

マニュアルページセクション 1: ユーザーコマンド

このソフトウェアおよび関連ドキュメントの使用と開示は、ライセンス契約の制約条件に従うものとし、知的財産に関する法律により保護されています。ライセンス契約で明示的に許諾されている場合もしくは法律によって認められている場合を除き、形式、手段に関係なく、いかなる部分も使用、複写、複製、翻訳、放送、修正、ライセンス供与、送信、配布、発表、実行、公開または表示することはできません。このソフトウェアのリバース・エンジニアリング、逆アセンブル、逆コンパイルは互換性のために法律によって規定されている場合を除き、禁止されています。

ここに記載された情報は予告なしに変更される場合があります。また、誤りが無いことの保証はいたしかねます。誤りを見つけた場合は、オラクル社までご連絡ください。

このソフトウェアまたは関連ドキュメントを、米国政府機関もしくは米国政府機関に代わってこのソフトウェアまたは関連ドキュメントをライセンスされた者に提供する場合は、次の通知が適用されます。

U.S. GOVERNMENT END USERS:

Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

このソフトウェアもしくはハードウェアは様々な情報管理アプリケーションでの一般的な使用のために開発されたものです。このソフトウェアもしくはハードウェアは、危険が伴うアプリケーション（人的傷害を発生させる可能性があるアプリケーションを含む）への用途を目的として開発されていません。このソフトウェアもしくはハードウェアを危険が伴うアプリケーションで使用する際、安全に使用するために、適切な安全装置、バックアップ、冗長性（redundancy）、その他の対策を講じることは使用者の責任となります。このソフトウェアもしくはハードウェアを危険が伴うアプリケーションで使用したこと起因して損害が発生しても、オラクル社およびその関連会社は一切の責任を負いかねます。

OracleおよびJavaはOracle Corporationおよびその関連企業の登録商標です。その他の名称は、それぞれの所有者の商標または登録商標です。

Intel、Intel Xeonは、Intel Corporationの商標または登録商標です。すべてのSPARCの商標はライセンスをもとに使用し、SPARC International, Inc.の商標または登録商標です。AMD、Opteron、AMDロゴ、AMD Opteronロゴは、Advanced Micro Devices, Inc.の商標または登録商標です。UNIXは、The Open Groupの登録商標です。

このソフトウェアまたはハードウェア、そしてドキュメントは、第三者のコンテンツ、製品、サービスへのアクセス、あるいはそれらに関する情報を提供することがあります。オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスに関して一切の責任を負わず、いかなる保証もいたしません。オラクル社およびその関連会社は、第三者のコンテンツ、製品、サービスへのアクセスまたは使用によって損失、費用、あるいは損害が発生しても一切の責任を負いかねます。

目次

はじめに	17
紹介	21
Intro(1)	22
User Commands	29
acctcom(1)	30
adb(1)	33
addbib(1)	34
alias(1)	36
allocate(1)	40
amt(1)	42
appcert(1)	43
apptrace(1)	52
apropos(1)	57
ar(1)	59
arch(1)	64
as(1)	65
asa(1)	71
at(1)	73
atq(1)	81
atrm(1)	82
audioconvert(1)	83
audiocctl(1)	87
audioplay(1)	89
audiorecord(1)	91
audiotest(1)	94

auths(1)	95
auto_ef(1)	99
awk(1)	102
banner(1)	108
basename(1)	109
basename(1B)	111
bc(1)	112
bdiff(1)	116
bfs(1)	117
biff(1B)	122
break(1)	123
builtin(1)	125
cal(1)	127
calendar(1)	128
cat(1)	130
cd(1)	133
cdrw(1)	137
checknr(1)	144
chgrp(1)	146
chkey(1)	149
chmod(1)	151
chown(1)	177
chown(1B)	181
ckdate(1)	182
ckgid(1)	185
ckint(1)	187
ckitem(1)	189
ckkeywd(1)	192
ckpath(1)	194
ckrange(1)	198
ckstr(1)	201
cksum(1)	204
cktime(1)	206
ckuid(1)	209
ckyorn(1)	212
clear(1)	214

cmp(1)	215
col(1)	217
comm(1)	219
command(1)	221
compress(1)	225
cp(1)	229
cpio(1)	235
cpp(1)	245
cputrack(1)	251
crle(1)	257
crontab(1)	270
csh(1)	277
csplit(1)	307
ct(1C)	310
ctags(1)	312
ctrunc(1)	315
ctstat(1)	318
ctwatch(1)	322
cu(1C)	324
cut(1)	331
date(1)	334
dc(1)	338
deallocate(1)	342
deroff(1)	344
df(1B)	345
dhcpinfo(1)	347
diff(1)	350
diff3(1)	354
diffmk(1)	356
digest(1)	357
dircmp(1)	359
dis(1)	360
disown(1)	363
dispgid(1)	364
dispuid(1)	365
dos2unix(1)	366

dpost(1)	368
du(1)	372
du(1B)	375
dump(1)	377
dumpcs(1)	380
echo(1)	381
echo(1B)	385
ed(1)	386
edit(1)	401
egrep(1)	406
eject(1)	409
elfdump(1)	411
elfedit(1)	418
elffile(1)	429
elfsign(1)	432
elfwrap(1)	436
encrypt(1)	439
enhance(1)	443
env(1)	445
eqn(1)	447
error(1)	451
ex(1)	455
exec(1)	466
exit(1)	470
expand(1)	473
exportfs(1B)	475
expr(1)	476
expr(1B)	480
exstr(1)	483
factor(1)	487
fastboot(1B)	488
fgrep(1)	489
file(1)	492
file(1B)	495
filebench(1)	497
filesync(1)	499

find(1)	507
finger(1)	516
fmt(1)	519
fmtmsg(1)	521
fold(1)	526
from(1B)	528
ftp(1)	529
gcore(1)	547
gencat(1)	549
geniconvtbl(1)	552
genmsg(1)	556
getconf(1)	562
getfacl(1)	569
getlabel(1)	573
getopt(1)	574
getoptcvt(1)	576
getopts(1)	579
gettext(1)	589
gettxt(1)	591
getzonepath(1)	593
glob(1)	594
gprof(1)	595
grep(1)	600
groups(1)	605
groups(1B)	606
grpck(1B)	607
hash(1)	608
head(1)	610
history(1)	612
hostid(1)	622
hostname(1)	623
iconv(1)	624
indxbib(1)	627
install(1B)	628
ipcrm(1)	630
ipcs(1)	632

isainfo(1)	637
isalist(1)	639
jobs(1)	640
join(1)	649
kbd(1)	653
kdestroy(1)	661
keylogin(1)	663
keylogout(1)	665
kill(1)	666
kinit(1)	672
klist(1)	680
kmdb(1)	683
kmfcfg(1)	690
kpasswd(1)	700
krb5-config(1)	701
ksh(1)	703
ksh88(1)	766
ktutil(1)	821
kvno(1)	823
lari(1)	825
last(1)	833
lastcomm(1)	835
ld(1)	837
ldapdelete(1)	865
ldaplist(1)	872
ldapmodify(1)	878
ldapmodrdn(1)	887
ldapsearch(1)	893
ldd(1)	904
ld.so.1(1)	911
let(1)	921
lex(1)	922
lgrpinfo(1)	934
limit(1)	941
line(1)	948
list_devices(1)	949

listusers(1)	953
llc2_autoconfig(1)	954
llc2_config(1)	955
llc2_stats(1)	957
ln(1)	965
ln(1B)	969
loadkeys(1)	972
locale(1)	973
localedef(1)	976
logger(1)	981
logger(1B)	983
login(1)	985
logname(1)	994
logout(1)	995
look(1)	996
lookbib(1)	997
lorder(1)	998
ls(1)	999
ls(1B)	1023
m4(1)	1026
mac(1)	1032
mach(1)	1035
machid(1)	1036
madv.so.1(1)	1037
mail(1)	1041
mail(1B)	1048
mailcompat(1)	1049
mailp(1)	1051
mailq(1)	1053
mailstats(1)	1055
mailx(1)	1057
make(1S)	1082
makekey(1)	1118
man(1)	1119
mconnect(1)	1128
mcs(1)	1129

mdb(1)	1131
mesg(1)	1183
mkdir(1)	1184
mkmsgs(1)	1186
mkstr(1B)	1188
mktemp(1)	1190
moe(1)	1195
more(1)	1197
mp(1)	1204
mpss.so.1(1)	1213
msgcc(1)	1216
msgcpp(1)	1219
msgcv(1)	1226
msgfmt(1)	1227
msggen(1)	1233
msgget(1)	1236
mt(1)	1237
mv(1)	1241
nawk(1)	1244
nc(1)	1265
ncab2clf(1)	1277
ncakmod(1)	1279
newform(1)	1280
newgrp(1)	1283
newtask(1)	1286
nice(1)	1289
nl(1)	1291
nm(1)	1294
nohup(1)	1300
nroff(1)	1305
od(1)	1308
on(1)	1315
optisa(1)	1317
pack(1)	1318
packagemanager(1)	1321
pagesize(1)	1324

pargs(1)	1325
passwd(1)	1327
paste(1)	1338
patch(1)	1341
pathchk(1)	1346
pax(1)	1349
perl(1)	1381
pfexec(1)	1389
pg(1)	1391
pgrep(1)	1396
pkcs11_inspect(1)	1400
pkg(1)	1402
pkgdepend(1)	1441
pkgdiff(1)	1447
pkgfmt(1)	1449
pkginfo(1)	1451
pkglint(1)	1454
pkgmerge(1)	1459
pkgmk(1)	1464
pkgmogrify(1)	1467
pkgparam(1)	1474
pkgproto(1)	1476
pkgrecv(1)	1478
pkgrepo(1)	1483
pkgsend(1)	1492
pkgsign(1)	1496
pkgtrans(1)	1498
pklogin_finder(1)	1501
pktool(1)	1503
plabel(1)	1522
plgrp(1)	1523
plimit(1)	1528
pmadvise(1)	1530
pmap(1)	1534
pm-updatemanager(1)	1547
ppgsz(1)	1549

ppriv(1)	1553
pr(1)	1558
praliases(1)	1563
prctl(1)	1564
preap(1)	1572
print(1)	1574
printenv(1B)	1577
printf(1)	1578
priocntl(1)	1587
proc(1)	1600
prof(1)	1605
profiles(1)	1609
projects(1)	1619
ps(1)	1621
ps(1B)	1634
ptree(1)	1637
pvs(1)	1639
pwd(1)	1644
radadrgen(1)	1645
ranlib(1)	1647
rcapstat(1)	1648
rcp(1)	1652
read(1)	1656
readonly(1)	1660
refer(1)	1662
regcmp(1)	1664
renice(1)	1666
rlogin(1)	1669
rm(1)	1674
rmformat(1)	1678
rmmount(1)	1683
roffbib(1)	1685
roles(1)	1687
rpcgen(1)	1689
rpm2cpio(1)	1695
rsh(1)	1696

runat(1)	1702
rup(1)	1705
rup(1C)	1706
ruptime(1)	1707
rusage(1B)	1708
rusers(1)	1710
rwho(1)	1711
sar(1)	1712
sccs(1)	1718
sccs-admin(1)	1728
sccs-cdc(1)	1733
sccs-comb(1)	1735
sccs-delta(1)	1737
sccs-get(1)	1740
sccs-help(1)	1746
sccs-prs(1)	1747
sccs-prt(1)	1751
sccs-rmdel(1)	1754
sccs-sact(1)	1755
sccs-sccsdiff(1)	1756
sccs-unget(1)	1757
sccs-val(1)	1758
scp(1)	1760
script(1)	1762
sdiff(1)	1763
sed(1)	1765
sed(1B)	1773
set(1)	1779
setfacl(1)	1790
setlabel(1)	1795
setpgrp(1)	1797
sftp(1)	1798
sh(1)	1803
shcomp(1)	1823
shell_builtins(1)	1825
shift(1)	1831

shutdown(1B)	1833
size(1)	1834
sleep(1)	1836
soelim(1)	1838
sort(1)	1839
sortbib(1)	1847
sotruess(1)	1849
spell(1)	1851
split(1)	1853
srchtxt(1)	1855
ssh(1)	1858
ssh-add(1)	1873
ssh-agent(1)	1876
ssh-http-proxy-connect(1)	1878
ssh-keygen(1)	1880
ssh-keyscan(1)	1885
ssh-socks5-proxy-connect(1)	1888
strchg(1)	1890
strings(1)	1893
strip(1)	1895
stty(1)	1897
stty(1B)	1907
sum(1)	1914
sum(1B)	1915
suspend(1)	1916
svcprop(1)	1917
svcs(1)	1923
symorder(1)	1931
sys-suspend(1)	1932
sysV-make(1)	1934
tabs(1)	1941
tail(1)	1945
talk(1)	1948
tar(1)	1951
tbl(1)	1965
tcopv(1)	1967

tee(1)	1968
telnet(1)	1970
test(1)	1984
test(1B)	1994
tftp(1)	1996
time(1)	1999
times(1)	2002
timex(1)	2003
tip(1)	2005
touch(1)	2014
touch(1B)	2019
tplot(1)	2020
tput(1)	2021
tr(1)	2026
tr(1B)	2031
trap(1)	2032
troff(1)	2035
true(1)	2038
truss(1)	2039
tset(1B)	2048
tsort(1)	2053
tty(1)	2055
type(1)	2056
typeset(1)	2057
ul(1)	2060
umask(1)	2061
uname(1)	2065
unifdef(1)	2068
uniq(1)	2070
units(1)	2073
unix2dos(1)	2075
updatehome(1)	2077
uptime(1)	2079
userattr(1)	2080
users(1B)	2081
uucp(1C)	2082

uuencode(1C)	2086
uuglist(1C)	2090
uustat(1C)	2091
uuto(1C)	2095
uux(1C)	2098
vacation(1)	2102
vc(1)	2106
vgrind(1)	2109
vi(1)	2113
vipw(1B)	2124
volcheck(1)	2125
volrmount(1)	2126
w(1)	2128
wait(1)	2130
wc(1)	2134
what(1)	2136
whatis(1)	2138
whereis(1B)	2139
which(1)	2141
who(1)	2142
whoami(1B)	2146
whocalls(1)	2147
whois(1)	2148
write(1)	2149
xargs(1)	2152
xgettext(1)	2158
xstr(1)	2160
yacc(1)	2162
yes(1)	2166
ypcat(1)	2167
ypmatch(1)	2168
yppasswd(1)	2169
ypwhich(1)	2171
zlogin(1)	2172
zonename(1)	2175
zonestat(1)	2176

はじめに

SunOS オペレーティングシステムの初心者でも、熟練したユーザーでも、オンラインのマニュアルページを使ってシステムおよびその機能に関する情報を入手できます。マニュアルページは、「それは何を行うのか」という質問に簡潔に回答するために作成されています。マニュアルページは一般にリファレンスマニュアルとなります。チュートリアルとしては作成されていません。

概要

次に、マニュアルページの各セクションと、それらのセクションで示される情報の概要を示します。

- セクション 1 では、オペレーティングシステムで使えるコマンドをアルファベット順に説明しています。
- セクション 1M では、主にシステムの保守と管理のために使用されるコマンドをアルファベット順に説明しています。
- セクション 2 では、すべてのシステムコールについて説明しています。これらの呼び出しのほとんどは、1つ以上のエラーを返します。エラー状態は、ほかの場合には返されない戻り値によって示されます。
- セクション 3 では、セクション 2 で説明されている UNIX システムプリミティブを直接呼び出す関数を除き、さまざまなライブラリにある関数について説明しています。
- セクション 4 では、各種ファイルの形式について説明しています。該当する場合は、ファイル形式を C 言語の構造体の宣言を使用して示します。
- セクション 5 では、文字セットテーブルなど、その他各種の情報について説明しています。
- セクション 7 では、特定のハードウェア周辺装置やデバイスドライバを参照する、さまざまな特殊ファイルについて説明しています。STREAMS ソフトウェアドライバ、モジュール、およびシステムコールの STREAMS 汎用セットについても説明します。
- セクション 9E では、開発者がデバイスドライバに組み込むことができる、DDI (デバイスドライバインタフェース)/DKI (ドライバ/カーネルインタフェース) 両用、DDI 専用、および DKI 専用のエントリポイントルーチンについて説明しています。

- セクション 9F では、デバイスドライバで利用できるカーネル関数について説明しています。
- セクション 9S では、ドライバとカーネルの間で情報を共有するためにドライバで使用されるデータ構造について説明しています。

マニュアルページの一般形式は次のとおりです。各マニュアルセクションのマニュアルページは、通常この順序に従っていますが、必要な項目だけを含んでいます。たとえば、記述すべきバグがコマンドにない場合、「使用上の留意点」というセクションはありません。各セクションの詳細については [intro](#) ページ、マニュアルページの一般的な情報については [man\(1\)](#) を参照してください。

名前 このセクションでは、コマンドまたは関数の名前および機能の概略を示します。

形式 このセクションでは、コマンドまたは関数の構文を示します。コマンドやファイルが標準パスに存在しない場合は、そのフルパス名が示されます。別の引数順序が必要な場合を除いて、オプションと引数はアルファベット順に、まず 1 文字の引数、次に引数付きのオプションが記述されます。

このセクションでは、次の特殊文字が使用されます。

[] 角括弧。角括弧に囲まれたオプションや引数は省略できます。角括弧がない場合、その引数は必須です。

... 省略記号。前の引数に複数の値を指定すること、または前の引数を複数回指定することができます (例: “filename...”)。

| 区切り文字。この文字で区切られている引数のうち 1 つだけを一度に指定できます。

{ } 中括弧。中括弧に囲まれたオプションや引数は相互に依存しているため、括弧内のすべてを一体として扱う必要があります。

プロトコル このセクションは、プロトコルが記述されているファイルを示すためにサブセクション 3R でのみ使用されます。

機能説明 このセクションでは、サービスの機能と動作を定義します。つまり、コマンドの機能を簡潔に説明します。オプションの説明や使用例はここでは示されていません。対話形式のコマンド、サブコマンド、要求、マクロ、および関数については「使用法」で説明します。

IOCTL このセクションは、セクション 7 のページだけに含まれます。[ioctl\(2\)](#) システムコールに適切なパラメータを提供するデバイスクラスのみが `ioctl` と呼ばれ、独自の見出しを生成しま

	す。特定のデバイスに関する <code>ioctl</code> 呼び出しは、(そのデバイスのマニュアルページに) アルファベット順に記述されています。 <code>ioctl</code> 呼び出しは、 <code>mtio(7I)</code> のように、すべて <code>io</code> で終わる特定クラスのデバイスに使用されます。
オプション	このセクションでは、コマンドオプションの一覧と各オプションの機能のサマリーを示します。オプションは、「形式」のセクションで示されている順序で、文字通りに記述されています。オプションに引数がある場合はそのオプションの下で説明され、該当する場合はデフォルト値が示されます。
オペランド	このセクションでは、コマンドのオペランドの一覧を示し、各オペランドがコマンドの動作にどのように影響を及ぼすかを説明します。
出力	このセクションでは、コマンドによって生成される出力 (標準出力、標準エラー、または出力ファイル) について説明します。
戻り値	値を返す関数のマニュアルページの場合、このセクションには、これらの値の一覧および値が返される条件が示されます。関数が 0 や -1 などの定数値だけを返す場合は、これらの値がタグ付き段落で一覧表示されます。それ以外の場合は、各関数の戻り値が 1 つの段落で説明されます。 <code>void</code> と宣言された関数は値を返さないため、「戻り値」では扱われません。
エラー	エラー発生時、ほとんどの関数はエラーコードを大域変数 <code>errno</code> に格納し、エラーの理由を示します。このセクションでは、関数が生成しうるすべてのエラーコードの一覧をアルファベット順で示し、各エラーの発生条件について説明します。複数の条件が同じエラーの原因になる場合、そのエラーコードの下に各条件が別々の段落で記述されます。
使用法	このセクションでは、詳細な説明を必要とする特殊な規則、機能、およびコマンドを示します。次に示すサブセクションは、組み込まれている機能を説明するために使用されます。 コマンド 修飾子 変数 式 入力文法
使用例	このセクションでは、コマンドまたは関数の使用例または使用方法を説明します。可能な場合は、コマンド行の入力とマシンの応答を含む完全な例が示されます。例の中で、プロンプトは <code>example%</code> と表示されるか、スーパーユーザーでなければならない

	い場合は <code>example#</code> と表示されます。例のあとに、説明、変数置換規則、または戻り値が示されます。ほとんどの例が、「形式」、「機能説明」、「オプション」、「使用法」のセクションの概念を示す実例となっています。
環境変数	このセクションでは、コマンドまたは関数が影響を与えるすべての環境変数の一覧を示し、その影響について簡単に説明します。
終了ステータス	このセクションでは、コマンドが呼び出し元のプログラムまたはシェルに返す値の一覧と、これらの値が返される条件を示します。通常、正常終了には0が返され、0以外の値はそれぞれのエラー状態を示します。
ファイル	このセクションでは、マニュアルページが参照するファイル、関連ファイル、およびコマンドが作成または必要とするファイルの一覧を示します。それぞれのあとに、サマリーまたは説明があります。
属性	このセクションでは、属性タイプとその対応する値を定義することにより、コマンド、ユーティリティ、およびデバイスドライバの特性を示します。詳細については、 attributes(5) を参照してください。
関連項目	このセクションでは、関連するマニュアルページ、当社のマニュアル、および外部の出版物を示します。
診断	このセクションでは、診断メッセージの一覧を示し、エラーの発生条件について簡単に説明します。
警告	このセクションでは、作業環境に重大な影響を与える可能性のある特殊な条件に関する警告の一覧を示します。これは診断の一覧ではありません。
注意事項	このセクションでは、このマニュアルページの他のどのセクションにも記載されない追加情報を示します。ユーザーへの参考情報として、特に関心を引く事項を扱います。ここで重要な情報が扱われることはありません。
使用上の留意点	このセクションでは、既知の問題について説明し、可能な場合は回避方法を示します。

参 照

紹介

名前	Intro, intro – コマンドおよびアプリケーションプログラムの序章
機能説明	このセクションでは、このオペレーティングシステムで使えるコマンドをアルファベット順に説明しています。 特別な範疇に収まるコマンドについては、次のように区別しています。 1B SunOS/BSD 互換性パッケージにだけ存在するコマンド。詳細は、『<i>SunOS/BSD Compatibility Package</i>』を参照してください。 1C ほかのシステムと通信するためのコマンド 1S SunOS システムにだけ存在するコマンド
その他のセクションについて	詳細は、SunOS リファレンスマニュアルの各セクションを参照してください。 ■ システム管理のコマンドについては、セクション 1M を参照してください。 ■ ファイルの形式については、セクション 4 を参照してください。 ■ 公式に使用できるファイルやその他のさまざまな情報については、セクション 5 を参照してください。 前述のコマンドや使用方法については、『System Administration Guide: Advanced Administration』を参照してください。
マニュアルページの コマンドの構文	特に説明しないかぎり、マニュアルページの「形式」の項で記述されるコマンドは、次の構文に従って、オプションやその他の引数を受け入れます。そして、次のように解釈されなければなりません。 <i>name</i> [-<i>option</i>...] [<i>cmdarg</i>...] where: [] 必須でない <i>option</i> (オプション) や <i>cmdarg</i> (引数) を囲みます。 ... <i>option</i> (オプション) や <i>cmdarg</i> (引数) が複数回発生することを意味します。 <i>name</i> 実行可能ファイルの名前です。 { } 中括弧で囲まれたオプション・引数は相互に依存しており、1 つの組として取り扱う必要があります。 <i>option</i> (常に “-” が先行します。) <i>noargletter</i>... または、<i>argletter optarg</i>[,...] <i>noargletter</i> 引数が必要ないオプション 1 文字を表します。複数の <i>noargletter</i> を指定する場合、1 つの “-” の後にまとめて指定できます (後述のガイドライン 5)。 <i>argletter</i> 引数が必要なオプション 1 文字を表します。

<i>optarg</i>	<i>argletter</i> に必要なオプション引数(文字列)です。複数の <i>optargs</i> を <i>argletter</i> に指定する場合、コンマで区切る、または、タブかスペース文字で区切って引用符で囲まなければなりません(後述のガイドライン 8)。
<i>cmdarg</i>	“-”で始まらないパス名(またはほかのコマンドの引数)。“-”だけを指定すると標準入力を表します。

特に指定しないかぎり、オペランドまたはオプション引数が数値である(つまり、数値を含む)場合、次のように処理されます。

- 数字は10進数の整数として認識されます。
- 0から2147483647までの数字は、構文上、数値であると認識されます。
- ユーティリティの機能説明において、負の数をオペランドまたはオプション引数として受け入れると記述されている場合、-2147483647から2147483647までの数字は、構文上、数値であると認識されます。
- ここに示す範囲よりも大きな範囲は許可されます。

コマンド構文規格: ガイドライン

ここで説明するコマンド構文のガイドラインは、既存のコマンドすべてに適用されているわけではありません。しかし、新規のコマンドはすべてこのガイドラインに従う予定です。すべてのシェルスクリプターはパラメータの構文解析に [getopts\(1\)](#) を使って、前後関係に依存するオプションが正しいかどうかチェックしなければなりません。これは、次に説明するガイドライン3から10までをサポートします。その他の規則については、コマンド自身がチェックしなければなりません。

1. コマンド名(前述の *name*)は、2文字から9文字までの長さでなければなりません。
2. コマンド名は、小文字と数字だけで構成されなければなりません。
3. オプション名(前述の *option*)は、1文字でなければなりません。
4. オプションには“-”が先行しなければなりません。
5. 引数なしのオプションは、1つの“-”の後に複数個まとめて指定できます。
6. オプションとオプションの最初の引数(前述の *optarg*)の間は、タブかスペース文字で区切らなければなりません。
7. オプションの引数は、必ず指定しなければなりません。
8. オプションに複数のオプションの引数が続く場合、それぞれをコンマで区切る、または、タブかスペース文字で区切って引用符で囲まなければなりません(たとえば、`-o xxx,z,yy` や `-o“xxx z yy”` など)。
9. コマンド行上では、オプションはオペランド(前述の *cmdarg*)より前に指定しなければなりません。
10. “--”を使って、オプションの終わりを示すことができます。

11. オプションの相対的な順番は問題になりません。
12. オペランド (前述の *cmdarg*) の相対的な順番は、その位置によって、コマンドが決めた意味に影響します。
13. "-" の前後にスペース文字を指定した場合 ("-") だけを指定した場合)、標準入力を表します。

Solaris を初めとする Sun 製品のために、前述のガイドラインを拡張する CLIP (コマンド行インタフェースパラダイム) という拡張セットが開発されました。この拡張セットの目的は、Sun のコマンド行構文を Linux システムで採用されている GNU コマンド行構文に近づけることです。これは、既存のユーティリティーを変更するわけではなく、また、この拡張セットをすべての新しいユーティリティーに適用するわけでもありません。つまり、必要に応じて、開発中のユーティリティーに適用するというだけのことです。

CLIP は、前述のガイドラインの完全な上位集合であり、IEEE 標準 1003.1-2001 (SUSv3) により近づいています。したがって、すべての GNU 構文を含むわけではありません。GNU 構文には、前述の IEEE 規則と衝突したり、あいまいな構造もあります。このような構造は許されません。

拡張 CLIP コマンド行構文は、次のようになります。

```
utility_name -a --longopt1 -c option_argument \
    -f option_argument --longopt2=option_argument \
    --longopt3 option_argument operand
```

この例では、ユーティリティーの名前は *utility_name* です。ユーティリティーの名前の後には、オプション、オプション引数、およびオペランドが続きます。これらをまとめて「引数」と呼びます。1つのハイフンの後に1つの文字または数字が続く引数 (たとえば、*-a*) のことを「短いオプション」と呼びます。2つのハイフンの後に複数の文字、数字、またはハイフンが続く引数 (たとえば、*--longopt1*) のことを「長いオプション」と呼びます。短いオプションと長いオプションをまとめて「オプション」と呼びます (古くは「フラグ」とも)。オプションには、オプション引数をとるものもあります (前述の例では、*-c option_argument*)。最後のオプション (および、オプション引数) の後に続く引数のことを「オペランド」と呼びます。ある引数が最初のオペランドであると認識されると、それ以降の引数はすべて、オペラントであると認識されます。

短いオプションとオプション引数の間には、空白が入っているものと、空白が入っていないものがあります。つまり、短いオプションとオプション引数が隣接していて、本当は2つの引数なのだが、1つの引数になっている場合があります。CLIP の仕様では、短いオプションとそのオプション引数の間には空白が入っていて、別々の引数である必要があります。しかし、昔のアプリケーションをサポートするために、いくつかの例外があります。

- ユーティリティーの「形式」節において、短いオプションとオプション引数の間にスペースが入っている場合 (前述の例では、`-c option_argument`)、アプリケーションは、短いオプションとオプション引数に、別々の引数を使用します。
- スペースが入っていない場合 (前述の例では、`-foption_argument`)、アプリケーションは、短いオプションとオプション引数が直接 (つまり、スペースなしに) 隣接していて、1つの引数であることを期待します。
- 前述の要件にもかかわらず、アプリケーションは、「形式」節でスペースが入っている場合も、そうでない場合も、さらには、短いオプションとオプション引数が1つの引数となっている場合も、別々の引数になっている場合も、その短いオプションとオプション引数を受け入れる必要があります。
- 長いオプションにオプション引数が付く場合、長いオプションとオプション引数の間には、必ず、等号記号(=)を入れる必要があると定義されています。しかし、ユーティリティーの「オプション」セクションにおいて、長いオプションとそのオプション引数の間に等号記号(=)が入っている場合でも (前述の例では、`--longopt2= option_argument`)、アプリケーションは、長いオプションとそのオプション引数の間に、空白を入れる方法も受け入れる必要があります (前述の例では、`--longopt1 option_argument`)。

CLIP は前述のガイドラインを拡張して、次のようなガイドラインを定義しています。

14. 同じような操作をグループにまとめるには、`command subcommand [options] [operands]` という形式が適切です。サブコマンド名はコマンド名と同じ規約 (ガイドライン1と2) に従います。
15. 長いオプションには「-」を前に付けて、適切な文字セットの英数字とハイフンだけを使用します。長いオプションの名前には、通常、1つから3つまでの単語を使用して、それぞれをハイフンで区切ります。
16. 長いオプションにオプション引数を指定するには、`--name=argument` の形式を使用する必要があります。ただし、`--name argument` の形式も受け入れます。
17. すべてのユーティリティーは、次の2つの標準の長いオプションをサポートする必要があります。`--version` (短いオプションの同義語は `-V`) と `--help` (短いオプションの同義語は `-?`) です。`--version` の場合、短いオプションの同義語と同じオプションがすでに使用されている場合は、短いオプションの同義語を変更してもかまいません。ただし、短いオプションの同義語は、必ず、用意しておく必要があります。これらのオプションに遭遇した時点で、引数の処理は停止され、適切な出力が表示され、ユーティリティーは正常に終了します。
18. 短いオプションは1つの長いオプションとのみ対応し、同様に、長いオプションは1つの短いオプションとのみ対応する必要があります。昔の習慣

や、コミュニティーで使用する同等のユーティリティーとの互換性を保つために、同等なオプションを用意できます。

19. 短いオプションの名前は、次の規則に従って、長いオプションの名前から決定される必要があります。
 1. 短いオプションの名前には、長いオプションの名前の最初の1文字を選択します。
 2. 最初の1文字がほかの短いオプションの名前と衝突する場合は、目立つ子音を選択します。
 3. 最初の1文字も目立つ子音もほかの短いオプションの名前と衝突する場合は、目立つ母音を選択します。
 4. 長いオプションの名前のどの文字も使用できない場合は、任意の文字を選択します。
20. 長いオプションの名前が1文字だけの場合、短いオプションの名前にも同じ文字を使用する必要があります。1文字だけの長いオプションは避けるべきです。1文字だけの長いオプションを使用するのは、その文字がオプションの意味をもっとも端的に表すという、ごくまれな場合だけです。
21. サブコマンドの形式は、通常、拡張 CLIP ガイドラインのガイドライン1で説明した形式である必要があります。サブコマンドを省略する場合、コマンドにオペランドを指定してはいけません。この場合、遭遇したときに引数の処理を停止させるオプションだけを指定できます。この形式のコマンドを呼び出すときに、サブコマンドも引数も省略すると、エラーになります。このガイドラインは、コマンドとサブコマンドの構成として、<コマンド> --help、<コマンド> -?、<コマンド> --version、および<コマンド> -v という共通な形式を受け入れるために提供されます。

これらのガイドラインのいくつかは、ユーティリティーの作者にしか関係のないものです。これらのガイドラインをここで紹介したのは、この構文に従ってユーティリティーを作成したいユーザーのためです。

謝辞

Oracle America, Inc. gratefully acknowledges The Open Group for permission to reproduce portions of its copyrighted documentation. Original documentation from The Open Group can be obtained online at <http://www.opengroup.org/bookstore/>.

The Institute of Electrical and Electronics Engineers and The Open Group, have given us permission to reprint portions of their documentation.

In the following statement, the phrase "this text" refers to portions of the system documentation.

Portions of this text are reprinted and reproduced in electronic form in the SunOS Reference Manual, from IEEE Std 1003.1, 2004 Edition, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 6,

Copyright (C) 2001-2004 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between these versions and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html>.

This notice shall appear on any product containing this material.

関連項目

[getopts\(1\)](#), [wait\(1\)](#), [exit\(2\)](#), [getopt\(3C\)](#)

診断

終了時、すべてのコマンドはステータスを表す2バイトを返します。1つは、システムから提供され、終了の原因を示します。もう1つ(正常な終了において)は、プログラムから提供されます([exit\(2\)](#)を参照)。前のバイトが0の場合、正常な終了を表します。後のバイトが0の場合、正常な実行を表します。後のバイトがゼロでない場合、間違った引数を指定した、または不良で受け入れることができないデータを指定したなどの障害を示します。このバイトは、「終了コード」、「終了ステータス」、「リターンコード」などさまざまな呼ばれ方をします。そして、特別な使い方がある場合に限って説明されます。

警告

NULL文字を含むファイルを処理しているときに、予測していなかった結果を出すコマンドがあります。通常このようなコマンドはテキスト入力行を文字列として扱っているので、行中のNULL文字(つまり文字列の終端)に出会うと混乱してしまうからです。

参 照

User Commands

名前	acctcom – プロセスアカウンティングファイルの検索と出力
形式	acctcom [-abfhikmqrtv] [-C sec] [-e time] [-E time] [-g group] [-H factor] [-I chars] [-l line] [-n pat]
機能説明	acctcom は、<i>filename</i> 引数で指定されたファイル、標準入力、もしくは <code>/var/adm/pacct</code> の内容を <code>acct.h(3HEAD)</code> で規定した形式で読み込み、条件に従って選択したレコードを標準出力に書き出します。個々のレコードは、1つのプロセスの実行を表します。必ず出力されるのは、COMMAND NAME、USER、TTYNAME、START TIME、END TIME、REAL (SEC)、CPU (SEC)、MEAN SIZE (K) です。また指定された場合にだけ出力されるのは、F(fork()/exec() フラグ:1 は exec() なしの fork())、STAT (システム終了ステータス)、HOG FACTOR、KCORE MIN、CPU FACTOR、CHARS TRNSFD、および BLOCKS READ (読み書きされたブロックの合計数) です。 スーパーユーザー特権で実行されたコマンドについては、コマンド名の前に '#' が付加されます。また認識されている端末のいずれにも関連していないプロセスについては、TTYNAME フィールドに '?' が出力されます。 <i>filename</i> 引数を指定しないとき、標準入力端末もしくは <code>/dev/null</code> に結びつけられていれば(シェル中で '&' を使う場合と同じ状態)、<code>/var/adm/pacct</code> が読み込まれます。対応していなければ標準入力を読み込まれます。 <i>filename</i> 引数を指定すると、記述した順序でそれらのファイルが読み込まれます。各ファイルは正方向に、すなわちプロセスの終了時間の昇順に読まれます。通常 <code>/var/adm/pacct</code> は、現在の検査対象ファイルを表します。処理量の多いシステムでは、このようなファイルを複数個用意して、現在のファイル以外のファイルを <code>/var/adm/pacct incr</code> として存在させることが必要となる場合があります。
オプション	サポートしているオプションは、次のとおりです。 -a 選択されたプロセスに関する平均値の統計情報を表示します。この統計情報は、全出力レコードの後に出力されます。 -b レコードを逆方向に、すなわち最新のコマンドから順番に読み込みます。入力先が標準入力の場合には、このオプションは無視されます。 -f 出力上に fork()/exec() フラグとシステム終了ステータス情報を書き出します。このオプションによる出力では、数値はいずれも 8 進数表示となります。 -h 平均メモリーサイズの代わりに、プロセスが実行中に使用した合計 CPU 時間の割合を表示します。この値は「ホッグ係数」と呼ばれ、(合計 CPU 時間)/(経過時間) で計算されます。 -i 出力上に、入出力回数を示す欄を表示します。 -k メモリサイズの代わりに、実行時に使われたキロバイト/分ごとのコアサイズの合計を表示します。 -m 平均コアサイズを表示します (デフォルト)。

- q 出力レコードはいっさい表示せずに、-a オプションで指定するような統計情報だけを出力します。
- r CPU 係数を出力します。これは (ユーザー時間) / ((システム時間) + (ユーザー時間)) で計算されます。
- t システム CPU 時間とユーザー CPU 時間を区別して出力します。
- v 各欄のヘッダを出力しません。
- C *sec* (システム時間) + (ユーザー時間) で計算される合計 CPU 時間が *sec* で示す秒数を超えているプロセスだけを選択します。
- e *time* *time* で示す時間もしくはそれ以前に存在していたプロセスだけを選択します。
- E *time* *time* で示す時間もしくはそれ以前に終了したプロセスだけを選択します。なお -S と -E の両オプションを指定し、*time* として同じ時間を指定した場合には、その時間に存在していたプロセスだけが選択されます。
- g *group* *group* で示すグループに属しているプロセスだけを選択します。*group* 引数には、グループ ID とグループ名のどちらでも指定できます。
- H *factor* 「ホッグ係数」値が *factor* で示す値を超えているプロセスだけを選択します。「ホッグ係数」に関しては、前述の -h オプションの項を参照してください。
- I *chars* *chars* で示す値を超える数の文字を転送するプロセスだけを選択します。
- l *line* /dev/term/*line* 端末に属するプロセスだけを選択します。
- n *pattern* *pattern* で示すパターンに一致するコマンドだけを選択します。*pattern* 引数として、[regcmp\(3C\)](#) で述べるような正規表現も指定できます。ただし + 記号は「1 回以上の出現」を表します。
- o *output-file* 選択されたプロセスレコードを、*output-file* で示すファイルに入力データ形式でコピーします。標準出力への書き出しは行いません。
- O *sec* システム CPU 時間が *sec* で示す秒数を超えているプロセスだけを選択します。
- s *time* *time* で示す時間もしくはそれ以降に存在していたプロセスだけを選択します。*time* の指定形式は *hr[:min[:sec]]* です。
- S *time* *time* で示す時間もしくはそれ以降に開始されたプロセスだけを選択します。

- u user

user で示すユーザーに属しているプロセスだけを選択します。user 引数として指定できるのは、ユーザー ID、ログイン名 (これはユーザー ID に変換される)、'#' (スーパーユーザー特権で実行されたプロセスを表す)、および '?' (認識されていないユーザー ID に対応しているプロセスを表す) です。
- ファイル

/etc/group

システムグループファイル

/etc/passwd

システムパスワードファイル

/var/adm/pacctincr

稼動中のプロセスアカウンティングファイル
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/accounting/legacy-accounting
CSI	有効

- 関連項目

[ps\(1\)](#), [acct\(1M\)](#), [acctcms\(1M\)](#), [acctcon\(1M\)](#), [acctmerg\(1M\)](#), [acctprc\(1M\)](#), [acctsh\(1M\)](#), [fwtmp\(1M\)](#), [runacct\(1M\)](#), [su\(1M\)](#), [acct\(2\)](#), [regcmp\(3C\)](#), [acct.h\(3HEAD\)](#), [utmp\(4\)](#), [attributes\(5\)](#)
- 『Oracle Solaris の管理: 一般的なタスク』
- 注意事項

acctcom は、すでに終了したプロセスに関する情報を出力するものです。現在稼動中のプロセスについての情報は、[ps\(1\)](#) により得ることができます。

名前 adb – 汎用デバッガ

形式 adb [-kw] [-I *dir*] [-P *prompt*] [-V *mode*] [*object* [*core*]]

機能説明 adb ユーティリティーは、対話型の汎用デバッガです。このユーティリティーは、ファイルの検査に使用でき、プログラムを実行するための制御環境を提供します。

adb ユーティリティーは、[mdb\(1\)](#) ユーティリティーへのリンクとして実装されるようになりました。mdb(1) は、ユーザープロセスおよび稼働中のオペレーティングシステムまたはオペレーティングシステムクラッシュダンプの検査に使用できる低レベルデバッグユーティリティーです。新しい mdb(1) ユーティリティーは、adb のマクロファイル処理を含む、adb の既存の構文や機能との完全な後方互換性を備えています。『[Oracle Solaris Modular Debugger Guide](#)』および mdb(1) のマニュアルページには、adb 互換モードなどの mdb の機能について説明されています。adb 互換モードは、adb リンクが実行される場合にデフォルトで有効になります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/debug/mdb

関連項目 [mdb\(1\)](#), [attributes\(5\)](#)

『[Oracle Solaris Modular Debugger Guide](#)』

名前 addbib – 参考文献データベース (bibliographic database) の作成または拡張

形式 addbib [-a] [-p *promptfile*] *database*

機能説明 addbib が起動したときに、最初の Instructions? プロンプトに対して y と応答すると指示が与えられます。n (または RETURN) を入力すると、指示をスキップします。addbib は、さまざまな参考文献に関するフィールドを示し、端末からの応答を読み取って、出力レコードを *database* に送信します。NULL 応答 (RETURN のみ) は、そのフィールドを省略することを意味します。-(マイナス符号) は、前のフィールドに戻ることを意味します。行末にバックスラッシュを付けると、フィールドを次の行に続けることができます。Continue? プロンプトが繰り返されているときは、y (または RETURN) を入力して再開するか、n または q と入力して現在のセッションを終了するか、あるいは任意のシステムエディタ (vi(1)、ex(1)、ed(1) を参照) を使用して *database* を編集できます。

オプション サポートしているオプションは、次のとおりです。

- a 抄録に対するプロンプトを抑制します。デフォルトでは、抄録が求められます。抄録は、CTRL-D で終了します。
- p *promptfile* *promptfile* に定義された新しいプロンプトの枠組みを使用します。このファイルには、プロンプト文字列、TAB、*database* に書き込まれるキー文字が含まれます。

使用法

参考文献キー文字 以下に、最も一般的なキー文字とその意味を示します。addbib は英語でプロンプトを表示するため、ユーザーはこれらのキー文字を知っている必要はありませんが、参考文献ファイルを後で編集する場合にはこれらの情報が必要になります。

- %A 著者名
- %B 参考資料を収録する文献
- %C 都市 (発行場所)
- %D 発行日
- %E 参考資料を収録した文献の編集者
- %F 脚注番号またはラベル (refer で与えられる)
- %G 政府発注番号
- %H 参考資料の前に印刷された見出し解説
- %I 発行者 (発行元)
- %J 参考資料を収録した定期刊行物
- %K 参考資料の検索に使用するキーワード
- %L refer の -k オプションで使用されるラベルフィールド

%M	ベル研究所によるメモ (未定義)
%N	ボリューム内の番号
%O	参考資料の最後に印刷された他の解説
%P	ページ番号
%Q	企業または外国の著者 (著作権なし)
%R	レポート、ペーパー、または論文 (未公表)
%S	シリーズのタイトル
%T	資料または文献のタイトル
%V	ボリューム番号
%X	抜粋 — refer ではなく、roffbib で使用される
%Y,Z	refer では無視される

使用例

例1 参考文献ファイルの編集

Aを除いて、各フィールドを1回だけ指定する必要があります。関連フィールドだけを提供する必要があります。

```
%A   Mark Twain
%T   Life on the Mississippi
%I   Penguin Books
%C   New York
%D   1978
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目

[ed\(1\)](#), [ex\(1\)](#), [indxbib\(1\)](#), [lookbib\(1\)](#), [refer\(1\)](#), [roffbib\(1\)](#), [sortbib\(1\)](#), [vi\(1\)](#), [attributes\(5\)](#)

名前	alias, unalias – コマンドまたはコマンド群の別名または省略形の生成と削除
形式	<pre>/usr/bin/alias [alias-name[= string...]] /usr/bin/unalias alias-name... /usr/bin/unalias -a</pre>
csH	<pre>alias [name [def]] unalias pattern</pre>
ksh88	<pre>alias [-tx] [name[= value]...] unalias name... unalias [-a]</pre>
ksh	<pre>alias [-ptx] [name[= value]...] unalias [-a] [name...]</pre>
機能説明	alias および unalias ユーティリティーは、コマンドまたはコマンド群の別名あるいは省略形を作成または削除します。これらのユーティリティーの動作は、C シェル環境と Korn シェル環境では異なります。
/usr/bin/alias	alias ユーティリティーは、別名定義を作成または再定義するか、あるいは既存の別名定義を標準出力に書き出します。別名定義は、コマンド名を置き換える文字列を指定するものです。 別名定義は、現在のシェルの実行環境、およびそのシェルの全サブシェルの実行環境に影響を及ぼします。このドキュメントに記述されているように用いれば、別名定義は現在のシェルの親プロセスにも、シェルが呼び出すユーティリティーの環境にも影響を及ぼしません。
/usr/bin/unalias	unalias ユーティリティーは、指定された別名の定義を削除します。それにより、現在のシェルの実行環境から別名が削除されます。-a オプションは、現在の実行環境からすべての別名定義を削除します。
csH	alias は別名 <i>name</i> に <i>def</i> で指定した別名の定義を割り当てます。その割り当てられた <i>def</i> はワードの並びで、エスケープされたヒストリ置換のメタシンタックスを含んでいてもかまいません。<i>name</i> に alias または unalias を使用することはできません。<i>def</i> を省略すると、別名 <i>name</i> が現在の定義と共に表示されます。<i>name</i> と <i>def</i> の両方を省略すると、現在あるすべての別名が表示されます。 実装状態の制限によって、別名の定義は、それが使われる前のコマンド行で入力されていなければなりません。 unalias は <i>pattern</i> が示すファイル名置換パターンに一致する別名を破棄します。‘unalias *’ と指定すると、すべての別名を破棄できます。

ksh88 引数なしの場合、このコマンドは標準出力上に *name=value* という形式の別名のリストを表示します。*value* が指定された名前に対しては別名を定義します。*value* の末尾にスペースがあると、次のワードが別名置換指定かどうかをチェックします。*-t* フラグは、検索済みの別名を設定または一覧表示します。検索済み別名の値は、指定した *name* に対応する完全パス名になります。*PATH* の値を再設定するとこの値は未定義になりますが、別名は検索済みのままです。*-t* フラグを省略すると、*value* が指定されていない引数リスト内の各 *name* について、別名の名前と値を表示します。*-x* フラグは、エクスポートされた別名を設定または表示します。エクスポートされた別名は、名前で起動されるスクリプト用に定義されます。*name* が指定されているが、*value* は指定されておらず、*name* に対する別名も定義されていない場合は、終了ステータスは0以外になります。

unalias を指定すると *name* が示す *alias* を *alias* リストから削除できます。

ksh *alias* は、別名定義を作成または再定義したり、既存の別名定義を標準出力に書き出したりします。

別名定義は、コマンド名を置き換える文字列を指定するものです。別名には、シェルに対する特殊文字を除く、任意の出力可能な文字を含めることができます。別名の末尾がスペース (SPACE) またはタブ (TAB) の場合は、別名により置き換えられるコマンド名に続く語も別名かどうかチェックされます。

名前が指定されていない場合は、すべての別名の名前と値が標準出力に書き込まれます。名前は指定されているが *=value* が指定されていない場合は、名前に対応する別名の現在値が名前ごとに標準出力に書き込まれます。*=value* が指定されている場合は、別名が作成または再定義されます。

alias は宣言コマンドとしてシェルに内蔵されているため、引数に対するフィールド分割およびパス名展開は実行されません。チルド展開は *value* に対して行われます。別名定義が影響を与えるのは、現在のシェル環境により読み込まれたスクリプトだけです。このシェルにより実行されるスクリプトには影響しません。

unalias は、指定された各別名の定義を現在のシェル実行環境から削除します。*-a* が指定されている場合は、すべての別名を削除します。すでに読み込まれ、実行されているコマンドには影響を与えません。

オプション *unalias* では、次のオプションがサポートされています。

-a 現在のシェルの実行環境から、すべての別名定義を削除します。

ksh88 *alias* では次のオプションがサポートされています。

-t 検索済みの別名を設定または一覧表示します。

ksh *alias* では次のオプションがサポートされています。

-p 出力をシェルへの入力に使用可能な *alias* コマンドの形式にして、現在の別名を再作成できるようにします。

-t 検索済みの別名を指定します。

検索済みの別名は、コマンド名をコマンドのパス名に関連付けます。これらは、PATH 変数の設定解除時にリセットされます。検索済み別名の機能は、廃止または互換性がなくなる可能性があります。

-x 無視されます。このオプションは廃止されています。

unalias では、次のオプションがサポートされています。

-a すべての別名定義を削除します。*name* オペランドは省略可能であり、指定されている場合は無視されます。

オペランド 次のオペランドがサポートされています。

alias *alias-name* 別名定義を標準出力に書き出します。

unalias *alias-name* 削除する別名を指定します。

alias-name=string *alias-name* で示す別名に、*string* で示す文字列を割り当てます。

オペランドを1つも指定しないと、すべての別名定義が標準出力に書き込まれます。

出力 オペランドがすべて省略された場合、または *name* オペランドだけが指定された場合の、別名の表示形式は次のとおりです。

```
"%s=%s\n" name, value
```

文字列 *value* は、シェルへ再入力できるよう、適切な引用符を付加して出力されます。

使用例 例1 コマンドの出力を変更する

ls ユーティリティの出力を複数カラム形式の注釈付きに変更します。

```
example% alias ls="ls -CF"
```

例2 コマンド履歴ファイルにある直前のエントリを繰り返す

次の例は、コマンドの履歴ファイルにある直前の入力を繰り返す単純な "redo" コマンドを生成します。

```
example% alias r='fc -s'
```

例3 コマンドの出力オプションを指定する

次の例は、du ユーティリティがディスク出力を 1024 バイト単位にまとめるようにします。

```
example% alias du="du -k"
```

例4 別名である引数を処理する

次の例では、別名である引数を処理できるよう、nohup ユーティリティーを設定します。

```
example% alias nohup="nohup "
```

環境変数 alias と unalias の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

alias >0 alias-name オペランドで指定した名前の1つが別名定義を持っていなかったか、もしくはエラーが発生しました。

unalias >0 alias-name オペランドで指定した名前の1つが正しい別名定義を表していなかったか、もしくはエラーが発生しました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

csh, ksh88

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

ksh

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

関連項目 [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [shell_builtins\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 allocate – デバイスの割り当て

形式 allocate [-s] [-w] [-F] [-U *uname*] [-z *zonename*] *device*
allocate [-s] [-w] [-F] [-U *uname*] [-z *zonename*] -g *dev-type*

機能説明 allocate ユーティリティーは、デバイス割り当てメカニズムを使用してデバイスの所有者を管理します。これにより、各デバイスが、ある時点で、資格のある 1 ユーザーによってのみ使用されることが保証されます。

device 引数には、操作対象のデバイスを指定します。デバイス所有者の整合性を維持するため、そのデバイスに関連付けられているすべてのデバイス特殊ファイルで割り当て操作が実行されます。

デフォルトの割り当て操作により、*device* に関連付けられたデバイス特殊ファイルが現在のプロセスの *uid* に割り当てられます。

デバイスを割り当てることができるのは、承認されたユーザーのみです。必要な承認は、[device_allocate\(4\)](#) で指定されます。

システムが Trusted Extensions で構成されている場合、allocate はデバイスの clean プログラムを実行してから、呼び出し側にそのデバイスへのアクセスを許可します。デバイスがマウント可能なファイルシステムを含むリムーバブルメディアの場合、呼び出し側が選択すると allocate によってメディアがマウントされます。

オプション サポートしているオプションは、次のとおりです。

- F *device* 空きデバイスまたは事前に割り当てられたデバイスを強制的に割り当てます。このオプションは、デバイスを特定のユーザーに割り当てまたは再割り当てするときに -u オプションとともによく使用されます。solaris.device.revoke 承認を保持するユーザーだけがこのオプションを使用できます。
- g *dev-type* デバイスタイプが *dev-type* に一致するデバイスを割り当てます。dev-type 引数には、操作対象のデバイスタイプを指定します。
- s サイレント。診断出力をすべて抑止します。
- U *uname* 割り当て操作の実行時に、現在のプロセスのユーザー ID ではなく、ユーザー ID *uname* を使用します。solaris.device.revoke 承認を保持するユーザーだけがこのオプションを使用できます。

Trusted Extensions では次のオプションがサポートされています。

- w ウィンドウ環境でデバイスのクリーニングプログラムを実行します。ウィンドウバージョンのプログラムが存在する場合は、そのプログラムが使用されます。そうでない場合は、標準バージョンが端末ウィンドウで実行されます。
- z *zonename* *zonename* で指定されたゾーンにデバイスを割り当てます。

オペランド	次のオペランドがサポートされています。 <i>device</i> 割り当てるデバイスの名前を指定します。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 20 指定したデバイスのエントリが存在しません。 ほかの値 エラーが発生した。
ファイル	/etc/security/device_allocate /etc/security/device_maps /etc/security/dev/* /etc/security/lib/*
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

呼び出しは不確実です。オプションは不確実です。出力は非インタフェースです。

関連項目 [deallocate\(1\)](#), [list_devices\(1\)](#), [device_allocate\(1M\)](#), [dminfo\(1M\)](#), [mkdevalloc\(1M\)](#), [mkdevmaps\(1M\)](#), [device_allocate\(4\)](#), [device_maps\(4\)](#), [attributes\(5\)](#)

Controlling Access to Devices

注意事項 このマニュアルページで説明している機能は、Solaris 監査が有効な場合にのみ使用できます。

このマニュアルページで説明している機能は、[device_allocate\(1M\)](#) サービスが有効な場合にのみ使用できます。

Trusted Extensions で構成されているシステムでは、この機能はデフォルトで有効です。

/etc/security/dev、[mkdevalloc\(1M\)](#)、および [mkdevmaps\(1M\)](#) は、Solaris オペレーティング環境の将来のリリースではサポートされない可能性があります。

名前	amt – 抽象マシンテストの実行
形式	amt [-s]
機能説明	amt コマンドは、Common Criteria セキュリティ認定システムで使用するためのコマンドです。このコマンドを使用して、Controlled Access Protection Profile のオブジェクト再利用要件を適用するために必要な低レベルの関数が正しく機能することを確認します。/usr/bin/amt は、システムに固有のテストを実行するシェルスクリプトです。32 ビットシステムの場合、テストは 32 ビットアプリケーションとして実行されます。64 ビットシステムの場合、テストは 2 回実行されます (32 ビットアプリケーションとして 1 回、64 ビットアプリケーションとして 1 回)。 amt は、-s オプションを使用して出力が抑止されている場合を除き、実行した各テストの結果を pass または fail で一覧表示します。
オプション	次のオプションがサポートされています。 -s 出力を抑止します。
終了ステータス	次のエラー値が返されます。 0 すべてのテストで合格。 >0 表示された数のテストで不合格。 <0 コマンド行の引数が不正。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [attributes\(5\)](#)

名前	appcert - アプリケーションレベルの製品検査による Solaris インタフェースの不安定な使用の確認
形式	appcert [-h] [-n] [-f <i>infile</i>] [-w <i>working_dir</i>] [-B] [-L] [-S] { <i>obj</i> <i>dir</i> }...
機能説明	appcert ユーティリティーは、Solaris アプリケーションバイナリインタフェース (ABI) へのアプリケーションの準拠を検査します。Solaris ABI は、アプリケーションが安全かつ安定して使用できる Solaris の実行時ライブラリインタフェースを定義します。具体的には、appcert は不安定な実行時インタフェースへの依存、および Solaris の将来のリリースで製品の動作に障害を発生させる可能性があるそのほかの危険を特定します。 appcert は次の内容をチェックします。 ■ Solaris ライブラリでの非公開シンボルの使用法。これらは非公開シンボル、つまり、開発者の使用が意図されていない関数またはデータです。Solaris ライブラリが相互の呼び出しに使用するインタフェースです。これらのシンボルにより、セマンティック動作が変更されるだけでなく、完全に削除されてしまう可能性もあるため (いわゆる降格シンボル)、アプリケーションがこれらのいずれにも依存していないことを確認することをお勧めします。 ■ 静的リンク。特に、これは、対応する共有オブジェクト <i>.so</i> の動的なリンクではなく、アーカイブ <i>libc.a</i>、<i>libsocket.a</i>、および <i>libnsl.a</i> の静的リンクを示します。ある Solaris ライブラリから別の Solaris ライブラリへの非公開シンボル呼び出しのセマンティクスはリリースごとに変更される場合があるため、ライブラリコードをバイナリオブジェクトに固定することはお勧めしません。 ■ 非結合シンボル。これらは、appcert が実行されたときに動的リンカーが解決できなかったライブラリシンボル (つまり、関数またはデータ) です。この問題は、環境の問題 (<i>LD_LIBRARY_PATH</i> など) または構築の問題 (コンパイル時に <i>-llib</i> や <i>-z defs</i> を指定しないなど) である可能性があります。このような問題を指摘したり、より深刻な問題を示したりする場合に、これらのシンボルにフラグが付けられます。 appcert によって、製品全体を簡単に検査できます (製品が多数のプログラムの集合体で、共有オブジェクトをサポートしている場合)。具体的には、appcert で製品のインストール先ディレクトリを調べます。 このタスクを実行するため、appcert は、製品内部のオブジェクトファイル (実行可能オブジェクトまたは共有オブジェクト) ごとにインタフェースの依存関係のプロファイルを作成し、依存関係にあるすべての Solaris システムインタフェースを特定します。(appcert は、この特定のために Solaris 実行時リンカーを使用します。) 次に、これらの依存関係プロファイルを Solaris ABI の定義と比較して、非公開の (アプリケーションレベルで使用する場合に安全でなく不安定な) インタフェースを識別します。

appcert は単純なロールアップレポートを生成し、製品のコンポーネントに問題があった場合は、そのコンポーネントと問題の内容を示します。開発者は、このレポートを使用することで、製品の安定性を複数のリリースにわたって検査できます。

appcert は、公開 (安全かつ安定した) Solaris インタフェースと非公開 (非 ABI) インタフェースの両方について、完全なインタフェース依存関係情報を生成します。必要に応じて、この情報を製品コンポーネントごとに確認することもできます。

重要: appcert は、チェック対象のアプリケーションを実行する環境と同じ環境で実行する必要があります。「注意事項」を参照してください。

オプション

サポートしているオプションは、次のとおりです。

- B appcert をバッチモードで実行すると、出力レポートはバイナリごとに 1 行ずつ表示されます。行の先頭の PASS はバイナリに問題が見つからなかったこと、FAIL は何らかの問題が見つかったこと、INC はバイナリを完全にはチェックできなかったことを意味します。ただし、これらのラベルを単純に文字どおりに解釈することは避けてください。たとえば、PASS は appcert で警告が発せられなかったことを意味しているにすぎません。これらの文字列は左寄せされるため、`grep ^FAIL ...` などを使用して選択できます。
- f *infile* 確認するファイルの一覧 (各行に 1 ファイル) を含む *infile* ファイルを指定します。この一覧は、コマンド行オペランドで処理された一覧に追加されます (下記の「オペランド」の項を参照)。
- h 使用法についての情報を出力します。
- L appcert は、製品を検査して共有オブジェクトの存在を確認します。共有オブジェクトが見つかった場合、そのオブジェクトが存在するディレクトリを `LD_LIBRARY_PATH` に追加します。このフラグを使用すると、appcert によるこの動作は無効になります。
- n このオプションを指定すると、ディレクトリを検索してチェックするバイナリを探すときに、シンボリックリンクをたどらなくなります。[find\(1\)](#) を参照してください。
- S Solaris ライブラリディレクトリ (`/usr/openwin/lib:/usr/dt/lib`) を `LD_LIBRARY_PATH` に追加します。
- w *working_dir* ライブラリコンポーネントを実行するディレクトリを特定して、一時ファイルを作成します (デフォルトは `/tmp`)。

オペランド

次のオペランドがサポートされています。

- {*obj* | *dir*} ... チェック対象の製品を構成するオブジェクトや、そのオブジェクトを含むディレクトリの完全な一覧。appcert はディレクトリを再帰的

に検索して、オブジェクトファイルを探します。オブジェクトでないファイルは無視されます。

終了ステータス 次の終了ステータスが返されます。

- 0 appcert の実行に成功し、バイナリ安定性問題は見つかりませんでした。
- 1 appcert の実行に失敗しました。
- 2 確認した一部のオブジェクトに、バイナリ安定性問題が見つかりました。
- 3 チェック可能なバイナリオブジェクトが見つかりませんでした。

制限事項 検査対象のオブジェクトファイルがライブラリに依存している場合、その依存関係を (コンパイラの `-l` スイッチを使用して) 内部に記録する必要があります。

検査対象のオブジェクトファイルがほかの共有ライブラリに依存している場合、appcert の実行時に、`LD_LIBRARY_PATH` または `RUNPATH` を使用してそれらのライブラリへのアクセスが可能になっている必要があります。

64 ビットアプリケーションをチェックするには、64 ビット Solaris カーネルがマシンで実行されている必要があります。[isalist\(1\)](#) を参照してください。また、現在、64 ビットアプリケーションでは静的リンクのチェックは実行されません。

appcert は、次を検査できません。

- 完全または部分的に静的にリンクされているオブジェクトファイル。

完全に静的にリンクされたオブジェクトは、不安定と報告されます。

- 実行権が設定されていない実行可能ファイル。

これらのファイルはスキップされます。実行権のない共有オブジェクトはスキップされません。

- `setuid root` であるオブジェクトファイル。

これらのファイルは、[ldd\(1\)](#) の制限のためにスキップされます。チェックするにはアクセス権のコピーや変更を行ってください。

- ELF でない実行可能ファイル (シェルスクリプトなど)。
- C 言語以外の Solaris インタフェース (C++, Java など)。

Solaris ライブラリの呼び出しが C であれば、コード自体が C である必要はありません。

出力ファイル

appcert は、検索結果を作業ディレクトリ内の次のファイルに記録します (デフォルトの作業ディレクトリは `/tmp/appcert.?????`)。

- Index** チェック済みバイナリと、そのバイナリに固有の出力が存在する作業ディレクトリ内のサブディレクトリ間のマッピング。
- Report** appcert の実行時に標準出力に表示されたロールアップレポートのコピー。
- Skipped** appcert がチェックを求められたがスキップせざるをえなかったバイナリの一覧、および各バイナリをスキップした簡単な理由。

また、`appcert.????/objects/` のサブディレクトリ内にある次のファイルには、オブジェクトごとの情報が含まれます。

- check.demoted_symbols** 降格された Solaris シンボルと思われるシンボルの一覧
- check.dynamic.private** オブジェクトにより直接結合が行われる非公開の Solaris シンボルの一覧。
- check.dynamic.public** オブジェクトにより直接結合が行われる公開 Solaris シンボルの一覧。
- check.dynamic.unbound** `ldd -r` を実行したときに、動的リンカーによりバインドされなかったシンボルの一覧。利便性を高めるため、`file not found` という記載のある `ldd` 出力行も含まれます。
- summary.dynamic** プリント形式で書き込まれた、検査済みオブジェクトの動的バインドのサマリー (各 Solaris ライブラリから使用される公開シンボルと非公開シンボルのテーブルも含まれる)。

ほかのファイルは、appcert によって内部的に使用される一時ファイルです。

出力メッセージ

非公開シンボルの使用

非公開シンボルは、開発者や外部での使用が意図されていない Solaris ライブラリの関数またはデータ変数です。これらのシンボルは、Solaris ライブラリが相互の呼び出しや通信に使用するインタフェースです。これらのシンボルには、**pv(1)** の出力内でシンボルのバージョン名 `SUNWprivate` が付けられます。

非公開シンボルにより、セマンティック動作が変更されるだけでなく、完全に削除されてしまう可能性もあるため (「降格」または「非推奨」のシンボル)、アプリケーションがどの非公開シンボルにも依存しないようにしてください。

降格シンボル

降格シンボルとは、Solaris ライブラリの関数またはデータ変数のうち、かつてはそのライブラリに対して非公開で、その後の Solaris リリースで削除された (または、適用範囲がそのライブラリに対してローカルになった可能性のある) 関数また

はデータ変数です。アプリケーションがこれらの降格シンボルのいずれかへの呼び出しを直接行なった場合、シンボルが削除されたりリリースまたはそれ以降のリリースでは呼び出しに失敗します (再配置エラー)。

まれに、降格シンボルが以降のリリースで復帰することがありますが、それでもなお、アプリケーションが動作しないリリースが存在することに変わりはありません。

Sun Microsystems Inc. は、Solaris 2.5.1 から 2.6 への移行でライブラリスコープのほとんどを実行しました。このアクションは、バイナリの安定性を向上させるために実行されました。これらの内部インタフェースを完全に見えなくする (つまり、動的にリンクできなくする) ことで、開発者が不注意または故意にこれらのインタフェースを呼び出すことがなくなります。詳細は、『[リンカーとライブラリ](#)』のバージョン管理の章を参照してください。

非結合シンボル

非結合シンボルとは、アプリケーションにより参照され、appcert の実行時に動的リンカーが解決できなかったライブラリシンボル (つまり関数またはデータ) のことです。注: appcert はアプリケーションを実際に行うわけではないため、動的リンクに影響を及ぼす一部の環境特性が適切に設定されない場合があります。

非結合シンボルは、バイナリ安定性問題があることを必ずしも示すわけではありません。appcert の実行時に、実行時動的リンカーがそれらのシンボルを解決できなかったことを示すだけです。

非結合シンボルが生成される原因として、LD_LIBRARY_PATH が正しく設定されていない場合があります。これが正しく設定されていることを確認し、すべてのバイナリオブジェクトが依存関係にあるすべてのライブラリ (製品固有のライブラリ、Solaris ライブラリ、または他社製のライブラリ) を検索できるようにしてください。その後、appcert を再度実行します。

シェルスクリプトを記述して環境を正しく設定してから、チェックするバイナリ上で appcert を実行する際に役立つ場合があります。

非結合シンボルが生成される別のよくある原因として、構築時にコンパイラおよび [ld\(1\)](#) に `-l` スイッチが指定されなかったために、テスト対象の共有オブジェクトで動的依存関係が記録されなかった場合があります。このため、共有オブジェクトでは、リンク対象の実行可能ファイルに適切な依存関係が記録されている必要があります。

このような共有オブジェクトでは、標準的な方法でリンクを設定する (実行可能ファイルの構築時に指定する) か、または動的に開く (実行可能ファイルが、実行中のある時点で共有オブジェクトに対して [dlopen\(3C\)](#) を呼び出すなど) ことができます。どちらの場合でも、appcert の実行時に非結合シンボルが生成される可能性があります。前者の場合は、通常、appcert の実行前に LD_LIBRARY_PATH を適切に設定することによって解決できます。後者の場合 (dlopen) は、解決が困難になるのが普

通です。状況によっては、LD_PRELOAD を適切に設定して必要なライブラリを事前に読み込むことができますが、この方法が常にうまくいくとは限りません。

環境が正しく設定され、非結合シンボルが生成されないようになっていることを確認するにはどうすればよいでしょうか。バイナリ上で `ldd -r` を実行しても、「file not found」エラーや「symbol not found」エラーが出力されないように設定する必要があります。動的リンクの詳細については、[ld.so.1\(1\)](#) および [ldd\(1\)](#) を参照してください。

いずれの場合でも、appcert は、非結合シンボルがより深刻な問題を示していると考えられる場合に警告フラグを設定します。非結合シンボルは、降格シンボル（ライブラリから削除されたか、適用範囲がローカルに設定されたシンボル）への依存関係を示す場合があります。降格シンボルへの依存関係は、深刻なバイナリ安定性問題につながります。

ただし、環境を適切に設定すれば、ほとんどの非結合シンボルは取り除かれます。通常、可能な場合には常に、構築時にライブラリの依存関係を記録しておくことをお勧めします。これは、バイナリオブジェクトがより明確に定義され、自己完結されるようにするためです。また、共有オブジェクトを構築する際に `-z defs` フラグを使用して、コンパイル時にすべてのシンボル解決を強制することをお勧めします。詳細については、[ld\(1\)](#) を参照してください。

結合が見つからない

appcert は、テスト対象のバイナリオブジェクトごとに `/bin/ldd -r` を実行します。これにより、環境変数 `LD_DEBUG="files,bindings"` が設定されます。（詳細については、[ldd\(1\)](#) および [ld.so.1\(1\)](#) を参照してください）。何らかの理由でこのコマンドが失敗すると、appcert は動的シンボル結合情報を保持せず、「no bindings」が検出されます。

次のいずれかが真の場合、appcert の実行に失敗する可能性があります。

- バイナリオブジェクトに読み取り権がありません。
- バイナリオブジェクトが SUID または SGID で、かつユーザーが十分な特権を備えていません。
- バイナリオブジェクトが、実行権ビットの設定されていない実行可能ファイルです。
- バイナリオブジェクトが完全に静的にリンクされています。
- バイナリオブジェクトにライブラリ依存関係の情報が記録されていません。

そのほかにも理由は考えられます（メモリー不足など）。一般に、このフラグは、アクセス権または環境のために appcert がオブジェクトを完全には検査できなかったことを意味します。動的結合が記録できるようにするため、アクセス権または環境の変更を試みてください。

廃止ライブラリ	廃止ライブラリとは、使用が推奨されておらず、将来のリリリースで Solaris から完全に削除される可能性のあるライブラリのことです。これらに依存するアプリケーションは将来の Solaris リリースで動作しなくなる可能性があるため、appcert はこれらにフラグを設定します。廃止ライブラリのインタフェース (非公開インタフェースを含む) はすべて凍結されており、変更されることはありません。
sys_errlist/sys_nerr の使用	sys_errlist シンボルまたは sys_nerr シンボルを直接使用すると、参照が sys_errlist 配列の末尾を越えてしまう危険があります。これらのシンボルは、32 ビットバージョンの Solaris では非推奨であり、64 ビットバージョンにはまったく存在しません。代わりに strerror(3C) を使用してください。
強いシンボルの使用と弱いシンボルの使用	「弱い」シンボル (socket など) に関連付けられている「強い」シンボル (_socket など) は、非公開 (将来動作が変更される可能性のあるもの) として予約されています。アプリケーションから直接参照するシンボルは、弱いシンボルだけにしてください (通常、強いシンボルは「_」で始まります)。 注: 特定の構築環境では、ソースコードが非公開シンボルを参照していないように思える場合でも、弱い (公開) シンボルの依存関係ではなく、強い (非公開) シンボルの依存関係がバイナリ内に記録されます。その場合でも、この発生原因を突き止めて、依存関係を修正するようにしてください。
注意事項	appcert は、チェック対象のアプリケーションを実行するのと同じ環境で実行する必要があります。そうでない場合は、Solaris ライブラリ内のインタフェースへの参照を正しく解決できない場合があります。次の手順を実行します。 1. LD_LIBRARY_PATH およびその他のすべての環境特性が、アプリケーション実行時の設定になっていることを確認します。また、製品の一部である非 Solaris 共有オブジェクトの存在するディレクトリが含まれていて、参照時に検出できるようになっていることを確認します。 2. チェック対象のすべてのバイナリが、次のようになっていることを確認します。 ■ 動的にリンクされた ELF オブジェクトである ■ 実行可能ファイルに実行権が設定されている (共有オブジェクトの場合は不要) ■ SUID root ではない (それ以外の場合、チェックを行うには自分が root である必要があるため、必要に応じて SUID ではないコピーを作成してチェックする必要があります)。 シェルスクリプトを記述して環境を正しく設定してから appcert を実行する際に役立つ場合があります。 発生する可能性のある問題は、次のとおりです。 ■ appcert が、Solaris ライブラリの一部であると思われる非結合シンボルを報告します。

これは、Solaris 依存関係が記録されていない共有オブジェクトにアクセスするためにアプリケーションが [dlopen\(3C\)](#) を使用することが原因と思われます。動的リンカーが共有オブジェクト上呼び出されることはなく、Solaris シンボル結合の解決に使用できる依存関係情報がほかに存在しないため、この場合には appcert はシンボルの使用を解決できません。これは、Solaris 以外のシンボルでも発生することがあります。

この問題を回避するには、共有オブジェクトの構築時に、コンパイル行で `-llib` オプションを使用して Solaris ライブラリへの依存関係が明示的に記録されていることを確認します (cc(1) および [ld\(1\)](#) を参照)。

- appcert が、アプリケーションのソースコード内で参照されていない Solaris 非公開シンボルがアプリケーションで使用されていることを報告します。

ほとんどの場合、この問題は、そのシンボルを参照する Solaris ライブラリの静的リンクが原因です。appcert は動的リンカーを使用してシンボルを解決するため、appcert は、静的にリンクされたライブラリをアプリケーションコードの一部とみなします (ある意味では事実ですが)。これは、Solaris ヘッダーファイルでマクロ置換を実行した結果として発生することもあります。

この問題を回避するには、可能なかぎり、Solaris ライブラリアーカイブをアプリケーション内に静的にリンクしないようにします。

- appcert がライブラリを Solaris の一部として認識しません。
廃止された Solaris ライブラリの中には、シンボルのバージョン管理前に廃止された非常に古いライブラリがあります。このため、appcert はそのようなライブラリを Solaris の一部として認識できません。

使用上の留意点 「公開」および「非公開」という用語を「安定」および「不安定」と同等の意味で使用することは、混乱を招く可能性があります。特に、実験的なインタフェースや開発中のインタフェースは、ドキュメント化され、使用が推奨されているという意味では公開されていると言えます。ただし、そのようなインタフェースを使用して構築されたアプリケーションは、以降のリリースでは動作しない可能性があるという意味で不安定です。したがって、appcert の目的から、これらのインタフェースは開発が終了するまで非公開と分類されます。逆に、廃止されたインタフェースは、過去に公開されて安定しており、appcert では引き続き公開として扱われている場合でも、最終的には消滅するという意味で不安定です。さいわいなことに、このどちらかの状況が存在することはめったにありません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/appcert
インタフェースの安定性	確実

関連項目

[cc\(1\)](#), [find\(1\)](#), [isalist\(1\)](#), [ld\(1\)](#), [ldd\(1\)](#), [ld.so.1\(1\)](#), [pvs\(1\)](#), [dlopen\(3C\)](#), [strerror\(3C\)](#),
[Intro\(4\)](#), [attributes\(5\)](#)

名前 appttrace – Solaris 共有ライブラリに対するアプリケーション関数呼び出しの追跡

形式 appttrace [-f] [-F [!] *tracefromlist*] [-T [!] *tracetolist*] [-o *outputfile*] [[-tv] [!] *call* ,...] *command* [

機能説明

appttrace ユーティリティは、*command* により指定された実行可能プログラムを実行して、プログラムの *command* が Solaris 共有ライブラリに対して行うすべての関数呼び出しを追跡します。appttrace における追跡では、プログラムによる関数呼び出しごとに、呼び出されたライブラリインタフェースの名前、渡された引数の値、および戻り値が報告されます。

デフォルトでは、appttrace は、実行可能オブジェクトからそれが依存するすべての共有オブジェクトに対する直接呼び出しを追跡します。間接呼び出し (つまり、実行可能プログラムが依存する共有オブジェクト間で行われた呼び出し) は、デフォルトでは報告されません。

追加共有オブジェクトとの間で実行される呼び出しは、-F または -T オプションを指定することにより追跡できます (下記参照)。

デフォルトのレポート形式は、呼び出しごとに 1 行であり、参照によって渡された引数またはデータ構造体のフォーマット出力はありません。

追加引数の詳細を示すフォーマット出力は、-v オプションを使用すると得られます (下記参照)。

デフォルトでは、共有オブジェクトによって提供されるインタフェースは、呼び出された場合、すべて追跡されます。ただし、-t または -v オプションのいずれか、あるいは両方を使用することにより、追跡するインタフェース群を制限できます。

通常、実行時にリンクされる動的オブジェクト (実行可能オブジェクトと依存対象のすべての共有オブジェクト) 間の呼び出しを追跡することができるため、追跡された各呼び出しのレポートには、呼び出し元のオブジェクトの名前が示されます。

appttrace は、プロシージャリンケージテーブルを介して動的オブジェクト間で発生するすべてのプロシージャ呼び出しを追跡します。そのため、そのテーブルを介して結合されているプロシージャ呼び出しだけが追跡されます。『[リンカーとライブラリ](#)』を参照してください。

オプション

サポートしているオプションは、次のとおりです。

- f [fork\(2\)](#) によって作成されたすべての子プロセスを追跡します。また、各行の先頭にプロセス ID を表示します。
- F [!] *tracefromlist* コンマで区切られた共有オブジェクトのリストからの呼び出しを追跡します。これらの共有オブジェクトからの呼び出しだけが追跡されます。デフォルトでは、メインの実行可能オブジェクトからの呼び出しだけが追跡されます。共有オブジェクトのベース名だけが必要になります。たとえば、*libc* は

`/usr/lib/libc.so.1` に一致します。また、`fnmatch(5)` で説明されているように、シェルのワイルドカード文字がサポートされています。`!` が前に付いたリストは、そこからの呼び出しが追跡から除外されるオブジェクト群を定義します。`command` からの呼び出しの追跡が必要な場合、`command` は `tracefromlist` のメンバーでなければなりません。

- `-o outputfile` `appttrace` の出力を `outputfile` に送ります。デフォルトでは、`appttrace` の出力は、追跡されるプロセスの `stderr` ストリームに送られます。
- `-t [!]call, ...` 関数呼び出しを追跡、または除外します。コンマで区切られたリスト `call` に指定された呼び出しが追跡されます。リストが `!` で始まる場合、指定した関数呼び出しが追跡出力から除外されます。デフォルト値は `-t *` です。シェルスタイルのワイルドカードを使用できます。
- `-T [!]tracetolist` コンマで区切られた共有オブジェクトのリストへの呼び出しを追跡します。デフォルトでは、すべての共有オブジェクトへの呼び出しが追跡されます。前述のとおり、ベース名だけが必要であり、ワイルドカードを使用できます。`!` が前に付いたリストは、そこへの呼び出しが追跡から除外されるオブジェクト群を示します。
- `-v [!]call, ...` 指定された関数呼び出しの引数と戻り値を示す冗長な、フォーマット出力を提供します (前述の `-t` オプションと同様)。`truss(1)` と違って、`-v` オプションで指定した呼び出しは、`-t` オプションで指定する必要はありません。たとえば、`appttrace -v open` は、`truss -t open -v open` と同等です。

使用例

例1 `date` コマンドの追跡

```
% appttrace date
-> date      -> libc.so.1:atexit(0xff3bf9ac, 0x22000, 0x0) ** NR
-> date      -> libc.so.1:atexit(0x11550, 0xfefeeef80, 0xab268) ** NR
-> date      -> libc.so.1:setlocale(0x6, 0x11560, 0x0) ** NR
-> date      -> libc.so.1:textdomain(0x11564, 0xfefce156, 0xff160200) ** NR
-> date      -> libc.so.1:int getopt(int = 0x1,
                    const char * * = 0xffbffa5c,
                    const char * = 0x11574 "a:u")
<- date      -> libc.so.1:getopt() = 0xffffffff
-> date      -> libc.so.1:time_t time(time_t * = 0x225c0)
<- date      -> libc.so.1:time() = 0x41ab6e82
-> date      -> libc.so.1:char * nl_langinfo(nl_item = 0x3a)
<- date      -> libc.so.1:nl_langinfo() = 0xfefd3e10
-> date      -> libc.so.1:struct tm * localtime(const time_t * = 0x225c0)
<- date      -> libc.so.1:localtime() = 0xff160240
```

例1 date コマンドの追跡 (続き)

```

-> date      -> libc.so.1:memcpy(0xffbfff9cc, 0xff160240, 0x24) ** NR
-> date      -> libc.so.1:size_t strftime(char * = 0x225c4 "",
                                size_t = 0x400,
                                const char * = 0xfefd3e10 "%a %b %e %T %Z %Y",
                                const struct tm * = 0xffbfff9cc)
<- date      -> libc.so.1:strftime() = 0x1c
-> date      -> libc.so.1:int puts(const char * = 0x225c4
                                "Mon Nov 29 10:46:26 PST 2004")
                                Mon Nov 29 10:46:26 PST 2004
<- date      -> libc.so.1:puts() = 0x1d
-> date      -> libc.so.1:exit(0x0, 0x22400, 0x0) ** NR

```

例2 冗長性セットによる特定のインタフェース群の追跡

% **appttrace -v localtime, strftime, puts date**

```

-> date      -> libc.so.1:struct tm * localtime(const time_t * = 0x225c0)
                                arg0 = (const time_t *) 0x225c0
                                return = (struct tm *) 0xff160280 (struct tm) {
                                tm_sec: (int) 0x4
                                tm_min: (int) 0x34
                                tm_hour: (int) 0xa
                                tm_mday: (int) 0x1d
                                tm_mon: (int) 0xa
                                tm_year: (int) 0x68
                                tm_wday: (int) 0x1
                                tm_yday: (int) 0x14d
                                tm_isdst: (int) 0
                                }
<- date      -> libc.so.1:localtime() = 0xff160280
-> date      -> libc.so.1:size_t strftime(char * = 0x225c4 "",
                                size_t = 0x400,
                                const char * = 0xfefd3e10 "%a %b %e %T %Z %Y",
                                const struct tm * = 0xffbfff99c)
                                arg0 = (char *) 0x225c4 ""
                                arg1 = (size_t) 0x400
                                arg2 = (const char *) 0xfefd3e10 "%a %b %e %T %Z %Y"
                                arg3 = (const struct tm *) 0xffbfff99c (struct tm) {
                                tm_sec: (int) 0x4
                                tm_min: (int) 0x34
                                tm_hour: (int) 0xa
                                tm_mday: (int) 0x1d
                                tm_mon: (int) 0xa
                                tm_year: (int) 0x68
                                tm_wday: (int) 0x1
                                tm_yday: (int) 0x14d
                                tm_isdst: (int) 0

```

例2 冗長性セットによる特定のインタフェース群の追跡 (続き)

```

    }
    return = (size_t) 0x1c
<- date      -> libc.so.1:strftime() = 0x1c
-> date      -> libc.so.1:int puts(const char * = 0x225c4
                        "Mon Nov 29 10:52:04 PST 2004")
    arg0 = (const char *) 0x225c4 "Mon Nov 29 10:52:04 PST 2004"
                        Mon Nov 29 10:52:04 PST 2004
    return = (int) 0x1d
<- date      -> libc.so.1:puts() = 0x1d

```

** NR - 関数呼び出しの戻り値は追跡できません。

ファイル

appttrace の基本実行時サポートは Solaris 実行時リンカー ([ld.so.1\(1\)](#)) のリンク監査機能によって提供され、appttrace コマンドは、`/usr/lib/abi` に格納されている監査オブジェクト (`appttrace.so.1`) に依存してこの機能を使用します。

制限事項

一般に、appttrace は、変数引数リストを受け入れる関数への呼び出しを追跡することはできません。一部の特別な場合、特に `printf` および `scanf` ファミリでは、この制限事項に対処するために、適切なコーディング方法がいくつか用いられています。

appttrace ユーティリティは、戻り値のタイプが `struct` または `union` である関数呼び出しの戻り値を追跡することができません。

スタックを調べようとする関数、または呼び出し元に関する情報を抽出しようとする関数は追跡できません。たとえば、`[gs]setcontext()`、`[sig]longjmp()`、`[sig]setjmp()`、`vfork()` がこれに相当します。

[exit\(2\)](#) などの値を返さない関数では、戻り値については追跡されません。

セキュリティ上の理由から、適切な特権を持つプロセスだけが appttrace を使用して `setuid/setgid` プログラムを追跡できます。

`<varargs.h>` の取り込みを必要とする関数 ([vwprintw\(3XCURSES\)](#) や [vwscanw\(3XCURSES\)](#) など) を追跡するときは、引数の出力は書式化されません。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/appttrace (32 ビット) SUNWcstlx (64 ビット)

属性タイプ	属性値
インタフェースの安定性	不確実

関連項目 [ld.so.1\(1\)](#), [truss\(1\)](#), [vwprintw\(3XCURSES\)](#), [vwscanw\(3XCURSES\)](#), [attributes\(5\)](#), [fnmatch\(5\)](#)

『リンカーとライブラリ』

名前	apropos – キーワード検索による検索コマンド
形式	apropos keyword...
機能説明	apropos ユーティリティーは、<i>keyword</i> を含んでいる各マニュアルページの名前、セクション番号、サブセクション名、<i>keyword</i>、および簡単な説明を表示します。 これらの情報が含まれるインデックスファイルは、man(1) および man(5) で説明されているように SMF サービスによって自動的に作成されるか、または catman(1M) と <code>-w</code> オプションを使用して手動で作成します。 各語は個別に処理され、大文字小文字の区別はありません。英単語の語幹解釈および部分一致もサポートされます。ほかの単語の一部である単語も考慮されます。たとえば、<code>compile</code> を検索するとき、<code>apropos</code> は <code>compiler</code> のすべてのインスタンスも検索します。 <code>apropos</code> は機能的には <code>-k</code> オプションを指定した man(1) コマンドと同等であるため、詳細は man(1) を参照してください。
使用例	例1 名前行に <i>keyword</i> が含まれているマニュアルページを検索する次を実行してください。 <pre>example% apropos password</pre> および <pre>example% apropos editor</pre> 行が <code>filename(section) ...</code> で始まっている場合、次を実行できます。 <pre>man -s section filename</pre> これにより、その <i>filename</i> が示すファイルのマニュアルページを出力することができます。 例2 <code>printf()</code> サブルーチンのマニュアルページを検索する次を実行してください。 <pre>example% apropos format</pre> そのあとで <pre>example% man -s 3s printf</pre> これにより <code>printf()</code> サブルーチンのマニュアルページが得られます。
ファイル	<code>/usr/share/man/man_index/*</code> 目次とキーワードのデータベース。 次のファイルが生成されます。 ■ <code>/usr/share/man/man_index/man.idx</code>

- /usr/share/man/man_index/man.dic
- /usr/share/man/man_index/man.frq
- /usr/share/man/man_index/man.pos

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools
CSI	有効
インタフェースの安定性	確実

関連項目 [man\(1\)](#), [whatis\(1\)](#), [catman\(1M\)](#), [attributes\(5\)](#), [man\(5\)](#)

名前 ar – 移植可能なアーカイブまたはライブラリの保守

形式

```
/usr/bin/ar -d [-SVv] archive file...
/usr/bin/ar -m [-abiSVv] [posname] archive file...
/usr/bin/ar -p [-SsVv] archive [file]...
/usr/bin/ar -q [-cSVv] archive file...
/usr/bin/ar -r [-abciuSVv] [posname] archive file...
/usr/bin/ar -t [-SsVv] archive [file]...
/usr/bin/ar -x [-CSsTVv] archive [file]...
/usr/xpg4/bin/ar -d [-SVv] archive file...
/usr/xpg4/bin/ar -m [-abiSVv] [posname] archive file...
/usr/xpg4/bin/ar -p [-SsVv] archive [file]...
/usr/xpg4/bin/ar -q [-cSVv] archive file...
/usr/xpg4/bin/ar -r [-abciuSVv] [posname] archive file...
/usr/xpg4/bin/ar -t [-SsVv] archive [file]...
/usr/xpg4/bin/ar -x [-CSsTVv] archive [file]...
```

機能説明

ar ユーティリティーは、1つのアーカイブファイルとして結合されたファイル群の保守を実行します。このコマンドは、おもにライブラリファイルの作成および更新のために使用されますが、他の似たような目的で使うこともできます。ar が使用するマジック文字列とファイルヘッダーは、印刷可能な ASCII 文字で構成されます。アーカイブが印刷可能なファイルの集まりであれば、アーカイブ全体も印刷可能です。

ar はアーカイブを作成する際に、すべてのマシン間で移植できるような形式でヘッダーを生成します。移植可能なアーカイブの形式や構造については、[ar.h\(3HEAD\)](#) で詳しく説明されています。アーカイブシンボルテーブル (ar.h(3HEAD) で説明) は、リンクエディタ [ld\(1\)](#) がオブジェクトファイルのライブラリ群にまたがった複数の受け渡しを効率よく有効にするために使用します。アーカイブシンボルテーブルは、アーカイブ中にオブジェクトファイルが1つでも存在している場合にだけ、ar により作成または保守されます。アーカイブシンボルテーブルは、特殊な名前が付けられたファイル内にあり、そのファイルは常にアーカイブの先頭にあります。このファイルに関してはユーザーに対しては表示されず、ユーザーがアクセスすることはできません。ar を使ってアーカイブを生成または更新するたびに、シンボルテーブルは再構築されます。また後述する -s オプションを使えば、シンボルテーブルを強制的に再構築することができます。

オプション

サポートしているオプションは、次のとおりです。

- a
file が示す新たなファイルを、*archive* が示すアーカイブ中の *posname* が示すファイルの後に配置します。
- b
file が示す新たなファイルを、*archive* が示すアーカイブ中の *posname* が示すファイルの前に配置します。
- c
デフォルトでは、*archive* で示すアーカイブが生成されるときに標準エラー出力に診断メッセージが書き込まれますが、このオプションはそのメッセージ出力を抑制します。
- C
抽出されたファイルが、ファイルシステム中の同一名のファイルを置き換えないようにします。特に -T オプションも指定した場合にはファイル名の後半が切り捨てられてしまうので、この C オプションにより前半部分が一致するファイルを置き換えてしまうことを防げます。
- d
file が示すファイルを *archive* が示すアーカイブから削除します。
- i
file が示す新たなファイルを、*archive* が示すアーカイブ中の *posname* が示すファイルの前に配置します。これは -b オプションと同等です。
- m
file が示すファイルを移動します。-a、-b、または -i が *posname* オペランドと一緒に指定された場合、-m オプションは *file* を新しい位置に移動します。そうでない場合は、*archive* が示すアーカイブの末尾に移動します。-
- p
archive が示すアーカイブ中の、*file* が示すファイルの内容を標準出力に印刷します。*file* が 1 つも指定されていなければ、そのアーカイブ中のすべてのファイルの内容が、アーカイブ内でのファイルの順序に従って出力されます。
- q
file が示すファイルを、*archive* が示すアーカイブの最後尾に追加します。位置を指定するオプション -a、-b、および -i はいずれも無効です。指定されたファイルと同じ名前のファイルがすでにそのアーカイブ中に存在しているかどうかは、チェックされません。このオプションを使えば二次的な動作を抑止できるので、大きなアーカイブを少しずつ作成していく場合に便利です。
- r
archive が示すアーカイブ中で、*file* が示すファイルの置換または追加を行います。指定したアーカイブが存在していなければ、新たにアーカイブを作成し (-c オプション指定時を除く) 診断メッセージを標準エラー出力に書き出します。アーカイブは存在していて、ファイル名が 1 つも指定されなかった場合の実行結果は定義されていません。ファイルの置換が発生しても、アーカイブ中での

ファイルの順序は変わりません。この `-r` オプションとともに `-u` オプションも指定されていると、最終更新日時がアーカイブ中の対応するファイルより新しいファイルだけが置き換えられます。`-a`、`-b`、または `-i` オプションが指定された場合、*posname* 引数の指定も必須となります。このとき、新規ファイルは `-posname` で示すファイルの後 (`-a`) または前 (`-b` または `i`) に置かれます。それ以外の場合、新規ファイルはアーカイブの最後尾に置かれます。

-s

`ar` を呼び出したコマンドに、アーカイブの内容を更新するようなオプションが指定されていない場合でも、アーカイブのシンボルテーブルを強制的に再構築します。このオプションは、アーカイブ上で `strip(1)` コマンドを実行した後でアーカイブシンボルテーブルの内容を復元するのに便利です。

-S

アーカイブシンボルテーブルの作成時に、64 ビット対応のシンボルテーブル形式の使用を強制します。デフォルトでは、4G バイト未満のすべてのアーカイブに 32 ビット形式が使用されます。32 ビットの制限を超える、より大きなアーカイブにはより大きな形式が使用されます。

-t

アーカイブの内容を示す目次を生成します。*file* で指定したファイルが目次に含まれます。*file* を 1 つも指定しないと、アーカイブ中の全ファイル名が、アーカイブ中での順番に含まれます。

-T

抽出されたファイルのアーカイブ名の長さが、ファイルシステムでサポートする最大長を超えている場合、超過した部分を切り捨てます。デフォルトでは、最大長を超える長さの名前を持つファイルを抽出しようとするときエラーとなり、ファイルの抽出は行われず診断メッセージが出力されます。

-u

古いファイルを更新します。`-r` オプションと一緒に指定すると、最終更新日時が *archive* 中の対応するファイルよりも新しいファイルについてだけ、置換が行われます。

-v

詳細な情報を出力します。他のオプション `-d`、`-r`、または `-x` も一緒に指定すると、アーカイブやファイルの生成と保守作業についてファイルごとの詳細な情報を出力します。`--p` オプションも一緒に指定すると、ファイルの内容の前にファイルの名前を標準出力に書き出します。`--t` オプションも一緒に指定すると、アーカイブ中のファイルに関する大量の情報が出力されます。`--x` オプションも一緒に指定すると、抽出作業の前にファイル名が出力されます。`-` アーカイブに書き込むと、メッセージを標準エラーに書き出します。-

-V

自身のバージョン番号を標準エラー出力に印刷します。

`/usr/xpg4/bin/ar` 次のオプションは、`/usr/xpg4/bin/ar` でサポートされています。

- v アーカイブに書き込んだ場合(メッセージを標準エラーに書き出しません)を除いて、`/usr/bin/ar` 版と同じです。
- x `file` で指定したファイル群を `archive` が示すアーカイブから抽出します。アーカイブの内容は変わりません。`file` を1つも指定しないと、アーカイブ中の全ファイルが抽出されます。抽出されたファイルの名前が、出力先のディレクトリがサポートしている最大長を超えている場合、結果は定義されていません。抽出されたファイルの最終更新時間は、アーカイブから抽出された時間に設定されます。

オペランド 次のオペランドがサポートされています。

`archive` アーカイブのパス名。

`file` パス名。アーカイブ中のファイル名と比較する際には、ファイル名の最終コンポーネントだけが使用されます。最終パス名コンポーネント ([basename\(1\)](#) を参照) に同一の名前が複数指定された場合、その結果は不確定です。アーカイブに追加または置換されたファイルの名前は、それが正当なものであれば、実装先システムのアーカイブの形式により切り捨てられることはありません。

`posname` アーカイブ中での配置位置を相対的に表すためのファイルの名前。詳しくは `-m` と `-r` オプションの説明を参照してください。

環境変数 `ar` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、LC_TIME、および NLSPATH。

`TMPDIR` 一時ファイルが存在する場合にデフォルトのディレクトリにオーバーライドするパス名を調べます。

`TZ` `ar -tv` により書き込まれる日時文字列の計算に使用されるタイムゾーンを調べます。TZ が指定されていない、または NULL の場合は、デフォルトのタイムゾーンが使用されます。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

<code>/usr/bin/ar</code>	属性タイプ	属性値
	使用条件	system/linker
	インタフェースの安定性	確実

/usr/xpg4/bin/ar

属性タイプ	属性値
使用条件	system/xopen/xcu4
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目

[basename\(1\)](#), [cpio\(1\)](#), [elffile\(1\)](#), [file\(1\)](#), [ld\(1\)](#), [lorder\(1\)](#), [strip\(1\)](#), [tar\(1\)](#),
[ar.h\(3HEAD\)](#), [a.out\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

引数リスト中に同じファイル名を2度記述すると、アーカイブ中にそのファイルが2個置かれます。

慣習として、アーカイブのファイル名には `.a` という接尾辞を付けるのが一般的です。

ELF オブジェクトをアーカイブファイルに挿入する際、これらのオブジェクトを8バイトの境界に合わせるため、`ar` により文字 `\n` が追加される場合があります。こうしたパディングにより、[ld\(1\)](#) がアーカイブにアクセスする際の効率が向上します。この方法でパディングが行われるのは、ELF オブジェクトファイルだけです。その他のアーカイブメンバーは変更されません。この種のパディングが行われたオブジェクトをアーカイブから抽出するとき、生成される出力にパディングは含まれません。

`ar` を個別に呼び出して個々のファイルを既存のアーカイブに挿入するよりも、新しいアーカイブを最初から作成する方が時間を短縮できます。可能であれば、既存のアーカイブを削除し、`ar` を1回呼び出してアーカイブを再作成することをお勧めします。

1つのアーカイブ全体のサイズは、4Gバイトを超えてもかまいません。ただし、アーカイブ内の個別のファイルは、アーカイブファイルの形式により4Gバイト以内に制限されています。[ar.h\(3HEAD\)](#) を参照してください。

アーカイブ内の各ファイルのユーザー ID およびグループ ID の最大値は、アーカイブファイルの形式により6桁の10進数に制限されています。999999を超えるユーザー ID やグループ ID は、自動的にユーザー ID `nobody` (60001) またはグループ ID `nobody` (6001) に設定されます。[ar.h\(3HEAD\)](#) を参照してください。

名前

arch - 現在のホストのアーキテクチャーの表示

形式

arch [-k | *archname*]

機能説明

arch ユーティリティは、現在のホストシステムのアプリケーションアーキテクチャーを表示します。このコマンドは、従来オプションなしで使用されてきたため、SunOS 5.x SPARC ベースのシステムはすべて、アプリケーションアーキテクチャーとして sun4 を返します。このコマンドは使用しないようにしてください。(「注意事項」のセクションを参照)

システムは、どの実行可能プログラムがどのマシンで実行されるかを定義するアーキテクチャーによって大まかに分類できます。カーネルアーキテクチャーとアプリケーションアーキテクチャー(通常は単に「アーキテクチャー」)は、区別することができます。ベースとなるハードウェアが異なるために稼動するカーネルが異なるマシンでも、同じアプリケーションプログラムを実行できる場合があります。

オプション

-k sun4u などのカーネルアーキテクチャーを表示します。このオプションは、マシンで稼動する特定の SunOS カーネルを定義し、カーネルに明示的に依存するプログラム(たとえば [ps\(1\)](#)) に対してのみ意味を持ちます。

オペランド

次のオペランドを指定できます。

archname *archname* を使用すると、このアプリケーションアーキテクチャー用のアプリケーションバイナリを現在のホストシステムで実行できるかどうかを判断できます。*archname* は、sun4、i86pc などの有効なアプリケーションアーキテクチャーでなければなりません。

archname 用のアプリケーションバイナリを現在のホストシステムで実行できる場合は、TRUE (0) が返されます。そうでない場合は、FALSE (1) が返されます。

終了ステータス

次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[mach\(1\)](#), [ps\(1\)](#), [uname\(1\)](#), [attributes\(5\)](#)

注意事項

このコマンドは以前のリリースとの互換性のために提供されており、使用しないようにしてください。代わりに、[uname](#) コマンドを使用してください。使用法については、[uname\(1\)](#) を参照してください。

名前	as – アセンブラ
形式	<pre> SPARC as [-hwcap={1 0}] [-L] [-m] [-m32] [-m64] [-n] [-o outfile] [-ul] [-P] [-Dname] [-Dname=def] [-Ipath] [-Uname....] [-Q[y n]] [-s] [-S[a b c l A B C L]] [-V] [-xarch=v] [-xF] [-Y[m c],path] [-YI,path] filename... x86 as [-a32] [-m] [-m32] [-m64] [-n] [-H] [-nH] [-o outfile] [-P] [-Dname] [-Dname=def] [-Ipath] [-Uname....] [-KPIC] [-Q[y n]] [-s] [-S[a b c l A B C L]] [-V] [-xchip=v] [-xmodel=[a]] [-Y[m d],path] [-YI,path] filename... </pre>
機能説明	as コマンドは、アセンブリ言語ソースファイルからオブジェクトファイルを生成します。
オプション	このセクションは、3 つに分かれています: ■ 共通オプション (SPARC と x86 の両方に共通のオプション) ■ SPARC のオプション ■ x86 のオプション
共通オプション	-Dname -Dname=def -P オプションが有効な場合、これらのオプションは、as コマンドでは解釈されずに cpp(1) プリプロセッサに渡されます。そうでない場合は無視されます。 -Ipath -P オプションが有効な場合、このオプションは、as コマンドでは解釈されずに cpp(1) プリプロセッサに渡されます。そうでない場合は無視されます。 -i as に対し、プリプロセッサからの行番号情報を無視するよう指示します。 -m アセンブラへの入力時に m4(1) マクロプロセッサを実行します。 -m32 -m64 32 ビットまたは 64 ビット ELF 形式のオブジェクトコードを生成します。 -n アセンブル実行時にすべての警告を抑制します。

-o *outfile*

アセンブリの出力を *outfile* に格納します。デフォルトでは、出力ファイル名は、入力ファイル名から *.s* 接尾辞 (もしあれば) を削除して、*.o* 接尾辞を付加することによって作成されます。

-P

アセンブル中のファイルに対して、C プリプロセッサ **cpp(1)** を実行します。このプリプロセッサは、入力ファイルを連結したのではなく、各入力ファイルに対して実行されます。このプリプロセッサによる出力結果はアセンブラに渡されます。

-Q[y|n]

y オプションを指定した場合、出力オブジェクトファイルの注釈セクションに「アセンブラバージョン」情報が生成されます。*n* オプションを指定すると、情報は抑制されます。

-S[a|b|c|l|A|B|C|L]

発行されたコードの逆アセンブリを標準出力に送ります。次の各文字を *-s* オプションに付けると、次の処理が実行されます。

- a アドレスによる逆アセンブル
- b *.bof* による逆アセンブル
- c 注釈による逆アセンブル
- l 行番号による逆アセンブル。

大文字を指定すると、対応するオプションが無効になります。デフォルトは *-Sc* です。

-s

すべてのスタブを *.stabs* セクションに置きます。デフォルトでは、スタブは、静的リンカー **ld(1)** によって最終実行中に削除される *stabs.excl* セクションに置かれます。*-s* オプションを使用すると、*.stab* セクションが静的リンカーによって削除されないため、スタブは最終実行可能プログラム中に残ります。

-Uname

-P オプションが有効な場合、このオプションは、**as** コマンドでは解釈されずに **cpp(1)** プリプロセッサに渡されます。そうでない場合は無視されます。

-Ym,path

使用する *m4* のバージョンへのパスを指定します。

-YI,path

#include ヘッダーファイルを検索するパスを示します。

SPARC のオプション

-hwcap={1|0}

ハードウェア機能セクションの生成を有効 (1) にするか、抑制 (0) します。デフォルトでは、このセクションを生成します。

-L

領域を節約するために通常は破棄される一時ラベルを含む、すべてのシンボルを ELF シンボルテーブルに保存します。

-ul

すべての未定義シンボルをローカルとして扱います。

-Yc,path

使用する cpp のバージョンへのパスを指定します。

-xarch=sparc

SPARC-V9 アーキテクチャーで定義されている命令をアセンブラが受け入れることができるようにします。生成されるオブジェクトコードは、-m32 では ELF32 形式、-m64 では ELF64 形式になります。これは、Oracle Solaris V8 システム (V8 プロセッサを持つマシン) では実行されません。これは、Oracle Solaris V8+ システムで実行されます。

-xarch=sparcvis

アセンブラが、SPARC-V9 アーキテクチャーで定義されている命令、および Visual Instruction Set (VIS) バージョン 1.0 の命令を受け入れることを可能にします。生成されるオブジェクトコードは、-m32 では V8+ ELF32 形式、-m64 では ELF64 形式になります。これは、V8 プロセッサを使用する Oracle Solaris システムでは実行されません。これは、V8+ プロセッサを使用する Oracle Solaris システムで実行されます。

-xarch=sparcvis2

アセンブラが、UltraSPARC-III 拡張機能を使用して、SPARC-V9 アーキテクチャーで定義された命令と、Visual Instruction Set (VIS) バージョン 2.0 の命令を受け入れることを可能にします。生成されるオブジェクトコードは、-m32 では V8+ ELF32 形式、-m64 では ELF64 形式になります。

-xarch=sparcvis3

SPARC-V9 ISA の SPARC VIS バージョン 3 (SPARC-V9 命令セットの命令) に加え、Visual Instruction Set (VIS) バージョン 1.0 を含む UltraSPARC 拡張機能、Visual Instruction Set (VIS) バージョン 2.0 を含む UltraSPARC-III 拡張機能、積和演算命令、および Visual Instruction Set (VIS) バージョン 3.0 で定義されている命令を受け入れます。

-xarch=sparcfmaf

SPARC-V9 ISA の sparcfmaf バージョンに加え、Visual Instruction Set (VIS) バージョン 1.0 を含む UltraSPARC 拡張機能、Visual Instruction Set (VIS) バージョン 2.0 を含む UltraSPARC-III 拡張機能、および浮動小数点積和演算用 SPARC64 VI 拡張機能で定義されている命令を受け入れます。

-xarch=sparcima

SPARC-V9 ISA の sparcima バージョン (SPARC-V9 命令セットの命令) に加え、Visual Instruction Set (VIS) バージョン 1.0 を含む UltraSPARC 拡張機能、Visual

Instruction Set (VIS) バージョン 2.0 を含む UltraSPARC-III 拡張機能、浮動小数点積和演算用 SPARC64 VI 拡張機能、および整数積和演算用 SPARC64 VII 拡張機能で定義されている命令を受け入れます。

-xarch=sparc4

SPARC-V9 ISA の sparc4 バージョンで定義された命令 (SPARC-V9 命令セットの命令)、VIS 1.0 を含む拡張機能で定義された命令、VIS 2.0 を含む UltraSPARC-III 拡張機能で定義された命令、浮動小数点積和演算命令、VIS 3.0、および SPARC4 命令を受け入れます。

-xarch=v9

-m64 -xarch=sparc と同等

-xarch=v9a

-m64 -xarch=sparcvis と同等

-xarch=v9b

-m64 -xarch=sparcvis2 と同等

-xF

Oracle Solaris Studio パフォーマンスアナライザで使用される追加情報を生成します。入力ファイルにスタブ (デバッグ指令) が含まれない場合、アセンブラは Oracle Solaris Studio アナライザに必要なデフォルトスタブを生成します。Oracle Sun Studio マニュアルページの dbx(1) も参照してください。

x86 のオプション

-a32

64 ビットモードでの 32 ビットアドレスを許可します。

-H

ハードウェア機能セクションを生成します。(これはデフォルトです。)

-nH

ハードウェア機能セクションの生成を抑制します。

-KPIC

絶対再配置を使用するアドレス参照を検査して、警告を発行します。

-xchip=v

複数のエンコードから選択可能な場合、指定されたチップに適したものを選択します。特に、適切な no-op バイトシーケンスを使用してコード配置パディングを実行し、指定されたチップ用に定義されていない命令が使用される場合は警告します。

アセンブラは、次の認識される -xchip 値の命令セットを受け入れます:

generic	汎用 x86 命令セット。
native	このホストプロセッサ。
core2	Intel Core2 プロセッサ。
nehalem	Intel Nehalem プロセッサ。

opteron	AMD Opteron プロセッサ。
penryn	Intel Penryn プロセッサ。
pentium	Intel Pentium アーキテクチャー。
pentium_pro	Intel Pentium Pro アーキテクチャー。
pentium3	Intel Pentium 3 スタイルプロセッサ。
pentium4	Intel Pentium 4 スタイルプロセッサ。
sandybridge	Intel Sandy Bridge プロセッサ。
westmere	Intel Westmere プロセッサ。
amdfam10	AMD FAM10 プロセッサ。
ivybridge	Intel Ivy Bridge プロセッサ。
haswell	Intel Haswell プロセッサ。

-xmodel=[small | medium | kernel]

-m64 の場合のみ、データアクセス用の R_X86_64_32S 再配置可能タイプを kernel の下に生成します。それ以外の場合、R_X86_64_32 を small の下に生成します。SHN_AMD64_LCOMMON および .lbcomm のサポートが medium の下に追加されました。デフォルトは small です。

-Yd,path

使用する cm4defs のバージョンへのパスを指定します。

環境変数

TMPDIR

通常、as コマンドは、一時ファイルを /tmp ディレクトリ内に作成します。環境変数 TMPDIR を選択したディレクトリに設定することにより、別のディレクトリを指定できます。(TMPDIR が有効なディレクトリではない場合、as は /tmp を使用します)。

ファイル

デフォルトでは、as は一時ファイルを /tmp に作成します。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/
インタフェースの安定性	確実

関連項目

[cpp\(1\)](#), [ld\(1\)](#), [m4\(1\)](#), [nm\(1\)](#), [strip\(1\)](#), [tmpnam\(3C\)](#), [a.out\(4\)](#), [attributes\(5\)](#)

注意事項

SPARC プラットフォームでは、-P フラグが出現すると cpp シンボル __sparc が設定され、-m32 フラグの場合には __sparcv8 が、-m64 フラグの場合には __sparcv9 がそれぞれ設定されます。

x86/x64 では、-P フラグが出現するとシンボル `__i386` が設定され、-m64 フラグの場合には `__amd64` が設定されます。

-m (**m4(1)**) マクロプロセッサを起動) オプションを使用する場合、m4 はアセンブラシンボルのキーワードと、実際の m4 マクロのキーワードとを区別できないため、m4 のキーワードを入力ファイルのシンボル(変数、関数、ラベル)として使用することはできません。

適切なライブラリリンクを保証するため、可能な場合は常に、Oracle Solaris Studio C コンパイラ(cc)などのコンパイルシステムインタフェースプログラム経由でアセンブラにアクセスしてください。Oracle Solaris Studio マニュアルページの `cc(1)` を参照してください。

名前	asa – FORTRAN キャリッジ制御出力の印刷可能形式への変換
形式	asa [-f] [file]...
機能説明	asa ユーティリティーは、その入力ファイルを標準出力に書き込んで、テキストファイル内のキャリッジ制御文字をラインプリンタ制御シーケンスにマッピングします。 各行の最初の文字が入力から削除され、次の処理が実行されます。 次に示すように、削除される文字によって実行される処理が異なります。 SPACE 残りの行が変更なしで出力されます。 0 これは、残りの入力行があとに続く NEWLINE 制御シーケンスで置き換えられます。 1 これは、残りの入力行があとに続く NEWPAGE 制御シーケンスで置き換えられます。 + 前の行の第 1 カラムに戻って印刷を行う制御シーケンスで置き換えられます。残りの入力行は、その行の第 1 カラムから印刷されます。 入力行の第 1 カラムに前述以外の文字がある場合、asa はその文字をスキップして、残りの行を変更せずに印刷します。 ファイル名を指定しないで asa を呼び出すと、標準入力を使用されます。
オプション	次のオプションがサポートされています。 -f 各ファイルを新しいページから開始します。
オペランド	次のオペランドを指定できます。 file 入力に使用されるテキストファイルのパス名。file オペランドを指定しないか、または「-」を指定すると、標準入力を使用されます。
環境変数	asa の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
終了ステータス	次の終了ステータスが返されます。 0 入力ファイルはすべて正常に出力されました。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目 [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	at, batch – 指定した時間にコマンドを実行
形式	<pre> /usr/bin/at [-c -k -s] [-m] [-f file] [-p project] [-q queueName] -t time /usr/bin/at [-c -k -s] [-m] [-f file] [-p project] [-q queueName] timespec... /usr/bin/at -l [-p project] [-q queueName] [at_job_id. ...] /usr/bin/at -r at_job_id. ... /usr/bin/batch [-p project] /usr/xpg4/bin/at [-c -k -s] [-m] [-f file] [-p project] [-q queueName] -t time /usr/xpg4/bin/at [-c -k -s] [-m] [-f file] [-p project] [-q queueName] timespec... /usr/xpg4/bin/at -l [-p project] [-q queueName] [at_job_id. ...] /usr/xpg4/bin/at -r at_job_id. ... /usr/xpg4/bin/batch [-p project] </pre>

機能説明

at ユーティリティーは、一群のコマンドを標準入力から読み込み、それを1つの at-job として統合し、指定された時間に実行します。

この at-job は、シェルを別途呼び出して実行されます。このシェルは、別のプロセスグループで、制御端末なしで動作しているものです。ただし、環境変数、現在の作業用ディレクトリ、ファイル作成マスク ([umask\(1\)](#) を参照)、システムリソースの制限 (sh および ksh88 の場合のみ。 [ulimit\(1\)](#) を参照) については、at ユーティリティーを実行した時点のものが保持されて、at-job 実行時に使用されます。

at-job が投入されると、at_job_id と実行開始予定時間が標準エラー出力に書き込まれます。at_job_id は、at-job の識別子で、英数字とピリオドだけで構成される文字列です。投入時点で、そのジョブが一意に識別できるような名前を at_job_id としてシステムが割り当てます。

ユーザーへの通知やジョブの標準出力および標準エラー出力の処理方法に関しては、-m オプションの項で説明します。

at や batch (後述) を使用できるのは、ファイル /usr/lib/cron/at.allow 中に名前が登録されているユーザーだけです。このファイルが存在していない場合は、ファイル /usr/lib/cron/at.deny を確認して、そのユーザーの at 使用を拒否すべきかどうかを決定します。どちらのファイルも存在しないときは、solaris.jobs.user の承認を受けたユーザーだけがジョブを投入できます。at.deny だけが存在しその内容が空の場合には、どのユーザーもジョブを投入できます。at.allow と at.deny の両ファイルは、どちらも1行に1つのユーザー名という形式です。

ユーザーのアカウントがロックされていると、cron ジョブおよび at ジョブは実行されません。shadow(4) で定義されているように、ロックされていないアカウントだけが、ジョブまたはプロセスを実行します。

batch

batch ユーティリティーは、あとで実行すべきコマンド群を読み込みます。

次のコマンドの形式を見てください。

```
/usr/bin/batch [-p project]
/usr/xpg4/bin/batch [-p project]
```

前述のコマンドの形式は、それぞれ次のコマンドの形式と同等の意味を持ちます。

```
/usr/bin/at -q b [-p project] now
/usr/xpg4/bin/at -q b -m [-p project] now
```

このうち **b** は at の特殊な待ち行列で、バッチジョブ専用に使います。バッチジョブは、バッチ待ち行列に投入されるとただちに実行されます。同時並行での実行が許されているジョブ数の制限によって、投入されたジョブの実行が遅れることがあります。queuedefs(4) を参照してください。

オプション

シェルの種類を指定するオプション **-c**、**-k**、**-s** がすべて省略された場合、デフォルトとして SHELL 環境変数によりシェルが決定されます。

/usr/xpg4/bin/at および /usr/xpg4/bin/batch では、SHELL が設定されていないか NULL の場合、/usr/xpg4/bin/sh が使用されます。

/usr/bin/at および /usr/bin/batch では、SHELL が設定されていないか NULL の場合、/bin/sh が使用されます。

サポートしているオプションは、次のとおりです。

- c** C シェル。csh(1) を使って at-job を実行します。
- k** Korn シェル。/bin/ksh を使って at-job を実行します。
- s** Bourne シェル。sh(1) を使って at-job を実行します。
- f file** at-job の元になるファイルとして標準入力以外のファイルを使用するとき、そのファイルのパスを指定します。
- l** (小文字のエル)at_job_id が指定されなかったときは、コマンドを呼び出したユーザー用にスケジュールしたジョブをすべて報告します。at_job_id が指定されていれば、そのジョブに関する情報だけを報告します。
- m** at-job の実行が終了したら、その旨をメールで当該ユーザーに通知します。at-job が生成した標準出力と標準エラー出力の内容も、他の出力先が指定されない限り、ユーザーにメールで送られます。なおメールは、ジョブが何の出力も生成しなかった場合でも送付されます。

-m を省略すると、ジョブの標準出力と標準エラー出力の内容は、他の出力先が指定されない限り、メールで当該ユーザーに通知されます。そのような出力が生成されなければ、ジョブの終了は通知されません。

- p *project* どのプロジェクトで at ジョブまたは batch ジョブを実行するかを指定します。-l オプションと共に使用すると、指定した特定のプロジェクトだけを検索します。*project* の値全体が数値である場合は、まずプロジェクト名として解釈され、次にプロジェクト ID として解釈されます。デフォルトでは、ユーザーの現在のプロジェクトが使用されます。
- q *queuenam*e *queuenam*e で示す待ち行列にジョブをスケジュールします。-l オプションも一緒に指定すると、その待ち行列だけが検索の対象となります。*queuenam*e として指定できるのは、a から z までの英小文字です。デフォルトでは、at-job は待ち行列 a にスケジュールされています。また待ち行列 b はバッチジョブ用に予約されています。待ち行列 c は cron ジョブ用に予約されているので、-q オプションの引数として使うことはできません。
- r *at_job_id* 以前の at ユーティリティーでスケジュールされたジョブのうち、*at_job_id* で示す識別子を持ったジョブを削除します。
- t *time* *time* オプション引数によって指定された時間に実行するようジョブを投入します。オプション引数は、touch(1) ユーティリティーによって指定される形式である必要があります。

オペランド

次のオペランドがサポートされています。

- at_job_id* 以前の at ユーティリティーによりジョブがスケジュールされたときに報告された名前。
- timespec* ジョブを投入し実行する日時を指定します。すべての *timespec* の値は、スペース文字で区切られて連結されていると見なされます。日付と時間の値は、そのユーザーのタイムゾーン (tz 変数が決定) で表されていると見なされます。ただし後述する *time* オペランドでタイムゾーン名を指定した場合を除きます。

C ロケールの場合、日時を指定する文字列は次に述べる 3 つの部分で構成されます。C ロケールの LC_TIME カテゴリからのすべての値は大文字と小文字の区別なく認識されます。

- time* *time* は、1、2、または 4 桁の数で指定します。1 桁または 2 桁の場合は「何時」を示す値として、4 桁の場合は「何時何分」を示す値として解釈されます。2 つの数をコロンで区切り、*hour:minute* の形式で「何時何分」を指定することもできます。時間指定の直後に AM または PM

(LC_TIME ロケールカテゴリの `am_pm` キーワードから得られる値) 表示を付加することもできます。そうしない場合は、24 時間制で記述された時間と見なされます。GMT、UCT、または ZULU (大文字小文字の区別なし) のタイムゾーン名を、ユニバーサル時間を調整する時間の指定に続けることもできます。その他のタイムゾーンは `tz` 環境変数を使用して指定できます。また、`time` フィールドは、C ロケールの次のトークンのいずれかにすることも可能です。

`midnight` 12:00 am (真夜中) を表します (00:00)。

`noon` 12:00 pm (正午) を表します。

`now` 現在の日時を表します。つまり `at now` という指定は、ただちに `at-job` を投入するという意味ですが、すぐに実行されるかどうかはその時点でのジョブのスケジューリング状況に依存します。

date 日付を示す *date* の指定は任意で、「月」の名前 (LC_TIME ロケールカテゴリの `mon` または `abmon` キーワードから得られる値) の後に「日」を表す数値を記述する (さらにその後にコンマと「年」を表す数値があってもよい) 方法と、曜日 (LC_TIME ロケールカテゴリの `day` または `abday` キーワードから得られる値) を記述する方法があります。C ロケールでは 2 つの特殊な日付が認識されます。

`today` 現在の日付が示す日、つまり当日を表します。

`tomorrow` 現在の日付が示す日の次の日、つまり翌日を表します。

date を省略すると、指定された時間が現時間より後であれば当日、現時間より前であれば翌日とみなされます。指定した月の値が当月よりも前で「年」を省略した場合は、翌年とみなされます。

increment 省略可能な *increment* は、正の符号 (+) が前に付加された数値で、あとに次のいずれかが付加されます。minutes、hours、days、weeks、months、または years。複数形を示す *s* は省略できます。また +1 と同等の意味を持つキーワード `next` も使用できます。たとえば次のコマンドは同等です。

`at 2pm + 1 week`

`at 2pm next week`

使用法

ここに示す at コマンド行の形式は、C ロケールに対してだけ保証されています。その他のロケールで

は、midnight、noon、now、mon、abmon、day、abday、today、tomorrow、minutes、hours、d の各指定はサポートされていません。

コマンドの実行は、別のプロセスグループで制御端末なしで動作しているシェルを別途呼び出して行うので、コマンドを呼び出した環境でのオープンファイル記述子やトラップ、優先順位などは失われてしまいます。

使用例

at

例1 端末での指定例

端末でのコマンドシーケンスの例を示します。

```
$ at -m 0730 tomorrow
sort < file >outfile
<EOT>
```

例2 出力先のリダイレクト

次のシーケンスは、出力先を標準エラー出力からパイプに変更するもので、コマンドプロシージャの中で使用すると便利です。なお、出力先変更指定の記述順序は重要なので注意してください。

```
$ at now + 1 hour <<!
diff file1 file2 2>&1 >outfile | mailx mygroup
```

例3 ジョブ自身による再スケジュール

ジョブ自身に再スケジュールさせるため、at-job の中から at を呼び出すことができます。たとえば、次の my.daily という名前の日常処理スクリプトは毎日実行されます (ただし、このような作業では crontab を使う方法のほうが一般的です)。

```
# my.daily runs every day
at now tomorrow < my.daily
daily-processing
```

例4 時間、オペランド指定

C ロケールの *timespec* の 3 つの部分の間隔は、明示的に記述してあれば非常に柔軟です。時間やオペランド指定の例を次に示します。

```
at 0815am Jan 24
at 8 :15amjan24
at now "+ 1day"
at 5 pm FRIday
at '17
    utc+
    30minutes'
```

batch	例5 端末での指定例 端末でのコマンドシーケンスの例を示します。 <pre>\$ batch sort <file >outfile <EOT></pre> 例6 出力先のリダイレクト 次のシーケンスは、出力先を標準エラー出力からパイプに変更するもので、コマンドプロシージャの中で使用すると便利です。なお、出力先変更指定の記述順序は重要なので注意してください。 <pre>\$ batch <<! diff file1 file2 2>&l >outfile mailx mygroup !</pre>
環境変数	at と batch の実行に影響を与える次の環境変数の詳細については、environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、NLSPATH、および LC_TIME。 DATEMSK 環境変数 DATEMSK が設定されていれば、at はその値を、書式文字列を含むテンプレートファイルの完全パス名として使用します。この文字列は書式記述子とテキスト文字から構成され、環境変数 LANG または LC_TIME を適切に設定することによって、異なる言語で許容される日付表示形式をより豊富に提供するために使用します。利用可能な書式記述子の一覧については、getdate(3C) のマニュアルページを参照してください。「オペランド」のセクションで説明している <i>time</i> および <i>date</i> 引数、特殊名の <i>noon</i>、<i>midnight</i>、<i>now</i>、<i>next</i>、<i>today</i>、<i>tomorrow</i>、および <i>increment</i> 引数の書式は、DATEMSK が設定されている場合には認識されません。 SHELL at-job の呼び出しに使用するコマンドインタプリタの名前を指定します。この変数が設定されていないか値が NULL の場合には、sh が使用されます。sh 以外の値に設定されていれば、そのシェルを使用します。このとき、どのシェルを使うかを表す警告メッセージが出力されます。 TZ タイムゾーンを指定します。ジョブは、<i>timespec</i> または <i>-t time</i> が示す時間に実行するために投入されますが、この時間は TZ 変数が示すタイムゾーンに対応した値です。<i>timespec</i> にタイムゾーン指定が含まれていれば、TZ が示すゾーンに代わってそちらが使用されます。<i>timespec</i> にタイムゾーン指定が含まれておらず、TZ も未設定か NULL の場合、デフォルトのタイムゾーンが使用されます。
終了ステータス	次の終了ステータスが返されます。

0 at ユーティリティーによるジョブの投入、削除、または一覧表示が正常終了しました。

>0 エラーが発生したため、ジョブはスケジュールされません。

ファイル

/usr/lib/cron/at.allow at と batch の両ユーティリティーへのアクセス権を持つユーザーの一覧。1行に1ユーザー名の形式

/usr/lib/cron/at.deny at と batch の両ユーティリティーへのアクセスを拒否するユーザーの一覧。1行に1ユーザー名の形式

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/at

属性タイプ	属性値
使用条件	system/core-os
CSI	未対応
インタフェースの安定性	確実
標準	standards(5) を参照してください。

/usr/xpg4/bin/at

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	未対応
インタフェースの安定性	標準

/usr/bin/batch

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	標準

/usr/xpg4/bin/batch

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	標準

関連項目

[auths\(1\)](#), [crontab\(1\)](#), [csh\(1\)](#), [date\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [touch\(1\)](#), [ulimit\(1\)](#), [umask\(1\)](#), [cron\(1M\)](#), [getdate\(3C\)](#), [auth_attr\(4\)](#), [shadow\(4\)](#), [queuedefs\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

待ち行列を使用しているかどうかにかかわらず、`cron(1M)`の実行は常に100ジョブに限られています。

`cron`ではジョブの実行に遅れの出る場合があります。これらの遅れによって `cron` ジョブの処理がハングしたように見えることがあります。すべてのジョブは最後には実行されますが、極端に遅れが生じた場合には、`cron`を終了してから再起動することが唯一の回避策です。

名前	atq – 指定時間 to 実行するために待ち行列に入っているジョブの表示
形式	atq [-c] [-n] [username]...
機能説明	atq ユーティリティーは、現在のユーザーについて、待ち行列に入っている at ジョブを表示します。at(1) は、ユーザーがコマンドを後日実行できるようにするユーティリティーです。solaris.jobs.admin 承認されたユーザーが atq を実行すると、待ち行列内のすべてのジョブが表示されます。 オプションを指定しない場合は、ジョブは実行される時間順に表示されます。 承認されたユーザーが username を指定しないで atq を実行した場合は、待ち行列全体が表示されます。username を指定すると、指定したユーザーに属するジョブだけが表示されます。
オプション	サポートしているオプションは、次のとおりです。 -c 待ち行列に入っているジョブが作成された順序 (つまり、at コマンドが実行された時間順) で表示します。 -n 現在待ち行列内にあるジョブの合計数だけを表示します。
ファイル	/var/spool/cron/atjobs at ジョブ用のスプール領域
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [at\(1\)](#), [atrm\(1\)](#), [auths\(1\)](#), [cron\(1M\)](#), [auth_attr\(4\)](#), [attributes\(5\)](#)

名前 atrm - at または batch によってスプールされたジョブの削除

形式 atrm [-afi] [[job #] [user]...]

機能説明 atrm ユーティリティーは、[at\(1\)](#) コマンドによって作成されたがまだ実行されていない遅延実行ジョブを削除します。これらのジョブと関連ジョブ番号のリストは、[atq\(1\)](#) を使用して表示できます。

ユーザーが指定したジョブを所有している場合、atrm は、そのユーザーが指定した各ジョブ番号や、そのユーザーが指定したユーザーに属するすべてのジョブを削除します。

solaris.jobs.admin 特権を持つ場合のみ、他のユーザーに属するジョブを削除できます。

- オプション サポートしているオプションは、次のとおりです。
- a すべて。現在のユーザーによって作成されたすべての未実行ジョブを削除します。特権ユーザーによって実行された場合は、待ち行列全体がフラッシュされます。
 - f 強制。指定されたジョブの削除に関するすべての情報が抑制されます。
 - i 対話型。atrm は、ジョブを削除するかどうかを尋ねます。応答が肯定の場合、ジョブは削除されます。

ファイル /var/spool/cron/atjobs at ジョブ用のスプール領域

環境変数 atrm の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの yesexpr キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、yesexpr に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、yesexpr に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#) を参照してください。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [at\(1\)](#), [atq\(1\)](#), [auths\(1\)](#), [cron\(1M\)](#), [auth_attr\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [locale\(5\)](#)

名前	audioconvert - オーディオファイル形式の変換
形式	audioconvert [-pF] [-f <i>outfmt</i>] [-o <i>outfile</i>] [[-i <i>infmt</i>] [<i>file</i>]...] ...
機能説明	audioconvert は、サポートされている一連のオーディオコードとファイル形式の間でオーディオデータを変換します。このユーティリティは、オーディオデータの圧縮と圧縮解除、raw オーディオデータファイルへのオーディオファイルヘッダーの追加、-law やリニア PCM などの標準データコード間の変換を行うために使用できます。 ファイル名が指定されていない場合、audioconvert は、標準入力ストリームからデータを読み取って、標準出力にオーディオファイルを書き込みます。それ以外の場合、入力ファイルは順番に処理、連結されて、出力ファイルに書き込まれます。 入力ファイルには、オーディオデータ形式を識別するオーディオファイルヘッダーが含まれています。オーディオデータに認識可能なヘッダーが含まれていない場合は、rate、encoding、および channels の各キーワードを使用して入力データ形式を識別し、-i オプションで形式を指定する必要があります。 出力ファイル形式は、-f 指定の形式オプションを用いて、最初の入力ファイルの形式を更新することによって決定されます。-p を指定しないと、後続のすべての入力ファイルが、この結果決定された形式に変換されて連結されます。format=raw が出力形式オプションに指定されていない限り、出力ファイルには、オーディオファイルヘッダーが含まれます。 入力ファイルは、-p オプションを使用して適宜変換できます。-p が有効な場合、各入力ファイルの形式は、出力形式を決定する -f オプションに従って変更されます。既存のファイルは、変換されたデータによって上書きされます。 file(1) コマンドは、Sun オーディオファイルのオーディオデータ形式をデコードして出力します。
オプション	サポートしているオプションは、次のとおりです。 -p <i>In Place</i>: 入力ファイルは、-f オプションで指定された形式に個々に変換されて、再度書き込まれます。ターゲットファイルがシンボリックリンクの場合、元のファイルが再度書き込まれます。-o オプションは、-p と同時に指定することはできません。 -F <i>Force</i>: このオプションは、audioconvert が -i オプションによって形式が指定されている入力ファイルのファイルヘッダーをすべて無視するようにします。-F を指定しないと、audioconvert は、有効なオーディオファイルヘッダーを含む入力ファイルに対する -i オプションを無視します。 -f <i>outfmt</i> <i>Output Format</i>: このオプションは、出力ファイルのファイル形式とデータコードを指定するために使用されます。指定されていないフィールドのデフォルト値は、入力ファイル形式から決定されます。有効なキーワードと値については、以下の項で説明します。

- o outfile** *Output File:* すべての入力ファイルが連結され、出力形式に変換されて、指定の出力ファイルに書き込まれます。-o と -p のいずれも指定しないと、連結された出力は標準出力に書き込まれます。-p オプションは、-o と同時に指定することはできません。
- i infmt** *Input Format:* このオプションは、raw 入力ファイルのデータコードを指定するために使用されます。通常、入力データ形式は、オーディオファイルヘッダーから決定されますこのオプションは、有効なオーディオファイルヘッダーが付いていないオーディオデータを変換するときに必要です。オーディオファイルヘッダーを含む入力ファイルに -i を指定すると、-F が指定されている場合を除き、入力形式文字列は無視されます。形式指定構文は、-f 出力ファイル形式と同じです。
- file** *File Specification:* 指定のオーディオファイルは連結され、出力形式に変換されて、書き込まれます。ファイル名が指定されていないか、または特殊ファイル名_ が指定されている場合、オーディオデータは標準入力から読み取られます。
- ?** *Help:* コマンド行の使用法に関するメッセージを出力します。

形式指定

入力および出力形式指定の構文は、次のとおりです。

keyword=value[,keyword=value ...]

間にスペースは入りません。一意の値は *keyword=* を前に付けなくても指定できます。

- rate** オーディオサンプリング率は、1 秒あたりのサンプル数で指定されます。数値の後に続く文字 *k* は、その数値に 1000 を掛けた値を意味します。(たとえば、44.1k=44100)。一般的に使用されるサンプル率の標準は、8k、16k、32k、44.1k、および 48k です。
- channels** インタリーブされたチャネルの数は整数で指定されます。1 つまたは 2 つのチャネルデータを指定するために、それぞれ *mono* または *stereo* という語を使用することもできます。
- encoding** このオプションは、デジタルオーディオデータ表示を指定します。コードは、精度を暗黙に (*ulaw* は暗黙に 8 ビット精度を示す)、または名前の一部として明示的に (たとえば、*linear16*) 指定します。有効なコード値は次のとおりです。
- ulaw** CCITT G.711 -law コード。これは、電話レベルの音声に主に使用される 8 ビット形式です。

alaw	CCITT G.711 A-law コード。これは、ヨーロッパで電話レベルの音声に主に使用される 8 ビット形式です。
linear8, linear16, linear32	リニアパルスコード変調 (PCM) コード。名前は、精度のビット数を示します。linear16 は通常、高品質オーディオデータに使用されます。
pcm	linear16 と同じです。
g721	CCITT G.721 圧縮形式。このコードは、4 ビット精度の適応デルタパルスコード変調 (ADPCM) を使用します。これは、主に圧縮 -law 音声データに使用されます (2:1 の圧縮率を達成)。
g723	CCITT G.723 圧縮形式。このコードは、3 ビット精度の適応デルタパルスコード変調 (ADPCM) を使用します。これは、主に圧縮 -law 音声データに使用されます (8:3 の圧縮率を達成)。オーディオ品質は、G.721 と同等ですが、非スピーチデータに使用した場合、品質が低下することがあります。

次のコード値も、サンプル率、チャネル、およびコードを設定するための短縮形として使用できます。

voice encoding=ulaw,rate=8k,channels=mono と同等

cd encoding=linear16,rate=44.1k,channels=stereo と同等

dat encoding=linear16,rate=48k,channels=stereo と同等

format このオプションは、オーディオファイル形式を指定します。有効な形式は、次のとおりです。

sun Sun 互換ファイル形式 (デフォルト)。

raw この形式は、raw オーディオデータ (オーディオヘッダーなし) を読み取るか書き込む場合に使用します。あるいは、外部オーディオファイル形式をインポートするために offset と組み合わせて使用します。

offset (-i only) オーディオデータの開始を見つけるためのバイトオフセットを指定します。このオプションは、認識不能なファイルヘッダーを含むオーディオデータをインポートするために使用できます

使用法

ファイルのサイズが 2G バイト (2^{31} バイト) 以上ある場合の audioconvert の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 音声データを記録および圧縮してから格納する
音声データを記録し、圧縮してからファイルに保存します。

```
example% audiorecord | audioconvert -f g721 > mydata.au
```

例2 2つの音声ファイルを連結する
データ形式に関係なく、2つの Sun 形式のオーディオファイルを連結して、8 ビットの -law、16 KHz のモノラルファイルを出力します。

```
example% audioconvert -f ulaw,rate=16k,mono -o outfile.au infile1 infile2
```

例3 ディレクトリを Sun 形式に変換する
raw 音声データファイルを含むディレクトリを、適宜 Sun 形式に変換します (各ファイルにファイルヘッダーを追加します)。

```
example% audioconvert -p -i voice -f sun *.au
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
アーキテクチャー	SPARC, x86
使用条件	audio/audio-utilities
インタフェースの安定性	確実

関連項目

[audioplay\(1\)](#), [audiorecord\(1\)](#), [file\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

注意事項

マルチチャネルデータをモノラルに変換するために使用されるアルゴリズムは、単にチャンネルを足し合わせるにより実装されます。入力データが完全に同相している場合 (モノラルファイルがステレオに変換されて再びモノラルに変換される場合のように)、結果のデータには若干の変形が含まれることがあります。

名前	audiocctl – オーディオミキサー制御のコマンド行アプリケーション
形式	<pre>audiocctl list-devices audiocctl show-device [-v] [-d device] audiocctl show-control [-v] [-d device] [control ...] audiocctl set-control [-v] [-d device] control value audiocctl save-controls [-d device] [-f] file audiocctl load-controls [-d device] file</pre>
機能説明	audiocctl コマンドを使用して、オーディオミキサーのさまざまな機能を制御したり、オーディオミキサーやオーディオデバイスの情報を取得したりします。audiocctl コマンドは、次のデータ型に対して処理を実行します。 <i>device</i> オーディオデバイス (audiohd#0 など)。これを受け入れるサブコマンドでは、オプション <i>-d</i> への引数としてこれが受け入れられます。指定しない場合、デフォルトのオーディオデバイスが使用されます。/dev/sound/0、/dev/dsp1、/dev/audio など、オーディオデバイスに関連付けられているどのデバイスノードも動作します。 <i>control</i> ミキサー制御の名前 (volume など)。 <i>value</i> 制御の値。具体的な形式は、制御のタイプによって異なります。モノラルの値には、通常、0 以上 100 以下の整数が 1 つ使用されます。ステレオの値には、左右のチャンネルを表す、モノラルと同じ一対の値が使用されます。ブール値は、オンまたはオフを示します。列挙する場合、1 つ以上の名前で構成される単一の値を指定します。 <i>file</i> 制御設定が記述された ASCII テキストファイル。
オプション	各サブコマンドには、独自のオプションセットを設定します。ただし、一部のサブコマンドは、詳細情報の出力要求を示す特殊フラグ <i>-v</i> をサポートします。
サブコマンド	サポートされているサブコマンドは次のとおりです。 <pre>audiocctl list-devices</pre> システムのオーディオデバイスをすべて一覧表示します。 <pre>audiocctl show-device [-v] [-d devices]</pre> デバイスに関する一般的な情報を表示します。 <pre>audiocctl show-control [-v] [-d device] [control ..]</pre> デバイスの制御設定値を表示します。指定された制御が表示されます。制御名を指定しない場合、すべての制御値が表示されます。 <pre>audiocctl set-control [-v] [-d device] control value</pre> 制御値を、指定した値に変更します。

`audiocctl save-controls [-f] [-d device] file`
全ミキサー制御値の現在の状態を、指定したファイルに保存します。-f を指定しないかぎり、ファイルがすでに存在する場合、このコマンドは安全に終了します。

`audiocctl load-controls [-d device] file`
すべてのミキサー制御について、指定したファイルに保存されている状態を復元します。

環境変数 `AUDIODEV` -d および -a オプションを指定しないと、`AUDIODEV` 環境変数が参照されます。設定する場合、`AUDIODEV` には、ユーザーのデフォルトオーディオデバイスのフルパス名を含めます。

ファイル `/dev/audiocctl/dev/sound/{0...n}ctl`

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
アーキテクチャー	SPARC, x86
使用条件	system/io/audio
インタフェースの安定性	下記を参照。

`audiocctl` コマンドおよびそのサブコマンドは「確実」です。人間が読める形式の出力は「非インタフェース」です。デバイス名、制御名、および値は「不確実」です。`save-controls` および `load-controls` サブコマンドによって使用される状態ファイルの形式は、「確実」で「非公開」です。

関連項目 [audioconvert\(1\)](#), [audioplay\(1\)](#), [audiorecord\(1\)](#), [open\(2\)](#), [attributes\(5\)](#)

名前	audioplay - オーディオファイルの再生
形式	audioplay [-iV] [-v vol] [-d dev] [file]...
機能説明	audioplay ユーティリティは、指定されたオーディオファイル(ファイル名が指定されていない場合は標準入力)をオーディオデバイスにコピーします。入力ファイルが指定されておらず、かつ標準入力/ttyの場合、プログラムはエラーメッセージを出力して終了します。 入力ファイルには、有効なオーディオファイルヘッダーが含まれていなければなりません。このヘッダーのコード情報はオーディオデバイスの機能に対応しており、データ形式に互換性がない場合は、エラーメッセージが出力されて、その入力ファイルはスキップされます。圧縮されたADPCM (G.721) モノラルオーディオデータは、自動的に圧縮解除されてから再生されます。 サンプリング周波数のわずかな違い (1% 未満) は、通常無視されます。これにより、たとえば、8000 Hz の周波数だけしかサポートしていないオーディオデバイスでも、8012 Hz でサンプリングされたデータを再生することができます。-v オプションが指定されている場合、周波数の違いには、警告メッセージによってフラグが付けられます。
オプション	サポートしているオプションは、次のとおりです。 -d dev <i>Device:</i> dev 引数は、出力が送信される代替オーディオデバイスを指定します。-d オプションを指定しないと、AUDIODEV 環境変数が参照されます (下記参照)。あるいは、/dev/audio がデフォルトのオーディオデバイスとして使用されます。 -i <i>Immediate:</i> オーディオデバイスを利用できない場合 (つまり、別のプロセスが現在書き込み権を持つ場合)、audioplay は通常、デバイスへのアクセス権を取得するまで待機します。-i オプションが指定されている場合、audioplay はエラーメッセージを出力して、デバイスが使用中であれば即座に終了します。 -v vol <i>Volume:</i> 出力ボリュームが指定の値に設定されてから再生が開始され、audioplay が終了すると元のレベルにリセットされます。vol 引数は、0 - 100 (0 と 100 を含む) の間の整数値です。この引数を指定しないと、出力ボリュームは、いずれかのプロセスによって最後に設定されたレベルのままになります。 -V <i>Verbose:</i> オーディオデバイスへのアクセスを待機している場合、またはサンプル率のずれが検出された場合、標準エラー出力にメッセージを書き込みます。 -? <i>Help:</i> コマンド行の使用法に関するメッセージを出力します。

オペランド	<i>file</i> <i>File Specification</i> : コマンド行に指定されたオーディオファイルは、順番に再生されます。ファイル名が指定されていない場合は、標準入力ストリーム (tty ではない場合) が再生されます (このストリームには、オーディオファイルヘッダーも含まれていなければなりません)。特殊なファイル名 - を使用すると、ファイルの代わりに標準入力ストリームを読み取ることができます。相対パス名を指定すると、AUDIOPATH 環境変数が参照されます (下記参照)。
使用法	ファイルのサイズが 2G バイト (2^{31} バイト) 以上ある場合の <code>audioplay</code> の動作については、 largefile(5) を参照してください。
環境変数	AUDIODEV <code>-d</code> 引数が指定されていない場合の、書き込み先オーディオデバイスの完全パス名。AUDIODEV 変数が設定されていない場合は、<code>/dev/audio</code> が使用されます。 AUDIOPATH 相対パス名で名前が指定されているオーディオファイルの検索先ディレクトリをコロンで区切ったリスト。現在のディレクトリ (.) は、検索パスに明示的に指定できます。AUDIOPATH 変数が設定されていない場合は、現在のディレクトリだけが検索されます。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
アーキテクチャー	SPARC, x86
使用条件	audio/audio-utilities
インタフェースの安定性	確実

関連項目 [audioconvert\(1\)](#), [audiocctl\(1\)](#), [audiorecord\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [audio\(7I\)](#)

使用上の留意点 `audioplay` が現在サポートしているオーディオ形式変換は限定されています。オーディオファイルの形式がオーディオデバイスによってサポートされていない場合は、まずこのファイルを変換する必要があります。たとえば、進行状況に合わせて音声形式に変換するには、次のコマンドを使用します。

```
example% audioconvert -f voice myfile | audioplay
```

形式変換がオーディオ出力に追いつけない場合があります。その場合は、一時ファイルに変換してから、データを再生する必要があります。

名前	audiorecord - オーディオファイルの記録
形式	<pre>audiorecord [-af] [-v vol] [-c channels] [-s rate] [-e encoding] [-t time] [-i info] [-d dev] [-T au aif[f] wav] [file[.au .aif[f]] .wav]</pre>
機能説明	audiorecord ユーティリティは、オーディオデバイスのオーディオデータを指定されたオーディオファイル(ファイル名が存在しない場合は、標準出力)にコピーします。出力ファイルが指定されておらず、かつ標準出力が tty の場合、プログラムはエラーメッセージを出力して終了します。 デフォルトでは、モノラルのオーディオデータは 8kHz で記録され、-law 形式でエンコードされます。オーディオデバイスが拡張構成をサポートする場合、-c、-s、および -e オプションを使用してデータ形式を指定できます。出力ファイルには、オーディオファイルのヘッダーが接頭辞として付くので、その出力ファイルにエンコードされているデータ形式が分かります。 記録は即座に開始され、SIGINT シグナル(たとえば、Control-C)を受信するまで続けられます。-t オプションを指定した場合、audiorecord は指定された量のデータを記録した後に停止します。 オーディオデバイスが利用できない場合、つまり、ほかのプロセスが読み取りアクセスを行なっている場合、audiorecord はエラーメッセージを出力して、即座に終了します。
オプション	サポートしているオプションは、次のとおりです。 -h? <i>Help:</i> コマンド行の使用法に関するメッセージを出力します。 -a <i>Append:</i> 指定されたオーディオファイルの終わりに、データを追加します。この場合、指定されたオーディオファイルのオーディオデータ形式をオーディオデバイスがサポートする必要があります。 -c channels <i>Channels:</i> オーディオチャネルの数を指定します。この値は、整数(1 または 2) または文字列(mono または stereo)のどちらでも指定できます。デフォルトの値は mono です。 -d dev <i>Device:</i> 代替のオーディオデバイスを指定します。dev 引数は、入力元となるオーディオデバイスです。-d オプションを指定しないと、AUDIODEV 環境変数が参照されます(下記参照)。あるいは、/dev/audio がデフォルトのオーディオデバイスとして使用されます。 -e encoding <i>Encoding:</i> オーディオデータのエンコード方法を指定します。この値は、ulaw、alaw、または linear のうちの 1 つです。デフォルトのエンコード方法は ulaw です。

-f	<i>Force</i> : -a オプションを指定した場合、オーディオデバイスのサンプル率は、オリジナルのファイルが記録されたときのサンプル率に一致する必要があります。-a オプションと一緒に f オプションも指定した場合、サンプル率の違いは無視されます。そして、警告メッセージが標準エラーに出力されます。
-i info	<i>Information</i> : 出力ファイルのヘッダーの「information (情報)」フィールドに、info 引数に指定された文字列を設定します。このオプションは、-a オプションと一緒に指定してはいけません。
-s rate	<i>Sample Rate</i> : サンプル率を 1 秒あたりのサンプル数で指定します。数値の後に続く文字 k は、その数値に 1000 を掛けた値を意味します。(たとえば、44.1k = 44100)。デフォルトのサンプル率は 8kHz です。
-t time	<i>Time</i> : 記録できる最長時間を指定します。time 引数は、浮動小数点数として、あるいは、形式「hh:mm:ss.dd」として指定できます。前者の場合、小数部分は秒数を意味します。後者の場合、時間 (hh) と分 (mm) は省略できます。
-T au aif[f] wav	作成するオーディオファイルの種類を指定します。-a オプションを指定した場合、ファイルの種類は、オリジナルのファイルの種類に一致する必要があります。ファイルの接尾辞にかかわらず、ファイルの種類はこのオプションに指定したファイルの種類に設定されます。このオプションを指定しない場合、ファイルの種類はファイルの接尾辞によって決定されます。
-v vol	<i>Volume</i> : 入力ボリューム (記録ゲイン) を vol に設定してから、記録を開始します。そして、audiorecord の終了時に、以前のレベルに戻します。vol 引数は、0 - 100 (0 と 100 を含む) の間の整数値です。この引数を指定しないと、入力ボリュームは、いずれかのプロセスによって最後に設定されたレベルのままになります。
オペランド	file[.au .aif[f]] .wav <i>File Specification</i>: 指定されたオーディオファイルを作成または追加します。ファイル名が存在せず、かつ標準出力が tty でない場合、あるいは、特別なファイル名「-」を指定した場合、出力は標準出力に送られます。 -T オプションを指定しない場合、ファイルの種類はファイルの接尾辞によって決定されます。接尾辞が認識されない場合、デフォルトは .au です。-T オプションを指定した場合、ファイルの接尾辞にかかわらず、ファイルの種類は T オプションに指定したファイルの種類に設定されます。

- 使用法** ファイルのサイズが2G バイト (2^{31} バイト) 以上ある場合の `audiorecord` の動作については、[largefile\(5\)](#) を参照してください。
- 環境変数** `AUDIODEV` 入力元のオーディオデバイスのフルパス名 (`-d` オプションを指定しない場合)。`AUDIODEV` 変数が設定されていない場合は、`/dev/audio` が使用されます。
- 属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
アーキテクチャー	SPARC, x86
使用条件	audio/audio-utilities
インタフェースの安定性	確実

- 関連項目** [audioconvert\(1\)](#), [audiocctl\(1\)](#), [audioplay\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [audio\(7I\)](#)

名前	audiotest - オーディオデバイスのテスト
形式	audiotest [-24571] [dev] ...
機能説明	audiotest ユーティリティーは、指定したオーディオデバイス (何も指定しない場合はシステム上で見つかったすべてのオーディオデバイス) のテストを実行します。このテストでは、各チャンネルでのオーディオサンプルの再生、および再生レートの測定によるクロックずれの検出が行われます。
オプション	サポートしているオプションは、次のとおりです。 -1 ループモード。テストは無限ループで実行されます。 -2 ステレオ (2 チャンネル) モード。これはデフォルトモードです。2 つのチャンネルが存在すると仮定して再生が行われます。 -4 4 チャンネルモード (4 チャンネルサラウンド)。4 つのサラウンドチャンネルが存在すると仮定して、テストが行われます。 -5 サラウンドサウンドモード (5.1)。このテストでは、左、右、左サラウンド、右サラウンド、および中央のチャンネルをチェックします。低周波エフェクトチャンネルのテストは行われません。 -7 サラウンドサウンドモード (7.1)。テストは左、右、左サラウンド、右サラウンド、左後方サラウンド、右後方サラウンド、および中央のチャンネルをチェックします。低周波エフェクトチャンネルのテストは行われません。
オペランド	dev テストするデバイスのパス (たとえば /dev/dsp0)。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
アーキテクチャー	SPARC, x86
使用条件	audio/audio-utilities
インタフェースの安定性	確実

関連項目 [audioconvert\(1\)](#), [audiocntl\(1\)](#), [audiorecord\(1\)](#), [attributes\(5\)](#), [audio\(71\)](#)

使用上の留意点 [audiotest](#) では、物理デバイスでサポートされる実際のオーディオチャンネルの数を検出することはできません。

[audiotest](#) では、低周波エフェクト (LFE) チャンネルのテストは行われません。

オーディオの取り込みや音量の制御など、高度なデバイス機能のテストは行われません。

名前 auths – 承認の管理および一覧表示

形式

```
auths [user]...

auths list [-S repository] [-v] [-u user]

auths info [-S repository] [-v] [authorization]

auths check [-u user] authorization

auths add [-S repository] -t description
        [-h help_file_path] authorization

auths modify [-S repository] [-t description]
        [-h help_file_path] authorization

auths remove [-S repository] authorization
```

機能説明

auths コマンドは、自分や任意指定のユーザーまたは役割に付与された承認を標準出力に書き込みます。承認は、制限付き機能をユーザーが実行できるかどうかを決定するために、特定の特権付きプログラムによって検査される特権です。

このコマンドではまた、ローカルファイルのネームサービスまたはLDAP ネームサービスの [auth_attr\(4\)](#) データベース内の承認とそのプロパティの作成および変更も行います。また、auths コマンドは、自分や必要に応じて指定されたユーザーまたは役割に付与された承認を標準出力に出力します。

管理者が追加、変更、または削除サブコマンドを使用して [auth_attr\(4\)](#) データベース内の承認を管理できるようにするには、その管理者に Rights Management プロファイルが付与されている必要があります。

各ユーザーはゼロまたはそれ以上の承認を持つことができます。承認は、承認を作成した組織と制御対象の機能とを識別する完全指定名で表現されます。Java の規約に準じて、承認の階層コンポーネントはドット (.) で区切られ、作成元組織のインターネットドメイン名を逆順にした形式で始まり、承認クラス内の特定の機能で終わります。承認をドット (.) で終わらせることはできません。

アスタリスク (*) はクラス内のすべての承認を示します。

ユーザーの承認は、[user_attr\(4\)](#) および `/etc/security/policy.conf` ファイル ([policy.conf\(4\)](#) を参照) 内で検索されます。承認は、[user_attr\(4\)](#) 内で直接、または [prof_attr\(4\)](#) を通して間接的に指定できます。また、`/etc/security/policy.conf` ファイル内で、承認をデフォルトの承認として直接、またはデフォルトプロファイルとして間接的に、システムの各ユーザーに割り当てることもできます。

サブコマンド

```
add [-S repository] -t description [-h help_file_path] authorization
    指定された承認 (authorization) を指定されたネームサービスリポジトリ
    (repository) 内に作成します。
```

リポジトリオプションが指定されていない場合、承認はそのファイルのネームサービス内に作成されます。

check [-u user] authorization

指定された承認 (*authorization*) が指定されたユーザー名 (*user*) または現在のユーザーに付与されているかどうかを確認します。

ユーザーに正しい承認が付与されている場合、auths は終了コード 0 で終了します。それ以外の場合は、1 より大きい終了コードを返します。

info [-S repository] [-v] [authorization]

指定された承認 (*authorization*) が指定されたネームサービスリポジトリ (*repository*) 内に存在するかどうかを確認するか、または [nsswitch.conf\(4\)](#) に基づいて検索します。指定された承認が存在する場合は、その承認が一覧表示され、auths はリターンコード 0 で終了します。

承認が指定されていない場合、auths は、指定されたネームサービスリポジトリ内に存在するすべての承認を出力するか、または [nsswitch.conf\(4\)](#) に基づいて出力します。

list [-S repository] [-v] [-u user]

指定されたユーザー (*user*) または (ユーザー名が指定されていない場合は) 現在のユーザーに割り当てられているすべての承認をネームサービスリポジトリ (*repository*) に基づいて一覧表示します。

リポジトリが指定されていない場合は、[nsswitch.conf\(4\)](#) に基づいて情報が検索されます。

modify [-S repository] [-t description] [-h help_file_path]

指定されたネームサービスリポジトリ内の既存の承認を変更します。リポジトリが指定されていない場合は、[nsswitch.conf\(4\)](#) に基づいて検索されたネームサービス内の承認が変更されます。

remove [-S repository] authorization

指定されたネームサービスリポジトリ (*repository*) 内の既存の承認 (*authorization*) を削除します。

リポジトリが指定されていない場合は、[nsswitch.conf\(4\)](#) に基づいて検索された最初のネームサービスから承認が削除されます。

オプション

auths サブコマンドは次のオプションをサポートします。

-h help_file_path 承認に関する情報を含むヘルプファイルの場所を設定します。

-S repository 変更または検索されるネームサービスリポジトリ (リポジトリ) を指定します。サポートされるリポジトリオプションは files と ldap です。

このオプションが省略されている場合は、[nsswitch.conf\(4\)](#) に基づいて検索されます。

-t description 承認のテキストでの説明を指定します。

- `-u user` 承認を一覧表示または確認するユーザー名 (*user*) を指定します。
- このオプションが省略されている場合は、現在のユーザーが使用されます。
- `-v` 承認の説明を出力します。

使用例

例1 auths コマンドの使用

auths の出力は次のようになります。

```
example% auths tester01 tester02
tester01 : solaris.system.date,solaris.jobs.admin
tester02 : solaris.system.*
example%
```

tester01 内の承認名を区切るコンマのあとにスペースはありません。

次のコマンドは、ユーザー tester01 に割り当てられている承認を一覧表示します。

```
example% auths list -u tester01
```

```
tester01:
solaris.jobs.admin
solaris.system.date
```

例2 承認の一覧表示

次のコマンドは、ユーザー tester01 に割り当てられている承認を説明とともに一覧表示します。

```
example% auths list -v -u tester01
tester01:
solaris.jobs.admin
Manage All Jobs
solaris.system.date
Set Date & Time
```

例3 承認の一覧表示

次のコマンドは、ネームサービス内の承認を説明とともに一覧表示します。

```
example% auths info -v solaris.user.manage
solaris.user.manage:
Manage user accounts
example%
```

例4 承認の追加

次のコマンドは、承認 `solaris.foo.manage` を説明 `manage foo` およびヘルプファイル `AuthFoo.html` とともにファイルのネームサービスリポジトリに追加します。

```
example% auths add -t "manage foo"\  
-h /home/abc/AuthFoo.html solaris.foo.manage
```

例5 承認の変更

次の例では、承認 `solaris.foo.manage` を変更し、説明を `manage foo and bar` に設定したあと、LDAP でヘルプファイルを `AuthFooBar.html` に設定します。

```
example% auths -S ldap modify -t "manage foo and bars"\  
-h /home/abc/AuthFooBar.html solaris.foo.manage
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。
- 2 承認されていないユーザー。

ファイル `/etc/user_attr`

 `/etc/security/auth_attr`

 `/etc/security/policy.conf`

 `/etc/security/prof_attr`

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [profiles\(1\)](#), [roles\(1\)](#), [getauthattr\(3C\)](#), [auth_attr\(4\)](#), [policy.conf\(4\)](#), [prof_attr\(4\)](#), [user_attr\(4\)](#), [attributes\(5\)](#)

名前	auto_ef - エンコーディングの自動検出																																										
形式	<pre>/usr/bin/auto_ef [-e <i>encoding_list</i>] [-a] [-l <i>level</i>] [<i>file</i> ...] /usr/bin/auto_ef -h</pre>																																										
機能説明	auto_ef ユーティリティーは、指定されたファイルのエンコーディングを識別します。このユーティリティーは、iconv コード変換を使用して特定のコード変換がファイルで成功したかどうかを判別し、またファイルに出現する文字シーケンスの頻度分析を実行して、エンコーディングを判断します。 文字列がバイナリ、文字テーブル、ローカライズ済みの楕リスト、またはクロノグラムの場合か、文字列またはファイルのサイズが非常に小さい場合 (100 バイト未満など)、auto_ef ユーティリティーが予期しない出力を生成する可能性があります。 <table> <tr><td>ASCII</td><td></td></tr> <tr><td>ISO-2022-JP</td><td>JIS</td></tr> <tr><td>eucJP</td><td>日本語 EUC</td></tr> <tr><td>PCCK</td><td>日本語 PC 漢字コード、CP932、シフト JIS</td></tr> <tr><td>UTF-8</td><td></td></tr> <tr><td>ko_KR.euc</td><td>韓国語 EUC</td></tr> <tr><td>ko_KR.cp949</td><td>統合型ハングル</td></tr> <tr><td>ISO-2022-KR</td><td>ISO-2022 韓国語</td></tr> <tr><td>zh_CN.iso2022-CN</td><td>ISO-2022 CN/CN-EXT</td></tr> <tr><td>zh_CN.euc</td><td>簡体字中国語 EUC、GB2312</td></tr> <tr><td>GB18030</td><td>簡体字中国語 GB18030/GBK</td></tr> <tr><td>zh_TW-big5</td><td>BIG5</td></tr> <tr><td>zh_TW-euc</td><td>繁体字中国語 EUC</td></tr> <tr><td>zh_TW.hkscs</td><td>香港 BIG5</td></tr> <tr><td>iso-8859-1</td><td>西ヨーロッパなど</td></tr> <tr><td>iso-8859-2</td><td>東ヨーロッパなど</td></tr> <tr><td>iso-8859-5</td><td>キリル文字など</td></tr> <tr><td>iso-8859-6</td><td>アラビア語</td></tr> <tr><td>iso-8859-7</td><td>ギリシャ語</td></tr> <tr><td>iso-8859-8</td><td>ヘブライ語</td></tr> <tr><td>CP1250</td><td>windows-1250、ISO-8859-2 に対応</td></tr> </table>	ASCII		ISO-2022-JP	JIS	eucJP	日本語 EUC	PCCK	日本語 PC 漢字コード、CP932、シフト JIS	UTF-8		ko_KR.euc	韓国語 EUC	ko_KR.cp949	統合型ハングル	ISO-2022-KR	ISO-2022 韓国語	zh_CN.iso2022-CN	ISO-2022 CN/CN-EXT	zh_CN.euc	簡体字中国語 EUC、GB2312	GB18030	簡体字中国語 GB18030/GBK	zh_TW-big5	BIG5	zh_TW-euc	繁体字中国語 EUC	zh_TW.hkscs	香港 BIG5	iso-8859-1	西ヨーロッパなど	iso-8859-2	東ヨーロッパなど	iso-8859-5	キリル文字など	iso-8859-6	アラビア語	iso-8859-7	ギリシャ語	iso-8859-8	ヘブライ語	CP1250	windows-1250、ISO-8859-2 に対応
ASCII																																											
ISO-2022-JP	JIS																																										
eucJP	日本語 EUC																																										
PCCK	日本語 PC 漢字コード、CP932、シフト JIS																																										
UTF-8																																											
ko_KR.euc	韓国語 EUC																																										
ko_KR.cp949	統合型ハングル																																										
ISO-2022-KR	ISO-2022 韓国語																																										
zh_CN.iso2022-CN	ISO-2022 CN/CN-EXT																																										
zh_CN.euc	簡体字中国語 EUC、GB2312																																										
GB18030	簡体字中国語 GB18030/GBK																																										
zh_TW-big5	BIG5																																										
zh_TW-euc	繁体字中国語 EUC																																										
zh_TW.hkscs	香港 BIG5																																										
iso-8859-1	西ヨーロッパなど																																										
iso-8859-2	東ヨーロッパなど																																										
iso-8859-5	キリル文字など																																										
iso-8859-6	アラビア語																																										
iso-8859-7	ギリシャ語																																										
iso-8859-8	ヘブライ語																																										
CP1250	windows-1250、ISO-8859-2 に対応																																										

CP1251	windows-1251、ISO-8859-5 に対応
CP1252	windows-1252、ISO-8859-1 に対応
CP1253	windows-1253、ISO-8859-7 に対応
CP1255	windows-1255、ISO-8859-8 に対応
koi8-r	iso-8859-5 に対応

デフォルトでは、`auto_ef` は、指定されたファイル内のテキストに対するもっとも可能性の高い1つのエンコーディングを返します。このファイルについて可能性のあるすべてのエンコーディングを取得するには、`-a` オプションを使用します。

また、デフォルトでは、`auto_ef` はもっとも高速な処理を使用してファイルを検査します。より正確な結果を得るには、`-l` オプションを使用します。

限定されたエンコーディングセットを使用してデータを検査するには、`-e` オプションを使用します。

オプション

サポートしているオプションは、次のとおりです。

- `-a` 可能性のあるすべてのエンコーディングを、可能性の高い順に **0.0 - 1.0** の範囲のスコアを付けて表示します。スコアが高いほど可能性も高くなります。たとえば、

```
example% auto_ef -a test_file
eucJP          0.89
zh_CN.euc      0.04
ko_KR.euc      0.01
```

このオプションを指定しない場合は、もっとも高いスコアを持つ1つのエンコーディングだけが表示されます。
- `-e encoding_list` 指定したエンコーディングだけを使ってデータを検査します。たとえば、*encoding_list* が「`ko_KR.euc:ko_KR.cp949`」に指定されている場合、`auto_ef` は `CP949` と `ko_KR.euc` だけを使用してテキストを検査します。このオプションを指定しない場合、`auto_ef` はすべてのエンコーディングでテキストを検査します。エンコーディングをコロン(:)で区切って、複数のエンコーディングを指定できます。
- `-h` 使用方法に関するメッセージを表示します。
- `-l level` 判断のレベルを指定します。*level* の値は、**0**、**1**、**2**、または**3**にできます。レベルを**3**にすると最善の結果が得られますが、時間のかかる場合があります。レベル**0**はもっとも高速ですが、高いレベルに比べて結果の正確性が低くなる場合があります。デフォルトのレベルは**0**です。

オペランド 次のオペランドがサポートされています。

file 検査対象のファイル名。

使用例 例1 ファイルのエンコードを検査する

```
example% auto_ef file_name
```

例2 ファイルのエンコーディングをレベル2で検査する

```
example% auto_ef -l 2 file_name
```

例3 eucJP または ko_KR.euc のみでファイルのエンコーディングを検査する

```
example% auto_ef -e "eucJP:ko_KR.euc" file_name
```

終了ステータス 次の終了ステータスが返されます。

0 正常終了

1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/auto_ef
インタフェースの安定性	確実

関連項目 [auto_ef\(3EXT\)](#), [libauto_ef\(3LIB\)](#), [attributes\(5\)](#)

『国際化対応言語環境の利用ガイド』

名前 awk - パターン走査およびパターン処理の言語

形式 `/usr/bin/awk [-f progfile] [-Fc] [' prog '] [parameters] [filename]...`
`/usr/xpg4/bin/awk [-FcERE] [-v assignment]... 'program' -f progfile... [argument]...`

機能説明 `/usr/xpg4/bin/awk` ユーティリティーについては [nawk\(1\)](#) のマニュアルページで説明しています。

`/usr/bin/awk` ユーティリティーは、*filename* で指定したファイルを走査し、*prog* で指定したパターンのいずれかと一致する行を探します。*prog* 文字列は、シェルと区別できるように単一引用符 (') で囲みます。*prog* で示される各パターンとともに、*filename* 内に一致する行が見つかったときに実行されるアクションを記述することができます。一連のパターン-アクション文は *prog* に直接書いても良く、`-f progfile` オプションで示されるファイル内に指定してもかまいません。ファイルは順番に読み込まれます。ファイルが指定されないときは、標準入力を使用されます。ファイル名 '-' は標準入力を意味します。

オプション サポートしているオプションは、次のとおりです。

`-f progfile` `awk` は *progfile* から読み込んだ一連のパターンを使用します。
`-Fc` フィールドセパレータ (FS) として *c* を使用します。下記の FS の説明を参照してください。

使用法

入力行 各行は、各パターン-アクション文のパターン部分と比較されます。一致すると、関連するアクションが実行されます。`var=value` の形式の *filename* はファイル名ではなく変数の割り当てとみなされ、その場合には、そのファイルがオープンされるであろう時点で行われます。この方法によって割り当てられた変数は BEGIN ルール内では使用できず、以前に指定されたファイルがすべて読み込まれた後で割り当てられます。

入力行は通常、スペースで区切られたフィールドで構成されています。(このデフォルト値は FS 組み込み変数または `-Fc` オプションを使用して変更できます)。デフォルトでは先行する空白文字を無視し、空白文字またはタブ、あるいはその両方でフィールドを区切るようになっています。ただし、変数 FS にスペースを含まない値が代入されていると、先行スペースを無視しません。フィールドは \$1、\$2、のように表されます。...;\$0 は全行を意味します。

パターン-アクション文

パターン-アクション文は次の形式をとります。

```
pattern { action }
```

パターンかアクションのどちらかを省略することができます。アクションが指定されていないときは、一致した行を印刷します。パターンが指定されていないときは、全行に対してアクションが実行されます。パターン-アクション文は復帰改行またはセミコロンで区切られます。

パターンは、正規表現および関係式を任意の論理演算子 (!, ||, &&, および括弧) によって組み合わせたものです。関係式は、次のいずれかです。

```
expression relop expression
expression matchop regular_expression
```

ここで、*relop* とは C 言語の 6 つの関係演算子のうちのどれかで、*matchop* とは ~ (含む) または !~ (含まない) のいずれかを示します。*expression* は、数値演算式、関係演算式、特殊式、

```
var in array
```

またはこれらを論理演算で組み合わせたものです。

egrep(1) と同様、正規表現はスラッシュで囲まれていなければなりません。パターン内の独立した正規表現はその行すべてに適用されます。正規表現は関係式の中にも現われます。パターンは、コンマで区切られた 2 つのパターンからなることもあります。この場合、関連するアクションは、最初のパターンが一致した行と 2 番目のパターンが一致した行の間にあるすべての行に対して実行されます。

BEGIN と END は特殊なパターンで、最初の入力行を読む前と最後の入力行を読んだ後に制御を確保するのに使用します。この 2 つの特殊パターンは他のパターンと組み合わせて使うことはできません。

組み込み変数

組み込み変数には次のものがあります。

FILENAME	現入力ファイル名
FS	入力フィールドセパレータの正規表現 (デフォルトは空白文字とタブ)
NF	現レコード中のフィールド数
NR	現レコード番号
OFMT	数値の出力形式 (デフォルトは %.6g)
OFS	出力フィールドセパレータ (デフォルトは空白文字)
ORS	出力レコードセパレータ (デフォルトは復帰改行文字)
RS	入力レコードセパレータ (デフォルトは復帰改行文字)

アクションは一連の文です。使用できる文は次のうちのいずれかです。

```
if ( expression ) statement [ else statement ]
while ( expression ) statement
do statement while ( expression )
for ( expression ; expression ; expression ) statement
for ( var in array ) statement
break
continue
```

```

{ [ statement ] . . . }
expression      # commonly variable = expression
print [ expression-list ] [ >expression ]
printf format [ ,expression-list ] [ >expression ]
next            # skip remaining patterns on this input line
exit [expr]     # skip the rest of the input; exit status is expr

```

文は、セミコロン、復帰改行、右括弧のうちのいずれかで終了します。式のリストが空の場合は入力行全体を意味します。式は、文字列または数字と +、-、*、/、%、^、および連結(空白文字で示される)の各演算子で構成されます。演算子 ++、--、+=、-=、*=、/=、%=、^=、>、>=、<、<=、==、!=、?: も式の中に記述できます。変数は、スカラー、配列要素(x[i] で表される)またはフィールドです。変数は NULL 文字列またはゼロで初期化されます。配列の添字は、必ずしも数字である必要はなく、文字列でもかまいません。これによって、ある種の連想記憶形式を使用できます。文字列定数は、二重引用符 (") で囲みます。C のエスケープ文字はエスケープとして認識します。

print 文はその引数を標準出力に出力します。>expression が指定されたときはファイルへ、|cmd が指定されたときはパイプへ出力します。出力は、現在の出力フィールドセパレータで区切られた各引数を持つ、出力レコードセパレータで終了します。printf 文は、その書式に従って式のリストの書式を定めます([printf\(3C\)](#) 参照)。

組み込み関数

演算関数は次のとおりです。

cos(x) x をラジアン単位とし、 x の余弦を返します。(usr/xpg4/bin/awk のみ。[nawk\(1\)](#) を参照)。

sin(x) x をラジアン単位とし、 x の正弦を返します。(usr/xpg4/bin/awk のみ。[nawk\(1\)](#) を参照)。

exp(x) x の指数関数を返します。

log(x) x の自然対数を返します。

sqrt(x) x の平方根を返します。

int(x) 引数を切り捨てて、整数にします。つまり x が 0 よりも大きい場合は、0 に向かって切り捨てます。

文字列関数は次のとおりです。

index(s, t) 文字列 s の中で文字列 t が最初に出現する位置を返します。出現しなければ 0 を返します。

int(s) 整数値になるよう、 s を切り捨てます。 s が指定されていないければ、\$0 が使われます。

<code>length(s)</code>	引数を文字列として解釈しその長さを返します。引数がない場合は行全体の長さを返します。
<code>split(s, a, fs)</code>	文字列 <i>s</i> を <i>a</i> [1]、 <i>a</i> [2]、... <i>a</i> [<i>n</i>] の配列要素に分割し、値 <i>n</i> を返します。この分割は、正規表現 <i>fs</i> によって行われ、 <i>fs</i> が指定されていない場合はフィールドセパレータ FS によって行われます。
<code>sprintf(fmt, expr, expr, ...)</code>	<i>fmt</i> で指定した printf(3C) 形式に従って式の書式を定め、その結果得られた文字列を返します。
<code>substr(s, m, n)</code>	文字列 <i>s</i> 内の <i>m</i> 番目から始まる長さ <i>n</i> の部分文字列を返します。

出入力用の関数は次のとおりです。

getline \$0 に、現入力ファイルの次の入力レコードを設定します。getline は正常終了時には 1 を、ファイルの終わりに達すると 0 を、またエラー発生時には -1 を返します。

大規模ファイルの動作

ファイルが 2G バイト(2³¹ バイト) 以上ある場合の awk の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 72 文字以上の行の出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な awk スクリプトです。このスクリプトは、72 文字を超える行を出力します。

```
length > 72
```

例2 フィールドの逆順出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な awk スクリプトです。このスクリプトは、最初の 2 フィールドを逆順に出力します。

```
{ print $2, $1 }
```

例3 入力フィールドを区切った、フィールドの逆順出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な awk スクリプトです。このスクリプトは、最初の 2 つの入力フィールドをコンマ、空白文字、またはタブで区切って逆順出力します。

```
BEGIN { FS = ",[ \t]*|[ \t]+" }  
      { print $2, $1 }
```

この例は、`/usr/xpg4/bin/awk` でのみ有効です。

例4 最初のカラムを合計して、合計値と平均値の出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な `awk` スクリプトです。このスクリプトは、最初のカラムを合計して、合計値と平均値を出力します。

```
{ s += $1 }
END { print "sum is", s, " average is", s/NR }
```

例5 フィールドの逆順出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な `awk` スクリプトです。このスクリプトは、フィールドを逆順に出力します。

```
{ for (i = NF; i > 0; --i) print $i }
```

例6 `start/stop` の間にあるすべての行の出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な `awk` スクリプトです。このスクリプトは、`start/stop` の間にあるすべての行を出力します。

```
/start/, /stop/
```

例7 最初のフィールドが前行と異なるすべての行の出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な `awk` スクリプトです。このスクリプトは、最初のフィールドが前行と異なるすべての行を出力します。

```
$1 != prev { print; prev = $1 }
```

例8 ページ番号付きでのファイルの出力

次の例は、`awk -f examplescript` の形式のコマンドで実行可能な `awk` スクリプトです。このスクリプトは、ページ番号付きで5ページからファイルを出力します。

```
/Page/      { $2 = n++; }
              { print }
```

例9 指定した数値から始まるページ番号でファイルを出力

このプログラムが `prog` のファイルに記録されている場合、次のコマンドは5ページから始まるページ番号で `input` ファイルを出力します。

```
example% awk -f prog n=5 input
```

環境変数

`awk` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、NLSPATH、および PATH。

LC_NUMERIC 数値入力の解釈、数値と文字列との変換、数値出力のフォーマットに用いる、小数点文字を決定します。`awk` プログラム (コマンド行引数で指定される代入も含む) の処理で認識される小数点文字は、ロ

ケールに関係なくピリオド (POSIX ロケールの小数点文字) です。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/awk

属性タイプ	属性値
使用条件	system/core-os
CSI	未対応

/usr/xpg4/bin/awk

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [egrep\(1\)](#), [grep\(1\)](#), [nawk\(1\)](#), [sed\(1\)](#), [printf\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 入力行にスペースが含まれる場合、出力時に保証されません。

数値と文字の間の明示的な変換は行われません。式を数値として扱いたい場合は `0` を加え、文字として扱いたい場合は NULL 文字列 ("") を連結してください。

名前	banner – make posters
形式	banner <i>strings</i>
機能説明	banner prints its arguments (each up to 10 characters long) in large letters on the standard output.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/extended-system-utilities

関連項目 [echo\(1\)](#), [attributes\(5\)](#)

名前	basename, dirname – パス名の部分的な抽出
形式	<pre>/usr/bin/basename string [suffix] /usr/xpg4/bin/basename string [suffix] dirname string</pre>
機能説明	basename ユーティリティーは、<code>/</code>で終わるすべてのプレフィックスと <i>suffix</i> (<i>string</i> 中にある場合) を <i>string</i> から削除して、その結果を標準出力に出力します。このユーティリティーは通常、シェルプロシージャ内の置換マーク (<code>' '</code>) の中で使用されます。
/usr/bin	<i>suffix</i> は、 expr(1) のマニュアルページに定義されているパターンです。
/usr/xpg4/bin	<i>suffix</i> は、含まれるどの文字にも特殊な意味が付加されていない文字列です。 dirname ユーティリティーは、<i>string</i> 内のパス名の最後のレベルを除く、すべてのレベルを出力します。
使用例	例1 環境変数を設定する 次の例は、引数 <code>/home/sms/personal/mail</code> を指定して呼び出しを行なった場合で、環境変数 <code>NAME</code> に <code>mail</code> というファイルを設定し、環境変数 <code>MYMAILPATH</code> に文字列 <code>/home/sms/personal</code> を設定します。 <pre>example% NAME='basename \$HOME/personal/mail' example% MYMAILPATH='dirname \$HOME/personal/mail'</pre> 例2 ファイルをコンパイルして出力を移動する 次のシェルプロシージャは、引数 <code>/usr/src/bin/cat.c</code> を指定して呼び出しを行なった場合で、指定したファイルをコンパイルし、出力を現在のディレクトリ内の <code>cat</code> というファイルに移動します。 <pre>example% cc \$1 example% mv a.out 'basename \$1 .c'</pre>
環境変数	basename と dirname の実行に影響を与える次の環境変数についての詳細は、environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLS_PATH。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

/usr/bin

属性タイプ	属性値
使用条件	system/core-os

/usr/xpg4/bin

属性タイプ	属性値
使用条件	system/xopen/xcu4
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目

[expr\(1\)](#), [basename\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 basename – display portions of pathnames

形式 /usr/ucb/basename *string* [*suffix*]

機能説明 The basename utility deletes any prefix ending in '/' and the *suffix*, if present in *string*. It directs the result to the standard output, and is normally used inside substitution marks (' ') within shell procedures. The *suffix* is a string with no special significance attached to any of the characters it contains.

使用例 例 1 Using the basename command.

This shell procedure invoked with the argument /usr/src/bin/cat.c compiles the named file and moves the output to cat in the current directory:

```
example% cc $1
```

```
example% mv a.out 'basename $1 .c'
```

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [sh\(1\)](#), [attributes\(5\)](#)

名前	bc – arbitrary precision arithmetic language
形式	<code>/usr/bin/bc [-c] [-l] [file]...</code> <code>/usr/xpg6/bin/bc [-c] [-l] [file]...</code>
機能説明	The bc utility implements an arbitrary precision calculator. It takes input from any files given, then reads from the standard input. If the standard input and standard output to bc are attached to a terminal, the invocation of bc is <i>interactive</i> , causing behavioral constraints described in the following sections. bc processes a language that resembles C and is a preprocessor for the desk calculator program dc, which it invokes automatically unless the -c option is specified. In this case the dc input is sent to the standard output instead.
使用法	The syntax for bc programs is as follows: <i>L</i> Means a letter a–z, <i>E</i> Means an expression: a (mathematical or logical) value, an operand that takes a value, or a combination of operands and operators that evaluates to a value, <i>S</i> Means a statement.
Comments	Enclosed in <code>/*</code> and <code>*/</code> .
Names (Operands)	Simple variables: <i>L</i> . Array elements: <i>L</i> [<i>E</i>] (up to BC_DIM_MAX dimensions). The words <code>ibase</code> , <code>obase</code> (limited to BC_BASE_MAX), and <code>scale</code> (limited to BC_SCALE_MAX).
Other Operands	Arbitrarily long numbers with optional sign and decimal point. Strings of fewer than BC_STRING_MAX characters, between double quotes ("). (<i>E</i>) <code>sqrt (<i>E</i>)</code> Square root <code>length (<i>E</i>)</code> Number of significant decimal digits. <code>scale (<i>E</i>)</code> Number of digits right of decimal point. <code><i>L</i> (<i>E</i>, ..., <i>E</i>)</code>
Operators	<code>+ - * / % ^</code> (% is remainder; ^ is power) <code>++ --</code> (prefix and postfix; apply to names) <code>== <= >= != < ></code> <code>= += -= *= /= %= ^=</code>
Statements	<i>E</i> <code>{ <i>S</i> ; ... ; <i>S</i> }</code> <code>if (<i>E</i>) <i>S</i></code>


```
while ( E ) S
for ( E ; E ; E ) S
null statement
break
quit

.string
```

Function Definitions

```
define L ( L , ... , L ) {
  auto L , ... , L
  S ; ... S
  return ( E )
}
```

Functions in -l Math Library

s (x)	sine
c (x)	cosine
e (x)	exponential
l (x)	log
a (x)	arctangent
j (n, x)	Bessel function

All function arguments are passed by value.

The value of a statement that is an expression is printed unless the main operator is an assignment. Either semicolons or new-lines may separate statements. Assignment to `scale` influences the number of digits to be retained on arithmetic operations in the manner of `dc`. Assignments to `ibase` or `obase` set the input and output number radix respectively.

The same letter may be used as an array, a function, and a simple variable simultaneously. All variables are global to the program. `auto` variables are stacked during function calls. When using arrays as function arguments or defining them as automatic variables, empty square brackets must follow the array name.

オプション The following operands are supported:

	-c	Compiles only. The output is <code>dc</code> commands that are sent to the standard output.
/usr/bin/bc	-l	Defines the math functions and initializes <code>scale</code> to 20, instead of the default zero.
/usr/xpg6/bin/bc	-l	Defines the math functions and initializes <code>scale</code> to 20, instead of the default zero. All math results have the scale of 20.

オペランド

The following operands are supported:

file A pathname of a text file containing bc program statements. After all cases of *file* have been read, bc reads the standard input.

使用例

例 1 Setting the precision of a variable

In the shell, the following assigns an approximation of the first ten digits of π to the variable *x*:

```
x=$(printf "%s\n" 'scale = 10; 104348/33215' | bc)
```

例 2 Defining a computing function

Defines a function to compute an approximate value of the exponential function:

```
scale = 20
define e(x){
    auto a, b, c, i, s
    a = 1
    b = 1
    s = 1
    for(i=1; 1==1; i++){
        a = a*x
        b = b*i
        c = a/b
        if(c == 0) return(s)
        s = s+c
    }
}
```

例 3 Printing the approximate values of the function

Prints approximate values of the exponential function of the first ten integers:

```
for(i=1; i<=10; i++) e(i)
```

or

```
for (i = 1; i <= 10; ++i) {            e(i) }
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of bc: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

終了ステータス

The following exit values are returned:

0 All input files were processed successfully.

unspecified An error occurred.

ファイル

/usr/lib/lib.b mathematical library

/usr/include/limits.h to define BC_ parameters

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [dc\(1\)](#), [awk\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 The bc command does not recognize the logical operators && and | | .
The for statement must have all three expressions (E's).

名前	bdiff - 大規模ファイルの diff
形式	bdiff filename1 filename2 [n] [-s]
機能説明	bdiff は diff と同様の機能を持ち、<i>filename1</i> と <i>filename2</i> 引数で示した 2 つのファイルの相違箇所を見つけ出します。本コマンドは、diff が処理可能な限度を超えた大きなファイルを比較するために提供されています。どちらかの引数にファイル名の代わりに - を指定すると、標準入力と比較対象となります。 bdiff は両ファイルの先頭にある共通部分を無視し、残りの部分を <i>n</i> 行ずつのセグメントに分割し、個々のセグメントを比較するために diff を呼び出します。ファイル名以外の 2 つのオプションを両方とも指定する場合、前述の「形式」で示した順序で記述しなければなりません。 bdiff の出力は diff の出力とまったく同じ形式です。行番号もファイルの分割を考慮して連続したものが割り当てられます。したがって出力リストは、ファイル全体が一度に比較されたかのように見えます。ただし分割により比較が行われているため、bdiff は、全体を一度に比較した場合に比べて、必ずしもファイルの微妙な差を探し出せるわけではないことに注意してください。
オプション	<i>n</i> 分割の単位とする行数を指定します。デフォルトは 3500 です。3 番目の引数が記述されていてその値が数値の場合、自動的に行数指定と見なされます。本引数は、デフォルトの行数である 3500 行が diff にとって多すぎて処理できない、といった場合に使用すると便利です。 -s サイレントオプション。bdiff が通常出力する診断メッセージを抑止します。ただし bdiff が呼び出す diff からの診断メッセージは、本オプションが指定されていても出力されます。
使用法	ファイルが 2 ギガバイト (2 ³¹ バイト) 以上ある場合の bdiff の動作については、 largefile(5) を参照してください。
ファイル	/tmp/bd????
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目	diff(1) , attributes(5) , largefile(5)
診断	診断メッセージの意味については help を使用してください。

名前	bfs - 大型ファイル用のスキャナ
形式	/usr/bin/bfs [-] <i>filename</i>
機能説明	bfs コマンドは、ほとんど ed(1) と同じですが、読み取り専用で、より大きなファイルを処理する点が異なります。ファイルは最大 1024K バイト、32K 行で、1 行あたりに復帰改行を含めて最大 512 文字 (16 ビットマシンの場合は 255 文字) を含めることができます。bfs ではファイルがバッファにコピーされないため、ファイルの走査は一般に ed(1) よりも効率的です。csplit(1) を使用して、大きなファイルを編集のために扱いやすい大きさのいくつかのファイルに分割できる場合、bfs は、大きなファイルの各セクションを識別するのに役立ちます。 通常、w(write) コマンドで書き込んだファイルのサイズが出力されるのと同様に、走査しているファイルのサイズが出力されます。オプションの - は、サイズの出力を抑制します。p とキャリッジリターンを入力すると、ed(1) の場合と同様に入力を求めるプロンプトとして * が出力されます。プロンプトは、再度 p とキャリッジリターンを入力するとオフにできます。メッセージは、プロンプトがオンになっている場合には、エラーに対してメッセージが出力されることに注意してください。 ed(1) で説明されているアドレス表現式はすべてサポートされています。また、正規表現を、/ と ? 以外の 2 つの記号で囲むことができます。 > 折り返しのない下方検索を示します。 < 折り返しのない上方検索を示します。 マーク名には多少の違いがあります。つまり、a - z の文字だけが使用可能であり、26 のマークすべてが記憶されます。
bfs コマンド	e、g、v、k、p、q、w、=、!、および NULL の各コマンドは、ed(1) の説明どおりに動作します。——、+++—、+++＝、-12、+4p などのコマンドは受け入れられません。1、10p、および 1,10 は、いずれも最初の 10 行を出力します。f コマンドは、走査中のファイルの名前だけを出力します。ファイル名の記憶機能はありません。w コマンドは、出力先の変更、切り捨て、または圧縮機能には影響されません (以下の xo、xt、および xc の各コマンドを参照)。次の追加コマンドを使用できます。 xf <i>file</i> 指定した <i>file</i> からコマンドを取り出します。ファイルの終わり、割り込みシグナルの受信、またはエラーの発生により、コマンドの読み取りは、xf を含むファイルに戻ります。xf コマンドは、10 の深さまでネストすることができます。 xn 現在使用中のマークをリストします (マークは、k コマンドで設定されます)。 xo [<i>file</i>] p および NULL コマンドからの出力を、指定した <i>file</i> に送ります。このファイルは、umask の設定 (umask(1) を参照) で別の指定がされているのでな

ければ、必要に応じてモード 666 (だれでも読み書きが可能) で作成されます。 *file* を指定しないと、出力は標準出力に送られます。出力先の変更を行うたびに、ファイルの切り捨てや作成が行われることに注意してください。

: label

コマンドファイル内に *label* 定義します。 *label* は復帰改行で終了し、 *:* (コロン) と *label* の先頭の間にあるブランクは無視されます。ラベルは参照されなくてもよいので、このコマンドは、コマンドファイル内に注釈を挿入するためにも使用できます。

(., .)xb/regular expression/label

このコマンドが正常に実行されると、 *label* への (上方または下方) ジャンプが行われます。次の条件のいずれかが成立すると、ジャンプは失敗します。

1. アドレスが 1 と \$ の間にない。
2. 2 番目のアドレスが 1 番目のアドレスよりも小さい。
3. 最初と最後の行を含む指定した範囲内に、正規表現と一致する行がない。

正常終了すると、一致した行に *.* (ドット) が設定されて、 *label* へのジャンプが行われます。このコマンドは、アドレスが間違っているときにもエラーメッセージを出さない唯一のコマンドです。他のコマンドを実行する前にアドレスが間違っていないかを検査するために使用できます。コマンド、 *xb/^/ label* は無条件ジャンプであることに注意してください。

xb コマンドは、端末以外から読み取られた場合にのみ使用できます。パイプから読み取られた場合は、下方ジャンプだけしか実行できません。

xt number

p および NULL コマンドからの出力を、 *number* で指定した文字数まで切り捨てます。初期値は 255 です。

xv [digit] [spaces] [value]

xv に続いて指定された *digit* が変数名になります。コマンド *xv5100* と *xv5 100* は、どちらも値 100 を変数 5 に割り当てます。コマンド *xv61, 100p* は、値 1、100p を変数 6 に割り当てます。変数を参照するには、変数名の前に % を付けます。たとえば次の例は、前述の割り当てを変数 5 と 6 に使用します。

```
1,%5p
1,%5
%6
```

すべてで、最初の 100 行が出力されます。

```
g/%5/p
```

前述のコマンドは、文字 **100** をグローバルに検索して、一致した各行を出力します。`%` の特殊な意味をエスケープするには、その前に `\` を付ける必要があります。

```
g/".*\%[cds]/p
```

文字、10 進整数、または文字列の `%c`、`%d`、または `%s` 形式 (たとえば、`printf` のような文) を照合してリストするために使用できます。`xv` コマンドの機能には、このほかに UNIX コマンドの出力の最初の行を変数に格納できるというものがあります。そのための条件は、*value* の最初の文字が `!` でなければならないということです。例:

```
.w junk
xv5!cat junk
!rm junk
!echo "%5"
xv6!expr %6 + 1
```

前述のコマンドは、現在の行を変数 35 に入れ、それを出力して、変数 36 を 1 増分します。*value* の最初の文字としての `!` の特殊な意味をエスケープするには、その前に `\` を付けます。

```
xv7!\!date
```

前述のコマンドは、値 `!date` を変数 7 に格納します。

```
xbz label
xbn label
```

これらの 2 つのコマンドは、UNIX コマンド (`!` コマンド) の実行で最後に保存されたリターンコード、または 0 以外の値をそれぞれテストして、指定された *label* にジャンプします。次の 2 つの例はいずれも、文字列 *size* を含む次の 5 行を検索するものです。

例 1:

```
xv55
: l
/size/
xv5!expr %5 - 1
!if 0%5 != 0 exit 2
xbrn l
```

例 2:

```
xv45
: l
/size/
xv4!expr %4 - 1
!if 0%4 = 0 exit 2
xbz l
```

xc [switch]

switchが1の場合、p および NULL コマンドからの出力は圧縮されます。switchが0の場合は圧縮されません。引数がない場合、xc は switch を反転します。switchは、最初は圧縮を行わないように設定されています。圧縮された出力では、タブとブランクからなる文字列は1つのブランクに置き換えられ、空白行は出力されません。

オペランド

次のオペランドを指定できます。

filename

最大 1024K バイト、32K 行、1 行あたりに復帰改行を含め最大 512 文字 (16 ビットマシンの場合は 255 文字) を収納するすべてのファイル。filename には、[csplit\(1\)](#) によって編集用に扱いやすい大きさに分割された大きなファイルの 1 つのセクションを指定することができます。

終了ステータス

次の終了ステータスが返されます。

0

ファイルにもコマンドにもエラーはなく、正常終了しました。

>0

エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[csplit\(1\)](#), [ed\(1\)](#), [umask\(1\)](#), [attributes\(5\)](#)

診断

プロンプトがオフの場合、コマンドにエラーがあると?だけが表示されます。プロンプトがオン場合は、詳しいエラーメッセージが表示されます。

- 名前

biff – give notice of incoming mail messages
- 形式

/usr/ucb/biff [y | n]
- 機能説明

biff turns mail notification on or off for the terminal session. With no arguments, biff displays the current notification status for the terminal.

If notification is allowed, the terminal rings the bell and displays the header and the first few lines of each arriving mail message. biff operates asynchronously. For synchronized notices, use the MAIL variable of [sh\(1\)](#) or the mail variable of [csh\(1\)](#).

A 'biff y' command can be included in your ~/.login or ~/.profile file for execution when you log in.
- オプション

y Allow mail notification for the terminal.

n Disable notification for the terminal.
- ファイル

~/.login User's login file

~/.profile User's profile file
- 属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

- 関連項目

[csh\(1\)](#), [mail\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

名前	break, continue – while、for、foreach、until の各ループ制御から抜け出す、または続行するためのシェル組み込み関数
形式	
sh	break [n] continue [n]
csch	break continue
ksh88	*break [n] *continue [n]
ksh	+break [n] +continue [n]
機能説明	
sh	break ユーティリティーは for ループまたは while ループ中であれば、ループを終了します。n を指定すると、n レベル分ブレイクします。 continue は for ループまたは while ループの次の繰り返しを実行します。n を指定すると、n 番目のループから実行します。
csch	break は foreach または while 内の最も内側にあるループの end の次から実行を再開します。現在の行の残りのコマンドが実行されます。これによって、複数レベルのブレイクを break コマンドのリストとして 1 行にすべて書き込めます。 continue は while または foreach 内の最も内側にあるループの、次の繰り返しから実行します。
ksh88	break は for ループ、while ループ、until ループ、または select ループがあれば終了します。n を指定すると、n レベル分だけループを終了します。処理に含まれるループ数より n が大きい場合には、もっとも外側のループで終了します。 continue は for ループ、while ループ、until ループ、または select ループの次の繰り返しを実行します。n を指定すると、n 番目のループから実行します。処理に含まれるループ数より n が大きい場合には、最も外側のループが使用されます。 このマニュアルページでは、1 つまたは 2 つのアスタリスク (*) が先頭に付加されている ksh88(1) コマンドは、次の方法で特殊な処理を受けます。 1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。 2. 入出力のリダイレクトは変数代入後に行われます。 3. エラーが発生すると、それを含むスクリプトは中止されます。

- 4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号の後に実行され、ワード分割とファイル名生成は実行されません。

ksh break は、もっとも内側の for、select、while、または until ループを終了するシェルの特殊組み込み関数です。また、n が指定されている場合は、n 番目のループを終了します。ループの次のコマンドから、実行が続行されます。

n が指定されている場合、これは正の整数 (≥1) でなければなりません。処理に含まれるループ数より n が大きい場合には、最後のループが終了します。

continue は、シェルの特殊組み込み関数で、もっとも内側の for、select、while、until ループの先頭 (存在する場合)、または n 番目のループの先頭 (n が指定されている場合) で、実行を続行します。

n が指定されている場合、これは正の整数 (≥1) でなければなりません。処理に含まれるループ数より n が大きい場合には、最後のループが使用されます。

このマニュアルページでは、1 つまたは 2 つの + 記号が先頭に付加されている ksh(1) コマンドは、特殊な組み込みコマンドであり、次のように処理されます。

- 1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
- 2. 入出力のリダイレクトは変数代入後に行われます。
- 3. エラーが発生すると、それを含むスクリプトは中止されます。
- 4. 組み込みコマンドは、有効な関数名ではありません。
- 5. 変数代入形式で、++ から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

属性 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 csh(1), exit(1), ksh(1), ksh88(1), sh(1), attributes(5)

名前	builtin – ksh 組み込み関数によるシェル組み込みの追加、削除、または表示
形式	builtin [-ds] [-f lib] [pathname ...]
機能説明	ksh builtin コマンドは、現在のシェル環境内で組み込みコマンドを追加、削除、または表示します。組み込みコマンドは現在のシェルのプロセス内で実行されるため、現在のシェルに副作用を引き起こす場合があります。ほとんどのシステムでは、組み込みコマンドの起動時間は、個別のプロセスを作成するコマンドの 10 分の 1 から 100 分の 1 未満になります。 指定した <i>pathname</i> ごとに、パス名のベース名に基づいて組み込み名が決定されます。ベース名ごとに、シェルは、組み込み名の先頭に <code>b_</code> が付加された C レベルの関数を現在のシェル内で検索します。<i>pathname</i> にスラッシュ (/) が含まれる場合、組み込みコマンドは <i>pathname</i> にバインドされています。パス名にバインドされた組み込みコマンドが実行されるのは、パス検索時に見つかった最初の実行可能ファイルが <i>pathname</i> である場合だけです。それ以外の場合、組み込みコマンドはパス検索を実行する前に見つかります。 <i>pathname</i> が指定されていない場合、builtin は現在の組み込みコマンドの一覧、または <code>-s</code> オプションが指定されているときには特殊な組み込みコマンドだけを標準出力に表示します。パス名にバインドされた組み込みコマンドのフルパス名が表示されます。 組み込みコマンドを含むライブラリを指定するには、<code>-f</code> オプションを使用します。ライブラリに <code>lib_init()</code> という名前の関数が含まれる場合、ライブラリの読み込み時にこの関数が引数 0 を付けて呼び出されます。<code>lib_init()</code> 関数を使用して組み込みコマンドを読み込むには、適切な C レベル関数を呼び出します。この場合、C レベル関数の名前に制限はありません。 C レベル関数は、3 つの引数を付けて呼び出されます。最初の 2 つの引数は <code>main()</code> と同じで、3 番目の引数はポインタです。 制限付きシェルから ksh builtin コマンドを呼び出すことはできません。
オプション	サポートしているオプションは、次のとおりです。 <code>-d</code> 指定した各組み込みコマンドを削除します。特殊な組み込みコマンドは削除できません。 <code>-f lib</code> 動的リンクの存在するシステムでは、組み込みコマンドの読み込みおよび検索は共有ライブラリ <i>lib</i> 内で行われます。 ライブラリの検索は、<code>\$PATH</code> およびシステム依存ライブラリディレクトリ内で行われます。システム依存共有ライブラリの接頭辞または接尾辞は省略できます。ライブラリが読み込まれると、そのシンボルを現在および後続の builtin の呼び出しで使用できるようになります。複数のライブラリを指定するには、builtin を別個に呼び出します。ライブラリは、指定された順序とは逆の順序で検索されます。

次のオペランドがサポートされています。

次の終了ステータスが返されます。

>0 エラーが発生した。

次の例では、builtin コマンド `mycmd` をライブラリ `libfoo.so` から読み込みます。

著者 David Korn、dgk@research.att.com

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

関連項目 [ksh\(1\)](#), [whence\(1\)](#), [attributes\(5\)](#)

名前	cal – カレンダの表示
形式	cal [[month] year]
機能説明	cal ユーティリティーは、グレゴリ暦のカレンダを標準出力に書き込みます。year オペランドを指定すると、その年のカレンダが書き込まれます。オペランドを指定しないと、現在の月のカレンダが書き込まれます。
オペランド	次のオペランドがサポートされています。 <i>month</i> 表示する月を、1 (1 月) - 12 (12 月) の 10 進数で指定します。デフォルト値は、現在の月です。 <i>year</i> カレンダを表示する年を、1 - 9999 の 10 進数で指定します。デフォルト値は、現在の年です。
環境変数	cal の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_TIME、LC_MESSAGES、および NLS_PATH。 TZ 現在の月の値を計算するのに使用するタイムゾーンを決定します。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目	calendar(1) , attributes(5) , environ(5) , standards(5)
注意事項	1752 年の 9 月のカレンダ出力は、通常のものとは異なります。この月は、うるう年の調整の欠落を補うために 11 日分省かれています。このカレンダを見るには、次のように入力します。 cal 9 1752 コマンド cal 83 は、西暦 1983 年ではなく、西暦 83 年を出力します。 年は、常に 1 月から始まるものと見なされます。

名前	calendar – reminder service
形式	calendar [-]
機能説明	The calendar utility consults the file calendar in the current directory and writes lines that contain today's or tomorrow's date anywhere in the line to standard output. Most reasonable month-day dates such as Aug. 24, august 24, 8/24, and so forth, are recognized, but not 24 August or 24/8. On Fridays and weekends “tomorrow” extends through Monday. calendar can be invoked regularly by using the crontab(1) or at(1) commands. When the optional argument - is present, calendar does its job for every user who has a file calendar in his or her login directory and sends them any positive results by mail(1). Normally this is done daily by facilities in the UNIX operating system (see cron(1M)). If the environment variable DATEMSK is set, calendar will use its value as the full path name of a template file containing format strings. The strings consist of conversion specifications and text characters and are used to provide a richer set of allowable date formats in different languages by appropriate settings of the environment variable LANG or LC_TIME; see environ(5). See strftime(3C) for the list of allowable conversion specifications.
使用例	例 1 Possible contents of a template The following example shows the possible contents of a template: <pre>%B %eth of the year %Y</pre> %B represents the full month name, %e the day of month and %Y the year (4 digits). If DATEMSK is set to this template, the following calendar file would be valid: <pre>March 7th of the year 1989 <Reminder></pre>
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of calendar: LC_CTYPE, LC_TIME, LC_MESSAGES, NLSPATH, and TZ.
終了ステータス	<pre>0 Successful completion. >0 An error occurred.</pre>
ファイル	<pre>/etc/passwd system password file /tmp/cal* temporary files used by calendar /usr/lib/calprog program used to determine dates for today and tomorrow</pre>
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目	<code>at(1)</code> , <code>crontab(1)</code> , <code>mail(1)</code> , <code>cron(1M)</code> , <code>ypbind(1M)</code> , <code>strftime(3C)</code> , <code>attributes(5)</code> , <code>environ(5)</code>
注意事項	Appropriate lines beginning with white space will not be printed. Your calendar must be public information for you to get reminder service. calendar's extended idea of "tomorrow" does not account for holidays. The <code>-</code> argument works only on calendar files that are local to the machine; calendar is intended not to work on calendar files that are mounted remotely with NFS. Thus, 'calendar <code>-</code>' should be run only on diskful machines where home directories exist; running it on a diskless client has no effect. calendar is no longer in the default root crontab. Because of the network burden 'calendar <code>-</code>' can induce, it is inadvisable in an environment running <code>ypbind(1M)</code> with a large <code>passwd.byname</code> map. If, however, the usefulness of calendar outweighs the network impact, the super-user may run 'crontab <code>-e</code>' to edit the root crontab. Otherwise, individual users may wish to use 'crontab <code>-e</code>' to edit their own crontabs to have cron invoke calendar without the <code>-</code> argument, piping output to mail addressed to themselves.

名前 cat – ファイルの連結と表示

形式 /usr/bin/cat [-nbsuvet] [file...]

機能説明 cat ユーティリティーは *file* を指定された順に読み込み、標準出力に出力します。したがって、

```
example% cat file
```

は、*file* を端末に出力します。

```
example% cat file1 file2 >file3
```

は、*file1* と *file2* を連結して、結果を *file3* に出力します。入力ファイルが指定されなかったときは、cat は標準入力ファイルから読み込みます。

オプション /usr/bin/cat では次のオプションを使用できます。

-b -n と同様に、行番号を振ります。ただし、空行は数えません。

-n 各出力行の前に、行番号を振ります。

-s cat は、ファイルが存在しないとき何もしません。

-u 出力はバッファリングされません

デフォルトでは、出力はバッファリングされます。

-v タブを除く非印刷文字、NEWLINE、およびフォームフィードが明確に出力されます。ASCII 制御文字 (8 進数の 000 から 037) は ^n として出力されます。ここで *n* は対応する ASCII 文字で、8 進数で 100 から 137 (@、A、B、C、.、.、X、Y、Z、[、\、]、^、_) の範囲の文字です。DEL 文字 (8 進数で 0177) は ^? として出力されます。その他の非印刷文字は M-*x* として出力されます。ここで、*x* は ASCII の下位 7 ビットで示される文字です。

-v オプションを使用した場合、次のオプションも使用できます。

-e \$ 記号が各行の最後 (NEWLINE の前) に出力されます。

-t タブは ^I として、フォームフィードは ^L としてそれぞれ出力されます。

-e オプションおよび -t オプションは、-v オプションが指定されていなければ無視されます。

オペランド 次のオペランドを指定できます。

file 入力ファイルのパス名。*file* の指定がない場合は、標準入力を使用します。*file* として - が指定されると、cat はその時点で標準入力を読み込みます。このように - をいくつでも *file* として指定できますが、cat はそのたびに標準入力をクローズしたり再オープンすることはありません。

使用法	ファイルが2G バイト (2^{31} バイト) 以上ある場合の <code>cat</code> の動作については、 largefile(5) を参照してください。
使用例	例1 1つのファイルを書き出す 次のコマンドは、<code>myfile</code> というファイルの内容を標準出力に書き出します。 <pre>example% cat myfile</pre> 例2 2つのファイルを1つのファイルに連結する 次のコマンドは、2つのファイル <code>doc1</code> と <code>doc2</code> を連結してその結果を <code>doc.all</code> に書き出します。 <pre>example% cat doc1 doc2 > doc.all</pre> 例3 <code>cat</code> の呼び出し1回で2組の入力を連結する このコマンドは、標準入力が端末の場合に、1回の <code>cat</code> の呼び出しで端末から2組の入力データを取得します。 <pre>example% cat start - middle - end > file</pre> このコマンドは、標準入力が端末の場合には、1回の <code>cat</code> の呼び出しで、端末から2組の入力データを取得するものです。 標準入力が通常ファイルの場合、 <pre>example% cat start - middle - end > file</pre> 次のコマンドと同等になります。 <pre>cat start - middle /dev/null end > file</pre> なぜなら、1つ目の <code>-</code> 指定に対して <code>cat</code> は標準入力ファイルの内容をすべて読み取ってしまい、2つ目の <code>-</code> 指定に対してはただちにファイルの終わり (EOF) が検出されるためです。
環境変数	<code>cat</code> の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。
終了ステータス	次の終了ステータスが返されます。 0 入力ファイルはすべて正常に出力されました。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [touch\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 `cat` の出力先を入力中のファイルに変更すると、入力ファイルのデータが失われます。たとえば、次のように指定します。

```
example% cat filename1 filename2 > filename1
```

この例では `filename1` の元データが失われます。

名前	cd, chdir, pushd, popd, dirs – 現在の作業用ディレクトリの変更
形式	<code>/usr/bin/cd [directory]</code>
sh	<code>cd [argument]</code> <code>chdir [argument]</code>
csh	<code>cd [dir]</code> <code>chdir [dir]</code> <code>pushd [+n dir]</code> <code>popd [+n]</code> <code>dirs [-l]</code>
ksh88、ksh	<code>cd [-L] [-P] [arg]</code> <code>cd old new</code>

機能説明

<code>/usr/bin/cd</code>	<code>/usr/bin/cd</code> ユーティリティは、 <code>cd</code> ユーティリティ自身だけの現在のディレクトリを変更します。これは、後述するシェル組み込みの <code>cd</code> とは対照的です。 <code>/usr/bin/cd</code> はプロセスの呼び出しには影響しませんが、あるディレクトリを現在のディレクトリとして設定できるかどうかを決定するのに使用できます。
sh	Bourne シェルに組み込まれている <code>cd</code> は、現在のディレクトリを <i>argument</i> で指定されたディレクトリに変更します。シェルパラメータ <code>HOME</code> は、 <i>argument</i> のデフォルト値です。シェルパラメータ <code>CDPATH</code> は、 <i>argument</i> を含むディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:)で区切ります。デフォルトのパスは空の文字列です(現在のディレクトリの指定)。現在のディレクトリは空のパス名で指定します。このパス名は、等号の直後か、パスリスト内にある区切り文字のコロンの間に指定します。 <i>argument</i> の先頭文字が <code>/</code> 、 <code>.</code> 、 <code>..</code> の場合、検索パスは使用しません。それ以外の場合は、パス中の各ディレクトリで <i>argument</i> を検索します。 <code>cd</code> は、 <i>argument</i> 中で実行(検索)権を持っていない限りなりません。コマンドを実行するたびに新しいプロセスが生成されるため、 <code>cd</code> を通常のコマンドとして実装した場合は効率が悪くなります。そのため、 <code>cd</code> コマンドは、シェルに組み込まれています。(pwd(1) 、 sh(1) 、 chdir(2) を参照)
	<code>chdir</code> は、 <code>cd</code> を呼び出すもうひとつの方法です。
csh	<i>dir</i> 引数を指定しない場合、Cシェルに組み込まれている <code>cd</code> は、シェル変数 <code>HOME</code> の値を新たな作業用ディレクトリとして使用します。 <i>dir</i> を指定した場合、それが <code>/</code> 、 <code>.</code> 、または <code>..</code> で始まる完全なパス名であれば、その <i>dir</i> が新たな作業ディレクトリとなります。それ以外の場合は、シェル変数 <code>CDPATH</code> が指定するパスと相対関係を持つディレクトリの中から該当するものを探し出します。 <code>CDPATH</code> の構文は <code>PATH</code> シェル変数と同一で、セマンティクスも似ています。 <code>cd</code> は <i>dir</i> に対する実行(検索)権を持っていない限りなりません。コマンドを実行するたびに新しいプロセスが生

成されるため、`cd` を通常のコマンドとして実装した場合は、効率が悪くなります。そのため、`cd` コマンドは、C シェルに組み込まれています。`(pwd(1)、sh(1)、chdir(2))` を参照)

`chdir` はシェルの作業用ディレクトリを *dir* が示すディレクトリに変更します。引数を指定しないと、そのユーザーのホームディレクトリに変更します。*dir* が現在のディレクトリには見つからない相対パス名の場合、変数 `cdpath` 内のディレクトリリストを検索します。*dir* が `/` で始まる値を持つシェル変数の名前である場合、その変数の値が示すディレクトリに変更します。

`pushd` はディレクトリスタックにディレクトリをプッシュ(押し込む)します。引数を指定しないと、トップの2つの構成要素を交換します。

`+n` *n* 番目のエントリとトップスタックを交換しそれに `cd` します。

dir 現在の作業用ディレクトリをスタックにプッシュし、*dir* を新たな作業用ディレクトリとします。

`popd` はディレクトリスタックからポップして(取り出して)、新たに先頭となったディレクトリへ `cd` します。ディレクトリスタックの構成要素は、トップディレクトリを0として番号付けられます。

`+n` スタック内の *n* 番目のエントリを破棄します。

`dirs` はディレクトリスタックを出力します。現在のディレクトリが最も左に現れるように時間順に出力されます。`-l` 引数を指定すると、`~` を使った省略形ではなく、完全な形式で出力されます。

ksh88、ksh

Korn シェルに組み込まれた `cd` コマンドは、前述の2つの形式のいずれかで入力します。第1の形式は、現在のディレクトリを *arg* に変更します。*arg* が `-` の場合、ディレクトリを直前のディレクトリに変更します。シェル変数 `HOME` の値がデフォルトの *arg* になります。`PWD` 環境変数は、現在のディレクトリに設定されます。`PWD` が変更された場合、`OLDPWD` 環境変数も古い作業用ディレクトリに変更されます。つまり、`cd` を呼び出す直前における現在の作業用ディレクトリのことです。シェル変数 `CDPATH` は、*arg* を含むディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(`:`)で区切ります。デフォルトのパスは空の文字列です(現在のディレクトリの指定)。現在のディレクトリは空のパス名で指定します。このパス名は、等号の直後か、パスリスト内にある区切り文字のコロンの間に指定します。*arg* の先頭文字が `/`、`.`、または `..` の場合、検索パスは使用しません。それ以外の場合は、パス中の各ディレクトリで *arg* を検索します。成功しなかった場合、`cd` は、`PWD` の値、スラッシュ文字、および *arg* を連結したパス名のディレクトリへ変更しようとして失敗します。

`-L` オペランドのドットドット(`..`)を論理的に処理します。ドットドット部分が処理される前に、シンボリックリンク部分は解釈処理されません。

`-P` オペランドのドットドット(`..`)を物理的に処理します。ドットドット部分が処理される前に、シンボリックリンク部分は解釈処理されます。

-L と -P の両方のオプションを指定した場合、最後に呼び出された方のオプションが使用され、他方のオプションは無視されます。-L オプションと -P オプションをどちらも指定しない場合、オペランドのドットドット (..) は論理的に処理されます。

cd の第 2 の形式は、PWD 中の現在のディレクトリ名における *old* という文字列を *new* という文字列に置換し、この新規のディレクトリへ変更しようとしています。

cd コマンドは rksh では実行できません。コマンドを実行するたびに新しいプロセスが生成されるため、cd を通常のコマンドとして実装した場合は、効率が悪くなります。そのため、cd コマンドは、ksh に組み込まれています。(pwd(1)、sh(1)、chdir(2)を参照)

オペランド

次のオペランドがサポートされています。

directory 新たな作業用ディレクトリとなるディレクトリの絶対または相対パス名。cd が相対パス名をどのように解釈するかは、環境変数 CDPATH の設定により異なります。

出力

CDPATH に設定されている空でないディレクトリ名が用いられる場合、新たな作業用ディレクトリの絶対パス名が以下のような形式で標準出力に出力されます。

```
"%s\n", <new directory>
```

それ以外の場合には、何も出力されません。

環境変数

cd の実行に影響を与える次の環境変数についての詳細は、environ(5)を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

CDPATH コロンで区切られた、ディレクトリを示すパス名のリスト。*directory* オペランドの先頭文字がスラッシュ (/) でなく、先頭部分がドットでもドットドットでもない場合には、cd はこのリスト内のパス名を順番に検索し、環境変数 CDPATH に指定されている名前のディレクトリから *directory* を探します。その結果、最初に見つかったディレクトリ名が新たな作業用ディレクトリとなります。ディレクトリのパス名として空の文字列を指定すると、それは現在のディレクトリと見なされません。CDPATH は、設定されていないときには空の文字列として扱われます。

HOME *directory* オペランドが省略されたときに用いるホームディレクトリの名前

OLDPWD 直前の作業用ディレクトリのパス名。この変数は、cd- で使用されます。

PWD 現在の作業用ディレクトリのパス名。この変数は、そのディレクトリに移った後に cd により設定されます。

0	ディレクトリが正常に変更されました。
>0	エラーが発生した。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

名前	<code>cdrw</code> – CD の読み込みと書き込み
形式	<pre> cdrw -i [-vSCO] [-d device] [-p speed] [image-file] cdrw -a [-vSCO] [-d device] [-p speed] [-T audio-type] audio-file1 [audio-file2]... cdrw -x [-v] [-d device] [-T audio-type] track-number out-file cdrw -c [-vSC] [-d device] [-p speed] [-m tmp-dir] [-s src-device] cdrw -b [-v] [-d device] all session fast cdrw -L [-v] [-d device] cdrw -M [-v] [-d device] cdrw -l [-v] cdrw -h </pre>
機能説明	<code>cdrw</code> コマンドを使用すると、データ CD および音楽 CD を作成することができます。また、音楽 CD から音楽トラックを取り出したり、データ DVD を作成したりできます。<code>cdrw</code> コマンドで CD あるいは DVD を作成するには、CD デバイスあるいは DVD デバイスが MMC 準拠である必要があります。 <code>-d</code> オプションでデバイスを指定しない場合、<code>cdrw</code> はシステムに接続された CD ライターあるいは DVD ライターを探します。このようなデバイスがシステム上で1つだけ見つかった場合、<code>cdrw</code> コマンドはそのデバイスをデフォルトの CD ライターあるいは DVD ライターとして扱います。 2つ以上の CD ライターあるいは DVD ライターがシステムに接続されている場合は、<code>-d</code> オプションによりデバイスを指定してください。CD ライターのデバイス名は、<code>/dev/rdisk/cNtNdNsN</code>、<code>cNtNdNsN</code>、<code>cNtNdN</code> のように指定されます。また、ボリュームマネージャで使用する <code>cdrom</code> や <code>cdrom1</code> などのシンボル名でも指定できます。<code>-l</code> オプションを使用すると、CD ライターあるいは DVD ライターのリストを表示できます。 USB 外部記憶装置クラスに準拠した CD-RW あるいは DVD-RW デバイスをシステムに追加する方法については、scsa2usb(7D) を参照してください。
データ CD の作成	データ CD を作成する場合、<code>cdrw</code> はトラック書き込みモード (Track-At-Once) を使用します。CD メディアに書き込むデータのファイルを指定するには、<code>-i</code> オプションを使用します。このオプションを指定しない場合、<code>cdrw</code> は標準入力からデータを読み込みます。 どちらの場合でも、データを準備するときには、通常、<code>mkisofs</code> コマンドを使用して、ファイルおよびファイル情報を CD で使用される High Sierra フォーマットに変換します。コマンドの使用方法については「使用例」を参照してください。

データ DVD の作成 `cdrw` は、`mkisofs` から生成されたイメージを使用することによって、シングルセッションのデータ DVD を DVD+RW デバイスあるいは DVD-RW デバイス上で作成できます。これらのディスクは HFSFS ファイルシステムとしてマウントできます。データ DVD を作成するとき、`cdrw` は Disk-At-Once (DAO) 書き込みモードを使用します。このモードは、書き込みが完了したときにメディアを閉じることによって、それ以上セッションが追加されないようにします。DAO モードではイメージのサイズがあらかじめ判明している必要があるため、イメージを DVD メディアに書き込むときには、そのイメージをあらかじめ準備しておく必要があります。

音楽 CD の作成 `-a` オプションを使用すると、音楽 CD を作成できます。このオプションには、1 つあるいは複数の音楽ファイルを指定できます。すべての音楽ファイルはサポートされた音楽形式でなければなりません。現在対応している形式は以下のものです。

`sun` レッドブック CD-DA 形式の `Sun .au` ファイル

`wav` レッドブック CD-DA 形式の RIFF (`.wav`) ファイル

`cda` 生の CD 音楽データを含む `.cda` ファイル (リトルエンディアン 16 bit PCM ステレオ 44.1 KHz サンプリング)

`aur` 生の CD データを含む `.aur` ファイル (ビッグエンディアン)

音楽形式が指定されていない場合は、`cdrw` はファイルの拡張子により音楽形式を認識しようとします。ファイル拡張子の太文字と小文字は区別されません。`-t` オプションにより音楽形式を指定した場合は、指定されたすべてのファイルの音楽形式として扱われます。また、`-cdrw` は音楽トラックを書き込んだ後セッションをクローズします。したがって、書き込まれるトラックは 1 つのコマンド行で指定する必要があります。

音楽の取り出し `cdrw` に `-x` オプションを指定すると、音楽 CD から音楽データを取り出すこともできます。CD にはレッドブック CD-DA 形式のトラックが存在する必要があります。デフォルトでは、出力の形式はファイルの拡張子に応じたものになります。`-t` オプションを使用すると、出力形式を `sun`、`wav`、`cda`、`aur` のいずれかに指定できます。

CD のコピー `cdrw` はシングルセッションのデータ CD-ROM やレッドブックの音楽 CD のコピーに使用できます。CD をコピーする場合、`cdrw` は指定されたソースデバイスを探します。`-c` オプション使用時にソースデバイスが指定されていない場合は、現在の書き込み用 CD デバイスがソースデバイスとみなされます。`cdrw` は一時ファイルに音楽トラックを取り出し、現在の書き込み用 CD デバイ스에空の書き込み可能 CD-R/CD-RW メディアが入っているかどうかを調べます。メディアが見つからない場合、空の書き込み可能 CD メディアを現在の CD ライターに挿入します。デフォルトの一時ディレクトリに十分な領域がない場合、`-m` オプションを使用して代替のディレクトリを指定できます。

CD-RW あるいは DVD-RW メディアの消去

ユーザーは再書き込みする前に CD-RW メディアを消去する必要があります。-b オプションは以下の消去方法をサポートしています。

session 最後のセッションを消去する
fast メディア全体を最小限消去する
all メディア全体を完全に消去する

session 消去では **cdrw** は最後のセッションを消去します。セッションが1つしか書き込まれていない CD-RW (たとえば、このツールで作成したデータあるいは音楽 CD-RW) の場合、記録された部分だけが消去されます。ほかの空の部分は処理されません。したがって、メディア全体を消去する場合よりも高速です。DVD メディアの場合は、-b **session** を指定するとメディア全体が消去されます。

fast 消去は、最初のセッションの PMA と TOC だけを削除することで、メディア全体を最小限消去します。メディア上にあるユーザーデータや後続のトラックなどは消去しません。それでも、このメディアは空のディスクとして処理されます。メディアを完全に消去する必要がある場合は、**all** オプションを使用します。

all 消去は、マルチセッションの CD、最後のセッションがクローズされていない場合、CD のステータスが不明な場合、ユーザーが CD 全体を消去したい場合に使用してください。この場合、**cdrw** はディスク全体を消去します。

DVD+RW メディアは消去をサポートしません。DVD+RW メディアを再利用するには、単に、新しいイメージをそのメディアに書き込みます。その後、**cdrw** は既存のメディアをフォーマットおよび上書きします。

デバイスリストおよびメディアのステータスのチェック

システムに接続されている CD ライターあるいは DVD ライターのリストを表示するには、-l オプションを使用します。また、メディアによっては -M オプションによって、消去のステータスや内容リスト (TOC) が表示されます。また、-M オプションは、最後のセッションの開始アドレスや次の書き込み可能アドレスも表示します。この情報は、-o オプションを指定してマルチセッション CD を作成するときと一緒に使用されます。詳細は、SUNWfsman パッケージ内の **mkisofs(8)** のマニュアルページ (`/usr/share/man/man8/mkisofs.8`) を参照してください。

オプション

サポートしているオプションは、次のとおりです。

- a 音楽 CD を作成します。少なくとも1つの音楽ファイル (*audio-file*) を指定する必要があります。最大 99 までの音楽トラックを指定することができます。
- b CD-RW あるいは DVD-RW メディアを消去します。消去する方法として、**all**、**fast**、または **session** のいずれかの引き数を指定します。DVD+RW メディアは消去をサポートしません。しかし、消去しなくても、そのまま上書きできます。

- c CDをコピーします。他に引き数の指定がない場合、cdrwはデフォルトのCD書き込みデバイスをソースデバイスとみなします。この場合、コピー処理はソースメディアを一時ディレクトリに読み込み、コピー処理を続けるために空のメディアをドライブに入れるようにユーザーに要求します。
- C このオプションは廃止されています。

このオプションは、cdrwでドライブを照会してメディアの容量を確認する場合に使用します。これは、現在のデフォルトの動作です。
- d CDあるいはDVD書き込みを行うデバイスを指定します。
- h ヘルプ。使用法を表示します。
- i データCDあるいはDVDを作成するイメージファイルを指定します。ファイルのサイズは、メディアに書き込める容量未満である必要があります。また、ファイルの位置は、NFSマウントされたファイルシステムではなく、ローカルで利用できる位置である必要があります。これは、CD書き込み処理が、途切れない連続したデータの供給を必要とするためです。
- l システムで利用できるすべてのCDライターあるいはDVDライターのリストを表示します。
- L ディスクを閉じます。前回の書き込み操作の後、メディアが開いたままである場合に、そのメディアを閉じて、それ以上書き込まないようにします。この処理を行うことができるのは、再書き込み可能CD-RWメディアだけです。
- m CDあるいはDVDをコピーする際にトラックデータを置く一時ディレクトリ(デフォルトの一時ディレクトリ以外)を指定します。代替の一時ディレクトリが必要になる理由は、CDに含まれるデータ量が非常に多い場合があるためです。たとえば、80分の音楽CDの場合は800Mバイト、DVDの場合は4.7Gバイトのデータ量になることがあります。デフォルトの一時ディレクトリには十分な容量がない可能性があります。
- M メディアのステータスを表示します。cdrwはメディアが空かどうか、内容リスト(TOC)、最後のセッションの開始アドレス、および、ディスクがオープンな場合は次の書き込み可能アドレスを表示します。DVD+RWは消去をサポートしません。また、メディアには常にいくつかの内容が入っています。
- O ディスクをオープンのままにします。cdrwは通常セッションをクローズしますが、マルチセッションCDを作成する場合は次のセッションを追加するために、オープンのままにしておきます。
- p CDライターの書き込み速度を設定します。たとえば、-p4は4倍速で書き込みます。オプションの指定がない場合、cdrwはCDライターのデフォルトの書き込み速度を使用します。オプションが指定された場合、cdrwはドライブに指定された速度での書き込みを設定しようとしますが、実際にドライブがその速度で書き込めるという保証はありません。

- s CDをコピーするソースデバイスを指定します。
 - S 疑似書き込みモードです。このモードでは、cdrwはCDライターのレーザーをオフにして動作します。したがってメディアにはデータは書き込まれません。システムがCDライターに指定された書き込み速度で、十分にデータを提供できるかどうか調べる場合にこのオプションを使用します。
- CD-R、CD-RW (MRW 形式をのぞく)、DVD-R、およびDVD-RW メディアは、疑似書き込みモード (-s) をサポートしています。DVD-RAM、DVD+R、DVD+RW、すべてのMRW形式のメディア、およびその他の一部のメディアは、疑似書き込みモード (-s) をサポートしていません。
- T 音楽CDを作成するために読み出す音楽ファイルあるいは取り出す音楽ファイルの形式を指定します。音楽形式 (*audio-type*) には、sun、wav、cda、aur のいずれかを指定できます。
 - v 冗長モード。
 - x 音楽トラックから音楽データを取り出します。

使用例

例1 データCDまたはデータDVDを作成する

```
example% cdrw -i /local/iso_image
```

例2 ディレクトリからのCDあるいはDVDの作成

以下は、/home/foo ディレクトリツリーからCDあるいはDVDを作成する例です。

```
example% mkisofs -r /home/foo 2>/dev/null | cdrw -i -p 1
```

例3 トラック番号を使用した音楽データの取り出し

以下は、トラック番号が1の音楽トラックを /home/foo/song1.wav に取り出す例です。

```
example% cdrw -x -T wav 1 /home/foo/song1.wav
```

例4 wavファイルの使用

以下は、ハードディスク上のwavファイルから音楽CDを作成する例です。

```
example% cdrw -a song1.wav song2.wav song3.wav song4.wav
```

例5 CD-RW あるいはDVD-RW メディアの消去

以下は、再書き込み可能メディアを消去する例です。

```
example% cdrw -b all
```

例6 複数のドライブでのデータ CD あるいは DVD の作成

以下は、複数の CD、DVD-R あるいは DVD-RW ドライブが接続されているシステムでデータ CD あるいは DVD を作成する例です。

```
example% cdwr -d c1t6d0s2 -i /home/foo/iso-image
```

例7 データ書き込み速度のチェック

以下は、システムが書き込み処理に対して十分な速度で、CD-RW あるいは DVD ドライブにデータを提供できるかどうかを確認する例です。

```
example% cdwr -S -i /home/foo/iso-image
```

例8 高優先度での実行

以下は、優先順位を上げて cdwr を実行する例です (ルートのみ)。

```
example# priocntl -e -p 60 cdwr -i /home/foo/iso-image
```

例9 マルチセッションディスクの作成

以下は、1 番目のセッションイメージを mkisofs で作成し、ディスクに記録した後、そのディスクを開いたままにしておく例です。

```
example% cdwr -O -i /home/foo/iso-image
```

オープンしたままのディスクには、mkisofs でイメージを作成する際に、cdwr が報告するセッション開始アドレスおよび次の書き込み可能なアドレスを使用して、ソフトウェアを追加することができます。

```
example% cdwr -M
```

Track No.	Type	Start address
1	Data	0
Leadout	Data	166564

Last session start address: 162140
Next writable address: 173464

```
example% mkisofs -o /tmp/image2 -r -C 0,173464 -M \  
/dev/rdisk/c0t2d0s2 /home/foo
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	media/cdrw

関連項目	audioconvert(1), priocntl(1), policy.conf(4), attributes(5), rbac(5), scsa2usb(7D), sd(7D) SUNWfsman パッケージ内の mkisofs(8) (/usr/share/man/man8/mkisofs.8) 『Oracle Solaris 11.1 の管理: デバイスとファイルシステム』
注意事項	CD 書き込み中、システムは一定の転送速度でドライブにデータを供給し続ける必要があります。CD 作成中は I/O 処理を最小限にし、不要なアプリケーションは終了しておくことをお勧めします。 コピーの作成や音楽トラックの取り出しには MMC 準拠のソース CD-ROM ドライブの使用をお勧めします。CD ライターはこの目的のために使用できます。 CD に書き込む前に、-M オプションを使用して、メディアが空であることを確認します。-s オプションの疑似モードを使用すると、システムが要求された書き込み速度でデータを提供できるかどうかをテストできます。ドライブがバッファアンダーラン保護をサポートする場合、cdrw はその保護を有効にして、ほとんどの機能停止から回復できます。システムが一定の速度でデータを提供できず、頻繁に機能停止する場合には、-p オプションを使用して、書き込み速度を落としてください。priocntl(1) コマンドを使用すると、優先順位を上げて cdrw を実行できます。 -p オプションを使用するのは、CD-R/RW ドライブの書き込み速度を変更できることが判明している場合だけです。一般に普及しているドライブでは、ドライブによって書き込み速度の設定コマンドの扱いが異なるため、注意してこのオプションを使用してください。 cdrw コマンドは、rbac(5) を使用してデバイスに対するユーザーのアクセスを制御しています。デフォルトでは、cdrw にアクセスできるユーザーに制限はありませんが、特定のユーザーしかアクセスできないように設定することもできます。詳細は、『Oracle Solaris 11.1 の管理: デバイスとファイルシステム』を参照してください。 CD を root 以外のユーザーで作成するには、hal を有効にして、ユーザーがコンソールにログインしている必要があります。hal (svc:/system/hal:SMF サービス) はデフォルトで有効になっているため、通常は特に操作する必要はありません。 ユーザーはコンソールにログインしている必要があります。/dev/console でもかまいません。以前は、ユーザーは telnet や ssh を使用してリモートでログインし、CD を作成できました。管理者がデフォルト構成を変更して solaris.device.cdrw 承認していないかぎり、この操作は有効です。policy.conf(4) を参照してください。

名前 checknr - nroff と troff 用の入力ファイルをチェックして誤りを報告

形式 checknr [-fs] [-a . x1 . y1 . x2 . y2 xn .yn] [-c . x1 . x2 . x3 xn] [filename]...

機能説明 checknr は [nroff\(1\)](#) または [troff\(1\)](#) 用の入力ファイルの内容をチェックして、起こりうるエラーを見つけ出します。見つけれられるエラーの種類は、区切り記号が対になっていない、未定義のコマンドが記述されている、などです。ファイル名を指定しないと、checknr は標準入力の内容をチェックします。チェックされる区切り記号は次のとおりです。

- 次の区切り記号で記述するフォント変更指定。 \fx ... \fP
- 次の区切り記号で記述するサイズ変更指定。 \sx ... \s0
- マクロのうち、開始と . 終了とが常に対となつて指定されるべきもの。たとえば、.TS と .TE マクロ

checknr は、[ms\(5\)](#) と [me\(5\)](#) マクロパッケージに対してもチェックできます。

checknr コマンドは、作者が checknr でのチェックを想定して記述したドキュメントの検査に適しています。特に \f と \s の両コマンドに関しては、特定のスタイルで記述されていなければなりません。具体的には、個々の \fx 指定は必ず対応する \fP 指定で終了し、同様に \sx 指定は必ず \s0 指定で終了していなければなりません。終了指定の代わりに元のフォントまたはサイズ指定を明示的に記述しても、指定としては有効ですし、実際に多くのドキュメントはそうように書かれていることでしょう。しかしそのようなドキュメントを checknr でチェックすると、大量のメッセージが出力されることになります。いずれにせよ、\fP や \s0 を使った形式の方が優れていますので、checknr に処理させるかどうかに関わらずこの形式を使うようお勧めします。

オプション

-f \f によるフォント変更指定を無視します。

-s \s によるサイズ変更指定を無視します。

-a .x1 .y1 ... マクロの対をリストに追加します。マクロは .DS と .DE のようになっている必要があり、そのようになっているかどうかを検査されます。-a の後には、6 文字一組でマクロの対を記述します。その 6 文字の内訳は、先頭がピリオド、次の 2 文字が開始マクロ名、次がピリオド、最後の 2 文字が終了マクロ名です。たとえば .BS と .ES の対を定義したければ、'-a.BS.ES' と指定します。

-c .x1 ... たとえ未定義であっても無視すべきコマンドを指定します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目 [eqn\(1\)](#), [nroff\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#), [me\(5\)](#), [ms\(5\)](#)

使用上の留意点 1 文字からなるマクロ名は、-a オプションを使って指定することができません。

名前	chgrp – ファイルのグループ所有権の変更
形式	<pre>chgrp [-fhR] group file... chgrp -s [-fhR] groupsid file... chgrp -R [f] [-H -L -P] group file... chgrp -s -R [f] [-H -L -P] groupsid file...</pre>
機能説明	chgrp ユーティリティーは、<i>file</i> で指定された各ファイルのグループ ID を、<i>group</i> で示す値に設定します。

file の個々のファイルに対し、chgrp は、[chown\(2\)](#) 関数を以下の引数とともに呼び出した場合と同等の動作を行います。

- *file* の値を *path* 引数に指定
- そのファイルのユーザー ID を *owner* 引数に指定
- 指定されたグループ ID を *group* 引数に指定

適切な特権を持ったプロセスが chgrp を呼び出さない限り、正常終了時に通常ファイルのセットユーザー ID ビットとセットグループ ID はクリアされます。他の種類のファイルのセットユーザー ID ビットとセットグループ ID もクリアされることがあります。

オペレーティングシステムは、所有者変更を制限する構成オプション `{_POSIX_CHOWN_RESTRICTED}` を持っています。このオプションが有効になっていると、ファイルの所有者は自分自身が属するグループへしかファイルのグループを変更できません。このオプションが有効かどうかに関わらず、スーパーユーザーだけが所有者 ID を変更できます。構成オプションを設定する場合は、`/etc/system` ファイルに次の行を挿入してください。

```
set rstchown = 1
```

このオプションを無効にする場合は、`/etc/system` ファイルに次の行を挿入してください。

```
set rstchown = 0
```

デフォルトでは `{_POSIX_CHOWN_RESTRICTED}` は有効です。[system\(4\)](#) と [fpathconf\(2\)](#) を参照してください。

オプション 以下のオプションがサポートされています。

/usr/bin/chgrp および /usr/xpg4/bin/chgrp	-f	強制。エラーは報告しません。
	-h	ファイルがシンボリックリンクであるとき、そのシンボリックリンクのグループを変更します。このオプションが指定されていない場合は、そのシンボリックリンクによって参照されるファイルのグループが変更されます。
	-H	コマンド行に指定したファイルがディレクトリタイプのファイルを参照するシンボリックリンクである場合、グループの変更の対象となるのは、そのシ

ンボリックリンクが参照するディレクトリ、および、そのディレクトリより下にあるファイル階層内のすべてのファイルです。ファイル階層の検索中にシンボリックリンクに出会った場合、対象となるファイルは変更されますが、再帰は発生しません。

- L ファイルがシンボリックリンクである場合、グループの変更の対象となるのは、そのシンボリックリンクが参照するファイルです。コマンド行に指定したファイル、あるいは、ファイル階層の検索中に会ったファイルがディレクトリタイプのファイルを参照するシンボリックリンクである場合、グループの変更の対象となるのは、そのシンボリックリンクが参照するディレクトリ、および、そのディレクトリより下にあるファイル階層内のすべてのファイルです。
- P コマンド行に指定したファイル、あるいは、ファイル階層の検索中に会ったファイルがシンボリックリンクである場合、グループの変更の対象となるのは、そのシンボリックリンク自身です。このオプションは、シンボリックリンクが参照するファイル階層は検索しません。
- s 指定されたグループは Windows SID です。このオプションでは、SID を格納可能なファイルシステム (ZFS など) が必要になります。

-H、-L、または -P は相互排他的なオプションですが、これらを複数指定しても、エラーであるとは判断されません。最後に指定したオプションが `chgrp` の動作を決定します。

<code>/usr/bin/chgrp</code>	-R 再帰。chgrp はディレクトリおよびすべてのサブディレクトリを検索し、指定されたグループ ID を設定していきます。シンボリックリンクに出会うと、(-h または -P オプションが指定されていなければ) 対象となるファイルのグループが変更されます。しかし、-H または -L オプションを指定しない限り、再帰は発生しません。
<code>/usr/xpg4/bin/chgrp</code>	-R 再帰。chgrp はディレクトリおよびすべてのサブディレクトリを検索し、指定されたグループ ID を設定していきます。シンボリックリンクに出会うと、(-h または -P オプションが指定されていなければ) 対象となるファイルのグループが変更されます。-H、-L、または -P のいずれのオプションも指定しない限り、-L オプションがデフォルトのモードとして使用されます。

オペランド 次のオペランドがサポートされています。

<i>group</i>	グループデータベースから得られるグループ名、または数値のグループ ID。どちらの場合でも、 <i>file</i> で指定した各ファイルに与えるグループ ID を表します。 <i>group</i> が数値で、それがグループ名としてグループデータベースに存在していると、そのグループ名に対応したグループ ID 番号がグループ ID として用いられます。
<i>file</i>	グループ ID を変更するファイルのパス名

- 使用法

ファイルが2ギガバイト (2³¹ バイト) 以上ある場合の chgrp の動作については、[largefile\(5\)](#) を参照してください。
- 環境変数

chgrp の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
- 終了ステータス

次の終了ステータスが返されます。

0 ユーティリティーの実行が正常終了し、要求されたすべての変更が行われた。
 >0 エラーが発生した。
- ファイル

/etc/group グループファイル
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/chgrp

属性タイプ	属性値
使用条件	system/core-os
CSI	有効。「注意事項」を参照してください。
インタフェースの安定性	確実
標準	standards(5) を参照してください。

/usr/xpg4/bin/chgrp

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効。「注意事項」を参照してください。
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目

[chmod\(1\)](#), [chown\(1\)](#), [id\(1M\)](#), [chown\(2\)](#), [fpathconf\(2\)](#), [group\(4\)](#), [passwd\(4\)](#), [system\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)
- 注意事項

chgrp は *group* 名を除いて CSI に対応しています。
- 148

マニュアルページセクション 1: ユーザーコマンド • 最終更新日 2008 年 7 月 11 日

名前	chkey – ユーザーの Secure RPC 鍵ペアの変更
形式	chkey [-p] [-s nis files ldap] [-m <mechanism>]
機能説明	chkey は、ユーザーの Secure RPC 公開鍵と秘密鍵のペアを変更するために使用されます。chkey は、古い Secure RPC パスワードを要求するプロンプトを出力し、秘密鍵を解読することによりそのパスワードが正しいことを検証します。ユーザーが、keylogin(1) を使用して秘密鍵を解読し、keyserv(1M) を使用して秘密鍵を保存する処理をまだ行っていない場合、chkey は、ローカルの keyserv(1M) デーモンを使用して秘密鍵を登録します。Secure RPC パスワードがログインパスワードと一致しない場合、chkey はログインパスワードの入力を求めるプロンプトを出力します。chkey は、ログインパスワードを使用して、ユーザーの Diffie-Hellman (192 ビット) 秘密鍵を暗号化します。chkey は、構成された認証メカニズム用に、ほかの Diffie-Hellman 鍵を暗号化することもできます。 chkey は、ログインパスワードと Secure RPC パスワードが同一であることを保証し、パスワードのシャドウ化を可能にします。shadow(4) を参照してください。 鍵ペアは、<code>/etc/publickey</code> ファイル (publickey(4) を参照) または NIS <code>publickey</code> マップに格納できます。新たに生成された秘密鍵は、ローカルの keyserv(1M) デーモンによって登録されます。 特別なメカニズムのための鍵は、<code>-m</code> オプションに認証メカニズム名を付けて指定することにより、変更または再暗号化できます。複数の <code>-m</code> オプションを使用すると、1 つまたは複数の鍵を変更できます。 <code>publickey</code> のソースが <code>-s</code> オプションで指定されていない場合、chkey は、ネームサービススイッチ構成ファイルの <code>publickey</code> エントリを参照します。nsswitch.conf(4) を参照してください。<code>publickey</code> エントリがソースを 1 つだけ指定する場合、chkey は、指定されたネームサービスの鍵を変更します。ただし、複数のネームサービスがリストされている場合、chkey は更新するソースを決定できず、エラーメッセージを表示します。ユーザーは、<code>-s</code> オプションを使用してソースを明示的に指定する必要があります。 root 以外のユーザーは、<code>files</code> データベース内の各自の鍵ペアを変更できません。
オプション	サポートしているオプションは、次のとおりです。 <code>-p</code> ユーザーのログインパスワードを使用して既存の秘密鍵を再暗号化します。 <code>-s nis</code> NIS データベースを更新します。 <code>-s files</code> <code>files</code> データベースを更新します。 <code>-s ldap</code> LDAP データベースを更新します。 <code>-m <mechanism></code> 指定したメカニズムの秘密鍵を変更または再暗号化します。

ファイル /etc/nsswitch.conf
 /etc/publickey

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [keylogin\(1\)](#), [keylogout\(1\)](#), [keyserv\(1M\)](#), [newkey\(1M\)](#), [nsswitch.conf\(4\)](#), [publickey\(4\)](#),
[shadow\(4\)](#), [attributes\(5\)](#)

名前	chmod – ファイルのアクセス権モードの変更
形式	<pre>chmod [-fR] <i>absolute-mode</i> file...</pre> <pre>chmod [-fR] <i>symbolic-mode-list</i> file...</pre> <pre>chmod [-fR] <i>acl_operation</i> file...</pre> <pre>chmod [-fR] [-@ <i>named_attribute</i>]...<i>attribute_specification_list</i> file...</pre>
機能説明	chmod ユーティリティーはファイルモードの変更や割り当てを行います。 chmod を使用すると、ファイルやディレクトリのアクセス制御リスト (ACL) の変更や、通常のファイル、ディレクトリ、および不透明な拡張属性ファイルのブール型読み書きシステム属性の変更も行えます。
絶対モード	絶対モードのコマンド行は、次の形式で指定します。 <pre>chmod [<i>options</i>] <i>absolute-mode</i> file...</pre> ここで、<i>absolute-mode</i> は、以下のように定義されている 8 進数 <i>nnnn</i> を使用して指定します。 <i>n</i> 0 から 7 の数字を表します。絶対モードは、以下のモードのいくつかの論理和をとったものです。 4000 実行時にユーザー ID を設定します。 20 # 0 # が 7、5、3 または 1 のとき、実行時にグループ ID を設定します。 # が 6、4、2 または 0 のとき、強制ロックを許可します。 ディレクトリに対して使用されると、グループ ID の継承については BSD のセマンティクスに基づいてファイルが作成されます。このオプションを使用すると、そのディレクトリ中に作成されるファイルやサブディレクトリは、カレントプロセスのグループ ID ではなく、ディレクトリのグループ ID を引き継ぎます。ディレクトリの場合、setgid ビットをセットまたはクリアできるのはシンボリックモードを使用する場合だけです。 1000 スティッキビットをオンにします。 chmod(2) 参照。 0400 所有者による読み取りを許可します。 0200 所有者による書き込みを許可します。 0100 所有者による実行 (ディレクトリ内での検索) を許可します。

0700

所有者による読み取り、書き込み、実行 (検索) を許可します。

0040

グループによる読み取りを許可します。

0020

グループによる書き込みを許可します。

0010

グループによる実行 (ディレクトリ内での検索) を許可します。

0070

グループによる読み取り、書き込み、実行 (検索) を許可します。

0004

その他のユーザーによる読み取りを許可します。

0002

その他のユーザーによる書き込みを許可します。

0001

その他のユーザーによる実行 (ディレクトリ内での検索) を許可します。

0007

その他のユーザーによる読み取り、書き込み、実行 (検索) を許可します。

ディレクトリに対して、絶対モードでは `setgid` のセットやクリアは行えません。シンボリックモードで `g+s` (または `g-s`) を使って行う必要があります。

シンボリック
モード

シンボリックモードのコマンド行は、次の書式で指定します。

```
chmod [options] symbolic-mode-list file...
```

ここで、*symbolic-mode-list* は、次の形式のシンボリックモードの式をコンマで区切ったリスト (スペースを含めない) です。

```
[who] operator [permissions]
```

処理は、記述された順序で実行されます。1つの演算子 (*operator*) のあとに *permissions* として複数のアクセス権文字を指定すると、その演算子で示された処理が同時に行われます。

who

u、g、o、a の文字の 1 つ以上の組み合わせ (または省略) で、誰のアクセス権が変更または割り当てられるのかを指定します。

u

ユーザーのアクセス権

g

グループのアクセス権

o
その他のアクセス権

a
全アクセス権(ユーザー、グループ、その他)

who が省略されると、デフォルトとして *a* と解釈されますが、ファイルモード作成マスク (詳細は [sh\(1\)](#) または [csh\(1\)](#) の *umask* の項を参照) の設定値は考慮されません。*who* が省略されると、*chmod* はユーザーマスクの制限を置き換えません。

operator

アクセス権をどのように変更するかを指定する演算子で、*+*、*-*、*=* のいずれかです。

+
アクセス権を付加します。

permissions を省略すると、何も付加されません。

who が省略されると、対応するビットがファイルモード生成マスク中にセットされている場合を除き、*permissions* が示すファイルモードビットが付加されます。

who が指定されていれば、*permissions* が示すファイルモードビットが付加されます。

-
アクセス権を除去します。

permissions を省略すると、何も行われません。

who が省略されると、対応するビットがファイルモード生成マスク中にセットされている場合を除き、*permissions* が示すファイルモードビットをクリアします。

who が指定されていれば、*permissions* が示すファイルモードビットをクリアします。

=
指定されたアクセス権をそのまま割り当てます。

who が省略されると、すべてのファイルモードビットをクリアします。*who* が指定されていれば、*who* が示すファイルモードビットをクリアします。

permissions を省略すると、他には何も行われません。

who が省略されると、対応するビットがファイルモード生成マスク中にセットされている場合を除き、*permissions* が示すファイルモードビットが付加されます。

`who` が指定されていれば、`permissions` が示すファイルモードビットが付加されます。

`=` は、他のシンボリック演算子と違って、`who` が示す他のすべてのビットをリセットしてしまうという絶対的な効果があります。`permission` の省略は、`=` を使用してすべてのアクセス権を除去するときのみ便利です。

permission

以下の文字の適切な組み合わせで指定します。

- `l`
強制ロッキング
- `r`
読み取り権
- `s`
ユーザーまたはセットグループ ID
- `t`
スティッキビット
- `w`
書き込み権
- `x`
実行権
- `X`
実行権 (ファイルがディレクトリの場合、または他のユーザークラスのいずれかに実行権がある場合)
- `u`、`g`、`o`
`permission` がそれぞれ現在のユーザー、グループ、またはその他のモードから除去されることを意味します。

ファイルへのアクセス権は、ユーザー ID 番号 (UID) およびグループ ID 番号 (GID) によって異なります。アクセス権は、3 つのユーザーモードがそれぞれ 3 つの文字を持つように記述されます。

ユーザー	グループ	その他
rwX	rwX	rwX

この例 (ユーザー、グループ、その他が、当該ファイルの読み取り、書き込み、実行権を持っている) は、アクセス権を与える際の 2 つのカテゴリを示しています。1 つはアクセスクラス、もう 1 つはアクセス権そのものです。

文字 *s* は、*u* または *g* と一緒に指定したときのみ有効です。また、*t* は *u* とのみ指定できます。

強制的なファイルおよびレコードのロック (*l*) は、プログラムがアクセスしている間、そのファイルの読み込み、書き込み権がロックされることを意味します。

セットグループ ID ビットがオンになっているディレクトリは、'*ls -ld*' で *-----s---* または *-----l---* が出力されることで判定できますが、このようなディレクトリ中でファイルやサブディレクトリを生成する際には、その親ディレクトリのセットグループ ID が与えられます。カレントプロセスのセットグループ ID ではありません。

グループ実行権を与えて、同時に実行時にファイルをロックすることはできません。さらに、セットグループ ID ビットをオンにして同時に実行時にファイルをロックすることはできません。したがって、以下の例は無効であり、エラーメッセージが出力されます。

```
chmod g+x,+l file
chmod g+s,+l file
```

ファイルまたはディレクトリの所有者 (またはスーパーユーザー) だけがそのファイルまたはディレクトリのモードを変更できます。スーパーユーザーだけがディレクトリでないファイルのスティッキビットを設定できます。スーパーユーザーでなければ、*chmod* はスティッキビットをマスクしますが、エラーは返しません。ファイルのセットグループ ID ビットをオンにするには、自分自身のグループ ID がファイルのそれと一致していて、さらにグループ実行権が設定されていなければなりません。

ACL 操作

アクセス制御リスト (ACL) とは、アクセス制御エントリ (ACE) のリストのことです。各エントリに特定クラスの利用者のアクセス権が定義されています。ACE のリストには、ゼロから始まる番号が付けられています。ACL 内での ACE の位置は *index* と呼ばれます。このインデックスは、次に示す多くの *chmod* コマンドの引数として使用されます。ACL と ACE の詳細は、『[Oracle Solaris 11.1 の管理: ZFS ファイルシステム](#)』を参照してください。

chmod を含む Oracle Solaris ユーティリティーは、NFSv4 と新しい POSIX ドラフト ACL 仕様の両方をサポートします。これらの仕様に、*acl_specification* フィールドの構文と意味が詳しく説明されています。これら 2 つの ACL 仕様については、次の各サブセクションで説明します。

ACL 操作のコマンド行は、次の形式で指定します。

```
chmod [options] A[index]- file ...
chmod [options] A-acl_specification file ...
chmod [options] A[index]{+|=}acl_specification file ...
```

ここで、*acl_specification* は、次の形式のコンマ区切りのリスト (スペースを含まない) です:

A[index]+acl_specification

acl_specification に指定されたアクセス制御エントリ (ACE) を、ファイルの ACL の先頭に追加します。ファイルシステムによっては、ファイルへの適用時に ACL が並べ替えられることがあります。オプションの *index* が指定された場合、*index* の前に新しい ACE が挿入されます。

A-

ファイル上の現在の ACL に対するすべての ACE を削除し、そのファイルの現在のモードだけを表現した新しい ACL で現在の ACL を置き換えます。

Aindex-

index の番号で指定された ACE を削除します。

A-acl_specification

acl_specification で指定された ACE が現在のファイルの ACL 内に存在している場合、そのエントリを削除します。

A=acl_specification

ファイルの ACL 全体を *acl_specification* で置き換えます。

A[index]=acl_specification

ファイル上の現在の ACL 内で、特定のインデックス番号以降の ACE を置き換えます。ACE が複数指定された場合、*acl_specification* 内の後続の各 ACE によって、現在の ACL 内の対応する ACE が置き換えられます。

POSIX ドラフト ACL 仕様 (UFS でサポート)

UFS でサポートされる POSIX ドラフト ACL は、次のようにコロン (:) で区切られたフィールドとして指定されます。

user::perms

所有者のアクセス権。

user:username:perms

特定のユーザーのアクセス権。

group::perms

ファイルのグループメンバーのアクセス権。

group:groupname:perms

特定のグループのアクセス権。

other::perms

ファイル所有者やファイルグループのメンバー以外のユーザーのアクセス権。

mask:perms

ACL マスク。マスクエントリは、ユーザー (所有者以外) とグループに許可される最大のアクセス権を示します。

default:user::perms

所有者のデフォルトアクセス権。

`default:user:username:perms`

特定のユーザーのデフォルトアクセス権。

`default:group::perms`

ファイルのグループメンバーのデフォルトアクセス権。

`default:group:groupname:perms`

特定のグループのデフォルトアクセス権。

`default:other:perms`

ファイルの所有者やファイルグループのメンバー以外のユーザーのデフォルトアクセス権。

`default:mask:perms`

デフォルトの ACL マスク。

前述の指定方法を使えば、ACL を次のように指定できます。

`user:tom:rw-,mask:rwx,group:staff:r-x`

NFSv4 ACL 仕様 (NFSv4 と ZFS でサポート)

NFSv4 ACL は、より豊富な ACL セマンティックを提供します。これらは、許可エントリと拒否エントリの両方を提供するほか、より細粒度の高いアクセス権と拡張された継承制御を提供します。

NFSv4 ACL は、次のようにコロン(:) で区切られたフィールドとして指定されます。

`owner@:<perms>[:inheritance flags]:<allow|deny>`

ファイルの所有者のアクセス権。

`group@:<perms>[:inheritance flags]:<allow|deny>`

ファイルのグループメンバーのアクセス権。

`everyone@:<perms>[:inheritance flags]:<allow|deny>`

ファイルの所有者とグループメンバーを含むすべてのユーザーのアクセス権。

`user:<username>:<perms>[:inheritance flags]:<allow|deny>`

特定のユーザーのアクセス権。

`usersid:<sid string>:<perms>[:inheritance flags]:<allow|deny>`

特定のユーザーのアクセス権。ただし、ユーザーは SID で指定されます。

`group:<groupname>:<perms>[:inheritance flags]:<allow|deny>`

特定のグループのアクセス権。

`groupsid:<sid string>:<perms>[:inheritance flags]:<allow|deny>`

特定のグループのアクセス権。ただし、グループは SID で指定されます。

`sid:<sid string>:<perms>[:inheritance flags]:<allow|deny>`

特定の SID のアクセス権。ただし、この SID はユーザーまたはグループのどちらでもかまいません。

アクセス権を指定する際には、異なる3つの chmod ACL 形式を使用できます。冗長、コンパクト、定位置です。冗長形式では、単語を使ってアクセス権を指定し、複数のアクセス権はスラッシュ (/) 文字で区切ります。コンパクト形式ではアクセス権文字を使用し、定位置形式ではアクセス権文字とハイフン (-) を使用します。ハイフンはアクセス権がないことを示します。

冗長モード用のアクセス権とコンパクトモードおよび定位置モード用の短縮形について、次に説明します。短縮形は括弧内に示しています。

read_data (r)

ファイルのデータを読み取る権限。

list_directory (r)

ディレクトリの内容を表示するためのアクセス権。

write_data (w)

ファイルのデータを変更する権限。ファイルのオフセット範囲内の任意の場所。

add_file (w)

ディレクトリに新しいファイルを追加するためのアクセス権。

append_data (p)

ファイルのデータを変更する機能。ただし、変更できるのは EOF 以降のみです。

現在、この権限はサポートされていません。

add_subdirectory (p)

ディレクトリにサブディレクトリを作成する権限。

read_xattr (R)

ファイルの拡張属性を読み取る機能。

write_xattr (w)

拡張属性を作成したり拡張属性ディレクトリに書き込んだりする機能。

execute (x)

ファイルを実行する権限。

read_attributes (a)

ファイルの基本属性 (ACL 以外) を読み取る機能。

write_attributes (A)

ファイルまたはディレクトリに関連付けられた時間を任意の値に変更するためのアクセス権。

delete (d)

ファイルを削除するためのアクセス権。

削除アクセス権の動作については、『[Oracle Solaris 11.1 の管理: ZFS ファイルシステム](#)』を参照してください。

`delete_child(D)`

ディレクトリ内のファイルを削除する権限。

削除アクセス権の動作については、『[Oracle Solaris 11.1 の管理: ZFS ファイルシステム](#)』を参照してください。

`read_acl(c)`

ファイルの ACL を読み取る権限。

`write_acl(C)`

ファイルの ACL を書き込む権限。

`write_owner(o)`

ファイルの所有者を変更する権限。

`synchronize(s)`

同期付きの読み取り権と書き込み権を使ってサーバー上のファイルにローカルアクセスする権限。

現在、この権限はサポートされていません。

コンパクト ACL 形式では、14 個の一意の文字を使ってアクセス権を指定します。

定位置 ACL 形式では、`ls -l` の形式に似た定位置引数として、アクセス権を指定します。ハイフン (-) は、その位置のアクセス権が許可されていないことを示しますが、このハイフンは省略可能であり、指定する必要があるのは必須文字だけです。

前述の文字は、定位置表記で指定する順番に並んでいます。

これらの文字を使えば、アクセス権を次のように指定できます。

`rxw--D-----`

ハイフンを削除すれば、文字列が次のようにコンパクトになります。

`rxwD`

いくつかの特殊アクセス権セットや別名もサポートされています。次のアクセス権セットの使用方法は、冗長アクセス権を指定する方法と同じです。

`full_set`

すべてのアクセス権。

`modify_set`

`write_acl` と `write_owner` を除くすべてのアクセス権。

`read_set`

`read_data`、`read_acl`、`read_attributes`、および `read_xattr`。

`write_set`

`write_data`、`append_data`、`write_attributes`、および `write_xattr`。

省略可能な継承フラグの指定形式は3つあります。最初の形式では、単語を使ってさまざまな継承フラグを指定します。複数の継承フラグはスラッシュ (/) 文字で区切ります。

`file_inherit(f)`

新たに作成されたすべてのファイルに継承します。

`dir_inherit(d)`

新たに作成されたすべてのディレクトリに継承します。

`inherit_only(i)`

ディレクトリに対して指定された場合、そのディレクトリには適用せず、新たに作成されたファイルやディレクトリだけに適用します。このフラグを使用する際には、`file_inherit` または `dir_inherit`、あるいはその両方も指定する必要があります。

`no_propagate(n)`

ディレクトリ内のオブジェクトにはACLエントリを継承しますが、レベルが1つ下がった時点でその継承を停止すべきであることを示します。このフラグを使用する際には、`file_inherit` または `dir_inherit`、あるいはその両方も指定する必要があります。

前述の継承フラグは、コンパクト形式で指定することもでき、また、`ls -v` の形式に似た定位置引数として指定することもできます。定位置ACL形式でのハイフン文字は、その位置の継承フラグが指定されていないことを示します。

これらの文字を使って継承フラグを指定する場合、次のいずれの同等の方法を使ってもかまいません。

```
file_inherit/dir_inherit/no_propagate
```

```
fd-n--
```

```
fdn
```

この継承モデルを使えば、ACLエントリを次のように指定できます。

```
user:tom:read_data/write_data/read_attributes:file_inherit:allow
user:fred:read_data:file_inherit/dir_inherit:deny
user:bob:read_data:allow
```

属性操作

属性操作のコマンド行は、次の形式で指定します。

```
chmod [options] attribute_specification_list file ...
```

`attribute_specification_list` は、文字 `s` とそれに続く1つ以上の `attribute_specifications` のコンマ区切りのリストです。各 `attribute_specification` の形式を次に示します。

```
[operator]attribute_specifier
```

`operator` は次のいずれかです。

+
関連する *attribute_specifier* で指定された各属性は、*attribute_specifier* で指定された値に一致するように調整されます。

-
関連する *attribute_specifier* で指定された各属性は、*attribute_specifier* で指定された値の逆に一致するように調整されます。

=
関連する *attribute_specifier* で指定された各属性は、*attribute_specifier* で指定された値に一致するように調整されます。*attribute_specifier* で指定されていない、現在のファイルに関連付けられたブール型の読み書き拡張システム属性は、クリアされます。

attribute_specification で *operator* が指定されていない場合、chmod は + が指定されているかのように動作します。

attribute_specifier は、次のいずれかの形式になります。

a
現在のファイルに関連付けられているすべてのブール型読み書き拡張システム属性を設定します。

c[*compact_attribute_list*]

c{'*compact_attribute_list*'}

compact_attribute_list で識別される各ブール型読み書き拡張システム属性を設定します。

v[*verbose_attribute_setting*]

v['{'*verbose_attribute_setting_list*'}]

verbose_attribute_setting で識別される各ブール型読み書き拡張システム属性を設定します。

compact_attribute_list は、このセクションで後述する属性名と略字のリストに記載されている、隣接するゼロ個以上の属性の略語のリストです。*compact_attribute_list* には、任意の数のハイフン (-) を含めることができます。これらは無視されます。

verbose_attribute_setting は、このセクションで後述する属性名と略字のリストに記載されている属性名です。必要に応じて、その直前に *no* を指定することもできます。属性名を *no* なしで指定する場合は、その属性が設定されます。それ以外の場合は、属性がクリアされます。

verbose_attribute_setting_list は、ゼロ個以上の *verbose_attribute_setting* のコンマ区切りのリストです。

1つのファイルに指定された複数の操作は累積され、そのすべてがファイルオペランドに対して1つの属性設定操作として設定されます。属性が

attribute_specification_list 内で複数回指定されている場合、最後に指定された操作が適用されます。

次に属性名と略字のリストを示します。

属性名
略字

hidden
H

sparse
s

system
S

readonly
R

archive
A

nounlink
u

immutable
i

appendonly
a

nodump
d

av_quarantined
q

av_modified
m

オプション

サポートしているオプションは、次のとおりです。

-f
強制。chmod はファイルモードの変更に失敗しても何もしません。

-R
ディレクトリ引数を再帰的に検索して、各ファイルのモードを設定します。シンボリックリンクに出会うと、対象のファイルのモードが変更されます。しかし、再帰は発生しません。

-@*named_attribute*
ファイルオペランド自体ではなく、各ファイルオペランドの名前付き拡張属性ファイルに対して属性操作を実行します。複数の **-@** 操作が指定されている場合、名前付き属性ファイルのそれぞれに属性指定モードが適用されます。

名前付き属性 * は、chmod に伝えられると、ファイルオペランドに関連付けられたすべての拡張属性ファイルを意味すると見なされます。これは特殊ファイル、および .. を参照しません。

.. の名前付き属性は、chmod に伝えられると、ファイルオペランド自体を意味すると見なされます。これにより、chmod を 1 回呼び出して、ファイルオペランドの指定された名前付き属性ファイルおよびファイルオペランド自体に属性指定モードを適用することが可能になります。

オペランド

次のオペランドがサポートされています。

absolute-mode

symbolic-mode-list

file オペランドのいずれかで指定された各ファイルのファイルモードビットに加えられる変更を示します。詳細は、このマニュアルページの「機能説明」の項の「絶対モード」および「シンボリックモード」を参照してください。

acl_operation

ファイルの ACL で実行される変更内容を表現します。詳細は、「機能説明」の項の「AOL 操作」を参照してください。

acl_operation は次のいずれかです。

A[number] -

A-acl_specification

A[index]{+|=}acl_specification

attribute_specification_list

ファイルの属性に対して実行される変更内容を表現します。詳細は、このマニュアルページの「機能説明」の項の「属性操作」を参照してください。

file

ファイルモードビットを変更するファイルのパス名

使用法

ファイルが 2 ギガバイト (2^{31} バイト) 以上ある場合の chmod の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 実行権を拒否する

次の例では、すべてのユーザーに対して実行権を拒否しています。

```
% chmod a-x file
```

例2 読み取り専用アクセス権を許可する

次の例では、すべてのユーザーに対して読み取り権だけを許可しています。

```
% chmod 444 file
```

例3 ファイルを読み取り可能かつ書き込み可能にする

次の例では、グループおよびその他のユーザーがファイルを読み書きできるようにしています。

```
% chmod go+rw file
% chmod 066 file
```

例4 ファイルをロックしてアクセスできないようにする

次の例では、ファイルをロックしてアクセスできないようにしています。

```
$ chmod +l file
```

例5 ファイルの読み取り、書き込み、実行、セットグループIDの各アクセス権を許可する

次の例では、ファイルの読み取り権、書き込み権、および実行権をすべてのユーザーに対して許可するとともに、セットグループIDをオンにしています。

```
$ chmod a=rwx,g+s file
$ chmod 2777 file
```

例6 ZFS ファイル上で新しいACLエントリを先頭に追加する

次の例では、ZFS ファイル上で新しいACLエントリを先頭に追加しています。

まず、現在のACLを表示します。

```
$ ls -v file.3
-rw-r--r--  1 marks  staff           0 Oct  9 15:49 file.3
0:owner@:execute:deny
1:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
2:group@:write_data/append_data/execute:deny
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

次のコマンドを入力します。

```
$ chmod A+user:lp:read_data:deny file.3
```

新しいACLを表示します。

```
$ ls -v file.3
-rw-r--r--+ 1 marks  staff           0 Oct  9 15:49 file.3
0:user:lp:read_data:deny
1:owner@:execute:deny
2:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
3:group@:write_data/append_data/execute:deny
```

例6 ZFS ファイル上で新しいACLエントリを先頭に追加する (続き)

```
4:group@:read_data:allow
5:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
6:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

例7 UFS ファイル上で新しいPOSIXドラフトACLエントリを先頭に追加する

次の例では、UFS ファイル上で新しいPOSIXドラフトACLエントリを先頭に追加しています。

まず、現在のACLを表示します。

```
$ ls -v file.2
-rw-r--r--  1 marks    staff          0 Oct  9 15:52 file.2
 0:user::rw-
 1:group::r--          #effective:r--
 2:mask:r--
 3:other:r--
```

次のコマンドを入力します。

```
$ chmod A+user:lp:-wx file.2
```

新しいACLを表示します。

```
$ ls -v file.2
-rw-r--r--+ 1 marks    staff          0 Oct  9 15:52 file.2
 0:user::rw-
 1:user:lp:-wx          #effective:---
 2:group::r--          #effective:r--
 3:mask:r--
 4:other:r--
```

例8 ZFS ファイル上でACLエントリを特定の位置に挿入する

次の例では、ZFS ファイルシステム上でACLエントリを特定の位置に挿入しています。また、コンパクトACL形式の使い方も示しています。

まず、ACLを表示し、新しいACEの挿入位置を確認します。

```
% ls -V file.1
-rw-r--r--+ 1 root      root          0 Oct  6 12:16 file.1
 user:lp:rw-----:-----:allow
 owner@:--x-----:-----:deny
 owner@:rw-p---A-W-Co-:-----:allow
 group@:-wxp-----:-----:deny
 group@:r-----:-----:allow
 everyone@:-wxp---A-W-Co-:-----:deny
 everyone@:r-----a-R-c--s:-----:allow
```

例8 ZFS ファイル上で ACL エントリを特定の位置に挿入する (続き)

次に、新しいエントリを位置3に挿入します。これにより、位置3から6に現在存在しているエントリが、下方に移動します。

次のコマンドを入力します。

```
$ chmod A3+user:marks:r:deny file.1
```

新しい ACL を表示します。

```
$ ls -V file.1
-rw-r--r--+ 1 root      staff          0 Feb  3 14:13 file.1
  user:lp:rw-----:-----:allow
  owner@:--x-----:-----:deny
  owner@:rw-p---A-W-Co:-----:allow
  user:marks:r-----:-----:deny
  group@:-wxp-----:-----:deny
  group@:r-----:-----:allow
  everyone@:-wxp---A-W-Co:-----:deny
  everyone@:r-----a-R-C--s:-----:allow
```

例9 UFS ファイル上で POSIX ドラフト ACL を特定の位置に挿入する

このファイルシステムは、ACL をファイルシステム内に格納する際にその ACL を並べ替えます。次の例は、この動作を説明したものです。

```
$ ls -v file.1
-rw-r--r--+ 1 root      root          0 Sep 29 16:10 file.1
  0:user::rw-
  1:user:lp:rw-      #effective:r--
  2:group::r--      #effective:r--
  3:mask:r--
  4:other:r--
```

ここで、インデックス位置3にエントリを挿入します。コマンドは機能しますが、ファイルシステムによって ACL が並べ替えられます。

```
$ chmod A3+user:marks:rw- file.1
$ ls -v file.1
-rw-r--r--+ 1 root      root          0 Sep 29 16:10 file.1
  0:user::rw-
  1:user:lp:rw-      #effective:r--
  2:user:marks:rw-   #effective:r--
  3:group::r--      #effective:r--
  4:mask:r--
  5:other:r--
```

ACL エントリは、要求された位置3に実際には挿入されず、最終的に位置2に挿入されています。

例10 ZFS ファイル上の ACL エントリを削除する

次の例では、ACL から lp エントリを削除しています。

```
$ ls -v file.3
-rw-r--r--+ 1 marks    staff          0 Oct  9 15:49 file.3
0:user:lp:read_data:deny
1:owner@:execute:deny
2:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
3:group@:write_data/append_data/execute:deny
4:group@:read_data:allow
5:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
6:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow

$ chmod A-user:lp:read_data:deny file.3
$ ls -v file.3
-rw-r--r--  1 marks    staff          0 Oct  9 15:49 file.3
0:owner@:execute:deny
1:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
2:group@:write_data/append_data/execute:deny
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

例11 UFS ファイル上の POSIX ドラフト ACL エントリを削除する

次の例では、ACL から lp エントリを削除しています。

```
$ ls -v file.2
-rw-r--r--+ 1 marks    staff          0 Oct  9 15:52 file.2
0:user::rw-
1:user:lp:-wx      #effective:---
2:group::r--      #effective:r--
3:mask:r--
4:other:r--

$ chmod A-user:lp:-wx file.2
$ ls -v file.2
-rw-r--r--  1 marks    staff          0 Oct  9 15:52 file.2
0:user::rw-
1:group::r--      #effective:r--
2:mask:r--
3:other:r--
```

例12 ZFS ファイル上の特定のACLエントリをインデックス番号を指定して削除する
次のようなACLがあるとします。

```
$ ls -v file
0:group:staff:read_data/write_data/execute/read_acl:allow
1:user:bin:read_data:deny
2:user:bin:read_data:allow
3:owner@:write_data/append_data:deny
4:owner@:read_data/write_xattr/execute/write_attributes/write_acl
  /write_owner:allow
5:group@:write_data/append_data:deny
6:group@:read_data/execute:allow
7:everyone@:write_data/append_data/write_xattr/write_attributes
  /write_acl/write_owner:deny
8:everyone@:read_data/read_xattr/execute/read_attributes/read_acl
  /synchronize:allow
```

bin に対する2つめのユーザーエントリを削除します。

```
$ chmod A2- file
$ ls -v file
0:group:staff:read_data/write_data/execute/read_acl:allow
1:user:bin:read_data:deny
2:owner@:write_data/append_data:deny
3:owner@:read_data/write_xattr/execute/write_attributes/write_acl
  /write_owner:allow
4:group@:write_data/append_data:deny
5:group@:read_data/execute:allow
6:everyone@:write_data/append_data/write_xattr/write_attributes
  /write_acl/write_owner:deny
7:everyone@:read_data/read_xattr/execute/read_attributes/read_acl
  /synchronize:allow
```

例13 UFS ファイル上の特定のPOSIXドラフトACLエントリを削除する
次の例では、次のACLからlpエントリをインデックス番号を指定して削除しています。

```
$ ls -v file.1
-rw-r--r--+ 1 root    root          0 Sep 29 16:10 file.1
0:user::rw-
1:user:lp:rw-          #effective:r--
2:group::r--          #effective:r--
3:mask:r--
4:other:r--

$ chmod A1- file.1
$ ls -v
-rw-r--r--+ 1 root    root          0 Sep 29 16:10 file.1
0:user::rw-
```


例13 UFS ファイル上の特定の POSIX ドラフト ACL エントリを削除する (続き)

```
1:group::r--          #effective:r--
2:mask:r--
3:other:r--
```

例14 ファイルから ACL をすべて削除する

次のコマンドは、NFSv4/ZFS ACL、POSIX ドラフト ACL のどちらでも動作します。

次のような ACL があるとします。

```
$ ls -v file.3
-rw-r--r--+ 1 marks    staff          0 Oct  9 15:49 file.3
0:user:lp:read_data/write_data:allow
1:user:marks:read_acl:allow
2:owner@:execute:deny
3:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
4:group@:write_data/append_data/execute:deny
5:group@:read_data:allow
6:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
7:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

既存の ACL は実質的に削除され、ファイルのアクセス権ビットを表現した ACL で置き換えられます。

```
$ chmod A- file.3
$ ls -v file.3
-rw-r--r-- 1 marks    staff          0 Oct  9 15:49 file.3
0:owner@:execute:deny
1:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
2:group@:write_data/append_data/execute:deny
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

例15 ZFS ファイル上の ACL エントリ全体を置き換える

ACL の全体を置き換えるには、次の chmod 構文を使用します。

```
$ chmod A=owner@:read_data/write_data:allow,group@:read_data/
  write_data:allow,user:lp:read_data:allow file.4
$ ls -v file.4
-rw-rw----+ 1 marks    staff          0 Oct  9 16:12 file.4
```

例 15 ZFS ファイル上の ACL エントリ全体を置き換える (続き)

```
0:owner@:read_data/write_data:allow
1:group@:read_data/write_data:allow
2:user:lp:read_data:allow
```

例 16 UFS ファイル上の POSIX ドラフト ACL 全体を置き換える

この操作のほうが多少複雑になっています。置換 ACL には、ファイルの所有者、グループ所有者、その他のユーザー、およびマスクを表現するために必要なエントリと、ほかに設定する必要のあるすべてのエントリを含める必要があります。

```
$ chmod A=user::rw-,group::rw-,other::---,mask:r--,
      user:lp:r-- file.3
$ ls -v file.3
-rw-r-----+ 1 root      root          0 Oct  9 16:14 file.3
0:user::rw-
1:user:lp:r--      #effective:r--
2:group::rw-      #effective:r--
3:mask:r--
4:other:---
```

例 17 ZFS ファイル上の特定のエントリを置き換える

次のような ACL があるとします。

```
$ ls -v file.5
-rw-r--r--+ 1 marks     staff          0 Oct  9 16:18 file.5
0:user:marks:read_data:allow
1:owner@:execute:deny
2:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
3:group@:write_data/append_data/execute:deny
4:group@:read_data:allow
5:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
6:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow
```

ここで、ユーザー marks のアクセスタイプを、許可から拒否に変更します。

```
$ chmod A0=user:marks:read_data:deny file.5
$ ls -v file.5
-rw-r--r--+ 1 marks     staff          0 Aug 23 09:11 file.5
0:user:marks:read_data:deny
1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
  /write_acl/write_owner:allow
2:group@:write_data/append_data/execute:deny
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
```

例17 ZFS ファイル上の特定のエントリを置き換える (続き)

```

/write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
:allow

```

例18 UFS ファイル上の特定の POSIX ドラフト ACL エントリを置き換える
次のような ACL があるとします。

```

$ ls -lv file.4
-rw-r--r--+ 1 marks    staff          0 Oct  9 16:21 file.4
0:user::rw-
1:user:lp:rwx      #effective:r--
2:group::r--      #effective:r--
3:mask:r--
4:other:r--

```

ここで、lp のアクセス権を、rwx から r-- に変更します。

```

$ chmod A1=user:lp:r-- file.4

$ ls -lv file
-rw-r--r--+ 1 marks    staff          0 Oct  9 16:21 file.4
0:user::rw-
1:user:lp:r--      #effective:r--
2:group::r--      #effective:r--
3:mask:r--
4:other:r--

```

例19 ZFS ファイル上で ACL 継承フラグを設定する

継承フラグを設定できるのは、ZFS ファイルの場合だけです。ACL をディレクトリに対して設定する場合には、いくつかの継承フラグがオプションで設定できます。

ユーザー lp に対するある ACL エントリを、ディレクトリ内で新たに作成されたファイルに継承する必要があると想定します。まず、継承可能な ACL エントリをディレクトリ上に作成する必要があります。

```

$ chmod A+user:lp:read_data:file_inherit:allow test.dir
$ ls -ldv test.dir
drwxr-xr-x+ 2 marks    staff          2 Aug 23 09:08 test.dir/
0:user:lp:read_data:file_inherit:allow
1:owner@::deny
2:owner@:list_directory/read_data/add_file/write_data/add_subdirectory
  /append_data/write_xattr/execute/write_attributes/write_acl
  /write_owner:allow
3:group@:add_file/write_data/add_subdirectory/append_data:deny
4:group@:list_directory/read_data/execute:allow
5:everyone@:add_file/write_data/add_subdirectory/append_data/write_xattr

```

例19 ZFS ファイル上で ACL 継承フラグを設定する (続き)

```

/write_attributes/write_acl/write_owner:deny
6:everyone@:list_directory/read_data/read_xattr/execute/read_attributes
/read_acl/synchronize:allow

```

この `lp` エントリは、ディレクトリ `test.dir` 内で新たに作成されたファイルに継承されます。

```

$ touch test.dir/file.test
$ ls -v test.dir/file.test
-rw-r--r--+ 1 marks      staff      0 Oct  9 16:29 test.dir/file.test
0:user:lp::deny
1:user:lp:read_data:allow
2:owner@:execute:deny
3:owner@:read_data/write_data/append_data/write_xattr/
  write_attributes/write_acl/write_owner:allow
4:group@:write_data/append_data/execute:deny
5:group@:read_data:allow
6:everyone@:write_data/append_data/write_xattr/execute/
  write_attributes/write_acl/write_owner:deny
7:everyone@:read_data/read_xattr/read_attributes/read_acl/
  synchronize:allow

```

ユーザー `lp` のエントリが、新たに作成されたファイルに継承されています。複数の継承フラグの組み合わせを指定することが可能です。たとえば、`lp` エントリがディレクトリにも継承されるようにするには、次のようなコマンドを使用します。

```

$ chmod A+user:lp:read_data:file_inherit/\
  dir_inherit:allow test.dir

```

例20 ZFS ファイルのシステム属性を置換する

次の例では、ZFS ファイルのシステム属性を置換します。

```

$ chmod S=v{archive,hidden,readonly,system,appendonly,\
  nonodump,immutable,noav_modified,noav_quarantined,\
  nounlink} file1

```

または

```

$ chmod S=c{AHRSaui} file1

```

または

```

$ chmod S=c{AHRSa-i--u} file1

```

または

```

$ chmod S=cAHRSaui file1

```

または

例 20 ZFS ファイルのシステム属性を置換する (続き)

```
$ chmod -@ '..' S=cAHRsaiu file1
```

適切な特権を前提として、file1 のシステム属性 archive、hidden、readonly、system、appendonly、immutable、および nounlink が設定されます。適切な特権を前提として、file1 のシステム属性 nodump、av_modified、および av_quarantined がクリアされます。

例 21 ZFS ファイルのすべてのシステム属性をクリアする

次の例では、ZFS ファイルのシステム属性をすべてクリアします。

```
$ chmod S-a file1
```

または

```
$ chmod -@ '..' S-a file1
```

適切な特権を前提として、file1 のブール型読み書きシステム属性がすべてクリアされます。

例 22 ZFS ファイルの名前付き属性ファイルのシステム属性を設定する

次の例では、ZFS ファイル自体ではなく、ZFS ファイルの名前付き属性ファイルのシステム属性を設定します。

```
$ chmod -@ myattr S+vhhidden file1
```

これにより、隠しシステム属性が、ファイル file1 自体ではなく、その名前付き属性ファイル myattr に設定されます。

例 23 ZFS ファイルのすべての名前付き属性ファイルのシステム属性を設定する

次の例では、ZFS ファイル自体ではなく、ZFS ファイルのすべての名前付き属性ファイルのシステム属性を設定します。

```
$ chmod -@ '*' S+a file1
```

例 24 ZFS ファイルのすべての名前付き属性ファイルのシステム属性を設定する

次の例では、ZFS ファイルのすべての名前付き属性ファイル、および ZFS ファイル自体のシステム属性を設定します。

```
$ chmod -@ '..' -@ '*' S+vhhidden file1
```

これにより、file1 のすべての名前付き属性ファイル、およびそのファイル自体に対して隠しシステム属性が設定されます。

例25 ディレクトリ階層を再帰的に検索する

次の例では、ディレクトリ階層を再帰的に検索して、すべての名前付き属性ファイル、ZFS ファイルオペランド、およびディレクトリ自体のシステム属性をすべて設定します。

```
$ chmod -R -@ '..' -@ '*' S+a directory1
```

これにより、`directory1` のディレクトリ階層内にあるすべての通常ファイルおよびディレクトリの名前付き属性ファイルすべて、および `directory1` 自体に対し、隠しシステム属性が設定されます。

例26 ZFS ファイルの hidden および system システム属性を設定する

次の例では、ZFS ファイルの `hidden` および `system` システム属性を設定します。

```
$ chmod S+cHS file1
```

または

```
$ chmod S+vhidden,+vsystem file1
```

または

```
$ chmod S+v{hidden,system} file1
```

または

```
$ chmod S+c{-HS-----} file1
```

または

```
$ chmod S-v{nohidden,nosystem} file1
```

または

```
$ chmod S-v{hidden,system},+v{hidden,system} file1
```

例27 ZFS ファイルのすべてのシステム属性をクリアする

次の例では、ZFS ファイルのシステム属性をすべてクリアします。

```
$ chmod S-a file1
```

または

```
$ chmod S=v{} file1
```

次の2つの例では、指定された最後の属性操作が優先されます。

この例では、置換属性名リスト (`{}`) により `file1` のシステム属性がすべてクリアされます。

```
$ chmod S+cHS,=v{} file1
```

例 27 ZFS ファイルのすべてのシステム属性をクリアする (続き)

この例では、クリア属性操作 (-a) により file1 のシステム属性がすべてクリアされます。

```
$ chmod S+vhhidden,+vsystem,-a file1
```

例 28 ファイルのすべてのブール型読み書きシステム属性の値を設定する

次の例では、あるファイルのすべてのブール型読み書きシステム属性の値を、別のファイルのブール型読み書きシステム属性と同じに設定します。

```
$ chmod S=v'ls -/v file1|sed -n '2s/.*/{p/' file2
```

適切な特権および file1 と file2 がサポートされる同じシステム属性を保持しているという前提で、file1 に設定されたすべてのシステム属性が file2 に対しても設定されます。クリアされた file1 のすべてのシステム属性が、file2 でもクリアされます。

環境変数

chmod の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

関連項目

[getfacl\(1\)](#), [ls\(1\)](#), [setfacl\(1\)](#), [chmod\(2\)](#), [fgetattr\(3C\)](#), [acl\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

『Oracle Solaris 11.1 の管理: ZFS ファイルシステム』

注意事項

ディレクトリのセットグループ ID ビットをそっくり置き換えることはできません。g+s または g-s を使用してください。

chmod は規定に反しないかぎり、無意味なモード作成でも許可します (たとえば、テキストファイルを実行可能にする)。chmod は強制ロックが意味をなすか否かを見るために、ファイルのタイプをチェックすることはありません。

ファイルシステムが *nosuid* オプションでマウントされている場合、*setuid* の実行は許可されません。

chmod を使用して ACL エントリを持つファイル上のファイルグループ所有者のアクセス権を変更すると、ファイルグループ所有者のアクセス権と ACL マスクの両方が、新しいアクセス権に変わります。新しい ACL マスクのアクセス権は、そのファイル上に ACL エントリを持つ追加ユーザーおよびグループの実効アクセス権を変更する場合があるので注意が必要です。[getfacl\(1\)](#) または [ls\(1\)](#) コマンドを使用して、すべての ACL エントリに適切なアクセス権が設定されていることを確認してください。

名前 chown – ファイルの所有者の変更

形式

```
/usr/bin/chown      /usr/bin/chown [-fhR] owner[:group] file...
                   /usr/bin/chown -s [-fhR] ownersid[:groupsid] file...
                   /usr/bin/chown -R [-f] [-H | -L | -P] owner[:group] file...
                   /usr/bin/chown -s -R [-f] [-H | -L | -P] ownersid[:groupsid] file...

/usr/xpg4/bin/chown /usr/xpg4/bin/chown [-fhR] owner[:group] file...
                   /usr/xpg4/bin/chown -s [-fhR] ownersid[:groupsid] file...
                   /usr/xpg4/bin/chown -R [-f] [-H | -L | -P] owner[:group] file...
                   /usr/xpg4/bin/chown -s -R [-f] [-H | -L | -P] ownersid[:groupsid] file...
```

機能説明

`/usr/bin/chown` および `/usr/xpg4/bin/chown` `chown` ユーティリティは、`file` オペランドが示す各ファイルのユーザー ID を、`owner` オペランドが示す値に変更し、さらに `group` が指定されていれば、グループ ID もその値に変更します。

`chown` がスーパーユーザー以外のユーザーによって起動された場合、セットユーザー ID ビットはクリアされます。

ファイルの所有者 (またはスーパーユーザー) だけが、ファイルの所有者を変更できます。

ファイルシステムでは、マウントポイントオプション `rstchown` を使用して所有者変更を制限します。このオプションが有効なときは、ファイルの所有者はそのファイルの所有者 ID を変更することができません。このオプションが有効かどうかにかかわらず、スーパーユーザーだけが所有者 ID を変更できます。

`chown` は、各ファイルの所有者を `owner` に変更します。`owner` は、ユーザー名または数値のユーザー ID で指定できます。各ファイルのグループ所有権も、ユーザー名に続けて `:group` を指定することで `group` に変更できます。

オプション

`/usr/bin/chown` および `/usr/xpg4/bin/chown` サポートしているオプションは、次のとおりです。

- f 強制。エラーは報告しません。
- h ファイルがシンボリックリンクであるとき、そのシンボリックリンクの所有者を変更します。このオプションが指定されていない場合は、そのシンボリックリンクによって参照されるファイルの所有者が変更されます。
- H コマンド行に指定したファイルがディレクトリタイプのファイルを参照するシンボリックリンクである場合、所有者の変更の対象となるのは、そのシン

ボリックリンクが参照するディレクトリと、その下にあるファイル階層内のすべてのファイルです。ファイル階層の検索中にシンボリックリンクに出会った場合、対象となるファイルの所有者は変更されますが、再帰は発生しません。

- L ファイルがシンボリックリンクである場合、所有者の変更の対象となるのは、そのシンボリックリンクが参照するファイルです。コマンド行に指定したファイル、あるいは、ファイル階層の検索中に会ったファイルがディレクトリタイプのファイルを参照するシンボリックリンクである場合、所有者の変更の対象となるのは、そのシンボリックリンクが参照するディレクトリ、および、そのディレクトリより下にあるファイル階層内のすべてのファイルです。
- P コマンド行に指定したファイル、あるいは、ファイル階層の検索中に会ったファイルがシンボリックリンクである場合、所有者の変更の対象となるのは、そのシンボリックリンク自身です。このオプションは、シンボリックリンクが参照するファイル階層は検索しません。
- s 所有者やグループの引数は、Windows SID 文字列です。このオプションでは、SID を格納可能なファイルシステム (ZFS など) が必要になります。

-H、-L、または -P は相互排他的なオプションですが、これらを複数指定しても、エラーであるとは判断されません。最後に指定したオプションが `chown` の動作を決定します。

/usr/bin/chown

サポートしているオプションは、次のとおりです。

- R 再帰。chown はディレクトリおよびすべてのサブディレクトリを検索し、指定された所有者 ID を設定していきます。シンボリックリンクに出会うと、(-h または -P オプションが指定されていなければ) 対象となるファイルの所有者が変更されます。しかし、-H または -L オプションを指定しない限り、再帰は発生しません。

/usr/xpg4/bin/chown

サポートしているオプションは、次のとおりです。

- R 再帰。chown はディレクトリおよびすべてのサブディレクトリを検索し、指定された所有者 ID を設定していきます。シンボリックリンクに出会うと、(-h または -P オプションが指定されていなければ) 対象となるファイルの所有者が変更されます。-H、-L、または -P のいずれのオプションも指定しない限り、-L オプションがデフォルトのモードとして使用されます。

オペランド

次のオペランドがサポートされています。

`owner[:group]` `file` に割り当てるユーザー ID とグループ ID (グループ ID は省略可能) を指定します。owner 部分は、ユーザーデータベースから得られるユーザー名、または数値のユーザー ID のどちらかでなければなりません。どちらの場合でも、`file` で指定した各ファイルに与

えるユーザー ID を表します。owner が数値で、それがユーザー名としてユーザーデータベースに存在していると、そのユーザー名に対応したユーザー ID 番号がユーザー ID として用いられます。同様に、group 部分を指定する場合は、グループデータベースから得られるグループ名、または数値のグループ ID のどちらかでなければなりません。どちらの場合でも、各ファイルに与えるグループ ID を表します。group が数値で、それがグループ名としてグループデータベースに存在していると、そのグループ名に対応したグループ ID 番号がグループ ID として用いられます。

file ユーザー ID を変更するファイルのパス名

使用法 ファイルが 2G バイト (2³¹ バイト) 以上ある場合の chown の動作については、[largefile\(5\)](#) を参照してください。

使用例 例1 階層内のすべてのファイルの所有者を変更
次のコマンドを実行すると、階層に含まれるすべてのファイルの所有者が変更されます。このとき、シンボリックリンクも変更されますが、リンクの参照先の所有者は変更されません。

example% **chown -R -h owner[:group] file...**

環境変数 chown の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 ユーティリティーの実行が正常終了し、要求されたすべての変更が行われた。
- >0 エラーが発生した。

ファイル /etc/passwd システムパスワードファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/chown

属性タイプ	属性値
使用条件	system/core-os
CSI	有効。「注意事項」を参照してください。
インタフェースの安定性	確実
標準	standards(5) を参照してください。

/usr/xpg4/bin/chown

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効。「注意事項」を参照してください。
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [chgrp\(1\)](#), [chmod\(1\)](#), [chown\(2\)](#), [fpathconf\(2\)](#), [passwd\(4\)](#), [system\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 [chown](#) は *owner* 名および *group* 名を除いて CSI に対応しています。

名前	chown – change owner
形式	/usr/ucb/chown [-fR] owner[.group] filename...
機能説明	chown changes the owner of the <i>filenames</i> to <i>owner</i>. The owner can be either a decimal user ID (UID) or a login name found in the password file. An optional <i>group</i> can also be specified. The group can be either a decimal group ID (GID) or a group name found in the GID file. In the default case, only the super-user of the machine where the file is physically located can change the owner. The system configuration option { _POSIX_CHOWN_RESTRICTED } and the privileges PRIV_FILE_CHOWN and PRIV_FILE_CHOWN_SELF also affect who can change the ownership of a file. See chown(2) and privileges(5).
オプション	The following options are supported: -f Do not report errors. -R Recursively descend into directories setting the ownership of all files in each directory encountered. When symbolic links are encountered, their ownership is changed, but they are not traversed.
使用法	See largefile(5) for the description of the behavior of chown when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
ファイル	/etc/passwd Password file
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目	chgrp(1) , chown(2) , group(4) , passwd(4) , attributes(5) , largefile(5) , privileges(5)

名前	ckdate, errdate, helpdate, valdate – 日付の入力要求とその検証
形式	<pre>ckdate [-Q] [-W width] [-f format] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] /usr/sadm/bin/errdate [-W width] [-e error] [-f format] /usr/sadm/bin/helpdate [-W width] [-h help] [-f format] /usr/sadm/bin/valdate [-f format] input</pre>
機能説明	ckdate ユーティリティーは、ユーザーに入力を要求するプロンプトを出力し、ユーザーの応答を検証します。このユーティリティーでは、ユーザーに日付の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値(ユーザーがRETURNキーで応答した場合に返される値)をオプションにより定義できます。ユーザーの応答は、date コマンドで定義されている書式に一致したものでなければなりません。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用されるスペース(復帰改行を含む)はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。 ckdate コマンドには、3つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errdate(エラーメッセージを書式化して表示する)、helpdate(ヘルプメッセージを書式化して表示する)、およびvaldate(応答を検証する)です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。errdate および helpdate の各モジュールに format が定義されている場合、メッセージには、指定した書式も含まれます。
オプション	サポートしているオプションは、次のとおりです。 -d default <i>default</i> をデフォルト値として定義します。デフォルト値は、どのような書式も満たす必要はありません。 -e error <i>error</i> をエラーメッセージとして定義します。 -f format 入力を検証するときの基準となる書式を指定します。次の書式と定義を指定できます。 %b = 省略された月の名前 (jan, feb, mar) %B = 完全な月の名前 %d = 日 (01 - 31) %D = %m/%d/%y の形式の日付 (デフォルトの書式) %e = 日 (1 - 31、1桁の数字の前にはブランクが挿入されます)

- `%h` = 省略形の月の名前。 `%b` と同じ
- `%m` = 月 (01 - 12)
- `%y` = 西暦中の年 (たとえば、89)
- `%Y` = CCYY の形式の年 (たとえば、1989)
- `-h help` `help` をヘルプメッセージとして定義します。
- `-k pid` ユーザーが中断を選択した場合、プロセス ID `pid` のプロセスにシグナルを送信するようにします。
- `-p prompt` `prompt` をプロンプトメッセージとして定義します。
- `-Q` 有効な応答として終了 (quit) を使用できないようにします。
- `-s signal` 終了が選択された場合、`-k` オプションで定義されたプロセス ID `pid` のプロセスに、シグナル `signal` を送信するようにします。シグナルを指定しないと、`SIGTERM` を送信します。
- `-W width` プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、`width` の行長に書式化します。

オペランド 次のオペランドを指定できます。

`input` 書式基準と照合される入力

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、`-w` オプションで負の行長が指定された、または使用法に誤りがあった。
- 3 ユーザー終了 (quit)。
- 4 無効な書式引数。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [attributes\(5\)](#)

注意事項 `ckdate` のデフォルトのプロンプトは、次のとおりです。

Enter the date [?,q]:

デフォルトのエラーメッセージは、次のとおりです。

ERROR - Please enter a date. Format is <format>.

デフォルトのヘルプメッセージは、次のとおりです。

Please enter a date. Format is <format>.

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に q が返されます。valdate モジュールは、出力を生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	ckgid, errgid, helpgid, valgid – グループ ID の入力要求とその検証
形式	<pre>ckgid [-Q] [-W width] [-m] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] /usr/sadm/bin/errgid [-W width] [-e error] /usr/sadm/bin/helpgid [-W width] [-m] [-h help] /usr/sadm/bin/valgid input</pre>
機能説明	ckgid は、ユーザーに入力を要求するプロンプトを出力し、ユーザーの応答を検証します。このユーティリティでは、ユーザーに既存のグループ ID の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーがキャリッジリターンで応答した場合に返される値) をオプションにより定義できます。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用するスペース (復帰改行を含む) はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。 ckgid コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errgid(エラーメッセージを書式化して表示する)、helpgid(ヘルプメッセージを書式化して表示する)、および valgid(応答を検証する)です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。
オプション	サポートしているオプションは、次のとおりです。 -d <i>default</i> <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e <i>error</i> <i>error</i> をエラーメッセージとして定義します。 -h <i>help</i> <i>help</i> をヘルプメッセージとして定義します。 -k <i>pid</i> ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -m ユーザーがヘルプを要求した場合、あるいはユーザーの入力が不正な場合は、グループ名の一覧を表示します。 -p <i>prompt</i> <i>prompt</i> をプロンプトメッセージとして定義します。 -Q 有効な応答として終了 (quit) を使用できないようにします。

- s *signal*

終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- W *width*

プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

- オペランド

次のオペランドを指定できます。
input /etc/group と照合される入力
- 終了ステータス

次の終了ステータスが返されます。

0 正常終了。
1 入力で EOF が検出された、-w オプションで負の行長が指定された、または使用法に誤りがあった。
3 ユーザー終了 (quit)。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

- 関連項目

[attributes\(5\)](#)
- 注意事項

ckgid のデフォルトのプロンプトは、次のとおりです。

Enter the name of an existing group [?,q]:

デフォルトのエラーメッセージは、次のとおりです。

ERROR: Please enter one of the following group names: [List]

ckgid の -m オプションを使用すると、有効なグループのリストがここに表示されます。

デフォルトのヘルプメッセージは、次のとおりです。

ERROR: Please enter one of the following group names: [List]

ckgid の -m オプションを使用すると、有効なグループのリストがここに表示されます。

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に q が返されます。valgid モジュールは、出力を生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	ckint, errint, helpint, valint – 整数値の入力要求とその検証
形式	<pre>ckint [-Q] [-W width] [-b base] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] /usr/sadm/bin/errint [-W width] [-b base] [-e error] /usr/sadm/bin/helpint [-W width] [-b base] [-h help] /usr/sadm/bin/valint [-b base] input</pre>
機能説明	ckint ユーティリティーは、ユーザーに入力を要求するプロンプトを出力し、ユーザーの応答を検証します。このユーティリティーでは、ユーザーに整数の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値(ユーザーがキャリッジリターンで応答した場合に返される値)をオプションにより定義できます。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用されるスペース(復帰改行を含む)はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。 ckint コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errint(エラーメッセージを書式化して表示する)、helpint(ヘルプメッセージを書式化して表示する)、および valint(応答を検証する)です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。errint と helpint の各モジュールに <i>base</i> が定義されている場合、メッセージには入力する整数の基数が含まれます。
オプション	サポートしているオプションは、次のとおりです。 -b base 入力する整数の基数を定義します。2 から 36 までの数字でなければなりません。デフォルト値は 10 です。 -d default <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e error <i>error</i> をエラーメッセージとして定義します。 -h help <i>help</i> をヘルプメッセージとして定義します。 -k pid ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -p prompt <i>prompt</i> をプロンプトメッセージとして定義します。 -Q 有効な応答として終了 (quit) を使用できないようにします。

- s *signal*

終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- W *width*

プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

オペランド 次のオペランドを指定できます。

input 基数 *base* の基準と照合される入力

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、-w オプションで負の行長が指定された、または使用法に誤りがあった。
- 3 ユーザー終了 (quit)。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [attributes\(5\)](#)

注意事項 ckint のデフォルトのプロンプトは、次のとおりです (基数は 10)。

Enter an integer [?,q]:

デフォルトのエラーメッセージは、次のとおりです (基数は 10)。

ERROR - Please enter an integer.

デフォルトのヘルプメッセージは、次のとおりです (基数は 10)。

Please enter an integer.

基数を 10 以外の数に設定した場合は、前述のメッセージは "integer" から "base *base* integer" に変更されます。

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に *q* が返されます。valint モジュールは、出力を生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	ckitem, erritem, helpitem – メニューの作成、およびメニュー項目の入力要求とその検証
形式	<pre>ckitem [-Q] [-W width] [-uno] [-f filename] [-l label] [-i invis] [,]... [-m max] [-d default] [-h help] [-e error] [-p prompt] [-k pid] [-s signal] [choice ...]] /usr/sadm/bin/erritem [-W width] [-e error] [choice ...] /usr/sadm/bin/helpitem [-W width] [-h help] [choice ...]</pre>
機能説明	ckitem ユーティリティは、メニューを作成し、ユーザーにプロンプトを出力してメニュー項目から1つの項目を選択するように促します。さらに、ユーザーの応答入力を検証します。このコマンドのオプションでは、プロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値(ユーザーがキャリッジリターンで応答した場合に返される値)を定義します。 デフォルトでは、メニューは書式化されており、各項目の前に番号が付けられ、端末上に複数の列で出力されます。列の長さは、選択した最大長の項目によって決まります。項目はアルファベット順に表示されます。 メッセージの長さはすべて最大70文字に制限され、自動的に書式化されます。定義で使用するスペース(復帰改行を含む)はすべて削除されます。<code>-w</code> オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。 ckitem コマンドには、2つのビジュアルツールモジュールがリンクされています。これらのモジュールは、<code>erritem</code>(エラーメッセージを書式化して表示する)と<code>helpitem</code>(ヘルプメッセージを書式化して表示する)です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。これらのモジュールに <i>choice</i> が定義されている場合、メッセージには利用可能なメニューの選択項目が含まれます。
オプション	サポートしているオプションは、次のとおりです。 <code>-d default</code> <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 <code>-e error</code> <i>error</i> をエラーメッセージとして定義します。 <code>-f filename</code> <i>filename</i> を表示されるメニュー項目のリストを含むファイルとして定義します。(このファイルの書式は、<code>token<tab>description</code> です。ポンド記号(#)で始まる行は注釈として指定され、無視されます。) <code>-h help</code> <i>help</i> をヘルプメッセージとして定義します。

<code>-i invis</code>	非表示メニュー項目(メニューに表示されない選択項目)を定義します。(たとえば、非表示項目として使用される"all"は、有効なオプションですが、メニューには表示されません。任意の数の非表示項目を定義できます。)非表示項目があることを、プロンプトメッセージまたはヘルプメッセージのどちらかで、ユーザーに知らせるようにします。
<code>-k pid</code>	ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。
<code>-l label</code>	メニューの上に表示するラベル、 <i>label</i> を定義します。
<code>-m max</code>	ユーザーが選択できるメニュー項目の最大数を定義します。デフォルトは1です。
<code>-n</code>	メニュー項目をアルファベット順で表示しないようにします。
<code>-o</code>	1つのメニュートークンだけが返されるようにします。
<code>-p prompt</code>	<i>prompt</i> をプロンプトメッセージとして定義します。
<code>-Q</code>	有効な応答として終了(quit)を使用できないようにします。
<code>-s signal</code>	終了が選択された場合、 <code>-k</code> オプションで定義されたプロセス ID <i>pid</i> のプロセスに、シグナル <i>signal</i> を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
<code>-u</code>	メニュー項目に番号を付けずに表示するようにします。
<code>-W width</code>	プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、 <i>width</i> の行長に書式化します。

オペランド 次のオペランドを指定できます。

choice メニュー項目を定義します。項目はスペースか復帰改行で区切ります。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、`-w` オプションで負の行長が指定された、`-f` オプションでファイルが開けない、あるいは使用法に誤りがあった。
- 3 ユーザー終了(quit)。
- 4 選択すべき項目がありません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[attributes\(5\)](#)

注意事項

ユーザーは、メニュー項目に番号が付いている場合はその項目の番号を、あるいは、その項目を一意に識別するのに必要な長さの文字列を入力できます。長いメニューはページに分割され、各ページには10個の項目が表示されます。

メニュー項目が、`-f` オプションで指定したファイルとコマンド行の両方に定義されている場合、メニュー項目は通常、アルファベット順に表示されます。ただし、アルファベット順での表示を抑制する `-n` オプションが使用されている場合は、ファイルに定義された項目が最初に表示され、次にコマンド行に定義されたオプションが表示されます。

ckitem のデフォルトのプロンプトは次のとおりです。

```
Enter selection [?,??,q]:
```

1つの疑問符はヘルプメッセージを表示してから、プロンプトを再表示します。2つの疑問符は、ヘルプメッセージを表示してから、メニューラベル、メニュー、およびプロンプトを再表示します。

番号を入力した場合のデフォルトのエラーメッセージは、次のとおりです。

```
ERROR: Bad numeric choice specification
```

文字列を入力した場合のデフォルトのエラーメッセージは、次のとおりです。

```
ERROR: Entry does not match available menu selection. Enter the number
of the menu item you wish to select, the token which is associated
with the menu item, or a partial string which uniquely identifies the
token for the menu item. Enter ?? to reprint the menu.
```

デフォルトのヘルプメッセージは、次のとおりです。

```
Enter the number of the menu item you wish to select, the token
which is associated with the menu item, or a partial string which
uniquely identifies the token for the menu item. Enter ? to
reprint the menu.
```

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード3と共にqが返されます。

名前	ckkeywd – キーワードの入力要求とその検証
形式	ckkeywd [-Q] [-W width] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] keyword [...]
機能説明	ckkeywd は、ユーザーに入力を要求するプロンプトを出力し、ユーザーの応答を検証します。このユーティリティでは、ユーザーにキーワードリストのいずれかの入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーがキャリッジリターンで応答した場合に返される値) をオプションにより定義できます。このコマンドから返される応答は、定義済みキーワードリストのいずれかと一致しなければなりません。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用されるスペース (復帰改行を含む) はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。
オプション	サポートしているオプションは、次のとおりです。 -d <i>default</i> <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e <i>error</i> <i>error</i> をエラーメッセージとして定義します。 -h <i>help</i> <i>help</i> をヘルプメッセージとして定義します。 -k <i>pid</i> ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -p <i>prompt</i> <i>prompt</i> をプロンプトメッセージとして定義します。 -Q 有効な応答として終了 (quit) を使用できないようにします。 -s <i>signal</i> 終了が選択された場合、-k オプションで定義されたプロセス ID <i>pid</i> のプロセスに、シグナル <i>signal</i> を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。 -W <i>width</i> プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、<i>width</i> の行長に書式化します。
オペランド	次のオペランドを指定できます。 <i>keyword</i> 応答を照合するときの基準となるキーワード、またはキーワードのリストを定義します。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、-w オプションで負の行長が指定された、選択すべきキーワードがない、あるいは、使用法に誤りがあった。
- 3 ユーザー終了 (quit)。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [attributes\(5\)](#)

注意事項 ckkeyword のデフォルトのプロンプトは、次のとおりです。

Enter appropriate value [*keyword*,[. . .],?,q]:

デフォルトのエラーメッセージは、次のとおりです。

ERROR: Please enter one of the following keywords: *keyword*,[. . .],q

デフォルトのヘルプメッセージは、次のとおりです。

keyword,[. . .],q

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に q が返されます。

名前

ckpath, errpath, helppath, valpath – パス名の入力要求とその検証

形式

```
ckpath [-Q] [-W width] [-a | l] [-b | c | f | y] [-n [o | z]] [-rtwx] [-d default] [-h help] [-e error]
/usr/sadm/bin/errpath [-W width] [-a | l] [-b | c | f | y] [-n [o | z]] [-rtwx] [-e error]
/usr/sadm/bin/helppath [-W width] [-a | l] [-b | c | f | y] [-n [o | z]] [-rtwx] [-h help]
/usr/sadm/bin/valpath [-a | l] [-b | c | f | y] [-n [o | z]] [-rtwx] input
```

機能説明

ckpath ユーティリティは、ユーザーに入力を要求するプロンプトを出力し、ユーザーの応答を検証します。このユーティリティでは、ユーザーにパス名の入力を促すプロンプトメッセージ、ヘルプメッセージおよびエラーメッセージ、およびデフォルト値(ユーザーがRETURN キーで応答した場合に返される値)をオプションにより定義できます。

パス名は、最初のオプショングループで指定した基準に従わなければなりません。基準が定義されていない場合、パス名はまだ存在していない通常のファイルに対するものでなければなりません。-a(絶対)と-l(相対)のいずれも指定されていない場合は、どちらかが有効であるとみなされます。

メッセージの長さはすべて最大 79 文字に制限され、自動的に書式化されます。メッセージ定義内で1つのスペース文字の後にあるタブと復帰改行は削除されますが、スペースは削除されません。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。

プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「使用例」の項を参照)が表示されます。

ckpath コマンドには、3つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errpath(標準出力上にエラーメッセージを書式化して表示する)、helppath(標準出力上にヘルプメッセージを書式化して表示する)、およびvalpath(応答を検証する)です。

オプション

サポートしているオプションは、次のとおりです。

- a パス名は絶対パスでなければなりません。
- b パス名はブロック型特殊ファイルでなければなりません。
- c パス名は文字型特殊ファイルでなければなりません。
- d default default をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。
- e error error をエラーメッセージとして定義します。
- f パス名は通常ファイルでなければなりません。
- h help help をヘルプメッセージとして定義します。

- k *pid* ユーザーが中断を選択した場合、プロセス ID *pid* のプロセスにシグナルを送信するようにします。
- l パス名は相対パスでなければなりません。
- n パス名が存在してはなりません (新規のものでなければなりません)。
- o パス名が存在していなければなりません (既存のものでなければなりません)。
- p *prompt* *prompt* をプロンプトメッセージとして定義します。
- Q 有効な応答として終了 (quit) を使用できないようにします。
- r パス名は読み取り可能でなければなりません。
- s *signal* 終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- t パス名は作成可能 (処理可能) でなければなりません。パス名が、まだ存在していない場合には作成されます。
- w パス名は書き込み可能でなければなりません。
- W *width* プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。
- x パス名は実行可能でなければなりません。
- y パス名はディレクトリでなければなりません。
- z パス名には、0 バイトを超えるサイズのファイルがなければなりません。

オペランド 次のオペランドを指定できます。

input 検証オプションと照合される入力

使用例 ckpath のデフォルトメッセージのテキストは、使用されている基準オプションによって異なります。

例1 デフォルトのプロンプト

ckpath (-a オプションを使用) のデフォルトプロンプトの例は、次のとおりです。

```
example% ckpath -a
Enter an absolute pathname [?,q]
```

例2 デフォルトのエラーメッセージ

デフォルトのエラーメッセージ (-a オプションを使用) の例は、次のとおりです。

例2 デフォルトのエラーメッセージ (続き)

```
example% /usr/sadm/bin/errpath -a
ERROR: A pathname is a filename, optionally preceded by parent
       directories.
       The pathname you enter: - must begin with a slash (/)
```

例3 デフォルトのヘルプメッセージ

デフォルトのヘルプメッセージ (-a オプションを使用) の例は、次のとおりです。

```
example% /usr/sadm/bin/helppath -a
A pathname is a filename, optionally preceded by parent directories.
The pathname you enter: - must begin with a slash (/)
```

例4 終了オプション

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に q が返されます。終了を入力すると、後に復帰改行が続きます。

例5 valpath モジュールの使用

valpath モジュールは、標準エラー出力に使用法に関するメッセージを出力します。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

```
example% /usr/sadm/bin/valpath
usage: valpath [-a|l][b|c|f|y][n|[o|z]]rtwx] input
.
.
.
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、-w オプションで負の行長が指定された、または使用法に誤りがあった。
- 2 同時に指定できないオプション。
- 3 ユーザー終了 (quit)。
- 4 同時に指定できないオプション。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[signal.h\(3HEAD\)](#), [attributes\(5\)](#)

名前	ckrange, errrange, helprange, valrange – 整数の入力要求とその検証
形式	<pre>ckrange [-Q] [-W width] [-l lower] [-u upper] [-b base] [-d default] [-h help] [-e error] [-p prompt] [/usr/sadm/bin/errrange [-W width] [-e error] [-l lower] [-u upper] [-b base] /usr/sadm/bin/helprange [-W width] [-h help] [-l lower] [-u upper] [-b base] /usr/sadm/bin/valrange [-l lower] [-u upper] [-b base] input</pre>
機能説明	ckrange ユーティリティーは、ユーザーに指定範囲内の整数の入力を要求して、ユーザーの応答が有効かどうかを検証します。このユーティリティーでは、ユーザーに指定範囲内の整数の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーが RETURN キーで応答した場合に返される値) をオプションにより定義します。 また、このコマンドは、有効な入力範囲も定義します。下限または上限のいずれかが定義されていない場合、範囲は一方の限界だけに制限されます。 メッセージの長さはすべて最大 79 文字に制限され、自動的に書式化されます。メッセージ定義内で 1 つのスペース文字の後にあるタブと復帰改行は削除されますが、スペースは削除されません。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「使用例」の項を参照)が表示されます。 ckrange コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errrange (標準出力上にエラーメッセージを書式化して表示する)、helprange (標準出力上にヘルプメッセージを書式化して表示する)、および valrange (応答を検証する) です。 注: input 引数として負の値を指定すると、valrange 内の getopt で混乱が生じます。引数の前に - を付けると、getopt は処理を停止します。getopt のパラメータ処理については、getopt(1) および Intro(1) を参照してください。getopt は、位置指定パラメータを解析して、有効なオプションかどうかを検査するために使用されます。
オプション	サポートしているオプションは、次のとおりです。 -b base 入力する整数の基数を定義します。2 から 36 までの数字でなければなりません。デフォルト値は 10 です。基数を変換するには strtoul(3C) を使用します。出力は常に、基数を 10 として実行されます。 -d default default をデフォルト値として定義します。default は strtoul(3C) を使用して指定した基数に変換できます。指定した基数に無効な文字があると、strtoul はエラーを示すことなく変換を終了します。 -e error error をエラーメッセージとして定義します。

- h *help* *help* をヘルプメッセージとして定義します。
- k *pid* ユーザーが中断を選択した場合、プロセス ID *pid* のプロセスにシグナルを送信するようにします。
- l *lower* *lower* を範囲の下限として定義します。デフォルト値は、マシンの負の最大整数です。
- p *prompt* *prompt* をプロンプトメッセージとして定義します。
- Q 有効な応答として終了 (quit) を使用できないようにします。
- s *signal* 終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- u *upper* *upper* を範囲の上限として定義します。デフォルト値は、マシンの正の最大整数です。
- W *width* プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

オペランド

次のオペランドを指定できます。

input 上限、下限、および基数と照合される入力

使用例

例1 デフォルトのプロンプト

ckrange のデフォルトのプロンプトは、次のとおりです (基数は 10)

```
example% ckrange
Enter an integer between lower_bound and
upper_bound [lower_bound-upper_bound,?,q]:
```

例2 デフォルトのエラーメッセージ

デフォルトのエラーメッセージは、次のとおりです (基数は 10)。

```
example% /usr/sadm/bin/errrange
ERROR: Please enter an integer between lower_bound \
and upper_bound.
```

例3 デフォルトのヘルプメッセージ

デフォルトのヘルプメッセージは、次のとおりです (基数は 10)。

```
example% /usr/sadm/bin/helprange
Please enter an integer between lower_bound and upper_bound.
```

例4 基数を 10 以外の数に変更した場合のメッセージ

基数を 10 以外の数に設定した場合、メッセージは "integer" から "base base integer" に変更されます。たとえば、次のように指定します。

例4 基数を10以外の数に変更した場合のメッセージ (続き)

```
example% /usr/sadm/bin/helprange -b 36
```

例5 終了(quit)オプションの使用

終了オプションを選択した場合(かつ使用できる場合)は、リターンコード3と共にqが返されます。終了を入力すると、後に復帰改行が続きます。

例6 valrangeモジュールの使用

valrangeモジュールは、標準エラー出力に使用法に関するメッセージを作成します。正常終了した場合は0、失敗した場合には0以外の値を返します。

```
example% /usr/sadm/bin/valrange
usage: valrange [-l lower] [-u upper] [-b base] input
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力でEOFが検出された、-wオプションで負の行長が指定された、または使用法に誤りがあった。
- 2 使用方法のエラー。
- 3 ユーザー終了(quit)。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [Intro\(1\)](#), [getopt\(1\)](#), [strtol\(3C\)](#), [attributes\(5\)](#), [signal.h\(3HEAD\)](#)

名前	ckstr, errstr, helpstr, valstr – 文字列の入力要求とその検証
形式	<pre>ckstr [-Q] [-W width] [[-r regexp] [...]] [-l length] [-d default] [-h help] [-e error] [-p prompt] /usr/sadm/bin/errstr [-W width] [-e error] [-l length] [[-r regexp] [...]] /usr/sadm/bin/helpstr [-W width] [-h help] [-l length] [[-r regexp] [...]] /usr/sadm/bin/valstr [-l length] [[-r regexp] [...]] input</pre>
機能説明	ckstr ユーティリティーは、ユーザーに入力を要求してその応答を検証します。このユーティリティーでは、文字列の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値(ユーザーが RETURN キーで応答した場合に返される値)をオプションにより定義できます。 このコマンドから返される応答は、定義済みの正規表現に一致していなければならず、指定された長さ以内でなければなりません。正規表現が指定されていない場合、有効な入力、内部にスペースを含まず、また先行や後続するスペースがない、指定された長さ以内の文字列でなければなりません。長さが指定されていない場合、長さは検査されません。 メッセージの長さはすべて最大 79 文字に制限され、自動的に書式化されます。メッセージ定義内で 1 つのスペース文字の後にあるタブと復帰改行は削除されますが、スペースは削除されません。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「使用例」の項を参照)が表示されます。 ckstr コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errstr(標準出力上にエラーメッセージを書式化して表示する)、helpstr(標準出力上にヘルプメッセージを書式化して表示する)、および valstr(応答を検証する)です。
オプション	サポートしているオプションは、次のとおりです。 -d default <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e error <i>error</i> をエラーメッセージとして定義します。 -h help <i>help</i> をヘルプメッセージとして定義します。 -k pid ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -l length 入力の最大長を指定します。 -p prompt <i>prompt</i> をプロンプトメッセージとして定義します。 -Q 有効な応答として終了(quit)を使用できないようにします。

- r *regexp*** 入力を検証するときの基準となる正規表現、*regexp* を指定します。スペースを含めることができます。複数の式が定義されている場合、応答はその内のいずれかに一致しなければなりません。
- s *signal*** 終了が選択された場合、**-k** オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、**SIGTERM** を送信します。
- W *width*** プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

オペランド 次のオペランドを指定できます。

input 書式の長さや正規表現の基準に対して検証される入力

使用例

例1 デフォルトのプロンプト

ckstr のデフォルトのプロンプトは、次のとおりです。

```
example% ckstr
Enter an appropriate value [?,q]:
```

例2 デフォルトのエラーメッセージ

デフォルトのエラーメッセージは、適用される妥当性検査のタイプによって異なります。ユーザーには、長さまたはパターン照合のいずれかが失敗したことが通知されます。デフォルトのエラーメッセージは、次のとおりです。

```
example% /usr/sadm/bin/errstr
ERROR: Please enter a string which contains no embedded,
leading or trailing spaces or tabs.
```

例3 デフォルトのヘルプメッセージ

デフォルトのヘルプメッセージも、適用される妥当性検査のタイプによって異なります。正規表現が定義されている場合、メッセージは次のようになります。

```
example% /usr/sadm/bin/helpstr -r regexp
Please enter a string which matches the following pattern:
regexp
```

他のメッセージは、文字列の長さの要件と定義を指定します。

例4 終了(quit)オプションの使用

終了オプションを選択した場合(かつ使用できる場合)は、リターンコード 3 と共に **q** が返されます。終了を入力すると、後に復帰改行が続きます。

例5 valstrモジュールの使用

valstrモジュールは、標準エラー出力に使用法に関するメッセージを出力します。正常終了した場合は0、失敗した場合には0以外の値を返します。

```
example% /usr/sadm/bin/valstr
usage: valstr [-l length] [[-r regexp] [ . . . ]] input
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力でEOFが検出された、-wオプションで負の行長が指定された、または使用法に誤りがあった。
- 2 無効な正規表現。
- 3 ユーザー終了 (quit)。

属性 属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [signal.h\(3HEAD\)](#), [attributes\(5\)](#)

名前	cksum – ファイルのチェックサムとサイズの出力
形式	/usr/bin/cksum [<i>file</i> ...]
機能説明	cksum コマンドは、各入力ファイルの循環冗長性検査 (CRC) を算出し、その結果を各ファイルのオクテット数と共に標準出力に書き込みます。 正常に処理された各ファイルについて cksum 方式は次の書式でチェックサム情報を出力します。 <pre>"%u %d %s\n" <checksum>, <# of octets>, <path name></pre> file オペランドが指定されなかった場合、パス名と先行する空白は省略されます。 使用される CRC は、参照されるイーサネット標準における CRC エラーチェックで使用される多項式に基づいています。 CRC チェックサムのコード化は、生成多項式によって定義されます。 $G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$ 指定したファイルに対応する CRC 値は、数学的には次の手順で定義されます。 1. 評価される n ビットは、2 を法とする $n-1$ 次多項式 $M(x)$ の係数と見なされます。これらの n ビットは、ファイルからのビットであり、最上位ビットはファイルの先頭のオクテットの最上位ビットであり、最後のビットは最後のオクテットの最下位ビットです。これらの n ビットには、オクテットの数が整数になるように、(必要であれば) 0 のビットが埋め込まれ、そのあとにファイルの長さを表す 1 つまたは複数のオクテットが、最下位オクテットを先頭にして 2 進値で続きます。この整数を表すことができる最小のオクテット数が使用されます。 2. $M(x)$ の値は、x^{32} で乗算されて (つまり、32 ビット左にシフト)、2 を法とした除算を使用して $G(x)$ で除算されます。結果として、次数 ≤ 31 の除余 $R(x)$ が生成されます。 3. $R(x)$ の係数は、32 ビットのシーケンスとみなされます。 4. ビットシーケンスの補数がとられ、その結果が CRC になります。 オペランド 次のオペランドを指定できます。 <i>file</i> 検査するファイルのパス名。 <i>file</i> オペランドを指定しないと、標準入力を使用されます。 使用法 cksum コマンドは通常、疑いのあるファイルを同じファイルの信頼できるバージョンと簡単に比較して、ノイズの多いメディアを介して送信されたファイルが正しく受信できたかどうかを確認するのに使用します。ただし、この比較は、暗号化した場合ほど安全ではありません。損傷を受けたファイルが、元のファイルと同じ CRC を生成する確率は天文学的低い値です。ただし、意図的に同じ CRC を生成することは困難ではありますが、不可能ではありません。

cksum の入力ファイルはどんなタイプでもかまいませんが、文字型特殊デバイスファイルでは期待通りの結果が得られないことがあります。本書では、入力時に使用されるブロックサイズを規定していないため、文字型特殊ファイルのチェックサムでは、これらのファイル内のデータがすべて処理されるとはかぎりません。

アルゴリズムは、オクテット単位に分割されたビットストリームで表現されます。ファイルが2つのシステム間で送信されるときに、何らかのデータ変換(8ビット文字を9ビットバイトに移行したり、リトルエンディアンのバイト順序をビッグエンディアンに変更するなど)が行われる場合、同じCRC値を得ることは期待できません。このような変換を実行する実装では、cksum を拡張して、そのような状況に対処します。

2G バイト (2^{31} バイト) 以上のファイルを検出した場合の cksum の動作については、[largefile\(5\)](#) を参照してください。

環境変数 cksum の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 ファイルはすべて正しく処理されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [digest\(1\)](#), [sum\(1\)](#), [bart\(1M\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前 cktime, errtime, helptime, valtime – 時間の入力要求とその検証

形式 cktime [-Q] [-W *width*] [-f *format*] [-d *default*] [-h *help*] [-e *error*] [-p *prompt*] [-k *pid*] [-s *signal*]
 /usr/sadm/bin/errtime [-W *width*] [-e *error*] [-f *format*]
 /usr/sadm/bin/helptime [-W *width*] [-h *help*] [-f *format*]
 /usr/sadm/bin/valtime [-f *format*] *input*

機能説明 cktime ユーティリティは、ユーザーに入力を要求してその応答を検証します。このユーティリティでは、ユーザーに時間の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーが RETURN キーで応答した場合に返される値) をオプションにより定義できます。ユーザー応答は、時間について定義されている書式に一致しなければなりません。

メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用するスペース (復帰改行を含む) はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。

プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ(「注意事項」の項を参照)が表示されます。

cktime コマンドには、3つのビジュアルツールモジュールがリンクされています。これらのモジュールは、errtime (エラーメッセージを書式化して表示する) と helptime (ヘルプメッセージを書式化して表示する) と、valtime (応答を検証する) です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。errtime および helptime の各モジュールに format が定義されている場合、メッセージには指定した書式も含まれます。

オプション サポートしているオプションは、次のとおりです。

-d *default* *default* をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。

-e *error* *error* をエラーメッセージとして定義します。

-f *format* 入力を検証するときの基準となる書式を指定します。次の書式と定義を指定できます。

```
%H = hour (00 - 23)
%I = hour (00 - 12)
%M = minute (00 - 59)
%p = ante meridian or post meridian
%r = time as %I:%M:%S %p
%R = time as %H:%M (the default format)
%S = seconds (00 - 59)
%T = time as %H:%M:%S
```

- h *help* *help* をヘルプメッセージとして定義します。
- k *pid* ユーザーが中断を選択した場合、プロセス ID *pid* のプロセスにシグナルを送信するようにします。
- p *prompt* *prompt* をプロンプトメッセージとして定義します。
- Q 有効な応答として終了 (quit) を使用できないようにします。
- s *signal* 終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- W *width* プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

オペランド 次のオペランドを指定できます。

input 書式基準と照合される入力

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 入力で EOF が検出された、-w オプションで負の行長が指定された、あるいは使用法に誤りがあった。
- 3 ユーザー終了 (quit)。
- 4 無効な書式引数。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [attributes\(5\)](#)

注意事項 cktime のデフォルトのプロンプトは、次のとおりです。

Enter a time of day [?,q]:

デフォルトのエラーメッセージは、次のとおりです。

ERROR: Please enter the time of day. Format is <format>.

デフォルトのヘルプメッセージは、次のとおりです。

Please enter the time of day. Format is <format>.

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に `q` が返されます。`valtime` モジュールは、出力を生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	ckuid, erruid, helpuid, valuid – ユーザー ID の入力要求とその検証
形式	<pre>ckuid [-Q] [-W width] [-m] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] /usr/sadm/bin/erruid [-W width] [-e error] /usr/sadm/bin/helpuid [-W width] [-m] [-h help] /usr/sadm/bin/valuid input</pre>
機能説明	ckuid ユーティリティーは、ユーザーに入力を要求してその応答を検証します。このユーティリティーでは、ユーザーに既存のユーザー ID の入力を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーが RETURN キーで応答した場合に返される値) をオプションにより定義できます。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用するスペース (復帰改行を含む) はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ (「注意事項」の項を参照) が表示されます。 ckuid コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、erruid (エラーメッセージを書式化して表示する) と helpuid (ヘルプメッセージを書式化して表示する) と、valuid (応答を検証する) です。これらのモジュールは、FML オブジェクトと組み合わせて使用する必要があります。この場合、FML オブジェクトはプロンプトを定義します。
オプション	サポートしているオプションは、次のとおりです。 -d <i>default</i> <i>default</i> をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e <i>error</i> <i>error</i> をエラーメッセージとして定義します。 -h <i>help</i> <i>help</i> をヘルプメッセージとして定義します。 -k <i>pid</i> ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -m ヘルプが要求された場合、またはユーザーがエラーを犯した場合は、すべてのログインのリストを表示します。 -p <i>prompt</i> <i>prompt</i> をプロンプトメッセージとして定義します。 -Q 有効な応答として終了 (quit) を使用できないようにします。

- s *signal*

終了が選択された場合、-k オプションで定義されたプロセス ID *pid* のプロセスに、シグナル *signal* を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。
- w *width*

プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、*width* の行長に書式化します。

オペランド

次のオペランドを指定できます。
input /etc/passwd と照合される入力

- 終了ステータス
- 次の終了ステータスが返されます。
- 0

正常終了。
- 1

入力で EOF が検出された、-w オプションで負の行長が指定された、または使用法に誤りがあった。
- 2

使用方法のエラー。
- 3

ユーザー終了 (quit)。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[attributes\(5\)](#)

注意事項

ckuid のデフォルトのプロンプトは、次のとおりです。
Enter the login name of an existing user [?,q]:

デフォルトのエラーメッセージは、次のとおりです。
ERROR - Please enter the login name of an existing user.

-m オプションを使用した場合のデフォルトのエラーメッセージは、次のとおりです。
ERROR: Please enter one of the following login names: <List>

デフォルトのヘルプメッセージは、次のとおりです。
Please enter the login name of an existing user.

-m オプションを使用した場合のデフォルトのヘルプメッセージは、次のとおりです。
Please enter one of the following login names: <List>

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に `q` が返されます。`valuid` モジュールは出力を何も生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	ckyorn, erryorn, helpyorn, valyorn – 「はい」 / 「いいえ」 の入力要求とその検証
形式	ckyorn [-Q] [-W width] [-d default] [-h help] [-e error] [-p prompt] [-k pid [-s signal]] /usr/sadm/bin/erryorn [-W width] [-e error] /usr/sadm/bin/helpyorn [-W width] [-h help] /usr/sadm/bin/valyorn input
機能説明	ckyorn は、ユーザーに入力を要求してその応答を検証します。このユーティリティーでは、「はい (yes)」または「いいえ (no)」の応答を促すプロンプトメッセージ、ヘルプメッセージとエラーメッセージ、およびデフォルト値 (ユーザーが RETURN キーで応答した場合に返される値) をオプションにより定義します。 メッセージの長さはすべて最大 70 文字に制限され、自動的に書式化されます。定義で使用されるスペース (復帰改行を含む) はすべて削除されます。-w オプションは、自動書式化を取り消します。メッセージ定義の最初または最後にチルド文字がある場合は、そこにデフォルトテキストが挿入されて、指定したテキストとデフォルトテキストの両方を表示することができます。 プロンプトメッセージ、ヘルプメッセージ、またはエラーメッセージが定義されていない場合は、デフォルトメッセージ (「注意事項」の項を参照) が表示されます。 ckyorn コマンドには、3 つのビジュアルツールモジュールがリンクされています。これらのモジュールは、erryorn (エラーメッセージを書式化して表示する) と helpyorn (ヘルプメッセージを書式化して表示する) と、valyorn (応答を検証する) です。
オプション	サポートしているオプションは、次のとおりです。 -d default default をデフォルト値として定義します。デフォルト値は検証されないため、どのような基準も満たす必要はありません。 -e error error をエラーメッセージとして定義します。 -h help help をヘルプメッセージとして定義します。 -k pid ユーザーが中断を選択した場合、プロセス ID <i>pid</i> のプロセスにシグナルを送信するようにします。 -p prompt prompt をプロンプトメッセージとして定義します。 -Q 有効な応答として終了 (quit) を使用できないようにします。 -s signal 終了が選択された場合、-k オプションで定義されたプロセス ID <i>pid</i> のプロセスに、シグナル <i>signal</i> を送信するようにします。シグナルを指定しないと、SIGTERM を送信します。 -W width プロンプトメッセージ、ヘルプメッセージ、およびエラーメッセージを、<i>width</i> の行長に書式化します。

- オペランド

次のオペランドを指定できます。

input y、yes、または n、no (大文字小文字の任意の組み合わせ) に対して検証される入力
- 終了ステータス

次の終了ステータスが返されます。

0 正常終了。

1 入力で EOF が検出された、-w オプションで負の行長が指定された、または使用法に誤りがあった。

2 使用方法のエラー。

3 ユーザー終了 (quit)。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

- 関連項目

[attributes\(5\)](#)
- 注意事項

ckyorn のデフォルトのプロンプトは、次のとおりです。

Yes or No [y,n,?,q]:

デフォルトのエラーメッセージは、次のとおりです。

ERROR - Please enter yes or no.

デフォルトのヘルプメッセージは、次のとおりです。

To respond in the affirmative, enter y, yes, Y, or YES.

To respond in the negative, enter n, no, N, or NO.

終了オプションを選択した場合 (かつ使用できる場合) は、リターンコード 3 と共に q が返されます。valyorn モジュールは、出力を生成しません。正常終了した場合は 0、失敗した場合には 0 以外の値を返します。

名前	clear – 端末画面の消去
形式	clear [<i>term</i>]
機能説明	clear は、可能であれば端末画面を消去します。 <i>term</i> オペランドで端末タイプが指定されていない場合は、環境内で調べ、そのあとで terminfo データベースを調べ、画面を消去する方法を決定します。
オペランド	<i>term</i> 端末タイプを指定します。通常は、環境変数 TERM からデフォルト値を使用するので必要ありません。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [tput\(1\)](#), [attributes\(5\)](#)

名前	cmp - 2つのファイルの比較
形式	<code>/usr/bin/cmp [-l -s] file1 file2 [skip1] [skip2]</code>
機能説明	cmp は、2つのファイル <i>file1</i> と <i>file2</i> を比較します。2つのファイルの内容が同じである場合、cmp は何も出力しません。デフォルトでは、ファイルが異なる場合、最初の差異が発生するバイトおよび行が標準出力に書き込まれます。バイト数と行数の初めをそれぞれ 1 とします。 <i>skip1</i> と <i>skip2</i> はそれぞれ <i>file1</i> と <i>file2</i> への初期バイトオフセットであり、8 進数と 10 進数のいずれかになります。先行する 0 は 8 進数を意味します。 <i>file1</i> と <i>files2</i> のどちらかが - の場合、cmp はそのオペランドに標準入力を使用します。
オプション	サポートしているオプションは、次のとおりです。 -l 各差異部分について、10 進数のバイト位置と差異部分のバイト長 (8 進数) を書き出します。 -s 差異のあるファイルについて何も書き出しません。ゼロ以外の終了ステータスのみを返します。
オペランド	次のオペランドがサポートされています。 <i>file1</i> 比較する 1 つ目のファイルのパス名。 <i>file1</i> として - を指定すると、標準入力とみなされます。 <i>file2</i> 比較する 2 つ目のファイルのパス名。 <i>file2</i> として - を指定すると、標準入力とみなされます。 <i>file1</i> と <i>file2</i> がどちらも標準入力を指す場合、または同じ先入れ先出し型特殊ファイル、ブロック型特殊ファイル、あるいは文字型特殊ファイルを指す場合、エラーで終了します。
使用法	ファイルが 2 ギガバイト (2^{31} バイト) 以上ある場合の cmp の動作については、 largefile(5) を参照してください。
使用例	例1 ファイルをバイトごとに比較する 次の例では、<i>file1</i> と <i>file2</i> をバイトごとに比較します。 <pre>example% cmp file1 file2 0 1024</pre> 比較を行う前に <i>file2</i> を 1024 バイト分とばしています。
環境変数	cmp の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

- 終了ステータス
- 次のエラー値が返されます。
- 0

2つのファイルは同一です。
- 1

2つのファイルは異なっていました。1つのファイルがもう一方のファイルの先頭部分と同一の場合も、このコードが返されます。
- >1

エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[comm\(1\)](#), [diff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	col - 逆方向改行フィルタ												
形式	col [-bfp _x]												
機能説明	col ユーティリティーは、標準入力を読んで標準出力に書き出します。col は、逆方向改行コードおよび順方向半改行と逆方向半改行 が示す行の重ね打ちを実行します。-x オプションが指定されない限り、入力中の空白文字を可能であればタブ文字に変換します。col は nroff(1) の .rt コマンドによる複数カラム出力、および tbl(1) プリプロセッサを使用すると得られる出力にフィルタをかけるときに便利です。 col は ASCII 制御文字の SO および SI を、代替文字セットの開始および終了とみなします。各入力文字が属する文字セットは記憶されていて、出力上では各文字が正しい文字セットで印刷されるように必要に応じて SI と SO が生成されます。 入力データ中で認識される制御文字は、スペース文字、バックスペース、タブ、改行文字、復帰改行文字、SI、SO、VT、逆方向改行、順方向半改行、そして逆方向半改行です。VT 文字は、逆方向全改行の代わりに使用できます。これは、このプログラムの初期タイプと互換性を保つためです。これら以外に出力上にコピーされる文字は、すべて印刷可能文字です。 前述の制御機能や改行文字の ASCII コードを以下の表に示します。ESC は ASCII のエスケープ文字を表します。8 進数コードでは 033 となります。ESC- という表示は、ESC の後に文字 x を伴った合計 2 文字の並びを示します。 <table> <tr> <td>逆方向改行</td><td>ESC-7</td></tr> <tr> <td>逆方向半改行</td><td>ESC-8</td></tr> <tr> <td>順方向半改行</td><td>ESC-9</td></tr> <tr> <td>仮想タブ (VT)</td><td>013</td></tr> <tr> <td>テキスト開始 (SO)</td><td>016</td></tr> <tr> <td>テキスト終了 (SI)</td><td>017</td></tr> </table>	逆方向改行	ESC-7	逆方向半改行	ESC-8	順方向半改行	ESC-9	仮想タブ (VT)	013	テキスト開始 (SO)	016	テキスト終了 (SI)	017
逆方向改行	ESC-7												
逆方向半改行	ESC-8												
順方向半改行	ESC-9												
仮想タブ (VT)	013												
テキスト開始 (SO)	016												
テキスト終了 (SI)	017												
オプション	-b col 使用中の出力装置がバックスペースに対応できないとみなします。この場合、同じ箇所に 2 文字以上が現われたときは最後に読んだ 1 文字が出力されます。 -f col は、入力時には 1/2 行単位の移動を認めますが、出力時には通常認めません。その代わり、行間に現われたテキストは次の全改行の位置に移動します。この処理は、-f (fine) オプションによって抑制できます。この場合、col からの出力には順方向半改行 (ESC-9) が含まれる可能性があります。逆方向改行または逆方向半改行が含まれることは決してありません。 -p 通常 col は、入力時に見つかった認識できないエスケープシーケンスをすべて無視します。-p オプションは、逆方向改行による重ね打ちが発生することを前提として、col がこれらのシーケンスを通常の文字として出力できるよ												

うにします。そのエスケープシーケンスのテキスト上の位置を完全に把握していないかぎり、このオプションは使用しないでください。

- x 出力上において空白文字を可能であればタブ文字に変換する、という col の動作を抑止します。タブ位置は、8 カラムごとに設定されていると見なされます。

環境変数 col の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次のエラー値が返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目 [nroff\(1\)](#), [tbl\(1\)](#), [ascii\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 col が使用する入力フォーマットは nroff に -T37 または -Tlp オプションを付けた場合の出力と一致します。col が最終的に出力する装置が半改行を認識できないときは -T37 オプション (および col の -f オプション) を、認識できるときは -Tlp をそれぞれ使用してください。

col は、128 行以上の行の逆戻りおよび 1 行当たり 800 字以上の処理はできません。

その文書の先頭行を越えて逆戻りする改行動作は無視されます。結果として、先頭行にスーパースクリプトを使用しないでください。

名前	comm - 2つのファイルに共通な行または共通でない行の表示
形式	<code>/usr/bin/comm [-options] file1 file2</code>
機能説明	comm は、現在のロケールの照合順序に従ってソートされているはずの2つのファイル <i>file1</i> と <i>file2</i> を読み込み、出力として3つのテキスト列を生成します。 1 <i>file1</i> の行のみ。 2 <i>file2</i> の行のみ。 3 両方のファイルの行。 いずれかのファイルの行が現在のロケールの照合順序に従ってソートされていない場合、結果は不確定です。 <i>file1</i> または <i>file2</i> が - の場合、comm は標準入力を使用します。開始位置は現在の位置です。
オプション	サポートしているオプションは、次のとおりです。 -1 <i>file1</i> だけに存在している行の出力を抑制します。 -2 <i>file2</i> だけに存在している行の出力を抑制します。 -3 <i>file1</i> と <i>file2</i> の両方に存在している行の出力を抑制します。
オペランド	次のオペランドがサポートされています。 <i>file1</i> 比較する1つ目のファイルのパス名。 <i>file1</i> として - を指定すると、標準入力とみなされます。 <i>file2</i> 比較する2つ目のファイルのパス名。 <i>file2</i> として - を指定すると、標準入力とみなされます。
使用法	ファイルが2G バイト (2³¹ バイト) 以上ある場合の comm の動作については、largefile(5) を参照してください。
使用例	例1 ファイルによって指定された、ユーティリティのリストを出力する <i>file1</i>、<i>file2</i>、<i>file3</i> の3つのファイルに、ユーティリティのリストが正しくソートされて書かれている場合、 <pre>example% comm -23 file1 file2 comm -23 - file3</pre> このコマンドは、<i>file1</i> だけに含まれているユーティリティだけを出力します。次のエントリを見てください。 <pre>example% comm -12 file1 file2 comm -12 - file3</pre> このエントリは、3つのファイルすべてに含まれているユーティリティを出力します。また、次のエントリを見てください。

例1 ファイルによって指定された、ユーティリティーのリストを出力する (続き)

```
example% comm -12 file2 file3 | comm -23 -file1
```

このエントリは、*file2* と *file3* には含まれているが *file1* には含まれていないユーティリティーを出力します。

環境変数 `comm` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 入力ファイルはすべて、(指定されたとおりに) 正常に出力されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [cmp\(1\)](#), [diff\(1\)](#), [sort\(1\)](#), [uniq\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	command – 単純コマンドの実行
形式	
/usr/bin/command	<pre>command [-p] command_name [argument]...</pre> <pre>command [-v -V] command_name</pre>
ksh	<pre>command [-pvxV] [command_name [argument...]]</pre>
機能説明	command ユーティリティーを使用すると、シェルは関数の検索を実行せずに、指定された引数を単純コマンドとして扱います。 <i>command_name</i> が特殊組み込みユーティリティーのいずれかの名前と同じ場合、特殊プロパティは発生しません。その他の点では、<i>command_name</i> が関数の名前でない場合、通常、command (オプションなし) の効果は、command を省略した場合と同じになります。 command ユーティリティーは、シェルによるコマンド名の解釈方法についての情報も提供します。-v および -V を参照してください。
ksh	-v や -V オプションを指定しない場合、command は <i>argument</i> で指定された引数を使って <i>command_name</i> を実行します。通常行われるシェル関数の検索は抑制されます。また、command が特殊な組み込みコマンドの場合、特殊プロパティを削除することで、失敗してもそれを実行するスクリプトが終了されないようにします。 -v または -V オプションが指定されている場合、command は whence(1) と同等です。
オプション	/usr/bin/command では次のオプションを使用できます。 -p すべての標準ユーティリティーの検索を可能にする PATH のデフォルト値を使用して、コマンド検索を実行します。 -v 現在のシェル実行環境で <i>command_name</i> を呼び出すためにシェルが使用するパスまたはコマンドを示す文字列を標準出力に書き込みますが、<i>command_name</i> を呼び出しません。 ■ ユーティリティー、正規の組み込みユーティリティー、スラッシュ文字を含む <i>command_names</i>、および PATH 変数を使用して検出される、実装で提供されている関数は、絶対パス名として書き込まれます。 ■ シェル関数、特殊組み込みユーティリティー、PATH 検索に関連しない正規の組み込みユーティリティー、およびシェル予約語は、単純に名前として書き込まれます。 ■ 別名は、別名定義を表すコマンド行として書き込まれます。 ■ それ以外の場合、出力には何も書き込まれず、終了ステータスに名前が検出されなかったことが示されます。 -V 現在のシェル実行環境で、<i>command_name</i> オペランドで指定された名前をシェルが解釈する方法を示す文字列を、標準出力に書き込みます

が、*command_name* を呼び出しません。この文字列の書式は規定されていませんが、出力文字列は指定した *command_name* が属するカテゴリを示し、次の情報を含みます。

- ユーティリティー、正規の組み込みユーティリティー、および *PATH* 変数を使用して検出される、実装で提供されている関数では、そのように識別され、文字列に絶対パス名が含まれています。
- これ以外のシェル関数は関数として識別されます。
- 別名は別名として識別され、その定義は文字列に含まれています。
- 特殊組み込みユーティリティーは、特殊組み込みユーティリティーとして識別されます。
- *PATH* 検索に関連しない正規の組み込みユーティリティーは、正規の組み込みユーティリティーとして識別されます。
- シェル予約語は、予約語として識別されます。

ksh の *command* では次のオプションがサポートされています。

- p *PATH* の値で定義されているパスではなく、デフォルトパスを検索します。
- v 次と同等です。
whence *command* [*argument* ...]
- V 次と同等です。
whence -v *command* [*argument* ...]
- x 引数が多すぎるためにコマンドが失敗した場合は、コマンドが複数回呼び出され、呼び出しごとに引数のサブセットが使用されます。複数の引数に展開する最初のワードの前の引数と、複数の引数に展開する最後のワードのあとの引数は、各呼び出しに渡されます。終了ステータスは呼び出しの最大終了ステータスです。

オペランド 次のオペランドがサポートされています。

argument *command_name* への引数として扱われる文字列の 1 つ。
command_name ユーティリティーまたは特殊組み込みユーティリティーの名前。

使用例

例 1 新しい作業ディレクトリを常に出力する *cd* を作成する

次の例では、常に新しい作業ディレクトリを一度だけ出力する *cd* を作成します。

```
cd() {
    command cd "$@" >/dev/null
    pwd
}
```

例2 スクリプトが親によるなりすましを回避する「安全なシェルスクリプト」の開始
 次の例では、スクリプトが親によるなりすましを回避する「安全なシェルスクリプト」を開始しています。

```
IFS='
# The preceding value should be <space><tab><newline>.
# Set IFS to its default value.
\unalias -a
# Unset all possible aliases.
# Note that unalias is escaped to prevent an alias
# being used for unalias.
unset -f command
# Ensure command is not a user function.
PATH="$(command -p getconf _CS_PATH):$PATH"
# Put on a reliable PATH prefix.
# ...
```

この時点で、PATHで呼び出されるディレクトリに適切なアクセス権が付与されていれば、スクリプトは、スクリプトが呼び出すユーティリティーはすべて目的のものであることを保証できます。実装拡張機能が存在でき、それによって呼び出し時にユーザー関数が存在できることが前提となっているため、慎重を期しています。この機能は本書では規定されていませんが、拡張機能としては禁止されていません。たとえば、ENV変数は、ユーザー起動スクリプトを使用するスクリプトの呼び出しに先行します。このようなスクリプトでは、アプリケーションになりすます関数を定義している場合があります。

環境変数

commandの実行に影響を与える次の環境変数については、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

PATH コマンド検索中に使用される検索パスを決定します。-p オプションが指定されている場合は除きます。

終了ステータス

/usr/bin/command -v または -V オプションを指定すると、次の終了値が返されます。

0 正常終了。
 >0 *command_name* が検出できなかったか、エラーが発生しました。

それ以外の場合は、次の終了値が返されます。

126 *command_name* で指定したユーティリティーは見つかりましたが、呼び出せませんでした。

127 *command* ユーティリティーでエラーが発生したか、あるいは *command_name* で指定したユーティリティーが見つかりませんでした。

それ以外の場合、`command` の終了ステータスは、`command_name` 引数で指定した単純コマンドの終了ステータスになります。

`ksh` *command* が呼び出されると、`command` の終了ステータスが *command* の終了ステータスになります。それ以外の場合は、次のいずれかになります。

- 0 *command_name* が正常に完了しました。
- >0 `-v` または `-V` が指定され、エラーが発生しました。
- 126 *command_name* は見つかったが、呼び出すことができませんでした。
- 127 *command_name* は見つかりませんでした。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/command

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

ksh

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

関連項目 [ksh\(1\)](#), [sh\(1\)](#), [type\(1\)](#), [whence\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 compress, uncompress, zcat – compress, uncompress files or display expanded files

形式 compress [-fv/] [-b *bits*] [*file*]...
 compress -c [-fv] [-b *bits*] [*file*]
 uncompress [-fv] [-c | -/] [*file*]...
 zcat [*file*]...

機能説明

compress The compress utility attempts to reduce the size of the named files by using adaptive Lempel-Ziv coding. Except when the output is to the standard output, each file is replaced by one with the extension .Z, while keeping the same ownership modes, change times and modification times, ACLs, and extended attributes. The compress utility also attempt to set the owner and group of *file.z* to the owner and group of *file*, but does not fail if this cannot be done. If appending the .Z to the file pathname would make the pathname exceed 1023 bytes, the command fails. If no files are specified, the standard input is compressed to the standard output.

The amount of compression obtained depends on the size of the input, the number of *bits* per code, and the distribution of common substrings. Typically, text such as source code or English is reduced by 50–60%. Compression is generally much better than that achieved by Huffman coding (as used in [pack\(1\)](#)) and it takes less time to compute. The *bits* parameter specified during compression is encoded within the compressed file, along with a magic number to ensure that neither decompression of random data nor recompression of compressed data is subsequently allowed.

uncompress The uncompress utility restores files to their original state after they have been compressed using the compress utility. If no files are specified, the standard input is uncompressed to the standard output.

This utility supports the uncompressing of any files produced by compress. For files produced by compress on other systems, uncompress supports 9- to 16-bit compression (see -b).

zcat The zcat utility writes to standard output the uncompressed form of files that have been compressed using compress. It is the equivalent of uncompress -c. Input files are not affected.

オプション The following options are supported:

- b *bits* Sets the upper limit (in bits) for common substring codes. *bits* must be between 9 and 16 (16 is the default). Lowering the number of bits result in larger, less compressed files.
- c Writes to the standard output; no files are changed and no .Z files are created. The behavior of zcat is identical to that of 'uncompress -c'.
- f When compressing, forces compression of *file*, even if it does not actually reduce the size of the file, or if the corresponding *file.Z* file already exists.

If the `-f` option is not specified, and the process is not running in the background, prompts to verify whether an existing file should be overwritten. If the response is affirmative, the existing file is overwritten. When uncompressing, does not prompt for overwriting files. If the `-f` option is not specified, and the process is not running in the background, prompts to verify whether an existing file should be overwritten. If the standard input is not a terminal and `-f` is not specified, writes a diagnostic message to standard error and exits with a status greater than 0.

- v Verbose. Writes to standard error messages concerning the percentage reduction or expansion of each file.
- / When compressing or decompressing, copies any extended system attributes associated with the source file to the target file and copies any extended system attributes associated with extended attributes of the source file to the corresponding extended attributes associated with the target file. If any extended system attributes cannot be copied, the original file is retained, a diagnostic is written to `stderr`, and the final exit status is non-zero.

オペランド

The following operand is supported:

- file* A path name of a file to be compressed by `compress`, uncompressed by `uncompress`, or whose uncompressed form is written to standard out by `zcat`. If *file* is `-`, or if no *file* is specified, the standard input is used.

使用法

See [largefile\(5\)](#) for the description of the behavior of `compress`, `uncompress`, and `zcat` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `compress`, `uncompress`, and `zcat`: `LANG`, `LC_ALL`, `LC_COLLATE`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

Affirmative responses are processed using the extended regular expression defined for the `yesexpr` keyword in the `LC_MESSAGES` category of the user's locale. The locale specified in the `LC_COLLATE` category defines the behavior of ranges, equivalence classes, and multi-character collating elements used in the expression defined for `yesexpr`. The locale specified in `LC_CTYPE` determines the locale for interpretation of sequences of bytes of text data as characters, the behavior of character classes used in the expression defined for the `yesexpr`. See [locale\(5\)](#).

終了ステータス

The following error values are returned:

- 0 Successful completion.
- 1 An error occurred.
- 2 One or more files were not compressed because they would have increased in size (and the `-f` option was not specified).

>2 An error occurred.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
CSI	Enabled
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[ln\(1\)](#), [pack\(1\)](#), [fgetattr\(3C\)](#), [fsetattr\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [locale\(5\)](#), [standards\(5\)](#)

診断

Usage: `compress [-fv/] [-b bits] [file ...]`

`compress c [-fv] [-b bits] [file ...]`

Invalid options were specified on the command line.

Usage: `uncompress [-fv] [-c | -/] [file] ...`

Invalid options were specified on the command line.

Missing `maxbits`

`Maxbits` must follow `-b`, or invalid `maxbits`, not a numeric value.

file: not in compressed format

The file specified to `uncompress` has not been compressed.

file: compressed with *xxbits*, can only handle *yybits*

file was compressed by a program that could deal with more *bits* than the `compress` code on this machine. Recompress the file with smaller *bits*.

file: already has `.Z` suffix -- no change

The file is assumed to be already compressed. Rename the file and try again.

file: already exists; do you wish to overwrite (y or n)?

Respond `y` if you want the output file to be replaced; `n` if not.

`uncompress`: corrupt input

A SIGSEGV violation was detected, which usually means that the input file is corrupted.

Compression:*xx.xx*%

Percentage of the input saved by compression. (Relevant only for `-v`.)

-- not a regular file: unchanged

When the input file is not a regular file, (such as a directory), it is left unaltered.

-- has *xx* other links: unchanged

The input file has links; it is left unchanged. See [ln\(1\)](#) for more information.

– – file unchanged

No savings are achieved by compression. The input remains uncompressed.

– –filename too long to tack on .Z

The path name is too long to append the .Z suffix.

– –cannot preserve extended attributes. file unchanged

Extended system attributes could not be copied.

注意事項

Although compressed files are compatible between machines with large memory, -b 12 should be used for file transfer to architectures with a small process data space (64KB or less).

compress should be more flexible about the existence of the .Z suffix.

名前 cp – ファイルのコピー

形式

```

/usr/bin/cp [-fip@/] source_file target_file
/usr/bin/cp [-fip@/] source_file... target
/usr/bin/cp -r | -R [-H | -L | -P] [-fip@/] source_dir... target
/usr/bin/cp -R | -R [-H | -L | -P] [-fip@/] source_dir... target
/usr/xpg4/bin/cp [-fip@/] source_file target_file
/usr/xpg4/bin/cp [-fip@/] source_file... target
/usr/xpg4/bin/cp -r | -R [-H | -L | -P] [-fip@/] source_dir... target
/usr/xpg4/bin/cp -R | -R [-H | -L | -P] [-fip@/] source_dir... target

```

機能説明

前述の「形式」の項で示す第1の形式では、*source_file* および *target_file* はディレクトリファイルではありません。また、両者は同じ名前ではありません。cp ユーティリティーは、*source_file* が示すファイルの内容を、*target_file* が示す宛先パスにコピーします。*target_file* がすでに存在していると、cp はその内容を上書きしますが、ファイルのモード(さらに ACL が有効な場合は ACL)、所有者、およびグループは変更しません。*target_file* の最新更新時間および *source_file* の最新アクセス時間の値は、コピーが行われた時間に設定されます。*target_file* が存在しなければ、cp は *target_file* という名のファイルを作成します。このファイルのモードは *source_file* と同じになります。ただしコマンドを発行したユーザーがスーパーユーザーでない限り、スティッキビットは設定されません。この場合、*target_file* の所有者とグループは(*target_file* を作成した)ユーザーのものと同じになります。ただし、新たに作成されたファイル(*target_file*)を含むディレクトリに setgid ビットがセットされている場合を除きます。このディレクトリに setgid ビットがセットされている場合、新たに作成されたファイル(*target_file*)のグループは、そのファイルを作成したユーザーのグループではなく、そのファイルを含むディレクトリのグループになります。*target_file* が別のファイルへのリンクである場合、リンク先のファイルに対して *source_file* の内容を上書きします。*target_file* からのリンクはそのまま残ります。

第2の形式の cp は、*source_file* が示す1つ以上のファイルを、*target* が示すディレクトリにコピーします。*source_file* がディレクトリの場合、*target* が存在しない場合、および *target* がディレクトリでない場合は、いずれもエラーとなります。

第3または第4の形式の cp は、*source_dir* が示す1つ以上のディレクトリを、*target* が示すディレクトリにコピーします。*-r* または *-R* のどちらかを指定しなければなりません。個々の *source_dir* に関して、cp はその中のすべてのファイルとサブディレクトリをコピーします。

オプション /usr/bin/cp と /usr/xpg4/bin/cp では次のオプションがサポートされています。

-f リンク解除。宛先ファイルのファイル記述子が得られない場合、宛先ファイルのリンクを解除して処理を続けようとします。

- H 処理は、*source_file* オペランドとして指定したシンボリックリンクが参照するファイルの種類と内容に基づいて行われます。

source_file オペランドがシンボリックリンクである場合、*source_file* オペランドのシンボリックリンクが参照するファイルがコピーされます。ファイル階層の検索中に検出された他のすべてのシンボリックリンクは維持されます。

- i 対話型。cp では、コピーが既存の *target* を上書きする際に、確認を求めます。これは、入力が端末から送られているかどうかに関係なく実行されます。確認のプロンプトでエラーが発生した場合は、ユーザーによる否定応答と同等です。肯定応答を返すと、コピー処理は続行されます。他の応答では、cp による *target* の上書きは実行されません。
- L 処理は、*source_file* オペランドとして指定したシンボリックリンク、あるいは、ファイル階層の検索中に会ったシンボリックリンクが参照するファイルの種類と内容に基づいて行われます。

シンボリックリンクが参照するファイルをコピーします。ファイル階層の検索中に検出されたシンボリックリンクは維持されません。

- p 保持。cp ユーティリティーは、*source_file* の内容を複製するだけではなく、その ACL、アクセス日時と更新日時、拡張属性、拡張システム属性、ファイルモード、および所有者 ID とグループ ID を保持しようとします。

cp がアクセス日時と更新日時、拡張属性、またはファイルモードを保持できない場合、cp はそれを失敗とは見なしません。cp が所有者 ID とグループ ID を保持できない場合、コピーは失敗しませんが、cp はメッセージを表示せずにターゲットのファイルモードから *S_ISUID* および *S_ISGID* ビットをクリアします。cp がこれらのビットをクリアできない場合、コピーは失敗します。cp が ACL または拡張システム属性を保持できない場合、コピーは失敗します。コピーが失敗すると、診断メッセージが *stderr* に書き込まれ、(残りのオペランドすべての処理後に) cp がゼロ以外の終了ステータスで終了します。

- P 処理は、*source_file* オペランドとして指定したシンボリックリンク、あるいは、ファイル階層の検索中に会ったシンボリックリンク自身に行われます。

シンボリックリンクをコピーします。ファイル階層の検索中に検出されたシンボリックリンクが保存されます。

- r 再帰。cp はそのディレクトリ全体、つまりディレクトリ内の全ファイルをコピーするだけでなく、ディレクトリにあるすべてのサブディレクトリとそのサブディレクトリ中の全ファイルも *target* にコピーします。-H、-L、または -P のいずれのオプションも指定しない限り、-L オプションがデフォルトのモードとして使用されます。

- R パイプが読まれるのではなく複写される、という点を除いて -r と同じです。
- @ 拡張属性を保持します。cp は、ファイルのデータとともに、すべてのソースファイルの拡張属性を宛先ファイルにコピーしようとします。
- / 拡張属性と拡張システム属性を保持します。cp ユーティリティーは、ファイルのデータに加え、拡張属性と拡張システム属性を各ソースファイルからコピーし、拡張属性に関連付けられた拡張システム属性を宛先ファイルにコピーしようとします。cp が拡張属性または拡張システム属性をコピーできない場合、診断メッセージが stderr に書き込まれ、(残りのオペランドすべての処理後に)ゼロ以外の終了ステータスで終了します。

-H、-L、または -P は相互排他的なオプションですが、これらを複数指定しても、エラーであるとは判断されません。最後に指定したオプションが cp の動作を決定します。

/usr/bin/cp

-p オプションが -@ オプションまたは -/ オプションとともに指定されている場合、/usr/bin/cp の動作は次のようになります。

- -p と -@ の両方が任意の順序で指定されている場合、拡張属性をコピーできないとコピーは失敗します。
- -p と -/ の両方が任意の順序で指定されている場合、拡張システム属性をコピーできないとコピーは失敗します。

/usr/xpg4/bin/cp

-p オプションが -@ オプションまたは -/ オプションとともに指定されている場合、/usr/xpg4/bin/cp の動作は次のようになります。

- -p と -@ の両方が指定されている場合は、最後に指定されたオプションによって、拡張属性を保持できないとコピーが失敗するかどうかが決まります。
- -p と -/ の両方が指定されている場合は、最後に指定されたオプションによって、拡張システム属性を保持できないとコピーが失敗するかどうかが決まります。

オペランド

次のオペランドがサポートされています。

<i>source_file</i>	コピーされる通常ファイルのパス名
<i>source_dir</i>	コピーされるディレクトリのパス名
<i>target_file</i>	1つのファイルをコピーする際に出力として用いられるファイルのパス名。既存のファイルでも存在していないファイルでもよい
<i>target</i>	コピーしたファイルを出力するディレクトリのパス名

使用法

ファイルが2ギガバイト (2³¹ バイト) 以上ある場合の cp の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 1つのファイルをコピーする

次の例は、1つのファイルをコピーします。

```
example% cp goodies goodies.old
```

```
example% ls goodies*
goodies goodies.old
```

例2 一群のファイルをコピーする

次の例は、一群のファイルを目的のディレクトリにコピーします。

```
example% cp ~/src/* /tmp
```

例3 あるディレクトリをコピーする

次の例は、あるディレクトリを、最初は新しいディレクトリにコピーして、次に既存のディレクトリにコピーします。

```
example% ls ~/bkup
/usr/example/fred/bkup not found
```

```
example% cp -r ~/src ~/bkup
```

```
example% ls -R ~/bkup
x.c y.c z.sh
```

```
example% cp -r ~/src ~/bkup
```

```
example% ls -R ~/bkup
src x.c y.c z.sh
src:
x.c y.c z.s
```

例4 拡張ファイルシステム属性をコピーする

次の例では、拡張ファイルシステム属性をコピーします。

```
$ ls -/ c file1
-rw-r--r--  1 foo  staff          0 Oct 29 20:04 file1
{AH-----m--}
```

```
$ cp -/ file1 file2
$ ls -/c file2
-rw-r--r--  1 foo  staff          0 Oct 29 20:17 file2
{AH-----m--}
```

例5 拡張システム属性のコピーに失敗する

次に、拡張システム属性のコピーに失敗する例を示します。

例5 拡張システム属性のコピーに失敗する (続き)

```
$ ls -/c file1
-rw-r--r--  1 foo    staff          0 Oct 29 20:04 file1
              {AH-----m--}

$ cp -/ file1 /tmp
cp: Failed to copy extended system attributes from file1 to /tmp/file1

$ ls -/c /tmp/file1
-rw-r--r--  1 foo    staff          0 Oct 29 20:09 /tmp/file1
              {}
```

環境変数 cp の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの `yesexpr` キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、`yesexpr` に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、`yesexpr` に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#)を参照してください。

終了ステータス 次の終了ステータスが返されます。

- 0 すべてのファイルを正常にコピーしました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#)を参照してください。

/usr/bin/cp

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

/usr/xpg4/bin/cp

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実

関連項目	chmod(1) , chown(1) , setfacl(1) , utime(2) , fgetattr(3C) , attributes(5) , environ(5) , fsattr(5) , largefile(5) , locale(5) , standards(5)
注意事項	ソースファイルのアクセス権モードは、コピー時にそのまま保存されます。 -- によって、ユーザーはコマンド行オプションの終端を明示的にマーク付けできるので、cp は - で始まるファイル名引数を認識できるようになります。

名前	cpio - アーカイブからのファイルの抽出または復元
形式	<pre>cpio -i [-bBcdfkmPrsStuvV6@/] [-C bufsize] [-E file] [-H header] [-I [-M message]] [-R id] [pattern]... cpio -o [-aABcLPvV@/] [-C bufsize] [-H header] [-O file [-M message]] cpio -p [-adlLmPuvV@/] [-R id] directory</pre>
機能説明	cpio コマンドは、ファイルを cpio アーカイブへコピーしたり、cpio からコピーしたりします。cpio アーカイブは複数のボリュームにわたることが出来ます。-i、-o、-p の各オプションは、実行する処理を選択します。以下に、これらの各処理を説明します。これらの処理は相互に排他的です。
復元モード	cpio -i (copy in、復元) は、以前に cpio -o コマンドが生成した出力を標準入力としてファイルを抽出します。pattern の 1 つに一致する名前を持つファイルのみが選択されます。pattern についての詳細は、後述する「オペランド」の項および sh(1) の説明を参照してください。抽出されたファイルは条件に応じて作成され、以下に述べるオプションに基づいて現在のディレクトリにコピーされます。ファイルのアクセス権は前の cpio o コマンドと同じです。所有者とグループは、現在のユーザーが {PRIV_FILE_CHOWN_SELF} 特権を保持している場合を除き、現在のユーザーと同じです。 chown(2) を参照してください。このような場合は、所有者およびグループは、前に実行した cpio -o コマンドの結果と同じになります。cpio -i が作成しようとするファイルがすでに存在していて、さらに既存のファイルが同じか新しいバージョンのときは、メッセージが出力されファイルは置換されないのので注意してください。-u オプションは既存のファイルを無条件に置換するときに使用します。
保管モード	cpio -o (copy out、保管) は、ファイルのパス名のリストを標準入力から読み込んで、それらのファイルをパス名およびステータス情報と共に、cpio アーカイブ形式で標準出力へコピーします。出力は、デフォルトでは 8192 バイト境界でブロック化されますが、(-B または -c オプションを使用して) ユーザーが指定したブロックサイズまたは (CTC テープのように) デバイスに依存したブロックサイズでの出力も可能です。
パスモード	cpio -p (pass、パス) は、ファイルのパス名のリストを標準入力から読み込んで、条件に応じてファイルを作成し、以下のオプションに基づいてコピー先のディレクトリへコピーします。 ソースファイルの基盤となるファイルシステムが pathconf(2) で報告されるホールの検出をサポートしており、かつそのファイルが疎ファイルで、宛先ファイルがシーク可能である場合、その疎ファイル内のホールはパスモードに保持されます。それ以外の場合は、ホールが 0 で埋められます。 cpio は 4 バイトワードを前提としています。 文字デバイスへの出力 (-o) または文字デバイスからの入力 (-i) 時に、cpio が媒体の終わりに到達し、さらに -o オプションや -I オプションが使用されていないと、cpio は次のメッセージを表示します。

To continue, type device/file name when ready.

継続するには、媒体を交換して文字型特殊デバイス名を入力して RETURN キーを押してください。

オプション

サポートしているオプションは、次のとおりです。

- i (copy in、復元) 標準入力からアーカイブを読み込んで、条件に応じて、アーカイブに含まれているファイルを抽出し、現在のディレクトリツリーに配置します。
- o (copy out、保管) ファイルのパス名のリストを標準入力から読み込んで、そのファイルを `cpio` 形式で標準出力へコピーします。
- p (pass、パス) ファイルのパス名のリストを標準入力から読み込んで、条件に応じて、読み込んだファイルを宛先のディレクトリツリーへコピーします。

前述の -i、-o、-p オプションのいずれかを指定したあとに、次のオプションを任意の順序で指定できます。

- 0 名前に NEWLINES を含むファイルをアーカイブできるように、NEWLINE ではなく NULL 文字で終わるファイル名のリストを読み込みます。この種のファイル名リストを生成する方法としては、`find` とともに `-print0` オプションを使用する方法があります。
このオプションは、保管モードとコピーパスモードで使用できます。
- a コピー後に入力ファイルのアクセス時間をリセットして、`cpio` によるアクセスの痕跡を消去します。`cpio -pla` が指定されていると、リンクされたファイルのアクセス時間はリセットされません。
- A ファイルをアーカイブへ追加します。`-A` オプションには `-o` が必要です。このオプションは、ファイル、またはハードディスクパーティションのアーカイブにのみ有効です。アーカイブ内に以前から存在する、リンクされたファイルへの影響は予測不能です。
- b 各ワード内のバイト順を逆にします。`-i` オプションとのみ使用してください。
- B 入出力を 5120 バイトでブロック化します。このオプションと `-c` オプションを使用していないときのデフォルトのバッファサイズは 8192 バイトです。`-B` は `-p` (pass、パス) オプションには適用されません。
- c 可搬性のために、ASCII 文字形式でヘッダー情報を読み書きします。このヘッダー形式に関してユーザー ID またはグループ ID の制限はありません。SVR4 をベースとしたマシン間ではこのオプションを使用してください。また種類が不明なマシン間では `-Hodc` オプションを使用してください。`-c` オプションは、SVR4 ベースのシステムでだけサポートされている拡張デバイス番号を使うことを意味しています。

す。SunOS 4 または Interactive UNIX と、Solaris 2.6 オペレーティング環境またはその互換バージョンとの間でファイルを転送する場合は、`-H odc` を使用してください。

- `-C bufsize` 入出力を *bufsize* 単位でブロック化します。*bufsize* は正の整数で置き換えられます。このオプションと `-B` オプションを使用していないときのデフォルトのバッファサイズは 8192 バイトです。`-C` は `-p` (pass、パス) オプションには適用されません。
- `-d` 必要に応じてディレクトリを作成します。
- `-E file` アーカイブから抽出するファイル名のリストを含む (1 行 1 ファイル名の) 入力ファイル (*file*) を指定します。
- `-f` *pattern* で指定されたものを除くすべてのファイルを抽出します。*pattern* については「オペランド」の項を参照してください。
- `-H header` *header* 形式でヘッダー情報を読み書きします。コピー先とコピー元のマシンが異なるタイプのときは常にこのオプション (または `-c` オプション) を使用してください。このオプションは、`-c` および `-6` と同時には使用できません。

header として有効な値は以下のとおりです。

<code>bar</code>	<code>bar</code> ヘッダーと形式。 <code>-i</code> オプション (読み取り専用) とのみ使用されます。
<code>crc</code> <code>CRC</code>	拡張デバイス番号とファイル単位のチェックサムを持った ASCII ヘッダー。このヘッダー形式に関してユーザー ID またはグループ ID の制限はありません。
<code>odc</code>	スモールデバイス番号を持った ASCII ヘッダー。これは IEEE/P1003 データ交換標準による <code>cpio</code> のヘッダーと形式です。他のヘッダー形式と比べて、最も幅広い可搬性を提供します。POSIX に準拠したシステム間でファイル転送を行う場合の公式の形式です (standards(5) を参照)。SunOS 4 または Interactive UNIX と通信する場合には、この形式を使用してください。このヘッダー形式では、ヘッダーにユーザー ID とグループ ID を最大 262143 まで格納できます。
<code>tar</code> または <code>TAR</code>	<code>tar</code> ヘッダーと形式。これは 2097151 までのユーザー ID とグループ ID を格納できる、古い <code>tar</code> ヘッダー形式です。 <code>-i</code> オプションと一緒に使用して、古い形式のアーカイブを読み込むためにのみ用意されています。

このアーカイブ形式を `-o` オプションと一緒に使用すると、"ustar" 形式を指定した場合と同じ効果が得られます。つまり、出力アーカイブは ustar 形式になり、アーカイブの読み込みには `-Hustar` を使用する必要があります。

ustar | USTAR IEEE/P1003 データ交換標準の tar ヘッダーと形式。このヘッダー形式では、ヘッダーにユーザー ID とグループ ID を最大 2097151 まで格納できます。

前述の制限よりも大きいユーザー ID およびグループ ID を持つファイルは 60001 のユーザー ID およびグループ ID で格納されます。大規模ファイル (8G バイト - 1G バイト) を転送するには、ヘッダー形式は tar または TAR、ustar または USTAR、odc のいずれかのみを使用できます。

- I *file* 入力アーカイブとして、標準入力の代わりに *file* の内容を読み込みます。*file* が文字型特殊デバイスで、現在の媒体をすべて読み終えた場合、処理を続けるために媒体を交換して RETURN キーを押してください。このオプションは `-i` オプションとのみ使用できます。
- k 破壊されたファイルヘッダーや I/O エラーを読み飛ばします。破壊されたりシーケンスが乱れたりした媒体からファイルをコピーしたい場合は、このオプションによって正常なヘッダーを持つファイルだけを読むことができます。ほかの cpio アーカイブを含む cpio アーカイブの場合、エラーが発生すると cpio は途中で終了できます。cpio はより小さいアーカイブのヘッダーである可能性のある次の正常なヘッダーを検索し、より小さいアーカイブのトレーラが見つかった時点で終了します。`-i` オプションとのみ使用してください。
- l パスモードでは、可能な場合は必ずリンク元とリンク先の間にハードリンクを作成します。`-L` オプションを同時に指定した場合は、シンボリックリンクによって参照されているファイルに対してハードリンクを作成します。そうでない場合は、シンボリックリンク自体に対してハードリンクを作成します。このオプションは、`-p` オプションとのみ使用してください。
- L シンボリックリンクをたどります。シンボリックリンクの宛先がディレクトリであった場合は、参照されているディレクトリを、そのリンクの名前で保存します。そうでない場合は、参照されているファイルを、そのリンクの名前で保存します。
- m 以前のファイル更新時間を保持します。このオプションは、コピー中のディレクトリには無効です。
- M *message* メディア交換時の *message* を定義します。オプション `-O` または `-I` を使用して、文字型特殊デバイスを指定しているときに、媒体の終わりに

達したときに出力されるメッセージを定義します。次の媒体のシーケンス番号を表示するのに1つの%dを使用できます。

- O *file* cpio の出力先を、標準出力から *file* に変更します。 *file* が文字型特殊デバイスで現在の媒体が一杯のとき処理を継続するには、媒体を交換してキャリッジリターンを押してください。 -o オプションとのみ使用できます。
- P ACL を保持します。このオプションを出力用に使用した場合、既存の ACL は、拡張された属性を除くほかの属性とともに標準出力に書き込まれます。ACL は、特殊なファイルタイプを持つ特殊ファイルとして作成されます。このオプションを入力用に使用した場合、ACL は他の属性とともに標準入力から抽出されます。このオプションは特殊ファイルタイプを認識できます。ACL を持つ cpio アーカイブを旧バージョンの cpio によって抽出した場合、エラーが発生することに注意してください。ACL がすべてのシステムにサポートされているとはかぎらず、可搬性がないため、このオプションは -c オプションとともに使用しないでください。可搬性を保つために ASCII ヘッダーを使用してください。
- r 対話形式でファイル名を変更します。キャリッジリターンだけを押しと、そのファイルを飛ばします。"." と入力すると、元のパス名が使用されます。cpio -p とは使用できません。
- R *id* 各ファイルの所有者とグループ情報をユーザー ID に再割り当てします(ID は passwd データベース内の有効なログイン ID でなければなりません)。このオプションが有効なのは、ID が呼び出し元のユーザーまたはスーパーユーザーの場合だけです。「注意事項」を参照してください。
- s 各ハーフワード毎にバイトを交換します。
- S 各ワード毎にハーフワードを交換します。
- t 入力の内容を表示します。入力したファイルに拡張属性が含まれている場合は、それらの属性も表示します。ファイルは生成されません。-t と -v は同時には使用できません。
- u 無条件にコピーを実行します。通常、古いファイルは同一名の新しいファイルを置換しませんが、古いディレクトリは新しいディレクトリを更新します。
- v 冗長。ファイル名のリストと拡張属性の名前を出力します。-t オプションと一緒に使用すると ls -l コマンドの出力のようになります (ls(1) 参照)。

- V 詳細表示。入力または出力した各ファイルの内容を完全に表示します。すべてのファイル名を表示せずに `cpio` が動作していることをユーザーが確認するときに便利です。
- 6 UNIX System Sixth Edition 形式のアーカイブ形式ファイルを処理します。`-i` オプションとのみ使用してください。このオプションは、`-c` および `-H` オプションと同時に使用できません。
- @ 拡張属性をアーカイブに含めます。デフォルトでは、`cpio` は拡張属性をアーカイブに含めません。このフラグを指定すると `cpio` は、アーカイブ内のファイルに拡張属性が存在するかどうかを検査し、存在する場合は、その拡張属性を通常のファイルと同様にアーカイブに含めます。拡張属性ファイルは、特殊なファイルタイプを持つ特殊ファイルとしてアーカイブに収められます。`-@` フラグをオプション `-i` または `-p` と一緒に使用すると、`cpio` は、拡張属性のデータを通常のファイルデータと一緒に復元します。拡張属性ファイルは、通常のファイルのデータの一部としてのみ抽出できます。拡張属性ファイルだけを明示的に抽出しようとするすると無視されます。
- / 拡張システム属性をアーカイブに含めます。デフォルトでは、`cpio` は拡張システム属性をアーカイブに含めません。このフラグを指定すると `cpio` は、アーカイブ内のファイルに拡張システム属性が存在するかどうかを検査し、存在する場合は、そのシステム属性を通常のファイルと同様にアーカイブに含めます。拡張属性ファイルは、特殊なファイルタイプを持つ特殊ファイルとしてアーカイブに収められます。`-/` フラグをオプション `-i` または `-p` と一緒に使用すると、`cpio` は、拡張システム属性のデータを通常のファイルデータと一緒に復元します。拡張システム属性ファイルは、通常のファイルのデータの一部としてのみ抽出できます。拡張属性ファイルだけを明示的に抽出しようとするすると無視されます。

オペランド

次のオペランドがサポートされています。

directory `cpio -p` の対象となる既存ディレクトリのパス名。

pattern パターンマッチング用の表現方法を利用した式。これはシェルがファイル名のパターンマッチングに利用する式 ([sh\(1\)](#) を参照) や正規表現に似たものです。以下に示すメタキャラクタが定義されています。

- * 空の文字列を含み、あらゆる文字列と一致します。
- ? 任意の単一文字と一致します。
- [...] 括弧内のいずれか 1 文字を表します。2 つの文字を `-` で区切って指定すると、システムのデフォルトの照合シーケンスに従ってその 2 文字の間のすべての文字 (2 文字自身も含む) と一致します。先頭の `[` の次の文字が `!` である場合、結果は不確定です。

!(感嘆符) は否定を表します。たとえば `!abc*` と指定すると、文字列 `abc` で始まるファイル名とは一致しません。

pattern 指定において、メタキャラクタ `?`、`*`、`[...]` はスラッシュ (`/`) と一致し、バックスラッシュ (`\`) はエスケープと一致します。複数 *pattern* も指定可能で、*pattern* が何も指定されなければデフォルト値として `*` (すなわち、すべてのファイルを選択する) が採用されます。

各パターンは二重引用符で囲む必要があります。そうしないと、現在のディレクトリ内のファイルが使用されることがあります。

使用法 ファイルが2ギガバイト (2³¹ バイト) 以上ある場合の `cpio` の動作については、[largefile\(5\)](#) を参照してください。

使用例 以下に `cpio` の使用例を示します。

例1 標準入力を使用する

```
example% ls | cpio -oc > ../newfile
```

前述の例のように標準入力をパイプ経由で `cpio -o` に渡すと、ファイルがグループ化されて1つのアーカイブファイル (`../newfile`) にまとめられます。`-c` オプションは (`-H` オプションと同様に)、アーカイブファイルのほかのマシンとの互換性を保証します。ファイル名のリストをパイプ経由で `cpio` に渡すには、[ls\(1\)](#) の代わりに [find\(1\)](#)、[echo\(1\)](#)、[cat\(1\)](#) などを使用できます。ファイルではなくデバイスへ出力先を変更することもできます。

例2 ディレクトリにファイルを抽出する

```
example% cat newfile | cpio -icd "memo/a1" "memo/b*"
```

この例では、`cpio -i` は `cpio -o` の出力を使用して (`cat` を使用してパイプに渡している)、パターン (`memo/a1`、`memo/b*`) に一致するファイルを抽出します。さらに必要に応じて現在のディレクトリ下にディレクトリを作成し (`-d` オプション)、適切なディレクトリにファイルを格納します。`-c` オプションは互換性のあるヘッダーで入力ファイルが作成されているときに使用します。何もパターンを指定しないと、`newfile` 内のすべてのファイルをディレクトリに格納します。

例3 別のディレクトリにファイルをコピーまたはリンクする

```
example% find . -depth -print | cpio -pdlmv newdir
```

この例では、`cpio -p` はパイプ経由でファイル名を読み込んで、それらのファイルを別のディレクトリ (`newdir`) へコピーまたはリンク (`-l` オプション) します。`-d` オプションは必要に応じてディレクトリを作成します。`-m` オプションは変更時間を保持します。(`cpio` に渡すパス名を作成するには [find\(1\)](#) の `-depth` オプションを使用してください。これによって読み取りのみが許可されたディレクトリ下にファイルを作成しようとするときに発生する問題を排除できます。) コピー先のディレクトリ

例3 別のディレクトリにファイルをコピーまたはリンクする (続き)

newdir が存在している必要があります。

find と一緒に cpio を使用する場合、cpio のオプションに -L を使用しているときは find のオプションに -follow を (逆の場合も同様) 使用してください。それ以外の場合、予期しない結果になります。

マルチリールアーカイブに対しては、古いボリュームのマウントを解除し、新しいボリュームをマウントし、さらに次の装置名を入力して (通常、最初のリールと同じ) 次のテープに引き継いでください。停止するには、RETURN キーを押すと cpio が終了します。

環境変数 cpio の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照してください。LC_COLLATE、LC_CTYPE、LC_MESSAGES、LC_TIME、TZ、および NLSPATH。

TMPDIR cpio の一時ファイルは、デフォルトでは /var/tmp に作成されます。それ以外の場合は、TMPDIR で指定したディレクトリに作成されます。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

関連項目 [ar\(1\)](#), [cat\(1\)](#), [echo\(1\)](#), [find\(1\)](#), [ls\(1\)](#), [pax\(1\)](#), [setfacl\(1\)](#), [sh\(1\)](#), [tar\(1\)](#), [chown\(2\)](#), [archives.h\(3HEAD\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 cpio アーカイブで使用できるパス名の最大長は、そのアーカイブに含まれているヘッダータイプによって決まります。以下の表に、サポートされているヘッダータイプと、そのヘッダータイプで許可されている最大パス長を示します。

ヘッダータイプ	コマンド行オプション	最大パス長
BINARY	“-o”	256
POSIX	“-oHodc”	256

ヘッダータイプ	コマンド行オプション	最大パス長
ASCII	"-oc"	1023
CRC	"-oH crc"	1023
USTAR	"-oH ustar"	255

コマンド行オプション“-o -H tar”を指定した場合は、作成されるアーカイブのタイプは USTAR になります。つまり、このアーカイブをコマンド行オプション“-i -H tar”を使用して読み込むとエラーが発生します。このアーカイブの読み込みには、コマンド行オプション“-i -H ustar”を使用してください。“-i -H tar”オプションは、古い tar アーカイブ形式を読み込むためのものです。

選択されたヘッダー形式に対してユーザー ID またはグループ ID が大きすぎるファイルについてはエラーメッセージが出力されます。ユーザー ID またはグループ ID のすべての値をサポートできるアーカイブを作成するには、-H crc あるいは -c を使用してください。

スーパーユーザーだけが特殊ファイルをコピーできます。

512 バイトを 1 ブロックとみなします。

ファイルのアクセス権が 000 で、なんらかの文字データを持ち、さらにユーザーが root でない場合、そのファイルは保管または復元されません。

{PRIV_FILE_CHOWN_SELF} 特権を持つユーザーにより、そして特に {POSIX_CHOWN_RESTRICTED} が有効でない(上書きされる場合を除き、この特権を事実上すべてのユーザーに許可する)システムに対して、cpio が復元またはパスモードで呼び出されると、抽出またはコピーされたファイルの所有者およびグループが、呼び出し元のユーザーとは異なる、元のアーカイブ済みファイルの所有者およびグループによって決定される可能性があります。これはユーザーが意図したことではない場合があります。ユーザーの ID を指定した場合には、必要に応じて、-R オプションを指定することでファイルの所有権を保持できます。

ヘッダーに書かれている i ノード番号 (/usr/include/archives.h) は unsigned short (2 バイト) です。このため、この i ノード番号は 0 から 65535 までの値となります。ハードリンクされたファイルはこの i ノード番号の範囲に入る必要があります。これはベンダーが異なるマシン間で cpio アーカイブを移動する場合に問題となることがあります。

テープからハードディスクへファイルを取り出したりコピーしたりする場合は、ハードディスクからテープへコピーした時と同じブロック化因数を使用してください。したがって、-B オプションまたは -c オプションを指定してください。

-p および -o の処理中は、標準入力上のファイルリストは、cpio によって一時ファイルに保存されます。

大規模ファイルに対しては、新しい [pax\(1\)](#) 形式とそれをサポートするコマンド (`tar` など)を使用する必要があります。現在、`cpio` コマンドは最新の POSIX 標準の一部ではありませんので、`pax` の方が推奨されます。

名前	cpp - C 言語プリプロセッサ
形式	<pre>/usr/lib/cpp [-BCHMPRT] [-undef] [-Dname] [-Dname = def] [-Idirectory] [-Uname] [-Ydirectory] [input-file [output-file]]</pre>
機能説明	cpp は、C 言語プリプロセッサです。cpp は、ほかの Sun コンパイラの第 1 パスプリプロセッサとしても使用されます。 cpp はマクロプロセッサとしても使用できますが、その出力がコンパイラの第 2 パスへの入力として使用できる形式となっているため、一般にマクロプロセッサとしての使用はお勧めできません。このため、コンパイルコマンドの使用が cpp の望ましい呼び出し方法です。汎用のマクロ処理については、m4(1) を参照してください。 cpp は、オプションとして 2 つのファイル名を引数として受け入れます。input-file および output-file は、それぞれ、プリプロセッサ用の入力ファイルと出力ファイルです。デフォルトでは、標準入力と標準出力が使用されます。
オプション	サポートしているオプションは、次のとおりです。 -B C++ の注釈インジケータ // をサポートします。このインジケータを使用すると、// の後の行は注釈として扱われます。 -C すべての注釈 (cpp の指令行にあるものは除く) をそのまま渡します。デフォルトでは、cpp は C 言語スタイルの注釈を削除します。 -H インクルードファイルのパス名を、1 行に 1 つずつ標準エラー出力に出力します。 -M メイクファイル依存関係のリストを生成して、標準出力に書き込みます。このリストは、入力ファイルから生成されるオブジェクトファイルが、入力ファイルだけでなく、参照されるインクルードファイルにも依存することを示します。 -p 最初の 8 文字だけを使用してプリプロセッサシンボルを識別し、指令を含む行の終わりに余分なトークンがあれば警告を発行します。 -P C コンパイラの次のパスで使用される行制御情報を生成せずに、入力を前処理します。 -R 再帰マクロを可能にします。 -T 最初の 8 文字だけを使用して、異なるプリプロセッサシンボルを識別します。このオプションは、常に最初の 8 文字だけを使用するシステムとの下方互換性のために提供されています。 -undef 事前定義されているすべてのシンボル初期定義を削除します。 -Dname name を 1 と定義します。これは、cpp コマンド行に -Dname=1 オプションを指定した場合と同じです。または、

- `#define name 1`
 の行が `cpp` の処理するソースファイル内にある場合と同じです。
- `-Dname=def` `#define` 指令と同様に `name` を定義します。これは、
`#define name def`
 の行が `cpp` の処理するソースファイル内にある場合と同じです。 `-D` オプションは `-U` オプションよりも優先順位が低くなります。つまり、`-U` オプションと `-D` オプションの両方で同じ名前を使用した場合は、オプションの順序に関係なくその名前は定義されません。
- `-Idirectory` / 以外で始まる名前を持つ `#include` ファイルの検索パスに `directory` を挿入します。 `directory` は、標準の `include` ディレクトリリストの前に挿入されます。したがって、名前が二重引用符 (") で囲まれた `#include` ファイルは、まず `#include` 行を含むファイルのあるディレクトリで検索され、次に `-I` オプションで指定されたディレクトリで検索されて、最後に標準リストのディレクトリで検索されます。名前が山括弧 (< >) で囲まれた `#include` ファイルの場合、`#include` 行を含むファイルのあるディレクトリは検索されません。この検索順序の詳細については、以下の「詳細」を参照してください。
- `-Uname` `name` の初期定義をすべて削除します。ここで、`name` は個々のプリプロセッサによって事前定義されたシンボルです。以下に、システムのアーキテクチャーに応じて事前定義されているシンボルリストの一部を示します。
- オペレーティングシステム: `ibm, gcos, os, tss and unix`
 ハードウェア: `interdata, u3b20d,`
 `ns32000, i386, sparc, および sun`
 UNIX システムからの派生: `RES および RT`
 lint コマンド: `lint`
- シンボル `sun, sparc, unix` は、すべての Sun システムで定義されています。
- `-Ydirectory` `#include` ファイルを検索する場合は、標準のディレクトリリストの代わりに、ディレクトリ `directory` を使用します。

使用法

指令 すべての `cpp` 指令行は、ハッシュシンボル (#) で始まります。適切なインデントーションを行うために、スペース (SPACE または TAB 文字) を # の後に入れることができます。

```
#define name token-string
```

これ以降、*name* を *token-string* で置き換えます。

```
#define name(argument [, argument]...) token-string
```

name と (の間にスペースを入れることはできません。括弧で囲んだ引数のリストが後に続く、以降の *name* を *token-string* で置き換えます。ここで、*token-string* 中の各 *argument* は、コンマで区切られたリスト内の対応するトークンで置き換えられます。引数を持つマクロが展開されると、引数は展開された *token-string* にそのまま入れられます。*token-string* 全体が展開されると、cpp は、新たに作成された *token-string* の先頭から、展開すべき名前の検索を再開します。

```
#undef name
```

シンボル *name* の定義をすべて削除します。*name* の後の指令行には、トークンを付加することはできません。

```
#include "filename "
```

```
#include <filename>
```

この位置に *filename* の内容を読み込みます。このデータは、現在のファイルの一部であるかのように cpp によって処理されます。*<filename>* の表記法を使用すると、*filename* は標準の *include* ディレクトリのみで検索されます。詳細については、前述の *-I* および *-Y* オプションを参照してください。最後の " または > の後の指令行にはトークンを付加することはできません。

```
#line integer-constant "filename"
```

C コンパイラの次のパスのための行制御情報を生成します。*integer-constant* は次の行の行番号として解釈され、*filename* はその行を含むファイルとして解釈されます。*filename* が指定されていない場合、現在のファイル名は変更されません。オプションの *filename* の後の指令行には、トークンを付加することはできません。

```
#if constant-expression
```

対応する *#else*、*#elif*、または *#endif* 指令までの後続行は、*constant-expression* が 0 以外の値に評価された場合のみ出力されます。*&&*、*|*、および *,* を含む、C 言語における代入以外のすべての 2 項演算子を、*constant-expression* 中使用できます。*?:* 演算子と単項演算子 *!*、および *~* も *constant-expression* 中使用できます。

これらの演算子の優先順位は、C における優先順位と同じです。また、単項演算子 *defined* は、次の 2 つの形式で *constant-expression* 中使用できます。つまり、*defined (name)* または *defined name* です。これにより、*#ifdef* および *#ifndef* の各ディレクトリ (下記参照) は *#if* 指令で有効になります。cpp によって認識されている演算子、整数定数、および名前だけを *constant-expression* 内で使用する必要があります。特に、*sizeof* 演算子は使用できません。

#ifdef name

対応 **#else**、**#elif**、または **#endif** までの後続行は、*name* が **#define** 指令または **--D** オプションのいずれかによって定義されており、*name* が **#undef** 指令の対象となっていない場合にのみ出力されます。指令行上の *name* の後に付加されたトークンは無視されます。

#ifndef name

対応する **#else**、**#elif**、または **#endif** までの後続行は、*name* が定義されていないか、あるいは、その定義が **#undef** 指令の対象となっていない場合にのみ出力されます。*name* の後の指令行には、トークンを付加することはできません。

#elif constant-expression

#if、**#ifdef**、または **#ifndef** 指令と、対応する **#else** または **#endif** 指令の間には、任意の数の **#elif** 指令を置くことができます。**#elif** 指令に続く行は、次の条件がすべてあてはまる場合にのみ出力されます。

- 先行する **#if** 指令で *constant-expression* が 0 と評価され、先行する **#ifdef** で *name* が定義されていない、あるいは先行する **#ifndef** 指令で *name* が定義されていた。
- すべての **#elif** 指令の対象となっている *constant-expression* が 0 に評価されました。
- 現在の *constant-expression* が 0 以外に評価されました。

constant-expression が 0 以外に評価された場合、後続の **#elif** と **#else** 指令は、対応する **#endif** まで無視されます。**#if** 指令で使用できる *constant-expression* はすべて、**#elif** 指令でも使用できます。

#else

これは、条件指令の意味を反転します。つまり逆の意味になります。先行する条件指令が、行を含めることを示す場合、**#else** と対応する **#endif** の間の行は無視されます。先行する条件指令が、行を無視することを示す場合、後続の行は出力に含まれます。条件指令と対応する **#else** 指令はネストできます。

#endif

条件指令 **#if**、**#ifdef**、または **#ifndef** のいずれかによって開始した行セクションを終了します。このような各指令には、対応する **#endif** が必要です。

マクロ

マクロの正規パラメータは、文字定数や引用文字列内で発生する場合でも、**#define** 指令の本体で認識されます。たとえば、次の指令を見てください。

```
#define abc(a)|'a|
abc(xyz)
```

出力は次のようになります。

```
# 1 ""
|'xyz |
```


2 番目の行は復帰改行です。最後の 7 文字は `|'|xyz|` (縦線、逆引用符、縦線、`x`、`y`、`z`、縦線) です。マクロ名は、通常の走査中は、文字定数や引用文字列内では認識されません。したがって、

```
#define abc xyz
printf("abc");
```

前述の指令は、2 番目の行で `abc` を展開しません。これは、`abc` が、`#define` マクロ定義の一部ではない引用文字列内にあるためです。

マクロは、`#define` または `#undef` の処理中には展開されません。したがって、

```
#define abc zingo
#define xyz abc
#undef abc
xyz
```

前述の指令は `abc` を作成します。`#ifdef` または `#ifndef` のすぐ後にあるトークンは展開されません。

マクロは、別のマクロ呼び出しへの実際のパラメータを判別する走査中は展開されません。

```
#define reverse(first,second)second first
#define greeting hello
reverse(greeting,
#define greeting goodbye
)
```

の出力は次のようになります。

```
#define hello goodbye hello
```

出力

出力は入力ファイルのコピーからなり、変更が加えられており、次の形式の行が追加されています。

```
#lineno " filename" "level"
```

後続の出力行の元のソース行番号とファイル名、および、これがインクルードファイルを入力した後の最初のそのような行であるか (レベル 1)、インクルードファイルが終了した後の最初のそのような行であるか (レベル 2)、あるいはそれ以外のそのような行であるか (`level` は空) を示します。

詳細

このセクションでは、使用法の詳細を示します。

ディレクトリの検索順序

`#include` ファイルは、次の順序で検索されます。

1. `#include` 要求を含むファイルのディレクトリ (つまり `#include` は、要求が行われたときに、検索されているファイルに対応します)。
2. `-I` オプションで指定された複数のディレクトリ。左から右の順に検索されます。

3. 標準ディレクトリ (UNIX システム上の /usr/include)

特殊名	cpp は、2 つ特別な名前を認識します。名前 <code>_LINE_</code> は、cpp が認識する現在の行番号 (10 進整数) として定義されます。 <code>_FILE_</code> は、cpp が認識する現在のファイル名 (C 文字列) として定義されます。これらは、他の定義済み名と同様に、任意の場所 (マクロ内を含む) で使用できます。
改行文字	NEWLINE 文字は、文字定数または引用文字列を終了します。エスケープされた NEWLINE 文字 (つまり、バックスラッシュが先行する NEWLINE) を <code>#define</code> 文の本体で使用すると、定義を次の行に続けることができます。エスケープされた NEWLINE は、マクロ値には含まれません。
注釈	注釈は削除されます (コマンド行に <code>-c</code> オプションが指定されている場合を除く)。また、注釈がトークンを終了する場合をのぞき注釈は、無視されます。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 0 以外 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	developer/build/make

関連項目 [m4\(1\)](#), [attributes\(5\)](#)

診断 cpp によって生成されるエラーメッセージは、読めばわかるものです。エラーが発生した行番号とファイル名が、診断と共に出力されます。

注意事項 NEWLINE 文字が、展開されるマクロの引数リストで検出された場合、以前のバージョンの cpp の中には、NEWLINE 文字が検出されて展開された場合のように、NEWLINE 文字を出力するものがあります。現在のバージョンの cpp は、NEWLINE 文字を SPACE 文字で置き換えします。

 インクルードファイルの標準ディレクトリは環境によって異なる場合があるため、次の形式の `#include` 指令を使用する必要があります。

`#include <file.h>`

 次のような絶対パスを使用する `#include` 指令は使用しないようにしてください。

`#include "/usr/include/file.h"`

 cpp は、絶対パス名が使用されていれば警告を出します。

 コンパイラでは 8 ビットの文字列と注釈を使用できますが、コンパイラ以外では 8 ビットは使用できません。

名前	cputrack – CPU パフォーマンスカウンタによるプロセスとLWP 動作の監視
形式	<pre>cputrack -c eventspec [-c eventspec]... [-efntvD] [-N count] [-o pathname] [-T interval] command cputrack -c eventspec [-c eventspec]... -p pid [-efntvD] [-N count] [-o pathname] [-T interval] cputrack -h</pre>
機能説明	cputrack ユーティリティーを使用すると、システムで実行中のプロセスまたはプロセスファミリの動作を CPU パフォーマンスカウンタで監視できます。<i>interval</i> に <i>-T</i> オプションが指定されている場合、cputrack は <i>interval</i> 秒間隔でアクティビティーをサンプリングする処理を永久に繰り返します。<i>-N</i> オプションに <i>count</i> が指定されている場合は、追跡されるプロセスごとに統計が <i>count</i> 回繰り返されます。どちらも指定されていない場合、間隔は 1 秒になります。<i>command</i> およびオプションの <i>args</i> が指定されている場合、cputrack は指定された CPU パフォーマンスイベントの監視中に設定された引数を使用してコマンドを実行します。あるいは、<i>-p</i> オプションを使用すると、既存のプロセスのプロセス ID を指定できます。 cputrack は非特権プログラムであるため、truss(1) に適用されるのと同じ制限に従います。たとえば、setuid(2) 実行可能ファイルは追跡できません。
オプション	サポートしているオプションは、次のとおりです。 -c eventspec 監視する CPU パフォーマンスカウンタ用のイベントセットを指定します。これらのイベント指示子の構文を次に示します。 <pre>[picn=]eventn[,attr[n][=val]][, [picn=]eventn [,attr[n][=val]],...,]</pre> -h オプションを使用すると、使用可能なイベントと属性の一覧を取得できます。これにより、使用法に関するメッセージが表示されます。明示的なカウンタの割り当ては省略できます。その場合、cpustat は有効なカウンタの選択を自動的に試みます。 属性値は、strtoll(3C) に適した形式 (16 進数、8 進数、または 10 進数) で表現できます。イベント指示子の属性に明示的な値が指定されていない場合、デフォルト値の 1 が使用されます。対応するカウンタ番号のない属性は、指示子内のすべてのカウンタに適用されます。 これらのイベント指示子のセマンティクスを確認するには、CPU の製造元のドキュメントでイベントに関する記述を参照してください。 -c オプションは複数指定できます。その場合、cputrack はサンプルごとに異なるイベント設定に切り替えます。 -D デバッグモードを有効にします。 -e すべての exec(2)、または execve(2) システムコールを追跡します。

- f
fork(2)、fork1(2)、またはvfork(2) システムコールによって作成されたすべての子を追跡します。
- h
ユーティリティーの使用方法、プロセッサに依存するカウンタのプログラミング方法、および詳細情報の参照先に関する拡張ヘルプメッセージを出力します。
- n
すべてのヘッダー出力を省略します (cputrack がパイプラインの先頭である場合)。
- N *count*
終了前に取得する CPU パフォーマンスカウンタのサンプルの最大数を指定します。
- o *outfile*
cputrack の出力に使用するファイルを指定します。
- p *pid*
引数を、プロセスカウンタのコンテキストを追加して監視しなければならない既存プロセスのプロセス ID として解釈します。
- t
出力にプロセッサのサイクル数の列を追加します (現在のアーキテクチャーで可能な場合)。
- T *interval*
CPU パフォーマンスカウンタのサンプリング間隔を秒単位で指定します。間隔が短すぎると、一部のサンプリングがスキップされることがあります。「警告」を参照してください。
- v
詳細情報の出力を有効にします。

使用法

オペレーティングシステムは、プロセスの追跡に一定の制限を課します。具体的には、ユーザーは、自身が読み取りできないオブジェクトファイルを持つコマンドを追跡することはできません。setuid と setgid を持つコマンドは、特権ユーザー以外では追跡できません。特権ユーザーによって実行されないかぎり、cputrack は、set-id または読み取り不可能なオブジェクトファイルの exec() を実行するすべてのプロセスの制御を失います。このようなプロセスは、cputrack から独立しているにもかかわらず、exec() から見ると正常に処理を続行します。

-f オプションを使用した場合、システムがユーザーごとのプロセススロットを使い果たしてしまう場合があります。これは、cputrack が、追跡するプロセスごとに 1 つの制御プロセスを実行するためです。

cputrack が出力する時間は、ハードウェアカウンタが実際にサンプリングされた時間に対応します。この時間は、[gethrtime\(3C\)](#) と同じタイムベースから決定されます。

cputrack ユーティリティーは、パフォーマンスカウンタのコンテキストを検査対象の各プロセスに追加します。このコンテキストが存在する場合は、パフォーマンスカウンタをシステム上のさまざまなプロセス間で多重化できますが、[cpustat\(1M\)](#) ユーティリティーと同時に使用できません。

cpustat ユーティリティーのインスタンスが実行されると、cpustat のインスタンスをすべて終了しないかぎり、cputrack をさらに実行しようとしても失敗します。

場合によっては、cputrack の柔軟性が非常に高く、多くの統計情報が出力されるために、監視コードが必要以上にアプリケーションに追加されることがあります。ただし、より厳密な制御をした方が望ましい場合があります。アプリケーション自体、および cputrack によりアプリケーション内に注入されるエージェント LWP が同じパフォーマンスカウンタのコンテキストを使用するため、アプリケーションでカウンタコンテキストを対話的に処理して、興味深い機能を実現できます。[cpc_enable\(3CPC\)](#) を参照してください。

-t オプションで有効されたプロセッサのサイクル数は、パフォーマンスカウンタのレジスタに適用された設定に関係なく、常にユーザーモードとシステムモードの両方に適用されます。

cputrack の出力は [nawk\(1\)](#) や [perl\(1\)](#) で簡単に解析できるように設計されているため、スクリプト内に cputrack を埋め込むことによってパフォーマンスツールを構成できます。あるいは、[libcpc\(3LIB\)](#) や [libpctx\(3LIB\)](#) の機能を使用すると、cputrack が構築されているのと同じ API を使用してツールを直接構築できます。[cpc\(3CPC\)](#) を参照してください。

cputrack はパフォーマンスカウンタのコンテキストを使用して LWP ごとに個別のパフォーマンスカウンタを管理しますが、カウント可能なイベントの一部は、特に(キャッシュ失敗率)のようにプロセス間で共有されるリソースが限定されている場合、システムで発生するほかのアクティビティーの影響を必ず受けます。このようなイベントでは、[cpustat\(1M\)](#) を使用してシステムの動作全体を監視してみるのもよいでしょう。

-T *interval* オプションでは、*interval* をゼロに指定すると定期的なサンプリングは実行されません。パフォーマンスカウンタがサンプリングされるのは、プロセスが LWP を作成または破棄したり、[fork\(2\)](#)、[exec\(2\)](#)、または [exit\(2\)](#) を呼び出したりする場合だけです。

使用例

SPARC

例1 パフォーマンスカウンタを使用してクロックサイクルをカウントする

この例では、UltraSPARC-III+ プロセッサを搭載したマシンでユーティリティーを使用しています。カウンタは、`sleep(1)` コマンドの実行中に、プロセッサのクロックサイクルおよびユーザーモードで送信された命令をカウントするように設定されています。

```
example% cputrack -c pic0=Cycle_cnt,pic1=Instr_cnt sleep 10
```

time	lwp	event	pic0	pic1
1.007	1	tick	765308	219233
2.007	1	tick	0	0
4.017	1	tick	0	0
6.007	1	tick	0	0
8.007	1	tick	0	0
10.007	1	tick	0	0
10.017	1	exit	844703	228058

例2 外部キャッシュの参照と失敗をカウントする

この例では、UltraSPARC マシンで単純なシェルスクリプトの `fork()` および `exec()` を追跡しながら、より詳細な情報を出力します。カウンタは、外部キャッシュの参照と失敗の数を測定します。pic0 および pic1 の名前は、混同される可能性がなければ明示的に指定する必要はありません。

```
example% cputrack -fev -c EC_ref,EC_hit /bin/ulimit -c
```

time	pid	lwp	event	pic0	pic1
0.007	101142	1	init_lwp	805286	20023
0.023	101142	1	fork		# 101143
0.026	101143	1	init_lwp	1015382	24461
0.029	101143	1	fini_lwp	1025546	25074
0.029	101143	1	exec	1025546	25074
0.000	101143	1	exec		\
					# '/usr/bin/sh /usr/bin/basename\
					/bin/ulimit'
0.039	101143	1	init_lwp	1025546	25074
0.050	101143	1	fini_lwp	1140482	27806
0.050	101143	1	exec	1140482	27806
0.000	101143	1	exec		# '/usr/bin/expr \
					//bin/ulimit : \(.[^/]\)/*\$: .*/\(..*\) : \(.*\)\$ //bin/ulimi'
0.059	101143	1	init_lwp	1140482	27806
0.075	101143	1	fini_lwp	1237647	30207
0.075	101143	1	exit	1237647	30207
			unlimited		
0.081	101142	1	fini_lwp	953383	23814
0.081	101142	1	exit	953383	23814

x86

例3 命令をカウントする

この例では、Pentium III マシンで日付を出力するためにアプリケーションおよびカーネルで実行された命令の数を表示します。

```
example% cputrack -c inst_retired,inst_retired,nouser1,sys1 date
```

```

time lwp      event      pic0      pic1
Fri Aug 20 20:03:08 PDT 1999
0.072  1      exit      246725    339666
```

例4 TLBのヒット回数をカウントする

この例では、Pentium 4 マシンでプロセッサ固有の属性を使用して TLB のヒット回数をカウントする方法を示します。

```
example% cputrack -c ITLB_reference,emask=1 date
```

```

time lwp      event      pic0
Fri Aug 20 20:03:08 PDT 1999
0.072  1      exit      246725
```

警告

[cpustat\(1M\)](#) ユーティリティのいずれかのインスタンスを実行すると、既存のすべてのパフォーマンスカウンタのコンテキストがマシン全体で強制的に無効にされます。その結果、cputrack コマンドのすべての呼び出しが不特定なエラーで途中終了する可能性があります。

cpustat が、Solaris でサポートされていない CPU パフォーマンスカウンタの存在するシステムで呼び出されると、次のメッセージが表示されます。

```
cputrack: cannot access performance counters - Operation not applicable
```

このエラーメッセージは、cpc_open() が失敗したことを意味し、[cpc_open\(3CPC\)](#) に記述されています。この問題および実行可能な解決方法については、このドキュメントを参照してください。

短い間隔を要求した場合、cputrack が目的のサンプルレートに追従できないことがあります。この場合は、サンプルの一部が省略されることがあります。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	diagnostic/cpu-counters
インタフェースの安定性	確実

関連項目

nawk(1), perl(1), proc(1), truss(1), prstat(1M), cpustat(1M), exec(2), exit(2), fork(2),
setuid(2), vfork(2), gethrtime(3C), strtoll(3C), cpc(3CPC), cpc_bind_pctx(3CPC),
cpc_enable(3CPC), cpc_open(3CPC), libcpc(3LIB), libpctx(3LIB), proc(4),
attributes(5)

名前	crle – 実行時リンク環境の構成
形式	<pre>crle [-64] [-a name] [-A name] [-c conf] [-e env] [-E env] [-f flags] [-i name] [-I name] [-g name] [-G name] [-l dir] [-o dir] [-s dir] [-t] [-u] [-v]</pre>
機能説明	crle ユーティリティを使用すると、実行時リンク構成ファイルを作成および表示できます。構成ファイルは、プロセスの開始時に実行時リンカー ld.so.1(1) によって読み取られ、解釈されます。実行時リンカーは、全プロセスのデフォルト構成ファイルを読み取ろうとします。32 ビットプロセスの場合、デフォルトの構成ファイルは <code>/var/ld/ld.config</code> です。64 ビットプロセスの場合、デフォルトの構成ファイルは <code>/var/ld/64/ld.config</code> です。 引数が指定されていないか、<code>-c</code> オプションのみが指定されている場合、crle は構成情報を表示します。この情報には、構成ファイルの内容、すべてのシステムデフォルト、および構成ファイルの再生成に必要なコマンド行が含まれます。ほかのオプションとともに使用すると、新しい構成ファイルが作成または更新されます。 LD_CONFIG ファミリ環境変数のいずれかを設定して、実行時リンカーを代替構成ファイルに接続することもできます。LD_CONFIG は、32 ビットと 64 ビットの両方のプログラムに適用されます。32 ビットと 64 ビットの構成ファイルは異なるため、1 つの構成ファイルを両方のオブジェクトクラスに使用することはできません。このため、あるクラスのプログラムが別のクラスのプログラムを実行する場合、LD_CONFIG がプログラムの実行に悪影響を及ぼすことがあります。特に、32 ビットバージョンの標準の Solaris ユーティリティがその 64 ビットバージョンを実行するということがよくあります。この場合、LD_CONFIG を正しく使用することはできません。したがって、適切なクラスのプロセスが正しくターゲットとして指定された LD_CONFIG_32 および LD_CONFIG_64 環境変数を使用することをお勧めします。 標準の場所である <code>/var/ld</code> に不正な構成ファイルを作成すると、プログラムが実行されず、その結果回復が困難になる可能性があります。このような状況にならないようにするために、新しい構成ファイルは一時的な場所に作成することをお勧めします。次に、適切な LD_CONFIG 環境変数をこの新しい構成ファイルに設定します。この設定によって、実行時リンカーが、デフォルトではなく新しい構成ファイルを使用するようになります。検証が終わったら、必要に応じて新しい構成ファイルをデフォルトの場所に移動できます。いつでも環境変数 LD_NOCONFIG を任意の値に設定して、実行時リンカーに構成ファイルを無視するように指示できます。この設定は、実験を行う場合に役立ちます。 構成ファイルには次の情報を含めることができます。 デフォルト検索パス 実行時リンカーは、指定した検索パスを使って、オブジェクトの動的な依存関係を検索します。この検索パスは、LD_LIBRARY_PATH 定義のコンポーネントで始まり、その後ろにオブジェクトの runpath のコンポーネントが続きます。最後に、オブジェクトのクラスに固有のデフォルト検索パスが使用されます。検索パ

スの最後のコンポーネントは、構成ファイル内に記述できます。通常、この機能を使用するときは、システムデフォルトと組み合わせて機能を拡張してください。-l および -u オプションを参照してください。

トラストディレクトリ

実行時リンカーは、セキュリティ保護されたアプリケーションを処理する際、LD_LIBRARY_PATH 検索および \$ORIGIN トークン拡張の使用を制限します。『リンカーとライブラリ』の「セキュリティ」を参照してください。また、事前に読み込まれたライブラリや監査ライブラリを検索できるディレクトリも制限されます。事前に読み込まれたライブラリや監査ライブラリに関連付けるパス名は、既知のトラストディレクトリに制限されます。トラストディレクトリは、構成ファイル内に記述できます。通常、この機能を使用するときは、システムデフォルトと組み合わせて機能を拡張してください。-s および -u オプションを参照してください。

環境変数

実行時リンカーによって解釈されるすべての環境変数は、構成ファイル内で指定できます。

ディレクトリキャッシュ

定義済みディレクトリ内部の共有オブジェクトの場所を、構成ファイル内でキャッシュとして管理できます。このディレクトリキャッシュによって、アプリケーションの依存関係を検索する際のオーバーヘッドを減らすことができます。

代替オブジェクト

ディレクトリキャッシュとともに、共有オブジェクトの代替オブジェクトを指定して実行時に使用できます。これらの代替オブジェクトは、ユーザーが指定できます。また、crle を使用すると、既知のメモリー位置に固定された共有オブジェクトのコピーとして代替オブジェクトを作成できます。これらの固定代替オブジェクトは、元の共有オブジェクトよりも実行時の処理が少なく済みます。

追加のデフォルト検索パスやトラストディレクトリを定義することは、管理者が他社のソフトウェアを1つの場所にインストールする場合に役立ちます。定義しない場合は、適切な runpath でコーディングされていない可能性があるアプリケーションの検索パスを変更することになります。

代替オブジェクトを宣言すると、シンボリックリンクを使用したり、LD_LIBRARY_PATH を設定したりせずに、依存関係を置き換えることができます。

実行時リンカーによって解釈される環境変数を宣言すると、すべてのアプリケーションの定義を集中管理できます。

ディレクトリキャッシュおよび crle によって生成される代替オブジェクトを使用すると、アプリケーション実行時の起動オーバーヘッドを減らすことができます。代替オブジェクトは、多数の依存関係が必要なアプリケーションや依存関係の再配置に大きな負荷がかかるアプリケーションで役に立つ場合があります。

す。*position-dependent* コードを含む共有オブジェクトでは、再配置に大きな負荷がかかることがよくあります。システムには負のパス検索などの負荷の軽減に役立つキャッシュ機能が多数備わっているため、`crle` を使用したディレクトリキャッシュを作成すると、一部のごく限定的な場合を除いて影響が最小限で済みます。

`crle` によって生成される代替オブジェクトが構成ファイル内で指定されている場合、実行時リンカーは最小限の一貫性検証を実行します。代替オブジェクトは元のオブジェクトと照合されます。この照合の目的は、アプリケーション構成情報が基盤となるシステムコンポーネントと同期しなくなった場合にアプリケーション障害を回避することです。こうした状況が発生すると、動的リンクシステムコンポーネントの柔軟性が損なわれることがあります。このようなアプリケーション障害は、診断が非常に困難な場合があります。ディレクトリキャッシュ情報の検証は行われません。キャッシュを再構築しないかぎり、ディレクトリ構造へのどのような変更もプロセスで検出できません。

多くの場合、システム共有オブジェクトは適切に調整されているため、キャッシュされることによるメリットはほとんどありません。通常、ディレクトリキャッシュと代替オブジェクトの機能はユーザーアプリケーションや共有オブジェクトに適用されますが、一部のごく限定的な場合にしか機能は向上しません。

`crle` は、`-I` および `-G` オプションを使用したときに検出される共有オブジェクトの代替オブジェクトを、`dldump(3C)` を使用して作成します。代替オブジェクトは、直前の `-o` オプションで指定されたディレクトリ内に作成されます。デフォルトでは、構成ファイルが作成されたディレクトリになります。`dldump()` によって使用されるフラグは、`-f` オプションを使用して指定されるか、またはデフォルトで `RTLD_REL_RELATIVE` になります。

オプション

以下のオプションがサポートされています。

-64

64 ビットオブジェクトを処理するよう指定します。デフォルトは 32 ビットです。`-64` は、64 ビット固有の構成ファイルを作成する場合に使用します。

-a *name*

name の代替パス名を作成します。代替パス名は構成ファイルに追加されます。

実際の代替ファイルは、ユーザーが準備する必要があります。このオプションは複数回指定できます。*name* がディレクトリの場合、ディレクトリ内の各共有オブジェクトがキャッシュに追加されます。*name* が存在しない場合、*name* は、存在しないファイルとしてキャッシュ内でマークされます。

通常、このオプションは `-o` オプションとともに使用されます。

-A *name*

name のオプションの代替パス名を作成します。この代替パス名は構成ファイルに追加されます。

このオプションは、`-a` オプションと同じように動作します。ただし、実行時に代替が使用できない場合には元のオブジェクト *name* が使用されます。これは、補助フィルタを使用するのと同じような動作です。『リンカーとライブラリ』の「補助フィルタの生成」を参照してください。

通常、このオプションは `-o` オプションとともに使用されます。

`-c conf`

構成ファイル名 *conf* を使用するよう指定します。このオプションを指定しないと、デフォルトの構成フィルが使用されます。

`-e env`

置換可能な環境変数 *env* を指定します。このオプションが意味を持つのは、実行時リンカーに適用可能な環境変数だけです。このオプションは複数回指定できます。このオプションは、`-E` オプションに似ています。ただし、構成ファイル定義、および同じ名前のプロセス環境定義を実行時に解決する方法が異なります。

構成ファイルで指定された定義は、プロセス環境定義で上書きしたり、NULL 値のプロセス環境定義で無効にしたりできます。

つまり、これらの構成ファイル定義は、実行時にプロセス環境で置換または削除できます。

`-E env`

永続的な環境変数 *env* を指定します。このオプションが意味を持つのは、実行時リンカーに適用可能な環境変数だけです。このオプションは複数回指定できます。このオプションは、`-e` オプションに似ています。ただし、構成ファイル定義、および同じ名前のプロセス環境定義を実行時に解決する方法が異なります。

実行時リンカーにとって意味がある環境変数定義は、2つのどちらかのカテゴリに分けられます。単数定義は、`LD_NOLAZYLOAD=1` や `LD_DEBUG_OUTPUT=file` などの定義です。リスト定義は1つ以上の値を含めることができる定義で、`LD_LIBRARY_PATH=path` や `LD_DEBUG=files,details` などがあります。

構成ファイルに記述された単数定義は、プロセス環境定義よりも優先されます。構成フィルに記述されたリスト定義は、プロセス環境定義に追加されます。構成ファイル内で記述された定義を、NULL 値のプロセス環境定義で無効にすることはできません。

つまり、これらの構成ファイル定義は、実行時にプロセス環境で置換および削除できません。

`-f flags`

代替オブジェクトの生成に使用される `dl_dump(3C)` の呼び出しに対してシンボリック *flags* 引数を指定します。`/usr/include/dlfcn.h` で定義された任意の RTLD_REL フラグを使用できます。「|」文字を使用すると、複数のフラグを OR 接続できます。この場合、シェルによって拡張されないようにするため、文字列を引用符で囲んでください。*flags* 値が指定されていない場合、デフォルトのフラグは `RTLD_REL_RELATIVE` です。

-i name

name を構成キャッシュに個別に追加します。このオプションは複数回指定できます。*name* には共有オブジェクトまたはディレクトリを指定できます。*name* がディレクトリの場合、ディレクトリ内の各共有オブジェクトがキャッシュに追加されます。*name* が存在しない場合、*name* は、存在しないディレクトリとしてキャッシュ内でマークされます。

-I name

-i と同じように動作するだけでなく、**dldump(3C)** を使用して、処理対象のすべての共有オブジェクトの代替を作成します。**-f** フラグに **RTLD_REL_EXEC** が含まれている場合は、*name* を動的実行可能ファイルにすることができ、そのファイルの代替が作成されます。作成されるキャッシュはこのアプリケーションに固有であるため、この方法で指定できる動的実行可能ファイルは1つだけです。

-g name

グループ *name* を構成キャッシュに追加します。各オブジェクトが展開されて、依存関係が特定されます。このオプションは複数回指定できます。*name* には、動的実行可能ファイル、共有オブジェクト、またはディレクトリを指定できます。*name* が共有オブジェクトの場合、共有オブジェクトおよびその依存関係がキャッシュに追加されます。*name* がディレクトリの場合、ディレクトリ内の各共有オブジェクトとその依存関係がキャッシュに追加されます。

-G name

-g オプションと同じように動作するだけでなく、**dldump(3C)** を使用して、処理対象のすべての共有オブジェクトの代替を作成します。*name* が動的実行可能ファイルであり、かつ **-f** フラグに **RTLD_REL_EXEC** が含まれている場合は、その動的実行可能ファイルの代替も作成されます。作成されるキャッシュはこのアプリケーションに固有であるため、この方法で指定できる動的実行可能ファイルは1つだけです。

-l dir

ELF オブジェクトの新しいデフォルト検索ディレクトリ *dir* を指定します。このオプションは複数回指定できます。

32 ビット ELF オブジェクトのデフォルト検索パスは順番に **/lib**、**/usr/lib** です。64 ビット ELF オブジェクトのデフォルト検索パスは、順番に **/lib/64**、そして **/usr/lib/64** です。

このオプションを使用すると、デフォルトの検索パスが置き換えられます。このため、通常は、**-l** オプションを使用して、適用される新しいパスに対応する元のシステムデフォルトを指定する必要があります。ただし、**-u** オプションが有効で、かつ構成ファイルが存在しない場合は、システムデフォルトが新しい構成ファイルに追加されます。これらのデフォルトは、**-l** オプションで指定する新しいパスの前に追加されます。

-o dir

-a または **-A** オプションとともに使用される場合、代替オブジェクトが存在するディレクトリ *dir* を指定します。代替オブジェクトが **crle** によって作成される場

合、このオプションで代替の作成場所を指定します。このオプションを指定しない場合、代替オブジェクトは構成ファイルが作成されるディレクトリ内に存在します。このオプションは複数回指定でき、ディレクトリ *dir* は、以降のコマンド行オプションの代替を検索するために使用されます。代替オブジェクトで元の関連オブジェクトをオーバーライドすることは許可されません。

通常、このオプションは *-a* または *-A* オプションとともに使用されます。

-s dir

セキュリティー保護された ELF オブジェクトの新しいトラストディレクトリ *dir* を指定します。このオプションは複数回指定できます。

セキュリティー保護されたオブジェクトの定義については、[SECURITY in ld.so.1\(1\)](#) を参照してください。セキュリティー保護されたアプリケーションに適用される実行時制限については、『[リンカーとライブラリ](#)』の「[セキュリティー](#)」を参照してください。

セキュリティー保護された 32 ビット ELF オブジェクトのデフォルトのトラストディレクトリは順番に */lib/secure*、*/usr/lib/secure* です。セキュリティー保護された 64 ビット ELF オブジェクトの場合、デフォルトのトラストディレクトリは順番に */lib/secure/64*、*/usr/lib/secure/64* です。

このオプションを使用すると、デフォルトのトラストディレクトリが置き換えられます。このため、通常は、*-l* オプションを使用して、適用される新しいディレクトリに対応する元のシステムデフォルトを指定する必要があります。ただし、*-u* オプションが有効で、かつ構成ファイルが存在しない場合は、システムデフォルトが新しい構成ファイルに追加されます。これらのデフォルトは、*-l* オプションで指定する新しいディレクトリの前に追加されます。

-t

廃止。「注意事項」を参照してください。

-u

構成ファイルの更新を要求します (通常は、新しい情報を追加して更新します)。ほかのオプションを指定しないと、既存のすべての構成ファイルが検査され、その内容が再計算されます。引数を追加すると、再計算された内容に情報を追加できます。「注意事項」を参照してください。

構成ファイルが存在しない場合は、ほかの引数で指定されたとおりに構成ファイルが作成されます。*-l* および *-s* オプションを指定すると、システムデフォルトが構成ファイルに適用されてから、これらのオプションで指定したディレクトリが適用されます。

構成ファイルを、通常ならファイルの先頭に記述されるシステム識別情報が存在しない、古い形式にすることもできます。この場合、*crle* は生成されるファイルにシステム識別情報を記録しないため、古いバージョンの Solaris との互換性が維持されます。「注意事項」を参照してください。

-v

詳細表示モードを指定します。構成ファイルの作成時には、処理中のファイルの追跡情報が標準出力に書き込まれます。構成ファイルの内容の出力時には、ディレクトリとファイルに関するより広範囲な情報が表示されます。

デフォルトでは、実行時リンカーは、処理対象の 32 ビットアプリケーションごとに構成ファイル `/var/ld/ld.config` の読み取りを試みます。また、64 ビットのアプリケーションごとに `/var/ld/64/ld.config` を読み取ります。実行時リンカーは、代替アプリケーションを処理する際に、`$ORIGIN/ld.config.app-name` 構成ファイルを使用します (存在する場合)。「注意事項」を参照してください。LD_CONFIG 環境変数を設定すると、アプリケーションから代替構成ファイルを参照できるようになります。アプリケーションの構築時に構成ファイル名をアプリケーションに記録して、代替構成ファイルを指定することもできます。[ld\(1\)](#) の `-c` オプションを参照してください。

使用例

例1 一時構成ファイルを試す

次の例では、ELF オブジェクトの新しいデフォルト検索パスを使用して一時構成ファイルを作成します。環境変数 LD_CONFIG_32 を使用して、すべての 32 ビットプロセスでこの構成ファイルを使用するよう実行時リンカーに指示します。

```
$ crle -c /tmp/ld.config -u -l /local/lib
$ crle -c /tmp/ld.config

Configuration file [version 4]: /tmp/ld.config
Platform:      32-bit MSB SPARC
Default Library Path (ELF): /lib:/usr/lib:/local/lib
Trusted Directories (ELF): /lib/secure:/usr/lib/secure \
                          (system default)

Command line:
crle -c /tmp/ld.config -l /lib:/usr/lib:/local/lib

$ LD_CONFIG_32=/tmp/ld.config date
Thu May 29 17:42:00 PDT 2008
```

例2 ELF オブジェクトの新しいデフォルト検索パスの更新および表示

次の例では、ELF オブジェクトの新しいデフォルト検索パスを更新して表示します。

```
# crle -u -l /local/lib
# crle

Configuration file [version 4]: /var/ld/ld.config
Platform:      32-bit MSB SPARC
Default Library Path (ELF): /lib:/usr/lib:/local/lib
Trusted Directories (ELF): /lib/secure:/usr/lib/secure \
                          (system default)
```

例2 ELF オブジェクトの新しいデフォルト検索パスの更新および表示 (続き)

```

Command line:
  crle -l /lib:/usr/lib:/local/lib

# crle -u -l /ISV/lib
# crle

Configuration file [version 4]: /var/ld/ld.config
Platform      32-bit MSB SPARC
Default Library Path (ELF):  /lib:/usr/lib:/local/lib:/ISV/lib
Trusted Directories (ELF):   /lib/secure:/usr/lib/secure \
                             (system default)

```

```

Command line:
  crle -l /lib:/usr/lib:/local/lib:/usr/local/lib

```

この例では、最初はデフォルトの構成ファイルがありませんでした。このため、システムデフォルトに新しい検索パス `/local/lib` が追加されます。次の更新で、検索パス `/ISV/lib` が、すでに構成ファイルに作成されているこれらのパスに追加されます。

例3 不正な構成ファイルを回復する

次の例では、デフォルトの場所に不正な構成ファイルを作成します。環境変数 `LD_NOCONFIG` を含む構成ファイルをすべて無視するよう実行時リンカーに指示することで、このファイルを削除できます。一時構成ファイルを作成し、環境変数 `LD_CONFIG` を使用してこれらのファイルを試すことをお勧めします。

```

# crle -l /local/lib
# date
ld.so.1: date: fatal: libc.so.1: open failed: \
  No such file or directory
Killed
# LD_NOCONFIG=yes rm /var/ld/ld.config
# date
Thu May 29 17:52:00 PDT 2008

```

この構成ファイルにはシステムのデフォルト検索パスが含まれていないため、`date` ユーティリティーは必要なシステム依存関係を見つけることができません。この場合に使用しなければならないオプションは `-u` でした。

例4 ELF オブジェクトの新しいデフォルト検索パスと新しいトラストディレクトリの作成および表示

次の例では、ELF オブジェクトの新しいデフォルト検索パスと新しいトラストディレクトリを作成して表示します。

例4 ELFオブジェクトの新しいデフォルト検索パスと新しいトラストディレクトリの作成および表示 (続き)

```
# crle -l /local/lib -l /lib -l /usr/lib -s /local/lib
# crle
```

```
Configuration file [version 4]: /var/ld/ld.config
Platform:      32-bit MSB SPARC
Default Library Path (ELF):  /local/lib:/lib:/usr/lib
Trusted Directories (ELF):   /local/lib
```

```
Command line:
crle -l /local/lib:/lib:/usr/lib -s /local/lib
```

この構成ファイルを使用すると、他社のアプリケーションを /local/bin に、関連する依存関係を /local/lib にインストールできます。デフォルトの検索パスを使用すると、LD_LIBRARY_PATH を設定しなくてもアプリケーションが依存関係を見つけることができます。この例では、デフォルトのトラストディレクトリも置き換えられています。

例5 ELFオブジェクトのディレクトリキャッシュの作成

次の例では、ELFオブジェクトのディレクトリキャッシュを作成します。

```
$ crle -i /usr/dt/lib -i /usr/openwin/lib -i /lib -i /usr/lib \
    -c config
$ ldd -s ./main
....
find object=libc.so.1; required by ./main
search path=/usr/dt/lib:/usr/openwin/lib (RUNPATH/RPATH ./main)
trying path=/usr/dt/lib/libc.so.1
trying path=/usr/openwin/lib/libc.so.1
search path=/lib (default)
trying path=/lib/libc.so.1
libc.so.1 => /lib/libc.so.1

$ LD_CONFIG=config ldd -s ./main
....
find object=libc.so.1; required by ./main
search path=/usr/dt/lib:/usr/openwin/lib (RUNPATH/RPATH ./main)
search path=/lib (default)
trying path=/lib/libc.so.1
libc.so.1 => /lib/libc.so.1
```

この構成では、システムライブラリ libc.so.1 がディレクトリ /usr/dt/lib にも /usr/openwin/lib にも存在しないことがキャッシュに反映されています。このため、アプリケーションの runpath でこれらのディレクトリが検索対象となっていて、このシステムファイルの検索では無視されます。

例6 ELF実行可能ファイルの代替オブジェクトキャッシュの作成

次の例では、ELF実行可能ファイルの代替オブジェクトキャッシュを作成します。

```
$ crle -c /local/$HOST/.xterm/ld.config.xterm \
      -f RTLD_REL_ALL -G /usr/openwin/bin/xterm
$ ln -s /local/$HOST/.xterm/xterm /local/$HOST/xterm
$ ldd /usr/local/$HOST/xterm
libXaw.so.5 => /local/$HOST/.xterm/libWaw.so.5 (alternate)
libXmu.so.4 => /local/$HOST/.xterm/libXmu.so.4 (alternate)
....
libc.so.1 => /local/$HOST/.xterm/libc.so.1 (alternate)
....
```

この構成を使用すると、新しいxtermおよびその依存関係が作成されます。これらの新しいオブジェクトが相互に完全に再配置されるため、元のオブジェクトより起動が高速化されます。このアプリケーションの実行には、独自の構成ファイルが使用されます。このモデルでは、通常、ほかのアプリケーションがldd(1)やtruss(1)などの構成ファイルを誤って使用してしまうことがないため、環境変数LD_CONFIGを使用する場合に比べて柔軟性が高くなります。

例7 ELF共有オブジェクトを置き換えるための代替オブジェクトキャッシュの作成

次の例では、ELF共有オブジェクトを置き換えるための代替オブジェクトキャッシュを作成します。

```
$ ldd /usr/bin/vi
libcurses.so.1 => /lib/libcurses.so.1
....

# crle -a /lib/libcurses.so.1 -o /usr/ucblib
# crle

Configuration file [version 4]: /var/ld/ld.config
Platform:      32-bit MSB SPARC
Default Library Path (ELF): /lib:/usr/lib (system default)
Trusted Directories (ELF): /lib/secure:/usr/lib/secure \
                          (system default)

Directory: /lib
libcurses.so.1 (alternate: /usr/ucblib/libcurses.so.1)
....

$ ldd /usr/bin/vi
libcurses.so.1 => /usr/ucblib/libcurses.so.1 (alternate)
....
```

この構成では、通常なら/usr/lib/libcurses.so.1に解決される依存関係がすべて/usr/ucblib/libcurses.so.1に解決されます。

例8 置換可能な環境変数および永続的な環境変数を設定する

次の例では、置換可能な環境変数と永続的な環境変数を設定します。

```
# crle -e LD_LIBRARY_PATH=/local/lib \
      -E LD_PRELOAD=preload.so.1
# crle
.....
Environment Variables:
  LD_LIBRARY_PATH=/local/lib (replaceable)
  LD_PRELOAD=preload.so.1 (permanent)

.....
$ LD_DEBUG=files LD_PRELOAD=preload.so.2 ./main
.....
18764: file=preload.so.2; preloaded
18764: file=/local/lib/preload.so.2 [ ELF ]; generating link map
.....
18764: file=preload.so.1; preloaded
18764: file=/local/lib/preload.so.1 [ ELF ]; generating link map
.....
```

この構成ファイルでは、置換可能な検索パスが、事前に読み込まれた永続的なオブジェクトとともに指定されており、そのオブジェクトはプロセス環境定義に追加されます。

終了ステータス 構成ファイルを作成または表示すると、0が返されます。それ以外の場合は、どのようなエラー条件でも、診断メッセージとゼロ以外の値がともに返されます。

注意事項 元のアプリケーションにDT_FLAGS_1およびDT_FEATURE_1のいずれかの`.dynamic`タグが含まれる場合、代替アプリケーションにタグを付けてアプリケーション固有の構成ファイルを使用できます。これらのエントリが存在しない場合は、LD_CONFIG環境変数を使用して構成ファイルを指定する必要があります。後者の方法の場合、フォークされたすべてのアプリケーションからこの環境変数が見えるようになるため、注意してください。

-u オプションを使用するには、crleのバージョン2以降が必要です。このバージョンレベルは、構成ファイルの内容を表示することで確認できます。

```
$ crle
```

```
Configuration file [2]: /var/ld/ld.config
.....
```

バージョン2の構成ファイルでは、crleは構成ファイルの再生成に必要なコマンド行引数を作成できます。このコマンド行を作成すると、-uオプションを使用して完全な更新を実行できます。バージョン1の構成ファイルを更新することはできませんが、crleですべての更新要件を計算するには、構成ファイルの内容が十分ではない可能性があります。

構成ファイルには、プラットフォーム固有のバイナリデータが含まれます。ある特定の構成ファイルは、ソフトウェアによって、同じマシナリクラスとバイト順序でのみ解釈できます。構成ファイルには、ファイルの先頭にシステム識別情報が含まれています。この情報は、構成ファイルとの互換性を確認するために `crle` と実行時によって使用されます。また、この情報によって、`file(1)` コマンドは構成ファイルを正しく識別できます。下位互換性を確保するため、この情報が不足している古いファイルも受け入れられますが、新しいファイルでは可能な識別とエラー検査は行われません。システム情報が含まれていない以前のファイルに対する更新 (`-u`) 操作の処理中に、`crle` がその結果にシステム識別情報を追加することはありません。

Oracle Solaris 11 は、SPARC ハードウェア上の SunOS 4.x AOUT 実行可能ファイルに対するサポートを中止しました。この変更の前は、`-t` オプションによって ELF または AOUT のオブジェクトタイプを表すための切り替えが提供され、それが以降のすべての `-l` または `-s` オプションに影響を与えました。`-t` オプションは現在廃止されているため、AOUT 固有の指示はすべて無視され、その影響に対する警告が表示されます。同じ理由で、`crle` を使用して AOUT の情報を含む古い構成ファイルを検査すると、その情報が廃止されたことを知らせる警告メッセージが生成されます。

ファイル

```
/var/ld/ld.config
    32 ビットアプリケーション用のデフォルトの構成ファイル。

/var/ld/64/ld.config
    64 ビットアプリケーション用のデフォルトの構成ファイル。

/var/tmp
    一時構成ファイルのデフォルトの場所。 tempnam\(3C\) を参照してください。

/usr/lib/lddstub
    dldump\(3C\) 32 ビットオブジェクトに対して使用されるスタブアプリケーション。

/usr/lib/64/lddstub
    dldump\(3C\) 64 ビットオブジェクトに対して使用されるスタブアプリケーション。

/usr/lib/libcrle.so.1
    dldump\(3C\) 32 ビットオブジェクトに対して使用される監査ライブラリ。

/usr/lib/64/libcrle.so.1
    dldump\(3C\) 64 ビットオブジェクトに対して使用される監査ライブラリ。
```

環境変数

`crle` によって参照される環境変数はありません。ただし、いくつかの環境変数は、`crle` によって作成される構成ファイルの処理に関する実行時リンカーの動作に影響を及ぼします。

```
LD_CONFIG、LD_CONFIG_32、および LD_CONFIG_64
    代替構成ファイルを指定します。

LD_NOCONFIG、LD_NOCONFIG_32、および LD_NOCONFIG_64
    構成ファイルの処理を無効にします。
```

LD_NODIRCONFIG、LD_NODIRCONFIG_32、およびLD_NODIRCONFIG_64
構成ファイルからのディレクトリキャッシュ処理を無効にします。

LD_NOENVCONFIG、LD_NOENVCONFIG_32、およびLD_NOENVCONFIG_64
構成ファイルからの環境変数処理を無効にします。

LD_NOOBJALTER、LD_NOOBJALTER_32、およびLD_NOOBJALTER_64
構成ファイルからの代替オブジェクト処理を無効にします。

属性

属性についての詳細は、[attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/linker
インタフェースの安定性	確実

関連項目

[file\(1\)](#), [ld\(1\)](#), [ld.so.1\(1\)](#), [dldump\(3C\)](#), [tempnam\(3C\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

名前	crontab – ユーザーの crontab ファイル
形式	<pre>/usr/bin/crontab [filename] /usr/bin/crontab -e [username] /usr/bin/crontab -l [username] /usr/bin/crontab -r [username] /usr/xpg4/bin/crontab [filename] /usr/xpg4/bin/crontab -e [username] /usr/xpg4/bin/crontab -l [username] /usr/xpg4/bin/crontab -r [username] /usr/xpg6/bin/crontab [filename] /usr/xpg6/bin/crontab -e [username] /usr/xpg6/bin/crontab -l [username] /usr/xpg6/bin/crontab -r [username]</pre>
機能説明	crontab ユーティリティーは、crontab ファイルをコピー、作成、表示、削除することにより、cron(cron(1M)) を使ってユーザーのアクセスを管理します。オプションなしで crontab を実行すると、指定したファイル(ファイルを指定しない場合は標準入力)を、全ユーザーの crontab が登録されているディレクトリにコピーします。 crontab にファイル名 (<i>filename</i>) を指定して実行すると、コマンドを実行したユーザーの crontab エントリが上書きされます。
アクセス制御	ユーザー: 次の場合 crontab にアクセスできます。 ■ /etc/cron.d/cron.allow に自分の名前が登録されている場合 ■ /etc/cron.d/cron.allow が存在せず、/etc/cron.d/cron.deny に自分の名前が登録されていない場合 ユーザー: 次の場合 crontab にアクセスできません。 ■ /etc/cron.d/cron.allow が存在しており、かつ自分の名前がそこに登録されていない場合 ■ /etc/cron.d/cron.allow が存在せず、/etc/cron.d/cron.deny に自分の名前が登録されている場合 ■ /etc/cron.d/cron.allow も /etc/cron.d/cron.deny も存在しない場合、ジョブを送ることができるのは、solaris.jobs.user の承認を受けているユーザーだけです。

- Solaris 監査が有効な場合、ユーザーのシェルは監査されず、ユーザーは `crontab` の所有者になりません。これは、ユーザーが SSH のいずれかのバージョンなど、監査パラメータを設定しないプログラムを介してログインする場合に起こります。

なお `allow` または `deny` の規則が `root` に適用されるのは、`allow` または `deny` ファイルが存在している場合だけです。

`allow` と `deny` の両ファイルは、いずれも 1 行に 1 ユーザー名が記述される形式になっています。

入力形式

`crontab` ファイル内の各行は、6 つのフィールドで構成されています。各フィールドの間は、スペースまたはタブで区切られています。最初の 5 つのフィールドは、次の内容を指定する整数パターンです。

```
minute (0-59),
hour (0-23),
day of the month (1-31),
month of the year (1-12),
day of the week (0-6 with 0=Sunday).
```

整数値の代わりに、アスタリスク (有効な値全部を表す) や、コンマで区切った形式の要素リストを指定することもできます。要素は、数値 1 個で指定しますが、ある数値からある数値までの範囲を指定する場合は、数値 2 個をマイナス記号 (-) で区切って指定します。ここで指定された時間は、現在有効なタイムゾーンで解釈されます。`crontab` ファイルの先頭には、`/etc/default/init` 内でシステム全体に設定されているタイムゾーンが記述されています。ユーザーは、次のように記述して行を追加できます。

```
TZ=timezone
```

すべての後続エントリは、新たな `TZ=timezone` 行が現れるまで、そのタイムゾーンを使って解釈されます。日付は、2 つのフィールド (日および曜日) で指定できます。どちらのフィールドも、要素のリストとして指定された場合に付加されません。「使用例」を参照してください。

`crontab` ファイル内の行の 6 番目のフィールドは、指定した時間にシェルによって実行されるコマンド文字列です。このフィールド内の % 文字 (\ によってエスケープされているものは除く) は、復帰改行 (NEWLINE) 文字に変換されます。

シェルによって実行されるのは、このコマンドフィールドの第 1 行 (' % ' または行の終わりまで) のみです。その他の行は、標準入力として実行されます。空行、または ' # ' で始まる行はコメントであり、無視されます。

シェルは `$HOME` ディレクトリから呼び出されます。`$TZ` の場合と同様に、`$SHELL` と `$HOME` のどちらも次の行を記述することで設定できます。

```
SHELL=/usr/bin/someshell
```

または

```
HOME=somedirectory
```

これによって、別の HOME または SHELL エントリが現れるまで、crontab 内の残りのエントリすべてで優先されます。この呼び出しには、現在有効な \$SHELL のベース名の `arg0` が使用されます。`.profile` や同等のファイルを実行するユーザーは、crontab ファイル内でそのことを明示的に指定する必要があります。cron は、HOME、LOGNAME、SHELL、TZ、PATH を定義し、すべてのシェルのデフォルト環境を指定します。ユーザーの cron ジョブのデフォルト PATH は `/usr/bin` で、root の cron ジョブのデフォルトは `/usr/sbin:/usr/bin` です。デフォルトパスは `/etc/default/cron` 中に設定できます ([cron\(1M\)](#) を参照)。TZ、HOME、および SHELL 環境変数は、その時点で crontab ファイル内で有効なこれらの環境変数に合わせて設定します。

コマンドの標準出力や標準エラー出力をリダイレクトするように指定していないと、生成されたすべての出力またはエラーが、ユーザーにメールで通知されることになります。

crontab 環境変数

次の変数がサポートされています。

HOME

代替ディレクトリの選択をユーザーに許可して、コマンドの実行前に cron がディレクトリを変更できるようにします。例:

```
HOME=/var/tmp
```

SHELL

後続のコマンドの実行に使用するシェルの名前。例:

```
SHELL=/usr/bin/ksh
```

TZ

cron のエントリを実行するタイムゾーンの選択をユーザーに許可します。これは、実行するコマンドの環境と、エントリのタイミングの両方に影響を及ぼします。たとえば、アイスランドのタイムゾーンでエントリを実行するには、次のように指定します。

```
TZ=Iceland
```

これらの各変数は、その変数をリセットする後続行によって変数がリセットされるまで、crontab ファイル内の後続行すべてに影響を及ぼします。このため、1 つの crontab ファイル内で複数のタイムゾーンを指定することが可能です。

これらの環境変数を設定しない行は、UNIX 標準に準拠した crontab エントリと同じです。UNIX 標準に準拠した crontab エントリについては、このマニュアルページのほかの部分に記載されています。

複数のタイム
ゾーンの cron
ジョブを設定する

cron デーモンのデフォルトタイムゾーンは、cron エントリのシステム全体のタイムゾーンとして設定されます。また、さらに /etc/default/init を使用してシステム全体のデフォルトタイムゾーンとして設定されます。

夏時間や冬時間などが有効になっている場合には、切り替え期間に予定されているジョブは、1 回だけ実行されたり、2 回実行されたり、またはまったく実行されないこともあります。

オプション

サポートしているオプションは、次のとおりです。

-e 現在のユーザーの crontab ファイルのコピーを編集します。crontab が存在しない場合は、空のファイルを作成して編集します。編集が終了すると、このファイルがユーザーの crontab ファイルとしてインストールされます。

-e オプションを指定してファイルを編集するときに起動するエディタは、環境変数 EDITOR によって決まります。crontab のジョブを登録するときは、必ず crontab を使用してください。crontab ファイルを直接編集してジョブを追加しないでください。これは、cron はこの方法による変更を認識しないためです。

crontab ファイル内のすべての行を削除すると、古い crontab ファイルが復元されます。すべての行を削除する正しい方法は、-r オプションを使用して crontab ファイルを削除する方法です。

username が指定された場合、現在のユーザーの crontab ファイルではなく、指定のユーザーの crontab ファイルを編集します。これを行えるのは、root と solaris.jobs.admin 承認を持つユーザーだけです。

-l crontab を起動したユーザーの crontab ファイルの内容を表示します。-l オプションのあとにユーザー名を指定して、指定のユーザーの crontab ファイルを表示できるのは、root または solaris.jobs.admin の承認を受けているユーザーだけです。

-r crontab ディレクトリからユーザーの crontab を削除します。r オプションのあとにユーザー名を指定して、指定のユーザーの -crontab ファイルを削除できるのは、root または solaris.jobs.admin の承認を受けているユーザーだけです。

使用例

例1 core ファイルを削除する

平日 (月 - 金) の午前 3 時 15 分に core ファイルを削除する例です。

```
15 3 * * 1-5 find $HOME -name core 2>/dev/null | xargs rm -f
```

例2 誕生日のお祝いを送る

次の例では、誕生日のお祝いメールが送信されます。

```
0 12 14 2 * mailx john%Happy Birthday!%Time for lunch.
```

例3 日付と曜日を同時に指定する

この例では、毎月1日と15日、さらに毎週月曜日にコマンドが実行されます。

```
0 0 1,15 * 1
```

曜日だけ、もしくは日付だけを指定する場合には、もう一方のフィールドに*を指定します。例:

```
0 0 * * 1
```

毎週月曜日にコマンドが実行されます。

例4 環境変数を使用する

次のエントリは、crontab による特定の環境変数のサポートを利用しています。

```
TZ=GMT
HOME=/local/home/user
SHELL=/usr/bin/ksh
0 0 * * * echo $(date) > midnight.GMT
TZ=US/Pacific
0 0 * * * echo $(date) > midnight.PST
TZ=US/Eastern
HOME=/local/home/myuser
SHELL=/bin/csh
```

先行するエントリにより、2つのジョブが実行されます。1番目はGMTタイムゾーンの午前0時に実行され、2番目はPSTタイムゾーンの午前0時に実行されます。両方とも、Korn シェルを使用してディレクトリ /local/home/user 内で実行されます。このファイルでは、最終的にTZ、HOME、およびSHELL エントリにより、これらの変数がデフォルト値に戻されます。

環境変数

crontab の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

/usr/bin/crontab

EDITOR -e オプションを指定したときに呼び出すエディタを指定します。VISUAL 環境変数がこの環境変数よりも優先されます。デフォルトのエディタは [vi\(1\)](#) です。

PATH crontab の環境内の PATH には、エディタの検索に使用する検索パスを指定します。

VISUAL -e オプションを指定したときに呼び出すビジュアルエディタを指定します。VISUAL が指定されていない場合、環境変数 EDITOR で指定されたエディタが使用されます。この設定がされていない場合、デフォルトは [vi\(1\)](#) です。

`/usr/xpg4/bin/crontabEDITOR` `-e` オプションを指定したときに呼び出すエディタを指定します。デフォルトエディタは、`/usr/xpg4/bin/vi` です。

`/usr/xpg6/bin/crontabEDITOR` `-e` オプションを指定したときに呼び出すエディタを指定します。デフォルトエディタは、`/usr/xpg6/bin/vi` です。

終了ステータス 次の終了ステータスが返されます。

- `0` 正常終了。
- `>0` エラーが発生した。

ファイル	<code>/etc/cron.d</code>	主 <code>cron</code> ディレクトリ
	<code>/etc/cron.d/cron.allow</code>	許可されているユーザーのリスト
	<code>/etc/default/cron</code>	<code>cron</code> のデフォルト設定を含む
	<code>/etc/cron.d/cron.deny</code>	許可されていないユーザーのリスト
	<code>/var/cron/log</code>	アカウンティング情報
	<code>/var/spool/cron/crontabs</code>	<code>crontab</code> のスプール空間

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

<code>/usr/bin/crontab</code>	属性タイプ	属性値
	使用条件	<code>system/core-os</code>
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

<code>/usr/xpg4/bin/crontab</code>	属性タイプ	属性値
	使用条件	<code>system/xopen/xcu4</code>
	インタフェースの安定性	標準

<code>/usr/xpg6/bin/crontab</code>	属性タイプ	属性値
	使用条件	<code>system/xopen/xcu6</code>
	インタフェースの安定性	標準

関連項目 [atq\(1\)](#), [atrm\(1\)](#), [auths\(1\)](#), [ed\(1\)](#), [sh\(1\)](#), [vi\(1\)](#), [cron\(1M\)](#), [su\(1M\)](#), [auth_attr\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

誤って引数なしで `crontab` コマンドを入力した場合、CTRL-D を使って処理を中止しないでください。これにより、`crontab` ファイル内のすべてのエントリが削除されます。処理の中止には、CTRL-C を使用してください。

`cron` を更新するときには、まず、既存の `crontab` を調べて、更新時近くにイベントがスケジュールされているかどうかをチェックします。イベントが予定されていた時刻より後に更新プロセスが完了すると、当該エントリが失われる可能性があります。この現象は次のように発生します。ユーザーの `crontab` ファイルの内部ビューを更新するように `crontab` から通知されると、`cron` はまず、既存の `crontab` ファイルの内部ビューとスケジュールされている内部イベントを削除します。次に、新しい `crontab` ファイルを読み取って、`crontab` の内部ビューと内部イベントを構築し直します。この最後の段階には (特に、`crontab` ファイルが大きいときには) 時間がかかります。既存の `crontab` エントリが更新時のきわめて近くに予定されていると、予定された時間のあとに最後の段階が完了することがあります。安全を期すために、新しいジョブは少なくとも現在の日付と時間の 60 秒後に起動するようにしてください。

1 つの `crontab` ファイルを一度に複数変更すると、予期しない結果になる可能性があります。

TZ、SHELL、および HOME 変数を `crontab` ファイルに追加する際、これらの変数をデフォルト以外に変更することが想定されていないアプリケーションで `crontab` ファイルが共有される可能性がある場合は注意が必要です。ファイルの末尾で値をデフォルトにリセットすると、問題が発生する危険を最小限に抑えることができます。

名前	csh - C に似た構文を持つシェル・コマンドインタプリタ
形式	csh [-bcefinstvVxX] [argument]...
機能説明	csh は、C シェルの略で、C 言語に似た構文を持つコマンドインタプリタです。Bourne シェルにはない多くの便利な対話型の機能を備えています。たとえば、ファイル名の補完、コマンド別名指定、履歴置換、ジョブ制御、多くの組み込みコマンドなどの機能です。Bourne シェルと同様に、C シェルは変数、コマンド、およびファイル名置換機能を提供します。
初期化と終了	C シェルを起動すると、まず通常ホームディレクトリにある <code>.cshrc</code> が読み取り可能であるか、当該ユーザーがその所有者であるか、または当該ユーザーのグループ ID がそのグループ ID と一致するかチェックし、一致していれば <code>.cshrc</code> ファイルを実行します。もし、C シェルを <code>'</code> で始まる名前前で起動すると、そのシェルはログインシェルとして実行されます (login(1) を起動するときの動作と同じ)。 シェルがログインシェルであった場合、実行は連続して行われます。最初に <code>/etc/.login</code> 中のコマンドが実行されます。次にユーザーのホームディレクトリにある <code>.cshrc</code> ファイルのコマンドが実行されます。その後、シェルはユーザーのホームディレクトリにある <code>.login</code> ファイルのコマンドを実行します。このとき、このファイルにも <code>.cshrc</code> と同様のアクセス権チェックが実施されます。通常 <code>.login</code> ファイルには、端末の種類や環境を指定するコマンドが入っています。(ファイルのインタプリタの説明については、「コマンドの実行」と exec(2) を参照。) ログインシェルが終了したとき、シェルはユーザーのホームディレクトリにある <code>.logout</code> ファイルのコマンドを実行します。このとき、このファイルにも <code>.cshrc</code> と同様のアクセス権チェックが実施されます。
対話型動作	起動処理が終了すると、C シェルはプロンプト <code>hostname%</code> (特権ユーザーなら <code>hostname#</code>) を表示して、端末からコマンドの読み取りを開始します。以降、シェルは以下の処理を繰り返します。まず入力コマンドを 1 行読み取り、複数のワードに分解します。次にこれらのワードを履歴に記憶し、構文解析します (下記の「使用方法」を参照)。最後に、現在の行にある各コマンドを実行します。
非対話型動作	非対話形式で動作しているとき、シェルは端末からの入力を促すプロンプトを表示しません。非対話形式の C シェルは、コマンド行で引数として与えられたコマンドを実行するか、スクリプトと呼ばれるファイルからコマンドを読み取り、実行します。
オプション	サポートしているオプションは、次のとおりです。 -b オプションの処理を強制的に中断します。以降のコマンド行引数は、C シェルのオプションとしては解釈されません。これによって、オプションを明確にスクリプトに渡すことができます。このオプションが設定されていなければ、シェルは <code>set-user-ID</code> または <code>set-group-ID</code> スクリプトを実行しません。

- c 最初の引数(必ず指定する)を実行します。残りの引数は、引数リスト変数 `argv` に格納され、`csh` に直接渡されます。
- e コマンドが異常終了したり、0以外の終了ステータスを返したとき、処理を終了します。
- f 高速起動。開始時に、`.cshrc` ファイルおよび `.login` ファイル(ログインシェルするとき)を読みません。
- i 強制的に対話型にします。標準入力端末(文字特殊装置)でなくても、コマンド行入力を促すプロンプトを表示します。
- n 構文解析はしますが、コマンドは実行しません。このオプションは、Cシェルスクリプトの構文エラーをチェックするのに使用できます。
- s 標準入力からコマンドを受け取ります。
- t コマンド行を1行読み取って実行します。後続の入力行へコマンド行が継続することを示すには、復帰改行(NEWLINE)を'\'(バックスラッシュ)でエスケープします。
- v 冗長。定義済み変数 `verbose` を設定します。コマンド入力のエコーを、履歴置換後(他の置換の前)かつ実行前に行います。
- V `.cshrc` を読み取る前に `verbose` 変数を設定します。
- x エコー。`echo` 変数を設定します。すべての置換後、実行の前にコマンドをエコーします。
- X `.cshrc` を読み取る前に `echo` 変数を設定します。

-c、-i、-s、および -t の各オプションを除き、最初の必須 *argument* はコマンド名またはスクリプト名とみなされます。そして0番目の引数として渡され、後続の引数はそのコマンドまたはスクリプトの引数リストへ追加されます。

使用法

ファイル名の補完

`filec` 変数を設定することによって、対話型Cシェルは、部分的に入力されたファイル名またはユーザー名を補完することができます。ファイル名を部分的にあいまいに入力してESC文字を入力すると、シェルは作業用ディレクトリから一致するファイルを検索して、残りの文字を埋めます。

ファイル名を部分的に入力してEOF文字(通常はCTRL-D)を入力すると、シェルは一致するすべてのファイル名を一覧表示します。そして、入力された不完全なコマンド行の内容を表示して、再度プロンプトを表示します。

最後のワード(入力値の一部)がチルド(~)で始まる場合、シェルは現在のディレクトリのファイル名ではなく、ユーザー名を補完しようとします。

端末は、ピープ音を鳴らすことによって、エラーまたは複数のファイル名が一致したことを知らせます。この音は、`nobeep` 変数を設定することによって抑止できます。`ignore` 変数に特定の接尾辞のリストを設定しておけば、その接尾辞を持つファイルを除外することができます。ただし、その接尾辞を持つファイル名以外に一致するものが見つからなければ、この変数は無視されます。なお EOF 文字によるファイル名のリスト出力には、`ignore` が示す除外対象ファイル名も含まれます。

字句構造

シェルは、入力行をスペース文字およびタブでワードに分解します。ただし、以下に説明する例外を除きます。`&`、`|`、`;`、`<`、`>`、`(`、および `)` の各文字は、それぞれ別個のワードを形成します。ただし対で使用すると、1 対で 1 つのワードを形成します。これらのシェルメタキャラクタは、他のワードの一部になることもできます。その場合、`\` (バックスラッシュ) を前に付けると、メタキャラクタの持つ特別な意味を抑制できます。`\` の付いた復帰改行文字は、スペース文字と同等です。

さらに、単一引用符 (`'`)、二重引用符 (`"`)、または逆引用符 (```) で囲まれた文字列は、ワードの一部を生成します。このような文字列の中に含まれるスペース文字またはタブを含めたメタキャラクタは、別個のワードを形成しません。逆引用符 (```) または二重引用符 (`"`) の対で囲まれた場合、`\` (バックスラッシュ) の付いた復帰改行は本当の意味での復帰改行文字です。各引用符のその他の機能については、後述の「変数置換」、「コマンド置換」、「ファイル名置換」の各項で説明しています。

シェルの入力端末ではない場合、`#` 文字は、入力行が最後までコメント行であることを表します。`\` が前に付いたり、一対の引用符で囲まれると、その特別な意味は抑制されます。

コマンド行構文解析

単純コマンド (*simple command*) は、一連のワードで構成されます。最初のワード (リダイレクションの一部ではない) は、実行するコマンドを指定します。1 つの単純コマンド、または、`|` や `|&` 文字で区切られた一連の単純コマンド群は、パイプライン (*pipeline*) を形成します。`|` を使用すると、直前のコマンドの標準出力が、それに続くコマンドの標準入力に変更 (リダイレクション) されます。`|&` を使用すると、標準エラー出力および標準出力の両方の出力先が、パイプラインを介してリダイレクションされます。

パイプラインはセミコロン (`;`) で区切ることができます。この場合、それらは順次実行されます。`&&` または `|` で区切られたパイプラインは、条件に従って実行されます。つまり左側のパイプラインの実行が成功するか失敗するかによって、右側のパイプラインが実行されるかどうかが決まります。

1 つのパイプラインまたはパイプラインの並びを、括弧 `"` で囲むことができます。囲まれた全体が単純コマンドとなり、他のパイプラインまたはパイプラインの並びのコンポーネントとなることができます。

パイプラインの並びは、`&` を付けることによって非同期的にもしくは「バックグラウンドで」実行できます。この場合シェルは、プロンプトの出力をパイプライン

の並びが終了するまで待ちません。ただちにジョブ番号 (後述の「ジョブ制御」を参照) と関連するプロセス ID を表示してから プロンプトを表示します。

履歴置換

履歴置換を使用すれば、以前入力したコマンド行のワードを、これから入力するコマンド行で使用できます。これにより、綴りの訂正、複雑なコマンドや引数の繰り返し入力が簡単になります。コマンド行は履歴リストに保存されます。履歴リストのサイズは `history` 変数によって変更できます。どのような場合でも、最後に入力されたコマンドは必ず保持されます。履歴置換の指定は `!` で始まり (`histchars` 変数により別の文字に変更可能)、コマンド行のどこに現われてもかまいません。ただし履歴置換のネストはできません。`!` を `\` でエスケープすれば、その特別な意味を抑制できます。

履歴置換を含んだ入力行は、展開された後で、他の置換が起こる前またはコマンドが実行される前に、端末上に表示されます。

イベント指示子

イベント指示子は、履歴リスト内の コマンド行エントリを参照するものです。

<code>!</code>	次の文字がスペース文字、タブ、復帰改行、 <code>=</code> 、または <code>(</code> でなければ、履歴置換を開始します。
<code>!!</code>	直前のコマンドを指します。他の文字を加えずにこれだけを入力すると、直前のコマンドを繰り返します。
<code>!n</code>	<i>n</i> 番のコマンド行を指します。
<code>!-n</code>	入力中のコマンドから <i>n</i> 個前のコマンドを指します。
<code>!str</code>	<i>str</i> で始まる最新のコマンドを指します。
<code>!?str?</code>	<i>str</i> を含む最新のコマンドを指します。
<code>!?str? additional</code>	<i>str</i> を含む最新のコマンドを指し、その参照コマンドに <i>additional</i> を追加します。
<code>!{command} additional</code>	<i>command</i> で始まる最新のコマンドを指し、その参照コマンドに <i>additional</i> を追加します。
<code>^previous_word^replacement^</code>	文字列 <i>previous_word</i> を文字列 <i>replacement</i> に置き換えて、直前のコマンド行を繰り返します。これは履歴置換と同等です。 <code>!:s/previous_word/replacement/.</code> 以前の特定のコマンドを再実行し、さらに同じような置換を行う場合、たとえば 6 番目のコマンドを再実行するには、次のようにします。 <code>!:6s/previous_word/replacement/.</code>

ワード指示子	A ':' (コロン) はイベント指示子とワード指示子とを区切ります。ワード指示子が ^、\$、*、-、または % で始まるときは省略できます。直前のコマンドからワードを選択する場合、イベント指示子の 2 番目の ! は省略できます。たとえば、!!:1 と !!:1 は、両方とも直前のコマンドの最初のワードを指します。また、!!\$ と !\$ は、両方とも直前のコマンドの最後のワードを指します。ワード指示子には以下のものがあります。 # 今までに入力したすべてのコマンド行 0 最初に入力したワード (コマンド) <i>n</i> <i>n</i> 番目の引数 ^ 最初の引数。1 と同じ \$ 最後の引数 % 最新の ?s による検索で一致したワード <i>x</i>-<i>y</i> ワードの範囲。-<i>y</i> は 0-<i>y</i> の省略形 * すべての引数。イベント中に 1 ワードしか存在しないときは NULL <i>x</i>* <i>x</i>-\$ の省略形 <i>x</i>- <i>x</i>* と同じであるが、ワード \$ を省略
修飾子	オプションのワード指示子の後に、: で始まる修飾子のうちの 1 つを追加できます。 h パス名の後部分のコンポーネントを除去して、前部分を残します。 r '.xxx' の形の接尾辞を除去して、ベース名を残します。 e 接尾辞以外はすべて除去して、拡張部分を残します。 s/<i>l</i>/<i>r</i>/ <i>l</i> を <i>r</i> で置き換えます。 t パス名の前部分のコンポーネントを除去して、後部分を残します。 & 前の置換を繰り返します。 g 各ワード内の最初の一致が発生した箇所を、前述のオプションに接頭辞を付けて変更します (たとえば g&)。 p 新しいコマンドを表示するだけで、実行はしません。 q 置換されたワードをクォートして、それ以上の置換をエスケープします。 x q と同じですが、スペース文字、タブ、復帰改行文字 (NEWLINE) ごとにワードに分割します。

`g` を先頭に付加しないと、`l` に一致する最初の文字列だけが変更されます。一致する文字列がなければ、エラーとなります。

置換部分の左側は正規表現ではなく文字列です。`/` の箇所には、区切り文字としてどのような文字でも使用できます。区切り文字用の文字はバックスラッシュで囲まれます。右側の `&` 文字は、左側のテキストで置換されます。`&` はバックスラッシュでクォートすることができます。`l` が `NULL` のとき、直前の文字列における `l`、または `!?` における文脈検索文字列 `s` を使用します。同様に `r` の直後に復帰改行がある場合、文脈検索の最右にある `?` は省略できます。

イベントが指定されないと、履歴リファレンスは前のコマンドか、(もしあれば) そのコマンド行上での前の履歴リファレンスを参照します。

高速置換 `^lr^` これは履歴置換と同等です。

`!:s/l/r/.`

別名(エイリアス) C シェルが持っている別名のリストは、`alias` および `unalias` コマンドを使用してユーザーが作成、表示、および変更できます。シェルは、各コマンドの最初のワードが既存の別名に一致するかどうかをチェックします。一致すれば、そのワードを別名に置き換えて再度コマンド処理を実行します。履歴置換のメカニズムは、そのコマンドが前の入力行であったかのように行うことが可能です。これによって履歴置換は(定義中ではバックスラッシュでエスケープされる)、別名が使用されているとき、実際のコマンド行引数で置き換えることができます。履歴置換が1度も呼ばれていなければ、引数は変更されません。

別名はネストできます。すなわち、別名の定義の中に別の別名を入れてもかまいません。ネストされた別名は履歴置換が実行される前に展開されます。これは次のようなパイプラインに役立ちます。

`alias lm 'ls -l \!* | more'`

これは呼び出されたときに、`ls(1)` の出力を `more(1)` にパイプするものです。

最初のワードを除いて、別名を自分自身の定義内、また定義が参照しているほかのどの別名内にも書いてはいけません。このようなループが見つかったら、エラーメッセージが表示されます。

I/O リダイレクション

以下のメタキャラクタは、それに続くワードが、コマンドの標準入力、標準出力、標準エラーのリダイレクション先のファイル名であることを示します。このワードは、コマンドの他の部分とは別個に展開されたファイル名、変数、またはコマンドです。

`<` 標準入力をリダイレクションします。

`< < word` 標準入力を `word` と一致する行まで読み取り、それらの行を一時ファイルに格納します。`word` がエスケープされるかクォートされていないければ、格納された行に対して、変数およびコマンド置換が行わ

れます。そして、一時ファイルを標準入力としてパイプラインが起動されます。*word* は変数、ファイル名、およびコマンド置換の対象にはなりません。また、各行はシェルによって置換が実行される前に *word* と比較されます。

>>! >& >&! ファイルへの標準出力をリダイレクションします。ファイルが存在しなければ作成します。存在すれば上書きします。このとき、以前の内容は失われます。

変数 *noclobber* が設定されていれば、既存のファイルを破壊しません。この変数は、*!* 形式のいずれかが使用されていないかぎり、端末および */dev/null* へのリダイレクションも防ぎます。*&* 形式は、標準出力および標準エラー (診断出力) の両方の出力先を、指定ファイルに変更します。

> >> >& >! > >&! 標準出力へ追加します。*>* と似ていますが、上書きするのではなくファイルの最後に追加します。*noclobber* が設定されていると、*!* 形式のうちの1つを使用しないかぎり、存在しないファイルに対してはエラーになります。*&* 形式は、標準出力および標準エラーの両方をファイルへ追加します。

変数置換

C シェルには一群の変数が備わっており、それぞれは名前と値の対で構成されています。変数には最大 128 個の文字と数字を含めることができます。先頭は文字にします。下線 (*_*) は文字と見なされます。変数の値は、空白で区切られた 0 個以上のワードです。シェルが 128 文字以内の変数をサポートしている場合は、変数 *SUNW_VARLEN* が定義されます。128 文字までの変数名をサポートしていない場合は、以前のバージョンのシェルが使用されており、シェル変数名の長さは最大 20 文字になります。

値を参照するには、変数名の前に *\$* を付けます。(以下に述べる) ある種の参照は、値から特定のワードを選択したり、変数に関する別の情報を表示したりするのに使用できます。括弧は、入力行の他の文字から参照を区別するために使用できます。

変数置換は、入力行の解析、別名の展開、入出力先のリダイレクションが行われた後に実行されます例外は、入出力先のリダイレクションの変数参照 (リダイレクション中に置換される) および逆引用符で囲まれた文字列 (「コマンド置換」の項を参照) です。

\$ の前に ** を付加すれば、変数の置換を抑制できます。ただし、二重引用符で囲まれた中では常に置換されます。単一引用符に囲まれていれば、変数置換は抑制されます。スペース文字、タブ、または復帰改行が続く場合は、*\$* はエスケープされます。

変数を生成、表示、および破壊するためには、*set* または *unset* コマンドを使用します。いくつかの変数はシェルによって管理または使用されます。たとえば *argv*

変数には、シェルの引数リストのイメージが入っています。シェルに使用される変数の多くはトグルです。つまり、シェルはその変数の値を知る必要はなく、単にその変数が設定されているか否かを知っているだけです。

数値の値は、数字として ((@) 組み込みコマンドで扱うように) 処理できます。数値処理では、空の値は 0 とみなされます。複数ワードからなる値の 2 番目以降のワードは無視されます。たとえば `verbose` 変数が何らかの値 (空の値も含む) に設定されていれば、コマンド入力は端末にエコーされます。

コマンドおよびファイル名の置換は、変数置換の結果生成されたワードに適用されます。ただし、二重引用符で抑制されている場合、`noglob` が設定されている場合 (ファイル名置換の抑制)、および `:q` 修飾子で参照がクォートされている場合を除きます。二重引用符の中では、参照が展開され、クォートされた文字列 (の一部) を形成します。つまり、複数ワードからなる値は、スペース文字を埋め込んだ 1 つの文字列に展開されます。`:q` 修飾子を参照に使用すると、スペースで区切られたワードに展開され、各ワードにはそれ以降のコマンドまたはファイル名の置換を防ぐために、クォートされます。

後述する例外を除き、設定されていない変数を参照するとエラーになります。

`$var`

`${var}`

`var` が示す変数の値から得られる、スペース文字で区切られたワード群に置き換えられます。`var` が環境変数の場合、その値が返されます (ただし `:` 修飾子と以下に述べる他の形式は使用できない)。

`$var[index]`

`${var[index]}`

`var` が示す変数の値から、特定のワードだけを選択します。`index` は 1 つの数字、`-` で区切られた 2 つの数字、またはアスタリスクで、これに対して変数置換が行われます。ワード群には 1 から始まるインデックス値が付けられています。`*` はすべてのワードを選択します。もし (`$argv[-2]` のように) 範囲の最初の数字を省略すると、デフォルトとして 1 とみなされます。もし (`$argv[1-]` のように) 範囲の最後の数字を省略すると、デフォルトとして `$#var` (ワード数) とみなされます。第 2 引数が省略されている (または範囲内の) ときは、範囲が空であってもエラーにはなりません。

`$#name`

`${#name}`

変数内のワード数を指定します。

`$0`

`setuid` シェルスクリプトを除いてはコマンドを読み取っているファイル名に置き換わります。ファイルの名前がわからないときはエラーになります。

`$n`

`${n}`

`$argv[n]` と同等です。

`$*` `$argv[*]` と同等です。

`:gh`、`:gt`、および `:gr` のように、修飾子 `:e`、`:h`、`:q`、`:r`、`:t`、および `:x` が適用できます (「履歴置換」の項を参照)。中括弧 `{ }` を使用するときは、修飾子の中括弧内に書きます。現在のインプリメントでは、このような修飾子を 1 展開あたり 1 つだけ認めています。

次の参照は、修飾子 `:` を使って修飾することはできません。

`$?var`
 `${?var}` `var` が設定されていれば文字列 1 に、設定されていなければ文字列 0 に置換します。

`$?0` 現在の入力ファイル名がわかっているときは文字列 1 に、わかっていなければ文字列 0 に置換します。

`$$` (親) シェルのプロセス番号に置換します。

`$<` 標準入力から得る行の内容に置換し、それ以降は何も解釈しません。キーボードからの入力を C シェルスクリプトで読み取るときに使用できます。

コマンドおよび ファイル名置換

コマンドおよびファイル名置換は、組み込みコマンドの引数に選択的に適用されます。評価されていない表現は、展開されません。非組み込みコマンドについては、コマンド名のファイル名展開と、引数リストのファイル名展開は別々に実行されます。展開は、入出力先のリダイレクションが終わった後、サブシェル内で行われます。

コマンド置換

逆引用符で囲まれた (`'...'`) コマンドはサブシェルによって実行されます。その標準出力はスペース文字、タブ、復帰改行の箇所でワードに分解されます。空のワードは捨てられます。この分解されたテキストは、現在のコマンド行上の逆引用符で囲まれた文字列を置き換えます。二重引用符で囲まれると、復帰改行文字の箇所だけでワードに分解され、スペース文字、タブはそのまま保存されます。ただし、最後の復帰改行は無視されます。したがって、コマンド置換がワードの一部を生成することも可能です。

ファイル名置換

`*`、`?`、`[`、または `{` のうちのいずれかの文字を含むクォートされていないワード、または `~` で始まるワードは、以下のようにアルファベット順にソートされたファイル名のリストに展開 (グローピングとも呼ばれる) されます。

`*` (0 個以上の) どの文字とも一致します。

`?` どの単一文字とも一致します。

`[...]` 括弧内の文字リストまたは範囲内のどの単一文字とも一致します。リストは文字列です。範囲とは、ダッシュ (`-`) で区切られた 2 つの文字を指し、ASCII の順で ([ascii\(5\)](#) を参照) その文字間にあるすべての文字も含みます。

`{str, str, ...}` コンマで区切られたリスト内の各文字列 (またはファイル名マッチングパターン) に展開します。この場合、前述のパターンマッチング表現とは違って、この構造の展開はソートされません。たとえば `{b,a}` は、`'a' 'b'` ではなく、`'b' 'a'` に展開されます。特別な例として、文字 `{および}` は、文字列 `{ }` と共に、展開されずに引き渡されます。

`~[user]` 変数 `home` の値によって示されるホームディレクトリ、または `user` のパスワードエントリによって示される `user` のホームディレクトリを意味します。

`*`、`?`、および `[...]` だけがパターンマッチングを意味します。パターンに一致するファイル名が存在しないときはエラーになります。`'.'` (ドット文字) は、それがファイル名またはパス名のコンポーネントの最初の文字である場合、明示的に一致しなければなりません。`/` (スラッシュ) も同様に明示的に一致させる必要があります。

式と演算子

多くの C シェルの組み込みコマンドは、式を受け入れます。この式の演算子は C の演算子と類似していて、その優先度も C での優先度と同じです。通常これらの式は `@`、`exit`、`if`、`set`、および `while` コマンド内に現われ、コマンド実行用のフロー制御の規定に使用します。式のコンポーネントはスペースで区切られます。

式の値が `NULL` または値がないと `0` とみなされます。式の結果はすべて文字列で、10 進数データも文字列で表現します。

C シェルの演算子を、以下に優先度の高いものから順にグループ分けして示します。

<code>(...)</code>	グループ化
<code>>~</code>	1 の補数
<code>!</code>	論理否定
<code>* / %</code>	乗算、除算、剰余 (これらは右結合なので、予想した結果が得られない場合があります。その場合、括弧を使って明示的にグループ化してください)。
<code>+ -</code>	加算、減算 (これらも右結合)
<code><< >></code>	ビット単位の左シフト、右シフト
<code>< > <= >=</code>	より小さい、より大きい、以下、以上
<code>= = != == !=</code>	等しい、等しくない、ファイル名置換パターン一致 (下記参照)、ファイル名置換パターン不一致
<code>&</code>	ビット単位の AND (論理積)
<code>^</code>	ビット単位の XOR (排他論理和)

	ビット単位の OR (包括論理和)
&&	AND (論理積)
	OR (論理和)

演算子 ==、!=、=~、および !~ は、引数を文字列として比較します。他の演算子は数字を使用します。演算子 =~ および !~ は、それぞれ左側の文字列が右側のファイル名置換パターンと一致するか否かをチェックします。これを使えば、文字列間のパターンマッチングだけ実行したいとき、switch 文を使用する必要がなくなります。

以下に示すようなファイルに関する照会も可能です。

-r filename	ユーザーが読み取り権を持っていれば真すなわち 1 を、そうでなければ偽すなわち 0 を返します。
-w filename	ユーザーが書き込み権を持っていれば、真です。
-x filename	ユーザーが実行権 (またはディレクトリの検索権) を持っていれば、真です。
-e filename	filename が存在していれば、真です。
-o filename	ユーザーが filename を所有していれば、真です。
-z filename	filename のサイズが 0 のとき、真です。
-f filename	filename がプレーンファイルなら、真です。
-d filename	filename がディレクトリなら、真です。

filename が存在しない、またはアクセスできない場合、すべての照会に対して偽を返します。

コマンドが成功したか否かの照会もできます。

{ command }	command が正常に動作していれば、この式は真すなわち 1 として、そうでなければ偽すなわち 0 として評価されます。注: 逆に、command 自体は正常に動作していれば通常 0 を、問題が発生すれば別の値を返します。ステータスを直接知りたければ、この式ではなく status 変数の値を使用します。

制御フロー

シェルはスクリプト内の制御フロー、および制限付きで端末からの制御フローを規定する多くのコマンドを持っています。これらのコマンドはシェルに入力を再度読み込ませるか (ループさせる)、ある条件下で入力をスキップさせる (分岐させる) ことによって動作します。

foreach、switch、while、if...then、および else の各組み込みコマンドは、入力行の最初のワードとして記述しなければなりません。

もし、シェルの入力が見つからないのにループが読み込まれているときは、その入力はバッファリングされます。シェルは、ループによって示される再読み込みを実行するために、内部バッファを検索します(この範囲において、逆方向の `goto` コマンドは、入力が見つからない場合でも成功します)。

コマンドの実行

コマンドがCシェルの組み込みコマンドの場合、シェルはそれを直接実行します。そうでない場合、シェルはその名前で実行可能なファイルを検索します。コマンド名が/を含んでいる場合、シェルはそれをパス名とみなし検索します。コマンド名が/を含んでいない場合、変数 `path` 内の各ディレクトリ内でコマンドを検索することによって、パス名として扱えるようにします。シェルは、検索を高速にするためにハッシュテーブルを使用して(`rehash` 組み込みコマンドの説明を参照)、該当するファイルが存在しないディレクトリを除外します。このハッシュは `-c` または `-t` オプション、または `unhash` 組み込みコマンドによって使用不能にできます。

特別な例として、スクリプト名に/が含まれず、ワード `shell` の別名が存在するときは、その式をコマンド行に(何も変更を加えずに)付加します。システムは、この特別な別名の最初のワードを実行します。このワードは、完全なパス名でなければなりません。別名定義中の残りのワードは、入力行のテキストと共に引数として扱われます。

適切な実行権を持つパス名が見つかり、シェルは新しいプロセスをフォークし、`execve()` システムコール([exec\(2\)](#)を参照)を使用して、そのパス名を引数と共にカーネルに引き渡します。カーネルは、希望するプログラムで新しいプロセスをオーバレイします。ファイルが実行可能なバイナリ([a.out\(4\)](#)形式)なら、カーネルは新しいプロセスを引き継ぎ、実行を開始します。ファイルがテキストファイルで、先頭行が `#!` で始まる場合、次のワードが、そのスクリプトを解釈するためのシェル(またはコマンド)のパス名とみなされます。先頭行の後続のワードは、そのシェルのオプションとみなされます。カーネルは、引数としてそのスクリプト名を使用し、指定されたシェルを起動(オーバレイ)します。

前述のいずれの条件にもあてはまらない場合は、カーネルはそのファイルをオーバレイできず、`execve()` コールは失敗します([exec\(2\)](#)を参照)。Cシェルは、以下に述べるように新しいシェルを起動して、そのファイルを実行しようとします。

- ファイルの先頭文字が `#` の場合、Cシェルを起動
- それ以外の場合は Bourne シェルを起動

シグナル処理

シェルは通常 `QUIT` シグナルを無視します。バックグラウンドジョブは、ハングアップ(`HUP`)などキーボードから生成されたシグナルを感知しません。他のシグナルはCシェルがその環境から引き継いだ値を持っています。シェルのスクリプト内における割り込みおよび終了シグナルの処理は、組み込みコマンド `onintr` で制御できます。`TERM` (終了)シグナルは、ログインシェルが受け取るか、あるいは子プロセスへ引き渡されます。ログインシェルが `.logout` ファイルを読み込んでいる間は、いかなる場合でも割り込みは許されません。

ジョブ制御

シェルは、番号の付いたジョブを各コマンドの並びと関連付けて、バックグラウンドで動作中のコマンド、または TSTP シグナル (通常は CTRL-Z) によって停止したコマンドの動作を追跡します。コマンドまたはコマンドの並び (セミコロンで区切られたリスト) をメタキャラクタ `&` を使用してバックグラウンドで起動した場合、シェルは角括弧で囲まれたジョブ番号が付いた行、および関連するプロセス番号のリストを表示します。

[1] 1234

現在のジョブリストを見るには、組み込みコマンド `jobs` を使用します。最後に停止したジョブ (停止したジョブがない場合は、最後にバックグラウンドに投入されたジョブ) を「現在のジョブ」といい、`+` で示します。前のジョブは `-` で示します。現在のジョブが終了したり、フォアグラウンドに移された場合、前のジョブが新たな現在のジョブになります。

ジョブを操作するには、組み込みコマンド `bg`、`fg`、`kill`、`stop`、および `%` を使用します

ジョブの参照は `%` で始まります。パーセント記号だけの指定は、現在のジョブを示します。

`%%+ %%` 現在のジョブ。

`%-` 前のジョブ。

`%j` 次のようにしてジョブ *j* を参照します。'`kill -9 %j`'。ここで、*j* はジョブ番号、またはジョブを起動したコマンド行を一意に表す文字列です。たとえば '`fg %vi`' は、停止した *vi* ジョブをフォアグラウンドに移します。

`%?string` コマンド行が *string* を含んでいるジョブ

バックグラウンドで動作中のジョブは、端末からの読み取り時に停止します。バックグラウンドジョブは、通常出力を生成しますが、'`stty tostop`' コマンドを使用して抑制することも可能です。

ステータスレポート

対話型で動作している場合、シェルは各ジョブのステータスを追跡し、ジョブが終了したり停止したりするとレポートを出力します。通常、入力表示を乱さないように、プロンプト出力時にメッセージを表示します。変数 `notify` が設定されているときは、シェルはステータスの変更を即座に報告します。デフォルトでは、`notify` コマンドは現在のプロセスをマークします。バックグラウンドのジョブの起動後に `notify` と入力すれば、そのジョブをマークできます。

コマンド

組み込みコマンドは C シェル内で実行されます。組み込みコマンドが、パイプラインのいずれかのコンポーネント (最後のコンポーネントを除く) として現われると、それはサブシェル内で実行されます。

`:` NULL コマンド。このコマンドは解釈されますが、実行はされません。

<code>alias [name [def]]</code>	別名 <i>name</i> に <i>def</i> を割り当てます。 <i>def</i> は一群のワードで、エスケープされた履歴置換のメタシンタックスを含めることができます。 <i>name</i> に <code>alias</code> または <code>unalias</code> を使用することはできません。 <i>def</i> を省略すると、現在の別名の定義 <i>name</i> が表示されます。 <i>name</i> と <i>def</i> の両方を省略すると、すべての別名と定義が表示されます。
<code>bg [%job ...]</code>	バックグラウンドで、現在のジョブまたは指定されたジョブを実行します。
<code>break</code>	<code>foreach</code> または <code>while</code> の最も内側のループの <code>end</code> の次から実行を再開します。現在の行の残りのコマンドが実行されません。これによって、複数レベルのブレイクを <code>break</code> コマンドのリストとして1行にすべて書き込めます。
<code>breaksw</code>	<code>switch</code> からブレイクして、 <code>endsw</code> の直後から再開します。
<code>case label:</code>	<code>switch</code> 文のラベル。
<code>cd [dir]</code> <code>chdir [dir]</code>	シェルの作業用ディレクトリを <i>dir</i> が示すディレクトリに変更します。引数を指定しないと、そのユーザーのホームディレクトリに変更します。 <i>dir</i> が現在のディレクトリには見つからない相対パス名の場合、変数 <code>cdpath</code> 内のディレクトリリストを検索します。 <i>dir</i> が / で始まる値を持つシェル変数の名前である場合、その変数の値が示すディレクトリに変更します。
<code>continue</code>	<code>while</code> または <code>foreach</code> の最も内側のループの、次の繰り返しから実行します。
<code>default:</code>	<code>switch</code> 文のデフォルトケースにラベルを付けます。デフォルトはすべての <code>case</code> ラベルの後に現れなければなりません。コマンド行の残りのコマンドが、最初に実行されます。
<code>dirs [-l]</code>	ディレクトリスタックを出力します。現在のディレクトリが最も左に現れるように時間順に出力されます。 <code>-l</code> 引数を指定すると、~ を使った省略形ではなく、完全な形式で出力されます。
<code>echo [-n] list</code>	<i>list</i> 内のワードをスペース文字で区切って、シェルの標準出力へ書き出します。オプション <code>-n</code> を指定しないと、出力は復帰改行で終了します。 <code>echo</code> が UNIX コマンドのフルパス名なしで呼び出された場合、ユーザーの <code>PATH</code> の構成に関係なく、デフォルトでは <code>csh</code> は、組み込みの <code>echo</code> を実行します (<code>echo(1)</code> を参照)。

<code>eval argument . . .</code>	引数をシェルへの入力として読み取り、それをコマンドとして実行します。通常この指定は、コマンドまたは変数置換の結果として生成されたコマンドを実行するために使用します。 <code>eval</code> の使用例については、 tset(1B) を参照してください。
<code>exec command</code>	現在のシェルの代わりに <code>command</code> を実行します。シェルは終了します。
<code>exit [(expr)]</code>	呼び出し元のシェルまたはシェルスクリプトが終了し、ステータス変数の値、または式 <code>expr</code> で指定された値が返されます。
<code>fg [%job]</code>	現在のジョブ、または指定された <code>job</code> をフォアグラウンドへ移します。
<code>foreach var (wordlist)</code> ... <code>end</code>	変数 <code>var</code> を、 <code>wordlist</code> の各メンバーに連続して設定します。 <code>var</code> の値が変わるたびに、 <code>foreach</code> と <code>end</code> との間にあるコマンドの並びを実行します。 <code>foreach</code> と <code>end</code> は異なる行に、単独で現われなければなりません。 ループの現在の繰り返しを終了するには、組み込みコマンド <code>continue</code> を使用します。またコマンド <code>foreach</code> の実行を終了するには、組み込みコマンド <code>break</code> を使用します。このコマンドが端末から読み取られるときは、ループ内の文が実行される前に、 <code>?</code> をプロンプトしてから、ループが読み取られます。
<code>glob wordlist</code>	<code>wordlist</code> 内のファイル名を展開します。エスケープ <code>\</code> を認識しない点を除いて、 <code>echo</code> と同様です。出力のワードは <code>NULL</code> 文字によって区切られます。
<code>goto label</code>	指定する <code>label</code> は、展開されるとラベルを生成するコマンドとファイル名です。シェルは可能なかぎり入力をさかのぼり、 <code>label:</code> という形式の行を探します。 <code>label:</code> の前にはスペース文字またはタブ文字がある可能性もあります。指定された行の次から実行が再開します。 <code>while</code> または <code>for</code> 組み込みコマンドと、対応する <code>end</code> との間にあるラベルへジャンプするとエラーになります。
<code>hashstat</code>	<code>path</code> 変数の内部ハッシュテーブルがコマンドの検索(実行を伴わない)にどの程度有効に働いているかを示す統計情報行を出力します。ハッシュ関数がヒットの可能性を示しているパスの各コン

ポーネントと、'/'で始まらない各コンポーネントについて、実行が試みられます。この統計は *cdpath* 変数ではなく *path* 変数だけの有効性を示します。

`history [-hr] [n]`

履歴リストを表示します。*n* を指定すると、*n* 個の最新のイベントを表示します。

-r 出力を、古い履歴からではなく最近の履歴から順に並べ変えます。

-h 先頭に番号を付加せずに履歴のリストを出力します。これは、*source* に対する -h オプションを使用して、ソーシングに適したファイルを作成するのに使用します。

`if (expr) command`

指定された式が真ならば、引数付きの単一の *command* を実行します。*command* 上での変数置換を最初に実行し、同時に `if` コマンドの残りの部分を実行します。*command* は単純コマンドでなければならず、パイプライン、コマンドリスト、括弧付きコマンドリストは指定できません。なお *expr* が偽で *command* が実行されなくても、入出力先のリダイレクションが実行されてしまうので注意してください(これはバグです)。

`if (expr) then`

`...`

`else if (expr2) then`

`...`

`else`

`...`

`endif`

expr が真ならば、最初の `else` までのコマンドを実行します。*expr2* が真ならば、`else if` と 2 番目の `else` の間にあるコマンドを実行します。それ以外の場合は、`else` と `endif` の間にあるコマンドを実行します。`else if` の組はいくつでもかまいませんが `else` は 1 つしか許されません。`endif` は 1 つだけで、これは必須です。`else` および `endif` というワードは、各行における空白を除く最初の文字でなければなりません。`if` は、その行に単独で現われるか、`else` の後に現われるかのいずれかです。

`jobs [-l]`

ジョブ制御下で活動中のジョブをリストします。

- `kill [sig] [pid] [%job] . . .`
`kill -l`
- `limit [-h] [resource [max-use]]`
- l** 通常の情報の他に、プロセス ID を表示します。
- TERM (終了) シグナル (デフォルトの場合) または指定されたシグナルを、指定されたプロセス ID、指定されたジョブ、または現在のジョブへ送信します。シグナルは番号または名前で指定します。シグナルを送るプロセスまたはジョブにデフォルトはありません。そのため、`kill` だけを入力しても現在のジョブにはシグナルを送信しません。送信中のシグナルが TERM (終了) または HUP (ハングアップ) の場合、そのジョブまたはプロセスには CONT (継続) シグナルも送られます。
- l** 送信可能なシグナル名の一覧を表示します。
- 現在のプロセス、またはそれが生成したすべてのプロセスについて、各プロセスが指定された *resource* を *max-use* 以上消費しないよう制限します。文字列 `unlimited` は、現在の制限値 (ある場合) の削除を要求します。*max-use* を省略すると、現在の制限値を表示します。*resource* を省略すると、すべての制限値を表示します。システムで利用可能な最大制限値を調べるには `sysdef(1M)` コマンドを実行してください。`sysdef` により報告される値は 16 進数ですが、`bc(1)` コマンドを使って 10 進数に変換できます。
- h** 現在の制限値ではなく強い制限値を使用します。強い制限値は現在の制限値を制限します。強い制限値を上げることができるのは特権ユーザーだけです。
- resource* として指定できるものは次のとおりです。
- | | |
|--------------------------------|--|
| <code>cputime</code> | プロセス当たりの最大 CPU 使用時間 (秒) |
| <code>filesize</code> | 最大の単一ファイル容量。ファイルシステムのサイズと機能により制限されます。 <code>df(1M)</code> を参照してください。 |
| <code>datasize</code> (ヒープサイズ) | プロセスの (スタックも含めた) 最大データ |

サイズ。システムの仮想記憶サイズ。 [swap\(1M\)](#) を参照。

stacksize プロセスの最大スタックサイズ。デフォルトのスタックサイズは 2^{64} バイトです。 [limit\(1\)](#) を使用して、シェルのデフォルトのスタックサイズを変更できます。

coredumpsize 最大コアダンプのファイルサイズ。ファイルシステムのサイズに制限します。

descriptors ファイル記述子の最大数。 [sysdef\(1M\)](#) を実行します。

memorysize 仮想記憶の最大サイズ

max-use は数値で、以下の単位を付加して指定することもできます。

nh (cputime の) 時間

nk *n* キロバイト。これは cputime を除くすべての値のデフォルト単位です。

nm *n* メガバイトまたは (cputime の) 分

mm:ss (cputime の) 分と秒

たとえば、0 メガバイトにコアファイルダンプのサイズを制限するには次のように入力します。

limit coredumpsize 0M

login [*username* | -p]

ログインシェルを終了して [login\(1\)](#) を起動します。 *.logout* ファイルは処理しません。 *username* を省略すると、login がユーザー名をプロンプトしてきます。

-p 現在の環境変数を保存します。

logout

ログインシェルを終了します。

nice [+n |-n] [command]

シェルまたは *command* のプロセス優先度の値を *n* だけ増減させます。優先度の値が大きいほど、プロセスの優先度は低くなり、実行速度は遅くなります。*command* を指定すると、そのコマンドは常にサブシェルで実行され、指定された値は1つの if コマンドの範囲内でのみ有効です。*command* を省略すると、nice は現在のシェルの値を増やします。増やす値を省略すると、nice はプロセス優先度の値を 4 に設定します。プロセス優先度の値の範囲は、-20 から 20 までです。*n* の値がこの範囲外の場合、最低値または最高値に設定されます。

+n プロセス優先度の値を *n* だけ増やします。

-n *n* だけ減らします。この引数は特権ユーザーだけが使用できます。

nohup [command]

HUP を無視して *command* を実行します。*command* 引数を省略すると、その後のスクリプト全体において HUP を無視します。*command* を指定すると、そのコマンドは常にサブシェル内で実行されます。このとき、単純な if 文中のコマンドに対する制限事項が適用されます。& を使って切り離れたすべてのプロセスに対し、このコマンドが有効になります。

notify [%job]...

現在のジョブまたは指定されたジョブのステータスが変化したとき、非同期的にユーザーに知らせます。

onintr [-|label]

割り込み時のシェルの動作を制御します。引数を指定しないと、onintr はデフォルトの動作を復元します(すなわち、シェルはシェルスクリプトを終了して、端末のコマンド入力レベルに戻ります)。- 引数を指定すると、シェルはすべての割り込みを無視します。*label* 引数を指定すると、割り込みを受信するか割り込みのために子プロセスが終了したときに、シェルは goto *label* を実行します。

popd [+n]

ディレクトリスタックをポップして、新しいトップディレクトリへ cd します。ディレクトリスタックの構成要素は、トップディレクトリを 0 として番号付けられます。

	+n スタック内の <i>n</i> 番目のエントリを破棄します。
pushd [+n <i>dir</i>]	ディレクトリスタックにディレクトリをプッシュします。引数を指定しないと、トップの2つの構成要素を交換します。 +n <i>n</i> 番目のエントリとトップスタックを交換しそれに cd します。 dir 現在の作業用ディレクトリをスタックにプッシュし、<i>dir</i> を新たな作業用ディレクトリとします。
rehash	新しく追加されたコマンドに合わせて、<i>path</i> 変数内にリストされたディレクトリの内容の内部ハッシュテーブルを再計算します。新しく追加されたディレクトリに合わせて、<i>cdpath</i> 変数内にリストされたディレクトリの内容の内部ハッシュテーブルを再計算します。
repeat <i>count</i> <i>command</i>	<i>command</i> を <i>count</i> 回繰り返します。<i>command</i> は1行の if 文と同様の制限に従います。
set [<i>var</i> [= <i>value</i>]] set <i>var</i> [<i>n</i>] = <i>word</i>	引数を設定しないと set はすべてのシェル変数の値を表示します。複数ワードからなる値は括弧でくくられて表示されます。引数 <i>var</i> のみを設定すると、set は空の (NULL) 値を <i>var</i> が示す変数に割り当てます。引数を <i>var</i> = <i>value</i> の形式で設定すると、set は <i>var</i> が示す変数に <i>value</i> が示す値を割り当てます。<i>value</i> は次のいずれかです。 word 単一ワード (もしくは引用符付きの文字列) (wordlist) スペースで区切られた、括弧付きワードの並び 値は、割り当てられる前に、コマンドおよびファイル名展開されます。set <i>var</i>[<i>n</i>] = <i>word</i> 形式は、複数ワードからなる値の <i>n</i> 番目のワードを <i>word</i> に置き換えます。
setenv [<i>VAR</i> [<i>word</i>]]	引数を設定しないと setenv はすべての環境変数を表示します。引数 <i>VAR</i> を指定すると、setenv は環境変数 <i>VAR</i> に空の値 (NULL) を設定します。(慣習上、環境変数名は大文字で指定されるのが通

常)。VAR と word の両引数を指定すると、setenv は、環境変数 NAME に単一ワードまたはクォートされた文字列である値 word を設定します。最もよく使用される環境変数 USER、TERM、PATH は、自動的に csh 変数 user、term、path から (へ) インポート (エクスポート) されます。したがって、これらの変数に setenv を使用する必要はありません。さらにシェルは、csh 変数 cwd が変更されるたびに、その値を環境変数 PWD へ設定します。

環境変数

LC_CTYPE、LC_MESSAGES、LC_TIME、LC_COLLATE、LC_NUMERIC は、C シェル内で変更されると新しい値が即座に有効になります。

LC_* 変数 (LC_CTYPE、LC_MESSAGES、LC_TIME、LC_COLLATE、LC_NUMERIC、LC_MONETARY) のいずれも環境に設定されていなければ (environ(5) を参照)、それぞれ対応するロケールのカテゴリにおける csh の動作は、環境変数 LANG によって決定されます。LC_ALL が設定されていれば、その内容が LANG 変数やその他の LC_* 変数より優先されます。上記の変数が環境にまったく設定されていなければ、C ロケール (米国スタイル) が csh の動作を決定します。

LC_CTYPE	csh の文字の処理方法を決定します。LC_CTYPE に有効な値が設定されていると、csh は、そのロケールにあった文字を含むテキストやファイル名を表示および処理できます。
LC_MESSAGES	診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、そして肯定応答および否定応答の正しい形も決定します。C ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。
LC_NUMERIC	基数文字、小数点 (C ロケールでは (.))、および 3 桁ごとのセパ

	レータ、空文字列 (C ロケールでは ("")) の値を決定します。
<code>shift [variable]</code>	<code>argv</code> のコンポーネント (または <code>variable</code> が指定されればその変数のコンポーネント) を、左へずらして最初のコンポーネントを切り捨てます。未設定の変数および <code>NULL</code> 値に対してはエラーとなります。
<code>source [-h] name</code>	<code>name</code> からコマンドを読み取ります。 <code>source</code> コマンドはネストできますが、あまり深くネストするとシェルのファイル記述子が不足する可能性があります。ソースファイル中のエラーは、それがいかなるレベルであろうと、入れ子にされたすべての <code>source</code> コマンドを終了させます。 <code>-h</code> <code>name</code> が示す、履歴のリスト上のファイルからコマンドを持ってきますが、実行はしません。
<code>stop %jobid...</code>	現在のジョブまたは 指定されたバックグラウンドジョブを停止します。
<code>stop pid...</code>	指定された <code>pid</code> (プロセス ID 番号) の実行を停止します (ps(1) を参照)。
<code>suspend</code>	<code>^Z</code> を使用して停止シグナルが送信されたときと同様に、トラック内でシェルを停止します。このコマンドは、 <code>su</code> コマンドによって開始されたシェルを停止するときによく使用します。
<code>switch (string)</code> <code>case label:</code> ... <code>breaksw</code> ... <code>default:</code> ... <code>breaksw</code> <code>endsw</code>	各 <code>label</code> を、指定された <code>string</code> (初めにコマンドおよびファイル名展開される) とマッチングします。ファイルメタキャラクタ <code>*</code> 、 <code>?</code> 、および <code>[...]</code> は、 <code>case</code> ラベルとして使用できます。このラベルは変数展開されます。一致するラベルが見つかる前にデフォルトラベルを検出すると、そのデフォルトのラベルの次から実行が開始されます。各 <code>case</code> 文および <code>default</code> 文は、行の先頭にな

ければなりません。コマンド `breaksw` は `endsw` の次から実行を続けます。それ以外の場合、C の場合と同様に、後続の `case` および `default` 文に制御が移ります。一致するラベルがなくデフォルトもない場合、`endsw` の次から実行が続行されます。

`time [command]`

引数を省略すると、現在の C シェルとその子プロセスが消費した時間についての情報を出力します。`command` を指定すると、`command` を実行し、その実行時間の情報を出力します。現在 `time` 組み込みコマンドは出力の最後 6 フィールドに対する値を計算しません。これらのフィールドに 0 の値を表示します。

example `%time ls -R`

```
9.0u 11.0s 3:32 10% 0+0k 0+0io 0pf+0w
```

(`time` 変数については、下の「環境変数と定義済みシェル変数」を参照)。

`umask [value]`

ファイル生成マスクを表示します。`value` を指定すると、ファイル生成マスクが設定されます。`value` は 8 進数で指定され、どんなビットもオフにできますが、新たにアクセス権を追加するようなビットをオンにできません。よく使われる値として 077、002、または 022 があります。077 は、自分以外の誰に対しても何のアクセス権も与えません。002 は、グループユーザーに対しては全アクセス権、グループ外ユーザーに対しては読み取り (およびディレクトリ検索) 権を与えます。022 は、グループおよびグループ外ユーザーに対して読み取り (およびディレクトリ検索) 権を与えますが、書き込み権を与えません。

`unalias pattern`

`pattern` が示すファイル名置換パターンに一致する別名を破棄します。‘`unalias *`’ と指定すると、すべての別名が破棄されます。

`unhash`

`path` と `cdpath` 変数の内部ハッシュテーブルを使用不能にします。

`unlimit [-h] [resource]`

`resource` に関する制限を削除します。`resource` が指定されないと、すべてのリソースの制限値が削除されます。リソース名の一覧については、前述の `limit` コマンドの説明を参照してください。

`-h` 対応する強い制限値を削除します。これは特権ユーザーだけしか実行できません。

<code>unset <i>pattern</i></code>	<code><i>pattern</i></code> が示すファイル名置換パターンに一致する名の変数を削除します。‘ <code>unset *</code> ’ と指定すると、すべての変数が削除されます。ただしこれによって、たいへんな副作用を引き起こします。
<code>unsetenv <i>variable</i></code>	環境から <code><i>variable</i></code> が示す変数を削除します。unset のようなパターンマッチングは行いません。
<code>wait</code>	プロンプトする前に、バックグラウンドジョブの終了 (または割り込み) を待ちます。
<code>while (<i>expr</i>)</code> <code>...</code> <code>end</code>	<code><i>expr</i></code> が真 (0 以外) であるかぎり、 <code>while</code> と、対応する <code>end</code> 文の間のコマンドを繰り返します。ループを途中で終了させるには <code>break</code> を、または先頭から再開するには <code>continue</code> を使用します。 <code>while</code> と <code>end</code> は入力行に単独で現われなければなりません。シェルの入力端末の場合、疑問符のプロンプトを表示してコマンドを受け付け、 <code>end</code> コマンドが入力されるとループ内でコマンドを実行します。
<code>%[<i>job</i>] [&]</code>	現在のジョブまたは <code><i>job</i></code> で指定されたジョブをフォアグラウンドへ移します。アンパサンドを付加すると、ジョブをバックグラウンドで実行し続けます。
<code>@[<i>var</i>=<i>expr</i>]</code> <code>@[<i>var</i>[<i>n</i>]=<i>expr</i>]</code>	引数を指定しないと、すべてのシェル変数の値を表示します。引数を指定すると、 <code><i>var</i></code> が示す変数の値、またはその値の <code><i>n</i></code> 番目のワードを、 <code><i>expr</i></code> が示す式の評価値に設定します。 <code>[<i>n</i>]</code> を指定するときは、 <code><i>var</i></code> が示す変数およびその <code><i>n</i></code> 番目のコンポーネントが既に存在していなければなりません。 式が文字 <code>></code> 、 <code><</code> 、 <code>&</code> 、または <code> </code> を含んでいる場合、少なくともその部分は括弧で囲む必要があります。 演算子 <code>*=</code> 、 <code>+=</code> などは C と同様に使用できます。名前と代入演算子とを区切っている空白は、省略可能です。ただし、指定しないと 1 ワードとみなされる式のコンポーネントを区切る場合は、スペースは必須です。 特別な接尾演算子の <code>++</code> と <code>--</code> は、 <code><i>name</i></code> をそれぞれ増加・減少させます。

環境変数と定義済みシェル変数

C シェルは、起動したプロセスに自動的に値がエクスポートされる環境変数と、そうでないシェル変数を区別します。この点は Bourne シェルとは異なります。両タイプの変数とも変数置換においては同等に扱われます。初期化の際、シェルは、変数 `argv`、`cwd`、`home`、`path`、`prompt`、`shell`、および `status` を設定します。シェルは、環境変数 `USER`、`TERM`、`HOME` をそれぞれシェル変数 `user`、`term`、`home` へコピーします。そして、前述のシェル変数が再設定されるたびに、対応する環境変数へその値を書き戻します。`PATH` と `path` は同様に扱われます。`path` は `.cshrc` または `.login` ファイル内で 1 度設定するだけです。環境変数 `PWD` は、`cwd` が変更されるたびに設定されます。以下のシェル変数の意味は(事前に)以下の通り定義済みです。

<code>argv</code>	引数リスト。シェルの今回の呼び出しの際に渡されたコマンド行引数リストを含んでいます。この変数は、 <code>\$1</code> や <code>\$2</code> などの定位置パラメータの値を決定します。
<code>cdpath</code>	<code>cd</code> 、 <code>chdir</code> 、および <code>popd</code> コマンドが検索するディレクトリのリスト。これらのコマンドが受け取ったディレクトリ引数が現在のディレクトリのサブディレクトリでないとき、この変数が示すディレクトリ群が検索されます。
<code>cwd</code>	現在のディレクトリの完全なパス名
<code>echo</code>	実行前に置換後のコマンドをエコーします。
<code>figignore</code>	ファイル名を補完させるときに無視するファイル名接尾辞のリスト。典型的な例は <code>‘.o’</code> の 1 ワードだけ。
<code>filec</code>	ファイル名補完を可能にします。その際、CTRL-d 文字の EOT および ESC 文字が端末の入力行の最後に入力されると特別な意味を持ちます。
<code>EOT</code>	前に指定した文字列で始まるファイル名のリストをすべて出力します。
<code>ESC</code>	前に指定した文字列を最も長い明確な拡張子に置き換えます。
<code>hardpaths</code>	設定すると、ディレクトリスタック内のパス名は、シンボリックリンクのコンポーネントを含まないように展開されます。
<code>histchars</code>	2 文字の文字列。最初の文字は、履歴置換文字として <code>!</code> を置き換えます。2 番目の文字は、高速置換のためのキャレット (^) を置き換えます。
<code>history</code>	履歴リストに保存される行数。数字が非常に大きいと、C シェルのメモリーすべてを消費してしまいます。設定を行わない場合、C シェルは最新のコマンドだけを保存します。
<code>home</code>	ユーザーのホームディレクトリ。ファイル名展開の <code>~</code> は、この変数の値を指します。

ignoreeof	設定すると、シェルは端末からの EOF を無視します。これにより、CTRL-D を誤って入力して C シェルを終了させてしまうことが防げます。
mail	C シェルがメールの有無をチェックするファイルのリスト。値の最初のワードが数字の場合、メールをチェックする間隔を秒数 (デフォルトは 5 秒) で表します。
nobeep	あいまいなファイル名を C シェルに拡張させる際、一致したものを見つけたときに発生するビーブ音を抑制します。
noclobber	既存のファイルが間違っって破壊されないように出力先のリダイレクションを制限します。> は新規のファイルへのみ出力先をリダイレクションできます。>> は既存のファイルへのみ出力先をリダイレクションできます。
noglob	ファイル名置換を禁止します。これは、シェルスクリプト内で一度ファイル名 (もしあれば) を取得した後、これ以上拡張したくないときに便利です。
nonomatch	パターンが一致しなかったとき、エラーではなくファイル名置換パターンを返します。パターンが間違っているときは、エラーを返します。
notify	設定すると、シェルはプロンプトの発行まで待つことなく、ジョブの終了を即座に報告します。
path	コマンドを検索するディレクトリのリストです。path は環境変数 PATH から初期化されます。この環境変数は、path を変更するたびに C シェルによって更新されます。NULL ワードは現在のディレクトリを指定します。デフォルトは通常、(/usr/bin .) です。.cshrc ファイルまたは .login ファイル (ログインシェルのみ) 内に設定することにより、csh の起動時にこの初期検索パスをオーバーライドできます。path が設定されないと、完全なパス名だけが実行されます。対話型の C シェルは、通常 .cshrc の読み取り後と path の再設定時には、必ずリストされたディレクトリの内容をハッシングします。新しいコマンドが追加されたときは、rehash コマンドを使用してテーブルを更新します。
prompt	対話型 C シェルのプロンプト文字列。非対話型シェルの prompt 変数は未設定のままです。対話型でのみ有効な .cshrc ファイル内の別名および他のコマンドは、'if (\$?prompt == 0) exit' という記述の後におくことができ、これによって、非対話型シェルの起動時間が短縮できます。prompt 文字列内の ! は、現在のイベント番号で置換されます。デフォルトプロンプトは通常の hostname% または特権ユーザー用の hostname# です。

\$prompt の設定には次の 3 つの意味があります。

	\$prompt が設定されていない	非対話型 シェル、\$?prompt を評価する
	\$prompt が設定されているが、== "" である	.cshrc が which(1) コマンドによって呼び出されました。
	\$prompt が設定されており、!= "" である	通常の対話型シェル。
savehist	ユーザーがログアウトしたときに ~/.history に保管される履歴リストの行数。 savehist の値が大きすぎると、C シェルの起動処理が遅くなります。	
shell	C シェルが存在するファイル。実行ビットがセットされているがシステムにより実行できないようなファイルを解釈するために、シェルをフォークする際に使用します。	
status	最新のコマンドによって返されたステータス。そのコマンドが異常終了した場合、ステータスに 0200 が加算されます。異常終了した組み込みコマンドは、終了ステータス 1 を返します。他のすべての組み込みコマンドは、ステータスを 0 に設定します。	
time	コマンドの自動タイミングを制御します。1つまたは2つの値が与えられます。最初の値は、報告するしきい値の CPU 時間(秒)です。2番目の値は、どのリソースについて報告するかを示すタグとテキストの文字列です。タグは、先頭にパーセント記号(%)を付加した1つの英大文字で表します。認識できないタグはテキストとして表示します。 %D 使用された、共有でないデータ領域の平均サイズ(キロバイト) %E コマンド実行時間 %F ページフォルト回数 %I ブロック入力操作の数 %K 使用された、共有でないスタック領域の平均サイズ(キロバイト) %M プロセスの実行中に使用された最大実記憶領域 %O ブロック出力操作の数 %P エラップス時間(E)に占める CPU 時間(ユーザー時間(U)とシステム時間(S)の合計)のパーセンテージ %S ユーザープロセス内で、カーネルによって消費された CPU 時間(秒) %U ユーザープロセスに使用された CPU 時間(秒)	

- %W

スワップの回数
- %X

使用されたの共有メモリ領域の平均サイズ(キロバイト)
- デフォルトの出力は、%U、%S、%E、%P、%X、%D、%I、%O、%F、%W の順です。
- verbose

履歴置換の後で各コマンドを表示します。
- 大規模ファイルの動作

ファイルが2ギガバイト(2³¹バイト)以上ある場合のcshの動作については、[largefile\(5\)](#)を参照してください。
- ファイル

~/.cshrc

各シェルの起動時に読み取られる

~/.login

ログイン時、ログインシェルによって.cshrcの後に読み取られる

~/.logout

ログアウト時、ログインシェルによって読み取られる

~/.history

次のログイン時に使用できるよう履歴を保管する

/usr/bin/sh

‘#’で始まらないシェルスクリプト用のBourneシェル

/tmp/sh*

‘<<’用の一時ファイル

/etc/passwd

‘~name’用のホームディレクトリを指定したソース
- 属性

属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

- 関連項目

[bc\(1\)](#), [echo\(1\)](#), [limit\(1\)](#), [login\(1\)](#), [ls\(1\)](#), [more\(1\)](#), [pfcsh\(1\)](#), [pfexec\(1\)](#), [ps\(1\)](#), [sh\(1\)](#), [shell_builtins\(1\)](#), [tset\(1B\)](#), [which\(1\)](#), [df\(1M\)](#), [swap\(1M\)](#), [sysdef\(1M\)](#), [access\(2\)](#), [exec\(2\)](#), [fork\(2\)](#), [pipe\(2\)](#), [a.out\(4\)](#), [ascii\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [termio\(7I\)](#)
- 診断

You have stopped jobs.

ジョブ制御下で停止しているジョブ上でCシェルの終了しようとした。すぐに再度Cシェルの終了すれば正常に終了しますが、停止しているジョブも終了します。
- 警告

シェルスクリプトをsetuidして使用することは避けてください。
- 注意事項

ワードは1024バイト以下でなければなりません。引数リストはシステムによって1,048,576バイト以下に制限されています。しかし、ファイル名の展開が適用できるコマンドの引数の最大数は1706です。コマンド置換は、引数リストに許されているのと同じ文字数までしか展開できません。ループを検出するために、シェルは別名置換を1行あたり20に制限しています。

コマンドが停止後再開されたとき、最初にコマンドを開始したディレクトリが現在のディレクトリと異なる場合、シェルは現在のディレクトリを表示します。これは、ジョブがディレクトリを内部で変更している可能性があるので、誤解を招きます(すなわち、間違っています)。

シェルの組み込み関数を停止または再開することはできません。*abc*のような形式のコマンドの並びも、停止が試されるとうまく処理できません。*b*で一時停止するとシェルは*c*を実行しません。このことは、別名から展開された場合に特に注意が必要です。この現象は、コマンドの並びを括弧で囲んでサブシェルに実行させると避けられます。

ループ内のコマンドは`?`でプロンプトしてきますが、履歴リストには格納されません。

制御構造は組み込みコマンドとみなすより構文解析をするべきです。これによって、制御コマンドはどこにでも置くことができ、`|`と組み合わせたり、`&`および`;`などのメタシンタックスと組み合わせることができるようになるはずです。

コマンド置換の出力に、修飾子`:`を使えるようにすべきです。この`:`修飾子を変数置換に使用するには2つの問題があります。1つはすべての修飾子が使用できるわけではないこと、もう1つは一度の置換に1つの修飾子しか許されないことです。

履歴置換内の`g`(グローバル)フラグは、すべてのワード内のすべての一致箇所ではなく、各ワード内の最初の一致箇所にしかな適用されません。通常のテキストエディタは、置換コマンドに`g`が指定されると、すべての一致箇所の指定とみなすのが一般的です。

クォートの規則は複雑です。二重引用符内で変数置換を強要するエスケープ文字を無効にするということは、混乱を招いて理解しにくくなり、Bourne シェルとの一貫性も保たれません。

シェルが、シンボリックリンクを無視してしまうことがあります。`hardpaths` 変数を設定すればこれを緩和できます。

次に示すように、組み込みコマンドを複数回使ってできた重複したパス名は、すべてユーザーが手動で取り除かなければなりません:

```
set path = pathnames
setenv PATH = pathnames
```

シェルスクリプトまたは`.cshrc` ファイルが次のようなコマンドを発行したときに、よくパス名が重複します。

```
'set path=(/usr/local /usr/hosts $path)'
```

このコマンドは、指定したディレクトリがパス名リスト内にあることを保証する場合に使用されます。

標準出力と標準エラーを分けて出力する唯一の方法は、以下のようにサブシェルを起動することです。

```
command > outfile ) >& errorfile
```

一般的な使用には十分耐えられますが、C シェルを少し複雑に使うと予期しない結果を招くことがあります。

csh をログインシェルとして起動しても、`.login` がホームディレクトリに存在しない場合は、csh は `/etc/.login` を読み取ります。

存在しないコマンドのインタプリタを実行しようとするシェルスクリプトを、シェルが処理した場合、シェルはシェルスクリプトが存在しないという間違った診断メッセージを返します。

使用上の留意点 現在 `time` 組み込みコマンドは出力の最後 6 フィールドに対する値を計算しません。これらのフィールドに `0` の値を表示します。

```
example %time ls -R
          9.0u 11.0s 3:32 10% 0+0k 0+0io 0pf+0w
```

名前	csplit – 指定した文脈に従ってファイルを分割
形式	csplit [-ks] [-f <i>prefix</i>] [-n <i>number</i>] <i>file</i> <i>arg1</i> ... <i>argn</i>
機能説明	csplit ユーティリティーは、 <i>file</i> オペランドが示すファイルを読み込み、その内容のすべてまたは一部を <i>arg</i> オペランドの指示に従って他のいくつかのファイルに書き出し、その出力ファイルのサイズを表示します。
オプション	サポートしているオプションは、次のとおりです。 -f <i>prefix</i> 生成されるファイルを、順番に <i>prefix00</i>、<i>prefix01</i>、...、<i>prefixn</i> と名付けます。デフォルトは <i>xx00</i> ... <i>xxn</i> です。<i>prefix</i> 引数が長すぎて、生成されるファイル名の長さが 14 バイトを超えてしまう場合はエラーとなります。その場合、ファイルは 1 つも生成されず、csplit は診断メッセージを伴って終了します。 -k 作成済みのファイルをそのままにしておきます。エラー発生時、デフォルトでは csplit は作成したファイルを削除します。 -n <i>number</i> 生成するファイル名の可変部分の桁数を <i>number</i> で指定します。デフォルト値は 2 です。 -s 生成するファイルのサイズに関する情報を出力しません。
オペランド	次のオペランドがサポートされています。 <i>file</i> 分割対象ファイルのパス名。- を指定すると標準入力とみなされます。 オペランド <i>arg1</i> ... <i>argn</i> は、以下の組み合わせで指定できます。 /<i>regexp</i>[/<i>offset</i>] 現在の行から、<i>regexp</i> で示す正規表現の評価 (<i>offset</i> 指定があればそれも考慮) により得られる行の直前の行までを内容とするファイルを生成します。<i>regexp</i> は、基本正規表現の規則に従っていなければなりません。正規表現には '\/' や '\%' を使用できます。バックslash (\) はシェルにとって特別な意味を持つため、このような形式は単一引用符で適切に引用する必要があります。<i>offset</i> は行数を表す正または負の整数で、指定は任意です。指定するときは、符号 + または - を先頭に付加しなければなりません。なお、この指定に従って行を選択したとき、生成されるファイルの行数がゼロになってしまう、あるいは入力ファイル中の行数を超えてしまう場合、結果は不確定です。ファイル生成後、現在の行は、正規表現の評価により得られた行に設定されます。<i>regexp</i> のパターンマッチングは、常に現在の行からファイルの終わりまでが対象となります。 %<i>regexp</i>%[/<i>offset</i>] このオペランドは、前述の /<i>regexp</i>[/<i>offset</i>] 指定と同じ意味ですが、入力ファイルから選択した行によるファイルの生成は行われません。

line_no 現在の行から、行番号 *line_no* で示す行の直前の行までを内容とするファイルを作成します。作成したファイル中では、各行に1から始まる番号が振られます。現在の行は *line_no* で示す行に移ります。

{*num*} 直前のオペランドを繰り返します。この指定は、前述のどのオペランドの後でも指定できます。*rexp* のタイプのオペランドの後に指定すれば、そのオペランドは *num* 回繰り返されます。*line_no* オペランドの後に指定すれば、ファイルはその位置から *line_no* 行ごとに *num* 回だけ分割されます。

オペランドが示す行が、現在の位置からファイルの終わりまでの間に存在していない場合、エラーが報告されます。

使用法 ファイルが2ギガバイト (2^{31} バイト) 以上ある場合の `csplit` の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 ファイルを分割および結合する

次の例は、4つのファイル `cobol00...cobol03` を作成します。

```
example% csplit -f cobol filename \
'/procedure division/' /par5./ /par16./
```

分割されたファイルを編集した後、次のようにすると、再度1つにまとめることができます。

```
example% cat cobol0[0-3] > filename
```

この例は、元のファイルを上書きします。

例2 ファイルを等分に分割する

次の例では、100行毎に10,000行までファイルを分割します。`-k` オプションは10,000行未満しかなくても、作成されたファイルを保存します。ただし、エラーメッセージはそのまま出力されます。

```
example% csplit -k filename 100 {99}
```

例3 各Cルーチンごとにファイルを作成する

`prog.c` が通常のCのコード化規則 (ルーチンの最後の行の1文字目が `}` だけである) に従っていれば、この例では、`prog.c` 中のCルーチン (21個まで) に対し、それぞれファイルを作成します。

```
example% csplit -k prog.c '%main(%' '/^')/+1' {20}
```

環境変数

`csplit` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [sed\(1\)](#), [split\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

診断 診断メッセージは、次に示すもの以外は、メッセージテキストを読めば意味がわかるようになっています。

arg – out of range 指定された引数が、現在の位置とファイルの終わりまでの間に存在する行を指してしないことを意味します。

名前	ct – spawn login to a remote terminal
形式	ct [<i>options</i>] <i>telno</i> ...
機能説明	The ct utility dials the telephone number of a modem that is attached to a terminal and spawns a login process to that terminal. The <i>telno</i> is a telephone number, with equal signs for secondary dial tones and minus signs for delays at appropriate places. (The set of legal characters for <i>telno</i> is 0 through 9, -, =, *, and #. The maximum length <i>telno</i> is 31 characters). If more than one telephone number is specified, ct will try each in succession until one answers; this is useful for specifying alternate dialing paths. ct will try each line listed in the file /etc/uucp/Devices until it finds an available line with appropriate attributes, or runs out of entries. After the user on the destination terminal logs out, there are two things that could occur depending on what type of port monitor is monitoring the port. In the case of no port monitor, ct prompts: Reconnect? If the response begins with the letter n, the line will be dropped; otherwise, ttymon will be started again and the login: prompt will be printed. In the second case, where a port monitor is monitoring the port, the port monitor reissues the login: prompt. The user should log out properly before disconnecting.
オプション	The following options are supported: -h Normally, ct will hang up the current line so that it can be used to answer the incoming call. The -h option will prevent this action. The -h option will also wait for the termination of the specified ct process before returning control to the user's terminal. -sspeed The data rate may be set with the -s option. <i>speed</i> is expressed in baud rates. The default baud rate is 1200. -v If the -v (verbose) option is used, ct will send a running narrative to the standard error output stream. -wn If there are no free lines ct will ask if it should wait, and for how many minutes, before it gives up. ct will continue to try to open the dialers at one-minute intervals until the specified limit is exceeded. This dialogue may be overridden by specifying the -wn option. <i>n</i> is the maximum number of minutes that ct is to wait for a line. -xn This option is used for debugging; it produces a detailed output of the program execution on stderr. <i>n</i> is a single number between 0 and 9. As <i>n</i> increases to 9, more detailed debugging information is given.

ファイル /etc/uucp/Devices
 /var/adm/ctlog

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp

関連項目 [cu\(1C\)](#), [login\(1\)](#), [uucp\(1C\)](#), [ttypmon\(1M\)](#), [attributes\(5\)](#)

注意事項 The ct program will not work with a DATAKIT Multiplex interface.

For a shared port, one used for both dial-in and dial-out, the ttypmon program running on the line must have the -r and -b options specified (see [ttypmon\(1M\)](#)).

名前 `ctags-ex` および `vi` で使用するタグファイルの作成

形式 `/usr/bin/ctags [-aBFtuvwx] [-f tagfile] file...`
`/usr/xpg4/bin/ctags [-aBFuvwx] [-f tagfile] file...`

機能説明 `ctags` ユーティリティーは、C、C++、Pascal、FORTRAN、[yacc\(1\)](#)、[lex\(1\)](#) の指定されたソースから [ex\(1\)](#) 用のタグファイルを作成します。タグファイルは、ファイルのグループにおける指定されたオブジェクト (この場合は関数と型定義) の位置を示します。タグファイルの各行には、オブジェクト名、それが定義されているファイル、およびオブジェクト定義のアドレス指定が含まれます。関数はパターンによって検索され、型定義は行番号によって検索されます。指定子は、行の個別のフィールドに指定され、スペースまたは TAB 文字で区切られます。タグファイルを使用すると、`ex` はこれらのオブジェクト定義を簡単に検索できます。

通常 `ctags` は、`tags` というファイルにタグの記述を入れます。これは、`-f` オプションによって変更にできます。

名前が `.c` または `.h` で終わるファイルは、C または C++ のソースファイルとみなされ、C または C++ ルーチンとマクロ定義が含まれていないか調べられます。名前が `.cc`、`.C`、または `.cxx` で終わるファイルは、C++ ソースファイルとみなされます。名前が `.y` で終わるファイルは、`yacc` ソースファイルとみなされます。名前が `.l` で終わるファイルは、`lex` ファイルとみなされます。これ以外のファイルは、まず Pascal または FORTRAN のルーチン定義が含まれていないか調べられます。これらの定義が含まれていないファイルは、再度 C 定義について検索されます。

タグ `main` は、C または C++ プログラムでは特別に処理されます。タグ `formed` は、`file` の先頭に `M` を付け、最後に `.c`、`.cc`、`.C`、または `.cxx` があれば削除し、先頭のパス名コンポーネントも削除して、書式が作成されます。これにより、複数のプログラムが置かれているディレクトリでも `ctags` を利用できます。

オプション 出力に関するオプションの優先順位は、`-x`、`-v`、そして残りのオプションという順になります。サポートしているオプションは、次のとおりです。

- `-a` 既存の `tags` ファイルに出力を追加します。
- `-B` 後方検索パターン (`? .. ?`).
- `-f tagfile` `tags` ではなく、ファイル `tagfile` にタグ記述を入れます。
- `-F` 前方検索パターン (`/ .. /`) を使用します (デフォルト)。
- `-t` 型定義のタグを作成します。`/usr/xpg4/bin/ctags` は、デフォルトで、型定義のタグを作成します。
- `-u` タグ内の指定したファイルを更新します。つまり、これらのファイルへの参照はすべて削除されて、ファイルに新しい値が追加されます。注意: このオプションを指定すると実行速度が遅くなります。通常、単に `tags` ファイルを作成し直す方が、速く実行できます。

- v 標準出力上に関数名、ファイル名、およびページ番号をリストする索引を出力します (64 行のページを想定)。出力は、辞書編集方式の順にソートされるため、出力を `sort -f` に通しておくのがよいでしょう。
- w 警告診断を抑制します。
- x オブジェクト名、各オブジェクトが定義されている行番号とファイル名、およびその行のテキストを作成して、標準出力に出力します。これは、オフラインで読み取ることでできる関数索引として出力可能な単純な索引です。

オペランド 次の *file* オペランドがサポートされています。

- file.c* .c 接尾辞で終了するベース名を持つファイルは、C 言語のソースコードとみなされます。
- file.h* .h 接尾辞で終了するベース名を持つファイルは、C++ 言語のソースコードとみなされます。
- file.f* .f 接尾辞で終了するベース名を持つファイルは、FORTRAN 言語のソースコードとみなされます。

使用法 -v オプションは、オプションとして提供される BSD 互換性パッケージに含まれている `vgrind` で主に使用されます。

使用例 例1 アルファベット順にエントリを生成する

-v オプションを付けて `ctags` を使用すると、`vgrind` に対して必ずしも適切とはいえない順序でエントリが作成されます。結果をアルファベット順で生成するには、出力を `sort -f` に送るようにします。

```
example% ctags -v filename.c filename.h | sort -f > index
example% vgrind -x index
```

例2 タグファイルを作成する

`sourcedir` をルートとするディレクトリ階層に C ソースのタグファイルを作成するには、まず空のタグファイルを作成してから、`find(1)` を実行します。

```
example% cd sourcedir ; rm -f tags ; touch tags
example% find . \( -name SCCS -prune -name \
    '*.c' -o -name '*.h' \) -exec ctags -u {} \;
```

スペースは、ここで示されたとおりに正確に入力してください。

環境変数 `ctags` の実行に影響を与える次の環境変数については、[environ\(5\)](#) のマニュアルページを参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

ファイル tags 出力タグファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/ctags

属性タイプ	属性値
使用条件	developer/base-developer-utilities

/usr/xpg4/bin/ctags

属性タイプ	属性値
使用条件	system/xopen/xcu4
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [ex\(1\)](#), [lex\(1\)](#), [vgrind\(1\)](#), [vi\(1\)](#), [yacc\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 FORTRAN と Pascal の 関数、サブルーチン、およびプロシージャは、非常に単純な方法で認識されます。ブロック構造とみなして処理されることはありません。異なるブロックに同じ名前の 2 つの Pascal プロシージャがある場合、一方しか判別されません。

C または Pasca 関数1、および FORTRAN 関数を検索するかどうかを決定する手法は、あまり出来の良いものではありません。

ctags ユーティリティーは、`#ifdefs` を認識しません。

ctags ユーティリティーは、Pascal の型を把握している必要があります。型定義の検出は、入力の形式が適切かどうかによります。`-tx` を使用すると、型定義の最後の行だけが示されます。

名前	ctrun – execute command in a process contract
形式	<code>/usr/bin/ctrun [options] command [argument]...</code>
機能説明	The <code>ctrun</code> utility starts a command in a newly created process contract. <code>ctrun</code> holds the contract and can be instructed to output or respond to events that occur within the contract. For additional information about process contracts, see contract(4) and process(4).
オプション	The following options are supported: -A <i>aux</i> Sets the process contract creator's auxiliary field. -i <i>event</i>, [<i>event</i> ...] Sets the informative and fatal events, respectively. -f <i>event</i>, [<i>event</i> ...] The following are valid <i>events</i>: - core A member process dumped core. - core events are informative by default. - empty The last member of the process contract exited. - exit A member process exited. - fork A process was added to the process contract. - hwerr A member process encountered a hardware error. - hwerr events are fatal by default. - signal A member process received a fatal signal from a process in a different process contract. Only <code>core</code>, <code>hwerr</code>, and <code>signal</code> events can be made fatal. More events can be delivered than requested if <code>ctrun</code> requires them for its own purposes. For example, <code>empty</code> messages are always requested if a lifetime of contract is specified. See <code>-l</code>. -F <i>fmri</i> Sets the process contract service FMRI field. To set this field the caller is required to have the {PRIV_CONTRACT_IDENTITY} in its effective set. -l <i>lifetime</i> The following valid <i>lifetime</i> values are supported: - child <code>ctrun</code> exits when the command exits, regardless of whether the contract is empty. - contract <code>ctrun</code> exits only when the contract exits. This is the default.

	none	ct run exits immediately, orphaning the contract.
-o <i>option</i> , [<i>option</i> ...]	The following <i>options</i> are supported:	
	noorphan	Kills all processes in the contract if the holder (ct run) exits. This option is invalid when a lifetime of none is specified.
	pgrponly	If a fatal error occurs, kills at most the process group of which the errant process is a member.
	regent	The contract inherits inheritable contracts when abandoned by member processes.
-r <i>count</i>	If the contract encounters a fault, this option attempts to restart the command <i>count</i> times. If <i>count</i> is 0, the attempt to restart continues indefinitely. By default, ct run does not attempt to restart the command. This option is invalid if a lifetime other than contract is specified or if the pgrponly option is used.	
-t	If the contract created by ct run inherited subcontracts from its member processes, attempts to transfer them to the new contract when restarting. This option is invalid unless -r is also specified.	
-v	Displays contract events and ct run actions as they occur.	
-V	Displays verbose contract events, as are displayed by the -v option of ctwatch. Implies -v.	

オペランド

The following operands are supported:	
<i>argument</i>	One of the strings treated as an argument to <i>command</i> .
<i>command</i>	The command to be passed to <code>execvp(2)</code> . See exec(2) .

使用例

例 1 Running a Shell in a New Process Contract

The following example runs a shell in a new process contract:

```
example% ctrun -l child -o pgrponly ksh
```

The `-l child` option argument is specified so that ct run won't wait until all children of the shell have exited. `-o pgrponly` is specified because an interactive ksh puts each job in a new process group, and an error in one job is unlikely to affect the others.

例2 Running a Simple Server

The following example runs a simple server:

```
example% ctrun -r 0 -t -f hwerr,core,signal server
```

The `-r 0` and `-t` options are specified to indicate that if the server encounters a fatal error, `ctrun` should try to restart it. The `-f` option makes “`hwerr`”, “`core`”, and “`signal`” fatal events.

終了ステータス

If *command* is specified and successfully invoked (see [exec\(2\)](#)), the exit status of `ctrun` is the exit status of *command*. Otherwise, `ctrun` exits with one of the following values:

- 123 The child process exited abnormally.
- 124 `ctrun` encountered an internal error.
- 125 Invalid arguments were provided to `ctrun`.
- 126 *command* was found but could not be invoked.
- 127 *command* could not be found.

ファイル

/system/contract/process/*

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	See below.

Human Readable Output is Uncommitted. Invocation is Committed.

関連項目

[ctstat\(1\)](#), [ctwatch\(1\)](#), [exec\(2\)](#), [contract\(4\)](#), [process\(4\)](#), [attributes\(5\)](#)

名前	ctstat – display active system contracts
形式	<code>/usr/bin/ctstat [-a] [-i <i>contractid...</i>] [-t <i>type...</i>] [-v] [-T u d] [<i>interval</i> [<i>count</i>]]</code>
機能説明	The <code>ctstat</code> utility allows a user to observe the contracts active on a system. Unless you specify the <code>-i</code> or <code>-t</code> option, <code>ctstat</code> displays statistics on all contracts in the system.
オプション	The following options are supported: <code>-a</code> Display all contracts regardless of state. By default, only those contracts which are in the owned, inherited, or orphan states are displayed. <code>-i <i>contractid...</i></code> Request status on the specified contracts, identified by their numeric contract identifier (<i>contract_id</i>). This option accepts lists as arguments. Items in the list can be separated by commas, or enclosed in quotes and separated by commas or spaces. <code>-T u d</code> Display a time stamp. Specify <code>u</code> for a printed representation of the internal representation of time. See time(2). Specify <code>d</code> for standard date format. See date(1). <code>-t <i>type...</i></code> Request status on contracts of the specified type (<i>type</i>). This option accepts lists as arguments. Items in the list can be separated by commas, or enclosed in quotes and separated by commas or spaces. The following types are supported: process Process contracts <code>-v</code> Verbose output.
オペランド	The following operands are supported: <i>interval</i> Report once each <i>interval</i> seconds. <i>count</i> Print only <i>count</i> reports.

Output

The following list defines the column headings and the meanings of a `ctstat` report:

CTID

The contract ID of the contract.

ZONEID

The zone ID of the contract's creator.

TYPE

The contract type.

STATE

The state of the contract:

owned

Contract is owned by a process.

inherited

The contract owner has exited abnormally and the contract has been inherited by the owner's process contract.

orphan

The contract owner has abandoned the contract, the contract owner exited abnormally and the contract was not inherited by the owner's process contract, or the process contract which had inherited the contract was abandoned by its owner.

dead

The contract is no longer active. It is removed from the system automatically when all references to it (open file descriptors, contract templates, and events) have been released.

HOLDER

If the contract is in the `owned` state, the pid of the process that owns the contract. If the contract is in the `inherited` state, the id of the regent process contract.

EVENTS

The number of unacknowledged critical events pending.

QTIME

The time until quantum ends, or - if no negotiation is in progress.

NTIME

The time until negotiation ends, or - if no negotiation is in progress.

使用例

例 1 Reporting on all Contracts in the System

The following example reports on all contracts in the system:

```
example% ctstat -a
```

CTID	TYPE	STATE	HOLDER	EVENTS	QTIME	NTIME
1	process	owned	100579	0	-	-

例 1 Reporting on all Contracts in the System (続き)

2	process dead	-	1	-	-
3	process inherit	1	3	-	-
4	process orphan	-	0	-	-

例 2 Obtaining a Verbose Report of All Contracts in the System

The following example obtains a verbose report of all contracts in the system:

example% ctstat -av

CTID	TYPE	STATE	HOLDER	EVENTS	QTIME	NTIME
1	process owned		100579	0	-	-
	informative event set: none					
	critical event set:		hwerr core			
	fatal event set:		hwerr			
	parameter set:		none			
	member processes:		100600 100601			
	inherited ctids:		none			
	service fmri:		svc:/system/init:default			
	svc_fmri ctid:		1			
	creator:		sched			
	aux:					
2	process dead	-		1	-	-
	informative event set: none					
	critical event set:		none			
	fatal event set:		hwerr core			
	parameter set:		pgrponly			
	member processes:		none			
	inherited ctids:		none			
	service fmri:		svc:/system/power:default			
	svc_fmri ctid:		19			
	creator:		svc.startd			
	aux:		start			

終了ステータス The following exit values are returned:

- 0 Successful completion.
- 1 An error occurred.
- 2 Invalid arguments.

ファイル

/system/contract/*

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	See below.

The human readable output is Uncommitted. The invocation is Committed.

関連項目

[ctrun\(1\)](#), [ctwatch\(1\)](#), [contract\(4\)](#), [process\(4\)](#), [attributes\(5\)](#)

名前

形式

機能説明

オプション

オペランド

Output

使用例

ctwatch – watch events in a contract or group of contracts

/usr/bin/ctwatch [-f] [-r] [-v] contract-type... | contract-id...

The ctwatch utility allows a user to observe the events occurring within a set of contracts or contract types. By default, ctwatch watches all contracts.

The following options are supported:

- f

Report events starting at the front of the event queue. Normally, ctwatch reports only events which occur after it has been invoked. With the -f option, any events that still exist in the contracts' event queues when ctwatch is invoked (for example, unacknowledged critical events) are also reported.

- r

Reliably watches all messages. Normally, the system may drop informative events and acknowledged critical events at any time, so ctwatch isn't guaranteed to see them all. This option may only be used if the ctwatch is invoked with the {PRIV_CONTRACT_EVENT} privilege asserted in its effective set.

- v

Request verbose event descriptions.

The following operands are supported:

contract-type

Valid contract types are:

process

Process contracts.

contract-id

A valid contract id.

The following list defines the column headings and the meanings of a ctwatch report:

CTID

The contract ID generating the event.

EVID

The event ID.

CRIT

Whether the event is informative, critical, or initiates an exit negotiation. Values are info, crit, or neg, respectively.

ACK

The event has been acknowledged. Values are yes or “no”.

CTTYPE

The contract type.

SUMMARY

A type-specific summary of the event.

例 1 Watching a process contract

example% ctwatch -r 1

CTID

EVID

CRIT

ACK

CTTYPE

SUMMARY

1

2

crit

no

process

pid 100569 was created

1

3

info

no

process

pid 100569 encountered hardware error

1

4

info

no

process

pid 100568 exited

1

5

info

no

process

pid 100569 exited

例 1 Watching a process contract (続き)

```
1      6      crit no process contract empty
```

終了ステータス The following exit values are returned:

0 Successful completion.

1 An error occurred.

2 Invalid arguments.

ファイル /system/contract/*

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	See below.

Human Readable Output is Uncommitted. Invocation is Committed.

関連項目 [ctrun\(1\)](#), [ctstat\(1\)](#), [contract\(4\)](#), [process\(4\)](#), [attributes\(5\)](#), [privileges\(5\)](#)

注意事項 Ordering of events is only guaranteed within a single contract, or within a single type when a type is specified.

ctwatch can only observe those events which are generated by contracts owned or authored by processes with the same effective user ID as ctwatch, unless the {PRIV_CONTRACT_OBSERVER} privilege is asserted in its effective set.

名前	cu – call another UNIX system
形式	cu [-c <i>device</i> -l <i>line</i>] [-s <i>speed</i>] [-b <i>bits</i>] [-h] [-n] [-t] [-d] [-o -e] [-L] [-C] [-H] <i>telno</i> <i>systemname</i> [<i>local-cmd</i>]
機能説明	The command cu calls up another UNIX system, a terminal, or possibly a non-UNIX system. It manages an interactive conversation with possible transfers of files. It is convenient to think of cu as operating in two phases. The first phase is the connection phase in which the connection is established. cu then enters the conversation phase. The -d option is the only one that applies to both phases.
オプション	cu accepts many options. The -c, -l, and -s options play a part in selecting the medium. The remaining options are used in configuring the line. -b <i>bits</i> Forces <i>bits</i> to be the number of bits processed on the line. <i>bits</i> is either 7 or 8. This allows connection between systems with different character sizes. By default, the character size of the line is set to the same value as the current local terminal, but the character size setting is affected by LC_CTYPE also. -c <i>device</i> Forces cu to use only entries in the "Type" field (the first field in the /etc/uucp/Devices file) that match the user specified <i>device</i>, usually the name of a local area network. -C Runs the <i>local-cmd</i> specified at the end of the command line instead of entering interactive mode. The stdin and stdout of the command that is run refer to the remote connection. -d Prints diagnostic traces. -e Sets an EVEN data parity. This option designates that EVEN parity is to be generated for data sent to the remote system. -h Sets communication mode to half-duplex. This option emulates local echo in order to support calls to other computer systems that expect terminals to be set to half-duplex mode. -H Ignores one hangup. This allows the user to remain in cu while the remote machine disconnects and places a call back to the local machine. This option should be used when connecting to systems with callback or dialback modems. Once the callback occurs subsequent hangups will cause cu to terminate. This option can be specified more than once. For more information about dialback configuration, see remote(4) and 『Solaris のシステム管理 (IP サービス)』 -l <i>line</i> Specifies a device name to use as the communication line. This can be used to override the search that would otherwise take place for the first available line having the right speed. When the -l option is used without the -s option, the speed of a line is taken from the /etc/uucp/Devices file record in which <i>line</i> matches the second field (the Line field). When the -l and -s options are both

used together, cu will search the /etc/uucp/Devices file to check if the requested speed for the requested line is available. If so, the connection will be made at the requested speed, otherwise, an error message will be printed and the call will not be made. In the general case where a specified device is a directly connected asynchronous line (for instance, /dev/term/a), a telephone number (*telno*) is not required. The specified device need not be in the /dev directory. If the specified device is associated with an auto dialer, a telephone number must be provided.

- L Goes through the login chat sequence specified in the /etc/uucp/Systems file. For more information about the chat sequence, see 『Solaris のシステム管理 (IP サービス)』
- n Requests user prompt for telephone number. For added security, this option will prompt the user to provide the telephone number to be dialed, rather than taking it from the command line.
- o Sets an ODD data parity. This option designates that ODD parity is to be generated for data sent to the remote system.
- s *speed* Specifies the transmission speed (300, 1200, 2400, 4800, 9600, 19200, 38400). The default value is "Any" speed which will depend on the order of the lines in the /etc/uucp/Devices file.
- t Dials a terminal which has been set to auto answer. Appropriate mapping of carriage-return to carriage-return-line-feed pairs is set.

オペランド

The following operands are supported:

- telno* When using an automatic dialler, specifies the telephone number with equal signs for secondary dial tone or minus signs placed appropriately for delays of 4 seconds.
- systemname* Specifies a uucp system name, which can be used rather than a telephone number; in this case, cu will obtain an appropriate direct line or telephone number from a system file.

使用法

Connection Phase

cu uses the same mechanism that **uucp(1C)** does to establish a connection. This means that it will use the uucp control files /etc/uucp/Devices and /etc/uucp/Systems. This gives cu the ability to choose from several different media to establish the connection. The possible media include telephone lines, direct connections, and local area networks (LAN). The /etc/uucp/Devices file contains a list of media that are available on your system. The /etc/uucp/Systems file contains information for connecting to remote systems, but it is not generally readable.

Note: cu determines which `/etc/uucp/Systems` and `/etc/uucp/Devices` files to use based upon the name used to invoke cu. In the simple case, this name will be "cu", but you could also have created a link to cu with another name, such as "pppcu", in which case cu would then look for a "service=pppcu" entry in the `/etc/uucp/Sysfiles` file to determine which `/etc/uucp/Systems` file to use.

The *telno* or *systemname* parameter from the command line is used to tell cu what system you wish to connect to. This parameter can be blank, a telephone number, a system name, or a LAN specific address.

telephone number	A telephone number is a string consisting of the tone dial characters (the digits 0 through 9, *, and #) plus the special characters = and -. The equal sign designates a secondary dial tone and the minus sign creates a 4 second delay.
system name	A system name is the name of any computer that uucp can call; the uname(1C) command prints a list of these names.
LAN address	The documentation for your LAN will show the form of the LAN specific address.

If cu's default behavior is invoked (not using the `-c` or `-l` options), cu will use the *telno* or *systemname* parameter to determine which medium to use. If a telephone number is specified, cu will assume that you wish to use a telephone line and it will select an automatic call unit (ACU). Otherwise, cu will assume that it is a system name. cu will follow the uucp calling mechanism and use the `/etc/uucp/Systems` and `/etc/uucp/Devices` files to obtain the best available connection. Since cu will choose a speed that is appropriate for the medium that it selects, you may not use the `-s` option when this parameter is a system name.

The `-c` and `-l` options modify this default behavior. `-c` is most often used to select a LAN by specifying a Type field from the `/etc/uucp/Devices` file. You must include either a *telno* or *systemname* value when using the `-c` option. If the connection to *systemname* fails, a connection will be attempted using *systemname* as a LAN specific address. The `-l` option is used to specify a device associated with a direct connection. If the connection is truly a direct connection to the remote machine, then there is no need to specify a *systemname*. This is the only case where a *telno* or *systemname* parameter is unnecessary. On the other hand, there may be cases in which the specified device connects to a dialer, so it is valid to specify a telephone number. The `-c` and `-l` options should not be specified on the same command line.

Conversation Phase

After making the connection, cu runs as two processes. The *transmit* process reads data from the standard input and, except for lines beginning with ~, passes it to the remote system. The *receive* process accepts data from the remote system and, except for lines beginning with ~, passes it to the standard output. Normally, an automatic DC3/DC1 protocol is used to control input from the remote so the buffer is not overrun. Lines beginning with ~ have special meanings.

Commands

The *transmit* process interprets the following user initiated commands:

~.	Terminates the conversation.
~!	Escapes to an interactive shell on the local system.
~!cmd . . .	Runs <i>cmd</i> on the local system (via <code>sh -c</code>).
~\$cmd . . .	Runs <i>cmd</i> locally and send its output to the remote system.
~%cd	Changes the directory on the local system. Note: ~!cd will cause the command to be run by a sub-shell, probably not what was intended.
~%take from [to]	Copies file <i>from</i> (on the remote system) to file <i>to</i> on the local system. If <i>to</i> is omitted, the <i>from</i> argument is used in both places.
~%put from [to]	Copies file <i>from</i> (on local system) to file <i>to</i> on remote system. If <i>to</i> is omitted, the <i>from</i> argument is used in both places.
~~ line	Sends the line ~ <i>line</i> to the remote system.
~%break	Transmits a BREAK to the remote system (which can also be specified as ~%b).
~%debug	Toggles the -d debugging option on or off (which can also be specified as ~%d).
~t	Prints the values of the termio structure variables for the user's terminal (useful for debugging).
~l	Prints the values of the termio structure variables for the remote communication line (useful for debugging).
~%ifc	Toggles between DC3/DC1 input control protocol and no input control. This is useful when the remote system does not respond properly to the DC3 and DC1 characters (can also be specified as ~%nostop).
~%ofc	Toggles the output flow control setting. When enabled, outgoing data may be flow controlled by the remote host (can also be specified as ~%nostop).
~%divert	Allows/disallows unsolicited diversions. That is, diversions not specified by ~%take.
~%old	Allows/disallows old style syntax for received diversions.
~%nostop	Same as ~%ifc.

The *receive* process normally copies data from the remote system to the standard output of the local system. It may also direct the output to local files.

The use of `~%put` requires `stty(1)` and `cat(1)` on the remote side. It also requires that the current erase and kill characters on the remote system be identical to these current control characters on the local system. Backslashes are inserted at appropriate places.

The use of `~%take` requires the existence of `echo(1)` and `cat(1)` on the remote system, and that the remote system must be using the Bourne shell, `sh`. Also, `tabs` mode (see `stty(1)`) should be set on the remote system if tabs are to be copied without expansion to spaces.

When `cu` is used on system X to connect to system Y and subsequently used on system Y to connect to system Z, commands on system Y can be executed by using `~`. Executing a tilde command reminds the user of the local system `uname`. For example, `uname` can be executed on Z, X, and Y as follows:

```
uname
Z
~[X] !uname
X
~~[Y] !uname
Y
```

In general, `~` causes the command to be executed on the original machine. `~ ~` causes the command to be executed on the next machine in the chain.

使用例

例 1 Dialling a system

To dial a system whose telephone number is 9 1 201 555 1234 using 1200 baud (where dialtone is expected after the 9):

```
example% cu -s 1200 9=12015551234
```

If the speed is not specified, "Any" is the default value.

例 2 Logging in to a system on a direct line

To login to a system connected by a direct line:

```
example% cu -l /dev/term/b
```

or

```
example% cu -l term/b
```

例 3 Dialling a system with specific line and speed

To dial a system with a specific line and speed:

```
example% cu -s 1200 -l term/b
```


例 4 Using a system name

To use a system name:

example% **cu systemname**

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of cu: LC_CTYPE, LC_MESSAGES, and NLSPATH.

終了ステータス

The following exit values are returned:

0 Successful completion.

>0 An error occurred.

ファイル

/etc/uucp/Devices device file

/etc/uucp/Sysfiles system file

/etc/uucp/Systems system file

/var/spool/locks/* lock file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp

関連項目

[cat\(1\)](#), [echo\(1\)](#), [stty\(1\)](#), [tip\(1\)](#), [uname\(1\)](#), [ct\(1C\)](#), [uuname\(1C\)](#), [uucp\(1C\)](#), [remote\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#)

『Solaris のシステム管理 (IP サービス)』

注意事項

The cu utility takes the default action upon receipt of signals, with the exception of:

SIGHUP Close the connection and terminate.

SIGINT Forward to the remote system.

SIGQUIT Forward to the remote system.

SIGUSR1 Terminate the cu process without the normal connection closing sequence.

The cu command does not do any integrity checking on data it transfers. Data fields with special cu characters may not be transmitted properly. Depending on the interconnection hardware, it may be necessary to use a ~. to terminate the conversion, even if stty 0 has been used. Non-printing characters are not dependably transmitted using either the ~%put or ~%take commands. ~%put and ~%take cannot be used over multiple links. Files must be moved one link at a time.

There is an artificial slowing of transmission by cu during the ~%put operation so that loss of data is unlikely. Files transferred using ~%take or ~%put must contain a trailing newline, otherwise, the operation will hang. Entering a Control-D command usually clears the hang condition.

名前	cut - ファイルの各行から選択されたフィールドをカット
形式	<pre> /usr/bin/cut -b list [-n] [file]... /usr/bin/cut -c list [file]... /usr/bin/cut -f list [-d delim] [-s] [file]...</pre>
機能説明	cut は、1 つ以上のファイルからバイト、文字、または文字で区切られたフィールドをカットし、連結して標準出力に出力します。 <i>option</i> の引数リストには、コンマまたは空白文字で区切られた正数およびその範囲を指定します。範囲には、3 つの形式があります。最初の形式は、2 つの正の整数をハイフンで区切る (最小値-最大値) ものです。これは、小さい値から大きい値までのすべてのフィールドを表します。2 番目の形式は、正数の前にハイフンを付ける (-最大値) ものです。これは、1 から最大値までのすべてのフィールドを表します。最後の形式は、正数の後ろにハイフンを付ける (最小値-) ものです。これは、最小値から最後のフィールドまでのフィールドすべて (両端の数値を含む) を表します。リスト内の要素は、繰り返したり、重複させたり、任意の順序で指定したりできます。出力の順序は、入力と同じです。 -b、-c、-f オプションのうち、1 つだけを指定する必要があります。 ファイルが指定されていないか - である場合、cut は標準入力からの入力をカットします。現在のオフセットがファイルの開始位置として定義されます。 サポートしているオプションは、次のとおりです。 -blist -b の後続く <i>list</i> はバイト位置を指定します (たとえば、-b1-72 は各行の最初の 72 バイトを渡します)。-b と -n を一緒に使用すると、複数バイト文字が分割されないように <i>list</i> が調整されます。 -clist -c に続く <i>list</i> は文字位置を指定します (たとえば、-c1-72 は各行の最初の 72 文字を渡します)。 -d delim -d に続く -delim はフィールド区切り文字です (f オプションの場合のみ)。デフォルトは TAB です。シェルにとって意味のある空白や他の文字は引用符で囲まなければなりません。delim は複数バイト文字でもかまいません。 -flist -f に続く <i>list</i> は区切り文字によってファイル内で区切られると見なされるフィールドのリストです (-d を参照)。たとえば、-f1,7 は最初と 7 番目のフィールドだけをコピーします。フィールド区切り文字のない行は、-s が指定されていないかぎりそのまま渡されます (テーブルのサブヘディングに有効)。

- n

文字を分割しません。-b list と -n が一緒に使用されると、list は複数バイト文字が分割されないように調整されます。
- s

-f オプションが指定されたとき、区切り文字のない行を抑止します。指定されない場合、区切り文字のない行はそのまま渡されます。
- オペランド

次のオペランドがサポートされています。

file 入力ファイルのパス名。file オペランドが指定されない場合、または file オペランドが - である場合には、標準入力を使用されます。
- 使用法

ファイルが2ギガバイト(2³¹バイト)以上ある場合の cut の動作については、[largefile\(5\)](#)を参照してください。
- 使用例

例1 ユーザーIDを割り当てる
ユーザーIDとユーザー名を組み合わせで出力するには次のようにします。

example% cut -d: -f1,5 /etc/passwd

例2 現在のログイン名を設定する
name に現在のログイン名を設定するには次のようにします。

example\$ name=\$(who am i | cut -f1 -d' ')
- 環境変数

cut の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。
- 終了ステータス

次の終了ステータスが返されます。

0 入力ファイルはすべて正常に出力されました。

>0 エラーが発生した。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[grep\(1\)](#), [paste\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前 date – 日付と時間の出力と設定

形式

```

/usr/bin/date [-u] [+format]

/usr/bin/date [-a [-]sss.fff]

/usr/bin/date [-u] [ [mmdd] HHMM | mmddHHMM [cc] yy] [.SS]

/usr/xpg4/bin/date [-u] [+format]

/usr/xpg4/bin/date [-a [-]sss.fff]

/usr/xpg4/bin/date [-u]
    [ [mmdd] HHMM | mmddHHMM [cc] yy] [.SS]

```

機能説明 date ユーティリティーは、日付と時間を標準出力に出力するほか、システムの日付と時間の設定も行います。デフォルトでは、現在の日付と時間を出力します。

月や曜日の名前を各国語に変換する仕様がサポートされています。使用される月や曜日の名前は、環境変数 LC_TIME で指定されたロケールに基づいて決定します ([environ\(5\)](#) 参照)。

c ロケールのデフォルト形式は次のようになっています。

```
%a %b %e %T %Z %Y
```

たとえば、次のように指定します。

```
Fri Dec 23 10:10:42 EST 1988
```

オプション サポートしているオプションは、次のとおりです。

- a [-]sss.fff システムクロックをゆっくり調整します。調整の単位は sss.fff 秒です (fff は秒の小数部)。調整は、プラスとマイナスの両方が可能です。システムクロックは、指定された秒だけ速くまたは遅くなります。日付の調整を行うことができるのはスーパーユーザーだけです。
- u 日付の表示や設定を行う際に、通常のローカル時間に変換しないでグリニッジ標準時間 (GMT - 標準時間) を使用します。

オペランド 次のオペランドがサポートされています。

+format 引数が + で始まる場合、date の出力は、format と現時間を strftime() に渡すことによって得られる値になります。date は、[strftime\(3C\)](#) のマニュアルページにある変換規則と、%c の変換規則を使用します。この変換規則は、/usr/bin/date と /usr/xpg4/bin/date のどちらが使われているかにより異なります。

```

/usr/bin/date      ロケールの日付と時間の表現。date のデフォルト出力

```

/usr/xpg4/bin/date 西暦 (年の上2桁)。つまり、年を100で割って端数を切り捨てた数字 (00-99)。

さらに、日付には %N もサポートされています。これは、Epoch (1970 年 1 月 1 日の 00:00:00 UTC) からの現在時間のナノ秒部分を 10 進数 [000000000-999999999] で表現します。この変換仕様では、strftime() で指定されているオプションのフラグ文字、オプションのフィールド幅、またはその両方を指定できます。ただし、指定されたフィールド幅が9未満の場合、実際の日付出力には、ナノ秒部分の左から指定された桁数の数字だけが含まれています。

文字列は常に復帰改行文字で終わります。引数に空白文字が含まれるときは、引用符で囲んでください。詳細については「使用例」の項を参照してください。

<i>mm</i>	月
<i>dd</i>	日
<i>HH</i>	時 (24 時 (間) 表示)
<i>MM</i>	分
<i>SS</i>	秒
<i>cc</i>	西暦 (年の上2桁)。つまり、年を100で割って端数を切り捨てた数字 (00-99)。たとえば、1988 年の場合、 <i>cc</i> は 19 で、2007 年の場合、 <i>cc</i> は 20 です。
<i>yy</i>	年の下2桁。 <i>cc</i> を指定しない場合、「69-99」という範囲は「1969 年から 1999 年まで (1969 年と 1999 年を含む)」を示し、「00-68」の範囲は「2000 年から 2068 年まで (2000 年と 2068 年を含む)」を示します。

月 (*mm*)、日 (*dd*)、年 (*yy*)、世紀 (*cc*) は省略できます。この場合は、現在の値がデフォルト値になります。例を示します。

example% **date 10080045**

この例では、日付と時刻が 10 月 8 日午前 12 時 45 分に設定されます。年を指定していないので、現在の年がデフォルトで使用されます。システムは GMT で動作していますが、**date** はローカルの標準時間の変換を処理します。日付を変更することができるのは、スーパーユーザーだけです。日付と時間を正しく設定し終わると、**date** は、デフォルトのフォーマットで新しい日付を表示します。また、**date** コマンドは正しいタイムゾーン情報を決定するために TZ を使用しています ([environ\(5\)](#) 参照)。

- 使用例

例1 出力の生成

次のコマンドを見てください。

```
example% date '+DATE: %m/%d/%y%nTIME:%H:%M:%S'
```

このコマンドによって、次の内容が出力されます。

```
DATE: 08/01/76
```

```
TIME: 14:45:05
```

例2 現在の時間の設定

次のコマンドは現在の時間を 12:34:56 に設定します。

```
example# date 1234.56
```

例3 グリニッジ平均時による現在の時間の設定

次のコマンドは現在の時刻を 2000 年 1 月 1 日、午前 12 時 30 分に設定します。

```
example# date -u 010100302000
```

コマンドを実行すると、次のように表示されます。

```
Thu Jan 01 00:30:00 GMT 2000
```
- 環境変数

date の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_TIME、LC_MESSAGES、およびNLSPATH。

TZ

-u オプションが省略されたときに、時間と日付の出力に使用するタイムゾーンを指定します。この TZ 変数が設定されず、-u も省略されている場合は、システムのデフォルトのタイムゾーンが使用されます。
- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/date	属性タイプ	属性値
	使用条件	system/core-os
	CSI	有効

/usr/xpg4/bin/date

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[strftime\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断

no permission スーパーユーザーではないので日付を変更できません。

bad conversion 日付設定の構文が正しくありません。

注意事項

標準タイムゾーンから別のタイムゾーンに切り替わる日 (たとえば、夏時間が始まる日または終わる日) を現在の日付とし、標準タイムゾーンの時間から別のタイムゾーンの時間に切り替わることを現在の時間にしようとした場合の結果は、保証されません。

システムの日時の変更を行うために、ウィンドウ環境で `date` コマンドを使用すると、不具合が発生する可能性があります。この場合の結果は保証されません。また、ウィンドウ環境以外のマルチユーザーモードでも、システムの日時を大幅に変更したりすると、結果が不安定になることがあります。システムの日時を変更するには、コマンド `date -a` を使用することをお勧めします。

03:14:07 UTC Jan 19, 2038 を超える日時にシステム日時を設定すること、つまりその日時を越えてシステム日時が進行することは、Solaris ではサポートされていません。

名前 dc – 電卓機能

形式 /usr/bin/dc [filename]
/usr/xpg6/bin/dc [filename]

機能説明 dc は、任意の精度の算術演算パッケージです。通常 dc は、10 進整数を処理しますが、入力の数値、出力の数値、および少数の桁数を指定することもできます。dc の全体構造は、スタック式(逆ポーランド式)計算機です。引数を指定すると、入力は指定したファイルから読み取られ、ファイルの終わりに到達すると標準入力から読み取られます。

bc は、中置記法と、機能を実装するための C 言語スタイルの構文を備えた、dc のプリプロセッサです。bc は、プログラムに対する合理的な制御構造も規定しています。詳細については、[bc\(1\)](#) を参照してください。

使用法

/usr/bin/dc,
/usr/xpg6/bin/dc

次の構文は、/usr/bin/dc と /usr/xpg6/bin/dc の両方で認識されます。

- number* 数値が、スタックにプッシュされます。数は、0-9 の数字からなる連続した文字列です。負の数を入力するには、下線 (_) を前に付けます。数値には、小数点が含まれる場合があります。
- sx* スタックの先頭の値がポップされ、レジスタ *x* に格納されます。ここで *x* は任意の文字です。*s* を大文字にすると、*x* はスタックとみなされ、値はそこにプッシュされます。
- lx* レジスタ *x* の値がスタックにプッシュされます。レジスタ *x* は変更されません。すべてのレジスタは、最初は 0 に設定されます。*l* を大文字にすると、レジスタ *x* はスタックとみなされ、その先頭の値がポップされ、メインスタックにプッシュされます。
- d* スタックの先頭の値が複製されます。
- p* スタックの先頭の値が出力されます。先頭の値はそのままです。
- P* スタックの先頭の値を ASCII 文字列として解釈し、それをスタックから削除し、出力します。
- f* スタック上のすべての値が出力されます。
- q* プログラムを終了します。文字列を処理している場合、再帰レベルは 2 レベルだけポップします。
- Q* プログラムを終了します。スタックの先頭の値がポップされ、文字列実行レベルはその値の分ポップします。
- x* スタックの先頭要素を文字列とみなし、それを dc コマンドの文字列として実行します。

<code>x</code>	スタックの先頭の値をそのスケールファクタで置き換えます。
<code>[...]</code>	角括弧で囲まれた ASCII 文字列をスタックの先頭に入れます。
<code><x>x=x</code>	スタックの先頭の要素 2 個がポップされ、比較されます。レジスタ <i>x</i> は、指定された関係が成立していれば評価されます。
<code>v</code>	スタックの先頭の要素をその平方根で置き換えます。引数に小数部分があるときは、それが考慮されますが、スケールファクタは無視されます。
<code>!</code>	残りの行をシェルコマンドとして解釈します。
<code>c</code>	スタック上のすべての値がポップされます。
<code>i</code>	スタックの先頭の値がポップされ、以後の入力の基数として使用されます。
<code>I</code>	入力の基数をスタックの先頭にプッシュします。
<code>o</code>	スタックの先頭の値がポップされ、以後の出力の基数として使用されます。
<code>O</code>	スタックの先頭に出力の基数をプッシュします。
<code>k</code>	スタックの先頭の値がポップされ、その値が負ではないスケールファクタとして使用されます。出力の際には、適切な桁数の保持に、また乗算、除算、および累乗の際には、位置取りの保持に使用されます。スケールファクタ、入力の基数、および出力の基数の相互作用は、これらすべてを一度に変更した場合には妥当なものになります。
<code>K</code>	現在のスケールファクタをスタックの先頭にプッシュします。
<code>z</code>	スタックレベルをスタックにプッシュします。
<code>Z</code>	スタックの先頭の数その長さを置き換えます。
<code>?</code>	入力行が入力ソース (通常は端末) から取られて実行されます。
<code>Y</code>	dc デバッグ情報を表示します。
<code>; :</code>	<code>bc(1)</code> が配列演算に使用します。

`/usr/bin/dc`

次の構文は、`/usr/bin/dc` で認識されます。どのような場合でも、結果のスケールが使用されます。

`+ - / * % ^` スタックの先頭の 2 つの値が、加算 (+)、減算 (-)、乗算 (*)、除算 (/)、剰余計算 (%)、または累乗 (^) されます。2 つの値は、スタックからポップされます。結果は、スタックのその場所にプッシュされます。指数の小数部分は無視されます。

/usr/xpg6/bin/dc

次の構文は、/usr/xpg6/bin/dc で認識されます。除算の結果には、20 のスケールが使用されます。

+ - / * % ^ スタックの先頭の2つの値が、加算(+)、減算(-)、乗算(*)、除算(/)、剰余計算(%)、または累乗(^)されます。2つの値は、スタックからポップされます。結果は、スタックのその場所にプッシュされます。指数の小数部分は無視されます。

除算の前に設定されたスケールが結果のスケールになります。

使用例

例1 n! の最初の10個の値を出力

この例では、n! の最初の10個の値を出力します:

```
[!a1+dsa*pla10>y]sy
0sa1
lyx
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[bc\(1\)](#), [attributes\(5\)](#)

診断

x is unimplemented	x は 8 進数です。
out of space	空きリストを使い果たしました (桁数が多すぎます)。
out of stack space	スタックへのプッシュが多すぎます (スタックのオーバーフロー)。
empty stack	スタックからのポップが多すぎます (スタックのアンダーフロー)。
nesting depth	ネストが深すぎます。
divide by 0	0 による除算です。
sqrt of neg number	負の数の平方根は定義されていません (虚数は扱えません)。
exp not an integer	dc は整数の累乗だけを処理します。
exp too big	最大許容指数は 999 です。
input base is too large	入力の基数 x: 2<= x <= 16。
input base is too small	入力の基数 x: 2<= x <= 16。

output base is too large	出力の基数は、BC_BASE_MAX 以下でなければなりません。
invalid scale factor	スケールファクタは1以上でなければなりません。
scale factor is too large	スケールファクタはBC_SCALE_MAX 以下でなければなりません。
symbol table overflow	指定された変数が多すぎます。
invalid index	インデックスは1以上でなければなりません。
index is too large	インデックスは、BC_DIM_MAX 以下でなければなりません。

名前	deallocate – デバイスの解放
形式	<pre>deallocate [-s] [-w] [-F] [-z zonename] [-c dev-class -g dev-type device] deallocate [-s] [-w] [-F] [-z zonename] -I</pre>
機能説明	deallocate コマンドは、割り当てられたデバイスを解放します。これは、デバイスに関連付けられているすべてのデバイス特殊ファイルの所有権とアクセス権をリセットし、そのデバイスへのアクセスを無効にします。deallocate は、device_allocate(4) に指定された内容に従って、そのデバイスのデバイスクリーニングプログラムを実行します。 デフォルトの deallocate 操作では、ユーザーに割り当てられているデバイスが解放されます。
オプション	サポートしているオプションは、次のとおりです。 -c dev-class 指定されたデバイスクラスのすべてのデバイスを解放します。 -F device <i>device</i> で指定されたファイルに関連付けられているデバイスを強制的に解放します。solaris.device.revoke 承認を保持するユーザーだけがこのオプションを使用できます。 -I すべての割り当て可能なデバイスを強制的に解放します。solaris.device.revoke 承認を保持するユーザーだけがこのオプションを使用できます。このオプションは、システムの初期化時のみに使用してください。 -s サイレント。診断出力をすべて抑止します。 Trusted Extensions を使用したシステムの構成時には、次のオプションがサポートされます。 -g dev-type <i>dev-type</i> に一致するデバイスタイプのデバイスを解放します。 -w ウィンドウ環境でデバイスのクリーニングプログラムを実行します。ウィンドウバージョンのプログラムが存在する場合は、そのプログラムが使用されます。そうでない場合は、標準バージョンが端末ウィンドウで実行されます。 -z zonename <i>zonename</i> で指定されたゾーンからデバイスを解放します。
オペランド	次のオペランドがサポートされています。 device <i>device</i> で指定されたデバイスを解放します。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。

20 指定したデバイスのエントリが存在しません。

ほかの値 エラーが発生した。

ファイル

/etc/security/device_allocate

/etc/security/device_maps

/etc/security/dev/*

/etc/security/lib/*

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

呼び出しは不確実です。オプションは不確実です。出力は非インタフェースです。

関連項目

[allocate\(1\)](#), [list_devices\(1\)](#), [device_allocate\(1M\)](#), [dminfo\(1M\)](#), [mkdevalloc\(1M\)](#), [mkdevmaps\(1M\)](#), [device_allocate\(4\)](#), [device_maps\(4\)](#), [attributes\(5\)](#)

Controlling Access to Devices

注意事項

このマニュアルページで説明している機能は、Solaris 監査が有効な場合にのみ使用できます。

このマニュアルページで説明している機能は、[device_allocate\(1M\)](#) サービスが有効な場合にのみ使用できます。

Trusted Extensions で構成されているシステムでは、この機能はデフォルトで有効です。

/etc/security/dev、[mkdevalloc\(1M\)](#)、および [mkdevmaps\(1M\)](#) は、Solaris オペレーティング環境の将来のリリースではサポートされない可能性があります。

名前	deroff - nroff/troff、tbl、およびeqn 構造体の削除
形式	deroff [-m [m s l]] [-w] [-i] [filename...]
機能説明	deroff は、各 <i>filenames</i> を順に読み取り、すべての troff(1) 要求、マクロ呼び出し、バックスラッシュ構文、 eqn(1) 構文 (.EQ 行と .EN 行の間、および区切り文字の間にある)、および tbl(1) 記述を削除し、それらをスペース (ブランクとブランク行) で置き換えて、ファイルの残りを標準出力に書き込みます。deroff は、インクルードされたファイルのチェーン (.so および .nx troff コマンド) に従います。ファイルがすでにインクルードされている場合、そのファイルを指定する .so は無視されて、そのファイルを指定する .nx は実行を終了します。入力ファイルが指定されていない場合、deroff は標準入力を読み込みます。
オプション	-m -m オプションの後には、m、s、または l を続けることができます。-mm オプションを使用するとマクロが解釈されて、実行テキストだけが出力されます (つまり、マクロ行からのテキストは出力されません)。-ml オプションは -mm オプションを強制し、mm マクロに対応するリストを削除します。 -w -w オプションが指定されている場合、出力はワードリストです。1 行ごとに 1 ワードが示され、他の文字はすべて削除されます。そうでない場合、出力は元のものに従い、前述のとおり削除されます。テキスト中の「ワード」は、少なくとも 2 つの文字を含み、英字、数字、アンパサント (&)、およびアポストロフィ (') からなる任意の文字列です。一方、マクロ呼び出しでは、「ワード」は少なくとも 2 文字から始まり、合計で 3 文字以上の文字列です。区切り文字は、英字、数字、アポストロフィ、およびアンパサント以外の任意の文字です。末尾のアポストロフィとアンパサントは、「ワード」から削除されます。 -i deroff に .so および .nx コマンドを無視させます。-
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目 [eqn\(1\)](#), [nroff\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#)

注意事項 deroff は、完全な troff インタプリタではないため、微妙な構文では混乱が生じることがあります。そのような場合、構文中の実際の問題を示すよりも多くのエラーメッセージ出力されます。

-ml オプションは、入れ子構造になっているリストを正しく処理できません。

名前	df – display status of disk space on file systems
形式	<code>/usr/ucb/df [-a] [-i] [-t type] [filesystem...] [filename...]</code>
機能説明	The <code>df</code> utility displays the amount of disk space occupied by currently mounted file systems, the amount of used and available space, and how much of the file system's total capacity has been used. If arguments to <code>df</code> are path names, <code>df</code> produces a report on the file system containing the named file. Thus '<code>df .</code>' shows the amount of space on the file system containing the current directory.
オプション	The following options are supported: -a Report on all filesystems including the uninteresting ones which have zero total blocks (that is, auto-mounter). -i Report the number of used and free inodes. Print '*' if no information is available. -t <i>type</i> Report on filesystems of a given type (for example, nfs or ufs).
使用例	例1 Using <code>df</code> A sample of output for <code>df</code> looks like: <pre>example% df Filesystem kbytes used avail capacity Mounted on sparky:/ 7445 4714 1986 70% / sparky:/usr 42277 35291 2758 93% /usr</pre> Note that <code>used+avail</code> is less than the amount of space in the file system (kbytes); this is because the system reserves a fraction of the space in the file system to allow its file system allocation routines to work well. The amount reserved is typically about 10%; this can be adjusted using <code>tunefs</code> (see tunefs(1M)). When all the space on a file system except for this reserve is in use, only the super-user can allocate new files and data blocks to existing files. When a file system is overallocated in this way, <code>df</code> can report that the file system is more than 100% utilized.
ファイル	<code>/etc/mnttab</code> List of file systems currently mounted <code>/etc/vfstab</code> List of default parameters for each file system
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [du\(1\)](#), [quot\(1M\)](#), [tunefs\(1M\)](#), [mnttab\(4\)](#), [attributes\(5\)](#)

名前 dhcpcinfo - DHCP を介して受信されたパラメータ値の表示

形式 dhcpcinfo [-c] [-i *interface*] [-n *limit*] [-v 4|6] *code*

dhcpcinfo [-c] [-i *interface*] [-n *limit*] [-v 4|6] *identifier*

機能説明 dhcpcinfo ユーティリティは、コマンド行で要求されたパラメータの DHCP 提供値を出力します。このパラメータは、DHCP 仕様の数値コードか、[dhcp_inittab\(4\)](#) にリストされているニモニック識別子のどちらかによって識別されます。このコマンドは、システムブート時に [init\(1M\)](#) によってブートされるシェルスクリプトのコマンド置換で使用されます。このコマンドは、システムブート時またはイベントスクリプト内で、DHCP クライアントデーモンと最初に通信を行います。詳細は、[dhcpcagent\(1M\)](#) を参照してください。このコマンドはまず DHCP クライアントデーモンの [dhcpcagent\(1M\)](#) と通信して、要求されたインタフェースで DHCP が正常に終了したかどうかを確認します。DHCP が、要求されたインタフェースで正常に終了していれば、dhcpcinfo は要求されたパラメータの値を取り出します。dhcpcinfo が表示したパラメータ値は、その終了ステータスを検査してから使用します。[exit\(1\)](#) を参照してください。

すべての DHCP パラメータのニモニック識別コードの一覧は、[dhcp_inittab\(4\)](#) を参照してください。DHCPv4 パラメータの詳細は、「RFC 2132, DHCP Options and BOOTP Vendor Extensions」を参照してください。DHCPv6 パラメータの詳細は、「RFC 3315, Dynamic Host Configuration Protocol for IPv6 (DHCPv6)」を参照してください。

出力形式 dhcpcinfo からの出力は、1 行または複数行の ASCII テキストからなります。出力の書式は、要求されたパラメータによって異なります。行ごとに返される値の数と、特定のパラメータで出力される行の合計数は、[dhcp_inittab\(4\)](#) で規定されているように、それぞれ、パラメータ指定の精度と最大値によって決まります。

各値の書式は、[dhcp_inittab\(4\)](#) で規定されているオプションのデータ型によって決まります。使用できるデータ型と書式は次のとおりです。

データ型	表記形式	dhcp_inittab(4) 型
符号なし数値	1 つまたは複数の 10 進数	UNUMBER8, UNUMBER16, UNUMBER32, UNUMBER64
符号付き数値	1 つまたは複数の 10 進数、負符号を付けることができる	SNUMBER8, SNUMBER16, SNUMBER32, SNUMBER64
IP アドレス	小数点付きの表記	IP
IPv6 アドレス	コロン区切りの表記	IPv6
オクテット	0x の後に 2 桁の 16 進数が続く文字列	OCTET
文字列	0 個またはそれ以上の ASCII 文字	ASCII

データ型	表記形式	dhcpc_inittab(4) 型
DUID	DHCP 一意識別子のテキスト	DUID
ドメイン名	RFC 1035 の書式に基づく標準の ドット区切りドメイン名	DOMAIN

オプション

サポートしているオプションは、次のとおりです。

- c 出力を標準の書式で表示します。これは、精度 1 の OCTET 書式と同じです。
- i interface DHCP パラメータの値を取り出すインタフェースを指定します。このオプションを指定しないと、プライマリインタフェースが使用されます。

ifconfig(1M) によりシステムのプライマリインタフェースが選択されていない場合、または -i によりこのコマンドのプライマリインタフェースが選択されていない場合、システムはインタフェースを自動的に選択してそれを現在のコマンド呼び出しのプライマリインタフェースとみなします。このときに選択されるのは、名前が語彙的に先頭にあり、DHCP パラメータが付加されているインタフェースです。選択内容が、システムの状態に影響を及ぼすことはありません。プライマリインタフェースの設定には、ifconfig(1M) を使用します。

dhcpcagent(1M) eventhook スクリプトを操作する場合には、プライマリ選択に依存するのではなく、-i を使用して目的のインタフェースを指定することをお勧めします。

DHCPv6 では、インタフェース名は、dhcpcagent により作成されたいずれかの論理インタフェースではなく、物理インタフェースの名前を使用してください。
- n limit 表示される値のリストを limit で指定された行数に制限します。
- v4 | 6 クエリー検索を行う DHCP バージョンを指定します。DHCPv4 では -v4 を、DHCPv6 では -v6 をそれぞれ使用します。

オペランド

次のオペランドがサポートされています。

- code DHCP 仕様で定義されている、要求された DHCP パラメータの数値コード。ベンダーオプションには、DHCPv4 では 256 を、DHCPv6 では 65536 をそれぞれ実際のベンダーコードに追加して指定します。
- identifier dhcpc_inittab(4) にリストされている、要求された DHCP パラメータのニーモニック記号

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 2 処理が正常に終了しなかった。DHCP クライアントデーモンが実行されていない、インタフェースが構成できなかった、あるいは十分な DHCP 応答が受信されなかった。
- 3 不正な引数。
- 4 処理がタイムアウトした。
- 6 起きるはずのないシステムエラーが発生しました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [dhcpcagent\(1M\)](#), [ifconfig\(1M\)](#), [init\(1M\)](#), [dhcp_inittab\(4\)](#), [attributes\(5\)](#)

Alexander, S., および R. 『RFC 2132, DHCP Options and BOOTP Vendor Extensions』、Alexander, S. (Silicon Graphics, Inc)、R. Droms (Bucknell University) 共著、1997 年 3 月

『RFC 3315, Dynamic Host Configuration Protocol for IPv6 (DHCPv6)』、Droms, R. (Cisco Systems) 著、2003 年 7 月

『RFC 1035, Domain names - implementation and specification』、Mockapetris, P.V. (ISI) 著、1987 年 11 月

名前 diff-2つのファイルの比較

形式 diff [-bitw] [-c | -e | -f | -h | -n | -u] *file1 file2*

diff [-bitw] [-C *number* | -U *number*] *file1 file2*

diff [-bitw] [-D *string*] *file1 file2*

diff [-bitw] [-c | -e | -f | -h | -n | -u] [-l] [-r] [-s]
[-S *name*] *directory1 directory2*

機能説明 diff ユーティリティーは *file1* と *file2* で指定された2つのファイルの内容を比較し、*file1* を *file2* に一致させるのに必要な変更リストを標準出力に書き込みます。変更リストの大きさは最小限に留められます。ごくまれな場合を除き、diff は、必要最低限の差異を生成するようにします。2つのファイルの内容が同一であれば、何も出力しません。

通常は、次のようなフォーマットの行を出力します。

```
n1 a n3,n4
n1,n2 d n3
n1,n2 c n3,n4
```

このうち *n1* と *n2* は *file1* 中の行を表し、*n3* と *n4* は *file2* 中の行を表します。これらの行は *file1* を *file2* へ変換するための **ed(1)** コマンドに似ています。a と d とを入れ替えて、各出力行の内容を右から左へ逆向きに読めば、*file2* を *file1* に変換する方法が確認できます。ed の場合と同様に、一致するペアは *n1=n2* または *n3=n4* のようになり、1つの番号に省略されます。

最初のファイルで影響を受けた行には、'<'が付きます。2番目のファイルで影響を受けた行には、'>'が付きます。

オプション サポートしているオプションは、次のとおりです。

- b 末尾のスペース(スペースとタブ文字)を無視し、他のスペース列は等価に扱います。
- i 大文字、小文字の区別を無視します。たとえば、A と a は等しいとみなします。
- t タブ文字を出力行上で展開します。通常または -c による出力は、各行の先頭に文字を追加するので、元の行のインデントに悪影響を与え、結果として出力行を読むにくくします。このオプションは、元の行のインデントを保持します。
- w すべてのスペース(スペース文字とタブ文字)を無視し、他のスペース文字は等価に扱います。たとえば、'if (a = b)' は 'if(a= =b)' と同じとみなします。

次のオプションは相互に排他的です。

- c 差異行の前後3行を追加して出力します。このオプションを使用すると、出力フォーマットが多少変わります。つまり、出力はファイルのIDと作成日で始まり、各差異部分は*からなる行で区切られます。*file1*から削除された行には「-」が付きます。*file2*に追加された行には+が付きます。1つのファイルから他のファイルへ変更された行には、双方のファイルに「!」が付きます'。
- C *number* 基本的に -c オプションと同じです。差異行の前後 *number* 行を追加して出力します。
- D *string* Cのプリプロセッサの制御によって *file1* と *file2* のマージ版を作成します。*string* を定義しないでコンパイルすると *file1* のコンパイル結果と同等になり、*string* を定義すると、*file2* を生成します。
- e *file1* から *file2* を再生するために、エディタ *ed* の *a*、*c*、*d* コマンドのスクリプトを生成します。次のシェルスクリプトは、この -e オプションを使用して、複数版のファイルを管理するものです。元のファイル(\$1)と *diff* によって作成される版から版への *ed* スクリプト(\$2,\$3,...)だけが必要です。最新版が標準出力へ表示されます。
- ```
(shift; cat $*; echo ' 1,$p') | ed - $1
```
- f 類似したスクリプト(*ed* では使えない)を反対順に生成します。
- h 高速で単調な処理を行います。
- このオプションが有効に機能するのは、変更範囲が小さく、十分に区別されている場合です。無限長のファイルでは機能しません。
- h とともに使用できるのは --b だけです。
- このオプションを指定した場合、*diff* はディレクトリの下階層を処理しません。
- n -e と同様のスクリプトを生成しますが、順序は反対で、各挿入または削除コマンドに変更行数を出力します。
- u 差異行の前後3行を追加して出力します。-c オプションと似ていますが、*u* オプションの場合、文脈が統合されて出力されます。つまり、*file1* で削除または変更された行には - が付き、*file2* で追加または変更された行には+が付きます。そして、変更された行は両方のファイルとも出力されますが、追加、削除、および同一の行は1度だけ出力されます。また、*file1* と *file2* の分別も異なります。-c オプションの場合は \*\*\* と —— が出力されますが、-u オプションの場合は —— と +++ が出力されます。変更された行はそれぞれ次のような行で分離されます。

```
@@ -n1,n2 +n3,n4 @@
```

**-U *number*** 出力形式は **-u** オプションと同じですが、出力する文脈の行数が *number* になります。

次のオプションは、ディレクトリを比較するのに使用します。

**-l** ロングフォーマットで出力を生成します。diff で処理する前に、各テキストファイルは **pr(1)** によってページ付けされます。他の差異は記憶され、すべてのテキストファイルの差異が報告された後で集計されます。

**-r** サブディレクトリに出会うと、diff を再帰的に実行します。

**-s** 同一のファイルを報告します。そうしないと、同一のファイルは報告されません。

**-S *name*** ファイル *name* からディレクトリ diff を開始します。

オペランド 次のオペランドがサポートされています。

*file1*  
*file2* 比較するファイルまたはディレクトリのパス名。*file1* と *file2* のどちらかに **-** を指定すると、そのファイルの代わりに標準入力を使用されます。

*directory1*  
*directory2* 比較するディレクトリのパス名

*file1* と *file2* のうちどちらか一方だけがディレクトリの場合、diff が比較するのは、ディレクトリでない方のファイルの内容と、当該ディレクトリ中のファイルのうち名前がディレクトリでない方のファイル名の最終コンポーネントと同一のファイルの内容です。

使用法 ファイルが2ギガバイト ( $2^{31}$  バイト) 以上ある場合の diff の動作については、[largefile\(5\)](#) を参照してください。

使用例 例1 diff コマンドを使用する  
次のコマンドでは、*dir1* がディレクトリで *x* という名のディレクトリを含んでいて、*dir2* もディレクトリで *x* という名のディレクトリを含んでいて、*dir1/x* と *dir2/x* の両方が *date.out* というファイルを含んでいて、さらに *dir2/x* が *y* というファイルを含んでいると仮定します。以下のような出力が生成されます。

```
example% diff -r dir1 dir2
Common subdirectories: dir1/x and dir2/x
```

```
Only in dir2/x: y
```

```
diff -r dir1/x/date.out dir2/x/date.out
```



例1 diff コマンドを使用する (続き)

1c1

< Mon Jul 2 13:12:16 PDT 1990

---

> Tue Jun 19 21:41:39 PDT 1990

## 環境変数

diff の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、LC\_TIME、およびNLSPATH。

TZ -c または -c オプションで出力される時間表示のタイムゾーン(タイムゾーン)に影響を与えるロケールを指定します。

## 終了ステータス

次の終了ステータスが返されます。

0 違いは見つかりませんでした。

1 違いが見つかりました。

>1 エラーが発生した。

## ファイル

/tmp/d????? 比較に用いる一時ファイル

/usr/lib/diffh -h オプション用の実行可能ファイル

## 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

## 関連項目

[bdiff\(1\)](#), [cmp\(1\)](#), [comm\(1\)](#), [dircmp\(1\)](#), [ed\(1\)](#), [pr\(1\)](#), [sdiff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

## 注意事項

オプション -e または -f 指定で生成される編集用スクリプトは、ピリオド(.) だけからなる行の作成に関しては慎重です。

Missing NEWLINE at end of file は、問題のファイルの最終行に復帰改行がないことを意味します。最終行に差異がある場合、フラグが付けられて出力されます。しかし、出力には差異がないように見えます。

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | diff3 - 3つのファイルの内容比較                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 形式   | diff3 [-exEX3] filename1 filename2 filename3                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 機能説明 | <p>diff3 は、1つのファイルの3つのバージョンを比較します。異なった内容を持つ範囲を次のようなコードを使って通知します。</p> <pre>====      全ファイルが異なる ====1     filename1 が異なる ====2     filename2 が異なる ====3     filename3 が異なる</pre> <p>示された範囲の内容をどのように変更すれば比較対象のほかのファイルの内容と一致させられるかは、次のいずれかの方法で表されます。</p> <p><i>f</i> : <i>n1</i> a            ファイル<i>f</i>の行番号<i>n1</i>の後にテキストを追加します。<i>f</i>は1、2、または3</p> <p><i>f</i> : <i>n1</i> , <i>n2</i> c        行番号<i>n1</i>から<i>n2</i>の範囲のテキストを変更します。<i>n1</i>と<i>n2</i>が等しい場合には、この範囲表記が省略されて、<i>n1</i>となります。</p> <p>示された範囲の元の内容が文字<i>c</i>の後に出力されます。2つのファイルの内容が同一のときは、小さい方の番号のファイルの内容は省略されます。</p> <p>次に示すコマンドは、その実行により生成されたスクリプトを<i>filename1</i>に適用します。</p> <pre>(cat script; echo ' 1,\$'p)   ed - filename1</pre> <p>オプション</p> <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-e    <i>filename2</i> と <i>filename3</i> との間のすべての相違点、つまり通常であればフラグ==== と ====3 とで示される相違点を <i>filename1</i> に組み込むための、ed(1) エディタ用スクリプトを生成します。</li> <li>      単一のドット(.)だけから成るテキスト行は、-e オプションを無効にします。</li> <li>-E    <i>filename2</i> と <i>filename3</i> との間のすべての相違点を組み込むためのスクリプトを生成しますが、その際、重複する相違点(つまり通常の一覧でフラグ====で示される相違点)を異なる方法で処理します。両ファイルからの重複する行は、編集スクリプトにより、&lt;&lt;&lt;&lt;&lt;&lt; 行と &gt;&gt;&gt;&gt;&gt;&gt; 行で囲まれて挿入されず。</li> <li>-x    ==== フラグにより示される相違箇所だけを取り込んだスクリプトを生成します。</li> </ul> |

- X    フラグ `====` で示される相違点だけを組み込むためのスクリプトを生成しますが、その際、`-E` オプションと同じ方法でそれらの相違点を処理します。
- 3    `====3` フラグにより示される相違箇所だけを取り込んだスクリプトを生成します。

**使用法**            ファイルが2ギガバイト ( $2^{31}$  バイト) 以上ある場合の `diff3` の動作については、[largefile\(5\)](#) を参照してください。

**ファイル**            `/tmp/d3*`  
                       `/usr/lib/diff3prog`

**終了ステータス**    次の終了ステータスが返されます。

- 0      正常終了。
- 1      エラーが発生した。差異が見つかったか、致命的なエラーが発生しました。
- >1    致命的なエラーが発生しました。

戻り値は、[diff\(1\)](#) やほかのベンダー版の `diff3` とは異なります。

**属性**                属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

**関連項目**            [diff\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

**注意事項**            64K バイトを超える大きさのファイルは処理できません。

|      |                                                                                                                                                                                                                                                                                                                                                                                                                      |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | diffmk - troff 入力ファイル間の差異の表示                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式   | diffmk <i>oldfile newfile markedfile</i>                                                                                                                                                                                                                                                                                                                                                                             |
| 機能説明 | <p>diffmk は、2つのバージョンのファイルを比較して、troff(1) および nroff(1) に対する変更マーク (.mc) コマンドを含む第3のバージョンを作成します。oldfile と newfile は、ファイルの新旧のバージョンを示します。diffmk は、newfile と oldfile が異なる場合に、troff(1) の変更マーク要求 (.mc) を挿入した newfile からテキスト形式の markedfile を生成します。markedfile が書式化されたときは、変更されたり挿入されたテキストの各行の右端に   が表示されます。削除されたテキストの位置には、1つの * が表示されます。</p>                                                                                      |
| 使用法  | <p>ファイルのサイズが 2G バイト (2<sup>31</sup> バイト) 以上ある場合の diffmk の動作については、largefile(5) を参照してください。</p>                                                                                                                                                                                                                                                                                                                         |
| 使用例  | <p>例1 diffmk コマンドの使用例</p> <p>diffmk を適切な troff 要求と共に使用して、変更がマークされたプログラムリストを作成することもできます。次のコマンド行を見てください。</p> <pre>example% diffmk old.c new.c marked.c ; nroff reqs marked.c   pr</pre> <p>ファイル reqs には、次の troff 要求が記述されています。</p> <pre>.pl 1 .ll 77 .nf .eo .nh</pre> <p>これらの要求はそれぞれ、改ページを削除し、行の長さを調整し、「詰め込みなし」モードを設定し、エスケープ文字を無視して、ハイフネーションを無効にします。</p> <p>文字   と * が不適切な場合は、sed(1) を使用して markedfile を全体的に変更できます。</p> |
| 属性   | <p>属性についての詳細は、マニュアルページの attributes(5) を参照してください。</p>                                                                                                                                                                                                                                                                                                                                                                 |

| 属性タイプ | 属性値           |
|-------|---------------|
| 使用条件  | text/doctools |

|         |                                                                                                                                                 |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| 関連項目    | diff(1), nroff(1), sed(1), troff(1), attributes(5), largefile(5)                                                                                |
| 使用上の留意点 | <p>体裁を整えるために、一部の出力を手動で調整しなければならない場合があります。書式に関する要求だけに差違がある場合、期待するような出力が得られないことがあります。たとえば、.sp が .sp 2 に変更されている場合、出力行の前後の行に変更マークが付けられることがあります。</p> |

|        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前     | digest - メッセージダイジェストの計算                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 形式     | <code>/usr/bin/digest -l   [-v] -a <i>algorithm</i> [<i>file</i>]...</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 機能説明   | <code>digest</code> ユーティリティーは、指定されたアルゴリズムを使用して、指定されたファイルまたは <code>stdin</code> のメッセージダイジェストを計算します。複数のファイルが指定された場合、出力の各行が単一ファイルのダイジェストになります。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| オプション  | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-a <i>algorithm</i></b>      暗号化または復号化プロセスで使用するアルゴリズムの名前を指定します。詳細については、「使用法」の「Algorithms」を参照してください。</p> <p><b>-l</b>                      システムで利用可能なアルゴリズムの一覧を表示します。この一覧は、暗号化フレームワークの構成によって変わります。</p> <p><b>-v</b>                      詳細情報を出力します。アルゴリズム名とファイル名を出力に含めます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 使用法    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| アルゴリズム | <p>これらのアルゴリズムは、暗号化フレームワークによって提供されます。このコマンドでサポートされる各アルゴリズムは、アクセスを容易にするための PKCS #11 メカニズムの別名です。たとえば、<code>sha1</code> は <code>CKM_SHA_1</code> の別名です。</p> <p>これらの別名は <b>-a</b> オプションとともに使用され、大文字と小文字が区別されません。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 使用例    | <p><b>例1 出力をシミュレートする</b></p> <p>次の例では、一般的な <code>md5sum</code> プログラムの出力をシミュレートします。</p> <pre>example\$ digest -v -a md5 /usr/bin/vi md5 (/usr/bin/vi) = e4e3588c5212903847c66d36b1a828a5</pre> <p><b>例2 ファイルのダイジェストを生成する</b></p> <p>次の例では、ファイル <code>/etc/motd</code> の <code>sha1</code> ダイジェストを生成します。</p> <pre>example\$ digest -a sha1 /etc/motd 9498a4f5303d056ad3ecae826b59f41448d63790</pre> <p><b>例3 ディレクトリマニフェストを生成する</b></p> <p>次の例では、<code>sha1</code> を使用してディレクトリマニフェストを生成します。</p> <pre>example\$ digest -v -a sha1 /usr/lib/inet/* sha1 (/usr/lib/inet/certdb) = f6d43e6e395d50db24d34e4af4828598c8918b16 sha1 (/usr/lib/inet/certlocal) = 7f74ba4a019b809c7023212b4bda10d9485e071d sha1 (/usr/lib/inet/certrldb) = 1f845d30b8d02066647de04311e74549049852ed</pre> |

例3 ディレクトリマニフェストを生成する (続き)

```
sha1 (/usr/lib/inet/dhcp) = e3db5e4ff40a69d13f2497254526c2015d2c37b3
sha1 (/usr/lib/inet/dsvclockd) = b61aad7ed6a0f82145c3c26aedc613ab4a1f032e
sha1 (/usr/lib/inet/in.dhcpd) = 382210180c826fbb2e747236c489062bac8cc30b
sha1 (/usr/lib/inet/in.iked) = be6061fad725d37256e773dc85f8bd5248649463
sha1 (/usr/lib/inet/in.mpathd) = 5bd6bf0340fd5c4cc0c53f2df158302a0e85f9d0
sha1 (/usr/lib/inet/in.ndpd) = fdb768aebe7e5eb4465e1c1bb5e679b496f5c5c6
sha1 (/usr/lib/inet/in.ripngd) = 4f56a0df2d4a252f581a73c2e84143b920d0b66b
sha1 (/usr/lib/inet/ncacnfd) = 7219542b5585a8d1104d7ce4a2ced07d8a260ea3
sha1 (/usr/lib/inet/ppp) = c96ee458549871a6ffdf2674a888b01d0c9e9740
sha1 (/usr/lib/inet/pppoe) = 5f022498d79dacacd947cddadc64f171822e3dee
sha1 (/usr/lib/inet/pppoe) = 252bd2f0863dbc1b05fffae72821a2a95609b8ad
sha1 (/usr/lib/inet/slpad) = dfa24cc0f0b05f790546d4f0948a9094f7089027
sha1 (/usr/lib/inet/wanboot) = a8b8c51c389c774d0be2ae43cb85d1b1439484ae
sha1 (/usr/lib/inet/ntpd) = 5b4aff102372cea801e7d08acde9655fec81f07c
```

例4 利用可能なアルゴリズムの一覧を表示する

次の例では、ダイジェストに利用可能なアルゴリズムの一覧を表示します。

```
example$ digest -l
sha1
md5
sha224
sha256
sha384
sha512
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 確実             |

関連項目 [cksum\(1\)](#), [encrypt\(1\)](#), [mac\(1\)](#), [bart\(1M\)](#), [cryptoadm\(1M\)](#), [libpkcs11\(3LIB\)](#), [attributes\(5\)](#), [pkcs11\\_softtoken\(5\)](#)

|         |                                                                                                                                                                                                                                                                     |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | dircmp – ディレクトリの比較                                                                                                                                                                                                                                                  |
| 形式      | dircmp [-ds] [-w <i>n</i> ] <i>dir1 dir2</i>                                                                                                                                                                                                                        |
| 機能説明    | dircmp コマンドは、 <i>dir1</i> と <i>dir2</i> を調べて、ディレクトリの内容についての情報を表形式で生成します。各ディレクトリに固有なファイルのリストが、すべてのオプションについて生成されます。オプションを指定しないと、両方のディレクトリに共通して存在するファイルの内容が同じかどうかを示すリストが出力されます。                                                                                        |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <p>-d      両方のディレクトリで同じ名前を持つファイルの内容を比較して、2つのファイルを一致させるために変更すべき箇所のリストを出力します。リストの形式については、<a href="#">diff(1)</a> を参照してください。</p> <p>-s      同じファイルについてのメッセージを抑制します。</p> <p>-w <i>n</i>   出力行の幅を <i>n</i> 文字に変更します。デフォルトの幅は 72 文字です。</p> |
| オペランド   | <p>次のオペランドがサポートされています。</p> <p><i>dir1</i><br/> <i>dir2</i>    比較するディレクトリのパス名</p>                                                                                                                                                                                    |
| 使用法     | ファイルのサイズが 2G バイト ( $2^{31}$ バイト) 以上ある場合の dircmp の動作については、 <a href="#">largefile(5)</a> を参照してください。                                                                                                                                                                   |
| 環境変数    | dircmp の実行に影響を与える次の環境変数については、 <a href="#">environ(5)</a> を参照してください。LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。                                                                                                                                                     |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0      正常終了。</p> <p>&gt;0    エラーが発生した。(ディレクトリの内容の違いはエラーとはみなされません。)</p>                                                                                                                                                                   |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                       |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目      [cmp\(1\)](#), [diff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | dis - オブジェクトコードの逆アセンブラ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式    | dis [-onqCLV] [-d <i>sec</i> ] [-D <i>sec</i> ] [-F <i>function</i> ]<br>[-l <i>string</i> ] [-t <i>sec</i> ] <i>file</i> ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 機能説明  | dis コマンドは、 <i>file</i> のアセンブリ言語リストを作成します。 <i>file</i> は、オブジェクトファイルでもオブジェクトファイルのアーカイブでもかまいません。このリストには、アセンブリ文と、それらの文を生成したバイナリコードの 8 進数表記または 16 進数表記が含まれます。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| オプション | <p>オプションは、逆アセンブラによって解釈され、任意の順序で指定できます。</p> <p>サポートしているオプションは、次のとおりです。</p> <p>-C            逆アセンブリで復号化された C++ シンボル名を表示します。</p> <p>-d <i>sec</i>        指定されたセクションをデータとして逆アセンブルし、セクションの始めからのデータのオフセットを出力します。</p> <p>-D <i>sec</i>        指定されたセクションをデータとして逆アセンブルし、データの実アドレスを出力します。</p> <p>-F <i>function</i>    コマンド行に指定された各オブジェクトファイルの指定された関数だけを逆アセンブルします。-F オプションは、コマンド行で複数回指定できます。</p> <p>-l <i>string</i>      <i>string</i> で指定されたアーカイブファイルを逆アセンブルします。たとえば、コマンド <code>dis -l x -l z</code> を発行して、<code>LIBDIR</code> にあると想定される <code>libx.a</code> と <code>libz.a</code> を逆アセンブルします。</p> <p>              このオプションは廃止または、互換性が無くなる可能性があります、Solaris の将来のリリースでは、削除されるかもしれません。</p> <p>-L            以降の出力を標準出力へ書き込むために、シンボルテーブル内の C 言語ソースラベルの検索を行います。</p> <p>              このオプションは廃止または、互換性が無くなる可能性があります、Solaris の将来のリリースでは、削除されるかもしれません。</p> <p>-n            すべてのアドレスを数値で表示します。デフォルトでは、アドレスはシンボル名を使用して表示されます。</p> <p>-o            数字を 8 進数で出力します。デフォルトでは 16 進数です。</p> <p>-q            静寂モード。ヘッダーまたは関数の開始ラベルを出力しません。</p> <p>-t <i>sec</i>        指定されたセクションをテキストとして逆アセンブルします。</p> <p>-V            実行中の逆アセンブラのバージョン番号を標準エラー出力に出力します。</p> |



このオプションは廃止または、互換性が無くなる可能性があります。  
り、Solaris の将来のリリースでは、削除されるかもしれません。

-d、-D、または -t オプションを指定した場合、ユーザーが指定した各ファイルの指定されたセクションだけが逆アセンブルされます。そうしないと、テキストを含むすべてのセクションが逆アセンブルされます。

出力において、[5] のように、行の始めにある角括弧で囲まれた数字は、ブレークポイント可能な行番号が次の命令から始まることを示します。これらの行番号が出力されるのは、デバッグ情報を追加指定してファイルをコンパイルした場合だけです。

オペランド 次のオペランドを指定できます。

*file* オブジェクトファイルまたはオブジェクトファイルのアーカイブ ([ar\(1\)](#) を参照) のパス名。

環境変数 `dis` の実行に影響を与える次の環境変数については、[environ\(5\)](#) を参照してください。LC\_CTYPE、LC\_MESSAGES、および NLSPATH。

**LIBDIR** この環境変数に値が設定されている場合は、それをライブラリ検索用のパスとして使用します。この環境変数に NULL の値が設定されている場合、あるいは、この環境変数が設定されていない場合は、デフォルトで `/usr/lib` にあるライブラリが検索されます。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

ファイル `/usr/lib` デフォルトの **LIBDIR**

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                |
|-------------|------------------------------------|
| 使用条件        | developer/base-developer-utilities |
| インタフェースの安定性 | 下記を参照。                             |

人間が読める形式の出力は「不確実」です。コマンド行オプションは「確実」です。

関連項目 [ar\(1\)](#), [as\(1\)](#), [ld\(1\)](#), [a.out\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#)

**診断**

コマンド行中のエラー、または指定したファイルで検出された問題は、診断メッセージに示されます。

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | disown – ksh 組み込み関数を使用したジョブと現在のシェルとの関連付けの解除                                                                                                                                                                                                                                                                                                                                                                                       |
| 形式      | disown [job ...]                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明    | <p>ksh disown コマンドは、現在のシェルによるログインセッションの終了時に、現在のシェルが指定の各ジョブに HUP シグナルを送信するのを防止します。</p> <p>job を省略すると、disown は最後に開始または停止されたバックグラウンドジョブに HUP シグナルを送信します。</p>                                                                                                                                                                                                                                                                         |
| オペランド   | <p>次のオペランドがサポートされています。</p> <p>job     disown が動作する 1 つまたは複数のジョブを指定します。</p> <p>job には次のいずれかを指定します。</p> <ul style="list-style-type: none"> <li>number       プロセス ID を示します。</li> <li>-number      プロセスグループ ID を示します。</li> <li>%number     ジョブ番号を示します。</li> <li>%string      名前が string で始まるジョブを示します。</li> <li>%%?string    名前に string を含むジョブを示します。</li> <li>%%+ または %%    現在のジョブを示します。</li> <li>%%-          前回のジョブを示します。</li> </ul> |
| 終了ステータス | <p>0     正常終了。</p> <p>&gt;0    指定された 1 つ以上のジョブが存在しません。</p>                                                                                                                                                                                                                                                                                                                                                                        |
| 使用例     | <p>例1 ジョブとの関連付けを解除する</p> <p>次の例では、ジョブ 1 との関連付けを解除します。</p> <pre>example% disown %1</pre>                                                                                                                                                                                                                                                                                                                                           |
| 著者      | David Korn、dkgk@research.att.com                                                                                                                                                                                                                                                                                                                                                                                                  |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                     |

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 不確実            |

関連項目     [bg\(1\)](#), [jobs\(1\)](#), [ksh\(1\)](#), [wait\(1\)](#), [attributes\(5\)](#)

|         |                                                               |
|---------|---------------------------------------------------------------|
| 名前      | dispgid - すべての有効なグループ名リストの表示                                  |
| 形式      | dispgid                                                       |
| 機能説明    | dispgid は、システム上のすべてのグループ名のリストを表示します (行ごとに1つのグループ)。            |
| 終了ステータス | 次の終了ステータスが返されます。<br>0    正常終了。<br>1    グループファイルが読み取れなかった。     |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。 |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目    [attributes\(5\)](#)

|         |                                                               |
|---------|---------------------------------------------------------------|
| 名前      | dispuid - すべての有効なユーザー名リストの表示                                  |
| 形式      | dispuid                                                       |
| 機能説明    | dispuid は、システム上のすべてのユーザー名のリストを表示します (名前ごとに 1 行)。              |
| 終了ステータス | 次の終了ステータスが返されます。<br>0 正常終了。<br>1 パスワードファイルが読み取れなかった。          |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。 |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | dos2unix – DOS 形式から ISO 形式へのテキストファイルの変換                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 形式    | dos2unix [-ascii] [-iso] [-7] [-437   -850   -860   -863   -865] <i>originalfile</i> <i>convertedfile</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 機能説明  | <p>dos2unix ユーティリティーは DOS 拡張文字セットの文字を、対応する ISO 標準文字に変換します。</p> <p>このコマンドは、DOS と SunOS のどちらからでも起動できます。ただしファイル名は、コマンドが起動された環境の命名規則を満たさなければなりません。</p> <p>元のファイルと変換されたファイルが同じである場合、dos2unix は、変換後のファイルで元のファイルを上書きします。</p>                                                                                                                                                                                                                                                                                                                                                                   |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-ascii    余分なキャリッジリターンを削除して、DOS 形式のテキストファイル内のファイルの終わり文字を、SunOS の要件に合うように変換します。</p> <p>-iso       これはデフォルトです。DOS 拡張文字セットの文字を、対応する ISO 標準文字に変換します。</p> <p>-7        8 ビットの DOS グラフィックス文字を 7 ビット空間文字に変換して、SunOS がファイルを読み取れるようにします。</p> <p>非 i386 システム上では、dos2unix はキーボードの種類を取得して、使用するコードページを決定します。そうでない場合、キーボードの種類はデフォルトで US です。使用するコードページを無効にするには、次のオプションを使用します。</p> <p>-437    米国英語のコードページを使用します。</p> <p>-850    複数言語のコードページを使用します。</p> <p>-860    ポルトガル語のコードページを使用します。</p> <p>-863    フランス系カナダ語のコードページを使用します。</p> <p>-865    デンマーク語のコードページを使用します。</p> |
| オペランド | <p>次のオペランドは必須です。</p> <p><i>originalfile</i>    ISO 形式に変換される DOS 形式の元のファイル</p> <p><i>convertedfile</i>    DOS 形式の元のファイルから変換された ISO 形式の新しいファイル</p>                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 属性    | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

## 関連項目

[unix2dos\(1\)](#), [ls\(1\)](#), [attributes\(5\)](#)

## 診断

**File *filename* not found, or no read permission**

指定した入力ファイルが存在しないか、読み取り権が与えられていません。SunOS の `ls -l` コマンド ([ls\(1\)](#) を参照) で検査します。

**Bad output filename *filename*, or no write permission**

指定された出力ファイルが無効であるか、あるいは、そのファイルまたはそのファイルを含むディレクトリの書き込み権がありません。また、ドライブが書き込み保護されていないか検査してください。

**Error while writing to temporary file**

現在のドライブに十分な領域がないために、ファイルの変換中にエラーが発生しました。DIR コマンドを使用して、現在のドライブ上の領域を検査してください。また、デフォルトのドライブが書き込み可能であることを (書き込み保護されていないことを) 確認してください。このエラーが発生した場合、元のファイルは変更されません。

**Translated temporary file name = *filename*.****Could not rename temporary file to *filename*.**

プログラムが、ファイル変換の最終段階を実行できませんでした。変換されたファイルは、前述のメッセージの 2 行目に示された名前で保存されます。

名前  
形式

dpost – PostScript プリント用の troff ポストプロセッサ

```
dpost [-c num] [-e num] [-m num] [-n num] [-o list] [-w num] [-x num] [-y num] [-F dir] [-H dir]
/usr/lib/lp/postscript/dpost
```

機能説明

dpost は、[troff\(1\)](#) によって作成された入力ファイル (*files*) を PostScript に変換して、結果を標準出力に書き込みます。入力ファイル (*files*) を指定しないか、または *-* が *files* の 1 つである場合は、標準入力を読み取られます。

*files* は、troff によって作成する必要があります。/usr/lib/font/devpost にあるデフォルトのフォントファイルは、最も適切で効率的な出力を作成します。これらのフォントファイルは、720 dpi の解像度を前提としており、*-Tpost* オプションを付けて troff を呼び出すことによってファイルのフォーマットに使用できます。以前のバージョンの eqn および pic プリプロセッサには、troff が *files* のフォーマットに使用する解像度を認識させる必要があります。使用しているシステムにこれらのバージョンがインストールされている場合は、eqn では *-r720* オプションを、pic では *-T720* を使用してください。

dpost は、解像度を想定していません。最初の *x res* コマンドは、入力ファイル DESC.out ファイルの変換に使用する解像度を設定します。一般に、/usr/lib/font/devpost/DESC.out は、バイナリフォントファイルに使用する解像度を定義します。また、PostScript プロローグは、適切なユーザー座標システムの設定を行います。

オプション

- c num*    各ページのコピーを *num* 部ずつ出力します。デフォルトでは、コピーは 1 部だけ出力されます。
- e num*    テキスト符号化レベルを *num* に設定します。認識される選択肢は 0、1、および 2 です。出力ファイルのサイズと出力時間は、*num* の値が増加すると減少します。レベル 2 の符号化は、一般にレベル 0 よりも約 20 パーセント高速です。レベル 0 がデフォルトであり、前のバージョンの dpost と基本的に同じ出力を作成します。
- m num*    各論理ページを、*num* で指定した係数で拡大します。ページは、各ページの左上端付近にある原点に対して均等にスケール変更されます。デフォルトの拡大率は 1.0 です。
- n num*    用紙 1 枚ごとに *num* で指定した数の論理ページを出力します。ここで、*num* には任意の正の整数を指定できます。デフォルトでは、*num* は 1 に設定されます。
- o list*    コンマで区切られた *list* 内で番号が与えられているページを出力します。このリストには、単一の番号 *N* と *N1-N2* の範囲が含まれます。*N1* がなければ最小番号のページを意味し、*N2* がなければ最大を意味します。ページ範囲は、物理的な用紙ではなく、論理ページを表現したものです。たとえば、2 つの論理ページを用紙に出力するとき、範囲 4 を指定した場合は、4 ページのレイアウトを含む 2 枚の用紙が出力されま



す。ページ範囲を 3 - 4 に指定して、2 つの論理ページを 1 枚の用紙に要求すると、ページ 3 と 4 のレイアウトだけが一枚の用紙に出力されます。

- p *mode*    *portrait* (縦方向) または *landscape* (横方向) のどちらかの *mode* でファイルを出力します。*mode* の最初の文字だけが有効です。デフォルトのモードは *portrait* です。
- w *num*    *troff* グラフィックスコマンドを *num* ポイントに導入するために使用するライン幅を設定します。ポイントは、1 インチの約 1/72 です。デフォルトでは、*num* は 0.3 ポイントに設定されます。
- x *num*    原点を正の x 軸に沿って *num* インチだけ変換します。デフォルトの座標システムでは、原点はページの左上端に固定されていて、正の x はページの右側、正の y はページの左側になります。正の *num* は、すべてを右に移動します。デフォルトのオフセットは 0 インチです。
- y *num*    原点を正の y 軸に沿って *num* インチだけ変換します。正の *num* はテキストをページ内で上に移動します。デフォルトのオフセットは 0 インチです。
- F *dir*    *dir* をフォントディレクトリとして使用します。デフォルトの *dir* は */usr/lib/font* であり、*dpost* はディレクトリ */usr/lib/font/devpost* からバイナリフォントファイルを読み込みます。
- H *dir*    *dir* をホスト上のフォントディレクトリとして使用します。このディレクトリのファイルは、完全な PostScript フォント記述でなければならない、適切な 2 文字の *troff* フォント名に対応する名前を割り当てる必要があります。各フォントファイルは、必要な場合にのみ、各ジョブの実行中に 1 回だけ出力ファイルにコピーされます。デフォルトのディレクトリはありません。
- L *file*    *file* を PostScript プロローグとして使用します。デフォルトは */usr/lib/lp/postscript/dpost.ps* です。
- O    PostScript ピクチャーのインクルードを無効にします。ネットワーク化された環境でスプーラが *dpost* を実行する場合に推奨されるオプションです。
- T *name*    デバイス *name* のフォントファイルを、使用可能な PostScript フォントの最善の記述として使用します。デフォルトでは、*name* は *post* に設定され、*dpost* は */usr/lib/font/devpost* からバイナリファイルを読み込みます。

## 使用例

例 1 *dpost* コマンドを使用する

使用しているシステム上に旧バージョンの *eqn* および *pic* がインストールされている場合は、次のようなコマンド行によって、最善の体裁の出力を得ることができます。

例1 dpost コマンドを使用する (続き)

```
example% pic -T720 file | tbl | eqn -r720 | troff -mm -Tpost | dpost
```

それ以外の場合は、次のステータスが返されます。

```
example% pic file | tbl | eqn | troff -mm -Tpost | dpost
```

このコマンド行でも最善の結果が得られます。

終了ステータス 次の終了ステータスが返されます。

- 0            正常終了。
- 0 以外      エラーが発生した。

ファイル

```
/usr/lib/font/devpost/*.out
/usr/lib/font/devpost/charlib/*
/usr/lib/lp/postscript/color.ps
/usr/lib/lp/postscript/draw.ps
/usr/lib/lp/postscript/forms.ps
/usr/lib/lp/postscript/ps.requests
/usr/lib/macros/pictures
/usr/lib/macros/color
```

属性            属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値                                  |
|-------|--------------------------------------|
| 使用条件  | print/lp/filter/postscript-lp-filter |

関連項目        [troff\(1\)](#), [attributes\(5\)](#)

注意事項        出力ファイルが、Adobe のファイル構造規則に準拠していないことがよくあります。

dpost は、どのデバイスに合わせてフォーマットされたファイルでも処理できますが、エミュレーションは負荷の重い処理であり、出力時間と出力ファイルのサイズは簡単に倍増してしまいます。そのため、troff がサポートするすべてのデバイスで利用できる文字セット、またはフォントを実装しようとする試みは、行われていません。欠落している文字はスペースで置き換えられ、認識不能なフォントは通常、Times フォント (R、I、B、またはBI) のいずれかにデフォルト設定されます。

x res コマンドは最初の x init コマンドの前になければならず、また、すべての入力ファイル (*files*) を同じ出力デバイスに合わせて用意する必要があります。

-T オプションは使用しないようにしてください。このオプションの唯一の目的は、異なる解像度、文字セット、またはフォントを使用する他の PostScript フォントとデバイス記述ファイルを使用できるようにすることです。

レベル0の符号化は完全にテストされている唯一のコード体系ですが、レベル2は高速であり、試してみる価値があります。

名前                    du – summarize disk usage

形式                    /usr/bin/du [-dorx] [-a | -s] [-h | -k | -m] [-H | -L]  
                         [file ...]  
  
                         /usr/xpg4/bin/du [-dorx] [-a | -s] [-h | -k | -m] [-H | -L]  
                         [file ...]

機能説明                The du utility writes to standard output the size of the file space allocated to, and the size of the file space allocated to each subdirectory of, the file hierarchy rooted in each of the specified files. The size of the file space allocated to a file of type directory is defined as the sum total of space allocated to all files in the file hierarchy rooted in the directory plus the space allocated to the directory itself. This sum will include the space allocated to any extended attributes encountered.

Files with multiple links will be counted and written for only one entry. The directory entry that is selected in the report is unspecified. By default, file sizes are written in 512-byte units, rounded up to the next 512-byte unit.

/usr/xpg4/bin/du        When du cannot obtain file attributes or read directories (see [stat\(2\)](#)), it will report an error condition and the final exit status will be affected.

オプション                The following options are supported for /usr/bin/du and /usr/xpg4/bin/du:

- a    In addition to the default output, report the size of each file not of type directory in the file hierarchy rooted in the specified file. Regardless of the presence of the -a option, non-directories given as *file* operands will always be listed.
- d    Do not cross filesystem boundaries. For example, the command, du -d / reports usage only on the root partition.
- h    All sizes are scaled to a human readable format, for example, 14K, 234M, 2.7G, or 3.0T. Scaling is done by repetitively dividing by 1024.
- H    If a symbolic link to a directory is specified on the command line, process the symbolic link by using the directory which the symbolic link references, rather than the link itself.
- k    Write the files sizes in units of 1024 bytes, rather than the default 512-byte units.
- L    Process symbolic links by using the file or directory which the symbolic link references, rather than the link itself.
- m    Write the files sizes in units of megabytes, rather than the default 512-byte units.
- o    Do not add child directories' usage to a parent's total. Without this option, the usage listed for a particular directory is the space taken by the files in that directory, as well as the files in all directories beneath it. This option does nothing if -s is used.

- r   Generate diagnostic messages about unreadable directories and files whose status cannot be obtained. /usr/bin/du is silent if these conditions arise and -r is not specified. /usr/xpg4/bin/du acts as though -r is always specified.
- s   Instead of the default output, report only the total sum for each of the specified files.
- x   When evaluating file sizes, evaluate only those files that have the same device as the file specified by the file operand.

Specifying more than one of the options in the mutually exclusive pair, -H and -L, is not considered an error. The last option specified determines the output format.

Specifying more than one of the options in the mutually exclusive set of options -h, -k, and -m is not considered an error. The last option specified determines the output format.

## オペランド

The following operand is supported:

*file*   The path name of a file whose size is to be written. If no *file* is specified, the current directory is used.

## Output

The output from du consists of the amount of the space allocated to a file and the name of the file.

## 使用法

See [largefile\(5\)](#) for the description of the behavior of du when encountering files greater than or equal to 2 Gbyte (  $2^{31}$  bytes).

## 環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of du: LANG, LC\_ALL, LC\_CTYPE, LC\_MESSAGES, and NLSPATH.

## 終了ステータス

The following exit values are returned:

- 0   Successful completion.
- >0   An error occurred.

## 属性

See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/du

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE |
|---------------------|-----------------|
| Availability        | system/core-os  |
| CSI                 | Enabled         |
| Interface Stability | Committed       |

/usr/xpg4/bin/du

| ATTRIBUTE TYPE | ATTRIBUTE VALUE   |
|----------------|-------------------|
| Availability   | system/xopen/xcu4 |

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE |
|---------------------|-----------------|
| CSI                 | Enabled         |
| Interface Stability | Standard        |

関連項目      [ls\(1\)](#), [stat\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

                『[Oracle Solaris の管理: 一般的なタスク](#)』

注意事項      A file with two or more links is counted only once. If, however, there are links between files in different directories where the directories are on separate branches of the file system hierarchy, du will count the excess files more than once.

                Files containing holes will result in an incorrect block count.

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | du – display the number of disk blocks used per directory or file                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式    | <code>/usr/ucb/du [-adkLr] [-o   -s] [filename]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 機能説明  | <p>The du utility gives the number of kilobytes contained in all files and, recursively, directories within each specified directory or file <i>filename</i>. If <i>filename</i> is missing, '.' (the current directory) is used.</p> <p>A file that has multiple links to it is only counted once.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| オプション | <p>The following options are supported:</p> <ul style="list-style-type: none"> <li>-a Generates an entry for each file.</li> <li>-d Does not cross file system boundaries. For example, <code>du -d /</code> reports usage only on the root partition.</li> <li>-k Writes the files sizes in units of 1024 bytes, rather than the default 512-byte units.</li> <li>-L Processes symbolic links by using the file or directory that the symbolic link references, rather than the link itself.</li> <li>-o Does not add child directories' usage to a parent's total. Without this option, the usage listed for a particular directory is the space taken by the files in that directory, as well as the files in all directories beneath it. This option does nothing if the -s option is used.</li> <li>-r Generates messages about directories that cannot be read, files that cannot be opened, and so forth, rather than being silent (the default).</li> <li>-s Only displays the grand total for each of the specified <i>filenames</i>.</li> </ul> |

Entries are generated only for each directory in the absence of options.

## 使用例

例 1 Showing usage of all subdirectories in a directory

This example uses du in a directory. The `pwd(1)` command was used to identify the directory, then du was used to show the usage of all the subdirectories in that directory. The grand total for the directory is the last entry in the display:

```
example% pwd
/usr/ralph/misc
example% du
5 ./jokes
33 ./squash
44 ./tech.papers/lpr.document
217 ./tech.papers/new.manager
401 ./tech.papers
144 ./memos
80 ./letters
388 ./window
93 ./messages
```

例 1 Showing usage of all subdirectories in a directory (続き)

```
15 ./useful.news
1211 .
```

環境変数

If any of the LC\_\* variables, that is, LC\_CTYPE, LC\_MESSAGES, LC\_TIME, LC\_COLLATE, LC\_NUMERIC, and LC\_MONETARY (see [environ\(5\)](#)), are not set in the environment, the operational behavior of du for each corresponding locale category is determined by the value of the LANG environment variable. If LC\_ALL is set, its contents are used to override both the LANG and the other LC\_\* variables. If none of the above variables is set in the environment, the "C" (U.S. style) locale determines how du behaves.

- LC\_CTYPE

Determines how du handles characters. When LC\_CTYPE is set to a valid value, du can display and handle text and filenames containing valid characters for that locale. du can display and handle Extended Unix Code (EUC) characters where any individual character can be 1, 2, or 3 bytes wide. du can also handle EUC characters of 1, 2, or more column widths. In the "C" locale, only characters from ISO 8859-1 are valid.
- LC\_MESSAGES

Determines how diagnostic and informative messages are presented. This includes the language and style of the messages, and the correct form of affirmative and negative responses. In the "C" locale, the messages are presented in the default form found in the program itself (in most cases, U.S. English).

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE   |
|----------------|-------------------|
| Availability   | compatibility/ucb |

関連項目

[pwd\(1\)](#), [df\(1M\)](#), [du\(1\)](#), [quot\(1M\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項

Filename arguments that are not directory names are ignored, unless you use -a.

If there are too many distinct linked files, du will count the excess files more than once.



|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | dump - オブジェクトファイルの選択部分のダンプ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 形式    | <pre>dump [-aCcghLorstV [-p]] [-T index [, indexn]] filename... dump [-afhorstL [-p] [v]] filename... dump [-hsr [-p] [-d number [, numbern]]] filename... dump [-hsrt [-p] [-n name]] filename...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明  | <p>dump ユーティリティーは、<i>filename</i> 引数で指定された各オブジェクトファイルの選択部分をダンプします。</p> <p>dump ユーティリティーはシェルスクリプトで使用するのに最適です。ただし、出力の読みやすさとしては、<a href="#">elfdump(1)</a> コマンドを推奨します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| オプション | <p>dump ユーティリティーは、オブジェクトファイルとオブジェクトファイルのアーカイブの両方を受け入れます。dump ユーティリティーは次のオプションに従い、<i>filename</i> 引数で指定された各ファイル进行处理します。</p> <ul style="list-style-type: none"> <li>-a           アーカイブの各メンバーのアーカイブヘッダーをダンプします。</li> <li>-c           文字列テーブルをダンプします。</li> <li>-C           復号化した C++ シンボルテーブル名をダンプします。</li> <li>-f           各ファイルヘッダーをダンプします。</li> <li>-g           アーカイブのシンボルテーブル内のグローバルシンボルをダンプします。</li> <li>-h           セクションヘッダーをダンプします。</li> <li>-L           可能であれば、動的リンク情報と静的共有ライブラリ情報をダンプします。</li> <li>-o           各プログラムの実行ヘッダーをダンプします。</li> <li>-r           再配置情報をダンプします。</li> <li>-s           セクションの内容を 16 進数でダンプします。</li> <li>-t           シンボルテーブルエントリをダンプします。</li> <li>-T <i>index</i> または<br/>-T <i>index1,index2</i>   <i>index</i> で定義された索引付きシンボルテーブルエントリ、または <i>index1</i> と <i>index2</i> で定義された範囲のエントリだけをダンプします。</li> <li>-V           バージョン情報を出力します。</li> </ul> <p>次の修飾子を前述のオプションと組み合わせて使用すると、オプションの機能が変わります。</p> |

- d *number***  
**-d *number1,number2***      *number* で示されたセクション番号、または *number1* から *number2* までの範囲のセクションをダンプします。修飾子 *d* はオプション *-h*、*-s*、および *-r* と共に使用できます。修飾子 *-d* をオプション *-h* または *-s* と共に使用すると、引数はセクション番号またはセクションの範囲として扱われます。修飾子 *-d* をオプション *-r* と共に使用すると、引数は、再配置が適用されるセクションの番号またはセクションの範囲として扱われます。たとえば、*.text* セクションに対応するすべての再配置エントリを出力するには、修飾子 *-d* の引数としてセクション番号を指定します。*.text* がファイル内のセクション番号 2 である場合、`dump -r -d 2` を実行すると、対応するすべてのエントリが出力されます。特定の再配置セクションを出力するには、*raw* データを出力する場合は `dump -s -n name` を使用し、解釈済みのデータを出力する場合は `dump -sv -n name` を使用します。
- n *name***      指定されたエントリにのみ対応する情報をダンプします。修飾子 *n* はオプション *-h*、*-s*、*-r*、および *-t* と共に使用できます。修飾子 *-n* をオプション *-h* または *-s* とともに使用すると、引数はセクション名として扱われます。修飾子 *-n* をオプション *-t* または *-r* とともに使用すると、引数はシンボル名として扱われます。たとえば、`dump -t -n .text` を使用すると、名前が *.text* のシンボルに対応するシンボルテーブルエントリがダンプされます。また、`dump -h -n .text` を使用すると、*.text* 用のセクションのセクションヘッダー情報がダンプされます。
- p**      ヘッダーの出力を抑制します。
- v**      情報を、数値ではなくシンボル表記でダンプします。修飾子 *v* は次のオプションと共に使用できます。
- a**      (日付、ユーザー ID、グループ ID)
  - f**      (クラス、データ、タイプ、マシン、バージョン、フラグ)
  - h**      (タイプ、フラグ)
  - L**      (値)
  - o**      (タイプ、フラグ)
  - r**      (名前、タイプ)
  - s**      (可能であれば、セクションの内容を解釈する)

-t (タイプ、バインド)

修飾子 `-v` をオプション `-s` とともに使用すると、解釈可能なすべてのセクション (文字列テーブルやシンボルテーブルなど) が解釈されます。たとえば、`dump -sv -n .symtab filename...` を実行すると、`dump -tv filename...` と同じ書式で出力されます。一方、`dump -s -n .symtab filename...` を実行すると、raw データが 16 進数で出力されます。修飾子の付かない `dump -sv filename...` を実行すると、ファイル内のすべてのセクションがダンプされます。このとき、解釈可能なセクションはすべて解釈され、解釈されないセクション (`.text` や `.data` など) は raw データとしてダンプされます。

`dump` ユーティリティーは情報を有意義な方法で出力します。つまり、出力する情報により、文字、16 進数、8 進数、または 10 進数を適切に使い分けます。

#### 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値                                |
|-------|------------------------------------|
| 使用条件  | developer/base-developer-utilities |

#### 関連項目

[elfdump\(1\)](#), [elffile\(1\)](#), [file\(1\)](#), [nm\(1\)](#), [ar.h\(3HEAD\)](#), [a.out\(4\)](#), [attributes\(5\)](#)

名前 dumpcs – 現在のロケールのコードセット一覧表を表示

形式 dumpcs [-0123vw]

機能説明 dumpcs は、ユーザーの現在のロケールに対応するコードセットの印字可能文字を 16 進数のコード値とともに表示します。ユーザーの表示端末はそのロケールの文字を表示できるものとします。オプションなしで起動した場合は、現在のロケールの印字可能文字をすべて表示します。

1 つ以上の数字オプションを指定した場合には、指定したコードセット番号に対応する現在のロケールの EUC コードセットが、コードセット順に表示されます。非印字文字は、その文字が属しているコードセットに固有な表示幅分の個数の ASCII スペース文字で置き換えられ、非印字文字であることが直前のアスタリスク (\*) で示されます。

- オプション
- 0 ASCII または EUC プライマリコードセットの文字を表示します。
  - 1 EUC 補助コードセット 1 の文字を表示します (現在のロケールで使用されている場合)。
  - 2 EUC 補助コードセット 2 の文字を表示します (現在のロケールで使用されている場合)。
  - 3 EUC 補助コードセット 3 の文字を表示します (現在のロケールで使用されている場合)。
  - v 「冗長」。非印字文字をリストしてコード表を表示します。このオプションがなく、非印字文字が 1 行以上連続した場合、最初の行の最初のカラムにアスタリスクがあるだけの 1 つの行で置き換えられます。
  - w コード値を、対応するワイド文字の値で置き換えます。

環境変数 環境変数 LC\_CTYPE と LANG は、dumpcs の文字分類を制御します。dumpcs を起動すると、これらの環境変数は、LC\_CTYPE、LANG の順にチェックされます。たとえば、LANG を新しく設定しても LC\_CTYPE の現在有効な文字分類にオーバーライドすることはありません。有効な値がない場合、文字分類はデフォルトで POSIX.1 の C ロケールに設定されます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [localedef\(1\)](#), [attributes\(5\)](#)

注意事項 dumpcs は EUC ロケールでのみ使用できます。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | echo – 引数の出力                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 形式    | /usr/bin/echo [ <i>string</i> ]...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 機能説明  | <p>echo ユーティリティーは、空白文字で区切られ、復帰改行 (NEWLINE) で終わる引数を標準出力に出力します。引数が指定されていない場合は、NEWLINE 文字のみを書き出します。</p> <p>echo は、コマンドファイル内で診断メッセージを生成したり、既知のデータをパイプに送ったり、環境変数の内容を表示したりする際に便利です。</p> <p>C シェル、Korn シェル、Bourne シェル には、echo 組み込みコマンドがあります。デフォルトでは echo 組み込みコマンドは、ユーザーがフルパス名なしで echo を呼び出した場合に実行されます。 <a href="#">shell_builtins(1)</a> を参照してください。sh の echo、ksh88 の echo、ksh の echo、および /usr/bin/echo はバックスラッシュの付いたエスケープ文字を認識します。ただし、sh の echo は、警告文字として \a を認識しません。さらに、ksh88 および ksh の echo には -n オプションは付きません。一方、csh の echo と /usr/ucb/echo には -n オプションが付きますが、バックスラッシュの付いたエスケープ文字は認識しません。sh と ksh88 は、/usr/ucb/echo が最初に PATH で見つかるかどうかを判別し、見つかった場合は、組み込み echo の動作を /usr/ucb/echo に合わせます。</p> |
| オペランド | <p>次のオペランドを指定できます。</p> <p><i>string</i>    標準出力に書き込まれる文字列。任意のオペランドが“-n”である場合、オプションではなく文字列として扱います。以下の文字の並びは、引数の中で特別に解釈されます。</p> <ul style="list-style-type: none"> <li>\a    警告文字</li> <li>\b    バックスペース</li> <li>\c    復帰改行なしで行を出力します。引数内で \c に続く文字はすべて無視されます。</li> <li>\f    用紙送り (form-feed)</li> <li>\n    復帰改行</li> <li>\r    キャリッジリターン</li> <li>\t    タブ</li> <li>\v    垂直タブ</li> <li>\\    バックスラッシュ</li> <li>\0<i>n</i>    <i>n</i> は、ASCII コードを 1、2、または 3 桁の 8 進数で表した 8 ビット文字です。</li> </ul>                                                                                                                                                                                       |

## 使用法

移植性の必要なアプリケーションには、`-n` (第1の引数として) またはエスケープシーケンスを使用しないでください。

`printf(1)` ユーティリティーは、以下のように `echo` ユーティリティーの従来の動作をすべてエミュレートするために使用できます。

- Solaris 2.6 オペレーティング環境またはその互換バージョンの `/usr/bin/echo` は次と同等です。

```
printf "%b\n" "$*"

/usr/ucb/echo は次と同等です。

if ["X$1" = "X-n"]

then

 shift

 printf "%s" "$*"

else

 printf "%s\n" "$*"

fi
```

新しいアプリケーションでは `echo` の代わりに `printf` を使用することをお勧めします。

## 使用例

例1 現在のディレクトリがルートからどれくらい離れているかを調べる

`echo` を使用して、現在のディレクトリがルートディレクトリ (`/`) から見ていくつ目のサブディレクトリになるかを、次のようにして判定することができます。

- 現在の作業中のディレクトリのフルパス名を表示する
- パスに組み込まれたスラッシュ文字を、スペース文字に変換するために `tr` を通して出力をパイプする
- ユーザーのパスの名前をカウントするために `wc -w` を通して出力をパイプする

```
example% /usr/bin/echo $PWD | tr '/' ' ' | wc -w
```

それぞれの機能については、`tr(1)` と `wc(1)` を参照してください。

以下に、復帰改行 (NEWLINE) なしで文字列を表示するための別の方法を示します。

```
例2 /usr/bin/echo
example% /usr/bin/echo "$USER's current directory is $PWD\c"

例3 sh/ksh88 シェル
example$ echo "$USER's current directory is $PWD\c"

例4 csh シェル
example% echo -n "$USER's current directory is $PWD"

例5 /usr/ucb/echo
example% /usr/ucb/echo -n "$USER's current directory is $PWD"
```

- 環境変数
- uname の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。
- 終了ステータス
- 次のエラー値が返されます。  
0     正常終了。  
>0    エラーが発生した。
- 属性
- 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

- 関連項目
- [ksh\(1\)](#), [printf\(1\)](#), [shell\\_builtins\(1\)](#), [tr\(1\)](#), [wc\(1\)](#), [echo\(1B\)](#), [ascii\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)
- 注意事項
- エスケープ規則 `\0n` を使用して 8 ビット文字を表す際、*n* の前に必ずゼロ (0) を付けなければなりません。  
  
たとえば、`echo 'WARNING:\07'` と入力した場合、`WARNING:` が出力され、端末の“ベル”が鳴ります。“07”の前に“\”を付けるときには、これを保護する単一 (または二重) 引用符 (または 2 つのバックスラッシュ) を使用する必要があります。  
  
`\0` の後には、8 進の出力文字を形成する最大 3 桁の文字が使用されます。`\0n` の後に、この 8 進表記に含まれない数値をさらに表示したい場合は、*n* には全 3 桁を使用しなければなりません。たとえば、“ESC 7”と表示したい場合は、`\0` の後に、2 桁の“33”だけでなく、3 桁の“033”を使用しなければなりません。

|          |            |                        |         |
|----------|------------|------------------------|---------|
| 2 digits | Incorrect: | echo "\0337"   od -xc  |         |
|          | produces:  | df0a                   | (hex)   |
| 3 digits |            | 337                    | (ascii) |
|          | Correct:   | echo "\00337"   od -xc |         |
|          | produces:  | 1b37 0a00              | (hex)   |
|          |            | 033 7                  | (ascii) |

各文字の 8 進表記については、[ascii\(5\)](#) を参照してください。



| 名前             | echo – echo arguments to standard output                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                |                 |              |                   |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|-----------------|--------------|-------------------|
| 形式             | <code>/usr/ucb/echo [-n] [argument]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |                |                 |              |                   |
| 機能説明           | <p>echo writes its arguments, separated by BLANKs and terminated by a NEWLINE, to the standard output.</p> <p>echo is useful for producing diagnostics in command files and for sending known data into a pipe, and for displaying the contents of environment variables.</p> <p>For example, you can use echo to determine how many subdirectories below the root directory (/) is your current directory, as follows:</p> <ul style="list-style-type: none"> <li>■ echo your current-working-directory's full pathname</li> <li>■ pipe the output through <code>tr</code> to translate the path's embedded slash-characters into space-characters</li> <li>■ pipe that output through <code>wc -w</code> for a count of the names in your path.</li> </ul> <pre>example% /usr/bin/echo "echo \$PWD   tr '/' ' '   wc -w"</pre> <p>See <a href="#">tr(1)</a> and <a href="#">wc(1)</a> for their functionality.</p> <p>The shells <a href="#">csh(1)</a>, <a href="#">ksh(1)</a>, and <a href="#">sh(1)</a>, each have an echo built-in command, which, by default, will have precedence, and will be invoked if the user calls echo without a full pathname. <code>/usr/ucb/echo</code> and <code>csh's echo()</code> have an <code>-n</code> option, but do not understand back-slashed escape characters. <code>sh's echo()</code>, <code>ksh's echo()</code>, and <code>/usr/bin/echo</code>, on the other hand, understand the black-slashed escape characters, and <code>ksh's echo()</code> also understands <code>\a</code> as the audible bell character; however, these commands do not have an <code>-n</code> option.</p> |                |                 |              |                   |
| オプション          | <code>-n</code> Do not add the NEWLINE to the output.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                |                 |              |                   |
| 属性             | See <a href="#">attributes(5)</a> for descriptions of the following attributes:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                |                 |              |                   |
|                | <table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>compatibility/ucb</td></tr> </tbody> </table>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | ATTRIBUTE TYPE | ATTRIBUTE VALUE | Availability | compatibility/ucb |
| ATTRIBUTE TYPE | ATTRIBUTE VALUE                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                |                 |              |                   |
| Availability   | compatibility/ucb                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                |                 |              |                   |
| 関連項目           | <a href="#">csh(1)</a> , <a href="#">echo(1)</a> , <a href="#">ksh(1)</a> , <a href="#">sh(1)</a> , <a href="#">tr(1)</a> , <a href="#">wc(1)</a> , <a href="#">attributes(5)</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                |                 |              |                   |
| 注意事項           | The <code>-n</code> option is a transition aid for BSD applications, and may not be supported in future releases.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                |                 |              |                   |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | ed, red – テキストエディタ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 形式   | <pre>/usr/bin/ed [-s   -] [-p <i>string</i>] [<i>file</i>]<br/><br/>/usr/xpg4/bin/ed [-s   -] [-p <i>string</i>] [<i>file</i>]<br/><br/>/usr/xpg6/bin/ed [-s   -] [-p <i>string</i>] [<i>file</i>]<br/><br/>/usr/bin/red [-s   -] [-p <i>string</i>] [<i>file</i>]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明 | <p>ed ユーティリティーは標準のテキストエディタです。<i>file</i> が指定されていると ed は e コマンド (下記参照) を指定されたファイルについてシミュレートします。その結果、このファイルは ed のバッファに読み込まれ、編集できるようになります。</p> <p>ed ユーティリティーは、編集されるファイルのコピーに対して操作を行います。つまり、コピーに対して行われた変更は w (write) コマンドが実行されるまで、そのファイルに対して有効になりません。編集されるテキストのコピーはバッファという一時ファイルに存在します。バッファは1つしかありません。</p> <p>red ユーティリティーは ed の制限付きバージョンです。red は現在のディレクトリ内のファイルしか編集できず、<code>!shell command</code> によるシェルコマンドの実行もできません。これらの制限を無視しようとするエラーメッセージ (<i>restricted shell</i>) が表示されます。</p> <p>ed と red は、両方とも <a href="#">fspec(4)</a> フォーマット機能をサポートしています。デフォルトの端末モードは <code>stty -tabs</code> または <code>stty tab3</code> で、タブ位置は 8 カラムごとに設定されます (<a href="#">stty(1)</a> を参照)。ただし <i>file</i> の先頭行にフォーマット指定が記述されていれば、その指定がデフォルトモードをオーバーライドします。たとえば、タブ位置を 5、10、15 に設定し、最大行長を 72 に設定するには、<i>file</i> の最初の行に次のように指定します。</p> <pre>&lt;:t5,10,15 s72:&gt;</pre> <p>ed コマンドは単純で、その構造は規則的です。最初に 0 から 2 個までのアドレス、次に 1 文字のコマンド、最後に(あれば) コマンドのパラメータが続きます。アドレスはバッファ内の 1 行または複数行を指定するものです。アドレスが必要なコマンドはすべてデフォルトのアドレスを持っているので、アドレスを省略することがよくあります。</p> <p>一般に、1 行に 1 つのコマンドだけ指定します。コマンドの中には、テキストを入力するものもあります。この時テキストは、バッファの中の適切な場所に格納されます。ed がテキストを受け入れていることを「入力モード」とあるといいます。このモードではコマンドは認識されません。すべての入力そのまま受け付けられるだけです。入力モードから抜けるには、行の先頭でピリオド (.) だけを入力して、キャリッジリターンを押します。</p> |

`/usr/bin/ed` ed が引数のあるコマンドを実行する場合、デフォルトのシェル `/usr/bin/sh` が使われます ([sh\(1\)](#) 参照)。

`/usr/xpg4/bin/ed` およ `び /usr/xpg6/bin/ed` ed が引数のあるコマンドを実行する場合、`/usr/xpg4/bin/sh` が使われます ([ksh88\(1\)](#) を参照)。

正規表現 ed ユーティリティーでは「正規表現」規則がサポートされていますが、ある程度制限されています。正規表現は、アドレスの中では行を指定するために、また、いくつかのコマンド(たとえば、`s`)の中では行のうちの置換される部分を指定するために用いられます。ed におけるアドレス指定方法を理解するには、常に「現在行」が存在することを認識する必要があります。一般に、現在行はコマンドによって影響を受けた最後の行です。現在行が受ける影響については、各コマンドの説明の箇所ですべて述べます。

国際化された標準の正規表現は、システムに与えられたすべてのロケールで使用されます。詳細については [regex\(5\)](#) を参照してください。

ed コマンド コマンドには 0、1、または 2 個のアドレスを必要とします。アドレスを必要としないコマンドにアドレスを指定するとエラーになります。アドレスが必要なコマンドに必要な数のアドレスを指定しないと、デフォルトのアドレスが採用されます。必要以上にアドレスが指定されると、最後のアドレスの方から使用されます。

通常、アドレスはコンマ( , )によってお互いに区切られます。セミコロン( ; )によっても区切られます。後者の場合、最初のアドレスが計算され、現在行( . )がその値に設定されます。その後、2 番目のアドレスが計算されます。この機能は、順方向および逆方向検索の開始行を決定するのに使用できます(前述の規則 5 および 6 を参照)。2 つのアドレスが指定されている場合には、2 番目のアドレスはバッファ内の特定の行、つまり最初のアドレスに対応する行に続く行に対応している必要があります。

`/usr/xpg6/gbin/ed` の場合、区切り記号のコンマまたはセミコロンの前後では、アドレスを省略できます。アドレスを省略した場合は、アドレスは次のように解釈されます。

| 指定    | 結果        |
|-------|-----------|
| ,     | 1,\$      |
| ,addr | 1,addr    |
| addr, | addr,addr |
| ;     | 1;\$      |
| ;addr | 1;addr    |
| addr; | addr;addr |

アドレス、アドレス区切り記号、またはアドレスオフセット間にある空白は無視されます。

以下に示す ed コマンドのリストでは、コマンドの前の括弧はアドレスの一部ではなく、デフォルトのアドレスを示します。

各アドレス部分の先頭には任意の2個の空白文字を付加できます。またコマンド文字の先頭にも任意の数の空白文字を付加できます。接尾文字(l、n、またはp)を指定するのであれば、コマンドの直後に記述しなければなりません。

e、E、f、r、wの各コマンドには、省略可能な *file* パラメータがあります。これを指定する場合には、コマンド文字との間に最低1個の空白文字を置くことが必要です。

バッファ全体を書き換えた最後のwコマンド実行後にバッファの内容が変更されているとき、eまたはqコマンドによりエディタバッファを破壊しようとする、edは警告を発します。具体的には以下の文字列を標準出力に書き出します。

```
"?\n"
```

なおHコマンドにより「ヘルプモード」が起動されていれば、状況を説明するメッセージが続いて出力されます。この警告出力後も、edはコマンドモードのままで、現在の行番号は変わりません。ここで続けてeまたはqコマンドを再度入力すれば、そのコマンドが実行されます。

標準入力から次のコマンドを読み込もうとしてファイルの終わりを検出した場合、edユーティリティはqコマンドが入力された場合と同じように動作します。

一般に、1行に2つ以上のコマンドを指定するとエラーです。しかし、すべてのコマンド(e、f、r、wを除く)は、l、n、またはpコマンド(それぞれ、現在行をリストする、番号付けする、出力する)を接尾辞として付けることができます(l、n、およびpコマンドを参照)。

```
(.)a
<text>
```

append コマンドは、0行以上のテキスト行を受け入れ、それをバッファ内のアドレスで指定された行の直後に追加します。現在行(.)は、最後に挿入された行に設定されます。挿入行がない場合は、アドレスで指定された行に設定されます。このコマンドではアドレス0は有効です。この場合、バッファの先頭にテキストを追加します。端末から入力できる最大文字数は1行あたり256文字です(復帰改行文字も含む)。

```
(...)c
<text>
```

change コマンドは、バッファからアドレスで指定された行を削除し、0行以上のテキスト行をバッファ内に受け入れ、削除された行と置換します。現在行(.)は最終入力行に設定されます。最終入力行がない場合

は、削除されなかった最初の行に設定されます。バッファの最終行を削除した場合、現在の行番号は新たに最終行となった行のアドレスに設定されます。削除によりバッファ内に行が残っていなければ、現在の行番号はゼロに設定されます。

`/usr/xpg4/bin/ed` このコマンドでは、アドレス 0 は有効ではありません。

`/usr/xpg6/bin/ed` このコマンドでは、アドレス 0 は有効です。この場合、アドレス 1 が指定されたと解釈されます。

**(.,.)d** delete コマンドは、バッファからアドレスで指定された行を削除します。削除された最後の行の直後の行が現在行に設定されます。バッファの最終行を削除した場合、新たに最終行となった行が現在行になります。削除によりバッファ内に行が残っていなければ、現在の行番号はゼロに設定されます。

**e file** edit コマンドは、バッファの全内容を削除して *file* の内容をバッファ内に読み込みます。現在行(.)は、バッファの最終行に設定されます。*file* が指定されないと、(もし、あれば) 現在記憶されているファイル名を使用します(**f** コマンド参照)。**-s** オプションが指定された場合を除き、読み込んだバイト数が以下の形式で標準出力に書き込まれます。

`"%d\n"` <読み込んだバイト数>

*file* は、後で使用する **e**、**E**、**r**、および **w** コマンドのデフォルトのファイル名として記憶されます。*file* の代わりに **!** を指定すると、**!**以降の文字列はシェル (**sh(1)**) コマンドと見なされ、その出力が読み込まれます。このようなシェルコマンドは現在のファイル名として記憶されません。後述の「診断」も参照してください。**e** コマンドが正常に終了すると、すべてのマークは捨てられます。バッファ全体が最後に書き換えた後でその内容が変更されているとき、前述したように警告が発せられます。

**E file** Edit コマンドは、基本的に、**e** コマンドと同じです。ただし、最後の **w** コマンドを実行してからバッファの内容が変更されたか否かをチェックしません。

**f file** *file* が指定されると、**f** コマンドは、現在記憶されているパス名を *file* に変更します。次に、パス名を変更したか否かに関わらず、現在記憶されているパス名 (変更した場合は新しいパス名) を以下の形式で標準出力に書き出します。

`"%s\n"` パス名

現在の行番号は変わりません。

**(1,\$)g/RE/command list** global コマンドは、まず *RE* で指定された正規表現と一致する各行をマークします。そして、該当するすべての行

に対して、まず現在行(.)をその行に設定してから、*command list*で指定したコマンドリストを実行します。g コマンドが終了したとき、現在の行番号の値は、コマンドリスト中の最後のコマンドが指定した値となっています。一致する行が見つからなかった場合には、現在の行番号の値は変わりません。単一コマンドまたはコマンドリストの最初のコマンドは global コマンドと同一行に現われなければなりません。コマンドが複数行にまたがる場合、最終行以外の各行はバックスラッシュ(\)で終わらなければなりません。a、i、およびc コマンドおよび関連する入力に許可されます。入力モードを終了する. が *command list* の最後の行の場合、省略できます。空の *command list* は p コマンドと同等です。g、G、v、V、および! コマンドは、*command list* 内に書くことができません。「注意事項」および「ファイル」の前の最後のパラグラフも参照してください。RE を区切る文字として、スラッシュの代わりに、スペースと復帰改行以外の任意の文字を使うことができます。また区切り文字を RE 中で実際の文字として使いたければ、その前にバックスラッシュを付加してください。

(1,\$)G/RE/

対話型 Global コマンドでは、最初に、RE で指定された正規表現と一致する各行をマークします。そして、該当する各行について、その行が標準出力に書き込まれ、現在行(.)がその行に変更され、この時点で、1つのコマンド(a、c、i、g、G、v、およびV コマンドをのぞく)が入力でき、入力されたコマンドは実行されます。コマンドが実行されると、次のマークされた行が出力され、以下同じ処理を繰り返します。復帰改行は NULL コマンドとして動作します。&は、現在起動中のG内で、最後に実行された NULL でないコマンドを再実行します。注意:G コマンド実行中のコマンド入力はバッファ内のどの行でもアドレス指定したり、影響を与えたりできません。現在の行番号の最終的な値は、最後に正常終了したコマンドが設定した値です(なおGで入力したコマンドが失敗した場合、またはNULL コマンドが入力された場合は、G自身が「最後に正常終了したコマンド」となります)。一致する行が見つからなかった場合には、現在の行番号の値は変わりません。G コマンドは SIGINT シグナルによって終了させることができます。G コマンドは、割り込みシグナル(ASCII DEL または BREAK)で終了できます。RE を区切る文字として、スラッシュの代わりに、スペースと復帰改行以外の任意の文字を使うこと

ができます。また区切り文字を *RE* 中で実際の文字として使いたければ、その前にバックスラッシュを付加してください。

**h** help コマンドは、最後の ? 診断の理由を説明する短いエラーメッセージを出力します。現在の行番号は変わりません。

**H** Help コマンドは、ed を以後発生するすべての ? 診断のエラーメッセージを出力するモードにします。もしあれば前の ? も説明します。H コマンドは、このモードを交互にオンおよびオフにします。最初はオフです。現在の行番号は変わりません。

**(.,.)i**  
**<text>**  
**.**

insert コマンドは、0 またはそれ以上の行のテキストを受け入れ、バッファ内のアドレス指定された行の前に挿入します。現在行 (.) は、最後に挿入された行に設定されます。挿入行がない場合は、アドレスで指定された行に設定されます。i コマンドと a コマンドは、入力テキストの挿入位置だけ異なります。端末から入力できる最大文字数は 1 行あたり 256 文字です (復帰改行文字も含む)。

/usr/xpg4/bin/ed このコマンドでは、アドレス 0 は有効ではありません。

/usr/xpg6/bin/ed このコマンドでは、アドレス 0 は有効です。この場合、アドレス 1 が指定されたと解釈されます。

**(.,.+1)j** join コマンドは、適当な復帰改行文字を削除することによって連続する行を結合します。1 つのアドレスしか指定されないと j コマンドは何もしません。行の結合が行われると、現在の行番号は結合された側の行のアドレスに設定されます。そうでない場合、現在の行番号は変わりません。

**(.)kx** mark コマンドは、x という名前でアドレス行をマークします。x は ASCII の小文字 (a-z) でなければなりません。x を指定すると、このマークされた行を指すようになります。現在行 (.) は変更されません。

**(.,.)l** list コマンドは、アドレスされた行をすべての情報が見える形で標準出力へ書き出します。\\、\ a、\ b、\ f、\ r、\ t、\ v はそれぞれ対応するエスケープシーケ



ンスとして書き込まれます。テーブル中にある `\n` は適用外です。テーブル中にある非印字文字に関しては、各バイトごとに(最上位バイトから)3桁の8進数で、前にバックスラッシュが付加された形式で出力されます。

長い行は折り返され、折り返しの発生地点はバックスラッシュと復帰改行文字の書き込みによって示されます。折り返し地点の長さは不特定ですが、出力装置に適した値になっています。各行の終わりは `$` でマークされます。`/usr/xpg6/bin/ed` コマンドを使用する場合、各行の終わりは折り返しを示す `$` でマークされます。また、テキスト内の `$` 文字の前にはバックスラッシュが付加されます。`!` コマンドは、`e`、`E`、`f`、`q`、`Q`、`r`、`w`、`!` 以外のすべてのコマンドの後に付けられます。現在の行番号は、最後に出力された行のアドレスに設定されます。

`(... )ma`

`move` コマンドは、アドレス行を `a` で示される行の後に移動します。アドレス `0` を `a` に書いても有効で、その場合アドレス行をファイルの先頭に移動します。アドレス `a` が移動する行の範囲内にあるとエラーになります。現在行 `(.)` は移動した最後の行に設定されます。

`(... )n`

`number` コマンドは、行番号とタブの後に、アドレス行を出力します。現在行 `(.)` は出力した最後の行に設定されます。`n` コマンドは `e`、`E`、`f`、`q`、`Q`、`r`、`w`、または `!` 以外のすべてのコマンドの後に続きます。

`(... )p`

`print` コマンドは、アドレス行を標準出力に出力します。現在行 `(.)` は出力した最後の行に設定されます。`p` コマンドは、`e`、`E`、`f`、`q`、`Q`、`r`、`w`、または `!` 以外のすべてのコマンドの後に付けられます。たとえば、`dp` というコマンドは、現在行を削除して新しい現在行を出力します。

`P`

`P` コマンドは、後続のすべてのコマンドにアスタリスク `(*)` (`-p` 指定時は文字列) によるプロンプトをつけて入力を促します。`P` コマンドは、このモードを交互にオンまたはオフします。初期値は、`-p` オプション指定時はオン、省略時はオフです。現在行は変わりません。

`q`

`quit` コマンドは、`ed` を終了します。バッファ全体が書き換えられた後でバッファの内容が変更されていると、警告が発せられます。後述の「診断」を参照してください。

`Q`

エディタは、最後の `w` コマンドの後、バッファが変更されたか否かをチェックしないで終了します。



`($)r file`

`read` コマンドは、*file* の内容をバッファ内に読み込みます。*file* が指定されないと、(もしあれば) 現在記憶されているファイル名を使用します(`e` および `f` コマンド参照)。*file* が `ed` 起動後参照された最初のファイル名でないかぎり、現在記憶されているファイル名は変更されません。`r` コマンドにはアドレス `0` は有効です。この場合、バッファの先頭にファイルを読み込みます。読み込みに成功すると、`-s` オプションが省略されていれば、読み込まれた文字数が以下の形式で標準出力に書き込まれます。

`%d\n, <number of bytes read>`

現在行 (`.`) は読み込まれた最後の行に設定されます。*file* の代わりに `!` を指定すると、`!`以降の文字列はシェルコマンド (`sh(1)` 参照) と見なされ、その出力が読み込まれます。たとえば、`$r !ls` は編集集中のファイルの最後に現在のディレクトリを追加します。このようなシェルコマンドは現在のファイル名として記憶されません。

```
(. . .)s/RE/replacement/
(. . .)s/RE/replacement/count,count=[1-2047]
(. . .)s/RE /replacement/g
(. . .)s/RE/replacement/\
(. . .)s/RE/replacement/n
(. . .)s/RE/replacement /p
```

`substitute` コマンドは、各アドレス行について、*RE* で示された正規表現を検索します。これらの置換コマンドは、任意の数だけ指定できます。一致が発生した各行に対して、グローバル置換指示子 `g` がコマンドの後にあれば、すべての (重ならない) 一致した文字列を *replacement* に置換します。グローバル指示子がなければ、一致した文字列の最初のものだけを置換します。数字 *count* がコマンドの後にあれば、各アドレス行内で一致した文字列のうち *count* 番目のものだけを置換します。すべてのアドレス行について置換が失敗するとエラーになります。正規表現 *RE* と *replacement* を区切るには、スラッシュ (`/`) の代わりにスペース文字と復帰改行以外のすべての文字が使用できます。現

在行(.)は置換が発生した最後の行に設定されます。また区切り文字を *RE* 中で実際の文字として使いたければ、その前にバックスラッシュを付加してください。「ファイル」の前の最後のパラグラフも参照してください。

*replacement* 内のアンパサンド(&)は、現在行上で正規表現 *RE* と一致した文字列に置き換えられます。この場合の & の特別な意味は、\ を前につけることによって抑制できます。\\*n* (*n* は数字) は、指定された正規表現 *RE* の \ ( と \ ) で囲まれた *n* 番目のサブ正規表現と一致するテキストに置換されます。ネストされた括弧付きサブ正規表現が存在する場合、*n* は左から数えた \ ( の発生回数によって決まります。文字 % が *replacement* 内の唯一の文字であるとき、最後の置換コマンドで使った *replacement* を現在の置換コマンドの *replacement* として使用します。ただしそれ以前に置換コマンドがなかった場合、このような % の使い方はエラーとなります。% は、複数の文字の置換文字内にあるとき、または \ が前に付くときには、その特別な意味を失います。*replacement* を先頭から終端まで走査する際にバックスラッシュ (\) が検出されると、後続の文字は特殊な意味を持っていたとしてもその意味を失います。&、\、% および数字以外の文字に対し、どのような特殊な意味が与えられているかは不特定です。

1 行を分割するには、復帰改行文字で置換します。*replacement* 内の復帰改行文字は、\ を前に付けてエスケープしなければなりません。このような置換は、*g* または *v* コマンドリストの一部としては実行できません。

ん現在の行番号は、置換が行われた最後の行のアドレスに設定されます。置換がまったく行われないと、現在の行番号は変わりません。行が分割されると、新たな現在の行番号を決定するため、分割で発生した両方の行で置換が行われたものと見なされます。置換文字列が、置換後の文字列と同一である場合も、置換が行われたこととなります。

置換コマンドは以下に示す指示子をサポートします。

- count*    各アドレス行で見つかった *RE* のうち、*count* 番目のものだけを置換します。*count* は 1 から 2047 までの値でなければなりません。
- g*        最初のものだけでなく、すべての重なっていない *RE* を一括して置換します。*g* と *count* の両方を指定した場合、結果は不確定です。
- l*        置換を行なった最後の行の内容を標準出力に書き出します。出力形式は *l* コマンドのものと同じです。
- n*        置換を行なった最後の行の内容を標準出力に書き出します。出力形式は *n* コマンドのものと同じです。
- p*        置換を行なった最後の行の内容を標準出力に書き出します。出力形式は *p* コマンドのものと同じです。

(...)*ta*

*t* コマンドは、*m* コマンドと同じように動作します。ただし、アドレス行のコピーがアドレス *a* (0 でもよい

)の後に置かれます。現在行(.)はコピーされた最後の行に設定されます。

u

undo コマンドは、バッファの内容を変更した最後のコマンドの実行結果を無効にします。無効にできるコマンドは、最後に実行した a、c、d、g、i、j、m、r、s、t、u、v、G、または V コマンドです。グローバルコマンドの g、G、v、または V でバッファを変更していた場合、一括してその変更を無効にします。グローバルコマンドで変更が行われていない(たとえば g/RE/p) 場合、u コマンドは何も意味を持ちません。現在の行番号は、無効にしたコマンドの開始直前に設定されていた値に戻ります。

(1,\$)v/RE/command list

v コマンドは、基本的に、グローバルコマンド g と同じです。ただし、最初の段階でマークされる行は RE で示す正規表現に一致しないものです。

(1,\$)V/RE/

V コマンドは、基本的に、対話型グローバルコマンド G と同じです。ただし、最初にマークされる行は RE で示す正規表現に一致しないものです。

(1,\$)w file

write コマンドはアドレス行を *file* に書き込みます。*file* が存在しない場合は、ファイル生成マスクが他のモードで指定されていないかぎり、モード 666 (すべてのユーザーが読み込み、書き込み可能) で作成します。[sh\(1\)](#) 上での特殊コマンド `umask` の説明を参照してください。*file* が ed 起動後参照された最初のファイル名でないかぎり、現在記憶されているファイル名は変更されません。ファイル名が指定されないと、現在記憶されているファイ

ル名 (ある場合) を使用します (e および f コマンドを参照)。現在行 (.) は変更されません。コマンドが正常終了すると、-s オプションが省略されていれば、書き込まれた文字数が以下の形式で出力されます。

```
"%d\n", <number of bytes written>
```

*file* の代わりに ! を指定した場合、! 以降のテキストは、アドレス行が標準入力であるシェル ([sh\(1\)](#) 参照) コマンドと見なされ、その出力が読み込まれます。このようなシェルコマンドは、現在のパス名として記憶されません。このような ! を伴った w コマンドは、「バッファ全体を書き換えた最後の w コマンド」と見なされます。

(1, \$)W *file*

W コマンドは基本的に上述の write コマンドと同じです。ただし、アドレス行を *file* (存在する場合) の最後に追加します。*file* が存在しない場合、上述の w コマンドで述べたようにファイルを作成します。

(\$)=

アドレス行の行番号が、次に示す形式で標準出力に表示されます。

```
"%d\n" <line number>
```

このコマンドによって、現在の行番号は変更されません。

! *shell command*

! 以降のテキストを UNIX システムシェル ([sh\(1\)](#) 参照) に送信し、コマンドとして解釈します。コマンドテキストにエスケープされていない % 文字を指定すると、記憶されているファイル名に置換されます。! がシェルコマンドの最初の文字として現われる場合、それは前のシェルコマンドのテキストで置換されます。つまり、!! は最後のシェルコマンドを繰り返します。% または ! による置換が実行されると、変更された行の内容が *command* の実行前に標準出力に書き込まれます。! コマンドは以下のメッセージを書き出します。

```
"!\n"
```

-s オプションが省略されていれば、終了時に標準出力に書き出します。現在の行番号は変わりません。

(.+1)<new-line> 行にアドレスだけを指定すると、そのアドレス行を出力します。復帰改行文字だけの場合、.+1p と同じです。つまり、バッファ内を進むのに使用します。現在の行番号は、書き出した行のアドレスに設定されます。

割り込みシグナル (ASCII DEL または BREAK) が送信されると、ed は "?\n" を出力して、そのコマンドレベルに戻ります。

ed ユーティリティはすべてのシグナルに対して標準的な動作を行います。ただし次の2つのシグナルは例外です。

**SIGINT** ed ユーティリティは現在の動作を中断し、文字列 "?\n" を標準出力に書き出し、コマンドモードに戻ります。

**SIGHUP** バッファが空でなく、最後の書き込み処理以降に変更されている場合、ed ユーティリティはファイル中にバッファのコピーを生成しようとし、その対象ファイルとして、まず現在のディレクトリ中の ed.hup というファイルが選ばれます。それが失敗すると、環境変数 HOME が示すディレクトリ中の ed.hup というファイルが選ばれます。いずれの場合も ed は、コマンドモードに戻らないで終了します。

いくつかのサイズ制限があります。1行は512文字以下、グローバル・コマンドリストは256文字以下、ファイルのパス名は255文字以下です(スラッシュを含む)。行数の制限はユーザーのメモリ容量によって異なります。1行には1ワード必要です。

ファイルを読むとき、ed は ASCII と NUL 文字を破棄します。

ファイルが復帰改行文字で終了していないとき、ed はそれを追加して、その旨を説明するメッセージを表示します。

正規表現 RE または置換文字列の終端区切り文字 (たとえば、/) が復帰改行文字の直前の文字のとき、区切り文字は省略できます。いずれの場合もアドレスされた行が出力されます。次の各組のコマンドは同等です。

s/s1/s2      s/s1/s2/p

g/s1          g/s1/p

?s1          ?s1?

不正なコマンドが投入されると、ed は以下の文字列を書き出します。

"?\n"

このとき H コマンドにより「ヘルプモード」が有効になっていれば、状況を説明するメッセージが付加されます。前述の文字列を標準出力に出力後、ed はコマンドモードを継続します。現在の行番号は変わりません。

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| オプション   | <p><code>-p string</code> ユーザーがプロンプト文字列を指定するのを許可します。デフォルトではプロンプト文字列はありません。</p> <p><code>-s   -;</code> <code>e</code>、<code>r</code>、および <code>w</code> コマンドによる文字カウント、<code>e</code> および <code>q</code> コマンドからの診断、および <code>!shell command</code> の後の <code>!</code> プロンプトを出力しません。</p>                                                                                                                                                  |
| オペランド   | <p>次のオペランドを指定できます。</p> <p><code>file</code> この <code>file</code> 引数を指定すると、<code>ed</code> は、標準入力からコマンドを読み込む前に、パス名 <code>file</code> で示されるファイルに <code>e</code> コマンドを適用したかのように動作します。</p>                                                                                                                                                                                                                                                  |
| 使用法     | ファイルが 2G バイト (2 <sup>31</sup> バイト) 以上ある場合の <code>ed</code> と <code>red</code> の動作については、 <a href="#">largefile(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                         |
| 環境変数    | <code>ed</code> の実行に影響を与える次の環境変数についての詳細は、 <a href="#">environ(5)</a> を参照してください。HOME、LANG、LC_ALL、LC_CTYPE、LC_COLLATE、LC_MESSAGES、および NLSPATH。                                                                                                                                                                                                                                                                                            |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p><code>0</code> ファイルにもコマンドにもエラーはなく、正常終了しました。</p> <p><code>&gt;0</code> エラーが発生した。</p>                                                                                                                                                                                                                                                                                                                          |
| ファイル    | <p><code>\$TMPDIR</code> この環境変数が <code>NULL</code> でない場合、その値は一時作業ファイルのディレクトリ名として <code>/var/tmp</code> の代わりに使用されます。</p> <p><code>/var/tmp</code> <code>/var/tmp</code> が存在する場合、一時作業ファイルのディレクトリ名として使用されます。</p> <p><code>/tmp</code> 環境変数 <code>TMPDIR</code> が存在しないか <code>NULL</code> で、さらに <code>/var/tmp</code> が存在しない場合、<code>/tmp</code> が一時作業ファイルのディレクトリ名として使用されます。</p> <p><code>ed.hup</code> 端末がハングアップしたとき、作業ファイルがここに保存されます。</p> |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                           |

/usr/bin/ed,  
/usr/bin/red

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin/ed

| 属性タイプ | 属性値               |
|-------|-------------------|
| 使用条件  | system/xopen/xcu4 |

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

/usr/xpg6/bin/ed

| 属性タイプ       | 属性値               |
|-------------|-------------------|
| 使用条件        | system/xopen/xcu6 |
| CSI         | 有効                |
| インタフェースの安定性 | 標準                |

関連項目 [bfs\(1\)](#), [edit\(1\)](#), [ex\(1\)](#), [grep\(1\)](#), [ksh88\(1\)](#), [sed\(1\)](#), [sh\(1\)](#), [stty\(1\)](#), [umask\(1\)](#), [vi\(1\)](#), [fspec\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [standards\(5\)](#)

診断 ?            コマンドエラー

[?file](#)        アクセス不可能なファイル (詳細については [help](#) および [Help](#) コマンドを使用してください)。

最後に [w](#) コマンドを使用して全バッファーを書き出した後にバッファーの内容が変更された場合、[e](#) または [q](#) コマンドによって [ed](#) のバッファーの内容を破棄しようとすると、[ed](#) はユーザーに対して警告を発します。[?](#) を出力して編集を続行するか否かを聞いてきます。ここで再度 [e](#) または [q](#) コマンドを入力すると、これらのコマンドは実行されます。[-s](#) コマンド行オプションは、上述の機能を禁止します。

注意事項 [-](#) オプションは、サポートはされていますが、ドキュメント内ではコマンド構文規格に準拠する [-s](#) オプションで置き換えました ([Intro\(1\)](#) 参照)。

[g](#) または [v](#) コマンドに対して [!](#) コマンドは無効です。

[!](#) コマンドと [e](#)、[r](#)、および [w](#) コマンドからのエスケープコマンド [!](#) は、エディタが制限付きシェル ([sh\(1\)](#) 参照) から起動されているときには使用できません。

正規表現 [RE](#) 内の [\n](#) シーケンスは復帰改行文字と一致しません。

エディタの入力がコマンドファイル (たとえば、`ed file < ed_cmd_file`) からの場合、最初にエラーが発生した時点でエディタは終了します。

`/usr/bin/ed` の場合、環境変数 `LD_PRELOAD` を使用して代替の `malloc()` ライブラリを読み込むと、問題が発生することがあります。



|    |                                                                                                                                                                                                                                                                                                             |
|----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前 | edit - テキストエディタ (ex の初心者ユーザー用の変形版)                                                                                                                                                                                                                                                                          |
| 形式 | <pre> /usr/bin/edit [-  -s] [-l] [-L] [-R] [-r [filename]] [-t tag] [-v] [-V] [-x] [-wn] [-C] [+comm /usr/xpg4/bin/edit [-  -s] [-l] [-L] [-R] [-r [filename]] [-t tag] [-v] [-V] [-x] [-wn] [-C] [- /usr/xpg6/bin/edit [-  -s] [-l] [-L] [-R] [-r [filename]] [-t tag] [-v] [-V] [-x] [-wn] [-C] [- </pre> |

## 機能説明

edit ユーティリティは、コマンド指向エディタの使用を望む、初心者ユーザー対象に推奨されるもので、テキストエディタ ex の変形です。このエディタは、次のオプションが自動的に設定され、ex とまったく同じ様に動作します。

|        |     |
|--------|-----|
| 初心者    | ON  |
| report | ON  |
| 表示モード  | ON  |
| マジック   | OFF |

edit の使用を始めるにあたり、役に立つ簡単な説明を次に示しますが、CRT 端末を使用している場合には、画面エディタ vi について学ぶ方をお勧めします。

既存のファイルの内容を編集する場合は、シェルに対し、コマンド `edit name` で始めます。edit は、そのユーザー編集用のファイルのコピーを作成してユーザーが編集できるようにし、そのファイル内の行数および文字数を表示します。新規のファイルを作成する場合も、コマンド `edit` にファイル名を付け、`edit name` で始めます。この場合には、エディタは `[New File]` と表示して、このファイルが新規のファイルであることを示します。

edit コマンドのプロンプトは、コロン (:) であり、エディタを起動した後は、このコロンが表示されていなければなりません。既存のファイルを編集する場合は、edit のバッファ (編集中のファイルのコピーの名前) に何行か入っています。編集開始時には、edit はファイルの最終行を現在行とします。edit のコマンドのほとんどは、対象行をユーザーが指定しないかぎり、現在行を対象とします。したがって、`print(p と省略できます)` を指定し、キャリッジリターンを入力すると (これは、edit のすべてのコマンドの後に必要です)、現在行が表示されます。現在行を `delete (d)` で削除すると、通常、ファイル中にある次の行が新しい現在行として表示されます。最終行を `delete` すると、新しい最終行が現在行になります。

空のファイルに入力したい場合、または、新しい行を追加したい場合は、`append(a)` コマンドを使用します。このコマンドを実行 (`append` の後に、キャリッジリターンを入力) した後は、edit は、ドット (.) だけからなる行を入力するまで、端末から入力行を読み取ります。入力行は、現在行の後に追加されます。入力した行の最終行が現在行になります。insert(i) コマンドも `append` と同じ様な働きをしますが、入力行は、現在行の後ではなく、前に挿入されます。

`edit` ユーティリティーは、バッファ内の行に行番号を付けます。先頭行が、行番号 1 になります。コマンド `1` を実行すると、`edit` はバッファの先頭行を表示します。このときに、コマンド `d` を実行すると、`edit` は先頭行を削除し、2 行目を 1 行目とし、現在行 (新しい 1 行目) を表示して現在どこにいるかわかるようにします。一般には、現在行は常に、その直前のコマンドの対象となっていた行です。

`substitute(s)` コマンド `s/old/new/` を使用すると、現在行の中のあるテキストを置き換えることができます。この場合、`old` には置き換えたい旧文字列を指定し、`new` には `old` と置き換えたい新文字列を指定します。

`filename(f)` コマンドは、編集中のバッファ内の行数を表示し、このバッファに変更を加えた場合には `[Modified]` と表示します。バッファを変更した後は、`write(w)` コマンドを実行してファイルの内容をセーブすることができます。`quit(q)` コマンドを実行すると、エディタを終了することができます。`edit` を起動したけれども、そのバッファに変更を加えなかった場合は、`write` コマンドを実行してファイルに書き込む必要はありません (ただし、実行しても問題はありませぬ)。バッファに変更を加えた後で、ファイルに書き込まずに `edit` を `quit` コマンドで終了しようとする、メッセージ `No write since last change (:quit!overrides)` が表示され、`edit` は次のコマンドの入力を待ちます。そのバッファを書き込む必要がない場合は、`quit` コマンドの後に感嘆符を付けて (`q!`) 実行してください。バッファは廃棄されて復元不能になり、シェルに戻ります。

`d` および `a` コマンドが使える、ファイル内の行を示す行番号の指定ができるようになると、必要とするあらゆる変更を行うことができます。ただし、`edit` をもう少し頻繁に使用する場合には、少なくとも、他に 2、3 のコマンドを覚えておいたほうが便利です。

`change(c)` コマンドは、現在行を入力行に変更します。`append` の場合と同様、ドット (.) だけからなる行を入力するまで、複数の行を入力できます。`change` では、変更したい行の行番号を指定して、複数行を変更することができます。3,5c のように指定してください。同じ方法を使って、複数行を表示することもできます。1,23p と指定すると、ファイルの先頭から 23 行目までを表示することができます。

`undo(u)` コマンドは、直前に実行したコマンドにより変更されたバッファを、元に戻します。したがって、`substitute` コマンドを実行し、その結果が望んだものではなかった場合、`u` コマンドを実行すると元の行の内容が復元されます。`undo` コマンドを使って `undo` コマンドを取り消すこともできます。`edit` は、コマンドがバッファの複数行を対象にした場合、警告メッセージを表示します。`write` や `quit` などのコマンドは、取り消しできないので注意してください。

バッファ内の次の行を見たい場合は、キャリッジリターンを入力してください。複数の行を見たい場合は、キャリッジリターンの代わりに、`^D` (コントロールキーを押しながら、`d` を押します) を入力してください。これによって、CRT に画面半分の行が表示されるか、または、ハードコピー端末に 12 行が出力されます。`z` コマン

ドを実行すると、前後のテキストを見ることができます。現在行がテキスト表示の中心に表示され、最後に表示される行が現在行になります。''を入力すると、z コマンドを実行する前の現在行に戻ることができます。z コマンドには、他にもオプションがあります。z- は、元の現在行を最後の行とする1画面分のテキスト(または24行)を表示します。z+ は、現在行の次の1画面分を表示します。1画面分より少ない行数で表示したい場合は、z.11 と入力すると、現在行の直前の5行と直後の5行が表示されます(z.n と入力すると、n が奇数の場合は、現在行が中心となり、全部で n 行が表示されます。n が偶数の場合は、現在行を中心として、n-1 行が表示されます)。他のコマンドの後に行数を指定することができます。たとえば、コマンド d5 を使用すると、現在行と、現在行から数えて5行目までの、合計5行を削除することができます。

ファイル内で何かを探す場合には、行番号がわかっている場合は、行番号を使用することができます。ただし、行の挿入や削除を行なった場合には、行番号は変わっているので、これはあまりあてになりません。ファイルの前方および後方に文字列を検索することができます。/text/ という形式のコマンドを使用するとファイルの先頭の方に向かって text を検索でき、?text? を使用するとファイルの最後の方に向かって text を検索できます。検索がファイルの終わりに到達しても text が見つからなかった場合は、ラップして元の場所まで検索を続けます。この検索の便利な機能に /^text/ という形式の検索があり、これは、行の先頭にある text を検索します。同様に /text\$/ は、行の終わりにある text を検索します。どちらのコマンドも、最後に付ける / または ? を省略することができます。

現在行を表す記号名として、ドット(.)があり、これは行の範囲指定の際に最も便利です。., \$p のように指定すると、現在行から、ファイルの最終行までが表示されます。ファイルの最終行に移動するには、最終行を表す記号名 \$ を使用して参照することができます。したがって、コマンド \$d は、ファイルの最終行を削除し、現在行は変わりません。行参照では、算術式も使用できます。たとえば、\$-5 は、最終行の前の行から数えて5番目の行を表し、.+20 は、現在行の次の行から数えて20番目の行を表します。

現在行を見つけるには、'.=' と入力します。テキストのある部分を、ファイル内またはファイル間で、移動またはコピーする場合に便利です。コピーまたは移動したい行の、最初と最後の行番号を調べます。10行目から20行目を移動する場合は、10,20d a と入力すると、これらの行はファイルから削除され、a という名前のバッファに入ります。edit には、このようなバッファが、a から z まで26個あります。バッファ a の内容を現在行の後に入れるには、put a と入力します。これらの行を別のファイルに移動またはコピーする場合には、行をコピーした後に edit(e) コマンドを実行します。このとき、edit chapter2 のように、e コマンドの後に編集したい別のファイルの名前を指定します。削除しないで行をコピーする場合には、d の代わりに yank(y) を使用してください。1つのファイル内だけで移動またはコピーする場合には、名前付きバッファを使用する必要はありません。たとえば、10行目から20行目をファイルの末尾に移動するには、10,20m \$ と入力します。

## オプション

次のオプションは、[ex\(1\)](#)内で `set` コマンドを使用することによって、オン/オフを切り替えることができます。

- C                    暗号化オプション。前述の `-x` オプションと同様ですが、唯一の違いは `ex` の `c` コマンドをシミュレートする点です。`c` コマンドは `ex` の `x` コマンドに似ていますが、読み取られたテキストは、すべて暗号化されていると仮定します。
- l                    LISP プログラムの編集用にセットアップします。
- L                    エディタまたはシステムのクラッシュが原因で保存されたファイルの名前をすべて表示します。
- R                    読み取り専用モード。読み取り専用フラグが設定され、ファイルの不慮の上書きを防ぎます。
- r *filename*        エディタもしくはシステムのクラッシュが発生した後で、*filename* が示すファイルを編集します。(クラッシュが発生したときにバッファの中にあった *filename* のバージョンを回復します)。
- t *tag*              *tag* が示すタグを含むファイルを編集します。そのタグが定義されている地点が編集開始地点となります。
- v                    `vi` を使った編集の画面表示を開始します。単に `vi` コマンドを入力しても同様に実行できます。
- V                    冗長。`ex` コマンドが標準入力によって読み込まれる場合に、その入力を標準エラーに反映します。シェルスクリプト中の `ex` コマンドを実行する場合に役に立ちます。
- x                    暗号化オプション。これを指定すると、`edit` は、`ex` の `x` コマンドをシミュレートし、ユーザーにキーの入力を求めます。このキーを使って、`crypt` コマンドのアルゴリズムに従い、テキストを暗号化したり、暗号を解除したりします。`x` コマンドは、読み取ったテキストが暗号化されているかどうかを判別します。また、一時バッファファイルも `-x` オプションに対して入力したキーの変形バージョンを用いて暗号化されます。
- wn                  デフォルト時のウィンドウサイズとして *n* を設定します。これは、低速の回線でエディタを使用するとき役に立ちます。
- +*command* | -c *command*    *command* で示したエディタコマンドを冒頭に実行して編集処理を開始します。通常は、検索または位置指定用のコマンドが用いられます。

- | -s ユーザーへのすべての対話型フィードバックを抑制します。編集スクリプト実行時に役に立ちます。

*filename* 引数は編集対象の1つまたは複数のファイルを示します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

| 属性タイプ | 属性値               |
|-------|-------------------|
| 使用条件  | system/xopen/xcu4 |
| CSI   | 有効                |

| 属性タイプ | 属性値               |
|-------|-------------------|
| 使用条件  | system/xopen/xcu6 |
| CSI   | 有効                |

関連項目 [ed\(1\)](#), [ex\(1\)](#), [vi\(1\)](#), [attributes\(5\)](#), [XPG4\(5\)](#)

**注意事項** 暗号化オプションは、セキュリティ管理ユーティリティーパッケージで提供されていますが、米国においてのみ使用可能です。

名前 egrep – 完全な正規表現を使用したファイル内のパターン検索

形式

```

/usr/bin/egrep [-bchilnsv] -e pattern_list [file...]
/usr/bin/egrep [-bchilnsv] -f file [file...]
/usr/bin/egrep [-bchilnsv] pattern [file...]
/usr/xpg4/bin/egrep [-bchilnqsvx] -e pattern_list [-f file] [file...]
/usr/xpg4/bin/egrep [-bchilnqsvx] [-e pattern_list] -f file [file...]
/usr/xpg4/bin/egrep [-bchilnqsvx] pattern [file...]

```

機能説明

egrep (*expression grep*) ユーティリティーは、ファイルの中の文字パターンを検索し、そのパターンを含む行をすべて出力します。egrep は、パターンマッチング用に完全な正規表現 (英数字および特殊文字のフルセットを使用する文字列値による表現) を使用します。また、高速決定性アルゴリズムを使用しており、これは指数的に増加する領域を必要とすることもあります。

ファイルが指定されない場合は、egrep は、標準入力を入力とみなします。検索された各行は、通常、標準出力に出力されます。複数の入力ファイルがある場合は、検索された各行の前にファイル名が出力されます。

/usr/bin/egrep

/usr/bin/egrep ユーティリティーは [regex\(5\)](#) のマニュアルページに記述されているように、完全な正規表現を受け入れますが、\*(*、\*)*、\*(*、\*)*、\*{*、\*}*、\*<*、\*>* と \*\n* を除きます。また次の各項目が追加されます。

1. 後ろに + が付いている完全な正規表現。これは、その完全な正規表現が 1 回以上現れるものに一致します。
2. 後ろに ? が付いている完全な正規表現。これは、その完全な正規表現が 0 回または 1 回現れるものに一致します。
3. | または 復帰改行 (NEWLINE) で区切られた複数の完全な正規表現。これは、どちらかの完全な正規表現と一致する文字列に一致します。
4. グループ化のために、丸かっこ ( ) で囲まれた完全な正規表現。

完全な正規表現の中で \$、\*、[、^、|、(、)、\ の各文字を使用する際には注意が必要です。これらの文字は、シェルにとっても特別な意味があるからです。完全な正規表現全体を単一引用符で囲む (') のがもっとも安全です。

演算子の優先順位は、[ ]、\* ? +、連結、|、復帰改行の順です。

/usr/xpg4/bin/egrep

/usr/xpg4/bin/egrep ユーティリティーは、[regex\(5\)](#) のマニュアルページの「EXTENDED REGULAR EXPRESSIONS」の項で述べる 正規表現を使用します。

オプション

/usr/bin/egrep と /usr/xpg4/bin/egrep では次のオプションがサポートされています。



- b 検索された各行の先頭にその行のブロック番号を付けます。文脈によりブロック番号を見つけ出す場合に便利です (最初のブロックは0です)。
- c 一致したパターンを含む行の行数だけを出力します。
- e *pattern\_list* *pattern\_list* (–で始まる完全な正規表現) を検索します。
- f *file* *file* から完全な正規表現のリストを取り込みます。
- h 複数ファイルの検索時にファイル名の出力を抑制します。
- i 比較時に大文字と小文字を区別しません。
- l 一致した行が1行でもあるファイルのファイル名だけを、1ファイルずつ復帰改行で区切って出力します。パターンが2度以上見つかるときは、ファイルの名前を繰り返しません。
- n 各行の先頭にファイル内の行番号を付けます (最初の行は1です)。
- s エラーメッセージだけを表示します。エラーステータスをチェックするのに有用です。
- v 一致するパターンを含む行以外のすべての行を出力します。

`/usr/xpg4/bin/egrep` 次のオプションは、`/usr/xpg4/bin/egrep` でのみサポートされています。

- q 非出力。行が一致したかどうかに関わらず、標準出力には何も書き出しません。入力行に一致した行があった場合、0の終了ステータスで処理を終了します。
- x 固定文字列全体または正規表現と完全に一致する入力行だけを、一致する行とみなします。

オペランド 次のオペランドがサポートされています。

*file* パターンを検索するファイルのパス名。*file* オペランドを指定しないと、標準入力を使用されます。

`/usr/bin/egrep` *pattern* 入力の検索時に用いるパターンを指定します。

`/usr/xpg4/bin/egrep` *pattern* 入力の検索時に用いる1つまたは複数のパターンを指定します。このオペランドは、`-epattern_list` と指定されたものとして扱われます。

使用法 ファイルが2ギガバイト ( $2^{31}$  バイト) 以上ある場合の `egrep` の動作については、[largefile\(5\)](#) を参照してください。

環境変数 `egrep` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC\_COLLATE、LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。

- 終了ステータス    次の終了ステータスが返されます。
- 0    一致するものが1つ以上見つかりました。
  - 1    一致するものが1つも見つかりませんでした。
  - 2    (一致するものが見つかった場合でも) 構文エラーが検出された、またはアクセスできないファイルがあった。

属性    属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| /usr/bin/egrep | 属性タイプ | 属性値            |
|----------------|-------|----------------|
|                | 使用条件  | system/core-os |
|                | CSI   | 未対応            |

| /usr/xpg4/bin/egrep | 属性タイプ | 属性値               |
|---------------------|-------|-------------------|
|                     | 使用条件  | system/xopen/xcu4 |
|                     | CSI   | 有効                |

関連項目    [fgrep\(1\)](#), [grep\(1\)](#), [sed\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [regexp\(5\)](#), [XPG4\(5\)](#)

注意事項    理想的には、grep コマンドは1つだけにすべきですが、広い範囲における消費領域と時間のかねあいに対応できるような1つのアルゴリズムはありません。

1行は仮想記憶に使用できるサイズに制限されています。

/usr/xpg4/bin/egrep    /usr/xpg4/bin/egrep ユーティリティーは /usr/xpg4/bin/grep -E と同一です。 [grep\(1\)](#) を参照してください。移植性の必要なアプリケーションは、/usr/xpg4/bin/grep -E の方を使用してください。



|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |               |                                               |                 |                                               |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------|-----------------------------------------------|-----------------|-----------------------------------------------|
| 名前              | eject – ドライブからの媒体 (CD-ROM など) の取り出し                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |               |                                               |                 |                                               |
| 形式              | <code>eject [-dflqt] [ [ <i>device</i>   <i>nickname</i> ] ]</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |               |                                               |                 |                                               |
| 機能説明            | <p>eject ユーティリティーは、手動取り出しボタンを備えていない、またはマウント済みなどの理由でロックされている可能性のある、リムーバブルメディアのデバイスに対して使用します。デバイスの指定には、名前やニックネームを使用できます。デバイスが指定されていない場合は、デフォルトデバイスが使用されます。</p> <p>このコマンドに応答するのは、プログラム制御下で eject コマンドをサポートしているデバイスだけです。</p> <p>手動でのみ取り出すことができるメディアで eject コマンドを実行すると、メディアの取り出し以外のすべての作業が行われます。たとえば、ファイルシステムがマウントされていれば、そのファイルシステムはマウント解除されます。この場合、eject は、手動でメディアを取り出す準備ができたことを知らせるメッセージを表示します。</p> <p>マウントされているファイルシステムが存在するデバイスからメディアを物理的に取り出してはなりません。eject はデバイスにマウントされているファイルシステムを自動的に検出し、メディアを取り出す前にファイルシステムをマウント解除しようとします。mount(1M) を参照してください。マウント解除が失敗した場合、eject は警告メッセージを出力して終了します。デバイスにマウント済みパーティションが含まれている場合でも、-f を使用するとイジェクトを指定できます。</p> <p>デバイスのフロントパネルの物理メディア取り出しボタンを押しても、対応するドライブに対して eject を呼び出すのと同じ効果があります。すべてのドライブにこの機能があるわけではありません。</p> |               |                                               |                 |                                               |
| オプション           | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"><li>-d デフォルトで取り出されるデバイス名を表示します。</li><li>-f デバイスがビジー状態の場合でも取り出しを強制的に実行します。</li><li>-l 取り出し可能なデバイスのパスとニックネームを表示します。</li><li>-q メディアが存在するかどうかを照会します。</li><li>-t ドライブに CD-ROM トレイを閉じるコマンドを発行します。</li></ul> <p>すべてのデバイスがこのコマンドをサポートしているわけではありません。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                      |               |                                               |                 |                                               |
| オペランド           | <p>次のオペランドがサポートされています。</p> <table><tr><td><i>device</i></td><td>eject の実行対象となるデバイスを /dev ディレクトリ中のデバイス名で指定します。</td></tr><tr><td><i>nickname</i></td><td>eject の実行対象となるデバイスを、eject が認識できるニックネームで指定します。</td></tr></table>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | <i>device</i> | eject の実行対象となるデバイスを /dev ディレクトリ中のデバイス名で指定します。 | <i>nickname</i> | eject の実行対象となるデバイスを、eject が認識できるニックネームで指定します。 |
| <i>device</i>   | eject の実行対象となるデバイスを /dev ディレクトリ中のデバイス名で指定します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |               |                                               |                 |                                               |
| <i>nickname</i> | eject の実行対象となるデバイスを、eject が認識できるニックネームで指定します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |               |                                               |                 |                                               |

ニックネームにはボリュームラベルやデバイスタイプ (cdrom など) を使用できます。

使用例

例1 メディアを取り出す  
次の例では、ボリュームラベルを指定してメディアを取り出します。

```
example> eject 'My Pictures'
```

終了ステータス

- 次の終了ステータスが返されます。
- 0 操作が正常に終了した、あるいは、-q オプションを指定した場合にはメディアがドライブに入っていた。
  - 1 操作が失敗しました。あるいは、-q オプションを指定した場合にはメディアがドライブに入っていなかった。
  - 2 無効なオプションが指定されました。
  - 3 ioctl() 要求が失敗しました。
  - 4 手動でメディアを取り出す準備ができた。

ファイル

/dev/sr0 デフォルトの CD-ROM ファイル (将来のリリースでは削除される予定)

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目

[volcheck\(1\)](#), [mount\(1M\)](#), [rmmount\(1M\)](#), [ioctl\(2\)](#), [attributes\(5\)](#)

**名前** elfdump - オブジェクトファイルの選択部分のダンプ

**形式** elfdump [-cCdegGhHiklmnPrsSuvy] [-p | -w file] [-I *index-expr*] [-N *name*] [-O *osabi*] [-T *type*] *filename*...

**機能説明** elfdump ユーティリティーは、指定されたオブジェクトファイルの選択部分をシンボルでダンプします。オプションを指定することにより、オブジェクトファイルの特定部分を抽出できます。

elfdump ユーティリティーは、[dump\(1\)](#) ユーティリティーと機能的に似ています。elfdump のインタフェースの方が dump よりも新しく、ユーザーフレンドリです。ただし、シェルスクリプト内で使用するような場合は、dump ユーティリティーの方が適しています。

elfdump は、[ar\(1\)](#) で作成したアーカイブファイルの検査にも使用できます。その場合、アーカイブ内の各オブジェクトは、指定されたオプションに従って処理されます。

elfdump を使用すると、任意の ELF オブジェクトの ELF ヘッダー、プログラムヘッダーの配列、およびセクションヘッダーの配列を表示できます。また、次のタイプのセクションに含まれているデータを表示することもできます。

| Category                  | Option | ELF Section Type                                                |
|---------------------------|--------|-----------------------------------------------------------------|
| Dynamic                   | -d     | SHT_DYNAMIC                                                     |
| Global Offset Table (GOT) | -G     | Special. See below.                                             |
| Group                     | -g     | SHT_GROUP                                                       |
| Capabilities              | -H     | SHT_SUNW_cap                                                    |
| Hash Table                | -h     | SHT_HASH                                                        |
| Interpreter               | -i     | Special, see below.                                             |
| Move                      | -m     | SHT_SUNW_move                                                   |
| Note                      | -n     | SHT_NOTE                                                        |
| Relocation                | -r     | SHT_RELA<br>SHT_REL                                             |
| Stack Unwind/Exceptions   | -u     | Special. See below.                                             |
| Syminfo                   | -y     | SHT_SUNW_syminfo                                                |
| Symbol Sort               | -S     | SHT_SUNW_symsort<br>SHT_SUNW_tlssort                            |
| Symbol Table              | -s     | SHT_SYMTAB<br>SHT_DYNSYM<br>SHT_SUNW_LDYNSYM<br>SHT_SUNW_versym |
| Versioning                | -v     | SHT_SUNW_verdef<br>SHT_SUNW_verneed                             |

インタプリタおよびグローバルオフセットテーブルのセクションには特別な ELF セクションタイプはありませんが、代わりに、よく知られた名前 (それぞれ `.interp` と `.got`) を持つ SHT\_PROGBITS セクションとして実装されています。elfdump は、これらの特殊なセクションを認識して表示できます。

スタック巻き戻しおよび例外処理に使用されるセクションは、使用されているコンパイラとプラットフォームによって、ELF セクションタイプが `SHT_PROGBITS` または `SHT_AMD64_UNWIND` になります。これらのセクションは、`.eh_frame`、`.eh_frame_hdr`、および `.exception_ranges` の名前で認識されます。

表示される情報を絞るためのオプションなしで実行されると、`elfdump` は、各オブジェクトの使用可能なすべての情報を表示します。

表示される情報の詳細は、『[リンカーとライブラリ](#)』を参照してください。

## オプション

サポートしているオプションは、次のとおりです。

`-c`  
セクションヘッダー情報をダンプします。

`-C`  
C++ シンボル名を復号化します。

`-d`  
`.dynamic` セクションの内容をダンプします。

`-e`  
ELF ヘッダーをダンプします。

`-g`  
`.group` セクションの内容をダンプします。

`-G`  
`.got` セクションの内容をダンプします。

`-h`  
`.hash` セクションの内容をダンプします。

`-H`  
`.SUNW_cap` 機能セクションの内容をダンプします。

`-i`  
`.interp` セクションの内容をダンプします。

### `-I index-expr`

検査するセクションまたはプログラムヘッダーを、特定のインデックスまたはインデックス範囲で修飾します。たとえば、次を使用して、ファイルの3番目のセクションヘッダーを表示できます。

```
example% elfdump -c -I 3 filename
```

`index-expr` には、上記の例に示したように、特定の項目を指定する負でない単一の整数値を指定できます。または、`index-expr` は、項目の範囲を示すコロン(:)で区切られた2つの負でない単一の整数値で構成することもできます。次の例は、ファイル内の3番目、4番目、および5番目のプログラムヘッダーを表示します。

```
example% elfdump -p -I 3:5 filename
```

インデックスの範囲を指定する場合、2 番目の値を省略すると、ファイル内の最後の項目を意味することができます。たとえば、次の文は 10 番目から最後まですべてのセクションヘッダーを一覧表示します。

```
example% elfdump -c -I 10: filename
```

マッチングオプション (-I、-N、-T) の詳細は、「マッチングオプション」を参照してください。

-k

ELF チェックサムを計算します。 [gelf\\_checksum\(3ELF\)](#) のマニュアルページを参照してください。

-l

省略しない長いセクション名を表示します。

-m

D.SUNW\_move セクションの内容をダンプします。

-n

.note セクションの内容をダンプします。デフォルトでは、elfdump は、このデータを解釈せずに 16 進形式で表示します。コアファイルは例外です。 [core\(4\)](#) で説明されているコアファイルの note のサブセットは、elfdump によって解釈され、高いレベルの形式 NT\_PRSTATUS、NT\_PRPSINFO、NT\_PLATFORM、NT\_AUXV、NT\_ASRS、NT\_PSTATUS および NT\_ZONENAME で表示されます。

-N *name*

検査するセクションまたはプログラムヘッダーを、特定の名前で修飾します。たとえば、複数のシンボルテーブルが入っているファイルにおいて .dynsym シンボルテーブルだけを表示するには、次を使用します。

```
example% elfdump -N .dynsym filename
```

ELF プログラムヘッダーには名前がありません。-p オプションを指定した場合、*name* は、プログラムヘッダータイプを指し、-N オプションの動作は -T オプションの動作と同じになります。たとえば、次を使用して、インタプリタを識別するプログラムヘッダーを表示できます。

```
example% elfdump -p -N PT_INTERP filename
```

マッチングオプション (-I、-N、-T) の詳細は、「マッチングオプション」を参照してください。

-O *osabi*

オブジェクトを解釈するときに適用するオペレーティングシステムの ABI を指定します。*osabi* には、`/usr/include/sys/elf.h` に含まれているいずれかの `ELFOSABI_` 定数の名前または値を指定できます。利便性のため、`ELFOSABI_` 接頭辞をこれらの名前から省略できます。*osabi* では、2 つの値が完全にサポートされ

ています。solaris は、Solaris オペレーティングシステムのネイティブ ABI です。none は、汎用の ELF ABI です。ほかのオペレーティングシステムの ABI のサポートは、不完全であるか、存在しない可能性があります。文字列を取得できない項目は、数値形式で表示されます。

-O を使用せず、オブジェクトの ELF ヘッダーで汎用以外の ABI が指定されている場合、オブジェクトによって指定されている ABI が使用されます。オブジェクトで汎用の ELF ABI が指定されている場合、elfdump は、.note.ABI-tag セクションを検索し、見つかった場合は、linux ABI を持つオブジェクトとしてそのオブジェクトを識別します。それ以外の場合、汎用の ELF ABI を指定しているオブジェクトは、solaris ABI に準拠すると想定されます。

-p

プログラムヘッダーをダンプします。マッチングオプション (-I、-N、-T) を使用すると、個々のプログラムヘッダーを指定できます。詳細は、「マッチングオプション」を参照してください。

-p オプションと -w オプションを同時に指定することはできません。1 回の elfdump 呼び出しでは、どちらかのオプションしか使用できません。

-P

プログラムヘッダーからの情報に基づいて代替のセクションヘッダー情報を生成および使用し、ファイルに含まれているセクションヘッダー情報を無視します。ファイルにセクションヘッダーがない場合、警告メッセージが出力され、このオプションが自動的に選択されます。セクションヘッダーは、プログラムを実行するためにシステムで使用されることはありません。そのため、悪意のあるプログラムでは、そのセクションヘッダーが削除されていたり、偽の情報を提供するように変更されていたりする場合があります。対照的に、プログラムが実行可能であるためには、プログラムヘッダーは正確である必要があります。プログラムヘッダーから生成された合成のセクションヘッダーを使用すると、変更されたセクションヘッダーを持つファイルを調べることができます。

-r

再配置セクション (つまり、.rel[a]) の内容をダンプします。

-S

シンボルテーブルセクション .SUNW\_ldynsym、.dynsym、および .symtab の内容をダンプします。アーカイブの場合は、アーカイブシンボルテーブルもダンプします。マッチングオプション (-I、-N、-T) を使用すると、個々のセクションを指定できます。アーカイブシンボルテーブルは特殊なセクション名 -NARSYM で指定できます。

コアファイルの場合、shndx フィールドには有効な値が入っていないため、shndx フィールドの値は「unknown」になります。

ver ヘッダーの下に、標準のシンボルテーブル情報に加えて、シンボルのバージョン定義索引も出力します。

マッチングオプション (-I、-N、-T) の詳細は、「マッチングオプション」を参照してください。

-S

シンボルソートセクション `.SUNW_dynsym`sort および `.SUNW_dynlssort` で指定された順序でソートされたシンボルテーブルセクション `.SUNW_ldynsym` および `.dynsym` の内容をダンプします。スレッドローカル記憶領域 (TLS) シンボルは、オフセットでソートされます。通常のシンボルはアドレスでソートされます。ソートセクションによって参照されていないシンボルは表示されません。

-T*type*

検査するセクションまたはプログラムヘッダーを、特定のタイプで修飾します。たとえば、複数のシンボルテーブルが入っているファイルにおいて `.dynsym` シンボルテーブルだけを表示するには、次を使用します。

```
example% elfdump -T SHT_DYNSYM filename
```

*type* の値には、数値、または `/usr/include/sys/elf.h` で定義されているいずれかの `SHT_` シンボル名を指定できます。`SHT_` 接頭辞は省略可能であり、*type* では大文字と小文字は区別されません。そのため、上記の例は次のように記述することもできます。

```
example% elfdump -T dynsym filename
```

-p オプションを指定した場合、*type* はプログラムヘッダータイプを指すため、特定のプログラムヘッダーを表示できます。たとえば、次を使用して、インタプリタを識別するプログラムヘッダーを表示できます。

```
example% elfdump -p -T PT_INTERP filename
```

*type* の値には、数値、または `/usr/include/sys/elf.h` で定義されているいずれかの `PT_` シンボル名を指定できます。`PT_` 接頭辞は省略可能であり、*type* では大文字と小文字は区別されません。そのため、上記の例は次のように記述することもできます。

```
example% elfdump -p -T interp filename
```

マッチングオプション (-I、-N、-T) の詳細は、「マッチングオプション」を参照してください。

-u

スタックフレーム巻き戻しおよび例外処理に使用されるセクションの内容をダンプします。

-v

バージョンセクション `.SUNW_version` の内容をダンプします。

-w*file*

マッチングオプション (-I、-N、-T) を使用して指定されたセクションの内容を、指定されたファイルに書き込みます。たとえば、ファイル内の `.text` セクションを抽出するには、次のように指定します。

```
example% elfdump -w text.out -N .text filename
```

マッチングオプション(-I、-N、-T)の詳細は、「マッチングオプション」を参照してください。

-p オプションと -w オプションを同時に指定することはできません。1回のelfdump呼び出しでは、どちらかのオプションしか使用できません。

-y

.SUNW\_syminfo セクションの内容をダンプします。シンボルの属性は、文字トークンによって伝達されます。

- A シンボル定義は、補助フィルタとして機能します。
- B Dとともに割り当てられ、シンボル参照は、関連付けられている依存関係定義に直接結合すべきです。
- C シンボル定義はコピー再配置の結果です。
- D シンボル参照は、定義を含む依存関係へ直接関連付けられます。
- F シンボル定義は、標準フィルタとして機能します。
- I シンボル定義は割り込み処理として機能します。
- L レイジー読み込みの必要がある依存関係に対するシンボル参照です。
- N 外部参照をこのシンボル定義に直接結合することはできません。
- P シンボルは、遅延(延期された)依存関係と関連付けられています。
- S シンボルは、機能と関連付けられています。

オペランド 次のオペランドを指定できます。

*filename* 指定されたオブジェクトファイルの名前

## 使用法

### マッチングオプション

オプション -I、-N、および -T は、まとめてマッチングオプションと呼ばれます。これらのオプションは、インデックス、名前、またはタイプによって、検査するプログラムヘッダーまたはセクションの範囲を絞り込むために使用されます。

マッチングオプションの厳密な解釈は、使用されているほかのオプションによって異なります。

- -p オプションとともに使用した場合、マッチングオプションはプログラムヘッダーを参照します。-I は、プログラムヘッダーのインデックスを指します。-T は、プログラムヘッダーのタイプを指します。プログラムヘッダーには名前がないため、-N オプションは、プログラムヘッダーに対しては -T と同様に動作します。



- マッチングオプションは、オプション `-c`、`-g`、`-m`、`-n`、`-r`、`-s`、`-S`、`-u`、`-w` のいずれかとともに使用された場合、セクションをインデックス、名前、またはタイプで選択するために使用されます。
- オプション `-c`、`-g`、`-m`、`-n`、`-p-r`、`-s`、`-S`、`-u`、`-w` のいずれも指定されず、マッチングオプションが単独で使用された場合、`elfdump` は各オブジェクトを検査し、一致したいずれかのセクションの内容を表示します。

任意の数およびタイプのマッチングオプションを組み合わせ、対象となる `elfdump` の呼び出しで使用できます。この場合、`elfdump` は使用されたすべてのマッチングオプションと一致する、すべての項目の上位集合を表示します。この機能を使用すると、各項目の指定にもっとも簡単な形式を使用し、複雑な項目のグループを選択できます。

ファイル `liblddbg.so`     リンカーデバッグ用ライブラリ

属性     属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値           |
|-------------|---------------|
| 使用条件        | system/linker |
| インタフェースの安定性 | 確実            |

関連項目     [ar\(1\)](#), [dump\(1\)](#), [elffile\(1\)](#), [file\(1\)](#), [nm\(1\)](#), [pvs\(1\)](#), [elf\(3ELF\)](#), [core\(4\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | elfedit - ELF ファイルの検査または編集                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式   | elfedit [-adr] [-e cmd] [-L path] [-o default   simple   num] [infile] [outfile]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 機能説明 | <p>elfedit は、既存の ELF オブジェクトの内容を検査または変更するためのツールです。特に、elfedit は、オブジェクトに含まれている ELF メタデータを変更するために使用されます。ELF ヘッダー、セクションヘッダーテーブル、プログラムヘッダーテーブル、動的セクション、ハードウェアおよびソフトウェア機能、文字列テーブル、シンボルテーブルなどの、オブジェクトに含まれているほとんどの ELF データへのアクセスが提供されます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 構文   | <p>elfedit は、コマンド行 (-e オプション) または標準入力からのコマンドを処理します。標準入力が端末である場合、elfedit では端末編集機能のほか、広範囲のコマンド完了が提供されます。ELF では、特殊な整数値やビットマスクのために多くの標準の記号名が使用されます。elfedit は、このような名前に関する、可能性のあるほとんどの完了を認識します。elfedit コマンドの入力中はいつでも TAB キーを押すことができ、それにより、elfedit は現在のカーソル位置にあるテキストの使用法に関するメッセージや既知の完了を表示します。</p> <p>elfedit の機能は、モジュールの形式で構成されています。各モジュールは、関連する機能に焦点を絞った一連のコマンドを提供します。コマンドは、モジュールとコマンド名をコロン(:)の区切り記号で結合する(空白は入れない)ことによって指定されます。たとえば、dyn:runpath は、dyn モジュールによって提供される runpath コマンドを示します。モジュール名は一意である必要があります。特定のモジュール内のコマンド名はそのモジュール内では一意ですが、その同じコマンド名を複数のモジュールで使用できます。</p> <p>一部のモジュールでは、コマンドの 1 つが、そのモジュールのデフォルトコマンドとして指定されます。このコマンドは、ユーザーがモジュール名のみを指定した場合に実行されます。ほとんどの elfedit モジュールでは、そのモジュールがサポートする ELF ファイルの部分に関して elfdump ユーティリティによって表示されるのと同じ情報を生成する、dump という名前のコマンドが提供されます。モジュールでは一般に、dump がデフォルトコマンドとして指定されます。</p> <p>elfedit コマンドを実行するために使用される構文は、UNIX コマンド行 ユーティリティを使用しているすべてのユーザーが使いやすいように考慮されています。この構文は、空白で区切られたトークンで構成されます。最初のトークンはコマンド名です。コマンドのあとに、ハイフン(-)文字で始まる引数である各オプションが指定されます。オプションのあとに、プレーン引数(オペランド)が指定されます。特定のコマンドに対してオプションやオペランドを 0 個以上指定できますが、この両方が存在する場合は、常にオプションをプレーン引数の前に指定します。特殊オプション -- (2つのハイフン)を使用して、オプションの終わりを表すことができます。このオプションが見つかると、残りの引数はすべて、-で始まっていたとしてもプレーン引数であると見なされます。</p> <p>elfedit トークン内の文字の解釈は、使用されている引用の形式によって異なります。</p> |

## 引用なし

単一引用符 (') または二重引用符 (") の外側では、バックスラッシュ (\) がエスケープ文字として機能します。バックスラッシュ文字が見つかり、elfedit はこの文字を無視し、後続の文字を (その文字自体がバックスラッシュであっても) 文字どおりに処理します。この機能を使用すると、文字列を2つの個別のトークンに分割しなくても、コマンドの文字列引数に空白文字を挿入できます。同様に、この機能を使用すると、引用符またはバックスラッシュをリテラル文字として挿入できます。

## 単一引用符

単一引用符 (') 内では、空白文字によってトークンは区切られず、空白文字はトークン内のリテラル文字として解釈されます。二重引用符 (") およびバックスラッシュ (\) 文字はリテラル文字として解釈され、特別な意味を持ちません。

## 二重引用符

二重引用符 (") 内では、空白文字によってトークンは区切られません。単一引用符文字は文字どおり解釈され、引用機能を持ちません。バックスラッシュ (\) は、文字列リテラル内の C プログラミング言語で使用される場合と同様に動作するエスケープ文字です。

\a      警告 (ベル)

\b      バックスペース

\f      フォームフィード

\n      改行

\r      復帰

\t      水平タブ

\v      垂直タブ

\\      バックスラッシュ

\'      単一引用符

\"      二重引用符

\ooo    8 進定数であり、ooo は 1 桁から 3 桁までの 8 進数 (0...7)

バックスラッシュのあとのその他の文字はすべてエラーです。

コアのコマンドは、sys という名前の内部モジュールに属しています。その他のモジュールはすべて、動的にロード可能な共有可能オブジェクトとしてパッケージ化されます。elfedit は、特定のモジュールを必要とするコマンドが実行された場合、または sys:load コマンド実行の結果として、必要に応じてモジュールをロードします。その特殊な組み込みステータスのため、さらにはそのコマンドが頻繁に使用されるため、elfedit では、sys: 接頭辞を含めなくても sys モジュールの

コマンドを指定できます(たとえば、`sys:load`ではなく`load`)。その他のすべてのモジュールのコマンドにアクセスするには、完全な `module:cmd` 形式を指定する必要があります。

elfedit は、次の標準モジュールとともに提供されます。

|         |                        |
|---------|------------------------|
| cap     | 機能セクション                |
| dyn     | 動的セクション                |
| ehdr    | ELF ヘッダー               |
| phdr    | プログラムヘッダー配列            |
| shdr    | セクションヘッダー配列            |
| str     | 文字列テーブルセクション           |
| sym     | シンボルテーブルセクション          |
| syminfo | Syminfo セクション          |
| sys     | コアに組み込まれた elfedit コマンド |

ステータスおよび  
コマンドドキュメント

ステータスおよびコマンドドキュメント

`status(sys:status)` コマンドは、現在の elfedit セッションに関する情報を表示します。

- 入力および出力ファイル
- オプション設定
- モジュール検索パス
- ロードされたモジュール

どの elfedit モジュールにも、コマンドごとの広範囲なオンラインドキュメントが、UNIX のマニュアルページと同様の形式で含まれています。`help(sys:help)` コマンドは、この情報を表示するために使用されます。elfedit についての詳細を表示するには、elfedit を起動し、ヘルプコマンドを引数なしで使用します。

```
% elfedit
> help
```

elfedit はウェルカムメッセージと、elfedit およびヘルプシステムの使用方法に関する詳細情報を表示します。

モジュールのサマリー情報を取得するには:

```
> help module
```

モジュールが提供する特定のコマンドの詳細なドキュメントを取得するには、次のようにします。

```
> help module:command
```

例としての dyn モジュールおよび dyn:runpath コマンドの使用:

```
> help dyn
```

```
> help dyn:runpath
```

help(sys:help) を使用すると、それ自体に関するヘルプを取得できます。

```
> help help
```

モジュール検索パス

elfedit モジュールは、必要に応じてロードされる共有可能オブジェクトとして実装されます。あるモジュールが必要になると、elfedit は、そのモジュールを実装している共有可能オブジェクトを見つけるためにモジュールパスを検索します。このときのパスは、一連のディレクトリ名をコロン(:)文字で区切ったものです。そのパスには、通常の文字に加えて、次のいずれかのトークンも含めることができます。

- %i 現在の命令セットアーキテクチャー (ISA) の名前 (sparc、sparcv9、i386、amd64) に展開されます。
- %I 64 ビット ISA に展開されます。これは 64 ビットバージョンの elfedit では %i と同じものですが、32 ビットバージョンでは空の文字列に展開されます。
- %O 変更対象のパスの古い値まで展開されます。これは、デフォルトのパスの前またはあとにディレクトリを追加する場合に役立ちます。
- %r elfedit プログラムを保持しているファイルシステムツリーのルート。elfedit がこのツリー内の usr/bin/elfedit としてインストールされていることを前提にしています。標準のシステムでは、これは単純に標準のシステムのルートディレクトリ (/) です。elfedit のコピーをほかの場所にインストールできる開発システムでは、%r を使用することにより、一致した一連のモジュールが使用されることを保証できます。
- %% 1 つの % 文字に展開されます。

elfedit のデフォルトのモジュール検索パスは次のとおりです。

```
%r/usr/lib/elfedit/%I
```

トークンを展開すると、これは次のようになります。

```
/usr/lib/elfedit 32 ビットの elfedit
/usr/lib/elfedit/sparcv9 64 ビットの elfedit (sparc)
/usr/lib/elfedit/amd64 64 ビットの elfedit (x86)
```

デフォルト検索パスは、ELFEDIT\_PATH 環境変数を設定するか、または -L コマンド行オプションを使用することによって変更できます。この両方を指定した場合は、-L オプションが環境変数より優先されます。

## オプション

サポートしているオプションは、次のとおりです。

**-a**

autoprint モードを有効にします。autoprint が有効になっている場合、elfedit は、ELF ファイルが変更されたときに生成される変更された値を出力します。この出力は、-o オプションを使用して変更できる現在の出力形式で示されます。デフォルトの出力形式は、elfdump(1) ユーティリティーによって使用される形式です。autoprint モードは、elfedit が対話的に使用されている場合 (標準入力と標準出力が端末である場合) にデフォルトになります。そのため、-a オプションは、elfedit が非対話型コンテキストで使用されている場合にのみ意味を持ちます。対話型セッションで autoprint を無効にするには、elfedit コマンドを使用します。

```
> set a off
```

**-d**

このオプションが設定されていると、elfedit は、処理されている ELF オブジェクトの内部の動作や詳細が記述された情報メッセージを発行します。これは、内部で実行されている動作を深く理解したい場合に役立ちます。

**-e cmd**

編集コマンドを指定します。複数の -e オプションを指定できます。0 コマンド行に編集コマンドが存在する場合、elfedit はバッチモードで動作します。ファイルを開いたあと、elfedit は各コマンドを指定された順序で実行します。そのあと、変更されたファイルが保存され、elfedit が終了します。バッチモードは、シェルスクリプトやメイクファイルから単純な操作を実行する場合に役立ちます。

**-L path**

elfedit モジュールを検索するためのデフォルトのパスを設定します。モジュールは、このマニュアルページの「モジュール検索パス」セクションで説明されています。

**-o default|simple|num**

ELF データを表示するために使用される形式。このオプションによって、セッションの現在の形式が確立されます。この形式は、elfedit セッション内から set(sys:set) コマンドを使用するか、またはセッション内で実行される個々のコマンドに -o オプションを指定することによって変更できます。

**default**    デフォルトの形式では、人間が見ることを目的にした形式で出力を表示します。この形式は、elfdump ユーティリティーによって使用される形式と同じです。

**num**        整数値は常に、整数の形式で表示されます。文字列は、包含する文字列テーブルへの整数オフセットとして表示されます。

**simple**      ELF ファイル内から文字列を表示する場合は、文字列のみが表示されます。整数値は、可能な場合は記号定数として表示され、それ以外

の場合は整数の形式で表示されます。タイトルやヘッダーなどの補足的な出力は表示されません。

**-r**

読み取り専用モード。入力ファイルは読み取り専用アクセス用に開かれ、編集セッションの結果は保存されません。elfedit では、**-r** が指定された場合、*outfile* 引数は許可されません。ファイルを変更する意図がない場合は、読み取り専用モードを強くお勧めします。誤った変更に対する追加の保護を提供することに加えて、ユーザーが書き込み権を持っていないファイルの検査も可能になります。

## オペランド

次のオペランドがサポートされています。

### *infile*

処理する ELF オブジェクトを含む入力ファイル。

実行可能ファイル (ET\_EXEC)、共有オブジェクト (ET\_DYN)、または再配置可能オブジェクトファイル (ET\_REL) を指定できます。アーカイブは、直接にはサポートされていません。アーカイブ内のオブジェクトを編集するには、オブジェクトを抽出し、そのコピーを編集してから、それをアーカイブ内に戻す必要があります。

*infile* が存在しない場合、elfedit は、**sys:** モジュールからのコマンドの実行のみが許可される制限されたモードで実行されます。このモードは主に、**help** (**sys:help**) コマンドから使用可能なコマンドドキュメントへのアクセスを許可するためのものです。

*infile* が存在し、*outfile* が指定されていない場合、elfedit は所定のファイルを編集し、その結果を同じファイルに書き込むため、元のファイルの内容が上書きされます。通常、elfedit をこのモードで使用することはやめ、出力ファイルを指定することをお勧めします。結果として得られるファイルがテストされ、検証されたあと、そのファイルを元のファイルの場所に移動できます。

**-r** オプションを使用すると、*infile* を読み取り専用アクセス用に開くことができます。これは、変更したくない既存のファイルを検査する場合に役立ちます。

### *outfile*

出力ファイル。*infile* と *outfile* の両方が存在する場合は、*infile* が読み取り専用アクセス用に開かれ、変更されたオブジェクト内容が *outfile* に書き込まれます。

## 使用法

システムでサポートされている場合、elfedit は 64 ビットアプリケーションとして実行されるため、2G バイト (2<sup>31</sup> バイト) 以上のファイルを処理できます。

起動時に、elfedit は **libelf** を使用して入力ファイルを開き、その内容のコピーを編集のためにメモリー内にキャッシュします。そのあと、1 つ以上のコマンドを実行できます。変更されたオブジェクトを必要に応じて出力ファイルに書き込むことによってセッションが完了し、次に終了します。



*infile*が存在しない場合、*elfedit* は、*sys* モジュールからのコマンドの実行のみが許可される制限されたモードで実行されます。このモードは主に、*help*(*sys:help*) コマンドから使用可能なコマンドドキュメントへのアクセスを許可するためのものです。

1つ以上の *-e* オプションが指定されている場合、指定されたコマンドは指定された順序で実行されます。*elfedit* は、指定されたコマンドの直後に *write*(*sys:write*) と *quit*(*sys:quit*) への暗黙の呼び出しを追加します。それにより、出力ファイルが書き込まれ、*elfedit* プロセスが終了します。シェルスクリプトやメイクファイルでは、この形式の使用が便利です。

*-e* オプションが指定されていない場合、*elfedit* は標準入力からコマンドを読み取り、それらのコマンドを指定された順序で実行します。呼び出し側は、このモードで実行中に作業を保存して終了するには *write*(*sys:write*) コマンドと *quit*(*sys:quit*) コマンドを明示的に発行する必要があります。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 致命的なエラーが発生しました。
- 2 無効なコマンド行オプションが指定された。

使用例 次の例では、シェルプロンプト (%) と *elfedit* プロンプト (>) での *elfedit* の対話型使用が示されています。ユーザーは、これらのいずれの文字も入力しないでください。

#### 例1 実行可能ファイルの実行パスの変更

次の例では、共有可能オブジェクト用の *lib* ディレクトリが隣接する *bin* ディレクトリ内にインストールされた、*prog* という名前の実行可能ファイルを前提にしています。次のコマンドは、その実行可能ファイルの *runpath* を *lib* ディレクトリに設定します。

```
elfedit -e 'dyn:runpath $ORIGIN/../lib'
```

*-e* オプションの引数での単一引用符の使用は、シェルがこのコマンド全体を *elfedit* に確実に1つの引数として渡すようにするために必要です。

あるいは、*elfedit* を非バッチモードでを使用することによって、同じ操作を実行できます。

```
% elfedit prog
> dyn:runpath $ORIGIN/../lib
 index tag value
 [30] RUNPATH 0x3e6 $ORIGIN/../lib
> write
> quit
```



### 例1 実行可能ファイルの実行パスの変更 (続き)

runpathなどの要素や、必要なエントリの追加または変更は、オブジェクト内にパディングが存在する場合にのみ実現できる可能性があります。「注意事項」を参照してください。

### 例2 ハードウェア機能ビットの削除

実行のためにオプションハードウェアのサポートを必要とするオブジェクトは、必要な機能を指定したビットのマスクを含む機能セクションを使用して構築されます。実行時リンカー(ld.so.1)は、このマスクを実行中のシステムの属性に対してチェックして、現在のシステムが特定のオブジェクトを実行できるかどうかを判定します。そのシステム上で使用できない機能を必要とするプログラムは、実行を妨げられます。

このチェックによって、必要なハードウェアサポートを明示的にチェックしない不備なプログラムが紛らわしい方法でクラッシュすることが防止されます。ただし、記述されるプログラムが、実行時にシステムの機能を明示的にチェックすることは不便な場合があります。このようなプログラムは、ハードウェアでサポートされている場合に使用する最適化されたコードを備える一方、それ以外の場合は、より低速ではあっても実行可能な汎用のフォールバックバージョンを提供していることがあります。この場合は、ハードウェア互換性マスクによって、このようなプログラムの古いハードウェア上での実行が妨げられます。このような場合は、関連するビットをマスクから削除すると、そのプログラムを実行できます。

次の例では、SSE3 CPU 拡張機能を使用する x86 バイナリから AV\_386\_SSE3 ハードウェア機能を削除します。これにより、SSE3 を使用するための機能を検証する責任が、実行時リンカーからプログラム自体に移転されます。

```
elfedit -e 'cap:hw1 -and -cmp sse3' prog
```

### 例3 オブジェクトからの情報の読み取り

elfedit を使用すると、対象を絞った特定の情報をオブジェクトから抽出できます。次のシェルコマンドは、ファイル /usr/bin/ls に含まれているセクションヘッダーの数を読み取ります。

```
% SHNUM=elfedit -r -onum -e 'ehdr:e_shnum' /usr/bin/ls'
% echo $SHNUM
29
```

Solaris のバージョンや、使用しているマシンの種類に応じて、異なる値が取得されることがあります。-r オプションを指定すると、ファイルが読み取り専用で開かれるため、通常のアクセス権を持つユーザーはファイルを開くことができ、また重要なシステム実行可能ファイルが誤って破損されることが防止されます。num 出力形式は、無関係なテキストを含まない、目的の値のみを取得するために使用されます。

例3 オブジェクトからの情報の読み取り (続き)

同様に、次のコマンドは、シンボル `unlink` のシンボルの型を C 実行時ライブラリから抽出します。

```
% TYPE='elfedit -r -osimple -e 'sym:st_type unlink' \
/lib/libc.so'
% echo $TYPE
STT_FUNC
```

例4 実行可能ファイルの ASLR 設定の指定

実行可能ファイルのデフォルトのアドレス空間配置のランダム化 (ASLR) 動作は、`DT_SUNW_ASLR` 動的セクションエントリを使って指定します。次に、指定されたプログラムの ASLR を有効または無効にする方法を示します。

```
% elfedit prog
> dyn:sunw_aslr enable
index tag value
[40] SUNW_ASLR 0x2 ENABLE
> dyn:sunw_aslr disable
index tag value
[40] SUNW_ASLR 0x1 DISABLE
```

環境変数

- ELFEDIT\_PATH** デフォルトのモジュール検索パスを変更します。モジュール検索パスは、このマニュアルページの「モジュール検索パス」セクションで説明されています。
- LD\_NOEXEC\_64** 64 ビットの `elfedit` の自動実行を抑止します。デフォルトでは、64 ビットバージョンの `elfedit` は、システムが 64 ビットに対応している場合に実行されます。
- PAGER** `elfedit` から画面への出力を対話的に提供します。設定されていない場合は、`more` が使用されます。[more\(1\)](#) を参照してください。

ファイル

- `/usr/lib/elfedit` 編集コマンドを提供するために必要に応じてロードされる `elfedit` モジュールのデフォルトディレクトリ。
- `~/.teclarc` コマンド行編集のための個人用 `tecla` カスタマイズファイル。[tecla\(5\)](#) を参照してください。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値           |
|-------------|---------------|
| 使用条件        | system/linker |
| インタフェースの安定性 | 確実            |

関連項目 `dump(1)`, `elfdump(1)`, `ld.so.1(1)`, `more(1)`, `nm(1)`, `pvs(1)`, `sxadm(1M)`, `elf(3ELF)`, `libelf(3LIB)`, `tecla(5)`, `attributes(5)`

### 『リンカーとライブラリ』

## 警告

`elfedit` は、ELF システムのテストおよび開発のためのツールとして設計されています。このツールでは、オブジェクト内のほぼすべての ELF メタデータの検査および変更のための機能が提供されます。無効な、または使用できない ELF ファイルを生成する可能性のある編集でも、何も出力せずに許可されます。ユーザーは、ELF の形式と、それを制御している規則や規約に関する知識を持っているものと想定されています。`elfedit` を使用する場合は、『リンカーとライブラリ』が役立ちます。

`elfedit` を使用すると、ユーザーはオブジェクト内の ELF メタデータを変更できますが、実際のプログラムのコードを理解したり、変更したりすることはできません。そのため、タイプ、サイズ、配置などの ELF 属性を、そのファイルの実際の内容に一致しない方法で設定すると、壊れていて、使用できない出力オブジェクトが生成される可能性があります。このような変更はリンカーコンポーネントのテストに役立つ場合がありますが、それ以外の場合は避けるようにしてください。

`dyn:runpath` コマンドを使用してオブジェクトの `runpath` を変更するなどの、より高いレベルの操作は安全であり、このセクションで詳細に説明している種類のリスクを冒すことなく実行できます。

## 注意事項

`elfedit` でサポートされているすべての ELF 操作を、すべての ELF オブジェクトに対して正常に実行できるわけではありません。`elfedit` は、ファイル内にある既存のセクションによって制約されます。

特に注意する点の1つは、`elfedit` が特定のオブジェクトの `runpath` を変更できない可能性があることです。`runpath` を変更するには、次の条件を満たしている必要があります。

- 目的の文字列が動的な文字列テーブル内にすでに存在するか、またはこのセクション内に新しい文字列を追加するための予約された十分な領域が存在する必要があります。オブジェクトに文字列テーブルの予約領域が存在する場合は、`.dynamic DT_SUNW_STRPAD` 要素の値がその領域のサイズを示します。これをチェックするには、次の `elfedit` コマンドを使用できます。

```
% elfedit -r -e 'dyn:tag DT_SUNW_STRPAD' file
```

- 動的セクションにすでに `runpath` 要素が存在するか、またはこの要素を入力できる使用可能な未使用の動的スロットが存在する必要があります。既存の `runpath` の存在をテストするには、次のようにします。

```
% elfedit -r -e 'dyn:runpath' file
```

動的セクションは、タイプ `DT_NULL` の要素を使用して、そのセクション内にある配列を終了します。最後の `DT_NULL` は変更できませんが、この要素が複数存在す

る場合、elfeditはそのいずれかを runpath 要素に変換できます。追加の動的スロットをテストするには、次のようにします。

```
% elfedit -r -e 'dyn:tag DT_NULL' file
```

古いオブジェクトには、このような操作を完了するために必要な追加の領域が存在しません。これに必要な領域は、Solaris Express Community Edition リリースで導入されました。

ある操作が失敗した場合は、-d (debug) オプションを使用して出力された詳細情報が、その原因を見つけるために非常に役立ちます。

elfedit モジュールは、ELF 構造内のフィールドを直接操作するコマンドはそのフィールドと同じ名前を持つが、より高いレベルの概念を実装するコマンドはそれに当てはまらないという規約に従っています。たとえば、ELF ヘッダー内の `e_flags` フィールドを操作するためのコマンドには `ehdr:e_flags` という名前が付けられています。そのため一般には、モジュールを識別し、ELF フィールドの名前を持つコマンドを探すことによって、そのフィールドを変更するためのコマンドが見つかります。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | elffile - ELF ファイルタイプの識別                                                                                                                                                                                                                                                                                                                                                                                                  |
| 形式    | elffile [-s basic   detail   summary] filename...                                                                                                                                                                                                                                                                                                                                                                         |
| 機能説明  | <p>elffile ユーティリティーは、ELF オブジェクトとそれに関連するファイルタイプでを使用することを目的にした、file コマンドの特殊なバリエーションです。elffile は、次の種類のファイルを識別できます。</p> <p>アーカイブ</p> <p>file によって提供される情報に加えて、elffile はアーカイブメンバーのタイプを識別します。</p> <p>ELF オブジェクト/実行時リンカーの構成ファイル</p> <p>elffile は、file と同じ出力を提供します。</p> <p>その他の種類のファイルはすべて、ELF 以外として報告されます。このようなファイルをそれ以上分類しようとする試みは行われません。file ユーティリティーは、汎用ファイルの識別に推奨されます。</p>                                                    |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-s basic   detail   summary</p> <p>提供される出力の形式を指定します</p> <p>basic</p> <p>file によって使用されるのと同じ形式の 1 行の説明を生成します。</p> <p>detail</p> <p>アーカイブ以外の場合、summary の出力は basic と同じです。アーカイブを処理する場合は、基本出力行のあとに、アーカイブメンバーごとの 1 行の出力が続きます。</p> <p>summary</p> <p>アーカイブ以外の場合、summary の出力は basic と同じです。アーカイブを処理する場合は、アーカイブ内容のサマリーの説明が基本出力の最後に追加されます。-s オプションが指定されていない場合、elffile はデフォルトで summary 形式を使用します。</p> |
| 注意事項  | <p>サマリー形式を使用しているときにアーカイブに対して生成される出力は、そのアーカイブの内容によって異なります。アーカイブに、同じプラットフォームに対するオブジェクトの同種コレクションが含まれている場合は、そのプラットフォームの詳細が 1 つのオブジェクトに使用されるのと同じ形式で示されます。それ以外の場合は、サマリーの説明が生成されます。詳細形式を使用すると、個々のアーカイブメンバーに関するより具体的な情報を取得できます。</p> <p>サマリー形式と詳細形式では、アーカイブのすべてのメンバーを検査する必要があります。実行の速度はアーカイブメンバーの数に比例し、きわめて大きなアーカイブでは遅くなる場合があります。</p>                                                                                              |

## 使用例

## 例1 アーカイブのサマリー出力の表示

次の例では、異なる内容を含むアーカイブに対する `elffile` からのサマリー出力を表示します。次のアーカイブが使用されます。

```
same_elf.a
 1つのプラットフォームに対する ELF オブジェクト。

mixed_elf.a
 複数のプラットフォームに対する ELF オブジェクト。

mixed.a
 ELF オブジェクトおよび ELF 以外のファイル。

not_elf.a
 ELF 以外のファイル。
```

アーカイブのサマリー出力は、アーカイブメンバーのタイプによって異なります。

```
example% elffile same_elf.a mixed_elf.a mixed.a not_elf.a
same_elf.a: current ar archive, 32-bit symbol table,
 ELF 64-bit LSB relocatable AMD64 Version 1
mixed_elf.a: current ar archive, 32-bit symbol table,
 mixed ELF content
mixed.a: current ar archive, 32-bit symbol table,
 mixed ELF and non-ELF content
not_elf.a: current ar archive, non-ELF content
```

## 例2 アーカイブの詳細出力のフィルタ処理

`elffile` からの詳細出力では、アーカイブの1行の出力が生成され、そのあとにアーカイブメンバーごとの1行の出力が続きます。この出力を容易にフィルタ処理して、情報をさまざまな形式で提供できます。次の例では、64ビットのx86システム用の再配置可能オブジェクトを含む `libCstd`. アーカイブを使用してこれを示しています。このアーカイブに対するフィルタ処理されていない `elffile` の詳細出力は次のとおりです。

```
example% elffile -s detail libCstd.a
libCstd.a: current ar archive, 32-bit symbol table
libCstd.a(bitset.o): ELF 64-bit LSB relocatable AMD64 Version 1 [CMOV]
libCstd.a(complex.o): ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE CMOV FPU]
libCstd.a(limits.o): ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE FPU]
libCstd.a(limitsinit.o): ELF 64-bit LSB relocatable AMD64 Version 1
libCstd.a(stdexcept.o): ELF 64-bit LSB relocatable AMD64 Version 1 [SSE CMOV]
...
```

この出力は、各オブジェクトに、実行のために必要なハードウェア機能のタグが付けられていることを示しています。これらの機能タグは、各オブジェクト内のコードによって異なります。次のコマンドは、`elffile` からの出力をフィルタ処理

して、一意の各機能マスクを識別し、アーカイブ内の各マスクを含むオブジェクトの数をカウントします。sed コマンドは、出力からアーカイブメンバー名を削除するために使用され、その結果、同じ機能マスクを持つすべてのアーカイブメンバーの出力が同じになります。sort コマンドは、これらの同じ行をグループ化するために使用され、uniq コマンドは、一意の各グループをグループごとの1行に置き換え、各行の先頭にその行がグループ内で出現した回数を示すために使用されます。

```
example% elffile -s detail libCstd.a | sed 's,(.*)',, ' | sort -f | uniq -c
 1 libCstd.a: current ar archive, 32-bit symbol table
777 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1
 1 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [CMOV FPU]
126 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [CMOV]
 12 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [FPU]
 69 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE CMOV]
 2 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 CMOV]
 3 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE CMOV FPU]
 3 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE CMOV]
 1 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE FPU]
 2 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2 SSE]
220 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE2]
 4 libCstd.a: ELF 64-bit LSB relocatable AMD64 Version 1 [SSE]
```

終了ステータス 次の終了ステータスが返されます。

0 正常終了  
>0 エラーが発生しました

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値           |
|-------------|---------------|
| 使用条件        | system/linker |
| インタフェースの安定性 | 確実            |

関連項目 [ar\(1\)](#), [dump\(1\)](#), [elfdump\(1\)](#), [file\(1\)](#)

『リンカーとライブラリ』

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | elfsign – バイナリへの署名                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 形式    | <pre> /usr/bin/elfsign sign [-v] -k <i>private_key</i> -c <i>certificate_file</i> -e <i>elf_object</i> [-F <i>format</i>] [<i>file</i>]...  /usr/bin/elfsign sign [-v] -c <i>certificate_file</i> -e <i>elf_object</i> -T <i>token_label</i> [-P <i>pin_file</i>] [-F <i>format</i>] [<i>file</i>]...  /usr/bin/elfsign verify [-c <i>certificate_file</i>] [-v] -e <i>elf_object</i> [<i>file</i>]...  /usr/bin/elfsign request -r <i>certificate_request_file</i> {-k <i>private_key</i>   -T <i>token_label</i>}  /usr/bin/elfsign list -f <i>field</i> -c <i>certificate_file</i>  /usr/bin/elfsign list -f <i>field</i> -e <i>elf_object</i> </pre>                                                                                |
| 機能説明  | <p><b>list</b>      1つの証明書ファイルまたは署名されたelfオブジェクトからの標準出力情報を一覧表示します。選択されたフィールドは1行に表示されます。指定されたフィールドが指定されたファイルに適用されない場合、コマンドは標準出力なしで終了します。このサブコマンドのこの出力は、スクリプトやほかのコマンドで使用されることを目的にしています。</p> <p><b>request</b>    秘密鍵と PKCS#10 証明書要求を生成します。Solaris 暗号化フレームワークで使用される PKCS#10 証明書要求。秘密鍵がトークンデバイスで作成される場合、elfsign は、そのトークンデバイスを更新するために必要な PIN の入力を求めます。証明書を取得するには、電子メールアドレス <code>solaris-crypto-req_ww@oracle.com</code> に PKCS#10 証明書要求を送信する必要があります。</p> <p>elfsign のユーザーは、Solaris 暗号化フレームワークで使用されるバイナリに署名する前に、まず証明書要求を生成して証明書を取得する必要があります。</p> <p><b>sign</b>      指定された秘密鍵と証明書ファイルを使用して、elfオブジェクトに署名します。</p> <p><b>verify</b>    既存の署名されたオブジェクトを検証します。-c が指定されていない場合は、指定された証明書を使用するか、または <code>/etc/crypto/certs</code> で適切な証明書を検索します。</p> |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-c <i>certificate_file</i>      PEM/PKCS#7 または ASN.1 BER 形式の X.509 証明書へのパスを指定します。</p> <p>-e <i>elf_object</i>            署名または検証されるオブジェクトへのパスを指定します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |



- e** オプションは、複数のオブジェクトを署名または検証するために複数回指定できます。
- F *format*** **sign** サブコマンドの場合は、署名の形式を指定します。有効な形式オプションは次のとおりです
- rsa\_md5\_sha1** デフォルト形式の Solaris 10 および更新。rsa\_md5\_sha1 形式は「廃止」です。
- rsa\_sha1** このリリースのデフォルト形式。
- rsa\_md5\_sha1** 以外の形式には、署名が適用された時間を示す、署名による情報タイムスタンプが含まれています。このタイムスタンプは暗号的に安全ではなく、また検証の一部としても使用されません。
- f *field*** **list** サブコマンドの場合は、出力にどのフィールドを表示するかを指定します。
- 証明書ファイルの有効なフィールド指定子は次のとおりです。
- subject** サブジェクト DN (識別名)
- issuer** 発行者 DN
- elf** オブジェクトの有効なフィールド指定子は次のとおりです。
- format** 署名の形式
- signer** オブジェクトを署名するために使用される証明書のサブジェクト DN
- time** 署名が適用された時間 (ロケールのデフォルト形式)
- k *private\_key*** PKCS#11 トークンを使用していないときの秘密鍵ファイルの場所を指定します。このファイルは、Solaris 固有の形式の RSA 秘密鍵ファイルです。**request** サブコマンドとともに使用された場合、これは新しく生成された鍵の出力ファイルです。
- k** オプションと **-T** オプションの両方を指定すると、エラーになります。
- P *pin\_file*** トークンデバイスにアクセスするための PIN を保持しているファイルを指定します。*pin\_file* で PIN が指定されていない場合、elfsign は PIN の入力を求めます。

- T オプションを指定せずに -P オプションを指定すると、エラーになります。
- r *certificate\_request\_file* PKCS#10 形式の証明書要求ファイルへのパスを指定します。
- T *token\_label* pktool によって提供された、秘密鍵を保持している PKCS#11 トークンデバイスのラベルを指定します。
- T オプションと -k オプションの両方を指定すると、エラーになります。
- v より詳細な情報を要求します。追加の出力には署名者と、署名形式に含まれている場合は、オブジェクトが署名された時間が含まれます。これは、安定した解析可能な出力ではありません。

オペランド 次のオペランドを指定できます。

*file* 署名または検証される 1 つ以上の elf オブジェクト。-e オプションを使用して、またはほかのすべてのオプションのあとに、少なくとも 1 つの elf オブジェクトを指定する必要があります。

## 使用例

例1 ファイル内の鍵/証明書を使用した ELF オブジェクトの署名

```
example$ elfsign sign -k myprivatekey -c mycert -e lib/libmylib.so.1
```

例2 elf オブジェクトの署名の検証

```
example$ elfsign verify -c mycert -e lib/libmylib.so.1
elfsign: verification of lib/libmylib.so.1 passed
```

例3 証明書要求の生成

```
example$ elfsign request -k mykey -r req.pkcs10
Enter Company Name / Stock Symbol or some other globally
unique identifier.
This will be the prefix of the Certificate DN: SUNW
```

例4 オブジェクトに関する情報の判定

```
example$ elfsign list -f format -e lib/libmylib.so.1
rsa_md5_sha1
```

```
example$ elfsign list -f signer -e lib/libmylib.so.1
CN=VENDOR, OU=Software Development, O=Vendor Inc.
```

終了ステータス 次の終了ステータスが返されます。

| 値 | 意味                                                   | サブコマンド              |
|---|------------------------------------------------------|---------------------|
| 0 | 操作の成功                                                | sign/verify/request |
| 1 | 無効な引数                                                |                     |
| 2 | ELF オブジェクトの検証に失敗しました                                 | verify              |
| 3 | ELF オブジェクトを開けません                                     | sign/verify         |
| 4 | 証明書ロードをできないか、または無効な証明書です                             | sign/verify         |
| 5 | 非公開鍵をロードできないか、秘密鍵が無効か、またはトークンラベルが無効な署名です             | sign                |
| 6 | 署名の追加に失敗しました                                         | sign                |
| 7 | 署名されていないオブジェクトを検証しようとしているか、またはオブジェクトがELF ファイルではありません | verify              |

ファイル /etc/crypto/certs -c フラグが使用されていない場合に verify サブコマンドで検索されるディレクトリ。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                |
|-------------|------------------------------------|
| 使用条件        | developer/base-developer-utilities |
| インタフェースの安定性 | 下記を参照。                             |

elfsign コマンドおよびサブコマンドは「確実」です。アプリケーションはelfsign の出力形式に依存すべきではありませんが、list サブコマンドの出力形式は「確実」です。

関連項目 [date\(1\)](#), [pktool\(1\)](#), [cryptoadm\(1M\)](#), [libpkcs11\(3LIB\)](#), [attributes\(5\)](#)

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | elfwrap - ELF ファイルにデータをラップする                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 形式   | elfwrap [-64] [-o <i>relobj-file</i> ] [-z target=sparc   x86]<br><i>data-file</i> ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 機能説明 | <p>elfwrap ユーティリティーは、1 つ以上のデータファイルから ELF 再配置可能オブジェクトファイルを作成します。再配置可能オブジェクトは、個々のセクション内の各データファイルを、セクションの参照に使用できるシンボルとともにカプセル化します。再配置可能オブジェクトを含めておくと、以後のリンク編集で便利です。ユーザーは、関連付けられているシンボルを使用すると、カプセル化されたデータを参照できます。</p> <p>デフォルトでは、elfwrap が実行されるマシンに適した 32 ビット ELF 再配置可能オブジェクトが作成されます。-64 オプションを使用すると、64 ビット ELF 再配置可能オブジェクトを作成できます。-z target オプションを使用すると、特定の機械タイプ用の再配置可能オブジェクトを作成できます。</p> <p>注-elfwrap でカプセル化されたデータはすべて、対象のターゲットに適した形式である必要があります。</p> <p>デフォルトでは、再配置可能オブジェクト <code>a.wrap.o</code> が作成されます。-o オプションを使用すると、別の再配置可能オブジェクト名を指定できます。</p> <p>各データファイルの <code>basename(1)</code> を使用して、さまざまな ELF 情報を作成します。たとえば、入力データファイルが <code>ISV/isv-data</code> である場合、次の ELF 情報が再配置可能オブジェクト内に作成されます。</p> <p><code>.isv-data</code> という名前の ELF セクション<br/>このセクションには、入力データファイルのすべての内容が含まれます。</p> <p><code>isv-data_start</code> という名前の ELF シンボル<br/>このシンボルは、<code>.isv-data</code> セクションの開始アドレスを示します。</p> <p><code>isv-data_end</code> という名前の ELF シンボル<br/>このシンボルは、<code>.isv-data</code> セクションのあとにある、最初の場所のアドレスを示します。</p> <p>オプション</p> <p>サポートしているオプションは、次のとおりです。</p> <p>-64<br/>64 ビット ELF 再配置可能オブジェクトを作成します。</p> <p>-o <i>relobj-file</i><br/><i>relobj-file</i> という名前の再配置可能オブジェクトを生成します。</p> <p>-z target=sparc   x86<br/>出力される再配置可能オブジェクトの機械タイプを指定します。サポートされるターゲットは、sparc および x86 です。-64 オプションも存在していて、対応する 64 ビット機械タイプが使用されている場合を除き、指定されたターゲット用の 32 ビット機械タイプが使用されます。デフォルトでは、elfwrap が実行される</p> |

マシン用の 32 ビットの再配置可能オブジェクトが生成されます。

## 使用例

次の例は、システム passwd ファイルとシステム group ファイルを再配置可能オブジェクト passgroup.o 内にカプセル化します。

```
example% elfwrap -o passgroup.o /etc/passwd /etc/group
example% elfdump -s passgroup.o | egrep "passwd|group"
[2] 0x00000000 0x00000000 SECT LOCL D 0 .passwd
[3] 0x00000000 0x00000000 SECT LOCL D 0 .group
[7] 0x00000000 0x000002f0 OBJT GLOB D 0 .passwd passwd_start
[8] 0x000002f0 0x00000000 OBJT GLOB D 0 .passwd passwd_end
[9] 0x00000000 0x00000121 OBJT GLOB D 0 .group group_start
[10] 0x00000121 0x00000000 OBJT GLOB D 0 .group group_end
example% strings -N.passwd passgroup.o | head -1
root:x:0:0:Super-User:/:usr/sbin/sh
example% strings -N.group passgroup.o | head -1
root::0:
```

この再配置可能オブジェクトは、次のユーザーコードから参照できます。

```
example% cat main.c
#include <stdio.h>

extern char passwd_start, passwd_end;

void main()
{
 char *pstart = &passwd_start, *pend = &passwd_end;
 char *str, *lstr;

 for (lstr = str = pstart; str < pend; str++) {
 if ((*str == '\n') && (str != (pend - 1))) {
 (void) printf("%.s", (++str - lstr), lstr);
 lstr = str;
 }
 }
}

example% cc -o main main.c passgroup.o
example% ./main
root:x:0:0:Super-User:/:usr/sbin/sh
....
nobody4:x:65534:65534:SunOS 4.x NFS Anonymous Access User:/:
```

## ファイル

a.wrap.o 作成されるデフォルトの再配置可能オブジェクトファイル。

## 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                |
|-------------|------------------------------------|
| 使用条件        | developer/base-developer-utilities |
| インタフェースの安定性 | 确实                                 |

関連項目

[elfdump\(1\)](#), [ld\(1\)](#), [strings\(1\)](#), [elf\(3ELF\)](#), [attributes\(5\)](#), [ddi\\_modopen\(9F\)](#)  
『リンカーとライブラリ』

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | encrypt, decrypt – ファイルの暗号化または復号化                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 形式   | <pre> /usr/bin/encrypt -l  /usr/bin/encrypt -a algorithm [-v]     [-k key_file   -K key_label [-T token_spec]]     [-i input_file] [-o output_file]  /usr/bin/decrypt -l  /usr/bin/decrypt -a algorithm [-v]     [-k key_file   -K key_label [-T token_spec]]     [-i input_file] [-o output_file] </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 機能説明 | <p>このユーティリティは、指定されたアルゴリズムを使用して、指定されたファイルまたは stdin を暗号化または復号化します。出力ファイルが指定されていない場合、標準出力に出力されます。cryptoadm -i オプションと -o オプションが同じファイルを指定している場合は、暗号化された出力が同じファイルシステム内の一時作業ファイルに書き込まれてから、元のファイルを置き換えるために使用されます。</p> <p>復号化時に cryptoadm -i オプションと -o オプションが同じファイルを指定している場合は、平文によって暗号化テキストファイルが置き換えられます。</p> <p>encrypt の出力ファイルと decrypt の入力ファイルには、次の情報が含まれています。</p> <ul style="list-style-type: none"> <li>■ 出力形式バージョン番号 (ネットワークバイトオーダーの 4 バイト)。現在のバージョンは 1 です。</li> <li>■ 鍵生成関数で使用する繰り返し (ネットワークバイトオーダーの 4 バイト)。</li> <li>■ IV (ivlen バイト)[1]。iv データは、1 ブロックサイズと等しいランダムバイトにより生成されます。</li> <li>■ 鍵生成に使用される salt データ (16 バイト)。</li> <li>■ 暗号化テキストデータ。</li> </ul> <p>サポートしているオプションは、次のとおりです。</p> <p><b>-a algorithm</b>      暗号化または復号化プロセスで使用するアルゴリズムの名前を指定します。詳細については、「使用法」の「アルゴリズム」を参照してください。</p> <p><b>-i input_file</b>      入力ファイルを指定します。input_file が指定されていない場合、デフォルトは標準入力です。</p> <p><b>-k key_file</b>      暗号化アルゴリズムのキー値を含むファイルを指定します。PKCS#11 仕様で規定されているように、各アルゴリズムには固有のキー素材要件があります。-k が指定されていない場合、encrypt は <a href="#">getpassphrase(3C)</a> を使用してキー素材の入力を要求します。鍵ファイルのサイズにより鍵の長さが決まります。ま</p> |

た、可変長鍵を使用した暗号化用の 128 ビット長鍵の生成には、端末から設定されたパスフレーズが常に使用されます。

鍵ファイルの生成については、[pktool\(1\)](#) の `genkey` サブコマンドを参照してください。また、[dd\(1M\)](#) を使用することもできます。

- K *key\_label*      PKCS#11 トークン内で対称トークンキーのラベルを指定します。
- l                    システムで利用可能なアルゴリズムの一覧を表示します。この一覧は、暗号化フレームワークの構成によって変わります。キーサイズはビット単位で表示されます。
- o *output\_file*      出力ファイルを指定します。*output\_file* が指定されていない場合のデフォルトは標準出力です。ファイルへのリダイレクトなしに標準出力が使用される場合、暗号化または復号化された raw データにより端末エミュレーションが中断されるため、端末ウィンドウがハングアップしたように見える場合があります。これは、バイナリファイルを表示しているときに時折発生する状態によく似ています。
- T *token\_spec*      -K が指定されている場合に、デフォルトのソフトトークンオブジェクトストア以外の PKCS#11 トークンを指定します。  
  
*token\_spec* は次の形式になります。  
*token\_name* [:*manuf\_id* [:*serial\_no*]]  
 トークンラベルの末尾にスペースがある場合でも、このオプションではユーザーがスペースを入力する必要はありません。  
  
 トークン識別文字列はコロンで区切ります。いずれかの部分に文字としてコロン(:)を含める場合、そのコロンをバックスラッシュで(\)でエスケープする必要があります。コロン(:)がない場合、文字列全体が(最大 32 文字)がトークンラベルと見なされます。コロンが(:)1 つだけある場合、文字列はトークンラベルと作成者となります。
- v                    詳細な情報を表示します。Verbose を参照してください。

## 使用法

### アルゴリズム

-l オプションを指定すると、サポートされるアルゴリズムと、鍵の最小サイズおよび最大サイズが表示されます。これらのアルゴリズムは、暗号化フレームワークによって提供されます。サポートされる各アルゴリズムは PKCS #11 メカニズムの別名であり、特定のアルゴリズムタイプでもっともよく使用され、もっとも制限が少ないバージョンです。たとえば、`des` は `CKM_DES_CBC_PAD` の別名、`arcfour` は `CKM_RC4` の別名です。パディングや ECB のない派生アルゴリズムは、サポートされていません。



これらの別名は `-a` オプションとともに使用され、大文字と小文字が区別されません。

**パスフレーズ** `-k` オプションが暗号化および復号化タスクで使用されない場合、ユーザーはパスフレーズの入力を求められます。パスフレーズは、PKCS #5 で指定されている PBKDF2 アルゴリズムを使用して処理され、より安全な鍵に変換されます。

暗号化および復号化でパスフレーズが使用される場合、ユーザーが入力したパスフレーズが PBKDF2 アルゴリズムを使用して暗号化鍵に変換されます。詳細は、<http://www.rsasecurity.com> の「PKCS #5 v2.0.」を参照してください。

**詳細** コマンドに入力ファイルが指定されている場合、進行状況が画面に表示されます。進行状況の表示では、25% 完了するごとにパイプ記号 (|) が表示されます。標準入力からの入力の場合は、40KB 読み取られるたびにピリオド (.) が表示されます。両方の入力方法が完了すると、「Done」が出力されます。

**使用例** 例1 使用できるアルゴリズムの一覧表示

次の例では、使用できるアルゴリズムが一覧表示されます。

```
example$ encrypt -l
 Algorithm Keysize: Min Max

 aes 128 128
 arcfour 8 128
 des 64 64
 3des 192 192
```

例2 AESを使用して暗号化する

次の例では、AES を使用して暗号化し、暗号化鍵の入力を求めます。

```
example$ encrypt -a aes -i myfile.txt -o secretstuff
```

例3 鍵ファイルの作成後に AES を使用して暗号化する

次の例では、鍵ファイルの作成後に AES を使用して暗号化します。

```
example$ pktool genkey keystore=file keytype=aes keylen=128 \
 outkey=key
example$ encrypt -a aes -k key -i myfile.txt -o secretstuff
```

例4 インパイプを使用して暗号化されたテープバックアップを指定する

次の例では、インパイプを使用して、暗号化されたテープバックアップを提供します。

```
example$ ufsdump 0f - /var | encrypt -a arcfour \
 -k /etc/mykeys/backup.k | dd of=/dev/rmt/0
```

例5 インパイプを使用してテープバックアップを復元する  
次の例では、インパイプを使用してテープバックアップを復元します。

```
example$ decrypt -a arcfour -k /etc/mykeys/backup.k \
-i /dev/rmt/0 | ufsrestore xvf -
```

例6 3DES アルゴリズムを使用して入力ファイルを暗号化する  
次の例では、des3key ファイルに格納された 192 ビット鍵を使用して inputfile  
ファイルを暗号化します。

```
example$ encrypt -a 3des -k des3key -i inputfile -o outputfile
```

例7 DES トークンキーを使用して入力ファイルを暗号化する  
次の例では、ソフトトークンキーストア内の DES トークンキーを使用して入力  
ファイルを暗号化します。DES トークンキーは、[pktool\(1\)](#) で生成できます。

```
example$ encrypt -a des -K mydeskey \
-T "Sun Software PKCS#11 softtoken" -i inputfile \
-o outputfile
```

終了ステータス    次の終了ステータスが返されます。

- 0      正常終了。
- >0    エラーが発生した。

属性                    属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 确实             |

関連項目                [digest\(1\)](#), [pktool\(1\)](#), [mac\(1\)](#), [dd\(1M\)](#), [getpassphrase\(3C\)](#), [libpkcs11\(3LIB\)](#),  
[attributes\(5\)](#), [pkcs11\\_softtoken\(5\)](#)

『Oracle Solaris 11.1 の管理: セキュリティーサービス』

RSA PKCS#11 v2.11: <http://www.rsasecurity.com>

RSA PKCS#5 v2.0: <http://www.rsasecurity.com>

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | enhance – 拡張されたコマンド行編集機能                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式   | enhance <i>command</i> [ <i>argument</i> ]...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明 | <p>enhance プログラムは、ソースコードを利用できない他社のアプリケーションのユーザーに、拡張されたコマンド行編集機能を提供します。これは、アプリケーションと実端末間に仮想端末を配置することにより実現されます。これは、<b>tecla</b> コマンド行編集ライブラリを使って実端末からの入力を読み取り、その入力行をすぐに仮想端末経由でアプリケーションに転送します。アプリケーションからの出力はすべて、変更されずに実端末に戻されます。</p> <p>アプリケーションが出力の生成を 10 分の 1 秒より長く停止する場合、拡張プログラムは常に最新の未完了の出力行をプロンプトとして処理し、その後ユーザーが入力した未完了の入力行をすべて再表示します。ユーザーに気づかれない程度のわずかな遅延も、プログラムが正しく動作する上で必要ありません。これは単に、入力行がしばしば再表示されるために出力速度が低下することを避けるための最適化です。</p> <p><b>Tecla</b> ライブラリにより提供されるユーザーレベルのコマンド行編集機能の説明は、<a href="#">tecla(5)</a> のマニュアルページに記載されています。</p> |
| 不完全性 | <p>未解決の大きな問題の 1 つは、制御端末による入力のエコーを実行するかしないかを変更するアプリケーションの処理方法です。たとえば、<b>ftp</b> や <b>telnet</b> などのパスワードの入力を求めるプログラムは、制御端末に対して一時的にユーザーの入力をエコーしないように指示します。この要求は仮想端末のアプリケーション側に送られるため、拡張プログラムはこれが起きたことを識別できず、ユーザーがパスワードを入力している最中も、入力を制御端末にエコーし続けます。</p> <p>さらに、ホストアプリケーションの実行前に、拡張プログラムはまず仮想端末を <b>noecho</b> モードに設定するため、プログラムに送信されるものが重複してエコーされることはありません。<b>noecho</b> モードに切り替わったプログラムがその後、以前に有効だった端末モードを復元せずに、エコーを明示的に復元した場合は、新しい行を入力するたびに次の行に重複するコピーが表示されます。</p>                                                                             |
| ファイル | <p><b>/usr/lib/libtecla.so</b>      <b>tecla</b> ライブラリ</p> <p><b>~/.teclarc</b>                <b>tecla</b> 個人用カスタマイズファイル。</p>                                                                                                                                                                                                                                                                                                                                                                                                               |
| 属性   | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |

| 属性タイプ       | 属性値              |
|-------------|------------------|
| 使用条件        | library/libtecla |
| インタフェースの安定性 | 確実               |

関連項目 [libtecla\(3LIB\)](#), [attributes\(5\)](#), [tecla\(5\)](#)

|               |                                                                                                                                                                                                                                                                                                             |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前            | env - コマンド実行のための環境の設定                                                                                                                                                                                                                                                                                       |
| 形式            | <pre>/usr/bin/env [-i   -] [name=value]... [utility [arg... ]]</pre> <pre>/usr/xpg4/bin/env [-i   -] [name=value]... [utility [arg... ]]</pre>                                                                                                                                                              |
| 機能説明          | <p>env ユーティリティーは現在の環境を読み込み、引数の指定に従ってその値を変更し、<i>utility</i> 引数で指定されたユーティリティーを変更された状態の環境で実行します。</p> <p>任意指定の引数も、<i>utility</i> に渡されます。<i>utility</i> が省略された場合、変更後の環境の内容が標準出力に書き込まれます。この出力には、名前と値が <i>name=value</i> の形式で1行に1組ずつ書かれます</p>                                                                      |
| /usr/bin      | env が引数のあるコマンドを実行する場合、デフォルトのシェル /usr/bin/sh が使われます ( <a href="#">sh(1)</a> 参照)。                                                                                                                                                                                                                             |
| /usr/xpg4/bin | env が引数のあるコマンドを実行する場合、/usr/xpg4/bin/sh が使われます ( <a href="#">ksh88(1)</a> を参照)。                                                                                                                                                                                                                              |
| オプション         | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-i   -</b>    現在のシェルから引き継がれるはずの環境はすべて無視します。引数で指定された環境だけを用いて <i>utility</i> を実行します。</p>                                                                                                                                                                                 |
| オペランド         | <p>次のオペランドがサポートされています。</p> <p><b>name=value</b>    この形式でオペランドを指定すると、<i>name</i> が示す環境変数に <i>value</i> が示す値を設定して実行環境を変更し、さらに <i>utility</i> を実行する前に、受け継いだ環境にその指定を割り当てます。</p> <p><b>utility</b>    実行するユーティリティーの名前。シェルの特殊な組み込みユーティリティーを指定した場合の結果は定義されていません。</p> <p><b>arg</b>    実行するユーティリティーに引数として渡す文字列。</p> |
| 使用例           | <p>例1 新しいPATHの値でユーティリティーを呼び出す</p> <p>以下にユーティリティーを示します。</p> <pre>example% env -i PATH=/mybin mygrep xyz myfile</pre> <p>ここでは環境変数 PATH を設定し、これを唯一の環境設定として mygrep というユーティリティーを実行しています。つまり mygrep は、PATH が示す /mybin 中に存在していなければなりません。</p>                                                                       |
| 環境変数          | <p>env の実行に影響を与える次の環境変数についての詳細は、<a href="#">environ(5)</a> を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。</p> <p><b>PATH</b>    <i>utility</i> で指定したユーティリティーの位置を決定します。PATH が env への <i>name=value</i> オペランドとして指定された場合は、指定された値が <i>utility</i> で</p>                                                |

指定したユーティリティーの検索に使用されます。

終了ステータス *utility* で指定したユーティリティーが実行された場合、そのユーティリティーの終了ステータスが *env* の終了ステータスとなります。そうでない場合は、*env* は次の終了値を返します。

- 0            正常終了。
- 1-125      エラーが発生した。
- 126        *utility* は見つかったが呼び出すことができませんでした。
- 127        *utility* は見つかりませんでした。

属性                    属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/xopen/xcu4                       |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目                [ksh88\(1\)](#), [sh\(1\)](#), [exec\(2\)](#), [profile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | eqn, neqn, checkeq – 数学的記述のタイプセット                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式    | <pre>eqn [-d xy] [-f n] [-p n] [-s n] [file]... neqn [file]... checkeq [file]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明  | <p>eqn および neqn は、数式を記述するのに便利な言語プロセッサです。eqn は <b>troff(1)</b> 用のプリプロセッサで、troff の出力を印刷できる装置用に提供されています。neqn は <b>nroff(1)</b> 用のプリプロセッサで、端末での出力用に提供されています。通常この2つのコマンドは、以下の形式で指定します。</p> <pre>example% eqn file ...   troff example% neqn file ...   nroff</pre> <p>ファイル名を表す <i>file</i> 引数を省略すると、eqn または neqn は標準入力から読み込みます。数式の開始を示すには、行の先頭に <b>.EQ</b> を記述します。同様に数式の終了は、行の先頭に <b>.EN</b> を記述して表します。この2つの行は変換されないので、センタリングや番号付けなどを行うマクロパッケージ中に定義しておくこともできます。また、一対の文字を「区切り記号」として設定し、区切り記号に囲まれたテキストを eqn 入力として処理させることもできます。</p> <p>区切り記号や <b>.EQ/.EN</b> が存在しない、または対で指定されていない場合、checkeq はメッセージを出力します。</p>                                                                                                                                                                                 |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-dxy    コマンド行引数で設定される数式の区切り記号として、文字 <i>x</i> と <i>y</i> を設定します。ただしこの方法よりも、<b>.EQ</b> と <b>.EN</b> の間で <b>delim xy</b> を使って区切り記号を指定する方法がより一般的です。<i>x</i> と <i>y</i> には同じ文字を指定することも可能です。テキスト中に <b>delim off</b> と記述すると、区切り記号は有効でなくなります。区切り記号にも <b>.EQ</b> と <b>.EN</b> にも囲まれていないテキストは、すべてそのまま渡されます。</li> <li>-fn    ドキュメント全体を通じて使用するフォントとして <i>n</i> を指定します。このグローバルフォントの設定は、ドキュメントの本文中に <b>gfont n</b> 命令を指定して変更することもできます。<i>n</i> はフォント指定です。</li> <li>-pn    下付きおよび上付きの添字のサイズを、直前の文字サイズより <i>n</i> ポイントだけ小さくします。<b>-p</b> オプションを省略すると、添字のサイズは3ポイント小さくなります。</li> <li>-sn    ドキュメント全体を通じて使用する文字サイズとして <i>n</i> を指定します。このグローバルサイズの設定は、ドキュメントの本文中に <b>gsize n</b> 命令を指定して変更することもできます。<i>n</i> はポイントサイズです。</li> </ul> |

オペランド 次のオペランドがサポートされています。

*file* eqn または neqn によって処理される nroff のファイルまたは troff のファイル。

## Eqn 言語

この説明を nroff を使って端末画面に表示した場合、端末画面の制限から neqn による出力箇所は正確には表示できません。出力の正確な表示を確認するために、このページを印刷してご覧ください。

eqn 中のトークンは、中括弧、二重引用符、チルド、山型記号、スペース文字、タブ、または復帰改行文字で区切られます。中括弧 {} は、グループ分けに用いられます。一般的には、たとえば  $x$  のような 1 つの文字が記述できる箇所であれば、中括弧で囲んだ複雑な記述を代わりに指定できます。チルド (~) は出力中における 1 文字分のスペースを、山型記号 (^) は半文字分のスペースを表します。

下付きおよび上付きの添字:

これらは、キーワード sub と sup を使って生成できます。

$x \text{ sub } i$

の出力は次のようになります。  $x_i$

$a \text{ sub } i \text{ sup } 2$

の出力は次のようになります。  $a_i^2$

$e \text{ sup } \{x \text{ sup } 2 + y \text{ sup } 2\}$

の出力は次のようになります。  $e^{x^2+y^2}$

分数:

分数は、キーワード over で指定します。

$a \text{ over } b$

この出力は次のようになります。  $\frac{a}{b}$

平方根の式:

平方根の式は、キーワード sqrt で指定します。

$1 \text{ over sqrt } \{ax \text{ sup } 2 + bx + c\}$  この出力結果は、次のようになります。

$$\frac{1}{\sqrt{ax^2+bx+c}}$$

制限値:

キーワード from と to は、種々の指定における最小値と最大値を表します。

$\lim \text{ from } \{n \rightarrow \text{inf}\} \text{ sum from } 0 \text{ to } n \text{ } x \text{ sub } i$  の出力は次のようになります。

$$\lim_{n \rightarrow \infty} \sum_{i=0}^n x_i$$



括弧:

大括弧、中括弧などを適切な高さで出力するには、左括弧に `left` を、右括弧には `right` をそれぞれ使用します。

```
left [x sup 2 + y sup 2 over alpha right] ~=-1
```

の出力は次のようになります。  $\left[ x^2 + \frac{y^2}{\alpha} \right] = 1$

なお、`right` 文節は省略することができます。キーワード `left` と `right` の直後に指定できる文字は、大括弧、中括弧、縦棒、上端と下端を表す `c` と `f`、何もない旨を示す `"` (対になるべき括弧のうち右括弧だけを使う場合に便利) です。

分数を縦に重ねる:

分数を縦に重ねるには、`pile`、`lpile`、`cpile`、または `rpile` を使用します。

```
pile {a above b above c}
```

の出力は次のようになります。 
$$\begin{array}{c} a \\ b \\ c \end{array}$$

何重に積み重ねてもかまいません。文字を合わせる位置は、`lpile` は左詰め、`pile` と `cpile` はともにセンタリング (ただし縦方向の間隔が異なる)、そして `rpile` は右詰めとなります。

行列:

行列は `matrix` というキーワードで生成されます。

```
matrix { lcol { x sub i above y sub 2 } ccol { 1 above 2 } }
 x_i 1
```

の出力は次のようになります。  $y_2 \begin{array}{c} x_i \\ 1 \end{array}$

カラムを右詰めにするには `rcol` を使用します。

文字の上下に付加する記号:

発音符記号のように文字の上下に付加する記号

は、`dot`、`dotdot`、`hat`、`tilde`、`bar`、`vec`、`dyad`、`under` を使って指定できます。

```
x dot = f(t) bar
```

出力は次のようになります。  $\dot{x} = \overline{f(t)}$

```
y dotdot bar ~=- n under
```

出力は次のようになります。  $\ddot{y} = \underline{n}$

```
x vec ~=- y dyad
```

出力は次のようになります。  $\vec{x} = \overleftrightarrow{y}$

文字のサイズとフォント:  
文字のサイズやフォントの変更は、size *n* または size  $\pm n$ 、roman、italic、bold、font *n* で指定します。ドキュメント全体を通じてグローバルに使用する文字サイズとフォントは、gsize *n* と gfont *n* をドキュメント中に指定するか、またはコマンド行引数の -*sn* と -*fn* を使って変更できます。

表示引数の位置:  
一連の表示引数の位置をそろえることもできます。先頭の数式において、そろえたい表示引数の直前に mark と記述します。さらに後続の数式において、それと合わせたい表示引数の直前に lineup と記述します。

短縮形:  
入力の短縮形を定義したり既存のキーワードを再定義するには、define を使用します。次に例を示します。

define *thing*%*replacement*%

これにより *thing* というトークンが新たに定義され、その後このトークンが現れるたびに *replacement* に置き換えられます。なお%の位置には、任意の文字(ただし *replacement* に含まれていないもの)を指定できます。

キーワードと短縮形:  
sum int inf のようなキーワード、および >= → や != のような短縮形も処理されます。

ギリシャ文字:  
ギリシャ文字は alpha または GAMMA のように、大文字・小文字のうち希望する方のつづりで出力できます。

数学用語:  
sin、cos、log のような数学用語は自動的にローマン字体で出力されます。

\(bu(●) のような4文字からなる troff(1) のエスケープは、どこでも使用できます。二重引用符に囲まれた文字列". ."は、そのまま渡されます。これによりキーワードをテキストとして入力でき、また(他の方法が使えないとき) troff との通信用に使うことができます。

属性

属性についての詳細は、マニュアルページの attributes(5) を参照してください。

| 属性タイプ | 属性値           |
|-------|---------------|
| 使用条件  | text/doctools |

関連項目

nroff(1),tbl(1),troff(1),attributes(5),ms(5)

使用上の留意点

数字や括弧をボードで出力したい場合、bold "12.3" のように引用符で囲んでください。

450

マニュアルページセクション 1: ユーザーコマンド • 最終更新日 2002 年 7 月 12 日

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | error - ソース行の右側へのコンパイラエラーメッセージの挿入                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 形式    | error [-n] [-q] [-s] [-v] [-t <i>suffixlist</i> ] [-I <i>ignorefile</i> ] [ <i>filename</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明  | <p>error は、コンパイラや言語プロセッサにより生成されたエラーメッセージを解析します。error は、従来の読みづらい省略形だらけのエラーメッセージを解析し、エラーメッセージとソースコードを同時に表示できるようにします。</p> <p>error は、指定されたファイル (<i>filename</i>) または標準入力からエラーメッセージを読み取り、次の処理を行います。</p> <ul style="list-style-type: none"> <li>■ 各エラーメッセージを生成した言語プロセッサを判別します。</li> <li>■ エラー行を含むファイルの名前と行番号を判別します。</li> <li>■ ソースファイル内でエラー行の直前にエラーメッセージを挿入します。</li> </ul> <p>言語プロセッサや内容を特定できなかったエラーメッセージは、ファイルに挿入されず標準出力に送られます。error がソースファイルを処理するのは、すべての入力を読み込んだ後だけです。</p> <p>error は、その標準入力エラーメッセージの送信元とパイプを介して接続されている状態で実行するように設計されています。言語プロセッサの中には、エラーメッセージを独自の標準エラーファイルに格納するものも、標準出力に送信するものがあります。そのため、両方のエラーソースがまとめて error にパイプで連結されているべきです。たとえば、csh 構文を使用している場合、次のコマンドは、lint の実行時に <a href="#">make(1S)</a> が実行するすべてのプログラムによって生成されたすべてのエラーメッセージを解析します。</p> <pre>example% make -s lint   &amp; error -q -v</pre> <p>error は、<a href="#">as(1)</a>、<a href="#">cpp(1)</a>、<a href="#">ld(1)</a>、<a href="#">make(1S)</a> などのコンパイラが生成したエラーメッセージを識別できます。Pascal 以外のすべての言語において、エラーメッセージは 1 行に制限されています。エラーメッセージの中には、複数のファイル内の複数の行を示すものもあります。この場合、error はエラーメッセージを複製し、該当するすべての場所にエラーメッセージを挿入します。</p> |
| オプション | <p>-n            ファイルを処理しません。すべてのエラーメッセージは標準出力に送られます。</p> <p>-q            error はファイルを処理するかどうかを尋ねます。継続するには、質問に対して y または n で答えます。-q オプションを指定しない場合、すべての参照されるファイル (破棄されるエラーメッセージを参照するファイルを除く) が処理されます。</p> <p>-s            エラーの分類に従った統計を出力します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

- v** すべてのファイルを処理した後で、処理したすべてのファイルを **vi** で開き、最初に処理したファイル内の最初のエラーの位置を示します。**vi(1)** が見つからない場合、**ex(1)** または **ed(1)** を標準の場所から実行しようとしています。
- t suffixlist** 後続の引数を接尾辞リストとして扱います。接尾辞リストに現れない接尾辞を持つファイルは処理されません。接尾辞リストはドットで区切られたリストで、\*ワイルドカードも使用できます。たとえば、接尾辞リスト
- ```
.c.y.f*.h
```
- を使用すると、**error** は **.c**、**.y**、**.f***、**.h** で終了するファイルを処理します。

error は割り込みと終了を捕捉し、適切に終了します。

使用例

例1 **error** コマンドの使用

次の C シェル (**/usr/bin/csh**) の例では、**error** は FORTRAN コンパイラから入力を受け取ります。

```
example% f77 -c any.f |& error options
```

K シェル (**/usr/bin/ksh**) を使用した場合の例は、次のとおりです。

```
example% f77 -c any.f 2>&1 | error options
```

使用法

error は、次の 6 つの分類基準に従ってエラーメッセージを処理します。

- 同期** 言語プロセッサの中には、現在処理中のファイルを示す簡単なエラーメッセージを生成するものがあります。言語プロセッサが各エラーメッセージ中にファイル名を含めない場合、**error** はこのような簡単なエラーメッセージを使用してファイル名を判別します。このような同期メッセージは、**error** により完全に取り除かれます。
- 破棄** **lint** からのエラーメッセージの中で、2 つの **lint** ライブラリ、**/usr/lib/lint/llib-lc** と **/usr/lib/lint/llib-port** のいずれかに関連するメッセージは破棄され、間違っこれらのライブラリが処理されないようにします。このようなメッセージも **error** により完全に取り除かれます。
- 無視** **lint** からのエラーメッセージの中で、あまり大切な診断メッセージを生成することがわかっている関数を示すメッセージは無視することができます。無視されたエラーメッセージはソースファイルには挿入されず、標準出力に書き込まれます。無視する関数名は、ユーザーのホームディレクトリにある **.errorrc** ファイル、または **-I**

オプションで指定したファイルから取られます。このファイルが存在しない場合、エラーメッセージは無視されません。このファイルが存在するときは、無視する関数名はそのファイル内で1行に1つずつ記述されていなければなりません。

- ファイル固有でない 判別できないエラーメッセージは1つのグループにまとめられ、ファイルが処理される前に標準出力に書き込まれます。このようなエラーメッセージは、ソースファイルには挿入されません。
- ファイルに固有 特定のファイルは示されているが特定の行が示されていないエラーメッセージは、そのファイルが処理されるときに標準出力に書き込まれます。
- 本当のエラー 判別可能なエラーメッセージは、そのエラーメッセージが示すファイルに挿入される候補となります。

ソースファイルに挿入されるのは、本当のエラーメッセージだけです。他のエラーメッセージは、`error`により完全に取り除かれるか、標準出力に書き込まれます。`error`はソースファイル内で、エラーメッセージの行番号が示す行の前にエラーメッセージを挿入します。各エラーメッセージはその言語におけるコメントとなります。内部的には、エラーの先頭には文字列`###`が付けられ、エラーの終わりには文字列`%%`が付けられます。これにより、エディタで作業するとき、エラーを簡単に検出し削除できます。さらに、各エラーメッセージには、そのエラーメッセージが示す行の行番号が入っています。記述規則に則ったソースプログラムであれば、エラーメッセージを削除しなくてもそのまま再コンパイルでき、エラーメッセージ自体が新たなエラーを引き起こすこともありません。CやPascalのように記述規則を持たない言語で書かれた、記述の整っていないソースプログラムでは、コメントを別のコメント内に挿入することは可能ですが、新たにコンパイルするときに問題が発生する原因となります。これを回避するには、コメントの終わりと同じ行に言語文がないように、ソースプログラムの記述方法を変更します。

- ファイル `~/ .errorrc` lintのエラーメッセージで無視する関数名
`/dev/tty` ユーザーの端末
- 属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities

- 関連項目 [as\(1\)](#), [cpp\(1\)](#), [csh\(1\)](#), [ed\(1\)](#), [ex\(1\)](#), [make\(1S\)](#), [ld\(1\)](#), [vi\(1\)](#), [attributes\(5\)](#)

使用上の留意点 `tty` デバイスは、ユーザー入力用に直接開いてください。

リンク付きのソースファイルは、リンクを1つだけ持つファイルのコピーを新たに作成します。

言語プロセッサのエラーメッセージの形式を変更すると、`error` がエラーメッセージを判別できなくなります。

`error` は純粋に機械的な処理を行うため、構文として重要でない1つのエラーによって発生した大量のエラーをフィルタすることはできません。このようなエラーは手動で破棄してください。

Pascal のエラーメッセージは実際にエラーが発生した行の後に置かれますが、`error` はメッセージを前に置きます。'|' マークによるエラー箇所の指定も `error` のメッセージではずれてしまいます。

`error` は、ある程度高速な CRT で動作するように設計されています。低速の端末では使いづらく、ハードコピー端末で使用するようには設計されていません。

名前

ex – text editor

形式

```

/usr/bin/ex [-| -s] [-l] [-L] [-R] [-r [file]] [-t tag]
            [-v] [-V] [-wn] [+command | -c command] file...

/usr/xpg4/bin/ex [-| -s] [-l] [-L] [-R] [-r [file]]
                [-t tag] [-v] [-V] [-wn]
                [+command | -c command] file...

/usr/xpg6/bin/ex [-| -s] [-l] [-L] [-R] [-r [file]]
                [-t tag] [-v] [-V] [-wn]
                [+command | -c command] file...

```

機能説明

The ex utility is the root of a family of editors: ex and vi. ex is a superset of [ed\(1\)](#), with the most notable extension being a display editing facility. Display based editing is the focus of vi.

If you have a CRT terminal, you can wish to use a display based editor; in this case see [vi\(1\)](#), which is a command which focuses on the display-editing portion of ex.

If you have used ed you find that, in addition to having all of the ed commands available, ex has a number of additional features useful on CRT terminals. Intelligent terminals and high speed terminals are very pleasant to use with vi. Generally, the ex editor uses far more of the capabilities of terminals than ed does, and uses the terminal capability data base (see [terminfo\(4\)](#)) and the type of the terminal you are using from the environment variable TERM to determine how to drive your terminal efficiently. The editor makes use of features such as insert and delete character and line in its `visual` command (which can be abbreviated vi) and which is the central mode of editing when using the vi command.

The ex utility contains a number of features for easily viewing the text of the file. The `z` command gives easy access to windows of text. Typing `^D` (CTRL-D) causes the editor to scroll a half-window of text and is more useful for quickly stepping through a file than just typing return. Of course, the screen-oriented `visual` mode gives constant access to editing context.

The ex utility gives you help when you make mistakes. The `undo` (u) command allows you to reverse any single change which goes astray. ex gives you a lot of feedback, normally printing changed lines, and indicates when more than a few lines are affected by a command so that it is easy to detect when a command has affected more lines than it should have.

The editor also normally prevents overwriting existing files, unless you edited them, so that you do not accidentally overwrite a file other than the one you are editing. If the system (or editor) crashes, or you accidentally hang up the telephone, you can use the editor `recover` command (or `-r file` option) to retrieve your work. This gets you back to within a few lines of where you left off.

The ex utility has several features for dealing with more than one file at a time. You can give it a list of files on the command line and use the `next` (n) command to deal with each in turn. The `next` command can also be given a list of file names, or a pattern as used by the shell to

specify a new set of files to be dealt with. In general, file names in the editor can be formed with full shell metasyntax. The metacharacter '%' is also available in forming file names and is replaced by the name of the current file.

The editor has a group of buffers whose names are the ASCII lower-case letters (a-z). You can place text in these named buffers where it is available to be inserted elsewhere in the file. The contents of these buffers remain available when you begin editing a new file using the `edit` (e) command.

There is a command `&` in `ex` which repeats the last `substitute` command. In addition, there is a confirmed substitute command. You give a range of substitutions to be done and the editor interactively asks whether each substitution is desired.

It is possible to ignore the case of letters in searches and substitutions. `ex` also allows regular expressions which match words to be constructed. This is convenient, for example, in searching for the word "edit" if your document also contains the word "editor."

`ex` has a set of options which you can set to tailor it to your liking. One option which is very useful is the `autoindent` option that allows the editor to supply leading white space to align text automatically. You can then use `^D` as a backtab and space or tab to move forward to align new code easily.

Miscellaneous useful features include an intelligent `join` (j) command that supplies white space between joined lines automatically, commands `<` and `>` which shift groups of lines, and the ability to filter portions of the buffer through commands such as `sort`.

オプション

The following options are supported:

<code>- -s</code>	Suppresses all interactive user feedback. This is useful when processing editor scripts.
<code>-l</code>	Sets up for editing LISP programs.
<code>-L</code>	Lists the name of all files saved as the result of an editor or system crash.
<code>-R</code>	Readonly mode. The <code>readonly</code> flag is set, preventing accidental overwriting of the file.
<code>-r file</code>	Edits <i>file</i> after an editor or system crash. (Recovers the version of <i>file</i> that was in the buffer when the crash occurred.)
<code>-t tag</code>	Edits the file containing the <i>tag</i> and positions the editor at its definition. It is an error to specify more than one <code>-t</code> option.
<code>-v</code>	Starts up in display editing state, using <code>vi</code> . You can achieve the same effect by typing the <code>vi</code> command itself.

- V

Verbose. When ex commands are read by means of standard input, the input is echoed to standard error. This can be useful when processing ex commands within shell scripts.
- wn

Sets the default window size to *n*. This is useful when using the editor over a slow speed line.
- +*command* | -c *command*

Begins editing by executing the specified editor *command* (usually a search or positioning command).
- /usr/xpg4/bin/ex, /usr/xpg6/bin/ex

If both the -t *tag* and the -c *command* options are given, the -t *tag* is processed first. That is, the file containing the tag is selected by -t and then the command is executed.

オペランド The following operand is supported:

file A path name of a file to be edited.

使用法 This section defines the ex states, commands, initializing options, and scanning pattern formations.

- ex States

Command

Normal and initial state. Input prompted for by “:”. The line kill character cancels a partial command.
- Insert

Entered by a, i, or c. Arbitrary text can be entered. Insert state normally is terminated by a line having only “.” on it, or, abnormally, with an interrupt.
- Visual

Entered by typing vi. Terminated by typing Q or ^\ (Control-\\).

ex Command Names and Abbreviations	Command Name	Abbreviation	Command Name	Abbreviation	Command Name	Abbreviation
	abbrev	ab	map		set	se
	append	a	mark	ma	shell	sh
	args	ar	move	m	source	so
	change	c	next	n	substitute	s
	copy	co	number	nu	unabbrev	unab
	delete	d	preserve	pre	undo	u
	edit	e	print	p	unmap	unm
	file	f	put	pu	version	ve
	global	g	quit	q	visual	vi

	insert w	i	read r	write	
	join	j	recover	rec	xit x
	list	l	rewind	rew	yank ya
Join Command Arguments	Join	[range] j[oin][!] [count] [flags]			
	If count is specified:				
	/usr/bin/ex, /usr/xpg6/bin/ex				
	If no address is specified, the join command behaves as if 2addr were the current line and the current line plus count (. , . + count). If one address is specified, the join command behaves as if 2addr were the specified address and the specified address plus count (addr , addr + count).				
	/usr/xpg4/bin/ex				
	If no address is specified, the join command behaves as if 2addr were the current line and the current line plus count - 1 (. , . + count - 1). If one address is specified, the join command behaves as if 2addr were the specified address and the specified address plus count - 1 (addr , addr + count - 1).				
	/usr/bin/ex, /usr/xpg4/bin/ex, /usr/xpg6/bin/ex				
	If two or more addresses are specified, the join command behaves as if an additional address, equal to the last address plus count - 1 (addr1 , . . . , lastaddr , lastaddr + count - 1), was specified. If this results in a second address greater than the last line of the edit buffer, it is corrected to be equal to the last line of the edit buffer.				
	If no count is specified:				
	/usr/bin/ex, /usr/xpg4/bin/ex, /usr/xpg6/bin/ex				
	If no address is specified, the join command behaves as if 2addr were the current line and the next line (. , . +1). If one address is specified, the join command behaves as if 2addr were the specified address and the next line (addr , addr +1).				
Additional ex Command Arguments	/usr/bin/ex, /usr/xpg6/bin/ex	For the following ex commands, if count is specified, it is equivalent to specifying an additional address to the command. The additional address is equal to the last address specified to the command (either explicitly or by default) plus count - 1. If this results in an address greater than the last line of the edit buffer, it is corrected to equal the last line of the edit buffer.			
	/usr/xpg4/bin/ex	For the following ex commands, if both a count and a range are specified for a command that uses them, the number of lines affected is taken from the count value rather than the range. The starting line for the			

command is taken to be the first line addressed by the range.

Abbreviate	ab[brev] word rhs
Append	[line]a[ppend][!]
Arguments	ar[gs]
Change	[range] c[hange][!] [count]
Change Directory	chd[ir][!] [directory]; cd[!] [directory]
Copy	[range] co[py] line [flags]; [range] t line [flags]
Delete	[range] d[etele] [buffer] [count] [flags]
Edit	e[dit][!] [+line][file]; ex[!] [+line] [file]
File	f[ile] [file]
Global	[range] g[lobal] /pattern/ [commands]; [range] v /pattern/ [commands]
Insert	[line] i[nsert][!]
List	[range] l[ist] [count] [flags]
Map	map[!] [x rhs]
Mark	[line] ma[rk] x; [line] k x
Move	[range] m[ove] line
Next	n[ext][!] [file ...]
Open	[line] o[pen] /pattern/ [flags]
Preserve	pre[serve]
Put	[line] pu[t] [buffer]
Quit	q[uit][!]
Read	[line] r[ead][!] [file]
Recover	rec[over] file
Rewind	rew[ind][!]
Set	se[t] [option[=[value]]...] [nooption...] [option?...] [all]
Shell	sh[ell]
Source	so[urce] file
Suspend	su[spend][!]; st[op][!]

Tag	ta[g][!] tagstring
Unabbreviate	una[bbrev] word
Undo	u[ndo]
Unmap	unm[ap][!] x
Visual	[line] v[isual] [type] [count] [flags]
Write	[range] w[rite][!] [>>] [file]; [range] w[rite][!] [file]; [range] wq[!] [>>] [file]
Write and Exit	[range] x[it][!] [file]
Yank	[range] ya[nk] [buffer] [count]
Adjust Window	[line] z [type] [count] [flags]
Escape	! command [range]! command
Scroll	EOF
Write Line Number	[line] = [flags]
Execute	@ buffer; * buffer

/usr/bin/ex, /usr/xpg4/bin/ex, /usr/xpg6/bin/ex

For the following ex commands, if *count* is specified, it is equivalent to specifying an additional address to the command. The additional address is equal to the last address specified to the command (either explicitly or by default) plus *count*-1. If this results in an address greater than the last line of the edit buffer, it is corrected to equal the last line of the edit buffer.

Number	[range] nu[mber] [count] [flags]; [range] # [count] [flags]
Print	[range] p[rint] [count] [flags]
Substitute	[range] s[ubstitute] [/pattern/repl/[options] [count] [flags]]
Shift Left	[range] < [count] [flags]
Shift Right	[range] > [count] [flags]
Resubstitute	[range] & [options] [count] [flags]; [range] s[ubstitute] [options] [count] [flags]; [range] ~ [options] [count] [flags]

ex Commands	&	resubst
	CR	print next
	>	rshift

	<	lshift
	^D	scroll
	z	window
	!	shell escape
ex Command Addresses	<i>n</i>	line <i>n</i>
	.	current
	\$	last
	+	next
	-	previous
	<i>+n</i>	<i>n</i> forward
	%	1,\$
	<i>/pat</i>	next with <i>pat</i>
	<i>?pat</i>	previous with <i>pat</i>
	<i>x-n</i>	<i>n</i> before <i>x</i>
	<i>x,y</i>	<i>x</i> through <i>y</i>
	' <i>x</i>	marked with <i>x</i>
	"	previous context
Initializing Options	EXINIT	place set's here in environment variable
	\$HOME/.exrc	editor initialization file
	./exrc	editor initialization file
	set <i>x</i>	enable option <i>x</i>
	set no <i>x</i>	disable option <i>x</i>
	set <i>x=val</i>	give value <i>val</i> to option <i>x</i>
	set	show changed options
	set all	show all options
	set <i>x?</i>	show value of option <i>x</i>

Useful Options and Abbreviations	autoindent	ai	supply indent
	autowrite	aw	write before changing files
	directory		pathname of directory for temporary work files
	exrc	ex	allow vi/ex to read the .exrc in the current directory. This option is set in the EXINIT shell variable or in the .exrc file in the \$HOME directory.
	ignorecase	ic	ignore case of letters in scanning
	list		print ^I for tab, \$ at end
	magic		treat . [* special in patterns
	modelines		first five lines and last five lines executed as vi/ex commands if they are of the form ex:command: or vi:command:
	number	nu	number lines
	paragraphs	para	macro names that start paragraphs
	redraw		simulate smart terminal
	report		informs you if the number of lines modified by the last command is greater than the value of the report variable
	scroll		command mode lines
	sections	sect	macro names that start sections
	shiftwidth	sw	for <>, and input ^D
	showmatch	sm	to) and } as typed
	showmode	smd	show insert mode in vi
	slowopen	slow	stop updates during insert
	term		specifies to vi the type of terminal being used (the default is the value of the environment variable TERM)
	window		visual mode lines
	wrapmargin	wm	automatic line splitting
	wrapscreen	ws	search around end (or beginning) of buffer
Scanning Pattern Formation	^		beginning of line
	\$		end of line
	.		any character

\<	beginning of word
\>	end of word
[<i>str</i>]	any character in <i>str</i>
[^ <i>str</i>]	any character not in <i>str</i>
[<i>xy</i>]	any character between <i>x</i> and <i>y</i>
*	any number of preceding characters

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of ex: HOME, LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_MESSAGES, NLSPATH, PATH, SHELL, and TERM.

COLUMNS	Override the system-selected horizontal screen size.
EXINIT	Determine a list of ex commands that are executed on editor start-up, before reading the first file. The list can contain multiple commands by separating them using a vertical-line () character.
LINES	Override the system-selected vertical screen size, used as the number of lines in a screenful and the vertical screen size in visual mode.

終了ステータス The following exit values are returned:

0	Successful completion.
>0	An error occurred.

ファイル

/var/tmp/Exnnnnnn	editor temporary
/var/tmp/Rxnnnnnn	named buffer temporary
/usr/lib/expreserve	preserve command
/usr/lib/exrecover	recover command
/usr/lib/exstrings	error messages
/usr/share/lib/terminfo/*	describes capabilities of terminals
/var/preserve/login	preservation directory (where login is the user's login)
\$HOME/.exrc	editor startup file
./ .exrc	editor startup file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/ex

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
CSI	Enabled

/usr/xpg4/bin/ex

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/xopen/xcu4
CSI	Enabled
Interface Stability	Committed
Standard	See standards(5) .

/usr/xpg6/bin/ex

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/xopen/xcu6
CSI	Enabled
Interface Stability	Standard

関連項目 [ed\(1\)](#), [edit\(1\)](#), [grep\(1\)](#), [sed\(1\)](#), [sort\(1\)](#), [vi\(1\)](#), [curses\(3CURSES\)](#), [term\(4\)](#), [terminfo\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

[『Solaris Advanced User’s Guide』](#)

Author The vi and ex utilities are based on software developed by The University of California, Berkeley California, Computer Science Division, Department of Electrical Engineering and Computer Science.

注意事項 Several options, although they continue to be supported, have been replaced in the documentation by options that follow the Command Syntax Standard (see [Intro\(1\)](#)). The `–` option has been replaced by `-s`, a `-r` option that is not followed with an option-argument has been replaced by `-L`, and `+command` has been replaced by `-c command`.

The message file too large to recover with `-r` option, which is seen when a file is loaded, indicates that the file can be edited and saved successfully, but if the editing session is lost, recovery of the file with the `-r` option is not possible.

The `z` command prints the number of logical rather than physical lines. More than a screen full of output can result if long lines are present.

File input/output errors do not print a name if the command line `-s` option is used.

The editing environment defaults to certain configuration options. When an editing session is initiated, `ex` attempts to read the `EXINIT` environment variable. If it exists, the editor uses the values defined in `EXINIT`, otherwise the values set in `$HOME/.exrc` are used. If `$HOME/.exrc` does not exist, the default values are used.

To use a copy of `.exrc` located in the current directory other than `$HOME`, set the `exrc` option in `EXINIT` or `$HOME/.exrc`. Options set in `EXINIT` can be turned off in a local `.exrc` only if `exrc` is set in `EXINIT` or `$HOME/.exrc`. In order to be used, `.exrc` in `$HOME` or the current directory must fulfill these conditions:

- It must exist.
- It must be owned by the same userid as the real userid of the process, or the process has appropriate privileges.
- It is not writable by anyone other than the owner.

There is no easy way to do a single scan ignoring case.

The editor does not warn if text is placed in named buffers and not used before exiting the editor.

Null characters are discarded in input files and cannot appear in resultant files.

名前 exec, eval, source – 他のコマンドを実行するためのシェル組み込み関数

形式

sh exec [*argument*]...
 eval [*argument*]...
csh exec *command*
 eval *argument*...
 source [-h] *name*
ksh88 *exec [*argument*]...
 *eval [*argument*]...
ksh +exec [-c] [-a *name*] [*command* [*argument* ...]]
 +eval [*argument*]...

機能説明

sh exec コマンドはこのシェルの代わりに、*argument* で指定されたコマンドを、新規プロセスは生成せずに実行します。入出力引数が指定可能で、その他の引数が指定されない場合は、これによってシェルの入出力が変更されます。

eval の組み込みの *argument* をシェルへの入力として読み取り、生成されるコマンドを実行します。

csh exec は現在のシェルの代わりに *command* を実行します。シェルは終了します。

eval は引数をシェルへの入力として読み取り、生成されるコマンドを実行します。通常この指定は、コマンドまたは変数置換の結果として生成されたコマンドを実行するために使用します。

source は *name* からコマンドを読み取ります。source コマンドはネストできますが、あまり深くネストするとシェルのファイル記述子が不足する可能性があります。ソースファイル中のエラーは、それがいかなるレベルであろうと、入れ子にされたすべての source コマンドを終了させます。

-h *name* が示す、履歴のリスト上のファイルからコマンドを持ってきますが、実行はしません。

ksh88 exec 組み込み関数を使用して *arg* を指定すると、このシェルの代わりに引数で指定されたコマンドを、新規プロセスは生成せずに実行します。入出力引数が指定可能で、現在のプロセスに影響を及ぼす場合があります。引数を指定しない場合は、ファイル記述子が入出力リダイレクションリストの指定どおりに変更されることになります。この場合、この機能を使って開かれた、ファイル記述子番号が2より大きいファイルは、別のプログラムを起動すると閉じられます。

`eval` に続く引数をシェルへの入力として読み取り、生成されるコマンドを実行します。

このマニュアルページでは、1つまたは2つの* (アスタリスク) が先頭に付加されている [ksh88\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

ksh

`exec` は特殊な組み込みコマンドであり、これを使用するとファイル記述子を操作したり、現在のシェルを新しいコマンドに置き換えたりできます。

`command` を指定した場合、`command` を実行してその完了を待機するのではなく、現在のシェルプロセスが `command` によって置き換えられます。シェルでは、可能な場合は常に `exec` メカニズムが内部で暗黙的に使用されるため、パフォーマンスを向上させるために `exec` を使用する必要はありません。

オペランドを指定しない場合、`exec` を使用して、ファイルを開いたり閉じたりできます。また、すべてのコマンドで使用できる標準のリダイレクトメカニズムを使用して、現在のシェル環境で 0 から 9 までのファイル記述子を操作することもできます。別のプログラムが呼び出されたときにファイルが閉じられるように、この方法で開かれた、ファイル記述子番号が 3 以上のファイルには `close-on-exec` フラグが設定されます。

`exec` は特殊なコマンドであるため、どのようなエラーが発生しても、`exec` を呼び出したスクリプトは終了します。これは、`command` ユーティリティーから `exec` を呼び出すことで防止できます。

`exec` を制限付きシェルから呼び出して、ファイルを作成したり、書き込みまたは追加を行うためにファイルを開いたりすることはできません。

`eval` は、スペースで区切られた複数の *argument* を連結してコマンドを生成する、シェルの特殊組み込みコマンドです。生成される文字列は、シェルへの入力と見なされ、現在の環境で評価されます。コマンドワードは、2回 (*argument* を生成するために1回、および生成されたコマンドをシェルが実行するときにもう1回) 展開されます。*argument* が指定されていなくても、エラーにはなりません。

このマニュアルページでは、1つまたは2つの+記号が先頭に付加されている `ksh` コマンドは特殊組み込みコマンドであり、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++ から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

オプション

ksh `exec` では次のオプションがサポートされています。

- a *name* `argv[0]` が、コマンドの *name* に設定されます。
- c 現在の `exec` コマンドの一部である変数代入を除き、実行前にすべての環境変数をクリアします。

終了ステータス

ksh88 `exec` により、次の終了値が返されます。

- 0 正常終了。
- 1-125 リダイレクトエラーが発生しました。
- 127 *command* が見つかりませんでした。
- 126 *command* は見つかりましたが、実行可能なユーティリティではありません。

ksh `exec` により、次の終了値が返されます。*command* を指定した場合、`exec` は何も返しません。

- 0 正常終了。すべての入出力のリダイレクトが成功しました。
- >0 エラーが発生した。

`eval` により、次の終了値が返されます。

argument を指定しない場合、終了ステータスは0となります。それ以外の場合は、*argument* オペランドによって定義されているコマンドの終了ステータスとなります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

名前 exit, return, goto – シェルの連続した処理を分岐して実行するためのシェル組み込み関数

形式

sh	exit [<i>n</i>] return [<i>n</i>]
csH	exit [(<i>expr</i>)] goto <i>label</i>
ksh88	*exit [<i>n</i>] *return [<i>n</i>]
ksh	+exit [<i>n</i>] +return [<i>n</i>]

機能説明

sh exit はシェルまたはシェルスクリプトを *n* で指定した終了ステータスで終了させます。*n* を省略すると、最後に実行されたコマンドの終了ステータスがシェルの終了ステータスになります (EOF によっても、シェルは終了します)。

return は関数を、*n* が示す戻り値で終了させます。*n* を省略すると、リターンステータスは最後に実行されたコマンドのリターンステータスになります。

csH exit はシェルまたはシェルスクリプトを終了させ、ステータス変数の値または式 *expr* で指定された値が返されます。

goto 組み込み関数は *label* をコマンド中で検索の引数として指定します。シェルは可能なかぎり入力をさかのぼり、*label:* という形式の行を探します。*label:* の前にはスペース文字またはタブ文字がある可能性もあります。指定された行の次から実行が再開します。while または for 組み込みコマンドと、対応する end との間にあるラベルへジャンプするとエラーになります。

ksh88 exit はシェルまたはシェルスクリプトを *n* で指定した終了ステータスで終了させます。具体的には、指定した値の最下位 8 ビットが終了ステータスの値となります。*n* を省略すると、最後に実行されたコマンドの終了ステータスがシェルの終了ステータスになります。トラップ実行中に exit が発生した場合、ここで言う最後に実行されたコマンドとは、トラップ呼び出し直前に実行されたコマンドを指します。なお、ignoreeof オプション (後述の set を参照) が有効になっているシェルを除き、ファイルの終わりを検出した場合もシェルが終了します。

return はシェル関数または '. ' スクリプトを、*n* で指定された戻りステータスで呼び出し側スクリプトに戻します。具体的には、指定した値の最下位 8 ビットが終了ステータスの値となります。*n* を省略すると、戻りステータスは最後に実行されたコマンドの戻りステータスになります。return を関数や '. ' スクリプト実行中以外で起動すると、結果は exit と同一になります。

このマニュアルページでは、1つまたは2つの*(アスタリスク)が先頭に付加されている `ksh88(1)` コマンドは、次のような特殊な扱いを受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、**を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

ksh

`exit` は、シェルの特殊組み込みコマンドであり、`exit` を呼び出したシェルを終了させます。EXITトラップが設定されている場合は、シェルが終了する前に、そのトラップが呼び出されます。

`n` が指定されている場合、終了ステータスを設定するために使用されます。

`return` は、シェルの特殊組み込みコマンドであり、`return` を呼び出した関数またはドットスクリプトを終了させます。`return` を関数またはドットスクリプトの外部で呼び出した場合、`return` は `exit` と同等になります。

`return` を、`function` 予約語構文を使用して定義されている関数の内部で呼び出した場合、関数が戻る前に、呼び出し側のコンテキストで、関数内に設定されているすべての EXIT トラップが呼び出されます。

`n` が指定されている場合、終了ステータスを設定するために使用されます。

このマニュアルページでは、1つまたは2つの+記号が先頭に付加されている `ksh` コマンドは特殊組み込みコマンドであり、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

終了ステータス

ksh

`exit` で `n` が指定されている場合、終了ステータスは `n` の値の最下位 8 ビットとなります。それ以外の場合、終了ステータスは、前のコマンドの終了ステータスとなります。トラップの内部で呼び出された場合、前のコマンドとは、トラップを呼び出したコマンドを意味します。

return で n が指定されている場合、終了ステータスは n の値の最下位 8 ビットとなります。それ以外の場合、終了ステータスは、前のコマンドの終了ステータスとなります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [break\(1\)](#), [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

名前	expand, unexpand – タブ文字をスペース文字に展開する、またはその反対
形式	<pre>expand [-t <i>tablist</i>] [<i>file</i>]...</pre> <pre>expand [-<i>tabstop</i>] [-<i>tab1</i>, <i>tab2</i>,... , <i>tabn</i>] [<i>file</i>]...</pre> <pre>unexpand [-a] [-t <i>tablist</i>] [<i>file</i>]...</pre>
機能説明	expand は 1 つ以上の <i>file</i> (または標準入力) のタブ文字をスペース文字に展開して標準出力へコピーします。バックスペース文字は出力中に保存され、タブのカラム幅の計算の際にカラム幅を 1 減算します。expand はタブ文字を含む文字ファイルの前処理 (ソートをする前や、特定のカラムを探す前など) を行う際に役に立ちます。 unexpand は 1 つ以上の <i>file</i> (または標準入力) を、タブ文字を復活させて標準出力へコピーします。オプションの指定がないときは、行頭のスペース文字とタブ文字だけがタブ列に置き換えられます。-a オプションを指定するとこの指定は無効となります(「オプション」の項参照)。
オプション	expand では次のオプションがサポートされています。 -t <i>tablist</i> タブの位置を指定します。引数 <i>tablist</i> は、1 つまたは複数の 10 進整数からなります。複数個指定する場合には、昇順に並べて空白文字またはコンマで区切らなければなりません。1 つの整数だけを指定すると、そのカラム数ごとにタブが設定されます。デフォルトでは 8 カラムおきです。複数の整数を指定すると、それらのカラム位置にタブが設定されます。 指定する各カラム位置 (<i>N</i>) は、ゼロより大きい整数でなければなりません。またカラム位置は昇順に指定する必要があります。行を出力する際、カラム位置 <i>N</i> にタブを進めるということは、次の文字が <i>N</i>+1 カラムに出力されることとなります。 複数のタブ位置が指定され、その最後のタブ位置を超えた地点でタブ文字の出力を処理する必要がある場合、expand はそのタブを 1 つのスペース文字に置き換えて出力します。 -tabstop 1 つの数を指定し、その個数分のスペース文字ごとに、タブを設定します。省略時の値は 8 です。 -tab1, tab2,...,tabn 引数で指定された位置にタブ文字を設定します。- unexpand では次のオプションがサポートされています。 -a 置き換えていく際に 2 つ以上スペース文字が連続していたらタブ文字を挿入します。より小さな出力ファイルを生成します。 -t <i>tablist</i> タブの位置を指定します。引数 <i>tablist</i> は、1 つまたは複数の 10 進整数からなります。複数個指定する場合には、昇順に並べて空白文字またはコンマで区切らなければなりません。1 つの整数だけを指定する

と、そのカラム数ごとにタブが設定されます。デフォルトでは8カラムおきです。複数の整数を指定すると、それらのカラム位置にタブが設定されます。指定する各カラム位置 (*N*) は、ゼロより大きい整数でなければなりません。またカラム位置は昇順に指定する必要があります。行を出力する際、カラム位置 *N* にタブを進めるということは、次の文字が *N*+1 カラムに出力されることになります。-t オプションを指定しない場合、デフォルトとして -t 8 を指定したことと同等となります。ただし、後述するように -a との関連については異なります。

複数のタブ位置が指定された場合、その最後のタブ位置を超えた地点ではスペース文字からタブ文字への文字変換は発生しません。

-t オプションを指定すると、-a オプションは意味を持たなくなり、タブ変換は先行する空白文字だけに制限されることはありません。

オペランド expand と unexpand では次のオペランドがサポートされています。

file 入力に用いるテキストファイルのパス名。

環境変数 expand と unexpand の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [tabs\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	exportfs – translates exportfs options to share/unshare commands
形式	/usr/sbin/exportfs [-aiuv] [-o <i>options</i>] [<i>pathname</i>]
機能説明	exportfs translates SunOS 4.x exportfs options to the corresponding share/unshare options and invokes share/unshare with the translated options. With no options or arguments, exportfs invokes share to print out the list of all currently shared NFS filesystems. exportfs is the BSD/Compatibility Package command of share(1M) and unshare(1M). Use share(1M)/ unshare(1M) whenever possible.
オプション	-a Invokes shareall(1M), or if -u is specified, invokes unshareall(1M). -i Ignore options in /etc/dfs/dfstab. -u Invokes unshare(1M) on <i>pathname</i>. -v Verbose. -o <i>options</i> Specify a comma-separated list of optional characteristics for the filesystems being exported. exportfs translates <i>options</i> to share-equivalent options. (see share(1M) for information about individual options).
属性	See attributes(5) for descriptions of the following attributes:
関連項目	share(1M) , shareall(1M) , unshare(1M) , unshareall(1M) , attributes(5)

名前 `expr` - 引数を式として評価する

形式
`/usr/bin/expr argument...`
`/usr/xpg4/bin/expr argument...`
`/usr/xpg6/bin/expr argument...`

機能説明

`/usr/bin/expr`,
`/usr/xpg4/bin/expr` `expr` ユーティリティーは式を評価して、その結果を標準出力に書き出します。文字 `0` はゼロ値として書き込まれ、何もない場合は `NULL` 文字列として書き込まれます。

`/usr/xpg6/bin/expr` `expr` ユーティリティーは式を評価して、その結果とその後に `NEWLINE` を標準出力に書き出します。`expr` 処理からの結果がない場合、`NEWLINE` が標準出力に書き込まれます。

オペランド `argument` オペランドは式として評価されます。式の項は空白文字で区切る必要があります。シェルに対する特殊文字は、エスケープする必要があります ([sh\(1\)](#) 参照)。空白文字またはその他の特殊文字を含む文字列は、引用符で囲む必要があります。式の長さは `LINE_MAX` (2048 文字) に制限されます。

演算子とキーワードを以下に示します。優先度の低いものから順に並べてあり、等しい優先度の演算子は、`{ }` 記号内にグループ分けされています。演算子はすべてその左に関与します。

`expr \| expr` 1 番目の `expr` が `NULL` または `0` のどちらでもない場合、1 番目の `expr` の評価を返します。それ以外の場合、2 番目の `expr` が `NULL` でなければ、2 番目の `expr` の評価を返します。そうでなければ、`0` を返します。

`expr \& expr` `expr` が両方とも `NULL` または `0` 以外の場合、最初の `expr` を返します。それ以外の場合は、`0` を返します。

`expr { =, \>, \>=, \<, \<=, != } expr` 両引数とも整数の場合、整数比較の結果を返します。それ以外の場合、ロケール固有の連立シーケンスを使用して、文字列比較の結果を返します。各比較の結果は、指定された関係が `TRUE` の場合は `1`、`FALSE` の場合は `0` になります。

`expr { +, - } expr` 整数値引数の加減。

`expr { \*, /, % } expr` 整数値引数の乗除または剰余。

`expr : expr` マッチング演算子 `:` (コロン) は、最初の引数と 2 番目の引数を比較します。2 番目の引数は国際化された基本正規表現である必要があります。ただ

し、すべてのパターンが文字列の最初にアンカーされるという点を除きます。つまり、文字列の最初の文字から始まるシーケンスだけが正規表現と一致することができます。regex(5) および「注意事項」を参照してください。通常、`/usr/bin/expr` のマッチング演算子は一致したバイト数を返し、`/usr/xpg4/bin/expr` のマッチング演算子は一致した文字数を返します (失敗時は 0)。2 番目の引数に少なくとも 1 つの、基本正規表現のサブエクスプレッション `[\\(\\...\\)]` が含まれる場合、マッチング演算子は `\\1` に対応する文字列を返します。

<i>integer</i>	数字だけからなる引数。先頭に負記号を付けることもできます。
<i>string</i>	<i>integer</i> 引数、または演算子符号の 1 つとして識別することができない文字列の引数です。
<i>index</i> 、 <i>length</i> 、 <i>match</i> 、および <i>substr</i> の 4 つの演算子の優先度はすべて同じです。	
<i>index string character-list</i>	<i>character-list</i> のバイトが <i>string</i> のバイトと一致する、 <i>string</i> の最初のバイトを報告します (1 からカウントします)。 <i>character-list</i> のバイトが <i>string</i> のバイトと一致しない場合は、0 が返されます。
<i>length string</i>	<i>string</i> の長さ (バイト数) を返します。終端 NULL 文字はそのカウントには含まれません。
<i>match string regular-expression</i>	<i>expr</i> : <i>expr</i> マッチング演算子と同義です。
<i>substr string integer-1 integer-2</i>	位置 <i>integer-1</i> から始まる、長さが <i>integer-2</i> バイトのバイトシーケンスを <i>string</i> から抽出します (1 からカウントします)。 <i>integer-1</i> が <i>string</i> のバイト数よりも大きい値である場合、 <i>expr</i> は NULL 文字列を返します。 <i>string</i> にあるバイト数よりも多く抽出しようとした場合、 <i>expr</i> は <i>string</i> から残りのすべてのバイトを返します。 <i>integer-1</i> または <i>integer-2</i> のどちらかが負の値である場合、結果は不確定です。

使用例

例1 整数をシェル変数に追加する
 シェル変数 *a* に 1 を追加します。

```
example$ a='expr $a + 1'
```

例2 パス名の要素を返す

以下は [basename\(1\)](#) をエミュレートする例です。これは、\$a というパス名の最後の要素を返します。\$a が /usr/abc/file か、ただの file のいずれかと等価の場合に、この例は、file を返します。(引数としての単独の / に注意してください。expr は / を除算演算子とみなします。後述の「注意事項」を参照。)

```
example$ expr $a : '.*\/(.*\)' \| $a
```

例3 //文字を使用して式を単純にする

前の例を改善したものです。// 文字を追加してあるので、除算演算子についてのあいまいさはなくなり、式全体が単純になっています。

```
example$ expr // $a : '.*\/(.*\)'
```

/usr/bin/expr

例4 変数にバイト数を返す

```
example$ expr "$VAR" : '.*'
```

/usr/xpg4/bin/expr

例5 変数に文字数を返す

```
example$ expr "$VAR" : '.*'
```

環境変数

expr の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

式を評価した結果として、expr は次の終了値を返します。

- 0 式が NULL でも 0 でもない場合。
- 1 式が NULL または 0 である場合。
- 2 不正な式の場合。
- >2 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効。「注意事項」を参照してください。
インタフェースの安定性	下記を参照。
標準	standards(5) を参照してください。

match、substr、length、および index 演算子は「不確実」です。ほかはすべて「確実」です。

関連項目	basename(1) , ed(1) , sh(1) , Intro(3) , attributes(5) , environ(5) , regex(5) , standards(5)
診断	syntax error 演算子とオペランドのエラー non-numeric argument 数値ではない文字列に対して演算を行う
注意事項	次の3つの演算子はCSIに対応していません。これらは、<code>/usr/xpg4/bin/expr</code> および <code>/usr/xpg6/bin/expr</code> で使用することもできません。 <i>index string character-list</i> <i>length string</i> <i>substr string integer-1 integer-2</i> シェルによる引数の処理後、<code>expr</code> は、値による場合を除き、演算子とオペランドの違いを区別できなくなります。<code>\$a</code> が <code>=</code> の場合、 <pre>example\$ expr \$a = '='</pre> というコマンドは、次のコマンドと等価になります。 <pre>example\$ expr = = =</pre> これは、引数が <code>expr</code> に渡される(さらに、すべて引数が <code>=</code> 演算子と見なされる)ためです。次のコマンドは正しく動作します。 <pre>example\$ expr X\$a = X=</pre>
正規表現	旧バージョンのあるバージョンとは違って、 <code>expr</code> はシステムに提供されたすべてのロケールに対して、国際化された基本正規表現を使用します。国際化された正規表現に関しては regex(5) のマニュアルページを参照してください。

名前	expr – evaluate arguments as a logical, arithmetic, or string expression														
形式	/usr/ucb/expr <i>argument</i> ...														
機能説明	The <i>expr</i> utility evaluates expressions as specified by its arguments. After evaluation, the result is written on the standard output. Each token of the expression is a separate argument, so terms of the expression must be separated by blanks. Characters special to the shell must be escaped. Note: 0 is returned to indicate a zero value, rather than the null string. Strings containing blanks or other special characters should be quoted. Integer-valued arguments may be preceded by a unary minus sign. Internally, integers are treated as 32-bit, two's-complement numbers. The operators and keywords are listed below. Characters that need to be escaped are preceded by '\'. The list is in order of increasing precedence, with equal precedence operators grouped within { } symbols. <table> <tr> <td><i>expr</i> \ <i>expr</i></td><td>Returns the evaluation of the first <i>expr</i> if it is neither NULL nor 0; otherwise, returns the evaluation of the second <i>expr</i> if it is not NULL; otherwise, 0.</td></tr> <tr> <td><i>expr</i> \& <i>expr</i></td><td>Returns the first <i>expr</i> if neither <i>expr</i> is NULL or 0, otherwise returns 0.</td></tr> <tr> <td><i>expr</i> { =, \, \, \<, \<=, != } <i>expr</i></td><td>Returns the result of an integer comparison if both arguments are integers, otherwise returns the result of a lexical comparison.</td></tr> <tr> <td><i>expr</i> { +, - } <i>expr</i></td><td>Addition or subtraction of integer-valued arguments.</td></tr> <tr> <td><i>expr</i> { \, /, % } <i>expr</i></td><td>Multiplication, division, or remainder of the integer-valued arguments.</td></tr> <tr> <td><i>string</i> : <i>regular-expression</i> <i>match string regular-expression</i></td><td>The two forms of the matching operator above are synonymous. The matching operators : and <i>match</i> compare the first argument with the second argument which must be a regular expression. Regular expression syntax is the same as that of regexp(5), except that all patterns are “anchored” (treated as if they begin with ^) and therefore ^ is not a special character, in that context. Normally, the matching operator returns the number of characters matched (0 on failure). Alternatively, the \ . . . \ pattern symbols can be used to return a portion of the first argument.</td></tr> <tr> <td><i>substr string integer-1 integer-2</i></td><td>Extracts the substring of <i>string</i> starting at position <i>integer-1</i> and of length <i>integer-2</i> characters. If <i>integer-1</i> has a value greater than the length of <i>string</i>, <i>expr</i> returns</td></tr> </table>	<i>expr</i> \ <i>expr</i>	Returns the evaluation of the first <i>expr</i> if it is neither NULL nor 0; otherwise, returns the evaluation of the second <i>expr</i> if it is not NULL; otherwise, 0.	<i>expr</i> \& <i>expr</i>	Returns the first <i>expr</i> if neither <i>expr</i> is NULL or 0, otherwise returns 0.	<i>expr</i> { =, \, \, \<, \<=, != } <i>expr</i>	Returns the result of an integer comparison if both arguments are integers, otherwise returns the result of a lexical comparison.	<i>expr</i> { +, - } <i>expr</i>	Addition or subtraction of integer-valued arguments.	<i>expr</i> { \, /, % } <i>expr</i>	Multiplication, division, or remainder of the integer-valued arguments.	<i>string</i> : <i>regular-expression</i> <i>match string regular-expression</i>	The two forms of the matching operator above are synonymous. The matching operators : and <i>match</i> compare the first argument with the second argument which must be a regular expression. Regular expression syntax is the same as that of regexp(5) , except that all patterns are “anchored” (treated as if they begin with ^) and therefore ^ is not a special character, in that context. Normally, the matching operator returns the number of characters matched (0 on failure). Alternatively, the \ . . . \ pattern symbols can be used to return a portion of the first argument.	<i>substr string integer-1 integer-2</i>	Extracts the substring of <i>string</i> starting at position <i>integer-1</i> and of length <i>integer-2</i> characters. If <i>integer-1</i> has a value greater than the length of <i>string</i> , <i>expr</i> returns
<i>expr</i> \ <i>expr</i>	Returns the evaluation of the first <i>expr</i> if it is neither NULL nor 0; otherwise, returns the evaluation of the second <i>expr</i> if it is not NULL; otherwise, 0.														
<i>expr</i> \& <i>expr</i>	Returns the first <i>expr</i> if neither <i>expr</i> is NULL or 0, otherwise returns 0.														
<i>expr</i> { =, \, \, \<, \<=, != } <i>expr</i>	Returns the result of an integer comparison if both arguments are integers, otherwise returns the result of a lexical comparison.														
<i>expr</i> { +, - } <i>expr</i>	Addition or subtraction of integer-valued arguments.														
<i>expr</i> { \, /, % } <i>expr</i>	Multiplication, division, or remainder of the integer-valued arguments.														
<i>string</i> : <i>regular-expression</i> <i>match string regular-expression</i>	The two forms of the matching operator above are synonymous. The matching operators : and <i>match</i> compare the first argument with the second argument which must be a regular expression. Regular expression syntax is the same as that of regexp(5) , except that all patterns are “anchored” (treated as if they begin with ^) and therefore ^ is not a special character, in that context. Normally, the matching operator returns the number of characters matched (0 on failure). Alternatively, the \ . . . \ pattern symbols can be used to return a portion of the first argument.														
<i>substr string integer-1 integer-2</i>	Extracts the substring of <i>string</i> starting at position <i>integer-1</i> and of length <i>integer-2</i> characters. If <i>integer-1</i> has a value greater than the length of <i>string</i> , <i>expr</i> returns														

a null string. If you try to extract more characters than there are in *string*, *expr* returns all the remaining characters from *string*. Beware of using negative values for either *integer-1* or *integer-2* as *expr* tends to run forever in these cases.

index string character-list

Reports the first position in *string* at which any one of the characters in *character-list* matches a character in *string*.

length string

Returns the length (that is, the number of characters) of *string*.

(*expr*)

Parentheses may be used for grouping.

使用例

例 1 Adding an integer to a shell variable

Add 1 to the shell variable *a*.

```
a='expr $a + 1'
```

例 2 Returning a path name segment

Return the last segment of a path name (that is, the filename part). Watch out for / alone as an argument: *expr* will take it as the division operator (see BUGS below).

```
# 'For $a equal to either "/usr/abc/file" or just "file"'  
expr $a : '.*\/ \ $a'
```

例 3 Using // characters to simplify the expression

The addition of the // characters eliminates any ambiguity about the division operator and simplifies the whole expression.

```
# A better representation of example 2.  
expr // $a : '.*\/'
```

例 4 Returning the value of a variable

Returns the number of characters in \$VAR.

```
expr $VAR : '.*'
```

終了ステータス

expr returns the following exit codes:

- 0 If the expression is neither NULL nor 0.
- 1 If the expression is NULL or 0.
- 2 For invalid expressions.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [sh\(1\)](#), [test\(1\)](#), [attributes\(5\)](#), [regex\(5\)](#)

診断

syntax error	for operator/operand errors
non-numeric argument	if arithmetic is attempted on such a string
division by zero	if an attempt to divide by zero is made

使用上の留意点 After argument processing by the shell, `expr` cannot tell the difference between an operator and an operand except by the value. If `$a` is an `=`, the command:

```
expr $a = '='
```

looks like:

```
expr = = =
```

as the arguments are passed to `expr` (and they will all be taken as the `=` operator). The following works:

```
expr X$a = X=
```

Note: the `match`, `substr`, `length`, and `index` operators cannot themselves be used as ordinary strings. That is, the expression:

```
example% expr index expurgatorious length
syntax error
example%
```

generates the 'syntax error' message as shown instead of the value 1 as you might expect.

名前	exstr - ソースファイルからの文字列の抽出
形式	exstr <i>filename</i>... exstr -e <i>filename</i>... exstr -r [-d] <i>filename</i>...
機能説明	exstr ユーティリティーは、C 言語のソースファイルから文字列を抽出し、それをメッセージ検索関数の呼び出し (gettext(3C) を参照) に置き換えます。このユーティリティーは、printf コマンドや printf ルーチンへの引数として指定された文字列だけではなく、二重引用符で囲まれている文字列もすべて抽出します。第 1 の形式では、exstr はソースファイルからすべての文字列を取り出して、標準出力上に書き出します。各文字列の先頭には、ソースファイル名とコロン(:) が付加されます。 最初のステップとして、まず exstr -e を使って一群の文字列を入力ファイルから取り出し、別のファイルに保存します。次にそれらのうちの文字列が、あとでメッセージ検索関数で検索できるように変換可能かを見つけ出します。次に、変換可能でない行を削除し、変換可能な行に関してはメッセージファイル名とメッセージ番号を第 4 項目 (<i>msgfile</i>) と第 5 項目 (<i>msgnum</i>) として追加します。指定するメッセージファイルは、mkmsgs(1) で作成したもので、<code>/usr/lib/locale/locale/LC_MESSAGES</code> ディレクトリ中に存在している必要があります。このパス中の <code>locale</code> は、テキスト文字列が書かれている言語に対応しています。詳しくは setlocale(3C) を参照してください。指定するメッセージ番号は、メッセージファイル中の文字列のシーケンス番号に一致している必要があります。 次に、このようにして編集したファイルを入力に指定して exstr -r を実行します。これにより元の C 言語のソースファイルの新バージョンとして、文字列がメッセージ検索関数 <code>gettext()</code> の呼び出しに置き換えられているファイルが生成されます。<i>msgfile</i> と <i>msgnum</i> の 2 つのフィールドは、<code>gettext()</code> へ渡す第 1 引数を構築するために用いられます。<code>gettext()</code> への第 2 引数は、実行時にメッセージ検索が失敗したときに印刷されます。この引数は、-d オプションが指定された場合を除き、NULL 文字列です。 このユーティリティーは、すべての場合に文字列を変換できるわけではありません。たとえば、静的初期化文字列を関数呼び出しに置き換えることはできません。また文字列が、変換できないエスケープシーケンスの形式をとっている場合もあります。既存のコードを破壊しないようにするため、exstr -e を使って生成したファイルの中身をよく検査して、関数呼び出しに置換できない文字列を含んでいる行を必ず削除してください。また場合によっては、文字列が抽出されてメッセージ検索関数の呼び出しに置き換えられるように、コードを修正する必要があります。

オプション

サポートしているオプションは、次のとおりです。

- e 指定した C 言語のソースファイルから、文字列リストと位置情報を取り出します。リストは以下に示す形式で標準出力上に書き込まれます。

```
file:line:position:msgfile:msgnum:string
```

各情報の意味は次のとおりです。

```
file      C 言語のソースファイル
line      ファイル中の行番号
position   その行における文字位置
msgfile    NULL 文字列
msgnum     NULL 文字列
string     抽出されたテキスト文字列
```

この出力先としてファイルを指定することも可能です。ファイルに書き出せば、あとでその内容を編集して *msgfile* と *msgnum* 用に使いたい値を追加することができます。

```
msgfile    string を置き換えるテキスト文字列を含んでいるファイル。mkmsgs(1) ユーティリティーを使って、この名前を持つファイルを作成して適切なディレクトリに置いておく必要があります。
```

```
msgnum     msgfile 中のシーケンス番号
```

次の手順として、`exstr -r` を使って *file* の *string* を置き換えます。

- r C 言語のソースファイル中の文字列を、メッセージ検索関数 `gettext()` の呼び出しに置き換えます。
- d このオプションは、`-r` オプションとともに使用します。実行時に `gettext()` を呼び出してメッセージの検索が成功しなかった場合、抽出された文字列が印刷されます。国際的環境で動作してメッセージを複数の言語で印刷するようなアプリケーションプログラム上では、`exstr` が提供する機能を使います。`exstr` は、テキスト文字列を、メッセージデータベース中の文字列を指す関数呼び出しに置き換えます。どのデータベースを用いるかは、環境変数 `LC_MESSAGES` の実行時の値により決まります (`environ(5)` を参照)。

使用例

例 1 `exstr` の使用例

`example.c` というファイル中に、次のような 2 つの文字列が含まれていると想定します。

```
main()
```

```
{
```

例1 exstr の使用例 (続き)

```

    printf("This is an example\n");

    printf("Hello world!\n");

}

```

このとき、以下のように `example.c` を引数として指定して `exstr` を実行すると、これらの文字列が取り出されて 標準出力上に書き込まれます。

```
example% exstr example.c
```

これにより次の出力が得られます。

```
example.c:This is an example\n
example.c:Hello world!\n

```

次に `-e` オプションと出力ファイル名を指定した例を示します。

```
example% exstr -e example.c > example.stringsout
```

この場合には、`example.stringsout` というファイルに以下の出力が得られます。

```
example.c:3:8::This is an example\n
example.c:4:8::Hello world!\n

```

次にこの `example.stringsout` ファイルを編集して、これらの文字列が検索関数呼び出しに置き換えられる前に、`msgfile` と `msgnum` の両フィールド用に使用したい値を追加します。たとえばメッセージファイルの名前が `ux` で、ファイル中の文字列のシーケンス番号が 1 と 2 であったとすると、`example.stringsout` ファイルを編集して次のような内容に変更します。

```
example.c:3:8:UX:1:This is an example\n
example.c:4:8:UX:2:Hello world!\n

```

このように準備を整えたら、`-r` オプション付きで `exstr` ユーティリティーを実行して、ソースファイル中の文字列をメッセージ検索関数 `gettext()` の呼び出しに置き換えることができます。

```
example% exstr -r example.c <example.stringsout >intlexample.c
```

これにより次の出力が得られます。

```
extern char *gettext();

main()

{

```

例1 exstr の使用例 (続き)

```
printf(gettxt("UX:1", ""));

printf(gettxt("UX:2", ""));

}
```

次の例を見てください。

```
example% exstr -rd example.c <example.stringsout >intlexample.c
```

この場合には、抽出された文字列が `gettext()` 関数への第2引数として用いられます。

```
extern char *gettext();

main()
{
    printf(gettxt("UX:1", "This is an example\n"));

    printf(gettxt("UX:2", "Hello world!\n"));

}
```

ファイル

`/usr/lib/locale/locale/LC_MESSAGES/*` [mkmsgs\(1\)](#) が生成したファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/locale

関連項目

[gettext\(1\)](#), [mkmsgs\(1\)](#), [printf\(1\)](#), [srchtxt\(1\)](#), [gettext\(3C\)](#), [printf\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

診断

`exstr` が発行するエラーメッセージは、メッセージテキストを読めば意味が分かるようになっています。エラーメッセージはコマンド行で見つかったエラー、また入力ファイル中で見つかった形式エラーを表します。

名前	factor – 素因数の分解
形式	factor [integer]
機能説明	factor は、10^{14} 以下の正の整数の、すべての素因数を標準出力に書き込みます。素因数は適切な回数だけ書き込まれます。 factor を引数なしで実行した場合、factor は整数が入力されるのを待ちます。整数が入力されると、factor はその整数を素因数に分解し、適切な回数だけ出力して、再び整数が入力されるのを待ちます。0 または数値以外の文字を入力すると、factor は終了します。 引数 (整数) を指定して factor を実行すると、factor はその整数を出力し、素因数に分解して、すべての素因数を前述のと同様に出力し、終了します。引数が 0 または数値以外の文字の場合、factor は 0 を出力して終了します。 整数を素因数分解する最大時間は $\text{sqrt}(n)$ (n は入力する整数) に比例します。n が素数または素数の二乗である場合、factor は、素因数の計算にこの最大時間を必要とします。
オペランド	integer 10^{14} 以下の正の整数
終了ステータス	0 正常終了。 1 エラーが発生した。
診断	範囲外の数値や意味のない文字の入力に対しては、factor は「Ouch!」というエラーメッセージを出力します。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/extended-system-utilities

関連項目 [attributes\(5\)](#)

- 名前

fastboot, fasthalt – reboot/halt the system without checking the disks
- 形式

/usr/ucb/fastboot [*boot-options*]

/usr/ucb/fasthalt [*halt-options*]
- 機能説明

fastboot and fasthalt are shell scripts that invoke reboot and halt with the proper arguments.

These commands are provided for compatibility only.
- 属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

- 関連項目

[fsck\(1M\)](#), [halt\(1M\)](#), [init\(1M\)](#), [reboot\(1M\)](#), [init.d\(4\)](#), [attributes\(5\)](#)

名前	fgrep – ファイル内の固定文字列のみを検索
形式	<pre> /usr/bin/fgrep [-bchilnsvx] -e pattern_list [file...] /usr/bin/fgrep [-bchilnsvx] -f file [file...] /usr/bin/fgrep [-bchilnsvx] pattern [file...] /usr/xpg4/bin/fgrep [-bchilnsvx] -e pattern_list [-f file] [file...] /usr/xpg4/bin/fgrep [-bchilnsvx] [-e pattern_list] -f file [file...] /usr/xpg4/bin/fgrep [-bchilnsvx] pattern [file...] </pre>
機能説明	fgrep (固定 grep) ユーティリティーは、ファイルで文字列を検索し、その文字列を含むすべての行を出力します。fgrep は文字列を検索するため、式に一致するパターンを検索する grep(1) や egrep(1) とは異なります。 \$、*、[, ^、 、(、)、および \ の文字は、fgrep により文字どおりに解釈されます。すなわち、fgrep は、egrep の場合と異なり、正規表現をすべて認識できるわけではありません。これらの文字はシェルに対して特別の意味を持ちます。そのため、安全を期すために、全 <i>string</i> を単一引用符 (') で囲むことをお勧めします。 ファイルを何も指定しないと、fgrep は標準入力を想定します。通常、発見された各行は標準出力にコピーされます。複数の入力ファイルがある場合、表示された各行の前にファイル名が出力されます。
オプション	次のオプションは、<code>/usr/bin/fgrep</code> と <code>/usr/xpg4/bin/fgrep</code> の両方のコマンドでサポートされています。 -b 各行の先頭に、その行が発見されたブロック番号を出力します。これは、ブロック番号を文脈別に見つける際に役立ちます。最初のブロックは0です。 -c パターンを含む行の数だけを出力します。 -e pattern_list pattern_list にある <i>string</i> を検索します。これは、<i>string</i> が - ではじまる場合に有効です。 -f pattern-file pattern-file からパターンのリストを取り出します。 -h 複数のファイルを検索するときにファイルの出力を抑制します。 -i 比較中に大文字と小文字の区別を無視します。 -l 一致する行のあるファイルの名前を1行ずつ復帰改行文字で区切って出力します。パターンが2度以上見つかるときは、ファイルの名前を繰り返しません。 -n 各行の先頭に、その行のファイル中の行番号を出力します。最初の行は1です。

- s

エラーメッセージを表示するだけです。エラーステータスをチェックするのに有用です。
- v

パターンを含む行を除いたすべての行を出力します。
- x

完全に一致する行だけを出力します。

/usr/xpg4/bin/fgrep

次のオプションは、`/usr/xpg4/bin/fgrep` のみでサポートされています。

-q

非出力。行が一致したかどうかに関わらず、標準出力には何も書き出しません。入力行に一致した行があった場合、0の終了ステータスで処理を終了します。

オペランド

次のオペランドがサポートされています。

file

パターンを検索するファイルのパス名を指定します。*file* を指定しないと、標準入力を読み込まれます。

/usr/bin/fgrep

pattern 入力の実行時に用いるパターンを指定します。

/usr/xpg4/bin/fgrep

pattern 入力の実行時に用いる1つまたは複数のパターンを指定します。オペランドは `-e pattern_list` が指定されたものとして扱われます。

使用法

ファイルが2ギガバイト (2³¹ バイト) 以上ある場合の `fgrep` の動作については、[largefile\(5\)](#) を参照してください。

環境変数

`fgrep` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス

次の終了ステータスが返されます。

0

一致するものが1つ以上見つかりました。

1

一致するものが1つも見つかりませんでした。

2

構文エラーが検出された、またはアクセスできないファイルがあった (一致するものが見つかった場合でも)。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/fgrep

属性タイプ	属性値
使用条件	system/core-os

/usr/xpg4/bin/fgrep

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効

関連項目	ed(1) , egrep(1) , grep(1) , sed(1) , sh(1) , attributes(5) , environ(5) , largefile(5) , XPG4(5)
注意事項	理想的には <code>grep</code> コマンドを1つだけにすべきなのですが、空間と時間を調節できるだけの広範なアルゴリズムはありません。 1行は仮想記憶に使用できるサイズに制限されています。
<code>/usr/xpg4/bin/fgrep</code>	<code>/usr/xpg4/bin/fgrep</code> ユーティリティーは <code>/usr/xpg4/bin/grep -F</code> と同じです (grep(1) 参照)。移植性が必要なアプリケーションでは <code>/usr/xpg4/bin/grep -F</code> を使用してください。

名前 file – ファイルタイプの判別

形式

```

/usr/bin/file [-dh] [-m mfile] [-M Mfile] [-f ffile]
file...

/usr/bin/file [-dh] [-m mfile] [-M Mfile] -f ffile

/usr/bin/file -i [-h] [-f ffile] file...

/usr/bin/file -i [-h] -f ffile

/usr/bin/file -c [-d] [-m mfile] [-M Mfile]

/usr/xpg4/bin/file [-dh] [-m mfile] [-M Mfile] [-f ffile]
file...

/usr/xpg4/bin/file [-dh] [-m mfile] [-M Mfile] -f ffile

/usr/xpg4/bin/file -i [-h] [-f ffile] file...

/usr/xpg4/bin/file -i [-h] -f ffile

/usr/xpg4/bin/file -c [-d] [-m mfile] [-M Mfile]
```

機能説明

file ユーティリティーは、*file* で指定した各ファイル、およびオプションとして指定した *ffile* 中に記述された個々のファイルを分類します。指定したファイルが通常ファイル以外である場合、そのタイプを判別します。判別されるタイプには、ディレクトリ、FIFO、ブロック特殊ファイル、文字特殊ファイルなどがあります。ファイル長がゼロの通常ファイルは、空ファイルと判別します。

file がテキストファイルである場合、file は最初の 512 バイトを検査し、そのプログラミング言語を判定します。*file* がシンボリックリンクである場合、デフォルトでは、file はそのリンクをたどって参照先のファイルをテストします。

file 引数が再配置可能オブジェクト、実行可能ファイル、または共有オブジェクトである場合、file コマンドはそのファイルの実行要件に関する情報を出力します。この情報には、マシンのクラス、バイト順、静的リンクまたは動的リンク、およびソフトウェア能力またはハードウェア能力に関する要件が入っています。*file* が実行時のリンク先の構成ファイルである場合、file は、マシンクラスやバイト順序などの、ターゲットのプラットフォームに関する情報を出力します。

/usr/lib/locale/locale/LC_MESSAGES/magic ファイルが存在する場合、file はデフォルトで、このマジックファイルを使用してマジック番号を持つファイルを識別します。たとえば、日本語ロケールの場合は */usr/lib/locale/ja/LC_MESSAGES/magic* を使用します。*/usr/lib/locale/locale/LC_MESSAGES/magic* が存在しない場合、file は */etc/magic* ファイルを使用します。マジック番号は、ファイルタイプを表す数値定数または文字列定数です。*/etc/magic* の形式については [magic\(4\)](#) を参照してください。

file が存在しない、読み取りできない、またはステータスが判別できない場合でも、終了ステータスに影響を与えるようなエラーとは見なされません。この場合、ファイルの処理は行われたがタイプは判断できなかったということが示されます。

オプション

サポートしているオプションは、次のとおりです。

- c マジックファイルにフォーマットエラーがないかどうかをチェックします。効率上の問題から、この妥当性検査は通常実行しません。
- d システムのデフォルトの位置依存および文脈依存テストをファイルに適用します。
- f *ffile* *ffile* は、検査すべきファイルの一覧が記述されているファイルの名前を表します。
- h シンボリックリンクに遭遇したとき、ファイルをシンボリックリンクとして識別します。-h を指定しない場合でも、*file* がシンボリックリンクであり、存在しないファイルを参照する場合、*file* ユーティリティーは、-h が指定されたように、*file* をシンボリックリンクとして識別します。
- i ファイルが通常ファイルである場合、それ以上ファイルのタイプを分類せず、そのファイルを「通常ファイル」として識別します。

-m *mfile*

/usr/bin/file *mfile* を */etc/magic* に代わる代替マジックファイルとして使用します。

/usr/xpg4/bin/file 位置依存テストを含むファイルの名前を指定します。このテストは、ファイルを分類するときに適用されます ([magic\(4\)](#) を参照)。-d オプションまたは -M オプションを指定せず、-m オプションだけを指定した場合、-m オプションで指定した位置依存テストの後に、システムのデフォルトの位置依存テストが適用されます。

- M *Mfile* 位置依存テストを含むファイルの名前を指定します。このテストは、ファイルを分類するときに適用されます ([magic\(4\)](#) を参照)。-d オプションと一緒に指定しない限り、システムのデフォルトの位置依存テストまたは文脈依存テストは適用されません。

-d オプション、-m オプション、あるいは、その両方と一緒に、--M オプションを指定した場合、あるいは、-d オプションと一緒に、-m オプションを指定した場合、これらのオプションで指定した位置依存テストはすべて、これらのオプションに指定した順番どおりに適用されます。

オペランド	次のオペランドがサポートされています。 <i>file</i> テストするファイルのパス名。
使用法	ファイルが2G バイト (2 ³¹ バイト) 以上ある場合の <i>file</i> の動作については、 largefile(5) を参照してください。
使用例	例1 引数として指定したファイルがバイナリの実行可能ファイルであるかどうかを検査する 次の例は、引数として指定したファイルがバイナリの実行可能ファイルであるかどうかを検査します。 <pre>file "\$1" grep -Fq executable && printf "%s is executable.\n" "\$1"</pre>
環境変数	<i>file</i> の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。
終了ステータス	次の終了ステータスが返されます。 <i>0</i> 正常終了。 <i>>0</i> エラーが発生した。
ファイル	/etc/magic <i>file</i> のマジック番号ファイル
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目	crle(1) , elfdump(1) , elffile(1) , ls(1) , magic(4) , attributes(5) , environ(5) , largefile(5) , standards(5)
注意事項	<i>file</i> ユーティリティでは、最初にアーカイブメンバーがアーカイブから別個のファイルに抽出されていないかぎり、アーカイブメンバーを検査することはできません。 <i>elffile</i> ユーティリティでは、所定の場所にあるアーカイブメンバーを検査できるため、ELF オブジェクトおよびアーカイブでは、このユーティリティを使用することをお勧めします。

名前	file – determine the type of a file by examining its contents
形式	<code>/usr/ucb/file [-f <i>ffile</i>] [-cL] [-m <i>mfile</i>] <i>filename</i>...</code>
機能説明	file performs a series of tests on each <i>filename</i> in an attempt to determine what it contains. If the contents of a file appear to be ASCII text, file examines the first 512 bytes and tries to guess its language. file uses the file <code>/etc/magic</code> to identify files that have some sort of <i>magic number</i>, that is, any file containing a numeric or string constant that indicates its type.
オプション	-c Check for format errors in the magic number file. For reasons of efficiency, this validation is not normally carried out. No file type-checking is done under -c. -f <i>ffile</i> Get a list of filenames to identify from <i>ffile</i>. -L If a file is a symbolic link, test the file the link references rather than the link itself. -m <i>mfile</i> Use <i>mfile</i> as the name of an alternate magic number file.
使用例	例1 Using file on all the files in a specific user's directory. This example illustrates the use of file on all the files in a specific user's directory: <pre>example% pwd /usr/blort/misc example% /usr/ucb/file *</pre> <pre>code: mc68020 demand paged executable code.c: c program text counts: ascii text doc: roff,nroff, or eqn input text empty.file: empty libz: archive random library memos: directory project: symboliclink to /usr/project script: executable shell script titles: ascii text s5.stuff: cpio archive</pre> example%
環境変数	The environment variables LC_CTYPE, LANG, and LC_default control the character classification throughout file. On entry to file, these environment variables are checked in the following order: LC_CTYPE, LANG, and LC_default. When a valid value is found, remaining environment variables for character classification are ignored. For example, a new setting for LANG does not override the current valid character classification rules of LC_CTYPE. When none of the values is valid, the shell character classification defaults to the POSIX.1 “C” locale.

```
ファイル      /etc/magic
```

属性 See `attributes(5)` for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [magic\(4\)](#), [attributes\(5\)](#)

使用上の留意点 file often makes mistakes. In particular, it often suggests that command files are C programs.
file does not recognize Pascal or LISP.

名前	filebench – ファイルシステムのパフォーマンスを測定および比較するためのフレームワークまたはワークロード
形式	<code>/usr/benchmarks/filebench/bin/filebench profile</code> <code>/usr/benchmarks/filebench/bin/filebench -c stats_dir...</code>
機能説明	filebench は、ワークロードを実行してファイルシステムのパフォーマンスを測定および比較します。 詳細な情報は、パフォーマンスのコミュニティー (http://hub.opensolaris.org/bin/view/Main/) で入手できます。
オプション	サポートしているオプションは、次のとおりです。 <code>-c stats_dir...</code> 指定されたディレクトリを比較した HTML ファイル (<code>index.html</code>) を生成します。 <code>stats_dir</code> は、結果が格納される 1 つまたは複数のディレクトリです。
オペランド	次のオペランドがサポートされています。 <code>profile</code> .prof で終わる構成ファイルの名前を指定します。構成ファイルには、次の内容を指定します。 ■ 実行するワークロード ■ 実行するパラメータ ■ 処理対象のディレクトリパス ■ 結果を格納するディレクトリパス
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 1 エラーが発生した。 2 無効なコマンド行オプションが指定された。
使用例	例1 マルチストリームシーケンシャル読み込みのワークロードを実行する 次の例では、 <code>sqread.pro</code> という名前の構成ファイルに記述されているワークロードを実行します。 <code># filebench sqread</code> 例2 複数の実行を比較する 次の例では、以前の 2 回の実行結果を比較します。 この例では、filebench を以前に 2 回実行したときの結果が <code>/stats/wombat-zfs-noel-Jun_27_2007-15h_45m_33s</code> および <code>/stats/wombat-ufs-noel-Jun_27_2007-15h_52m_11s</code> ディレクトリに格納されているものとします。

例2 複数の実行を比較する (続き)

この例では、現在の作業ディレクトリに `index.html` という名前の HTML ファイルが生成されます。

```
# filebench -c /stats/wombat-zfs-noel-Jun_27_2007-15h_45m_33s \  
/stats/wombat-ufs-noel-Jun_27_2007-15h_52m_11s
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	benchmark/filebench
インタフェースの安定性	不確実

関連項目 [attributes\(5\)](#)

名前	filesync – 通常ファイル、ディレクトリファイル、特殊ファイルの同期
形式	<pre>filesync [-aehmnqvy] [-o src dst] [-f src dst old new] [-r directory]... filesync [-aehmnqvy] -s source-dir -d dest-dir filename...</pre>
機能説明	filesync ユーティリティは、複数のコンピュータシステム (通常は、サーバーとポータブルコンピュータ) 間でファイルを同期させます。filesync は、通常ファイル、ディレクトリファイル、特殊ファイルの同期を行います。本来、モバイルシステムでの使用を目的としていますが、filesync は据え置き型のシステムにおけるバックアップやファイルの複製などにも便利です。 システム間でファイルを同期させた場合、それぞれのシステム上の対応するファイルが同じものになります。一方または両方のシステムでファイルを変更すると、対応するファイルは同じではなくなります (つまり、同期していません)。もう一度ファイルを同じにするには、ファイル間の違いを調整しなければなりません。filesync がどのようにファイルを調整して同期させるかについての詳細は、「ファイルの調整と同期」を参照してください。 filesync コマンドには2つの形式があります。最初の形式の filesync はファイル引数なしで呼び出します。この形式の filesync は \$HOME/.packingrules ファイルに指定されているファイルおよびシステム間の違いを調整します。\$HOME/.packingrules は、filesync 用のパッキング規則リストで、同期させるファイルのリストが入っています。packingrules(4) を参照してください。 2 番目の形式の filesync は、ソースシステム上のディレクトリから宛先システム上のディレクトリに特定のファイルをコピーします。さらに、この形式の filesync は、引数 (<i>filename</i>) として指定した (1 つまたは複数の) ファイルを \$HOME/.packingrules に追加します。ソースシステムと宛先システム上のディレクトリを指定する方法については、<code>-s</code> オプションと <code>-d</code> オプションを参照してください。ファイル (<i>filename</i>) 引数を指定する方法については、「オペランド」の項を参照してください。 filesync コマンドは複数累積できます。つまり、filesync コマンドに指定したファイルは既存のパッキング規則ファイルに追加されます。「複数の filesync コマンド」を参照してください。 ファイルの調停と同期 filesync はコンピュータシステム間でファイルを同期させるために、次の2つの処理を行います。 1. filesync は、両方のシステム上にあるパッキング規則ファイルに指定されているディレクトリとファイルを調べ、同じかどうかを判定します。異なるファイルには調整が必要です。 filesync はまた、監視されるすべてのファイルについて、\$HOME/.filesync-base ファイル内のベースラインサマリーを保守します。このファイルには、最後の調整時におけるファイルの名前、タイプ、およびサイズのリストが入っています。

2. ベースラインファイル内の情報と指定されたオプションに基づいて (「filesync の衝突の解決」を参照)、filesync はどちらのコピーが正しいものかを判別し、正しくない方のシステムに対して必要な変更を加えます。この変更が加えられると、2つのコピーは再び同じになります (つまり、同期しています)。

ソースファイルが変更され、宛先ファイルが変更されていない場合、ソースファイルの変更は宛先システムに伝播されます。宛先ファイルが変更され、対応するソースファイルが変更されていない場合、宛先ファイルの変更はソースシステムに伝播されます。両方のシステム上でファイルが変更されたが、それでもまだファイルが同じでない場合、手動で衝突を解決するようにユーザーに促す警告メッセージが出力されます。「filesync の衝突の解決」を参照してください。

filesync の衝突の解決

両方のシステム上のファイルが変更された場合、filesync は、どちらのバージョンを選択すべきかを決定しようとします。どちらのバージョンを選択すればいいのかを自動的に決定できない場合、filesync は警告メッセージを出力し、2つの互換性のないファイルを調整しないままにします。

この場合、ユーザーが手動で違いを解決しなければなりません。あるいは、どちらのバージョンを選択するかを filesync に指示しなければなりません。衝突を解決する方法を filesync に指示するには、`-o` オプションと `-f` オプションを使用します (「オプション」の項を参照)。

あるいは、衝突するファイルごとに、ユーザーが2つのバージョンを調べ、どちらを変更せずにおくかを決定し、手動で2つのバージョンを同期させます (つまり、所有権や保護モードを、コピー・削除・変更して修正します)。その後、filesync を再実行すると、衝突がまだ残っているかどうかわかります。

パッキング規則ファイル

パッキング規則ファイル `$HOME/.packingrules` には、同期すべきファイルのリストが入っています。このファイルの構文については、[packingrules\(4\)](#) を参照してください。

`$HOME/.packingrules` ファイルは、ユーザーが filesync を filename 引数付きで呼び出すと自動的に作成されます。filesync のオプションを使用することにより、ユーザーは `$HOME/.packingrules` ファイル内のパッキング規則を増やすことができます。

パッキング規則ファイルは手動でも作成および編集できます。任意のエディタで `$HOME/.packingrules` を編集すると、恒久的な変更を `$HOME/.packingrules` に加えたり、コマンド行からは利用できないより強力なオプション (IGNORE コマンドなど) を使用できます。複雑なワイルドカード式を入力する場合も、`$HOME/.packingrules` ファイルを編集する方がはるかに簡単です。

ベースラインファイル

`$HOME/.filesync-base` は、filesync ベースラインサマリーファイルです。filesync は `$HOME/.filesync-base` 内の情報を使用して、調整処理と同期処理の間にファイル間の差異を識別します。ユーザーはベースラインファイルを作成および編集する必

要はありません。ベースラインファイルはfilesyncにより自動的に作成されます。このファイルには、保守されるすべてのファイルについて、最後の調整時における状態が記録されます。

複数の filesync コマンド

しばらくすると、同期させたいファイルのセットが変わってしまうことがあります。たとえば、ノートブックには現在進行中のプロジェクトに関するファイルだけを保存したいと考えることはよくあることです。これまで係ってきたすべてのプロジェクトに関するファイルを同期させようとする、ノートブックのハードディスクは古いファイルでいっぱいになってしまいます。filesync コマンドを実行すると、不要なファイルの更新に時間を費やしてしまいます。

ユーザーがノートブック上のファイルを削除すると、filesync はサーバー上の対応するファイルも削除しようとします。これは好ましいことではありません。この場合、filesync に指示を与え、一部のファイルを同期させないようにする方法が必要になります。そのためには、次の2つの方法を利用します。

1. `$HOME/.packingrules` を編集します。削除したいファイルの規則を削除します。
2. `$HOME/.packingrules` を削除します。その後、filesync コマンドを使用して、同期させたいファイルを指定します。

どちらの方法も有効です。使いやすい方を選んでください。変更したいファイルの規則が少ない場合、`$HOME/.packingrules` を編集する方が簡単です。変更したいファイルの規則が多い場合は、`$HOME/.packingrules` を削除し、最初から作り直す方が簡単です。

filesync によるファイルセットの同期を停止すれば、サーバー上のファイルを気にすることなく、ノートブック上のファイルを削除できます。

モバイルマシン

filesync を使用してモバイルマシンとサーバー間でファイルを同期させる場合、パッキング規則ファイルとベースラインファイルは、サーバー上ではなく、モバイルマシン上に格納します。ノートブックにログインするときに、HOME 環境変数がノートブック上のディレクトリを正しく指定していない場合、FILESYNC 環境変数を使用すれば、パッキング規則ファイルとベースラインファイルの代替位置を指定できます。

各モバイルマシンは独自のパッキング規則ファイルとベースラインファイルを持っていなければなりません。サーバーにベースラインファイルが格納されており、複数のモバイルマシンがサーバーのベースラインに基づいて調整を試みると、ファイルの同期は正しくなくなります。この場合、モバイルマシンはファイルの状態を正確に記述していないベースラインファイルを使用することになり、結果として間違った調整が行われます。

単一のベースラインファイルを複数のマシン間で共有する際の危険性を回避するために、filesync はデフォルトの規則を新しいパッキング規則ファイルごとに追加します。このデフォルト規則により、パッキング規則ファイルとベースラインファイルがコピーされるのを防ぐことができます。

オプション

サポートしているオプションは、次のとおりです。

-a

ACL (Access Control List, アクセス制御リスト) を検査し、新しいファイルと変更されたファイルのすべてに対して ACL の同期を強制します。特定のファイルで ACL を設定できない場合、filesync はそのファイルに対する ACL 同期を停止します。

ファイルシステムの中には、ACL をサポートしていないものがあります。ACL をサポートしているファイルシステムとサポートしていないファイルシステム間で ACL を同期させることはできません。このようなファイルシステム間で ACL を同期させようとする、多数のエラーメッセージが生成されます。

-d dest-dir

filename をコピーする宛先システム上のディレクトリを指定します。-s *source-dir* オプションと *filename* オペランドと共に使用します。-s オプションおよび「オペランド」の項を参照してください。

-e

すべての違いにフラグを立てます。filesync を root ユーザーとして実行していない限り、保護モードや所有権に関するすべての衝突を完全に解決することは不可能です。通常、ファイルの所有権または保護モードを変更できない場合、filesync は所有権と保護モードの衝突を無視します。ただし、-e (すべての同期) フラグを指定した場合、filesync はこのような違いにもフラグを立てます。

-f src|dst|old|new

-f オプションは、衝突する変更をどのように解決するかを filesync に指示します。両方のシステム上でファイルが変更されているときに、-f オプションを指定すると、filesync は指定されたシステム上の変更をそのまま (有効) にし、指定されなかったシステム上の変更を破棄します。

-f *src* を指定すると、ソースシステム上のファイルを有効にします。-f *dst* を指定すると、宛先システム上のファイルを有効にします。-f *old* を指定すると、古いバージョンのファイルを有効にします。-f *new* を指定すると、新しいバージョンのファイルを有効にします。

-f オプションと -o オプションを組み合わせるときは、それぞれの *src* と *dst* が競合しないように指定しなければなりません。-f オプションと -o オプションで競合が発生した場合、-f オプションが無視されます。-o オプションを参照してください。

-h

エラー発生時に停止します。通常、ファイルのコピー時に読み取りエラーまたは書き込みエラーが発生した場合、filesync はエラーメッセージを出力し、別のファイルを調整しようとします。-h オプションを指定した場合、filesync はこのようなエラーが発生するとすぐに停止し、別のファイルの調整は行いません。

-m

ファイルの両方のコピーにおいて変更時間が同じであることを保証します。デフォルトでは、新たにコピーされるファイルの変更時間は調整時の時間に設定されます。ファイルの変更の順番は変更時間の昇順で決定されます。つまり、ファイルが伝播される順番は、ソースファイルの変更時間の相対的な順番と同じです。一般的に、2つのシステム間には時間のずれがあり、これらのシステム間で変更時間を転送すると、おかしい結果になることがある、という点に注意してください。

filesync でディレクトリ内のいくつかのファイル(すべてではない)を更新すると、make プログラムが混乱してしまうことがあります。たとえば、filesync が .c ファイルの同期は取るが、.o ファイルを無視する場合、変更された .c ファイルが(変更される前の .c ファイルから生成された) .o ファイルよりも前の変更時間を示すことがあります。

-n

実際の変更は行いません。-n オプションを指定した場合、filesync は、ファイルにどのような変更が加えられており、どのような調整が必要であるかを判別し、その情報を標準出力に表示します。ファイルの変更は行われません(パッキング規則ファイルも含む)。

-n オプションと -o オプションの両方を指定すると、filesync は、o オプションで指定されたシステムを解析し、そのシステムで加えられた変更を報告します。マシンが接続されていない(つまり、サーバーにアクセスできない)ときにローカルマシン上で加えられた変更を知りたい場合には、-n オプションと -o オプションを組み合わせると便利です。-o オプションを参照してください。

-o src | dst

-o オプションを指定すると、ソースシステム(src)または宛先システム(dst)のどちらか一方だけの調整を行います。

-o src を指定すると、filesync はソースシステムから宛先システムへの変更だけを伝播します。宛先システム上で加えられた変更は無視されます。ソースディレクトリまたは宛先ディレクトリにアクセスできない場合、filesync は中断します。

-o dst を指定すると、filesync は宛先システムからソースシステムへの変更だけを伝播します。ソースシステム上で加えられた変更は無視されます。ソースディレクトリまたは宛先ディレクトリにアクセスできない場合、filesync は中断します。

-n オプションと -o オプションの両方を指定すると、filesync は、o オプションで指定されたシステムを解析し、そのシステム上で加えられた変更を報告します。マシンが接続されていない(つまり、サーバーにアクセスできない)ときに

このローカルマシン上で加えられた変更を知りたい場合には、`-n` オプションと `-o` オプションを組み合わせると便利です。`-n` オプションを参照してください。

`-f` オプションと `-o` オプションを組み合わせるときは、それぞれの `src` と `dst` が競合しないように指定しなければなりません。`-f` オプションと `-o` オプションで競合が発生した場合、`-f` オプションが無視されます。`-f` オプションを参照してください。

`-q`

実行した各再調整アクションを記述する標準の `filesync` メッセージを抑制します。

標準の `filesync` メッセージは、UNIX のシェルコマンドの形式(たとえば、`mv`、`ln`、`cp`、`rm`、`chmod`、`chown`、`chgrp`、`setfacl` など)で各調整アクションを記述します。

`-r directory`

`directory` だけに調整を制限します。複数のディレクトリを指定するときは、複数の `-r` オプションを指定します。

`-s source-dir`

コピーする `filename` が存在しているソースシステム上のディレクトリを指定します。`-d dest-dir` オプションと `filename` オペランドと共に使用します。`-d` オプションおよび「オペランド」の項を参照してください。

`-v`

ファイルが比較されるたびに、追加の情報を標準出力に表示します。

`-y`

安全検査の確認応答を省略します。モバイルマシンはドメイン間を移動することがあり、`filesync` が操作するファイルの多くは NFS 経由でアクセスされることが予想されます。したがって、間違ったファイルシステムまたはサーバー上において、ローカルの変更を調整する危険性があります。間違ったファイルシステムまたはサーバー上で調整が行われること、多数のファイルが間違っても変更または削除されます。そのような事態を回避するために、`filesync` は調整する前にいくつかの安全検査を実行します。多数のファイルを削除したい場合、あるいは高いレベルのディレクトリの `i` ノード番号を変更した場合、`filesync` は調整する前にユーザーにそのことをプロンプトで確認します。このような変更が行われることをすでに知っており、確認を省略したい場合は、`-y (yes)` オプションを使用して、そのような確認に対して自動的に `y (yes)` で応答します。

オペランド

次のオペランドがサポートされています。

`filename` 指定したソースディレクトリ (`source-dir`) 内にある、同期すべき通常ファイル、ディレクトリ、シンボリックリンク、または特殊ファイルの名前。複数のファイルを指定するには、各 `filename` をスペースで区

切って指定します。*filename* オペランドは **-s** オプションと **-d** オプションと共に使用します。詳しくは「オプション」の項を参照してください。

filename が通常ファイルの場合、指定した宛先ディレクトリ (*dest-dir*) 内に、その通常ファイルが (同じファイル名で) 複製されます。

filename がディレクトリの場合、指定した宛先ディレクトリ (*dest-dir*) 内に、そのディレクトリおよびそのディレクトリの下の子ディレクトリとファイルが再帰的に複製されます。

filename がシンボリックリンクの場合、指定した宛先ディレクトリ (*dest-dir*) 内に、そのシンボリックリンクのコピーが複製されます。

filename が特殊ファイルである場合は、同じメジャーまたはマイナーデバイス番号を持つ特殊ファイルが指定された宛先ディレクトリ (*dest-dir*) 内に複製されます。**filesync** を使用して特殊ファイルを作成できるのはスーパーユーザーだけです。

宛先ディレクトリ (*dest-dir*) に作成されたファイルは、ソースディレクトリ内のファイルと同じアクセス許可 (所有者、グループ、他のユーザーについてのアクセス権) を持ちます。

filename 内にエスケープされたシェルのワイルドカード文字が含まれている場合、そのワイルドカード文字は `$HOME/.packingrules` に格納され、**filesync** が実行されるたびに評価されます。

たとえば、次のコマンドは、現在 `$RHOME` 内にある 2 つの特殊ファイルが `$HOME` に複製されたことを確認します。

```
filesync -s $RHOME -d $HOME a.c b.c
```

次の例は、`$RHOME` 内にある `*.c` ファイルが (後で作成されなくても) すべて `$HOME` に複製されたことを確認します。

```
filesync -s $RHOME -d $HOME '*.c'
```

任意の宛先ファイルがすでに存在している場合、**filesync** は、これらのファイルが同じであることを確認し、同じでない場合は警告を発行します。

ファイルがコピーされた後は、ソースと宛先に区別はありません。 (**-o** オプションと **-f** オプションで指定した場合を除く)。

環境変数

FILESYNC

filesync のパッキング規則ファイルとベースラインファイルのデフォルトの位置を指定します。この環境変数のデフォルト値は `$HOME` です。パッキング規則ファイルとベースラインファイルの名前には接尾辞 `.packingrules` と `.filesync-base` が付加されます。

- LC_MESSAGES

診断メッセージや情報メッセージの表示方法を決定します。cロケールでは、メッセージはプログラム自身が使用しているデフォルトの形(通常、米語)で表示されます。
- 終了ステータス

一般に、すべてのファイルがすでに最新のものである場合、あるいは、すべてのファイルが正常に調整されている場合、filesyncは0の終了ステータスで終了します。ただし、-n オプションが指定された場合、あるいは、エラーが発生した場合は、終了ステータスは次の論理和になります。

0

競合がなく、すべてのファイルが最新になっています。

1

解決可能な競合がいくつかあります。

2

手動で解決しなければならない競合がいくつかあります。

4

指定したファイルがいくつか存在しません。

8

いくつかのファイルでなアクセス権が不足しています。

16

パッキング規則ファイルまたはベースラインファイルにアクセスするときにエラーが発生しました。

32

無効な引数

64

指定した src または dst ディレクトリのどちらか(あるいは、その両方)にアクセスできませんでした。

128

その他の障害が発生しました。
- ファイル

\$HOME/.packingrules

同期すべきファイルのリスト

\$HOME/.filesync-base

ベースラインサマリーファイル
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/network/network-clients

関連項目

[packingrules\(4\)](#), [attributes\(5\)](#)

名前	find - ファイルの検索
形式	<pre>/usr/bin/find [-H -L] path... expression</pre> <pre>/usr/xpg4/bin/find [-H -L] path... expression</pre>
機能説明	find ユーティリティーは、<i>path</i> で指定した各パス名に含まれているディレクトリ階層を再帰的に下降し、次に示す一次子に書き込まれたブール型の式 <i>expression</i> と一致するファイルを検索します。 find は、ファイル階層を任意の深さまで下降することができ、アプリケーションによって指定される <i>path</i> オペランドが PATH_MAX の要件を超えないかぎり、パスの長さの制限が原因でエラーになることはありません。 find は、無限ループ、つまり、最後に検出されたファイルの祖先である、前にアクセスしたディレクトリに入る動作を検出します。
オプション	サポートしているオプションは、次のとおりです。 -H コマンド行でシンボリックリンクに遭遇した場合、シンボリックリンク自身ではなく、シンボリックリンクが参照するファイルのファイル情報とファイルタイプを評価します。参照先のファイルが存在しない場合、シンボリックリンク自身のファイル情報およびファイルタイプを評価します。コマンド行以外でシンボリックリンクに遭遇した場合、シンボリックリンク自身のファイル情報およびファイルタイプを評価します。 -L シンボリックリンクに遭遇した場合、シンボリックリンク自身ではなく、シンボリックリンクが参照するファイルのファイル情報とファイルタイプを評価します。「注意事項」を参照してください。 相互排他的なオプションである -H と -L を複数指定しても、エラーとはみなされません。最後に指定したオプションが cp の動作を決定します。
オペランド	次のオペランドがサポートされています。 <i>path</i> ディレクトリ階層構造における、下降を開始する地点のパス名を指定します。 <i>expression</i> 第一引数は - で始まるか、あるいは引数自身が ! または (で、後続の引数は以下に説明する一次子と演算子で構成される式として解釈されます。以下の説明で、<i>n</i> がプライマリ引数として使われている場合、それは 10 進整数として解釈されます。この整数には、次に示すように先頭に正 (+) や負 (-) の符号を付加することも可能です。 +<i>n</i> <i>n</i> より大きい <i>n</i> <i>n</i> に等しい -<i>n</i> <i>n</i> より小さい

式

有効な式は次のとおりです。

- acl ファイルに拡張 ACL が定義されている場合、真になります。
- amin *n* ファイルが最後にアクセスされたのは *n* 分前です。
- atime *n* ファイルが *n* 日前にアクセスされている場合、真になります。*path* に含まれているディレクトリのアクセス時間は、find 自体によって変更されます。
- cmin *n* ファイルのステータスが最後に変更されたのは *n* 分前です。
- cpio *device* 常に真です。*device* 上に、現在のファイルを cpio フォーマット (5120 バイトレコード) で書き込みます。
- ctime *n* ファイルのステータスが *n* 日前に変更されている場合、真になります。
- depth 常に真です。ディレクトリ階層を下降して、あるディレクトリにあるすべてのエントリを、そのディレクトリ自体よりも先に検査します。これは、find と cpio(1) を組み合わせて書き込み権のないディレクトリ内のファイルを転送したい場合に役立ちます。
- exec *command* 実行したコマンドが終了ステータスとして 0 の値を返す場合、真です。コマンドの終わりには、コマンドの終了を示すセミコロン (;) を付ける必要があります。コマンド引数 {} は、現在のパス名に置き換えられます。-exec の最後の引数が {} であり、セミコロン (;) ではなくプラス (+) を指定した場合、コマンドが複数回呼び出され、{} がパス名のグループに置き換えられます。コマンドの呼び出しで、終了ステータスとして 0 以外の値を返す場合、find は 0 以外の終了ステータスを返します。
- follow 常に真であり、*expression* のどの位置に指定されていても常に評価されます。find コマンドを -H オプションまたは -L オプションを指定して呼び出すときに、-follow を使用した場合、動作は不確定です。シンボリックリンクをたどり、アクセスしたディレクトリを記憶します。これは、無限ループを検出するためです。無限ループは、シンボリックリンクが親ディレクトリを指している場合などに発生します。この式を find -type l 式と組み合わせて使用しないようにしてください。「注意事項」を参照してください。
- fstype *type* ファイルが属するファイルシステムの形式が *type* の場合、真です。
- group *gname* ファイルが *gname* というグループに属している場合、真です。*gname* が数値で、group(4) データベースに存在しない場合、これはグループ ID と見なされます。

<code>-iname <i>pattern</i></code>	<code>-name</code> と似ていますが、パターンと現在のファイル名のベース名との照合で、大文字と小文字は区別されません(「使用例」を参照してください)。 <code>-name</code> オプションとは異なり、先頭のピリオドは特別に扱われず、 <code>/usr/bin/find</code> と <code>/usr/xpg4/bin/find</code> の両方で、ワイルドカードファイル名生成文字を、 <code>.</code> で始まるファイル名と一致させることができます。
<code>-inum <i>n</i></code>	ファイルが <i>n</i> という <i>i</i> ノード番号を持つ場合、真です。
<code>-links <i>n</i></code>	ファイルにリンクが <i>n</i> 個ある場合、真です。
<code>-local</code>	ファイルシステムが <code>/etc/dfs/fstypes</code> で定義されたりリモートファイルシステムでない場合に、真です。 <code>/etc/dfs/fstypes</code> ファイルが存在しない場合、デフォルトのリモートファイルシステムとして <code>nfs</code> が使用されます。このオプションは、ローカルでないディレクトリの階層を下降して検索します。 <code>-ディレクトリ階層を下降しないでローカルファイルを検索する例</code> については、「使用例」の節を参照してください。
<code>-ls</code>	常に真です。現在のパス名と、このパスの統計情報を出力します。統計情報には以下の項目が含まれています。 ■ <i>i</i> ノード番号 ■ KB (1024 バイト) 単位のサイズ ■ 保護モード ■ ハードリンクの数 ■ <code>user</code> ■ <code>group</code> ■ サイズ (単位はバイト) ■ 変更時間。 ファイルが特殊ファイルの場合、サイズフィールドにはメジャーデバイス番号とマイナーデバイス番号が入ります。 ファイルがシンボリックリンクの場合、<code>'→'</code> の後に、リンクされたファイルのパス名が表示されます。書式は、<code>ls -gilds</code> の書式と同じです (ls(1B) 参照)。 この書式は、<code>ls</code> で設定されているのではなく、<code>find</code> 内部で設定されています。
<code>-mmin <i>n</i></code>	ファイルのデータが最後に変更されたのは <i>n</i> 分前です。
<code>-mount</code>	常に真です。検索の範囲を、指定されたディレクトリを持つファイルシステムに限定します。他のファイルシステムへのマウントポイントは表示しません。
<code>-mtime <i>n</i></code>	ファイルのデータが <i>n</i> 日前に変更されている場合、真です。

- name *pattern*** *pattern* が現在のファイル名のベース名に一致する場合は真。通常のシェルファイル名生成文字 ([sh\(1\)](#)) を参照) を使用できません。 *pattern* 内のエスケープ文字としては、バックスラッシュ (\) を使用します。 *find* をシェルから起動する場合は、 *pattern* をエスケープするか、 *pattern* 全体を引用符で囲む必要があります。
- /usr/bin/find* では、ピリオド文字 (.) を *pattern* の先頭に明示的に指定しない場合、ピリオド文字 (.) で始まる現在のファイル名は *pattern* に一致しません。 */usr/xpg4/bin/find* では、この区別は行われず、ワイルドカードファイル名生成文字はピリオド文字 (.) で始まるファイル名に一致します。
- ncpio *device*** 常に真です。 *device* 上に、現在のファイルを *cpio -c* フォーマット (5120 バイトレコード) で書き込みます。
- newer *file*** 現在のファイルが、 *file* 引数で示されているファイルの更新時刻以降に変更されている場合、真です。
- nogroup** ファイルが [group\(4\)](#) データベースにないグループに属している場合、真です。
- nouser** ファイルが [passwd\(4\)](#) データベースにないユーザーに属している場合、真です。
- ok *command*** 生成されるコマンド行の先頭に疑問符が出力されること、および肯定的な応答の場合にのみ実行されることを除き、 *-exec* と同様です。
- perm [-] *mode*** *mode* 引数は、ファイルモードビットを表します。書式は、 [chmod\(1\)](#) で説明されているシンボリックモードオペランド (*symbolic_mode_list*) と同じで、次のように解釈されます。最初に、すべてのファイルモードビットがオフになっているテンプレートを想定します。演算子記号 (*op*):
- + テンプレート上で、モードビットをオンにします
 - ビットをオフにします
 - = プロセスのファイルモード生成マスクの内容に関係なく、モードビットをオンにします
- このうち - 記号は、 *mode* 引数の最初の文字として記述することはできません。これは、ハイフンではじまるオプションとの混同を避けるためです。初期値としてすべてのモードビットがオフになっているので、最初の文字として - を指定する必要があるシンボリックモードはありません。

ハイフンが省略されると、一次子は、ファイルのアクセス権ビットの値が結果のテンプレートの値と等しければ、真と判定します。

mode 引数がハイフンではじまる場合は、結果のテンプレートのビットがすべてファイルのアクセス権ビットで設定されていれば、真と判定します。

-perm [-]onum ファイルのアクセス権フラグが8進数字 *onum* に正確に一致すれば、真になります ([chmod\(1\)](#) を参照)。*onum* の先頭がマイナス記号(-)の場合、ファイルのアクセス権フラグと比較されるのは、*onum* 中で設定されているビットだけになります。比較の結果、一致すれば、真と判定されます。

-print 常に真です。現在のパス名を出力します。

-print0 常に真です。現在のパス名を出力します。そのあとには、**-print** で使用される NEWLINE 文字ではなく、NULL 文字が続きます。

これにより、NEWLINE またはほかのタイプのスペースを含むファイル名を、**find** の出力を処理するプログラムが正しく解釈できるようになります。このオプションは、**cpio** および **xargs** の **-0** オプションに相当します。

-prune 常に真です。一致するディレクトリ構造内で、*pattern* より下位のディレクトリやファイルは検査しません(「使用例」を参照)。**-depth** が指定されている場合は、**-prune** の指定は無効になります。

-size n[c] ファイルが *n* ブロック長(1ブロックは512バイト)である場合、真です。*n* のあとに *c* を指定すると、サイズはバイト単位になります。

-type c ファイルの形式が *c* である場合、真です。ここで *c* は、**b**(ブロック型特殊ファイル)、**c**(文字型特殊ファイル)、**d**(ディレクトリ)、**D**(door)、**f**(プレーンファイル)、**l**(シンボリックリンク)、**p**(FIFO(名前付きパイプ))、**s**(ソケット)のいずれかです。

-user uname ファイルの所有者が *uname* に指定したユーザーである場合、真です。*uname* が数値で、[passwd\(4\)](#) データベースにログイン名として記載されていない場合、これはユーザー ID と見なされます。

-xdev **-mount** 一次子と同じ。

-xattr ファイルが拡張属性を持っている場合、真です。

複合式

一次子は、以下の演算子(優先度の高い順)を使用して組み合わせることができます。

- 1) *(expression)* 括弧で囲まれた式が真である場合、真 (括弧は、シェルの特殊文字とみなされるため、エスケープする必要がある)
- 2) *!expression* 一次子の否定 (! は単項 *not* 演算子)。
- 3) *expression[-a] expression* 一次子の連結 (*and* 演算子は 2 つの一次子を並置することにより示される)
- 4) *expression-oexpression* 一次子の代替 (*-o* は *or* 演算子)。

`find` を `cpio` と組み合わせて使用する場合、`cpio` に `-L` オプションが指定されているときは、`find` に `-L` オプションまたは `-follow` プリミティブを使用する必要があります。`cpio` に `L` オプションが指定されていない場合は、`find` に `L` オプションまたは `follow` プリミティブを使用してはいけません。そうしない場合、結果は不確定です。

expression を省略すると、式として `-print` が使用されます。*expression* を指定し、その式の中に `-exec`、`-ok`、`-ls`、`-print` のいずれも含まれていなければ、その式は次に示すものに置き換えられます。

(指定された式) `-print`

`-user`、`-group`、`-newer` の各一次子がそれぞれの引数を評価するのは 1 回だけです。`-exec` または `-ok` で指定したコマンドを呼び出しても、同ファイル中の後続の一次子は影響を受けません。

使用法

ファイルが 2G バイト (2³¹ バイト) 以上ある場合の `find` の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 ディレクトリ階層を出力する

次のコマンドは同等です。

```
example% find .
example% find . -print
```

どちらも、現在のディレクトリ以下の階層構造をすべて出力します。

例2 ファイルを削除する

次のコマンドは、ホームディレクトリにある `a.out` または `*.o` という名前のファイルのうち、1 週間アクセスされなかったものをすべて削除します。

```
example% find $HOME \( -name a.out -o -name '*.o' \) \
-atime +7 -exec rm {} \;
```


例3 すべてのファイル名(SCCSディレクトリを除く)を出力する

次の例は、現在のディレクトリとそれ以下のディレクトリ内にあるファイル名すべてを再帰的に出力します。ただし、SCCSディレクトリはスキップします。

```
example% find . -name SCCS -prune -o -print
```

例4 すべてのファイル名とSCCSディレクトリを出力する

次の例は、現在のディレクトリとそれ以下のディレクトリ内にあるファイル名すべてを再帰的に出力します。SCCSディレクトリの内容は出力しませんが、SCCSディレクトリ名は出力します。

```
example% find . -print -name SCCS -prune
```

例5 新しいファイルを検査する

次のコマンドは、-nt を指定した `test(1)` と基本的に同等です。

```
example$ if [ -n "$(find
file1 -prune -newer file2)" ]; then

printf %s\\n "file1 is newer than file2"
```

例6 24時間モードを使用してファイルを選択する

-atime、-ctime、-mtimeで指定する *n* の最小単位は24時間です。たとえば、ある日の23:59にアクセスされたファイルがあり、その2分後、つまり翌日の00:01に次のコマンドを実行したとします。

```
example% find . -atime -1 -print
```

この場合、ファイルは選択の対象となります。日付は変わっていても24時間は経過していないためです。午前0時が間にはさまっていても、24時間を単位とする計算には影響はありません。

例7 指定したファイルアクセス権に一致したファイルを出力する

次の例は、ユーザーによる読み取り・書き込み・実行、グループによる読み取り・実行、その他のユーザーによる読み取り・実行が許可されており、その他のアクセス権は許可されていないファイル名すべてを再帰的に出力します。

```
example% find . -perm u=rwx,g=rx,o=rx
```

次のように指定することもできます。

```
example% find . -perm a=rwx,g-w,o-w
```

例8 その他のユーザーによる書き込みが許可されているファイルを出力する

次の例は、その他のユーザーによる書き込みが許可されているファイル名すべてを再帰的に出力します (読み取り、書き込みが許可されているかどうかは関係ありません)。

```
example% find . -perm -o+w
```

例9 ローカルファイル (ローカルでないディレクトリ階層は下降しない) を出力する

```
example% find . ! -local -prune -o -print
```

例10 拡張属性を持つ名前空間内のファイルを出力する

```
example% find . -xattr
```

例11 大文字と小文字を区別せずにすべての PDF ファイル名を出力する

次の例は、.pdf、.PDF、.Pdf などの拡張子を持つすべてのファイル名を検索します。

```
example% find . -iname '*.pdf'
```

環境変数

find の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

PATH 一次子 -exec と -ok の *utility_name* の場所を判別します。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの *yesexpr* キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、*yesexpr* に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、*yesexpr* に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#) を参照してください。

終了ステータス 次の終了ステータスが返されます。

```
0      オペランドで指定されたパスはすべて正常に検査されました。
>0     エラーが発生した。
```

ファイル	/etc/passwd	パスワードファイル
	/etc/group	グループファイル
	/etc/dfs/fstypes	分散ファイルシステムパッケージを登録したファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[chmod\(1\)](#), [cpio\(1\)](#), [sh\(1\)](#), [test\(1\)](#), [ls\(1B\)](#), [acl\(2\)](#), [stat\(2\)](#), [umask\(2\)](#), [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [locale\(5\)](#), [standards\(5\)](#)

警告

次のオプションは廃止済みで、将来のリリースではサポートされなくなります。

-cpio device 常に真です。device 上に、現在のファイルを cpio フォーマット (5120 バイトレコード) で書き込みます。

-ncpio device 常に真です。device 上に、現在のファイルを cpio -c フォーマット (5120 バイトレコード) で書き込みます。

注意事項

find を使用して、一定時間内に変更されたファイルを判別する場合は、-print 引数の前に -mtime 引数を使用します。そうしないと、すべてのファイル名が出力されます。

Solaris のルートファイルシステム下にあるファイルは、実際には、mntfs や namefs などの仮想ファイルシステムのマウントポイントである場合もあります。ufs ファイルシステムと比較するときに、-mount または -xdev を find 式に指定した場合、このようなファイルは選択されません。

ほかのユーザーの管理下にあるファイルシステム階層を下降する場合に、-L または -follow オプションを使用することはお勧めしません。特に -exec を使用する場合は、シンボリックリンクによって、起動した階層から find コマンドが抜け出す可能性があります。find コマンドによって実行される型チェックと実行されるコマンドがファイル引数に対して操作する時間との間に固有の競合状態があるため、-exec コマンドが操作するファイルタイプを制限するのに -type を使用するのとは十分ではありません。

名前	finger - ローカルユーザーとリモートユーザーに関する情報の表示
形式	<pre>finger [-bfhilmpqsw] [username]... finger [-l] [username@hostname 1 [@hostname 2 .. .@hostname n]...] finger [-l] [@hostname 1 [@hostname 2 .. .@hostname n]...]</pre>
機能説明	引数を省略した場合、finger は複数行のフォーマットでログインしている全ユーザーに関して以下の情報を表示します。 ■ ユーザー名 ■ 正式な氏名 ■ 端末名 (書き込み権が拒否されていると先頭に*が付く) ■ アイドル時間 ■ ログイン時間 ■ リモートでログインしていたらホスト名 アイドル時間には3通りの表記方法があります。数値1つの場合は「分」を表し、2つの数値が:(コロン)をはさんでいる場合は「時」と「分」を表し、2つの数値がdをはさんでいる場合は「日」と「時」を表します。 <i>username</i> 引数を1つ以上指定すると、それらのユーザーに関するより詳細な情報が、当該ユーザーがログインされているいかに関わらず出力されます。<i>username</i> はローカルユーザー名である必要がありますが、そのユーザーの姓でも、名でも、アカウント名でもかまいません。引数の指定により複数行にわたって出力される詳細情報には、以下の内容が含まれます。 ■ 当該ユーザー名と正式名 ■ 当該ユーザーのホームディレクトリとログインシェル ■ 現在ログインされていればそのログイン時間、されていなければ最後にログインした時間。さらにログインを行なった端末またはホスト ■ 当該ユーザーが最後にメールを受信した時間、および最後にメールを読んだ時間 ■ もしあれば、当該ユーザーのホームディレクトリ内の <code>.project</code> ファイルの最初の行 ■ もしあれば、当該ユーザーのホームディレクトリ内の <code>.plan</code> の内容 注: <code>/etc/passwd</code> のコメント (GECOS) フィールドにコンマが組み込まれている場合、finger はそのコンマの後に続く情報を表示しません。 <i>username@hostname1[@hostname2...@hostnamen]</i> または <i>@hostname1[@hostname2...@hostnamen]</i> の形式で引数を指定すると、要求はまず <i>hostnamen</i> に送られ、それぞれ <i>hostname-1</i> を通して <i>hostname1</i> に送られます。本プログラムは finger user information protocol (finger ユーザー情報プロトコル RFC 1288 を参照) を使って、指定されたユーザー (指定されていれば) またはログインし

ているユーザーに関する情報を、指定されたりモートホストに照会します。表示された情報はサーバーによって異なります。

RFC 1288 で規定されるように `finger` は印刷可能である 7 ビットの ASCII データだけを送ります。この動作は、システム管理者が `/etc/default/finger` に `PASS` オプションを使用することによって修正できます。`PASS=low` を指定すると、10 進数 32 ASCII よりも小さい文字をすべて送ります。`PASS=high` を指定すると、10 進数 126 ASCII よりも大きい文字をすべて送ります。`PASS=low,high` または `PASS=high,low` を指定すると、32 ASCII よりも小さい文字と 126 ASCII よりも大きい文字をすべて送ります。

オプション

`username@hostname` 形式では `-l` オプションのみがサポートされている以外は、次のオプションがサポートされています。

- b 長形式の出力において、ユーザーのホームディレクトリとシェルの出力を抑制します。
- f 長形式以外の出力において、通常出力されるヘッダーの出力を抑制します。
- h 長形式の出力において、`.project` ファイルの出力を抑制します。
- i アイドル形式の出力を生成します。短形式との違いは、ログイン名、端末、ログイン時間、およびアイドル時間のみが出力されることです。
- l 長形式の出力を生成します。
- m 引数をユーザー名 (姓や名ではない) と見なしてマッチングを行います。
- p 長形式の出力において、`.plan` ファイルの出力を抑制します。
- q クイック形式の出力を生成します。短形式との違いは、ログイン名、端末、およびログイン時間のみが出力されることです。
- s 短形式の出力を生成します。
- w 短形式の出力において、正式な氏名の出力を抑制します。

ファイル

<code>\$HOME/.plan</code>	ユーザーの計画
<code>\$HOME/.project</code>	ユーザーのプロジェクト
<code>/etc/default/finger</code>	<code>finger</code> オプションのファイル
<code>/etc/passwd</code>	パスワードのファイル
<code>/var/adm/lastlog</code>	最後にログインした時間
<code>/var/adm/utmpx</code>	アカウンティング

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/network/network-servers

関連項目 [passwd\(1\)](#), [who\(1\)](#), [whois\(1\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

『RFC 1288, The Finger User Information Protocol』、Zimmerman, D. 著、Center for Discrete Mathematics and Theoretical Computer Science (DIMACS)、Rutgers University、1991 年 12 月

注意事項 本コマンドをリモート形式で実行する場合、使用可能なオプションは `finger user information protocol` (finger ユーザー情報プロトコル) により制限されます。

名前	fmt – 簡単なテキストフォーマッタ
形式	fmt [-cs] [-w <i>width</i> -width] [<i>inputfile</i>]...
機能説明	fmt は簡単なテキストフォーマッタで、行を詰めたり結合することによって -w <i>width</i> オプションで指定した文字数までの出力行を作成します。デフォルトの <i>width</i> は 72 カラムです。fmt は引数に指定した <i>inputfile</i> を結合します。inputfile を指定しなかった場合、fmt は標準入力からのテキストを書式化します。 空白行は、単語間の空白としてそのまま出力されます。fmt は、nroff(1) との互換性のため、.(ドット)で始まる行を詰めたり結合することはありません。また、メールヘッダーと判別された(つまり、最初の行が "From" で始まる)連続する空白以外の行も詰めたり結合することはありません。 インデントはそのまま出力されます。インデントが異なる入力行は結合されません (-c オプションを使用しない場合)。 また、fmt は、vi(1) のインラインテキストフィルタとしても使用できます。次の vi コマンドを入力します。

```
!}fmt
```

すると、カーソルの位置から段落の最後までのテキストが書式化されます。

オプション	-c クラウンマージンモード。1つの段落内の最初の2行のインデントを保持し、後続行の左マージンを2番目の行に揃えます。タブ付き段落に便利です。 -s 行の分割のみ。短い行を連結してより長い行にはしません。これにより、コード例などの書式化されたテキストが不当に連結されるのを防ぐことができます。 -w <i>width</i> -width 出力行を <i>width</i> カラムの幅にします。

オペランド	<i>inputfile</i> 入力ファイル

環境変数	fmt の実行に影響を与える環境変数 LC_CTYPE については、 environ(5) を参照してください。

属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目	nroff(1) , vi(1) , attributes(5) , environ(5)

注意事項

-width オプションはBSD 互換性のために提供されていますが、将来のリリースでは廃止される可能性があります。

名前	fmtmsg - stderr またはシステムコンソールへのメッセージ表示
形式	fmtmsg [-c <i>class</i>] [-u <i>subclass</i>] [-l <i>label</i>] [-s <i>severity</i>] [-t <i>tag</i>] [-a <i>action</i>] <i>text</i>
機能説明	fmtmsg は、メッセージの分類コンポーネントに基づいて、書式付きメッセージを stderr またはコンソールに書き出します。 書式付きメッセージは、標準コンポーネントで構成されます (「環境」の項の環境変数 MSGVERB を参照)。このうち分類コンポーネントとサブクラスコンポーネントの 2 つは、標準メッセージの一部としては表示されません。これらはメッセージの発行元を定義し、書式付きメッセージの表示先を指定するために使われます。
オプション	サポートしているオプションは、次のとおりです。 -c <i>class</i> メッセージ発行の原因となった状態の種類を表します。以下のキーワードが指定可能です。 hard 状態の発生元はハードウェア soft 状態の発生元はソフトウェア firm 状態の発生元はファームウェア -u <i>subclass</i> メッセージをさらに定義しメッセージの表示先を示すための、一連のキーワードを <i>subclass</i> に指定します。キーワード間はコンマで区切ります。以下のキーワードが指定可能です。 appl 状態の発生はアプリケーション中に起因しています。このキーワードは util および opsys と同時には指定できません。 util 状態の発生はユーティリティ中に起因しています。このキーワードは appl および opsys と同時には指定できません。 opsys 状態の発生はカーネル中に起因しています。このキーワードは appl および util と同時には指定できません。 recov アプリケーションをその状態から回復させます。このキーワードは nrecov と同時には指定できません。 nrecov アプリケーションをその状態から回復させません。このキーワードは recov と同時には指定できません。 print 標準エラー스트リーム stderr にメッセージを書き出します。 console システムコンソールにメッセージを書き出します。print と console を両方指定することも可能です。 -l <i>label</i> メッセージの発行元を指定します。

-s severity	エラーの重大度を指定します。標準レベルの重大度として <i>severity</i> に指定できるキーワードとその意味を以下に説明します。
halt	アプリケーションが重大なエラーを検出し、処理を中断しています。
error	アプリケーションがエラーを検出しました。
warn	アプリケーションが、通常は起こらない状態を検出しました。エラーの可能性あります。
info	アプリケーションは、エラーではない単なる情報を通知しています。
-t tag	メッセージの識別子を含んだ文字列を指定します。
-a action	エラー回復処理の第1ステップを述べた文字列を指定します。 <i>action</i> 引数は、それ全体が1つの文字列であると正しく解釈されるように記述しなければなりません。fmtmsg は、この <i>action</i> で指定した文字列の前に TO FIX: を付加します。
text	状態を表すテキスト文字列を指定します。 <i>text</i> 引数は、それ全体が1つの文字列であると正しく解釈されるように記述しなければなりません。

使用例

例1 標準メッセージ形式

次に示す fmtmsg は、標準メッセージ形式で完全なメッセージを生成し、それを標準エラー 스트リームに書き出す例です。

```
example% fmtmsg -c soft -u recov,print,appl -l UX:cat \
-s error -t UX:cat:001 -a "refer to manual" "invalid syntax"
```

これを実行すると次の出力が得られます。

```
UX:cat: ERROR: invalid syntax
TO FIX: refer to manual    UX:cat:138
```

例2 MSGVERB を使用する

環境変数 MSGVERB が次のように設定されているとします。

```
MSGVERB=severity:text:action
```

このとき前述の例1の fmtmsg コマンドを実行すると、以下の出力が得られます。

```
ERROR: invalid syntax
TO FIX: refer to manual
```

例3 SEV_LEVELを使用する

環境変数 SEV_LEVEL が次のように設定されているとします。

```
SEV_LEVEL=note,5,NOTE
```

このとき次の fmtmsg コマンドを実行します。

```
example% fmtmsg -c soft -u print -l UX:cat -s note \
-a "refer to manual" "invalid syntax"
```

これを実行すると次の出力が得られます。

```
NOTE: invalid syntax
TO FIX: refer to manual
```

また、stderr にメッセージが出力されます。

環境変数

fmtmsg の動作は、2つの環境変数 MSGVERB と SEV_LEVEL により制御されています。MSGVERB はシステム管理者によって、システム用の /etc/profile 中に設定されています。ユーザーは、このシステムに設定された値以外の MSGVERB 値を使用することができます。その方法としては、ユーザーの .profile ファイル中にある MSGVERB をリセットする、現在のシェルセッション内で値を変更する、の2通りがあります。SEV_LEVEL はシェルスクリプト中で使用できます。

メッセージを stderr に出力するとき、どのコンポーネントを選択すべきかを MSGVERB が fmtmsg に伝えます。MSGVERB の値は一連のキーワードで、キーワード間はコロンで区切ります。MSGVERB は次に示す形式で設定できます。

```
MSGVERB=[keyword[:keyword[:...]]]
export MSGVERB
```

keywords としては、label、severity、text、action、tag が指定できます。MSGVERB がコンポーネント用のキーワードを含んでいて、そのコンポーネントの値が NULL 文字列ではない場合、fmtmsg はメッセージを stderr に書き出す際にそのコンポーネントをメッセージ中に挿入します。MSGVERB がコンポーネント用のキーワードを含んでいなければ、そのコンポーネントはメッセージ表示には含まれません。前述のキーワードは任意の順序で指定できます。MSGVERB が定義されていない、値が NULL 文字列である、値の形式が正しくない、前述のもの以外のキーワードを含んでいる、といった場合には、fmtmsg はすべてのコンポーネントを選択します。

MSGVERB は、どのコンポーネントを選択するかを表しますが、これは表示用のメッセージに関してだけです。コンソールへ出力されるメッセージには、常にすべてのコンポーネントが含まれます。

SEV_LEVEL は重大度レベルを定義し、fmtmsg で使用できるように印刷文字列をそれらのレベルに対応させます。なお以下に示す標準重大度レベルは、変更することはできません。これ以外のレベルは、定義したり、再定義したり、削除したりすることが可能です。

0 (重大度は使用されません)

1 HALT

2 ERROR

3 WARNING

4 INFO

SEV_LEVEL は次の形式で設定できます。

description は3つのフィールドで構成され、フィールド間はコンマで区切ります。

```
SEV_LEVEL= [description[:description[:...]]]
export SEV_LEVEL
```

description=severity_keyword, level, printstring

最初のフィールド *severity_keyword* は、fmtmsg の `-s severity` オプションで指定できるキーワードのいずれかと同じ文字列です。

次のフィールド *level* は、評価の結果が正の整数となる文字列です。ただしその整数値は、標準重大度レベルとして予約されている 0、1、2、3、または 4 であってはいけません。キーワード *severity_keyword* が指定されていると、*level* の値は重大度の値として [fmtmsg\(3C\)](#) に渡されます。

3 番目のフィールド *printstring* は、前述の重大度値 *level* が用いられる際に fmtmsg が標準メッセージ形式で使用する文字列です。

SEV_LEVEL が定義されていない場合、またはその値が NULL 文字列の場合には、デフォルトの重大度レベル値だけが使用可能です。コロンで区切られた一連の *description* の中に、その内容がコンマで区切られた3つのフィールドで構成されていないものがあるとき、または第2フィールドの値が正の整数に評価されないものがあるとき、その *description* は無視されます。

終了ステータス 次の終了ステータスが返されます。

0 指定された機能はすべて正常に実行されました。

1 コマンド中に、構文の誤り、不正なオプション、またはオプションに対する不正な引数が検出されました。

2 機能は部分的に正常終了したが、メッセージは `stderr` に出力されませんでした。

- 4 機能は部分的に正常終了したが、メッセージはシステムコンソールに出力されませんでした。
- 32 要求された機能はいずれも正常に終了しませんでした。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

```
addseverity(3C), fmtmsg(3C), attributes(5)
```

名前	fold - 行の折り返し用フィルタ
形式	fold [-bs] [-w <i>width</i> -width] [<i>file</i>]...
機能説明	fold ユーティリティーは、入力ファイルから読み込んだ各行を、行の最大長が <i>width</i> で示すカラム位置 (-b オプション指定時はバイト数) になるように分割して複数行に表示するフィルタです。各出力行 (このセクションでは以降、セグメントと呼ぶ) が指定されたカラム位置 (またはバイト数) を超えない最大長になるように、NEWLINE 文字を挿入して行が分割されます。行が文字の途中で分割されることはありません。入力中の 1 つの文字が占有するカラム数より小さい値を <i>width</i> に指定した場合、その結果は予測できません。 入力中に CARRIAGE-RETURN、BACKSPACE、または TAB 文字が検出され、-b オプションが指定されていない場合、特殊な処理が行われます。 BACKSPACE 現在の行の幅の数が 1 だけ減らされます。ただし負になることはありません。fold は、BACKSPACE 文字の直前や直後に NEWLINE 文字を挿入することはありません。 CARRIAGE-RETURN 現在の行の幅の数が 0 に設定されます。fold は、CARRIAGE-RETURN 文字の直前や直後に NEWLINE 文字を挿入することはありません。 TAB 検出された各 TAB 文字は、カラム位置ポインタを次のタブストップへ進めます。タブストップは、各カラム位置 <i>n</i> (<i>n</i> は 8 で割った剰余が 1 に等しくなる数) にあります。
オプション	サポートしているオプションは、次のとおりです。 -b <i>width</i> の値はカラム数ではなくバイト数で示されていることを指定します。 -s 入力行に空白文字が含まれている場合、出力セグメントの長さが <i>width</i> カラムまたはバイトを超えない範囲で、最後の空白文字の直後で行を分割します。要件を満たす空白文字が存在しなければ、-s オプションは入力行のその出力セグメントに対して効果がありません。 -w <i>width</i> -width 出力セグメントの最大長をカラム数 (-b 指定時はバイト数) で指定します。<i>width</i> の値が正の整数でないと、エラーが発生します。デフォルトの値は 80 です。
オペランド	次のオペランドを指定できます。 <i>file</i> 行の折り返しを行うテキストファイルのパス名。<i>file</i> オペランドを指定しないと、標準入力を使用されます。

環境変数 `fold` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 指定されたファイルはすべて正常に処理されました。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [cut\(1\)](#), [pr\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 `fold` と [cut\(1\)](#) は、どちらも長い行を持つファイルから新たなテキストファイルを生成する目的で使われます。`fold` は行の内容を連続させるべき場合に使用するのに対して、`cut` は行 (またはレコード) の数を一定にしておく場合に使用します。

`fold` は、最大印刷幅 (通常は 80 または 132 カラム) を超える行は切り捨ててしまうようなプリンタにテキストファイルを出力させるためによく使われます。

下線が存在する場合には、`fold` は正しく動作しないことがあります。

名前	from – display the sender and date of newly-arrived mail messages
形式	<code>/usr/ucb/from [-s <i>sender</i>] [<i>username</i>]</code>
機能説明	The <code>from</code> utility prints out the mail header lines in your mailbox file to show you who your mail is from. If <i>username</i> is specified, <i>username</i> 's mailbox is examined instead of your own.
オプション	The following option is supported: <code>-s <i>sender</i></code> Only display headers for mail sent by <i>sender</i> .
使用法	See largefile(5) for the description of the behavior of <code>from</code> when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
ファイル	<code>/var/mail/*</code>
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [biff\(1B\)](#), [mail\(1B\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

名前	ftp - ファイル転送プログラム
形式	ftp [-adfginpstvx] [-m <i>GSS Mech</i>] [-T <i>timeout</i>] [<i>hostname</i> [<i>port</i>]]
機能説明	ftp コマンドは、インターネット標準ファイル転送プロトコル (FTP) のユーザーインタフェースとして機能します。このコマンドを使用すると、リモートネットワークサイトとファイルをやりとりすることができます。 ftp の通信相手であるホストおよびポート (任意指定) は、コマンド行に指定することができます。クライアントホスト名をコマンド行に指定した場合 ftp は、ただちにそのホストの FTP サーバーとの接続を確立しようとします。ホスト名がコマンド行で指定されなかった場合は、コマンドインタプリタに制御が移り、ユーザーからの指示を待ちます。指示を待っている間、ftp> というプロンプトが表示されます。
オプション	以下に示すオプションは、コマンド行中に指定することも、コマンドインタプリタに対して指定することもできます。 -a GSSAPI 認証だけを使用します。この認証に失敗した場合、接続を閉じます。 -d デバッグ処理を実施します。 -f ローカルのセキュリティ証明書をリモートサーバーに転送します。 -g ファイル名の "グロビング" (展開) 機能を無効にします。 -i 複数ファイルの転送中に、対話形式のプロンプトを出力しないようにします。 -m GSS-API メカニズムを使用することを指定します。デフォルトでは、kerberos_v5 メカニズムを使用します。サポートされる代替機構は、/etc/gss/mech に定義されています (mech(4) を参照)。 -n 初期接続時に "自動ログイン" を行いません。自動ログインが有効になっている場合、ftp はユーザーのホームディレクトリ内の .netrc ファイルを検査し、リモートマシンにログインするアカウントが記述されているかどうかを確認し、アカウントが見つからない場合は、リモートマシン上のアカウントのログイン名の入力を要求します (デフォルトはローカルマシン上のログイン名)。さらに、もし必要であれば、パスワードとログインするアカウントの入力も要求します。 -p データ転送を受動モードで行います。このオプションは、ファイアウォールを介して接続されているリモートホストと通信する場合に便利です。 -s 接続時にデフォルトですべてのリモートサーバーに送信される SYST コマンドをスキップします。このシステムコマンドは、プロトコルでデフォルトの ASCII モードではなくバイナリモードを自動的に使用できるようにします。

一部の古いサーバーでは `ftp` コマンドを処理できないため、そのようなサーバーとの相互運用性のためにこの指令が提供されています。

- t パケットトレース機能を有効にします (この機能はまだ実装されていません)。
- T *timeout* 大域接続タイマーを有効にします (秒単位、10 進数で指定)。タイマーには、サーバーにデータが送信されるとリセットされ、クライアントシステムにユーザーの入力を求めるプロンプトが表示されている間は無効になるという制御接続用のタイマーと、着信方向または発信方向のデータ接続を監視するためのタイマーの 2 種類があります。
- v データ転送統計情報のレポートとともに、リモートサーバーからのすべての応答を表示します。ftp が対話形式で動作している場合は、この機能はデフォルトで有効になります。
- x 認証と暗号化に GSSAPI の使用を試みます。データとコマンドのチャンネル保護は「private」に設定されます。

以下に示すコマンドは、コマンドインタプリタに対して指定することができます。

- ! *command* で指定したコマンドを、ローカルマシン上でシェルコマンドとして実行します。*command* を省略した場合、対話型シェルが呼び出されます。
- \$ *macro-name* [*args*] *macro-name* で指定したマクロを実行します。このマクロは `macdef` コマンドで定義したものです。引数 *args* は、そのままマクロに渡されます。
- account [*passwd*] リモートシステムに正常にログイン完了した後、リソースのアクセスに必要な補助パスワードを指定します。*password* を省略した場合、ユーザーはアカウントのパスワードを入力するように要求されます。このとき、入力したパスワードは画面上には表示されません。
- append *local-file* [*remote-file*] [*remote-file*] *local-file* で指定したローカルファイルを、リモートマシン上の *remote-file* で指定したファイルの末尾に追加します。*remote-file* を省略した場合、ローカルファイル名が `ntrans` または `nmap` の設定に従って変更され、リモートファイル名として使用されます。ファイル転送時、“表現タイプ”、“ファイル構造”、“転送モード”には、現在の設定値が使用されます。

<code>ascii</code>	“表現タイプ”を、デフォルトの“ネットワーク ASCII”に設定します。これがデフォルトです。
<code>bell</code>	各ファイル転送コマンドが終了するたびに、ベルを鳴らします。
<code>binary</code>	“表現タイプ”を、“イメージ”に設定します。
<code>bye</code>	リモートサーバーとのFTPセッションを終了し、 <code>ftp</code> コマンドの実行を終了します。なお EOF (ファイルの終わり) に到達した場合もセッションを終了し、コマンドの実行を終了します。
<code>case</code>	<code>mget</code> コマンドを実行している間、リモートコンピュータ上にあるファイル名の大文字と小文字を対応づける機能を有効にします (デフォルトでは無効になっています)。この機能を有効にすると、英大文字の名前を持つリモートコンピュータ上のファイルは、英小文字の名前でローカルディレクトリに書き込まれます。
<code>cd remote-directory</code>	リモートマシン上の作業ディレクトリを、 <i>remote-directory</i> で指定したディレクトリに変更します。
<code>cdup</code>	リモートマシン上の作業ディレクトリを、現在の作業ディレクトリの親ディレクトリに変更します。
<code>clear</code>	データ転送の保護レベルを「 <code>clear</code> 」に設定します。ADAT コマンドが失敗した場合、これがデフォルトの保護レベルになります。
<code>close</code>	リモートサーバーとのFTPセッションを終了し、コマンドインタプリタに戻ります。定義されたマクロがあれば削除されます。
<code>cr</code>	“ネットワーク ASCII” タイプのファイル検索中に復帰改行 (RETURN) を除去する処理のオン・オフの切り替えを行います。“ネットワーク ASCII” タイプのファイル転送時には、復帰改行/改行 (RETURN/LINEFEED シーケンス) によってレコードが認識されます。 <code>cr</code> がオン (デフォルト) のとき、復帰改行はこのシーケンスから取り除かれます。これにより、改行のみでレコードを区切っている UNIX システムの仕様に準拠できます。UNIX 以外のシステムのリモートホスト上

	のレコードには、改行のみを区切り記号として使用している場合があります。“ネットワーク ASCII”タイプの転送を行なったとき、これらの改行文字は <code>cr</code> がオフの場合にのみレコード区切り記号とは区別して扱われます。
<code>delete remote-file</code>	リモートマシン上の <i>remote-file</i> で示したファイルを削除します。
<code>debug</code>	デバッグモードのオン・オフの切り替えを行います。デバッグモードがオンのとき、ftp はリモートマシンに送られたコマンドを、先頭に <code>-></code> を付加して印刷します。
<code>dir [remote-directory [local-file]]</code>	<i>remote-directory</i> で示すリモートディレクトリの内容を出力します。出力先は <i>local-file</i> で示したローカルファイルです。 <i>remote-directory</i> を省略した場合は、現在の作業ディレクトリの内容を出力します。ローカルファイルを省略した場合、または <i>local-file</i> として <code>-</code> を指定した場合は、出力先は端末となります。
<code>disconnect</code>	<code>close</code> と同一機能です。
<code>form [format-name]</code>	“表現タイプ”のうちのキャリッジ制御フォーマットサブタイプを、 <i>format-name</i> で示したフォーマットに設定します。なお <i>format-name</i> として指定可能なものは <code>non-print</code> だけです。これはデフォルトの“ノンプリント”サブタイプに対応します。
<code>get remote-file [local-file]</code>	<i>remote-file</i> で示したリモートファイルを取り出し、 <i>local-file</i> で示した名前でローカルマシン上に格納します。 <i>local-file</i> を省略した場合、リモートマシン上のファイル名が <code>case</code> 、 <code>ntrans</code> 、 <code>nmap</code> 設定に従って変更され、ローカルファイル名として使用されます。ファイル転送時、“表現タイプ”、“ファイル構造”、“転送モード”に関しては、現在の設定値が用いられます。
<code>glob</code>	<code>mdelete</code> 、 <code>mget</code> 、 <code>mput</code> 時のファイル名の展開、あるいは“グローピング”機能のオン・オフの切り替えを行います。グローピング機能をオフにすると、ファイル名は展開されずそのまま使用されます。

mput 時のグロビングは、[sh\(1\)](#)での処理と同一です。mdelete と mget に関しては、各リモートファイル名はリモートマシン上で個々に展開され、リストはマージされません。

ディレクトリ名の展開は、通常ファイル名の展開とはかなり異なります。すなわちグロビングの結果は、リモートオペレーティングシステムと FTP サーバーにより決まります。その値は `mls remote-files` コマンドの実行によりあらかじめ得ることができます。

mget と mput は、ファイルのディレクトリサブツリーをすべて転送するためのものではありません。そのような転送を行いたいときは、[tar\(1\)](#) アーカイブをサブツリーに転送してください(その際には `binary` コマンドの実行で設定される“イメージ”という“表現タイプ”を使用してください)。

hash

個々のデータブロックを転送するたびにハッシュサイン (#) を印刷する機能のオン・オフの切り替えを行います。各データブロックのサイズは 8192 バイトです。

help [*command*]

command で示したコマンドの意味を説明するメッセージを出力します。*command* を省略した場合は、定義されているコマンドの一覧表が出力されます。

lcd [*directory*]

ローカルマシン上の作業ディレクトリを、*directory* で示したディレクトリに変更します。*directory* を省略した場合は、ユーザーのホームディレクトリが指定されたものとみなされます。

ls [-al | *remote-directory* [*local-file*]]

デフォルトでは *remote-directory* で示すリモートファイル上のディレクトリの内容を、要約した形式で出力します。このデフォルトの動作は `ls` コマンドが `dir` コマンドと同じになるように変更できます。この変更を行うには、`/etc/default/ftp` または環境において、`FTP_LS_SENDS_NLST` を「no」に設定します。詳細については、[ftp\(4\)](#) を参照してください。

-a オプションは、すべてのファイルを表示します。通常は表示されないドット(.)で始まるファイルも表示されます。-l オプションは、ファイルを詳細に表示します。各ファイルのモード、リンク数、所有者、グループ、サイズ(バイト数)、最後修正日時が表示されます。ファイルが特殊ファイルの場合、サイズフィールドには、サイズではなく、メジャーデバイス番号とマイナーデバイス番号が表示されます。ファイルがシンボリックリンクの場合は、ファイル名を出力したあとに、→に続いて参照されるファイルのパス名が出力されます。

remote-directory を省略した場合は、現在の作業ディレクトリの内容が出力されます。

ローカルファイルを省略した場合、または *local-file* として - を指定した場合は、出力先は端末となります。

`macdef macro-name`

マクロを定義します。後続の行で指定された内容が、*macro-name* という名前のマクロとして格納されます。マクロ定義の終わりは空の (NULL) 行 (ファイル中では連続した復帰改行 (NEWLINE) 文字、端末入力では連続した復帰改行 (RETURN) 文字) によって表されます。定義可能なマクロ数は最大 16 個、その合計文字数は最大 4096 文字です。いったん定義されたマクロは、`close` コマンドが実行されるまで定義されたままとなります。

マクロプロセッサは、\$ と \ を特殊文字として解釈します。\$ の後に 1 個 (または複数個) の数値を指定すると、それは、マクロ呼び出しコマンド行内の対応する引数に置き換えられます。\$ の後に文字 *i* を指定すると、そのマクロはループされる旨がマクロプロセッサに通知されます。つまり最初のパスでは \$*i* はマクロ呼び出しコマンド行内の第 1 引数で置き換えられ、2 回目の実行時には第 2 引数で置き換えられ、という方法で繰り返されます。\' の後に何らかの文字を指定すると、\' はその文字で置き換えられます。これにより、\$ を特殊文字ではなく通常の文字として

使用することができます。

`mdelete remote-files`

リモートマシン上にある、*remote-files* で示すファイルを削除します。

`mdir remote-files local-file`

`dir` と同機能ですが、複数のリモートファイル名を指定できる点が異なります。対話形式のプロンプトが有効な場合、最後の引数が `mdir` の出力先としてのターゲットのローカルファイルであることを確認するためのプロンプトを出力します。

`mget remote-files`

リモートマシン上にある、*remote-files* で示すファイル名を展開し、それにより生成された名前を持つ各ファイルに対して `get` を実行します。ファイル名の展開の詳細については、`glob` の説明を参照してください。生成されたファイル名は、`case`、`ntrans`、`nmap` の設定値に従って処理されます。ファイルはローカル作業ディレクトリに転送されます。ローカル作業ディレクトリは `lcd directory` により変更できます。新たなローカルディレクトリを作成するには、`!mkdir directory` を使用してください。

`mkdir directory-name`

`directory-name` で示したディレクトリをリモートマシン上に生成します。

`mls remote-files local-file`

`ls(1)` と同機能ですが、複数のリモートファイル名を指定できる点が異なります。対話形式のプロンプトが有効な場合、最後の引数が `mls` の出力先としてのターゲットのローカルファイルであることを確認するためのプロンプトを出力します。

`mode [ mode-name ]`

“転送モード” を、*mode-name* に設定します。なお *mode-name* として指定可能なものは `stream` だけです。これはデフォルトの“ストリーム”モードに対応します。現在のシステムでサポートしているモードは `stream` だけなので、これを指定しなければなりません。

`mput local-files`

`local-files` で示したローカルファイル名のリスト中のワイルドカードを展開し、それにより生成された名前を持つ各ファイルに対して `put` を実行します。ファイル名展開の詳細については `glob` の説明を参照してください。生成されたファイル名は、`ntrans` および `nmap` の設定値に従って処理されます。

`nlist [-al] remote-directory [local-file]`

リモートマシン上のディレクトリの内容を簡略に表示します。`-a` オプションまたは `-l` オプションを指定しない限り、表示されるファイルは `get` コマンドで取得できるファイルだけです。`remote-directory` を省略した場合は、現在の作業ディレクトリの内容が出力されます。

`-a` オプションは、すべてのファイルを表示します。通常は表示されないドット (.) で始まるファイルも表示されます。`-l` オプションは、ファイルを詳細に表示します。`ls` コマンドの `l` オプションと同じです。

`nmap [inpattern outpattern]`

ファイル名マッピングメカニズムを設定または解除します。引数を省略した場合は、解除となります。引数を指定した場合は、リモートファイル名は「対象のリモートファイル名指定のない `mput` および `put` コマンド」の実行時にマップされます。同様にローカルファイル名は「対象のローカルファイル名指定のない `mget` および `get` コマンド」の実行時にマップされます。

このコマンドは、異なったファイル名規約を持つ、UNIX 以外のシステムのリモートホストに接続する場合に便利です。マッピング処理は、`inpattern` と `outpattern` で設定されたパターンに従って行われます。`inpattern` は入力ファイル名 (すでに `ntrans` および `case` の設定により処理されている場合もある) 用のテンプレートです。`inpattern` 中に `$1`、`$2` ...、`$9` シーケンスを含めることによ

り、変数を使用したテンプレート処理も可能です。`$`を特殊文字ではなく通常の文字として扱う必要があれば、`\`を使用してください。その他の文字はすべて入力されたとおりに処理され、`nmap`の *inpattern* 変数の値を決定するために用いられます。

たとえば、*inpattern* として `$1.$2` を指定し、リモートファイル名が `mydata.data` のとき、`$1` の値は `mydata` に、`$2` の値は `data` になります。

outpattern は、マッピングの結果生成されるファイル名を決定しま

す。`$1`、`$2`、...、`$9` というシーケンスを指定すれば、*inpattern* のテンプレートから得られる値により置き換えられます。

シーケンス `$0` は、元のファイル名に置き換えられます。さらに、`[seq1, seq2]` というシーケンスは、`seq1` が `NULL` でなければ `seq1` に、`NULL` であれば `seq2` に置き換えられます。

たとえば、`nmap $1.$2.$3`

`[$1,$2].[$2,file]` というコマンドを実行すると、入力ファイル名が `myfile.data` または `myfile.data.old` のときは出力ファイル名が `myfile.data` になり、`myfile` のときは `myfile.file` になり、`.myfile` のときは `myfile.myfile` になります。また、たとえば `nmap $1 | sed "s/ *$//" > $1` のように、*outpattern* 中にスペース文字が含まれていても構いません。`$`、`[`、`]`、`,` の4つの各文字を、特殊文字ではなく通常の文字として扱う必要があれば、`\`を使用してください。

`ntrans [ inchars [ outchars ] ]`

ファイル名の文字変換メカニズムを設定または解除します。引数を省略した場合は、解除となります。引数を指定した場合は、リモートファイル名を構成する文字は「対象のリモートファイル名指定のない `mput` および `put` コマンド」の実行時に変換されます。同様にローカルファイ

ル名の文字は「対象のローカルファイル名指定のない `mget` および `get` コマンド」の実行時に変換されます。

このコマンドは、異なったファイル名規約を持つ、UNIX 以外のシステムのリモートホストに接続する場合に便利です。 `inchars` で指定した文字がファイル名に含まれていると、その文字は `outchars` 中の対応する文字に変換されます。 `inchars` 内におけるその文字の位置が `outchars` の長さを超えている場合、その文字はファイル名から削除されます。

`ntrans` コマンドを `ftp` で使用すると 16 文字のみを変換します。全アルファベットを転換する場合には、`case` (前述参照) を使います。

`open host [ port ]`

`host` が示すホストの FTP サーバーとの接続を確立します。オプションであるポート番号を指定すると、`ftp` はそのポートにおいて FTP サーバーと接続しようと試みます。また自動ログインオプションが有効 (デフォルト値) のとき、`ftp` はユーザーを自動的に FTP サーバーにログインしようと試みます。

`passive`

受動モードのオン・オフを切り替えます。受動モードがオンの場合、`ftp` クライアントは、データ接続用のポートをオープンし、そのポートのアドレスを返すことを要求する `PASV` コマンドを FTP サーバーに送信します。リモートサーバーはそのポートで待機し、クライアントはそのポートに接続します。受動モードがオフの場合、`ftp` クライアントは、リモートサーバーが接続するべきアドレスを指定する `PORT` コマンドを送信します。受動モードは、`ftp` クライアントへの接続が何らかの形で制御されている場合 (例: ファイアウォール配下にある場合など) に便利です。IPv6 対応の FTP サーバーに接続する場合は、`PASV` の代わ

	りに EPSV、PORT の代わりに EPRT が使用される場合があります。
<code>private</code>	データ転送の保護レベルを「private」に設定します。データの転送の信頼性(と整合性)は暗号化によって保護されます。ADAT コマンドが失敗した場合、データ転送の保護レベルは「clear」になります。
<code>prompt</code>	対話形式のプロンプトのオン・オフの切り替えを行います。オンに設定すれば、複数個のファイル転送を行う場合、読み書きの対象とするファイルを個別に選択することが可能となりますデフォルトはオンです。オフのときは、 <code>mget</code> や <code>mput</code> コマンドはすべてのファイルを転送し、 <code>mdelete</code> コマンドはすべてのファイルを削除します。
<code>protect protection-level</code>	データ転送の保護レベルを <i>protection-level</i> に設定します。有効な保護レベルは、「clear」(データ転送を保護しない)、「safe」(データ転送の整合性を暗号チェックサムで保護する)、「private」(データ転送の信頼性と整合性を暗号化で保護する)です。ADAT コマンドが失敗した場合、データ転送の保護レベルは「clear」のみになります。保護レベルを指定しない場合、現在の保護レベルが表示されます。デフォルトの保護レベルは「clear」です。
<code>proxy ftp-command</code>	セカンダリ制御接続上で FTP コマンドを実行します。すなわちこのコマンドは、2つのリモート FTP サーバーに対して同時に接続を確立し、それらのサーバー間での転送を可能とします。最初に発行する <code>proxy</code> コマンドは、セカンダリ制御接続を確立するための <code>open</code> コマンドでなければなりません。セカンダリ接続上で実行可能なその他のコマンドを知るには、 <code>proxy ?</code> コマンドを使用してください。
	以下に述べるコマンドは、 <code>proxy</code> を伴って入力された場合、次のように、通常とは

異なった動作をします。open は自動ログイン処理中には新たなマクロの定義は行いません。close は既存のマクロの削除は行いません。get および mget はプライマリ制御接続上のホストからセカンダリ制御接続上のホストへファイルを転送します。put、mput、および append は、セカンダリ制御接続上のホストからプライマリ制御接続上のホストへファイルを転送します。

第三者によるファイル転送は、セカンダリ制御接続上のサーバーが PASV コマンドをサポートしているか否かにより異なります。

put *local-file* [*remote-file*]

local-file で示すローカルファイルを、*remote-file* という名でリモートマシン上に格納します。*remote-file* を省略した場合は、ローカルファイル名が ntrans または nmap 設定に従って変更され、リモートファイル名として使用されます。ファイル転送時、“表現タイプ”、“ファイル構造”、“転送モード”には、現在の設定値が使用されます。

pwd

現在の作業ディレクトリの名前をリモートマシン上に出力します。

quit

bye と同一機能です。

quote *arg1 arg2* ...

指定された引数をそのままリモート FTP サーバーに送信します。FTP からは応答コードが 1 つ返されます。(有効な引数の一覧を表示するには、remotehelp コマンドを使用してください。)

quote コマンドの使用は、FTP プロトコルを熟知している、経験豊かなユーザーだけに限るべきです。

recv *remote-file* [*local-file*]

get と同一機能です。

reget *remote-file* [*local-file*]

reget コマンドの動作は get コマンドと似ています。ただし、ローカルファイル (*local-file*) が存在し、かつ、そのサイズがリモートファイル (*remote-file*) より小さい

	場合は、 <i>local-file</i> を転送中の <i>remote-file</i> の部分コピーであるとみなして、障害が発生したと考えられる時点から転送を継続するという点が異なります。このコマンドは、接続が途切れる可能性の高いネットワーク経由での大きなファイル転送に便利です。
<code>remotehelp [<i>command-name</i>]</code>	リモート FTP サーバーのヘルプ機能呼び出します。 <i>command-name</i> が指定されていれば、それもサーバーに渡されます。
<code>rename <i>from to</i></code>	リモートマシン上に存在する <i>from</i> で示したファイルの名前を <i>to</i> という名前に変更します。
<code>reset</code>	応答用のキューをクリアします。このコマンドは、コマンドと応答との順序を、リモート FTP サーバーに合わせ直します(再同期化)。リモートサーバーが FTP のプロトコルに違反する動作を行った場合、この再同期化処理が必要となることがあります。
<code>restart [<i>marker</i>]</code>	指定したマーカー (<i>marker</i>) の値で、直後にある <code>get</code> または <code>put</code> を再起動します。UNIX システムでは、 <i>marker</i> は通常ファイルへのバイトオフセットです。直後のコマンドが <code>mget</code> の場合、 <code>restart</code> は、1 つめの <code>get</code> に対して適用されます。 <i>marker</i> に <code>0</code> を指定した場合は、再起動マーカーはクリアされます。引数を指定しなかった場合は、現在の再起動ステータスが表示されます。
<code>rmdir <i>directory-name</i></code>	リモートマシン上の <i>directory-name</i> で示すディレクトリを削除します。
<code>runique</code>	ローカルシステムにファイルを格納する際に一意なファイル名を与える処理のオン・オフの切り替えを行います。この機能がオンのとき、 <code>get</code> または <code>mget</code> コマンドの対象となるローカルファイルの名前と同じファイル名がすでに存在していれば、ファイル名の終わりに <code>.1</code> を付加して一意な名前を新たに生成します。さらに

	その名前と同じファイル名もすでに存在していれば、 <code>.2</code> を使用します。このようにして数値を増加させていき、 <code>.99</code> に到達しても一意名を生成できない場合、ファイル転送は中止され、エラーメッセージが表示されます。一意なファイル名が生成できたら、その名前が報告されます。この <code>runique</code> 機能は、シェルコマンドで生成するローカルファイルの名前には影響を与えません。デフォルトはオフです。
<code>safe</code>	データ転送の保護レベルを「 <code>safe</code> 」に設定します。データ転送は、暗号チェックサムによって完全性が保護されます。ADAT コマンドが失敗した場合、データ転送の保護レベルは「 <code>clear</code> 」のみにになります。
<code>send local-file [remote-file]</code>	<code>put</code> と同一機能です。
<code>sendport</code>	<code>PORT</code> コマンドの使用のオン・オフの切り替えを行います。デフォルトでは、 <code>ftp</code> は各データ転送処理に対して接続を確立する際に <code>PORT</code> コマンドの使用を試みます。複数のファイル転送を実施する場合に、 <code>PORT</code> コマンドを用いると時間の遅れを防ぐことができます。 <code>PORT</code> コマンド実行が失敗すると、 <code>ftp</code> はデフォルトのデータポートを使用します。 <code>PORT</code> コマンドの使用がオフの(使用しない)状態では、データ転送処理に対しての <code>PORT</code> コマンドの使用の試みは抑止されます。このオフ指定は、 <code>PORT</code> コマンドを無視するのにもかかわらず受け入れた旨を(誤って)表示してしまうような FTP システムに接続されている場合に便利です。
<code>site arg1 [arg2] ...</code>	指定した引数をそのまま <code>SITE</code> コマンドとして、リモートの FTP サーバーに送信します。
<code>status</code>	<code>ftp</code> の現在のステータスを表示します。
<code>struct [struct-name]</code>	ファイル構造を <code>struct-name</code> で示す値に設定します。なお <code>struct-name</code> として指定可

	能なものは <code>file</code> だけです。これはデフォルトの“ファイル”構造にあたります。現在のシステムでサポートされている構造は <code>file</code> だけで、これ以外は指定できません。
<code>sunique</code>	リモートシステムにファイルを格納する際に一意なファイル名を与える処理のオン・オフの切り替えを行います。この機能を使うには、リモート FTP サーバーが <code>STOU</code> コマンドをサポートしていなければなりません。リモートサーバーは、生成した一意名を報告します。デフォルトはオフです。
<code>tcpwindow [size]</code>	データ接続に使用する TCP ウィンドウのサイズ (<code>size</code>) を設定します。<code>0</code> を指定すると、データ接続時の TCP ウィンドウサイズは明示的には設定されません。引数を指定しなかった場合は、現在の設定が表示されます。
<code>tenex</code>	“表現タイプ”を、TENEX マシンと通信するために必要な値に設定します。
<code>trace</code>	パケットトレース機能のオン・オフの切り替えを行います (未実装)。
<code>type [type-name]</code>	“表現タイプ”を、<code>type-name</code> に設定します。<code>type-name</code> として指定可能なものは次のとおりです。“ネットワーク ASCII”の場合は <code>ascii</code>、“イメージ”の場合は <code>binary</code> または <code>image</code>、バイト長が 8 の“ローカルバイトサイズ”の場合は <code>tenex</code> (TENEX マシンとの通信に使用)。<code>type-name</code> を省略した場合は、現在の表現タイプが表示されます。デフォルト時のタイプは“ネットワーク ASCII”です。
<code>user user-name [password] [account]</code>	リモート FTP サーバーに対して、ユーザー自身の名前などの属性を通知します。サーバーがパスワードを必要としているのに <code>password</code> を省略した場合は、<code>ftp</code> はローカルエコーをオフにしてからパスワードの入力を要求します。同様にサーバーがアカウントを必要としてい

るのに `account` 指定を省略した場合は、`ftp` はアカウントの入力を要求します。アカウントが指定された場合、(リモートサーバーがログイン時にアカウントを必要としなければ) ログイン処理完了後にアカウントコマンドがリモートサーバーに渡されます。この処理は、`ftp` が“自動ログイン使用不可”状態と呼ばれた場合を除き、FTP サーバーへ最初に接続が行われた時点で自動的に実行されます。

`verbose`

冗長表示モードのオン・オフの切り替えを行います。このモードがオンのとき、FTP サーバーからの応答はすべて表示されます。さらにファイル転送が終了するたびに、その転送処理の効率に関する統計データも報告されます。デフォルトは、`ftp` のコマンドが端末から送られてくる場合はオン、それ以外の場合はオフです。

? [*command*]

`help` と同一機能です。

スペースを含んだコマンド引数は、引用符 (") で囲むことができます。

省略できないコマンド引数を指定しない場合、`ftp` はその引数の入力を要求するプロンプトを出力します。

ファイル転送の中止

ファイル転送処理を中止したい場合、端末の割り込みキーを使用します。その転送処理が送信の場合は、ただちに中止されます。受信の場合には、リモートサーバーに対して FTP プロトコルの `ABOR` コマンドを送信し、以後受信するデータを破棄することにより中止されます。この処理に要する時間は、リモートサーバーが `ABOR` 処理をサポートしているかどうかで違ってきます。リモートサーバーが `ABOR` コマンドをサポートしていないと、要求されたファイルをリモートサーバーが完全に送信し終わるまで `ftp>` というプロンプトは現れません。

`ftp` がローカル処理を終えてリモートサーバーからの応答を待っている状態の場合には、端末から割り込みキーを入力しても無視されます。このモードでは遅延時間が長くなることがあります。その場合の理由としては、前述の `ABOR` 処理によるもの、またはリモートサーバー側の誤った動作 (たとえば、`ftp` プロトコル違反) によるものが考えられます。後者が原因の場合、ローカル `ftp` プログラムを手動で終了させることが必要となります。

ファイル命名規約

`ftp` コマンドの引数として指定されたローカルファイル名は、以下の規則に従って処理されます。

- 1) ファイル名として-を指定すると、読み込みの場合は標準入力、書き出しの場合は標準出力が使用されます。
- 2) ファイル名の最初の文字が|の場合、引数の残りの部分はシェルコマンドであると解釈されます。ftpはその引数を指定して **popen(3C)** を使ってシェルを呼び出し、そのシェルの標準入力からデータを読み込みます(または標準出力にデータを書き出します)。シェルコマンドが空白文字を含んでいる場合には、引用符で囲む必要があります(たとえば、|ls -lt)。この機能の便利な使い方の典型的な例は "dir | more" です。
- 3) 前述の2種類に該当しないローカルファイル名は、グロビング(展開)メカニズムが有効な場合、**sh(1)** が使用する規則に従って展開されます(詳細はglobの項を参照)。putのようにローカルファイル名を1つだけ必要とするftpコマンドの場合には、グロビング処理で最初に生成されるファイル名だけが用いられます。
- 4) mget または get コマンドに対してローカルファイル名を省略すると、リモートファイル名が case、ntrans、nmap の設定値に従って変更され、ローカルファイル名として使用されます。runique がオンの場合は、その名前がさらに変更されることがあります。
- 5) mput または put コマンドに対してリモートファイル名を省略すると、ローカルファイル名が ntrans および nmap の設定値に従って変更され、リモートファイル名として使用されます。sunique がオンの場合は、その名前をリモートサーバーがさらに変更することがあります。

ファイル転送用 パラメータ

FTP 仕様では、ファイル転送に影響を与えうるパラメータをいくつか指定できます。

“表現タイプ”として指定できる値は“ネットワーク ASCII”、“EBCDIC”、“イメージ”、バイト長指定付きの“ローカルバイトサイズ”(おもに PDP10 および PDP20 シリーズ用)です。“ネットワーク ASCII”および“EBCDIC”の場合には、さらにサブタイプ指定を伴います。このサブタイプは、垂直方向のフォーマット制御(復帰改行 (NEWLINE) 文字やフォームフィードなど)がそのまま渡される(“ノンプリント”)か、TELNET フォーマット中に存在する(“TELNET フォーマット制御”)か、あるいは ASA (FORTRAN) (“キャリッジ制御 (ASA)”) フォーマット中に存在するかを指定します。ftp がサポートする表現タイプは、“ネットワーク ASCII”(サブタイプは“ノンプリント”のみ)、“イメージ”、およびバイト長が 8 の“ローカルバイトサイズ”(TENEX マシンとの通信に使用)のみです。

“ファイル構造”に指定可能な値は“file”(レコード構造なし)、“record”、および“page”です。ftp はデフォルトである“file”のみをサポートします。

“転送モード”に指定できる値は“stream”、“block”、および“compressed”です。ftp はデフォルトである“stream”のみをサポートします。

- 使用法

ファイルが2ギガバイト(2³¹バイト)以上ある場合のftpの動作については、[largefile\(5\)](#)を参照してください。

ftp コマンドはIPv6に対応しています。[ip6\(7P\)](#)を参照してください。
- ファイル

~/.netrc
- 属性

属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	network/ftp
CSI	有効

- 関連項目

[ls\(1\)](#), [rcp\(1\)](#), [sh\(1\)](#), [tar\(1\)](#), [popen\(3C\)](#), [ftp\(4\)](#), [ftputers\(4\)](#), [mech\(4\)](#), [netrc\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [ip6\(7P\)](#)

Allman, M., Ostermann, S., Metz, C. 共著、『RFC 2428, FTP Extensions for IPv6 and NATs』、The Internet Society 発行。1998 年 9 月

『RFC 2228, FTP Security Extensions』、Lunt, S. J. 著インターネットドラフト、1993 年 11 月

Postel, Jon, Joyce Reynolds 著、『RFC 959, File Transfer Protocol (FTP)』、Network Information Center 発行、1985 年 10 月

Piscitello, D. 著、『RFC 1639, FTP Operation Over Big Address Records (FOOBAR)』、Network Working Group 発行、1994 年 6 月
- 注意事項

アカウントが/etc/ftputers ファイルにリストされていると、リモートFTPサーバーに明示的に拒否されることが原因で、ログインに失敗する場合があります。[ftputers\(4\)](#)を参照してください。

多くのコマンドの実行が正常に行われるか否かは、リモートサーバーの動きによって決まります。

以前は、表現タイプが“ネットワーク ASCII”の場合の4.2BSDコードハンドリング転送においてキャリッジリターンの扱いに誤りがありましたが、この誤りは修正されました。ただしその修正の影響で、表現タイプが“ネットワーク ASCII”のとき、4.2BSDとのバイナリファイル転送が正しく行われなくても知れません。表現タイプとして“イメージ”を使用してください。

名前	gcore - 実行中のプロセスのコアイメージを取得する
形式	gcore [-pgF] [-o filename] [-c content] process-id...
機能説明	gcore ユーティリティーは、指定した各プロセスのコアイメージを作成します。デフォルトでは、プロセス ID が <i>process-id</i> のプロセスのコアイメージファイルは、 <i>core.process-id</i> という名前になります。
オプション	サポートしているオプションは、次のとおりです。 -c content 指定したコンテンツを使用してコアイメージファイルを生成します。コンテンツの記述では、coreadm(1M) のトークンと同じトークンを使用します。-c オプションは、-p または -g フラグによって生成されるコアには適用されません。 -F 強制。別のプロセスが制御していても、ターゲットプロセスを捕捉します。 -g coreadm(1M) によって構成されたとおりに、大域コンテンツを使用してコアイメージファイルを大域コアファイルリポジトリ内に生成します。ユーザーが大域コアファイルリポジトリへのアクセス権を持っていない場合、コマンドは失敗します。 -o filename コアイメージファイル名の最初の部分の <i>core</i> を <i>filename</i> に置き換えます。<i>filename</i> には、coreadm(1M) のパスと同じ拡張トークンを含めることができます。 -p coreadm(1M) によって構成されたとおりに、プロセス固有のコンテンツを使用して、プロセス固有の場所にコアイメージファイルを生成します。ユーザーがプロセスごとのコアファイルリポジトリへのアクセス権を持っていない場合、このコマンドは失敗します。
オペランド	次のオペランドを指定できます。 <i>process-id</i> プロセス ID
使用法	-F フラグを使用するときは注意してください。2つの制御プロセスを1つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の <i>proc</i> ツールの適用時に何も実行していない場合のみ、安全性が確保されます。
終了ステータス	次の終了ステータスが返されます。 0 成功の場合。 0 以外 失敗した場合 (存在しないプロセス ID など)。
ファイル	<i>core.process-id</i> コアイメージ

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

コマンドの構文は「確実」です。出力形式は「不確実」です。

関連項目 [kill\(1\)](#), [coreadm\(1M\)](#), [setrlimit\(2\)](#), [core\(4\)](#), [proc\(4\)](#), [attributes\(5\)](#)

注意事項 gcore は、RLIMIT_CORE 値を使用した [setrlimit\(2\)](#) システムコールによる影響を受けません。

名前	gencat – 書式付きメッセージカタログの生成
形式	gencat <i>catfile</i> <i>msgfile</i> ...
機能説明	gencat コマンドは、<i>msgfile</i> というメッセージテキストのソースファイルを <i>catfile</i> という書式付きメッセージデータベースにマージします。<i>catfile</i> データベースがまだ存在していない場合は、データベースが作成されます。<i>catfile</i> が存在する場合、そのメッセージは新しい <i>catfile</i> に挿入されます。セット番号とメッセージ番号が衝突する場合、<i>msgfile</i> に定義された新しいメッセージテキストが <i>catfile</i> に現在入っている古いメッセージテキストと置き換わります。gencat に対するメッセージテキストのソースファイル (またはファイルのセット) 入力には、セット番号とメッセージ番号の両方、またはメッセージ番号だけを含めることができます。後者の場合、NL_SETD (nl_types.h(3HEAD) 参照) というセットが想定されます。
メッセージテキストのソースファイルフォーマット	メッセージテキストのソースファイルのフォーマットは次のとおりに定義します。メッセージテキストのソース行のフィールドは、ASCII スペース文字または ASCII タブ文字 1 つで区切ることにご注意ください。他の ASCII スペースまたは ASCII タブは、後続のフィールドの一部と見なされます。 \$set <i>n</i> comment ここで、<i>n</i> は次の \$set、\$delset、またはファイルの終わりまでの後続メッセージのセット識別子を指定します。<i>n</i> は (1-{NL_SETMAX}) の範囲内の数にする必要があります。1 個のソースファイル内のセット識別子は連続する必要はありません。セット識別子の後の文字列はコメントと見なされず。メッセージテキストのソースファイルに \$set 宣言を指定しないと、メッセージはすべてデフォルトのメッセージセット、NL_SETD に入ります。 \$delset <i>n</i> comment 既存のメッセージカタログから <i>n</i> というメッセージセットを削除します。セット番号の後の文字列はコメントと見なされます。注意: <i>n</i> は、有効なセットでない場合、無視されます。 \$comment ドル記号の \$ で始まり、その後に ASCII スペース文字または ASCII タブ文字が続く行は、コメントと見なされます。 <i>m</i> message-text <i>m</i> はメッセージ識別子であり、(1-{NL_MSGMAX}) の範囲内の数になります。<i>message-text</i> は直前の \$set 宣言で指定されたセット識別子とメッセージ識別子の <i>m</i> とともにメッセージカタログに格納されます。<i>message-text</i> が空で、ASCII スペースまたは ASCII タブのフィールドセパレータが存在する場合は、メッセージカタログに空の文字列を格納します。メッセージソース行にメッセージ番号があるが、フィールドセパレータも <i>message-text</i> もない場合、カタログからその番号 (もしあれば) の既存のメッセージを削除します。メッセージ識別子は連続する必要はありません。<i>message-text</i> の長さは、(0-{NL_TEXTMAX}) の範囲内にする必要があります。

\$quote *c*

この行は、オプションの引用符文字である *c* を指定します。この文字を使用して *message-text* を囲めば、メッセージソース行において後方のスペースまたはNULL(空)メッセージが見えるようにすることができます。デフォルト時、または空の *\$quote* 宣言を指定するとき、*message-text* の引用は認識されません。

メッセージテキストのソースファイル内の空の行は無視されます。

テキスト文字列には、下記の表に定義された特殊文字およびエスケープシーケンスを入れることができます。

説明	記号	シーケンス
改行	NL(LF)	\n
水平タブ	HT	\t
垂直タブ	VT	\v
バックスペース	BS	\b
キャリッジリターン	CR	\r
フォームフィード	FF	\f
バックスラッシュ	\	\\
ビットパターン	ddd	\ddd

\ddd というエスケープシーケンスは、バックスラッシュと、それに続く1つ、2つ、または3つの8進数からなります。これらの数字は、希望の文字の値を指定するのに使用します。バックスラッシュの後の文字が、指定された文字のいずれでもない場合、バックスラッシュは無視されます。

また、バックスラッシュの後に ASCII 改行文字を指定すれば、後続の行に文字列を継続できます。したがって、下記の2行は単一のメッセージ文字列を記述します。

```
1 This line continues \
to the next line
```

これは、次の行と等価です。

```
1 This line continues to the next line
```

オペランド 次のオペランドがサポートされています。

catfile

フォーマットされたメッセージカタログのパス名。-を指定すると、標準出力とみなされる。

msgfile メッセージテキストのソースファイルのパス名。*msgfile* の例として `-` を指定すると、標準入力とみなされる。メッセージテキストソースファイルの書式は前述の「メッセージテキストのソースファイルフォーマット」で定義されています。

環境変数 `gencat` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- `0` 正常終了。
- `>0` エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/locale
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [mkmsgs\(1\)](#), [catgets\(3C\)](#), [catopen\(3C\)](#), [gettext\(3C\)](#), [nl_types.h\(3HEAD\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	geniconvtbl - iconv コード変換テーブルの生成
形式	geniconvtbl [-fnq] [-p <i>preprocessor</i>] [-W <i>arg</i>] [-D <i>name</i>] [-D <i>name</i> = <i>def</i>] [-I <i>directory</i>] [-U <i>name</i>] [<i>infile</i>]...
機能説明	geniconvtbl ユーティリティは、フラットテキストファイルに定義されたコード変換規則を受け入れて、コード変換バイナリテーブルファイルを書き込みます。このファイルを使用して、ユーザー定義の iconv コード変換をサポートできます (iconv コード変換の詳細は iconv(1) および iconv(3C) を参照)。
オプション	サポートしているオプションは、次のとおりです。 -f 出力ファイルが存在する場合は、出力ファイルを上書きします。 -n 出力ファイルを生成しません。これは、入力ファイルの内容を確認する場合に役立ちます。 -p <i>preprocessor</i> デフォルトのプロセッサ <code>/usr/lib/cpp</code> の代わりに、指定された <i>preprocessor</i> を使用します。 -q 非出力オプション。警告メッセージとエラーメッセージを抑制します。 -W <i>arg</i> 引数 <i>arg</i> をプリプロセッサに渡します。このオプションが複数回指定されていると、すべての引数がプリプロセッサに渡されます。 -D<i>name</i> -D<i>name</i>=<i>def</i> -I<i>directory</i> -U<i>name</i> geniconvtbl はこれらのオプションを認識して、引数とともにプリプロセッサに渡します。
オペランド	次のオペランドを指定できます。 <i>infile</i> 入力ファイルのパス名。入力ファイルが指定されていない場合、geniconvtbl は標準入力ストリームから読み込みます。ユーザーは、必要に応じて複数の入力ファイルを指定できます。
出力	標準入力ストリームから入力が行われる場合、geniconvtbl は出力を標準出力ストリームに書き込みます。複数の入力ファイルが指定されている場合、geniconvtbl は各入力ファイルから読み込み、対応する出力ファイルに書き込みます。各出力ファイルの名前は、対応する入力ファイルに <code>.bt</code> を追加したものになります。 生成された出力ファイルは、iconv(1) および iconv(3C) でコード変換を使用する前に、次のディレクトリに移動する必要があります。 <code>/usr/lib/iconv/geniconvtbl/binarytables/</code>

出力ファイル名は、先頭から 1 文字以上の印字可能な ASCII 文字 (「fromcode」名)、パーセント文字 (%)、1 文字以上の印字可能な ASCII 文字 (「tocode」名)、接尾辞「.bt」の順にします。「fromcode」名と「tocode」名は、[iconv\(1\)](#) および [iconv_open\(3C\)](#) の実行時に iconv コード変換の特定に使用されます。適切に命名された出力ファイルを `/usr/lib/iconv/geniconvtbl/binarytables/` ディレクトリに配置してください。

使用例

例1 iconv コード変換バイナリテーブルを生成する

次の例では、出力ファイル名 `convertA2B.bt` のコード変換バイナリテーブルを生成します。

```
example% geniconvtbl convertA2B
```

例2 複数の iconv コード変換バイナリテーブルを生成する

次の例では、出力ファイルが `test1.bt` と `test2.bt` の 2 つのコード変換バイナリテーブルを生成します。

```
example% geniconvtbl test1 test2
```

例3 別のプリプロセッサを使用する

次の例では、指定されたプリプロセッサが入力ファイルを処理したあとで、コード変換バイナリテーブルを生成します。

```
example% geniconvtbl -p /opt/SUNWspro/bin/cc -W -E convertB2A
```

例4 バイナリテーブルを配置する

上記の最初の例で作成したバイナリテーブルを「fromcode」ABC から「tocode」DEF への変換のエンジンとして使用するには、スーパーユーザーになり、次の方法で名前の変更および配置を行います。

```
example# mv convertA2B.bt \
    /usr/lib/iconv/geniconvtbl/binarytables/ABC%DEF.bt
```

例5 ISO8859-1 を変更して UTF-8 コード変換に使用する

コード変換を定義する `geniconvtbl` ソースファイルを記述します。たとえば、`/usr/lib/iconv/geniconvtbl/srcs/ISO8859-1_to_UTF-8.src` を自分のディレクトリにコピーして、ソースファイルに必要な変更を加えることができます。変更が完了したら、バイナリテーブルを生成します。

```
example% geniconvtbl ISO8859-1_to_UTF-8.src
```

スーパーユーザーになり、一意の名前を持つ生成されたバイナリテーブルをシステムディレクトリに配置して、[iconv_open\(3C\)](#) がこのバイナリテーブルを検出できるようにします。

例5 ISO8859-1 を変更して UTF-8 コード変換に使用する (続き)

```
example su
Password:
example% cp ISO8859-1_to_UTF-8.bt \
        /usr/lib/iconv/geniconvtbl/binarytables/my-iso-8859-1%utf-8.bt
```

その後、iconv コード変換を実行できます。例:

```
example% iconv -f my-iso-8859-1 -t utf-8 testfile.txt
```

環境変数 geniconvtbl の実行に影響を与える環境変数 LANG および LC_CTYPE についての詳細は、[environ\(5\)](#) を参照してください。

終了ステータス 次の終了ステータスが返されます。

- 0 エラーは発生せず、出力ファイルは正常に作成されました。
- 1 コマンド行オプションは現在使用されていないか、不明のコマンド行オプションが指定されました。
- 2 無効な入力ファイルまたは出力ファイルが指定されました。
- 3 現在、入力ファイルの変換規則が正しく定義されていません。
- 4 入力ファイルの変換規則制限に達しました。[geniconvtbl\(4\)](#) の「注意事項」の項を参照してください。
- 5 これ以上のシステムリソースエラーはありません。
- 6 内部エラー。

ファイル /usr/lib/iconv/geniconvtbl/binarytables/*.bt
変換バイナリテーブル

/usr/lib/iconv/geniconvtbl/srcs/*
ユーザー参照用の変換ソースファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [cpp\(1\)](#), [iconv\(1\)](#), [iconv\(3C\)](#), [iconv_close\(3C\)](#), [iconv_open\(3C\)](#), [geniconvtbl\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [iconv\(5\)](#)

『Solaris Internationalization Guide for Developers』

注意事項

生成され、正しく配置された出力ファイル

/usr/lib/iconv/geniconvtbl/binarytables/*.bt は、32ビットと64ビットの両方の環境で使用されます。

名前 genmsg – generate a message source file by extracting messages from source files

形式 genmsg [-abdfmntx] [-c *message-tag*] [-g *project-file*]
 [-l *project-file*] [-m *prefix*] [-M *suffix*]
 [-o *message-file*] [-p *preprocessor*] [-s *set-tags*] *file...*

機能説明 The genmsg utility extracts message strings with calls to [catgets\(3C\)](#) from source files and writes them in a format suitable for input to [gencat\(1\)](#).

Invocation genmsg reads one or more input files and, by default, generates a message source file whose name is composed of the first input file name with `.msg`. If the `-o` option is specified, genmsg uses the option argument for its output file.

Command	Output File
genmsg prog.c	prog.c.msg
gensmg main.c util.c tool.c	main.c.msg
genmsg -o prog.msg mail.c util.c	prog.msg

genmsg also allows you to invoke a preprocessor to solve the dependencies of macros and define statements for the [catgets\(3C\)](#) calls.

Auto Message genmsg replaces message numbers with the calculated numbers based upon the project file if
Numbering the message numbers are -1, and it generates copies of the input files with the new message numbers and a copy of the project file with the new maximum message numbers.

A project file is a database that stores a list of set numbers with their maximum message numbers. Each line in a project file is composed of a set number and its maximum message number:

Set_number Maximum_message_number

In a project file, a line beginning with a number sign (#) or an ASCII space is considered as a comment and ignored.

genmsg also has the reverse operation to replace all message numbers with -1.

Comment Extraction genmsg allows you to comment about messages and set numbers to inform the translator how the messages should be translated. It extracts the comment, which is surrounded with the comment indicators and has the specified tag inside the comment, from the input file and writes it with a dollar (\$) prefix in the output file. genmsg supports the C and C++ comment indicators, `'/*', '*/', and '//'`.

Testing

genmsg generates two kinds of messages for testing, prefixed messages and long messages. Prefixed messages allow you to check that your program is retrieving the messages from the message catalog. Long messages allow you to check the appearance of your window program's initial size and position.

オプション

The following options are supported:

- a Append the output into the message file *message-file* that is specified by the -o option. If two different messages that have the same set and message number are found, the message in the specified message file is kept and the other message in the input file is discarded.
- b Place the extracted comment after the corresponding message in the output file. This option changes the placement behavior of the -s or -c option.
- c *message-tag* Extract message comments having *message-tag* inside them from the input files and write them with a '\$' prefix as a comment in the output file.
- d Include an original text of a message as a comment to be preserved along with its translations. With this option, the translator can see the original messages even after they are replaced with their translations.
- f Overwrite the input files and the project file when used with the -l or -r option. With the -r option, genmsg overwrites only the input files.
- g *project-file* Generate *project-file* that has a list of set numbers and their maximum message numbers in the input files.
- l *project-file* Replace message numbers with the calculated numbers based upon *project-file* if the message numbers are -1 in the input files, and then generate copies of the input files with the new message numbers and a copy of *project-file* with the new maximum message numbers. If *project-file* is not found, genmsg uses the maximum message number in the input file as a base number and generates *project-file*.
- m *prefix* Fill in the message with *prefix*. This option is useful for testing.
- M *suffix* Fill in the message with *suffix*. This option is useful for testing.
- n Add comment lines to the output file indicating the file name and line number in the input files where each extracted string is encountered.
- o *message-file* Write the output to *message-file*.
- p *preprocessor* Invoke *preprocessor* to preprocess macros and define statements for the [catgets\(3C\)](#) calls. genmsg first invokes the option argument as a preprocessor and then starts the normal process against the output from the preprocessor. genmsg initiates this process for all the input files.

- r Replace message numbers with -1. This is the reverse operation of the -l option.
- s *set-tag* Extract set number comments having *set-tag* inside them from the input files and write them with a '\$' prefix as a comment in the output file. If multiple comments are specified for one set number, the first one is extracted and the rest of them are discarded.
- t Generate a message that is three times as long as the original message. This option is useful for testing.
- x Suppress warning messages about message and set number range checks and conflicts.

オペランド *file* An input source file.

使用例

例1 Assigning Message Numbers and Generating New Files

Suppose that you have the following source and project files:

```
example% cat test.c
printf(catgets(catfd, 1, -1, "line too long\n"));
printf(catgets(catfd, 2, -1, "invalid code\n"));
```

```
example% cat proj
1   10
2   20
```

The command

```
example% genmsg -l proj test.c
```

would assign the calculated message numbers based upon `proj` and generate the following files:

`test.c.msg` Message file

`proj.new` Updated project file

`test.c.new` New source file

```
example% cat test.c.msg
$quote "
$set   1
11     "line too long\n"
$set   2
21     "invalid code\n"
```

```
example% cat proj.new
1   11
2   21
```

例1 Assigning Message Numbers and Generating New Files (続き)

```
example% cat test.c.new
printf(catgets(catfd, 1, 11, "line too long\n"));
printf(catgets(catfd, 2, 21, "invalid code\n"));
```

例2 Extracting Comments Into a File

The command

```
example% genmsg -s SET -c MSG test.c
example% cat test.c
/* SET: tar messages */
/* MSG: don't translate "tar". */
catgets(catfd, 1, 1, "tar: tape write error");
// MSG: don't translate "tar" and "-I".
catgets(catfd, 1, 2, "tar: missing argument for -I flag");
```

would extract the comments and write them in the following output file:

```
example% cat test.c.msg
$ /* SET: tar messages */
$set    1
$ /* MSG: don't translate "tar". */
1      "tar: tape write error"
$ // MSG: don't translate "tar" and "-I".
2      "tar: missing argument for -I flag"
```

例3 Generating Test Messages

The following command:

```
example% genmsg -m PRE: -M :FIX test.c
```

might generate the following messages for testing:

```
example% cat test.c.msg
1      "PRE:OK:FIX"
2      "PRE:Cancel:FIX"
```

例4 Parsing a Macro and Writing the Extracted Messages

Given the following input:

```
example% cat example.c
#include <nl_types.h>
#define MSG1    "message1"
#define MSG2    "message2"
#define MSG3    "message3"
#define MSG(n)  catgets(catd, 1, n, MSG ## n)
```

例 4 Parsing a Macro and Writing the Extracted Messages (続き)

```

void
main(int argc, char **argv)
{
    nl_catd catd = catopen(argv[0], NL_CAT_LOCALE);
    (void) printf("%s0\n", MSG(1));
    (void) printf("%s0\n", MSG(2));
    (void) printf("%s0\n", MSG(3));
    (void) catclose(catd);
}

```

The following command:

```
example% genmsg -p "cc -E" -o example.msg example.c
```

would parse the MSG macros and write the extracted messages in `example.msg`.

例 5 Assigning Calculated Message Numbers

Suppose that you have the following header, source, and project files:

```

example% cat ../inc/msg.h
#define WARN_SET    1
#define ERR_SET     2
#define WARN_MSG(id, msg) catgets(catd, WARN_SET, (id), (msg))
#define ERR_MSG(id, msg) catgets(catd, ERR_SET, (id), (msg))
example% example.c
#include "msg.h"
printf("%s", WARN_MSG(-1, "Warning error"));
printf("%s", ERR_MSG(-1, "Fatal error"));
example % proj
1      10
2      10

```

The command

```

example% genmsg -f -p "cc -E -I../inc" -l proj \
-o example.msg example.c

```

would assign each of the -1 message numbers a calculated number based upon `proj` and would overwrite the results to `example.c` and `proj`. Also, this command writes the extracted messages in `example.msg`.

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `genmsg`: `LC_MESSAGES` and `NLSPATH`.

終了ステータス The following exit values are returned:

0 Successful completion.

>0 An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/locale

関連項目 [gencat\(1\)](#), [catgets\(3C\)](#), [catopen\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 genmsg does not handle pointers or variables in the [catgets\(3C\)](#) call. For example:

```
const int set_num = 1;
extern int msg_num(const char *);
const char *msg = "Hello";
catgets(catd, set_num, msg_num(msg), msg);
```

When the auto message numbering is turned on with a preprocessor, if there are multiple -1's in the [catgets\(3C\)](#) line, genmsg replaces all of the -1's in the line with a calculated number. For example, given the input:

```
#define MSG(id, msg) catgets(catd, 1, (id), (msg))
if (ret == -1) printf("%s, MSG(-1, "Failed");
```

the command

```
genmsg -l proj -p "cc -E"
```

would produce:

```
#define MSG(id, msg) catgets(catd, 1, (id), (msg))
if (ret == 1) printf("%s, MSG(1, "Failed");
```

The workaround would be to split it into two lines as follows:

```
if (ret == -1)
    printf("%s, MSG(-1, "Failed");
```

名前

形式

機能説明

オプション

オペランド

getconf – get configuration values

/usr/bin/getconf [-v *specification*] *system_var*

/usr/bin/getconf [-v *specification*] *path_var* *pathname*

/usr/bin/getconf -a

/usr/xpg4/bin/getconf [-v *specification*] *system_var*

/usr/xpg4/bin/getconf [-v *specification*] *path_var* *pathname*

/usr/xpg4/bin/getconf -a

/usr/xpg6/bin/getconf [-v *specification*] *system_var*

/usr/xpg6/bin/getconf [-v *specification*] *path_var* *pathname*

/usr/xpg6/bin/getconf -a

In the first synopsis form, the `getconf` utility writes to the standard output the value of the variable specified by *system_var*, in accordance with *specification* if the `-v` option is used.

In the second synopsis form, `getconf` writes to the standard output the value of the variable specified by *path_var* for the path specified by *pathname*, in accordance with *specification* if the `-v` option is used.

In the third synopsis form, `confi g` writes to the standard output the names of the current system configuration variables.

The value of each configuration variable is determined as if it were obtained by calling the function from which it is defined to be available. The value reflects conditions in the current operating environment.

The following options are supported:

-a

Writes the names of the current system configuration variables to the standard output.

-v *specification*

Gives the specification which governs the selection of values for configuration variables.

The following operands are supported:

path_var

A name of a configuration variable whose value is available from the [pathconf\(2\)](#) function. All of the values in the following table are supported:

LINK_MAX

NAME_MAX

_POSIX_CHOWN_RESTRICTED

MAX_CANON

PATH_MAX

_POSIX_NO_TRUNC

MAX_INPUT

PIPE_BUF

_POSIX_VDISABLE

pathname A path name for which the variable specified by *path_var* is to be determined.

system_var A name of a configuration variable whose value is available from [confstr\(3C\)](#) or [sysconf\(3C\)](#). All of the values in the following table are supported:

ARG_MAX	BC_BASE_MAX
BC_DIM_MAX	BC_SCALE_MAX
BC_STRING_MAX	CHAR_BIT
CHARCLASS_NAME_MAX	CHAR_MAX
CHAR_MIN	CHILD_MAX
CLK_TCK	COLL_WEIGHTS_MAX
CS_PATH	EXPR_NEST_MAX
HOST_NAME_MAX	INT_MAX
INT_MIN	LFS64_CFLAGS
LFS64_LDFLAGS	LFS64_LIBS
LFS64_LINTFLAGS	LFS_CFLAGS
LFS_LDFLAGS	LFS_LIBS
LFS_LINTFLAGS	LINE_MAX
LONG_BIT	LONG_MAX
LONG_MIN	MB_LEN_MAX
NGROUPS_MAX	NL_ARGMAX
NL_LANGMAX	NL_MSGMAX
NL_NMAX	NL_SETMAX
NL_TEXTMAX	NZERO
OPEN_MAX	POSIX2_BC_BASE_MAX
POSIX2_BC_DIM_MAX	POSIX2_BC_SCALE_MAX
POSIX2_BC_STRING_MAX	POSIX2_C_BIND
POSIX2_C_DEV	POSIX2_CHAR_TERM
POSIX2_COLL_WEIGHTS_MAX	POSIX2_C_VERSION
POSIX2_EXPR_NEST_MAX	POSIX2_FORT_DEV
POSIX2_FORT_RUN	POSIX2_LINE_MAX

POSIX2_LOCALEDEF	POSIX2_RE_DUP_MAX
POSIX2_SW_DEV	POSIX2_SYMLINKS
POSIX2_UPE	POSIX2_VERSION
POSIX_ALLOC_SIZE_MIN	POSIX_REC_INCR_XFER_SIZE
POSIX_REC_MAX_XFER_SIZE	POSIX_REC_MIN_XFER_SIZE
POSIX_REC_XFER_ALIGN	POSIX_V6_ILP32_OFF32
POSIX_V6_ILP32_OFF32_CFLAGS	POSIX_V6_ILP32_OFF32_LDFLAGS
POSIX_V6_ILP32_OFF32_LIBS	POSIX_V6_ILP32_OFFBIG
POSIX_V6_ILP32_OFFBIG_CFLAGS	POSIX_V6_ILP32_OFFBIG_LDFLAGS
POSIX_V6_ILP32_OFFBIG_LIBS	POSIX_V6_LP64_OFF64
POSIX_V6_LP64_OFF64_CFLAGS	POSIX_V6_LP64_OFF64_LDFLAGS
POSIX_V6_LP64_OFF64_LIBS	POSIX_V6_LPBIG_OFFBIG
POSIX_V6_LPBIG_OFFBIG_CFLAGS	POSIX_V6_LPBIG_OFFBIG_LDFLAGS
POSIX_V6_LPBIG_OFFBIG_LIBS	POSIX_V6_WIDTH_RESTRICTED_ENVS
SYMLINK_MAX	SYMLOOP_MAX
_POSIX2_BC_BASE_MAX	_POSIX2_BC_DIM_MAX
_POSIX2_BC_SCALE_MAX	_POSIX2_BC_STRING_MAX
_POSIX2_CHARCLASS_NAME_MAX	_POSIX2_CHAR_TERM
_POSIX2_COLL_WEIGHTS_MAX	_POSIX2_C_BIND
_POSIX2_C_DEV	_POSIX2_C_VERSION
_POSIX2_EXPR_NEST_MAX	_POSIX2_FORT_DEV
_POSIX2_FORT_RUN	_POSIX2_LINE_MAX
_POSIX2_LOCALEDEF	_POSIX2_PBS
_POSIX2_PBS_ACCOUNTING	_POSIX2_PBS_CHECKPOINT
_POSIX2_PBS_LOCATE	_POSIX2_PBS_MESSAGE
_POSIX2_PBS_TRACK	_POSIX2_RE_DUP_MAX
_POSIX2_SW_DEV	_POSIX2_UPE
_POSIX2_VERSION	_POSIX_ADVISORY_INFO
_POSIX_AIO_LISTIO_MAX	_POSIX_AIO_MAX

_POSIX_ARG_MAX	_POSIX_ASYNC_IO
_POSIX_BARRIERS	_POSIX_CHILD_MAX
_POSIX_CLOCKRES_MIN	_POSIX_CLOCK_SELECTION
_POSIX_CPUTIME	_POSIX_DELAYTIMER_MAX
_POSIX_HOST_NAME_MAX	_POSIX_IPV6
_POSIX_JOB_CONTROL	_POSIX_LINK_MAX
_POSIX_LOGIN_NAME_MAX	_POSIX_MAX_CANON
_POSIX_MAX_INPUT	_POSIX_MONOTONIC_CLOCK
_POSIX_MQ_OPEN_MAX	_POSIX_MQ_PRIO_MAX
_POSIX_NAME_MAX	_POSIX_NGROUPS_MAX
_POSIX_OPEN_MAX	_POSIX_PATH_MAX
_POSIX_PIPE_BUF	_POSIX_PRIO_IO
_POSIX_RAW_SOCKETS	_POSIX_READER_WRITER_LOCKS
_POSIX_REGEX	_POSIX_RE_DUP_MAX
_POSIX_RTSIG_MAX	_POSIX_SAVED_IDS
_POSIX_SEM_NSEMS_MAX	_POSIX_SEM_VALUE_MAX
_POSIX_SHELL	_POSIX_SIGQUEUE_MAX
_POSIX_SPAWN	_POSIX_SPIN_LOCKS
_POSIX_SPORADIC_SERVER	_POSIX_SSIZE_MAX
_POSIX_SS_REPL_MAX	_POSIX_STREAM_MAX
_POSIX_SYMLINK_MAX	_POSIX_SYMLOOP_MAX
_POSIX_SYNC_IO	_POSIX_THREAD_ATTR_STACKADDR
_POSIX_THREAD_ATTR_STACKSIZE	_POSIX_THREAD_CPUTIME
_POSIX_THREAD_DESTRUCTOR_ITERATIONS	_POSIX_THREAD_KEYS_MAX
_POSIX_THREAD_PRIORITY_SCHEDULING	_POSIX_THREAD_PRIO_INHERIT
_POSIX_THREAD_PRIO_PROTECT	_POSIX_THREAD_PROCESS_SHARED
_POSIX_THREAD_SAFE_FUNCTIONS	_POSIX_THREAD_SPARADIC_SERVER
_POSIX_THREAD_THREADS_MAX	_POSIX_TIMEOUTS
_POSIX_TIMER_MAX	_POSIX_TRA_POSIX_TIMER_MAXCE

_POSIX_TIMESTAMP_RESOLUTION	
_POSIX_TRACE_EVENT_FILTER	_POSIX_TRACE_EVENT_NAME_MAX
_POSIX_TRACE_INHERIT	_POSIX_TRACE_LOG
_POSIX_TRACE_NAME_MAX	_POSIX_TRACE_SYS_MAX
_POSIX_TRACE_USER_EVENT_MAX	_POSIX_TTY_NAME_MAX
_POSIX_TYPED_MEMORY_OBJECTS	_POSIX_TZNAME_MAX
_POSIX_VERSION	_POSIX_V6_ILP32_OFF32
_POSIX_V6_ILP32_OFFBIG	_POSIX_V6_LP64_OFF64
_POSIX_V6_LPBIG_OFFBIG	_V6_ILP32_OFF32
_V6_ILP32_OFFBIG	_V6_LP64_OFF64
_V6_LPBIG_OFFBIG	RE_DUP_MAX
SCHAR_MAX	SCHAR_MIN
SHRT_MAX	SHRT_MIN
SSIZE_MAX	STREAM_MAX
TMP_MAX	TZNAME_MAX
UCHAR_MAX	UINT_MAX
ULONG_MAX	USHRT_MAX
WORD_BIT	XBS5_ILP32_OFF32
XBS5_ILP32_OFF32_CFLAGS	XBS5_ILP32_OFF32_LDFLAGS
XBS5_ILP32_OFF32_LIBS	XBS5_ILP32_OFF32_LINTFLAGS
XBS5_ILP32_OFFBIG	XBS5_ILP32_OFFBIG_CFLAGS
XBS5_ILP32_OFFBIG_LDFLAGS	XBS5_ILP32_OFFBIG_LIBS
XBS5_ILP32_OFFBIG_LINTFLAGS	XBS5_LP64_OFF64
XBS5_LP64_OFF64_CFLAGS	XBS5_LP64_OFF64_LDFLAGS
XBS5_LP64_OFF64_LIBS	XBS5_LP64_OFF64_LINTFLAGS
XBS5_LPBIG_OFFBIG	XBS5_LPBIG_OFFBIG_CFLAGS
XBS5_LPBIG_OFFBIG_LDFLAGS	XBS5_LPBIG_OFFBIG_LIBS
XBS5_LPBIG_OFFBIG_LINTFLAGS	_XOPEN_CRYPT
_XOPEN_ENH_I18N	_XOPEN_IOV_MAX

<code>_XOPEN_LEGACY</code>	<code>_XOPEN_NAME_MAX</code>
<code>_XOPEN_PATH_MAX</code>	<code>_XOPEN_SHM</code>
<code>_XOPEN_STREAMS</code>	<code>_XOPEN_VERSION</code>
<code>_XOPEN_XCU_VERSION</code>	<code>_XOPEN_XPG2</code>
<code>_XOPEN_XPG3</code>	<code>_XOPEN_XPG4</code>

The symbol `PATH` also is recognized, yielding the same value as the `confstr()` name value `CS_PATH`.

使用法

See [largefile\(5\)](#) for the description of the behavior of `/usr/bin/getconf` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

使用例

例1 Writing the Value of a Variable

This example illustrates the value of `{NGROUPS_MAX}`:

```
example% getconf NGROUPS_MAX
```

例2 Writing the Value of a Variable for a Specific Directory

This example illustrates the value of `NAME_MAX` for a specific directory:

```
example% getconf NAME_MAX /usr
```

例3 Dealing with Unspecified Results

This example shows how to deal more carefully with results that might be unspecified:

```
if value=$(getconf PATH_MAX /usr); then
if [ "$value" = "undefined" ]; then
echo PATH_MAX in /usr is infinite.
else
echo PATH_MAX in /usr is $value.
fi
else
echo Error in getconf.
fi
```

For example:

```
sysconf(_SC_POSIX_C_BIND);

and

system("getconf POSIX2_C_BIND");
```

in a C program could give different answers. The `sysconf` call supplies a value that corresponds to the conditions when the program was either compiled or executed, depending on the implementation. The `system` call to `getconf` always supplies a value corresponding to

例 3 Dealing with Unspecified Results (続き)

conditions when the program is executed.

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `getconf`: `LANG`, `LC_ALL`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

終了ステータス The following exit values are returned:

- 0 The specified variable is valid and information about its current state was written successfully.
- >0 An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [sh\(1\)](#), [pathconf\(2\)](#), [sysinfo\(2\)](#), [confstr\(3C\)](#), [sysconf\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	getfacl – 任意のファイル情報の表示
形式	getfacl [-ad] <i>file</i> ...
機能説明	通常ファイル、特殊ファイル、または名前付きパイプである引数ごとに、getfacl ユーティリティーは、所有者、グループ、およびアクセス制御リスト (ACL) を表示します。ディレクトリ引数ごとに、getfacl は、所有者、グループ、および ACL またはデフォルトの ACL、あるいはその両方を表示します。デフォルトの ACL が含まれるのはディレクトリだけです。 getfacl ユーティリティーは、ACL をサポートしていないファイルシステム上で実行できます。その場合は、基本のアクセス権ビットに基づいて ACL を報告します。 オプションが指定されていない場合、getfacl は、ファイル名、ファイル所有者、ファイルグループ所有者、および ACL とデフォルトの ACL (存在する場合) の両方を表示します。
オプション	サポートしているオプションは、次のとおりです。 -a ファイル名、ファイル所有者、ファイルグループ所有者、およびファイルの ACL を表示します。 -d ファイル名、ファイル所有者、ファイルグループ所有者、およびファイルのデフォルトの ACL (存在する場合) を表示します。
オペランド	次のオペランドがサポートされています。 <i>file</i> 通常ファイル、特殊ファイル、または名前付きパイプのパス名。
出力	ACL の出力の形式は次のとおりです。 <pre># file: filename # owner: uid # group: gid user::perm user:uid:perm group::perm group:gid:perm mask:perm other:perm default:user::perm default:user:uid:perm default:group::perm default:group:gid:perm default:mask:perm default:other:perm</pre> コマンド行で複数のファイルが指定される場合、各ファイルの ACL は空行で区切られます。

ACL エントリは、アクセスチェックが実行されるときに評価された順序で表示されます。ディレクトリ上に存在する可能性のあるデフォルトの ACL エントリは、アクセスチェックに影響を与えません。

最初の 3 行には、ファイル名、ファイル所有者、およびファイルグループ所有者が表示されます。-d オプションのみが指定され、ファイルにデフォルトの ACL が存在しない場合は、これらの 3 行のみが表示されます。

ユーザー ID のない user エントリは、ファイル所有者に付与されたアクセス権を示します。1 つ以上の追加のユーザーエントリは、指定されたユーザーに付与されたアクセス権を示します。

グループ ID のない group エントリは、ファイルグループ所有者に付与されたアクセス権を示します。1 つ以上の追加のグループエントリは、指定されたグループに付与されたアクセス権を示します。

mask エントリは、ACL マスクのアクセス権を示します。これらは、ファイル所有者を除く任意のユーザーエントリ、およびファイルグループ所有者を含む任意のグループエントリに許可される最大のアクセス権です。これらのアクセス権によって、ほかのエントリで指定されたアクセス権が制限されます。

other エントリは、ほかのユーザーに付与されたアクセス権を示します。

default エントリは、ディレクトリに対してのみ存在できます。これらのエントリは、ディレクトリ内で作成されるファイルに追加されたデフォルトのエントリを示します。

uid は、システムパスワードファイル /etc/passwd 内に uid のエントリが存在しない場合は、ログイン名またはユーザー ID です。gid は、システムグループファイル /etc/group 内に gid のエントリが存在しない場合は、グループ名またはグループ ID です。perm は、r(読み取り)、w(書き込み)、x(実行/検索)という個別の随意アクセス権を表す英字またはプレースホルダー文字で構成された 3 文字から成る文字列です。perm は、rwx の順序で表示されます。ACL エントリによってアクセス権が付与されていない場合は、プレースホルダー文字が表示されます。

chmod(1) コマンドを使用して ACL エントリを含むファイルに対するファイルグループ所有者のアクセス権を変更した場合は、ファイルグループ所有者のアクセス権と ACL マスクの両方が新しいアクセス権に変更されます。新しい ACL マスクのアクセス権によって、そのファイル上に ACL エントリを持つ追加のユーザーやグループの実効アクセス権が変更される場合があることに注意してください。

ACL マスクによって ACL エントリが制限されることを示すために、getfacl は、そのエントリのあとに、追加のタブ文字、ポンド記号(#)、および付与される実際のアクセス権を表示します。

使用例

例1 ファイル情報の表示

ACL の6つのエントリの長さを持つファイル `foo` の場合、次のコマンド

```
host% getfacl foo
```

によって次の内容が出力されます:

```
# file: foo
# owner: shea
# group: staff
user::rwx
user:spy: - - -
user:mookie:r - -
group::r - -
mask::rw -
other:: - - -
```

例2 chmod コマンドのあとの情報の表示

上の例を続けると、`chmod 700 foo` が発行されたあと、次のようになります。

```
host% getfacl foo
```

によって次の内容が出力されます:

```
# file: foo
# owner: shea
# group: staff
user::rwx
user:spy: - - -
user:mookie:r - -      #effective: - - -
group:: - - -
mask:: - - -
other:: - - -
```

例3 ACL にデフォルトのエントリが含まれている場合の情報の表示

デフォルトのエントリが含まれた ACL を含むディレクトリ `doo` の場合、次のコマンド

```
host% getfacl -d doo
```

によって次の内容が出力されます:

```
# file: doo
# owner: shea
# group: staff
default:user::rwx
default:user:spy: - - -
default:user:mookie:r - -
default:group::r - -
```

例3 ACLにデフォルトのエントリが含まれている場合の情報の表示 (続き)

```
default:mask:: ---
default:other:: ---
```

ファイル /etc/passwd システムパスワードファイル
/etc/group グループファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [chmod\(1\)](#), [ls\(1\)](#), [setfacl\(1\)](#), [acl\(2\)](#), [aclsort\(3SEC\)](#), [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

注意事項 getfacl からの出力は、setfacl -f コマンドへの入力のための正しい形式をしています。getfacl からの出力がファイルにリダイレクトされた場合は、そのファイルを setfacl への入力として使用できます。このようにして、ユーザーは、あるファイルの ACL を別のファイルに容易に割り当てることができます。

名前	getlabel – ファイルのラベルの表示
形式	/usr/bin/getlabel [-sS] filename...
機能説明	getlabel は、各 <i>filename</i> に関連付けられているラベルを表示します。オプションを指定しない場合、ラベルはデフォルト形式で表示されます。
オプション	-s <i>filename</i> に関連付けられているラベルを短形式で表示します。 -S <i>filename</i> に関連付けられているラベルを長形式で表示します。
終了ステータス	getlabel は、終了時に次のいずれかの値を返します。 0 正常終了。 1 使用方法の誤りのためエラー終了しました。 2 ラベルを変換できませんでした。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/trusted
インタフェースの安定性	下記を参照。

このコマンド行は「確実」です。出力は非インタフェースです。

関連項目 [setlabel\(1\)](#), [m_label\(3TSOL\)](#), [label_encodings\(4\)](#), [attributes\(5\)](#)

注意事項 このマニュアルページに記載されている機能を利用できるのは、システムが Trusted Extensions を使用して構成されている場合のみです。

名前	getopt - コマンドオプションの解析
形式	set -- 'getopt <i>optstring</i> \$ * '
機能説明	getopts コマンドが、getopt に取って代わります。詳細は、以下の「注意事項」の項を参照してください。

getopt は、シェルプロシージャによる解析を容易にするためにコマンド行のオプションを分解し、各オプションが正当であるかどうかの確認に使用します。*optstring* は、認識されるオプションの文字列です。getopt(3C) を参照してください。オプション文字のあとにコロン(:) が付いている場合は、そのオプションに引数があるとみなされます。コロンと引数の間は、スペースで区切られていても、区切られていなくてもかまいません。特殊なオプション -- は、オプションの終わりを表すのに使用されます。このオプションを明示的に使用すると getopt はこれを認識し、省略された場合には getopt がこれを生成します。いずれの場合でも、getopt はオプションの終わりにこの特殊オプションを付けます。シェルの位置パラメータ (\$1 \$2 ...) が再設定されて、各オプションの前に - が付き、各オプションはそのオプションの位置パラメータに入ります。各オプション引数も、同様にその引数の位置パラメータに入るように解析されます。

使用例

例1 コマンド引数の処理

次の部分コードに、オプション -a または -b、および引数の必要なオプション -o を使用するコマンドについて、引数の処理例を示します。

```
set -- 'getopt abo: $*'
if [ $? != 0 ]
then
    echo $USAGE
    exit 2
fi
for i in $*
do
    case $i in
        -a | -b)    FLAG=$i; shift;;
        -o)        OARG=$2; shift 2;;
        --)        shift; break;;
        esac
    done
```

このコードは、次のコマンドすべてを同一内容として受け入れます。

```
cmd -aoarg filename1 filename2
cmd -a -o arg filename1 filename2
cmd -oarg -a filename1 filename2
cmd -a -oarg -- filename1 filename2
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目 [Intro\(1\)](#), [getopts\(1\)](#), [getoptcv\(1\)](#), [sh\(1\)](#), [shell_builtins\(1\)](#), [getopt\(3C\)](#), [attributes\(5\)](#)

診断 `getopt` は、`optstring` に含まれていないオプション文字を検出すると、標準エラー出力にエラーメッセージを表示します。

注意事項 `getopt` は、次のメジャーリリースではサポートされません。このリリースにおいては、変換ツール `getoptcv` が提供されています。詳細については、[getopts\(1\)](#) および [getoptcv\(1\)](#) を参照してください。

オプションを再走査する際は、`optind` に 1 を再設定してください。

`getopt` は、オプションに続くオプション引数を、スペースおよび引用符で区切って指定できるというコマンド構文標準規則 8 の一部をサポートしません([Intro\(1\)](#) 参照)。たとえば、

```
cmd -a -b -o "xxx z yy" filename
```

は、正しく処理されません。この不完全性を正すためには、`getopt` コマンドの代わりに `getopts` コマンドを使用してください。

オプション引数を持つあるオプションに、`optstring` で示されているオプションの 1 つと同じ値が続いている場合 (前述の「使用例」の項を参照。ただし、次のコマンド行を使用する場合)、

```
cmd -o -a filename
```

`getopt` は、常に `-a` を `-o` のオプション引数として扱うので、`a` はオプションとして認識されません。この場合、この例の `for` ループは、`filename` 引数を通り越してシフトします。

名前	getoptcv - コマンドオプションを解析するために getopt に変換
形式	<pre>/usr/lib/getoptcv [-b] filename</pre> <pre>/usr/lib/getoptcv</pre>
機能説明	<code>/usr/lib/getoptcv</code> は <code>filename</code> 内のシェルスクリプトを読み取り、<code>getopt</code> の代わりに <code>getopts</code> を使用できるようにそれを変換し、その結果を標準出力に出力します。 <code>getopts</code> は、Bourne シェル組み込みコマンドの1つであり、定位置パラメータの解析およびオプションの妥当性チェックに使用されます。sh(1) を参照してください。このコマンドは、コマンド構文標準規格 (Intro(1) のルール 3 - 10 を参照) のうち、該当するルールすべてをサポートします。このコマンドは、<code>getopt</code> コマンドの代わりに使用してください。(「注意事項」の項を参照してください。)シェルの <code>getopts</code> 組み込みコマンドの構文を次に示します。 <pre>getopts optstring name [argument ...]</pre> <code>optstring</code> には、<code>getopts</code> を使用するコマンドが認識できる オプション文字を入れなければなりません。オプション文字の後にコロン(:)が付いている場合は、そのオプションには1つまたは複数の引数があるとみなされます。オプションと引数の間は、スペースによって区切られていなければなりません。 <code>getopts</code> は、呼び出されるたびに、次のオプションをシェル変数 <code>name</code> に入れ、処理すべき次の引数へのインデックスをシェル変数 <code>OPTARG</code> に入れます。シェルまたはシェル手続きが呼び出されるたびに、<code>OPTARG</code> の値は1に初期設定されます。 オプションにオプション引数が必要な場合、<code>getopts</code> はその引数をシェル変数 <code>OPTARG</code> に入れます。 不正なオプションが検出されると、<code>?</code> が <code>name</code> に入ります。 オプションの終わりが検出されると、<code>getopts</code> はゼロ以外の終了ステータスで終了します。特殊オプション <code>--</code> を使用して、オプションの終わりを表すことができます。 デフォルトでは、<code>getopts</code> は定位置パラメータを解析します。<code>getopts</code> コマンド行に追加の引数 (<code>argument...</code>) が指定されると、<code>getopts</code> は定位置パラメータの代わりにその引数を解析します。 新しいコマンドはすべて、Intro(1) で説明している コマンド構文標準規格に従うようにするために、定位置パラメータの解析、およびオプションがそのコマンドの有効なオプションであるかどうかのチェックに、<code>getopts</code> または <code>getopt</code> を使用しなければなりません(以下の「注意事項」の項を参照してください)。
オプション	次のオプションがサポートされています。 -b 変換されたスクリプトを、UNIX システムの旧リリースへ移植可能にします。<code>/usr/lib/getoptcv</code> は、<code>filename</code> 内のシェルスクリプトを変更し、変更後のシェルスクリプトを実行すると、実行時に <code>getopts</code> と <code>getopt</code> のどちらを呼

び出すか判別できるようにします。

使用例

例1 コマンド引数の処理

次に示すシェルスクリプトの一部では、オプション `-a` または `-b`、およびオプション引数の必要なオプション `-o` を使用するコマンドについて、引数の使用例を示します。

```
while getoptcs abo: c
do
    case $c in
        a | b)    FLAG=$c;;
        o)        OARG=$OPTARG;;
        \?)       echo $USAGE
                  exit 2;;
    esac
done
shift `expr $OPTIND - 1`
```

例2 同一内容のコード式

このコードは、次のコマンドすべてを同一内容として受け入れます。

```
cmd -a -b -o "xxx z yy" filename
cmd -a -b -o "xxx z yy" -filename
cmd -ab -o xxx,z,yy filename
cmd -ab -o "xxx z yy" filename
cmd -o xxx,z,yy b a filename
```

環境変数

`getopts` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_CTYPE、LC_MESSAGES、およびNLSPATH。

OPTIND この変数は、次に処理すべき引数の索引として `getoptcv` によって使用されます。

OPTARG この変数は、オプションが引数を使用している場合に、引数を格納するために `getoptcv` によって使用されます。

終了ステータス 次の終了ステータスが返されます。

0 `optstring` で指定した、または指定しなかったオプションが見つかりました。

>0 オプションの並びの終わりを検出した、またはエラーが発生しました。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

属性タイプ	属性値
CSI	有効

関連項目 [Intro\(1\)](#), [getopts\(1\)](#), [sh\(1\)](#), [shell_builtins\(1\)](#), [getopt\(3C\)](#), [attributes\(5\)](#)

診断 `getopts` は、`optstring` に含まれていないオプション文字を検出すると、標準エラー出力にエラーメッセージを表示します。

注意事項 現在の実装においては、次のようにコマンド構文規格（[Intro\(1\)](#)を参照）に従わない使用方法が認められていますが、このような使用法は、将来、システムのリリースではサポートされないので、使用しないでください。前述の「使用例」と同様に、`-a` および `-b` がオプションであり、`-o` はオプション引数を持つオプションとします。次の例では、ルール 5 (オプション引数を持つオプションは別のオプションと一緒にまとめることはできない) に違反しています。

```
example% cmd -abxxx filename
```

次の例では、ルール 6 (オプション引数を持つオプションの後には、スペースがなければならぬ) に違反しています。

```
example% cmd -ab oxxx filename
```

シェル変数 `OPTIND` の値を変更したり、異なる引数の集まりを解析したりした場合の結果は不確定です。

名前	getopts - ユーティリティーオプションの解析
形式	/usr/bin/getopts <i>optstring name</i> [<i>arg...</i>]
sh	getopts <i>optstring name</i> [<i>argument</i>]...
ksh88	getopts <i>optstring name</i> [<i>arg</i>]...
ksh	getopts [-a <i>name</i>] <i>optstring name</i> [<i>arg</i>]...

機能説明

/usr/bin/getopts getopts ユーティリティを使えば、引数リストからオプションやオプション引数を検索できます。

getopts は、呼び出されるたびに、*name* オペランドで指定されているシェル変数に次のオプションを入れ、シェル変数 OPTIND に処理すべき次の引数へのインデックスを入れます。シェルが呼び出されるたびに、OPTIND は 1 に初期設定されます。

オプション引数を必要とするオプションの場合、getopts はオプション引数をシェル変数 OPTARG に入れます。オプションがない場合または見つかったオプションにオプション引数がない場合、OPTARG は設定されません。

オプション文字の位置に、*optstring* オペランドに含まれていないオプション文字があると、*name* で指定されているシェル変数が疑問符 (?) 文字に設定されます。この場合、*optstring* の最初の文字がコロン (:) なら、シェル変数 OPTARG は見つかったオプション文字に設定されますが、標準エラー出力には何も出力されません。それ以外の場合、シェル変数 OPTARG は設定されず、標準エラー出力には診断メッセージが出力されます。この状況は、呼び出しアプリケーションに引数を渡す方法においてエラーがあったと判断され、getopts 処理のエラーとはなりません。

オプション引数がない場合は、以下のように処理されます。

- *optstring* の最初の文字がコロン (:) なら、*name* で指定されているシェル変数はコロン文字に設定され、シェル変数 OPTARG は見つかったオプション文字に設定されます。
- それ以外の場合、*name* で指定されているシェル変数は疑問符 (?) 文字に設定され、シェル変数 OPTARG は設定されず、標準エラー出力には診断メッセージが出力されます。この状況は、呼び出しアプリケーションに引数を渡す方法においてエラーがあったと判断され、getopts 処理のエラーとはなりません。この判断を示した診断メッセージが出力されますが、終了ステータスは 0 になります。

オプションの最後にくると、getopts はゼロより大きい戻り値で終了します。シェル変数 OPTIND は最初に現われた非オプション引数のインデックスに設定されます。このとき、最初に現われた -- 引数は、他の非オプション引数が先に現われない場合にオプションとしてみなされます。非オプション引数が 1 つもない場合は、\$# + 1 に設定され、*name* 変数は疑問符 (?) 文字になります。オプションの終わりを判断するのは、特殊オプション -- がある場合、- 以外で始まる引数が見つかった場合、またはエラーが発生した場合です。

シェル変数 `OPTIND` と `OPTARG` は `getopts` の呼び出し元のローカル変数であり、デフォルトではエクスポートされません。

name オペランドで指定されているシェル変数、`OPTIND`、および `OPTARG` は、現在のシェル実行環境に影響を及ぼします。

アプリケーションが `OPTIND` を 1 に設定すると、新しいパラメータセットが使用できるようになります。現在の位置パラメータまたは新しい *arg* 値です。すべての呼び出しで同じパラメータ (位置パラメータまたは *arg* オペランド) を使おうとしない場合、または `OPTIND` を 1 以外の値に変更した場合、1 つのシェル実行環境で `getopts` を複数回呼び出すと、結果は不確定です。

sh

`getopts` は、Bourne シェル組み込みコマンドの 1 つであり、定位置パラメータの解析およびオプションの妥当性チェックに使用されます。[sh\(1\)](#) を参照してください。このコマンドは、コマンド構文標準規格 ([Intro\(1\)](#) のルール 3 - 10 を参照) のうち、該当するルールすべてをサポートします。このコマンドは、`getopt` コマンドの代わりに使用してください。

optstring には、`getopts` を使用するコマンドが認識できるオプション文字を入れなければなりません。オプション文字の後にコロンが付いている場合は、そのオプションには 1 つまたは複数の引数があるとみなされます。オプションと引数の間は、スペースによって区切られていなければなりません。

`getopts` は、呼び出されるたびに、次のオプションをシェル変数 *name* に入れ、処理すべき次の引数へのインデックスをシェル変数 `OPTIND` に入れます。シェルまたはシェル手続きが呼び出されるたびに、`OPTIND` の値は 1 に初期設定されます。

オプションにオプション引数が必要な場合、`getopts` はその引数をシェル変数 `OPTARG` に入れます。

不正なオプションが検出されると、`?` が *name* に入ります。

オプションの終わりが検出されると、`getopts` はゼロ以外の終了ステータスで終了します。特殊オプション `-` を使用して、オプションの終わりを表すことができます。

デフォルトでは、`getopts` は定位置パラメータを解析します。`getopts` コマンド行に追加の引数 (*argument...*) が指定されると、`getopts` は定位置パラメータの代わりにその引数を解析します。

`/usr/lib/getoptcvt` は *filename* 内のシェルスクリプトを読み取り、`getopt` の代わりに `getopts` を使用できるようにそれを変換し、その結果を標準出力に出力します。

新しいコマンドはすべて、[Intro\(1\)](#) で説明しているコマンド構文標準規格に従うようにするために、定位置パラメータの解析、およびオプションがそのコマンドの有効なオプションであるかどうかのチェックに、`getopts` または `getopt` を使用しなければなりません。

getopts は、*optstring* に含まれていないオプション文字を検出すると、標準エラー出力にエラーメッセージを表示します。

現在の実装においては、次のようにコマンド構文標準規格 (Intro(1) を参照) に従わない使用方法が認められていますが、このような使用法は、将来、システムのリリースではサポートされないので、使用しないでください。以下のセクションの例と同様に、*-a* および *-b* がオプションであり、*-o* はオプション引数を持つオプションとします。

次の例では、ルール 5 (オプション引数を持つオプションは別のオプションと一緒にまとめることはできない) に違反しています。

```
example% cmd -aboxxx filename
```

次の例では、ルール 6 (オプション引数を持つオプションの後には、スペースがなければならない) に違反しています。

```
example% cmd -ab oxxx filename
```

シェル変数 *OPTIND* の値を変更したり、異なる引数の集まりを解析したりした場合の結果は不確定です。

ksh88

arg が正当なオプションを示しているかどうかをチェックします。*arg* を省略すると、定位置パラメータが使用されます。オプション引数は *+* または *-* で始まります。*+* または *-* 以外の文字で始まっているオプション、また *-* 引数があると、オプションの終わりとみなされます。*optstring* には、getopts が認識する文字を記述します。文字のあとに *:* が続く場合、そのオプションには引数があるとみなされます。オプションと引数とは空白で区切ることができます。

arg が *+* で始まる場合、getopts は *+* を起動するたびに *name* 内で見つかった次のオプション文字を設定します。次の *arg* のインデックスは *OPTIND* に格納されます。オプション引数がある場合は *OPTARG* に格納されます。

optstring 内で先頭に *:* がある場合、getopts は無効なオプション文字を *OPTARG* に格納し、*name* を *?* (未定義のオプションが指定された場合) または *:* (必要なオプション引数が省略されている場合) に設定します。getopts はエラーメッセージを表示します。オプションがなくなると、終了ステータスは 0 以外になります。

getopts は、従来の単一文字の短いオプションも、Sun の Command Line Interface Paradigm (CLIP) で定義された長いオプションもサポートします。

長いオプションは短いオプションの別名です。長いオプションを指定するには、対応する短いオプションのあとに、長いオプションを括弧に囲んで指定します。たとえば、短いオプション「*f*」の別名として、長いオプション「*file*」を指定する場合、次のスクリプト行を使用します。

```
getopts "f(file)" opt
```

コマンド行上では、長いオプションの前に「--」または「++」を指定します。前述の例の場合、コマンド行上の「--file」は「-f」と同等であり、「++file」は「+f」と同等です。

短いオプションの別名として長いオプションを複数指定することはできますが、これはCLIP仕様には違反することであり、使用するときには十分注意する必要があります。長いオプションを複数指定するときには、次のように、それぞれの長いオプションを括弧で囲む必要があります。

```
getopts "f:(file)(input-file)o:(output-file)"
```

前述の例の場合、「--file」と「--input-file」は両方とも「-f」と同等であり、「--output-file」は「-o」と同等です。

変数名は、常に、短いオプションに設定されます。コマンド行上で長いオプションを指定しても、変数名はそれと同等の短いオプションに設定されます。

シェルの `getopts` 組み込みコマンドに関する詳細は、このマニュアルページ内で前述した Bourne シェル (sh) の説明を参照してください。

ksh

`getopts` ユーティリティーを使えば、`arg` で指定された引数リストまたは定位置パラメータ (`arg` が省略されている場合) から、オプションや引数を検索できます。また、`optstring` の情報に基づいて、コマンドの使用法を示すメッセージやマニュアルページを生成することもできます。

`getopts` は、呼び出されるたびに、`name` オペランドで指定されているシェル変数に次のオプションを入れ、シェル変数 `OPTARG` に処理すべき次の引数へのインデックスを入れます。シェルが呼び出されると、`OPTARG` が 1 に初期化されます。オプションにオプション引数が必要であるか、許可される場合、`getopts` はそのオプション引数をシェル変数 `OPTARG` に入れます。それ以外の場合、`OPTARG` はオプションの設定時に 1 に、オプションの設定解除時に 0 にそれぞれ設定されます。

`optstring` 文字列は、英数字、特殊文字 (+、-、?、:)、スペース文字 (SPACE)、または [...] で囲まれた文字グループで構成されます。文字グループは、{...} 内に入れ子にできます。[...] グループの外にある 1 つの復帰改行文字 (NEWLINE) とそれに続く 0 個以上の空白文字は無視されます。オプションとコマンド引数の形式は、1 つ以上の空白行で区切られます。

各 [...] グループは、ラベル (省略可)、属性 (区切り文字は:、省略可)、および ? に続く説明文字列 (省略可) で構成されます。? から次の] の末尾までの文字は、オプション解析および短い使用法メッセージとして無視されます。これらは、詳細なヘルプまたはマニュアルページの生成に使用されます。: 文字をラベルに表示することはできません。ラベルでは ? 文字を ?? と指定し、説明文字列では] 文字を]] と指定する必要があります。2 つの \b (バックスペース) 文字の間のテキストは、そのテキストがボールド (太文字) で表示されることを示しています。2 つの \a (ベル) 文字の間のテキストは、そのテキストが強調またはイタリックで表示されることを示しています。2 つの \v (垂直タブ) 文字の間のテキストは、そのテキストが固定幅

フォントで表示されることを示しています。2つの `\f` (フォームフィード) 文字の間のテキストは、囲まれたテキストを名前とするシェル関数からの出力で置き換えられます。

このインタフェースからの出力はすべて、標準エラー出力に書き込まれます。

グループの種類を次に示します。

■ 次の形式のグループ

`[-[version][flag[number]]...[?text]]`

は最初のグループとして表示され、拡張インタフェースを使用可能にします。

version にはインタフェースのバージョン (現在は 1) を指定します。バージョンが省略された場合は、最新バージョンと見なされます。将来の拡張により *version* の値は大きくなってゆきますが、すべてのバージョンがサポートされます。通常、*text* には SCCS または CVS 識別文字列を指定します。数値 (省略可) を含むゼロ以上のフラグを指定して、オプション解析を制御できます。次のフラグがあります。

- c 複数回受け渡しできるように、この *optstring* をキャッシュします。同じプロセス内で何度も呼び出し可能な組み込みコマンドを最適化する場合に使用します。
 - i ヘルプの生成時にこの *optstring* を無視します。複数の受け渡しから得られる *optstring* 値を統合する場合に使用します。
 - l ヘルプメッセージに長いオプション名だけを表示します。
 - o オプション文字の接頭辞 - は省略可能です。これは、廃止された **ps(1)** オプション構文に対応しています。
 - p この数値は、長いオプション名の接頭辞として追加する必要のある - 文字の数を示します。デフォルト値は 2 です。指定可能な値は 0、1、2 です。たとえば、**dd(1M)** の場合は **p0**、**find(1)** の場合は **p1** と指定します。
 - s この数値は、マニュアルページのセクション番号を指定します。デフォルトは 1 です。
- `[option[!][=number][:longname][? text]]` 形式のオプション。この場合、最初のフィールドはオプション文字です。これは、オプションが一致した場合に *name* オペランド内に返される値です。オプション文字が存在しない場合は、2 桁以上の数値を指定してください。長いオプションが一致する場合、この数値が *name* オペランドの値として返されます。オプションの後に ! が続く場合、オプション文字の意味は *longname* の意味の逆になります。値を取らないオプションの場合、OPTARG は、逆オプション文字 ! では 0 に、それ以外の場合は 1 に設定されます。オプションの *=number* は、オプション文字ではなく *name* オペランドに返される数値を示します。*longname* は `--longname` で指定されます。これは、すべての長いオプションのうちで、あいまいさのない最短の接頭辞により照合され

ます。longname フィールド内の*は、追加の文字すべてが正確に一致するならば、その位置までの文字だけが一致する必要があることを示します。longname も説明テキストも存在しないオプションの場合、[と]は省略できます。

- オプション引数の指定。引数を取るオプションの後に文字列値を表す: または数値を表す#を付けてから、オプション引数を指定できます。オプション引数の指定では、フィールド1にオプション引数名が含まれます。「:」で区切られた残りのフィールドは、タイプ名およびゼロ以上の特殊属性ワード(listof、oneof、およびignorecase)になります。最後のフィールドには、「:=default」のようにデフォルトのオプション値を指定できます。オプション引数の指定に続いて、中括弧で囲んだオプション値の説明のリストを記述できます。引数を取る長いオプションは、--longname=valueのように指定します。:や#の後に?が続く場合、そのオプション引数は省略可能です。オプション文字の書式だけが指定されている場合は、次の引数が- または+ で始まるのであれば、オプションの引数値は設定されません。
- オプション値の説明。
- 引数の指定。オプション引数の指定に続いて、有効なオプション引数値のリストを{...}で囲んで指定できます。許可されるそれぞれの値は、[...]の中に値を含めて指定できます。その後に説明を記述できます。
- [+ \ n...] 形式のグループは、... の表す文字を固定幅フォントで表示します。改行は追加しません。
- [+name ?text] 形式のグループでは、セクション名と説明テキストを指定します。nameを省略すると、textは新しい段落に配置されます。
- [-name ?text] 形式のグループでは、IMPLEMENTATION セクションのエントリを指定します。

optstring の先頭文字が+の場合、+で始まる引数もオプションと見なされます。

先頭の: 文字、またはoptstring の先頭の+に続く: は、エラーの処理方法に影響を及ぼします。オプションの処理時にオプション文字またはoptstring で指定されていないlongname 引数を検出すると、name という名前のシェル変数が? 文字に設定されます。シェル変数OPTARG は、検出された文字に設定されます。オプション引数が存在しないか、その値が無効な場合、name が: 文字に設定され、シェル変数OPTARG が検出されたオプション文字に設定されます。先頭に: が存在しない場合、name は? 文字に設定され、OPTARG の設定は解除されます。エラーが発生した場合には、エラーメッセージが標準エラー出力に書き込まれます。

次の場合は、オプションの終わりが発生します。

1. 特別な引数-- が検出されました。
2. 先頭文字が- でない引数が検出されました。
3. ヘルプ引数が指定されています。
4. エラーが発生します。

OPTIND が値 1 に設定されている場合は、新しい引数セットを使用できます。

getopts も使用すると、コマンドの使用方法や詳細な説明を含むヘルプメッセージを生成できます。*args* は次のように指定します。

- ? 使用方法の概要を生成する場合に使用します。
- ?? 詳細な使用方法メッセージを生成する場合に使用します。
- ??man 書式付きのマニュアルページを生成する場合に使用します。
- ??api 簡単に解析できる使用方法メッセージを生成する場合に使用します。
- ??html マニュアルページを html 形式で生成する場合に使用します。
- ??nroff マニュアルページを nroff 形式で生成する場合に使用します。
- ??usage 現在の optstring を一覧表示する場合に使用します。
- ???name version=*n* を一覧表示する場合に使用します。オプション *name* が getopts により認識される場合、*n* は 0 よりも大きくなります。

オプションの終わりが検出されると、getopts は 0 以外の戻り値で終了し、変数 OPTIND が最初の非オプション引数のインデックスに設定されます。

オプション

ksh では次のオプションがサポートされています。

- a *name* 使用方法メッセージ内で、コマンド名の代わりに *name* を使用します。

オペランド

次のオペランドがサポートされています。

- optstring* getopts を呼び出すユーティリティによって識別されるオプション文字が入った文字列。文字の後ろにコロンが付いている場合、そのオプションには引数があることを意味します。引数は別個に指定する必要があります。アプリケーションは、オプション文字とそのオプション引数を別々に指定しなければなりませんが、そのように指定されているかどうかに関わらず、getopts は、引数を取るオプションに続く文字をその引数と解釈します。オプション引数として NULL を渡すには getopts 起動時に明示的に引数とする必要があります。getopt(3C) を参照してください。アプリケーションでは、疑問符 (?) とコロン (:) をオプション文字としては使用できません。英数字以外のオプション文字を使用した場合の結果は不確定です。オプション引数とオプション文字とを別々に指定しないと、OPTARG の値からオプション文字と - が取り除かれます。オプション文字がわからないとき、またはオプション引数がないとき、optstring の最初の文字によって、getopts のふるまいが決まります。

name `getopts` によって、見つかったオプション文字に設定されるシェル変数の名前。

デフォルトでは、`getopts` は、呼び出し側のシェルのプロシージャに引き渡す位置パラメータの構文を解析します。*arg* が指定されていれば、位置パラメータの代わりに解析されます。

使用法

`getopts` は現在のシェル実行環境に影響を及ぼすので、通常、シェルに組み込まれています。以下のように、サブシェルや別のユーティリティ実行環境で呼び出された場合は、

```
(getopts abc value "$@")
nohup getopts ...
find . -exec getopts ... \;
```

この場合、`getopts` は呼び出し側の環境のシェル変数には影響しません。

なお、位置パラメータが変更されても、シェル関数は呼び出し側シェルと `OPTIND` を共有します。引数の構文を解析するために `getopts` を使用したい関数は、処理の最初で `OPTIND` の値を保存し、戻る前に復元するのが一般的です。ただし、呼び出し側シェル用に関数が `OPTIND` を変更する場合があります。

使用例

例1 引数の解析と表示

以下に示すスクリプトは、引数を解析して表示する例です。

```
aflag=
bflag=
while getopts ab: name
do
    case $name in
        a)      aflag=1;;
        b)      bflag=1
                bval="$OPTARG";;
        ?)      printf "Usage: %s: [-a] [-b value] args\n" $0
                exit 2;;
    esac
done
if [ ! -z "$aflag" ]; then
    printf "Option -a specified\n"
fi
if [ ! -z "$bflag" ]; then
    printf 'Option -b "%s" specified\n' "$bval"
fi
shift $((OPTIND - 1))
printf "Remaining arguments are: %s\n" "$@"
```

例2 オプション付きコマンド用の引数の処理

以下のシェルスクリプトの一部は、コマンドに対する引数を処理する方法を示す例です。この例では、オプション引数を伴わない -a または -b と、オプション引数を伴う -o を使用します。

```
while getopts abo: c
do
    case $c in
        a | b)    FLAG=$c;;
        o)        OARG=$OPTARG;;
        \?)       echo $USAGE
                  exit 2;;
    esac
done
shift `expr $OPTIND - 1`
```

例3 等価コード表現

以下のコードは、すべて同じ処理を実行します。

```
cmd -a -b -o "xxx z yy" filename
cmd -a -b -o "xxx z yy" -- filename
cmd -ab -o xxx,z,yy filename
cmd -ab -o "xxx z yy" filename
cmd -o xxx,z,yy -b -a filename
```

環境変数

getopts の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

OPTIND この変数は、次に処理する引数へのインデックスとして getopts が使用します。

OPTARG この変数は、オプションに引数を使用された場合、その引数を保存するために getopts が使用します。

終了ステータス 次の終了ステータスが返されます。

- 0 *optstring* で指定した、または指定しなかったオプションが見つかりました。
- >0 オプションの並びの終わりを検出した、またはエラーが発生しました。

ksh

ksh により、次の終了値が返されます。

- 0 指定されたオプションが見つかりました。
- 1 オプションの並びの終わりを検出しました。
- 2 使用方法または情報のメッセージが生成されました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/getopts、sh、ksh88	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	确实
	標準	standards(5) を参照してください。

ksh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	不确实

関連項目 [Intro\(1\)](#), [getoptcvt\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [ps\(1\)](#), [sh\(1\)](#), [getopt\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 エラーが検出されたとき、*optstring* オペランドの最初の文字がコロン (:) 以外なら、以下の情報が入った診断メッセージが不特定なフォーマットで標準エラー出力に出力されます。

- 呼び出し側のプログラム名がメッセージ内に示されます。呼び出し側のプログラム名は、`getopts` が呼び出された時点で、シェル特殊パラメータ `0` の値を持ちます。次と等価の名前
`basename "$0"`
が使用できます。
- *optstring* にないオプションが見つかった場合、エラーと認識されて、無効なオプション文字がメッセージ内に示されます。
- オプション引数を必要とするオプションが見つかったが、オプション引数がない場合、エラーと認識されて、無効なオプション文字がメッセージ内に示されます。

名前	gettext – メッセージデータベースからのテキスト文字列の取得										
形式	<pre>gettext [-d <i>textdomain</i> --domain=<i>textdomain</i>] [<i>textdomain</i>] <i>msgid</i> gettext -s [-e] [-n] [-d <i>textdomain</i> --domain=<i>textdomain</i>] <i>msgid</i>...</pre>										
機能説明	gettext ユーティリティーは、msgfmt(1) で生成されたメッセージオブジェクトから、文字列 <i>msgid</i> に対応する変換されたテキスト文字列を取得します。メッセージオブジェクト名は、省略可能な引数 <i>textdomain</i> が存在する場合はその引数から、それ以外の場合は TEXTDOMAIN 環境変数から決定されます。ドメインが指定されていない場合や、対応する文字列が見つからない場合、gettext は <i>msgid</i> を出力します。 通常、gettext は /usr/lib/locale/lang/LC_MESSAGES 内でメッセージオブジェクトを探します。ここで、<i>lang</i> はロケール名です。存在する場合は、TEXTDOMAINDIR 環境変数によって、パス名のコンポーネントが <i>lang</i> に置き換えられます。 このコマンドは、タブに対する \t などの C エスケープシーケンスを解釈します。バックスラッシュを出力するには、\\ を使用します。独自の行にメッセージを生成するには、<i>msgid</i> の最後に \n を入力するか、またはこのコマンドを printf(1) とともに使用します。 -s オプションとともに使用された場合、gettext は echo(1) のように動作します。ただし、引数を単純に標準出力にコピーするわけではありません。代わりに、選択されたカタログ内にあるメッセージが変換されます。										
オプション	サポートしているオプションは、次のとおりです。 <table> <tr> <td>-d <i>textdomain</i></td><td></td></tr> <tr> <td>--domain=<i>textdomain</i></td><td><i>textdomain</i> がオペランドとして指定されていない場合は、ドメイン <i>textdomain</i> から変換されたメッセージを取得します。</td></tr> <tr> <td>-e</td><td>-s オプションとともに使用されている場合は、一部のエスケープシーケンスの展開を有効にします。</td></tr> <tr> <td>-n</td><td>-s オプションとともに使用されている場合は、末尾の復帰改行を抑止します。</td></tr> <tr> <td>-s</td><td>echo(1) のように動作します(上の「機能説明」を参照)。-s オプションが指定されている場合、デフォルトでは、C エスケープシーケンスの展開は実行されず、出力に改行文字が付加されます。</td></tr> </table>	-d <i>textdomain</i>		--domain= <i>textdomain</i>	<i>textdomain</i> がオペランドとして指定されていない場合は、ドメイン <i>textdomain</i> から変換されたメッセージを取得します。	-e	-s オプションとともに使用されている場合は、一部のエスケープシーケンスの展開を有効にします。	-n	-s オプションとともに使用されている場合は、末尾の復帰改行を抑止します。	-s	echo(1) のように動作します(上の「機能説明」を参照)。-s オプションが指定されている場合、デフォルトでは、C エスケープシーケンスの展開は実行されず、出力に改行文字が付加されます。
-d <i>textdomain</i>											
--domain= <i>textdomain</i>	<i>textdomain</i> がオペランドとして指定されていない場合は、ドメイン <i>textdomain</i> から変換されたメッセージを取得します。										
-e	-s オプションとともに使用されている場合は、一部のエスケープシーケンスの展開を有効にします。										
-n	-s オプションとともに使用されている場合は、末尾の復帰改行を抑止します。										
-s	echo(1) のように動作します(上の「機能説明」を参照)。-s オプションが指定されている場合、デフォルトでは、C エスケープシーケンスの展開は実行されず、出力に改行文字が付加されます。										
オペランド	次のオペランドがサポートされています。 <table> <tr> <td><i>textdomain</i></td><td>メッセージを取得するために使用されるドメイン名。これにより、-d または --domain オプション(存在する場合)によって指定がオーバーライドされます。</td></tr> </table>	<i>textdomain</i>	メッセージを取得するために使用されるドメイン名。これにより、-d または --domain オプション(存在する場合)によって指定がオーバーライドされます。								
<i>textdomain</i>	メッセージを取得するために使用されるドメイン名。これにより、-d または --domain オプション(存在する場合)によって指定がオーバーライドされます。										

	<i>msgid</i>	ローカライズされたメッセージを取得するための鍵。
環境変数	LANG	ロケール名を指定します。
	LC_MESSAGES	メッセージングのロケールを指定し、存在する場合はメッセージの LANG をオーバーライドします。
	TEXTDOMAIN	テキストドメイン名を指定します。これは、.mo サフィックスのないメッセージオブジェクトファイル名と同じです。
	TEXTDOMAINDIR	メッセージデータベースへのパス名を指定します。存在する場合は、/usr/lib/locale を置き換えます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [echo\(1\)](#), [msgfmt\(1\)](#), [printf\(1\)](#), [gettext\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#)

注意事項 これは、ライブラリルーチン [gettext\(3C\)](#) と等価なシェルです。

名前	gettxt - テキスト文字列のメッセージデータベースからの検索
形式	gettxt <i>msgfile</i> : <i>msgnum</i> [<i>dflt_msg</i>]
機能説明	gettxt は、<code>/usr/lib/locale/locale/LC_MESSAGES</code> というディレクトリ内のメッセージファイルからテキスト文字列を検索します。<code>locale</code> というディレクトリ名は、テキスト文字列を書き込む際の言語に該当します。setlocale(3C) を参照してください。 <i>msgfile</i> <i>msgnum</i> を検索する先の <code>/usr/lib/locale/ locale/LC_MESSAGES</code> ディレクトリ内のファイル名。<i>msgfile</i> の名前は、最大 14 文字の長さにできますが、<code>\0</code> (NULL) や、<code>/</code> (スラッシュ) または <code>:</code> (コロン) などの ASCII コードを入れることはできません。 <i>msgnum</i> <i>msgfile</i> から検索すべき文字列のシーケンス番号。<i>msgfile</i> 内の文字列は、1 から <i>n</i> まで連続して番号を付けます。ここで、<i>n</i> はファイル内の文字列の数です。 <i>dflt_msg</i> gettxt が <i>msgnum</i> の <i>msgfile</i> からの検索に失敗した場合に表示すべきデフォルト文字列。表示不可能な文字は、エスケープシーケンスとしてアルファベットで表す必要があります。

検索すべきテキスト文字列は、*msgfile* ファイル内にあります。このファイルは [mkmsgs\(1\)](#) ユーティリティで作成し、`/usr/lib/locale/locale/LC_MESSAGES` ディレクトリの下にインストールします。`.LC_MESSAGES` という環境変数を設定すると、対応したディレクトリを検索します。`LC_MESSAGES` を設定しないと、`LANG` 環境変数が使用されます。`LANG` を設定しないと、文字列の入ったファイルは、`/usr/lib/locale/C/LC_MESSAGES` ディレクトリの下になります。

gettxt が、要求された言語でのメッセージの検索に失敗すると、`/usr/lib/locale/C/LC_MESSAGES/msgfile` において同一のメッセージを検索します。これも失敗し、*dflt_msg* が存在して NULL 以外である場合、*dflt_msg* の値を表示します。*dflt_msg* が存在しないか NULL である場合、`Message not found!!` という文字列を表示します。

使用例

例1 `LANG` 環境変数および `LC_MESSAGES` 環境変数

`LANG` 環境変数または `LC_MESSAGES` 環境変数をそのデフォルト値以外に設定していない場合、たとえば、

```
example% gettxt UX:10 "hello world\n"
```

では、`/usr/lib/locale/C/UX/msgfile` から 10 番目のメッセージを検索します。検索が失敗すると、`"hello world"` というメッセージが表示され、その後に復帰改行が続きます。

環境変数 gettext の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LC_CTYPE および LC_MESSAGES。

- LC_CTYPE

gettext の文字の処理方法を決定します。LC_CTYPE に有効な値が設定されていると、gettext は、そのロケールに合った文字を含むテキストやファイル名を表示および処理できます。gettext は拡張 Unix コード (EUC) も表示および処理できます。この場合、文字は 1 バイト幅、2 バイト幅、3 バイト幅のいずれも使用できます。また、gettext は 1、2、またはそれ以上のカラム幅の EUC 文字も処理することができます。C ロケールにおいては、ISO 8859-1 の文字だけが有効です。
- LC_MESSAGES

診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、肯定・否定の応答形式も決定します。C ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。

ファイル

/usr/lib/locale/C/LC_MESSAGES/*

[mkmsgs\(1\)](#) が作成するデフォルトのメッセージファイル

/usr/lib/locale/locale/LC_MESSAGES/*

[mkmsgs\(1\)](#) が作成するさまざまな言語用のメッセージファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/locale
CSI	有効

関連項目 [exstr\(1\)](#), [mkmsgs\(1\)](#), [srchtxt\(1\)](#), [gettext\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

名前	getzonepath – 指定したラベルに対応するゾーンのルートパスの表示
形式	/usr/bin/getzonepath {sensitivity-label}
機能説明	getzonepath は、指定した機密ラベルに対応する、実行中のラベル付きゾーンのルートパス名を表示します。返されるパス名は、呼び出し元のルートパス名に対する相対パスで、指定された機密ラベルが付けられています。 呼び出し元が大域ゾーン内に存在する場合、呼び出し側のプロセスが <code>file_dac_search</code> 特権を保持しない限り、返されるパスをたどることはできません。 呼び出し側がラベル付きのゾーン内に存在する場合、呼び出し側のラベルが指定したラベルよりも優位でなければなりません。返されたパス名内部のファイルへのアクセスは、読み取り専用操作に限定されています。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/trusted
安定性	確実

診断	getzonepath は、終了時に次のいずれかの値を返します。 0 成功 1 使用上のエラー 2 異常終了。エラーメッセージとして getzonerootbylabel(3TSOL) のシステムエラー番号が返される

関連項目	getzonerootbylabel(3TSOL) , attributes(5) 『Trusted Extensions Developer’s Guide』の「 Acquiring a Sensitivity Label 」

注意事項	このマニュアルページに記載されている機能を利用できるのは、システムが Trusted Extensions を使用して構成されている場合のみです。

名前 glob – 単語リストを展開するためのシェル組み込み関数

形式

csh glob wordlist

機能説明

csh glob は wordlist に指定される単語リストを使用してファイル名を展開します。エスケープ \ を認識しない点を除いて、echo(1) と同様です。出力のワードは NULL 文字によって区切られます。

属性 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 csh(1), echo(1), attributes(5)

名前 gprof – display call-graph profile data

形式 gprof [-abcDlsz] [-e *function-name*] [-E *function-name*]
 [-f *function-name*] [-F *function-name*]
 [*image-file* [*profile-file*...]]
 [-n *number of functions*]

機能説明

The gprof utility produces an execution profile of a program. The effect of called routines is incorporated in the profile of each caller. The profile data is taken from the call graph profile file that is created by programs compiled with the -xpg option of cc(1), or by the -pg option with other compilers, or by setting the LD_PROFILE environment variable for shared objects. See [ld.so.1\(1\)](#). These compiler options also link in versions of the library routines which are compiled for profiling. The symbol table in the executable image file *image-file* (a.out by default) is read and correlated with the call graph profile file *profile-file* (gmon.out by default).

First, execution times for each routine are propagated along the edges of the call graph. Cycles are discovered, and calls into a cycle are made to share the time of the cycle. The first listing shows the functions sorted according to the time they represent, including the time of their call graph descendants. Below each function entry is shown its (direct) call-graph children and how their times are propagated to this function. A similar display above the function shows how this function's time and the time of its descendants are propagated to its (direct) call-graph parents.

Cycles are also shown, with an entry for the cycle as a whole and a listing of the members of the cycle and their contributions to the time and call counts of the cycle.

Next, a flat profile is given, similar to that provided by [prof\(1\)](#). This listing gives the total execution times and call counts for each of the functions in the program, sorted by decreasing time. Finally, an index is given, which shows the correspondence between function names and call-graph profile index numbers.

A single function may be split into subfunctions for profiling by means of the MARK macro. See [prof\(5\)](#).

Beware of quantization errors. The granularity of the sampling is shown, but remains statistical at best. It is assumed that the time for each execution of a function can be expressed by the total time for the function divided by the number of times the function is called. Thus the time propagated along the call-graph arcs to parents of that function is directly proportional to the number of times that arc is traversed.

The profiled program must call [exit\(2\)](#) or return normally for the profiling information to be saved in the gmon.out file.

オプション

The following options are supported:

-a Suppress printing statically declared functions. If this option is given, all relevant information about the static function (for instance, time samples, calls to other functions, calls from other functions) belongs to the function loaded just before the static function in the a.out file.

-
- b Brief. Suppress descriptions of each field in the profile.
 - c Discover the static call-graph of the program by a heuristic which examines the text space of the object file. Static-only parents or children are indicated with call counts of 0. Note that for dynamically linked executables, the linked shared objects' text segments are not examined.
 - C Demangle C++ symbol names before printing them out.
 - D Produce a profile file `gmon.sum` that represents the difference of the profile information in all specified profile files. This summary profile file may be given to subsequent executions of `gprof` (also with `-D`) to summarize profile data across several runs of an `a.out` file. See also the `-s` option.

As an example, suppose function A calls function B *n* times in profile file `gmon.sum`, and *m* times in profile file `gmon.out`. With `-D`, a new `gmon.sum` file will be created showing the number of calls from A to B as *n-m*.
 - e*function-name* Suppress printing the graph profile entry for routine *function-name* and all its descendants (unless they have other ancestors that are not suppressed). More than one `-e` option may be given. Only one *function-name* may be given with each `-e` option.
 - E*function-name* Suppress printing the graph profile entry for routine *function-name* (and its descendants) as `-e`, below, and also exclude the time spent in *function-name* (and its descendants) from the total and percentage time computations. More than one `-E` option may be given. For example:

`-E mcount -E mcleanup`

is the default.
 - f*function-name* Print the graph profile entry only for routine *function-name* and its descendants. More than one `-f` option may be given. Only one *function-name* may be given with each `-f` option.
 - F*function-name* Print the graph profile entry only for routine *function-name* and its descendants (as `-f`, below) and also use only the times of the printed routines in total time and percentage computations. More than one `-F` option may be given. Only one *function-name* may be given with each `-F` option. The `-F` option overrides the `-E` option.
 - l Suppress the reporting of graph profile entries for all local symbols. This option would be the equivalent of placing all of the local symbols for the specified executable image on the `-E` exclusion list.
 - n Limits the size of flat and graph profile listings to the top *n* offending functions.

- s Produce a profile file `gmon.sum` which represents the sum of the profile information in all of the specified profile files. This summary profile file may be given to subsequent executions of `gprof` (also with `-s`) to accumulate profile data across several runs of an `a.out` file. See also the `-D` option.
- z Display routines which have zero usage (as indicated by call counts and accumulated time). This is useful in conjunction with the `-c` option for discovering which routines were never called. Note that this has restricted use for dynamically linked executables, since shared object text space will not be examined by the `-c` option.

環境変数

PROFDIR If this environment variable contains a value, place profiling output within that directory, in a file named `pid.programname`. `pid` is the process ID and `programname` is the name of the program being profiled, as determined by removing any path prefix from the `argv[0]` with which the program was called. If the variable contains a null value, no profiling output is produced. Otherwise, profiling output is placed in the file `gmon.out`.

ファイル

a.out Executable file containing namelist

gmon.out Dynamic call-graph and profile

gmon.sum Summarized dynamic call-graph and profile

\$PROFDIR/pid.programname

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities

関連項目

[cc\(1\)](#), [ld.so.1\(1\)](#), [prof\(1\)](#), [exit\(2\)](#), [pcsample\(2\)](#), [profil\(2\)](#), [malloc\(3C\)](#), [malloc\(3MALLOC\)](#), [monitor\(3C\)](#), [attributes\(5\)](#), [prof\(5\)](#)

Graham, S.L., Kessler, P.B., McKusick, M.K., gprof: A Call Graph Execution Profiler Proceedings of the SIGPLAN '82 Symposium on Compiler Construction, SIGPLAN Notices, Vol. 17, No. 6, pp. 120-126, June 1982.

『リンカーとライブラリ』

注意事項

If the executable image has been stripped and does not have the `.symtab` symbol table, `gprof` reads the global dynamic symbol tables `.dynsym` and `.SUNW_ldynsym`, if present. The symbols in the dynamic symbol tables are a subset of the symbols that are found in `.symtab`. The `.dynsym` symbol table contains the global symbols used by the runtime linker. `.SUNW_ldynsym` augments the information in `.dynsym` with local function symbols. In the case where `.dynsym`

is found and `.SUNW_ldynsym` is not, only the information for the global symbols is available. Without local symbols, the behavior is as described for the `-a` option.

`LD_LIBRARY_PATH` must not contain `/usr/lib` as a component when compiling a program for profiling. If `LD_LIBRARY_PATH` contains `/usr/lib`, the program will not be linked correctly with the profiling versions of the system libraries in `/usr/lib/libp`.

The times reported in successive identical runs may show variances because of varying cache-hit ratios that result from sharing the cache with other processes. Even if a program seems to be the only one using the machine, hidden background or asynchronous processes may blur the data. In rare cases, the clock ticks initiating recording of the program counter may *beat* with loops in a program, grossly distorting measurements. Call counts are always recorded precisely, however.

Only programs that call `exit` or return from `main` are guaranteed to produce a profile file, unless a final call to `monitor` is explicitly coded.

Functions such as `mcount()`, `_mcount()`, `moncontrol()`, `_moncontrol()`, `monitor()`, and `_monitor()` may appear in the `gprof` report. These functions are part of the profiling implementation and thus account for some amount of the runtime overhead. Since these functions are not present in an unprofiled application, time accumulated and call counts for these functions may be ignored when evaluating the performance of an application.

64-bit profiling

64-bit profiling may be used freely with dynamically linked executables, and profiling information is collected for the shared objects if the objects are compiled for profiling. Care must be applied to interpret the profile output, since it is possible for symbols from different shared objects to have the same name. If name duplication occurs in the profile output, the module id prefix before the symbol name in the symbol index listing can be used to identify the appropriate module for the symbol.

When using the `-s` or `-D` option to sum multiple profile files, care must be taken not to mix 32-bit profile files with 64-bit profile files.

32-bit profiling

32-bit profiling may be used with dynamically linked executables, but care must be applied. In 32-bit profiling, shared objects cannot be profiled with `gprof`. Thus, when a profiled, dynamically linked program is executed, only the *main* portion of the image is sampled. This means that all time spent outside of the *main* object, that is, time spent in a shared object, will not be included in the profile summary; the total time reported for the program may be less than the total time used by the program.

Because the time spent in a shared object cannot be accounted for, the use of shared objects should be minimized whenever a program is profiled with `gprof`. If desired, the program should be linked to the profiled version of a library (or to the standard archive version if no profiling version is available), instead of the shared object to get profile information on the functions of a library. Versions of profiled libraries may be supplied with the system in the `/usr/lib/libp` directory. Refer to compiler driver documentation on profiling.

Consider an extreme case. A profiled program dynamically linked with the shared C library spends 100 units of time in some libc routine, say, `malloc()`. Suppose `malloc()` is called only from routine B and B consumes only 1 unit of time. Suppose further that routine A consumes 10 units of time, more than any other routine in the *main* (profiled) portion of the image. In this case, `gprof` will conclude that most of the time is being spent in A and almost no time is being spent in B. From this it will be almost impossible to tell that the greatest improvement can be made by looking at routine B and not routine A. The value of the profiler in this case is severely degraded; the solution is to use archives as much as possible for profiling.

使用上の留意点

Parents which are not themselves profiled will have the time of their profiled children propagated to them, but they will appear to be spontaneously invoked in the call-graph listing, and will not have their time propagated further. Similarly, signal catchers, even though profiled, will appear to be spontaneous (although for more obscure reasons). Any profiled children of signal catchers should have their times propagated properly, unless the signal catcher was invoked during the execution of the profiling routine, in which case all is lost.

名前	grep - ファイルにおけるパターンの検索
形式	<pre> /usr/bin/grep [-c -l -q] [-bhinsvw] <i>limited-regular-expression</i> [<i>filename</i>]... /usr/xpg4/bin/grep [-E -F] [-c -l -q] [-bhinsvx] -e <i>pattern_list</i>... [-f <i>pattern_file</i>]... [<i>file</i>]... /usr/xpg4/bin/grep [-E -F] [-c -l -q] [-bhinsvx] [-e <i>pattern_list</i>]... -f <i>pattern_file</i>... [<i>file</i>]... /usr/xpg4/bin/grep [-E -F] [-c -l -q] [-bhinsvx] <i>pattern</i> [<i>file</i>]...</pre>
機能説明	grep ユーティリティーは、テキストファイルにおいてパターンを検索し、そのパターンを含むすべての行を出力します。grep は、単純な非決定性アルゴリズムを使用します。 <i>pattern_list</i> での \$、*、[、^、 、(、)、\ などの文字の使い方に注意してください。これらの文字は、シェルにとっても特別な意味があるからです。次のように、<i>pattern_list</i> 全体を単一引用符で囲む('...')のがもっとも安全です。 ファイルが指定されない場合は、grep は、標準入力を入力とみなします。検索された各行は、通常、標準出力に出力されます。複数の入力ファイルがある場合は、検索された各行の前にファイル名が出力されます。
/usr/bin/grep	/usr/bin/grep ユーティリティーは、 regex(5) のマニュアルページに説明されている正規表現に似た限定正規表現を使用して、パターンマッチングを行います。
/usr/xpg4/bin/grep	-E と -F の両オプションは、/usr/xpg4/bin/grep の <i>pattern_list</i> の解釈の仕方に影響を与えます。-E を指定すると、/usr/xpg4/bin/grep は <i>pattern_list</i> を完全な正規表現と解釈します(詳細は -E の説明を参照)。-F を指定すると、grep は <i>pattern_list</i> を固定文字列と解釈します。両方のオプションを省略すると、grep は <i>pattern_list</i> を基本正規表現と解釈します(詳細は regex(5) のマニュアルページを参照)。
オプション	次のオプションは、/usr/bin/grep と /usr/xpg4/bin/grep の両方のコマンドでサポートされています。 -b 検索された各行の先頭にその行のブロック番号を付けます。文脈によりブロック番号を見つけ出す場合に便利です(最初のブロックは0です)。 -c パターンを含む行の数だけを出力します。 -h 一致した行を含むファイル名をその行の先頭に追加しません。複数のファイルを検索するときに使用されます。 -i 比較中に大文字と小文字の区別を無視します。 -l 一致した行があるファイルのファイル名だけを、1 ファイルずつ復帰改行文字で区切って出力します。パターンが2度以上見つかるときは、ファイルの名前を繰り返しません。

- n 各行の先頭にファイル内の行番号を付けます (最初の行は1です)。
- q 非出力。行が一致したかどうかに関わらず、標準出力には何も書き出しません。入力行に一致した行があった場合、0の終了ステータスで処理を終了します。
- s 存在しないファイルや読めないファイルに対してのエラーメッセージを抑制します。
- v パターンを含む行を除いたすべての行を出力します。
- w 表現が\<と\>で囲まれている場合のように、表現を単語として検索します。

/usr/xpg4/bin/grep 次のオプションは、/usr/xpg4/bin/grep のみでサポートされています。

- e *pattern_list* 入力の検索時に用いる1つまたは複数のパターンを指定します。複数のパターンを *pattern_list* に指定する場合、各々を復帰改行文字 (NEWLINE) で区切る必要があります。空のパターンは、復帰改行文字を2つ連続して *pattern_list* に指定することにより表されます。-E または -F オプションも指定した場合を除き、各パターンは基本正規表現として扱われます。grep コマンドを使って、複数の -e および -f オプションを指定することが可能です。行のマッチングにおいて、指定されたパターンはすべて使用されますが、評価の順序は不特定です。
- E 完全な正規表現を使ってマッチングを行います。指定した各パターンは完全な正規表現として扱われます。ある完全な正規表現の全体が入力行と一致したとき、その行は一致したことになります。空の完全な正規表現は、すべての行と一致します。各パターンは、[regex\(5\)](#) のマニュアルページに記述されているように、完全な正規表現として解釈されます。ただし \< (や \) は例外で、以下の指定も含まれます。
 1. 後ろに + が付いている完全な正規表現。これは、その完全な正規表現が1回以上現れるものに一致します。
 2. 後ろに ? が付いている完全な正規表現。これは、その完全な正規表現が0回または1回現れるものに一致します。
 3. | または復帰改行で区切られた複数の完全な正規表現。これは、どちらかの完全な正規表現と一致する文字列に一致します。
 4. グループ化のために、丸かっこ () で囲まれた完全な正規表現。

演算子の優先順位は、[], * ? +、次が連結、その次が | と復帰改行となります。

- f *pattern_file* *pattern_file* が示すパス名で指定したファイルから、1つまたは複数のパターンを読み込みます。*pattern_file* 中のパターンの終了は、復帰改行文字により表されます。空のパターンは、*pattern_file* 中に空の行を記述することにより表せます。**-E** または **-F** オプションも指定した場合を除き、各パターンは基本正規表現として扱われます。
- F 固定文字列を使ってマッチングを行います。指定された各パターンを、正規表現ではなく文字列と見なします。いずれかのパターンが連続したバイトとして入力行の中に現れた場合、その行は一致します。空の文字列はすべての入力行と一致します。詳細については、[fgrep\(1\)](#) を参照してください。
- x 固定文字列全体または正規表現と完全に一致する入力行だけを、一致する行とみなします。

オペランド 次のオペランドがサポートされています。

file パターンを検索するファイルのパス名。*file* オペランドを指定しないと、標準入力を使用されます。

/usr/bin/grep *pattern* 入力の検索時に用いるパターンを指定します。

/usr/xpg4/bin/grep *pattern* 入力の検索時に用いる1つまたは複数のパターンを指定します。オペランドは **-e *pattern_list*** が指定されたものとして扱われます。

使用法 **-c**、**-l**、および **-q** オプションを同時に指定することはできません。同時に指定した場合、**-q** は **-c** よりも優先され、**c** は **-l** よりも優先されます。

-e *pattern_list* オプションの指定は、*pattern_list* オペランドと同じ結果をもたらしますが、*pattern_list* がハイフン区切り文字で始まる場合には便利な方法です。また、複数のパターンを別個の引数として使用したい場合にも便利な方法です。

-e と **-f** の両オプションは、それぞれ複数個指定できます。入力テキスト行のマッチングにおいて、**grep** は指定されたすべてのパターンを使用します。なお、評価の順序は指定されないので注意してください。空の文字列をパターンとして使用したいときは、すべての行と一致するそのパターンを最初に記述し、それ以外のパターンをうまく無視させることができます。

-q オプションは、一群のファイル中に特定のパターンまたは文字列が存在しているかどうかを簡単に見分ける手段として使えます。複数のファイルを検索する場合、最初に一致するものを見つけたところで処理を終了するので、性能が向上します。また、引数として複数のファイルを設定する場合でも、それほど注意は必要ありません。なぜなら、前の方の *file* オペランドで設定したファイルでアクセスエラーや読み取りエラーが発生しても、そのあとで一致するものが見つかれば、**grep** は 0 の終了ステータスで終了するためです。

大規模ファイルの動作

ファイルが2G バイト (2^{31} バイト) 以上ある場合の `grep` の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 ある文字列をすべて検索する

以下の例は、`text.mm` というファイルを検索し、`Posix` という文字列 (大文字と小文字は区別しない) が現れている箇所すべてを行番号とともに出力します。

```
example% /usr/bin/grep -i -n posix text.mm
```

例2 空の行をすべて検索する

次の例は、標準入力中の空の行をすべてを見つけるものです。

```
example% /usr/bin/grep ^$
```

または

```
example% /usr/bin/grep -v .
```

例3 ある文字列を含む行を検索する

以下のコマンドはすべて、`abc` と `def` という2つの文字列のどちらかまたは両方を含んでいる行をすべて出力します。

```
example% /usr/xpg4/bin/grep 'abc
def'
example% /usr/xpg4/bin/grep -e 'abc
def'
example% /usr/xpg4/bin/grep -e 'abc' -e 'def'
example% /usr/xpg4/bin/grep -E 'abc|def'
example% /usr/xpg4/bin/grep -E -e 'abc|def'
example% /usr/xpg4/bin/grep -E -e 'abc' -e 'def'
example% /usr/xpg4/bin/grep -E 'abc
def'
example% /usr/xpg4/bin/grep -E -e 'abc
def'
example% /usr/xpg4/bin/grep -F -e 'abc' -e 'def'
example% /usr/xpg4/bin/grep -F 'abc
def'
example% /usr/xpg4/bin/grep -F -e 'abc
def'
```

例4 ある文字列と完全に一致する行を検索する

以下の2つのコマンドはともに、`abc` または `def` のどちらかの文字列と完全に一致している行をすべて出力します。

```
example% /usr/xpg4/bin/grep -E '^abc$ ^def$'
example% /usr/xpg4/bin/grep -F -x 'abc def'
```

環境変数 `grep` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 一致するものが1つ以上見つかりました。
- 1 一致するものが1つも見つかりませんでした。
- 2 (一致するものが見つかった場合でも) 構文エラーが検出された、またはアクセスできないファイルがありました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/grep	属性タイプ	属性値
	使用条件	system/core-os
	CSI	未対応

/usr/xpg4/bin/grep	属性タイプ	属性値
	使用条件	system/xopen/xcu4
	CSI	有効
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

関連項目 [egrep\(1\)](#), [fgrep\(1\)](#), [sed\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [regexp\(5\)](#), [standards\(5\)](#)

注意事項

/usr/bin/grep 1 行は仮想記憶に使用できるサイズに制限されています。入力行中に空文字がある場合、`grep` は最初の空文字までマッチングを行います。最初の空文字までのマッチングで一致した場合は、行全体が出力されます。

/usr/xpg4/bin/grep 入力ファイルが `LINE_MAX` バイトよりも長い行を含んでいる場合、またはバイナリデータを含んでいる場合の結果は不確定です。`LINE_MAX` は `/usr/include/limits.h` で定義されます。

名前	groups – ユーザーのグループメンバーシップの出力
形式	groups [user]...
機能説明	groups コマンドは、このコマンドを実行したユーザーの、または指定したユーザーの所属グループを標準出力に書き込みます。各ユーザーは /etc/passwd 内に指定されているグループに属し、さらに /etc/group 内に指定されているほかのグループにも属している可能性があります。/etc/passwd では、グループの ID (gid) は数値で指定されています。groups コマンドは gid をグループ名に変換して出力します。
使用例	出力の形式は次のとおりです。 example% groups tester01 tester02 tester01 : staff tester02 : staff example%
ファイル	/etc/passwd /etc/group
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

名前	groups – display a user's group memberships				
形式	/usr/ucb/groups [user]...				
機能説明	With no arguments, groups displays the groups to which you belong; else it displays the groups to which the user belongs. Each user belongs to a group specified in the password file /etc/passwd and possibly to other groups as specified in the file /etc/group. If you do not own a file but belong to the group which it is owned by then you are granted group access to the file.				
ファイル	/etc/passwd /etc/group				
属性	See attributes(5) for descriptions of the following attributes:				
	<table><tr><th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr><tr><td>Availability</td><td>compatibility/ucb</td></tr></table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	compatibility/ucb
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	compatibility/ucb				
関連項目	getgroups(2) , attributes(5)				
注意事項	This command is obsolete.				

名前	grpck – check group database entries
形式	/usr/sbin/grpck [filename]
機能説明	The grpck utility checks that a file in group(4) does not contain any errors; it checks the /etc/group file by default.
ファイル	/etc/group
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [groups\(1\)](#), [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

診断	Too many/few fields	An entry in the group file does not have the proper number of fields.
	No group name	The group name field of an entry is empty.
	Bad character(s) in group name	The group name in an entry contains characters other than lower-case letters and digits.
	Invalid GID	The group ID field in an entry is not numeric or is greater than 65535.
	Null login name	A login name in the list of login names in an entry is null.
	Logname not found in password file	A login name in the list of login names in an entry is not in the password file.
	Line too long	A line (including the NEWLINE character) in the group file exceeds the maximum length of 512 characters.
	Duplicate logname entry	A login name appears more than once in the list of login names for a group file entry.
	Out of memory	The program cannot allocate memory in order to continue.
	Maximum groups exceeded for logname	A login name's group membership exceeds the maximum, NGROUPS_MAX.

名前	hash, rehash, unhash, hashstat – ディレクトリの内容の内部ハッシュテーブルの評価
形式	<pre>/usr/bin/hash [utility] /usr/bin/hash [-r]</pre>
sh	<pre>hash [-r] [name]...</pre>
csch	<pre>rehash unhash hashstat</pre>
ksh88	<pre>hash [name]... hash [-r]</pre>

機能説明

`/usr/bin/hash` ユーティリティーは、見つかったユーティリティーの位置を現在のシェル環境がどのように記憶するか、その記憶方法を変更します。具体的には、指定された引数に従って、新たなユーティリティーをユーティリティー位置リストに追加したり、リストの内容を消去したりします。引数を指定しないと、リストの内容が報告されます。-r オプションを指定すると、シェルは記憶したすべての位置を忘れます。

シェルの組み込みユーティリティーとして提供されているものについては、hash は報告対象としません。

sh シェルは、各 *name* ごとに、それが示すコマンドの検索パス内の位置を決定し、記憶します。-r オプションを指定すると、シェルは記憶したすべての位置を破棄します。引数が与えられない場合、hash は記憶されたコマンドに関する情報を表示します。この出力表示において、*Hits* の列はシェルプロセスによってコマンドが呼び出された回数を表します。*Cost* は、検索パスのコマンドを見つけるのに必要な作業量です。コマンドが検索パスの「相対」ディレクトリにある場合、そのディレクトリの変更後にそのコマンドが格納された位置が再計算されます。この再計算が行われる予定のコマンドに対しては、*Hits* 情報の隣にアスタリスク (*) が示されます。*Cost* の値は、再計算が行われるたびに増加します。

csch `rehash` は新しく追加されたコマンドに合わせて、`path` 環境の変数に記憶しているディレクトリの内容の内部ハッシュテーブルを再計算します。

`unhash` は内部ハッシュテーブルを使用不能にします。

`hashstat` は内部ハッシュテーブルがコマンドの検索(実行を伴わない)にどの程度有効に働いているかを示す統計情報を出力します。ハッシュ関数がヒットの可能性を示しているパスの各コンポーネントと、'/' で始まらない各コンポーネントについて、実行しようとしています。

ksh88 シェルは、各 *name* ごとに、それが示すコマンドの検索パス内の位置を決定し、記憶します。**-r** オプションを指定すると、シェルは記憶したすべての位置を破棄します。引数が与えられない場合、**hash** は記憶されたコマンドに関する情報を表示します。

オペランド **hash** では次のオペランドがサポートされています。

utility 位置のリスト内で検索する、あるいはリストに追加するユーティリティ名。

出力 引数を 1 つも指定しないと、**hash** の標準出力が使用されます。その形式は決まっていますが、現在のシェル環境用の位置リストにある各ユーティリティのパス名は含まれています。このリストは、以前に実行された **hash** 呼び出し中に指定されていたユーティリティをすべて含んでおり、さらに通常のコマンド検索プロセスで呼び出され見つけられたユーティリティを含んでいることもあります。

環境変数 **hash** の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。**LANG**、**LC_ALL**、**LC_CTYPE**、**LC_MESSAGES**、および **NLSPATH**。

PATH **utility** の位置を指定します。

終了ステータス 次の終了値が返されます。

0 正常終了。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [csh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	head - ファイルの最初の数行の表示
形式	<code>/usr/bin/head [-number -n number] [filename]...</code>
機能説明	head ユーティリティーは、<i>filename</i> で指定された各ファイルについて、先頭の何行か (行数は <i>number</i> で指定) を標準出力に出力します。<i>filename</i> が指定されない場合、head は標準入力から行をコピーします。<i>number</i> のデフォルト値は 10 行です。 複数のファイルが指定された場合、各ファイルの先頭は次のようになります。 <pre>==> filename <==</pre> したがって、複数の小さいファイルの内容を、各ファイルを識別できるように表示する際、一般的な方法として、次のようにこのコマンドを使用できます。 <pre>example% head -9999 filename1 filename2 ...</pre>
オプション	サポートしているオプションは、次のとおりです。 -n number 各入力ファイルの先頭の何行を標準出力に書き出すか、その行数を <i>number</i> で指定します。<i>number</i> の値は正の整数でなければなりません。 -number <i>number</i> は正の整数で、意味は前述の -n number 指定と同じです。 オプションが指定されていない場合、head は -n 10 が指定されたものとして動作します。
オペランド	次のオペランドを指定できます。 <i>filename</i> 入力ファイルのパス名。<i>file</i> オペランドを指定しないと、標準入力が使われます。
使用法	ファイルが 2G バイト (2 ³¹ バイト) 以上ある場合の head の動作については、 largefile(5) を参照してください。
使用例	例1 ディレクトリ中のすべてのファイルの先頭 10 行を出力する 次の例は、ピリオドで始まるものを除くディレクトリ中のすべてのファイルの先頭の 10 行を出力します。 <pre>example% head *</pre>
環境変数	head の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [cat\(1\)](#), [more\(1\)](#), [pg\(1\)](#), [tail\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前 history, fc, hist – コマンドの履歴リストの処理

形式 /usr/bin/fc [-r] [-e editor] [first [last]]
 /usr/bin/fc -l [-nr] [first [last]]
 /usr/bin/fc -s [old=new] [first]

csh history [-hr] [n]

ksh88 fc -e - [old = new] [command]
 fc -s [old = new] [command]
 fc [-e ename] [-nlr] [first [last]]

ksh hist [-lnprs] [-e editor][-N num][first[last]]

機能説明

/usr/bin/fc fc ユーティリティーは、以前に対話型 sh に入力されたコマンドの一覧表示、または編集と再実行を行います。

コマンドの履歴リストは、番号によってコマンドを参照します。リスト中の最初の番号は任意に選択できます。番号とコマンドとの関係は変わることはありません。ただしユーザーがログインしたときに、どのプロセスもリストをアクセスしていない場合を除きます。この場合には、システムは番号をリセットするので、保存されているもっとも古いコマンドは別の番号 (通常は 1) で開始できます。番号が、HISTSIZE の値と 32767 の大きい方の値に到達したとき、シェルは番号を循環させるので、次のコマンドはより小さい数字 (通常は 1) で開始できます。ただし、このように番号が循環しても、fc は時間の順序は把握しています。たとえば 4 つのコマンドにそれぞれ 32766、32767、1 (循環後)、2 という番号が与えられたとき、32767 は 1 よりも大きい値であっても 1 以前に実行されたものであると認識しています。

コマンドが編集されたとき (-l オプション省略時)、結果として生成された行が履歴リストの最後尾に記録され、sh によって再実行されます。編集作業を起動した fc コマンド自身は履歴リストには記録されません。エディタがゼロ以外の終了ステータスを返した場合、履歴リストへの記録や再実行は抑制されます。fc コマンドに指定したコマンド行変数割り当てやリダイレクト演算子は、fc コマンド自身および生成されるコマンドの両方に対して有効です。次の例を見てください。

```
fc -s -- -l 2>/dev/null
```

この例は以前のコマンドを再度呼び出すものですが、fc および以前のコマンドの両方に対して標準エラー出力を抑制します。

csh 履歴のリストを表示します。n を指定すると、n 個の最新の履歴を表示します。
 -r 出力を、古い履歴からではなく最近の履歴から順に並べ変えます。

-h 先頭に番号を付加せずに履歴のリストを出力します。-h オプションを使用することにより、csh 組み込みコマンドの `source(1)` の入力に適したファイルを出力できます。

履歴置換:

履歴置換を使用すれば、以前入力したコマンド行のワードを、これから入力するコマンド行で使用できます。これにより、綴りの訂正、複雑なコマンドや引数の繰り返し入力が簡単になります。コマンド行は履歴リストに保存されます。履歴リストのサイズは `history` 変数によって変更できます。`history` シェル変数は、履歴ファイルに保存されるコマンド行の最大数に設定されます。たとえば

```
set history = 200
```

は、履歴リストに最新の 200 コマンド行のトラックを保管します。設定を行わない場合、C シェルは最新のコマンドだけを保存します。

履歴置換の指定は `!` で始まり (`histchars` 変数によりほかの文字に変更可能)、コマンド行のどこに現われてもかまいません。ただし履歴置換のネストはできません。`!` を `\` でエスケープすれば、その特別な意味を抑制できます。

履歴置換を含んだ入力行は、展開された後で、他の置換が起こる前またはコマンドが実行される前に、端末上に表示されます。

イベント指示子

イベント指示子は、履歴リスト内のコマンド行エントリを参照するものです。

<code>!</code>	次の文字がスペース文字、タブ、復帰改行、 <code>=</code> 、または <code>(</code> でなければ、履歴置換を開始します。
<code>!!</code>	直前のコマンドを指します。他の文字を加えずにこれだけを入力すると、直前のコマンドを繰り返します。
<code>!n</code>	<i>n</i> 番のコマンド行を指します。
<code>!-n</code>	入力中のコマンドから <i>n</i> 個前のコマンドを指します。
<code>!str</code>	<i>str</i> で始まる最新のコマンドを指します。
<code>!?str?</code>	<i>str</i> を含む最新のコマンドを指します。
<code>!?str? additional</code>	<i>str</i> を含む最新のコマンドを指し、その参照コマンドに <i>additional</i> を追加します。
<code>!{command} additional</code>	<i>command</i> で始まる最新のコマンドを指し、その参照コマンドに <i>additional</i> を追加します。

`^previous_word^replacement^` 文字列 `previous_word` を文字列 `replacement` に置き換えて、直前のコマンド行を繰り返します。これは履歴置換と同等です。

文字列 `previous_word` を文字列 `replacement` に置き換えて、直前のコマンド行を繰り返します。これは履歴置換と同等です。

`!!:s/previous_word/replacement/.`

以前の特定のコマンドを再実行し、さらに同じような置換を行う場合、たとえば6番目のコマンドを再実行するには、次のようにします。

`!!:6s/previous_word/replacement/.`

ワード指示子:

A `:'` (コロン) はイベント指示子とワード指示子とを区切ります。ワード指示子が `^`、`$`、`*`、`-`、または `%` で始まるときは省略できます。直前のコマンドからワードを選択する場合、イベント指示子の2番目の `!` は省略できます。たとえば、`!!:1` と `!!:1` は、両方とも直前のコマンドの最初のワードを指します。また、`!!$` と `!!$` は、両方とも直前のコマンドの最後のワードを指します。ワード指示子には以下のものがあります。

`#` 今までに入力したすべてのコマンド行
`0` 最初に入力したワード (コマンド)
`n` n 番目の引数
`^` 最初の引数。1 と同じ
`$` 最後の引数
`%` 最新の `?s` による検索で一致したワード
`x-y` ワードの範囲。 `-y` は `0-y` の省略形
`*` すべての引数。イベント中に1ワードしか存在しないときは NULL
`x*` `x-$` の省略形
`x-` `x*` と同じであるが、ワード `$` を省略

修飾子:

オプションのワード指示子の後に、`:` で始まる1つ以上の修飾子を追加できます。

`h` パス名の後部分のコンポーネントを除去して、前部分を残します。
`r` `'...xxx'` の形の接尾辞を除去して、ベース名を残します。

e	接尾辞以外はすべて除去して、拡張部分を残します。
s/ <i>oldchars</i> / <i>replacements</i> /	<i>replacements</i> を <i>oldchars</i> に置換します。 <i>oldchars</i> は組み込まれたスペースを含む文字列ですが、イベント指示子にある <i>previous_word</i> はスペースを含みません。 ^ <i>oldchars</i> ^ <i>replacements</i> ^
t	パス名の前部分のコンポーネントを除去して、後部分を残します。
&	前の置換を繰り返します。
g	各ワード内の最初の一致が発生した箇所を、前述のオプションに接頭辞を付けて変更します(たとえば <i>g&</i>)。
p	新しいコマンドを表示するだけで、実行はしません。
q	置換されたワードをクォートして、それ以上の置換をエスケープします。
x	<i>q</i> と同じですが、スペース文字、タブ、復帰改行文字 (NEWLINE) ごとにワードに分割します。

g を先頭に付加しないと、*oldchars* に一致する最初の文字列だけが変更されます。一致する文字列がなければ、エラーとなります。

置換部分の左側は正規表現ではなく文字列です。/ の箇所には、区切り文字としてどのような文字でも使用できます。区切り文字用の文字はバックスラッシュで囲まれます。右側の & 文字は、左側のテキストで置換されます。& はバックスラッシュでクォートすることができます。 *oldchars* が NULL のとき、直前の文字列における *oldchars*、または *!?s* における文脈検索文字列 *s* を使用します。同様に *replacements* の直後に復帰改行がある場合、文脈検索の最右にある ? は省略できます。

イベントが指定されないと、履歴リファレンスは前のコマンドか、(もしあれば) そのコマンド行上での前の履歴リファレンスを参照します。

ksh88

次の形式で *fc* を指定します。

```
fc -e - [old=new] [command],
```

または

```
fc -s [old=new] [command],
```

この場合、*old=new* の置換を行なったあとで *command* が再実行されます。 *command* 引数を省略すると、最後に行なったコマンドが実行されます。

次の形式で *fc* を指定します。

```
fc [-e ename] [-nlr] [first [last]],
```

この場合、端末から最近入力された HISTSIZE 個のコマンドの中から、*first* から *last* までの範囲のコマンドを選択します。*first* と *last* の両引数は、数値または文字列で指定できます。文字列の場合、その文字列で始まる最新のコマンドを見つけます。負の数値は、現在のコマンド番号からのオフセットとなります。*-l* オプションを指定すると、標準出力上にコマンドを一覧表示します。*-l* を指定しないと、これらのキーボードコマンドの入ったファイル上で *e ename* で示すエディタプログラムを起動します。*ename* が省略されていると、変数 FCEDIT (デフォルトは /bin/ed) の値をエディタとして使用します。編集が完了すると、編集されたコマンドを実行します。*last* を指定しない場合、*first* と同一値に設定されます。*first* を指定しない場合、デフォルトは、編集については直前のコマンドに、一覧表示については -16 になります。*-r* オプションはコマンドの順序を逆にします。*-n* オプションは一覧表示時にコマンド番号の出力を抑制します(コマンド行編集の詳細については [ksh88\(1\)](#) を参照)。

HISTFILE シェルの起動時にこの変数が設定されていると、その値はコマンド履歴を格納するために使用されるファイルのパス名になります。

HISTSIZE シェルの起動時にこの変数が設定されていると、このシェルで使用可能な入力済みコマンドの数がこの値以上になります。デフォルト値は 128 です。

コマンド再入力:

端末装置から最近入力された HISTSIZE が示す個数 (デフォルトは 128 個) のコマンドのテキストは、履歴ファイルに保存されています。`$HOME/.sh_history` というファイルは、**HISTFILE** 変数が設定されていない場合、または変数が示すファイルが書き込み不可能な場合に使用されます。シェルは、同じ名前の **HISTFILE** を使用する対話型シェルすべてのコマンド履歴を使用できます。**fc** という特殊コマンドは、このファイルの一部をリスト表示または編集するときに使用します。編集またはリスト表示されるファイルの部分は、番号か、またはコマンドの最初の文字を指定することによって選択できます。単一のコマンドを指定することも、コマンドの範囲を指定することも可能です。**fc** の引数としてエディタプログラムが指定されていないと、**FCEDIT** という変数の値が使用されます。**FCEDIT** が未定義の場合は、`/bin/ed` が使われます。編集されたコマンドは、エディタを終了した時点で表示および再実行されます。エディタ名に `-` を指定すると、編集段階が省かれ、コマンドが再実行されます。この場合、`old=new` という形式の代入パラメータを使用すれば、実行前にコマンドを変更できます。たとえば、*r* が '`fc -e`' の別名として定義されているとき '`r bad=good c`' と入力すると、*c* という文字で始まるコマンドのうち最新のものが、その記述中の最初の *bad* という文字列を *good* に置き換えられて再実行されます。

複合コマンドの中に **fc** 組み込みコマンドを指定すると、すべてのコマンドが履歴ファイルから削除されます。

ksh

`hist` は、現在のシェル環境に以前に入力されたコマンドの一覧表示、編集、または再実行を行います。

コマンドの履歴リストは、番号によってコマンドを参照します。リスト中の最初の番号は任意に選択できます。ログインセッション中は、番号とそのコマンドの関係が変わることはありません。番号は 32767 に達すると 1 に戻りますが、順序は維持されます。

`l` オプションが指定されておらず、コマンドが編集された場合、結果の行は、履歴リストの末尾に入力され、現在のシェルによって再実行されます。編集作業を起動した `hist` コマンド自身は履歴リストには記録されません。エディタがゼロ以外 (non-zero) の終了ステータスを返した場合、履歴リストへの記録や再実行は抑制されます。コマンド行変数代入とリダイレクトは、`hist` コマンドおよび再実行されるコマンドの両方に対して有効です。

`first` と `last` は、コマンドの範囲を定義します。`first` と `last` には、次のいずれかを指定します。

- `number` コマンド番号を表す正の整数。+ 記号を `number` の前に指定できます。
- `-number` いくつ前のコマンドかを示す負の数。たとえば前回実行したコマンドなら -1 となります。
- `string` `string` は、`string` で始まるコマンドのうち、最後に実行したコマンドを示します。`string` には = を含めないようにしてください。

`first` を省略すると、前回のコマンドが使用されます。ただし、`-l` を指定した場合は、デフォルトで `first` は -16 になり、`last` は -1 になります。

`first` を指定し、`last` を省略した場合、`last` はデフォルトで `first` になります。ただし、`-l` を指定した場合は、デフォルトで -1 になります。

エディタを指定しない場合、`HISTEDIT` 変数 (設定されている場合) または `FCEDIT` 変数 (設定されている場合) によって指定されたエディタが使用されます。それ以外の場合は、`ed` が使用されます。

オプション

サポートしているオプションは、次のとおりです。

- `-e editor` `editor` が示すエディタを使ってコマンドを編集します。文字列 `editor` はユーティリティー名で、`PATH` 変数の値に従って検索されます。`-e` を指定しない場合、`FCEDIT` 変数の値がデフォルトとして用いられます。`FCEDIT` の値が NULL または未設定のときは、エディタとして `ed` が使用されます。
- `-l` (小文字のエル)エディタを呼び出して編集する代わりに、コマンドを一覧表示します。`first` と `last` の両オペランドで指定した範囲のコマンドを、`-r` オプションがあればそれに従って、順番にコマンド番号付きで表示します。

- n -l オプションによる一覧表示において、コマンド番号を出力しません。
- r コマンドの一覧表示 (-l 指定時) または編集 (-l および -s 省略時) において、順序を逆にします。
- s エディタを呼び出さずにコマンドを再実行します。

ksh

ksh では次のオプションがサポートされています。

- e *editor* history コマンドの編集に使用するエディタを指定します。*editor* の値は、-s オプションを指定するのと同様です。
- l コマンドの編集および再実行を行わずに、コマンドを一覧表示します。
- N *num* *num* コマンド前から開始します。
- n コマンドを一覧表示するときに、コマンド番号を抑制します。
- p 各オペランドの履歴展開の結果を標準出力に書き込みます。その他のすべてのオプションは無視されます。
- r コマンドの順序を逆にします。
- s エディタを呼び出さずにコマンドを再実行します。この場合、*old=new* 形式のオペランドを指定すると、コマンドで最初に出現した文字列 *old* を *new* に変更してからコマンドを再実行できます。

オペランド

次のオペランドがサポートされています。

first
last

表示または編集するコマンドを選択します。いくつまでさかのぼってコマンドをアクセスできるかは、HISTSIZE 変数の値により決まります。*first* と *last* の値は、それぞれ次のいずれかです。

[+]*number* コマンド番号を表す正の整数。過去に実行した各コマンドの番号は、-l オプションを使えば確認できます。

-*number* いくつ前のコマンドかを示す負の整数。たとえば直前に実行したコマンドなら -1 となります。

string 指定した文字列で始まっていたコマンドのうち、最後に実行したコマンド。*old=new* オペランドが -s オプションなしで指定された場合、文字列形式の *first* オペランドを使用するなら、その文字列中に等記号を含めることはできません。

「形式」の節で示した形式で、-s を指定する場合、*first* を省略すると、直前のコマンドが使用されます。

「形式」の項で示した形式で、-s を指定しない場合、

- *last* を省略すると、デフォルト値は *l* 指定時は直前のコマンドとなり、*-l* 省略時は *first* の値となります。
- *first* と *last* の両方を省略すると、*-l* 指定時は直前の 16 個のコマンドの表示、*l* 省略時は直前の 1 つのコマンドの編集となります。
- *first* と *last* の両方を指定すると、*first* から *last* までのすべてのコマンドが表示 (*-l* 指定時) または編集 (*-l* 省略時) されます。複数のコマンドを一度に、それぞれを新たな行で開始してエディタに渡せば、複数コマンドの編集が可能です。*first* が示すコマンドが *last* が示すものより新しい場合、*-r* 指定時と同じように逆の順序で表示または編集されます。次の例では、1 行目のコマンドは、2 行目の対応するコマンドと同等です。

```
fc -r 10 20      fc      30 40
fc      20 10      fc -r 40 30
```

- コマンドの範囲を使用する場合、履歴リストに存在していない値を *first* や *last* に指定してもエラーとはなりません。*fc* は、存在しているもっとも古いまたは新しい番号を、その代わりに使用します。たとえば、履歴リスト中に 10 個のコマンドが記録されていて、そのコマンド番号が 1 から 10 となっているとします。

```
fc -l
fc 1 99
```

この場合、いずれも 10 個のコマンドすべてを表示または編集することになります。

old=new 再実行対象のコマンド中に最初に現れた文字列 *old* を、ほかの文字列 *new* に置き換えます。

出力

-l オプションを使ってコマンドを表示する場合、その出力形式は次のとおりです。

```
"%d\t%s\n", <line number>, <command>
```

-l と *-n* の両オプションを指定すると、各コマンドの出力形式は次のようになります。

```
"\t%s\n", <command>
```

command が複数の行で構成されている場合、2 行目以降は次のように表示されます。

```
"\t%s\n", <continued-command>
```

使用例

例1 history と fc の使用例

csch	ksh88
% history	\$ fc -l
1 cd /etc	1 cd /etc
2 vi passwd	2 vi passwd
3 date	3 date
4 cd	4 cd
5 du .	5 du .
6 ls -t	6 ls -t
7 history	7 fc -l
% !d	\$ fc -e - d
du .	du .
262 ./SCCS	262 ./SCCS
336 .	336 .
% !da	\$ fc -e - da
Thu Jul 21 17:29:56 PDT 1994	Thu Jul 21 17:29:56 PDT 1994
%	\$ alias \!= 'fc -e - '
% !!	\$!
date	alias = 'fc -e - '
Thu Jul 21 17:29:56 PDT 1994	

環境変数

fc の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_CTYPE、LC_MESSAGES、 および NLSPATH。

- FCEDIT

この変数は、シェルによって展開されると、`-e editor` オプションの *editor* のデフォルト値を定義します。FCEDIT の値が NULL または未設定のときは、エディタとして [ed\(1\)](#) が使用されます。
- HISTFILE

コマンドの履歴ファイル名を表すパス名を定義します。HISTFILE 変数が設定されていないと、シェルはユーザーのホームディレクトリ内で `.sh_history` というファイルをアクセスまたは生成しようとします。この履歴ファイルへの読み込みおよび書き込みアクセスができない、または生成できない場合、シェルは何らかのメカニズムを使って履歴が正しくとれるようにします。(この項で述べる履歴ファイルの参照とは、このメカニズムが使用される場合があることを前提としています。)履歴ファイルの初期化時にのみ fc をアクセスするようにすることもできます。初期化は、fc または sh がユーザーからのコマンドに従って、最初にこのファイル、または ENV 変数が指定するファイル、または `/etc/profile` のようなシステム起動ファイルからのエントリ検索もしくはエントリの追加を試みたときに発生します。(なお履歴ファイル用の初期化処理は、システム起動ファイルの内容に依存させ

ることもできます。つまり、ユーザーが設定した HISTFILE や HISTSIZE 値を効果的に上書きするようなコマンドを、このファイルに記述することが可能です。たとえば `set -o no log` オプションが設定されていなければ、関数定義コマンドが履歴ファイルに記録されます。ENV ファイルの前に呼び出されるシステムスタートアップファイル中に、システム管理者が関数定義を記述しておけば、ユーザーが履歴ファイルの属性を変更するような動作が可能になる前に、履歴ファイルが初期化されます。)シェルが呼び出されると、最初に HISTFILE 変数が参照されます。ほかのシェルが呼び出されるまでは、HISTFILE に対する変更は反映されません。

HISTSIZE 以前のコマンドを最大いくつまでさかのぼってアクセスできるかを 10 進数を使って定義します。この変数が設定されていないと、128 以上の不特定の値がデフォルトとして用いられます。シェルが呼び出されると、最初に HISTSIZE 変数が参照されます。ほかのシェルが呼び出されるまでは、HISTSIZE に対する変更は反映されません。

終了ステータス 次の終了ステータスが返されます。

- `0` 一覧表示の正常終了。
- `>0` エラーが発生した。

それ以外の場合、終了ステータスは、fc または hist によって実行されたコマンドの終了ステータスとなります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [csh\(1\)](#), [ed\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [set\(1\)](#), [sh\(1\)](#), [source\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#)

名前	hostid - 現在のホストの数値識別子の出力
形式	/usr/bin/hostid
機能説明	hostid コマンドは、現在のホストの識別子を 16 進数で出力します。ホスト識別子をエミュレートする非大域ゾーン内で実行した場合、エミュレートされたホスト識別子が出力されます。通常、hostid を別のマシンで実行した場合、この数値は異なります。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [sysinfo\(2\)](#), [gethostid\(3C\)](#), [attributes\(5\)](#), [zones\(5\)](#)

名前	hostname – 現在のホストシステムの名前の設定または出力
形式	<code>/usr/bin/hostname [[-t] name-of-host]</code>
機能説明	<code>hostname</code> コマンドは、 <code>login</code> プロンプトの前に表示されている現在のホストの名前を出力します。スーパーユーザーは、引数を指定してホスト名を設定できます。 <code>-t</code> オプションを指定しない限り、ホスト名の変更は永続的です。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [uname\(1\)](#), [nodename\(4\)](#), [attributes\(5\)](#)

名前 iconv - コードセット変換ユーティリティー

形式 iconv [-cs] -f *frommap* -t *tomap* [*file*]...
iconv -f *fromcode* [-cs] [-t *tocode*] [*file*]...
iconv -t *tocode* [-cs] [-f *fromcode*] [*file*]...
iconv -l

機能説明 iconv ユーティリティーは、*file* で示すファイル中にある文字群または一連の文字群に対し、コードセットを変換してその結果を標準出力に書き出します。出力側のコードセットが対応する文字を持っていない文字は、実装で定義された文字に変換されます。

サポートされる変換処理の一覧と変換テーブルの場所は、[iconv\(5\)](#) のマニュアルページに記載されています。

オプション サポートしているオプションは、次のとおりです。

-c 入力ファイルのコードセットにおいて無効な文字を出力しません。-c オプションを指定しない場合、入力ストリームで無効な文字に遭遇したときの結果は、指定した変換用コードセットによって異なります。無効な文字とは、入力ファイルのコードセットにおいて無効な文字であるか、出力ファイルのコードセットにおいて対応する文字がない文字であるかのどちらかです。-c オプションは iconv の終了ステータスには影響しません。--f オプションの *fromcodeset* に *fromcode* を指定した場合、あるいは、-t オプションの *tocodeset* に *tocode* を指定した場合、-c オプションは無視されます。

-f *fromcodeset* 入力ファイルのコードセットを設定します。*fromcodeset* オプション引数は、次の2つの形式が認識されます。

fromcode *fromcode* オプション引数には、スラッシュ (/) 文字を使用してはいけません。これはコードセット記述の名前として解釈されます。

frommap *frommap* オプション引数には、スラッシュ文字を使用する必要があります。これは [charmap\(5\)](#) で定義されている *charmap* ファイルのパス名として解釈されます。このパス名が有効で読み取り可能な *charmap* ファイルを参照していない場合、その結果は定義されていません。

このオプションを指定しない場合、現在のロケールのコードセットが使用されます。

-l サポートされるすべての *fromcode* と *tocode* の値を標準出力に書き出します。

-s 無効な文字に関するメッセージを標準エラーに書き込みませ
 ん。-s を使用しない場合、入力ストリームで無効な文字に遭遇し
 たときの結果は、指定した変換用コードセットによって異なりま
 す。無効な文字とは、入力ファイルのコードセットにおいて無効
 な文字であるか、出力ファイルのコードセットにおいて対応する
 文字がない文字であるかのどちらかです。-s オプションは iconv
 の終了ステータスには影響しません。-f オプションの *fromcodeset*
 に *fromcode* を指定した場合、あるいは、-t オプションの *tocodeset*
 に *tocode* を指定した場合、-s オプションは無視されます。

-t *tocodeset* 出力ファイルのコードセットを設定します。 *tocodeset* オプション
 引数は、次の2つの形式が認識されます。

tocode *tocode* オプション引数には、スラッシュ (/) 文字を使用
 してはいけません。これはコードセット記述の名前とし
 て解釈されます。

tomap *tomap* オプション引数には、スラッシュ文字を使用する
 必要があります。これは [charmap\(5\)](#) で定義されている
 charmap ファイルのパス名として解釈されます。このパ
 ス名が有効で読み取り可能な charmap ファイルを参照し
 ていない場合、その結果は定義されていません。

このオプションを指定しない場合、現在のロケールのコード
 セットが使用されます。

-f オプションまたは -t オプションの一方を charmap として指定して、他方を
 charmap として指定しないか、他方を指定しない場合、あるいは、-f オプションま
 たは -t オプションを両方とも指定しない場合、iconv は失敗して、エラーになりま
 す。

オペランド 次のオペランドがサポートされています。

file 入力ファイルのパス名。file オペランドが指定されない、または - の場
 合、標準入力を使用されます。

使用例 例1 ファイルを変換して保存する

次の例は、ファイル mail1 の内容を 8859 から 646fr へコードセット変換し、その結
 果をファイル mail.local に書き出します。

```
example% iconv -f 8859 -t 646fr mail1 > mail.local
```

環境変数 iconv の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照し
 てください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

1

エラーが発生しました。
- ファイル

/usr/lib/iconv/iconv_data

変換テーブルでサポートしている変換のリスト
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目

[iconv\(3C\)](#), [iconv_open\(3C\)](#), [attributes\(5\)](#), [charmap\(5\)](#), [environ\(5\)](#), [iconv\(5\)](#), [iconv_unicode\(5\)](#), [standards\(5\)](#)
- 注意事項

2つのコードセットに共通な文字には、両方の charmap ファイルが同じシンボリック名を使用していることを確認してください。

-l オプションの出力形式は定義されていません。-l オプションはシェルスクリプトで使用するようには設計されていません。

fromcode または *toctype* をコードセット変換用に指定した場合、iconv は [iconv_open\(3C\)](#) 関数を使用します。iconv_open(3C) が指定されたコードセット変換を開くのに失敗した場合、iconv は適切な変換テーブルを検索します。iconv_open(3C) がサポートするコード変換については、[iconv\(5\)](#) および [iconv_locale\(5\)](#) を参照してください。
- 626

マニュアルページセクション 1: ユーザーコマンド • 最終更新日 2003 年 11 月 14 日

名前	indxbib – create an inverted index to a bibliographic database								
形式	indxbib <i>database-file</i> ...								
機能説明	indxbib makes an inverted index to the named <i>database-file</i> (which must reside within the current directory), typically for use by lookbib(1) and refer(1). A <i>database</i> contains bibliographic references (or other kinds of information) separated by blank lines. A bibliographic reference is a set of lines, constituting fields of bibliographic information. Each field starts on a line beginning with a '%', followed by a key-letter, then a blank, and finally the contents of the field, which may continue until the next line starting with '%'. indxbib is a shell script that calls two programs: /usr/lib/refer/mkey and /usr/lib/refer/inv. mkey truncates words to 6 characters, and maps upper case to lower case. It also discards words shorter than 3 characters, words among the 100 most common English words, and numbers (dates) < 1000 or > 2099. These parameters can be changed. indxbib creates an entry file (with a .ia suffix), a posting file (.ib), and a tag file (.ic), in the working directory.								
ファイル	<pre>/usr/lib/refer/mkey /usr/lib/refer/inv</pre> <table> <tr> <td><i>x.ia</i></td><td>entry file</td></tr> <tr> <td><i>x.ib</i></td><td>posting file</td></tr> <tr> <td><i>x.ic</i></td><td>tag file</td></tr> <tr> <td><i>x.ig</i></td><td>reference file</td></tr> </table>	<i>x.ia</i>	entry file	<i>x.ib</i>	posting file	<i>x.ic</i>	tag file	<i>x.ig</i>	reference file
<i>x.ia</i>	entry file								
<i>x.ib</i>	posting file								
<i>x.ic</i>	tag file								
<i>x.ig</i>	reference file								
属性	See attributes(5) for descriptions of the following attributes:								
	<table> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> <tr> <td>Availability</td><td>text/doctools</td></tr> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	text/doctools				
ATTRIBUTE TYPE	ATTRIBUTE VALUE								
Availability	text/doctools								
関連項目	addbib(1) , lookbib(1) , refer(1) , roffbib(1) , sortbib(1) , attributes(5)								
使用上の留意点	All dates should probably be indexed, since many disciplines refer to literature written in the 1800s or earlier. indxbib does not recognize pathnames.								

名前	install – install files
形式	<pre> /usr/ucb/install [-cs] [-g group] [-m mode] [-o owner] filename1 filename2 /usr/ucb/install [-cs] [-g group] [-m mode] [-o owner] filename... directory /usr/ucb/install -d [-g group] [-m mode] [-o owner] directory </pre>
機能説明	install is used within makefiles to copy new versions of files into a destination directory and to create the destination directory itself. The first two forms are similar to the cp(1) command with the addition that executable files can be stripped during the copy and the owner, group, and mode of the installed file(s) can be given. The third form can be used to create a destination directory with the required owner, group and permissions. Note: install uses no special privileges to copy files from one place to another. The implications of this are: ■ You must have permission to read the files to be installed. ■ You must have permission to copy into the destination file or directory. ■ You must have permission to change the modes on the final copy of the file if you want to use the -m option to change modes. ■ You must be superuser if you want to specify the ownership of the installed file with -o. If you are not the super-user, or if -o is not in effect, the installed file will be owned by you, regardless of who owns the original.
オプション	-c Copy files. In fact install <i>always</i> copies files, but the -c option is retained for backwards compatibility with old shell scripts that might otherwise break. -d Create a directory. Missing parent directories are created as required as in <code>mkdir -p</code>. If the directory already exists, the owner, group and mode will be set to the values given on the command line. -s Strip executable files as they are copied. -g <i>group</i> Set the group ownership of the installed file or directory. (staff by default.) -m <i>mode</i> Set the mode for the installed file or directory. (0755 by default.) -o <i>owner</i> If run as root, set the ownership of the installed file to the user-ID of <i>owner</i>.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [chgrp\(1\)](#), [chmod\(1\)](#), [chown\(1\)](#), [cp\(1\)](#), [mkdir\(1\)](#), [strip\(1\)](#), [install\(1M\)](#), [attributes\(5\)](#)

名前	ipcrm - メッセージキュー、セマフォセット、または共有メモリー ID の削除
形式	ipcrm [-z zone] [-m shmid] [-q msgqid] [-s semid] [-M shmkey] [-Q msgkey] [-S semkey]
機能説明	ipcrm は、1 つ以上のメッセージ、セマフォ、または共有メモリー識別子を削除します。
オプション	次のオプションがサポートされています。 -z zone ほかのオプションで指定された鍵は、指定されたゾーン内の機能を示しています (zones(5) を参照)。デフォルトは、このコマンドが実行されているゾーンです。このオプションは、コマンドが大域ゾーンで実行されている場合にのみ役立ちます。 これらの識別子は、次のオプションで指定されます。 -m shmid システムから共有メモリー識別子 <i>shmid</i> を削除します。最後の切り離しのあと、共有メモリーセグメントとそれに関連付けられたデータ構造が破棄されます。 -q msgqid システムからメッセージキュー識別子 <i>msgqid</i> を削除し、メッセージキューとそれに関連付けられたデータ構造を破棄します。 -s semid システムからセマフォ識別子 <i>semid</i> を削除し、セマフォのセットとそれに関連付けられたデータ構造を破棄します。 -M shmkey システムから、鍵 <i>shmkey</i> を使用して作成された共有メモリー識別子を削除します。最後の切り離しのあと、共有メモリーセグメントとそれに関連付けられたデータ構造が破棄されます。 -Q msgkey システムから、鍵 <i>msgkey</i> を使用して作成されたメッセージキュー識別子を削除し、メッセージキューとそれに関連付けられたデータ構造を破棄します。 -S semkey システムから、鍵 <i>semkey</i> を使用して作成されたセマフォ識別子を削除し、セマフォのセットとそれに関連付けられたデータ構造を破棄します。 削除の詳細は、msgctl(2)、shmctl(2)、および semctl(2) で説明されています。識別子と鍵を検索するには、<code>ipcs</code> コマンドを使用します。
環境変数	ipcrm の実行に影響を与える環境変数 <code>LANG</code> 、 <code>LC_ALL</code> 、 <code>LC_CTYPE</code> 、 <code>LC_MESSAGES</code> 、および <code>NLSPATH</code> については、 environ(5) を参照してください。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目

[ipcs\(1\)](#), [msgctl\(2\)](#), [msgget\(2\)](#), [msgrcv\(2\)](#), [msgsnd\(2\)](#), [semctl\(2\)](#), [semget\(2\)](#), [semop\(2\)](#), [shmctl\(2\)](#), [shmget\(2\)](#), [shmop\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#), [zones\(5\)](#)

名前	ipcs – プロセス間通信機能のステータスの報告
形式	ipcs [-aAbciJmopqstZ] [-D <i>mtype</i>] [-z <i>zone</i>]
機能説明	ipcs ユーティリティは、アクティブなプロセス間通信機能に関する情報を出力します。表示される情報は指定したオプションによって制御されます。オプションを指定しない場合、システムで現在アクティブなメッセージキュー、共有メモリー、およびセマフォに関する情報が短形式で出力されます。
オプション	サポートしているオプションは、次のとおりです。 -m アクティブな共有メモリーセグメントに関する情報を出力します。 -q アクティブなメッセージキューに関する情報を出力します。 -s アクティブなセマフォに関する情報を出力します。 -m、-q、または -s を指定した場合、指定されたオプションに関する情報のみが出力されます。いずれも指定しない場合、次のオプションに従って、3つのオプションすべてに関する情報が出力されます。 -a すべてのXCU5出力オプションを使用します。(これは、-b、-c、-o、-p、および-tの簡略表記です)。 -A すべての出力オプションを使用します。(これは、-b、-c、-i、-J、-o、-p、および-tの簡略表記です)。 -b 許容される最大サイズ(メッセージキューにおけるキューのメッセージの最大バイト数、共有メモリーにおけるセグメントの最大サイズ、およびセマフォの各セットにおけるセマフォの最大数)に関する情報を出力します。一覧の列の意味については、下記を参照してください。 -c 作成者のログイン名とグループ名を出力します。下記を参照。 -D <i>mtype</i> ipcs を呼び出したユーザーが読み取り権を持つメッセージキューで見つかった <i>mtype</i> タイプのすべてのメッセージの内容を、16進形式またはASCII形式で表示します。<i>mtype</i> が 0 の場合、すべてのメッセージが表示されます。<i>mtype</i> が負の値の場合、タイプが <i>mtype</i> の絶対値以下のすべてのメッセージが表示されます(msgrcv(2) および msgsnap(2) を参照してください)。 -i 共有メモリーセグメントへのISM接続の数を出力します。 -J 作成者のプロジェクトを出力します。 -o 未処理の操作に関する情報(キューにあるメッセージ数、メッセージキューのキューにあるメッセージの合計バイト数、共有メモリーセグメントに接続されているプロセス数)を出力します。

- p プロセス番号情報(メッセージを最後に送信したプロセスのプロセス ID、メッセージキューのメッセージを最後に受信したプロセスのプロセス ID、作成プロセスのプロセス ID、共有メモリーセグメントに対し最後に接続または切り離しを行なったプロセスのプロセス ID)を出力します。下記を参照。
- t 時間情報(すべての機能のアクセス権を変更した最後の制御操作の時間、メッセージキューでの最後の `msgsnd(2)` と最後の `msgrcv(2)` の時間、共有メモリーでの最後の `shmat(2)` と最後の `shmdt(2)` の時間(`shmop(2)`を参照)、セマフォでの最後の `semop(2)` の時間)を出力します。下記を参照。
- z zone 指定されたゾーン(`zones(5)`を参照)に関連付けられている機能に関する情報を出力します。ゾーンは、名前または数値 ID で指定できます。デフォルトでは、コマンドが実行されているゾーンに関する情報が表示されます。このオプションは、大域ゾーンで実行している場合にのみ効果を発揮します。
- Z 大域ゾーンで実行している場合に、すべてのゾーンに関する情報を出力します。それ以外の場合は、コマンドが実行されているゾーンに関する情報が出力されます。出力には、各機能に関連付けられているゾーンが含まれます。

`ipcs` の一覧における列見出しおよび列の意味については、次を参照してください。括弧内の文字は、対応する見出しを表示させるオプションを示しており、「all」は、見出しが常に表示されることを意味しています。注:これらのオプションは、各機能のどの情報が表示されるかを決定するだけで、どの機能が一覧表示されるかは決定しません。

T (all)	機能のタイプ。
q	メッセージキュー
m	共有メモリーセグメント
s	セマフォ
ID (all)	機能エントリの識別子。
KEY (all)	機能エントリを作成するために、 <code>msgget(2)</code> 、 <code>semget(2)</code> 、または <code>shmget(2)</code> に対する引数として使用されるキー(注:セグメントが削除された場合、セグメントに接続されていたすべてのプロセスがセグメントを切り離すまで、共有メモリーセグメントのキーは <code>IPC_PRIVATE</code> に変更されます)。
MODE (all)	機能のアクセスモードとフラグ。モードは、次のように解釈される 11 文字で構成されます。最初の 2 文字は次のように解釈されます。

- R プロセスは `msgrcv(2)` で待機中です。
- S プロセスは `msgsnd(2)` で待機中です。
- 対応する特別なフラグが設定されていません。

残りの9文字は、3文字ずつの3組に分けられます。最初の組は、所有者のアクセス権を示します。次の組は、機能エントリのユーザーグループに属するほかのユーザーのアクセス権を示します。最後の組は、その他のすべてのユーザーのアクセス権を示します。各組内の1つ目の文字は機能エントリの読み取り権を、2つ目の文字は書き込み権または変更権を示します。最後の文字は現在は使用されていません。

各アクセス権の意味は、次のとおりです。

- r 読み取り権が付与されています。
- w 書き込み権が付与されています。
- a すべてのアクセス権が付与されています。
- 指定されたアクセス権は与えられていません。

OWNER (all)	機能エントリの所有者のログイン名。
GROUP (all)	機能エントリの所有者のグループのグループ名。
CREATOR (a, A, c)	機能エントリの作成者のログイン名。
CGROUP (a, A, c)	機能エントリの作成者のグループのグループ名。
CBYTES (a, A, o)	関連付けられているメッセージキューにある現時点で未処理のメッセージのバイト数。
QNUM (a, A, o)	関連付けられているメッセージキューにある現時点で未処理のメッセージの数。
QBYTES (a, A, b)	関連付けられているメッセージキューにある未処理のメッセージで許可されている最大バイト数。
LSPID (a, A, p)	関連付けられているキューにメッセージを最後に送信したプロセスのプロセス ID。
LRPID (a, A, p)	関連付けられているキューからメッセージを最後に受信したプロセスのプロセス ID。
STIME (a, A, t)	関連付けられているキューにメッセージが最後に送信された時間。
RTIME (a, A, t)	関連付けられているキューからメッセージを最後に受信した時間。

CTIME (a, A, t)	関連付けられているエントリが作成または変更された時間。
ISMATCH (a, i)	関連付けられている共有メモリーセグメントへの ISM 接続の数。
NATCH (a, A, o)	関連付けられている共有メモリーセグメントに接続されているプロセスの数。
SEGSZ (a, A, b)	関連付けられている共有メモリーセグメントのサイズ。
CPID (a, A, p)	共有メモリーエントリの作成者のプロセス ID。
LPID (a, A, p)	共有メモリーセグメントを最後に接続または切り離したプロセスのプロセス ID。
ATIME (a, A, t)	共有メモリーセグメントへの接続が最後に完了した時間。
DTIME (a, A, t)	共有メモリーセグメントの切り離しが最後に完了した時間。
NSEMS (a, A, b)	セマフォエントリに関連付けられているセット内のセマフォの数。
OTIME (a, A, t)	セマフォエントリに関連付けられているセットでセマフォ操作が最後に完了した時間。
PROJECT (J, A)	機能エントリの作成者のプロジェクト名。
ZONE (Z)	機能が関連付けられているゾーン。

環境変数 `ipcs` の実行に影響を与える次の環境変数については、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLS_PATH。

TZ `ipcs` によって書き込まれる時間文字列のタイムゾーンを決定します。

ファイル
`/etc/group` グループ名
`/etc/passwd` ユーザー名

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目 [ipcrm\(1\)](#), [msgget\(2\)](#), [msgids\(2\)](#), [msgrcv\(2\)](#), [msgsnap\(2\)](#), [msgsnd\(2\)](#), [semget\(2\)](#), [semids\(2\)](#), [semop\(2\)](#), [shmctl\(2\)](#), [shmget\(2\)](#), [shmid\(2\)](#), [shmop\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#), [zones\(5\)](#)

注意事項

ipcs の実行中に状況が変化することがあります。提供される情報の正確性は、その情報が取得された時点にかぎり保証されます。

名前	isainfo – 命令セットアーキテクチャーの記述
形式	isainfo [[-v] [-b -n -k] [-x]]
機能説明	isainfo ユーティリティーは、現在実行中のシステムでサポートされる命令セットアーキテクチャーのさまざまな属性を識別するために使用されます。このコマンドが回答可能な質問には、64 ビットアプリケーションがサポートされているかどうか、実行中のカーネルで使用するデバイスドライバが 32 ビットか 64 ビットか、などが含まれます。 オプションを設定せずに isainfo を呼び出した場合、現バージョンのオペレーティングシステムでサポートされているアプリケーションのネイティブ命令セットの名前が表示されます。それらは、isalist(1) によって返されるリストのサブセットです。サブセットは、現在実行中のシステムでサポートされている基本アプリケーション環境に対応しています。
オプション	サポートしているオプションは、次のとおりです。 -b ネイティブ命令セットのアドレス空間のビット数を表示します。 -k オペレーティングシステムのカーネルコンポーネント (デバイスドライバや STREAMS モジュールなど) で使用される命令セットの名前を表示します。 -n オペレーティングシステムの現在のバージョンでサポートされる、移植性のあるアプリケーションで使用されるネイティブ命令セットの名前を表示します。 -v -b、-k、または -n オプションともに使用した場合、より詳細な情報を表示します。 -x プラットフォームでサポートされている、ネイティブ ABI に対する命令拡張を表示します。
使用例	例1 32 ビット x86 プラットフォーム上での isainfo の呼び出し 次の例では、32 ビット x86 プラットフォーム上で isainfo を呼び出します。 <pre>example% isainfo -v 32-bit i386 applications</pre> example% isainfo -k i386 例2 64 ビット SPARC プロセッサ上で 64 ビットオペレーティングシステムを実行しているシステムでの isainfo の呼び出し 次の例では、64 ビット SPARC プロセッサ上で 64 ビットオペレーティングシステムを実行しているシステムで isainfo を呼び出します。 <pre>example% isainfo sparcv9 sparc</pre>

例2 64ビット SPARC プロセッサ上で64ビットオペレーティングシステムを実行しているシステムでの isainfo の呼び出し (続き)

```
example% isainfo -n
sparcv9
example% isainfo -v
64-bit sparcv9 applications
32-bit sparc applications
example% isainfo -vk
64-bit sparcv9 kernel modules
```

例3 AMD Opteron CPU 上での isainfo -x の呼び出し

次の例では、AMD Opteron CPU 上で -x オプションを指定して isainfo を呼び出します。

```
example% isainfo -x
i386: fpu tsc cx8 sep cmov mmx ammx a3dnow a3dnowx fxsr sse sse2 pause
```

終了ステータス 0 以外 オプションが正しく指定されていないか、実行中のシステムの属性をコマンドが認識できません。エラーメッセージが標準エラー出力に出力されます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [isalist\(1\)](#), [uname\(1\)](#), [psrinfo\(1M\)](#), [getisax\(2\)](#), [sysinfo\(2\)](#), [attributes\(5\)](#), [isalist\(5\)](#)

名前	isalist – このプラットフォームで実行可能なネイティブ命令セットを表示する
形式	isalist
機能説明	isalist は、sysinfo(2) の SI_ISALIST コマンドが返すように、このプラットフォームで実行可能なネイティブ命令セットの名前を標準出力に書き込みます。 名前はスペースで区切られ、最適なパフォーマンスの順に表示されます。つまり、先に表示された命令セットには、あとに表示された命令セット以上の命令が含まれている場合があります。このため、このマシンではほとんどの場合、先に表示された命令セット用にコンパイルされたプログラムは、あとに表示された命令セット用にコンパイルされた同じプログラムよりも高速に動作します。 このマシンではほとんどの場合、このリストに表示されない命令セット用にコンパイルされたプログラムは、パフォーマンスの低下を引き起こすか、まったく動作しません。 システムが認識する命令セットの名前は、isalist(5) のリストに表示されます。これらの名前は、C 言語処理系の定義済みの名前やコンパイラオプションに一致する場合も一致しない場合もあります。 このコマンドは廃止または、互換性がなくなる可能性があり、Solaris の将来のバージョンでは削除されることがあります。命令セット拡張機能のより良い処理方法については、isainfo(1) を参照してください。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [isainfo\(1\)](#), [optisa\(1\)](#), [uname\(1\)](#), [sysinfo\(2\)](#), [attributes\(5\)](#), [isalist\(5\)](#)

名前 jobs, fg, bg, stop, notify – プロセスの実行の制御

形式

```
sh      jobs [-p | -l] [% job_id...]
        jobs -x command [arguments]
        fg [% job_id...]
        bg [% job_id...]
        stop % job_id...
        stop pid...

csh     jobs [-l]
        fg [% job_id]
        bg [% job_id]...
        notify [% job_id]...
        stop % job_id...
        stop pid...

ksh88   jobs [-lnp] [% job_id...]
        fg [% job_id...]
        bg [% job_id...]
        stop % job_id...
        stop pid...

ksh     jobs [-lnp] [job_id...]
        fg [job_id...]
        bg [job_id...]
```

機能説明

sh ジョブ制御が有効なとき、Bourne シェルに組み込まれた jobs は、停止中またはバックグラウンドで実行中のすべてのジョブを表示します。%*job_id* を省略すると、停止中またはバックグラウンドで稼働中のすべてのジョブが表示されます。次のオプションは、jobs の出力を変更または拡張するために使用します。

- l ジョブのプロセスグループ ID および作業ディレクトリを表示します。
- p ジョブのプロセスグループ ID のみを表示します。
- x *command* または *arguments* 中に見つかった *job_id* を、対応するプロセスグループ ID に置き換え、*command* に *argument* を渡して実行します。

シェルを `jsh` として呼び出すと、`sh` の説明で述べたすべての機能に加えて、ジョブ制御が可能になります。通常、ジョブ制御は対話型シェルに対してだけ可能です。非対話型シェルは、ジョブ制御の機能が追加されても、その恩恵を受けないのが通常です。

ジョブ制御が可能なとき、ユーザーが端末から入力したコマンドまたはパイプラインは、すべて `job_id` と呼ばれます。どのジョブも、必ず、次のいずれかの状態にあります。フォアグラウンド、バックグラウンド、または停止です。これらの用語の定義を次に示します。

1. フォアグラウンド状態にあるジョブは、制御している端末への読み取りおよび書き込み権を持っています。
2. バックグラウンド状態にあるジョブは、制御している端末への読み取りアクセスを拒否されていますが、条件付き書き込みアクセス権を持っています (`stty(1)` を参照)。
3. 停止ジョブは保留状態に置かれたジョブであり、通常は `SIGTSTP` シグナルにより、この状態になります (`signal.h(3HEAD)` を参照)。

シェルが起動するすべてのジョブには、ジョブ番号 (*job-id number*) と呼ばれる正の整数が割り当てられます。シェルはこの番号を把握し、特定のジョブを示す識別子として使用します。また、シェルは現在 (*current*) および直前 (*previous*) のジョブも把握しています。現在のジョブとは、最後に起動または再起動されたジョブです。前回のジョブとは、その直前のジョブです。

ジョブ識別子の正しい構文は次のような形式です。

`%job_id`

`job_id` は、次のいずれかの形式で指定できます。

<code>%</code> または <code>+</code>	現在のジョブ
<code>-</code>	前回のジョブ
<code>?<string></code>	<code>string</code> を含むコマンド行 (一意に表す) に対応したジョブ
<code>n</code>	ジョブ番号が <code>n</code> のジョブ
<code>pref</code>	コマンド名の先頭が <code>pref</code> のコマンド。たとえば <code>ls l name</code> がバックグラウンドで実行中だった場合、 <code>%ls</code> と指定すればこのコマンドを示すことができます。 <code>pref</code> は、引用符で囲まれないかぎり、空白文字を含めることができません。

ジョブ制御が有効なとき、`fg` は中断しているジョブの実行をフォアグラウンドで再開します。またバックグラウンドで実行中のジョブをフォアグラウンドに移動します。`%job_id` を省略した場合は、現在のジョブとみなされます。

ジョブ制御が有効なとき、bg は中断されているジョブの実行をバックグラウンドで再開します。%job_id を省略した場合は、現在のジョブとみなされます。

stop は、job_id を指定してバックグラウンドジョブの実行を中断、または pid (プロセス ID 番号) を指定してすべてのプロセスを中断します (ps(1) を参照)。

csh

C シェルに組み込まれた jobs は、引数なしでジョブ制御下で活動中のジョブを一覧表示します。

-l 通常の情報の他に、プロセス ID を表示します。

シェルは、番号の付いた job_id を各コマンドシーケンスと対応付けて、バックグラウンドで動作中のコマンド、または TSTP シグナル (通常は Control-Z) によって停止したコマンドの動作を追跡します。コマンドまたはコマンドシーケンス (セミコロンの区切られたリスト) をメタキャラクタ & を使用してバックグラウンドで起動した場合、シェルは角括弧で囲まれたジョブ番号と関連するプロセス番号のリストを表示します。次に例を示します。

[1] 1234

現在のジョブリストを見るには、組み込みコマンド jobs を使用します。最後に停止したジョブ (停止したジョブがない場合は、最後にバックグラウンドに投入されたジョブ) を「現在のジョブ」といい、+ で示します。前のジョブは - で示します。現在のジョブが終了したりフォアグラウンドに移された場合、前のジョブが新しく現在のジョブになります。

ジョブを操作するには、組み込みコマンド bg、fg、kill、stop、および % を使用します

ジョブの参照は % で始まります。パーセント記号だけの指定は、現在のジョブを示します。

%%+ %% 現在のジョブ。

%- 前のジョブ。

%j 次のようにしてジョブ j を参照します。'kill -9 %j'。ここで、j はジョブ番号、またはジョブを起動した コマンド行を一意に表す文字列です。たとえば 'fg %vi' は、停止した vi ジョブをフォアグラウンドに移します。

%?string コマンド行が string を含んでいるジョブ

バックグラウンドで動作中のジョブは、端末からの読み取り時に停止します。バックグラウンドジョブは、通常出力を生成しますが、'stty tostop' コマンドを使用して抑制することも可能です。

fg は現在のジョブまたは指定された job_id をフォアグラウンドへ移します。

bg はバックグラウンドで、現在のジョブまたは指定されたジョブを実行します。

stop は、*job_id* を指定してバックグラウンドジョブの実行を中断、または *pid* (プロセス ID 番号) を指定してすべてのプロセスを中断します (**ps(1)** を参照)。

notify は、現在のジョブまたは指定されたジョブのステータスが変化したとき、非同期的にユーザーに知らせます。

ksh88

jobs は、現在のシェル環境で開始されたジョブの状況を表示します。 **jobs** がジョブの終了ステータスを報告したとき、シェルはそのジョブのプロセス ID を、現在のシェル実行環境で把握しているプロセス ID のリストから削除します。

特定のジョブの報告だけが必要なら、*job_id* を使ってジョブを指定します。 *job_id* を 1 つも指定しないと、全ジョブに関する情報が出力されます。

次のオプションは、**jobs** の出力を変更または拡張するために使用します。

- l (小文字のエル) 個々のジョブに関して詳細な情報を出力します。具体的には、ジョブ番号、現在のジョブ、プロセスグループ ID、状態、ジョブを生成したコマンドを出力します。
- n 前回通知を受けたあとに停止または終了したジョブだけを表示します。
- p 選択されたジョブのプロセスグループリーダーのプロセスグループ ID だけを出力します。

デフォルトでは、**jobs** は、停止しているすべてのジョブのステータス、実行中のバックグラウンドジョブのステータス、そしてステータスが変化したのにシェルによりまだ報告されていないすべてのジョブのステータスを表示します。

set コマンドの **monitor** オプションを有効にすると、対話型シェルが **job** を各パイプラインと関連付けます。このオプションは、**jobs** コマンドが表示する現在のジョブのテーブルを維持し、これらのジョブに整数番号を割り当てます。ジョブを **&** で非同期的に起動すると、シェルは、次の形式の行を表示します。

```
[1] 1234
```

非同期的に起動されたジョブはジョブ番号 1 であり、プロセス ID が 1234 であるプロセスが 1 つ (トップレベル) あることを示します。

実行中のジョブがあるが、別の実行対象のジョブがある場合、**^Z** (Control-Z) キーを押せば、現在のジョブに **STOP** シグナルが送信されます。そうするとシェルは通常、ジョブが「停止」されたことを示し (後述の「出力」の項を参照)、プロンプトを表示します。これで、このジョブの状態を **bg** コマンドでバックグラウンドで処理するか、またはほかのコマンドを実行してから、**fg** というコマンドでジョブ

をフォアグラウンドに移すことができます。^Zはただちに有効になります。つまり、保留中の出力や読み取られていない入力がただちに中止されるという点で、割り込みに似ています。

シェル内のジョブを参照する方法はいくつかあります。そのジョブのいずれかのプロセスのIDを使っても、また次のいずれかを使っても参照できます。

<code>%number</code>	指定された番号のジョブ。
<code>%string</code>	コマンド行が <i>string</i> で始まっていたジョブ。履歴ファイルが有効なとき、この方法是对話モードでのみ使用できます。
<code>;%?string</code>	コマンド行が <i>string</i> を含んでいたジョブ。履歴ファイルが有効なとき、この方法是对話モードでのみ使用できます。
<code>%%</code>	現在のジョブ
<code>%+</code>	<code>%%</code> と同等
<code>%-</code>	直前のジョブ

シェルは、プロセスの状態が変更されると、ただちにそれを検出します。ジョブがブロックされてそれ以上進めない状態になると、プロンプトを発行する直前にだけ、シェルはその旨をユーザーに通知します。これは、ユーザーの作業の妨げにならないようにするためです。モニタモードが有効なとき、完了した各バックグラウンドジョブは、CHLD に設定されているトラップを起こします。ジョブの実行中または停止中にシェルを終了しようとする、「停止中(実行中)のジョブがある(You have stopped (running) jobs.)」旨の警告を受けます。jobs コマンドを使用すれば、どのジョブが該当するのかを確認できます。これを実行するか、またはただちにシェルを再終了しようとする、シェルは2度目の警告は出さず、停止中のジョブは終了します。

fg は、バックグラウンドジョブを、現在の環境からフォアグラウンドへ移します。fg を使ってジョブをフォアグラウンドへ移した場合、そのジョブのプロセスIDは、現在のシェル実行環境で把握しているプロセスIDのリストから削除されます。fg コマンドが使えるのは、ジョブ制御をサポートするシステム上だけです。job_id が指定されないと、現在のジョブをフォアグラウンドで実行します。

bg は、現在の環境で中断されたジョブを、バックグラウンドジョブとして実行することにより再開します。job_id が示すジョブがすでにバックグラウンドジョブを実行している場合、bg は何も行わず正常に終了します。bg を使ってジョブをバックグラウンドへ移した場合、あたかも非同期リストから起動されたかのように、そのジョブのプロセスIDは、「現在のシェル実行環境で把握しているプロセスID」の1つとなります。bg コマンドが使えるのは、ジョブ制御をサポートするシステム上だけです。job_id が省略された場合は、現在のジョブをバックグラウンドで実行します。

`stop` は、`job_id` を指定してバックグラウンドジョブの実行を中断、または `pid` (プロセス ID 番号) を指定してすべてのプロセスを中断します [ps\(1\)](#) を参照してください。

ksh

`jobs` は、現在のシェル環境によって開始された、指定されたジョブに関する情報を標準出力で表示します。情報には、[...] で囲まれたジョブ番号、ステータス、およびジョブを開始したコマンド行が含まれます。

`job_id` を省略した場合、`jobs` は、停止されたすべてのジョブ、バックグラウンドジョブ、およびシェルによる最後の報告以降にステータスを変更されたすべてのジョブのステータスを表示します。

`jobs` がジョブの終了ステータスを報告したとき、シェルはそのジョブを、現在のシェル環境で把握しているジョブのリストから削除します。

次のオプションは、`jobs` の出力を変更または拡張するために使用します。

- l 通常の情報に加えて、プロセス ID をジョブ番号のあとに表示します。
- n 最後のプロンプトが表示されてからステータスを変更されたジョブのみを表示します。
- p 指定されたジョブのプロセスグループリーダーの ID を表示します。

`job_id` は、次のいずれかの形式で `jobs`、`fg`、および `bg` に対し指定できます。

- `number` job のプロセス ID。
- `-number` job のプロセスグループ ID。
- `%number` ジョブ番号。
- `%string` 名前が `string` で始まるジョブ。
- `%?string` 名前に `string` が含まれるジョブ。
- `%+`
- `%%` 現在のジョブ。
- `%-` 前のジョブ。

`fg` は、指定されたジョブをフォアグラウンドに順に移行し、`CONT` 信号を送信してそれぞれの実行を開始します。`job_id` を省略した場合、最後に開始または停止されたバックグラウンドジョブがフォアグラウンドに移行されます。

`bg` は、指定されたジョブをバックグラウンドに移行し、`CONT` 信号を送信してそれらの実行を開始します。`job_id` を省略した場合、最後に開始または停止されたバックグラウンドジョブがバックグラウンドで再開または続行されます。

出力 `-p` オプションを指定すると、各プロセス ID に対して次に示す 1 行の情報が出力されます。

```
"%d\n", "process ID"
```

`p` を指定しない場合、`-l` オプションも省略されていれば、次の形式の一連の行が出力されます。

```
"[%d] %c %s %s\n", job-number, current, state, command
```

各フィールドの意味を次に説明します。

<i>current</i>	文字 <code>+</code> は、 <code>fg</code> および <code>bg</code> コマンド用のデフォルトとして使用するジョブを表します。このジョブは、 <code>job_id%</code> または <code>%</code> を使って指定することもできます。文字 <code>-</code> は、現在のデフォルトジョブが終了した場合にデフォルトとなるジョブを表します。このジョブは <code>job_id%</code> <code>-</code> を使って指定することもできます。その他のジョブに関しては、このフィールドは空白文字です。 <code>+</code> や <code>-</code> を使って表せるジョブの数は、どちらも最大 1 つです。停止中のジョブがあれば、現在のジョブも停止ジョブとなります。停止中のジョブが 2 つ以上あれば、以前のジョブも停止ジョブとなります。														
<i>job-number</i>	<code>wait</code> 、 <code>fg</code> 、 <code>bg</code> 、 <code>kill</code> の各ユーティリティー用にプロセスグループを識別するのに使用する番号。これらのユーティリティーを使うと、ジョブはジョブ番号の前に <code>%</code> を付加することにより識別できます。														
<i>state</i>	POSIX ロケールにある次の文字列のいずれかです。 <table><tr><td>Running</td><td>ジョブはシグナルによって中断されておらず、終了もしていないことを表します。</td></tr><tr><td>Done</td><td>ジョブは終了して、ゼロの終了ステータスを返したことを表します。</td></tr><tr><td>Done(<i>code</i>)</td><td>ジョブは正常に終了し、指定された <code>0</code> 以外の終了ステータス (<i>code</i> が示す 10 進数) を返したことを表します。</td></tr><tr><td>Stopped</td><td>ジョブが停止されたことを表します。</td></tr><tr><td>Stopped(SIGTSTP)</td><td>SIGTSTP シグナルがジョブを停止したことを表します。</td></tr><tr><td>Stopped(SIGSTOP)</td><td>SIGSTOP シグナルがジョブを停止したことを表します。</td></tr><tr><td>Stopped(SIGTTIN)</td><td>SIGTTIN シグナルがジョブを停止したことを表します。</td></tr></table>	Running	ジョブはシグナルによって中断されておらず、終了もしていないことを表します。	Done	ジョブは終了して、ゼロの終了ステータスを返したことを表します。	Done(<i>code</i>)	ジョブは正常に終了し、指定された <code>0</code> 以外の終了ステータス (<i>code</i> が示す 10 進数) を返したことを表します。	Stopped	ジョブが停止されたことを表します。	Stopped(SIGTSTP)	SIGTSTP シグナルがジョブを停止したことを表します。	Stopped(SIGSTOP)	SIGSTOP シグナルがジョブを停止したことを表します。	Stopped(SIGTTIN)	SIGTTIN シグナルがジョブを停止したことを表します。
Running	ジョブはシグナルによって中断されておらず、終了もしていないことを表します。														
Done	ジョブは終了して、ゼロの終了ステータスを返したことを表します。														
Done(<i>code</i>)	ジョブは正常に終了し、指定された <code>0</code> 以外の終了ステータス (<i>code</i> が示す 10 進数) を返したことを表します。														
Stopped	ジョブが停止されたことを表します。														
Stopped(SIGTSTP)	SIGTSTP シグナルがジョブを停止したことを表します。														
Stopped(SIGSTOP)	SIGSTOP シグナルがジョブを停止したことを表します。														
Stopped(SIGTTIN)	SIGTTIN シグナルがジョブを停止したことを表します。														

Stopped (SIGTTOU) SIGTTOU シグナルがジョブを停止したことを表します。

利用者側で、文字列 **Stopped** の代わりに **Suspended** を使うよう定義できます。ジョブをシグナルが終了した場合、**state** の形式は不特定ですが、ここに示したほかの **state** 形式とは明確に区別できるものです。その出力上で、ジョブを終了させたシグナルの名前または説明が与えられます。

command シェルに与えられた関連コマンド。

-l オプションを指定すると、プロセスグループ ID を示すフィールドが **state** フィールドの前に挿入されます。さらに、プロセスグループ内のより多くのプロセスが別の行に出力されることがあります。その内容は、プロセス ID と **command** フィールドだけです。

環境変数 **jobs**、**fg**、**bg** の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

sh、**csch**、**ksh88** **jobs**、**fg**、**bg** は、次の終了値を返します。

- 0 正常終了。
- >0 エラーが発生した。

ksh **jobs** は、次の終了値を返します。

- 0 各ジョブの情報が標準出力に書き込まれました。
- >0 1 つ以上のジョブが存在しません。

fg は、次の終了値を返します。

最後のジョブの終了ステータス 1 つ以上のジョブがフォアグラウンドに移行されました。

- 0 以外 1 つ以上のジョブが存在しないか、完了していません。

bg は、次の終了値を返します。

- 0 すべてのバックグラウンドジョブが開始されました。
- >0 1 つ以上のジョブが存在しないか、バックグラウンドジョブが存在しません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

csh、sh、ksh88

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	确实
標準	standards(5) を参照してください。

ksh

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不确实

関連項目

[csh\(1\)](#), [kill\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [ps\(1\)](#), [sh\(1\)](#), [stop\(1\)](#), [shell_builtins\(1\)](#), [stty\(1\)](#), [wait\(1\)](#), [signal.h\(3HEAD\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	join - リレーショナルデータベース演算子
形式	<pre> /usr/bin/join [-a filename] [-v filename] [-1 fieldnumber] [-2 fieldnumber] [-o list] [-e string][-t char] file1 file2 /usr/bin/join [-a filename] [-j fieldnumber] [-j1 fieldnumber] [-j2 fieldnumber] [-o list] [-e string][-t char] file1 file2 </pre>
機能説明	join は、ファイル <i>file1</i> とファイル <i>file2</i> で等結合を実行し、結果として生成される結合されたファイルを標準出力に書き込みます。デフォルトでは、フィールドは1つ以上のスペースまたはタブで区切られます。連続するスペースまたはタブ、あるいはその両方のうち、最後の1つ以外は無視されます。-t オプションを使用すると、フィールドの区切り文字を変更できます。 join フィールドとは、各ファイル内のフィールドであり、このフィールドでファイルが比較されます。デフォルトでは、join により、一致する join フィールドがある <i>files1</i> と <i>files2</i> 内の行のペアごとに1行が出力に書き込まれます。デフォルトの出力行は、join フィールド、<i>file1</i> の残りのフィールド、<i>file2</i> の残りのフィールドで構成されますが、これは、-o オプションを使用すると変更できます。また -a オプションを使えば、一致しなかった行も出力できます。-v オプションは、一致しなかった行だけを出力するために使用します。 ファイル <i>file1</i> とファイル <i>file2</i> は、結合されるフィールド上でソート -b の照合順序でソートされている必要があります。ソートされていない場合、結果は不確定です。 <i>file1</i> または <i>file2</i> が - の場合、join は標準入力を使用します。開始位置は現在の位置です。
オプション	次のオプションには <i>filename</i> という引数を使用するものがあります。この引数の値は1または2で、それぞれ <i>file1</i> または <i>file2</i> を表します。 -a <i>filename</i> 通常の出力に加え、ファイル <i>filename</i> 内の対にならなかった行についても出力します。<i>filename</i> の値は1または2です。-a 1 と -a 2 の両方を指定すると、対にならなかった行がすべて出力されます。 -e <i>string</i> -o オプションで選択したリスト内にある空の出力フィールドを <i>string</i> 文字列で置き換えます。 -j <i>fieldnumber</i> -1 <i>fieldnumber</i> -2 <i>fieldnumber</i> と同等です。フィールドには1から始まる番号が付きます。 -j1 <i>fieldnumber</i> -1 <i>fieldnumber</i> と同等です。フィールドには1から始まる番号が付きます。 -j2 <i>fieldnumber</i> -2 <i>fieldnumber</i> と同じ意味です。フィールドには1から始まる番号が付きます。

- o *list* 各出力行は、*list* に指定されたフィールドを含みます。*list* で選択されたフィールドのうち入力中に現れないものは、空の出力フィールドとして扱われます (-e オプションの説明を参照)。この各要素の形式は *filenumber.fieldnumber* または *0* で、後者は join フィールドを表します。特に指定しないかぎり、共通フィールドは出力しません。
- t *char* *char* で示す文字を区切り文字として使用します。1つの行に *char* が複数個あるとき、それらはすべて有効です。文字 *char* は、入力と出力の両方においてフィールド区切り文字として使われます。このオプションを指定する場合、照合される語は -b オプションなしで sort を実行した場合にも同一とみなされる語でなければなりません。
- v *filenumber* デフォルト形式の出力ではなく、*filenumber* 内の対にならなかった行だけを出力します。*filenumber* の値は 1 または 2 です。-v 1 と -v 2 の両方を指定すると、対にならなかった行がすべて出力されます。
- 1 *fieldnumber* ファイル 1 の *fieldnumber* 番目のフィールドで結合します。フィールド番号は 1 から始まる 10 進整数です。
- 2 *fieldnumber* ファイル 2 の *fieldnumber* 番目のフィールドで結合します。フィールド番号は 1 から始まる 10 進整数です。

オペランド

次のオペランドがサポートされています。

- file1* 結合するファイルのパス名。*file1* と *file2* のどちらかに - を指定すると、そのファイルの代わりに標準入力を使用されます。
- file2* 結合するファイルのパス名。*file1* と *file2* のどちらかに - を指定すると、そのファイルの代わりに標準入力を使用されます。

file1 と *file2* は、結合するフィールド (通常は各行の最初のフィールド) 上で LC_COLLATE によって決められた照合順序で昇順にソートされていなければなりません (sort(1) を参照)。

使用法

ファイルが 2G バイト (2³¹ バイト) 以上ある場合の join の動作については、largefile(5) を参照してください。

使用例

例1 パスワードファイルとグループファイルの結合

次のコマンド行は、パスワードファイルとグループファイルを結合し、数値グループ ID のマッチングを行い、ログイン名、グループ名、およびログインディレクトリを出力します。ファイルは、グループ ID フィールド上において ASCII の照合順序でソートされていると仮定します。

```
example% join -j1 4-j2 3 -o 1.1 2.1 1.6 -t:/etc/passwd /etc/group
```

例2 -o オプションの使用

-o 0 は、基本的には結合フィールドを統合したフィールドを表します。たとえば、phone という名のファイルは次のような内容だとします。

```
!Name      Phone Number
Don        +1 123-456-7890
Hal        +1 234-567-8901
Yasushi    +2 345-678-9012
```

また、fax というファイルは次のような内容だとします。

```
!Name      Fax Number
Don        +1 123-456-7899
Keith      +1 456-789-0122
Yasushi    +2 345-678-9011
```

この2つのファイルで、長いスペース部分は1つのタブ文字を表しているものとします。このとき次のようなコマンドを実行します。

```
example% join -t"tab" -a 1 -a 2 -e '(unknown)' -o 0,1,2,2.2 phone fax
```

すると次のような出力が生成されます。

```
!Name      Phone Number      Fax Number
Don        +1 123-456-7890          +1 123-456-7899
Hal        +1 234-567-8901          (unknown
Keith      (unknown)              +1 456-789-012
Yasushi    +2 345-678-9012          +2 345-678-9011
```

環境変数 join の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、LC_COLLATE、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 入力ファイルはすべて正常に出力されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

属性タイプ	属性値
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目
- [awk\(1\)](#), [comm\(1\)](#), [sort\(1\)](#), [uniq\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)
- 注意事項
- デフォルトでフィールドを分離する場合、照合順序は `-b` を付けた `sort` コマンドに対応した順序です。 `-t` の場合、照合順序は `b` を付けない `sort` コマンドの順序になります。

`join`、`sort`、`comm`、`uniq`、`awk` コマンドの規則は一律ではありません。

名前	kbd - キーボード状態の操作、キーボードの種類の表示、またはデフォルトのキーボードアポートシーケンス効果の変更
形式	<pre> kbd [-r] [-t] [-l] [-a enable disable alternate] [-c on off] [-d <i>keyboard device</i>] [-D <i>autorepeat delay</i>] [-R <i>autorepeat rate</i>] kbd [-i] [-d <i>keyboard device</i>] kbd -s [<i>language</i>] kbd -b [<i>keyboard console</i>] <i>frequency</i> </pre>
機能説明	kbd ユーティリティーは、キーボード状態の操作、キーボードの種類の表示、またはデフォルトのキーボードアポートシーケンス効果の変更を行います。アポートシーケンスは、シリアルコンソールデバイスにも適用されます。kbd ユーティリティーは、<code>/dev/kbd</code> デフォルトキーボードデバイスを設定します。
拡張機能説明	<code>-i</code> オプションは、keyclick および keyboard アポート設定のデフォルト値をキーボード構成サービス <code>svc:/system/keymap:default</code> から読み取って処理します。クリックをサポートするキーボードだけが、<code>-c</code> オプションに応答します。クリックをデフォルトで有効にするには、次のようにして <code>keymap</code> サービスの <code>keymap/keyclick</code> プロパティの値を追加または変更します。 <pre> \$ svccfg -s keymap:default setprop keymap/keyclick=true \$ svcadm refresh keymap </pre> 次に、コマンド <code>kbd -i</code> を実行して設定を変更します。<code>keymap/keyclick</code> プロパティの有効な設定は、<code>true</code> または <code>false</code> です。その他の値はすべて無視されます。<code>keymap</code> サービス内で <code>keymap/keyclick</code> プロパティが指定されていない場合、この設定は変更されません。 キーボードのアポートシーケンス効果を変更できるのは、スーパーユーザーが <code>-a</code> オプションを使用した場合だけです。通常このシーケンスは、キーボードの Stop-A、L1-A、および Shift-Pause (SPARC システム)、F1-A および Shift-Pause (x86 システム)、BREAK (大半のシステムのシリアルコンソール入力デバイス) です。 電気信号エラーに由来する BREAK 状態と、リモートの DCE により意図的に送信されたものとを区別することはできません。対処方法として、代替ブレークで <code>-a</code> オプションを使用してブレークの解釈を切り替えます。シーケンスが不正確に解釈されることを避けるため、代替ブレークシーケンスが有効な場合は、SLIP などのバイナリプロトコルをシリアルコンソールポート経由では実行しないようにしてください。 PPP はバイナリプロトコルではありますが、シリアル操作の妨げになる文字の使用を回避する機能を備えています。デフォルトの代替ブレークシーケンスは CTRL-m から CTRL-b、または 16 進数の 0D 7E 02 です。PPP の場合、ACCM で 0x00000004 または 0x00002000 を設定することによりこれを回避できます。これにより、CTRL-b または CTRL-m 文字のエスケープがそれぞれ強制されます。

Solaris PPP 4.0 でこれを実行するには、

```
asynmap 0x00002000
```

を `/etc/ppp/options` ファイルまたは接続で使用するほかの構成ファイルのいずれかに追加します。[pppd\(1M\)](#) を参照してください。

SLIP には同等の機能はないため、代替ブレイクシーケンスの使用中には決して使用しないでください。

代替ブレイクシーケンスは、キーボードアポートには影響を及ぼしません。代替ブレイクシーケンスの詳細については、[zs\(7D\)](#)、[se\(7D\)](#)、および [asy\(7D\)](#) を参照してください。

多くのシステムで、キーボードアポートシーケンスのデフォルト効果は、オペレーティングシステムを中断し、デバッグまたはモニターに入ることです。一部のシステムには、`secure` の位置を備えたキースイッチがあります。これらのシステムでは、キースイッチを `secure` の位置に設定すると、このコマンドを使用したソフトウェアのデフォルトセットがすべてオーバーライドされます。

キーボードアポートシーケンスのソフトウェアのデフォルト効果を恒久的に変更するには、最初に、次に示すように `keymap` サービスの `keymap/keyboard_abort` プロパティの値を追加または変更します。

```
$ svccfg -s keymap:default setprop keymap/keyboard_abort=disable
$ svcadm refresh keymap
```

次に、コマンド `kbd -i` を実行して設定を変更します。有効な設定は `enable`、`disable`、および `alternate` です。その他の値はすべて無視されます。`keymap` サービスに変数が指定されていない場合、設定は変更されません。

アポートシーケンスをハードウェア `BREAK` に設定するには、次に示すように、`keymap` サービスの `keymap/keyboard_abort` の値を設定します。

```
$ svccfg -s keymap:default setprop keymap/keyboard_abort=enable
$ svcadm refresh keymap
```

現在の設定を変更するには、コマンド `kbd -i` を実行します。アポートシーケンスを代替ブレイク文字シーケンスに設定するには、最初に `keymap` サービスの `keyboard_abort` プロパティの現在値を次のように設定します。

```
$ svccfg -s keymap:default setprop keymap/keyboard_abort=alternate
$ svcadm refresh keymap
```

次に、コマンド `kbd -i` を実行して設定を変更します。代替ブレイクシーケンスが有効な場合は、シリアルコンソールデバイスだけが影響を受けます。

デフォルトの自動リピート遅延を設定するには、`keymap` サービスの `repeat_delay` プロパティを目的の値に設定します(単位はミリ秒(ms))。入力ミスのためにキーボードが使用不可になることを避けるため、`KIOCRPTDELAY_MIN` (`/usr/include/sys/kbio.h` で定義) よりも小さい遅延値は `EINVAL` で拒否されます。

```
$ svccfg -s keymap:default setprop keymap/repeat_delay=500
$ svcadm refresh keymap
```

デフォルトの自動リピートレートを設定するには、`keymap` サービスの `repeat_rate` プロパティを目的の値に設定します (単位はミリ秒)。負およびゼロのリピートレートは、`EINVAL` で拒否されます。

```
$ svccfg -s keymap:default setprop keymap/repeat_rate=40
$ svcadm refresh keymap
```

delay および *rate* の現在の設定を変更するには、コマンド `kbd -i` を実行します。自動リピート遅延や自動リピートレートが有効な場合、コマンド行モードだけが影響を受けます。

デフォルトの言語を設定するには、`keymap` サービスの `keymap/layout` プロパティを目的の言語に設定します。カーネルでサポートされている言語は、`kbd -s` を実行して確認できます。その他の値は無視されます。次の例では、キーボードをスペイン語配列に設定します。

```
$ svccfg -s keymap:default setprop keymap/layout=Spanish
$ svcadm refresh keymap
```

次に、`kbd -i` を実行して設定を変更します。`Solaris` のリブート時に、スペイン語鍵テーブルがカーネルに読み込まれます。これらの配列は、`usb` および `ps/2` キーボードで有効です。

キーボードビープ音のデフォルトの周波数を設定するには、`keymap` サービスの `keymap/kbd_beeper_freq` プロパティを目的の値に設定します (単位はHZ)。これは、0 以上 32767 以下の値にしてください。それ以外の値は `EINVAL` で拒否されます。

```
$ svccfg -s keymap:default setprop keymap/kbd_beeper_freq=2000
$ svcadm refresh keymap
```

コンソールビープ音のデフォルトの周波数を設定するには、`keymap` サービスの `keymap/console_beeper_freq` プロパティを目的の値に設定します (単位はHZ)。これは、0 以上 32767 以下の値にしてください。それ以外の値は `EINVAL` で拒否されます。

```
$ svccfg -s keymap:default setprop keymap/console_beeper_freq=900
$ svcadm refresh keymap
```

キーボードビープ音およびコンソールビープ音の周波数の現在設定を変更するには、`kbd -i` を実行します。

オプション

サポートしているオプションは、次のとおりです。

-a enable | disable | alternate

キーボードアポートシーケンスの効果を有効または無効にするか、切り替えます。デフォルトでは、キーボードアポートシーケンスにより、大半のシステムのオペレーティングシステムが中断されます。通常このシーケンスは、キーボードの Stop-A、L1-A、Shift-Pause (SPARC システム)、F1-A、Shift-Pause (x86 システム)、および BREAK (シリアルコンソールデバイス) です。

このオプションを使って、デフォルトのキーボード動作を変更できます。-a オプションを使用できるのはスーパーユーザーだけです。

enable キーボードアポートシーケンスのデフォルト効果を有効にします (オペレーティングシステムを中断し、デバッグまたはモニターに入ります)。

disable デフォルトや代替の効果を無効にして、キーボードアポートシーケンスを無視します。

alternate コンソールで代替ブレイク文字シーケンスを受け取る
と、キーボードアポートシーケンスの代替効果を有効にします (オペレーティングシステムを中断し、デバッグまたはモニターに入ります)。代替ブレイクシーケンスは、ドライバ [zs\(7D\)](#)、[se\(7D\)](#)、[asy\(7D\)](#) により定義されます。シーケンスが不正確に解釈される危険があるため、この値の使用時にバイナリプロトコルをシリアルコンソールポート経由で実行することはできません。

-b keyboard | console

キーボードまたはコンソールのピープ音の周波数を設定します。

keyboard キーボードピープ音の周波数をオペランドに設定します (単位は HZ)。オペランドを参照してください。

	console	コンソールビープ音の周波数をオペランドに設定します (単位はHZ)。オペランドを参照してください。
-c on off		キーボードのクリック音を有効または無効にします。
	on	クリック音を有効にします。
	off	クリック音を無効にします。
-d keyboard device		設定するキーボードデバイスを指定します。デフォルトの設定は、 <code>/dev/kbd</code> です。
-D autorepeat delay		自動リピートの遅延を設定します (単位はミリ秒)。
-i		キーボードのプロパティを <code>keymap</code> サービスから取得して設定します。このオプションを <code>-d keyboard device</code> 以外のオプションとともに使用することはできません。 <code>-i</code> オプションを指定すると、キーボードコマンドは <code>keymap</code> サービスのキーボードプロパティから <code>keyclick</code> および <code>keyboard</code> のアボートデフォルト値を読み取って処理します。 <code>-i</code> オプションを使用できるのは、Device Security 権利プロファイルを持つユーザーまたは役割だけです。
-l		使用中のキーボードの配列コード、および使用中の自動リピート遅延と自動リピートレートを返します。
		このオプションを <code>-R</code> または <code>-D</code> オプションとともに使用すると、変更前の値が返されます。
-r		キーボードを電源投入時の状態にリセットします。
-R autorepeat rate		自動リピートレートを設定します (単位はミリ秒)。
-s [language]		キー配列をカーネル内に設定します。
		<i>language</i> が指定されている場合、配列は <i>language</i> に設定されます。 <i>language</i> が指定されていない場合、使用可能な配列のリストを表示し、ユーザーに <i>language</i> の指定を求めます。「オペランド」の項を参照してください。

次のオペランドがサポートされています。

frequency	カーネルに設定する周波数の値。この値の受信側の指定には、-b オプションを使用します。これは 0 から 32767 の間の値にしてください。それ以外の値を指定すると、EINVAL で拒否されます。

language	カーネルに設定する言語。言語が見つからない場合は、サポートされる言語を一覧表示して選択を求めます。これは、-s オプションにのみ当てはまります。

例1 キーボードの種類を表示する

```
example% kbd -t
Type 4 Sun keyboard
example%
```

次の例では、keymap サービスの設定に従ってキーボードのデフォルトを設定します。

```
example# kbd -i
example#
```

次の例では、キーボードの種類と配列コードを表示します。また、自動リピートの遅延およびレートの設定も表示します。

```
example% kbd -l
type=4
layout=43 (0x2b)
delay(ms)=500
rate(ms)=33
example%
```

次の例では、キーボードの自動リピート遅延を設定します。

```
example% kbd -D 300
example%
```

次の例では、キーボードの自動リピートレートを設定します。

例5 キーボードの自動リピートレートを設定する (続き)

```
example% kbd -R 50
example%
```

例6 キーボードの言語を選択および設定する
次の例では、指定された言語の一覧からキーボードの言語を選択および設定します。

```
example% kbd -s
1. Albanian
2. Belarusian
3. Belgian
4. Bulgarian
5. Croatian
6. Danish
7. Dutch
.....
16. Malta_UK
17. Malta_US
18. Norwegian
19. Portuguese
20. Russian
21. Serbia-And-Montenegro
22. Slove
```

To select the keyboard layout, enter a number [default n]:

```
example%
```

次の例では、指定されたキーボード言語を設定します。

```
example% kbd -s Dutch
example%
```

例7 キーボードビープ音の周波数を設定する
次の例では、キーボードビープ音の周波数を設定します。

```
example% kbd -b keyboard 1000
example%
```

ファイル

/dev/kbd キーボードデバイスファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[kmbd\(1\)](#), [loadkeys\(1\)](#), [svcs\(1\)](#), [inetd\(1M\)](#), [inetadm\(1M\)](#), [svcadm\(1M\)](#), [pppd\(1M\)](#), [keytables\(4\)](#), [attributes\(5\)](#), [smf\(5\)](#), [kb\(7M\)](#), [zs\(7D\)](#), [se\(7D\)](#), [asy\(7D\)](#), [virtualkm\(7D\)](#)

注意事項

一部のサーバーシステムのキースイッチには `secure` のキー位置があり、システムソフトウェアからこのキー位置を読み取ることが可能です。このキー位置は、キーボードアポートシーケンス効果の通常のデフォルトよりも優先され、デフォルトを変更してこの効果を無効にします。これらのシステムでキースイッチが `secure` の位置にある場合は、`kbd` ユーティリティーで設定可能なソフトウェアデフォルトで、キーボードアポートシーケンス効果を無効にすることはできません。

現在のところ、キーボードクリック設定の状態を判別する方法はありません。

`kdb` サービスは、サービス管理機能 `smf(5)` により、次のサービス識別子として管理されます。

```
svc:/system/keymap:default
```

有効化、無効化、または再起動要求など、このサービスに関する管理操作は、`svcadm(1M)` を使用して実行できます。このサービスを開始または再開する責任は `inetd(1M)` に委託されています。このサービスの構成を変更したり、構成情報を表示したりするには、`inetadm(1M)` を使用します。サービスステータスを照会するには、`svcs(1)` コマンドを使用します。

名前	kdestroy – Kerberos チケットの破棄
形式	/usr/bin/kdestroy [-q] [-c <i>cache_name</i>]
機能説明	kdestroy ユーティリティーは、ユーザーのアクティブな Kerberos 承認チケットを含む指定された資格キャッシュに 0 を書き込むことによって、それらのチケットを破棄します。資格キャッシュが指定されていない場合は、デフォルトの資格キャッシュが破棄されます。資格キャッシュが存在しない場合、kdestroy は、その影響に対するメッセージを表示します。 キャッシュを上書きしたあと、kdestroy はシステムからキャッシュを削除します。このユーティリティーは、操作の成功または失敗を示すメッセージを表示します。キャッシュを破棄できない場合、kdestroy は、端末のビーブ音を鳴らすことによってユーザーに警告します。 必要に応じて、ログアウトしたらチケットが自動的に破棄されるように kdestroy コマンドを .logout ファイル内に配置できます。
オプション	サポートしているオプションは、次のとおりです。 -c <i>cache_name</i> <i>cache_name</i> は、資格 (チケット) のキャッシュ名および場所として使用します。このオプションが使用されていない場合は、デフォルトのキャッシュ名および場所が使用されます。 -q 何も出力せずに実行されます。kdestroy がチケットの破棄に失敗した場合、端末のビーブ音は鳴りません。
環境変数	kdestroy は、次の環境変数を使用します。 KRB5CCNAME 資格 (チケット) のキャッシュの場所。構文および詳細については、krb5envvar(5) を参照してください。
ファイル	/tmp/krb5cc_ <i>uid</i> デフォルトの資格キャッシュ (<i>uid</i> は、ユーザーの 10 進数の UID です)。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
インタフェースの安定性	確実
コマンド引数	確実
コマンド出力	不確実

関連項目 [kinit\(1\)](#), [klist\(1\)](#), [attributes\(5\)](#), [kerberos\(5\)](#), [krb5envvar\(5\)](#)

使用上の留意点 指定された資格キャッシュ内のチケットのみが破棄されます。ルートインスタンスとパスワード変更のチケットを保持するために、個別のチケットキャッシュが使用されます。これらのファイルも、おそらく破棄すべきです。または、ユーザーのチケットをすべて1つの資格キャッシュ内に保持すべきです。

名前 keylogin – keyserv による秘密鍵の復号化および格納

形式 /usr/bin/keylogin [-r]

機能説明 keylogin コマンドはパスワードの入力を求め、それを使用してユーザーの秘密鍵を復号化します。この鍵は、/etc/publickey ファイル ([publickey\(4\)](#) を参照) またはユーザーのホームドメイン内の NIS マップ「publickey.byname」内にあります。各ソースとそれらの検索順序は、/etc/nsswitch.conf ファイルで指定されます。[nsswitch.conf\(4\)](#) を参照してください。復号化されると、ユーザーの秘密鍵は、ローカルのキーサーバープロセス [keyserv\(1M\)](#) によって格納されます。この格納された鍵は、NFS などの、任意のセキュリティー保護された RPC サービスに要求を発行するときに使用されます。プログラム [keylogout\(1\)](#) を使用すると、[keyserv](#) によって格納された鍵を削除できます。

呼び出し側の鍵を取得できない場合や、指定されたパスワードが正しくない場合、keylogin は失敗します。新しいユーザーまたはホストの場合は、[newkey\(1M\)](#) を使用して新しい鍵を追加できます。

システムで複数の認証メカニズムが構成されている場合、構成された各メカニズムの秘密鍵は [keyserv\(1M\)](#) によって復号化および格納されます。

オプション サポートしているオプションは、次のとおりです。

-r /etc/.rootkey ファイルを更新します。このファイルは、スーパーユーザーの暗号化されていない秘密鍵を保持します。このオプションを使用できるのはスーパーユーザーだけです。このオプションは、管理者がシステム起動時にスーパーユーザーとして keylogin を明示的に実行しなくても、スーパーユーザーとして実行されているプロセスが認証された要求を発行できるようにするために使用されます。[keyserv\(1M\)](#) を参照してください。-r オプションは、publickey データベース内のホストのエントリが変更され、publickey データベース内に格納されている実際の鍵のペアに関して /etc/.rootkey ファイルが古くなった場合に管理者によって使用されるべきです。/etc/.rootkey ファイルに対するアクセス権は、スーパーユーザーは読み書きできるが、システム上のその他のユーザーは読み書きできないように設定されています。

システムで複数の認証メカニズムが構成されている場合、構成された各メカニズムの秘密鍵は /etc/.rootkey ファイル内に格納されます。

ファイル /etc/.rootkey スーパーユーザーの秘密鍵

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

```
chkey(1),keylogout(1),login(1),keyserv(1M),newkey(1M),nsswitch.conf(4),
publickey(4),attributes(5)
```


名前	keylogout – keyserv によって格納された秘密鍵の削除
形式	/usr/bin/keylogout [-f]
機能説明	keylogout は、キーサーバープロセス keyserv(1M) によって格納された鍵を削除します。その鍵へのそれ以上のアクセスは取り消されます。ただし、現在のセッション鍵は、期限切れになるか、またはリフレッシュされるまで有効なままになる可能性があります。 keyserv によって格納された鍵を削除すると、セキュリティー保護された RPC サービスが必要なバックグラウンドジョブまたはスケジュールされた at(1) ジョブはすべて失敗します。マシン上に保持される鍵のコピーは1つだけであり、このコマンドの .logout ファイルでの呼び出しは同じマシン上のほかのセッションに影響を与えるため、お勧めできません。
オプション	サポートしているオプションは、次のとおりです。 -f スーパーユーザーの秘密鍵を keylogout で強制的に削除します。デフォルトでは、スーパーユーザーによって起動された NFS などの RPC サービスがすべて停止されるため、スーパーユーザーによる keylogout は禁止されています。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。
関連項目	at(1) , chkey(1) , login(1) , keylogin(1) , keyserv(1M) , newkey(1M) , publickey(4) , attributes(5)

名前 kill - プロセスの終了またはシグナル送出

形式 /usr/bin/kill -s *signal_name* *pid*...
/usr/bin/kill -l [*exit_status*]
/usr/bin/kill [-*signal_name*] *pid*...
/usr/bin/kill [-*signal_number*] *pid*...

機能説明 kill ユーティリティーは、各 *pid* オペランドによって指定されたプロセス (1 つまたは複数) にシグナルを送信します。

pid オペランドのそれぞれについて、kill ユーティリティーは次の引数で呼び出された **kill(2)** 関数と等価のアクションを実行します。

1. *pid* オペランドの値は、*pid* 引数として使用されます。
2. *sig* 引数は、-s オプション、-*signal_name* オプション、または -*signal_number* オプションで指定された値です。あるいは、これらのオプションが何も指定されていない場合は、SIGTERM で指定された値です。

シグナルが送信されるプロセスは、ユーザーがスーパーユーザーでないかぎり現在のユーザーに属していなければなりません。

kill のシェル組み込みバージョンの説明については、「注意事項」を参照してください。

オプション サポートしているオプションは、次のとおりです。

- l (小文字のエル)オペランドが指定されていない場合、該当するシステムでサポートされている *signal_name* のすべての値を書き出します。exit_status オペランドが指定され、それがシェル特殊パラメータ値 ? およびシグナルが終了したプロセスに対応する wait の場合は、そのプロセスを終了するシグナルに対応する *signal_name* が書き込まれます。exit_status オペランドが指定され、それがシグナル番号を示す符号なしの 10 進整数値なら、そのシグナルに対応する *signal_name* が書き込まれます。その他の場合、結果は不確定です。
- s *signal_name* <signal.h> に定義されている記号名の 1 つを使って、送信するシグナルを指定します。*signal_name* の値は、SIG 接頭辞なしで、大文字/小文字を区別しない方法で認識されます。さらに記号名 0 は、ゼロのシグナル値を表すものと認識されます。SIGTERM の代わりに、対応するシグナルが送信されます。
- signal_name* -s *signal_name* と同等です。
- signal_number* SIGTERM の代わりに使用するシグナルを表す、負でない 10 進整数 *signal_number* を指定します。**kill(2)** への有効な呼び出しにおける *sig* 引数と同じです。

オペランド	次のオペランドがサポートされています。
<i>pid</i>	次のうちのいずれかを指定します。 1. シグナルが送信されるプロセスまたはプロセスグループを指定する 10 進の整数。 <i>pid</i> オペランドの正の値、負の値、またはゼロで選択されたプロセスは、kill 関数で説明されているものと同じです。プロセス番号 0 が指定されると、プロセスグループ内のすべてのプロセスにシグナルが送信されます。最初の <i>pid</i> オペランドに負の値を指定する場合、オプションとして解釈されないように、前に -- を付加しなければなりません。 2. シグナルが送信されるバックグラウンドプロセスグループを識別するジョブ制御のジョブ ID。ジョブ制御ジョブ ID 表記が適用できるのは、現在のシェル実行環境で kill を呼び出すときだけです。 <i>pid</i> のジョブ制御ジョブ ID タイプが使用できるのは、ジョブ制御オプションをサポートしているシステムだけです。
<i>exit_status</i>	シグナル番号を指定する 10 進の整数またはシグナルが終了したプロセスの終了ステータス。
使用法	プロセス番号は ps(1) を使って知ることができます。 kill が独自のユーティリティー実行環境で動作するときは、ジョブ制御ジョブ ID 表記は、前述のような働きをする必要はありません。次の 2 つの例を見てください。 <pre>example% nohup kill %1 & example% system("kill %1");</pre> いずれにおいても kill は異なる環境で動作し、各ジョブ番号については関連性はありません。
出力	-l オプションが指定されなければ、標準出力は使用されません。 -l オプションが指定されると、各シグナルの記号名が次のフォーマットで書き込まれます。 <pre>"%s%c", <signal_name>, <separator></pre> この <signal_name> は、SIG 接頭辞を付けずに大文字で指定します。<separator> は復帰改行文字 (NEWLINE) またはスペース文字です。最後に書き込まれるシグナルの場合、<separator> は復帰改行文字になります。 -l オプションと <i>exit_status</i> オペランドの両方が指定されると、対応するシグナルの記号名が次のフォーマットで書き込まれます。 <pre>"%s\n", <signal_name></pre>

使用例

例1 強制終了シグナルを送信する
次のコマンドを見てください。

```
example% kill -9 100 -165
example% kill -s kill 100 -165
example% kill -s KILL 100 -165
```

送信側プロセスが指定されたプロセスへのシグナル送信を許可されていて、指定されたプロセスが存在するという条件で、いずれのコマンドも **100** という ID を持つプロセスと、**165** というプロセスグループ ID を持つすべてのプロセスに **SIGKILL** シグナルを送信します。

例2 最初の負の引数のあいまいさを避ける
シグナル番号またはプロセスグループを指定する最初の負の引数のあいまいさを避けるため、最初の負の数はいつもシグナルとします。そのため、プロセスグループ (たとえば 123) にデフォルトシグナルを送信する場合、アプリケーションは次のようなコマンドを使用しなければなりません。

```
example% kill -TERM -123
example% kill -- -123
```

環境変数

kill の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

次の終了ステータスが返されます。

0 各 *pid* オペランドに対して一致するプロセスが1つ以上見つかり、さらに1つ以上のプロセスに対して指定されたシグナルが処理されました。

>0 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/kill、csh、ksh88、sh

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

ksh

属性タイプ	属性値
使用条件	system/core-os

属性タイプ	属性値
CSI	有効
インタフェースの安定性	不確実

関連項目

[csh\(1\)](#), [getconf\(1\)](#), [jobs\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [ps\(1\)](#), [sh\(1\)](#), [shell_builtins\(1\)](#), [wait\(1\)](#), [kill\(2\)](#), [signal\(3C\)](#), [signal.h\(3HEAD\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

/usr/bin/kill

サポートされるリアルタイムシグナルの数は、[getconf\(1\)](#) の値 `_POSIX_RTSIG_MAX` によって定義されます。

sh

sh には kill コマンドに、*jobid* でプロセスを識別する機能を追加した、組み込みコマンドが用意されています。sh の構文を次に示します。

```
kill [ -sig ] [ pid ] [ %job ]...
kill -l
```

csh

C シェルと csh にも kill の組み込みコマンドが用意されています。構文は次のとおりです。

```
kill [-sig][pid][%job]...
kill -l
```

csh における kill の組み込みコマンドは **TERM** (終了) シグナル (デフォルトの場合) または指定されたシグナルを、指定されたプロセス ID、指定されたジョブ、または現在のジョブへ送信します。シグナルは番号または名前で指定します。シグナルを送るプロセスまたはジョブにデフォルトはありません。そのため、kill だけを入力しても現在のジョブにはシグナルを送信しません。送信中のシグナルが **TERM** (終了) または **HUP** (ハングアップ) の場合、そのジョブまたはプロセスには **CONT** (継続) シグナルも送られます。

-l 送信可能なシグナル名の一覧を表示します。

ksh88

ksh88 における kill の構文は次のとおりです。

```
kill [-sig][pid][%job]...
kill -l
```

ksh88 における kill は、**TERM** (終了) シグナルまたは指定されたシグナルのいずれかを、指定されたジョブまたはプロセスに送信します。シグナルは、番号または名前 ([signal.h\(3HEAD\)](#) に示される。ただし “SIG” という接頭辞を取り除いたもの) で指定します。送信するシグナルが **TERM** (終了) または **HUP** (ハングアップ) の場合、停止中のジョブまたはプロセスには **CONT** (継続) シグナルを送信します。*job* という引数は、活動中のジョブではないプロセスのプロセス ID を指定することもできます。第 2 の形式の kill -l は、シグナル番号とシグナル名をリスト表示します。

ksh

ksh における kill の構文は次のとおりです。

```
kill [-n signum] [-s signame] job ...
kill [-n signum] [-s signame] -l [arg ...]
```

-l が指定されていない最初の形式では、kill は、*job* で指定された 1 つ以上のプロセスにシグナルを送信します。この場合、シグナルが捕捉または無視されていないかぎり、通常、プロセスは終了します。

job には次のいずれかを指定します。

number *job* のプロセス ID。
-number *job* のプロセスグループ ID。
%number ジョブ番号。
%string 名前が *string* で始まるジョブ。
%?string 名前に *string* が含まれるジョブ。
%+
%% 現在のジョブ。
%- 前のジョブ。

-n オプションまたは -s オプションを使用してシグナルが指定されていない場合、SIGTERM シグナルが使用されます。

-l が指定されて、*arg* が指定されない場合、kill は、シグナルのリストを標準出力に書き込みます。それ以外の場合、*arg* には、シグナル名か、シグナル番号またはシグナルによって終了されたプロセスの終了ステータスを表す数値を指定できます。名前を指定した場合、対応するシグナル番号が標準出力に書き込まれます。数値を指定した場合、対応するシグナル名が標準出力に書き込まれます。

-l 前述のとおり、シグナルを送信するのではなく、シグナル名またはシグナル番号を一覧表示します。-n オプションおよび -s オプションは指定できません。

-n *signum* 送信するシグナルの番号を指定します。次の場合を除き、シグナル番号はプラットフォーム間の可搬性はありません。

0	シグナルなし。
1	HUP
2	INT
3	QUIT
6	ABRT
9	KILL

14 ALRM

15 TERM

-s signame 送信するシグナルの名前を指定します。シグナル名は、`<signal.h>` で定義されている名前から接頭辞 `SIG` を取り除いたもので、大文字と小文字は区別されません。`kill -l` は、現在のプラットフォーム上のシグナルのリストを生成します。

`ksh` における `kill` は、終了時に次のいずれかの値を返します。

- 0 各 `job` オペランドに対して一致するプロセスが1つ以上見つかり、さらに1つ以上の一致するプロセスに対して指定されたシグナルが正常に送信されました。
- >0 エラーが発生した。

名前	kinit – Kerberos チケット認可チケットの取得およびキャッシュ
形式	<pre> /usr/bin/kinit [-ARvV] [-p -P] [-f -F] [-a] [-c cache_name] [-C] [-E] [-k [-t keytab_file]] [-l lifetime] [-r renewable_life] [-s start_time] [-n] [-S service_name] [-X attribute=value]] [-T armor_ccache] [principal] </pre>
機能説明	kinit コマンドは、<i>principal</i> の初期のチケット認可チケット (資格) を取得してキャッシュするために使用されます。このチケットは、Kerberos システムによる認証に使用されます。Kerberos システムを使用できるのは、Kerberos 主体を持つユーザーだけです。Kerberos 主体については、kerberos(5) を参照してください。 kinit をオプションなしで使用すると、このユーティリティーは <i>principal</i> と Kerberos パスワードの入力を求め、ローカルの Kerberos サーバーを使用してログインの認証を試みます。<i>principal</i> は、必要に応じてコマンド行で指定できます。 Kerberos がログイン試行を認証した場合、kinit は初期のチケット認可チケットを取得し、それをチケットキャッシュ内に格納します。デフォルトでは、チケットはファイル <code>/tmp/krb5cc_uid</code> 内に格納されます。ここで、<i>uid</i> はユーザー ID 番号を指定します。チケットは、指定された有効期間が経過すると期限切れになります。そのあとは、kinit をふたたび実行する必要があります。kinit では、キャッシュの既存の内容はすべて破棄されます。 コマンド行で指定された値によって、Kerberos 構成ファイル内で <i>lifetime</i> と <i>renewable_life</i> に指定されている値がオーバーライドされます。 ログインセッションを終了する前にアクティブなチケットをすべて破棄するには、kdestroy(1) コマンドを使用できます。
オプション	サポートしているオプションは、次のとおりです。 -a ローカルアドレスを含むチケットを要求します。 -A アドレスを含まないチケットを要求します。 -c <i>cache_name</i> <i>cache_name</i> は、資格 (チケット) のキャッシュ名および場所として使用します。このオプションが使用されていない場合は、デフォルトのキャッシュ名および場所が使用されます。 -C 主体名の正規化を要求します。 -E 主体名を企業名として扱います。 -f 転送可能チケットを要求します。 -F 転送可能ではありません。転送可能チケットを要求しません。 あるホスト上で取得されたチケットは通常、別のホスト上では使用できません。クライアントは、チケットを転送可能と

してマークすることを要求できます。あるチケットに対して TKT_FLG_FORWARDABLE フラグが設定されたあと、ユーザーはこのチケットを使用して新しいチケットを要求できますが、別の IP アドレスを指定します。そのため、ユーザーは自分の現在の資格を使用して、別のマシン上で有効な資格を取得できます。このオプションを使用すると、ユーザーは転送可能でないチケットを明示的に取得できます。

`-k [-t keytab_file]`

ローカルホストの *keytab* ファイル内の鍵から取得されたホストのチケットを要求します。keytab ファイルの名前と場所は、`-t keytab_file` オプションで指定できます。それ以外の場合は、デフォルトの名前と場所が使用されます。

`-l lifetime`

有効期間 *lifetime* を持つチケットを要求します。`-l` オプションが指定されていない場合は、デフォルトのチケット有効期間 (各サイトで構成済み) が使用されます。最大のチケット有効期間 (各サイトで構成済み) より長いチケット有効期間を指定すると、最大の有効期間を持つチケットが生成されます。*lifetime* で指定できる有効な期間の形式については、「時間形式」セクションを参照してください。[kdc.conf\(4\)](#) および [kadmind\(1M\)](#) (サーバー主体の有効期間の値を検証するには `getprinc` コマンド) を参照してください。

返されるチケットの有効期間は、次の最小値です。

- コマンド行で指定された値。
- KDC 構成ファイルで指定された値。
- サーバー主体の Kerberos データベースで指定された値。kinit の場合は、`krbtgt/realm name` です。
- ユーザー主体の Kerberos データベースで指定された値。

`-n`

匿名の処理を要求します。

2つのタイプの匿名主体がサポートされています。完全に匿名の Kerberos の場合は、`pkinit` を KDC 上で構成し、`pkinit_anchors` をクライアントの `krb5.conf` 内に構成します。次に、`-n` オプションを使用して `@REALM` の形式の主体 (空の主体名のあとにアットマーク記号とレルム名) を指定します。KDC によって許可されている場合は、匿名チケットが返されます。

また、2番目の形式の匿名チケットもサポートされています。これらのレルムが公開されたチケットでは、クライアントの識別情報は非表示になりますが、クライアントのレルム

は表示されます。このモードでは、`kinit -n` を使用して通常の主体名を指定します。KDCでサポートされている場合は、匿名主体によって(レルムではなく)主体が置き換えられます。リリース 1.8 の時点で、完全に匿名の操作をサポートしているのは MIT Kerberos KDC だけです。

- p プロキシ可能チケットを要求します。
- P プロキシ可能ではありません。プロキシ可能チケットを要求しません。

プロキシ可能チケットは、チケット認可チケット内のアドレス以外の IP アドレスを使用してサービスのチケットを取得できるチケットです。このオプションを使用すると、ユーザーはプロキシ可能でないチケットを明示的に取得できます。

- r *renewable_life* *renewable_life* の合計有効期間を持つ更新可能なチケットを要求します。*renewable_life* で指定できる有効な期間の形式については、「時間形式」セクションを参照してください。[kdc.conf\(4\)](#) および [kadmin\(1M\)](#) (サーバー主体の有効期間の値を検証するには `getprinc` コマンド) を参照してください。

返されるチケットの更新可能な有効期間は、次の最小値です。

- コマンド行で指定された値。
- KDC 構成ファイルで指定された値。
- サーバー主体の Kerberos データベースで指定された値。`kinit` の場合は、`krbtgt/realms` です。
- ユーザー主体の Kerberos データベースで指定された値。

- R チケット認可チケットの更新を要求します。期限が切れたチケットは、そのチケットがまだ更新可能な期間内に含まれていないとしても更新できません。

- s *start_time* *start_time* から有効になる遅延チケットを要求します。遅延チケットは `invalid` フラグが設定された状態で発行されるため、使用する前に KDC にフィードバックする必要があります。*start_time* で指定できる有効な絶対時間または期間の形式については、「時間形式」セクションを参照してください。`kinit` は、期間の一致を試みる前に、まず絶対時間を一致させようとしています。

- S *service_name* 初期のチケットを取得するときに使用する代替サービス名を指定します。

- `-T armor_ccache` KDCでサポートされている場合は、すでにチケットが含まれている資格キャッシュの名前(ccache)を指定します。このccacheは、要求を保護することによって、攻撃者がその要求を攻撃するには保護チケットの鍵と、認証に使用される主体の鍵の両方を知る必要があるようにするために使用されます。
- また、この保護により、KDCからの応答が遷移中に変更されないことも保証されます。
- `-v` キャッシュ内の(*invalid* フラグが設定された)チケット認可チケットを検証のためにKDCに渡すことを要求します。チケットが要求された時間範囲内に含まれている場合、キャッシュは、検証されたチケットに置き換えられます。
- `-V` 詳細情報を出力します。ユーザーに認証やバージョンの確認などの詳細情報を表示します。
- `-X attribute[=value]` 事前認証プラグインに渡される事前認証の属性と値を指定します。*attribute* と *value* に指定できる値は、事前認証プラグインによって異なります。このオプションは、複数の属性を指定するために複数回指定できます。値が指定されていない場合は、*yes* であると見なされます。
- OpenSSL の pkinit 事前認証メカニズムでは、次の属性が認識されます。
- X509_user_identity=URI**
ユーザーの X509 識別情報をどこで見つけるかを指定します。
- 有効な URI タイプは FILE、DIR、PKCS11、PKCS12、および ENV です。詳細は、「pkinit の URI タイプ」セクションを参照してください。
- X509_anchors=URI**
信頼できる X509 アンカー情報をどこで見つけるかを指定します。
- 有効な URI タイプは FILE と DIR です。詳細は、「pkinit の URI タイプ」セクションを参照してください。
- flag_RSA_PROTOCOL[=yes]**
デフォルトの Diffie-Hellman プロトコルではなく、RSA の使用を指定します。

pkinitのURIタイプ FILE:*file-name*[,*key-file-name*]

このオプションには、コンテキスト固有の動作があります。

X509_user_identity *file-name* は、ユーザーの証明書が含まれた PEM 形式ファイルの名前を指定します。*key-file-name* が指定されていない場合は、ユーザーの秘密鍵も *file-name* 内に存在すると想定されます。それ以外の場合は、*key-file-name* が、秘密鍵を含むファイルの名前です。

X509_anchors *file-name* は、OpenSSL スタイルの ca-bundle ファイルの名前であると見なされます。ca-bundle ファイルは、base-64 でエンコードされているべきです。

DIR:*directory-name*

このオプションには、コンテキスト固有の動作があります。

X509_user_identity *directory-name* は、*.crt および *.key という名前のファイルを含むディレクトリを指定し、ここで、ファイル名の最初の部分は、証明書ファイルと秘密鍵ファイルの一致するペアと同じです。.crt で終わる名前を持つファイルが見つかった場合は、.key で終わる一致するファイルが、秘密鍵を含んでいると見なされます。このようなファイルが見つからない場合、.crt 内の証明書は使用されません。

X509_anchors *directory-name* は、各 CA 証明書が hash-of-ca-cert.# という名前のファイル内に格納されている、OpenSSL スタイルのハッシュされた CA ディレクトリであると見なされます。このインフラストラクチャーが推奨されますが、そのディレクトリ内のすべてのファイルが検査され、証明書 (PEM 形式) が含まれている場合は、それらのファイルが使用されます。

PKCS12:*pkcs12-file-name*

pkcs12-file-name は、ユーザーの証明書と秘密鍵が含まれた PKCS #12 形式ファイルの名前です。

PKCS11:[slotid=*slot-id*][:token=*token-label*][:certid=*cert-id*][:certlabel=*cert-label*]

すべてのキーワードおよび値が省略可能です。PKCS11 モジュール (たとえば、opensc-pkcs11.so) が、libpkcs11(3LIB) の下の暗号化プロバイダとしてインストールされている必要があります。slotid= または token=、あるいはその両方を指定すると、特定のスマートカードリーダーまたはトークンを強制的に使用できます (複数使用可能な場合)。certid= または certlabel=、あるいはその両方を指定すると、デバイス上の特定の証明書を強制的に選択できます。pkinit で使用する特定の証明書を選択するためのその他の方法については、pkinit_cert_match 構成オプションを参照してください。

ENV:*environment-variable-name*

environment-variable-name は、前のいずれかの値に準拠する値に設定されている環境変数の名前を指定します。たとえば、ENV:X509_PROXY と指定し、ここで、環境変数 X509_PROXY は FILE:/tmp/my_proxy.pem に設定されています。

時間形式

-s *start_time* オプションには、次の絶対時間の形式を使用できます。これらの例は、1999 年 7 月 2 日午後 1:35:30 の日付と時間に基づいています。

絶対時間の形式	例
<i>yymmddhhmm</i> [<i>ss</i>]	990702133530
<i>hhmm</i> [<i>ss</i>]	133530
<i>yy.mm.dd.hh.mm.ss</i>	99:07:02:13:35:30
<i>hh:mm</i> [: <i>ss</i>]	13:35:30
<i>ldate:ltime</i>	07-07-99:13:35:30
<i>dd-month-yyyy:hh:mm</i> [: <i>ss</i>]	02-july-1999:13:35:30

Variable

変数	説明
<i>dd</i>	日
<i>hh</i>	時間 (24 時間制)
<i>mm</i>	分
<i>ss</i>	秒
<i>yy</i>	世紀内の年 (0-68 は 2000 から 2068 まで、69-99 は 1969 から 1999 まで)
<i>yyyy</i>	世紀を含む年
<i>month</i>	ロケールの完全な、または短縮された月の名前
<i>ldate</i>	ロケールの適切な日付表現
<i>ltime</i>	ロケールの適切な時間表現

-l *lifetime*、-r *renewable_life*、および -s *start_time* オプションには、次の期間の形式を使用できます。これらの例は、14 日、7 時間、5 分、および 30 秒の期間に基づいています。

期間の形式	例
#d	14d
#h	7h
#m	5m
#s	30s
#d#h#m#s	14d7h5m30s
#h#m[#s]	7h5m30s
days-hh:mm:ss	14-07:05:30
hours:mm[:ss]	7:05:30

区切り記号	説明
d	日数
h	時間数
m	分数
s	秒数

変数	説明
#	数字
days	日数
hours	時間数
hh	時間 (24 時間制)
mm	分
ss	秒

環境変数

kinit は、次の環境変数を使用します。

KRB5CCNAME 資格 (チケット) のキャッシュの場所。構文および詳細については、[krb5envvar\(5\)](#) を参照してください。

ファイル

/tmp/krb5cc_uid デフォルトの資格キャッシュ (*uid* は、ユーザーの 10 進数の UID です)。

/etc/krb5/krb5.keytab ローカルホストの keytab ファイルのデフォルトの場所。

/etc/krb5/krb5.conf ローカルホストの構成ファイルのデフォルトの場所。[krb5.conf\(4\)](#) を参照してください。

属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。						
	<table><tr><th>属性タイプ</th><th>属性値</th></tr><tr><td>使用条件</td><td>service/security/kerberos-5</td></tr><tr><td>インタフェースの安定性</td><td>下記を参照。</td></tr></table>	属性タイプ	属性値	使用条件	service/security/kerberos-5	インタフェースの安定性	下記を参照。
属性タイプ	属性値						
使用条件	service/security/kerberos-5						
インタフェースの安定性	下記を参照。						
	コマンド引数は「確実」です。コマンド出力は「不確実」です。						
関連項目	kdestroy(1) , klist(1) , kadmin(1M) , ktkt_warnd(1M) , libpkcs11(3LIB) , kdc.conf(4) , krb5.conf(4) , attributes(5) , kerberos(5) , krb5envvar(5) , pam_krb5(5)						
注意事項	成功した場合、kinit は ktkt_warnd(1M) に、初期の資格 (チケット認可チケット) の期限が近づいてきたらユーザーに警告するよう通知します。						

名前	klist – 現在保持されている Kerberos チケットの一覧表示																												
形式	<pre>/usr/bin/klist [-e] [-c] [-f] [-s] [-a [-n]] [cache_name] [-k [-t]] [-K] [keytab_file]</pre>																												
機能説明	klist ユーティリティーは、資格キャッシュの名前、チケットが対象としている (チケットファイルに示されている) 主体の識別情報、ユーザーが現在保持しているすべての Kerberos チケットの主体名を、オーセンティケーターごとの発行および有効期限とともに出力します。主体名は <i>name/instance@realm</i> という形式で一覧表示され、インスタンスが含まれていない場合は「/」が省略され、レルムが含まれていない場合は「@」が省略されます。 <i>cache_file</i> または <i>keytab_name</i> が指定されていない場合、klist は、必要に応じてデフォルトの資格キャッシュまたは keytab ファイル内の資格を表示します。デフォルトでは、チケットはファイル <i>/tmp/krb5cc_uid</i> 内に格納されます。ここで、<i>uid</i> はユーザーの現在のユーザー ID です。																												
オプション	サポートしているオプションは、次のとおりです。 <table><tr><td>-a</td><td>資格内のアドレスのリストを表示します。可能な場合は、構成されているネームサービスを使用して、数値ネットワークアドレスを関連付けられたホスト名に変換します。</td></tr><tr><td>-c [cache_name]</td><td>資格キャッシュ内に保持されているチケットを一覧表示します。-c も -k も指定されていない場合は、これがデフォルトです。</td></tr><tr><td>-e</td><td>セッション鍵の暗号化タイプと、資格キャッシュ内の各資格または keytab ファイル内の各鍵のチケットを表示します。</td></tr><tr><td>-f</td><td>次の略語を使用して、資格内に存在するフラグを示します。<table><tr><td>a</td><td>匿名</td></tr><tr><td>A</td><td>事前認証済み</td></tr><tr><td>d</td><td>遅延</td></tr><tr><td>D</td><td>遅延可能</td></tr><tr><td>f</td><td>転送済み</td></tr><tr><td>F</td><td>転送可能</td></tr><tr><td>H</td><td>ハードウェア認証済み</td></tr><tr><td>i</td><td>無効</td></tr><tr><td>I</td><td>初期</td></tr><tr><td>O</td><td>委任として OK</td></tr></table></td></tr></table>	-a	資格内のアドレスのリストを表示します。可能な場合は、構成されているネームサービスを使用して、数値ネットワークアドレスを関連付けられたホスト名に変換します。	-c [cache_name]	資格キャッシュ内に保持されているチケットを一覧表示します。-c も -k も指定されていない場合は、これがデフォルトです。	-e	セッション鍵の暗号化タイプと、資格キャッシュ内の各資格または keytab ファイル内の各鍵のチケットを表示します。	-f	次の略語を使用して、資格内に存在するフラグを示します。 <table><tr><td>a</td><td>匿名</td></tr><tr><td>A</td><td>事前認証済み</td></tr><tr><td>d</td><td>遅延</td></tr><tr><td>D</td><td>遅延可能</td></tr><tr><td>f</td><td>転送済み</td></tr><tr><td>F</td><td>転送可能</td></tr><tr><td>H</td><td>ハードウェア認証済み</td></tr><tr><td>i</td><td>無効</td></tr><tr><td>I</td><td>初期</td></tr><tr><td>O</td><td>委任として OK</td></tr></table>	a	匿名	A	事前認証済み	d	遅延	D	遅延可能	f	転送済み	F	転送可能	H	ハードウェア認証済み	i	無効	I	初期	O	委任として OK
-a	資格内のアドレスのリストを表示します。可能な場合は、構成されているネームサービスを使用して、数値ネットワークアドレスを関連付けられたホスト名に変換します。																												
-c [cache_name]	資格キャッシュ内に保持されているチケットを一覧表示します。-c も -k も指定されていない場合は、これがデフォルトです。																												
-e	セッション鍵の暗号化タイプと、資格キャッシュ内の各資格または keytab ファイル内の各鍵のチケットを表示します。																												
-f	次の略語を使用して、資格内に存在するフラグを示します。 <table><tr><td>a</td><td>匿名</td></tr><tr><td>A</td><td>事前認証済み</td></tr><tr><td>d</td><td>遅延</td></tr><tr><td>D</td><td>遅延可能</td></tr><tr><td>f</td><td>転送済み</td></tr><tr><td>F</td><td>転送可能</td></tr><tr><td>H</td><td>ハードウェア認証済み</td></tr><tr><td>i</td><td>無効</td></tr><tr><td>I</td><td>初期</td></tr><tr><td>O</td><td>委任として OK</td></tr></table>	a	匿名	A	事前認証済み	d	遅延	D	遅延可能	f	転送済み	F	転送可能	H	ハードウェア認証済み	i	無効	I	初期	O	委任として OK								
a	匿名																												
A	事前認証済み																												
d	遅延																												
D	遅延可能																												
f	転送済み																												
F	転送可能																												
H	ハードウェア認証済み																												
i	無効																												
I	初期																												
O	委任として OK																												

- p プロキシ
- P プロキシ可能
- R 更新可能
- T 移行ポリシー確認済み
- k [keytab_file] keytab ファイル内に保持されている鍵を一覧表示します。
- K keytab ファイル内の各 keytab エントリにある暗号化鍵の値を表示します。
- n 逆引きアドレスの代わりに、数値 IP アドレスを示します。-a オプションでのみ有効です。
- s klist が暗黙のうちに実行されます (出力は生成されませんが、資格キャッシュが見つかったかどうかに応じて終了ステータスは引き続き設定されます。終了ステータスは、klist で資格キャッシュが見つかった場合は0、見つからない場合や、ローカルレルムの TGT の期限が切れている場合は1です。
- t keytab ファイル内の keytab エントリごとの時間エントリタイムスタンプを表示します。

- 環境変数
- klist は、次の環境変数を使用します。
- KRB5CCNAME 資格(チケット)のキャッシュの場所。構文および詳細については、[krb5envvar\(5\)](#)を参照してください。
- ファイル
- /tmp/krb5cc_uid デフォルトの資格キャッシュ (uid は、ユーザーの 10 進数の UID です)。

/etc/krb5/krb5.keytab ローカルホストの keytab ファイルのデフォルトの場所。

/etc/krb5/krb5.conf ローカルホストの構成ファイルのデフォルトの場所。[krb5.conf\(4\)](#)を参照してください。
- 属性
- 属性についての詳細は、マニュアルページの [attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
インタフェースの安定性	下記を参照。

コマンド引数は「確実」です。コマンド出力は「不確実」です。

関連項目	kdestroy(1) , kinit(1) , krb5.conf(4) , attributes(5) , krb5envvar(5) , kerberos(5)
使用上の留意点	サービス鍵ファイルとしてファイルを読み取り中、エラーチェックはほとんど実行されません。

名前	kmdb – 原状態カーネルデバッガ
形式	
ブート時の読み込み	SPARC <pre>ok boot [device-specifier] -k [-d] [boot-flags]</pre> <pre>ok boot [device-specifier] kmdb [-d] [boot-flags]</pre> x86 <pre>kernel\$ /platform/i86pc/kernel/\$ISADIR/unix -k [-d] [boot-flags]</pre>
実行時の読み込み	<pre>mdb -K</pre>
機能説明	kmdb は、ライブカーネルコンテキストでの mdb(1) のユーザーインタフェースと機能を実装した、対話型のカーネルデバッガです。kmdb を使用すると、カーネル実行の制御およびライブカーネル状態の検査と変更を実行できます。kmdb の読み込みは、ブートセッションの開始時またはシステムのブート後に実行できます。 このマニュアルページでは、kmdb に固有の機能、および mdb(1) と比較して異なる kmdb の機能について説明します。mdb(1)、および kmdb により実装される機能の詳細については、mdb(1) のマニュアルページおよび『Oracle Solaris Modular Debugger Guide』を参照してください。
読み込みと読み込み解除	ブート時の読み込み 要求があると、カーネル実行時リンカー (krtld) は kmdb を読み込んでから制御をカーネルに渡します。-d フラグが指定されていると、デバッガはシステムの制御を取得してから unix オブジェクトの初期関数を実行します。-d が指定されていない場合、kmdb は読み込まれますが、明示的にそのモードに入るまで制御は取得しません。後述の「デバッガのエントリ」のセクションを参照してください。ブート時に kmdb の読み込みを実行するブートコマンドの一覧については、前述の「形式」のセクションを参照してください。このコマンドを使って、SPARC マシンのブート時に kmdb が常に読み込まれるようにする方法については、eeprom(1M) を参照してください。 ブート時に読み込まれた kmdb は、システムのリブートによってのみ読み込みを解除できます。 kmdb の一部の機能はカーネルサービスの存在に依存しているため、ブート時に読み込まれた kmdb からすぐに使用することはできません。特に dmods の読み込みと読み込み解除は、モジュールサブシステムが初期化されるまで使用できません。要求は、処理可能になるまで待ち行列に入れられます。同様に、仮想アドレスから物理アドレスへの変換も VM システムが初期化されるまで使用できません。この機能が使用可能になるまで、変換を試みても失敗します。 実行時の読み込み mdb(1) に -K フラグを使用すると、システムのブート後に kmdb を読み込むこともできます。この方法で読み込まれた場合、kmdb はただちにシステムの制御を取

得します。実行時に読み込まれた kmdb を読み込み解除するには、[mdb\(1\)](#) に `-U` フラグを使用するか、デバッガ内で `::quit dcmd` に `-u` フラグを使用します。

端末タイプ

読み込み時に、kmdb はシステムコンソールで使用されている端末タイプの判別を試みます。デバッグ対象のシステムにシステムコンソール用のキーボードとローカルディスプレイが接続されている場合、kmdb はマシンに適した端末タイプ (SPARC の場合には「sun」、x86 の場合には「sun-color」) を使用します。シリアルコンソールが使用されている場合、ブート時に読み込まれる kmdb のデフォルトの端末タイプは「vt100」になります。実行時に読み込まれる kmdb のデフォルトは、[mdb\(1\)](#) により要求される端末タイプになります。[mdb\(1\)](#) は、`-T` フラグで上書きされていないかぎり、`TERM` 環境変数の値で指定された端末タイプを要求します。`::term` を使うと、現在の端末タイプを表示できます。

デバッガのエントリ

デバッガのエントリは、明示的または暗黙的に要求できます。暗黙的なエントリは、ブレークポイントまたはその他の実行制御機能の使用時に検出されます。詳細は、「実行制御」のセクションを参照してください。

デバッガのエントリを明示的に要求するプライマリな方法には、ローカルコンソールでシステムのキーボードアポートシーケンスを使用する方法と、シリアルコンソールで `BREAK` 文字を使用する方法があります。アポートシーケンスは、SPARC システムのローカルコンソールでは `STOP-A` または `Shift-Pause`、x86 システムのローカルコンソールでは `F1-A` または `Shift-Pause` です。アポートシーケンスおよびそれを無効にする方法については、[kbd\(1\)](#) を参照してください。

デバッガのエントリを要求する 2 番目の方法は、[mdb\(1\)](#) コマンドを使用するものです。デバッガの読み込み後に [mdb\(1\)](#) を `-K` フラグを指定して呼び出すと、デバッガのエントリがトリガーされます。

カーネルパニックおよび kmdb が読み込まれると、デフォルトでパニックルーチンが kmdb に入り、ライブデバッグが実行されます。ダンプデバイスが指定されていて、かつ `::cont` に入ると、デバッガが終了してクラッシュダンプが実行されます。パニックの発生時にカーネルが kmdb に入らないようにするには、`nopanicdebug` 変数を 1 に設定します。kmdb を使用するか、次の行を `/etc/system` に含めて、`nopanicdebug` 変数を 1 に設定します。

```
set nopanicdebug = 1
```

これは、kmdb を読み込んだ状態を維持するが、デバッガに入ることなくパニックが常にクラッシュダンプをトリガーするようにする場合に役立ちます。

実行制御

ほとんどの場合、kmdb によりカーネルに提供される実行制御機能は、[mdb\(1\)](#) プロセスのターゲットにより提供される実行制御機能を反映しています。ブレークポイント (`::bp`)、ウォッチポイント (`::wp`)、`::continue`、および `::step` のさまざまな特性を使用できます。

カーネルにより提供される無制限のユーザープロセスウォッチポイントとは異なり、kmdb は 1 つの CPU ウォッチポイントセットに制限されており、これによってウォッチポイントの数、サイズ、種類が限定されます。ウォッチポイントがハードウェアのサポートするウォッチポイントと互換性がない場合、`::wp` コマンドはそのウォッチポイントの作成を許可しません。

デバッグモジュール (dmods)

mdb(1) の場合と同様に、kmdb をインストールすると、多数のサブシステム固有のデバッグモジュール (dmods) もインストールされます。dmods は、サポートするサブシステムの読み込みや読み込み解除に合わせて、自動的に読み込みまたは読み込み解除されます。`::load` や `::unload` を使用すると、dmods を明示的に読み込んだり、読み込み解除したりすることもできます。

kmdb は、カーネルの機能を使って dmods の読み込みと読み込み解除を行います。また、システムの実行を再開して要求された各操作を実行する必要があります。dmod の読み込みまたは読み込み解除が完了すると、システムが停止して、自動的にデバッグが再度エントリされます。dmod の読み込み時には、要求された dmod の読み込みが成功または失敗すると処理が完了します。どちらの場合でも、ステータスメッセージが出力されます。

プロセッサ固有の機能

一部の機能は、個々のプロセッサタイプに固有のものです。この種の機能の例には、さまざまな x86 プロセッサで提供されている分岐トレースがあります。これらのプロセッサ固有の機能にアクセスするには、それをサポートしているシステムにのみ存在するプロセッサ固有の `dcmd` を使用します。プロセッサ固有のサポートを利用できるかどうかは、`::status dcmd` の出力に示されます。デバッグは、カーネルに基づいてプロセッサタイプを判断します。デバッグは特定のプロセッサタイプをサポートしている場合でも、カーネルによりプロセッサの識別が完了するまでサポートは公開されません。

カーネルマクロ

デバッグに内蔵されているコンパイル済みマクロセットにアクセスできます。使用できるのはコンパイル済みのマクロだけです。**mdb(1)** とは異なり、`$< dcmd` を使用して任意の場所からマクロを読み込むことはできません。使用可能なマクロの一覧を表示するには、`$M` コマンドを使用します。

組み込み dcmd

このセクションでは、kmdb に固有の `dcmd`、および kmdb の `dcmd` の中で **mdb(1)** と動作が異なるものの一覧を示します。

```
[address]:bp [+/-dDestT] [-c cmd] [-n count] sym ...
address :b [cmd ...]
```

指定された場所にブレークポイントを設定します。`::bp dcmd` は、指定されたアドレスまたはシンボル (dcmd の前にある明示的な式で指定されたオプションのアドレスを含む) ごと、そして、dcmd の後ろにある文字列または即値ごとにブレークポイントを設定します。引数には、指定された特定の仮想アドレスを示すシンボル名または即値を指定できます。

シンボル名を指定した場合は、まだ評価できないシンボルを参照できます。まだ読み込んでいないオブジェクト内のオブジェクト名や関数名を参照できます。こ

の場合、ターゲット内のブレークポイントは延期され、指定された名前に一致するオブジェクトが読み込まれるまでアクティブ(有効)になりません。オブジェクトが読み込まれると、このブレークポイントは自動的に有効になります。

-d、-D、-e、-s、-t、-T、-c、および -n オプションは、`::evset dcmd` で使用するのと同じです。`::evset` については、[mdb\(1\)](#) を参照してください。`dcmd` の `:b` 形式を使用した場合、ブレークポイントは `dcmd` の前にある式で指定した仮想アドレスだけに設定されます。`:b dcmd` の後ろにある引数は連結され、コールバック文字列となります。この文字列にメタキャラクタが含まれる場合、文字列を引用符で囲む必要があります。

`::branches [-v]`

(x86 のみ)

CPU が選択した最後の分岐を表示します。この `dcmd` は x86 システムでのみサポートされます。また、プロセッサ固有のサポートが検出されて有効化されている場合にのみ使用できます。表示される分岐の数と種類は、CPU の提供する分岐トレース機能によって異なります。-v オプションを使用すると、指定した分岐より前の命令が表示されます。

`[function] ::call [arg [arg ...]]`

指定した関数を、指定した引数を使って呼び出します。呼び出された関数は、読み込まれたモジュールのシンボルテーブル内の関数として表示されます。文字列引数は、参照により渡されます。呼び出しが完了すると、その関数の戻り値が表示されます。

この `dcmd` は、きわめて慎重に使用する必要があります。呼び出しが行われても、カーネルは再開されません。呼び出された関数は、使用可能なカーネルサービスに関してどのような想定も行えません。また、ブロックするような操作や呼び出しを実行してはなりません。ユーザーは、呼び出された関数により発生する副作用についても警戒しておく必要があります。カーネルの安定性が影響を受ける可能性があるためです。

`[addr] ::cpuregs [-c cpuid]`

指定された CPU の現在の汎用レジスタセットを、`::regs` で使用される書式で表示します。

`[addr] ::cpustack [-c cpuid]`

指定された CPU の C スタックバックトレースを出力します。表示されるバックトレースは、デバッガが指定された CPU をエントリまたは停止した時点のものです。

`addr[,len] ::in [-L len]`

(x86 のみ)

`addr` で指定された入出力ポートから `len` バイトを読み込みます。-L オプションが指定されている場合は、このオプションが繰り返し回数の値よりも優先されます。読み取り長は 1、2、または 4 バイトにする必要があります、ポートアドレスはこの長さと同じ配置にする必要があります。

`addr[,len]::out [-L len] value`

(x86 のみ)

値を `addr` で指定された `len` バイトの入出力ポートに書き込みます。-L オプションが指定されている場合は、このオプションが繰り返し回数の値よりも優先されます。書き込み長は 1、2、または 4 バイトにする必要があります、ポートアドレスはこの長さと同じ配置にする必要があります。

`::quit [-u]`

`$q`

デバッガを終了します。-u オプションを使用すると、システムが再開されて、デバッガが読み込み解除されます。ブート時にデバッガが読み込まれた場合、-u オプションは使用できません。-u オプションを使用すると、SPARC システムが終了して、ブート PROM の `ok` プロンプトが表示されます。go コマンドを使用するとデバッガを再度エントリできます。x86 システムでは、マシンをリブートするための許可を求めるプロンプトが表示されます。

`::step [over|out|branch]`

ターゲットを 1 命令だけ実行します。サブルーチンの呼び出しまで進めるには、オプションの `over` 引数を使用します。オプションの `out` 引数を指定した場合、制御が現在の関数から戻ってくるまで、ターゲットプログラムは実行を継続します。

オプションの `branch` 引数は、x86 システムでプロセッサ固有のサポートが検出されて有効化されている場合にのみ使用できます。`::step branch` が指定されている場合、次の分岐命令が検出されるまでターゲットプログラムは実行を継続します。

SPARC システムでは、`::step dcmd` を使用して「`ta`」命令を実行することはできません。同様に、x86 システムで、これを使用して「`int`」命令を実行することもできません。実行によりデバッガで解決できないトラップが生成された場合は、その効果に対するメッセージが出力されて実行が失敗します。

`cpuid::switch`

`cpuid:x`

指定された CPU を代表として使用します。スタックトレース、汎用レジスタダンプ、および類似の機能で、新しい代表 CPU がデータソースとして使用されます。新しい代表 CPU では完全な実行制御機能を使用できます。

`::term`

現在の端末タイプを表示します。

`addr[,len]::wp [+/-dDestT] [-rwx] [-pi] [-n count] [-c cmd]`

`addr[,len]:a [cmd ...]`

`addr[,len]:p [cmd ...]`

`addr[,len]:w [cmd ...]`

指定されたアドレスにウォッチポイントを設定します。これは、デフォルトで仮想アドレスとして解釈されます。-p オプションを指定した場合、アドレスは物

理アドレスとして解釈されます。x86 プラットフォームでは、`-i` オプションを使用するとウォッチポイントを入出力ポート上に設定できます。`-i` オプションを指定した場合、アドレスは入出力ポートのアドレスとして解釈されます。

監視される領域の長さをバイト数で設定するには、`dcmd` の前に繰り返し回数を指定します。長さを明示的に設定しない場合、デフォルトは1バイトです。`::wp dcmd` を使用すると、読み取り (`-r` オプション)、書き込み (`-w` オプション)、または実行 (`-x` オプション) のアクセス権の任意の組み合わせでのトリガーが可能となるようにウォッチポイントを構成できます。

`-d`、`-D`、`-e`、`-s`、`-t`、`-T`、`-c`、および `-n` オプションは、`::evset dcmd` で使用するのと同じです。`::evset` については、[mdb\(1\)](#) を参照してください。`:a dcmd` は、指定されたアドレスに読み取り権のウォッチポイントを設定します。`:p dcmd` は、指定されたアドレスに実行権のウォッチポイントを設定します。`:w dcmd` は、指定されたアドレスに書き込み権のウォッチポイントを設定します。`:a`、`:p`、および `:w dcmd` の後ろにある引数は連結され、コールバック文字列となります。この文字列にメタキャラクタが含まれる場合、文字列を引用符で囲む必要があります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/kernel (デバッグ)
	developer/debug/mdb (dmod)
インタフェースの安定性	確実

関連項目 [mdb\(1\)](#), [boot\(1M\)](#), [dumpadm\(1M\)](#), [eeprom\(1M\)](#), [kernel\(1M\)](#), [system\(4\)](#), [attributes\(5\)](#)

『Oracle Solaris Modular Debugger Guide』

SPARC のみ [kbd\(1\)](#)

注意事項

- デバッグで使用可能なメモリーに関する制限
- デバッグで使用可能なメモリー領域は、デバッグの読み込み時に割り当てられ、その時点で固定されます。`dcmd` が使用可能なメモリーより多くのメモリーの割り当てを試みると、終了させられます (可能な場合)。デバッグは、メモリー不足の状態からの正常な回復を試みますが、それができない場合にはシステムを強制終了することがあります。この制約は、特に 32 ビットの x86 システムで深刻な問題になります。
- パフォーマンスへの影響
- `kmdb` の読み込みにより、システムのパフォーマンスが悪影響を受けます。これは、デバッグがカーネルメモリーおよびその他の限定されたシステムリソースを消費するためです。

kmdb のブートによ
る panic() スタック
の取得

[eeprom\(1M\)](#) を使用してブート時にシステムが常に kmdb を読み込むように設定すると、SPARC マシンで発生する `panic()` のトラブルシューティングに役立つことがあります。パニックが発生すると、システムがリブートを開始することにより、パニックスタックがコンソールから消去されます。kmdb をブートすることで、パニックスタックを取得して解釈できるようになります。ブート時に kmdb の読み込みを指定する方法については、[eeprom\(1M\)](#) の例を参照してください。

名前	kmfcfg – 鍵管理ポリシーおよびプラグイン構成ユーティリティー
形式	kmfcfg <i>subcommand</i> [<i>option</i> ...]
機能説明	ユーザーは kmfcfg コマンドを使用すると、鍵管理フレームワーク (KMF) ポリシーデータベースを構成できます。KMF ポリシーデータベース (DB) は、KMF フレームワークから管理される鍵と証明書の使用を制限します。 kmfcfg を使用すると、システムのデフォルトのデータベースファイル /etc/security/kmfpolicy.xml またはユーザー定義のデータベースファイルのポリシー定義を一覧表示、作成、変更、削除、インポート、およびエクスポートできます。 プラグイン構成の場合、ユーザーは kmfcfg を使用して、プラグイン情報の表示、KMF プラグインのインストール/アンインストール、およびプラグインオプションの変更を行えます。

サブコマンド サポートされているサブコマンドは次のとおりです。

create

ポリシーデータベースファイルに新しいポリシーを追加します。

create サブコマンドの書式は次のとおりです。

```
create [dbfile=dbfile] policy=policyname
    [ignore-date=true|false]
    [ignore-unknown-eku=true|false]
    [ignore-trust-anchor=true|false]
    [validity-adjusttime=adjusttime]
    [ta-name=trust anchor subject DN]
    [ta-name=trust anchor subject DN | search]
    [ta-serial=trust anchor serial number]
    [ocsp-responder=URL]
    [ocsp-proxy=URL]
    [ocsp-use-cert-responder=true|false]
    [ocsp-response-lifetime=timelimit]
    [ocsp-ignore-response-sign=true|false]
    [ocsp-responder-cert-name=Issuer DN]
    [ocsp-responder-cert-serial=serial number]
    [crl-basefilename=basefilename]
    [crl-directory=directory]
    [crl-get-crl-uri=true|false]
    [crl-proxy=URL]
    [crl-ignore-crl-sign=true|false]
    [crl-ignore-crl-date=true|false]
    [keyusage=digitalSignature|nonRepudiation
        |keyEncipherment | dataEncipherment |
        keyAgreement |keyCertSign |
        cRLSign | encipherOnly | decipherOnly],[...]
```

```
[ekunames=serverAuth | clientAuth |
      codeSigning | emailProtection |
      ipsecEndSystem | ipsecTunnel |
      ipsecUser | timeStamping |
      OCSPSigning],[...]
[ekuoids=OID,OID,OID...]
[mapper-name=name of the mapper]
[mapper-dir=dir where mapper library resides]
[mapper-path=full pathname of mapper library]
[mapper-options=mapper options]
```

create サブコマンドは次のオプションをサポートします。

crl-basefilename=filename

crl-directory=directory

これら2つの属性は、CRL ファイルの場所を指定するために使用されます。**crl-basefilename** 属性はCRL ファイルのベースファイル名を表します。**crl-directory** 属性は、CRL ファイルのディレクトリを表します。デフォルトで現在のディレクトリに設定されています。

crl-get-crl-uri 属性が **true** に設定され、**crl-basefilename** が指定されていない場合、キャッシュされたCRL ファイルの **basefilename** が、CRL ファイルの取得に使用されるURI のベース名になります。

crl-get-crl-uri 属性が **false** に設定されている場合、入力CRL ファイルを示すために **crl-basefilename** を指定する必要があります。**crl-get-crl-uri** はデフォルトで **false** に設定されています。

これら2つの属性は、ファイルベースのCRL プラグインにのみ適用されます。現在のファイルベースのCRL プラグインは **file** および **pkcs11** キーストアです。**nss** キーストアの場合、CRL の場所は常にNSS 内部データベースです。

crl-get-crl-uri=true | false

証明書の検証の一環として、証明書の配布ポイント拡張機能のURI 情報を使用してCRL ファイルを取得し、動的にキャッシュするかどうかを構成します。

この属性はデフォルトで **false** に設定されています。

crl-ignore-crl-date=true | false

crl-ignore-crl-date が **true** に設定されている場合、CRL の有効期間は検査されません。

この属性はデフォルトで **false** に設定されています。

crl-ignore-crl-sign=true | false

crl-ignore-crl-sign が **true** に設定されている場合、CRL の署名は検査されません。

この属性はデフォルトで **false** に設定されています。

crl-proxy=URL

crl-get-crl-uri が true に設定されている場合に、CRL ファイルを動的に取得するプロキシサーバーの名前とポートを設定します。

ポート番号は省略できます。ポート番号を指定しない場合、デフォルト値は 8080 です。crl-proxy の設定は、たとえば `crl-proxy=webcache.sfbay:8080` のようになります。

dbfile=dbfile

新しいポリシーを追加する DB ファイル。指定しない場合、デフォルトは、システム KMF ポリシーデータベースファイル `/etc/security/kmfpolicy.xml` になります。

ekuoids=EKUOIDS

定義しているポリシーで必要になる拡張鍵用途 OID のコンマ区切りのリスト。OID は、1.2.3.4 のようにドット表記で表されます。ekuoids の設定は、たとえば `ekuoids=1.2.3.4,9.8.7.6.5` のようになります。

ekunames=EKUNAMES

定義しているポリシーで必要になる拡張鍵用途名のコンマ区切りリスト。EKUNAMES に使用できる値は、`serverAuth`、`clientAuth`、`codeSigning`、`emailProtection`、`ipsecEndSystem`、`ipsecTunnel`、`ipsecUser`、`timeStamping`、および `OCSPSigning` です。

OCSP、CRL、鍵用途、および拡張鍵用途の検査はデフォルトで無効になっています。これらのいずれかを有効にするには、特定の検査に対して 1 つまたは複数の属性を指定します。たとえば、`ocsp-responder` 属性を設定すると、OCSP 検査が有効になります。ekuname 属性または ekuoids 属性を設定すると、拡張鍵用途検査が有効になります。

ignore-date=true | false

このポリシーの日付を無視オプションを設定します。デフォルトではこの値は false です。true を指定した場合、ポリシーは、有効性を評価するときに、証明書で定義されている有効期限を無視します。

ignore-unknown-eku=true | false

このポリシーの不明な EKU を無視オプションを設定します。デフォルトではこの値は false です。true の場合、ポリシーは、拡張鍵用途拡張機能の認識できない EKU 値を無視します。

ignore-trust-anchor=true | false

このポリシーのトラストアンカーを無視オプションを設定します。デフォルトではこの値は false です。true を指定した場合、ポリシーは、検証時にトラストアンカーを使用して検証対象の証明書の署名を確認しません。

`keyusage=KVALUES`

定義しているポリシーで必要になる鍵用途値のコンマ区切りリスト。使用できる値

は、`digitalSignature`、`nonRepudiation`、`keyEncipherment`、`dataEncipherment`、`keyAgreement`、`decipherOnly` です。

`ocsp-ignore-response-sign=true | false`

この属性を `true` に設定した場合、OCSP 応答の署名は検証されません。この属性値はデフォルトで `false` に設定されています。

`ocsp-proxy=URL`

OCSP のプロキシサーバー名およびポートを設定します。ポート番号は省略できます。ポート番号を指定しない場合、デフォルト値は 8080 です。`ocsp-proxy` の設定は、たとえば `ocsp-proxy="webcache.sfbay:8080"` のようになります。

`ocsp-response-lifetime=timelimit`

応答が必要になる *freshness* 期間を設定します。 *timelimit*

は、*number-day*、*number-hour*、*number-minute*、または *number-second* で指定できます。`ocsp-response-lifetime` の設定は、たとえば `ocsp-response-lifetime=6-hour` のようになります。

`ocsp-responder-cert-name=IssuerDN`

`ocsp-responder-cert-serial=serialNumber`

これらの 2 つの属性は、OCSP 応答者証明書を表しま

す。`ocsp-responder-cert-name` には、証明書の発行者名を指定します。例については、`ta-name` オプションを参照してください。`ocsp-responder-cert-serial` はシリアル番号用で、`0x0102030405060708090a0b0c0d0e0f` のように 16 進数値で指定する必要があります。OCSP 応答者が証明書の発行者とは異なる場合、および OCSP 応答を検証する必要がある場合は、OCSP 応答者の証明書情報を指定するようにしてください。

`ocsp-responder=URL`

OCSP 検証方法で使用する OCSP 応答者の URL を設定します。たとえ

ば、`ocsp-responder=http://ocsp.verisign.com/ocsp/status` のように設定します。

`ocsp-use-cert-responder=true | false`

可能な場合は証明書で定義された応答者を常に使用するように、このポリシーを構成します。

`policy=policyname`

作成されるポリシーレコード。 *policyname* が必要です。

`ta-name=trust anchor subject DN | search`

`ta-name` は証明書を検証するために使用されるトラストアンカーを識別します。KMF ポリシーエンジンは、完全な PKIX パス検証を行うのではなく、単に、検証対象の証明書の親であるようにトラストアンカーを扱います。

明示的な Subject DN が指定されている場合、`ta-serial` 値と組み合わせて、使用する証明書を一意に識別する必要があります。また、識別された証明書は、選択されているキーストアで使用できる必要があります。

明示的な主体名やシリアル番号の代わりに値 `search` を使用する場合、KMF ポリシーエンジンは、検証対象の証明書の発行者名に一致する証明書を見つけようとし、それを検証に使用します。

`search` を使用する場合、`ta-serial` 値は無視されます。

`ta-serial=trust anchor serial number`

`ta-name` を明示的な主体名として指定する場合、その証明書のシリアル番号を `ta-serial` 値で指定する必要があります。シリアル番号は、たとえば `ta-serial=0x01020a0b` のように、16 進形式で指定する必要があります。

`validity-adjusttime=adjusttime`

証明書の有効期限の開始と終了に対する調整時間を設定します。時間は `number-day`、`number-hour`、`number-minute`、または `number-second` で指定できます。`validity-adjusttime` の設定は、たとえば `validity-adjusttime=6-hour.ta-name="Subject DN" ta-serial=serialNumber` のようになります。

これら 2 つの属性は、トラストアンカー証明書を表し、キーストアでトラストアンカーの証明書を見つけるために使用されます。`ta-name` には、トラストアンカー証明書の主体名の識別名を指定します。たとえば、`ta-name="O=Sun Microsystems Inc., \ OU=Solaris Security Technologies Group, \ L=Ashburn, ST=VA, C=US, CN=John Smith"` TA 証明書のシリアル番号。これは、発行者 DN とともに、キーストアで TA 証明書を見つけるために使用されます。シリアル番号は、たとえば `0x0102030405060708090a0b0c0d0e` のように、16 進数値として指定する必要があります。`ignore-trust-anchor` 属性の値が `false` の場合、トラストアンカー属性を設定する必要があります。

`mapper-name=name`

`mapper-dir=directory`

`mapper-path=path`

`mapper-options=options`

これらの 4 つのオプションは、証明書と名前のマッピングをサポートします。`mapper-name` はマッパーの名前を指定します。たとえば、`cn` 名は、マッパーオブジェクト `kmf_mapper_cn.so.1` を表します。`mapper-dir` は、デフォルトのマッパーディレクトリ `/lib/crypto` をオーバーライドします。`mapper-path` は、マッパーオブジェクトの完全パスを指定します。`mapper-options` は、ASCII のみから構成される最大 255 バイト長の文字列です。その書式はマッパー固有ですが、マッパーは、たとえば `casesensitive`、`ignoredomain` のような、オプションのコンマ区切りリストを受け入れることが期待されます。`mapper-path` および `mapper-name` は相互に排他的です。`mapper-dir` は、`mapper-name` が設定されている場合にのみ設定できます。`mapper-options` は、`mapper-name` または `mapper-path` が設定されている場

合にのみ設定できます。上記の誤った設定のいずれかを使用しようとする
と、エラーになり、ポリシーデータベースは変更されません。

delete

指定されたポリシー名に一致したポリシーを削除します。システムのデフォルト
ポリシー (default) は削除できません。

delete サブコマンドの書式は次のとおりです。

```
delete [dbfile=dbfile] policy=policyname
```

delete サブコマンドは次のオプションをサポートします。

dbfile=dbfile 指定されたファイルからポリシー定義を読み込みま
す。dbfile が指定されていない場合、デフォルトは、シス
テム KMF ポリシーデータベースファイル
/etc/security/kmfpolicy.xml になります。

policy=policyname 削除するポリシーの名前。システムデータベースを使用す
る場合、policyname が必要です。

export

あるポリシーデータベースファイルから別のポリシーデータベースファイル
に、ポリシーをエクスポートします。

export サブコマンドの書式は次のとおりです。

```
kmfcfg export policy=policyname outfile=newdbfile [dbfile=dbfile]
```

export サブコマンドは次のオプションをサポートします。

dbfile=dbfile エクスポートしたポリシーが読み込まれる DB ファイ
ル。dbfile が指定されていない場合、デフォルトは、シス
テム KMF ポリシーデータベースファイル
/etc/security/kmfpolicy.xml になります。

outfile=outputdbfile エクスポートしたポリシーが保存される DB。

policy=policyname エクスポートされるポリシーレコード。

help

kmfcfg コマンドのヘルプを表示します。

help サブコマンドの書式は次のとおりです。

```
help
```

import

あるポリシーデータベースファイルから別のポリシーデータベースファイル
に、ポリシーをインポートします。

import サブコマンドの書式は次のとおりです。

```
kmfcfg import policy=policyname infile=inputdbfile [dbfile=dbfile]
```

import サブコマンドは次のオプションをサポートします。

`policy=policyname` インポートされるポリシーレコード。
`infile=inputdbfile` ポリシーの読み込み元の DB ファイル。
`dbfile=outdbfile` 新しいポリシーを追加する DB ファイル。指定しない場合、デフォルトは、システム KMF ポリシーデータベースファイル `/etc/security/kmfpolicy.xml` になります。

list

引数がない場合、デフォルトのシステムデータベースから、すべてのポリシー定義を一覧表示します。

list サブコマンドの書式は次のとおりです。

```
list [dbfile=dbfile] [policy=policyname]
```

list サブコマンドは次のオプションをサポートします。

`dbfile=dbfile` 指定されたファイルからポリシー定義を読み込みます。指定しない場合、デフォルトは、システム KMF ポリシーデータベースファイル `/etc/security/kmfpolicy.xml` になります。

`policy=policyname` 指定された名前に一致したポリシーのポリシー定義のみを表示します。

modify

指定された名前に一致したポリシーを変更します。システムのデフォルトポリシー (default) は変更できません。

modify サブコマンドの書式は次のとおりです。

```
modify [dbfile=dbfile] policy=policyname
    [ignore-date=true|false]
    [ignore-unknown-eku=true|false]
    [ignore-trust-anchor=true|false]
    [validity-adjusttime=adjusttime]
    [ta-name=trust anchor subject DN]
    [ta-serial=trust anchor serial number]
    [ocsp-responder=URL]
    [ocsp-proxy=URL]
    [ocsp-use-cert-responder=true|false]
    [ocsp-response-lifetime=timelimit]
    [ocsp-ignore-response-sign=true|false]
    [ocsp-responder-cert-name=Issuer DN]
    [ocsp-responder-cert-serial=serial number]
    [ocsp-none=true|false]
    [crl-basefilename=basefilename]
    [crl-directory=directory]
```



```

[crl-get-crl-uri=true|false]
[crl-proxy=URL]
[crl-ignore-crl-sign=true|false]
[crl-ignore-crl-date=true|false]
[crl-none=true|false]
[keyusage=digitalSignature| nonRepudiation
    |keyEncipherment | dataEncipherment |
    keyAgreement |keyCertSign |
    cRLSign | encipherOnly | decipherOnly],[...]
[keyusage-none=true|false]
[ekunames=serverAuth | clientAuth |
    codeSigning | emailProtection |
    ipsecEndSystem | ipsecTunnel |
    ipsecUser | timeStamping |
    OCSPSigning],[...]
[ekuoids=OID,OID,OID]
[eku-none=true|false]
[mapper-name=name of the mapper]
[mapper-dir=dir where mapper library resides]
[mapper-path=full pathname of mapper library]
[mapper-options=mapper options]

```

modify サブコマンドは、create サブコマンドと同じオプションを多数サポートします。共通のオプションの説明については、create サブコマンドを参照してください。

modify サブコマンドは次の固有のオプションをサポートします。

crl-none=true false	crl-none が true に設定されている場合、CRL 検査が無効になります。この属性が true に設定されている場合、ほかの CRL 属性は設定できません。
dfile=[<i>dbfile</i>]	ポリシーを変更するデータベースファイル。指定しない場合、デフォルトは、システム KMF ポリシーデータベースファイル /etc/security/kmfpolicy.xml になります。
eku-none=true false	eku-none が true に設定されている場合、拡張鍵用途検査が無効になります。eku-none が true に設定されている場合、拡張鍵用途属性 ekuname および ekuoids は同時に設定できません。
keyusage-none=true false	keyusage-none が true に設定されている場合、鍵用途検査が無効になります。 この属性が true に設定されている場合、keyusage 属性は同時に設定できません。

<code>ocsp-none=true false</code>	<code>ocsp-none</code> が <code>true</code> に設定されている場合、OCSP 検査が無効になります。この属性が <code>true</code> に設定されている場合、ほかの OCSP 属性は同時に設定されません。
<code>policy=policyname</code>	変更するポリシーの名前。 <i>policyname</i> が必要です。システム KMF ポリシーデータベースの <i>default</i> ポリシーは変更できません。
<code>mapper-name=name</code> <code>mapper-dir=directory</code> <code>mapper-path=path</code> <code>mapper-options=options</code>	詳細は、 <code>create</code> サブコマンドを参照してください。

プラグインサブコマンド

`install keystore=keystore_name modulepath=pathname` [`option=option_str`]
 プラグインをシステムにインストールします。`modulepath` フィールドには、KMF プラグイン共有ライブラリオブジェクトへのパス名を指定します。*pathname* が完全パス名として指定されない場合、共有ライブラリオブジェクトは、`/lib/security/$ISA/` を基準とした相対パスであるとみなされます。*ISA* トークンは、実装で定義されたディレクトリ名に置き換えられます。これは、呼び出し側プログラムの命令セットのアーキテクチャーを基準にした相対パス名を定義します。

`list plugin`
 KMF プラグイン情報を表示します。

`plugin` キーワードがない場合、「サブコマンド」セクションで説明したように、`kmfcfg list` はポリシー情報を表示します。

`modify plugin keystore=keystore_name option=option_str`
`plugin` オプションを変更します。`plugin` オプションはプラグインが定義し、プラグインが具体的に解釈するので、このコマンドは任意のオプション文字列を受け入れます。

`plugin` キーワードがない場合、「サブコマンド」セクションで説明したように、`kmfcfg modify` はポリシー構成を更新します。

`uninstall keystore=keystore_name`
keystore_name のプラグインをアンインストールします。

使用例

例1 新しいポリシーの作成

次の例は、IPSEC と呼ばれる新しいポリシーをシステムデータベース内に作成します。

```
$ kmfcfg create IPSEC \
  ignore-trust-anchor=true \
  ocsp-use-cert-responder=true \
```

例1 新しいポリシーの作成 (続き)

```
keyusage=keyAgreement,keyEncipherment,dataEncipherment \  
ekuname=ipsecTunnel,ipsecUser
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

ファイル /etc/security/kmfpolicy.xml デフォルトのシステムポリシーデータベース

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

関連項目 [attributes\(5\)](#)

名前	kpasswd – ユーザーの Kerberos パスワードの変更
形式	/usr/bin/kpasswd [<i>principal</i>]
機能説明	kpasswd コマンドは、Kerberos 主体のパスワードを変更するために使用されます。kpasswd は、現在の Kerberos パスワードの入力を求め、このパスワードは、ユーザーの Kerberos レルムの KDC から <code>changepw</code> チケットを取得するために使用されます。kpasswd が <code>changepw</code> チケットを正常に取得した場合、ユーザーは 2 回新しいパスワードを入力するよう求められ、それによりパスワードが変更されます。 主体が、新しいパスワードに必要な文字クラスの長さまたは数、あるいはその両方を指定するポリシーによって管理されている場合、新しいパスワードはそのポリシーに準拠している必要があります。(5 つの文字クラスは、小文字、大文字、数字、句読文字、およびその他のすべての文字です。)
オペランド	次のオペランドを指定できます。 <i>principal</i> Kerberos 主体 <i>principal</i> のパスワードを変更します。それ以外の場合、主体は、kpasswd コマンドを呼び出しているユーザーの識別情報から決定されます。
ファイル	/tmp/ovsec_adm.xxxxxx パスワード変更操作の間だけ有効な一時的な資格キャッシュ。(xxxxxx はランダム文字列です。)
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
CSI	有効

関連項目	kerberos(5)
使用上の留意点	kpasswd が中断された場合、 <code>changepw</code> チケットは破棄されない可能性があります。

名前	krb5-config – インストールされている Kerberos ライブラリに対するリンク
形式	krb5-config [<code>--all</code> <code>--cflags</code> <code>--exec-prefix</code> <code>--help</code> <code>--libs library</code> <code>--prefix</code> <code>--vendor</code> <code>--version</code>]
機能説明	krb5-config は、インストールされている Kerberos ライブラリに対してプログラムをコンパイルおよびリンクするために必要な特殊なフラグを識別して表示します。
オプション	サポートしているオプションは、次のとおりです。 <code>--all</code> バージョン、ベンダー、接頭辞、および <code>exec-prefix</code> を表示します。 <code>--cflags</code> Kerberos が構築されたときのコンパイラフラグを表示します。 <code>--exec-prefix</code> Kerberos が構築されたときの <code>exec-prefix</code> を表示します。 <code>--help</code> 使用方法に関するメッセージを表示します。 これはデフォルトです。 <code>--libs library</code> <i>library</i> とリンクするために必要なコンパイラオプションを表示します。 次の <i>library</i> 値がサポートされています。 <code>krb5</code> Kerberos 5 アプリケーション <code>--prefix</code> Kerberos が構築されたときの接頭辞を表示します。 <code>--vendor</code> インストールされている Kerberos 実装のベンダーを表示します。 <code>--version</code> インストールされている Kerberos 実装のバージョンを表示します。
使用例	例1 <code>--cflags</code> オプションの使用 次の例では、libkrb5(3LIB) を使用するために必要な C コンパイラフラグを表示します。 <pre>% krb5-config --cflags -I/usr/include/kerberosv5</pre> 例2 <code>--libs</code> オプションの使用 次の例では、libkrb5(3LIB) に対してリンクするために必要な C コンパイラオプションを示します。 <pre>% krb5-config --libs -L/usr/lib -R/usr/lib -lkrb5</pre>

- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
インタフェースの安定性	流動的

関連項目

[libgss\(3LIB\)](#), [libkrb5\(3LIB\)](#), [attributes\(5\)](#)

名前	ksh, ksh93, rksh – Korn シェル。標準および制限付きコマンドとプログラミング言語
形式	<pre>ksh [±abcefghikmnoprstuvxBCD] [-R file] [±o option] ... [-] [arg ...] rksh [±abcefghikmnoprstuvxBCD] [-R file] [±o option] ... [-] [arg ...]</pre>
機能説明	ksh は、端末またはファイルから読み取られたコマンドを実行するコマンドおよびプログラミング言語です。rksh は、コマンドインタプリタである ksh の制限付きバージョンです。rksh は、標準シェルより機能が制限されたログイン名や実行環境を設定するために使用します。 シェルに対する引数の意味については、「呼び出し」を参照してください。
定義	メタキャラクタ (<i>metacharacter</i>) は、次のいずれかの文字として定義されます。 <pre>; & () < > NEWLINE SPACE TAB</pre> ブランク (<i>blank</i>) は、タブ (TAB) またはスペース文字 (SPACE) のことです。 識別子 (<i>identifier</i>) は英文字、数字、または下線の並びで、先頭文字は英文字または下線です。識別子は変数名のコンポーネントとして使用されます。 <i>vname</i> は、1 つの識別子、またはピリオド (.) で区切られた複数の識別子の並びです。前にピリオド (.) を付けることもできます。<i>vnames</i> は関数名および変数名として使用されます。 ワード (<i>word</i>) は、引用符なしのメタキャラクタを除く、現在のロケールによって定義された文字セットの文字の並びのことです。 コマンド (<i>command</i>) は、シェル言語の構文にそった文字の並びのことです。シェルは各コマンドを読み取り、指定された動作を直接実行するか、または動作を実行する別のユーティリティを起動します。組み込みコマンドとは、別のプロセスを作成せずにシェル自体が実行するコマンドです。一部のコマンドは、便宜のためだけに組み込まれているので、このマニュアルページでは説明しません。シェル環境で副作用が発生する組み込みコマンドと、パス検索の実行前に見つかる組み込みコマンド(「実行」を参照)については、このマニュアルページで説明します。歴史的な理由から、これらの組み込みコマンドの中には、ほかの組み込みコマンドと動作が異なり、特殊組み込みコマンドと呼ばれるものがあります。
コマンド	単純コマンド (<i>simple-command</i>) は、変数代入のリストです(「変数代入」を参照)。つまり、ブランクで区切られたワードの並びであり、変数代入のリストが前に置かれることもあります。このマニュアルページの「環境」セクションを参照してください。 先頭のワードは、実行すべきコマンドの名前を指定します。残りのワードは、このセクションで述べる場合を除き、呼び出されたコマンドに引数として渡されます。コマンド名は引数 0 として渡されます。exec(2) を参照してください。単純コマ

ンドの値 (*value*) は、コマンドの終了ステータスです。正常終了した場合、この値は 0 - 255 です。異常終了した場合は 256+*signal* です。終了ステータスに対応するシグナルの名前は、組み込みの kill ユーティリティーの -l オプションを使用すると取得できます。

パイプライン (*pipeline*) は、パイプ (|) で区切られた 1 つ以上のコマンドの並びです。最後のコマンドを除き、各コマンドの標準出力は pipe(2) によってその次のコマンドの標準入力と結合されます。最後とみなされるコマンドを除き、各コマンドは別々のプロセスとして実行されます。シェルは最後のコマンドが終了するのを待ちます。パイプラインの終了ステータスは、pipefail オプションが有効になっていないかぎり、最後のコマンドの終了ステータスになります。各パイプラインの前に予約語 ! を指定できます。この場合、パイプラインの終了ステータスは、最後のコマンドの終了ステータスが 0 以外であれば 0 に、0 であれば 1 になります。

リスト (*list*) は、1 つのパイプライン、または ;、&、|&、&&、または | で区切られた複数のパイプラインの並びです。終わりに、;、&、または |& を記述することもできます。これら 5 つの記号の中で、;、&、および |& の優先度は同じで、&& と || の優先度よりも低くなります。&& と || の優先度は同じです。

セミコロン (;) によって、直前のパイプラインが順次実行されます。アンパサンド記号 (&) によって、直前のパイプラインが非同期的に実行されます。つまりシェルはパイプラインが終了するのを待ちません。|& という記号によって、親シェルに対して双方向パイプが確立された直前のパイプラインが非同期的に実行されます。リダイレクト演算子である <& および >& を引数 p とともにコマンドに適用し、組み込みコマンドである read および print の -p オプションを使用すると、生成されたパイプラインの標準入出力を親シェルに書き込んだり、親シェルから読み取ったりできます。&& (||) という記号は、直前のパイプラインの戻り値が 0 の値 (0 以外の値) の場合にのみ、後続のリストを実行します。コマンドの区切りとして、セミコロンの代わりに 1 つ以上の復帰改行をリストに指定できます。リダイレクションで始まっておらず、while、until、または if *list* 内で実行されていない単純コマンドであるリストの最初のパイプラインの最初の項目の前に、セミコロンを指定できます。set 組み込みコマンドでの説明のように、showme オプションが有効になっていないかぎり、このセミコロンは無視されます。

コマンド (*command*) は、単純コマンドまたは次の一覧に示すコマンドのいずれかです。特に断わりのない限り、コマンドが返す値は、そのコマンド中で最後に実行された単純コマンドの値です。

```
for vname [ in word ... ] ;do list ;done
```

for コマンドが実行されるたびに、*vname* は、in *word* リストから次に得られる *word* に設定されます。in *word* ... を省略すると、for コマンドは、1 から開始するように設定された各定位置パラメータに対して do *list* を 1 回実行します。リストの *word* がなくなると、実行は終了します。「パラメータ展開」を参照してください。


```
(( [expr1] ; [expr2] ; [expr3] )) ;do list ;done
```

算術式 *expr1* が最初に評価されます。算術式 *expr2* は評価の結果が 0 になるまで繰り返し評価され、評価の結果が 0 以外であれば *list* が実行され、算術式 *expr3* が評価されます。いずれかの式を省略した場合、その式の評価が 1 になる場合と同様に処理されます。「算術評価」を参照してください。

```
select vname [ in word ... ] ;do list ;done
```

select コマンドは、標準エラー (ファイル記述子 2) に、一群のワードをそれぞれの前に番号を付けて出力します。*in word...* を省略すると、1 から始まる定位置パラメータが使用されます。「パラメータ展開」を参照してください。PS3 プロンプトが出力され、標準入力から行が読み取られます。この行が、リストに示された *word* のいずれかの番号からなる場合、*vname* が示す変数の値はこの番号に該当する *word* に設定されます。この行が空の場合は、再度選択リストを出力します。それ以外の場合は、*vname* 変数の値を NULL に設定します。標準入力から読み取られた行の内容は、REPLY 変数に保存されます。*break* またはファイルの終わり (EOF) に行き当たるまで、選択が発生するたびに *list* が実行されます。*list* の実行によって REPLY 変数が NULL に設定されると、次の選択を求める PS3 プロンプトの表示前に選択リストが出力されます。

```
case word in [ ( ! pattern [ | pattern ] ... ) list ;; ] ... esac
```

case コマンドは、*word* に一致する最初の *pattern* に対応した *list* を実行します。*pattern* の形式は、ファイル名生成に使用される形式と同じです。「ファイル名の生成」を参照してください。

;; 演算子を使用すると、*case* の実行が終了します。;; の代わりに ;& を使用した場合、後続のリストが存在するときは、そのリストが実行されます。

```
if list ;then list [ ;elif list ;then list ] ... [ ;else list ] ;fi
```

if のあとの *list* を実行後、0 の終了ステータスが返されると、最初の *then* のあとの *list* が実行されます。それ以外の場合、*elif* のあとの *list* が実行されます。この値が 0 の場合、次の *then* のあとの *list* が実行されます。連続した *elif list* がすべて失敗すると、*else list* が実行されます。*if list* が 0 以外の終了ステータスを持ち、*else list* が存在しない場合、*if* コマンドは 0 の終了ステータスを返します。

```
while list ;do list ;done
```

```
until list ;do list ;done
```

while コマンドは、*while list* を繰り返し実行し、リスト内の最後のコマンドの終了ステータスが 0 の場合、*do list* を実行します。それ以外の場合、ループは終了します。*do list* 中のコマンドを実行しない場合、*while* コマンドは 0 の終了ステータスを返します。ループ終了条件の判定を逆にするには、*while* の代わりに *until* を使用します。

```
(( (expression))
```

expression は、このマニュアルページで説明する算術評価の規則を使用して評価されます。算術式の値が 0 以外の場合、終了ステータスは 0 です。それ以外の場合、終了ステータスは 1 です。

(*list*;)

別の環境で *list* を実行します。入れ子で2つの開いた括弧を連続して記述する場合、このセクションで説明するように、算術コマンドとして評価されないようにスペースを挿入する必要があります。

単に *list* を実行します。メタキャラクタの (および) とは異なり、{ および } は予約語なので、認識されるには行の始めまたは ; のあとに現れる必要があります。

[[*expression*]]

expression を評価し、*expression* が真のときに 0 の終了ステータスを返します。*expression* については、「条件式」を参照してください。

function *varname* { *list* ; }

varname () { *list* ; }

varname が参照する関数を定義します。*varname* に . が含まれる関数はディシプリン関数と呼ばれ、*varname* のうち最後の . の前にある部分は既存の変数を参照している必要があります。

{ と } の間のコマンド群 (*list*) が関数の本体になります。function *varname* の構文で定義された関数を、. という特殊組み込みコマンドの引数として使用すると、その定義に *varname*() の構文を使用している場合と同等の動作をさせることができます。「関数」を参照してください。

time [*pipeline*]

pipeline を省略すると、現在のシェルおよび完了した子プロセスのユーザーおよびシステム時間が標準エラーに出力されます。それ以外の場合は、*pipeline* が実行され、経過時間、ユーザー時間、およびシステム時間が標準エラーに出力されます。TIMEFORMAT 変数は、時間情報の表示方法を指定する書式文字列に設定できます。TIMEFORMAT 変数についての詳細は、「シェル変数」を参照してください。

次の予約語は、コマンドの最初のワードであり、かつ引用符を付けずに記述されたときにのみ、予約語と認識されます。

```
case
do
done
else
elif
esac
for
fi
function
if
select
then
time
```

```
until
while
{ }
[[ ]]
!
```

変数代入

1つまたは複数の変数代入は、単純コマンドを開始すること
も、`typeset`、`export`、または `readonly` の特殊組み込みコマンドの引数として使用
することもできます。代入の構文は次の形式になります。

```
varname=word
varname[word]=word
```

`varname` と `=` の間、`=` と `word` の間にスペースを入れることはできません。

```
varname=(assignlist)
```

`varname` と `=` の間にスペースを入れることはできません。`assignlist` は次のいずれ
かにできます。

```
word...
```

インデックス付き配列の代入。

```
[word]=word...
```

連想配列の代入。`typeset -a` を前に付けると、代わりにインデックス付き配列
が作成されます。

```
assignment...
```

複合変数代入。これは、`varname.name` という形式のサブ変数を含んだ複合変
数 `varname` を作成します。ここで `name` は代入の名前の部分です。`varname` の
値にはすべての代入要素が含まれます。`varname` のサブ変数に行われた追加の
代入は、`varname` の値の一部としても表示されます。`assignment` を指定しない
場合、`varname` はサブシーケンスの子要素を定義できる複合変数になります。

```
typeset [options] assignment ...
```

入れ子になった変数代入。複数の代入を指定するには、`;` でそれぞれを区切り
ます。前の値は、代入前に設定解除されます。

また、前の値への加算または付加を指定するために、`=` の代わりに `+=` を使用で
きます。`+=` を算術タイプに適用すると、`word` は算術式として評価され、現在の値
に加算されます。文字列変数に適用すると、`word` で定義された値が、この値に
付加されます。複合代入の場合、前の値は設定解除されず、型に互換性があれ
ば、新しい値が現在の値に付加されます。

Comments

`#` で始まるワードがある場合、そのワードと、これ以降の復帰改行までの文字がす
べてコメントになり、無視されます。

別名

各コマンドの最初のワードに別名が定義されている場合、そのワードは別名のテキ
ストに置き換えられます。別名は任意の数の文字で構成されます。別名に使用で
きない文字は、メタキャラクタ、引用符、ファイル展開文字、パラメータ展開文

字、コマンド置換文字、および=です。置換文字列としては、「コマンド」セクションで示したメタキャラクタを含む有効なシェルスクリプトを指定できます。置換されたテキスト内にある各コマンドの最初のワードは、置換対象のものを除き、別名についてチェックされます。別名の最後の文字がブランクの場合、別名のあとのワードも別名置換についてチェックされます。

別名を使用すると、組み込みコマンドを再定義できますが、「コマンド」セクションで示した予約語は再定義できません。別名は `alias` コマンドで作成および一覧表示でき、`unalias` コマンドで削除できます。

別名化はスクリプトの読み取り時に実行されますが、スクリプトの実行中には行われません。別名を有効にするには、別名を参照するコマンドの読み取り前に、`alias` コマンドで別名を定義しておく必要があります。次に示す別名は、シェルにコンパイルされますが、設定解除または再定義が可能です。

```
autoload='typeset -fu'
command='command '
fc=hist
float='typeset -lE'
functions='typeset -f'
hash='alias -t --'
history='hist -l'
integer='typeset -li'
nameref='typeset -n'
nohup='nohup '
r='hist -s'
redirect='command exec'
source='command .'
stop='kill -s STOP'
suspend='kill -s STOP $$'
times='{ { time;} 2>&1;}'
type='whence -v'
```

チルド置換

別名置換を実行すると、各ワードが引用符なしのチルド(~)で始まっているかどうかチェックされます。チルド置換の場合、ワードは、パラメータ展開のワード部分も指します。「パラメータ展開」を参照してください。

この場合には、/までのワードについて、パスワードデータベースのユーザー名と一致するかどうかチェックされます。一致した場合は、~および一致したログイン名が一致したユーザーのログインディレクトリに置き換えられます。一致するユーザー名が見つからない場合、元のテキストは変更されません。~が単独で指定された場合や、後ろに/を伴って指定された場合は、`$HOME`に置き換えられます。~のあとに+または-を指定すると、それぞれ `$PWD` または `$OLDPWD` の値に置き換えられます。

さらに、変数代入を展開する場合で、代入する値が~で始まる時、および~がコロン(:)のあとに置かれているときに、チルド置換が試行されます。:は~ログイン名の終了にも使用されます。

コマンド置換 ドル記号 (\$) を前に付けた括弧か、一対の逆引用符 (") でコマンドを囲むと、その標準出力をワードの一部または全体として使用できます。末尾の復帰改行は削除されます。2 番目の (廃止された) 形式では、コマンドの実行前に、引用符間の文字列に含まれる特殊引用符文字が処理されます。「クォート」を参照してください。

`$(cat file)` というコマンド置換は、同じ動作で実行速度の速い `$(<file)` に置き換えることができます。`$(n<#)` というコマンド置換は、ファイル記述子 *n* の現在のバイトオフセットに展開されます。

算術置換 ドル記号を前に付けた二重括弧で囲まれた算術式 (つまり `$( ( (arithmetic_expression) ) )` の形式) は、二重括弧内の算術式の値に置き換えられます。

プロセス置換 プロセス置換は、UNIX オペレーティングシステムのバージョンが、開いたファイルに名前を付けるための `/dev/fd` ディレクトリをサポートしている場合にだけ使用できます。

`<(list)` または `>(list)` 形式の各コマンド引数は、*list* が示すプロセスを実行します。このプロセスは、`/dev/fd` 中のファイルに非同期的に接続されています。このファイルの名前がコマンドの引数になります。`>` が付いた形式を使用した場合は、このファイルに書き出すことで *list* に入力できます。`<` の形式を使用した場合は、引数として渡されたファイルに *list* プロセスからの出力が含まれます。

たとえば、

```
paste <(cut -f1 file1) <(cut -f3 file2) | tee \
    >(process1) >(process2)
```

これは *file1* からフィールド 1 をカットし、*file2* からフィールド 3 をカットし、その両者をペーストし、その結果を *process1* と *process2* に送ります。さらに標準出力に結果を表示します。引数としてコマンドに渡されるファイルは、UNIX の `pipe(2)` です。ファイルに対し `lseek(2)` を行えると想定しているプログラムは動作しません。

パラメータ展開 パラメータは、変数、1 つまたは複数の数字、または `*`、`@`、`#`、`?`、`-`、`$`、`!` のいずれかの文字です。変数は *vname* で指定されます。*vname* に `.` が含まれている変数を作成するには、最後の `.` の前にあるすべての文字から構成される *vname* 変数が、すでに存在している必要があります。変数には 1 つの値と、0 個以上の属性が割り当てられます。変数に値と属性を割り当てるには、`typeset` 特殊組み込みコマンドを使用します。シェルがサポートする属性については、`typeset` 特殊組み込みコマンドの節で後述します。エクスポートされた変数によって、値と属性が環境に渡されます。

シェルは、インデックス付き配列と連想配列の両方をサポートします。配列変数の要素は、添字 (subscript) によって参照されます。インデックス付き配列の添字は、`[` と `]` で囲まれた算術式で指定されます (「算術評価」を参照してください)。インデックス付き配列に値を代入するには、`set -A vname value ...` を使用します。添字の

値は必ず 0 から 1,048,575 の範囲内で指定します。インデックス付き配列は宣言する必要はありません。有効な添字を伴う変数の参照は正当であり、必要に応じて配列が作成されます。

連想配列は、typeset の -A オプションを使用して作成します。連想配列の添字は、[と] で囲まれた文字列で指定されます。

添字なしで配列を参照すると、0 の添字で配列を参照したときと同等の結果になります。

次によって変数に値を代入することもできます。

```
vname=value [vname=value] ...
```

または

```
vname[subscript]=value [vname[subscript]=value] ...
```

= の前後にスペースを入れることはできません。nameref は、別の変数を参照する変数です。nameref は、typeset の -n 属性を使用して作成します。nameref 変数が使用されたときは常に typeset コマンドの実行時点での変数の値が参照されます。nameref の名前にドット (.) を含めることはできません。変数名または関数名に . が含まれ、名前の最初の . までの部分が nameref の名前と一致する場合、nameref の部分を nameref が参照している変数の名前に置き換えることによって、参照される変数が取得されます。nameref を for ループのインデックスとして使用すると、リスト内の項目ごとに名前参照が確立されます。nameref を使用すると、関数への引数として名前が渡される変数を、関数内部で簡単に参照できます。たとえば、変数の名前が最初の引数として関数に渡される場合、その関数内のコマンドとして

```
typeset -n var=$1
```

を実行すると、var への参照および代入が、その関数に名前が渡される変数の参照および代入になります。浮動小数点属性 (-E または -F) または整数属性 (-i) のどちらかが vname に対して設定されている場合、このマニュアルページで説明するように、value は算術評価の対象になります。定位置パラメータは数値で設定されるパラメータで、set 特殊組み込みコマンドで値を代入できます。パラメータ \$0 は、シェルが呼び出されたときに、引数 0 から設定されます。文字 \$ は、置換可能なパラメータを示します。

```
${parameter}
```

シェルは、中括弧やメタキャラクタが含まれていても、\${ からこれに対応する } までのすべての文字を、同じワードの一部として読み取ります。パラメータの値 (もしあれば) に置き換えられます。中括弧が必要となるのは、parameter のあとに、その名前の一部として解釈してはならない文字、数字、または下線が続く場合、変数名に . が含まれる場合、または変数に添字が付いている場合です。1 つ

	または複数の数字からなるパラメータは、定位置パラメータになります。複数の数字からなる定位置パラメータは中括弧で囲む必要があります。<i>parameter</i> が * または @ のときは、\$1 から始まるすべての定位置パラメータに置き換えられ、フィールド区切り文字で区切られます。配列の <i>vname</i> に添字として * または @ を使用すると、各要素の値に置き換えられ、IFS の値の最初の文字で区切られます。
<code>\${#parameter}</code>	<i>parameter</i> が * または @ のときは、定位置パラメータの数に置き換えられます。それ以外の場合は、<i>parameter</i> の値の長さに置き換えられます。
<code>\${#vname[*]}</code> <code>\${#vname[@]}</code>	配列 <i>vname</i> 内の要素の数に置き換えられます。
<code>\${!vname}</code>	<i>vname</i> で参照される変数の名前に展開されます。<i>vname</i> が名前参照である場合を除いて、これは <i>vname</i> です。
<code>\${!vname[subscript]}</code>	<i>subscript</i> が * または @ でなければ、添字の名前に展開されます。<i>subscript</i> が * の場合は、<i>vname</i> の配列添字のリストが生成されます。配列でない変数の場合、変数が設定されていれば値は 0 になります。それ以外の場合は NULL です。<i>subscript</i> が @ の場合、<code>\${vname[*]}</code> と同じになります。ただし、二重引用符で囲んで使用した場合に、それぞれの配列の添字が別々の引数を生成します。
<code>\${!prefix*}</code>	名前が <i>prefix</i> で始まる変数の名前に展開されます。
<code>\${parameter:-word}</code>	<i>parameter</i> が設定されており、NULL でない場合は、その値に置き換えられます。それ以外の場合は、<i>word</i> に置き換えられます。 <i>word</i> は、置換された文字列として使用されるまで評価されません。 次の例では、<i>d</i> が設定されていないか、NULL である場合にのみ、<i>pwd</i> が実行されます。 <pre>print \${d:-\$(pwd)}</pre> 式からコロン(:)を省略すると、シェルは <i>parameter</i> が設定されているかどうかだけをチェックします。
<code>\${parameter:offset:length}</code> <code>\${parameter:offset}</code>	オフセットを算術式として展開することにより決定される (0 を基準とした) 文字から始まり、<i>length</i> に

よって定義された算術式によって決定される文字数から構成される *parameter* の値の部分に展開されます。

2 番目の形式では、値の残りの部分を使用されます。負のオフセットの場合は、*parameter* の最後から逆方向にカウントします。

シェルが演算子を `:-` と解釈しないように、1 つまたは複数のブランクをマイナスの記号の前に指定する必要があります。パラメータが `*` または `@` である場合や、`*` または `@` のインデックスが付けられた配列名である場合、*offset* と *length* はそれぞれ配列インデックスと要素数を指します。負の *offset* は、インデックス付き配列の最大の添字より 1 大きい数値を基準に計算されます。連想配列の順序は不特定です。

```
${parameter#pattern}
${parameter##pattern}
```

シェルの *pattern* が *parameter* の値の先頭に一致した場合、この展開式の値は、*parameter* の値から一致した部分を削除したものになります。それ以外の場合は、この *parameter* の値に置き換えられます。1 番目の形式では、一致した最小の *pattern* が削除され、2 番目の形式では、一致した最大の *pattern* が削除されます。*parameter* が `@` または `*` である場合や、`@` または `*` の添字が指定された配列変数である場合は、部分文字列操作が個々の要素に順番に適用されます。

```
${parameter%pattern}
${parameter%%pattern}
```

シェルの *pattern* が *parameter* の値の末尾に一致した場合、この展開式の値は、*parameter* の値から一致した部分を削除したものになります。それ以外の場合は、*parameter* の値に置き換えられます。1 番目の形式では、一致した最小の *pattern* が削除され、2 番目の形式では、一致した最大の *pattern* が削除されます。*parameter* が `@` または `*` である場合や、`@` または `*` の添字が指定された配列変数である場合は、部分文字列操作が個々の要素に順番に適用されます。

```
${parameter/pattern/string}
${parameter//pattern/string}
${parameter/#pattern/string}
```


`${parameter/%pattern/string}` *parameter*を展開し、一致した最長の *pattern* を、指定した *string* に置き換えます。*string* に `\n` が現れるごとに、*parameter* 中の *n* 番目のサブパターンに一致する部分に置き換えられます。

string が NULL の場合、*pattern* は削除され、*string* の先頭の `/` を省略できます。*parameter* が `@` または `*` である場合や、`@` または `*` の添字が指定された配列変数である場合は、置換操作が個々の要素に順番に適用されます。この場合、*word* の *string* 部分が要素ごとに繰り返し評価されます。

1 番目の形式では、最初に現れた *pattern* だけが置き換えられます。

2 番目の形式では、*pattern* に一致するごとに、指定した *string* に置き換えられます。

3 番目の形式では、パターンの一致は *string* の先頭に制限されます。

4 番目の形式では、パターンの一致は *string* の末尾に制限されます。

次のパラメータはシェルが自動的に設定します。

#	定位置パラメータ数 (10 進数)
-	呼び出し時、または <code>set</code> コマンドを使用してシェルに適用されるオプション。
?	最後に実行されたコマンドが返した 10 進数。
\$	このシェルのプロセス番号。
_	<code>_</code> の値は、最初は環境に引き渡される実行中のシェルまたはスクリプトの絶対パス名です。その後、直前のコマンドの最後の引数が代入されます。
	このパラメータは、非同期式のコマンドに関しては設定されません。このパラメータは、メールのチェック時にも、一致する MAIL ファイルの名前を保持するために使用されます。
!	最後に呼び出されたバックグラウンドコマンドのプロセス番号、または <code>bg</code> 組み込みコマンドでバックグラウンドで実行された最新のジョブのプロセス番号。
.sh.command	DEBUG トラップを処理するときに、この変数には実行しようとしている現在のコマンド行が含まれます。

<code>.sh.edchar</code>	この変数には、KEYBD トラップを処理するときに入力されたキーボード文字 (または、最初の文字が ESC、つまり ASCII 033 である場合には文字の並び) の値が含まれます。トラップアクションの一環として値が変更された場合は、そのトラップを引き起こしたキー (またはキーシーケンス) が新しい値に置き換えられます。このマニュアルページの「キーバインド」を参照してください。
<code>.sh.edcol</code>	最新の KEYBD トラップの時点でのカーソルの文字位置。
<code>.sh.edmode</code>	vi 挿入モード中に KEYBD トラップを処理するときには、値は ESC に設定されます。それ以外の場合は、 <code>.sh.edmode</code> は KEYBD トラップを処理するとき NULL になります。このマニュアルページの「vi 編集モード」セクションを参照してください。
<code>.sh.edtext</code>	直前の KEYBD トラップの時点における入力バッファーに含まれていた文字。KEYBD トラップを処理していない場合は、値は NULL です。
<code>.sh.file</code>	現在のコマンドを含むファイルのパス名。
<code>.sh.fun</code>	実行されている現在の関数の名前。
<code>.sh.match</code>	一致した条件付きパターンマッチングのあと、および #、%、または / の演算子を使用した変数展開のあとに、最新の一致およびサブパターン一致を保存するインデックス付き配列。0 番目の要素には完全一致を格納し、 <i>i</i> 番目の要素には <i>i</i> 番目の部分一致を格納します。展開された変数に新しい値が代入されると、 <code>.sh.match</code> 変数は設定解除されます。
<code>.sh.name</code>	ディシプリン関数が呼び出された時点での変数の名前に設定されます。
<code>.sh.subscript</code>	ディシプリン関数が呼び出された時点での変数の名前添字に設定されます。
<code>.sh.subshell</code>	サブシェルおよびコマンド置換の現在の深さ。
<code>.sh.value</code>	set または append ディシプリン関数が呼び出された時点での変数の値に設定されます。
<code>.sh.version</code>	このシェルのバージョンを識別する値に設定されます。
<code>LINENO</code>	実行しているスクリプトまたは関数内での現在の行番号。
<code>OLDPWD</code>	cd コマンドで設定された直前の作業用ディレクトリ。
<code>OPTARG</code>	getopts 組み込みコマンドで処理された最後のオプション引数の値。

OPTIND	getopts 組み込みコマンドで処理された最後のオプション引数のインデックス。
PPID	シェルの親のプロセス番号。
PWD	cd コマンドで設定された現在の作業用ディレクトリ。
RANDOM	この変数が参照されるたびに、0 から 32767 の間で均一に分散した乱整数を生成します。RANDOM に数値を代入すると、乱数の並びを初期化できます。
REPLY	この変数は、引数が指定されない場合、select 文および read 組み込みコマンドによって設定されます。
SECONDS	この変数を参照するたびに、シェルを起動してからの秒数が返されます。この変数に値を代入すると、その値と代入処理実行時からの秒数との合計値が参照時に返されます。

次の変数はシェルによって使用されます。

CDPATH

cd コマンドの検索パスを定義します。

COLUMNS

シェル編集モードおよび選択リスト出力用の編集ウィンドウの幅を定義します。

EDITOR

VISUAL 変数が設定されていない場合、VISUAL で説明するように、この変数の値のパターンがチェックされ、対応する編集オプションが有効になります。

このマニュアルページの「特殊コマンド」セクションの set コマンドを参照してください。

ENV

値に対してパラメータ展開、コマンド置換、および算術置換を実行して、シェルが呼び出されたときに実行されるスクリプトのパス名を生成します。このファイルは通常、別名および関数の定義に使用されます。デフォルト値は \$HOME/.kshrc です。

このマニュアルページの「呼び出し」セクションを参照してください。

ENV はシェルでは設定されません。

FCEDIT

hist コマンドのデフォルトのエディタ名を表す廃止済みの名前。HISTEDIT が設定されている場合、FCEDIT は使用されません。

シェルではデフォルト値を FCEDIT に指定しています。

FIGNORE

ファイル名の照合を実行するときに無視される一連のファイル名を定義するパターン。

FPATH

関数定義の検索パス。-u 属性を持つ関数が参照される場合や、コマンドが見つからなかった場合に、このパス内のディレクトリに対して、関数またはコマンドと同じ名前のファイルを検索します。このコマンドの名前の実行ファイルが見つかると、現在の環境に読み込まれ実行されます。PATH とは異なり、現在のディレクトリは、隣接するコロン(:) 文字や先頭または末尾のコロン(:) ではなく、ドット(.) で明示的に表す必要があります。

HISTCMD

履歴ファイル内の現在のコマンドの番号。

HISTEDIT

hist コマンドのデフォルトのエディタ名に使用する名前。

HISTFILE

シェルの起動時にこの変数が設定されていると、その値はコマンド履歴の格納に使用されるファイルのパス名になります。このマニュアルページの「コマンド行の再入力」セクションを参照してください。

HISTSIZE

シェルの起動時にこの変数が設定されていると、このシェルで使用可能な入力済みコマンドの数がこの値以上になります。デフォルト値は512です。

HOME

cd コマンドのデフォルト引数(ホームディレクトリ)。

HOME はシェルでは設定されません。HOME は [login\(1\)](#) で設定されます。

IFS

内部フィールド区切り文字。通常は、スペース文字、タブ、および復帰改行です。コマンド置換またはパラメータ展開の結果を区切る場合および組み込みコマンドの read でフィールドを区切る場合に使用します。"\$*" 置換で引数を区切るときには、IFS 変数の最初の文字が使用されます。このマニュアルページの「クォート」セクションを参照してください。

isspace 文字クラスに属していない IFS 文字、および ispace 文字クラスに属する隣接文字で IFS に含まれているものが、分割対象の文字列に現れるごとに、フィールドが区切られます。isspace 文字クラスに属する文字で IFS に含まれる1つまたは複数の文字によって、フィールドが区切られます。また、同じ ispace 文字が IFS の中で並んでいる場合、この文字は ispace クラスに属していない文字として扱われます。つまり、IFS が2つのタブ文字から構成されている場合、2つの隣接するタブ文字によって NULL フィールドが区切られます。

シェルではデフォルト値を IFS に設定しています。

LANG

この変数は、LC_ または LANG で始まる変数で明確に選択されていないすべてのロケールカテゴリを指定します。

LC_ALL

この変数は、LANG 変数とほかのすべての LC_ 変数の値をオーバーライドします。

LC_COLLATE

この変数は、文字照合情報のロケールカテゴリを指定します。

LC_CTYPE

この変数は、文字処理関数のロケールカテゴリを指定します。これは、パターンマッチング用の文字クラスを指定します。このマニュアルページの「ファイル名の生成」セクションを参照してください。

LC_NUMERIC

この変数は、小数点文字のロケールカテゴリを指定します。

LINES

この変数を設定すると、その値は選択 (select) リスト出力用のカラム長の決定に使用されます。選択リストは、LINES が示す行数の約 3 分の 2 がいっぱいになるまで垂直に出力されます。

MAIL

この変数にメールファイルの名前が設定されていて、MAILPATH 変数が設定されていない場合、シェルは指定されたファイルにメールが到着するとユーザーに通知します。

MAIL はシェルでは設定されません。一部のシステムでは、MAIL は [login\(1\)](#) で設定されます。

MAILCHECK

MAILPATH 変数または MAIL 変数で指定されたファイルの更新時間の変更をシェルが何秒ごとにチェックするかを指定します。デフォルト値は **600** 秒です。この時間が経過すると、シェルは次のプロンプトを出力する前にチェックします。

シェルではデフォルト値を MAILCHECK に指定しています。

MAILPATH

コロン (:) で区切ったファイル名のリスト。この変数を設定すると、シェルは直前の MAILCHECK 秒間に発生した指定ファイルに対する変更をユーザーに通知します。各ファイル名のあとには ? と、出力されるメッセージを指定できます。このメッセージには、変更されたファイルの名前に変数 \$_ を定義して、パラメータ展開、コマンド置換、および算術置換が行われます。デフォルトのメッセージは `you have mail in $_` です。

PATH

コマンド用の検索パス。profile の場合を除き、rksh の環境下で実行する場合、ユーザーは PATH を変更できません。このマニュアルページの「実行」セクションを参照してください。

シェルではデフォルト値を `PATH` に設定しています。

PS1

この変数の値は、パラメータ展開、コマンド置換、および算術置換用に展開され、1次プロンプト文字列を定義します。デフォルトは `$` です。1次プロンプト文字列内の文字 `!` は、コマンド番号で置き換えられます。`!` を2つ連続して指定すると、プロンプト文字列の出力時に `!` が1つ表示されます。このマニュアルページの「コマンド行の再入力」セクションを参照してください。

シェルではデフォルト値を `PS1` に設定しています。

PS2

2次プロンプト文字列を指定します。デフォルトは `>` です。

シェルではデフォルト値を `PS2` に設定しています。

PS3

`select` ループ内で使用する選択プロンプト文字列を指定します。デフォルトは `#?` です。

シェルではデフォルト値を `PS3` に設定しています。

PS4

この変数の値は、パラメータ評価、コマンド置換、および算術置換用に展開され、実行トレースの各行の前に指定されます。デフォルトでは、`PS4` は `+` です。`PS4` が設定解除されると、実行トレースプロンプトも `+` になります。

シェルではデフォルト値を `PS4` に設定しています。

SHELL

シェルのパス名は環境内に保持されます。呼び出し時に、この変数のベース名が `rsh`、`rksh`、`rksh`、または `krsh` の場合、シェルの機能は制限されます。

`SHELL` はシェルでは設定されません。一部のシステムでは、`SHELL` は `login(1)` で設定されます。

TIMEFORMAT

このパラメータの値は、`time` の予約語が接頭辞として付けられたパイプラインの時間情報を表示する方法を指定する書式文字列として使用されます。`%` 文字は、時間値またはほかの情報に展開される書式シーケンスの始まりを示します。

書式シーケンスとその意味を次に示します。

`%%`

リテラルな `%`。

`%[p][l]R`

秒数での経過時間。

`%[p][l]U`

ユーザーモードで消費された CPU 時間 (秒)。

`%[p][l]S`

システムモードで消費された CPU 時間 (秒)。

`%P`

CPU 時間の割合 (パーセント)。(U + S) / R で計算します。

中括弧は省略可能な部分を示します。省略可能な *p* は、精度を小数点以下の桁数で指定します。0 の値にすると、小数点と小数点以下は出力されません。最大で、小数点以下第 3 位まで表示できます。*p* の値が 3 より大きい場合は、3 として扱われます。*p* が指定されていないければ、3 の値が使用されます。

省略可能な *l* を指定すると、時間 (0 より大きい場合)、分、秒から構成される *HHhMMmSS.FFs* という長い形式が使用されます。*p* の値は、小数点以下を含めるかどうかを指定します。

ほかの文字はすべて変更されずに出力され、末尾に復帰改行が追加されます。設定解除されている場合は、デフォルト値 `$'`

`\nreal\t%2lR\nuser\t%2lU\nsys\t%2lS'` が使用されます。値が NULL の場合、時間情報は表示されません。

TMOUT

TMOUT は、0 より大きな値に設定した場合に、read 組み込みコマンドのデフォルトのタイムアウト値になります。select 複合コマンドは、端末から入力が行われているときに、TMOUT 秒後に終了します。それ以外の場合は、端末からの読み取り中、指定された秒数以内に行が入力されなければ、シェルは終了します。この値をユーザーが超えられないような大きい値に設定してシェルをコンパイルすることもできます。

シェルではデフォルト値を TMOUT に設定しています。

VISUAL

この変数の値がパターン `*[Vv][Ii]*` に一致した場合、vi オプションが有効になります。「特殊コマンド」を参照してください。値がパターン `*gmacs*` に一致した場合、gmacs オプションが有効になります。値がパターン `*macs*` に一致した場合、emacs オプションが有効になります。VISUAL の値は EDITOR の値をオーバーライドします。

フィールド分割

パラメータ展開とコマンド置換後、置換結果からフィールド区切り文字 (IFS に含まれている文字) を検索し、その文字が見つかった位置で分割します。分割されたそれぞれの文字列がフィールドになります。明示的な NULL フィールド ("" または '') は保持されます。暗示的な NULL フィールド、つまり値を持たないパラメータや出力のないコマンド置換から得られたフィールドは削除されます。

braceexpand (-B) オプションが設定されている場合、IFS から得られたそれぞれのフィールドは、1 つまたは複数の中括弧パターンが含まれているかどうかチェックされます。有効な中括弧パターンは、`{*,*}`、`{l1..l2}`、`{n1..n2}`、`{n1..n2%fmt}` `{n1..n2..n3}`、または `{n1..n2..n3%fmt}` です。ここで、* は任意の文字を表

し、*l1*、*l2*は英字であり、*n1*、*n2*、*n3*は符号付きの数値です。*fmt*はprintfで使用する場合と同様に指定された書式です。どの形式でも、{と}の間にある文字から生成されるそれぞれの文字列に対して、{よりも前にある文字列が前に指定され、}よりも後ろにある文字列が後方に付加されて、フィールドが作成されます。結果となるフィールドは、中括弧パターンを含んでいるかどうかチェックされます。

1番目の形式では、{と,の間、,と,の間、および,と}の間の文字列ごとにフィールドが作成されます。*で表されている文字列には、対応する{と}を引用符で囲まずに埋め込むことができます。それ以外の場合は、*を伴うそれぞれの{および}は、引用符で囲む必要があります。

2番目の形式では、*l1*と*l2*はCロケールでどちらも大文字か小文字にする必要があります。この場合、フィールドは*l1*から*l2*までの文字ごとに作成されます。

残りの形式では、フィールド*n1*から始まる数値ごとに作成されます。これは、*n1*を*n3*ずつ増分して、*n2*になるまで続きます。*n3*が指定されていない形式では、*n1* ≤ *n2*の場合は*n3*を1とみなし、それ以外の場合は-1とみなして動作します。*%fmt*が指定されている形式では、すべての書式フラグ、幅、および精度を指定でき、*fmt*の末尾には任意の指定子*cdiouxX*を指定できます。たとえば、{a,z}{1..5..3%02d}{b..c}xは、a01bx、a01cx、a04bx、a04cx、z01bx、z01cx、z04bx、およびz4cxの8つのフィールドに展開されます。

ファイル名の生成 分割後、-f オプションが設定されていなければ、各フィールドに*、?、(、および[の文字が含まれているかどうか調べられます。これらの文字のいずれかが含まれていると、そのワードはパターンとみなされます。

いずれかのパターン文字を含む各ファイル名コンポーネントは、そのディレクトリのパターンに一致する名前が辞書編集方式の順にソートされたセットに置き換えられます。パターンに一致するファイル名が見つからなかった場合、パターンに接頭辞として~(N)が付けられていなければ(この接頭辞が付けられた場合は削除されます)、ファイル名のそのコンポーネントは変更されません。FIGNOREが設定されている場合、FIGNOREの値で定義されたパターンに一致したファイル名のコンポーネントそれぞれは、一致したファイル名を生成するときに無視されます。名前.と..も無視されます。FIGNOREが設定されていない場合、ファイル名の各コンポーネントに対応するパターンの最初の文字が.の文字そのものでないかぎり、このコンポーネントの先頭にある.の文字は無視されます。パターンマッチングのほかの使用法では、/と.は特別に扱われません。

* NULL文字列を含むあらゆる文字列と一致します。ファイル名の展開に使用した場合でglobstarオプションが有効になっているときは、2つの隣接する*は単独で、すべてのファイルと、0個以上のディレクトリおよびサブディレクトリに一致します。2つの隣接する*に/を続けると、ディレクトリおよびサブディレクトリだけが一致します。

? 任意の単一文字と一致します。

[...] 囲まれた文字のいずれかと一致します。2つの文字を - で区切ると、その間にある任意の文字(その2つの文字も含む)に一致します。先頭の [の次の文字が ! である場合、[] で囲まれていない任意の文字と一致します。- は、最初の文字または最後の文字として文字セットに挿入できます。[と] の内側では、[:class:] という構文で文字クラスを指定できます。ここで *class* は、ANSI-C 標準で定義された次のクラスのいずれかになります。

alnum alpha blank cntrl digit graph
lower print punct space upper
word xdigit

word は、*alnum* に文字 `_` を加えたものと同等です。[と] の内側では、[*=c*] という構文で同値化クラスを指定できます。この構文は、文字 *c* と同じ1次照合ウェイト(現在のロケールで定義)を持つすべての文字に一致します。[と] の内側では、[*.symbol.*] が照合シンボル *symbol* に一致します。

pattern-list は、1つのパターン、または&または|で区切られた複数のパターンのリストです。&はすべてのパターンが一致する必要があることを示しますが、|は1つのパターンだけ一致すればよいことを示します。複合パターンは、次のサブパターンの1つまたは複数から構成できます。

?(<i>pattern-list</i>)	指定されたパターンのいずれかと任意に一致します。
*(<i>pattern-list</i>)	指定されたパターンの0回以上の出現と一致します。
+(<i>pattern-list</i>)	指定されたパターンの1回以上の出現と一致します。
{ <i>n</i> }(<i>pattern-list</i>)	指定されたパターンの <i>n</i> 回の発生に一致します。
{ <i>m,n</i> }(<i>pattern-list</i>)	指定されたパターンの <i>m</i> から <i>n</i> 回の出現と一致します。 <i>m</i> を省略すると、0が使用されます。 <i>n</i> を省略すると、 <i>m</i> 回以上の出現と一致します。
@(<i>pattern-list</i>)	指定されたパターンの1つと完全に一致します。
!(<i>pattern-list</i>)	指定されたパターンのいずれにも一致しないものに一致します。

デフォルトでは、各パターンまたはサブパターンは、全体的にもっとも長い一致が生成される可能性のあるもっとも長い文字列に一致します。複数に一致する場合、文字列の先頭にもっとも近いものが選択されます。ただし、それぞれの複合パターンで (の前に - を挿入すると、指定された *pattern-list* に一致するもっとも短いものが使用されます。

pattern-list が括弧で囲まれている場合、バックスラッシュ文字 \ は、文字クラスの内部にある場合でも特別に扱われます。すべての ANSI-C 文字エスケープは認識され、指定された文字に一致します。さらに、次のエスケープシーケンスが認識されます。

- \d digit クラスの任意の文字と一致します。
- \D digit クラス以外の任意の文字と一致します。
- \s space クラスの任意の文字と一致します。
- \S space クラス以外の任意の文字と一致します。
- \w word クラスの任意の文字と一致します。
- \W word クラス以外の任意の文字と一致します。

%(*pattern-pairs*) の形式のパターンは、入れ子になった文字式の照合に使用できるサブパターンです。各 *pattern-pair* は、2 文字の並びであり、& や | を含めることはできません。最初の *pattern-pair* は照合の開始文字と終了文字を指定します。以降の *pattern-pair* はそれぞれ、開始文字と終了文字の一致をカウントするときにスキップされる入れ子になったグループの開始文字と終了文字を表します。*pattern-pair* の最初の文字が次の文字以外の英数字である場合、動作は不確定です。

- D 終了文字が現れると、一致が見つからなくてもこのパターンの検索を終了します。
- E 終了文字が、エスケープ文字として解釈されます。
- L 終了文字が、一致の検索時にすべての文字を無視させる引用符文字として解釈されます。
- Q 終了文字が、一致の検索時にエスケープ文字以外のすべての文字を無視させる引用符文字として解釈されます。

%({}Q"E\) は、{ から始まり、対応する } が見つかるまでの文字を照合しますが、二重引用符で囲まれた文字列の中にあるか、エスケープ文字 \ が前に指定されている { または } はカウントしません。{} がない場合、このパターンは任意の C 言語文字列に一致します。

複合パターン内の各サブパターンには、パターン内の (の位置に応じて、1 から始まる番号が付けられます。\\n の並びは、サブパターンそのものと同じ文字列に一致します。ここで、n は 1 桁の数字で \\n は n 番目のサブパターンのあとに置かれます。

パターンには ~(*options:pattern-list*) の形式のサブパターンを含めることができます。*options* または *:pattern-list* のどちらかは省略できます。ほかの複合パターンとは異なり、これらのサブパターンは、番号が付けられたサブパターンではカウントされません。*options* が存在する場合、次の 1 つまたは複数から構成できます。

- + 以降のオプションを有効にします。これはデフォルトです。
- 以降のオプションを無効にします。

- E パターンの残りの部分で、**egrep(1)** コマンドに類似した拡張正規表現の構文を使用します。
- F パターンの残りの部分で、**fgrep(1)** の表現の構文を使用します。
- g 最長の一致を提示します (貪欲)。
これはデフォルトです。
- G パターンの残りの部分で、**grep(1)** コマンドに類似した基本正規表現の構文を使用します。
- i 大文字と小文字を区別せずに一致を扱います。
- K パターンの残りの部分で、シェルパターン構文を使用します。
これはデフォルトです。
- l パターンを左側で固定します。
これはKスタイルパターンのデフォルトです。
- N これは無視されます。ただし、これが最初の文字であり、ファイル名生成で
使用され、一致が存在しない場合、ファイルパターンは空の文字列に展開
されます。
- r パターンを右側で固定します。
これはKスタイルパターンのデフォルトです。

options と *:pattern-list* の両方を指定した場合、*pattern-list* だけにオプションは適用されます。それ以外の場合、これらのオプションは次の *~(...)* によって無効になるまで、または *~(...)* を含むサブパターンの末尾で無効になるまで、有効のままです。

クォート 「定義」で一覧表示したメタキャラクタにはそれぞれ、シェルに対する特別な意味があります。

- g 最長の一致を提示します (貪欲)。これはデフォルトです。
- i 大文字と小文字を区別せずに一致を扱います。

options と *:pattern-list* の両方を指定した場合、*pattern-list* だけにオプションは適用されます。それ以外の場合、オプションは後続の *~(...)* によって無効になるまで、または *~(...)* を含むサブパターンの末尾で無効になるまで、有効のままです。

このマニュアルページの「定義」のセクションに一覧表示したメタキャラクタはそれぞれ、シェルに対して特別な意味を持ち、クォートしないかぎり、ワードの終わりを表します。文字は、その前にバックスラッシュ (\) を指定するとクォートでき

ます。つまり、その文字そのものを示すことができます。`\NEWLINE` のペアは削除されます。`$` が前に指定されていない一対の単一引用符 (') で囲まれた文字はすべてクォートされます。単一引用符は、単一引用符で囲んで表すことはできません。単一引用符で囲まれた文字列の前にクォートされていない `$` がある場合、次の場合を除き、ANSI-C 文字列として処理されます。

`\0` 残りの文字列が無視されます。
`\cx` 文字 CTRL-x に展開します。
`\C[.name.]` 照合要素 *name* に展開します。
`\e` エスケープ文字 (ASCII 033) と同等です。
`\E` エスケープ文字 (ASCII 033) と同等です。

一対の二重引用符 (") の内側では、パラメータとコマンドの置換が発生し、`\` は `\`、`'`、`"`、および `$` をクォートします。二重引用符に囲まれた文字列の前の `$` は、`c` または POSIX ロケールでは無視され、その他のロケールでは、文字列がロケール固有の文字列に置き換えられる場合があります。`$*` と `$@` の意味は、クォートされない場合、または変数代入値やファイル名として使用される場合は同じです。ただし、コマンド引数として使用されている場合、`"$*"` は `"$1$d$2d..."` と同等です。ここで、*d* は IFS 変数の先頭文字です。これに対して、`"$@"` は `"$1"$2" ....` と同等です。逆引用符 (`) の内側では、`\` は文字 `\`、`'`、および `$` を引用します。逆引用符を二重引用符内で指定すると、`\` も `"` をクォートします。

予約語や別名が持つ特別な意味は、予約語のいずれかの文字をクォートすることで失われます。関数名や組み込みコマンド名の認識は、それらをクォートしても変わりません。

算術評価

シェルは算術置換に対して算術評価を実行して、算術コマンドを評価し、インデックス付き配列添字を評価し、組み込みコマンド `shift` および `let` への引数を評価します。算術評価は、書式オペランドでの数値書式指定子に対応する組み込みコマンド `printf` での引数オペランドに対しても実行されます。[printf\(1\)](#) を参照してください。評価は、倍精度浮動小数点演算または 4 倍精度浮動小数点 (このデータ型が利用できるシステムの場合) を使用して実行されます。浮動小数点定数は、ANSI-C プログラミング言語の浮動小数点規則に従います。シングルバイト文字定数しか認識されず、文字キャストは認識されませんが、整数定数は、ANSI-C プログラミング言語の整数定数規則に従います。定数の形式には `[base]n` を使用できます。*base* は底を表す 2 から 64 の範囲の 10 進数で、*n* はその底の数です。9 より大きな数字は、小文字英字、大文字英字、`@`、および `_` でそれぞれ表されます。底が 36 以下の場合には、大文字と小文字を区別せずに使用できます。

算術式には、C 言語の式と同一の構文、優先度、および結合規則が使用されます。浮動小数点の数量に適用される C 言語の演算子をすべて使用できます。さらに、演算子 `**` を累乗に使用できます。これは乗算よりも優先度が高く、結合されたままになります。算術変数または部分式の値がロング整数として表現できる場

合、C 言語の整数算術演算をすべて実行できます。算術式内では、パラメータ展開構文を使用しなくても、名前を変数を参照できます。変数を参照すると、その値は算術式として評価されます。

C 演算ライブラリにある次の演算ライブラリ関数のいずれも、算術式内で使用できます。

```
abs acos acosh asin asinh atan atan2 atanh cbrt
copysign cos cosh erf erfc exp exp2 expm1 fabs
fdim finite floor fma fmax fmod hypot ilogb
int isinf isnan lgamma log log2 logb
nearbyint nextafter nexttoward pow remainder
rint round sin sinh sqrt tan tanh tgamma trunc
```

倍精度浮動小数点の変数の内部表現は、**typeset** 特殊組み込みコマンドの **-E [n]** または **-F [n]** オプションで指定できます。**-E** オプションを指定すると、値の展開は、展開されるときに、科学表記法を使用して表現されます。省略可能なオプション引数 *n* は有効数字の桁数を定義します。**-F** オプションを指定すると、展開は、展開されるときに、浮動小数値として表現されます。省略可能なオプション引数 *n* は、この場合に、小数点以下の位数を定義します。

整数の変数の内部表現は、**typeset** 特殊組み込みコマンドの **-i [n]** オプションで指定できます。省略可能なオプション引数 *n* は、変数を展開するとき使用する底を指定します。底を指定しない場合、10 の底が使用されます。

算術評価は、**-E**、**-F**、または **-i** オプションを使用した変数に対する代入値ごとに実行されます。型が整数である変数に浮動小数点値を代入すると、小数部分が切り捨てられます。

プロンプト

シェルは、対話的に使用すると、パラメータ展開、コマンド置換、および算術置換に展開したあとで、コマンドを読み取る前に、**PS1** の値のプロンプトが表示されます。さらに、プロンプト内の単一の **!** はそれぞれコマンド番号に置き換えられます。**!!** が、プロンプト内に **!** を指定するには必要です。復帰改行を入力したあとで、コマンドを完了するためにさらに入力が必要な場合は、2 次プロンプト (**PS2** の値) が出力されます。

条件式

条件式 (*conditional expression*) は、ファイルの属性をテストしたり文字列を比較したりするときに、複合コマンドの **[[** とともに使用します。**[[** と **]]** の間のワードについては、フィールド分割とファイル名生成は実行されません。

各条件式は、次の単項式または 2 項式をいくつか組み合わせて構築できます。

-a file *file* が存在すれば、真です。

このオプションは **-e** と同じです。このオプションは廃止されています。

-b <i>file</i>	<i>file</i> が存在し、ブロック型特殊ファイルであれば、真です。
-c <i>file</i>	<i>file</i> が存在し、文字型特殊ファイルであれば、真です。
-d <i>file</i>	<i>file</i> が存在し、ディレクトリであれば、真です。
-e <i>file</i>	<i>file</i> が存在すれば、真です。
-f <i>file</i>	<i>file</i> が存在し、通常ファイルであれば、真です。
-g <i>file</i>	<i>file</i> が存在し、setgid ビットが設定されていれば、真です。
-G <i>file</i>	<i>file</i> が存在し、ファイルのグループがこのプロセスの実効グループ ID と一致していれば、真です。
-h <i>file</i>	<i>file</i> が存在し、シンボリックリンクであれば、真です。
-k <i>file</i>	<i>file</i> が存在し、スティッキービットが設定されていれば、真です。
-L <i>file</i>	<i>file</i> が存在し、シンボリックリンクであれば、真です。
-n <i>string</i>	<i>string</i> の長さが 0 以外であれば、真です。
-N <i>file</i>	<i>file</i> が存在し、更新時間が最終アクセス時間よりあとであれば、真です。
-o <i>option</i>	<i>option</i> という名前のオプションが有効であれば、真です。
-o ? <i>option</i>	<i>option</i> という名前のオプションが、有効なオプション名であれば、真です。
-O <i>file</i>	<i>file</i> が存在し、このプロセスの実効ユーザー ID がそのファイルを所有していれば、真です。
-p <i>file</i>	<i>file</i> が存在し、FIFO 特殊ファイルまたはパイプであれば、真です。
-r <i>file</i>	<i>file</i> が存在し、現在のプロセスで読み取り可能であれば、真です。
-s <i>file</i>	<i>file</i> が存在し、サイズがゼロより大きければ、真です。
-S <i>file</i>	<i>file</i> が存在し、ソケットであれば、真です。
-t <i>fildev</i>	ファイル記述子番号が <i>fildev</i> であるファイルが開いていて、端末デバイスに対応していれば、真です。
-u <i>file</i>	<i>file</i> が存在し、setuid ビットが設定されていれば、真です。
-w <i>file</i>	<i>file</i> が存在し、現在のプロセスで書き込み可能であれば、真です。

<code>-x file</code>	<code>file</code> が存在し、現在のプロセスで実行可能であれば、真です。 <code>file</code> が存在し、ディレクトリである場合は、現在のプロセスにそのディレクトリを検索する権限があれば、真です。
<code>-z string</code>	<code>string</code> の長さがゼロであれば、真です。
<code>file1 -ef file2</code>	<code>file1</code> と <code>file2</code> が存在し、同じファイルを参照していれば、真です。
<code>file1 -nt file2</code>	<code>file1</code> が存在し、 <code>file2</code> が存在しないか、または <code>file1</code> が <code>file2</code> よりも新しい場合は、真です。
<code>file1 -ot file2</code>	<code>file2</code> が存在し、 <code>file1</code> が存在しないか、または <code>file1</code> が <code>file2</code> よりも古い場合は、真です。
<code>string</code>	<code>string</code> がNULLでなければ、真です。
<code>string == pattern</code>	<code>string</code> が <code>pattern</code> と一致すれば、真です。 <code>pattern</code> の任意の部分をクォートして、文字列として一致させることができます。 <code>pattern</code> に一致した場合、 <code>.sh.match</code> 配列変数には、その一致とサブパターンの一致が含まれます。
<code>string = pattern</code>	<code>==</code> と同じですが、廃止されました。
<code>string != pattern</code>	<code>string</code> が <code>pattern</code> と一致しなければ、真です。 <code>string</code> が <code>pattern</code> に一致した場合、 <code>.sh.match</code> 配列には、その一致とサブパターンの一致が含まれます。
<code>string =~ ere</code>	<code>string</code> がパターン <code>~(E)ere</code> に一致すれば、真です。 <code>ere</code> は拡張正規表現です。
<code>string1 < string2</code>	それぞれの文字のASCII値に基づいて、 <code>string1</code> が <code>string2</code> より小さい場合に、真です。
<code>string1 > string2</code>	それぞれの文字のASCII値に基づいて、 <code>string1</code> が <code>string2</code> より大きい場合に、真です。

次のそれぞれの式で、`file`が`/dev/fd/n` (`n`は整数) という形式の場合は、記述子番号が`n`である開かれたファイルがテストされます。次の廃止された算術比較がサポートされています。

<code>exp1 -eq exp2</code>	<code>exp1</code> が <code>exp2</code> と等しければ、真です。
<code>exp1 -ge exp2</code>	<code>exp1</code> が <code>exp2</code> 以上であれば、真です。
<code>exp1 -gt exp2</code>	<code>exp1</code> が <code>exp2</code> より大きければ、真です。
<code>exp1 -le exp2</code>	<code>exp1</code> が <code>exp2</code> 以下であれば、真です。
<code>exp1 -lt exp2</code>	<code>exp1</code> が <code>exp2</code> 未満であれば、真です。
<code>exp1 -ne exp2</code>	<code>exp1</code> が <code>exp2</code> と等しくなければ、真です。

次のいずれかを使用して、これらの基本式から複合式を構築できます。優先度の高いものから順に並べてあります。

<i>(expression)</i>	<i>expression</i> が真であれば、真です。式をグループ化するとき 사용합니다。
<i>! expression</i>	<i>expression</i> が偽であれば、真です。
<i>expression1 && expression2</i>	<i>expression1</i> と <i>expression2</i> の両方が真であれば、真です。
<i>expression1 expression2</i>	<i>expression1</i> と <i>expression2</i> のどちらかが真であれば、真です。

入出力

コマンドの実行前に、シェルが解釈する特殊な表記を使って入出力先を変更(リダイレクト)できます。次の各項目は、単純コマンド内の任意の位置またはコマンドの前後に指定できますが、起動されたコマンドには引き渡されません。このセクションで示す場合を除き、*word* または *digit* を使用する前に、コマンド置換、パラメータ展開、および算術置換が発生します。シェルが対話型であり、パターンが単一のファイルに一致する場合にのみ、ファイル名生成が行われます。フィールド分割は実行されません。

次のリダイレクションのそれぞれで、*file* が */dev/sctp/host/port*、*/dev/tcp/host/port*、または */dev/udp/host/port* の形式である場合 (*host* はホスト名またはホストアドレス、*port* はサービスの名前または整数のポート番号)、リダイレクションは、対応するソケットへの *tcp*、*sctp*、または *udp* 接続を確立しようと試みます。

リダイレクト演算子の文字間にはスペースを入れられません。

<i><word</i>	<i>word</i> というファイルを標準入力(ファイル記述子 0)として使用します。
<i>>word</i>	<i>word</i> というファイルを標準出力(ファイル記述子 1)として使用します。ファイルが存在しない場合は、ファイルを作成します。ファイルが存在し、 <i>noclobber</i> オプションが有効である場合は、エラーになります。それ以外の場合は、長さが 0 になります。
<i>> word</i>	<i>></i> と同じですが、 <i>noclobber</i> オプションを無視します。
<i>>>word</i>	<i>word</i> というファイルを標準出力として使用します。ファイルが存在する場合、(ファイルの終わりまでシークしたあと)そのファイルに出力を追加します。そうでない場合は、ファイルを作成します。
<i><>word</i>	<i>word</i> というファイルを標準入力として読み書き用に開きます。
<i><<[-]word</i>	シェルへの入力として、すべてのクォートを削除したあとの <i>word</i> と同一の行まで、またはファイルの終わりまで読み取ります。 <i>word</i> に対してパラメータ置換、コマンド置換、算術置換、ファイル名生成を実行しません。 <i>here-document</i> が生成されて標準入力になります。 <i>word</i> のいずれかの文字がクォートされていると、ドキュメント

の文字は解釈されません。それ以外の場合、パラメータ展開、コマンド置換、および算術置換が行われ、`\NEWLINE`が無視されます。また、`\`を使って、`\`、`$`、`'`の文字をクォートする必要があります。`-`を`<<`のあとに付加すると、`word`とドキュメントから先行するタブがすべて取り除かれます。`#`を`<<`のあとに付加すると、先行するスペース文字とタブがドキュメントの先頭の行から取り除かれ、最大で同じだけのインデントが残りの行と`word`から取り除かれます。タブ位置は8カラムごとであると想定して、インデントは判断されます。

<code><<<word</code>	パラメータ展開、コマンド置換、算術置換が行われたあとに、 <code>word</code> が here-document の内容になる here-document の短い形式。
<code><&digit</code>	ファイル記述子 <code>digit</code> から標準入力複製されます。同様に、標準出力については <code>>&digit</code> を使用します。 dup(2) を参照してください。
<code><&digit-</code>	<code>digit</code> で指定されたファイル記述子は標準入力に移動します。同様に、標準出力については <code>>&digit-</code> を使用します。
<code><&-</code>	標準入力を閉じます。同様に、標準出力については <code>>&-</code> を使用します。
<code><&p</code>	並行プロセスからの入力を標準入力に移動します。
<code>>&p</code>	並行プロセスへの出力を標準出力に移動します。
<code><#((expr))</code>	算術式 <code>expr</code> を評価して、その結果得られた値のバイト数だけファイルの先頭から離れた位置に、ファイル記述子 <code>0</code> を配置します。 <code>expr</code> を評価するときに、 <code>CUR</code> と <code>EOF</code> の変数はそれぞれ、現在のオフセットとファイルの終わりのオフセットに評価されます。
<code>>#((expr))</code>	<code><#</code> と同じですが、ファイル記述子 <code>1</code> に適用されます。
<code><#pattern</code>	パターンを含む次の行の先頭を探します。
<code><##pattern</code>	<code><#</code> と同じですが、スキップされるファイルの部分が、標準出力にコピーされる点が異なります。

いずれかのリダイレクト演算子の前に、スペースを入れずに数字を付けると、参照されるファイル記述子番号は(デフォルトの `0` または `1` ではなく)その数字で指定された番号になります。`>&-`、`>#`、および `<#` 形式以外のリダイレクト演算子のいずれかの前に、スペースを入れずに `{varname}` を指定すると、シェルでは `>10` のファイル記述子番号が選択され、変数 `varname` に格納されます。`>&-`、または `>#` と `<#` のどちらかの形式の前に `{varname}` が指定されている場合、`varname` の値で、閉じるファイル記述子または配置するファイル記述子を定義します。例:

```
... 2>&1
```

これは、ファイル記述子 `2` をファイル記述子 `1` の複製として書き込み用に開くことを意味し、また、

```
exec [n]<file
```

これは、読み取り用に *file* を開き、ファイル記述子番号を変数 *n* に格納することを意味します。リダイレクションを指定する場合、記述する順序が重要になります。シェルは、評価時点での (*file_descriptor* と *file* の) 関連性について、各リダイレクションを評価します。例:

```
... 1>fname 2>&1
```

これは、まず *fname* というファイルにファイル記述子 1 を関連付けます。次に、ファイル記述子 1 (つまり *fname*) に関連するファイルに、ファイル記述子 2 を関連付けます。リダイレクションの向きが逆の場合は、まずファイル記述子 2 を端末に関連付け (ファイル記述子 1 はすでに端末に関連付けられているとして)、次にファイル記述子 1 をファイル *fname* に関連付けます。ジョブ制御が有効でない場合にコマンドのあとに & を指定すると、コマンドのデフォルトの標準入力はいずれも */dev/null* という空ファイルになります。それ以外の場合は、コマンドを実行するための環境に起動側シェルのファイル記述子 (入出力指定で変更可能) が含まれます。

環境

環境とは、実行されるプログラムに渡される、名前と値が対になったリストのことです。通常の引数リストと同様の方法で指定します ([environ\(5\)](#) 参照)。

名前は識別子、値は文字列である必要があります。シェルが環境と対話する方法はいくつかあります。シェルは、起動されると、環境を走査して、見つけた名前ごとに変数を作成し、対応する値および属性を設定し、さらに `export` というマークを付けます。実行されるコマンドは環境を引き継ぎます。ユーザーがこれらの変数の値を変更するか、新しい変数を作成した場合は、`export` コマンドまたは `typeset -x` コマンドを使用すれば、それらの値が環境の一部になります。したがって、実行されるコマンドが参照する環境は、シェルが最初に引き継いだ名前と値のペア (その値は現在のシェルで変更可能) に、`export` コマンドまたは `typeset -x` コマンドで指定した変数を加えたものになります。1 つまたは複数の変数代入を先頭に付加すれば、単純コマンドまたは関数の環境を拡張できます。変数代入引数は、*identifier=value* という形式のワードです。したがって、

```
TERM=450 cmd args
```

および

```
(export TERM; TERM=450; cmd args)
```

は、*cmd* の実行に関するかぎり、同じことを意味します。ただし、「組み込みコマンド」セクションで示した特殊組み込みコマンドのうち、ダガーが先頭に置かれているものが指定された場合を除きます。廃止された `-k` オプションを設定すると、変数代入引数はすべて環境に格納されます。これらの引数がコマンド名のあとに指定された場合も同様です。

次の例では、最初に `a=b c` を表示し、次に `c` を表示します。

```
echo a=b c
set -k
echo a=b c
```

この機能は、シェルの初期バージョン用に作成されたスクリプトで使用するためのものです。新しいスクリプトには使用しないでください。

関数

歴史的な理由から、関数の定義には、`name()` 構文と `function name` 構文の2つの方法が存在します。これらについては、このマニュアルページの「コマンド」セクションを参照してください。

シェル関数は内部で読み取られ、保存されます。別名は、関数を読み取るときに解釈されます。関数はコマンドと同様に実行され、引数は定位置パラメータとして渡されます。詳細は、このマニュアルページの「実行」セクションを参照してください。

`function name` 構文で定義され、名前呼び出された関数は、呼び出し側と同じプロセスで実行され、すべてのファイルと現在の作業用ディレクトリを呼び出し側と共有します。呼び出し側が受け取るトラップは、関数の内部でデフォルトの動作に再設定されます。関数が受け取らないか、または無視するトラップ条件がある場合、関数は終了し、その条件が呼び出し側に引き渡されます。関数内部で設定された `EXIT` に基づくトラップは、関数が完了したあとに呼び出し側の環境で実行されます。通常、変数は呼び出し側プログラムと関数の間で共有されます。ただし、関数内で `typeset` 特殊組み込みコマンドを使用すると、現在の関数をスコープとする局所変数を定義できます。関数内で別の関数を呼び出す際に、これらの局所変数を渡すには、呼び出しの前に変数代入リスト内に指定するか、または名前参照として渡される引数として指定します。関数内でエラーが発生すると、呼び出し側に制御が戻ります。

`name()` 構文で定義された関数と、`function name` 構文で定義され、特殊組み込みコマンドで呼び出された関数は、呼び出し側の環境で実行され、呼び出し側とすべての変数およびトラップを共有します。このような関数の実行中にエラーが発生すると、これらを含むスクリプトは中止されます。

`return` 特殊組み込みコマンドは、関数の呼び出しから戻るときに使用します。

`typeset` 特殊組み込みコマンドの `-f` または `+f` オプションを使用すると、関数名を一覧表示できます。`-f` では、関数のテキストがある場合は、これも一覧表示されます。`unset` 特殊組み込みコマンドの `-f` オプションを使用すると、関数を未定義状態にできます。

通常、シェルがシェルスクリプトを実行すると、関数は未設定になります。シェルの個別起動をまたがって定義する必要のある関数は、ディレクトリに置き、このディレクトリの名前を `FPATH` 変数に含めるようにしてください。これらは、`ENV` ファイルで指定することもできます。

ディシプリン関数 それぞれの変数に、0 個以上のディシプリン関数を関連付けられます。シェルは初期状態で `get`、`set`、`append`、および `unset` のディシプリン名を認識しますが、ほとんどのシステムでは、`builtin` 組み込みユーティリティが持つ C プログラミングインタフェース拡張機能を介して、実行時にほかのディシプリン名を追加できます。変数に対して `get` ディシプリンが定義されている場合、指定された変数が参照されるたびにこのディシプリンが呼び出されます。ディシプリン関数内で、変数 `.sh.value` に値が代入された場合、参照される変数は代わりにこの値に評価されます。変数に対して `set` ディシプリンが定義されている場合、指定された変数に値が代入されるたびにこのディシプリンが呼び出されます。変数に対して `append` ディシプリンが定義されている場合、指定された変数に値が付加されるたびにこのディシプリンが呼び出されます。変数 `.sh.value` にはディシプリンを呼び出す前の変数の値が格納され、ディシプリンの完了後、変数には `.sh.value` の値が代入されます。ディシプリン内で `sh.value` が設定解除された場合、その値は変更されません。変数に対して `unset` ディシプリンが定義されている場合、指定された変数が設定解除されるたびに呼び出されます。このディシプリン関数内から明示的に設定解除されないかぎり、変数は設定解除されません。

変数 `.sh.name` には、ディシプリン関数を呼び出す対象の変数の名前が含まれます。`.sh.subscript` は変数の添字です。`.sh.value` には、`set` ディシプリン関数内で代入される値が含まれます。`set` ディシプリンの場合、`.sh.value` を変更すると、代入される値が変更されます。

ジョブ `set` コマンドの `monitor` オプションを有効にすると、対話型シェルがジョブを各パイプラインと関連付けます。このオプションは、`jobs` コマンドが表示する現在のジョブのテーブルを維持し、これらのジョブに整数番号を割り当てます。ジョブを `&` で非同期に起動すると、シェルは、次の形式の行を表示します。

```
[1] 1234
```

非同期に起動されたジョブはジョブ番号 1 であり、プロセス ID が 1234 であるプロセスが 1 つ (トップレベル) あることを示します。

ジョブの実行中に停止する必要がある場合は、`CTRL-z` を押して、`STOP` シグナルを現在のジョブに送信します。シェルは通常、ジョブが停止されたことを示すメッセージを表示し、別のプロンプトを表示します。これで、このジョブの状態を `bg` コマンドでバックグラウンドで処理するか、またはほかのコマンドを実行してから、`fg` というコマンドでジョブをフォアグラウンドに移すことができます。`CTRL-z` はただちに効果を発揮します。つまり `CTRL-z` は、保留中の出力や読み取られていない入力 that だちに中止されるという点で、割り込みに似ています。

バックグラウンドで実行中のジョブは、端末から読み取ろうとすると停止します。通常バックグラウンドジョブは出力を生成できますが、`sttytostop` というコマンドを指定すればこの出力生成も無効にできます。この `tty` オプションを設定すると、バックグラウンドジョブは、入力の読み取り時と同様に出力を生成しようとする と停止します。

シェル内のジョブを参照する方法はいくつかあります。そのジョブのいずれかのプロセスの ID を使っても、また次のいずれかを使っても参照できます。

`%number` 指定された番号のジョブ。
`%string` コマンド行が `string` で始まるジョブ
`;%?string` コマンド行に `string` を含むジョブ
`%%` 現在のジョブ
`%+` `%%` と同等
`%-` 直前のジョブ

シェルは、プロセスの状態が変更されると、ただちにそれを検出します。ジョブがブロックされてそれ以上進めない状態になると、プロンプトを発行する直前にだけ、シェルはその旨をユーザーに通知します。これは、ユーザーの作業の妨げにならないようにするためです。set コマンドの `notify` オプションを設定すると、変更が起きるとすぐに、シェルはこれらのジョブ変更メッセージを出力します。

`monitor` オプションが有効になっていると、バックグラウンドジョブが完了するたびに、CHLD に対して設定されているトラップをトリガーします。

ジョブの実行または停止中にシェルを終了しようとする、停止中(実行中)のジョブがある旨の警告 (You have stopped(running) jobs) が表示されます。jobs コマンドを使用して、それらのジョブが何かを表示できます。ただちにシェルを再終了しようとする、シェルは2度目の警告は出さず、停止中のジョブは終了します。ログインシェルは、HUP シグナルを受信すると、disown 組み込みコマンドで所有解除されていない各ジョブに HUP シグナルを送信します。

シグナル 起動されたコマンドに対する INT シグナルと QUIT シグナルは、コマンドの後ろに & が指定され、ジョブの `monitor` オプションが有効でない場合、無視されます。それ以外の場合は、シェルが親から引き継いだ値がシグナルに含まれます。trap 組み込みコマンドを参照してください。

実行 コマンドが読み込まれるたびに、置換が実行されます。コマンド名は、このマニュアルページの「特殊組み込みコマンド」セクションで記した、いずれかのコマンドと一致した場合、現在のシェルプロセス内で実行されます。次に、コマンド名がユーザー定義関数と一致するかどうかチェックされます。一致する場合は、位置パラメータが保存され、関数呼び出しの引数に再設定されます。関数は現在のシェルプロセスでも実行されます。関数が完了するか、`return` を発行すると、位置パラメータリストが復元されます。function name 構文で定義された関数の場合、関数内の EXIT に設定されているトラップが実行されます。関数の終了値は、最後に実行されたコマンドの値です。コマンド名が、特殊組み込みコマンドやユーザー定義関数ではなく、組み込みコマンドの1つである場合は、現在のシェルプロセスで実行されます。

PATHというシェル変数は、コマンドを含んでいるディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:)で区切ります。デフォルトのパスは /bin:/usr/bin: で、/bin、/usr/bin、現在のディレクトリの順に指定されています。現在のディレクトリは、複数のコロンを連続して記述するか、パスリストの始めか終わりにコロンを付ければ指定できます。コマンド名にスラッシュ(/)が含まれている場合は、検索パスは使用されません。スラッシュが含まれていなければ、指定された名前の、ディレクトリではない実行可能ファイルが、パスの各ディレクトリで検索されます。実行可能ファイルが見つかった場合、および指定されたパス名に対応するコマンドの組み込みバージョンがあるとシェルが判断した場合は、現在のプロセスでこの組み込みコマンドが呼び出されます。実行可能ファイルが見つかり、このディレクトリが FPATH 変数の値にも含まれている場合は、設定済みの別名だけが展開される点を除いて、コマンドの引数である場合と同様にこのファイルは現在のシェル環境に読み込まれ、このマニュアルページで説明するように、指定された名前の関数が実行されます。実行可能ファイルが見つからず、ファイル .paths が見つかり、このファイルに FPATH=path という形式の行が含まれ(path は既存のディレクトリ)、指定されている名前のファイルがこのディレクトリに含まれている場合、このファイルは、. 特殊組み込みコマンドの引数である場合と同様に現在のシェル環境に読み込まれ、指定された名前の関数が実行されます。これら以外で実行可能ファイルが見つかった場合は、プロセスが作成され、**exec(2)** を使用してコマンドの実行が試行されます。

実行可能ファイルが見つかった場合、そのファイルが見つかったディレクトリ内で .paths という名前のファイルが検索されます。このファイルが見つかり、そこに BUILTIN_LIB=value という形式の行が含まれている場合、builtin -f のオプション引数である場合と同様に value で名前が付けられたライブラリが検索されます。指定された名前の組み込みコマンドが含まれている場合、この名前のコマンドではなく組み込みコマンドが実行されます。これ以外でこのファイルが見つかり、その最初の行または2行目に name= value の形式の行が含まれている場合、value で指定されたディレクトリをディレクトリリストの先頭に加えることによって、環境変数 name が変更されます。value が絶対ディレクトリでない場合は、実行可能ファイルが見つかったディレクトリに対する相対ディレクトリとしてディレクトリが指定されます。環境変数 name がまだ存在しない場合は、指定されたコマンドの環境リストに追加されます。

ファイルが実行権を有するが、それが a.out ファイルでない場合、シェルコマンドの入ったファイルとみなされます。そのファイルを読み取るときは、別のシェルが生成されます。この場合、エクスポートされていない変数はすべて削除されます。シェルコマンドファイルに読み取り権がない場合またはファイルに setuid および setgid ビットが設定されている場合、あるいはその両方の場合、シェルはエージェントを実行します。そして、このエージェントがアクセス権を設定し、開かれているファイルとして渡されたシェルコマンドファイルでシェルを実行します。括弧で囲まれたコマンドは、エクスポートされていない変数を削除することなく、サブシェルで実行されます。

コマンド行の再入力

端末装置から最近入力された HISTSIZE が示す個数 (デフォルトは 512 個) のコマンドのテキストは、履歴ファイルに保存されています。\$HOME/.sh_history というファイルは、HISTFILE 変数が設定されていない場合、または変数が示すファイルが書き込み不可能な場合に使用されます。シェルは、同じ名前の HISTFILE を使用する対話型シェルすべてのコマンド履歴を使用できます。hist という組み込みコマンドは、このファイルの一部を一覧表示または編集するときに使用します。編集またはリスト表示されるファイルの部分は、番号か、またはコマンドの最初の文字を指定することによって選択できます。単一のコマンドを指定することも、コマンドの範囲を指定することも可能です。hist の引数としてエディタプログラムが指定されていないと、HISTEDIT という変数の値が使用されます。HISTEDIT が設定されていない場合、廃止された変数 FCEDIT が使用されます。FCEDIT が未定義の場合は、/bin/ed が使用されます。エディタを終了したときに、編集されたコマンドは出力され再実行されます。ただし、書き込まずに終了した場合は除きます。-s オプション (廃止されたバージョンではエディタ名 -) を指定すると、編集段階が省かれ、コマンドが再実行されます。この場合、old=new という形式の代入パラメータを使用すれば、実行前にコマンドを変更できます。たとえば、'hist -s' の別名として定義されている設定済みの別名 r を使用して、'r bad=good c' と入力すると、c という文字で始まるコマンドのうち最新のものが、その記述中の最初の bad という文字列を good に置き換えて再実行されます。

インライン編集オプション

通常、端末装置から入力する各コマンド行では、コマンドのあとに復帰改行 (RETURN または LINE FEED) しか入力できません。emacs、gmacs、vi のいずれかのオプションが有効な場合、ユーザーはコマンド行を編集できます。いずれかの編集モードにするには、対応するオプションを設定します。編集オプションは、いずれかのオプション名で終了する値を VISUAL 変数または EDITOR 変数に代入するたびに、自動的に選択されます。

編集機能では、ユーザーの端末が RETURN を改行のないキャリッジリターンとして扱うこと、およびスペース文字がスクリーン上の現在の文字を上書きすることが必要です。

multiline オプションが有効になっていないかぎり、編集モードは、ユーザーがウィンドウを介して現在の行を見るという概念を実現します。ウィンドウ幅は、COLUMNS が定義されていればその値になり、未定義の場合は 80 になります。ウィンドウの幅が小さすぎて、プロンプトを表示すると入力用に 8 カラム以上残すことができない場合は、プロンプトが左端から切り捨てられます。ウィンドウ幅から 2 を引いた長さより行が長い場合は、ウィンドウの終わりにマークを表示してユーザーに通知します。カーソルが移動し、ウィンドウの境界に達すると、ウィンドウはカーソルを中心としてセンタリングされます。行がウィンドウの右端、左端、または両端を超えている場合は、それぞれ >、<、* のマークが使用されます。

各編集モードでは、検索コマンドから履歴ファイルにアクセスできます。パターンではなく文字列だけがマッチングの対象になります。ただし、文字列の先頭に^があると、マッチングの開始位置が行の先頭に限定されます。

それぞれの編集モードには、部分的に入力したワードに一致するファイルまたはコマンドを一覧表示する操作が用意されています。行の最初のワード

か、;、|、&、または(のあとの最初のワードに適用され、ワードの先頭が~でなく、ワードに/が含まれない場合、部分的なワードに一致する可能性のある、PATH変数で定義された別名、関数、実行可能なコマンドのリストが表示されます。それ以外の場合、指定されたワードに一致するファイルのリストが表示されます。部分的に入力されたワードにファイル展開文字が含まれない場合、これらのリストを生成する前に、*が付加されます。生成されたリストを表示したあとで、入力行が再描画されます。これらの操作はそれぞれ、コマンド名一覧表示、ファイル名一覧表示と呼ばれます。ほかにもコマンド名補完とファイル名補完と呼ばれる操作があります。これらの操作では、一致するコマンドまたはファイルのリストを作成しますが、リストを出力するのではなく、現在のワードを完全一致または部分一致と置き換えます。ファイル名補完の場合、一致が1つしかなければ、ファイルがディレクトリの場合は/が付加され、ファイルがディレクトリでない場合はスペースが付加されます。それ以外の場合は、一致したファイルすべてに共通するもっとも長い接頭辞がワードと置き換えられます。コマンド名補完の場合、ファイル名のうち最後の/以降の部分だけが、もっとも長いコマンド接頭辞の検出に使用されます。1つの名前しかこの接頭辞に一致しない場合、ワードはコマンド名にスペースを続けた文字列に置き換えられます。補完にタブを使用した場合で複数の一致が存在したときは、後続のタブで、一致する代替候補の番号付きリストが表示されます。選択番号に続けてタブを入力することにより、特定の項目を選択できます。

キーバインド

KEYBDトラップは、キーの入力時にキーを傍受して、実際にシェルで表示される文字を変更するために使用できます。このトラップは、端末からの読み取り中に文字(または、最初の文字がESCのときは文字の並び)が入力されるたびに実行されます。

変数 `.sh.edchar` には、トラップを生成した文字または文字の並びが含まれます。トラップアクションで `.sh.edchar` の値を変更すると、シェルは、元の値ではなく新しい値がキーボードから入力された場合と同じように動作します。変数 `.sh.edcol` は、入力時点のカーソルの入力カラム番号に設定されます。変数 `.sh.edmode` は、vi挿入モードのときにESCに、それ以外の場合にNULLに設定されます。`.sh.edchar` に代入された値の前に `${.sh.editmode}` を付けると、シェルは、まだ制御モードになっていない場合はこのモードに切り替わります。

このトラップは、編集指令への引数として入力された文字や、文字検索の入力を読み取っているときは呼び出されません。

emacs 編集モード

emacs または gmacs オプションを有効にすると、対応する編集モードに移行します。この2つのモードは、^Tの扱い方だけが異なります。編集を行うには、ユーザーは訂正が必要な位置にカーソルを移動し、必要に応じて文字やワード

を挿入または削除します。編集用のコマンドは、いずれも制御文字またはエスケープシーケンスです。制御文字は、キャレット (^) とそれに続く文字で表記されます。

たとえば、`^F` は CTRL/F を表します。これは、CTRL (コントロール) キーを押しながら `f` キーを押して入力します。シフトキーは押しません。(`^?` は DEL (削除) キーを表します。)

エスケープシーケンスは `M-` とそれに続く文字で表記されます。たとえば、`M-f` (メタ `f` と呼ぶ) は ESC キー (ASCII 033) に続いて `f` を押すことで入力します。`M-F` は ESC のあとに `F` を押すことを表します。

すべての編集コマンドは、行の先頭だけでなく、行のどこからでも実行できます。特に断りのないかぎり、編集コマンドのあとに RETURN キーや LINE FEED キーは入力しません。

<code>^F</code>	カーソルを 1 文字だけ前方 (右) に移動します。
<code>M-[C</code>	カーソルを 1 文字だけ前方 (右) に移動します。
<code>M-f</code>	カーソルを 1 ワードだけ前方に移動します。emacs エディタでは、「ワード」は英文字、数字、および下線からなる文字列を意味します。
<code>^B</code>	カーソルを 1 文字だけ後方 (左) に移動します。
<code>M-[D</code>	カーソルを 1 文字だけ後方 (左) に移動します。
<code>M-b</code>	カーソルを 1 ワードだけ後方に移動します。
<code>^A</code>	カーソルを行の先頭に移動します。
<code>M-[H</code>	カーソルを行の先頭に移動します。
<code>^E</code>	カーソルを行の末尾に移動します。
<code>M-[Y</code>	カーソルを行の末尾に移動します。
<code>^]char</code>	現在の行で文字 <i>char</i> が次に現れる位置にカーソルを移動します。
<code>M-^]char</code>	現在の行で文字 <i>char</i> が前に現れる位置にカーソルを移動します。
<code>^X^X</code>	カーソルとマークを入れ替えます。
<i>erase</i>	前の 1 文字を削除します。ユーザー定義の消去文字は、 <code>stty(1)</code> コマンドで定義され、通常は、 <code>^H</code> または <code>#</code> です。
<i>lnext</i>	次の文字の編集機能を無効にします。ユーザー定義の次のリテラル文字は、 <code>stty(1)</code> コマンドによって定義されます。定義されていない場合は <code>^V</code> です。
<code>^D</code>	現在の文字を削除します。

M-d	現在のワードを削除します。
M-^H	メタバックスペース。直前のワードを削除します。
M-h	直前のワードを削除します。
M-^?	メタ DEL。直前のワードを削除します。割り込み文字が^? (デフォルトの DEL) である場合、このコマンドは動作しません。
^T	emacs モードでは、現在の文字を前の文字と入れ替え、カーソルを進めます。gmacs モードでは、直前の2つの文字を入れ替えます。
^C	現在の文字を大文字にします。
M-c	現在のワードを大文字にします。
M-l	現在のワードを小文字にします。
^K	カーソルから行の末尾までを削除します。直前に指定された数値パラメータの値が現在のカーソル位置より小さい場合は、指定された位置の文字から現在のカーソル位置の文字までを削除します。直前に指定された数値パラメータの値が現在のカーソル位置より大きい場合は、現在のカーソル位置の文字から指定された位置の文字までを削除します。
^W	カーソル位置の文字からマークまでを抹消します。
M-p	カーソル位置からマークまでの領域をスタックにプッシュします。
kill	現在の行全体を抹消します。ユーザー定義の抹消文字は、 stty(1) コマンドで定義され、通常は^Gまたは@です。2つの抹消文字を連続して入力すると、そのあとのkill文字はすべて改行を意味します。プリンタ端末(印字式端末)を使用している場合に便利です。
^Y	行から最後に削除された項目を復元します。つまり、退避していた項目を行に戻します。
^L	改行を行い、現在の行を出力します。
M-^L	画面を消去します。
^@	NULL 文字。マークを設定します。
M-space	メタスペース文字。マークを設定します。
^J	復帰改行。現在の行を実行します。
^M	復帰。現在の行を実行します。
EOF	現在の行が NULL の場合のみ、end-of-file 文字 (通常は^D) をファイルの終わりとして処理します。

^P	直前のコマンドを取り出します。 ^P を入力するたびに、さらに前のコマンドにアクセスします。複数行にまたがるコマンドで最初の行でない場合は、1行戻ります。
M-[A	^P と同等です。
M-<	もっとも古いコマンド行の履歴を取り出します。
M->	もっとも新しいコマンド行の履歴を取り出します。
^N	次のコマンド行を取り出します。 ^N を入力するたびに、さらに次のコマンドにアクセスします。
M-[B	^N と同等です。
^Rstring	履歴をさかのぼって、 <i>string</i> を含むコマンド行を検索します。パラメータとして0を指定すると、順方向に検索します。 <i>string</i> の終わりは復帰改行 (RETURN または NEWLINE) で示します。 <i>string</i> の先頭に^が付加されると、その文字列で始まるコマンドだけを検索します。 <i>string</i> を省略すると、直前に指定した文字列を含む次のコマンド行にアクセスします。この場合、パラメータとして0を指定すると検索方向が逆になります。
^O	(Operate) 現在の行を実行し、現在の行に対する次の行を履歴ファイルから取り出します。
M-digits	エスケープ。数値パラメータを定義します。 <i>digits</i> は次のコマンドに対するパラメータとみなされます。パラメータを受け入れるコマンドは、 ^F 、 ^B 、 ERASE 、 ^C 、 ^D 、 ^K 、 ^R 、 ^P 、 ^N 、 ^J 、 M-. 、 M- 、 M-^J 、 M-_~ 、 M-= 、 M-b 、 M-c 、 M-d 、 M-f 、 M-h 、 M-l 、および M-^H です。
M-letter	(ソフトキー) 別名リストから、名前 <i>letter</i> の別名を検索します。 <i>letter</i> の別名が定義されている場合、入力待ち行列にその値を挿入します。 <i>letter</i> は、このセクションで示したいずれかのメタ関数を示す文字であってはいけません。
M-[letter	(ソフトキー) 別名リストから、名前 <i>letter</i> の別名を検索します。この名前の別名が定義されている場合、入力待ち行列にその値を挿入します。これは、多くの端末でファンクションキーをプログラムするために使用できます。
M-.	直前のコマンドの最後のワードを行に挿入します。数値パラメータが指定されると、最後のワードではなくその数値が示す位置のワードを挿入します。
M-_~	M-. と同じです。
M-*	現在のワードに対してファイル名生成を試行します。ワードがどのファイルにも一致しない場合や、特殊パターン文字を含んでいない場合、アスタリスクが付加されます。

M-ESC	このマニュアルページで説明したコマンド名補完またはファイル名補完。
^ITAB	このマニュアルページで説明したコマンド名補完またはファイル名補完を試行します。部分補完が起きた場合、これを繰り返すと、M-=を入力した場合と同様に動作します。一致が見つからない場合、またはスペース文字のあとに入力されている場合は、タブが挿入されます。
M-=	数値パラメータが前に付いていない場合、このマニュアルページで説明するように、一致するコマンドまたはファイル名のリストを生成します。それ以外の場合は、カーソルの下のワードが、もっとも新しく生成されたコマンドまたはファイルリストからの数値パラメータの値に対応する項目で置き換えられます。カーソルがワードの上にない場合、代わりにワードが挿入されます。
^U	次のコマンドのパラメータを4倍します。
\	次の文字をエスケープします。編集用文字、ユーザーが設定した消去文字、抹消文字、および割り込み文字(通常は^?)は、その前に\を指定すれば、コマンド行または検索文字列に入力できます。\ は、次の文字の編集機能(もしあれば)を無効にします。
M-^V	シェルのバージョンを表示します。
M-#	行の先頭が#でない場合、行の先頭と各復帰改行のあとに#が挿入され、行が入力されます。これによって、履歴ファイルにコメントを挿入できます。行の先頭が#である場合、#が削除され、各復帰改行のあとの#も1つずつ削除されます。

vi 編集モード

2つの入力モードがあります。初期状態では、コマンドを入力すると入力モードに移行します。編集するには、ユーザーは、エスケープ(ESC)キー(033)を入力して制御モードに移行し、訂正が必要な箇所にカーソルを移動してから、必要に応じて文字やワードを挿入または削除します。ほとんどの制御コマンドでは、コマンドの先頭に繰り返し数を指定できます。

ほとんどのシステムでは、viモードに移行した直後は標準処理が有効になります。端末の回線速度が1200ボー以上であり、制御文字が含まれているか、またはプロンプトが表示されてからの経過時間が1秒未満であれば、コマンドが再度表示されます。ESC文字を入力すると、コマンドの残りの部分に対する標準処理が終了し、ユーザーはコマンド行を変更できるようになります。この方式には、標準処理でrawモードの先行入力表示機能を利用できるという利点があります。

virawオプションも設定すると、端末の標準処理を常に無効にできます。このモードは、行の終わりの区切りを示す2つの代替記号をサポートしていないシステムでは暗黙的に設定されており、特定の端末で便利ことがあります。

入力編集コマンド

デフォルトでは、エディタは入力モードになります。

次の入力編集コマンドがサポートされています。

ERASE	ユーザーが <code>stty</code> コマンドで定義した消去文字。通常は <code>^H</code> または <code>#</code> 。前の 1 文字を削除します。
<code>^W</code>	直前の、空白で区切られたワードを削除します。一部のシステムでは、この機能には <code>viraw</code> オプションが必要です。
EOF	行の先頭の文字として使用すると、 <code>ignoreeof</code> オプションが設定されない限り、シェルが終了します。それ以外の場合、この文字は無視されます。
<code>lnext</code>	<code>stty(1)</code> によって定義した、ユーザー定義の次のリテラル文字。定義されていない場合は <code>^V</code> 。次の文字の編集機能がある場合はこれを無効にします。一部のシステムでは、この機能には <code>viraw</code> オプションが必要です。
<code>\</code>	次の ERASE または KILL 文字をエスケープします。
<code>^I TAB</code>	このマニュアルページで説明したコマンド名補完またはファイル名補完を試行し、入力モードに戻ります。部分補完が起きた場合、これを繰り返すと、制御モードから <code>=</code> を入力した場合と同様に動作します。一致が見つからない場合、またはスペース文字のあとに入力されている場合は、タブが挿入されます。

移動編集コマンド

移動編集コマンドはカーソルを移動します。

次の移動編集コマンドがサポートされています。

<code>[count]l</code>	カーソルを 1 文字だけ前方 (右) に移動します。
<code>[count][C</code>	カーソルを 1 文字だけ前方 (右) に移動します。
<code>[count]w</code>	英数字から構成されるワード 1 つ分だけカーソルを前方に移動します。
<code>[count]W</code>	空白のあとに続く次のワードの先頭にカーソルを移動します。
<code>[count]e</code>	カーソルをワードの終わりに移動します。
<code>[count]E</code>	空白で区切られた現在のワードの終わりにカーソルを移動します。
<code>[count]h</code>	カーソルを 1 文字だけ後方 (左) に移動します。
<code>[count][D</code>	カーソルを 1 文字だけ後方 (左) に移動します。
<code>[count]b</code>	カーソルを 1 ワードだけ後方に移動します。
<code>[count]B</code>	空白で区切られた直前のワードにカーソルを移動します。

[count]	<i>count</i> が示すカラムにカーソルを移動します。
[count]fc	現在の行で文字 <i>c</i> が次に現れる位置にカーソルを移動します。
[count]Fc	現在の行で文字 <i>c</i> が前に現れる位置にカーソルを移動します。
[count]tC	f と h を連続して実行した場合と同等です。
[count]Tc	F と l を連続して実行した場合と同等です。
[count];	直前の単一文字検索コマンドの f、F、t、または T を <i>count</i> の数だけ繰り返します。
[count],	直前の単一文字検索コマンドを <i>count</i> の数だけ逆方向で実行します。
0	カーソルを行の先頭に移動します。
^	カーソルを行の先頭に移動します。
[H	行に含まれる最初のブランク以外の文字にカーソルを移動します。
\$	カーソルを行の末尾に移動します。
[Y	カーソルを行の末尾に移動します。
%	現在の位置にある括弧記号 (、)、{、}、[、または] に対応する括弧記号にカーソルを移動します。このセクションで説明したどの文字の上にもカーソルが置かれていない場合は、いずれかの文字で最初に現れるものを行の残りから検索します。

検索編集コマンド

検索編集コマンドはコマンド履歴にアクセスします。

次の検索編集コマンドがサポートされています。

[count]k	直前のコマンドを取り出します。k を入力するたびに、さらに前のコマンドにアクセスします。
[count]-	直前のコマンドを取り出します。k を入力するたびに、さらに前のコマンドにアクセスします。 k と同等です。
[count][A	直前のコマンドを取り出します。k を入力するたびに、さらに前のコマンドにアクセスします。 k と同等です。
[count]j	次のコマンドを取り出します。j を入力するたびに、さらに次のコマンドにアクセスします。

- [*count*]+ 次のコマンドを取り出します。j を入力するたびに、さらに次のコマンドにアクセスします。
- j と同等です。
- [*count*][B 次のコマンドを取り出します。j を入力するたびに、さらに次のコマンドにアクセスします。
- j と同等です。
- [*count*]G コマンド番号 *count* を取り出します。デフォルトでは、もっとも古いコマンド履歴を取り出します。
- /string 履歴をさかのぼって、*string* を含むコマンドを検索します。*string* の終わりは復帰改行 (RETURN または NEWLINE) で示します。*string* の先頭に ^ が付加されると、その文字列で始まるコマンドだけを検索します。*string* が NULL の場合は、直前に指定された文字列を使用します。
- ?string 履歴を順方向に調べて、*string* を含むコマンドを検索します。*string* の終わりは復帰改行 (RETURN または NEWLINE) で示します。*string* の先頭に ^ が付加されると、その文字列で始まるコマンドだけを検索します。*string* が NULL の場合は、直前に指定された文字列を使用します。
- /と同じですが、順方向に検索します。
- n 直前の / または ? コマンドで指定されたパターンと一致する次のパターンを逆方向に検索します。
- N 直前の / または ? で指定されたパターンと一致する次のパターンを順方向に検索します。

テキスト変更編集コマンド

次のコマンドは行を変更します。

- a 入力モードに移行し、現在の文字のあとにテキストを入力します。
- A 行の終わりにテキストを追加します。*\$a* と同等です。
- [*count*]cmotion
c[*count*]motion 現在の文字から *motion* によりカーソルが移動する先までの文字を削除し、入力モードに移行します。*motion* が c の場合は、行全体を削除して入力モードに移行します。
- C 現在の文字から行の終わりまでを削除し、入力モードに移行します。*c\$* と同等です。

S	cc と同等です。
[count]s	入力モードで、カーソルの下にある文字を置き換えます。
D[count]dmotion	現在の文字から行の終わりまでを削除します。d\$ と同等です。
d[count]motion	現在の文字から <i>motion</i> によりカーソルが移動する先までの文字を削除します。 <i>motion</i> が d の場合は、行全体を削除します。
i	入力モードに移行し、現在の文字の前にテキストを入力します。
I	行の先頭にテキストを挿入します。0i と同等です。
[count]P	カーソルの前に、直前のテキスト変更を挿入します。
[count]p	カーソルのあとに、直前のテキスト変更を挿入します。
R	入力モードに移行して、画面上の文字を重ね打ちした文字に置き換えます。
[count]rc	現在のカーソル位置から始まる <i>count</i> 個の文字を <i>c</i> に置き換え、カーソルの位置を進めます。
[count]x	現在の文字を削除します。
[count]X	直前の文字を削除します。
[count].	直前のテキスト変更コマンドを繰り返します。
[count]~	現在のカーソル位置から始まる <i>count</i> 個の文字を、大文字の場合は小文字に、小文字の場合は大文字に変換して、カーソルの位置を進めます。
[count]_	直前のコマンドの <i>count</i> 個のワードを付加し、入力モードに移行します。 <i>count</i> を省略すると、最後のワードを使用します。
*	* を現在のワードのあとに付加し、ファイル名を生成しようとします。一致するものが見つからない場合は、ベルを鳴らします。見つかった場合は、ワードを一致したパターンで置換し、入力モードに移行します。
\	このマニュアルページで説明したコマンド名補完またはファイル名補完。

その他の編集コマンド

次のその他の編集コマンドがサポートされています。

[count]ymotion	
y[count]motion	現在の文字から <i>motion</i> によりカーソルが移動する先の文字までを退避します。退避した文字を削除用バッファーに入れます。テキストとカーソル位置は変更されません。

yy	現在の行を退避します。
Y	現在のカーソル位置から行の末尾まで、現在の行を退避します。 <i>y\$</i> と同等です。
u	直前のテキスト変更コマンドを取り消します。
U	現在の行で実行されたテキスト変更コマンドをすべて取り消します。
[<i>count</i>]V	コマンド <pre>hist -e \${VISUAL:-\${EDITOR:-vi}} <i>count</i></pre> を入力バッファに返します。<i>count</i> を省略した場合、現在の行が使用されます。
^L	改行を行い、現在の行を出力します。このコマンドは制御モードでのみ機能します。
^J	復帰改行。モードに関係なく、現在の行を実行します。
^M	復帰。モードに関係なく、現在の行を実行します。
#	コマンドの先頭文字が#であれば、その#および復帰改行に続くそれぞれの#を削除します。 先頭文字がほかの文字であれば、#をコマンドの各行の先頭に挿入後、行を送ります。 このコマンドは、現在の行をコメントとして履歴に挿入する場合や、履歴ファイルに含まれる以前のコメント化されたコマンドをコメント解除する場合に便利です。
[<i>count</i>]=	<i>count</i> が指定されていない場合、このマニュアルページで説明するように、一致するコマンドまたはファイル名のリストを生成します。 指定されている場合は、現在カーソルが置かれているワードを、もっとも新しく生成したコマンドまたはファイルリストからの <i>count</i> 項目に置き換えます。カーソルがワードの上になければ、現在のカーソル位置のあとに挿入されます。
@ <i>letter</i>	別名リストから、名前 <i>letter</i> の別名を検索します。この名前の別名が定義されている場合、処理用の入力待ち行列にその値を挿入します。
^V	シェルのバージョンを表示します。

組み込みコマンド

次の単純コマンドは、シェルプロセス中で実行されます。入出力のリダイレクトが可能です。特に断りのないかぎり、出力はファイル記述子1上に書き込まれ、構文エラーがなければ終了ステータスは0です。.:、true、false、echo、newgrp、およびloginを除くすべての組み込みコマンドで、オプションの終わりを示す--を受け入れます。また、オプション--manを、マニュアルページを標準エラーに出力するように求める要求と解釈し、-?を、使用法に関するメッセージを標準エラーに出力するように求めるヘルプ要求と解釈します。

1つまたは2つの++記号が先頭に付加されているコマンドは特殊組み込みコマンドであり、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

+ : [arg ...]

パラメータの展開だけを行います。

+ . name [arg ...]

name が *function name* 予約語構文で定義された関数である場合、この関数は、現在の環境で実行されます (*name()* 構文で定義されている場合と同様)。それ以外の場合で *name* がファイルを示しているときは、ファイルの全体が読み取られ、コマンドは現在のシェル環境で実行されます。PATHによって指定された検索パスを用いて、ファイルを含むディレクトリを探します。引数の *arg* が指定されている場合、.コマンドの処理中にこの引数が定位置パラメータになり、終了すると元の定位置パラメータが復元されます。そうでない場合、定位置パラメータは変更されません。終了ステータスは、最後に実行されたコマンドの終了ステータスです。

++ alias [-ptx] [name[=value]] ...

alias 引数なしの場合、このコマンドは標準出力上に *name=value* という形式の別名のリストを表示します。-p オプションを指定すると、それぞれの別名の前に *alias* というワードが挿入されます。1つまたは複数の引数が指定された場合、*value* が指定された *name* ごとに、*alias* が定義されます。*value* の末尾にスペースがあると、次のワードが別名置換指定かどうかをチェックします。廃止された -t オプションは、検索済みの別名を設定または一覧表示します。検索済み別名の値は、指定した *name* に対応する完全パス名になります。PATHの値を再設定するとこの値は未定義になりますが、別名は検索済みのままです。-t オプションを省略すると、*value* が指定されていない引数リスト内の各 *name* について、別名の名前と値を表示します。廃止された -x オプションは効果がありませ

ん。*name*が指定されているが、*value*は指定されておらず、*name*に対する別名も定義されていない場合は、終了ステータスは0以外になります。

bg [*job*...]

このコマンドを使用できるのは、ジョブ制御をサポートするシステム上だけです。指定された各*job*をバックグラウンドで実行します。*job*が省略された場合は、現在のジョブをバックグラウンドで実行します。*job*の形式については、このマニュアルページの「ジョブ」セクションを参照してください。

+ break [*n*]

for ループ、while ループ、until ループ、または select ループがあれば終了します。*n* を指定すると、*n* レベル分だけループを終了します。

builtin [-ds] [-f *file*] [*name*...]

name と -f オプションを指定しない場合、組み込みコマンドは標準出力に出力されます。-s オプションを指定すると、特殊組み込みコマンドだけが出力されます。それ以外の場合、それぞれの *name* は、組み込みコマンドの名前をベース名として持つパス名を表します。エントリポイントの関数名は、組み込みコマンド名の前に *b* を付けることによって決定されます。組み込みコマンド *mycommand* に対する ISO C/C++ プロトタイプは *bmycommand(int argc, char *argv[], void *context)* です。ここで、*argv* は *argc* 要素の配列であり、*context* は、`<ast/shell.h>` に記述された `Shell_t` 構造体への省略可能なポインタです。特殊組み込みコマンドは、パス名にバインドしたり、削除したりできません。-d オプションを指定すると、それぞれの特殊組み込みコマンドが削除されます。動的読み込みをサポートするシステムでは、-f オプションは、組み込みコマンドのコードを含んだ共有ライブラリの名前を指定します。共有ライブラリの接頭辞または接尾辞（それぞれシステムに応じて異なる）、あるいはその両方を省略できます。ライブラリが読み込まれると、そのシンボルを後続の builtin の呼び出しで使えるようになります。複数のライブラリを指定するには、builtin コマンドを別個に呼び出します。ライブラリは、指定された順序とは逆の順序で検索されます。ライブラリが読み込まれると、ライブラリ内で `lib_init()` という名前の関数が検索され、0 の引数を付けてこの関数が呼び出されます。

cd [-LP] [*arg*]

cd [-LP] *old new*

このコマンドには2つの形式があります。

第1の形式は、現在のディレクトリを *arg* に変更します。*arg* が - の場合、ディレクトリを直前のディレクトリに変更します。シェル変数 HOME の値がデフォルトの *arg* になります。PWD 変数は、現在のディレクトリに設定されます。シェル変数 CDPATH は、*arg* を含むディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:)で区切ります。デフォルトのパスは空の文字列です(現在のディレクトリを指定)。現在のディレクトリは空のパス名で指定します。このパス名は、等号の直後か、パスリスト内にある区切り文字のコロンの間に指定します。*arg* の先頭文字が / の場合、検索パスは使用しません。それ以外の場合は、パス中の各ディレクトリで *arg* を検索します。

cd の第 2 の形式は、PWD 中の現在のディレクトリ名に含まれる *old* という文字列を *new* という文字列に置換し、この新規のディレクトリへ変更しようとしします。デフォルトでは、ディレクトリ名を検索するときに、シンボリックリンク名は入力されたとおりに扱われます。これは、-L オプションと同等です。-P オプションを指定すると、ディレクトリを決定するときに、シンボリックリンクが解決されます。コマンド行の -L または -P の最後の指定によって、使用されるメソッドが決まります。cd コマンドは rksh では実行できません。

command [-pvVx] *name* [*arg*...]

-v または -V オプションを指定しない場合、*arg* で指定された引数で *name* が実行されます。

-p オプションを指定すると、PATH の値で定義されたパスではなく、デフォルトのパスが検索されます。*name* を探すときに、関数は検索されません。さらに、*name* が特殊組み込みコマンドを指す場合、先頭にダガーが付いた特殊なプロパティは考慮されません。たとえば、`redirect='command exec'` という定義済みの別名を使用すると、無効なリダイレクションが指定された場合にスクリプトが終了しません。

-x オプションを指定すると、引数が多すぎるためにコマンドの実行が `errno E2BIG` というエラーになる場合に、シェルは、各呼び出しで引数のサブセットを指定して、コマンド *name* を複数回呼び出します。複数の引数に展開する最初のワードの前の引数と、複数の引数に展開する最後のワードのあとの引数は、各呼び出しに渡されます。終了ステータスは呼び出しの最大終了ステータスです。

-v オプションを指定した場合、**command** は、このセクションで説明する `whence` 組み込みコマンドと同等になります。-V オプションを指定した場合、**command** は `whence -v` と同じように機能します。

+continue [*n*]

for ループ、while ループ、until ループ、または select ループの次の繰り返しを実行します。*n* を指定すると、*n* 番目のループから実行します。

disown [*job*...]

ログインシェルが終了するときに、シェルが、指定された各 *job*、または *job* が省略されている場合はすべての活動中のジョブに HUP シグナルを送信しません。

echo [*arg*...]

最初の *arg* の先頭が - ではなく、どの引数にもバックスラッシュ (\) が含まれていない場合、各引数をスペース文字で区切り、復帰改行で終了する形式で出力します。それ以外の場合は、echo の動作はシステムによって異なるため、このセクションで説明する `print` または `printf` を使用するようにしてください。使用法と説明については、[echo\(1\)](#) を参照してください。

+eval [*arg*...]

引数をシェルへの入力として読み取り、生成されるコマンドを実行します。

+exec [-c] [-a name...] [arg...]

arg を指定すると、このシェルの代わりに、引数で指定されたコマンドを (新規プロセスは生成せずに) 実行します。-c オプションを指定すると、exec の呼び出しに関連付けられた変数代入を適用する前に、環境がクリアされます。-a オプションを指定すると、最初の *arg* ではなく *name* が、新しいプロセスの *argv[0]* になります。入出力引数が指定可能で、現在のプロセスに影響を及ぼす場合があります。*arg* を指定しない場合は、ファイル記述子が、入出力リダイレクトリストの指定どおりに変更されることになります。この場合、この機能によりオープンされた 2 より大きい番号のファイル記述子は、別のプログラムを起動するとクローズされます。

+exit [n]

シェルを、*n* が示す終了ステータスで終了させます。具体的には、指定した値の最下位 8 ビットが終了ステータスの値となります。*n* を省略すると、最後に実行されたコマンドの終了ステータスがシェルの終了ステータスになります。ignoreeof オプションが有効になっているシェルを除き、ファイルの終わりを検出した場合もシェルが終了します。set を参照してください。

++export [-p] [name=value]] ...

name を指定しない場合、エクスポート属性を持つ各変数の名前と値は、再入力可能な形式で値がクォートされて出力されます。-p オプションを指定すると、それぞれの変数の前に export というワードが挿入されます。それ以外の場合は、指定された *name* に、あとで実行されるコマンドの環境へ自動的にエクスポートされるようにマークが付きます。

false

何も行わず、1 を返して終了します。until とともに使用すると無限ループになります。

fg [job...]

このコマンドを使用できるのは、ジョブ制御をサポートするシステム上だけです。指定された各 *job* を、指定された順序でフォアグラウンドで実行し、待機させます。*job* が指定されないと、現在のジョブをフォアグラウンドで実行します。*job* の記述形式については、「ジョブ」を参照してください。

getconf [name [pathname]]

name で指定された構成パラメータの現在の値を出力します。構成パラメータは、IEEE POSIX 1003.1 および IEEE POSIX 1003.2 標準で定義されます。pathconf(2) および sysconf(3C) を参照してください。

pathname 引数は、値がファイルシステムでの場所によって異なるパラメータで必要です。引数を指定しない場合、getconf は、現在の構成パラメータの名前と値を出力します。パス名 / は、*pathname* を必要とする各パラメータで使用されます。

getopts [-a name] optstring vname [arg...]

arg が正当なオプションを示しているかどうかをチェックします。*arg* を省略すると、定位置パラメータが使用されます。オプション引数は + または - で始まりま

す。+ や - 以外の文字で始まっているオプション、または引数が -- である場合、オプションは終了します。+ で始まっているオプションは、*optstring* が + で始まっている場合にのみ認識されます。*optstring* には、*getopts* が認識する文字を記述します。文字のあとに : が続く場合、そのオプションには引数があるとみなされます。オプションと引数とは空白で区切ることができます。オプション -? を指定すると、*getopts* は、標準エラーに使用法に関するメッセージを生成します。-a オプションを使用すると、使用法に関するメッセージに使用する名前を指定できます。これはデフォルトで \$0 です。*getopts* は、呼び出されるごとに、次に見つかったオプション文字を変数 *vname* 内に格納します。*arg* が + で始まっている場合、オプション文字の前に + が付けられます。次の *arg* のインデックスは *OPTIND* に格納されます。オプション引数がある場合は *OPTARG* に格納されます。*optstring* の先頭に : がある場合は、*getopts* は無効なオプション文字を *OPTARG* に格納し、*vname* を ? (未定義のオプションが指定された場合) または : (必要なオプション引数が省略されている場合) に設定します。*getopts* はエラーメッセージを表示します。オプションがなくなると、終了ステータスは 0 以外になります。:、+、-、?、[、および] のオプションを指定する方法はありません。オプション # は、最初のオプションとしてのみ指定できます。

```
hist [-e ename] [-nlr] [first [last ]]
```

```
hist -s [old=new] [command]
```

第1の形式では、端末から最近入力された *HISTSIZE* 個のコマンドの中から、*first* から *last* までの範囲のコマンドを選択します。*first* と *last* の両引数は、数値または文字列で指定できます。指定された文字列で始まる最新のコマンドを見つけるには、文字列を使用します。負の数値は、現在のコマンド番号からのオフセットとなります。-l オプションを指定すると、標準出力上にコマンドを一覧表示します。指定しないと、これらのキーボードコマンドの入ったファイル上で *ename* というエディタプログラムを起動します。*ename* を指定しない場合、変数 *HISTEDIT* の値が使用されます。*HISTEDIT* が設定されていない場合、*FCEDIT* (デフォルトは /bin/ed) がエディタとして使用されます。編集が完了すると、変更が保存されていれば、編集されたコマンドが実行されます。*last* を指定しない場合、*first* と同一値に設定されます。*first* を指定しない場合、デフォルトは、編集については直前のコマンドに、一覧表示については -16 になります。-r オプションはコマンドの順序を逆にします。-n オプションは一覧表示時にコマンド番号の出力を抑止します。第2の形式では、*command* は、このセクションで説明した *first* と解釈され、デフォルトで最後に実行されたコマンドに設定されます。*old=new* の省略可能な置換を行なったあとで、結果として得られるコマンドが実行されます。

```
jobs -lnp [job ...]
```

指定された各ジョブに関する情報を一覧表示します。*job* 引数を省略すると、活動中のジョブすべてに関する情報を一覧表示します。-l オプションは、通常の情報に加えてプロセス ID も表示します。-n オプションは、前回通知を受けたあとに停止または終了したジョブだけを表示します。-p オプションは、プロセスグループだけを表示します。*job* の記述形式については、「ジョブ」を参照してください。

```
kill [-s signame] job ...
```

```
kill [-n signum] job ...
```

```
kill -l [sig ...]
```

TERM(終了)シグナルまたは指定されたシグナルのいずれかを、指定されたジョブまたはプロセスに送信します。シグナルは、-n オプションの場合には番号で、-s オプションの場合には名前で指定されます。名前は、<signal.h> の定義から SIG という接頭辞を取り除いたものになります。ただし SIGCLD は CHLD という名前になります。下位互換性を確保するため、n および s を省略でき、番号または名前を - の直後に指定できます。送信するシグナルが TERM(終了) または HUP(ハングアップ) の場合、停止中のジョブまたはプロセスには CONT(継続) シグナルを送信します。*job* という引数は、活動中のジョブではないプロセスのプロセス ID を指定することもできます。*job* の記述形式については、「ジョブ」を参照してください。第3の形式の kill -l では、*sig* が指定されていない場合、シグナル名が一覧表示されます。それ以外の場合は、名前である *sig* ごとに、対応するシグナル番号が一覧表示され、番号である *sig* ごとに、*sig* の下位 8 ビットに対応するシグナル名が一覧表示されます。

```
let [arg ...]
```

各 *arg* は、評価の対象となる個々の算術式を表します。算術式の評価については、このマニュアルページの「算術評価」セクションを参照してください。終了ステータスは、最後の式の値が 0 以外の場合は 0 で、0 の場合には 1 です。

```
+newgrp [arg ...]
```

exec /bin/newgrp *arg* ... と同機能です。

```
print [-Renprs] [-u unit] [-f format] [arg ...]
```

オプションを指定しない場合、または - や -- のオプションを指定した場合、標準出力にそれぞれの *arg* が出力されます。-f オプションを指定すると、引数は、printf の記述に従って出力されます。この場合、e、n、r、R オプションが無視されます。それ以外の場合は、-R または -r が指定されていないかぎり、次のエスケープ規則が適用されます。

- \a 警告文字 (ASCII 07)
- \b バックスペース文字 (ASCII 010)
- \c print は、引数をこれ以上処理せず、復帰改行を追加せずに終了します。
- \f 用紙送り文字 (ASCII 014)
- \n 復帰改行文字 (ASCII 012)
- \r 復帰文字 (ASCII 015)
- \t タブ文字 (ASCII 011)
- \v 垂直タブ文字 (ASCII 013)
- \E エスケープ文字 (ASCII 033)
- \\ バックスラッシュ文字 \

`\0x` x で指定された 1、2、または 3 桁の 8 進数文字列で定義された文字

`-R` オプションは、`-n` を除く後続の引数およびオプションすべてを表示します。`-e` を指定すると、エスケープ規則が適用されます。これがデフォルトの動作です。前の `-r` の効果を逆転させます。`-p` オプションを指定すると、引数は、標準出力ではなく `|&` で生成されたプロセスのパイプに書き込まれます。`-s` オプションを指定すると、引数は、標準出力ではなく履歴ファイルに書き込まれます。`-u` オプションを使用すると、出力を格納する 1 桁のファイル記述子ユニット番号 *unit* を指定できます。デフォルト値は 1 です。`-n` オプションを指定した場合、復帰改行は出力に追加されません。

`printf format[arg...]`

引数 *arg* は、書式文字列 *format* に関連付けられた ANSI-C 書式設定規則に従って、標準出力に出力されます。引数の数が書式指定の数を超えた場合、書式文字列が、残りの引数の書式設定に再使用されます。次の拡張機能も使用できます。`%s` の代わりに `%b` 書式を使用すると、対応する *arg* 内のエスケープシーケンスを `print` の記述に従って展開できます。`%B` オプションを指定すると、各引数が変数名として扱われ、変数のバイナリ値が出力されます。これは、`b` の属性を持つ変数にもっとも役立ちます。`%H` 形式を `%s` の代わりに使用すると、HTML や XML で特殊な *arg* 内の文字をそのエンティティ名として出力できます。`%s` の書式の代わりに `%P` を使用すると、*arg* を拡張正規表現として解釈し、シェルパターンとして出力できます。`%s` の書式の代わりに `%R` を使用すると、*arg* をシェルパターンとして解釈し、拡張正規表現として出力できます。`%s` の書式の代わりに `%q` を使用すると、結果として得られる文字列を、シェルへの再入力に使用できるような形式にクォートできます。`%(date-format)T` の書式を使用すると、引数を日付/時間文字列として扱い、`date(1)` コマンドで定義されている *date-format* に従って日付/時間を書式設定できます。`%z` の書式は値が 0 のバイトを出力します。`%d` の書式の精度フィールドのあとには、`.` と出力の底を続けることができます。この場合、`#` フラグ文字を指定することにより、底となる数字が前に置かれるようになります。`#` フラグを、出力の底なしで *d* 指定子とともに使用すると、単位を示す `kMGtPE` のいずれかの接尾辞とともに、出力が 1000 の累乗の単位で表示されます。`#` フラグを、*i* 指定子とともに使用すると、単位を示す `KiMiGiTiPiEi` のいずれかの接尾辞とともに、出力が 1024 の累乗の単位で表示されます。指定したフィールド幅内で出力が中央になるように調整する `=` フラグが追加されています。

`pwd [-LP]`

現在の作業ディレクトリの値を出力します。`-L` オプションはデフォルトです。現在のディレクトリの論理名を出力します。`-P` オプションが指定された場合、すべてのシンボリックリンクが名前から解決されます。コマンド行の `-L` または `-P` の最後の指定によって、使用されるメソッドが決まります。

`read [-Aprs] [-d delim] [-n n] [[-N n] [[-t timeout] [-u unit] [vname?prompt] [vname...]]`

シェルの入力メカニズムです。1 つの行を読み取り、IFS が示す文字を、区切り文字として使用して、行の内容をいくつかのフィールドに分割します。エス

ケープ文字\`\`は、次の文字の特別な意味または行の継続に関する意味を取り除くために使用します。`-d` オプションを指定すると、読み取りは復帰改行ではなく `delim` の最初の文字まで続けられます。`-n` オプションを指定すると、行全体ではなく、最高で `n` バイトが読み取られますが、低速デバイスから読み取る場合には、何らかの文字を読み取ったらずぐに返ります。`-N` オプションを指定すると、ファイルの終わりにならないかぎり、または `-t` オプションのために読み取りがタイムアウトしないかぎり、正確に `n` バイトを読み取ります。`-r` で指定する raw モードでは、\`\` が持つこの特殊な意味は無視されます。第1フィールドを1番目の `vname` に、第2フィールドを2番目の `vname` にという順番で割り当ていき、余ったフィールドがあれば最後の `vname` に割り当てます。`vname` にバイナリ属性があり、`-n` または `-N` が指定された場合、読み取られたバイトは、直接変数に格納されます。`-v` が指定された場合、端末装置から読み取るときに、1番目の `vname` の値がデフォルト値として使用されます。`-A` オプションを指定すると、変数 `vname` が設定解除され、読み取られた各フィールドは、インデックス付き配列 `vname` の連続した要素に格納されます。`-p` オプションは、シェルが `|&` を使用して生成したプロセスの入力パイプから入力行を取り出します。`-s` オプションは、入力をコマンドとして履歴ファイルに保存します。`-u` オプションは、読み取り元となるファイル記述子番号を1桁の数値 `unit` で指定します。ファイル記述子は、`exec` という特殊組み込みコマンドで開くことができます。ユニット `n` のデフォルト値は0です。`-t` オプションは、端末またはパイプから読み取るときに、秒数でタイムアウトを指定します。`vname` を省略すると、REPLY の値をデフォルトとして使用します。`-p` オプションが指定されていてファイルの終わりを検出すると、このプロセスをクリアし別のプロセスを作成可能にします。最初の引数が?を含んでいると、シェルが対話型のとき、このワードの残りを標準エラーに対するプロンプトとして使用します。ファイルの終わりに到達するか、読み取りがタイムアウトしないかぎり、終了ステータスは0です。

`++readonly [-p] [ vname[=value]] ...`

`vname` を指定しない場合、読み取り専用属性を持つ各変数の名前と値は、再入力可能な形式で値がクォートされて出力されます。`-p` オプションを指定すると、それぞれの変数の前に `readonly` というワードが挿入されます。それ以外の場合は、指定された `vname` は読み取り専用としてマークされ、これらの名前は、以降の代入で変更できなくなります。

`++return [n]`

シェル関数またはスクリプトを、`n` で指定された終了ステータスで呼び出し側スクリプトに戻します。具体的には、指定した値の最下位8ビットが終了ステータスの値となります。`n` を省略すると、戻りステータスは最後に実行されたコマンドの戻りステータスになります。関数やスクリプトの外で `return` を呼び出すと、`exit` と同じ動作を行います。

`++set [ ±BCGabefhkmnoprstuvx] [±o [ option ] ] ... [ ±A vname] [arg...]`

`set` コマンドには、次のオプションを設定できます。

`-a`

定義される後続の変数すべてを自動的にエクスポートします。

- A
配列の代入。変数 *vname* の設定を解除し、*arg* リストから順々に値を代入します。*+A* を使用すると、最初に変数 *vname* の設定解除は行いません。
- b
バックグラウンドジョブの状態が変更されたときに、次のプロンプトを待機せずに、すぐにジョブ完了メッセージを出力します。
- B
中括弧パターンフィールドの生成を有効にします。これはデフォルトの動作です。
- C
シェルのリダイレクト (>) によって既存のファイルが上書きされるのを防ぎます。作成されたファイルは、*O_EXCL* モードで開かれます。これが有効なときにファイルを切り捨てるには、>| が必要です。
- e
コマンドの終了ステータスが 0 でない場合、*ERR* トラップ (設定されていれば) を実行し、終了します。このモードは、プロファイル読み取り時は無効です。
- f
ファイル名を生成しないようにします。
- G
ファイル名生成に使用した場合に、パターン **** 単独で、ファイルおよび 0 個以上のディレクトリおよびサブディレクトリに一致します。/ を続けると、ディレクトリおよびサブディレクトリだけが一致します。
- h
各コマンドは、最初に検出された時点で、検索済み別名になります。
- k
廃止。コマンド名に先行するものだけでなく、すべての変数代入引数をコマンドの環境に格納します。
- m
バックグラウンドジョブを個別のプロセスグループで実行し、完了時にメッセージを 1 行表示します。バックグラウンドジョブの終了ステータスは完了メッセージで報告されます。ジョブ制御を備えたシステムでは、このオプションは対話型シェルに対して自動的に有効になります。
- n
コマンドを読み取り、構文エラーがないかチェックします。実行は行いません。対話型シェルに対しては無視されます。
- o
オプション名が指定されていない場合、オプションとその現在の設定のリストが、標準出力に書き込まれます。*+* を付けて呼び出された場合、シェルに再

入力して設定を復元できる形式で、オプションが書き込まれます。このオプションを繰り返して、複数のオプションを有効または無効にすることができます。

このフラグのあとに指定する引数は、次のいずれかのオプション名です。

allexport

-a と同じです。

bgnice

バックグラウンドジョブをすべて低い優先度で実行します。これはデフォルトモードです。

braceexpand

-B と同じです。

emacs

コマンド入力用に、**emacs** 形式のインラインエディタを起動します。

errexit

-e と同じです。

globstar

-G と同じです。

gmacs

コマンド入力用に、**gmacs** 形式のインラインエディタを起動します。

ignoreeof

ファイルの終わりを検出してもシェルは終了しません。終了させるには **exit** コマンドを使用する必要があります。

keyword

-k と同じです。

markdirs

ファイル名生成によって生成されるディレクトリ名には、すべて最後に / を付加します。

monitor

-m と同じです。

multiline

組み込みエディタで、行の長さが画面の幅を超える場合に複数の行を使用します。これはすべての端末で機能するわけではありません。

noclobber

-c と同じです。

noexec

-n と同じです。

noglob

-f と同じです。

nolog

履歴ファイルに関数定義を保存しません。

notify

-b と同じです。

nounset

-u と同じです。

pipefail

パイプラインのすべてのコンポーネントが完了するまで、パイプラインは完了しません。戻り値は、最後に失敗した 0 以外のコマンドの値か、コマンドが失敗しなかった場合は 0 です。

privileged

-p と同じです。

showme

有効にした場合、セミコロン (;) が前に付いた単純コマンドまたはパイプラインは、**xtrace** オプションが有効になっている場合と同様に表示されますが、実行されません。それ以外の場合、前に付いた ; は無視されます。

trackall

-h と同じです。

verbose

-v と同じです。

vi

vi 形式のインラインエディタの挿入モードになり、033 というエスケープ文字を押すまで継続されます。これにより制御モードになります。Return で行を送信します。

viraw

各文字を **vi** モードで入力されたときと同様に処理します。

xtrace

-x と同じです。

オプション名を指定しないと、現在のオプション設定を表示します。

-p

\$HOME/.profile ファイルを処理しないようにし、**ENV** ファイルの代わりに **/etc/suid_profile** ファイルを使用します。このモードは、実効ユーザー ID (実効グループ ID) が実ユーザー ID (実グループ ID) と等しくないときには必ず有効になります。このモードを無効にすると、実効ユーザー ID が実ユーザー ID に、実効グループ ID が実グループ ID にそれぞれ設定されます。

- r
制限付きシェルを有効にします。このオプションは、一度設定すると設定解除できません。
- s
定位置パラメータを辞書編集方式の順にソートします。
- t
廃止。1つのコマンドを読み取り、実行し、終了します。
- u
置換を行う際に、設定されていないパラメータをエラーとして扱います。
- v
シェルへの入力行の読み取り時に、その内容を表示出力します。
- x
コマンドおよび引数の実行時に、その内容を表示出力します。
- どのオプションも変更しません。これは、\$1 を- で始まる値に設定する際に便利です。このオプションのあとに引数がない場合、定位置パラメータが設定解除されます。

廃止された機能として、最初の *arg* が - である場合は、-x および -v オプションが無効になり、次の *arg* が最初の引数として扱われます。- の代わりに + を使用すると、これらのオプションは無効になります。これらのオプションはシェル起動時にも使用できます。現在のオプションセットは、\$- で見つけられます。-A が指定されていないかぎり、残りの引数は定位置パラメータであり、\$1 \$2 に順に割り当てられます。引数が指定されていない場合は、すべての変数の名前と値が標準出力に出力されます。

+shift [*n*]

\$*n*+1 ... からの定位置パラメータの名前は、\$1 ... に変更されます。デフォルトの *n* は 1 です。パラメータ *n* に指定できる値は、評価結果が \$# 以下の負でない数になる算術式です。

+trap -p [*action*] [*sig*] ...

-p オプションを指定すると、引数として指定された、各トラップに関連付けられているトラップアクションが、適切にクォートされて出力されます。それ以外の場合は、*action* は、シェルがシグナル *sig* を受け取ったときに、eval への引数である場合と同様に処理されます。各 *sig* は、数値またはシグナルの名前です。trap コマンドは、シグナル番号の順序で実行されます。現在のシェルへの入力時に無視されたシグナルにトラップを設定しようとしても無効となります。action を省略して、最初の *sig* が数値である場合、または *action* が - である場合は、各 *sig* 用のトラップが元の値に再設定されます。action が NULL 文字列の場合、シェルおよびシェルが呼び出したコマンドは、このシグナルを無視します。sig が ERR の場合は、コマンドが 0 以外の終了ステータスで終わると必ず action が実行されます。sig が DEBUG の場合は、各コマンドの前に action が実行さ

れます。*action* が実行されているとき、変数 *.sh.command* には現在のコマンド行の内容が含まれます。*sig* が 0 または EXIT であり、*function name* 構文で定義されたある関数の内部で *trap* 文が実行された場合、コマンド *action* は関数の完了後に実行されます。*sig* が 0 または EXIT であり、*trap* が関数の外側で設定されている場合、コマンド *action* はシェルの終了時に実行されます。*sig* が KEYBD である場合、*emacs*、*gmacs*、または *vi* モードのときにキーが読み取られると必ず、*action* が実行されます。引数なしの *trap* コマンドは、コマンドの一覧を各々が対応しているシグナル番号とともに表示します。

true

何も行わず、0 を返して終了します。*while* とともに使用すると無限ループになります。

++typeset [±AHflabnprtux] [±EFLRZi[n]] [*vname*[=*value*]]

シェル変数と関数の属性と値を設定します。*function name* 構文で定義された関数内部で呼び出されると、変数 *vname* の新しいインスタンスが作成され、関数の完了時に変数の値と型が復元されます。

- の代わりに + を使用すると、これらのオプションは無効になります。*vname* 引数が指定されていない場合、変数の *vname* (および必要な場合は *value*) のリストが出力されます。- の代わりに + を使用すると、値は出力されません。-p オプションを指定すると、オプションの名前ではなく *typeset* とオプション文字が、それぞれの名前の前に出力されます。-p 以外のオプションを指定した場合は、指定されたすべてのオプションを持つ変数だけが出力されます。それ以外の場合、*vname* および属性を持つすべての変数の *attributes* が出力されます。

このコマンドには、次の属性を指定できます。

- a *vname* をインデックス付き配列として宣言します。複合変数の代入を除き、これは省略可能です。
- A *vname* を連想配列として宣言します。添字は、算術式ではなく文字列です。
- b 変数には、何バイトのデータでも保持できます。データはテキストにすることもバイナリにすることもできます。値は、データの base64 エンコーディングによって表されます。-z も指定した場合、バッファ内のデータのサイズ (バイト数) は、-z に関連付けられたサイズで決められます。代入された base64 文字列がこのデータサイズを超えた場合、切り捨てられます。それ以外の場合、値が 0 であるバイトが入力されます。データの base64 エンコーディングではなく、このバッファ内の実際のデータを出力するには、*printf* の書式である %B を使用できます。
- E *vname* を倍精度浮動小数点数として宣言します。*n* が 0 以外の場合、*vname* の展開時に使用される有効数字の桁数を定義します。それ以外の場合、10 桁の有効数字が使用されます。

-f 名前は、変数名ではなく関数名を指します。割り当てを実行することはできず、ほかの有効なオプションは -t、-u、および -x だけです。-t オプションは、この機能の実行トレースを有効にします。-u フラグは、この関数に「未定義」を示すマークを付けます。関数が参照されると、関数定義を見つけるために FPATH 変数が検索されます。-f 以外のオプションを指定しない場合は、関数定義が標準出力上に表示されます。+f を指定した場合、関数名のあとに、この関数が定義されたファイル(ある場合)の行番号とパス名を示したシェルコメントが記された行が表示されます。

-f では、-i 属性を指定できません。

-F *vname* を倍精度浮動小数点数として宣言します。*n* が 0 以外の場合、*vname* の展開時に使用される小数点以下の位数を定義します。それ以外の場合は、小数点以下 10 位が使用されます。

-H このオプションは、UNIX 以外のマシン上で、UNIX とホスト名ファイルとのマッピング情報を提供します。

-i *vname* を内部で整数を表す値として宣言します。代入の右側は、整数に代入する場合、算術式として評価されます。*n* が 0 以外であればその値を底として定義します。0 の場合、出力の底は 10 です。

-i 属性は、-R、-L、-Z、または -f と同時に指定できません。

-l 大文字をすべて小文字に変換します。大文字オプションの -u は無効になります。

-L 左詰めを行い、先行する空白文字を *value* から取り除きます。*n* は、0 以外であればフィールドの幅を定義します。0 の場合、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ右側にブランクが詰められ、長ければ切り捨てられます。-R オプションは無効になります。

-L では、-i 属性を指定できません。

-n 変数 *vname* の値で名前が定義されている変数への参照として、*vname* を宣言します。これは通常、名前が引数として渡された関数内部の変数を参照するために使用されます。

-R 右詰めを行い、先行する空白文字を挿入します。*n* は、0 以外であればフィールドの幅を定義します。0 の場合、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ左側にブランクが詰められ、長ければ終端が切り捨てられます。-L オプションは無効になります。

-R では、-i 属性を指定できません。

- r 指定された *vname* は読み取り専用としてマークされ、これらの名前は、以降の代入で変更できなくなります。
 - t 変数にタグを付けます。タグはユーザーが定義可能で、シェルに対して特別な意味を持ちません。
 - u 小文字をすべて大文字に変換します。小文字オプションの *-l* は無効になります。
 - x 指定された *vname* に対し、あとで実行されるコマンドの環境へ自動的にエクスポートされるようにマークを付けます。名前に *.* が含まれる変数は、エクスポートできません。
 - z 最初の、空白文字でない文字が数字で、かつ *-L* オプションが設定されていない場合、右詰めを行い先頭に 0 を詰めます。*-L* オプションも設定されている場合は、先頭の 0 を削除します。*n* は、0 以外であればフィールドの幅を定義します。0 の場合、フィールドの幅は最初に代入される値の幅で決定されます。
- z* では、*-i* 属性を指定できません。

`ulimit [-HSacdfmnpstv] [ limit ]`

リソース制限値を設定または表示します。システムによっては、次に挙げたすべての資源の制限をサポートしていないこともあります。*limit* を指定すると、指定したリソースの制限値が設定されます。*limit* の値は、各リソースに指定された単位の数値、または *unlimited* という値です。複数のリソースを指定すると、値の前に制限するリソース名と単位とが表示されます。

オプションをすべて省略すると、*-f* が指定されたものとみなします。

次に、使用可能なリソース制限値を示します。

- a 現在のリソース制限値をすべて表示します。
- c コアダンプのサイズをブロック (512 バイト) 単位で表します。
- d データ領域のサイズを K バイト単位で表します。
- f 現在のプロセスまたは子プロセスが書き込めるファイルのサイズをブロック (512 バイト) 単位で表します。読み込むファイルのサイズに制限はありません。
- H 指定したリソースに対して強い制限値を設定します。

強い制限値は、いったん設定したらあとで値を上げることはできません。

-H も *-S* も省略すると、指定した制限値が強い制限と弱い制限の両方に適用されます。*limit* 引数を省略すると、現在のリソース制限値が表示されます。このとき、*-H* が指定された場合を除き、表示されるのは弱い制限値です。

- m 物理メモリーのサイズを K バイト単位で表します。
- n ファイル記述子数に 1 を加えた値を表します。
- p パイプバッファリングのブロック (512 バイト) の数を表します。
- s スタック領域のサイズを K バイト単位で表します。
- S 指定したリソースに対して弱い制限値を設定します。

弱い制限値は、強い制限値を超えない範囲で値を上げることが可能です。

-H も -s も省略すると、指定した制限値が強い制限と弱い制限の両方に適用されます。*limit* 引数を省略すると、現在のリソース制限値が表示されます。このとき、-H が指定された場合を除き、表示されるのは弱い制限値です。

- t 各プロセスが CPU を使用する秒数を表します。
- v 仮想記憶のサイズを K バイト単位で表します。

umask [-S] [*mask*]

ユーザーファイルの作成時のマスクを *mask* 引数が示す値に設定します。*mask* には、[chmod\(1\)](#) で説明する記号値または 8 進数を指定できます。

記号値を指定すると、新しい umask 値は、*mask* を直前の umask 値の補数に適用した結果の補数になります。*mask* 引数を省略すると、マスクの現在の値を表示します。-s オプションを指定すると、モードは記号値として出力されます。それ以外の場合は、マスクは 8 進数で出力されます。

[umask\(2\)](#) を参照してください。

+unalias [-a] *name*

name のリストで指定された別名を別名リストから削除します。-a オプションを指定すると、すべての別名が設定解除されます。

+unset [-fnv] *vname*

vname のリストで指定された変数の設定を解除します。つまり、それらの変数の値と属性を消去します。読み取り専用の変数は設定を解除できません。-f オプションが設定されていると、名前は関数名を表します。-v オプションが設定されていると、名前は変数名を指します。-f オプションは -v をオーバーライドします。-n が設定され、*name* が名前参照である場合、*name* が参照する変数ではなく、*name* の設定が解除されます。デフォルトは -v です。LINENO、MAILCHECK、OPTARG、OPTIND、RANDOM、SECONDS、TMOUT、および _ の設定を解除すると、これらの変数の特殊な意味が削除されます。あとでこれらの変数に値を代入しても、特殊な意味はないままです。

wait [*job*]

指定されたジョブの終了を待ち、その終了ステータスを報告します。*job* を指定しないと、現在実行中のすべての子プロセスを待ちます。このコマンドの終了ス

テータスは、*job* を指定した場合は、待機する対象の最後のプロセスの終了ステータスです。それ以外は 0 になります。*job* の記述形式については、「ジョブ」を参照してください。

whence [-afpv] *name* ...

各 *name* ごとに、コマンド名として使用される場合にどのように解釈されるかを指示します。*-v* オプションを指定すると、より詳細に表示されます。*-f* オプションを指定すると、関数を検索しません。*-p* オプションを指定すると、コマンド名が別名、関数、または予約語である場合でも *name* のパスが検索されます。*-a* オプションは、*-v* オプションと似ていますが、指定した名前の解釈がすべて報告される点異なります。

呼び出し

シェルを **exec(2)** で呼び出し、0 番目の引数 (\$0) の最初の文字が *-* である場合は、シェルをログインシェルとみなし、*/etc/profile* からコマンドを読み取ります。次に、現在のディレクトリ内に *.profile* が存在するか、または *\$HOME/.profile* がある場合は、そのいずれかのファイルからコマンドを読み取ります。次に、対話型シェルの場合、コマンドは最初に */etc/ksh.kshrc* から読み取られ、続いて環境変数 *ENV* の値に対してパラメータ展開、コマンド置換、および算術置換を実行することによって指定された名前のファイルが存在する場合はこのファイルから読み取られます。*-s* オプションが存在せず、*arg* と、*arg* の名前のファイルが存在する場合、このスクリプトを読み取って実行します。それ以外の場合は、最初の *arg* に */* が含まれなければ、最初の *arg* に対してパス検索を実行し、実行するスクリプトの名前を判定します。*arg* が示すスクリプトには実行権が必要で、*setuid* と *setgid* の設定は無視されます。パス上でスクリプトが見つからない場合、*arg* は組み込みコマンドまたは組み込み関数の名前を示しているものとして処理されます。

続いてコマンドが記述に従って読み取られ、次のオプションがシェルの起動時にシェルによって解釈されます。

- c* *-c* オプションを指定すると、最初の *arg* からコマンドを読み取ります。残りの引数は、0 で始まる定位置パラメータになります。
- D* \$ が前に付いた二重引用符で囲まれている文字列の一覧が標準出力に出力され、シェルが終了します。この文字列セットは、ロケールが C や POSIX でない場合、各国語への変換の対象になります。コマンドは実行されません。
- i* *-i* オプションが指定された場合、またはシェル入出力が端末に接続されている場合 (**tcgetattr(3C)** を参照)、このシェルは対話型になります。この場合、*kill 0* が対話型シェルを終了しないように *TERM* を無視し、*wait* が割り込み可能になるように *INTR* を捕らえ、無視します。いずれの場合も、シェルは *QUIT* を無視します。
- R filename* *-R filename* オプションは、別のユーティリティで変数およびコマンドの定義および参照の検索に使用できる相互参照データベースを生成するために使用されます。
- r* *-r* オプションを指定すると、シェルは制限付きシェルになります。

-s **-s** オプションが存在する場合、または引数が残っていない場合は、標準入力からコマンドを読み取ります。「特殊コマンド」に記述されているコマンドの出力を除くシェル出力は、ファイル記述子2に書き込まれます。

ほかのオプションと引数については、**set** コマンドの箇所で説明されています。最初の引数としての省略可能な **-** は無視されます。

rksh のみ

rksh は、標準シェルより機能が制限されたログイン名や実行環境を設定するために使用します。

rksh の機能は、次の動作ができない点を除いて ksh と同じです。

- 制限されたオプションの設定解除。
- ディレクトリの変更。[cd\(1\)](#) を参照してください。
- SHELL、ENV、FPATH、または PATH の値または属性の設定と設定解除。
- / を含むパス名またはコマンド名の指定。
- 出力先のリダイレクト (>, >|, <>, および >>)。
- 組み込みコマンドの追加または削除。
- **command -p** を使用したコマンドの呼び出し。

これらの制限は、**.profile** ファイルと **ENV** ファイルの解釈後に有効となります。

実行するコマンドがシェル手続きである場合、rksh は ksh を呼び出してコマンドを実行します。したがって、利用できるコマンドの種類には制限がありますが、一般ユーザーは標準シェルの全機能を利用できるシェル手続きを使用できます。この方式では、一般ユーザーが同じディレクトリにおいて書き込み権と実行権を持っていないことを想定しています。これらの規則の実際の効果は、**.profile** の作成者が確実な設定処理を実行してユーザーを適切な(おそらく、ログインディレクトリ以外の)ディレクトリに置くことにより、ユーザーの動作を完全に制御できるという点にあります。システム管理者は、多くの場合、rksh で安全に起動できるコマンドのディレクトリ (たとえば、**/usr/rbin**) を設定します。

使用法

ファイルが 2G バイト (2³¹ バイト) 以上ある場合の ksh と rksh の動作については、[largefile\(5\)](#) を参照してください。

終了ステータス

次の終了ステータスが返されます。

0 以外

シェルで、構文エラーなどのエラーを検出したときに、0 以外を返します。

シェルの非対話型で使用している場合、シェルフファイルの実行は中止されます。ただし、エラーがサブシェル内部で発生した場合は、サブシェルが中止されます。

最後に実行されたコマンドの終了ステータス

最後に実行されたコマンドの終了ステータスを返します。

シェルが検出する実行時エラーは、コマンド名または関数名、およびエラー状態を表示することによって報告されます。エラーが発生した行の番号が1より大きい場合は、コマンド名または関数名のあとに角括弧 ([]) で囲んだ行番号も表示します。

詳細は、`ksh exit` コマンドを参照してください。

ファイル

- `/etc/profile`
システム初期化ファイル。ログインシェルに対して実行されます。
- `/etc/ksh.kshrc`
システム全体の起動ファイル。対話型シェルに対して実行されます。
- `$HOME/.profile`
個人用初期化ファイル。`/etc/profile` のあとでログインシェルに対して実行されます。
- `$HOME/.kshrc`
デフォルトの個人用初期化ファイル。`ENV` が設定されていない場合、対話型シェルに対して `/etc/ksh.kshrc` のあとで実行されます。
- `/etc/suid-profile`
代替初期化ファイル。実ユーザー (グループ) ID と実効ユーザー (グループ) ID が一致しない場合、個人用初期化ファイルの代わりに実行されます。
- `/dev/null`
NULL デバイス。

著者

David Korn、dgk@research.att.com

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	shell/ksh
インタフェースの安定性	下記を参照。

スクリプト型インタフェースは「不確実」です。環境変数、`.paths` 機能、および編集モードは「流動的」です。

関連項目

[cat\(1\)](#), [cd\(1\)](#), [chmod\(1\)](#), [cut\(1\)](#), [date\(1\)](#), [egrep\(1\)](#), [echo\(1\)](#), [egrep\(1\)](#), [env\(1\)](#), [fgrep\(1\)](#), [grep\(1\)](#), [login\(1\)](#), [newgrp\(1\)](#), [paste\(1\)](#), [printf\(1\)](#), [shell_builtins\(1\)](#), [stty\(1\)](#), [test\(1\)](#), [umask\(1\)](#), [vi\(1\)](#), [dup\(2\)](#), [exec\(2\)](#), [fork\(2\)](#), [ioctl\(2\)](#), [lseek\(2\)](#), [pathconf\(2\)](#), [pipe\(2\)](#), [sysconf\(3C\)](#), [ulimit\(2\)](#), [umask\(2\)](#), [rand\(3C\)](#), [tcgetattr\(3C\)](#), [wait\(3C\)](#), [a.out\(4\)](#), [profile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

『The New KornShell Command and Programming Language』、Bolsky, Morris I., Korn, David G. 共著、PrenticeHall 発行、1995 年

『POSIX-Part 2: Shell and Utilities, IEEE Std 1003.2-1992, ISO/IEC 9945-2』、IEEE、1993年。

注意事項

ksh スクリプトは、将来の ksh への拡張との衝突を避けるため、ISO C99、C++、および JAVA 言語の予約キーワードで使われる名前空間外でシェル関数名を選択するようにしてください。

あるコマンドを実行し、そのあとで同一名のコマンドが、検索パスにおいて元のコマンドがあるディレクトリの前に位置するディレクトリにインストールされた場合、シェルは元のコマンドの方を実行し続けます。新しい方のコマンドを実行するには、alias コマンドの -t オプションを使用してください。

非常に古いシェルスクリプトの中には、パイプ文字 (|) の同義語としてキャレット (^) を使用するものがあります。

複合コマンドの中に hist 組み込みコマンドを指定すると、すべてのコマンドが履歴ファイルから削除されます。

.file という組み込みコマンドは、コマンドの実行前にファイル全体を読み取ります。ファイル内の alias コマンドと unalias コマンドは、ファイル内に定義されたどのコマンドにも適用されません。

ジョブがフォアグラウンドプロセスを待っている間は、トラップは処理されません。このため、CHLD に対するトラップはフォアグラウンドジョブが終了するまで実行されません。

コンマが一部のロケールで小数点と解釈されないようにするために、算術式では、コンマ演算子のあとにスペース文字を残しておくことをお勧めします。

ほかのオペレーティングシステムに移植可能な .paths ファイルの作成には、いくつかの制限がある可能性があります。

システムが 64 ビットの命令セットをサポートしている場合、/bin/ksh は 64 ビットバージョンの ksh を実行します。

名前	ksh88, rksh88 – Korn シェル。標準/制限付きコマンドとプログラミング言語
形式	<pre> /usr/sunos/bin/ksh [± abCefhikmnoprstuvx] [± o option]... [arg]... /usr/sunos/bin/ksh -c [± abCefhikmnoprstuvx] [± o option]... command_string [command_name [arg...]] /usr/xpg4/bin/sh [± abCefhikmnoprstuvx] [± o option]... [arg]... /usr/xpg4/bin/sh -c [± abCefhikmnoprstuvx] [± o option]... command_string [command_name [arg...]] /usr/sunos/bin/rksh [± abCefhikmnoprstuvx] [± o option]... [arg]... /usr/sunos/bin/rksh -c [± abCefhikmnoprstuvx] [± o option]... command_string [command_name [arg...]] </pre>
機能説明	<code>/usr/xpg4/bin/sh</code> ユーティリティーは標準に準拠したシェルです。このユーティリティーは <code>/usr/sunos/bin/ksh</code> のすべての機能を提供します。ただし、動作が異なる場合を除きます。詳細については、「算術展開」節を参照してください。 <code>/usr/sunos/bin/ksh</code> は、端末またはファイルから読み取られたコマンドを実行するコマンドプログラミング言語です。rksh は、コマンドインタプリタである ksh の制限付きバージョンです。標準シェルよりも機能が制限されたログイン名や実行環境を設定するために使用します。シェルへの引数の意味については、「呼び出し」節を参照してください。
定義	メタキャラクタには、次のような文字があります。 <code>; & () < ></code> 復帰改行文字、スペース、タブ ブランク (<i>blank</i>) は、タブ (TAB) またはスペース文字 (SPACE) のことです。識別子 (<i>identifier</i>) は英文字、数字、または下線の並びで、先頭文字は英文字または下線です。識別子は関数 (<i>function</i>) や変数 (<i>variable</i>) の名前として使用します。ワード (<i>word</i>) は1つまたは複数の引用符なしのメタキャラクタで区切られた文字列です。 コマンド (<i>command</i>) は、シェル言語の構文にそった文字の並びのことです。シェルは各コマンドを読み取り、指定された動作を直接実行するか、または動作を実行する別のユーティリティーを起動します。特殊コマンド (<i>special-command</i>) とは、シェルが個別のプロセスを作成せずに実行するコマンドです。このドキュメントに記述されている副作用が発生する場合を除き、ほとんどの特殊コマンドは個別のユーティリティーとして実装できます。

コマンド

単純コマンド (*simple-command*) は空白で区切られたワードの並びで、その前に変数代入リストを指定できます。「環境」を参照してください。先頭のワードは、実行すべきコマンドの名前を指定します。残りのワードは、後述の場合を除き、呼び出されたコマンドに引数として渡されます。コマンド名は引数0として渡されます ([exec\(2\)](#) を参照)。単純コマンドの値 (*value*) は、正常終了した場合はコマンドの終了ステータスです。シグナルを受け取って異常終了した場合は、シグナル番号に 128 を足した値になります。シグナル値の一覧については、[signal.h\(3HEAD\)](#) を参照してください。なお、正常な終了ステータス値 129 - 255 と、シグナル番号 1 - 127 を受け取って異常終了した場合の値を見分けることはできません。

パイプライン (*pipeline*) は、パイプ (`|`) で区切られた 1 つ以上のコマンドの並びです。最後のコマンドを除き、各コマンドの標準出力は [pipe\(2\)](#) によってその次のコマンドの標準入力と結合されます。各コマンドは、別々のプロセスとして実行されます。シェルは最後のコマンドが終了するのを待ちます。パイプラインの終了ステータスが最後のコマンドの終了ステータスになります。

リスト (*list*) は、`;`、`&`、`&&`、または `|` で区切られた 1 つ以上のパイプラインの並びです。その並びの終わりに `;`、`&`、または `|&` を記述することもできます。これら 5 つの記号の中で、`;`、`&`、および `|&` の優先度は同じで、`&&` と `|` の優先度より低くなります。`&&` と `|` の優先度は同じです。セミコロン (`;`) によって、直前のパイプラインが順次実行されます。アンパサンド記号 (`&`) によって、直前のパイプラインが非同期的に実行されます。つまりシェルはパイプラインが終了するのを待ちません。`|&` という記号によって、親シェルに対して双方向パイプが確立された直前のコマンドまたはパイプラインが非同期的に実行されます。

生成されたコマンドの標準入出力は、親シェルが特殊コマンドの `read` および `print` (「特殊コマンド」を参照) の `-p` オプションを使用して書き込み、読み取ることができます。`&&` という記号は、直前のパイプラインの戻り値が 0 の場合にだけ後続のリストを実行します。反対に (`|` `|`) は、戻り値が 0 以外の場合にだけ後続のリストを実行します。コマンドの区切りとして、セミコロンの代わりに任意の数の復帰改行を *list* に指定できます。

コマンド (*command*) は、単純コマンドまたは次のいずれかです。特に断わりのない限り、コマンドが返す値は、そのコマンド中で最後に実行された単純コマンドの値です。

`for identifier [ in word ... ] ; do list ; done`

`for` コマンドが実行されるたびに、*identifier* は *in word* リストから次に得られる *word* に設定されます。*in word ...* を省略すると、`for` コマンドは設定された各位置パラメータに対して `do list` を 1 回実行します。「パラメータ置換」を参照してください。リストの *word* がなくなると、実行は終了します。

`select identifier [ in word ... ] ; do list ; done`

`select` コマンドは、標準エラー (ファイル記述子 2) に、一群のワードをそれぞれの前に番号を付けて出力します。*in word ...* を省略すると、定位置パラメータが使用されます。「パラメータ置換」を参照してください。PS3 プロンプトが出力

され、標準入力から行が読み取られます。この行が、リストに示された *word* のいずれかの番号からなる場合、*identifier* が示す変数の値はこの番号に該当する *word* に設定されます。この行が空の場合は、再度選択リストを出力します。空でない場合は、*identifier* 変数の値を NULL に設定します (NULL については「ブランクの解釈」を参照)。標準入力から読み取られた行の内容は、REPLY というシェル変数に保存されます。break またはファイルの終わり (EOF) に行き当たるまで、選択が発生するたびに *list* が実行されます。*list* の実行によって REPLY 変数が NULL に設定されると、次の選択を求める PS3 プロンプトの表示前に選択リストが出力されます。

```
case word in [ pattern [ | pattern ] ) list ; ; ] ... esac
```

case コマンドは、*word* に一致する最初の *pattern* に対応した *list* を実行します。pattern の形式は、ファイル名生成に使用される形式と同じです。「ファイル名の生成」を参照してください。

```
if list ; then list ; [ elif list ; then list ; ... ] [ else list ; ] fi
```

if のあとの *list* を実行後、*list* が 0 の終了ステータスを返すと、最初の then のあとの *list* を実行します。それ以外の場合、elif のあとの *list* を実行します。この値が 0 の場合、次の then のあとの *list* を実行します。これが失敗すると、else *list* を実行します。else *list* も then *list* も実行しない場合、if コマンドは 0 の終了ステータスを返します。

```
while list ; do list ; done
```

```
until list ; do list ; done
```

while コマンドは、while *list* を繰り返し実行し、*list* 中の最後のコマンドの終了ステータスが 0 の場合、do *list* を実行します。それ以外の場合、ループは終了します。do *list* 中のコマンドを実行しない場合、while コマンドは 0 の終了ステータスを返します。ループ終了条件の判定を逆にするには、while の代わりに until を使用します。

```
(list)
```

別の環境で *list* を実行します。入れ子で 2 つの開いた括弧を連続して記述する場合、算術評価を避けるためにスペースを挿入する必要があります。

```
{list}
```

単に *list* を実行します。メタキャラクタの (と) とは異なり、{ と } は「予約語」なので、認識されるためには行の始めまたは ; のあとに現れる必要があります。

```
[[expression]]
```

expression を評価し、*expression* が真のときに 0 の終了ステータスを返します。*expression* については、「条件式」を参照してください。

```
function identifier { list ; }
```

```
identifier( ) { list ; }
```

identifier で参照される関数を定義します。{ と } の間のコマンド群 (*list*) が関数の本体になります。「関数」を参照してください。

time pipeline

pipeline を実行し、標準エラーに経過時間、ユーザー時間、およびシステム時間を出力します。

次の予約語は、コマンドの最初に現れたとき、および引用符を付けずに記述されたときに認識されます。

```
!          if          then      else      elif      fi          case
esac       for         while     until     do        done      {  }
function   select      time      [[  ]]
```

Comments

でワードを始めると、そのワードおよび以降の復帰改行までの文字がすべて無視されます。

別名

各コマンドの最初のワードに別名が定義されている場合、そのワードは別名のテキストに置き換えられます。別名は任意の数の文字で構成されます。別名に使用できない文字は、メタキャラクタ、引用符、ファイル展開文字、パラメータ置換文字、コマンド置換文字、および= です。代入する文字列としては、前述のメタキャラクタを含む有効なシェルスクリプトを指定できます。置換されたテキスト内にある各コマンドの最初のワードは、置換対象のものを除き、別名についてチェックされます。別名の最後の文字がブランクの場合、別名のあとのワードも別名置換についてチェックされます。別名を使用して、特殊組み込み型コマンドを再定義できますが、前述の予約語は再定義できません。別名は `alias` コマンドで作成、表示、エクスポートし、`unalias` コマンドで削除します。エクスポートされた別名は、名前指定で起動されたスクリプトに対しても有効ですが、シェルを起動するたびに初期化し直す必要があります。「呼び出し」を参照してください。再帰的に別名化する際に無限ループの発生を防ぐため、シェルが現在同じ名前の別名を処理中でなければ、ワードは別名の値に置換されますが、処理中であれば置換されません。

別名化はスクリプトの読み取り時に実行されますが、スクリプトの実行中には行われません。したがって別名を有効にするには、別名を参照するコマンドの読み取り前に `alias` コマンドで別名を定義しておく必要があります。

別名は、完全パス名の省略形としてよく使用されます。別名化機能のオプションを使って、別名の値を、該当するコマンドの完全パス名に自動的に設定できます。これらの別名を検索済み (*tracked*) 別名と呼びます。検索済み別名の値は、対応するコマンドを最初に検索するときに定義され、`PATH` 変数を再設定するたびに未定義になります。これらの別名は検索済みのままとなり、次の参照時に値が再定義されます。複数の検索済み別名がシェル中にコンパイルされます。`set` コマンドの `-h` オプションは、参照された各コマンド名を検索済み別名にします。

次に示す「エクスポート済み別名」はシェルにコンパイルされ組み込まれますが、設定解除または再定義が可能です。

```
autoload='typeset -fu'
functions='typeset -f'
history='fc -l'
```

```
integer='typeset -i'
nohup='nohup '
r='fc -e -'
```

後方のブランク文字と予約語に関する例を示します。ユーザーが次のように入力したとします。

```
$ alias foo="/bin/ls "
$ alias while=""
```

ここで次のコマンドを実行します。

```
$ while true
> do
> echo "Hello, World"
> done
```

この結果、Hello, World という文字列が画面上に無限に表示されます。一方、次のように入力したとします。

```
$ foo while
```

この結果は / の ls 出力となります。foo に対する別名置換はスペース文字で終わりとなるため、次のワードが別名置換用にチェックされます。次のワード while も別名化されているので、置換が行われます。while は、コマンド名として正しい位置にないため、予約語として認識されません。

ユーザーが次のように入力したとします。

```
$ foo; while
```

この場合、while は通常どおり予約語とみなされます。

チルド置換

別名置換を実行すると、各ワードが引用符なしの ~ で始まっているかどうかチェックされます。結果が真なら、/ までのワードがユーザー名と一致するかどうかチェックされます。一致した場合は、~ および一致したログイン名が一致したユーザーのログインディレクトリに置き換えられます。これをチルド (*tilde*) 置換と呼びます。一致するユーザー名が見つからない場合、元のテキストは変更されません。~ が単独で指定された場合や、後ろに / を伴って指定された場合は、\$HOME に置き換えられます。~ のあとに + または - を指定すると、それぞれ \$PWD または \$OLDPWD に置き換えられます。

また、変数に代入する値が ~ で始まるときにも、チルド置換が試行されます。

チルド展開

チルド接頭辞 (*tilde-prefix*) は、ワードの先頭の引用符なしのチルド文字と、それに続く文字列からなります。この文字列は、ワード中に引用符なしのスラッシュが含まれている場合は最初のスラッシュの前にあるすべての文字であり、スラッシュが含まれていない場合はワード中のすべての文字です。代入では、複数のチルド接頭辞を使用できます。つまり、ワードの先頭 (代入を示す等号の直後)、引用符が付かないコロンの直後、あるいはその両方で使用できます。代入のチルド接頭辞の終わ

りは、引用符が付かないコロンまたはスラッシュが最初に現れた位置です。チルド接頭辞中に引用符付きの文字が1つもない場合は、先頭のチルドを除いた部分文字列が、ユーザーデータベースに登録されているログイン名である可能性があると思なされます。

移植可能なログイン名には、LOGNAME 環境変数の節に記載されている文字セット以外の文字を含めることはできません。ログイン名が NULL のとき、つまりチルド接頭辞がチルドだけからなるとき、チルド接頭辞は変数 HOME の値に置き換えられます。HOME が設定されていないと、その結果は不確定です。ログイン名が NULL でなければ、チルド接頭辞は getpwnam 関数を使って得られるログイン名に対応したホームディレクトリのパス名に置き換えられます。システムがログイン名を認識できない場合、結果は未定義です。

チルド展開は、通常はワードの先頭でのみ発生しますが、次のように従来の使用方法に基づく例外的な使い方もあります。

```
PATH=/posix/bin:~dgc/bin
```

これもチルド展開の対象となります。チルドがコロンの直後にあり、それに続く文字が1つも引用符で囲まれていないためです。次のような置換も発生しうするため、このような動作は抑止することも検討されました。

```
PATH=$(printf %s ~karels/bin : ~bostic/bin)
for Dir in ~maat/bin ~srb/bin .
do
    PATH=${PATH:+$PATH:}$Dir
done
```

最初のコマンドでは、各ディレクトリにコロンが明示的に指定されています。いずれの場合も、すべてのディレクトリ名が別個のワードとして認識されるため、シェルは各ディレクトリ名に対してチルド展開を実行します。

オペランド中の式の例を次に示します。

```
make -k mumble LIBDIR=~chet/lib
```

このような代入式は、シェル変数の代入と認識される条件を満たしていないので、チルド展開は行われません。ただし、コマンド自体が展開を行う場合は除きます (make は展開を行いません)。

将来的な実装のために、どのワード中でもチルド展開を強制的に実行する方法として、\$~ という特殊な文字列が提供されています。

ワードに引用符を付けてはいけな、という規則なので、次の例は同等ではありません。チルド展開が行われるのは、最後の例だけです。

```
\~hlj/  ~\hlj/  ~"hlj"/  ~hlj\  ~hlj/
```

Korn シェルの `~+` および `~-` 構造はこの規則を利用しているため、未定義のログイン名にチルドを付加した場合は結果を予測できませんが、一般に誤ったログイン名にチルドを付加するとエラーになります。HOME が未設定の場合、従来のシェルの中にはそれをエラーとするものもあるため、結果は不確定です。

コマンド置換

コマンドがドル記号に続く括弧で囲まれている場合 (つまり `$(command)` の形式)、または一対の逆引用符 (`"`) で囲まれている場合は、その標準出力をワードの一部または全体として使用できます。行末の復帰改行は削除されます。2 番目の古い形式では、コマンドの実行前に、引用符間の文字列に含まれる特殊引用符文字が処理されます。「クォート」を参照してください。`$(cat file)` というコマンド置換は、同じ動作で実行速度の速い `$(<file)` に置き換えることができます。入出力のリダイレクションを行わないほとんどの特殊コマンドのコマンド置換は、別のプロセスを作成せずに実行されます。

コマンド置換を使って、コマンド名自体をコマンドの出力に置き換えることもできます。コマンド置換は、コマンドを次のように囲んで記述すると行われます。

```
$(command)
```

または、逆引用符を使って次のように指定します。

```
'command'
```

シェルは、`command` をサブシェル環境で実行し、コマンド置換指定 (`command` のテキストと、それを囲む `$()` または逆引用符) をコマンドの標準出力で置き換え、置換指定の最後にあるいくつかの連続した復帰改行文字を除去することにより、コマンド置換を展開します。出力の途中で埋め込まれている復帰改行文字は除去されません。ただし、IFS の値と現在有効な引用符によっては、そのような復帰改行文字はフィールド区切り文字とみなされて、フィールド分割時に削除されることもあります。

逆引用符形式のコマンド置換では、バックスラッシュは文字そのものとしての意味を保持します。ただし、次に示す文字が直後に続く場合を除きます。

```
$ ' \
```

(ドル記号、逆引用符、バックスラッシュ)。対応する逆引用符の検索は、直前の文字がバックスラッシュでない逆引用符が見つかったと完了します。この検索では、シェルコメント、here-document (各種シェルの `<<` を使用する機能)、`$(command)` 形式の組み込みコマンド置換、または引用符付きの文字列の中でエスケープのない逆引用符が検出された場合、結果は未定義となります。先頭が `'...'` の並びで、終わりがこの並びではない、単一引用符または二重引用符で囲まれた文字列がある場合、結果は未定義となります。

`$(command)` 形式の指定では、左括弧の次の文字から、対応する右括弧までのすべての文字が `command` を構成します。この `command` には、次の場合を除いて任意の有効なシェルスクリプトを使用できます。

- リダイレクションだけで構成されるスクリプトの場合、結果は不確定です。

- 単一のサブシェルの場合、後述のような制限があります。

コマンド置換によって得られた結果は、あとでチルド展開、パラメータ展開、コマンド置換、または算術展開を行うためにフィールド分割またはパス名展開されることはありません。二重引用符に囲まれた内部でコマンド置換が発生した場合、その置換結果に対するコマンド置換は行われません。

コマンド置換は入れ子にできます。逆引用符形式の内部で入れ子を指定するには、内側の両方の逆引用符の直前にバックスラッシュを記述する必要があります。例:

```
\`command\`
```

逆引用符を使ったときの動作に一貫性がないという問題は、`$()` 形式のコマンド置換を使えば解決できます。例:

コマンド	出力
<code>echo \"\$x'</code>	<code>\\$x</code>
<code>echo `echo \"\$x`</code>	<code>\$x</code>
<code>echo \$(echo \"\$x')</code>	<code>\\$x</code>

また、逆引用符形式には、従来から組み込みコマンドの内容に対して制限があります。新しい `$()` 形式があらゆる種類の有効な組み込みスクリプトを処理できるのに対し、逆引用符形式は逆引用符を含んだ有効なスクリプトを処理できないことがあります。たとえば、次に示す例はいずれも有効な組み込みスクリプトですが、右側は正しく処理されるのに対し、左側は処理されません。

<code>echo `</code>	<code>echo \$(</code>
<code>cat <<eof</code>	<code>cat <<eof</code>
<code>a here-doc with `</code>	<code>a here-doc with)</code>
<code>eof</code>	<code>eof</code>
<code>`</code>	<code>)</code>
<code>echo `</code>	<code>echo \$(</code>
<code>echo abc # a comment with `</code>	<code>echo abc # a comment with)</code>
<code>`</code>	<code>)</code>
<code>echo `</code>	<code>echo \$(</code>
<code>echo "'</code>	<code>echo ')</code>

‘

)

逆引用符形式のコマンド置換は、このように動作に一貫性がないため、コマンド置換を入れ子にするアプリケーションや複雑なスクリプトを埋め込もうとするアプリケーションに使用することはお勧めしません。

コマンド置換が次のように単一のサブシェルからなる場合を考えます。

`$( (command) )`

移植可能なアプリケーションでは、`$(` と `(` を2つのトークンに(スペースを使って)分離する必要があります。これは算術展開との混同を避けるためです。

演算機能の拡張

ドル記号に続く二重括弧で囲まれた算術式(つまり `$( (arithmetic-expression) )`)の形式)は、二重括弧内の算術式の値に置き換えられます。算術展開は、算術式を評価してその値に置換するためのメカニズムを提供します。算術展開の形式は次のとおりです。

`$( (expression) )`

この式は、二重引用符で囲んだ場合と同じように扱われます。ただし、式の内部の二重引用符は特別な意味を持つとはみなされません。シェルは、パラメータ展開、コマンド置換、および引用符削除のために、式の中のトークンをすべて展開します。

次に、シェルはこの式を算術式とみなし、式の値に置き換えます。算術式は、次の例外を除いて、ISO Cの規則に従って処理されます。

- 整数の算術だけが必須です。
- `sizeof()` 演算子、および先頭と末尾の `++` 演算子と `--` 演算子は必須ではありません。
- 選択、繰り返し、ジャンプの各ステートメントはサポートされていません。
- `/usr/sunos/bin/ksh` および `/usr/sunos/bin/rksh` は、先頭の0から9までを10進定数として扱います。次に例を示します。

コマンド	/bin/ksh の結果	/usr/xpg4/bin/sh の結果
<code>echo \$((010+10))</code>	20	18
<code>echo \$((019+10))</code>	29	エラー
<code>[10 -le \$((011))]</code>	true	false

拡張機能として、シェルはこれら以外の算術式も認識できます。式が無効な場合、展開は失敗し、シェルは失敗を示すメッセージを標準エラー出力に書き出します。

算術展開を行う簡単な例を次に示します。

```
# repeat a command 100 times
x=100
while [ $x -gt 0 ]
do
    command
    x=$((x-1))
done
```

プロセス置換

この機能は SunOS が提供するもので、UNIX オペレーティングシステムのバージョンが、開いたファイルに名前を付けるための `/dev/fd` ディレクトリをサポートしている場合にだけ使用できます。<(list) または >(list) 形式の各コマンド引数は、list が示すプロセスを実行します。このプロセスは、`/dev/fd` 中のファイルに非同期的に接続されています。このファイルの名前がコマンドの引数になります。> が付いた形式を使用した場合は、このファイルに書き出すことで list に入力できます。< の形式を使用した場合は、引数として渡されたファイルに list プロセスからの出力が含まれます。例:

```
paste <(cut -f1 file1) <(cut -f3 file2) | tee >(process1) >(process2)
```

このコマンドは `file1` からフィールド 1 をカットし、`file2` からフィールド 3 をカットし、その両者をペーストし、その結果を `process1` と `process2` に送り、さらに標準出力に書き出します。このファイルは、引数としてコマンドに渡されますが、UNIX の `pipe(2)` になっているので、ファイル上で `lseek(2)` を行おうとするプログラムは動作しません。

パラメータ置換

パラメータ (*parameter*) は、識別子 (*identifier*)、1 つまたは複数の数字、または *、@、#、?、-、\$、! の文字のいずれかです。変数 (*variable*) は識別子が示すパラメータで、1 つの値 (*value*) といくつかの属性 (*attribute*) を持っています。属性がない場合もあります。typeset 特殊コマンドを使用すれば、変数に値と属性を代入できます。シェルがサポートする属性については、typeset 特殊コマンドの節で後述します。エクスポートされた変数によって、値と属性が環境に渡されます。

シェルは一次元配列機能をサポートします。配列変数の要素は、添字 (*subscript*) によって参照されます。添字を指定するには、最初に [、次に算術式、その次に] を記述します。「算術評価」を参照してください。配列に値を割り当てるときは、set -A name value ... を使用します。添字の値は常に 0 から 4095 の範囲内で指定します。配列を宣言する必要はありません。有効な添字を伴う変数の参照は正当であり、必要に応じて配列が作成されます。添字なしで配列を参照するのは、0 番目の要素を参照するのと同等の意味です。配列の identifier に添字として * または @ を使用すると、各要素の値に置換されます (フィールド区切り文字で区切られます)。

次の記述方法で変数に値を代入することもできます。

```
name=value [ name=value ] ...
```

name に -i という整数属性を設定すると、value は後述の算術評価を受けます。

定位置パラメータは数値で設定されるパラメータで、`set` 特殊コマンドで値を代入できます。パラメータ `$0` は、シェルが呼び出されたときに、引数 `0` から設定されます。1つまたは複数の数字からなるパラメータは、定位置パラメータになります。複数の数字からなる定位置パラメータは中括弧で囲む必要があります。

パラメータ展開

パラメータ展開の形式は次のとおりです。

`${expression}`

expression は、対応する `}` の直前までのすべての文字を含みます。バックスラッシュによりエスケープされている `}` や引用符で囲まれた文字列内の `}`、および埋め込まれた算術展開、コマンド置換、変数展開の中の文字は、対応する `}` の検索対象にはなりません。

パラメータ展開のもっとも単純な形式は次のとおりです。

`${parameter}`

parameter に値があれば、その値に置き換えられます。

パラメータの名前と記号は、中括弧 (`{}`) で囲むこともできます。この中括弧は、定位置パラメータが複数の数字からなる場合や、名前の一部として解釈される可能性のある文字が *parameter* に続く場合を除いて、省略可能です。対応する右中括弧は、中括弧のレベルを数えながら、引用符で囲まれた文字列やコマンド置換をスキップして見つけられます。

パラメータの名前や記号が中括弧で囲まれていない場合は、名前が表す記号が存在しているかどうかに関係なく、もっとも長い有効な名前が展開に使われます。シェルが入力を検索して名前の境界を判定するときは、すでに定義されている名前は考慮されません。たとえば、`F` というシェル変数が定義されているときに、次のコマンドを実行したとします。

```
echo $Fred
```

この場合、`$F` のあとに `red` が表示されることはありません。正しい名前になる可能性がある最長の文字列は `Fred` であり、そのような名前は定義されていないためです。

二重引用符内でパラメータ展開が発生した場合は、次の制限があります。

- 展開結果に対してパス名展開は行われません。
- `@` の場合を除き、展開結果に対してフィールド分割は行われません。

さらに、次のいずれかの形式を使ってパラメータ展開を変更できます。*word* の値が必要な場合 (*parameter* の状態による)、*word* はチルド展開、パラメータ展開、コマンド置換、および算術展開の対象になります。*word* が必要でない場合、展開は行われません。次のパラメータ展開の変更指定を区切る文字 `}` は、前述の方法および `dquote` のセクションで説明する方法によって判定されます。たとえば、`${foo-bar}xyz` という指定は、`foo` が設定されていれば `foo` の展開のあとに文字列 `xyz` が続き、`foo` が設定されていなければ文字列 `barxyz` になります。

<code>\${parameter:-word}</code>	デフォルト値の使用。 <i>parameter</i> が未設定または NULL の場合は、 <i>word</i> の展開に置換されます。それ以外の場合は <i>parameter</i> の値に置換されます。
<code>\${parameter:=word}</code>	デフォルト値の割り当て。 <i>parameter</i> が未設定または NULL の場合は、 <i>word</i> の展開が <i>parameter</i> に割り当てられます。どのような場合でも、 <i>parameter</i> の最終的な値に置換されます。この割り当て方法は変数だけに使用可能で、定位置パラメータや特殊パラメータには使用できません。
<code>\${parameter:?word}</code>	NULL または未設定のときエラー表示。 <i>parameter</i> が未設定または NULL の場合、 <i>word</i> の展開 (または <i>word</i> 省略時は未設定を表すメッセージ) が標準エラーに書き出され、シェルはゼロ以外の終了ステータスで終了します。それ以外の場合は <i>parameter</i> の値に置換されます。対話型シェルでは終了しません。
<code>\${parameter:+word}</code>	代替値の使用。 <i>parameter</i> が未設定または NULL の場合、NULL に置換されます。それ以外の場合は <i>word</i> の展開結果に置換されます。

前述のパラメータ展開では、形式内にコロンを指定すると、パラメータが未設定または NULL かどうかのテストになります。コロンを省略すると、パラメータが未設定かどうかだけのテストになります。次の 2 つの表は、コロンの効果についてまとめたものです。

	パラメータが NULL 以外に設定された場合	パラメータが NULL に設定された場合
<code>\${parameter:-word}</code>	<i>parameter</i> に置換	<i>word</i> に置換
<code>\${parameter-word}</code>	<i>parameter</i> に置換	NULL に置換
<code>\${parameter:=word}</code>	<i>parameter</i> に置換	<i>word</i> を代入
<code>\${parameter=word}</code>	<i>parameter</i> に置換	<i>parameter</i> に置換
<code>\${parameter:?word}</code>	<i>parameter</i> に置換	エラー、終了
<code>\${parameter?word}</code>	<i>parameter</i> に置換	NULL に置換
<code>\${parameter:+word}</code>	<i>word</i> に置換	NULL に置換
<code>\${parameter+word}</code>	<i>word</i> に置換	<i>word</i> に置換

	パラメータが未設定の場合
<code>\${parameter:-word}</code>	<i>word</i> に置換

	パラメータが未設定の場合
<code>\${parameter-word}</code>	<code>word</code> に置換
<code>\${parameter:=word}</code>	<code>word</code> を代入
<code>\${parameter=word}</code>	NULL を代入
<code>\${parameter:?word}</code>	エラー、終了
<code>\${parameter?word}</code>	エラー、終了
<code>\${parameter:+word}</code>	NULL に置換
<code>\${parameter+word}</code>	NULL に置換

「...に置換」は、式が表に示した値に置き換えられることを表します。「...を代入」は、*parameter* にその値が代入され、さらに式が置き換えられることを表します。

`${#parameter}` 「文字列の長さ」。パラメータの値の長さを文字単位で示します。*parameter* が * または @ のときは、\$1 から始まるすべての定位置パラメータ (フィールド区切り文字で区切られている) に置き換えられます。

次に示す 4 種類のパラメータ展開は、部分文字列を処理するためのものです。どの場合も、パターンの評価には正規表現ではなくパターンマッチングの表記 (patmat を参照) が使われます。*parameter* が * または @ のときは、\$1 から始まるすべての定位置パラメータ (フィールド区切り文字で区切られている) に置き換えられます。完全パラメータ展開文字列を二重引用符で囲んでも、次の 4 種類のパターン文字列は引用符付きになりません。ただし、中括弧の内部で文字列を囲んだ場合は、引用符付きになります。

- `${parameter%word}` 最小の接尾辞パターンの削除。*word* が展開されてパターンを生成します。その後、*parameter* に対してパラメータ展開が行われ、接尾辞中の *pattern* と一致する最小部分が削除されます。
- `${parameter%%word}` 最大の接尾辞パターンの削除。*word* が展開されてパターンを生成します。その後、*parameter* に対してパラメータ展開が行われ、接尾辞中の *pattern* と一致する最大部分が削除されます。
- `${parameter#word}` 最小の接頭辞パターンの削除。*word* が展開されてパターンを生成します。その後、*parameter* に対してパラメータ展開が行われ、接頭辞中の *pattern* と一致する最小部分が削除されます。

`${parameter##word}` 最大の接頭辞パターンの削除。*word*が展開されてパターンを生成します。その後、*parameter*に対してパラメータ展開が行われ、接頭辞中の*pattern*と一致する最大部分が削除されます。

例:

```
${parameter:-word}
```

次の例では、*x*がNULLまたは未設定の場合にのみ*ls*が実行されます。`$(ls)` コマンド置換の表記については、前述の「コマンド置換」で説明しています。

```
${x:-$(ls)}
```

```
${parameter:=word}
```

```
unset X
echo ${X:=abc}
abc
```

```
${parameter:?word}
```

```
unset posix
echo ${posix:?}
sh: posix: parameter null or not set
```

```
${parameter:+word}
```

```
set a b c
echo ${3:+posix}
posix
```

```
${#parameter}
```

```
HOME=/usr/posix
echo ${#HOME}
10
```

```
${parameter%word}
```

```
x=file.c
echo ${x%.c}.o
file.o
```

```
${parameter%%word}
```

```
x=posix/src/std
echo ${x%/*}
posix
```

```
${parameter#word}
```

```

x=$HOME/src/cmd
echo ${x#$HOME}
/src/cmd

${parameter##word}

x=/one/two/three
echo ${x##*/}
three

```

シェルが設定する
パラメータ

次のパラメータはシェルが自動的に設定します。

#	定位置パラメータ数 (10 進数)
—	呼び出し時に、または set コマンドによってシェルに与えられたフラグ。
?	最後に実行されたコマンドが返した 10 進数。
\$	このシェルのプロセス番号。
_	_ の値は、最初は環境 (<i>environment</i>) に引き渡される実行中のシェルまたはスクリプトの絶対パス名です。その後、直前のコマンドの最後の引数が代入されます。このパラメータは、非同期式のコマンドに関しては設定されません。このパラメータは、メールのチェック時にも、一致する MAIL ファイルの名前を保持するために使用されます。
!	最後に呼び出されたバックグラウンドコマンドのプロセス番号。
ERRNO	最後に失敗したシステムコールにより設定された <i>errno</i> の値。この値はシステムに依存し、デバッグのために使用されます。
LINENO	実行中のスクリプトまたは関数内での現在行の行番号。
OLDPWD	cd コマンドで設定された直前の作業用ディレクトリ。
OPTARG	getopts 特殊コマンドで処理された最後のオプション引数の値。
OPTIND	getopts 特殊コマンドで処理された最後のオプション引数のインデックス。
PPID	シェルの親のプロセス番号。
PWD	cd コマンドで設定された現在の作業用ディレクトリ。
RANDOM	この変数を参照するたびに、0 から 32767 の間で均一に分散した乱整数を生成します。RANDOM に数値を代入すると、乱数の並びを初期化できません。
REPLY	この変数は、引数指定のない select 文または read 特殊コマンドによって設定されます。

	SECONDS	この変数を参照するたびに、シェルを起動してからの秒数が返されます。この変数に値を代入すると、その値と代入処理実行時からの秒数との合計値が参照時に返されます。
シェルが使用する変数	次の変数はシェルによって使用されます。	
	CDPATH	cd コマンドの検索パスを指定します。
	COLUMNS	この変数を設定すると、その値によってシェル編集モード用と選択(select) リスト出力用の編集ウィンドウの幅が定義されます。
	EDITOR	この変数の値が <code>emacs</code> 、 <code>gmacs</code> 、または <code>vi</code> で終わり、 <code>VISUAL</code> 変数が設定されていない場合は、該当するオプションが有効になります。 <code>set</code> 特殊コマンドを参照してください。
	ENV	この変数は、対話型シェルが呼び出されたときにだけシェルによるパラメータ展開の対象になります。その結果得られた値は、現在の環境で実行するシェルコマンドを含むファイルのパス名として使用されます。このファイルは実行可能形式でなくてもかまいません。ENV を展開した値が絶対パス名でない場合、結果は不確定です。ユーザーの実ユーザー ID と実効ユーザー ID が異なっていたり、実グループ ID と実効グループ ID が異なっていたりすると、ENV は無視されます。 この変数を使って、シェルの呼び出しだけに有効な別名やほかの項目を設定できます。ENV が参照するファイルは <code>\$HOME/.profile</code> とは異なります。つまり、<code>.profile</code> が通常はセッションの起動時に実行されるのに対し、ENV ファイルはシェルを呼び出すたびに最初に実行されます。ENV の値はドットスクリプトと同じように解釈されます。つまり、コマンドは現在の環境で実行され、ファイルは実行可能でなくてもかまいませんが、読み取り可能である必要があります。ただし、ドットスクリプトとは異なり、PATH の検索は行われません。これは「トロイの木馬」型のセキュリティー侵入から保護するために使われます。
	FCEDIT	fc コマンドのデフォルトのエディタ名。
	FPATH	関数定義の検索パス。デフォルトでは、PATH 変数のあとで FPATH のディレクトリを検索します。実行可能ファイルを見つけると、それを現在の環境で実行します。 <code>-u</code> 属性を持つ関数が参照された場合は、PATH 変数の前に FPATH のディレクトリを検索します。設定済みの別名である <code>autoload</code> を使用すると、 <code>-u</code> 属性を持つ関数を生成できます。
	HISTFILE	シェルの起動時にこの変数が設定されていると、その値はコマンド履歴を格納するために使用されるファイルのパス名になります。「コマンド再入力」を参照してください。

HISTSIZE	シェルの起動時にこの変数が設定されていると、このシェルで使用可能な入力済みコマンドの数がこの値以上になります。デフォルト値は 128 です。
HOME	cd コマンドのデフォルト引数 (ホームディレクトリ)。
IFS	内部フィールド区切り文字。通常はスペース、タブ、および復帰改行です。コマンドまたはパラメータの置換によって生じるコマンドワードを区切る場合と、read 特殊コマンドでワードを区切る場合に使用します。\$* 置換で引数を区切るときは、IFS 変数の最初の文字を使用します。「クォート」を参照してください。
LANG	未設定または NULL の国際化変数に対するデフォルト値を指定します。国際化変数のいずれかが無効な値に設定されていると、変数が 1 つも定義されていないように動作します。
LC_ALL	LC_* 変数のデフォルト値を提供します。
LC_COLLATE	パターンマッチングに含まれる範囲式、同等クラス、および複数バイト文字照合要素の動作を決定します。
LC_CTYPE	シェルの文字処理方法を決定します。LC_CTYPE に有効な値が設定されていると、シェルはそのロケールに合った文字を含むテキストやファイル名を表示および処理できます。環境に LC_CTYPE (environ(5)) を参照) が設定されていないければ、シェルの動作は環境変数 LANG によって決定されます。LC_ALL が設定されていれば、その内容が LANG 変数やその他の LC_* 変数より優先されます。
LC_MESSAGES	メッセージをどの言語で出力するかを決定します。
LINENO	この変数は、各コマンドの実行前に、シェルによってスクリプトまたは関数内の現在の行番号 (1 から始まる連続した番号) を表す 10 進数に設定されます。ユーザーが LINENO を設定解除または再設定すると、現在のシェルの動作中は、この変数が持つ特別な意味が失われます。シェルが現在スクリプトも関数も実行していない場合、LINENO の値は不特定です。
LINES	この変数を設定すると、その値は選択 (select) リスト出力用のカラム長の決定に使用されます。選択リストは、LINES が示す行数の約 3 分の 2 がいっぱいになるまで垂直に出力されます。
MAIL	この変数にメールファイルの名前が設定されていて、MAILPATH 変数が設定されていない場合、シェルは指定されたファイルにメールが到着するとユーザーに通知します。
MAILCHECK	MAILPATH 変数または MAIL 変数で指定されたファイルの更新時間の変更をシェルが何秒ごとにチェックするかを指定します。デフォルト値は 600 秒です。この時間が経過すると、シェルは次のプロンプトを出力する前にチェックします。

MAILPATH	コロン(:)で区切ったファイル名のリスト。この変数を設定すると、シェルは直前のMAILCHECK秒間に発生した指定ファイルに対する変更をユーザーに通知します。各ファイル名のあとには?と、出力されるメッセージを指定できます。このメッセージには、変更されたファイルの名前として定義された\$_変数によるパラメータ置換が行われます。デフォルトのメッセージは you have mail in \$_ です。
NLSPATH	LC_MESSAGES 処理用のメッセージカタログの場所を決定します。
PATH	コマンド用の検索パス。「実行」を参照してください。rksh の環境下で実行する場合(.profile の場合を除く)、ユーザーはPATHを変更できません。
PPID	この変数は、シェルによってシェルを呼び出したプロセスのプロセスID(10進数)に設定されます。サブシェルでは、PPIDは現在のシェルの親と同じ値に設定されます。たとえば、echo \$PPID と (echo \$PPID) は同じ値を生成します。
PS1	この変数の値はパラメータ置換用に展開され、1次プロンプト文字列を定義します。デフォルトは\$です。1次プロンプト文字列内の文字!は、コマンド番号で置換されます。「コマンド再入力」を参照してください。!を2つ連続して指定すると、プロンプト文字列の出力時に!が1つ表示されます。
PS2	2次プロンプト文字列を指定します。デフォルトは>です。
PS3	select ループ内で使用する選択プロンプト文字列を指定します。デフォルトは#?です。
PS4	この変数の値は、パラメータ置換用に展開され、実行トレースの各行の前に出力されます。省略すると、実行トレースプロンプトは+になります。
PWD	シェルを初期化したときに、シェルによって現在の作業用ディレクトリの絶対パスに設定されます。このパスには、シンボリックリンクタイプのコンポーネント、ドット(.)のコンポーネント、およびドットドット(..)のコンポーネントは含まれません。アプリケーションがPWDの値を設定または設定解除した場合、cd および pwd ユーティリティーの動作は不確定です。
SHELL	シェルのパス名は環境内に保持されます。起動時に、この変数のベース名が rsh、rksh、または krsh の場合、シェルの機能は制限されます。
TMOUT	0 より大きい値に設定すると、PS1 プロンプトの出力後、指定された秒数以内にコマンドが入力されない場合にシェルが終了しま

す。この値をユーザーが超えられないような大きい値に設定してシェルをコンパイルすることもできます。

VISUAL この変数の値が `emacs`、`gmacs`、または `vi` で終わる場合、該当するオプションが有効になります。`set` 特殊コマンドを参照してください。

シェルは `PATH`、`PS1`、`PS2`、`PS3`、`PS4`、`MAILCHECK`、`FCEDIT`、`TMOUT`、および `IFS` にデフォルト値を割り当てますが、`HOME`、`SHELL`、`ENV`、および `MAIL` はシェルによって設定されません。ただし、`HOME` は `login(1)` によって設定されます。一部のシステムでは、`MAIL` や `SHELL` も `login` で設定されます。

ブランクの解釈 パラメータとコマンドの置換後、置換結果からフィールド区切り文字 (`IFS` にあるもの) を検索し、その文字が現れた位置で分割します。分割された各文字列が引数になります。明示的な `NULL` 引数 ("" または '') は保持されます。暗示的な `NULL` 引数 (値を持たないパラメータから得られるもの) は削除されます。

ファイル名の生成 置換後、各コマンドワードに `*`、`?`、および `[` が含まれているかどうかをチェックします。ただし、`-f` オプションが設定されていない場合にかぎります。これらの文字のいずれかがあると、そのワードはパターンとみなされます。このワードは、パターンと一致する、辞書編集方式の順にソートされたファイル名に置換されます。パターンと一致するファイル名が見つからない場合、ワードは変更されません。パターンをファイル名の生成に使用する場合、ファイル名の先頭のピリオド (`.`) またはスラッシュ (`/`) 直後のピリオド (およびスラッシュ自体) を明示的に一致させる必要があります。ピリオドで始まるファイル名は、括弧の中にピリオドがあるパターンとは一致しません。つまり、`ls .@(r*)` では `.restore` という名前のファイルが検出されますが、`ls @(.r*)` では検出されません。パターンマッチングのほかのケースでは、`/` と `.` は特殊文字とはみなされません。

`*` `NULL` 文字列を含め、任意の文字列と一致します。

`?` 任意の単一文字と一致します。

`[...]` 括弧内のいずれか 1 文字を表します。2 つの文字を `-` で区切ると、その間にある任意の文字 (その 2 つの文字も含む) に一致します。先頭の `[` の次の文字が `!` である場合、`[]` で囲まれていない任意の文字と一致します。`-` は、最初の文字または最後の文字として文字セットに挿入できます。

`pattern-list` は、`|` で区切られた 1 つまたは複数のパターンのリストです。複合パターンは、次のうちの 1 つまたは複数で構成されます。

`?(pattern-list)` 指定されたパターンのいずれかと任意に一致します。

`*(pattern-list)` 指定されたパターンの 0 回以上の出現と一致します。

`+(pattern-list)` 指定されたパターンの 1 回以上の出現と一致します。

`@(pattern-list)` 指定されたパターンのうち 1 つだけと一致します。

!(*pattern-list*) 指定されたパターンのうち1つだけを除き、あらゆるものと一致します。

クォート

前述(「定義」を参照)のメタキャラクタは、それぞれシェルに対して特別な意味を持ち、クォートしないかぎり、ワードの終わりを表します。文字は、その前に\を指定すればクォートされます。つまり、その文字自身を示すことができます。\\

NEWLINE のペアは削除されます。一對の単一引用符(' ')で囲まれた文字はすべてクォートされます。単一引用符を、単一引用符で囲んで表示させることはできません。一對の二重引用符(" ")で囲まれた文字列内では、パラメータとコマンドの置換が発生し、\は\\、'は\'、および\$をクォートします。*\$と@\$の意味は、クォートされていない場合や、パラメータの代入値またはファイル名として使用される場合は同じです。ただし、コマンド引数として使用する場合、*\$は\$1d \$2d...と同等になります。dはIFS変数の最初の文字です。一方、@\$は\$1\$2...と同じになります。一對の逆引用符(` `)で囲まれた中では、\は\\、'は\'、および\$をクォートします。逆引用符を二重引用符内で指定すると、\も"をクォートします。

予約語や別名を持つ特別な意味は、予約語のいずれかの文字をクォートすることで失われます。関数や特殊コマンドについては、その名前をクォートしても、関数名やコマンド名としての認識を変えることはできません。

算術評価

let という特殊コマンドには、整数演算を実行する機能が用意されています。評価はlong演算を使用して行われます。定数の形式は[base#]nです。baseは底を表す2から36の範囲の10進数で、nはその底の数です。baseを省略すると、底は10になります。

算術式には、C言語の式と同一の構文、優先度、および結合規則が使用されます。++、-、?:、,以外のすべての整数演算子がサポートされます。算術式内では、パラメータ置換構文を使用しなくても、名前で変数を参照できます。変数を参照すると、その値は算術式として評価されます。

変数の内部整数表記は、typeset 特殊コマンドの-i オプションで指定できます。算術評価は、-i 属性を備えた変数に対する代入値ごとに実行されます。底を指定しなかった場合は、変数への最初の代入によって底が決定します。この底は、パラメータ置換が発生したときに使用されます。

算術演算子の多くはクォートしなければならないので、代替形式のlet コマンドが提供されています。((で始まるコマンドについては、対応する))までの文字がすべてクォートされた表現とみなされます。具体的には、((...))はlet "...と同じ意味です。

プロンプト

シェルは、対話的に使用すると、コマンドを読み取る前にPS1のパラメータ展開値によるプロンプトを出力します。復帰改行を入力したあとで、コマンドを完了するためにさらに入力が必要な場合は、2次プロンプト(つまりPS2の値)が出力されます。

条件式

条件式 (*conditional expression*) は、ファイルの属性をテストしたり文字列を比較したりするときに、複合コマンドの `[[` とともに使用します。`[[` と `]]` の間のワードについては、ワード分割とファイル名生成は実行されません。各条件式は、次の単項式または2項式をいくつか組み合わせて構築できます。

- a *file* *file* が存在すれば、真です。
- b *file* *file* が存在し、ブロック型特殊ファイルであれば、真です。
- c *file* *file* が存在し、文字型特殊ファイルであれば、真です。
- d *file* *file* が存在し、ディレクトリであれば、真です。
- e *file* *file* が存在すれば、真です。
- f *file* *file* が存在し、通常ファイルであれば、真です。
- g *file* *file* が存在し、setgid ビットが設定されていれば、真です。
- h *file* *file* が存在し、シンボリックリンクであれば、真です。
- k *file* *file* が存在し、スティッキービットが設定されていれば、真です。
- n *string* *string* の長さがゼロでなければ、真です。
- o *option* *option* という名前のオプションが有効であれば、真です。
- p *file* *file* が存在し、FIFO 特殊ファイルまたはパイプであれば、真です。
- r *file* *file* が存在し、現在のプロセスで読み取り可能であれば、真です。
- s *file* *file* が存在し、サイズがゼロより大きければ、真です。
- t *fildes* ファイル記述子番号が *fildes* であるファイルが開いていて、端末デバイスに対応していれば、真です。
- u *file* *file* が存在し、setuid ビットが設定されていれば、真です。
- w *file* *file* が存在し、現在のプロセスで書き込み可能であれば、真です。
- x *file* *file* が存在し、現在のプロセスで実行可能であれば、真です。*file* が存在し、ディレクトリである場合は、現在のプロセスにそのディレクトリを検索する権限があります。
- z *string* *string* の長さがゼロであれば、真です。
- L *file* *file* が存在し、シンボリックリンクであれば、真です。
- O *file* *file* が存在し、このプロセスの実効ユーザー ID がそのファイルを所有していれば、真です。

<code>-G file</code>	<code>file</code> が存在し、ファイルのグループがこのプロセスの実効グループ ID と一致していれば、真です。
<code>-S file</code>	<code>file</code> が存在し、ソケットであれば、真です。
<code>file1 -nt file2</code>	<code>file1</code> が存在し、 <code>file2</code> よりも新しい場合は、真です。
<code>file1 -ot file2</code>	<code>file1</code> が存在し、 <code>file2</code> よりも古い場合は、真です。
<code>file1 -ef file2</code>	<code>file1</code> と <code>file2</code> が存在し、同じファイルを参照していれば、真です。
<code>string</code>	文字列 <code>string</code> が NULL 文字列でなければ、真です。
<code>string == pattern</code>	<code>string</code> が <code>pattern</code> と一致すれば、真です。
<code>string = pattern</code>	<code>==</code> と同じですが、廃止されました。
<code>string != pattern</code>	<code>string</code> が <code>pattern</code> と一致しなければ、真です。
<code>string1 < string2</code>	カテゴリ LC_COLLATE のロケール設定に応じて解釈された文字列に基づいて、 <code>string1</code> が <code>string2</code> より小さければ、真です。
<code>string1 > string2</code>	カテゴリ LC_COLLATE のロケール設定に応じて解釈された文字列に基づいて、 <code>string1</code> が <code>string2</code> より大きければ、真です。
<code>exp1 -eq exp2</code>	<code>exp1</code> が <code>exp2</code> と等しければ、真です。
<code>exp1 -ne exp2</code>	<code>exp1</code> が <code>exp2</code> と等しくなければ、真です。
<code>exp1 -lt exp2</code>	<code>exp1</code> が <code>exp2</code> 未満であれば、真です。
<code>exp1 -gt exp2</code>	<code>exp1</code> が <code>exp2</code> より大きければ、真です。
<code>exp1 -le exp2</code>	<code>exp1</code> が <code>exp2</code> 以下であれば、真です。
<code>exp1 -ge exp2</code>	<code>exp1</code> が <code>exp2</code> 以上であれば、真です。

前述のそれぞれの式で、`file` が `/dev/fd/n` (`n` は整数) という形式の場合は、記述子番号が `n` である開かれたファイルがテストされます。

次のいずれかを使用して、これらの基本式から複合式を構築できます。優先度の高いものから順に並べてあります。

<code>(expression)</code>	<code>expression</code> が真であれば、真です。式をグループ化するときに使用します。
<code>! expression</code>	<code>expression</code> が偽であれば、真です。
<code>expression1 && expression2</code>	<code>expression1</code> と <code>expression2</code> の両方が真であれば、真です。
<code>expression1 expression2</code>	<code>expression1</code> と <code>expression2</code> のどちらかが真であれば、真です。

入出力

コマンドの実行前に、シェルが解釈する特殊な表記を使って入出力先を変更(リダイレクト)できます。次の各項目は、単純コマンド内の任意の位置またはコマンドの前後に指定できますが、起動されたコマンドには引き渡されません。後述の場合を除き、*word* または *digit* を使用する前にコマンドとパラメータの置換が発生します。ファイル名生成が発生するのは、パターンが1つのファイルとだけ一致し、ブランク解釈が実行されない場合だけです。

< <i>word</i>	<i>word</i> というファイルを標準入力(ファイル記述子 0)として使用します。
> <i>word</i>	<i>word</i> というファイルを標準出力(ファイル記述子 1)として使用します。ファイルが存在しない場合は、ファイルを作成します。ファイルが存在し、-noclobber オプションが有効である場合は、エラーになります。その他の場合は、ファイルの長さがゼロになります。
> <i>word</i>	>と同じですが、-noclobber オプションを無視します。
>> <i>word</i>	<i>word</i> というファイルを標準出力として使用します。ファイルが存在する場合、(EOF までシークしたあと)そのファイルに出力を追加します。そうでない場合は、ファイルを作成します。
<< <i>word</i>	<i>word</i> というファイルを標準入力として読み書き用に開きます。
<< [-] <i>word</i>	シェルへの入力として <i>word</i> と同一の行まで、または EOF まで読み取ります。 <i>word</i> に対してパラメータ置換、コマンド置換、ファイル名生成を実行しません。 <i>here-document</i> が生成されて標準入力になります。 <i>word</i> のいずれかの文字がクォートされていると、ドキュメントの文字は解釈されません。クォートされている文字がなければ、パラメータとコマンドの置換が発生し、\NEWLINE が無視されます。また、\ を使って \、\$、'、および <i>word</i> の最初の文字をクォートする必要があります。- を << のあとに付加すると、 <i>word</i> とドキュメントから先行するタブがすべて取り除かれます。
<& <i>digit</i>	ファイル記述子 <i>digit</i> から標準入力複製されます (<i>dup(2)</i> を参照)。同様に、標準出力については >& <i>digit</i> を使用します。
<&-	標準入力を閉じます。同様に、標準出力については >&- を使用します。
<&p	並行プロセスからの入力を標準入力に移動します。
>&p	並行プロセスへの出力を標準出力に移動します。

前述のいずれかの前に数字が付く場合、参照されるファイル記述子番号は(デフォルトの 0 または 1 ではなく)その数字で指定された番号になります。例:

```
... 2>&1
```

前述の例では、ファイル記述子 2 をファイル記述子 1 から複製して、書き込み用に開きます。

リダイレクションを指定する場合、記述する順序が重要になります。シェルは、評価時点での(ファイル記述子とファイルの)関連性について、各リダイレクションを評価します。例:

```
... 1>fname 2>&1
```

これは、まず *fname* というファイルにファイル記述子 1 を関連付けます。次に、ファイル記述子 1 (つまり *fname*) に関連するファイルに、ファイル記述子 2 を関連付けます。リダイレクションの向きが逆の場合は、まずファイル記述子 2 を端末に関連付け(ファイル記述子 1 はすでに端末に関連付けられているとして)、次にファイル記述子 1 をファイル *fname* に関連付けます。

ジョブ制御が有効でない場合にコマンドのあとに & を指定すると、コマンドのデフォルトの標準入力はいずれも */dev/null* という空ファイルになります。それ以外の場合は、コマンドを実行するための環境に起動側シェルのファイル記述子(入出力指定で変更可能)が含まれます。

環境

環境 (*environment*) は、通常の引数リストが実行されるプログラムに引き渡される場合と同様の方法で引き渡される、名前と値の対の集まりです ([environ\(5\)](#) を参照)。名前は識別子、値は文字列である必要があります。シェルが環境と対話する方法はいくつかあります。シェルは、起動されると、環境を走査して、見つけた名前ごとに変数を作成し、対応する値を設定し、さらに *export* というマークを付けます。実行されるコマンドは環境を引き継ぎます。ユーザーがこれらの変数の値を変更するか、新しい変数を作成した場合は、*export* コマンドまたは *typeset -x* コマンドを使用すれば、それらの値が環境の一部になります。したがって、実行されるコマンドが参照する環境は、シェルが最初に引き継いだ名前と値のペア(その値は現在のシェルで変更可能)に、*export* コマンドまたは *typeset -x* コマンドで指定した変数を加えたものになります。

1 つまたは複数の変数代入を先頭に付加すれば、単純コマンドまたは関数の環境を拡張できます。変数代入引数は、*identifier=value* という形式のワードです。したがって、

```
TERM=450 cmd args
```

および

```
(export TERM; TERM=450; cmd args)
```

は、前述の *cmd* の実行に関するかぎり、同じことを意味します。ただし、アスタリスク (*) が先頭に付加されている特殊コマンドが指定された場合を除きます。

-k フラグを設定すると、変数代入引数はすべて環境に格納されます。これらの引数がコマンド名のあとに指定された場合も同様です。次では、最初に *a=b c* を表示し、次に *c* を表示します。

```
echo a=b c
set -k echo
a=b c
```

この機能は、シェルの初期バージョン用に作成されたスクリプトで使用するのためのものです。新しいスクリプトには使用しないでください。この機能は将来なくなる可能性があります。

関数

前述の「コマンド」節で説明した予約語 `function` は、シェル関数の定義に使用します。シェル関数は内部で読み取られ、保存されます。別名は、関数を読み取るときに解釈されます。関数はコマンドと同様に実行され、引数は定位置パラメータとして渡されます。「実行」を参照してください。

関数は、呼び出し側と同じプロセスで実行され、すべてのファイルと現在の作業用ディレクトリを呼び出し側と共有します。呼び出し側が受け取るトラップは、関数の内部でデフォルトの動作に再設定されます。関数が受け取らないか、または無視するトラップ条件がある場合、関数は終了し、その条件が呼び出し側に引き渡されます。

関数内部で設定された `EXIT` に基づくトラップは、関数が呼び出し側の環境で完了したあとに実行されます。これは、非 POSIX スタイルの関数だけに当てはまります。非 POSIX スタイルの関数は次のように宣言されます。

```
function func
```

一方、POSIX スタイルの関数は次のように宣言されます。

```
func()
```

通常、変数は呼び出し側プログラムと関数の間で共有されます。ただし、関数内で使用される `typeset` 特殊コマンドは、現在の関数とそれが呼び出す関数のすべてを適用範囲とする局所変数を定義します。

`return` 特殊コマンドは、関数の呼び出しから戻るときに使用します。関数内でエラーが発生すると、呼び出し側に制御が戻ります。

すべての関数の名前を一覧表示するには、`typeset -f` を使用します。すべての関数の名前とともにすべての関数のテキストも表示するには、`typeset -f` を使用します。特定の関数のテキストだけを表示するには、`typeset -f function-names` を使用します。`unset` 特殊コマンドの `-f` オプションを使用すると、関数を未定義状態にできます。

通常、シェルがシェルスクリプトを実行すると、関数は未設定になります。`typeset` コマンドの `-xf` オプションを使用すると、別個にシェルを起動しなくても実行されるスクリプトに関数をエクスポートできます。シェルの個別起動にまたがって定義する必要がある関数は、`typeset` の `-xf` オプションとともに `ENV` ファイルにも指定する必要があります。

関数定義コマンド 関数とは、新しい定位置パラメータとともに複合コマンドを呼び出すための単純コマンドとして使用されるユーザー定義の名前です。関数は「関数定義コマンド」を使って定義します。

関数定義コマンドの形式は次のとおりです。

```
fname() compound-command[io-redirect ...]
```

`fname` は名前でなければならず、これが関数名になります。実装によっては、拡張機能として、ほかの文字を関数名に使用できる場合があります。その場合は、関数と変数を別個の名前空間で管理します。

関数定義コマンド中の `()` は、2つの演算子で構成されます。したがって、空白文字と `fname`、`()`、`および` を混在させることもできますが、その必要はありません。

引数 *compound-command* は、複合コマンドを表します。

関数を宣言したときは、*compound-command* や *io-redirect* のテキストに対して `wordexp` の展開処理は行われません。すべての展開処理は、通常のように、関数が呼び出されるたびに行われます。同様に、*io-redirect* (省略可能) のリダイレクションや *compound-command* 内の変数代入は、関数の定義時ではなく関数の実行時に行われます。

関数を実行すると、特殊組み込みユーティリティーのために記述された構文エラーや変数代入のプロパティーが提供されます。

関数名が単純コマンドとして指定されると、それに対応した *compound-command* が実行されます。*compound-command* の実行中は、単純コマンドに指定したオペランドが一時的に定位置パラメータになります。特殊パラメータ `#` も、オペランド数を示す値に変更されます。特殊パラメータ `0` は変更されません。関数の実行が終了すると、定位置パラメータや特殊パラメータ `#` の値は関数実行前の値に復元されます。*compound-command* 内で特殊組み込みコマンド `return` が実行されると、関数の実行が終了し、関数呼び出しの次のコマンドから処理が再開されます。

単純コマンドが記述できるところであれば、関数定義を記述できます。次に例を示します。

```
# If variable i is equal to "yes",
# define function foo to be ls -l
#
[ "$i" = yes ] && foo() {
    ls -l
}
```

関数定義の終了ステータスは、関数が正常に宣言されれば0、そうでなければゼロより大きい値です。関数呼び出しの終了ステータスは、関数によって最後に実行されたコマンドの終了ステータスです。

ジョブ

`set` コマンドの `monitor` オプションを有効にすると、対話型シェルが `job` を各パイプラインと関連付けます。このオプションは、`jobs` コマンドが表示する現在のジョブのテーブルを維持し、これらのジョブに整数番号を割り当てます。ジョブを & で非同期に起動すると、シェルは、次の形式の行を表示します。

```
[1] 1234
```

非同期に起動されたジョブはジョブ番号 1 であり、プロセス ID が 1234 である (トップレベルの) プロセスが 1 つあることを示します。

ジョブの実行中に別のジョブを実行しようとする場合は、`^Z` (CTRL-Z) キーを押せば、現在のジョブに `STOP` シグナルが送信されます。シェルは通常、ジョブが「停止」されたことを示し、新しいプロンプトを表示します。これで、このジョブの状態を `bg` コマンドでバックグラウンドで処理するか、またはほかのコマンドを実行してから、`fg` というコマンドでジョブをフォアグラウンドに移すことができます。`^Z` はただちに有効になります。つまり `^Z` は、保留中の出力や読み取られていない入力 that ただちに中止されるという点で、割り込みに似ています。

バックグラウンドで実行中のジョブは、端末から読み取ろうとすると停止します。通常バックグラウンドジョブは出力を生成できますが、`stty tostop` というコマンドを指定すればこの出力生成も無効にできます。この `tty` オプションを設定すると、バックグラウンドジョブは、入力の読み取り時と同様に出力を生成しようとするると停止します。

シェル内のジョブを参照する方法はいくつかあります。そのジョブのいずれかのプロセスの ID を使っても、また次のいずれかを使っても参照できます。

```
%number    number が示す番号のジョブ
%string     コマンド行が string で始まるジョブ
%?string    コマンド行に string を含むジョブ
%%          現在のジョブ
%+          % と同等
%-          直前のジョブ
```

シェルは、プロセスの状態が変更されると、ただちにそれを検出します。ジョブがブロックされてそれ以上進めない状態になると、プロンプトを発行する直前にだけ、シェルはその旨をユーザーに通知します。これは、ユーザーの作業の妨げにならないようにするためです。

モニタモードが有効なとき、完了した各バックグラウンドジョブは、CHLD に設定されているトラップを起こします。

ジョブの実行中または停止中にシェルを終了しようとする、と、「停止中(実行中)のジョブがある (You have stopped(running) jobs.)」旨のメッセージによって警告を受けます。jobs コマンドを使用して、該当するジョブを確認できます。これを実行するか、またはただちにシェルを再終了しようとする、と、シェルは2度目の警告は出さず、停止中のジョブは終了します。nohup コマンドで起動したジョブの実行中にログアウトを行うと、次のような警告メッセージを受けます。

You have jobs running.

実際にログアウトするには、ログアウトを2回行う必要があります。その場合でも、バックグラウンドジョブの実行は継続されます。

シグナル 起動されたコマンドに対する INT シグナルと QUIT シグナルは、コマンドの後ろに & が指定され、ジョブの -monitor オプションが有効でない場合、無視されます。それ以外の場合は、シェルが親から引き継いだ値がシグナルに含まれます。trap 特殊コマンドのセクションを参照してください。

実行 コマンドが実行されるたびに、前述の置換が実行されます。コマンド名は、「特殊コマンド」のいずれかと一致する場合、現在のシェルプロセス内で実行されます。次に、コマンド名がユーザー定義関数のいずれかと一致するかどうかをチェックされます。一致する場合は、定位置パラメータが保存され、関数呼び出しの引数に再設定されます。関数が完了するか return を発行すると、定位置パラメータリストが復元され、関数内の EXIT に設定されているトラップが実行されます。関数の値は、最後に実行されたコマンドの値です。関数は現在のシェルプロセスでも実行されます。コマンド名が特殊コマンドやユーザー定義関数でない場合は、プロセスが作成され、exec(2) を使用してコマンドの実行が試みられます。

PATH というシェル変数は、コマンドを含んでいるディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:) で区切ります。デフォルトのパスは /usr/bin: です (/usr/bin および現在のディレクトリの順に指定)。現在のディレクトリは、複数のコロンを連続して記述するか、パスリストの始めか終わりにコロンを付ければ指定できます。コマンド名に / が含まれている場合は、検索パスは使用されません。そうでない場合は、パスにおける各ディレクトリに実行可能ファイルがあるか検索します。ファイルが実行権を持っているが、ディレクトリや a.out ファイルでない場合は、シェルコマンドの入ったファイルとみなされます。そのファイルを読み取るときは、サブシェルが生成されます。この場合、エクスポートされていない別名、関数、および変数はすべて削除されます。括弧で囲まれたコマンドは、エクスポートされていないものを削除することなく、サブシェルで実行されます。

コマンド行の再入力 端末装置から最近入力された HISTSIZE が示す個数(デフォルトは 128 個)のコマンドのテキストは、履歴ファイルに保存されています。\$HOME/.sh_history という

ファイルは、HISTFILE 変数が設定されていない場合、または変数が示すファイルが書き込み不可能な場合に使用されます。シェルは、同じ名前の HISTFILE を使用する対話型シェルすべてのコマンド履歴を使用できます。fc という特殊コマンドは、このファイルの一部をリスト表示または編集するときに使用します。編集またはリスト表示されるファイルの部分は、番号か、またはコマンドの最初の文字を指定することによって選択できます。単一のコマンドを指定することも、コマンドの範囲を指定することも可能です。fc の引数としてエディタプログラムが指定されていないと、FCEDIT という変数の値が使用されます。FCEDIT が未定義の場合は、/bin/ed が使用されます。編集されたコマンドは、エディタを終了した時点で表示および再実行されます。エディタ名に - を指定すると、編集段階が省かれ、コマンドが再実行されます。この場合、old=new という形式の代入パラメータを使用すれば、実行前にコマンドを変更できます。たとえば、r が 'fc -e -' の別名として定義されている場合に 'r bad=good c' と入力すると、c という文字で始まるコマンドのうち最新のものが、その記述中の最初の bad という文字列を good に置き換えて再実行されます。

インライン編集オプション

通常、端末装置から入力する各コマンド行では、コマンドのあとに復帰改行 (RETURN または LINEFEED) しか入力できません。emacs、gmacs、vi のいずれかのオプションが有効な場合、ユーザーはコマンド行を編集できます。いずれかのオプションを set すれば、対応する編集モードになります。編集オプションは、いずれかのオプション名で終了する値を VISUAL 変数または EDITOR 変数に代入するたびに、自動的に選択されます。

編集機能では、ユーザーの端末が RETURN を改行のないキャリッジリターンとして扱うこと、およびスペース文字がスクリーン上の現在の文字を上書きすることが必要です。

これらの編集モードは、ユーザーがウィンドウを介して現在の行を見るという概念を実現します。ウィンドウ幅は、COLUMNS が定義されていればその値になり、未定義の場合は 80 になります。ウィンドウの幅が小さすぎて、プロンプトを表示すると入力用に 8 カラム以上残すことができない場合は、プロンプトが左端から切り捨てられます。ウィンドウ幅から 2 を引いた長さより行が長い場合は、ウィンドウの終わりにマークを表示してユーザーに通知します。カーソルが移動し、ウィンドウの境界に達すると、ウィンドウはカーソルを中心としてセンタリングされます。行がウィンドウの右端を超えている場合は > マーク、左端を超えている場合は < マーク、左右両端を超えている場合は * マークがそれぞれ表示されます。

各編集モードでは、検索コマンドから履歴ファイルにアクセスできます。パターンではなく文字列だけがマッチングの対象になります。ただし、文字列の先頭に ^ があると、マッチングの開始位置が行の先頭に限定されます。

emacs 編集モード

emacs または gmacs オプションを有効にすると、対応する編集モードに移行します。この 2 つのモードは、^T の扱い方だけが異なります。編集を行うには、訂正が必要な位置にカーソルを移動し、必要に応じて文字やワードを挿入または削除します。編集用のコマンドは、いずれも制御文字またはエスケープシーケンスです。制

御文字は、キャレット (^) とそれに続く文字で表記されます。たとえば、`^F` は `CTRL-F` を表します。つまり、`CTRL` (コントロール) キーを押しながら `f` キーを押します。シフトキーは押しません。`^?` は `DEL` (削除) キーを表します。

エスケープシーケンスは `M-` とそれに続く文字で表記されます。たとえば、`M-f` (メタ `f` と呼ぶ) は `ESC` キー (ASCII コード 033) に続いて `f` を押すことで入力します。`M-F` は `ESC` キーに続いて `SHIFT` キーを押しながら (大文字の) `F` を入力することを表します。

すべての編集コマンドは、行の先頭だけでなく、行のどこからでも実行できます。特に断りのないかぎり、編集コマンドのあとに `RETURN` キーや `LINEFEED` キーは入力しません。

- `^F` カーソルを 1 文字だけ前方 (右) に移動します。
- `M-f` カーソルを 1 単語だけ前方に移動します。emacs エディタでは、「ワード」は英文字、数字、および下線からなる文字列を意味します。
- `^B` カーソルを 1 文字だけ後方 (左) に移動します。
- `M-b` カーソルを 1 単語だけ後方に移動します。
- `^A` カーソルを行の先頭に移動します。
- `^E` カーソルを行の末尾に移動します。
- `^]char` 現在の行で文字 *char* が次に現れる位置にカーソルを移動します。
- `M-^]char` 現在の行で文字 *char* が直前に現れた位置にカーソルを移動します。
- `^X^X` カーソルとマークを入れ替えます。
- erase* (ユーザーが `stty(1)` コマンドで定義した消去文字。通常は `^H` または `#`) 前の 1 文字を削除します。
- `^D` 現在の文字を削除します。
- `M-d` 現在のワードを削除します。
- `M-^H` (メタ - バックスペース) 直前の単語を削除します。
- `M-h` 直前のワードを削除します。
- `M-^?` (メタ - `DEL`) 直前のワードを削除します (割り込み文字が `^?` (デフォルトの `DEL`) の場合、このコマンドは動作しません)。
- `^T` emacs モードでは、現在の文字と次の文字を入れ替えます。gmacs モードでは、直前の 2 つの文字を入れ替えます。
- `^C` 現在の文字を大文字にします。
- `M-c` 現在のワードを大文字にします。

M-l	現在のワードを小文字にします。
^K	カーソルから行の末尾までを削除します。直前に指定された数値パラメータの値が現在のカーソル位置より小さい場合は、指定された位置の文字から現在のカーソル位置の文字までを削除します。直前に指定された数値パラメータの値が現在のカーソル位置より大きい場合は、現在のカーソル位置の文字から指定された位置の文字までを削除します。
^W	カーソル位置の文字からマークまでを抹消します。
M-p	カーソル位置からマークまでの領域をスタックにプッシュします。
kill	(ユーザーが <code>stty(1)</code> コマンドで定義した抹消文字。通常は ^G または @) 現在の行全体を抹消します。2 つの <i>kill</i> 文字を連続して入力すると、その後の <i>kill</i> 文字はすべて改行を意味します。プリンタ端末 (印字式端末) を使用している場合に便利です。
^Y	行から最後に削除された項目を復元します。つまり、記憶していた項目を行に戻します。
^L	復帰改行して、現在の行を表示します。
^@	(NULL 文字) マークを設定します。
M-space	(メタ - スペース) マークを設定します。
J	(NEWLINE) 現在の行を実行します。
M	(RETURN) 現在の行を実行します。
eof	現在の行が NULL の場合のみ、end-of-file 文字 (通常は ^D) を end-of-file として処理します。
^P	直前のコマンドを取り出します。^P を入力するたびに、さらに前のコマンドにアクセスします。複数行にまたがるコマンドで最初の行でない場合は、1 行戻ります。
M-<	もっとも古いコマンド行の履歴を取り出します。
M->	もっとも新しいコマンド行の履歴を取り出します。
^N	次のコマンド行を取り出します。^N を入力するたびに、さらに次のコマンドにアクセスします。
^Rstring	履歴をさかのぼって、 <i>string</i> を含むコマンド行を検索します。パラメータとして 0 を指定すると、順方向に検索します。 <i>string</i> の終わりは復帰改行 (RETURN または NEWLINE) で示します。 <i>string</i> の先頭に ^ が付加されると、その文字列で始まるコマンドだけを検索します。 <i>string</i>

を省略すると、直前に指定した文字列を含む次のコマンド行にアクセスします。この場合、パラメータとして0を指定すると検索方向が逆になります。

- ^O** (Operate) 現在の行を実行し、現在の行に対する次の行を履歴ファイルから取り出します。
- M-digits** (エスケープ) 数値パラメータを定義します。digitsは次のコマンドに対するパラメータとみなされます。パラメータを受け入れるコマンドは、`^F`、`^B`、`erase`、`^C`、`^D`、`^K`、`^R`、`^P`、`^N`、`^J`、`M-.`、`M-^J`、`M-_`、`M-b`、`M-c`、`M-d`、`M-f`、`M-h`、`M-l`、および`M-^H`です。
- M-letter** (ソフトキー) ユーザーの別名リスト中で、`_letter`という名前の別名を検索します。この名前の別名が定義されていれば、入力待ち行列にその値を挿入します。`letter`は、前述の各メタ関数を示す文字であってはいけません。
- M-[letter** (ソフトキー) ユーザーの別名リスト中で、`__letter`という名前の別名を検索します。この名前の別名が定義されていれば、入力待ち行列にその値を挿入します。この機能は、多くの端末でファンクションキーをプログラムするために使用できます。
- M-.** 直前のコマンドの最後のワードを行に挿入します。数値パラメータが指定されると、最後のワードではなくその数値が示す位置のワードを挿入します。
- M-_** `M-.`と同じです。
- M-*** アスタリスクがワードの最後に付加され、ファイル名が展開されます。
- M-ESC** ファイル名を補完します。現在のワードにアスタリスクを付加したものと一致するすべてのファイル名の最長の前方一致部分で、現在のワードを置き換えます。一致するものが1つしかない場合、ファイルがディレクトリなら`/`を付加し、ファイルがディレクトリでないならスペースを付加します。
- M-=** 現在のワードパターンのあとにアスタリスクを付加すれば一致するファイル名をリスト表示します。
- ^U** 次のコマンドのパラメータを4倍します。
- ** 次の文字をエスケープします。編集用文字、ユーザーが設定した消去文字、抹消文字、および割り込み文字(通常は`^?`)は、その前に`\`を指定すれば、コマンド行または検索文字列に入力できます。`\`は、次の文字の編集機能(もしあれば)を無効にします。
- ^V** シェルのバージョンを表示します。

M-# # を行の先頭に挿入したあと、その行を実行します。これによって、履歴ファイルにコメントを挿入できます。

vi 編集モード

2つの入力モードがあります。初期状態では、コマンドを入力すると入力モードに移行します。編集するときは、エスケープ (ESC) キー (033) を入力して制御モードに移行し、訂正が必要な箇所にカーソルを移動してから、必要に応じて文字やワードを挿入または削除します。ほとんどの制御コマンドでは、コマンドの先頭に繰り返し数を指定できます。

ほとんどのシステムでは、vi モードに移行した直後は標準処理が有効になります。端末の回線速度が 1200 ボー以上であり、制御文字が含まれているか、またはプロンプトが表示されてからの経過時間が 1 秒未満であれば、コマンドが再度表示されます。ESC 文字を入力すると、コマンドの残りの部分に対する標準処理が終了し、ユーザーはコマンド行を変更できるようになります。この方式には、標準処理で raw モードの先行入力表示機能を利用できるという利点があります。

viraw オプションも設定すると、端末の標準処理を常に無効にできます。このモードは、行の終わりの区切りを示す 2 つの代替記号をサポートしていないシステムでは暗黙的に設定されており、特定の端末で便利なことがあります。

入力編集コマンド

デフォルトでは、エディタは入力モードになります。

erase (ユーザーが **stty(1)** コマンドで定義した消去文字。通常は ^H または #) 前の 1 文字を削除します。

^W 直前の、空白で区切られたワードを削除します。

^D シェルを終了します。

^V 次の文字をエスケープします。編集用の文字、ユーザーの定義した消去文字または抹消文字は、その前に ^v を入力すれば、コマンド行または検索文字列に入力できます。^v は、次の文字の編集機能 (もしあれば) を無効にします。

**** 次の **erase** または抹消文字をエスケープします。

移動編集コマンド

次のコマンドはカーソルを移動します。

[count]l 1 つ先 (右) の文字にカーソルを移動します。

[count]w 1 つ先の英数字のワードにカーソルを移動します。

[count]W 空白のあとに続く次のワードの先頭にカーソルを移動します。

[count]e ワードの終わりにカーソルを移動します。

[count]E 空白で区切られた現在のワードの終わりにカーソルを移動します。

[count]h 1 つ前 (左) の文字にカーソルを移動します。

[count]b	1 つ前のワードにカーソルを移動します。
[count]B	ブランクで区切られた直前のワードに移動します。
[count]	<i>count</i> が示すカラムにカーソルを移動します。
[count]fc	現在の行で文字 <i>c</i> が次に現れる位置にカーソルを移動します。
[count]Fc	現在の行で文字 <i>c</i> が前に現れる位置にカーソルを移動します。
[count]tc	f と h を連続して実行した場合と同等です。
[count]Tc	F と l を連続して実行した場合と同等です。
[count];	直前の単一文字検索コマンドの f、F、t、または T を <i>count</i> の数だけ繰り返します。
[count],	直前の単一文字検索コマンドを <i>count</i> の数だけ逆方向で実行します。
0	行の先頭にカーソルを移動します。
^	行に含まれる最初のブランク以外の文字にカーソルを移動します。
\$	行の終わりにカーソルを移動します。
%	現在の位置にある括弧記号 (、)、{、}、[、または] に対応する括弧記号にカーソルを移動します。現在の文字が前述のいずれの括弧でもない場合は、行の残りから最初に現れる括弧を検索します。

検索編集コマンド 次の各コマンドはコマンド履歴にアクセスします。

[count]k	直前のコマンドを取り出します。k を入力するたびに、さらに前のコマンドにアクセスします。
[count]-	k と同等です。
[count]j	次のコマンドを取り出します。j を入力するたびに、さらに次のコマンドにアクセスします。
[count]+	j と同等です。
[count]G	<i>count</i> が示す番号のコマンドを取り出します。デフォルトでは、もっとも古いコマンド履歴を取り出します。
/string	履歴をさかのぼって、 <i>string</i> を含むコマンドを検索します。 <i>string</i> の終わりは復帰改行 (RETURN または NEWLINE) で示します。 <i>string</i> の先頭に ^ が付加されると、その文字列で始まるコマンドだけを検索します。 <i>string</i> が NULL の場合は、直前に指定された文字列を使用します。
?string	/ と同じですが、順方向に検索します。
n	直前の / または ? コマンドで指定されたパターンと一致する次のパターンを検索します。

	N	直前の / または ? コマンドで指定されたパターンと一致する次のコマンドを逆方向に検索します。以前に / コマンドで入力された <i>string</i> (文字列) の履歴を検索します。
テキスト変更編集 コマンド		次の各コマンドは行を変更します。
	a	入力モードに移行し、現在の文字のあとにテキストを入力します。
	A	行の終わりにテキストを追加します。\$a と同等です。
	[count]cmotion	現在の文字から <i>motion</i> によりカーソルが移動する先までの文字を削除し、入力モードに移行します。 <i>motion</i> が c の場合は、行全体を削除して入力モードに移行します。
	c[count]motion	
	C	現在の文字から行の終わりまでを削除し、入力モードに移行します。c\$ と同等です。
	[count]s	count 個の文字を削除して入力モードに移行します。
	S	cc と同等です。
	D	現在の文字から行の終わりまでを削除します。d\$ と同等です。
	[count]dmotion	現在の文字から <i>motion</i> によりカーソルが移動する先までの文字を削除します。 <i>motion</i> が d の場合は、行全体を削除します。
	d[count]motion	
	i	入力モードに移行し、現在の文字の前にテキストを入力します。
	I	行の先頭にテキストを挿入します。0i と同等です。
	[count]P	カーソルの前に、直前のテキスト変更を挿入します。
	[count]p	カーソルのあとに、直前のテキスト変更を挿入します。
	R	入力モードに移行して、画面上の文字を重ね打ちした文字に置き換えます。
	[count]rc	現在のカーソル位置から始まる count 個の文字を c に置き換え、カーソルの位置を進めます。
	[count]x	現在の文字を削除します。
	[count]X	直前の文字を削除します。
	[count].	直前のテキスト変更コマンドを繰り返します。
	[count]~	現在のカーソル位置から始まる count 個の文字を、大文字の場合は小文字に、小文字の場合は大文字に変換して、カーソルの位置を進めます。

その他の編集コマンド

<code>[count]_</code>	直前のコマンドの <i>count</i> 個のワードを付加し、入力モードに移行します。 <i>count</i> を省略すると、最後のワードを使用します。
<code>*</code>	<code>*</code> を現在のワードのあとに付加し、ファイル名を生成しようとします。一致するものが見つからない場合は、ベルを鳴らし、見つかった場合は、ワードを一致したパターンで置換し、入力モードに移行します。
<code>\</code>	ファイル名を補完します。現在のワードにアスタリスクを付加したものと一致するすべてのファイル名の最長の前方一致部分で、現在のワードを置き換えます。一致するものが1つしかない場合、ファイルがディレクトリなら <code>/</code> を付加し、ファイルがディレクトリでないならスペースを付加します。
その他の編集コマンド	その他の編集コマンドを次に示します。
<code>[count]ymotion</code> <code>y[count]motion</code>	現在の文字から <i>motion</i> によりカーソルが移動する先までの文字を記憶し、削除用バッファに入れます。テキストとカーソルは変更しません。
<code>Y</code>	現在の位置から行の終わりまでの文字を記憶します。 <code>y\$</code> と同等です。
<code>u</code>	直前のテキスト変更コマンドを取り消します。
<code>U</code>	現在の行で実行されたテキスト変更コマンドをすべて取り消します。
<code>[count]v</code>	<code>fc -e \${VISUAL:-\${EDITOR:-vi}}</code> <i>count</i> コマンドを入力 バッファに戻します。 <i>count</i> を省略すると、現在の行を使用します。
<code>^L</code>	復帰改行して、現在の行を表示します。制御モードでのみ有効です。
<code>J</code>	(NEWLINE) モードに関係なく、現在の行を実行します。
<code>M</code>	(RETURN) モードに関係なく、現在の行を実行します。
<code>#</code>	コマンドの先頭文字が <code>#</code> であれば、その <code>#</code> およびそのあとのそれぞれの改行直後の <code>#</code> を削除します。先頭文字がほかの文字であれば、 <code>#</code> をコマンドの各行の先頭に挿入後、行を送ります。現在の行をコメントとして履歴に挿入する場合や、履歴ファイルに含まれる以前のコメント化されたコマンドからコメントを削除する場合に便利です。
<code>=</code>	現在のワードのあとにアスタリスクを付加した場合に一致するファイル名を一覧表示します。

`@letter` ユーザーの別名リスト中で、`_letter`という名前の別名を検索します。この名前の別名が定義されていれば、処理待ちの入力待ち行頭にその値を挿入します。

特殊コマンド

次の単純コマンドは、シェルプロセス中で実行されます。入出力のリダイレクトが可能です。特に断りのないかぎり、出力はファイル記述子1上に書き込まれ、構文エラーがなければ終了ステータスは0です。1つまたは2つのアスタリスク(*)が先頭に付加されているコマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

*: [*arg* ...]
パラメータの展開だけを行います。

*. *file* [*arg* ...]
file 全体を読み取ってからコマンドを実行します。コマンドは現在のシェル環境において実行されます。PATHによって指定された検索パスを用いて、*file*を含むディレクトリを探します。引数の*arg*は(指定されていれば)定位置パラメータになります。そうでない場合、定位置パラメータは変更されません。終了ステータスは、最後に実行されたコマンドの終了ステータスです。

** *alias* [-tx] [*name*[=*value*]] ...
引数なしの場合、このコマンドは標準出力上に *name*=*value* という形式の別名のリストを表示します。*value*が指定された名前に対しては別名を定義します。*value*の末尾にスペースがあると、次のワードが別名置換指定かどうかをチェックします。*-t* フラグは、検索済みの別名を設定または一覧表示します。検索済み別名の値は、指定した *name* に対応する完全パス名になります。*PATH*の値を再設定するとこの値は未定義になりますが、別名は検索済みのままです。*-t* フラグを省略すると、*value*が指定されていない引数リスト内の各 *name* について、別名の名前と値を表示します。*-x* フラグは、エクスポートされた別名を設定または表示します。エクスポートされた別名は、名前で起動されるスクリプト用に定義されます。*name*が指定されているが、*value*は指定されておらず、*name*に対しての別名も定義されていない場合は、終了ステータスが0以外になります。

bg [%*job*...]

このコマンドを使用できるのは、ジョブ制御をサポートするシステム上だけです。指定された各 *job* をバックグラウンドで実行します。*job* が省略された場合は、現在のジョブをバックグラウンドで実行します。*job* の記述形式については、前述の「ジョブ」節を参照してください。

* `break [n]`

`for` ループ、`while` ループ、`until` ループ、または `select` ループがあれば終了します。`n` を指定すると、`n` レベル分だけループを終了します。処理に含まれるループ数より `n` が大きい場合には、もっとも外側のループで終了します。

* `continue [n]`

`for` ループ、`while` ループ、`until` ループ、または `select` ループの次の繰り返しを実行します。`n` を指定すると、`n` 番目のループから実行します。処理に含まれるループ数より `n` が大きい場合には、最も外側のループが使用されます。

`cd [-L] [-P] [arg]`

`cd old new`

このコマンドは2つの形式のいずれかで入力します。第1の形式は、現在のディレクトリを `arg` に変更します。`arg` が `-` の場合、ディレクトリを直前のディレクトリに変更します。シェル変数 `HOME` の値がデフォルトの `arg` になります。`PWD` 環境変数は、現在のディレクトリに設定されます。`PWD` が変更された場合、`OLDPWD` 環境変数も古い作業用ディレクトリに変更されます。つまり、`cd` を呼び出す直前における現在の作業用ディレクトリのことです。シェル変数 `CDPATH` は、`arg` を含むディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:) で区切ります。デフォルトのパスは空の文字列です(現在のディレクトリの指定)。現在のディレクトリは空のパス名で指定します。このパス名は、等号の直後か、パスリスト内にある区切り文字のコロンの間に指定します。`arg` の先頭文字が `/` の場合、検索パスは使用しません。それ以外の場合は、パス中の各ディレクトリで `arg` を検索します。成功しなかった場合、`cd` は、`PWD` の値、スラッシュ文字、および `arg` を連結したパス名のディレクトリへ変更しようとしています。

-L オペランドのドットドット(..`)`を論理的に処理します。ドットドット部分が処理される前に、シンボリックリンク部分は解釈処理されません。

-P オペランドのドットドット(..`)`を物理的に処理します。ドットドット部分が処理される前に、シンボリックリンク部分は解釈処理されます。

-L と -P の両方のオプションを指定した場合、最後に呼び出された方のオプションが使用され、他方のオプションは無視されます。-L オプションと -P オプションをどちらも指定しない場合、オペランドのドットドット(..`)`は論理的に処理されます。

`cd` の第2の形式は、`PWD` 中の現在のディレクトリ名に含まれる `old` という文字列を `new` という文字列に置換し、この新規のディレクトリへ変更しようとしています。`cd` コマンドは `rksh` では実行できません。

`command [-p] [command_name] [argument ...]`

`command [-v | -V] command_name`

`command` ユーティリティーを使用すると、シェルは関数の検索を実行せずに、指定された引数を単純コマンドとして扱います。-p フラグは、すべての標準ユーティリティーの検索を可能にする `PATH` のデフォルト値を使用して、コマン

ド検索を実行します。-v フラグは、標準出力に文字列を書き出します。この文字列は、現在のシェル実行環境でシェルが *command_name* を呼び出すために使用するパス名またはコマンドを表します。-v フラグも標準出力に文字列を書き出します。この文字列は、*command_name* オペランドに指定された名前を、シェルが現在のシェル実行環境でどのように解釈するかを表します。

`echo [ arg ... ]`

使用法と説明については、[echo\(1\)](#) を参照してください。

* `eval [ arg ... ]`

引数をシェルへの入力として読み取り、生成されるコマンドを実行します。

* `exec [ arg ... ]`

arg を指定すると、このシェルの代わりに、引数で指定されたコマンドを(新規プロセスは生成せずに)実行します。入出力引数が指定可能で、現在のプロセスに影響を及ぼす場合があります。引数を指定しない場合は、ファイル記述子が入出力リダイレクションリストの指定どおりに変更されることになります。この場合、この機能を使って開かれた、ファイル記述子番号が2より大きいファイルは、別のプログラムを起動すると閉じられます。

* `exit [ n ]`

呼び出し元のシェルまたはシェルスクリプトを *n* で指定した終了ステータスで終了させます。具体的には、指定した値の最下位8ビットが終了ステータスの値となります。*n* を省略すると、最後に実行されたコマンドの終了ステータスがシェルの終了ステータスになります。トラップ実行中に `exit` が発生した場合、ここで言う最後に実行されたコマンドとは、トラップ呼び出し直前に実行されたコマンドを指します。`ignoreeof` オプションが有効になっているシェルを除き、EOFを検出した場合もシェルが終了します。[set](#) を参照してください。

** `export [ name[=value] ] ...`

** `export -p`

指定された *name* に対し、あとで実行されるコマンドの「環境」へ自動的にエクスポートされるようにマークを付けます。

-p を指定したとき、`export` は、すべてのエクスポートされる変数の名前と値を、次の形式で標準出力に書き出します。

```
"export %s=%s\n", name, value
```

これは、*name* が設定されている場合です。

```
"export %s\n", name
```

これは、*name* が設定されていない場合です。

コマンドとしてシェルに再入力できる形式で書き込まれる(引用も適切に使用される)ため、エクスポートされる変数の名前と値をシェル実行環境で再現するときに便利です。ただし、この場合、次のような制限があります。

1. 出力時に値が設定されていた読み取り専用の変数は、元の状態には戻りません。

2. 出力時に値が設定されていなかった変数に、状態を保存したときから保存した出力をシェルに再入力するまでの時間が値として割り当てられた場合、その変数は未設定の状態には戻りません(つまり、その値は保持されます)。

```
fc [-e ename] [-nlr] [first [last]]
```

```
fc -e - [old=new] [command]
```

```
fc -s [old=new] [command]
```

第1の形式では、端末から最近入力された HISTSIZE 個のコマンドの中から、*first* から *last* までの範囲のコマンドを選択します。*first* と *last* の両引数は、数値または文字列で指定できます。文字列の場合、その文字列で始まる最新のコマンドを見つけます。負の数値は、現在のコマンド番号からのオフセットとなります。*-l* オプションを指定すると、標準出力上にコマンドを一覧表示します。指定しないと、これらのキーボードコマンドの入ったファイル上で *ename* というエディタプログラムを起動します。*ename* が省略されていると、変数 FCEDIT (デフォルトは /bin/ed) の値をエディタとして使用します。編集が完了すると、編集されたコマンドを実行します。*last* を指定しない場合、*first* と同一値に設定されます。*first* を指定しない場合、デフォルトは、編集については直前のコマンドに、一覧表示については -16 になります。*-r* オプションはコマンドの順序を逆にします。*-n* オプションは一覧表示時にコマンド番号の出力を抑制します第2の形式では、*old=new* の置換を行なったあとで *command* が再実行されます。*command* 引数を省略すると、最後に行なったコマンドが実行されます。

```
fg [%job...]
```

このコマンドを使用できるのは、ジョブ制御をサポートするシステム上だけです。指定された各 *job* をフォアグラウンドで実行します。*job* が指定されないと、現在のジョブをフォアグラウンドで実行します。*job* の記述形式については、前述の「ジョブ」節を参照してください。

```
getopts optstring name [arg ...]
```

arg が正当なオプションを示しているかどうかをチェックします。*arg* を省略すると、定位置パラメータが使用されます。オプション引数は + または - で始まります。+ または - 以外の文字で始まっているオプション、また - 引数があると、オプションの終わりともみなされます。*optstring* には、getopts が認識する文字を記述します。文字のあとに : が続く場合、そのオプションには引数があるとみなされます。オプションと引数とはブランクで区切ることができます。

arg が + で始まる場合、getopts は + を起動するたびに *name* 内で見つかった次のオプション文字を設定します。次の *arg* のインデックスは OPTIND に格納されます。オプション引数がある場合は OPTARG に格納されます。

optstring 内で先頭に : がある場合、getopts は無効なオプション文字を OPTARG に格納し、*name* を ? (未定義のオプションが指定された場合) または : (必要なオプション引数が省略されている場合) に設定します。getopts はエラーメッセージを表示します。オプションがなくなると、終了ステータスは 0 以外になります。使用法と説明については、getoptcvt(1) を参照してください。

getopts は、従来の単一文字の短いオプションも、Sun の Command Line Interface Paradigm (CLIP) で定義された長いオプションもサポートします。

長いオプションは短いオプションの別名です。長いオプションを指定するには、対応する短いオプションのあとに、長いオプションを括弧に囲んで指定します。たとえば、短いオプション「f」の別名として、長いオプション「file」を指定する場合、次のスクリプト行を使用します。

```
getopts "f(file)" opt
```

コマンド行上では、長いオプションの前に「--」または「++」を指定します。前述の例の場合、コマンド行上の「--file」は「-f」と同等であり、「++file」は「+f」と同等です。

短いオプションの別名として長いオプションを複数指定することはできますが、これは CLIP 仕様には違反することであり、使用するときには十分注意する必要があります。長いオプションを複数指定するときには、次のように、それぞれの長いオプションを括弧で囲む必要があります。

```
getopts "f:(file)(input-file)o:(output-file)"
```

前述の例の場合、「--file」と「--input-file」は両方とも「-f」と同等であり、「--output-file」は「-o」と同等です。

変数名は、常に、短いオプションに設定されます。コマンド行上で長いオプションを指定しても、変数名はそれと同等の短いオプションに設定されます。

```
hash [ name ... ]
```

```
hash [ -r ]
```

シェルは、各 *name* ごとに、それが示すコマンドの検索パス内の位置を決定し、記憶します。*-r* オプションを指定すると、シェルは記憶したすべての位置を忘れます。引数をまったく指定しないと、記憶されたコマンドに関する情報が表示されます。*hits* はシェルプロセスによってコマンドが呼び出された回数を表示します。*cost* は、検索パスのコマンドを見つけるのに必要な作業です。コマンドが検索パスの相対ディレクトリにある場合、そのディレクトリの変更後にそのコマンドが格納された位置が再計算されます。この再計算が行われるコマンドに対しては、*hits* 情報の隣にアスタリスク(*)が示されます。*Cost* の値は、再計算が行われるたびに増加されます。

```
jobs [ -lnp ] [ %job ... ]
```

指定された各ジョブに関する情報を一覧表示します。*job* 引数を省略すると、活動中のジョブすべてに関する情報を一覧表示します。*-l* フラグは、通常の情報に加えてプロセス ID も表示します。*-n* フラグは、前回通知を受けたあとに停止または終了したジョブだけを表示します。*-p* フラグは、プロセスグループだけを表示します。*job* の記述形式については、前述の「ジョブ」節と [jobs\(1\)](#) を参照してください。

```
kill [ -sig ] %job ...
```

```
kill [ -sig ] pid ...
```

kill -l

TERM(終了)シグナルまたは指定されたシグナルのいずれかを、指定されたジョブまたはプロセスに送信します。シグナルは番号または名前で指定します。名前は、[signal.h\(3HEAD\)](#)に示された名前から「SIG」という接頭辞を取り除いたものです。ただし、SIGCHDはCHLDという名前になります。送信するシグナルがTERM(終了)またはHUP(ハングアップ)の場合、停止中のジョブまたはプロセスにはCONT(継続)シグナルを送信します。*job*という引数は、活動中のジョブではないプロセスのプロセスIDを指定することもできます。*job*の記述形式については、「ジョブ」を参照してください。第2の形式のkill -lは、シグナル番号とシグナル名をリスト表示します。

let arg...

各*arg*は、評価の対象となる個々の算術式を表します。評価の方法については、前述の「算術評価」節を参照してください。

終了ステータスは、最後の式の値が0以外の場合は0で、0の場合には1です。

login argument...

'exec login argument....'と同機能です。使用法と説明については、[login\(1\)](#)を参照してください。

*** newgrp [arg ...]**

exec /bin/newgrp arg と同機能です。

print [-Rnprsu[n]] [arg ...]

シェルの出力メカニズムです。フラグを省略した場合、あるいは-または--フラグを指定した場合は、[echo\(1\)](#)で述べるように標準出力上に引数を表示します。出力ファイルが書き込み用に開いていない場合を除き、終了ステータスは0になります。

- n 復帰改行 (NEWLINE) の出力を抑止します。
- R | -r (raw モード) echo のエスケープ規則を無視します。-R オプションは、-nを除く後続の引数およびオプションすべてを表示します。
- p 標準出力の代わりに |&で生成されたプロセスのパイプに引数を出力します。
- s 標準出力の代わりに履歴ファイルに引数を書き込みます。
- u [*n*] 出力を格納するファイル記述子番号を、1桁の数値 *n* で指定します。デフォルト値は1です。

pwd [-L | -P]

ファイル名のドット(.)またはドットドット(..)を含まない、現在の作業用ディレクトリの絶対パス名を標準出力に書き出します。

- L PWD 環境変数に、現在のディレクトリの絶対パス名(ファイル名にドットまたはドットドットを含まない)が含まれている場合は、pwd はこのパス名を

標準出力に書き出します。それ以外の場合、`-L` オプションは `-P` オプションと同様の結果を書き出します。

`-P` 書き込まれる絶対パス名にファイル名が含まれません。パス名のコンテキストでは、シンボリックリンク形式でファイル名が参照されます。

`-L` と `-P` の両方が指定された場合は、最後に指定されたオプションが適用されます。`-L` も `-P` も指定されない場合、`pwd` は `-L` オプションと同様の結果を書き出します。

`read [-prsu[n]] [name?prompt] [name ...]`

シェルの入力機構。1つの行を読み取り、IFS が示す文字を区切り文字として使用して、複数のフィールドに分割します。エスケープ文字 (`\`) は、次の文字の特別な意味または行の継続に関する意味を取り除くために使用します。`-r` で指定する raw モードでは、`\` が持つこの特殊な意味は無視されます。第1フィールドを1番目の *name* に、第2フィールドを2番目の *name* に、という順番で割り当てていき、余ったフィールドがあれば最後の *name* に割り当てます。`-p` オプションは、シェルが `|&` を使用して生成したプロセスの入力パイプから入力行を取り出します。`-s` フラグは、入力をコマンドとして履歴ファイルに保存します。`-u` フラグは、読み取り元となるファイル記述子番号を1桁の数値 *n* で指定します。ファイル記述子は、`exec` という特殊コマンドで開くことができます。*n* のデフォルト値は `0` です。*name* を省略すると、`REPLY` の値をデフォルトとして使用します。入力ファイルが読み込み用に開いていない場合と EOF に到達した場合を除き、終了ステータスは `0` です。`-p` オプションが指定されていて EOF を検出すると、このプロセスをクリアして別のプロセスを作成可能にします。最初の引数が `?` を含んでいると、シェルが対話型のとき、このワードの残りを標準エラーに対するプロンプトとして使用します。EOF に到達しないかぎり、終了ステータスは `0` です。

**** readonly [name=value] ...**

**** readonly -p**

指定された *name* を読み取り専用にします。あとの代入でこれらの名前を変更できないようにします。

`-p` を指定したとき、`readonly` は、すべての読み取り専用変数の名前と値を、次の形式で標準出力に書き出します。

```
"readonly %s=%s\n", name, value
```

これは、*name* が設定されている場合です。

```
"readonly $s\n", name
```

これは、*name* が設定されていない場合です。

コマンドとしてシェルに再入力できる形式で書き込まれるため、同じ読み取り専用の属性と値をシェル実行環境で再現するときに便利です。保存しておいた出力をシェルに再入力するときには、次のような制限があります。

1. 出力時に値が設定されていた変数には、読み取り専用の属性は設定されません。
2. 出力時に設定されていなかった変数には、値は設定されません。

* `return [ n ]`

シェル関数または、スクリプトを、*n* で指定された戻りステータスで呼び出し側スクリプトに戻します。具体的には、指定した値の最下位 8 ビットが終了ステータスの値となります。*n* を省略すると、戻りステータスは最後に実行されたコマンドの戻りステータスになります。`return` を関数や、'.' スクリプト実行中以外で起動すると、結果は `exit` と同一になります。

`set [ ±abCefghkmnopstuvx ] [ ±o option ]... [ ±A name ] [ arg ... ]`

このコマンドのフラグの意味は次のとおりです。

- A 配列の代入。*name* で示される変数の設定を解除し、*arg* リストから順々に値を割り当てます。+A は、最初の変数設定を解除しません。
- a 定義される後続の変数すべてを自動的にエクスポートします。
- b バックグラウンドジョブの完了を非同期的にユーザーに通知します。次のメッセージが標準エラー出力に書き込まれます。

```
"[%d]%c %s%s\n", <job-number>, <current>, <status>, \
    whe<job-name>
```

各フィールドの意味を次に説明します。

<current> 文字 + は、fg または bg ユーティリティー用のデフォルトとして使用するジョブを表します。このジョブは、*job_id* %+ または %% を使って指定することもできます。文字 - は、現在のデフォルトジョブが終了したときにデフォルトとなるジョブを表します。このジョブは、*job_id* %- を使って指定することもできます。その他のジョブでは、このフィールドはスペース文字となります。+ や - を使って表せるジョブの数は、どちらも最大 1 つです。停止中のジョブがあれば、現在のジョブも停止ジョブとなります。停止中のジョブが 2 つ以上あれば、以前のジョブも停止ジョブとなります。

<job-number> wait、fg、bg、kill の各ユーティリティー用にプロセスグループを識別するのに使用する番号。これらのユーティリティーを使うと、ジョブはジョブ番号の前に % を付加することにより識別できます。

<status> 指定なし。

<job-name> 指定なし。

シェルは、ジョブの終了をユーザーに通知したときに、そのジョブのプロセス ID を現在のシェル実行環境で認識されているプロセス ID のリストから削除することがあります。非同期の通知はデフォルトでは無効です。

- c シェルのリダイレクト演算子「>」によって既存のファイルが上書きされるのを防ぎます。リダイレクト演算子「>|」は、個々のファイルに対して、このオプションよりも優先されます。-
- e コマンドの終了ステータスが 0 でない場合、ERR トラップ (設定されていれば) を実行し、終了します。このモードは、プロファイル読み取り時は無効です。
- f ファイル名を生成しないようにします。
- h 各コマンドは、最初に検出された時点で、検索済み別名になります。
- k コマンド名に先行するものだけでなく、すべての変数代入引数をコマンドの環境に格納します。
- m バックグラウンドジョブを個別のプロセスグループで実行し、完了時にメッセージを 1 行表示します。バックグラウンドジョブの終了ステータスは完了メッセージで報告されます。ジョブ制御を備えたシステムでは、このフラグは対話型シェルに対して自動的に有効になります。
- n コマンドを読み取り、構文エラーがないかチェックします。実行は行いません。対話型シェルに対しては無視されます。
- o 現在のオプション設定を標準出力に書き出します。コマンドとしてシェルに再入力できる形式で書き込まれるため、同じオプション設定を再現するときに便利です。
- o このフラグのあとに指定する引数は、次のいずれかのオプション名です。

allexport -a と同じです。

errexit -e と同じです。

bgnice バックグラウンドジョブをすべて低い優先度で実行します。これはデフォルトモードです。

emacs コマンド入力用に、emacs 形式のインラインエディタを起動します。

gmacs コマンド入力用に、gmacs 形式のインラインエディタを起動します。

ignoreeof EOF を検出してもシェルは終了しません。終了させるには exit コマンドを使用する必要があります。

keyword -k と同じです。

markdirs	ファイル名生成によって生成されるディレクトリ名には、すべて最後に / を付加します。
monitor	-m と同じです。
noclobber	リダイレクト演算子「>」によって既存のファイルが上書きされるのを防ぎます。このオプションが有効なときにファイルを切り捨てるには、> が必要です。-c と同等です。
noexec	-n と同じです。
noglob	-f と同じです。
nolog	履歴ファイルに関数定義を保存しません。
notify	-b と同等です。
nounset	-u と同じです。
privileged	-p と同じです。
verbose	-v と同じです。
trackall	-h と同じです。
vi	vi 形式のインラインエディタの挿入モードになります。033 というエスケープ文字を押すと、これにより制御モードになります。Return で行を送信します。
viraw	各文字を vi モードで入力されたときと同様に処理します。
xtrace	-x と同じです。

オプション名を指定しないと、現在のオプション設定を表示します。

- p \$HOME/.profile ファイルを処理しないようにし、ENV ファイルの代わりに /etc/suid_profile ファイルを使用します。このモードは、実効ユーザー ID が実ユーザー ID と等しくないとき、また実効グループ ID が実グループ ID と等しくないときには必ず有効になります。このモードを無効にすると、実効ユーザー ID が実ユーザー ID に、実効グループ ID が実グループ ID にそれぞれ設定されます。
- s 定位置パラメータを辞書編集方式の順にソートします。
- t 1つのコマンドを読み取り、実行したあと、終了します。
- u 置換を行う際に、設定されていないパラメータをエラーとして扱います。
- v シェル入力行の読み取り時に、その内容を表示します。
- x コマンドの実行時に、コマンドと引数の内容を表示します。

- -x フラグと -v フラグを無効にし、フラグに対する引数の検査を停止します。
 - どのフラグも変更しません。\$1 を - で始まる値に設定する際に便利です。このフラグの後に引数がない場合、定位置パラメータが設定解除されます。
- の代わりに + を使用すると、これらのフラグは無効になります。これらのフラグはシェル起動時にも使用できます。現在セットされているフラグは、\$- で見つけられます。-A を指定しないかぎり、残りの引数は定位置パラメータとなり、\$1 \$2 に順番に割り当てられます。引数を 1 つも指定しない場合には、すべての変数の名前と値を標準出力上に表示します。

* shift [*n*]

\$*n*+1 \$*n*+1 . . . の定位置パラメータを \$1 . . . という名前に変更します。*n* のデフォルト値は 1 です。パラメータ *n* に指定できる値は、評価結果が \$# 以下の負でない数になる算術式です。

stop%*jobid* ...

stop *pid* ...

stop は、*jobid* (ジョブ ID 番号) を指定するとバックグラウンドジョブの実行を中断し、*pid* (プロセス ID 番号) を指定するとすべてのプロセスを中断します。ps(1) を参照してください。

suspend

現在のシェルがログインシェルでない場合、その実行を中断します。

test *expression*

条件式を評価します。使用法と説明については、前述の「条件式」節と test(1) を参照してください。

* times

シェルおよびシェルから実行されたプロセスの、ユーザー時間およびシステム時間の累計値を表示します。

* trap [*arg sig* ...]

arg は、*sig* が示すシグナルをシェルが受信したときに読み取られ、実行されるコマンドです。*arg* は、トラップ設定時とトラップ取り出し時に 1 度ずつ検索されます。*sig* には、シグナル番号またはシグナル名を指定します。trap コマンドは、シグナル番号の順序で実行されます。現在のシェルへの入力時に無視されたシグナル番号にトラップを設定しようとしても無効となります。

arg が - の場合、シェルは各 *sig* をデフォルト値に再設定します。*arg* が NULL (') の場合、シェルは指定された各 *sig* が発生してもそれを無視します。それ以外の場合は、対応する *sig* が 1 つでも発生したときに、*arg* がシェルによって読み取られ、実行されます。トラップのアクションは、以前のアクション (デフォルトまたは明示的に設定されたもの) より優先して使用されます。トラップのアクションが完了すると、\$? の値はトラップが呼び出されたときの値になります。

sig は、EXIT、0 (EXIT と同義)、またはシンボル名を使って指定したシグナルから接頭辞 SIG を除いたものです。たとえば HUP、INT、QUIT、TERM などです。*sig* が 0 または EXIT であり、ある関数の内部で `trap` 文が実行されると、関数の終了後に *arg* の示すコマンドが実行されます。*sig* が 0 または EXIT であり、関数の外部にトラップが設定されている場合、シェルの終了時に *arg* の示すコマンドが実行されます。*sig* が ERR の場合は、コマンドが 0 以外の終了ステータスで終わると必ず *arg* が実行されます。*sig* が DEBUG の場合は、各コマンドのあとで *arg* が実行されます。

シェルが EXIT に対してトラップを実行する環境は、EXIT のトラップを取り出す前に実行された最後のコマンドの直後の環境と同じです。

トラップが呼び出されるたびに、*arg* 引数は `eval "$arg"` と同じように処理されます。

非対話型シェルの開始時に無視されたシグナルは、トラップもリセットもできません。ただし、トラップやりセットを試みても、エラーは報告されません。対話型シェルは、呼び出し時に無視されたシグナルをリセットしたりキャッチしたりできます。トラップは、そのシェルの動作中、ほかの `trap` コマンドにより明示的に変更されるまで有効であり続けます。

サブシェルを起動すると、トラップがデフォルト引数に設定されます。これは、サブシェル内で `trap` コマンドを使って新しいトラップを設定できないことを意味するものではありません。

引数なしの `trap` コマンドは、各シグナルに対応したコマンドの一覧を標準出力に書き出します。その形式は次のとおりです。

```
trap — %s %s ... <arg>, <sig> ...
```

この出力は、同じトラップ結果をもたらすコマンドとしてシェルに再入力できる形式で書き込まれます (引用も適切に使用されます)。例:

```
save_traps=$(trap)
...
eval "$save_traps"
```

トラップの名前や番号が正しくないと、ゼロ以外の終了ステータスが返されます。正しければ 0 が返されます。対話型シェルと非対話型シェルのどちらでも、無効なシグナル名やシグナル番号は構文エラーとみなされず、シェルは異常終了しません。

ジョブがフォアグラウンドプロセスを待っている間は、トラップは処理されません。このため、CHLD に対するトラップはフォアグラウンドジョブが終了するまで実行されません。

`type name ...`

コマンドは、*name* ごとに、コマンド名として使用される場合にどのように解釈されるかを指示します。

**** typeset [±HLRZfилrtux[n]] [name[=value]]...**

シェル変数と関数の属性と値を設定します。関数内で **typeset** を実行すると、**name** が示す変数の新しいインスタンスが生成されます。関数が完了すると、その変数の値と型が復元されます。このコマンドには、次の属性を指定できます。

- H このフラグは UNIX 以外のマシン上で、UNIX とホスト名ファイルとのマッピング情報を提供します。
- L 左詰めを行い、先行するブランクを *value* から取り除きます。*n* は、ゼロ以外であればフィールドの幅を定義します。そうでなければ、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ右側にブランクが詰められ、長ければ切り捨てられます。**-Z** フラグも設定されていれば、先行する 0 を削除します。**-R** フラグは無効になります。
- R 右詰めを行い、先行するブランクを挿入します。*n* は、0 以外であればフィールドの幅を定義します。0 の場合、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ左側にブランクが詰められ、長ければ終端が切り捨てられます。**-L** フラグは無効になります。
- Z ブランク以外の最初の文字が数字であり、**-L** フラグが設定されていない場合は、右詰めを行い、先頭に 0 を詰めます。*n* は、ゼロ以外であればフィールドの幅を定義します。そうでなければ、フィールドの幅は最初に代入される値の幅で決定されます。
- f 名前は、変数名ではなく関数名を指します。代入は行われません。このフラグとともに指定できるほかのフラグは、**-t**、**-u**、および **-x** だけです。**-t** フラグは、この関数の実行トレースを有効にします。**-u** フラグは、この関数に「未定義」を示すマークを付けます。関数が参照されると、関数定義を見つけるために **FPATH** 変数が検索されます。**-x** フラグを指定すると、名前呼び出されるシェル手続き全体で関数定義が有効になります。
- i パラメータを整数とします。これにより算術演算が高速化されます。*n* は、0 以外であればその値を底として定義します。0 の場合、最初の代入で底が決定されます。
- l 大文字をすべて小文字に変換します。大文字への変換を示す **-u** フラグを無効にします。
- r 指定された *name* を読み取り専用にします。あとの代入でこれらの名前を変更できないようにします。
- t 変数にタグを付けます。タグはユーザーが定義可能で、シェルに対して特別な意味を持ちません。

- u 小文字をすべて大文字に変換します。小文字への変換を示す -l フラグを無効にします。
- x 指定された *name* に対し、あとで実行されるコマンドの「環境」へ自動的にエクスポートされるようにマークを付けます。

-i 属性は、-R、-L、-Z、または -f と同時に指定できません。

- の代わりに + を使用すると、これらのフラグは無効になります。*name* 引数をまったく指定せずにフラグを指定すると、これらのフラグが設定されている変数の名前(および選択によっては値も)が一覧表示されます。(- の代わりに + を使用すると、値は出力されません。) *name* 引数とフラグを 1 つも指定しないと、すべての変数の名前と属性が表示されます。

`ulimit [-HSacdfnstv] [limit]`

リソース制限値を設定または表示します。使用可能なリソース制限値については、次のセクションで説明します。システムによっては、次に挙げたすべての資源の制限を提供していないこともあります。*limit* を指定すると、指定したリソースの制限値が設定されます。*limit* の値は、各リソースに指定された単位の数値、または `unlimited` という値です。文字列 `unlimited` は、現在の制限値(ある場合)の削除を要求します。-H フラグと -s フラグは、指定したリソースに対して強い制限と弱い制限のどちらを設定するかを表します。強い制限値は、いったん設定したらあとで値を上げることはできません。弱い制限値は、強い制限値を超えない範囲で値を上げることが可能です。-H も -s も省略すると、指定した制限値が強い制限と弱い制限の両方に適用されます。*limit* 引数を省略すると、現在のリソース制限値が表示されます。このとき、-H が指定された場合を除き、表示されるのは弱い制限値です。複数のリソースを指定すると、値の前に制限値の名前と単位が表示されます。

- a 現在のリソース制限値をすべて表示します。
- c コアダンプのサイズをブロック (512 バイト) 単位で表します。
- d データ領域のサイズを K バイト単位で表します。
- f 子プロセスが書き込むファイルのサイズをブロック (512 バイト) 単位で表します。読み込むファイルのサイズに制限はありません。
- n ファイル記述子数に 1 を加えた値を表します。
- s スタック領域のサイズを K バイト単位で表します。
- t 各プロセスが使用する秒数を表します。
- v 仮想記憶のサイズを K バイト単位で表します。

オプションをすべて省略すると、-f が指定されたものとみなします。

`umask [-S] [mask]`

ユーザーファイルの作成時のマスクを *mask* が示す値に設定します (`umask(2)` を参照)。 *mask* には、`chmod(1)` で説明する記号値または 8 進数を指定できます。記号値を指定すると、新しい *umask* 値は、*mask* を直前の *umask* 値の補数に適用した結果の補数になります。 *mask* 引数を省略すると、マスクの現在の値を表示します。 `-s` フラグは、シンボリック形式の出力を生成します。

`unalias name ...`

`unalias -a`

name のリストで指定された別名を別名リストから削除します。 `-a` オプションは、現在の実行環境からすべての別名定義を削除します。

`unset [-f] name ...`

name が示す変数の設定を解除します。つまり、それらの変数の値と属性を消去します。読み取り専用の変数は設定を解除できません。 `-f` フラグが設定されていると、*name* 引数は関数名を表しま

す。 `ERRNO`、`LINENO`、`MAILCHECK`、`OPTARG`、`OPTIND`、`RANDOM`、`SECONDS`、`TMOUT`、および `_` の設定を解除すると、これらの変数の特殊な意味が削除されます。あとでこれらの変数に値を代入しても、特殊な意味はないままです。

`*wait [job]`

指定された *job* の終了を待ち、その終了ステータスを報告します。 *job* を指定しないと、現在実行中のすべての子プロセスを待ちます。待つ対象のプロセスの終了ステータスが、このコマンドの終了ステータスになります。 *job* の記述形式については、「ジョブ」を参照してください。

`whence [-pv] name ...`

コマンドは、*name* ごとに、コマンド名として使用される場合にどのように解釈されるかを指示します。

`-v` フラグをつけると、より冗長に表示されます。

`-p` フラグをつけると、コマンド名が別名、関数、または予約語である場合でも *name* のパスが検索されます。

呼び出し

シェルを `exec(2)` で呼び出し、0 番目の引数 (`$0`) の最初の文字が `-` である場合は、シェルをログインシェルとみなし、`/etc/profile` からコマンドを読み取ります。次に、現在のディレクトリ内に `.profile` が存在するか、または `$HOME/.profile` がある場合は、そのいずれかのファイルからコマンドを読み取ります。次に、環境変数 `ENV` に設定されている値をパラメータ置換することによって指定されるファイルが存在する場合は、そのファイルからコマンドを読み取ります。 `-s` フラグが省略され、*arg* 引数が指定されている場合は、最初の *arg* に対してパス検索を実行し、実行するスクリプトの名前を判定します。 *arg* が示すスクリプトには読み取り権が必要で、`setuid` と `setgid` の設定は無視されます。パス上でスクリプトが見つからない場合、*arg* は組み込みコマンドまたは組み込み関数の名前を示しているものとして処理されます。次に、後述の方法でコマンドが読み取られます。シェルを起動すると、次のフラグがシェルによって解釈されます。

- c *command_string* オペランドからコマンドを読み取ります。特殊パラメータ 0 の値は、*command_name* オペランドの値と残りの *arg* オペランドに含まれる定位置パラメータ (\$1、\$2 など) から順に設定されます。標準入力から読み取られるコマンドはありません。
- s -s フラグが指定された場合、または引数が残っていない場合は、標準入力からコマンドを読み取ります。前述の特殊コマンドの出力を除くシェル出力は、ファイル記述子 2 に書き込まれます。
- i -i フラグが指定された場合、またはシェル入出力が端末に接続されている場合 (*ioctl*(2) を参照)、このシェルは対話型になります。この場合、kill 0 が対話型シェルを終了しないように TERM を無視し、wait が割り込み可能になるように INTR を捕らえ、無視します。いずれの場合も、シェルは QUIT を無視します。
- r -r フラグを指定すると、シェルは制限付きシェルになります。

他のフラグと引数については、前述の *set* コマンドの箇所の説明されています。

rksh の特記事項

rksh は、標準シェルより機能が制限されたログイン名や実行環境を設定するために使用します。rksh の機能は、次の動作ができない点を除いて ksh と同じです。

- ディレクトリの変更 (*cd*(1) を参照)
- SHELL、ENV、または PATH の値の設定
- / を含むパス名またはコマンド名の指定
- 出力先のリダイレクト (>、>|、<>、および >>)
- グループの変更 (*newgrp*(1) を参照)

前述の制限は、*.profile* ファイルと ENV ファイルの解釈後に有効となります。

実行するコマンドがシェル手続きである場合、rksh は ksh を呼び出してコマンドを実行します。したがって、利用できるコマンドの種類には制限はありますが、一般ユーザーは標準シェルの全機能を利用できるシェル手続きを使用することができます。このスキームは、一般ユーザーが同じディレクトリで書き込み権と実行権を持っていないことを想定しています。

これらの規則の実際の効果は、*.profile* の作成者が確実な設定処理を実行してユーザーを適切な (おそらく、ログインディレクトリ以外の) ディレクトリに置くことにより、ユーザーの動作を完全に制御できるという点にあります。

システム管理者は、多くの場合、rksh で安全に起動できるコマンドのディレクトリ (つまり、*/usr/rbin*) を設定します。

エラー

構文エラーなどのエラーを検出すると、シェルは 0 以外の終了ステータスを返します。そうでない場合は、シェルは最後に実行されたコマンドの終了ステータスを返します (前述の *exit* コマンドの説明を参照)。シェルを非対話型で使用している場合、シェルファイルの実行は中止されます。シェルが検出する実行時エラーは、コ

マンド名または関数名、およびエラー状態を表示することによって報告されます。エラーが発生した行の番号が1より大きい場合は、コマンド名または関数名のあとに角括弧([1])で囲んだ行番号も表示します。

非対話型シェルの場合は、特殊組み込みユーティリティやほかの種類のユーティリティがエラー状態を検出すると、シェルは診断メッセージを書き出し、次の表に示すように終了します。

エラー	特殊組み込み ユーティリティ	その他のユーティリ ティ
シェル言語の構文エラー	終了する	終了する
ユーティリティの構文エラー (オプションまたはオペランドのエラー)	終了する	終了しない
リダイレクトのエラー	終了する	終了しない
変数代入エラー	終了する	終了しない
展開エラー	終了する	終了する
コマンドが見つからない	該当せず	終了する場合があります
ドットスクリプトが見つからない	終了する	該当せず

展開エラーとは、シェル展開の実行時に発生するエラーです。たとえば、`${x!y}`は!が有効な演算子ではないのでエラーになります。これらのエラーを展開時ではなくトークン化時に検出できる実装では、これらのエラーを構文エラーとして扱うことができます。

前述の表で「終了する」(または「終了する場合があります」)と示されているエラーがサブシェル内で発生した場合、サブシェルはゼロ以外のステータスで終了します(または終了する場合があります)が、サブシェルを含むスクリプトはエラーによって終了しません。

前述の表のどの場合でも、対話型シェルは診断メッセージを標準エラーに書き出すだけで、終了はしません。

使用法 ファイルが2G バイト (2³¹ バイト) 以上ある場合の ksh と rksh の動作については、[largefile\(5\)](#) を参照してください。

終了ステータス 各コマンドには、ほかのシェルコマンドの動作に影響を与える可能性のある終了ステータスが定義されています。ユーティリティを除くコマンドの終了ステータスについては、このセクションで説明します。標準ユーティリティの終了ステータスについては、対応する各節で説明されています。

コマンドが見つからない場合、終了ステータスは 127 になります。コマンド名は見つかったが実行可能なユーティリティーではない場合、終了ステータスは 126 になります。シェルを使わないでユーティリティーを呼び出すアプリケーションでは、これらの終了ステータス値を使って同様のエラーを報告するようにしてください。

ワードの展開中またはリダイレクション中にコマンドが失敗すると、その終了ステータスはゼロより大きい値になります。

シェルは、特殊パラメータ ? 付きの終了ステータスを報告するときに、利用可能な終了ステータスの 8 ビットすべてを報告します。報告されたシグナルを受け取ったために終了したコマンドの終了ステータスは、128 より大きな値になります。

ファイル

/etc/profile

/etc/suid_profile

\$HOME/.profile

/tmp/sh*

/dev/null

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/sunos/bin/ksh、/usr/sunos/bin/rksh	属性タイプ	属性値
	使用条件	shell/ksh88
	CSI	有効

/usr/xpg4/bin/sh	属性タイプ	属性値
	使用条件	system/core-os
	CSI	有効
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

関連項目

[cat\(1\)](#), [cd\(1\)](#), [chmod\(1\)](#), [cut\(1\)](#), [echo\(1\)](#), [env\(1\)](#), [getoptcv\(1\)](#), [jobs\(1\)](#), [login\(1\)](#), [newgrp\(1\)](#), [paste\(1\)](#), [pfksh\(1\)](#), [pfexec\(1\)](#), [ps\(1\)](#), [shell_builtins\(1\)](#), [stty\(1\)](#), [test\(1\)](#), [vi\(1\)](#), [dup\(2\)](#), [exec\(2\)](#), [fork\(2\)](#), [ioctl\(2\)](#), [lseek\(2\)](#), [pipe\(2\)](#), [ulimit\(2\)](#), [umask\(2\)](#), [rand\(3C\)](#), [signal\(3C\)](#), [signal.h\(3HEAD\)](#), [wait\(3C\)](#), [a.out\(4\)](#), [profile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

Morris I.Bolsky、David G.Korn 共著、『The KornShell Command and Programming Language』、Prentice Hall 発行、1989 年。

- 警告** シェルスクリプトを `setuid` して使用することは避けてください。
- 注意事項** 検索済み別名であるコマンドを実行したあとで、検索パスの中で元のコマンドがあるディレクトリよりも前に位置するディレクトリに同じ名前のコマンドがインストールされても、シェルは元のコマンドの方を実行し続けます。新しい方のコマンドを実行するには、`alias` コマンドの `-t` オプションを使用してください。
- 非常に古いシェルスクリプトの中には、パイプ文字 `|` の同義語として `^` を使用するものがあります。
- 複合コマンドの中に `fc` 組み込みコマンドを指定すると、すべてのコマンドが履歴ファイルから削除されます。
- `.file` という組み込みコマンドは、コマンドの実行前にファイル全体を読み取ります。したがって、ファイル内の `alias` コマンドと `unalias` コマンドは、ファイル内に定義されたどの関数にも適用されません。
- 存在しないコマンドのインタプリタを実行しようとするシェルスクリプトを、シェルが処理した場合、シェルはシェルスクリプトが存在しないという間違った診断メッセージを返します。

名前	ktutil – Kerberos keytab 保守ユーティリティ																
形式	/usr/bin/ktutil																
機能説明	ktutil コマンドは、keytab ファイルで鍵リストを管理するための対話型のコマンド行インタフェースユーティリティです。管理するには、その前に keytab の鍵リストを読み込んでおく必要があります。また、ktutil コマンドを実行するユーザーには、keytab の読み取り/書き込み権が必要です。たとえば、keytab をルートユーザーが所有している場合 (一般的な場合)、ktutil は、適切なアクセス権が得られるように、ルートユーザーが実行する必要があります。																
コマンド	<table> <tr> <td><code>clear_list</code> <code>clear</code></td><td>現在の鍵リストをクリアします。</td></tr> <tr> <td><code>read_kt file</code> <code>rkt file</code></td><td>keytab を現在の鍵リストに読み込みます。読み込む keytab ファイル (<i>file</i>) を指定する必要があります。</td></tr> <tr> <td><code>write_kt file</code> <code>wkt file</code></td><td>現在の鍵リストを keytab ファイル (<i>file</i>) に書き込みます。書き込む keytab ファイルを指定する必要があります。keytab ファイルがすでに存在する場合は、現在の鍵リストが既存の keytab ファイルに付加されます。</td></tr> <tr> <td><code>add_entry number</code> <code>addent number</code></td><td>現在の鍵リストにエントリを追加します。鍵リストのロット番号でエントリを指定します。</td></tr> <tr> <td><code>delete_entry number</code> <code>delent number</code></td><td>現在の鍵リストからエントリを削除します。鍵リストのロット番号でエントリを指定します。</td></tr> <tr> <td><code>list</code> <code>l</code></td><td>現在の鍵リストを一覧表示します。</td></tr> <tr> <td><code>list_request</code> <code>lr</code></td><td>使用できる要求 (コマンド) を一覧表示します。</td></tr> <tr> <td><code>quit</code> <code>exit</code> <code>q</code></td><td>ユーティリティを終了します。</td></tr> </table>	<code>clear_list</code> <code>clear</code>	現在の鍵リストをクリアします。	<code>read_kt file</code> <code>rkt file</code>	keytab を現在の鍵リストに読み込みます。読み込む keytab ファイル (<i>file</i>) を指定する必要があります。	<code>write_kt file</code> <code>wkt file</code>	現在の鍵リストを keytab ファイル (<i>file</i>) に書き込みます。書き込む keytab ファイルを指定する必要があります。keytab ファイルがすでに存在する場合は、現在の鍵リストが既存の keytab ファイルに付加されます。	<code>add_entry number</code> <code>addent number</code>	現在の鍵リストにエントリを追加します。鍵リストのロット番号でエントリを指定します。	<code>delete_entry number</code> <code>delent number</code>	現在の鍵リストからエントリを削除します。鍵リストのロット番号でエントリを指定します。	<code>list</code> <code>l</code>	現在の鍵リストを一覧表示します。	<code>list_request</code> <code>lr</code>	使用できる要求 (コマンド) を一覧表示します。	<code>quit</code> <code>exit</code> <code>q</code>	ユーティリティを終了します。
<code>clear_list</code> <code>clear</code>	現在の鍵リストをクリアします。																
<code>read_kt file</code> <code>rkt file</code>	keytab を現在の鍵リストに読み込みます。読み込む keytab ファイル (<i>file</i>) を指定する必要があります。																
<code>write_kt file</code> <code>wkt file</code>	現在の鍵リストを keytab ファイル (<i>file</i>) に書き込みます。書き込む keytab ファイルを指定する必要があります。keytab ファイルがすでに存在する場合は、現在の鍵リストが既存の keytab ファイルに付加されます。																
<code>add_entry number</code> <code>addent number</code>	現在の鍵リストにエントリを追加します。鍵リストのロット番号でエントリを指定します。																
<code>delete_entry number</code> <code>delent number</code>	現在の鍵リストからエントリを削除します。鍵リストのロット番号でエントリを指定します。																
<code>list</code> <code>l</code>	現在の鍵リストを一覧表示します。																
<code>list_request</code> <code>lr</code>	使用できる要求 (コマンド) を一覧表示します。																
<code>quit</code> <code>exit</code> <code>q</code>	ユーティリティを終了します。																

使用例

例1 ファイルからの主体の削除

次の例では、host/denver@ACME.com 主体を /etc/krb5/krb5.keytab ファイルから削除します。既存の keytab からエントリを削除する場合は、最初に鍵リストを一時 keytab に書き込み、次に既存の keytab をその一時 keytab で上書きする必要があることに注意してください。これは、wkt コマンドが実際には現在の鍵リストを既存の keytab に付加してしまうので、これを使用して keytab を上書きできないためです。

例1 ファイルからの主体の削除 (続き)

```
example# /usr/krb5/bin/ktutil
ktutil: rkt /etc/krb5/krb5.keytab
ktutil: list
slot KVNO Principal
-----
1      8 host/vail@ACME.COM
2      5 host/denver@ACME.COM
ktutil:delent 2
ktutil:l
slot KVNO Principal
-----
1      8 host/vail@ACME.COM
ktutil:wkt /tmp/krb5.keytab
ktutil:q
example# mv /tmp/krb5.keytab /etc/krb5/krb5.keytab
```

ファイル /etc/krb5/krb5.keytab Kerberos クライアントの keytab ファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
インタフェースの安定性	下記を参照。

コマンド引数は「確実」です。コマンド出力は「不確実」です。

関連項目 [kadmin\(1M\)](#), [k5srvutil\(1M\)](#), [attributes\(5\)](#), [kerberos\(5\)](#)

名前	kvno – Kerberos 主体の鍵バージョン番号の出力
形式	kvno [-h] [-P] [-q] [-c <i>ccache</i>] [-e <i>etype</i>] [-S <i>sname</i>] [-U <i>for_user</i>] <i>princ_host</i> [<i>princ_host</i> ...]
機能説明	kvno は指定された Kerberos 主体のサービスチケットを取得し、それぞれの鍵バージョン番号を出力します。
オプション	サポートしているオプションは、次のとおりです。 -c <i>ccache</i> 使用する資格キャッシュの名前を指定します (デフォルト値でない場合)。 -e <i>etype</i> コマンド行で指定されたすべてのサービスのセッション鍵に対して要求される <i>etype</i> を指定します。 これは、特定の下位互換の状況で役立ちます。 -h 使用法の説明を出力し、終了します。 -P <i>princ_host</i> 引数を、制約された委譲を使用して資格が取得されるサービスとして扱うことを指定します。 このオプションは、プロトコル遷移と組み合わせて使用された場合にのみ有効です。 -q 出力を抑止します。 -S <i>sname</i> <i>krb5_sname_to_principal()</i> を主体名の構築に使用することを指定します。このフラグが指定されている場合は、<i>princ_host</i> 引数が (主体名ではなく) ホスト名として解釈され、<i>sname</i> がサービス名として解釈されます。 -U <i>for_user</i> <i>for_user</i> の代わりにチケットを取得するためにプロトコル遷移 (S4U2Self) を使用することを指定します。 制約された委譲が要求されない場合は、サービス名が資格キャッシュのクライアント主体に一致している必要があります。
オペランド	次のオペランドを指定できます。 <i>princ_host</i> -S オプションが指定されている場合は、鍵バージョン番号を取得するターゲットの主体の主体名またはホスト名のどちらかを指定します。
環境変数	kvno の実行に影響を与える環境変数 LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH については、environ(5) を参照してください。 KRB5CCNAME 資格 (チケット) のキャッシュの場所。

ファイル	/tmp/krb5cc_ <i>uid</i>	資格キャッシュのデフォルトの場所。 <i>uid</i> は、ユーザーの 10 進数の UID です。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/security/kerberos-5
インタフェースの安定性	確実

関連項目 [kdestroy\(1\)](#), [kinit\(1\)](#), [attributes\(5\)](#), [krb5envvar\(5\)](#)

名前	<code>lari</code> - 実行時インタフェースのリンク分析
形式	<pre>lari [-bCDsVv] [-a -i -o] file directory... lari [-CDosv] [-m [-d mapdir]] file</pre>
機能説明	<code>lari</code> ユーティリティーは、動的 ELF オブジェクトのインタフェース要件を分析します。2つの基本操作モードを利用できます。最初のモードでは、実行時インタフェース情報を表示します。2番目のモードでは、インタフェース定義を生成します。 動的オブジェクトは、外部コンシューマに提供するインタフェースを表すシンボル定義を提供します。実行時に、あるオブジェクトのシンボル参照から、別のオブジェクトのシンボル定義への結合が確立されます。<code>lari</code> は、指定されたオブジェクトのインタフェース定義と実行時結合の両方を分析します。 実行時インタフェース情報を表示するときに、<code>lari</code> は、ファイルまたはディレクトリの数を分析できます。<code>lari</code> は、コマンド行で指定したそれぞれの <i>file</i> を分析します。<code>lari</code> は、コマンド行で指定したそれぞれの <i>directory</i> の下位に再帰的に移動して、見つかったそれぞれのファイルを処理します。 インタフェース定義を生成するときに、<code>lari</code> は、コマンド行で指定した単一の <i>file</i> だけを処理できます。 <code>-D</code> オプションを指定しない場合、<code>lari</code> は、ldd(1) を使用することによって、動的 ELF オブジェクトとしてファイルを処理します。この処理では次のオプションを使用します。 <pre>-r and -e LD_DEBUG=files,bindings,detail</pre> これらのオプションは、オブジェクトを読み込む過程で確立されたすべての結合に関する情報を提示します。<code>ldd</code> を使用することによって、指定されたオブジェクトが実行されず、したがって、たとえば dlopen(3C) による、オブジェクトのユーザー制御の読み込みが行われないことに注意してください。実行プロセスからすべての結合情報を取得するために、次の環境変数を直接実行時リンカー ld.so.1(1) に渡すことができます。 <pre>LD_DEBUG=files,bindings,detail LD_DEBUG_OUTPUT=lari.dbg \ LD_BIND_NOW=yes</pre> 結果として得られるデバッグ出力 <code>lari.dbg.pid</code> は、<code>lari</code> で <code>-D</code> オプションを使用して処理できます。注: <code>lari</code> は、デバッグ出力で定義されたパス名を使用して処理された各オブジェクトを分析しようとします。したがって、完全な正確な分析を行うために、各オブジェクトには <code>lari</code> からアクセスする必要があります。デバッグ出力ファイルは、<code>C</code> ロケールで生成する必要があります。 <code>lari</code> は、インタフェース情報を表示するときに、処理された動的オブジェクトのインタフェースを分析し、デフォルトで、関心を引く情報を表示します。「拡張機能説明」の「関心を引く情報」を参照してください。表示される情報は、ほかの

ツールへのパイピングにも適しています。この機能は、開発者がプロセス結合を分析したり、複雑な結合シナリオをデバッグしたりするときに役立ちます。

`lari` を使ってインタフェース定義を生成すると、処理対象の動的オブジェクトのインタフェース要件をより正確に定義できるようになります。動的オブジェクトを作成するときに、明示的にバージョン定義されたインタフェースを定義するようにしてください。この定義は、外部ユーザーが利用できるシンボル定義を制御します。また、この定義によって、実行時のオブジェクトの実行コスト全体を削減できることがよくあります。インタフェース定義は、リンカーで `-M` オプションおよび関連付けられた `mapfile` 指令を使用することによって、作成時にオブジェクトに割り当てることができます。`mapfile` を使用してオブジェクトのバージョンを定義する方法の詳細は、『[リンカーとライブラリ](#)』を参照してください。これらの `mapfile` の初期バージョンは、`lari` で作成できます。

オプション

以下のオプションがサポートされています。

- a 分析対象のオブジェクトに関するすべてのインタフェース情報を表示します。注: このオプションからの出力は大量になることがあります。多くの場合、ほかの分析ツールへのパイピングに役立ちます。
- b インタフェース情報を、明示的に結合されているシンボルに制限します。注: `protected` と定義したシンボルは、定義用オブジェクト内から結合されている可能性があります。この結合はリンク編集時に満たされるので、実行時環境では表示されません。`protected` シンボルは、このオプションで表示されます。
- C `DemanglesC++` シンボル名を復号化します。このオプションは、実行時インタフェース情報を拡張するときに役立ちます。インタフェース定義の生成時に、`demangled` 名がコメントとして `mapfile` に追加されます。
- d `mapdir` `mapfile` が作成されるディレクトリ (`mapdir`) を定義します。デフォルトでは、現在の作業ディレクトリが使用されます。
- D 動的オブジェクトではなくデバッグ情報として入力ファイルを解釈します。
- i 関心を引くインタフェース結合情報を表示します。ほかの出力制御オプションが指定されていない場合、このモードがデフォルトです。「拡張機能説明」の「関心を引く情報」を参照してください。
- m 処理される動的オブジェクトごとに `mapfile` を作成します。これらの `mapfile` には、処理されている入力ファイルで必要になる各オブジェクトのインタフェース要件が反映されます。

- o インタフェース情報を、オーバーヘッドとみなされているシンボルに制限します。*mapfile* の作成時、オーバーヘッドシンボルがローカルシンボルとして項目化されます。「拡張機能説明」の「オーバーヘッド情報」を参照してください。
- s さらに分析するために、*ldd(1)* からの結合情報を保存します。「ファイル」を参照してください。
- V 関心を引くシンボル可視設定を追加します。*singleton* または *protected* と定義されたシンボルは、このオプションで識別されます。
- v すでにバージョン定義されているオブジェクトを無視します。バージョン定義されているオブジェクトのインタフェースはすでに定義されていますが、表示されるインタフェース情報にはその情報が含まれています。たとえば、バージョン定義されている共有オブジェクトであれば、特定のプロセスのオーバーヘッドシンボルが表示される場合があります。共有オブジェクトは多くの場合、複数のプロセスで使用できるように設計されているので、これらのオブジェクトで用意されているインタフェースは、単一のプロセスの要件を超える場合があります。したがって *-v* オプションを使用すると、インタフェース情報を表示するときにノイズを軽減できます。

lari から生成される実行時インタフェース情報は、次の形式になります。

[information]: *symbol-name* [demangled-name]: *object-name*

各行には、インタフェースシンボル (*symbol-name*) と、シンボルが定義されるオブジェクト (*object-name*) が記述されます。シンボルが関数を表す場合、シンボル名のあとに () が続きます。シンボルがデータオブジェクトを表す場合、シンボル名のあとに [] で囲まれたシンボルサイズが続きます。*-c* オプションが使用された場合、シンボル名は、シンボルの復号化された名前 (*demangled-name*) とともに表示されます。情報フィールドには、シンボルの使用について記述した次のトークンの 1 つまたは複数が表示されます。

- cnt:bnd* 2つの10進数値は、シンボル数 (*cnt*) と、このオブジェクトへの結合の数 (*bnd*) を示します。シンボル数は、分析対象のオブジェクトで見つかったこのシンボル定義の出現数です。1より大きな数は、シンボル定義が複数回出現したことを示しています。結合の数は、実行時リンカーによってこのシンボル定義に結合されたオブジェクトの数を示しています。
- E このシンボル定義は、外部オブジェクトから結合されています。
- S このシンボル定義は、同じオブジェクトから結合されています。
- D このシンボル定義は、直接結合されています。

- I** このシンボル定義は割り込みシンボルを規定します。明示的に割り込みオブジェクトであることを示すオブジェクトは、すべてのグローバルシンボルを割り込みシンボルとして定義します。[ld\(1\)](#) の `-z interpose` オプションおよび [ld.so.1\(1\)](#) の `LD_PRELOAD` 変数を参照してください。動的実行可能ファイル内の個々のシンボルは、`INTERPOSE mapfile` 指令を使用することによって、割り込みシンボルとして定義できます。
- C** このシンボル定義は、コピー再配置の参照データです。
- F** このシンボル定義は `filtee` にあります。
- P** このシンボルは `protected` と定義されています。このシンボルには、シンボルが宣言されたオブジェクトからの内部結合が存在する可能性があります。この属性を持つ内部結合には、別のシンボル定義が割り込むことはできません。
- A** このシンボル定義は、動的実行可能ファイル内でのプロシージャリンクエントリのアドレスです。
- U** このシンボル参照は [dlsym\(3C\)](#) などのユーザー要求に起因します。
- R** このシンボル定義はフィルタとして機能し、フィルタ対象へのリダイレクションを規定します。
- r** このシンボルへの結合は、シンボル検索中のある時点で拒否されました。直接結合要求が、直接結合を防止するタグが付けられているシンボルを見つけると、結合が拒否されます。このシナリオでは、シンボル検索は、デフォルトの検索モデルを使用して繰り返されます。結合はまだ、元の拒否されたシンボルに解決できます。デフォルト以外のシンボル検索が、`singleton` として識別されたシンボルを見つけたときにも、結合は拒否されます。この場合も、シンボル検索は、デフォルトの検索モデルを使用して繰り返されます。
- N** このシンボル定義は、定義に直接結合することを明示的に禁止します。
- これらのシンボル分類の詳細は、『[リンカーとライブラリ](#)』を参照してください。

拡張機能説明

関心を引く情報

デフォルトで、または `-i` オプションを明示的に使用すると、`lari` は、実行時インタフェース情報をフィルタ処理して、関心を引くイベントを示します。このフィルタ処理は主に、大きなアプリケーションから生成される情報量を減らすときに実行します。また、この情報を使用すると、複雑な結合シナリオをデバッグするときに注目すべき箇所がわかります。多くの場合で、問題領域を強調します。ただし、特定のアプリケーション用に意味のある情報を正確に分類することは困難です。`-a` オプションを使用すると、調査対象のアプリケーションに固有のイベントの結合情報を自由に検索できます。

関心を引くシンボル定義が検出された場合、同じシンボルのほかのすべての定義が出力されます。

関心を引くインタフェース情報がフォーカスしているのは、シンボルの複数の定義の存在です。この場合、通常、1つのシンボルがほかの1つまたは複数のシンボル定義に割り込みます。ある定義の結合数 (*bnd*) が0でなく、ほかのすべての定義の結合数が0である場合に、この割り込みが発生しています。コンパイル環境(つまりリンク環境)に起因する割り込みは、関心を引くものとは判断されません。これらの割り込みの出現例としては、コピー再配置 ([C]) や、プロシージャリネージアドレスへの結合 ([A]) などがあります。

多くの場合、割り込みは望ましいものです。その目的は、共有オブジェクトからのシンボル定義を多重定義する、つまり置き換えることです。`ld(1)` の `-z interpose` オプションを使用すると、割り込むオブジェクトに明示的にタグ ([I]) を付けることができます。これらのオブジェクトは、プロセスでオブジェクトを読み込む順番とは無関係に、安全にシンボルに割り込むことができます。ただし、明示的でない割り込みが使用される場合には注意してください。この割り込みは、プロセスを構成するオブジェクトの読み込み順序に起因するためです。

ユーザーが作成した多重定義シンボルは、関心を引く情報として `lari` から出力されます。この例では、`interpose1()` の2つの定義が存在しますが、`main` の定義だけが参照されます。

```
[2:1E]: interpose1(): ./main
[2:0]: interpose1(): ./libA.so
```

割り込みは、予期しないシンボル名の衝突によって生じた、想定外の好ましくないイベントである場合もあります。このような割り込みの症状としては、関数への参照が存在することがわかっているのに、その関数が呼び出されない場合などがあります。このシナリオは、前の例で説明したように、多重定義シンボルとして識別できます。しかし、オブジェクトが特定のシンボルを定義および参照する場合に、より意外なシナリオが多く発生します。

このシナリオの例は、2つの動的オブジェクトが同じ関数 `interpose2()` を定義し参照する場合です。このシンボルへの参照はすべて、プロセスで読み込まれる最初の動的オブジェクトに結合します。この場合、オブジェクト `libA.so` の `interpose2()` の定義は、オブジェクト `libB.so` の `interpose2()` の定義に割り込み、これを隠べします。`lari` からの出力は次のようになります。

```
[2:2E5]: interpose2(): ./libA.so
[2:0]: interpose2(): ./libB.so
```

多重定義シンボルに個別に結合する場合もあります。直接結合が有効になっている場合や ([D])、シンボルの可視性が `protected` の場合は ([P])、個別に結合されま

す。個別の結合は明示的に確立できますが、予期しない想定外の結合が存在する場合があります。直接結合されたシンボルと、可視性が `protected` のシンボルは、関心を引く情報として出力されます。

オーバーヘッド情報

`-o` オプションを使用した場合、`lari` は、オーバーヘッドとみなされるシンボル定義を表示します。

参照されないグローバルシンボルはオーバーヘッドとみなされます。オブジェクト内で提供されるシンボル情報によって、オブジェクトのテキストセグメントのサイズが必要以上に増加します。さらに、シンボル情報によって、実行時にオブジェクト内でほかのシンボル参照を検索するために必要な処理が増える場合があります。

同じオブジェクトからしか参照されないグローバルシンボルは、同じ特性を持っています。シンボル参照の実行時検索は、結果として、参照を行なった同じオブジェクトへの結合になり、追加のオーバーヘッドになります。

これらの両方のシンボル定義は、オブジェクトのインタフェースを定義することによってローカルスコープに降格できます。インタフェース定義は、リンカーで、`-m` オプションおよび関連付けられた `mapfile` 指令を使用することによって、作成時にファイルに割り当てることができます。`mapfile` の詳細は、『[リンカーとライブラリ](#)』を参照してください。`-m` オプションを付けて `lari` を使用して、これらの `mapfile` の初期バージョンを作成します。

`lari` を使用して `mapfile` を生成すると、バージョン定義された共有オブジェクトには、オーバーヘッドシンボルをローカルに降格すべきであることを示す `mapfile` が作成されます。このモデルでは、`lari` は、既存のインタフェース定義と比較するための `mapfile` を生成できます。`-v` オプションを使用して、`mapfile` を作成するとき、バージョン定義された共有オブジェクトを無視します。

コピー再配置はオーバーヘッドともみなされるため、通常は回避すべきです。コピーされたデータのサイズは、そのインタフェースの定義です。この定義は、データが定義されている共有オブジェクトのより新しいバージョンでのデータサイズを変更する機能を制限します。この制限、およびコピー再配置を処理するコストは、関数インタフェースを使用してデータを参照することによって回避できます。コピー再配置の `lari` からの出力は次のようになります。

```
[2:1EC]: __iob[0x140]: ./main
[2:0]: __iob[0x140]: ./libA.so.1
```

前の例で使用された `__iob` など、いくつかの小さなコピー再配置が存在することに注意してください。これは、システムライブラリのプログラミング操作のために歴史的な理由により存在しています。

オーバーヘッド情報の別の例は、動的実行可能ファイルのプロシージャリンケージテーブルエントリに、動的オブジェクトを結合する場合です。動的実行可能ファイルが外部関数を参照する場合、プロシージャリンケージテーブルエントリが作成さ

れます。この構造では、関数呼び出しが実際に行われるまで、参照結合を延期できます。動的オブジェクトがこの参照された同じ関数のアドレスを取得した場合、動的オブジェクトは、動的実行可能ファイルのプロシージャリリンクエントリに結合します。このタイプのイベントの例は、次の点を明らかにします。

```
[2:1EA]: foo(): ./main
[2:1E]: foo(): ./libA.so
```

このタイプの少数の結合は、通常は心配する必要はありません。ただし、ジャンプテーブルプログラミング技法などによって、これらの結合が多数発生する場合は、オーバーヘッドを起こす一因となることが考えられます。このタイプのアドレス再配置結合では、直接関数を呼び出したときに使用される遅延再配置処理ではなく、アプリケーション起動時の再配置処理が必要になります。このアドレスを使用するために、実行時に間接参照も必要になります。

使用例

例1 複数の結合がある場合の分析

次の例では、複数のシンボル定義が存在するプロセスの分析を示します。共有オブジェクト `libX.so` と `libY.so` は関数 `interpose()` を呼び出します。この関数は、アプリケーション `main` と、共有オブジェクト `libA.so` の両方に存在します。割り込みのため、両方の参照は `main` の `interpose()` の定義に結合します。

共有オブジェクト `libX.so` と `libY.so` は関数 `foo()` も呼び出します。この関数は、アプリケーション `main` と、共有オブジェクト `libA.so`、`libX.so`、および `libY.so` に存在します。`libX.so` と `libY.so` の両方は直接結合を有効にして構築されているので、各オブジェクトはそれぞれ独自の定義に結合します。

```
example% lari ./main
[3:0]: foo(): ./libA.so
[3:1SD]: foo(): ./libX.so
[3:1SD]: foo(): ./libY.so
[2:0]: interpose(): ./libA.so
[2:2EP]: interpose(): ./main
```

結合情報をさらに分析するために、結合データをその後の検査用に保存できます。たとえば、前の出力は、関数 `interpose()` が、`main` の外部にある2つのオブジェクトから呼び出されたことを示しています。結合の出力を検査することによって、この関数への参照が行われた場所を明らかにできます。

```
example% lari -s ./main
lari: ./main: bindings information saved as: /usr/tmp/lari.dbg.main
.....
example% fgrep foo /usr/tmp/lari.dbg.main
binding file=./libX.so to file=./main: symbol 'interpose'
binding file=./libY.so to file=./main: symbol 'interpose'
```

注: 結合の出力にはプロセス識別子、アドレス、その他の結合情報が伴うので、出力は通常、ここに示すよりも多くなります。

例2 インタフェース定義の生成

次の例では、アプリケーションとその依存関係のインタフェース定義を作成しますが、バージョンが定義されたシステムライブラリは無視します。アプリケーション main は、foo.so のインタフェース one()、two()、および three() を参照します。

```
example% lari -omv ./main
example% cat mapfile-foo.so
#
# Interface Definition mapfile for:
#     Dynamic Object: ./foo.so
#     Process:        ./main
#
foo.so {
    global:
        one;
        three;
        two;
    local:
        _one;
        _three;
        _two;
        *;
};
```

ファイル `$TMPDIR/lari.dbg.file` [ldd\(1\)](#) によって生成された結合の出力。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities
インタフェースの安定性	下記を参照。

人間が読める形式の出力は「不確実」です。オプションは「確実」です。

関連項目 [ld\(1\)](#), [ldd\(1\)](#), [ld.so.1\(1\)](#), [dlopen\(3C\)](#), [dlsym\(3C\)](#), [attributes\(5\)](#)
『リンカーとライブラリ』

名前	last – display login and logout information about users and terminals
形式	last [-a] [-n <i>number</i> -number] [-f <i>filename</i>] [<i>name</i> <i>tty</i>]...
機能説明	The <code>last</code> command looks in the <code>/var/adm/wtmpx</code> file, which records all logins and logouts, for information about a user, a terminal, or any group of users and terminals. Arguments specify names of users or terminals of interest. If multiple arguments are given, the information applicable to any of the arguments is printed. For example, <code>last root console</code> lists all of root's sessions, as well as all sessions on the console terminal. <code>last</code> displays the sessions of the specified users and terminals, most recent first, indicating the times at which the session began, the duration of the session, and the terminal on which the session took place. <code>last</code> also indicates whether the session is continuing or was cut short by a reboot. The pseudo-user <code>reboot</code> logs in when the system is shutdown and when it reboots. Thus, <pre>last reboot</pre> gives an approximate record of when the operating system instance was shutdown and when it rebooted. This can be used to calculate the availability of the operating system over time. <code>last</code> with no arguments displays a record of all logins and logouts, in reverse order. If <code>last</code> is interrupted, it indicates how far the search has progressed in <code>/var/adm/wtmpx</code>. If interrupted with a quit signal (generated by a CTRL-<code>\</code>), <code>last</code> indicates how far the search has progressed, and then continues the search.
オプション	The following options are supported: -a Displays the hostname in the last column. -f <i>filename</i> Uses <i>filename</i> as the name of the accounting file instead of <code>/var/adm/wtmpx</code>. -n <i>number</i> -<i>number</i> Limits the number of entries displayed to that specified by <i>number</i>. These options are identical; the -<i>number</i> option is provided as a transition tool only and is removed in future releases.
環境変数	Date and time format is based on locale specified by the <code>LC_ALL</code> , <code>LC_TIME</code> , or <code>LANG</code> environments, in that order of priority.
ファイル	<code>/var/adm/wtmpx</code> accounting file
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目

```
utmpx(4), attributes(5)
```

名前	lastcomm – display the last commands executed, in reverse order
形式	lastcomm [-f <i>file</i>] [-x] [<i>command-name</i>] ... [<i>user-name</i>] ... [<i>terminal-name</i>] ...
機能説明	The lastcomm command gives information on previously executed commands. lastcomm with no arguments displays information about all the commands recorded during the current accounting file's lifetime. If called with arguments, lastcomm only displays accounting entries with a matching <i>command-name</i>, <i>user-name</i>, or <i>terminal-name</i>. If extended process accounting is active (see acctadm(1M)) and is recording the appropriate data items, lastcomm attempts to take data from the current extended process accounting file. If standard process accounting is active, lastcomm takes data from the current standard accounting file (see acct(2)). If <i>terminal-name</i> is '-', there was no controlling TTY for the process. The process was probably executed during boot time. If <i>terminal-name</i> is '??', the controlling TTY could not be decoded into a printable name. For each process entry, lastcomm displays the following items of information: ■ The command name under which the process was called. ■ One or more flags indicating special information about the process. The flags have the following meanings: - F The process performed a fork but not an exec. - S The process ran as a set-user-id program. ■ The name of the user who ran the process. ■ The terminal which the user was logged in on at the time (if applicable). ■ The amount of CPU time used by the process (in seconds). ■ The date and time the process exited.
オプション	The following options are supported: -f <i>file</i> Uses <i>file</i> as the source of accounting data. <i>file</i> may be either an extended process accounting file or a standard process accounting file. -x Uses the currently active extended process accounting file. If extended processing accounting is inactive, no output will be produced.
使用例	例 1 Listing executions of named commands The command <pre>example% lastcomm a.out root term/01</pre> produces a listing of all the executions of commands named a.out by user root while using the terminal term/01.

例 2 Listing all user commands

The command

example% **lastcomm root**

produces a listing of all the commands executed by user root.

ファイル

/var/adm/pacct standard accounting file

/var/adm/exacct/proc extended accounting file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目

[last\(1\)](#), [acctadm\(1M\)](#), [acct\(2\)](#), [acct.h\(3HEAD\)](#), [core\(4\)](#), [attributes\(5\)](#)

名前
形式

ld - オブジェクトファイル用リンカー

```
ld [-32 | -64] [-Bdirect | nodirect] [-B dynamic | static]
[-B eliminate[=mode] | local[=mode]] [-B reduce] [-c name]
[-C] [-D [!]token1,[!]token2,...] [-e epsym]
[-f name | -F name] [-G] [-h name] [-i] [-l x] [-L path]
[-m] [-M mapfile] [-N string] [-o outfile] [-p auditlib]
[-P auditlib] [-Q y | n] [-r] [-R path] [-s]
[-S supportlib] [-t] [-u symname] [-V] [-Y P,dirlist]
[-z allexttract | defaultextract | weakextract ]
[-z altexec64] [-z aslr[=mode]] [-z ancillary[=outfile]]
[-z assert-deflib[=libname]] [-z deferred | nodeferred]
[-z defs | nodefs] [-z direct | nodirect]
[-z discard-unused=item1,item2,...] [-z endfiltee]
[-z fatal-warnings | nofatal-warnings] [-z finiarray=function]
[-z globalaudit] [-z guidance[=item1,item2,...]] [-z help ]
[-z ignore | record] [-z initarray=function] [-z initfirst]
[-z interpose] [-z lazyload | nolazyload]
[-z ld32=arg1,arg2,...] [-z ld64=arg1,arg2,...]
[-z loadfltr] [-z mapfile-add=name] [-z muldefs]
[-z nocompstrtab] [-z nodefaultlib] [-z nodelete]
[-z nodlopen] [-z nodump] [-z noldynsym] [-z nopartial]
[-z origin] [-z parent=object] [-z preinitarray=function]
[-z relaxreloc] [-z rescan-now] [-z rescan-start ... -z rescan-end]
[-z strip-class=[!]class1,[!]class2,...] [-z stub]
[-z symbolcap] [-z target=sparc|x86]
[-z text | textwarn | textoff] [-z verbose] [-z wrap=symbol]
filename ...
```

機能説明

リンカー ld は、シンボル定義へのシンボル参照を解決し、再配置を実行することによって再配置可能オブジェクトファイルを結合します。いずれの場合も、ld のデフォルトの出力は a.out ファイルとなります。「注意事項」を参照してください。

ld には、多数のオプションがあります。現代的なプログラミング習慣に関連するオプションは「形式」で定義され、以降のセクションで説明されています。その他のオプションは一般にはあまり使用されず、「セカンダリオプション」セクションで説明されています。

ld は、動的または静的の2つのモードのどちらかで動作します。動的モードがデフォルトですが、このモードは -r または -d オプションで管理できます。動的モードでは、引数として提供される再配置可能オブジェクトファイルが結合されて、動的実行可能ファイルが生成されます。このファイルは、実行時に、引数として指定した共有オブジェクトファイルに結合されます。-G オプションを指定した場合、再配置可能オブジェクトファイルが結合され、共有オブジェクトファイルが生成されます。この動的リンク環境では、リンカーと実行時リンカー ld.so.1(1) の作業が密接に結び付けられます。これらのユーティリティーはどちらも、関連テクノロジーおよびユーティリティーと合わせて『リンカーとライブラリ』で広範囲にわたって説明されています。

静的モードでは、`-r` オプションにより、引数として提供される再配置可能オブジェクトファイルが結合されて1つの再配置可能オブジェクトファイルが生成されます。Oracle Solaris オペレーティングシステムの最新のバージョンは、静的実行可能ファイルの作成をサポートしていません。Static Executablesを参照してください。このセクションのすべての説明は動的リンク、または `-r` オプションを指定した静的リンクを前提にしています。

いずれかの引数がライブラリである場合、`ld` はデフォルトでは、コマンド行でそのライブラリが検出された時点でそのライブラリを1回だけ検索します。ライブラリとして指定できるのは、共有オブジェクトまたは再配置可能アーカイブのどちらかです。[ar.h\(3HEAD\)](#) を参照してください。

共有オブジェクトは不可分の完全なユニットで、1つ以上の入力ファイルをリンクすることによって生成されます。リンカーが共有オブジェクトを処理すると、共有オブジェクトの全内容は、その結果作成された出力ファイルイメージの論理的な部分になります。共有オブジェクトが実際に取り込まれるのはプロセスの実行時であるため、リンク処理中に物理的なコピー処理が行われることはありません。この論理的な組み込みは、リンク編集プロセスにとって共有オブジェクト内に定義されたすべてのシンボルエントリが利用可能になることを意味しています。[『リンカーとライブラリ』の第4章「共有オブジェクト」](#) を参照してください。

アーカイブライブラリの場合、`ld` は、未解決の外部参照を定義するアーカイブメンバーのみをロードします。`ld` は、アーカイブメンバーが満足できる外部参照を解決するために、アーカイブライブラリのシンボルテーブルを順番に検索します。この検索は、アーカイブで外部参照を解決できなくなるまで繰り返されます。そのため、同じ外部シンボルを定義する複数のアーカイブメンバーが存在しないかぎり、ライブラリ内のメンバーの順序は機能的に重要ではありません。相互依存関係を持つアーカイブライブラリでは、複数のコマンド行定義、またはいずれかの `-z rescanner` オプションの使用が必要になる場合があります。[『リンカーとライブラリ』の「アーカイブ処理」](#) を参照してください。

`ld` はクロスリンカーで、SPARC または x86 ターゲットに対して 32 ビットオブジェクトまたは 64 ビットオブジェクトをリンクできます。`ld` の操作モードは、コマンド行の最初の再配置可能オブジェクトの ELF クラスおよび機械タイプによって制御されます。32 ビットオブジェクトと 64 ビットオブジェクトを混在させることはできません。同様に、1つの機械タイプのオブジェクトのみが許可されます。`-32`、`-64` および `-z target` オプションと `LD_NOEXEC_64` 環境変数を参照してください。

静的実行可能 ファイル

静的実行可能ファイルは、多くのリリースで作成しないように勧められています。実際、Solaris では、64 ビットのシステムアーカイブライブラリが提供されたことはありません。静的実行可能ファイルは、システムアーカイブライブラリに反して構築されるので、実行可能ファイルにはシステム実装の詳細が含まれます。この自己内包には、多数の欠点があります。

- この実行可能ファイルは、共有オブジェクトとして提供されるシステムパッチの恩恵を受けることができません。したがって、多くのシステムの改良を利用するには、この実行可能ファイルを再構築する必要があります。
- 将来のリリースでこの実行可能ファイルを実行できなくなる可能性があります。
- システム実装の詳細を複製すると、システムのパフォーマンスに悪影響を与えます。

Oracle Solaris 10 から、32 ビットのシステムアーカイブライブラリは提供されなくなりました。これらのライブラリ (特に、`libc.a`) がなくなるため、特別なシステム知識がなければ、静的実行可能ファイルは作成できません。ただし、静的リンクオプションを処理する `ld` の機能とアーカイブライブラリの処理に変更はありません。

オプション

以下のオプションがサポートされています。

-32 | **-64**

32 ビットまたは 64 ビットオブジェクトを作成します。

デフォルトでの生成されるオブジェクトのクラスは、コマンド行から処理される最初の ELF オブジェクトによって決まります。オブジェクトを指定してしない場合、クラスは、コマンド行から処理される最初のアーカイブ内で検出された最初のオブジェクトによって決まります。オブジェクトもアーカイブもない場合、リンカーは 32 ビットオブジェクトを作成します。

-64 オプションは、64 ビットオブジェクトを `mapfile` からのみ作成する場合に必要です。

-32 または **-64** オプションは、32 ビットと 64 ビットのオブジェクトが混在するアーカイブからすべてをリンクするというごくまれな場合にも使用できます。アーカイブの最初のオブジェクトが、作成される必要のあるオブジェクトのクラスではない場合、**-32** または **-64** オプションを使用してリンカーに指示できます。

-Bdirect | **nodirect**

これらのオプションで直接結合を制御します。**-Bdirect** は、シンボル参照と、シンボル定義を提供する依存関係との間の関係を記録することによって、直接結合情報を確立します。さらに、直接結合情報は、シンボル参照と、作成されるオブジェクト内の関連付けられた定義の間で確立できます。実行時リンカーは、デフォルトのシンボル検索を行わずに、この情報を使用して、関連付けられたオブジェクト中のシンボルを直接検索します。

直接結合情報は、リンク編集で指定される依存関係に対してのみ確立されます。そのため、**-z defs** オプションも追加するようにしてください。直接結合環境でシンボルに対して割り込もうとするオブジェクトは、**-z interpose** オプションにより、自身をインターポーターとして識別するべきです。**-Bdirect** を使用すると、**-z lazyload** もすべての依存関係に対して有効になります。

-Bnodirect オプションは、作成されているオブジェクトによって提供されるインタフェースとの直接結合をすべて回避させます。-zdirect オプションを指定すると、作成されているオブジェクトは外部インタフェースと直接結合を続けられます。『リンカーとライブラリ』の第9章「直接結合」を参照してください。

-Bdynamic | static

これらのオプションによって、ライブラリの取り込みが管理されます。-Bdynamic は、-l オプションのライブラリ検索が共有オブジェクトとアーカイブライブラリ名に展開されるように許可します。このオプションはデフォルトであり、動的モードでのみ有効です。-Bstatic は、-l オプションのライブラリ検索がアーカイブライブラリ名にのみ展開されるように制限します。これらのオプションは、トグルとしてコマンド行で何回でも指定できます。-Bstatic オプションが指定されている場合は、-Bdynamic が見つかるまで、共有オブジェクトは受け入れられません。-l オプションを参照してください。

-Beliminate[=*mode*] | local[=*mode*]

デフォルトではグローバル結合になるシンボルがすべて、シンボルテーブルから削除されるか、またはローカルな可視性に削減されます。STV_DEFAULT 可視性を持つグローバルシンボルは、リンカーによって、より制限的な可視性に削減される場合があります。その他のすべての STV_ 可視性を持つグローバルシンボルは、明示的な可視性を持っていると見なされます。明示的な可視性はリンカーによって尊重され、変更することはできません。『リンカーとライブラリ』の「SYMBOL_SCOPE/SYMBOL_VERSION 指令」を参照してください。シンボルの可視性は、コンパイラ指令を使用して、あるいはバージョンまたはインタフェース定義を定義する mapfiles を使用して明示的に定義できます。

Mapfile バージョンおよびインタフェース定義には、*auto-elimination* または *auto-reduction* 指令を含めることができます。『リンカーとライブラリ』の「SYMBOL_SCOPE/SYMBOL_VERSION 指令」を参照してください。これらの指令を指定すると、mapfile で明示的に定義されていないシンボル、または明示的な可視性を定義していないシンボルがそれぞれ、削除されるか、またはローカルに削減されます。以降の段落で説明されているように、シンボルの削除または削減は、すべての明示的なシンボル定義に加えて、生成されるオブジェクトのタイプによっても影響を受けます。-Beliminate オプションは、mapfile *auto-elimination* 指令と同じシンボル削除を要求します。-Blocal オプションは、mapfile *auto-reduction* 指令と同じシンボル削減を要求します。

どちらのオプションも、削除または削減されるグローバルシンボルの選択を制御するために、external または noexternal のモードで修飾できます。リンカーはデフォルトで、生成されるオブジェクトのタイプに適したモード (実行可能ファイルの場合は external、共有オブジェクトの場合は noexternal) になるため、この微調整は通常は必要ありません。

動的オブジェクトを構築するときは、一部のシンボルを確実にグローバルのままにして、外部の依存関係から参照可能にすることが必要になる場合があります。これは特に、動的実行可能ファイルの場合に当てはまります。実行可能

ファイルを構築するときは、実行時のプロセス初期化を可能にするコンパイル環境によって再配置可能オブジェクトが提供されます。これらの再配置可能オブジェクトには、システムの依存関係から参照されるグローバルシンボルを含めることができます。これらのシンボルは、実行時の実行を損なわないように、シンボルの自動削除または自動削減手法には関係なくグローバルのままになるようにするべきです。

`mode` を `external` として定義すると、外部の依存関係からのいずれかのシンボル参照が、構築されるオブジェクト内のシンボル定義に結合される可能性があるかどうかを判定するために、外部のすべての依存関係が分析されます。このような結合を満たすグローバルシンボルが削除されたり、ローカルに削減されたりすることはありません。動的実行可能ファイルを生成する場合は、このモードがデフォルトです。

`mode` を `noexternal` として定義すると、外部のすべての依存関係の分析が回避され、`mapfile` で明示的に定義されていない、または明示的な可視性を定義していないすべてのシンボルが削減されます。共有オブジェクトを生成する場合は、このモードがデフォルトです。

-B reduce オプションも参照してください。

-B reduce

再配置可能オブジェクトを生成している場合は、いずれかのシンボル可視性属性によって、あるいはバージョンまたはインタフェース定義を定義する `mapfiles` を使用して定義されたシンボル情報が削減されます。デフォルトでは、再配置可能オブジェクトが生成される場合、可視性属性、バージョン定義、またはインタフェース定義は出力イメージにのみ記録されます。動的実行可能ファイルまたは共有オブジェクトを作成する場合は、可視性属性、あるいは `mapfile` バージョンまたはインタフェース定義は常に、すべてのシンボル情報に適用されます。

-c name

実行時に使用する構成ファイル *name* を記録します。構成ファイルは、デフォルト検索パスの変更やディレクトリキャッシュの提供に使用されるほか、代替オブジェクトの依存関係を提供します。[crle\(1\)](#) を参照してください。

-C

診断メッセージに表示される C++ シンボル名を復号化します。

-D [!]]token1,[!]token2,...

token に指定したデバッグ情報を標準エラーに出力します。`help` は、使用可能なトークンをすべて指定した場合と同じ意味を持つ特殊なトークンです。[『リンクとライブラリ』の「デバッグ支援」](#)を参照してください。

-e epsym

--entry epsym

出力ファイルのエントリポイントのアドレスとして、シンボル *epsym* に設定します。

-f *name*

--auxiliary *name*

共有オブジェクトを構築するときのみ使用されます。このオプションを指定すると、共有オブジェクトのシンボルテーブルが、*name*に指定した共有オブジェクトのシンボルテーブルで、補助フィルタとして使用されるようになります。このオプションは複数回指定できますが、本オプションは -F と同時に指定することはできません。『リンカーとライブラリ』の「補助フィルタの生成」を参照してください。

-F *name*

--filter *name*

共有オブジェクトを構築するときのみ使用されます。このオプションを指定すると、共有オブジェクトのシンボルテーブルが、*name*に指定した共有オブジェクトのシンボルテーブルで、フィルタとして使用されるようになります。このオプションは複数回指定できますが、本オプションは -f と同時に指定することはできません。『リンカーとライブラリ』の「標準フィルタの生成」を参照してください。

-G

-shared

動的モード専用です。共有オブジェクトを生成します。未定義のシンボルも許容されます。『リンカーとライブラリ』の第4章「共有オブジェクト」を参照してください。

-h *name*

--soname *name*

動的モード専用です。共有オブジェクトの作成時、*name* をオブジェクトの動的セクションに記録します。ファイルシステムにおけるオブジェクトの名前ではなく、*name* がこのオブジェクトにリンクされた任意の動的オブジェクトに記録されます。このため、実行時リンカーは、実行時に検索する共有オブジェクト名として *name* を使用します。『リンカーとライブラリ』の「共有オブジェクト名の記録」を参照してください。

-i

LD_LIBRARY_PATH の設定を無視します。LD_LIBRARY_PATH の設定がリンク処理の妨げになる場合は、このオプションを使用して、実行時のライブラリ検索への影響を防ぎます。

-l *x*

--library *x*

ライブラリ *libx.so* (共有オブジェクトの慣例的な名前) または *libx.a* (アーカイブライブラリの慣例的な名前) を検索します。動的モードでは、-Bstatic オプションが有効になっている場合を除けば、ライブラリ検索パスに指定された各ディレクトリ内で、*libx.so* ファイルまたは *libx.a* ファイルが検索されます。ディレクトリ検索は、どちらかのファイルが入っているディレクトリを検出した時点で終了します。-l*x* が、*libx.so*、*libx.a* という形式の名前を持つ2つのファイルに展開される場合、*.so* ファイルが選択され、*libx.so* が見つからない

場合は `libx.a` が選択されます。静的モードを使用している場合や `-Bstatic` オプションが有効になっている場合は、`.a` ファイルだけが選択されます。`ld` は、ライブラリの名前を検出した時点でライブラリの検索を実行するので、`-l` の位置は重要な意味を持ちます。『リンカーとライブラリ』の「追加ライブラリとのリンク」を参照してください。

`-L path`

`--library-path path`

`path` をライブラリ検索ディレクトリに追加します。`ld` のライブラリ検索は、まず `-L` オプションで指定したディレクトリで行われ、次に標準ディレクトリで行われます。このオプションは、同一コマンド行の `-l` オプションより前に指定した場合のみ有効です。『リンカーとライブラリ』の「リンカーが検索するディレクトリ」を参照してください。

環境変数 `LD_LIBRARY_PATH` はライブラリ検索パスの補完に使用できますが、実行時環境によっても解釈されるため、`-L` オプションを使用することをお勧めします。「環境変数」の「`LD_LIBRARY_PATH`」を参照してください。

`-m`

メモリーマップ(入出力セクションのリスト)と、致命的ではない多重定義シンボルを生成し、標準出力に出力します。

`-M mapfile`

マップファイル `mapfile` を、`ld` への指令が記述されているテキストファイルとして読み取ります。このオプションは複数回指定できます。`mapfile` がディレクトリの場合、`stat(2)` で定義されるように、そのディレクトリ内のすべての通常ファイルが処理対象になります。『リンカーとライブラリ』の付録 B「System V Release 4 (バージョン 1) Mapfile」を参照してください。マップファイルの例は、`/usr/lib/ld` にあります。「ファイル」を参照してください。

`-N string`

このオプションを指定すると、作成されるオブジェクトの `.dynamic` セクションに `DT_NEEDED` エントリが追加されます。`DT_NEEDED` 文字列の値はコマンド行で指定した `string` です。このオプションは位置に依存します。このため、`DT_NEEDED .dynamic` エントリはリンク行にあるほかの動的な依存オブジェクトに対して相対的になります。デバイスドライバの再配置可能オブジェクト間の依存関係を指定するときに、このオプションをオプション `-dy` および `-r` と組み合わせて使用すると便利です。

`-o outfile`

`--output outfile`

`outfile` という名前の出力オブジェクトファイルを生成します。デフォルトのオブジェクトファイル名は、`a.out` です。

`-p auditlib`

監査ライブラリ `auditlib` を識別します。このライブラリは、作成されたオブジェクトの実行時に行われる監査に使用されます。自分自身の監査を行うような共有オブジェクトと依存関係にあるオブジェクトは、この共有オブジェクトの性

質を継承し、自分自身の監査を行います (-P オプションを参照)。[『リンカーとライブラリ』の「実行時リンカーの監査インタフェース」](#)を参照してください。

-P *auditlib*

監査ライブラリ *auditlib* を識別します。このライブラリは、作成されたオブジェクトの実行時に行われる依存関係の監査に使用されます。この監査は、このオブジェクトと依存関係にあり、監査を行う必要があるオブジェクトから継承される場合もあります -p オプションと -z *globalaudit* オプションを参照してください。[『リンカーとライブラリ』の「実行時リンカーの監査インタフェース」](#)を参照してください。

-Q *y|n*

-Q *y* を指定すると、*ident* 文字列が、出力ファイルの *.comment* セクションに追加されます。この文字列は、ファイル作成に使用された *ld* のバージョンを識別します。このため、*ld -r* を使用した場合など、複数のリンク手順を経て作成されたファイルには、複数の *ld ident*s があることになります。この識別は、*cc* コマンドのデフォルト時の動作と同じです。-Q *n* は、バージョンの識別を抑止します。*.comment* セクションは *mcs(1)* ユーティリティーで操作できます。

-r

--relocatable

複数の再配置可能オブジェクトファイルを結合して、1つの再配置可能オブジェクトを生成します。*ld* は、未解決の参照があってもメッセージを出力しません。このオプションは、-a と同時には使用できません。

-R *path*

-rpath *path*

複数のディレクトリをコロンで区切って指定します。このリストは、実行時リンカーにライブラリ検索ディレクトリを指定する際に使用されます。NULL 以外の文字列は、出力オブジェクトファイルに記録され、実行時リンカーに渡されます。このオプションを複数回指定する場合と、コロンで区切って *path* を指定する場合の結果は同じになります。[『リンカーとライブラリ』の「実行時リンカーが検索するディレクトリ」](#)を参照してください。

関連オブジェクト内で *runpath* を使用する方が、*LD_LIBRARY_PATH* 環境変数を使用するなどの方法でグローバル検索パスを設定するよりも適切です。オブジェクトの依存関係を探すために必要な *runpaths* のみ記録するようにしてください。*ldd(1)* を -U オプションとともに使用すると、動的オブジェクト内の未使用の *runpaths* を探すこともできます。

また、*runpath* で提供されるさまざまなトークンを使用すると、システムの機能やオブジェクトの位置を柔軟に識別できます。[『リンカーとライブラリ』の第6章「動的ストリングトークンによる依存関係の確立」](#)を参照してください。*\$ORIGIN* トークンは、動的オブジェクトをファイルシステム内の別の場所に再配置できるようにする場合に特に役立ちます。

-s

--strip-all

出力ファイルからシンボル情報をすべて取り除きます。このオプションは、`-z strip-class` オプションを `debug` と `symbol` クラス識別子とともに使用することと同等です。`-z relocsym` および `-z noldynsym` オプションも参照してください。

`-S supportlib`

`ld` によって共有オブジェクト `supportlib` が読み込まれ、リンクプロセスに関する情報が提供されます。`-s` オプションを使用して定義された共有オブジェクトは、`SGS_SUPPORT` 環境変数を使用して提供することもできます。『[リンカーとライブラリ](#)』の「[リンカーのサポートインタフェース](#)」を参照してください。

`-t`

サイズの異なる多重定義シンボルを検出した場合の警告の出力を抑止します。

`-u symname`

`--undefined symname`

`symname` を、未定義シンボルとしてシンボルテーブルに入力します。このオプションは、すべてのルーチンをアーカイブライブラリから読み込む場合に便利です。これは、最初のルーチンを読み込むために、未解決の参照が必要になるからです。コマンド行内でのこのオプションの位置は重要な意味を持ち、シンボルを定義するライブラリより前に配置する必要があります。『[リンカーとライブラリ](#)』の「[-u オプションを使用した追加シンボルの定義](#)」を参照してください。

`-V`

`--version`

使用する `ld` のバージョン情報を示すメッセージを出力します。

`-YP,dirlist`

ライブラリ検索に使用するデフォルトのディレクトリを変更します。`dirlist` には、複数のパスをコロンで区切って指定します。

`-z allextract|defaultextract|weakextract`

`--whole-archive|--no-whole-archive`

後続のすべてのアーカイブのオブジェクトの抽出条件を変更します。デフォルトでは、アーカイブメンバーは、未定義の参照を解決し、データ定義による仮の定義を行うために抽出されます。弱いシンボル参照は抽出を行いません。`-z allextract` または `--whole-archive` オプションでは、すべてのアーカイブメンバーがアーカイブから抽出されます。`-z weakextract` を指定すると、弱い参照によってアーカイブの抽出が行われます。`-z defaultextract` または `--no-whole-archive` オプションを指定すると、すでに指定されている抽出オプションを無効にし、デフォルトの状態に戻すことができます。『[リンカーとライブラリ](#)』の「[アーカイブ処理](#)」を参照してください。

`-z altexec64`

64 ビットの `ld` を実行します。これまでは、実行されたリンカーのクラスは、作成された ELF オブジェクトのクラスによって決まっていました。現在では、実行

されるリンカーのクラスは、ベースとなるシステムのクラスによって決まります。通常は64ビットです。このオプションは、下位互換性を確保するために残されています。

-z ancillary[=*outfile*]

通常は出力オブジェクトに追加される割り当て不可能なすべてのセクションを受信する補助的な出力ファイルを指定します。割り当て不可能なセクションは実行時には必要なく、主にデバッガやその他の可観測性ツールで使用されます。*outfile*が存在する場合は、指定された名前を使用して補助的なファイルが作成されます。*outfile*が存在しない場合、補助的なファイルにはプライマリ出力ファイルと同じ名前が与えられ、*.anc* 接尾辞が追加されます。『リンカーとライブラリ』の第2章「リンカー」を参照してください。

補助的なオブジェクトは、実行可能オブジェクトと共有オブジェクトでのみサポートされます。再配置可能オブジェクトを構築するために *-r* オプションが指定されている場合は、*-z ancillary* オプションを使用できません。*-z ancillary* オプションは、*-z stub* オプションと組み合わせて使用されている場合、何も出力せずに無視されます。また、*outfile* なしで指定され、*-o* オプションで指定された出力ファイルが */dev/null* などのデバイス特殊ファイルである場合の *-z ancillary* オプションも無視されます。

-z aslr[=*mode*]

実行可能ファイルのアドレス空間配置のランダム化 (ASLR) 動作を指定します。モード値は、*enabled* または *disabled* に設定できます。モードを省略した場合、ASLRは有効になります。

-z assert-deflib[=*libname*]

-l コマンド行オプションで指定したライブラリに対して、リンカーで指定したデフォルト検索パスの検査によって検出される警告メッセージを有効にします。*libname* の値を指定するとデフォルトのライブラリ警告機能が有効になり、指定したライブラリが、警告を出力しないライブラリの一覧に追加されます。複数の *-z assert-deflib* オプションを指定すると、警告を出力してはいけないライブラリを複数指定できます。

libname の値は、リンカーで検出されたときのように、バスコンポーネントのないライブラリファイル名にする必要があります。たとえば、次のように指定すると、デフォルトのライブラリ警告が有効になり、標準Cライブラリが除外されます。

```
ld ... -z assert-deflib=libc.so ...
```

-z assert-deflib は、同名のオブジェクトが複数存在し、使用ライブラリを厳密に管理する必要がある構築環境を主に対象とした特殊なオプションです。このオプションは、一般的な用途向けではありません。

-z deferred|nodeferred

動的依存関係への遅延の印の付加を有効または無効にします。遅延の印を付けられた動的依存関係にはレイジー読み込み可能な印も付けられ、初期プロセスの起

動では読み込まれません。遅延依存関係の読み込みは、そのレイジー参照への結合がはじめて行われたときに行われます。基本的なレイジー読み込み可能な依存関係とは異なり、遅延依存関係は `LD_BIND_NOW` プロセスの一部として、または `RTLD_NOW` フラグが設定された `dlopen(3C)` を介して処理されることはありません。『リンカーとライブラリ』の「動的依存関係の遅延読み込み」を参照してください。

遅延依存関係を `dlsym(3C)` および `RTLD_PROBE` ハンドルとともに使用すると、自然なコーディングスタイルを使用して柔軟な方法で機能テストを実行できます。

`-z defs | nodefs`

`--no-undefined`

`-z defs` オプションと `--no-undefined` オプションは、リンク終了時に未定義のシンボルがあると、致命的エラーを強制的に出力します。これは、実行可能オブジェクトを作成する場合のデフォルトの設定ですが、歴史的経緯から、共有オブジェクトを作成場合にはデフォルトではありません。共有オブジェクト作成時に、そのオブジェクトが自己完結していることを確認できるため、`-z defs` オプションを使用することをお勧めします。自己完結しているオブジェクトでは、シンボル参照がそのオブジェクト内またはそのオブジェクトの依存関係の中で解決されています。

`-z nodefs` オプションは、未定義のシンボルを許可します。歴史的経緯から、共有オブジェクトを作成する場合には、この動作がデフォルトになっています。このオプションを実行可能ファイルに対して指定した場合の、未定義のシンボルに対する参照の動作は不確定です。`-z nodefs` オプションを使用することは、お勧めしません。

`-z direct | nodirect`

コマンド行であとに続くすべての依存関係への直接結合を有効または無効にします。このオプションにより、対応するグローバルなオプション `-Bdirect` よりも直接結合を効率よく制御できます。`-z direct` オプションは、次の点で `-Bdirect` オプションとも異なります。直接結合の情報は、シンボル参照と作成されたオブジェクト内で関連付けられた定義の間では確立されません。レイジー読み込みは無効です。

`-z discard-unused= item1, item2, ...`

デフォルトでは、リンカーは、未使用の空のセクションを破棄します。その他のカテゴリの入力素材は、リンク編集集中に、未使用であると判定できます。`-z discard-unused` オプションを使用すると、このような項目の自動削除が可能になります。次の `item` トークンが認識されます。

セクション

未使用のセクションは、リンク編集から作成された出力ファイルから破棄されます。

ファイル

未使用の再配置可能オブジェクトファイルは、リンク編集から作成された出力ファイルから破棄されます。

入力の再配置可能オブジェクトファイルは、その再配置可能オブジェクトによって提供されるすべての割り当て可能セクションが未使用である場合に、未使用であると判定されます。`-z guidance` オプションの「必須でない再配置可能オブジェクトファイル」の説明も参照してください。

依存関係

未使用の、明示的な、共有オブジェクトの依存関係は、リンク編集から作成された出力ファイル内に記録されません。

明示的な依存関係とは、パス名を使用して、またはより一般には `-l` オプションを使用してコマンド行で定義された依存関係のことです。明示的な依存関係がほかのオブジェクトに依存する場合があります、これを暗黙の依存関係と呼びます。明示的な依存関係は、2つの条件が満たされた場合に、未使用であると判定されます。

- この依存関係によって提供されるグローバルシンボルが、構築されるオブジェクトから参照されていない。
- この依存関係が、どの暗黙の依存関係の要件も補償していない。

`-z guidance` オプションの「必須でない、または補償している依存関係」の説明も参照してください。

none

未使用の処理 (未使用の空のセクションを削除するデフォルトのアクションを含む) をすべて無効にします。

『[Oracle Solaris 11.1 リンカーとライブラリガイド](#)』の「使用されない対象物の削除」も参照してください。

`-z endfiltee`

フィルタ処理の対象オブジェクトに印をつけます。フィルタによる検索処理は、このオブジェクトを検出した時点で終了します。『[リンカーとライブラリ](#)』の「[「フィルティー」検索の縮小](#)」を参照してください。

`-z fatal-warnings | nofatal-warnings`

`--fatal-warnings | --no-fatal-warnings`

`-z fatal-warnings` と `--fatal-warnings` オプションを指定すると、リンカーは警告を致命的エラーとして扱います。

`-z nofatal-warnings` と `--no-fatal-warnings` オプションを指定すると、リンカーは警告を致命的でないエラーとして扱います。これはデフォルトの動作です。

`-z finiarray=function`

構築するオブジェクトの `.fini_array` セクションにエントリを追加します。`.fini_array` セクションが存在しない場合は、1つのセクションを新たに作成します。追加されたエントリは `function` を指すように初期化されます。『[リンカーとライブラリ](#)』の「[初期設定および終了セクション](#)」を参照してください。

-z globalaudit

このオプションは、**-P** オプションを使用して記録された監査ライブラリの定義を補足します。このオプションは、動的実行可能ファイルの構築時にのみ意味を持ちます。**-P** オプションを使用してオブジェクト内で定義された監査ライブラリを使用すると、通常、そのオブジェクトの直接の依存関係を監査できます。

-z globalaudit は、監査者をグローバルな監査者に昇格させることによって、すべての依存関係を監査できるようにします。『リンカーとライブラリ』の「監査インタフェースの呼び出し」を参照してください。

-P オプションと **-z globalaudit** オプションを使用して確立された監査者は、**LD_AUDIT** 環境変数を使用して確立された監査者と同等です。**ld.so.1(1)** を参照してください。

-z guidance[=item1,item2,...]

生成されるオブジェクトの品質を高めるか、それ以外のメリットがあると考えられる **ld** オプションを提案するガイダンスメッセージを表示します。提供される個々のガイダンスは、システムの進化に伴って変更される可能性があります。古いバージョンの **ld** で提供されていた廃止済みのガイダンスは、新しいバージョンでは削除されることがあります。同様に、新しいガイダンスが新しいバージョンの **ld** に追加されることがあります。したがって、ガイダンスには常に最新のベストプラクティスが表示されます。

ガイダンスを有効にすると同時に特定のガイダンスメッセージが表示されないようにするには、表示しないガイダンスのクラスを示す *item* トークンの一覧を指定します。これにより、不要なアドバイスは表示されなくなりますが、それ以外のガイダンスは活用できます。**ld** は認識されない *item* トークンを何も出力せず無視するため、指定した **ld** コマンド行を新旧さまざまなバージョンの Solaris で実行できます。

現在のバージョンの **ld** で提供されるガイダンスと、これらのメッセージを無効にするために使用する *item* トークンは次のとおりです。

必要な依存関係を指定する

動的実行可能ファイルと共有オブジェクトでは、必要とする依存関係をすべて明示的に定義する必要があります。ガイダンスでは、動的オブジェクトの作成時にシンボル参照が不十分な場合には、**-z defs** オプションの使用を推奨しています。このガイダンスは、**-z guidance=nofdefs** で無効にできます。

必須でない、または補償している依存関係を指定しない

動的実行可能ファイルおよび共有オブジェクトは、動的オブジェクトによって作成されたシンボル参照を満たさない明示的な依存関係を定義してはいけません。ガイダンスでは、必須でない、または未使用の依存関係を削除するよう推奨しています。未使用の依存関係は、2つのカテゴリのいずれかに分類できます。

- シンボル参照を満たさない明示的な依存関係。

- 構築される動的オブジェクトからのシンボル参照を満たさないが、暗黙の依存関係を補償している明示的な依存関係。-z discard-unused オプションの「依存関係」の説明を参照してください。

これらのどちらのカテゴリのガイダンス

も、-z guidance=nounused-dependencies または同義語の -z guidance=nounused で無効にできます。補償している依存関係のガイダンスは、-z guidance=nounused-compensators で無効にできます。

『Oracle Solaris 11.1 リンカーとライブラリガイド』の「使用されない対象物の削除」も参照してください。

必須でない再配置可能オブジェクトファイルを指定しない

作成される出力ファイルには、リンク編集に関与しているその他のオブジェクトによって割り当て可能セクションが参照されない再配置可能オブジェクトからのどのような情報も含まれてはいけません。ガイダンスでは、未使用の再配置可能オブジェクトを削除するよう推奨しています。このガイダンスは、-z guidance=nounused-files で無効にできます。

『Oracle Solaris 11.1 リンカーとライブラリガイド』の「使用されない対象物の削除」も参照してください。

レイジー読み込み

レイジー読み込みでは依存関係を識別する必要があります。ガイダンスでは、-z lazyload または -z nolazyload オプションが出現する前にすべての依存関係が処理される場合、-z lazyload オプションの使用を推奨しています。このガイダンスは、-z guidance=nolazyload で無効にできます。

直接結合

直接結合を使用して依存関係を参照する必要があります。ガイダンスでは、-B direct と -z direct オプションのどちらか、または -z nodirect オプションが出現する前に依存関係が処理される場合、B direct と z direct のどちらかのオプションの使用を推奨しています。このガイダンスは、-z guidance=nodirect で無効にできます。

ピュアテキストセグメント

動的オブジェクトでは、書き込み不可で割り当て可能なセクションに再配置を行なってはいけません。ガイダンスでは、テキストセグメントに対して再配置が残っていて、-z textwarn オプションも -z textoff オプションも検出されない場合、位置独立コード (PIC) でのオブジェクトのコンパイルを推奨しています。このガイダンスは、-z guidance=notext で無効にできます。

Mapfile の構文

すべての mapfile は、バージョン 2 の mapfile 構文を使用する必要があります。ガイダンスでは、バージョン 1 の構文を使用する mapfile が検出された場合、バージョン 2 の構文の使用を推奨しています。このガイダンスは -z guidance=nomapfile で無効にできます。

ライブラリ検索パス

ldによって不適切な依存関係が検出されると、何も出力されずに無視されます。たとえば、64ビットのオブジェクト生成時に検出された32ビットの依存関係は無視されます。このような依存関係は、誤った-Lオプションが設定されるなど、検索パスの設定が正しくないことから生じる可能性があります。この依存関係があっても問題ではありませんが、この依存関係を処理する必要はなく、解決の必要がある作成時の問題が隠れてしまうおそれがあります。ガイドンスでは、不適切な依存関係はすべて削除することを推奨しています。このガイドンスは、-z guidance=nolibpathで無効にできます。

また、-z guidance=noallでガイドンス機能をすべて無効にできます。より適切なオブジェクトを構築するためのガイドンスとアドバイスについての詳細は、『リンカーとライブラリ』の第7章「リンカーのクイックリファレンス」を参照してください。

-z help

--help

標準出力にコマンド行オプションのサマリーを出力して終了します。

-z ignore|record

リンク編集の一部として参照されない共有オブジェクトの依存関係は無視または記録します。

-z ignore および -z record は、コマンド行で検出された参照されない依存関係のldでの処理方法を切り替えるために使用される定位置オプションです。-z ignoreが検出されると、以降の参照されない依存関係はすべて、何も出力せずに無視されます。-z recordが検出されると、依存関係が参照されるかどうかには関係なく、すべての依存関係が記録されます。

デフォルトでは、依存関係が参照されるかどうかには関係なく、ldはすべての依存関係を記録します。定位置でない -z discard-unused=dependencies オプションを使用すると、この初期のデフォルトを変更できます。初期設定が確立されたあとは、-z ignore および -z record を使用してデフォルトの動作を変更できます。

-z initarray=function

構築するオブジェクトの .init_array セクションにエントリを追加します。 .init_array セクションが存在しない場合は、1つのセクションを新たに作成します。追加されたエントリは *function* を指すように初期化されます。『リンカーとライブラリ』の「初期設定および終了セクション」を参照してください。

-z initfirst

オブジェクトの実行時初期設定が完了してから、ほかのオブジェクトの実行時初期設定を同時処理します。また、同時プロセスから除去されたほかのすべてのオブジェクトの実行時最終設定が完了してから、オブジェクトの実行時最終設定が行われるようにします。このオプションは共有オブジェクトの作成時にのみ有効です。

-z interpose

直接結合よりも優先して検索するオブジェクトを指定します。実行時に、**-z interpose** オプションでオブジェクトにタグが付けられている場合、そのオブジェクトは明示的な割り込みとして識別されます。明示的な割り込みは、オブジェクトが **LD_PRELOAD** 環境変数を使用して読み込まれたときにも設定されます。オブジェクトの読み込み順序が原因で暗黙的な割り込みが発生することがありますが、この暗黙的な割り込みは実行時リンカーにとっては未知のものです。明示的な割り込みでは、オブジェクトの読み込み順序にかかわらず割り込みを発生させることができます。また、直接結合が有効な場合には、明示的な割り込みによって、実行時リンカーが任意の明示的な割り込み内でシンボルを検索できるようになります。

-z lazyload|nolazyload

遅延して読み込まれる動的な依存オブジェクトの指定を有効または無効にします。**lazyload** と指定された動的な依存オブジェクトは、初期プロセスの起動では読み込まれません。オブジェクトの読み込みは、はじめてこのオブジェクトへの結合が行われたときに行われます。注: レイジー読み込みには、正しい依存宣言と、プロセス内で使用される動的オブジェクトごとに関連する **runpaths** が必要です。『リンカーとライブラリ』の「動的依存関係の遅延読み込み」を参照してください。

-z ld32=arg1,arg2,...**-z ld64=arg1,arg2,...**

リンカーのクラスは、作成される出力ファイルのクラスおよびリンカーが実行されているオペレーティングシステムの機能によって変わります。**-z ld[32]64** オプションでは、すべてのリンカーの引数を定義できます。定義された引数は、それぞれ 32 ビットクラスまたは 64 ビットクラスのリンカーによって解釈のみが行われます。

たとえば、サポートするライブラリがクラスに固有のものである場合に、そのライブラリに正しいクラスを指定するには、次のようにします。

```
ld ... -z ld32=-Saudit32.so.1 -z ld64=-Saudit64.so.1 ...
```

-z loadfltr

フィルタ対象が実行時ただちに処理されることを要求するように、フィルタに指定します。通常、フィルタ処理は、シンボル参照がフィルタに結合されたときにはじめて行われます。このフラグを持つオブジェクトの実行時プロセスは、環境変数 **LD_LOADFLTR** が設定されている場合と同じように動作します。**ld.so.1(1)** を参照してください。

-z mapfile-add=name

name を既知の **mapfile** 条件入力式の述部の一覧に追加します。このオプションは、リンカーによって読み取られる最初の **mapfile** の先頭に次の行を挿入することと同等です。

```
$mapfile_version 2
$add name
```

-z mapfile-add で入力された名前を mapfile \$if および \$elif 指令とともに使用すると、mapfile 入力を条件に応じて処理できます。『リンカーとライブラリ』の第 10 章「mapfile」を参照してください。

-z muldefs

--allow-multiple-definition

複数のシンボル定義を許容します。デフォルトでは、再配置可能オブジェクト間で重複したシンボル定義が発生すると、致命的なエラーになります。ところが、このオプションを指定すると、エラーにはならず、最初の定義が有効な定義として使用されます。

-z nodelete

実行時に削除できないように、オブジェクトに指定します。このモードは、[dlopen\(3C\)](#) を RTLD_NODELETE モードとともに使用してプロセスにオブジェクトを追加する場合と同様です。

-z nodlopen

オブジェクトを、[dlopen\(3C\)](#) では利用できないように指定するか、[dlopen\(\)](#) によって指定されたオブジェクトとして指定します。または [dlopen\(\)](#) によって指定されたオブジェクトに必要な依存オブジェクトとして指定します。このオプションは共有オブジェクトの作成時にのみ有効です。

-z nodump

オブジェクトを、[dldump\(3C\)](#) では利用できないように指定します。

-z parent=*object*

出力オブジェクトのリンク先の親 オブジェクト (実行可能ファイルまたは共有オブジェクト) を指定します。通常、このオプションを使用するのは *plugin* 共有オブジェクトの作成時で、このオブジェクトは実行可能ファイルにより実行時に [dlopen\(\)](#) 関数を介してロードされます。親オブジェクトからのシンボルテーブルは、*plugin* オブジェクトからの参照を満たすために使用されます。『リンカーとライブラリ』の第 2 章「リンカー」を参照してください。

-z preinitarray=*function*

構築するオブジェクトの `.preinit_array` セクションにエントリを追加します。`.preinit_array` セクションが存在しない場合は、1つのセクションを新たに作成します。追加されたエントリは *function* を指すように初期化されます。『リンカーとライブラリ』の「初期設定および終了セクション」を参照してください。

-z relaxreloc

`ld` は、通常、削除された COMDAT セクションを参照するシンボルを使用して再配置を検出したときに致命的なエラーを出力します。`-z relaxreloc` が有効な場合、`ld` は、その代わりに、保持されていた COMDAT セクション内の同等のシンボルにこのような再配置をリダイレクトします。`-z relaxreloc` は、主にコンパイラ作成者を対象とした特殊なオプションで、一般的な用途向けではありません。

-z rescan-now

これらのオプションは、リンカーに渡されたアーカイブファイルを再走査します。デフォルトでは、アーカイブの検索は、コマンド行に指定されたときに一度だけ処理されます。通常アーカイブは、そのシンボル定義でアーカイブの前に指定されている参照を解決できるように、コマンド行の最後に指定します。ただし、実際には、アーカイブ間での相互依存関係を解決するために、アーカイブ自体を複数回指定しなければならないことがあります。

-z rescan-now は位置オプションで、コマンド行上で検出されるとただちにリンカーによって処理されます。コマンド行上でそのポイントに達したすべてのアーカイブはただちに処理され、シンボル参照を解決する追加のアーカイブメンバーを見つけ出そうとします。アーカイブ全体を走査しても新しい再配置可能オブジェクトが抽出されないと、アーカイブの再走査は終了します。

-z rescan は、位置独立型のオプションです。リンカーは、コマンド行全体の処理を完了するまで再走査作業を延期し、コマンド行上にあるすべてのアーカイブに対して最後の再走査作業を開始します。**-z rescan** の再走査作業では、初期化 (.init) セクションまたは最終設定 (.fini) セクションを含むオブジェクトが誤って処理され、それらのセクションのコードが動作しないことがあります。このため、**-z rescan** ではなく、**-z rescan-now** を使用することを推奨します。

-z rescan-start ... -z rescan-end

--start-group ... --end-group

-(... -)

アーカイブ再走査グループを定義します。これは位置構成で、終端区切りオプションが検出されるとリンカーによってただちに処理されます。グループ区切りオプション内で見つかったアーカイブはグループとして再処理され、シンボル参照を解決する追加のアーカイブメンバーを見つけ出そうとします。アーカイブ全体を走査しても新しい再配置可能オブジェクトが抽出されないと、アーカイブの再走査は終了します。アーカイブ再走査グループを入れ子にすることはできません。

-z strip-class=[!] class1,[!] class2,...

特定のクラスのセクションがすべての入力オブジェクトから削除されるため、これらのセクションは出力ファイルに追加されなくなります。このオプションを使用すると、出力ファイルから省略できるセクションに対してきめ細かな制御ができます。

以降の **strip** クラスの説明は、割り当て不可能なセクションにのみ適用されます。

各クラスのトークンを '!' の前に追加して、そのクラスの削除が禁止されていることを示すことができます。この定義は、**nonalloc** クラスと組み合わせる場合に役立つことがあります。たとえば、**'-z strip-class=nonalloc,!note'** を使用して、ノートセクションを除く、すべての割り当て不可のセクションを削除します。

取り除かれたセクションは、出力オブジェクトから完全に削除されます。`-z ancillary` オプションを使用すると、動的でないシンボルテーブル `.symtab` に関するこの動作と、それに関連したセクションが変更されます。デフォルトでは、シンボルテーブルは、プライマリオブジェクトと補助的なオブジェクトの両方に書き込まれます。取り除かれた場合、シンボルテーブルは補助的なオブジェクトにのみ書き込まれ、プライマリオブジェクト内に存在しないとしてマークされます。`.symtab` が `-z ancillary` を使用せずにオブジェクトから取り除かれた場合、このセクションは通常の方法で完全に削除されます。

次のクラスのセクションを定義できます。

`nonalloc`

割り当て不可のセクションをすべて削除します。これらのセクションは、`SHF_ALLOC` セクションフラグを含んでいないものとして識別されます。このクラスは、`symbol` クラスを除く、ほかのすべてのクラスをカプセル化します。`nonalloc` クラスは多くの場合、単独で十分であり、不要なすべてのセクションを削除できます。

`annotate`

すべての注釈セクションを削除します。これらのセクションは、メモリアクセスツールや、カバレッジ関連のツールによって使用される情報を提供します。これらのセクションは、`SHT_SUNW_ANNOTATE` セクションタイプを指定することで識別します。

`comment`

すべてのコメントセクションを削除します。これらのセクションは、`.comment` セクション名を指定することで識別します。あるいは、コメントセクションの操作には `mcs(1)` ユーティリティが一般的に使用されています。

`debug`

デバッグデータを含めるためによく使用されるセクションを削除します。これらのセクションは、`.compcom`, `.line`, `.stab*`、または `.debug*` セクション名を指定することで識別します。また、これらのセクションは、`SHT_SUNW_DEBUG*` セクションタイプを指定することで識別します。

`exclude`

実行可能セクションをすべて削除します。これらのセクションは、`SHF_EXCLUDE` セクションフラグを指定することで識別します。このクラスは、再配置可能オブジェクトを作成する場合に役立ちます。デフォルトでは、このようなセクションは、動的実行可能ファイルまたは共有オブジェクトの作成時には自動的に除外され、再配置可能オブジェクトの作成時には保持されます。

`note`

ノートセクションを削除します。これらのセクションは、`SHT_NOTE` セクションタイプを指定することで識別します。

symbol

出力ファイルが再配置可能オブジェクトでない場合に、割り当て不可のシンボルテーブルセクションと文字列テーブルセクションをすべて削除します。これらのセクションは、SHT_SYMTAB セクションタイプを指定することで識別します。関連する文字列テーブルもすべて削除されます。

-z stub

スタブ共有オブジェクトを生成します。スタブオブジェクトは共有オブジェクトで、mapfiles からすべて作成されます。また、コードやデータを持ちませんが、本物のオブジェクトと同じリンクインタフェースを提供します。スタブオブジェクトは、実行時には使用できません。ただし、スタブオブジェクトに対してアプリケーションを作成できます。その場合、スタブオブジェクトは、実行時に使用される本物のオブジェクトの名前を提供し、実行時に本物のオブジェクトを使用します。

スタブオブジェクトは共有オブジェクト用に対してのみ作成でき、エクスポートされる大域シンボルを定義する mapfile を提供する必要があります。そのため、-G および -M オプションが、-z stub の使用時に必要となります。スタブオブジェクトの作成時には、リンカーはコマンド行で指定されているオブジェクトとライブラリファイルをすべて無視するため、スタブを作成するためにこれらのファイルが存在している必要はありません。コンパイル手順が省略でき、リンカーの作業が比較的少ないため、スタブオブジェクトはすぐに作成できます。

「Stub Objects」 in 『Linker and Libraries Guide』を参照してください。

-z symbolcap

オブジェクトの機能を定義する再配置可能オブジェクトを、シンボルの機能を定義する再配置可能オブジェクトに変換します。[『リンカーとライブラリ』の「オブジェクト機能のシンボル機能への変換」](#)を参照してください。

-z target=sparc|x86

出力オブジェクトの機械タイプを指定します。サポートされるターゲットは、SPARC と x86 です。-64 オプションも存在していて、対応する 64 ビット機械タイプが使用されている場合を除き、指定されたターゲット用の 32 ビット機械タイプが使用されます。デフォルトでは、生成されるオブジェクトの機械タイプは、コマンド行から処理される最初の ELF オブジェクトによって決まります。オブジェクトを指定していない場合、機械タイプは、コマンド行から処理される最初のアーカイブ内で検出された最初のオブジェクトによって決まります。オブジェクトもアーカイブもない場合、リンカーはネイティブマシンを仮定します。このオプションは、mapfile からの入力のみで ld を使用してオブジェクトを直接作成するときに便利です。(-M の説明を参照)。また、最初のオブジェクトが目的の機械タイプのオブジェクトではない、異なる機械タイプのオブジェクトを含むアーカイブからすべてをリンクするというごくまれな場合にも便利です。

-z text | textoff | textwarn

これらのオプションは、動的モードでのみ使用できます。-z text オプションは、書き込み不可能な割り当て可能セクションに対する再配置が残っていた場

合、強制的に致命的エラーを発生させます。歴史的な理由から、この動作は、実行可能オブジェクトまたは共有オブジェクトを作成するときのデフォルトではありません。ただし、作成される動的オブジェクトのテキスト部分を実行中の複数のプロセス間で共有可能にするため、このオプションを使用することをお勧めします。オブジェクトをメモリーに読み込むときの再配置オーバーヘッドが最低限に抑えられることを共有テキストセグメントは保証します。『[リンカーとライブラリ](#)』の「[位置独立のコード](#)」を参照してください。

`-z textoff` オプションは、すべての割り当て可能セクション (書き込み不可能なセクションを含む) に対する再配置を許可します。共有オブジェクト作成時のデフォルトの設定です。

`-z textwarn` オプションは、書き込み不可能な割り当て可能セクションに対する再配置が残っていた場合、警告を表示します。実行可能オブジェクト作成時のデフォルトの設定です。

`-z verbose`

このオプションはリンクエディット時、警告診断を追加します。現在、このオプションによって次の警告が有効になります。

- 再配置の置き換えの疑わしい使用。
- 共有オブジェクト作成時の静的 TLS 再配置の制限された使用。
- シンボルの可視性の不一致。

将来、このオプションは、デフォルトで生成される詳細な診断を提供するように拡張される予定です。

`-zwrap=symbol`

`-wrap= symbol`

`--wrap= symbol`

ソースコードを変更しなくてもラッパーコードを出力オブジェクトにリンクできるようにするため、`symbol` への未定義の参照の名前を変更します。`-z wrap` が指定されている場合、`symbol` への未定義の参照はすべて `__wrap_symbol` 参照に変更され、`__real_symbol` への参照はすべて `symbol` 参照に変更されます。ユーザーは、`__wrap_symbol` 関数を含むオブジェクトを提供するものと想定されています。ラッパー関数では、ラップされる実際の関数を参照するために、`__real_symbol` を呼び出すことができます。

次に、`malloc(3C)` 関数のラッパーの例を示します。

```
void *
__wrap_malloc(size_t c)
{
    (void) printf("malloc called with %zu\n", c);
    return (__real_malloc(c));
}
```

すべてのオブジェクトをコンパイルするために `-z wrap=malloc` を使用してほかのコードをこのファイルとリンクする場合、`malloc` へのすべての呼び出しは

`__wrap_malloc` 関数を代わりに呼び出します。`__real_malloc` への呼び出しは、本物の `malloc` 関数を呼び出します。

本物の関数とラップされた関数は別々のソースファイルで保持するようにしてください。そうしないと、コンパイラまたはアセンブラが呼び出しの解決をリンカーに任せずに自分で行うため、ラップの発生が妨げられることがあります。

セカンダリオプション

次のオプションは、一般にはあまり使用されません。これらのオプションは、下位互換性、非常に特殊な機能、または機能強化されたバリエーションに置き換えられたオプションを提供します。

-a

静的モードでのみ、静的実行可能ファイルを生成します。未定義の参照があるとエラーを発行します (静的モードのデフォルトの動作)。-a オプションは、-r オプションと同時に使用することはできません。「機能説明」の「静的実行可能ファイル」を参照してください。

-b

動的モード専用です。動的実行可能ファイルに対して、共有オブジェクト内のシンボルを参照するような特殊な再配置処理を行いません。-b オプションを指定しない場合、リンカーは、テキストセグメントが読み取り専用になるように動的実行可能ファイル内でいくつかの方法を適用します。1つの方法は、共有オブジェクトで定義されている関数への参照に対して位置独立型の特別な再配置を作成することです。もう1つの方法は、共有オブジェクト内に定義されているデータオブジェクトが実行時に実行可能ファイルのメモリーイメージにコピーされるように設定することです。

このオプションは、特殊化された動的オブジェクトを対象としているため、一般的な用途に使用することはお勧めしません。-このオプションを使用すると、オブジェクトを共有可能にするために必要な特殊処理がすべて無効になります。また、64ビット実行可能オブジェクトの再配置を妨げることもあります。

-B group

共有オブジェクトとその依存オブジェクトを1つのグループにまとめます。グループ内のオブジェクトは、実行時に、グループ内のほかのメンバーに結合されます。このモードは、[dlopen\(3C\)](#) を `RTLD_GROUP` モードとともに使用してプロセスにオブジェクトを追加する場合と同様です。あるグループに属しているオブジェクトに対して明示的な依存関係を持つオブジェクトは、自分自身もそのグループのメンバーになります。

グループは自己完結している (依存関係がグループ内で解決されている) 必要があるため、-z `defs` オプションを追加して、グループが自己完結していることを確認してください。-

グループを確立すると、オブジェクトのグループの結合を制御するための基本的な手段が提供されます。ただし、より適切な制御は、直接結合で実現できます。-B `direct` オプションを参照してください。

-B symbolic

動的モード専用です。共有オブジェクトを作成する際、可能であれば、大域シンボルへの参照を共有オブジェクト内の参照定義に結合します。通常、共有オブジェクト内の大域シンボルへの参照は、定義が使用可能でも実行時まで結合されません。このため、実行可能オブジェクト内またはほかの共有オブジェクト内に定義された同一のシンボルによって、オブジェクト自体の定義が無効になる可能性があります。-z defs オプションによって無効にしないかぎり、ld はシンボルが未定義であることを知らせる警告を出力します。

このオプションは、特殊化された動的オブジェクトを対象としているため、一般的な用途に使用することはお勧めしません。-オブジェクトに必要な実行時の再配置処理を削減するには、直接結合の使用とともに、バージョン定義の作成をお勧めします。-B direct オプションを参照してください。

-dy | n

-dy が指定されている場合 (デフォルト)、ld は動的リンクを使用します。-dn が指定されている場合、ld は静的リンクを使用します。「機能説明」の「静的実行可能ファイル」および -B dynamic | static の項目を参照してください。

-I name**--dynamic-linker name**

実行可能オブジェクトの作成時、プログラムヘッダーに書き込まれるインタプリタのパス名として *name* を使用します。静的モードのデフォルトの設定では、インタプリタは使用されません。動的モードのデフォルトの設定では、実行時リンカー **ld.so.1(1)** が使用されます。どちらの場合も、-I *name* でデフォルトの設定を無効にできます。**exec(2)** は、a.out を読み込む際にこのインタプリタをロードし、a.out ではなくインタプリタに制御を渡します。

-z absexec

動的実行可能ファイルを構築するときのみ使用されます。このオプションは、外部の絶対的シンボルへの参照を、実行時ではなく今すぐに解決するよう指示します。これによって、ある条件が整った場合には、動的オブジェクトがスワップ領域を大量に消費する可能性のあるテキストの再配置を行わなくなります。

-z combrelloc | nocombreloc

デフォルトでは、ld は、実行可能ファイルまたは共有オブジェクトを作成するときに複数の再配置セクションを結合します。このセクション結合は再配置可能オブジェクトとは異なるもので、再配置可能セクションでは、再配置の適用が必要なセクションと 1 対 1 の関係が維持されます。-z nocombreloc オプションは、この再配置セクションのマージを無効にし、元の再配置可能オブジェクトで見つかった 1 対 1 の関係を保持します。

ld は、データ再配置セクションのエントリを、そのシンボル参照に従ってソートします。これにより、実行時のシンボル参照を最低限に抑えます。複数の再配置セクションが結合されている場合は、これにより、オブジェクトがメモ

リーに読み込まれるときの再配置オーバーヘッドを最低限に抑え、動的オブジェクトの実行時の読み込みを速くします。

これまでは、個々の再配置セクションはすべての実行可能ファイルまたは共有オブジェクトに引き継がれ、前述した再配置セクションのマージを有効にするには `-z combrelloc` オプションが必要でした。現在では、再配置セクションのマージはデフォルトになっています。`-z combrelloc` オプションは古い構築環境のために受け入れられますが、このオプションは不必要で効果はありません。

-z groupperm | nogroupperm

一意のグループにつながる依存関係の割り当て、または割り当ての解除を行います。グループに対して依存関係の割り当てを行うと、`-B group` オプションを使って依存関係を構築した場合と同じ効果が得られます。

-z nocompstrtab

ELF 文字列テーブルおよびコメントセクションの圧縮を無効にします。デフォルトでは、文字列の圧縮は、`SHT_STRTAB` セクション、`SHF_MERGE` と `SHF_STRINGS` のセクションフラグが設定されている `SHT_PROGBITS` セクション、およびコメントセクションに適用されます。

[mcs\(1\)](#) ユーティリティを `-c` オプションとともに使用すると、オブジェクトが構築されたあとにコメントセクションを圧縮できます。

-z nodefaultlib

オブジェクトが、実行時デフォルトライブラリ検索パス (`LD_LIBRARY_PATH` や実行パスの後に使用) を無視するように指定します。このように指定すると、すべての依存オブジェクトが実行パスを使用することになります。

-z noldynsym

動的実行可能ファイルまたは共有オブジェクトに `.SUNW_ldynsym` セクションが含まれないようにします。`.SUNW_ldynsym` セクションは、ローカル関数用のシンボルを提供することで `.dynsym` セクションを拡張します。ローカル関数のシンボルを使用すると、デバッグで、シンボル名などが削除されたプログラムからスタックトレースにローカル関数名を表示できるようになります。同様に、[dladdr\(3C\)](#) では、より正確な結果を出力できます。

`-z noldynsym` オプションも、`.SUNW_ldynsym` セクションに関連する2つのソートセクションが含まないようにします。`.SUNW_dynsymssort` セクションでは、通常の関数と変数シンボルにソートアクセスできます。`.SUNW_dyntlssort` セクションでは、スレッドローカル記憶領域 (TLS) 変数シンボルにソートアクセスできます。

`.SUNW_ldynsym`、`.SUNW_dynsymssort`、および `.SUNW_dyntlssort` セクションは、結果ファイルの割り当て可能なテキストセグメントの一部となりますが、[strip\(1\)](#) によって削除することはできません。したがって、これらのセクションが含まれないようにする唯一の方法は、`-z noldynsym` オプションを使用することです。

-z nopartial

部分的に初期化されたシンボルが、再配置可能なオブジェクト内に定義されている場合、このシンボルは、生成される出力ファイルで展開されます。

-z now

オブジェクトがレイジーでない実行時結合を要求するように指定します。このモードは、[dlopen\(3C\)](#) を RTLD_NOW モードとともに使用してプロセスにオブジェクトを追加する場合と同様です。また、このモードは、LD_BIND_NOW 環境変数を有効にする場合と同様です。[ld.so.1\(1\)](#) を参照してください。

-z origin

実行時ただちに \$ORIGIN を処理することを要求するように、オブジェクトに指定します。この動作は現在はデフォルトです。このオプションは、従来機能との互換性のためにのみ残されています。

-z redlocsym

SHT_SYMTAB シンボルテーブルから、SECT シンボル以外のすべてのローカルシンボルを削除します。ローカルシンボルを参照する配置はすべて更新され、SECT シンボルを参照するようになります。このオプションを使用すると、特殊なオブジェクトのシンボルテーブルのサイズを大幅に削減できます。-z strip-class および -z noldynsym オプションも参照してください。

-z redlocsym オプションは、オペレーティングシステムカーネル内部で使用されるような特殊なオブジェクトに対して有用ですが、一般的な用途には推奨されません。シンボルテーブル SHT_SYMTAB のサイズは実行時動作に影響を与えず、ローカルシンボルの削除はプロセスの可観測性に悪影響を与える場合があります。ローカルシンボルを削除すると、コンパイラドライバ -g オプションを使用して生成されるデバッグ情報が少なくなる可能性があります。また、ローカルシンボルを削除すると、通常は .SUNW_ldynsym セクションに書き込まれる情報も削除されるため、[pstack\(1\)](#) や [truss\(1\)](#) などのデバッグやツールの有効性が低下します。

-z rescan

リンク編集に提供されたアーカイブファイルの再スキャンを起動する位置独立オプション。リンカーは、コマンド行全体の処理を完了するまで再走査作業を延期し、コマンド行上にあるすべてのアーカイブに対して最後の再走査作業を開始します。-z rescan の再走査作業では、初期化 (.init) セクションまたは最終設定 (.fini) セクションを含むオブジェクトが誤って処理され、それらのセクションのコードが動作しないことがあります。このため、-z rescan ではなく、-z rescan-now を使用することを推奨します。

環境変数**LD_ALTEXC**

代替リンカーパス名。ld が実行され、この代替リンカーに制御が渡されます。この環境変数により、さまざまなコンパイラドライバから呼び出されるデフォルトのリンカーを上書きする汎用的な手段が提供されます。-z altexec64 オプションを参照してください。

LD_LIBRARY_PATH

-l オプションで指定されたライブラリを検索するディレクトリのリストを指定します。複数のディレクトリを指定する場合は、ディレクトリとディレクトリの間をコロンで区切ります。2つのディレクトリをセミコロンで区切る方式がもっとも一般的です。

dirlist1;dirlist2

次のように `ld` の呼び出し時に `-L` が複数回指定されている場合、

ld ... -Lpath1 ... -Lpathn ...

検索の順番は、次のようになります。

dirlist1 path1 ... pathn dirlist2 **LIBPATH**

ディレクトリリストにセミコロンが含まれていない場合、*dirlist2* と解釈されます。

`LD_LIBRARY_PATH` 環境変数も、動的な依存関係を検索する実行時リンカーに影響を及ぼします。

この環境変数には、`_32` または `_64` という接尾辞を指定できます。この接尾辞を追加することで、`LD_LIBRARY_PATH` 環境変数は 32 ビットのプロセスまたは 64 ビットのプロセス専用になり、接尾辞のない `LD_LIBRARY_PATH` 環境変数が有効な場合でも、優先的に使用されます。

LD_NOEXEC_64

64 ビットのリンカーが自動的に実行されないようにします。デフォルトでは、最初の再配置可能オブジェクトの ELF クラスが 64 ビットオブジェクトである場合は、自動的に 64 ビット版のリンカーが起動されます。32 ビット版のリンカーが作成できる 64 ビットイメージにはいくつかの制限がありますが、リンク編集の中には、32 ビットのリンカーを使用した方が速いものも存在します。

LD_OPTIONS

`ld` のデフォルトのオプションセットです。`LD_OPTIONS` の値は、コマンド行で `ld` の起動コマンドの直後に指定されたものと解釈されます。つまり、次のように指定した場合と同じ結果になります。

ld \$LD_OPTIONS ... other-arguments ...

LD_RUN_PATH

リンカーに実行パスを指定する代替手段です (`-R` オプションを参照)。 `LD_RUN_PATH` と `-R` オプションを両方指定すると、`-R` オプションが優先されます。

SGS_SUPPORT

リンカーによって読み込まれ、リンクプロセスの情報を与えられた共有オブジェクトを、コロンで区切った形式で一覧します。この環境変数には、`_32` または `_64` という接尾辞を指定できます。こうすることで、この環境変数を、`ld` の 32 ビットクラスまたは 64 ビットクラスのどちらか専用にできます。この指定

は、現在有効になっている接尾辞の付いていない環境変数の指定より優先されます。-s オプションを参照してください。

文字列 LD_ で始まる環境変数名は、ld と ld.so.1(1) の拡張用として予約されています。

ファイル	libx.so	共有オブジェクトライブラリ。
	libx.a	アーカイブライブラリ
	a.out	デフォルトの出力ファイル。
	LIBPATH	32 ビットライブラリの場合、デフォルトの検索パスは、順番に、/lib、そして /usr/lib です。64 ビットライブラリの場合、デフォルトの検索パスは、順番に、/lib/64、そして /usr/lib/64 です。
	/usr/lib/ld	リンク編集時に使用可能な複数の mapfiles を含むディレクトリです。これらの mapfiles により、メモリーレイアウトの定義、bss の整列、および実行不可能なスタックの定義など、さまざまな機能が提供されます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/linker
インタフェースの安定性	確実

関連項目 [as\(1\)](#), [crle\(1\)](#), [gprof\(1\)](#), [ld.so.1\(1\)](#), [ldd\(1\)](#), [mcs\(1\)](#), [pvs\(1\)](#), [strip\(1\)](#), [exec\(2\)](#), [stat\(2\)](#), [dlopen\(3C\)](#), [dldump\(3C\)](#), [elf\(3ELF\)](#), [ar.h\(3HEAD\)](#), [a.out\(4\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

注意事項 ld のデフォルト動作を指定するためのオプションは、歴史的な理由から残されています。しかし、動的オブジェクトが広く使用されている現在のプログラミング環境では、従来とは異なるデフォルトの方が有効である場合が少なくありません。ただし、従来のデフォルトは、既存のプログラム開発環境との互換性を維持するために残しておく必要があります。このマニュアルでは従来のデフォルトは、可能な限り、そのことを明示する方法で記述します。現在の推奨されるオプションについては、『リンカーとライブラリ』の第7章「リンカーのクイックリファレンス」を参照してください。

ld により作成されるファイルがすでに存在する場合、そのファイルは、すべての入力ファイルが処理されたあとにリンクが解除され、指定された名前での新しいファイルが作成されます。これにより、ld は、古いファイルの内容にアクセスしている既存のプロセスが実行を継続できるようにする一方で、同時にそのファイルの新しい

バージョンを作成できます。古いファイルにほかにリンクが含まれていない場合は、そのファイルを参照している最後のプロセスが終了すると、その削除されたファイルのディスク領域が解放されます。

Oracle Solaris 11 では、作成されるファイルがすでに存在するときの `ld` の動作が変更されました。古いバージョンでは、既存のファイルはその場所で書き換えられるため、そのファイルを使用している動作中のプロセスが壊れてしまう可能性があります。この変更は、ファイルシステム内にハードリンクが複数ある出力ファイルに影響します。以前は、すべてのリンクがそのまま維持され、新しいファイルの内容にもアクセスしていました。新しい `ld` の動作はこのようなリンクを解除するため、指定された出力ファイル名だけが新しいファイルを参照します。それ以外のリンクはすべて、古いファイルを引き続き参照します。一貫性のある動作を保証するには、リンカーの出力ファイルへの複数のハードリンクに依存するアプリケーションで、そのほかのファイル名を明示的に削除し、リンクし直す必要があります。

名前	ldapdelete – LDAP エントリ削除ツール
形式	<pre>ldapdelete [-n] [-v] [-c] [-d <i>debuglevel</i>] [-f <i>file</i>] [-D <i>bindDN</i>] [-w <i>passwd</i> -j <i>file</i>] [-J <i>[:criticality]</i>] [-?] [-H] [-h <i>ldaphost</i>] [-V <i>version</i>] [-i <i>locale</i>] [-k <i>path</i>] [-P <i>path</i>] [-N <i>certificate</i>] [-y <i>proxyid</i>] [-p <i>ldapport</i>] [-O <i>hoplimit</i>] [-o <i>attributename=value</i>] [-W <i>password</i>] [<i>dn</i>]...</pre>
機能説明	ldapdelete ユーティリティーは LDAP サーバーへの接続を開いてバインドし、1 つ以上のエントリを削除します。1 つ以上の <i>dn</i> 引数が指定されている場合は、それらの識別名を持つエントリが削除されます。<i>dn</i> 引数が指定されていない場合は、DN のリストが <i>file</i> から (-f オプションが指定されている場合)、または標準入力から読み取られます。
オプション	サポートしているオプションは、次のとおりです。 -a 分岐を削除するときに確認の質問を省略します。 -c 継続動作モード。エラーが報告されても、ldapdelete は削除を続行します。デフォルトでは、エラーを報告したあとに終了します。 -d <i>debuglevel</i> LDAP のデバッグレベルを設定します。ldapdelete に役立つデバッグレベルは次のとおりです。 1 トレース 2 パケット 4 引数 32 フィルタ 128 アクセス制御 複数のカテゴリのデバッグ情報を要求するには、マスクを追加します。たとえば、トレースとフィルタの情報を要求するには、33 の <i>debuglevel</i> を指定します。 -D <i>bindDN</i> 識別名 <i>bindDN</i> を使用してディレクトリにバインドします。

- E 認証応答制御を使用してバインド識別情報を公開 (報告) するようサーバーに要求します。
- f *file* 標準入力の代わりに *file* からエントリ削除情報を読み取ります。
- ? すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
- H すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
- h *ldaphost* LDAP サーバーが実行されている代替ホストを指定します。
- i *locale* コマンド行入力で使用する文字セットを指定します。デフォルトは、LANG 環境変数で指定された文字セットです。このオプションを使用して、指定した文字セットから UTF8 への変換を実行し、それによって LANG 設定をオーバーライドすることもできます。
- この引数を使用すると、バインド DN とターゲット DN を指定された文字セットに入力できます。ldapdelete ツールは、検索要求を処理する前に、これらの引数からの入力を変換します。たとえば、-i no は、バインド DN とターゲット DN がノルウェー語で提供されていることを示します。
- このオプションは、コマンド行入力にのみ影響を与えます。つまり、DN を含むファイルを指定した場合 (-f オプションを使用)、ldapdelete は、そのファイル内のデータを変換しません。
- j *filename* バインド DN 用のパスワードまたは SSL クライアントの鍵データベース用のパスワードを含む

	ファイルを指定します。パスワードを保護するには、このオプションをスクリプトで使い、セキュリティで保護されたファイルにパスワードを保存します。このオプションは、<code>-w</code> および <code>-W</code> オプションとは相互に排他的です。<code>-j</code> オプションは、<code>-j</code> と <code>-w/-W</code> の中ではより安全な代替オプションです。
<code>-J [:criticality[:value ::b64value b64value :fileurl]]</code>	<code>criticality</code> はブール値です (デフォルトは <code>false</code>)。
<code>-k path</code>	変換ルーチンを含むディレクトリへのパスを指定します。これらのルーチンは、ディレクトリサーバーでデフォルトではサポートされていないロケールを指定したい場合に使用されます。これは NLS のサポート用です。
<code>-M</code>	スマートリフェラルを管理します。これらが操作のターゲットである場合は、リフェラルに従って取得されたエントリではなく、リフェラルを含む実際のエントリを削除します。
<code>-n</code>	実行される処理を表示しますが、実際にはエントリを削除しません。オプション <code>-v</code> および <code>-d</code> と組み合わせるとデバッグに役立ちます。
<code>-N certificate</code>	証明書に基づくクライアント認証で使用する証明書名を指定します。たとえば、<code>-N "Directory-Cert"</code> と指定します。
<code>-o attributename=value</code>	SASL メカニズムや、セキュリティプロパティなどのその他のオプションの場合、動作モード、認可 ID、認証 IDなどを指定します。

各種の属性名とそれらの値は次のとおりです。

<code>secProp="number"</code>	SASL セキュリティプロパティの定義用。
<code>realm="value"</code>	SASL レalmを指定します(デフォルトは <code>realm=none</code>)。
<code>authzid="value"</code>	SASL バインドの認可 ID 名を指定します。
<code>authid="value"</code>	SASL バインドの認証 ID を指定します。
<code>mech="value"</code>	さまざまな SASL メカニズムを指定します。

`-O hopLimit`

削除するエントリの検索中に従うべきリフェラルホップの最大数を指定します。デフォルトでは、制限はありません。

`-p ldapport`

LDAP サーバーが待機している代替 TCP ポートを指定します。

`-P path`

クライアントの証明書データベースのパスとファイル名を指定します。例:

`-P /home/uid/.netscape/cert7.db`

このコマンドをディレクトリサーバーと同じホスト上で使用している場合は、そのサーバーの独自の証明書データベースを使用できます。例:

`-P installDir/ldap-serverID/alias/cert7.db`

サーバー認証のみを指定するには、`-P` オプションだけを使用します。

`-v`

冗長モードを使用し、診断が標準出力に書き込まれるようにします。

`-V version`

削除操作に使用される LDAP プロトコルのバージョン番号 (2 または 3 のどちらか) を指定します。LDAP v3 がデフォルトです。v3 をサポートしていないサーバーに接続しているときは、LDAP v2 を指定します。

`-W password`

`-P` オプションで指定されたクライアントの鍵データベースのパスワードを指定します。このオプションは、証明書に基づくクライアント認証では必須です。コマンド行で `password` を指定すると、システム上のほかのユーザーが `ps` コマンドを使用してそのパスワードを表示できるため、セキュリティの問題が発生します。ファイルからパスワードを指定するには、代わりに `-j` を使用します。このオプションは、`-j` とは相互に排他的です。

`-w passwd`

ディレクトリへの認証のパスワードとして `passwd` を使用します。認証に使用されるパスワードを指定するために `-w passwd` を使用すると、システムのほかのユーザーが `ps` コマンドを使用して、スクリプトファイルまたはシェルの履歴でそのパスワードを表示できるようになります。このオプションを指定せずに `ldapdelete` コマンドを使用した場合、このコマンドはパスワードの入力を求め、それを標準入力から

	読み取ります。-w オプションを指定せずに使用されると、パスワードがほかのユーザーに表示されなくなります。
-Y proxyid	削除操作で使用するプロキシ DN (プロキシ設定された認可 ID) を、通常はシェルの二重引用符 ("") 内に指定します。
-Z	証明書に基づくクライアント認証を提供するために SSL を使用することを指定します。このオプションには、-N と SSL パスワード、および証明書と鍵データベースを識別するために必要なその他のすべての SSL オプションが必要です。

オペランド 次のオペランドを指定できます。

dn 削除するエントリの 1 つまたは複数の識別名を指定します。

使用例

例1 エントリの削除

XYZ Corporation の組織エントリの真下にある `commonName Delete Me` で指定されたエントリを削除するには、次のコマンドを使用します。

```
example% ldapdelete -D "cn=Administrator, o=XYZ, c=US" \
    "cn=Delete Me, o=XYZ, c=US"
```

例2 SASL 認証を使用したエントリの削除

XYZ Corporation の組織エントリの真下にある `commonName 「Delete Me」` で指定されたエントリを削除するには、次のコマンドを使用します。

```
example% ldapdelete -o mech=DIGEST-MD5 -o secProp=noanonymous \
-o realm=none -o authid="dn:uid=foo,o=XYZ, c=US" \
    "cn=Delete Me, o=XYZ, c=US"
```

属性 属性についての詳細は、[attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

終了ステータス	次の終了ステータスが返されます。 0 正常終了。 0 以外 エラーが発生した。診断メッセージが標準エラー出力に書き込まれます。
関連項目	ldapadd(1) , ldapmodify(1) , ldapmodrdn(1) , ldapsearch(1) , ldap_get_option(3LDAP) , ldap_set_option(3LDAP) , attributes(5)
注意事項	- <i>M authentication</i> オプションは「廃止」です。

名前 ldaplist - 構成済みのプロファイルを使用した LDAP ディレクトリからのネーミング情報の検索および一覧表示

形式

```
/usr/bin/ldaplist [-dlv] [-h LDAP_server[:serverPort]] [-M domainName]
    [-N profileName] [-a authenticationMethod] [-P certifiPath]
    [-D bindDN] [-w bindPassword] [-j passwdFile]]
    [database [key]...]

/usr/bin/ldaplist -g

/usr/bin/ldaplist -h
```

機能説明

-h LDAP_server[:serverPort] オプションが指定されている場合、ldaplist はオプションによって指定されたサーバーへの接続を確立して、-N オプションで指定された DUAProfile を取得します。ldaplist は、取得された構成で説明されているディレクトリからの情報を一覧表示します。

デフォルトでは (-h LDAP_server[:serverPort] オプションが指定されていない場合)、このユーティリティーは、クライアントの初期化フェーズ中に [ldapclient\(1M\)](#) によって生成された LDAP 構成ファイルで定義されている LDAP ディレクトリサービスからのネーミング情報を検索して一覧表示します。このユーティリティーをデフォルトモードで使用するには、Oracle Solaris LDAP クライアントがあらかじめ設定されている必要があります。

このデータベースは、[nsswitch.conf\(4\)](#) で定義されているコンテナ名またはデータベース名のどちらかです。コンテナとは、ネームサービス情報を含むディレクトリ情報ツリー (DIT) 内のリーフ以外のエントリのことです。コンテナ名は、構成ファイルで定義されている defaultSearchBase を基準としたコンテナの LDAP 相対識別名 (RDN) です。たとえば、ou=people という名前のコンテナの場合、データベース名は nsswitch.conf で指定されたデータベースです。このデータベースはコンテナにマップされます。たとえば、passwd は ou=people にマップされます。無効なデータベースが指定されている場合、そのデータベースは汎用のコンテナ (たとえば、nisMapName=name) にマップされます。

この鍵は、データベース内で検索される属性値です。同じデータベース内で検索される複数の鍵を指定できます。この鍵は、attribute=value または value の 2 つの形式のどちらかで指定できます。最初の場合は、ldaplist からサーバーに検索キーが渡されます。後者の場合は、データベースの指定方法に応じて属性が割り当てられます。データベースがコンテナ名である場合は、「cn」属性タイプが使用されます。データベースが nsswitch.conf で定義されている有効なデータベース名である場合は、定義済みの属性タイプが使用されます (下の表を参照)。データベースが無効なデータベース名である場合は、cn が属性タイプとして使用されます。

ldaplist ユーティリティーは、RFC 2307bis (現在は IETF ドラフト) で定義されたスキーマに依存します。LDAP サーバー上に格納されるデータは、プロファイルにスキーママッピング定義が含まれていないかぎり、このスキーマに基づいて格納される必要があります。スキーママッピングについての詳細は、[ldapclient\(1M\)](#) を参照

してください。次の表は、データベース名からコンテナへのデフォルトのマッピング、LDAP オブジェクトクラス、および鍵で定義されていない場合に使用される属性タイプを示しています。

Database	Object Class	Attribute Type	Container
aliases	mailGroup	cn	ou=Aliases
automount	nisObject	cn	automountMapName=auto_*
bootparams	bootableDevice	cn	ou=Ethers
ethers	ieee802Device	cn	ou=Ethers
group	posixgroup	cn	ou=Group
hosts	ipHost	cn	ou=Hosts
ipnodes	ipHost	cn	ou=Hosts
netgroup	ipNetgroup	cn	ou=Netgroup
netmasks	ipNetwork	ipnetworknumber	ou=Networks
networks	ipNetwork	ipnetworknumber	ou=Networks
passwd	posixAccount	uid	ou=People
protocols	ipProtocol	cn	ou=Protocols
publickey	nisKeyObject	uidnumber	ou=People
		cn	ou=Hosts
rpc	oncRpc	cn	ou=Rpc
services	ipService	cn	ou=Services
printers	printerService	printer-uri	ou=printers
auth_attr	SolarisAuthAttr	nameT	ou=SolarisAuthAttr
prof_attr	SolarisProfAttr	nameT	ou=SolarisProfAttr
exec_attr	SolarisExecAttr	nameT	ou=SolarisProfAttr
user_attr	SolarisUserAttr	uidT	ou=people
projects	SolarisProject	SolarisProjectID	ou=projects

システムが Trusted Extensions を使用して構成されている場合は、次のデータベースだけが使用できます。

tnrhtp	ipTnetTemplate	ipTnetTemplateName	ou=ipTnet
tnrhdb	ipTnetHost	ipTnetNumber	ou=ipTnet

- automount データベースの場合、コンテナ列の auto_* は auto_home、auto_direct、... を表します。
- publickey データベースで、鍵が数字で始まっている場合は、UID 番号として解釈されます。鍵が数字以外で始まっている場合は、ホスト名として解釈されません。

ldaplist ユーティリティーは、鍵の中にワイルドカード「*」を使用した部分文字列の検索をサポートしています。たとえば、「my*」は、「my」で始まるすべての文字列に一致します。一部のシェル環境では、ワイルドカードが含まれた鍵を引用することが必要な場合があります。

鍵が指定されていない場合は、現在の検索 baseDN 内のすべてのコンテナが一覧表示されます。

オプション

サポートしているオプションは、次のとおりです。

-a *authenticationMethod*

認証方法を指定します。デフォルト値は、プロファイルで構成されているものです。サポートされる認証方法は次のとおりです。

```
simple
sasl/CRAM-MD5
sasl/DIGEST-MD5
tls:simple
tls:sasl/CRAM-MD5
tls:sasl/DIGEST-MD5
```

simple を選択すると、パスワードがネットワーク経由で平文で送信されます。これを使用することは決してお勧めできません。

さらに、認証を使用しないプロファイルを使用してクライアントが構成されている場合、つまり、*credentialLevel* 属性が **anonymous** に設定されているか、または *authenticationMethod* が **none** に設定されている場合、ユーザーはこのオプションを使用して認証方法を指定する必要があります。

-d

エントリではなく、指定されたデータベースの属性を一覧表示します。デフォルトでは、エントリが一覧表示されます。

-D *bindDN*

要求されたデータベースへの読み取り権を持つエントリを指定します。

-g

データベースのマッピングを一覧表示します。

-h

データベースのマッピングを一覧表示します。

このオプションは非推奨になりました。

-h *LDAP_server[:serverPort]*

エントリが読み取られる LDAP サーバーのアドレス (または名前) とポートを指定します。nsswitch.conf ファイルに指定されている現在のネームサービスが使用されます。認証方法で TLS が指定されていないかぎり、ポートのデフォルト値は 389 です。この場合、デフォルトの LDAP サーバーポート番号は 636 です。

IPv6 アドレスのアドレスとポート番号を指定するための形式は次のとおりです。

```
[ipv6_addr]:port
```

IPv4 アドレスのアドレスとポート番号を指定するには、次の形式を使用します。

ipv4_addr:port

ホスト名を指定する場合は、次の形式を使用します。

host_name:port

-j passwdFile

バインド DN のパスワードまたは SSL クライアントの鍵データベースのパスワードを含むファイルを指定します。パスワードを保護するには、このオプションをスクリプトで使用し、セキュリティ保護されたファイルにパスワードを保存します。

このオプションは、**-w** オプションとは相互排他的です。

-l

検索条件に一致する各エントリのすべての属性を一覧表示します。デフォルトでは、**ldaplist** は、検索されたエントリの識別名のみを一覧表示します。

-M domainName

指定されたサーバーによって提供されるドメインの名前を指定します。このオプションが指定されていない場合は、デフォルトドメイン名が使用されます。

-N profileName

DUAProfile の名前を指定します。**-H** オプションで指定されたサーバー上に、このような名前を持つプロファイルが存在すると想定されます。デフォルト値は **default** です。

-p certifPath

証明書データベースの場所への証明書のパスを指定します。値は、セキュリティデータベースファイルが置かれているパスです。これは、*authenticationMethod* および *serviceAuthenticationMethod* 属性で指定される TLS サポートに使用されます。デフォルトは **/var/ldap** です。

-w bindPassword

bindDN の認証に使用するパスワード。このパラメータがない場合、このコマンドはパスワードの入力を求めます。LDAP では、NULL パスワードはサポートされていません。

認証に使用されるパスワードを指定するために **-w bind_password** を使用すると、システムのほかのユーザーが **ps** コマンドを使用して、スクリプトファイルまたはシェルの履歴でそのパスワードを表示できるようになります。

- の値がパスワードとして指定されている場合、このコマンドはパスワードの入力を求めます。

-v 冗長モードを設定します。ldaplist ユーティリティーは、エントリを検索するために使用されるフィルタも表示します。このフィルタには「+++」の接頭辞が付けられます。

使用例

例1 ホストデータベース内のすべてのエントリの一覧表示

次の例では、hosts データベース内のすべてのエントリを一覧表示します。

```
example% ldaplist hosts
```

例2 非標準データベース ou=new 内のすべてのエントリの一覧表示

次の例では、非標準データベース内のすべてのエントリを一覧表示します。

```
example% ldaplist ou=new
```

例3 passwd データベース内の user1 の検索

次の例では、passwd データベース内の user1 を検索します。

```
example% ldaplist passwd user1
```

例4 services データベース内の 4045 のサービスポートを持つエントリの検索

次の例では、services データベース内の 4045 のサービスポートを持つエントリを検索します。

```
example% ldaplist services ipServicePort=4045
```

例5 passwd データベース内の new で始まるユーザー名を持つすべてのユーザーの検索

次の例では、passwd データベース内の new で始まるユーザー名を持つすべてのユーザーを検索します。

```
example% ldaplist passwd 'new*'
```

例6 hosts データベースの属性の一覧表示

次の例では、hosts データベースの属性を一覧表示します。

```
example% ldaplist -d hosts
```

例7 passwd データベース内の user1 の検索

次の例では、passwd データベース内の user1 を検索します。LDAP サーバーが明示的に指定されています。

```
example% ldaplist -H 10.10.10.10:3890 \  
-M another.domain.name -N special_duaprofile \  
-P password
```

例7 passwd データベース内の user1 の検索 (続き)

```
-D "cn=directory manager" -w secret \
user1
```

終了ステータス 次の終了ステータスが返されます。

- 0 いくつかのエントリに正常に一致しました。
- 1 テーブルを正常に検索しましたが、一致が見つかりませんでした。
- 2 エラーが発生した。エラーメッセージが出力されます。

ファイル

```
/var/ldap/ldap_client_file
/var/ldap/ldap_client_cred
```

クライアントの LDAP 構成を含むファイル。これらのファイルを手動で変更しないでください。その内容は人間が読めるとは限りません。これらのファイルを更新するには、[ldapclient\(1M\)](#) を使用します。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/network/nis
インタフェースの安定性	確実

関連項目

[ldapadd\(1\)](#), [ldapdelete\(1\)](#), [ldapmodify\(1\)](#), [ldapmodrdn\(1\)](#), [ldapsearch\(1\)](#), [idsconfig\(1M\)](#), [ldap_cachemgr\(1M\)](#), [ldapaddent\(1M\)](#), [ldapclient\(1M\)](#), [resolv.conf\(4\)](#), [attributes\(5\)](#), [ldap\(5\)](#)

注意事項

RFC 2307bis は、LDAP をネームサービスとして使用するためのアプローチを定義した、ドラフト段階にある IETF 情報ドキュメントです。

現在 `libldap.so.5` では StartTLS がサポートされていないため、提供されるポート番号は、StartTLS シーケンスの一部として使用されるポートではなく、TLS オープン中に使用されるポートを示します。たとえば、`-h foo:1000 -a tls:simple` は、セキュリティで保護されていないポート 1000 上のオープンな StartTLS シーケンスではなく、ホスト `foo` のポート 1000 上の生の TLS オープンを示します。ポート 1000 がセキュリティで保護されていない場合は、接続が確立されません。

名前 ldapmodify, ldapadd – LDAP エントリ追加および変更ツール

形式 ldapmodify [-a] [-c] [-r] [-n] [-v] [-F] [-b] [-A] [-q]
 [-H] [-?] [-E] [-J] [-Z] [-M] [-d *debuglevel*]
 [-D *bindDN*] [-j *filename*] [-J [:criticality]]
 [-B *baseDN*] [-V *version*] [-Y *proxyDN*] [-O *hopLimit*]
 [-i *locale*] [-k *path*] [-e *errorFile*] [-P *path*]
 [-N *certificate*] [-w *passwd*] [-o *attributename=value*]
 [-h *ldaphost*] [-W *password*] [-p *ldapport*] [-f *file*]
 [-l *nb-ldap-connections*]

ldapadd [-c] [-n] [-v] [-F]
 [[-b] [-A] [-q] [-H] [-?] [-E] [-J] [-Z] [-M] -d *debuglevel*]
 [-D *bindDN*] [-j *filename*] [-B *baseDN*] [-V *version*]
 [-Y *proxyDN*] [-O *hopLimit*] [-i *locale*] [-k *path*]
 [-e *errorFile*] [-P *path*] [-N *certificate*] [-w *passwd*]
 [-o *attributename=value*] [-h *ldaphost*] [-W *password*]
 [-p *ldapport*] [-f *file*] [-l *nb-ldap-connections*]

機能説明 ldapmodify ユーティリティーはLDAP サーバーへの接続を開いてバインドし、エントリを変更または追加します。エントリ情報は標準入力から、または -f オプションを使用して指定された *file* から読み取られます。ldapadd ユーティリティーは、ldapmodify ツールへのハードリンクとして実装されます。ldapadd として呼び出された場合は、-a (新しいエントリの追加) オプションが自動的にオンに設定されます。

ldapadd と ldapmodify はどちらも、同じエントリに対する重複した属性の名前と値のペアを拒否します。

オプション サポートしているオプションは、次のとおりです。

- a 新しいエントリを追加します。デフォルトでは、ldapmodify は既存のエントリを変更します。ldapadd として呼び出された場合は、このオプションが常に設定されます。
- A ASCII 以外のモード: -v オプションと組み合わせて、ASCII 以外の値を表示します。
- b バイナリファイル进行处理します。ldapmodify ツールは、入力内のすべての属性値を走査して、それが有効なファイル参照であるかどうかを判定します。参照が有効な場合は、そのファイルの内容を属性の値として使用します。この

	オプションは、属性のバイナリデータ (JPEG イメージなど) を入力するために使用されます。たとえば、対応する LDIF 入力は <code>jpegPhoto: /tmp/photo.jpg</code> になります。また、<code>ldapmodify</code> ツールは、ファイルの内容を直接含めるための LDIF <code>:<URL</code> の表記もサポートしています。										
<code>-B baseDN</code>	追加を実行するときのベース DN を、通常はシェルの二重引用符 (" ") 内に指定します。すべてのエントリがこの接尾辞の下に配置されるため、一括インポート機能が提供されます。										
<code>-c</code>	継続動作モードを指定します。エラーが報告されても、<code>ldapmodify</code> および <code>ldapadd</code> は変更を続行します。デフォルトでは、エラーを報告したあとに終了します。										
<code>-D bindDN</code>	識別名 <code>bindDN</code> を使用してディレクトリにバインドします。										
<code>-d debuglevel</code>	LDAP のデバッグレベルを設定します。<code>ldapmodify</code> および <code>ldapadd</code> に役立つデバッグレベルは次のとおりです。 <table> <tr> <td>1</td><td>トレース</td></tr> <tr> <td>2</td><td>パケット</td></tr> <tr> <td>4</td><td>引数</td></tr> <tr> <td>32</td><td>フィルタ</td></tr> <tr> <td>128</td><td>アクセス制御</td></tr> </table> 複数のカテゴリのデバッグ情報を要求するには、マスクを追加します。たとえば、トレースとフィルタの情報を要求するには、33 の <code>debuglevel</code> を指定します。	1	トレース	2	パケット	4	引数	32	フィルタ	128	アクセス制御
1	トレース										
2	パケット										
4	引数										
32	フィルタ										
128	アクセス制御										
<code>-e errorFile</code>	入力内の無効な更新文が、デバッグのために <code>errorFile</code> にコ										

	ピーされます。大きな LDIF 入力を処理しているときのエラーを修正するには、 <code>-c</code> オプションとともに使用します。
<code>-E</code>	認証応答制御を使用してバインド識別情報を公開 (報告) するようサーバーに要求します。
<code>-F</code>	<code>replica:</code> で始まる入力行の内容には関係なく、すべての変更を強制的に適用します。デフォルトでは、 <code>replica:</code> 行が使用中の LDAP サーバーホストおよびポートと比較されて、 <code>replog</code> レコードを適用するかどうか判定されます。
<code>-f file</code>	標準入力の代わりに <i>file</i> からエントリ変更情報を読み取ります。
<code>-?</code>	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
<code>-H</code>	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
<code>-h ldaphost</code>	LDAP サーバーが実行されている代替ホストを指定します。
<code>-i locale</code>	<code>-f LDIFfile</code> または標準入力で使用する文字セットを指定します。デフォルトは、 <code>LANG</code> 環境変数で指定された文字セットです。このオプションを使用して、指定した文字セットから UTF8 への変換を実行し、それによって <code>LANG</code> 設定をオーバーライドすることを選択できます。
<code>-j filename</code>	バインド DN 用のパスワードまたは SSL クライアントの鍵データベース用のパスワードを含むファイルを指定します。パスワードを保護するには、このオプションをスクリプトで使用し、セ

	セキュリティ保護されたファイルにパスワードを保存します。このオプションは、<code>-w</code>および<code>-W</code>オプションとは相互に排他的です。
<code>-J [:criticality[: value ::b64value b64value]: fileurl]]</code>	<code>criticality</code> はブール値です (デフォルトは <code>false</code>)。
<code>-k path</code>	変換ルーチンを含むディレクトリへのパスを指定します。これらのルーチンは、ディレクトリサーバーでデフォルトではサポートされていないロケールを指定したい場合に使用されます。これは NLS のサポート用です。
<code>-l nb-ldap-connections</code>	<code>ldapadd</code> または <code>ldapmodify</code> がディレクトリ内の変更を処理するために開く LDAP 接続の数を指定します。デフォルトは 1 つの接続です。
<code>-M</code>	スマートリフェラルを管理します。これらが操作のターゲットである場合は、リフェラルに従って取得されたエントリではなく、リフェラルを含むエントリを変更します。
<code>-n</code>	変更をプレビューしますが、エントリへの変更は行いません。<code>-v</code> および <code>-d</code> と組み合わせるとデバッグに役立ちます。
<code>-N certificate</code>	証明書に基づくクライアント認証で使用する証明書名を指定します。たとえば、<code>-N "Directory-Cert"</code> と指定します。
<code>-o attributename= value</code>	SASL メカニズムや、セキュリティプロパティーなどのその他のオプションの場合、動作モード、認可 ID、認証 IDなどを指定します。 各種の属性名とそれらの値は次のとおりです。

	<code>secProp="number"</code>	SASL セキュリ ティープロ パティーの 定義用。
	<code>realm="value"</code>	SASL レalm を指定しま す(デ フォルトは <code>realm=none</code>)。
	<code>authzid="value"</code>	SASL バイン ドの認可 ID 名を指定し ます。
	<code>authid="value"</code>	SASL バイン ドの認証 ID を指定しま す。
	<code>mech="value"</code>	さまざまな SASL メカニ ズムを指定 します。
<code>-O hopLimit</code>		変更するエントリの検索中に従う べき参照ホップの最大数を指定し ます。デフォルトでは、制限はあ りません。
<code>-p ldapport</code>		セキュリティ保護された LDAP サーバーが待機している代替 TCP ポートを指定します。
<code>-P path</code>		クライアントの証明書データ ベースのパスとファイル名を指定 します。例: <code>-P /home/uid/.netscape/cert7.db</code> このコマンドをディレクトリ サーバーと同じホスト上で使用し ている場合は、そのサーバーの独 自の証明書データベースを使用で きます。例:
<code>-P installDir/lapd-serverID/alias/cert7.db</code>		

	サーバー認証のみを指定するには、<code>-P</code> オプションだけを使用します。
<code>-r</code>	既存の値を指定された値に置き換えます。<code>ldapmodify</code> では、これがデフォルトです。<code>ldapadd</code> が呼び出された場合、または <code>-a</code> オプションが指定されている場合、<code>-r</code> オプションは無視されます。
<code>-v</code>	冗長モードを使用し、診断が標準出力に書き込まれるようにします。
<code>-V version</code>	削除操作に使用される LDAP プロトコルのバージョン番号 (2 または 3 のどちらか) を指定します。LDAP v3 がデフォルトです。v3 をサポートしていないサーバーに接続しているときは、LDAP v2 を指定します。
<code>-W password</code>	<code>-P</code> オプションで指定されたクライアントの鍵データベースのパスワードを指定します。このオプションは、証明書に基づくクライアント認証では必須です。コマンド行で <code>password</code> を指定すると、システム上のほかのユーザーが <code>ps</code> コマンドを使用してそのパスワードを表示できるため、セキュリティの問題が発生します。ファイルからパスワードを指定するには、代わりに <code>-j</code> を使用します。このオプションは、<code>-j</code> とは相互に排他的です。
<code>-w passwd</code>	ディレクトリへの認証のパスワードとして <code>passwd</code> を使用します。認証に使用されるパスワードを指定するために <code>-w passwd</code> を使用すると、システムのほかのユーザーが <code>ps</code> コマンドを使用して、スクリプトファイルまたはシェルの履歴でそのパスワードを

表示できるようになります。このオプションを指定せずに `ldapmodify` コマンドまたは `ldapadd` コマンドを使用した場合、このコマンドはパスワードの入力を求め、それを標準入力から読み取ります。 `-w` オプションを指定せずに使用されると、パスワードがほかのユーザーに表示されなくなります。

- `-Y proxyid` 変更操作で使用するプロキシ DN (プロキシ設定された認可 ID) を、通常はシェルの二重引用符 (" ") 内に指定します。
- `-Z` 証明書に基づくクライアント認証を提供するために SSL を使用することを指定します。このオプションには、 `-N` と SSL パスワード、および証明書と鍵データベースを識別するために必要なその他のすべての SSL オプションが必要です。

終了ステータス 次の終了ステータスが返されます。

- `0` 正常終了。
- `0 以外` エラーが発生した。診断メッセージが標準エラー出力に書き込まれません。

使用例 *file* の内容 (`-f` オプションが指定されていない場合は標準入力) の形式を以降の例に示します。

例1 エントリの変更
ファイル `/tmp/entrymods` には、次の変更指示が含まれています。

```
dn: cn=Modify Me, o=XYZ, c=US
changetype: modify
replace: mail
mail: modme@atlanta.xyz.com
-
add: title
title: System Manager
-
add: jpegPhoto
```

例1 エントリの変更 (続き)

```
jpegPhoto:< file:///tmp/modme.jpeg
-
delete: description
-
```

次のコマンド

```
example% ldapmodify -r -f /tmp/entrymods
```

Modify Me エントリを次のように変更します。

1. mail 属性の現在の値が modme@atlanta.xyz.com の値に置き換えられます。
2. System Manager の値を持つ title 属性が追加されます。
3. ファイル /tmp/modme.jpeg の内容を使用して、jpegPhoto 属性が属性値として追加されます。
4. description 属性が削除されます。

例2 新しいエントリの作成

ファイル /tmp/newentry には、新しいエントリを作成するための次の情報が含まれています。

```
dn: cn=Ann Jones, o=XYZ, c=US
objectClass: person
cn: Ann Jones
cn: Annie Jones
sn: Jones
title: Director of Research and Development
mail: ajones@londonrd.xyz.us.com
uid: ajones
```

コマンド

```
example% ldapadd -f /tmp/newentry
```

このファイル内の情報を使用して、Ann Jones の新しいエントリを追加します。

例3 IPv6 サーバー上の新しいエントリの作成

ファイル /tmp/newentry には、IPv6 サーバー上に新しいエントリを作成するための次の情報が含まれています。

```
dn: cn=Ann Jones, o=XYZ, c=US
objectClass: person
cn: Ann Jones
cn: Annie Jones
sn: Jones
```

例3 IPv6サーバー上の新しいエントリの作成 (続き)

```
title: Director of Research and Development
mail: ajones@londonrd.xyz.us.com
uid: ajones
```

コマンド

```
example% ldapadd -c -v -h '["fec0::111:a00:20ff:feaa:a364"]':389 \
-D cn=Directory Manager -w secret \
-f /tmp/entry
```

このファイル内の情報を使用して、Directory Manager の新しいエントリを追加します。

例4 エントリの削除

ファイル /tmp/badentry には、削除されるエントリに関する次の情報が含まれています。

```
dn: cn=Ann Jones, o=XYZ, c=US
changetype: delete
```

次のコマンド

```
example% ldapmodify -f /tmp/badentry
```

Ann Jones のエントリを削除します。

属性

属性についての詳細は、[attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目

[ldapdelete\(1\)](#), [ldaplist\(1\)](#), [ldapmodrdn\(1\)](#), [ldapsearch\(1\)](#), [ldapaddent\(1M\)](#), [ldap_cachemgr\(1M\)](#), [ldap_get_option\(3LDAP\)](#), [ldap_set_option\(3LDAP\)](#), [attributes\(5\)](#), [ldap\(5\)](#)

名前	ldapmodrdn – LDAP エントリ RDN 変更ツール	
形式	ldapmodrdn [-r] [-n] [-v] [-c] [-E] [-H] [-?] [-M] [-R] [-Z] [-V version] [-d debuglevel] [-D bindDN] [-w passwd] [-h ldaphost] [-i locale] [-j filename] [-J [:criticality]] [-k path] [-N certificate] [-O hopLimit] [-P path] [-W password] [-p ldapport] [-o attributename=value] [-f file] [-Y proxyDN] [dn rdn]	
機能説明	ldapmodrdn は LDAP サーバーへの接続を開いてバインドし、エントリの RDN を変更します。エントリ情報は標準入力から、-f オプションを使用して file から、または dn と rdn のコマンド行のペアから読み取られます。	
オプション	-c	継続動作モード。エラーが報告されても、ldapmodify は変更を続行します。デフォルトでは、エラーを報告したあとに終了します。
	-D bindDN	識別名 binddn を使用してディレクトリにバインドします。
	-d debuglevel	LDAP のデバッグレベルを設定します。ldapmodrdn に役立つ debuglevel の値は次のとおりです。 1 トレース 2 パケット 4 引数 32 フィルタ 128 アクセス制御 複数のカテゴリのデバッグ情報を要求するには、マスクを追加します。たとえば、トレースとフィルタの情報を要求するには、33 の debuglevel を指定します。
	-E	認証応答制御を使用してバインド識別情報を公開 (報告) するようサーバーに要求します。
	-f file	標準入力またはコマンド行の代わりに file からエントリ変更情報を読み取ります。

-?	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
-H	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
-h <i>ldaphost</i>	LDAP サーバーが実行されている代替ホストを指定します。
-i <i>locale</i>	-f <i>LDIFfile</i> または標準入力で使用する文字セットを指定します。デフォルトは、LANG 環境変数で指定された文字セットです。このオプションを使用して、指定した文字セットから UTF8 への変換を実行し、それによって LANG 設定をオーバーライドすることを選択できます。
-J [:criticality[:value][:b64value b64value :fileurl]]	criticality はブール値です (デフォルトは false)。
-j <i>filename</i>	バインド DN 用のパスワードまたは SSL クライアントの鍵データベース用のパスワードを含むファイルを指定します。パスワードを保護するには、このオプションをスクリプトで使い、セキュリティで保護されたファイルにパスワードを保存します。このオプションは、-w および -W オプションとは相互に排他的です。
-k <i>path</i>	変換ルーチンを含むディレクトリへのパスを指定します。これらのルーチンは、ディレクトリサーバーでデフォルトではサポートされていないロケールを指定したい場合に使用されます。これは NLS のサポート用です。
-M	スマートリフェラルを管理します。これらが操作のターゲットである場合は、リフェラルに従って

	取得されたエントリではなく、リ フェラルを含むエントリを変更し ます。
-n	変更をプレビューしますが、エン トリへの変更は行いません。-vお よび -d と組み合わせるとデバッグ に役立ちます。
-N <i>certificate</i>	証明書に基づくクライアント認証 で使用する証明書名を指定しま す。たとえば、-N "Directory-Cert" と指定します。
-n	実行される処理を表示します が、実際にはエントリを変更しま せん。-v と組み合わせるとデ バッグに役立ちます。
-o <i>attributename=value</i>	SASL メカニズムや、セキュリ ティープロパティーなどのその他 のオプションの場合、動作 モード、認可 ID、認証 ID などを 指定します。
	各種の属性名とそれらの値は次の とおりです。
<i>secProp="number"</i>	SASL セ キュリ ティープロ パティーの 定義用。
<i>realm="value"</i>	SASL レルム を指定しま す (デフォル トは <i>realm=none</i>)。
<i>authzid="value"</i>	SASL バイン ドの認可 ID 名を指定し ます。

	<code>authid="value"</code>	SASL バイン ドの認証 ID を指定しま す。
	<code>mech="value"</code>	さまざまな SASL メカニ ズムを指定 します。
<code>-O hopLimit</code>		変更するエントリの検索中に従う べき参照ホップの最大数を指定し ます。デフォルトでは、制限はあ りません。
<code>-P path</code>		クライアントの証明書データ ベースのパスとファイル名を指定 します。例: <code>-P /home/uid/.netscape/cert7.db</code> このコマンドをディレクトリ サーバーと同じホスト上で使用し ている場合は、そのサーバーの独 自の証明書データベースを使用で きます。例:
<code>-P installDir/ldap-serverID/alias/cert7.db</code>		サーバー認証のみを指定するに は、 <code>-P</code> オプションだけを使用しま す。
<code>-p ldapport</code>		セキュリティー保護された LDAP サーバーが待機している代替 TCP ポートを指定します。
<code>-R</code>		検索中に、返されたリフェラルに 自動的に従いません。
<code>-r</code>		古い RDN 値をエントリから削除し ます。デフォルトでは、古い値が 保持されます。
<code>-V version</code>		削除操作に使用される LDAP プロ トコルのバージョン番号 (2 または 3 のどちらか) を指定しま す。LDAP v3 がデフォルトで す。v3 をサポートしていない

	サーバーに接続しているときは、LDAP v2 を指定します。
-v	冗長モードを使用し、診断が標準出力に書き込まれるようにします。
-W <i>password</i>	-P オプションで指定されたクライアントの鍵データベースのパスワードを指定します。このオプションは、証明書に基づくクライアント認証では必須です。コマンド行で <i>password</i> を指定すると、システム上のほかのユーザーが ps コマンドを使用してそのパスワードを表示できるため、セキュリティの問題が発生します。ファイルからパスワードを指定するには、代わりに -j を使用します。このオプションは、 -j とは相互に排他的です。
-w <i>passwd</i>	ディレクトリへの認証のパスワードとして <i>passwd</i> を使用します。認証に使用されるパスワードを指定するために -w <i>passwd</i> を使用すると、システムのほかのユーザーが ps コマンドを使用して、スクリプトファイルまたはシェルの履歴でそのパスワードを表示できるようになります。このオプションを指定せずに ldapmodrtn コマンドを使用した場合、このコマンドはパスワードの入力を求め、それを標準入力から読み取ります。 -w オプションを指定せずに使用されると、パスワードがほかのユーザーに表示されなくなります。
-Y <i>proxyid</i>	変更操作で使用するプロキシ DN (プロキシ設定された認可 ID) を、通常はシェルの二重引用符 ("") 内に指定します。

	-Z 証明書に基づくクライアント認証を提供するためにSSLを使用することを指定します。このオプションには、-NとSSLパスワード、および証明書と鍵データベースを識別するために必要なその他のすべてのSSLオプションが必要です。						
入力形式	コマンド行引数 <i>dn</i> と <i>rdn</i> が指定されている場合は、<i>rdn</i> によって、DN <i>dn</i> で指定されているエントリのRDNが置き換えられます。 それ以外の場合は、<i>file</i> の内容 (-f オプションが指定されていない場合は標準入力) が1つ以上の行のペアで構成されている必要があります。 Distinguished Name (DN) Relative Distinguished Name (RDN) DN/RDN の各ペアを区切るには、1行以上の空行を使用します。						
使用例	ファイル /tmp/entrymods には、次の内容が含まれています。 cn=Modify Me, o=XYZ, c=US cn=The New Me 次のコマンド example% ldapmodify -r -f /tmp/entrymods 「Modify Me」 エントリの RDN を「Modify Me」 から「The New Me」に変更し、古い cn である「Modify Me」が削除されます。						
属性	属性についての詳細は、attributes(5) を参照してください。						
	<table><tr><th>属性タイプ</th><th>属性値</th></tr><tr><td>使用条件</td><td>system/core-os</td></tr><tr><td>インタフェースの安定性</td><td>確実</td></tr></table>	属性タイプ	属性値	使用条件	system/core-os	インタフェースの安定性	確実
属性タイプ	属性値						
使用条件	system/core-os						
インタフェースの安定性	確実						
関連項目	ldapadd(1), ldapdelete(1), ldapmodify(1), ldapsearch(1), attributes(5)						
診断	エラーが発生しなかった場合、終了ステータスは0です。エラーが発生すると0以外の終了ステータスが生成され、診断メッセージが標準エラーに書き込まれます。						

名前	ldapsearch - LDAP 検索ツール
形式	<pre> ldapsearch [-n] [-u] [-v] [-t] [-A] [-B] [-L] [-R] [-H] [-?] [-t] [-T] [-B] [-E] [-J] [-e] [-l] [-Z] [-r] [-M] [-d <i>debuglevel</i>] [-F <i>sep</i>] [-f <i>file</i>] [-D <i>bindDN</i>] [-j <i>filename</i>] [-V <i>version</i>] [-Y <i>proxyDN</i>] [-O <i>hopLimit</i>] [-i <i>locale</i>] [-k <i>path</i>] [-S [-] <i>attribute</i>] [-C <i>pattern</i>] [-c <i>authzid</i>] [-P <i>path</i>] [-N <i>certificate</i>] [-w <i>passwd</i>] [-h <i>ldaphost</i>] [-p <i>ldapport</i>] [-o <i>attributename=value</i>] [-b <i>searchbase</i>] [-s <i>scope</i>] [-a <i>deref</i>] [-l <i>timelimit</i>] [-z <i>sizelimit</i>] <i>filter</i> [<i>attrs</i>]...</pre>
機能説明	ldapsearch ユーティリティーは LDAP サーバーへの接続を開いてバインドし、フィルタ <i>filter</i> を使用して検索を実行します。 ldapsearch で 1 つ以上のエントリが検索された場合は、<i>attrs</i> で指定された属性が取得され、エントリと値が標準出力に表示されます。<i>attrs</i> が指定されていない場合は、すべての属性が返されます。
出力形式	1 つ以上のエントリが検索された場合、各エントリは、次の形式で標準出力に書き込まれます。 <pre> dn: Distinguished Name (DN) attributename: value attributename: value attributename: value ...</pre> 複数のエントリは、1 行の空行で区切られます。別の区切り文字を指定するために <i>-F</i> オプションが使用されている場合は、この文字が : 文字の代わりに使用されます。<i>-t</i> オプションが使用されている場合は、実際の値の代わりに一時ファイルの名前が返されます。<i>-A</i> オプションが指定されている場合は、「<i>attributename</i>」のみが返され、属性値は返されません。
オプション	サポートしているオプションは、次のとおりです。 <i>-A</i> 属性のみを取得します (値はなし)。これは、あるエントリ内に属性が存在するかどうかだけを確認する必要があり、具体的な値には関心がない場合に役立ちます。 <i>-a deref</i> 別名の間接参照の実行方法を指定します。<i>deref</i> の指定可能な値は、それぞれ、別名が間接参照されないこと、常に間接参照されること、検索時に間接参照されること、検索のための基本オブジェクトの検索時にのみ間接参照される

	ことを指定する never、always、search、findで す。デフォルトでは、別名は間接 参照されません。										
-B	ASCII 以外の値を表示し、LDIF 以 外の古い形式を使用します。この オプションを指定すると、デ フォルトの -L オプションが無効に なります。										
-b <i>searchbase</i>	デフォルト値の代わり に、 <i>searchbase</i> を検索の開始位置と して使用します。										
-C <i>pattern</i>	永続検索。接続を開いたまま保持 する検索を実行し、検索の範囲と フィルタに一致するエントリが追 加、変更、または削除された場合 は常に結果を表示します。このオ プションを指定する と、ldapsearch ツールは無期限に 動作します。停止するには、Ctrl + C キーを入力する必要があります。 パターン形式は次のとおり です。										
<code>ps:changeType[:changesOnly[:entryChangeControls]]</code>											
-c <i>authzid</i>	getEffectiveRights コントロール の <i>authzid</i> を指定します。例: <code>dn:uid=bjensen,dc=example,dc=com</code>										
-D <i>bindDN</i>	識別名 <i>bindDN</i> を使用してディレク トリにバインドします。										
-d <i>debuglevel</i>	LDAP のデバッグレベルを設定し ます。ldapsearch に役立つデ バッグレベルは次のとおりです。 <table><tr><td>1</td><td>トレース</td></tr><tr><td>2</td><td>パケット</td></tr><tr><td>4</td><td>引数</td></tr><tr><td>32</td><td>フィルタ</td></tr><tr><td>128</td><td>アクセス制御</td></tr></table>	1	トレース	2	パケット	4	引数	32	フィルタ	128	アクセス制御
1	トレース										
2	パケット										
4	引数										
32	フィルタ										
128	アクセス制御										

	複数のカテゴリのデバッグ情報を要求するには、マスクを追加します。たとえば、トレースとフィルタの情報を要求するには、33 の <code>debuglevel</code> を指定します。
-E	認証応答制御を使用してバインド識別情報を公開 (報告) するようサーバーに要求します。
-e	値の Base 64 エンコーディングを最小化します。
-F <i>sep</i>	<i>sep</i> を属性名と値の間のフィールド区切り文字として使用します。このオプションが指定されている場合、-L オプションは無視されます。
-f <i>file</i>	<i>file</i> から一連の行を読み取り、行ごとに1つのLDAP検索を実行します。この場合、コマンド行で指定された <i>filter</i> はパターンとして扱われ、最初に検出された %s が <i>file</i> の行に置き換えられます。 <i>file</i> が1つの - 文字である場合は、各行が標準入力から読み取られます。
-G <i>pattern</i>	仮想リスト表示。検索ターゲットのインデックスまたは値と、ターゲットの前後に返されるエントリの数によって決定される、すべての結果のうちの一部のみを取得します。このオプションは、サーバー上のソート順序を指定するために、常に -s および -x オプションが必要です。
-?	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。
-H	すべてのオプションを簡単に説明した使用法のヘルプテキストを表示します。

`-h ldaphost`

セキュリティー保護された LDAP サーバーが実行されている代替ホストを指定します。

`-i locale`

コマンド行入力で使用する文字セットを指定します。デフォルトは、LANG 環境変数で指定された文字セットです。このオプションを使用して、指定した文字セットから UTF8 への変換を実行し、それによって LANG 設定をオーバーライドすることもできます。この引数を使用すると、バインド

DN、ベース DN、および検索フィルタパターンを指定された文字セットに入力できま

す。ldapsearch ツールは、検索要求を処理する前に、これらの引数からの入力を変換します。たとえば、`-i no` は、バインド

DN、ベース DN、および検索フィルタがノルウェー語で提供されていることを示します。この引数は、コマンド行入力にのみ影響を与えます。検索フィルタを含むファイルを指定した場合 (`-f` オプションを使用)、ldapsearch は、そのファイル内のデータを変換しません。

`-j filename`

バインド DN 用のパスワードまたは SSL クライアントの鍵データベース用のパスワードを含むファイルを指定します。パスワードを保護するには、このオプションをスクリプトで使用し、セキュリティー保護されたファイルにパスワードを保存します。このオプションは、`-w` および `-W` オプションとは相互に排他的です。

`-J [:criticality[:value|:b64value|b64value|:fileurl]]`

criticality はブール値です (デフォルトは false)。

<code>-k path</code>	変換ルーチンを含むディレクトリへのパスを指定します。これらのルーチンは、ディレクトリサーバーでデフォルトではサポートされていないロケールを指定したい場合に使用されます。これは NLS のサポート用です。
<code>-L</code>	検索結果を LDIF 形式で表示します。このオプションを指定すると、 <code>-B</code> オプションもオンに設定されます。この動作がデフォルトです。
<code>-l timelimit</code>	検索が完了するまで、最大 <i>timelimit</i> 秒待ちます。
<code>-M</code>	スマートリフェラルを管理します。これらが操作のターゲットである場合は、リフェラルに従って取得されたエントリではなく、リフェラルを含むエントリを検索します。
<code>-N certificate</code>	証明書に基づくクライアント認証で使用する証明書名を指定します。たとえば、 <code>-N "Directory-Cert"</code> と指定します。
<code>-n</code>	実行される処理を表示しますが、実際には検索を実行しません。 <code>-v</code> および <code>-d</code> と組み合わせるとデバッグに役立ちます。
<code>-O hopLimit</code>	変更するエントリの検索中に従うべき参照ホップの最大数を指定します。デフォルトでは、制限はありません。
<code>-o attributename=value</code>	SASL メカニズムや、セキュリティプロパティーなどのその他のオプションの場合、動作モード、認可 ID、認証 IDなどを指定します。
	各種の属性名とそれらの値は次のとおりです。

	<code>secProp="number"</code>	SASL セキュリティプロパティの定義用。
	<code>realm="value"</code>	SASL レalm を指定します (デフォルトは <code>realm=none</code>)。
	<code>authzid="value"</code>	SASL バインドの認可 ID 名を指定します。
	<code>authid="value"</code>	SASL バインドの認証 ID を指定します。
	<code>mech="value"</code>	さまざまな SASL メカニズムを指定します。
<code>-P path</code>	クライアントの証明書データベースのパスとファイル名を指定します。例: <code>-P /home/uid/.netscape/cert7.db</code> このコマンドをディレクトリサーバーと同じホスト上で使用している場合は、そのサーバーの独自の証明書データベースを使用できます。例:	
<code>-P installDir/ldap-serverID/alias/cert7.db</code>	サーバー認証のみを指定するには、 <code>-P</code> オプションだけを使用します。	
<code>-p ldapport</code>	セキュリティー保護された LDAP サーバーが待機している代替 TCP ポートを指定します。	

-R	検索中に、返されたりフェラルに自動的に従いません。
-r	ldapsearch コマンドの出力を古い形式で表示します。
-S [-]attribute	検索によって返されたエントリをソートするための属性を指定します。ソート条件は、属性の値に関してアルファベット順、または -attribute の形式を指定した場合は逆アルファベット順です。ソートをより詳細に指定するために複数の -s オプションを指定できます。例: -S sn -S givenname デフォルトでは、エントリはソートされません。サーバー側のソートを実行するには、-x オプションを使用します。
-s scope	検索の範囲を指定します。scope の指定可能な値は、それぞれ、基本オブジェクト、1 レベル、またはサブツリーの検索を指定する base、one、sub です。デフォルトは sub です。
-T	個々の属性値内で行ブレークが使用されないように、検索結果の出力を書式設定します。
-t	取得された値を一連の一時ファイルに書き込みます。これは、jpegPhoto やオーディオなどの ASCII 以外の値を処理する場合に役立ちます。
-U	URL 形式 (-t オプションでのみ有効)。一時ファイルの出力を使用している場合、このツールの標準出力には、属性の値の代わりにファイルの URL が含まれます。例:

```
jpegPhoto:< file:/tmp/ldapsearch-jpegPhoto-YzaOMh
```

-u	出力にわかりやすい形式の識別名 (DN) を含めます。
-V <i>version</i>	削除操作に使用される LDAP プロトコルのバージョン番号 (2 または 3 のどちらか) を指定します。LDAP v3 がデフォルトです。v3 をサポートしていないサーバーに接続しているときは、LDAP v2 を指定します。
-v	冗長モードで動作し、診断が標準出力に書き込まれるようにします。
-W <i>password</i>	-P オプションで指定されたクライアントの鍵データベースのパスワードを指定します。このオプションは、証明書に基づくクライアント認証では必須です。コマンド行で <i>password</i> を指定すると、システム上のほかのユーザーが ps コマンドを使用してそのパスワードを表示できるため、セキュリティの問題が発生します。ファイルからパスワードを指定するには、代わりに -j を使用します。このオプションは、-j とは相互に排他的です。
-w <i>passwd</i>	ディレクトリへの認証のパスワードとして <i>passwd</i> を使用します。認証に使用されるパスワードを指定するために -w <i>passwd</i> を使用すると、システムのほかのユーザーが ps コマンドを使用して、スクリプトファイルまたはシェルの履歴でそのパスワードを表示できるようになります。このオプションを指定せずに ldapsearch コマンドを使用した場合、このコマンドはパスワードの入力を求め、それを標準入力から読み取ります。-w オプションを指定せずに使用されると、パス

-x	ワードがほかのユーザーに表示されなくなります。
-Y proxyDN	検索結果を、クライアント上で実行されている <code>ldapsearch</code> コマンドによってではなく、サーバー上でソートされるように指定するには、 <code>-s</code> オプションとともに使用します。これは、国際的な検索のように、マッチングルールに従ってソートしたい場合に役立ちます。サーバー上のソートがサポートされている場合は、通常、クライアント上のソートより高速です。
-Z	変更操作で使用するプロキシ DN (プロキシ設定された認可 ID) を、通常はシェルの二重引用符 ("") 内に指定します。
-z sizelimit	証明書に基づくクライアント認証を提供するために SSL を使用することを指定します。このオプションには、 <code>-N</code> と SSL パスワード、および証明書と鍵データベースを識別するために必要なその他のすべての SSL オプションが必要です。
	検索が完了するまで、最大 <code>sizelimit</code> エントリを取得します。

使用例

例1 サブツリーの検索の実行

次のコマンドは、「`mark smith`」の `commonName` を持つエントリに対する (デフォルト検索ベースを使用した) サブツリーの検索を実行します。 `commonName` 値と `telephoneNumber` 値が取得されて標準出力に表示されます。この出力を古い形式で表示するには、`-r` オプションを使用します。

```
example% ldapsearch "cn=mark smith" cn telephoneNumber
```

出力は次のようになります。

```
dn: Mark D Smith, ou=Sales, ou=Atlanta, ou=People, o=XYZ, c=US
cn: Mark Smith
cn: Mark David Smith
cn: Mark D Smith 1
```

例1 サブツリーの検索の実行 (続き)

```

cn: Mark D Smith
telephoneNumber: +1 123 456-7890

dn: Mark C Smith, ou=Distribution, ou=Atlanta, ou=People, o=XYZ, c=US
cn: Mark Smith
cn: Mark C Smith 1
cn: Mark C Smith
telephoneNumber: +1 123 456-9999

```

例2 デフォルト検索ベースを使用したサブツリーの検索の実行

次のコマンドは、mcs のユーザー ID を持つエントリに対するデフォルト検索ベースを使用した `-r` オプションによるサブツリーの検索を実行して、古い形式で表示します。各エントリのわかりやすい形式の DN がその DN 自体を含む行のあとに出力され、`jpegPhoto` 値と `audio` 値が取得されて一時ファイルに書き込まれます。

```
ldapsearch -r -u -t "uid=mcs" -r jpegPhoto audio
```

要求された各属性の 1 つ値を持つ 1 つのエントリが検索された場合、出力は次のようになります。

```

cn=Mark C Smith, ou=Distribution, ou=Atlanta, ou=People, o=XYZ, c=US
Mark C Smith, Distribution, Atlanta, People, XYZ, US
audio=/tmp/ldapsearch-audio-a19924
jpegPhoto=/tmp/ldapsearch-jpegPhoto-a19924

```

例3 1レベルの検索の実行

次のコマンドは、`organizationName` が XY で始まるすべての組織に対する `c=US` レベルでの 1 レベルの検索を実行します。

```
example% ldapsearch -s one -b "c=US" "o=XY*" o description
```

`organizationName` と `description` の属性値が取得されて標準出力に表示され、次のような出力が生成されます。

```

dn: o=XYZ      c=US
   o: XYZ
   description: XYZ Corporation

dn: o="XY Trading Company", c=US
   o: XY Trading Company
   description: Import and export specialists

dn: o=XYInternational, c=US
   o: XYInternational
   o: XYI

```

例 3 1 レベルの検索の実行 (続き)

o: XY International

例 4 IPv6 サーバー上でのサブツリーの検索の実行

次のコマンドは、IPv6 (つまり、-h) サーバー上での mcs のユーザー ID を持つエントリに対するデフォルト検索ベースを使用したサブツリーの検索を実行します。

```
example% ldapsearch -u -h '[fec0::111:a00:20ff:fea3:edcf]' \  
-t "uid=mcs" jpegPhoto audio
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。診断メッセージが標準エラー出力に書き込まれます。

属性 属性についての詳細は、[attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [ldapadd\(1\)](#), [ldapdelete\(1\)](#), [ldapmodify\(1\)](#), [ldapmodrdn\(1\)](#), [attributes\(5\)](#)

名前	ldd - 実行可能ファイルまたは共有オブジェクトの動的依存関係の表示
形式	<code>ldd [-d -r] [-c] [-D] [-e <i>envvar</i>] [-f] [-i] [-L] [-l] [-p] [-s] [-U -u] [-v] [-w] <i>filename</i>...</code>
機能説明	ldd ユーティリティーは、実行可能ファイルまたは共有オブジェクトの動的依存関係を表示します。ldd は実行時リンカー ld.so.1 を使用して診断情報を生成します。実行時リンカーは検査対象のオブジェクトを取得して、実行中のプロセスで使用されるのと同様に処理します。デフォルトでは、ldd は、すべてのレイジーな依存関係、および遅延依存関係の読み込みをトリガーします。 ldd は <i>filename</i> の読み込み時に読み込まれるすべての共有オブジェクトのパス名を表示します。ldd は、検査対象の共有オブジェクトに実行権が与えられているものとみなします。実行権が与えられていない場合、ldd は警告を出力してからファイルを処理します。 ldd はファイルを 1 つずつ入力処理します。入力ファイルごとに、ldd は次のいずれかを実行します。 ■ オブジェクトの依存関係が存在する場合はその依存関係を表示する ■ 依存関係が存在しない場合はそのまま続行する ■ 正常に処理できなかった場合はエラーメッセージを出力する ldd により検査される動的オブジェクトは実行されません。このため、ldd を実行しても、dlopen(3C) を使って明示的に追加された共有オブジェクトは一切表示されません。プロセスまたはコアファイルにより使用されているオブジェクトをすべて表示するには、pldd(1) を使用します。
オプション	ldd は、<i>filename</i> とそこで使用される共有オブジェクト間の互換性を調べることもできます。次のオプションを指定すると、ldd は <i>filename</i> の実行時に発生する未解決のシンボル参照について、警告を出力します。 -d 即時参照をチェックします。 -r 即時参照およびレイジー参照の両方をチェックします。 1 回の ldd 呼び出しでは、前述のオプションのいずれか一方だけしか指定できません。-- 即時参照の対象は通常、実行可能ファイルまたは共有オブジェクトコードが使用するデータ項目です。関数へのポインタ、そして、位置に依存する共有オブジェクトからの関数呼び出しです。レイジー参照は通常、位置に依存しない共有オブジェクトからのグローバル関数の呼び出し、または実行可能ファイルからの外部関数呼び出しです。各参照タイプの詳細は、『リンカーとライブラリ』の「再配置処理の実行に関する説明」を参照してください。オブジェクトの読み込みは、再配置処理の影響も受けます。詳細は、「使用法」の「レイジー読み込み」を参照してください。

デフォルトでは、一部の未解決のシンボル参照は報告されません。次のオプションを使用すると、未解決の参照を報告できます。これらのオプションは、`-d` オプションまたは `-r` オプションと組み合わせてはじめて効果を発揮します。

- p 明示的な親参照および外部参照に対する未解決のシンボルエラーを公開します。
- w 未解決の弱いシンボル参照を公開します。

共有オブジェクトでは、共有オブジェクトの呼び出し元によって指定されるシンボルを参照できます。これらの参照は、共有オブジェクトの作成時に、親から使用可能、または単純に外部であることを明示的に分類できます。[ld\(1\)](#) の `-Mmapfile` オプションと、`PARENT` および `EXTERN` シンボル定義キーワードを参照してください。動的実行可能ファイルの検査時に、解決できない親参照または外部参照には、エラーとしてフラグが設定されます。ただし、デフォルトでは、共有オブジェクトの検査時に、解決できない親参照または外部参照には、エラーとしてフラグが設定されません。`-p` オプションを `-d` オプションまたは `-r` オプションと組み合わせて使用すると、未解決の親参照または外部参照に、再配置エラーとしてフラグが設定されます。

再配置で使用するシンボルを弱い参照として定義できます。デフォルトでは、弱いシンボル参照が解決できない場合、再配置は無視され、再配置オフセットにゼロが書き込まれます。`-w` オプションを `-d` オプションまたは `-r` オプションと組み合わせて使用すると、弱いシンボル参照に対する未解決の再配置に、再配置エラーとしてフラグが設定されます。

`ldd` は依存関係についても確認できます。次に示す各オプションを使用すると、`ldd` は、*filename* が読み込まれたときに、未参照または未使用の依存関係が読み込まれた場合に、それらに対する警告を出力します。1つのシンボル参照が1つの依存関係にバインドされているときのみ、その依存関係が使用されていると考えられます。したがって、これらのオプションはシンボル参照が検査されているときのみ有効です。`-r` オプションが有効でない場合、`-d` が有効になります。

オブジェクトにより定義されているが、オブジェクトからバインドされていない依存関係は、未参照の依存関係です。*filename* が読み込まれたときにその他のオブジェクトによりバインドされていない依存関係は、未使用のオブジェクトです。

依存関係は、デフォルトのシステム上の場所か、検索パスで指定された場所で検出できます。検索パスは、環境変数 `LD_LIBRARY_PATH` など、グローバルに指定できます。検索パスは、動的オブジェクトで実行パスとして定義することもできます。[ld\(1\)](#) の `-R` オプションを参照してください。依存関係を満たすために使用されない検索パスは、不要なファイルシステム処理を発生させる原因となります。

- U 未参照または未使用の依存関係を表示します。未参照の依存関係が、*filename* で読み込まれたほかのオブジェクトによりバインドされていない場合、その

依存関係は未使用としても設定されます。巡回の外部にあるオブジェクトにバインドされていない巡回依存関係は、未参照ともみなされます。

このオプションは、未使用の検索パスも表示します。

-u 未使用のオブジェクトを表示します。

-u は **-u** の機能をすべて含んでいますが、**-u** または **-u** オプションのどちらか 1 つだけを **ldd** の 1 回の呼び出しの間に指定できます。**-r** オプションを使用するとき未参照または未使用であると判明したオブジェクトは、依存関係として削除する必要があります。これらのオブジェクトは参照を提供しませんが、*filename* が読み込まれるとき、不必要なオーバーヘッドを生みます。**-d** オプションを使用する場合、未参照または未使用と判明したオブジェクトは *filename* が読み込まれるときにすぐには要求されません。これらのオブジェクトはレイジー読み込みの候補です。詳細は、「使用法」の「レイジー読み込み」を参照してください。

未使用の依存関係を削除すると、実行時リンクのオーバーヘッドを下げるができます。未参照の依存関係を削除すると、実行時リンクのオーバーヘッドを下げるができますが、その効果はより少なくなります。ただし、未参照の依存関係の削除は、異なるオブジェクトと組み合わせられたときや、その他のオブジェクトの依存関係が変化するに伴って、依存関係が使用されなくなることを防ぐ役割を果たします。

未使用の検索パスを削除することで、依存関係の検出に必要な処理を削減できます。これは、ネットワーク経由でファイルサーバーのファイルにアクセスする場合に有効です。検索パスは **dlopen(3C)** の要件を満たすために、オブジェクト内に符号化されている場合があります。この検索パスは、このオブジェクトの依存関係を取得するためには必要がない場合があるため、**ldd** では未使用とみなされます。

次のオプションを使用できます。

-c 一切の構成ファイルの使用を禁止します。構成ファイルは、デフォルトの検索パスの変更、ディレクトリキャッシュの提供、および代替オブジェクト依存関係の提供に利用できます。**crle(1)** を参照してください。

-D 遅延依存関係の読み込みをスキップします。デフォルトでは、**ldd** は、レイジーな依存関係と遅延依存関係の両方の処理を強制します。**-L** オプションも参照してください。通常のプロセス実行時は、遅延依存関係が読み込まれるのは、遅延参照への最初の実行時結合が行われるときだけです。**-D** オプションを使用すると、**-d** オプションまたは **-r** オプションを使用しても、遅延依存関係の読み込みはトリガーされません。**ld(1)** の **-z deferred** オプションを参照してください。

-e *envvar* 環境変数 *envvar* を設定します。

このオプションは、LD_PRELOAD などの、ldd そのものに悪影響を与える可能性のある実行時リンカー環境変数を試す場合に便利です。

このオプションは、LD_DEBUG などの追加情報だけを検査中のオブジェクトから抽出する場合にも役立ちます。ld.so.1(1) と lari(1) を参照してください。

- f 安全でない実行可能ファイルの検査を強制します。スーパーユーザーが呼び出した場合、ldd はデフォルトで、安全でないことが判明した実行可能ファイルを一切処理しません。実行可能ファイルは、指定されたインタプリタが /lib または /usr/lib にない場合、安全でないとみなされます。インタプリタが判別できない場合にも、実行可能ファイルは安全でないとみなされます。「使用法」の「セキュリティー」を参照してください。
- i 初期化セクションの実行順序を表示します。表示される順序は、-d または -r オプションの使用によって変わります。「使用法」の「初期化順序」を参照してください。
- L レイジー読み込みを有効にします。デフォルトでは、ldd は、レイジーな依存関係と遅延依存関係の両方の処理を強制します。-D オプションも参照してください。通常のプロセス実行時は、レイジー読み込みがデフォルトの動作モードです。この場合、レイジーオブジェクト内で定義されているシンボルが参照されたときに、レイジーな依存関係またはフィルタだけがプロセスに読み込まれます。-L オプションとともに -d または -r オプションを使用すると、実行プロセスで発生する依存関係および読み込み順序を調べることができます。ld(1) の -z lazyload オプションを参照してください。
- l あらゆるフィルタを強制的に即時処理します (すべてのフィルタとその依存関係を表示するため)。フィルタの即時処理は現在、ldd のデフォルトの動作モードです。ただし、このデフォルトモードでは、検出されなかった外部フィルタはそのまま無視されます。-l オプションを使用すると、検出されなかった外部フィルタに起因するエラーメッセージが生成されます。
- s 共有オブジェクトの依存関係を調べるのに使用した検索パスを表示します。
- v *filename* の処理中に発生したすべての依存関係を表示します。このオプションを指定すると、依存関係が要求するバージョン情報も表示します。pvs(1) を参照してください。

使用法

セキュリティー スーパーユーザーは、検査対象の実行可能ファイルが信頼できると判明している場合に限り、-f オプションを使用します。信頼できない実行可能ファイルに -f オプ

ションを使用すると、システムのセキュリティが損なわれる可能性があります。検査対象の実行ファイルが信頼できるものかどうか不明な場合、スーパーユーザーは一時的に一般ユーザーになり、一般ユーザーとして `ldd` を呼び出す必要があります。

`:r` サブコマンドを使用しないかぎり、`dump(1)`、`elfdump(1)`、`elfedit(1)`、および `mdb(1)` を使用して、信頼できないオブジェクトを安全に検査できます。また、スーパーユーザー以外のユーザーは `mdb` の `:r` サブコマンドまたは `truss(1)` を使用することによって、それほどリスクを冒さずに信頼できない実行可能ファイルを検査できます。信頼できない実行可能ファイルで `ldd`、`mdb :r`、または `truss` を使用する場合は、リスクを最小限に抑えるために UID "nobody" を使用してください。

レイジー読み込み

レイジー読み込みを直接的に適用するには、レイジーな依存関係を指定します。`ld(1)` の `-z lazyload` オプションを参照してください。レイジー読み込みを間接的に適用するには、フィルタを指定します。`ld(1)` の `-f` オプションと `-F` オプションを参照してください。レイジー読み込み技法をオブジェクトに適用する場合、使用するオプションによっては、`ldd` の出力が異なることがあります。すべての依存関係がレイジーであるとされたオブジェクトの場合、`ldd` のデフォルトの動作により、すべての依存関係はそのオブジェクトで記録される順に出力されます。

```
example% ldd main
        libelf.so.1 => /lib/libelf.so.1
        libnsl.so.1 => /lib/libnsl.so.1
        libc.so.1 => /lib/libc.so.1
```

実行時にこのオブジェクトを使用した場合のレイジー読み込み動作は、`-L` オプションを使用することによって有効にできます。このモードの場合、遅延依存関係が読み込まれるのは、レイジーオブジェクト内で定義されているシンボルが参照されたときです。したがって、`-L` オプションを `-d` および `-r` オプションと組み合わせると、即時参照とレイジー参照のそれぞれを満たすのに必要な依存関係を調べることができます。

```
example% ldd -L main
example% ldd -d main
        libc.so.1 => /lib/libc.so.1
example% ldd -r main
        libc.so.1 => /lib/libc.so.1
        libelf.so.1 => /lib/libelf.so.1
```

この例の場合、出力される依存関係の順序は、オプションなしで実行した `ldd` の場合と異なります。また、`-r` オプションを使用した場合とも異なります。レイジー依存関係に対する参照は、実行中のプログラムと同じ順序では発生しません。

レイジー読み込みを調べると、参照を満たす必要のないオブジェクトも明らかになります。このようなオブジェクト(前述の例では `libnsl.so.1`)は、検査対象のオブジェクトを作成するために使用したリンク行から削除可能な候補です。

初期化順序

必要な依存関係が明示的に定義されていないオブジェクトでは、使用するオプションによって、lddによって表示される初期化セクションの順序が異なる場合があります。次に、簡単な適用例を示します。

```
example% ldd -i main
      libA.so.1 =>    ./libA.so.1
      libc.so.1 =>   /lib/libc.so.1
      libB.so.1 =>    ./libB.so.1

      init object=./libB.so.1
      init object=./libA.so.1
      init object=/lib/libc.so.1
```

再配置が適用されると、初期化セクションの順序は次のようになります。

```
example% ldd -ir main
      .....

      init object=/lib/libc.so.1
      init object=./libB.so.1
      init object=./libA.so.1
```

この場合、libB.so.1は/usr/lib/libc.so.1の関数を参照します。ただし、このライブラリに明示的な依存関係はありません。再配置が検出されてはじめて、依存関係が確立され、その結果、この暗黙的な依存関係により初期化セクションのソート順序が影響を受けます。

通常、アプリケーションの実行時に設定される初期化セクションのソート順序は、lddに-dオプションを指定した場合と同等です。最適な順序が得られるのは、すべてのオブジェクトでそれぞれの依存関係が完全に定義されている場合です。動的オブジェクトの作成が望ましい場合は、ld(1)でオプション-z defsおよび-z ignoreを使用します。

1つ以上の動的オブジェクトが相互に参照する場合には、循環型依存関係が生じる可能性があります。循環型依存関係は、各依存関係固有のソート順序を確立できないので、避けなければなりません。

オブジェクトファイルを静的に分析する方が望ましい場合は、dump(1)やelfdump(1)などのツールを使用して依存関係を調べることができます。

ファイル

/usr/lib/lddstub	共有オブジェクトの依存関係を検査するために読み込まれた32ビット擬似実行可能ファイル
/usr/lib/64/lddstub	共有オブジェクトの依存関係を検査するために読み込まれた64ビット擬似実行可能ファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/linker

関連項目 [crle\(1\)](#), [dump\(1\)](#), [elfdump\(1\)](#), [elfedit\(1\)](#), [lari\(1\)](#), [ld\(1\)](#), [ld.so.1\(1\)](#), [mdb\(1\)](#), [pldd\(1\)](#), [pvs\(1\)](#), [truss\(1\)](#), [dlopen\(3C\)](#), [attributes\(5\)](#)

[『リンカーとライブラリ』](#)

診断 `ldd` は共有オブジェクトのパス名の記録を標準出力に書き込みます。シンボルの解決問題に関する任意選択のリストは、標準エラー出力に書き込まれます。*filename* が実行可能ファイルまたは共有オブジェクトではない場合、あるいは、読み取り用にオープンできない場合、ゼロ以外の終了ステータスが返されます。

注意事項 共有オブジェクトで `-d` または `-r` オプションを使用すると、誤解を生じるような結果が出力される場合があります。`ldd` は共有オブジェクトにおける最悪の場合を解析します。しかし実際には、未解決として報告されたシンボルでも、その一部または全部が共有オブジェクトを参照する実行可能ファイルによって解決されることがあります。実行時リンカーのプリロードメカニズムを使用すると、検査対象のオブジェクトに依存関係を追加できます。`LD_PRELOAD` を参照してください。

`ldd` は実行時リンカーと同じアルゴリズムを使用して、共有オブジェクトを見つけます。

名前	ld.so.1 – 動的オブジェクト用の実行時リンカー
形式	<pre>/lib/ld.so.1 /lib/ld.so.1 [-e <i>envar</i>] <i>dynamic-object</i> [<i>object args</i>]...</pre>
機能説明	動的アプリケーションは、1つ以上の動的オブジェクトで構成されています。動的アプリケーションとは、通常、1つの動的実行可能ファイルと1つ以上の共有オブジェクト依存関係のことです。動的アプリケーションの初期化と実行の一部として、「インタプリタ」が呼び出されます。このインタプリタは、アプリケーションをその共有オブジェクト依存関係に結合する作業を完了させます。Solaris では、このインタプリタを実行時リンカーと呼びます。 動的実行可能ファイルのリンク編集では、特別な <code>.interp</code> セクションが、関連のプログラムヘッダーとともに作成されます。このセクションには、プログラムのインタプリタを指定するパス名が組み込まれています。インタプリタのパス名は、<code>-I</code> オプションを <code>ld(1)</code> (リンカー) に使用して実行可能ファイルが作成される場合に指定できます。リンカーによって提供されるデフォルトの名前は、実行時リンカーの名前 <code>ld.so.1</code> です。 動的実行可能ファイルの実行プロセスの間は、カーネルがファイルをマッピングし、必要なインタプリタを見つけ出します。<code>exec(2)</code> and <code>mmapobj(2)</code> を参照してください。カーネルはこのインタプリタ内でマッピングを行い、このインタプリタに制御を渡します。十分に情報がインタプリタに渡されるため、インタプリタは引き続きアプリケーションを結合および実行できます。 アプリケーションの初期化に加え、実行時リンカーは、アプリケーションが自分のアドレス空間を拡張できるサービスを使用できるようにするサービスも提供します。追加の共有オブジェクトがマッピングされ、共有オブジェクト内のシンボルが結合できるようになります。 実行時リンカーは次の関数を実行します。 ■ 構成ファイルが存在する場合は、そのファイルが処理されます。構成ファイルは、デフォルト検索パスの変更に使用されます。また、ディレクトリキャッシュや代替オブジェクトの依存関係を提供します。<code>crle(1)</code> のマニュアルページを参照してください。デフォルトでは、32 ビットオブジェクトの場合には、構成ファイル <code>/var/ld/ld.config</code> が使用されます。64 ビットオブジェクトの場合には、デフォルトの構成ファイル <code>/var/ld/64/ld.config</code> が使用されます。代替の構成ファイルは、<code>LD_CONFIG</code> 環境変数で指定できます。また、代替の構成ファイルは、<code>ld(1)</code> の <code>-c</code> オプションを使用すると、動的実行可能ファイル内にもエンコードできます。 ■ 実行時リンカーは、アプリケーションの動的な情報セクション <code>.dynamic</code> を分析し、どの共有オブジェクト依存関係が必要になるかを判断します。 ■ 次に、実行時リンカーはこれらの依存関係を検索し、マッピングします。その後、各依存関係の動的な情報セクションを分析し、さらに依存関係が必要かどうかを判断します。

- 共有オブジェクトの依存関係がすべて読み込まれると、実行時リンカーは必要な再配置をすべて行います。これらの再配置により、実行プロセスに備えて共有オブジェクトが結合されます。
- 共有オブジェクトの依存関係 (場合によっては、動的実行可能ファイル) によって提供された初期化関数がすべて呼び出されます。関数は、位相的にソートされた依存関係とは逆の順序で呼び出されます。循環型依存関係が存在すると、初期化関数は循環が排除されたソート順序で呼び出されます。`ldd(1)` を使用すると、共有オブジェクトの依存関係の初期化順序を表示できます。
- アプリケーションに制御が引き渡されます。
- アプリケーションの実行中は、実行時リンカーを呼び出すことで、遅延関数結合を実行できます。
- 共有オブジェクトがプロセスから削除されると、終了関数が呼び出されます。デフォルトでは、これらの関数は、位相的にソートされた依存関係の順序で呼び出されます。
- アプリケーションは、実行時リンカーのサービスを呼び出し、`dlopen(3C)` を使用して追加の共有オブジェクトを取得することもできます。これらのオブジェクトによって提供されるシンボルは、`dlsym(3C)` を使用して結合できます。

以前の各項目の詳細については、『リンカーとライブラリ』にあります。

実行時リンカーは、指定した検索パスを使って、オブジェクトの動的な依存関係を検索します。デフォルトの検索パスは、オブジェクト内に記録された `runpath` と、それに続く一連のデフォルトのパスです。32ビットオブジェクトの場合、デフォルトは、順番に、`/lib`、そして `/usr/lib` です。64ビットオブジェクトの場合、デフォルトは、順番に `/lib/64`、そして `/usr/lib/64` です。これらのデフォルトのコンポーネントは、`crle(1)` で作成された構成ファイルを使用して変更できます。`runpath` は、`-R` オプションを `ld(1)` に使用して動的オブジェクトを作成する場合に指定されます。環境変数 `LD_LIBRARY_PATH` は、デフォルトのディレクトリの前に検索されるディレクトリを指定するために使用できます。

コマンド行の使用法

通常、実行時リンカーは、実行時リンカーをインタプリタとして宣言した動的実行可能ファイルを実行することで間接的に呼び出されます。また、実行時リンカーはコマンド行から直接実行できます。このメカニズムは、実行時リンカーの新規実装を試す場合によく使用されます。コマンド行で指定された引数は、実行時リンカーに適用可能なオプションで構成されます。これらのオプションのあとには、実行する動的オブジェクトの名前とそのオブジェクトに必要なオプションが続きます。事実上、実行時リンカーは、動的オブジェクトによって指定されたすべてのインタプリタを置き換えます。

次のオプションがサポートされています。

- e *envvar* 実行時リンカー固有の環境変数を指定します。「環境変数」の節を参照してください。このオプションを使用する変数セットは、すべての環境変数、または同名の構成ファイル変数よりも優先されます。変数

LD_NOENVIRON は、-e オプションの処理後に処理すべき環境変数がないことを示すために指定できます。

環境変数

各環境変数には、_32 または _64 という接尾辞を指定できます。この接尾辞は、環境変数をそれぞれ 32 ビットまたは 64 ビットプロセス固有のものにします。この環境変数は、接尾辞の付いてない環境変数が有効な場合でも、それに優先します。値なしに指定された環境変数で、_32 または _64 の接尾辞が付いている場合は、事実上、関連する汎用の環境変数設定が取り消されます。

LD_AUDIT、LD_AUDIT_32、および LD_AUDIT_64

コロンで区切られたオブジェクトの一覧が実行時リンカーによって読み込まれます。各オブジェクトが読み込まれると、そのオブジェクトは Link-Auditing インタフェースルーチンがないかどうか検査されます。存在するルーチンは、『[リンカーとライブラリ](#)』で説明されている Link-Auditing インタフェースで指定されているとおりに呼び出されます。[ld\(1\)](#) の -p および -P オプションも参照してください。

LD_BIND_LAZY、LD_BIND_LAZY_32、および LD_BIND_LAZY_64

実行時リンカーがレイジー結合を実行するデフォルトのモードは、環境変数 LD_BIND_LAZY を NULL 以外の任意の値に指定することで、強制的に設定できます。この設定により、実行時リンカーは、プロセスに読み込まれたすべてのオブジェクトに対してレイジー参照再配置のみを実行します。個々のオブジェクトは、オブジェクトの読み込み時にレイジー参照再配置を実行するように要求できます。[ld\(1\)](#) の -z now オプションおよび RTLD_NOW モードの [dlopen\(3C\)](#) を参照してください。LD_BIND_LAZY が有効になっている場合、レイジー参照再配置の実行要求は抑制されます。

LD_BIND_LAZY と LD_BIND_NOW が両方とも指定されている場合は、LD_BIND_NOW が優先されます。

LD_BIND_NOW、LD_BIND_NOW_32、および LD_BIND_NOW_64

実行時リンカーがレイジー結合を実行するデフォルトのモードは、環境変数 LD_BIND_NOW を NULL 以外の任意の値に指定することで、強制的に設定できます。この設定により、実行時リンカーは、プロセスに読み込まれたすべての遅延のないオブジェクトに対して即時参照とレイジー参照の両方の再配置を実行します。個々のオブジェクトは、オブジェクトの読み込み時に遅延のないレイジー参照再配置を実行するように要求できます。[ld\(1\)](#) の -z now オプションおよび RTLD_NOW モードの [dlopen\(3C\)](#) を参照してください。遅延依存関係は、LD_BIND_NOW および RTLD_NOW の影響を受けません。[ld\(1\)](#) の -z deferred オプションを参照してください。

LD_BIND_NOW と LD_BIND_LAZY が両方とも指定されている場合は、LD_BIND_NOW が優先されます。

LD_CAP_FILES、LD_CAP_FILES_32、および LD_CAP_FILES_64

すべての代替機能に対して検証すべきコンマ区切りのファイル一覧を識別します。LD_PLATCAP、LD_MACHCAP、LD_HWCAP、および LD_SFCAP。

LD_CONFIG、LD_CONFIG_32、およびLD_CONFIG_64

代替構成ファイルを提供します。構成ファイルは、デフォルト検索パスの変更に使用されます。また、ディレクトリキャッシュや代替オブジェクトの依存関係を提供します。[crle\(1\)](#)のマニュアルページを参照してください。

LD_DEBUG、LD_DEBUG_32、およびLD_DEBUG_64

コンマまたはコロン区切りのトークン一覧を提供し、実行時リンカーがデバッグ情報を標準エラーに出力するようにします。`help`は、使用可能なトークンをすべて指定した場合と同じ意味を持つ特殊なトークンです。環境変数

`LD_DEBUG_OUTPUT`は、デバッグ情報の送信先となるファイルを指定するためにも与えられます。ファイル名の末尾には、デバッグ情報を生成するアプリケーションのプロセスIDが付けられます。[lari\(1\)](#)を参照してください。

LD_DEMANGLE、LD_DEMANGLE_32、およびLD_DEMANGLE_64

診断メッセージの一部として使用されるシンボル名はすべて、ELF ファイル内で定義されたとおりに表示されます。`LD_DEMANGLE`がNULL以外の値に設定された場合、実行時リンカーはすべてのC++シンボル名をデコード(復号化)しようとします。

LD_FLAGS、LD_FLAGS_32、およびLD_FLAGS_64

環境変数情報を与える代替手段を提供します。`LD_XXX`環境変数はすべて、`xxx`トークンとして指定できます。トークンが複数ある場合は、コンマで区切って指定できます。「使用例」を参照してください。

LD_HWCAP、LD_HWCAP_32、およびLD_HWCAP_64

代替ハードウェア機能の値を識別します。

```
LD_HWCAP=[+-]{token | number},....
```

接頭辞"+"を付けると、その機能が代替機能に追加されます。接頭辞 "-"を付けると、その機能が代替機能から削除されます。"+"がない場合は、その機能が代替機能を置き換えます。

LD_LIBRARY_PATH、LD_LIBRARY_PATH_32、およびLD_LIBRARY_PATH_64

`LD_LIBRARY_PATH`環境変数が設定されている場合、この変数は、実行時リンカーが動的依存関係を探すために使用する検索パスの拡張に使用されます。`LD_LIBRARY_PATH`は、デフォルトのディレクトリの前に検索されるディレクトリのコンマ区切りの一覧を指定します。`LD_LIBRARY_PATH`は、[ld\(1\)](#)にセマンティクスを追加します。

LD_LOADFLTR、LD_LOADFLTR_32、およびLD_LOADFLTR_64

フィルタは共有オブジェクト形式です。フィルタは、実行時に代替の共有オブジェクトを選択できるようにして、フィルタ内で定義されるシンボルに実装を提供します。[ld\(1\)](#)の `-f` および `-F` オプションを参照してください。デフォルトでは、フィルタに対してシンボル解決が発生するまで代替共有オブジェクト処理は延期されます。`LD_LOADFLTR`がNULL以外の値に設定されている場合、どのフィルタも読み込まれるとただちに処理されます。[ld\(1\)](#)の `-z loadfltr` オプションも参照してください。

LD_MACHCAP、LD_MACHCAP_32、およびLD_MACHCAP_64
 代替マシンハードウェアの名前を識別します。

LD_NOAUDIT、LD_NOAUDIT_32、およびLD_NOAUDIT_64
 ローカル監査ライブラリは、アプリケーションと共有オブジェクト内で定義できます。[ld\(1\)](#)の **-p** および **-P** オプションを参照してください。LD_NOAUDIT が NULL 以外の値に設定されている場合、実行時リンカーはローカルの監査ライブラリをすべて無視します。

LD_NOAUXFLTR、LD_NOAUXFLTR_32、およびLD_NOAUXFLTR_64
 補助フィルタは共有オブジェクト形式です。補助フィルタは、実行時に代替の共有オブジェクトを選択できるようにして、フィルタ内で定義されるシンボルに実装を提供します。[ld\(1\)](#)の **-f** オプションを参照してください。LD_NOAUXFLTR が NULL 以外の値が設定されている場合、実行時リンカーはこの代替共有オブジェクトの検索を無効にします。

LD_NOCONFIG、LD_NOCONFIG_32、およびLD_NOCONFIG_64
 デフォルトでは、実行時リンカーは構成ファイルを開いて処理しようとします。LD_NOCONFIG が NULL 以外の値に設定されている場合、実行時リンカーはこの構成ファイル処理を無効にします。

LD_NODIRCONFIG、LD_NODIRCONFIG_32、およびLD_NODIRCONFIG_64
 構成ファイル内で提供されるすべてのディレクトリキャッシュ情報は無視されるため、LD_NOCONFIG のサブセットを提供します。

LD_NODIRECT、LD_NODIRECT_32、およびLD_NODIRECT_64
 直接結合情報は、実行時リンカーに関連オブジェクト内のシンボルを直接検索するように指示します。[ld\(1\)](#)の **-B direct** オプションを参照してください。直接結合がない場合、実行時リンカーによって実行されるシンボル検索はデフォルトのモデルに従います。LD_NODIRECT が NULL 以外の値に設定されている場合、実行時リンカーは直接結合情報をすべて無視します。

LD_NOENVCONFIG、LD_NOENVCONFIG_32、およびLD_NOENVCONFIG_64
 構成ファイル内で提供されるすべての環境変数は無視されるため、LD_NOCONFIG のサブセットを提供します。

LD_NOLAZYLOAD、LD_NOLAZYLOAD_32、およびLD_NOLAZYLOAD_64
 レイジー読み込みとラベル付けされている依存関係は、その依存関係への明示的な参照が作成されるまでメモリーに読み込まれません。[ld\(1\)](#)の **-z lazyload** オプションを参照してください。LD_NOLAZYLOAD が NULL 以外の値に設定されている場合、実行時リンカーは依存関係のレイジー読み込みのラベルを無視し、その依存関係をただちに読み込みます。

LD_NOOBJALTER、LD_NOOBJALTER_32、およびLD_NOOBJALTER_64
 構成ファイル内で提供されるすべての代替オブジェクト依存関係は無視されるため、LD_NOCONFIG のサブセットを提供します。

LD_NOVERSION、LD_NOVERSION_32、およびLD_NOVERSION_64

デフォルトでは、実行時リンカーは、プライマリ実行可能ファイルおよびすべての依存関係についてバージョンの依存関係を検証します。LD_NOVERSIONがNULL以外の値が設定されている場合、実行時リンカーはこのバージョンチェックを無効にします。

LD_ORIGIN、LD_ORIGIN_32、およびLD_ORIGIN_64

\$ORIGINの即時処理は、環境変数LD_ORIGINをNULL以外の値に設定することによって開始できます。Solaris 9より前のバージョンでは、このオプションは、\$ORIGIN文字列トークンを使用する依存関係を検索する前にchdir(2)を呼び出すアプリケーションにとって便利なオプションでした。現在では、デフォルトで実行時リンカーが現在の作業用ディレクトリを作成するため、このオプションは不要です。

LD_PLATCAP、LD_PLATCAP_32、およびLD_PLATCAP_64

代替プラットフォームの名前を識別します。

LD_PRELOAD、LD_PRELOAD_32、およびLD_PRELOAD_64

スペース区切りの共有オブジェクトの一覧を提供します。これらのオブジェクトは、実行されるプログラムのあとに読み込まれますが、そのプログラムが参照するほかの共有オブジェクトより早く読み込まれます。事前に読み込まれたオブジェクトによって提供されるシンボル定義は、プログラムが参照する共有オブジェクトによって作成された参照に挿入されます。事前に読み込まれたオブジェクトによって提供されるシンボル定義は、プログラムによって提供されるシンボル定義には挿入されません。

LD_PROFILE、LD_PROFILE_32、およびLD_PROFILE_64

実行時リンカーによってプロファイルが作成される共有オブジェクトを定義します。プロファイルの作成が有効な場合、プロファイリングバッファファイルが作成され、マッピングされます。バッファファイル名はプロファイルが作成される共有オブジェクトの名前で、拡張子は.profileとなります。デフォルトでは、このバッファは/var/tmpの下に置かれます。また、環境変数LD_PROFILE_OUTPUTが、プロファイリングバッファを置く代替ディレクトリを示すために設定されます。

プロファイリングバッファには、profil(2)と呼び出し回数情報が含まれます。この情報は、ccの-xpgオプションにリンクされたプログラムによって生成されるgmon.out情報と同様です。指定された共有オブジェクトを使用し、この環境変数が設定されているときに動作するアプリケーションはすべて、プロファイルバッファにデータを蓄積します。「注意事項」も参照してください。プロファイルバッファ情報は、gprof(1)を使用して検査できます。

LD_PROFILEによるプロファイル作成方法は、コンパイルシステムによって提供されることがあるほかの方法の代替手段です。プロファイルが作成される共有オブジェクトでは設定を行う必要はまったくなく、LD_PROFILEをプロファイルが設定

されているアプリケーションと組み合わせるはいけません。共有オブジェクトのプロファイル作成について詳しくは、『[リンカーとライブラリ](#)』を参照してください。

LD_SFCAP、**LD_SFCAP_32**、および **LD_SFCAP_64**
代替ソフトウェア機能の値を識別します。

`LD_SFCAP=[+-]{token | number},...`

接頭辞 "+" を付けると、その機能が代替機能に追加されます。接頭辞 "-" を付けると、その機能が代替機能から削除されます。"+" がいない場合は、その機能が代替機能を置き換えます。

LD_SIGNAL、**LD_SIGNAL_32**、および **LD_SIGNAL_64**

致命的な実行時エラーの発生時に、実行時リンカーがプロセスを強制終了するために使用する「数値」シグナル番号を提供します。[thr_kill\(3C\)](#)を参照してください。デフォルトでは、SIGKILL が使用されます。たとえば、代替シグナル番号が 6 (SIGABRT) の場合、デバッグを支援するコアファイルを作成できます。[dlinfo\(3C\)](#) への RTLD_DI_SETSIGNAL 要求も参照してください。

文字列 'LD_' で始まる環境変数名は、[ld\(1\)](#) と `ld.so.1` の拡張用として予約されています。

セキュリティー セキュリティー保護されたプロセスでは、悪意のある依存関係の置換またはシンボルの割り込みを防ぐため、依存関係と `runpath` の評価にいくつかの制約があります。

実行時リンカーは、[issetugid\(2\)](#) システム呼び出しがプロセスに対して `true` を返した場合、そのプロセスを安全と分類します。

32 ビットオブジェクトの場合、実行時リンカーが認識しているデフォルトのトラストディレクトリは、`/lib/secure/` と `/usr/lib/secure/` です。64 ビットオブジェクトの場合、デフォルトのトラストディレクトリは `/lib/secure/64` と `/usr/lib/secure/64` です。ユーティリティー [crle\(1\)](#) を使用すれば、セキュリティー保護されたアプリケーション向けに追加のトラストディレクトリを指定できます。この方法を使用する場合には、管理者は、ターゲットディレクトリを悪意のある侵入から適切に保護する必要があります。

あるセキュリティー保護されたプロセスに対して `LD_LIBRARY_PATH` ファミリ環境変数が有効になっている場合、実行時リンカーの検索規則を拡張するために使用されるのは、この変数に指定された「トラスト」ディレクトリだけです。

セキュリティー保護されたプロセスでは、アプリケーションまたはその依存関係のいずれかによって提供される `runpath` コンポーネントは、そのコンポーネントがフルパス名 (つまり、'/' で始まるパス名) である場合に使用されます。

セキュリティー保護されたプロセスでは、`$ORIGIN` 文字列の拡張は、その文字列がトラストディレクトリに拡張されるときに限り許可されます。ただし、`$ORIGIN` を

展開することです。すでに依存関係を提供したディレクトリに一致する場合、そのディレクトリは暗黙にセキュリティ保護されます。このディレクトリは、追加の依存関係を提供するために使用できます。

セキュリティ保護されたプロセスでは、LD_CONFIGは無視されます。ただし、セキュリティ保護されたアプリケーションで記録された構成ファイルは使用されます。[ld\(1\)](#)の-cオプションを参照してください。記録される構成ファイルは、フルパス名(つまり、'/'で始まるパス名)にする必要があります。記録される構成ファイルが\$ORIGIN文字列を使用する場合、そのファイルは既知のトラストディレクトリに制限されます。セキュリティ保護されたアプリケーション内の構成ファイルを記録する開発者は、構成ファイルディレクトリを悪意のある侵入から適切に保護する必要があります。記録された構成ファイルがない場合、セキュリティ保護されたプロセスは、構成ファイルがあれば、デフォルトの構成ファイルを使用します。[crle\(1\)](#)を参照してください。

セキュリティ保護されたプロセスでは、LD_SIGNALは無視されます。

追加のオブジェクトは、LD_PRELOADまたはLD_AUDIT環境変数を使用すると、セキュリティ保護されたプロセスで読み込むことができます。これらのオブジェクトは「フル」パス名または「単純」ファイル名で指定しなければなりません。フルパス名は、既知の「トラスト」ディレクトリに限定されます。単純ファイル名(名前に'/'が付いていない)は、前述した検索パスの制約に従って配置されます。単純ファイル名は、既知の「トラスト」ディレクトリにのみ解決されることになります。

セキュリティ保護されたプロセスでは、単純ファイル名を構成する依存関係は、前述のパス名の制約を使用して処理されます。フルパス名または相対パス名で表示された依存関係は、そのまま使用されます。したがって、セキュリティ保護されたプロセスの開発者は、フルパス名または相対パス名依存関係として参照されるターゲットディレクトリを悪意のある侵入から適切に保護する必要があります。

セキュリティ保護されたプロセスを作成する場合、依存関係を表したり、[dlopen\(3C\)](#)パス名を作成したりするために相対パス名を使用してはいけません。この制約は、アプリケーションと「すべての」依存関係に適用するようにしてください。

使用例

例1 LD_FLAGSのグループ環境変数情報への使用

LD_FLAGSの次の使用法は、環境変数LD_BIND_NOWとLD_LIBRARY_PATHをそれぞれ32ビットアプリケーション用に設定することと同等です。

```
example% LD_FLAGS_32=bind_now,library_path=/lib/one:/lib/two
```

LD_FLAGSの次の使用法は、環境変数LD_LIBRARY_PATHとLD_PRELOADをそれぞれ64ビットアプリケーション用に設定することと同等です。

```
example% LD_FLAGS_64=library_path=/lib/one/64,preload=foo.so
```

ファイル

/lib/ld.so.1

デフォルトの実行時リンカー。

<code>/lib/libc.so.1</code>	SVID ABI 互換性のための代替インタプリタ
<code>/usr/lib/0@0.so.1</code>	NULL 文字ポインタをサポートする互換性ライブラリ。「注意事項」を参照してください。
<code>/lib/secure</code> および <code>/usr/lib/secure</code>	セキュリティー保護されたアプリケーション用の LD_PRELOAD の位置。
<code>/lib/secure/64</code> および <code>/usr/lib/secure/64</code>	セキュリティー保護された 64 ビットアプリケーション用の LD_PRELOAD の位置。
<code>/lib/64/ld.so.1</code>	64 ビットアプリケーション用のデフォルトの実行時リンカー。
<code>/usr/lib/64/0@0.so.1</code>	NULL 文字ポインタをサポートする 64 ビット互換性ライブラリ。「注意事項」を参照してください。
<code>/var/ld/ld.config</code>	32 ビットアプリケーション用のデフォルトの構成ファイル。
<code>/var/ld/64/ld.config</code>	64 ビットアプリケーション用のデフォルトの構成ファイル。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/linker

関連項目 [crle\(1\)](#), [gprof\(1\)](#), [lari\(1\)](#), [ld\(1\)](#), [ldd\(1\)](#), [exec\(2\)](#), [issetugid\(2\)](#), [mmapobj\(2\)](#), [profil\(2\)](#), [dladdr\(3C\)](#), [dlclose\(3C\)](#), [dldump\(3C\)](#), [dlerror\(3C\)](#), [dlinfo\(3C\)](#), [dlopen\(3C\)](#), [dlsym\(3C\)](#), [thr_kill\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

注意事項 LD_PROFILE を、[proc\(4\)](#) のユーザーなど、ほかのプロセス監視方法と組み合わせて使う場合は注意してください。プロセス監視方法が複数あると、デッドロック条件が発生してプロファイルバッファがロックされたままになるおそれがあります。ロックされたバッファは、プロファイル情報を記録しようとするプロセスをすべてブロックします。この問題が発生する可能性を減らすために、実行時リンカーのプロファイル実装は、起動時にプロセスが監視されているかどうかを確認します。その場合、プロセスのプロファイル作成は何も出力されずに無効にされます。ただし、このメカニズムでは、このプロセスに付随する監視プロセスをプロセスの実行中に取得できません。

ユーザー互換性ライブラリ `/usr/lib/0@0.so.1` では、`0` の値を位置 `0` で設定するメカニズムを利用できます。アプリケーションの中には、NULL 文字ポインタを NULL 文字列へのポインタと同じように処理するべきだと誤って解釈するものがあります。このようなアプリケーションでは、NULL 文字ポインタに対してアクセスがあるとセグメンテーション違反が発生します。`LD_PRELOAD` を使用して実行時にこのライブラリがそのようなアプリケーションに追加されると、こうした誤動作が起りやすい環境が作られます。しかし、ユーザー互換性ライブラリは、このようなアプリケーションを生成するためのものでも、こうした特殊なプログラミング作業を奨励するためのものでもありません。

ほとんどの場合、`/usr/lib/0@0.so.1` が存在していても問題はなく、これを必要としないプログラムに事前に読み込まれてもかまいません。ただし、例外があります。JVM (Java 仮想マシン) など一部のアプリケーションでは、NULL ポインタへのアクセスによってセグメント違反が発生する必要があります。JVM などのアプリケーションでは、`/usr/lib/0@0.so` を事前に読み込まないようにしてください。

名前	let - 1 つ以上の算術式を評価するためのシェル組み込みコマンド
形式	
ksh88	let <i>arg</i> ...
ksh	let [<i>expr</i> ...]
機能説明	
ksh88	各 <i>arg</i> は、評価の対象となる個々の算術式を表します。
ksh	let は、現在のシェル環境において各 <i>expr</i> を ANSIC 構文を使用した算術式として評価します。変数名はシェル変数であり、算術式として再帰的に評価されて数値が取得されます。let は、演算子をクォートしなくてもコマンド引数として渡すことができる ksh(1) の ((...)) 構文によって廃止されました。
終了ステータス	
ksh88	ksh88 は、次の終了値を返します。 0 最後の式の値が 0 以外。 1 最後の式の値が 0。
ksh	ksh は、次の終了値を返します。 0 最後の <i>expr</i> が 0 以外の値と評価されました。 >0 最後の <i>expr</i> が 0 と評価されたか、エラーが発生しました。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。
関連項目	ksh(1) , ksh88(1) , set(1) , typeset(1) , attributes(5)

名前	lex – 字句解析プログラムの生成
形式	lex [-cntv] [-e -w] [-V -Q [y n]] [file]...
機能説明	lex ユーティリティーは、文字入力の字句解析に用いる C プログラムを生成します。このプログラムは、yacc へのインタフェースとして使用できます。lex ソースコードから生成された C プログラムは、ISO C 標準に準拠しています。通常 lex ユーティリティーは、生成したプログラムを lex.yy.c ファイルに書き出します。lex の終了ステータスがゼロでない場合、このファイルの状態は不確定です。lex の入力言語に関する詳細は、「拡張機能説明」の項を参照してください。
オプション	サポートしているオプションは、次のとおりです。 -c C 言語の動作 (デフォルトのオプション) を示します。 -e EUC 文字を扱えるプログラムを生成します。本オプションは -w と同時に指定することはできません。yytext[] の型は unsigned char[] となります。 -n -v によって出力される合計情報の出力を抑止します。lex ソースコード中にテーブルのサイズが指定されず、-v オプションを省略した場合には、この -n が指定されたものと見なされます。 -t 生成したプログラムを、lex.yy.c ファイルではなく標準出力上に書き出します。 -v lex 統計のサマリーを標準エラー出力に書き出します (「lex の定義」の項にある lex テーブルサイズの説明を参照してください)。lex ソースコード中にテーブルサイズが指定され、-n オプションが指定されなかった場合、-v オプションを有効にできます。 -w EUC 文字を扱えるプログラムを生成します。本オプションは -e と同時に指定することはできません。-yytext[] の型は wchar_t[] となります。この点が e 指定と異なります。 -V 標準エラー出力上にバージョン情報を書き出します。 -Q[y n] -Qy は、ファイル lex.yy.c にバージョン情報を書き出します。-Qn はバージョン情報を出力しません。デフォルト指定は Qn です。
オペランド	次のオペランドを指定できます。 <i>file</i> 入力ファイルのパス名。このような複数の <i>file</i> を指定すると、すべてのファイルが連結されて 1 つの lex プログラムが生成されます。<i>file</i> オペランドを 1 つも指定しない場合、または <i>file</i> オペランドに - を指定した場合には、標準入力を用いられます。
出力	lex の出力ファイルについては下記を参照してください。

Stdout	-t オプションが指定されると、lex の C ソースコード出力のテキストファイルが標準出力に書き出されます。
Stderr	-t オプションが指定されると、lex ソースコード入力に関する情報メッセージ、エラーメッセージ、および警告メッセージが標準エラー出力に書き出されます。 -t オプションが指定されなければ、次のようになります。 1. lex ソースコード入力の内容に関する情報メッセージ、エラーメッセージ、および警告メッセージが標準出力または標準エラー出力に書き出されます。 2. -v オプションが指定され、-n オプションが指定されなかった場合は、lex 統計情報も標準エラー出力に書き出されます。-n オプションが指定されていない限り、lex の定義のセクション(「拡張機能説明」を参照)内にテーブルサイズが % 演算子で指定された場合も、この統計情報が生成されます。
出力ファイル	C ソースコードが入ったテキストファイルが lex.yy.c に、または -t オプションが指定された場合は標準出力に書き出されます。
拡張機能説明	各入力ファイルには lex ソースコードが入っています。これは、C プログラムフラグメントの形式で対応するアクションを定義した正規表現のテーブルです。 lex.yy.c がコンパイルされ lex ライブラリにリンクされると (-c89 または cc で ll オペランドを使って)、生成されたプログラムは標準入力から文字入力を読み込み、与えられた式と一致する文字列に分割します。 式が一致すると、次の処理が行われます。 ■ 一致した入力文字列は、NULL で終わっている文字列として <i>yytext</i> に残されます。<i>yytext</i> は、外部文字配列または文字列へのポインタのどちらかです。lex の定義のセクションで説明するように、%array または %pointer 宣言を使えば型を明示選択できますが、デフォルトは %array です。 ■ 照合文字列の長さが外部変数 <i>int yyleng</i> に設定されます。 ■ 式に対応するプログラムフラグメントまたはアクションが実行されます。 パターン照合の間、lex はパターンセットを検索し、一致するもっとも長い文字列を見つけ出します。複数のルールが同じ長さの文字列に一致する場合は、最初のルールが選択されます。 lex ソースの一般形式は、次のとおりです。 <pre> Definitions %% Rules %% User Subroutines </pre>

最初の%%はルール(正規表現とアクション)の開始を示すために必要なもので、2番目の%%はユーザーサブルーチンが続く場合にのみ必要なものです。

lex の Definitions で、空白文字で始まる行は、C プログラムフラグメントとみなされ、lex.yy.c ファイルの外部定義域にコピーされます。同様に、%{ と %} だけの区切り行に囲まれた部分もそのまま、lex.yy.c ファイルの外部定義域にコピーされます。

Rules セクションの始めに、ほかのルールが指定される前にこのような入力(空白文字で始まる、または %{ と %} だけの区切り行に囲まれる)がある場合、lex.yy.c の yylex 関数の変数宣言のあと、および yylex 内の最初のコード行の前に書き出されます。そのため yylex に入ったときに実行されるアプリケーションコードだけでなく、yylex にローカルなユーザー変数をここで宣言できます。

Rules セクションで、いくつかのルールのあとに空白文字で始まる、または %{ と %} だけの区切り行に囲まれる入力があったときに lex が実行するアクションは未定義です。このような入力があったとき、yylex 関数の定義にエラーが生じることがあります。

lex 内の定義

lex 内の定義は最初の%%区切り行より前にあります。このセクション内で %{ と %} 区切り行に囲まれていない行で、しかも空白文字以外で始まっている行は、lex 置換文字列を定義するものとみなされます。これらの行の形式は、次のとおりです。

name substitute

name が ISO C 標準の識別子の条件を満たしていない場合、結果は未定義です。*substitute* 文字列は、ルール内で使用されると *{name}* 文字列を置き換えます。この状況で *name* 文字列が認識されるのは、中括弧が使用されていて、大括弧や二重引用符で囲まれていないときだけです。

lex 内の定義セクションでは、%(パーセント記号)で始まりその後ろに s または S で始まる英数字語が続く行は、開始条件セットを定義します。%(パーセント記号)で始まりその後ろに x または X で始まる英数字語が続く行は、排他的な開始条件セットを定義します。生成されたスキナが %s 状態のときは、状態が指定されていないパターンもアクティブになります。%x 状態では、このようなパターンはアクティブになりません。最初の語を除いた行の残りは、空白文字で区切られた開始条件名とみなされます。開始条件名は、定義名と同じ方法で作成されます。「lex の正規表現」の項で説明するように、開始条件は、正規表現の照合をいくつかの状態に制限するときに使用できます。

lex 内の定義セクションで、次の排他的な宣言の2つのうちどちらかを使用します。

%array yytext の型を NULL で終わる文字配列と宣言します。

%pointer yytext の型を NULL で終わる文字配列へのポインタと宣言します。

`yytext` を変更するために、`%pointer` オプションと同時に `yyless` 関数を使用することはできません。

`%array` はデフォルトです。`%array` が指定されている (または `%array` と `%pointer` の 2 つとも指定されていない) 場合に、外部参照先を `yytext` するには次のように書式を宣言します。

```
extern char yytext[ ]
```

`%pointer` が指定されている場合、外部参照先の書式は次のとおりです。

```
extern char *yytext;
```

`lex` では、`lex` 内の定義セクションの特定の内部テーブルサイズの設定を宣言できます。次の表に、宣言を示します。

`lex` でのテーブルサイズ宣言

宣言	説明	デフォルト
<code>%pn</code>	ポジションの数	2500
<code>%nn</code>	状態の数	500
<code>%a n</code>	遷移の数	2000
<code>%en</code>	解析ツリーのノード数	1000
<code>%kn</code>	バック文字クラスの数	10000
<code>%on</code>	出力配列のサイズ	3000

`lex` により生成されたプログラムは、補助コードセットの EUC 文字を含む入力データを処理するため、`-e` または `-w` のいずれかのオプションの指定を必要とします。この両オプションがともに省略されると、`yytext` の型は `char[ ]` となり、生成されたプログラムは ASCII 以外のコードセットを扱うことはできません。

`-e` オプションを使用すると、`yytext` の型は `unsigned char[ ]` となり、`yyleng` は一致した文字列のバイト数を示すことになります。マクロ `input()`、`unput(c)`、および `output(c)` は、通常の ASCII `lex` と同様に、バイトを基準とした I/O を実行しなければなりません。`-e` オプションで使用する変数はほかに 2 つあります。それは `yywtext` と `yywleng` で、それぞれ `-w` オプション指定時の `yytext` および `yyleng` と同じ動きをします。

`-w` オプションを指定すると、`yytext` の型は `wchar_t[ ]` となり、`yyleng` には一致した文字列の文字数が記録されます。ユーザーがこのオプションを使って独自の `input()`、`unput(c)`、または `output(c)` マクロを指定するのであれば、それらはワイド文字 (`wchar_t`) の形式で EUC 文字を返したり受け取ったりするよう設計しなければ

ばなりません。これによって、ユーザーのプログラムと `lex` 内部との間に別のインタフェースを設け、ほかのプログラムの処理速度を上げることが可能となります。

lex 内のルール

`lex` ソースファイル内のルールは、左のカラムには正規表現、右のカラムにはその正規表現が認識されたときに実行されるアクション (C プログラムフラグメント) が入ったテーブルです。

```
ERE action
```

```
ERE action
```

```
...
```

行の拡張正規表現 (ERE) 部分と *action* は、1 つ以上の空白文字で区切られます。空白文字が入った正規表現は、次の条件の 1 つが満たされる場合に認識されます。

- 表現全体が二重引用符で囲まれている
- 二重引用符または大括弧内に空白文字がある
- 各空白文字の前にバックスラッシュがある

lex 内のユーザーサブルーチン

ユーザーサブルーチンセクション内のものはすべて、`lex.yy.c` の `yylex` のあとにコピーされます。

lex 内の正規表現

`lex` ユーティリティーは、[regex\(5\)](#) で記述されている拡張正規表現 (ERE) セットをサポートします。ただし次のように追加された構文と、例外となる構文があります。

... 二重引用符で囲まれた文字列は、二重引用符内の文字そのものを表します。ただし、バックスラッシュエスケープは認識されます (次の表を参照してください)。バックスラッシュエスケープシーケンスは、閉じる引用符で終端されます。たとえば `" \ 01"1` は、8 進数の 1 と後続の文字 1 という単一の文字列を表します。

`<state>r`

`<state1, state2, ...>r`

正規表現 *r* が照合されるのは、プログラムが *state*、*state1* などによって示される開始条件の 1 つにあるときだけです。詳細については、「`lex` 内のアクション」の項を参照してください。(以降本書では、印刷上の規則の例外として、たとえば `<state>` はメタ変数を表さず、記号を囲むリテラル山括弧文字を表します)。このように開始条件が認識されるのは、正規表現の始めだけです。

r/x

正規表現 *r* が照合されるのは、その後ろに正規表現 *x* が続いているときだけです。`yytext` で戻されるトークンは、*r* とだけ照合されます。*r* の末尾部分が *x* の先頭部分と一致する場合の結果は不確定です。*r* 式には末尾コンテキストや `$` (行の終わり) 演算子は含めることができません。*x* には `&` (行の始まり) 演算子、末尾コンテキスト、`$` 演算子は含めることができません。つまり `lex` 正規表現には 1 つの末尾コンテキストしか含めず、`^` 演算子が使用できるのはこのような式の始

めだけです。末尾コンテキスト演算子 / (スラッシュ) は、括弧でグループ化できないよう制限されています。

`{name}`

`name` が *Definitions* セクションからの置換記号の 1 つである場合、左右の中括弧も含めた文字列が *substitute* 値で置き換えられます。拡張正規表現内では、*substitute* 値は括弧で囲まれているものとして扱われます。`{name}` が大括弧や二重引用符で囲まれている場合、置換は行われません。

ERE 内では、バックスラッシュ文字 (`\\`, `\ a`, `\ b`, `\ f`, `\ n`, `\ r`, `\ t`, `\ v`) をエスケープシーケンスの始まりとみなします。さらに次の表のエスケープシーケンスが認識されます。

リテラル復帰改行文字 (NEWLINE) は、ERE 内ではありえません。復帰改行文字を表現するには、エスケープシーケンス `\ n` が使用されます。復帰改行文字は、ピリオド演算子とは照合できません。

lex 内のエスケープシーケンス

lex 内のエスケープシーケンス		
エスケープシーケンス	説明	意味
<code>\ digits</code>	バックスラッシュ文字とそのあとの 1、2、または 3 桁の 8 進数の最長シーケンス (01234567)。すべての桁が 0 なら (つまり NULL 文字なら)、動作は未定義	エンコーディングが 1、2、または 3 桁の 8 進数で表現される文字。複数バイト文字は、この種のエスケープシーケンスがいくつか連続したものを必要とします。そのとき、各バイトの先頭に <code>\</code> が必要です。
<code>\ xdigits</code>	バックスラッシュ文字と、そのあとに続く 16 進文字の最長シーケンス (01234567abcdefABCDEF)。すべての桁が 0 (つまり NULL 文字) の場合、動作は未定義です。	エンコーディングが 16 進整数で表現される文字。
<code>\ c</code>	バックスラッシュ文字と、そのあとに続くこの表に記載されていない文字。 (<code>\\</code> , <code>\ a</code> , <code>\ b</code> , <code>\ f</code> , <code>\ n</code> , <code>\ r</code> , <code>\ t</code> , <code>\ v</code>)	文字 <code>c</code> 、変更なし。

lex 用の拡張正規表現の優先順位は、下の表に示すとおりです。優先度の高い順に並んでいます。

エスケープ文字エントリは、これらが演算子であることを意味しませんが、真の演算子との関係を示すため、表に組み込まれています。この項で説明した配置方法の

制限のため、開始条件、末尾コンテキスト、およびアンカー指定は省略されています。これらは ERE の先頭または末尾にだけ指定できます。

lex 内での ERE 優先順位	
照合関連の括弧記号	[= =] [: :] [. .]
エスケープ文字	\<special character>
大括弧表現	[]
引用	". . ."
グループ化	()
定義	{name}
単一文字 RE 重複	* + ?
連結	
インターバル表現	{m,n}
択一	

ERE アンカー演算子 (^ と \$) は、表にはありません。lex 正規表現では、これらの演算子の使用が次のように制限されています。^ 演算子は正規表現の始めにしか使用できず、\$ 演算子は最後にしか使用できません。演算子は正規表現全体に適用されます。そのためたとえば、(^abc)|(def\$) などのパターンは未定義です。これは 2 つの別々のルールとして書くことができます。正規表現 ^abc と正規表現 def\$ です。この 2 つは、特殊な | アクション (下記を参照のこと) を介して共通なアクションを共有します。パターンが ^abc|def\$ と書かれたら、abc または def だけの行と一致します。

一般の ERE ルールと異なり、従来のほとんどの lex の実装では組み込み型アンカーは許可されていません。組み込み型アンカーの例としては、foo が完全な語として存在するときに foo と照合させるための (^)foo(\$) などのパターン用があります。これは、次の既存の lex 機能を使えば可能です。

```
^foo/[ \ n]|
" foo"/[ \ n]    /* found foo as a separate word */
```

なお \$ も末尾コンテキストの形式であり (\ n と等価)、その演算子の別のインスタンスを含む正規表現では使用できません (上の末尾コンテキストの説明を参照してください)。

追加の正規表現末尾コンテキスト演算子 / (スラッシュ) は、二重引用符で囲む (" / ")、前にバックスラッシュを付ける (\ /)、または大括弧で囲めば ([/])、通常の

文字として使用できます。開始条件 `<` と `>` 演算子は、正規表現の始めの開始条件においてだけ特別とみなされます。正規表現のその他の場所では、通常の文字として扱われます。

次の例を見れば、lex 正規表現と本書に出てくるその他の正規表現との違いがわかります。 r/x 形式の正規表現の場合、 r と一致する文字列が常に返されます。 x の先頭が r の末尾部分と一致する場合に混乱が生じることがあります。たとえば、正規表現 a^*b/cc と入力 $aaabcc$ が与えられると、 $yytext$ にはこの照合結果として $aaab$ が入ります。しかし正規表現 x^*/xy と入力 $xxxy$ が与えられると、 xxx は x^* に一致するため、 xx ではなく xxx が返されます。

ルール ab^*/bc では、 r の最後の b^* が末尾コンテキストの先頭まで一致するため、結果は不確定です。ただしこのルールが ab/bc なら、テキスト ab にテキスト bc が続く場合に一致します。この場合 r は x の先頭の b までに一致しないので、結果は確定的です。

lex 内のアクション ERE が一致したときに実行されるアクションは、C プログラムフラグメントまたは次に説明する特殊なアクションです。プログラムフラグメントには1つ以上のCステートメントと特殊なアクションが入っています。空のCステートメント `;` は、有効なアクションです。このようなルールのパターン部分に一致する `lex.yy.c` 入力内の文字列は、実際には無視またはスキップされます。ただしアクションの不在は無効であり、このような状況で lex が実行するアクションは未定義です。

C ステートメントと特殊アクションも含め、アクションについての仕様は、中括弧で囲まれていれば複数行にわたることができます。

```
ERE <one or more blanks> { program statement
program statement }
```

`lex.yy.c` プログラムへの入力内の文字列がどの表現とも一致しないときは、デフォルトアクションとして、文字列が出力にコピーされます。lex によって生成されるプログラムのデフォルトの動作は、入力を読み込んで出力へコピーするだけなので、`%` だけの最小 lex ソースプログラムは、入力をそのまま出力へコピーするだけのCプログラムを生成します。

4つの特殊アクションが使用可能です。

	ECHO;	REJECT;	BEGIN

	アクション <code> </code> は、次のルールのアクションがこのルールのアクションであることを意味します。ほかの3つのアクションとは異なり、 <code> </code> は中括弧で囲んだりセミコロンで終わることができません。それは、ほかの動作なしで、単独で指定する必要があります。

ECHO;	文字列 <code>yytext</code> の内容を出力に書き出します。

REJECT;	1つの式だけを入力内の文字列と一致させるのが普通です。REJECT は「現在の入力と一致する次の式まで継続する」という意味で、現在のルールが実行されたあと、2番目のルールが何であれ同じ入力に対し

て実行されます。そのため、1つの入力文字列または重なる入力文字列に対して複数のルールを一致させて実行できます。たとえば、正規表現 `xyz`、正規表現 `xy`、および入力 `xyz` が与えられると、通常は正規表現 `xyz` だけが一致します。次の照合の試行は `z` のあとから開始されます。`xyz` ルールの最後のアクションが `REJECT` なら、このルールと `xy` ルールの両方が実行されます。`yyllex` の別の部分への `goto` と同じように、制御の流れが継続しないような方法で、`REJECT` アクションを実装できます。`REJECT` を使用すると、スキャナはある程度大きくなり、実行が遅くなります。

BEGIN アクション

BEGIN *newstate*;

は、状態 (開始条件) を *newstate* に切り替えます。文字列 *newstate* が `lex` 内の定義セクションに開始条件として宣言されていないと、結果は不確定です。初期状態は、数字の `0` または `INITIAL` トークンで示されます。

次に説明する関数やマクロは、`lex` 入力内のユーザーコードをアクセスできます。`lex` の C コード出力に示されるかどうか、および `c89` や `cc` への `-ll` オペランドを通してだけアクセス可能かどうかは不確定です (`lex` ライブラリ)。

<code>int yylex(void)</code>	入力の字句を解析します。これは <code>lex</code> ユーティリティによって生成される主な関数です。この関数は、入力の終わりに達するとゼロを返します。その他の場合は、選択されたアクションによって決定されたゼロ以外の値 (トークン) を返します。
<code>int yymore(void)</code>	呼び出されると、次の入力文字列がいつ認識されるかを示します。置き換えるのではなく、 <code>yytext</code> の現在の値に付加されます。これに合わせて <code>yyleng</code> の値が調整されます。
<code>int yyless(int n)</code>	<code>NULL</code> で終わっている <code>yytext</code> 内の最初の <i>n</i> 文字を記憶し、残りの文字は読み取っていないものとします。これに合わせて <code>yyleng</code> の値が調整されます。
<code>int input(void)</code>	入力から次の文字を返します。ファイルが終わりのときはゼロを返します。ストリームポインタ <code>yyin</code> から、おそらく中間バッファを介して入力を得ます。そのためスキニングの開始後に <code>yyin</code> の値を変更した場合の影響は未定義です。読み込まれた文字列は、スキャナの入力ストリームからそのまま取り除かれます。
<code>int unput(int c)</code>	文字 <i>c</i> を入力に返します。次の式が一致するまで <code>yytext</code> と <code>yyleng</code> は未定義です。入力された文字より多い文字に対して <code>unput</code> を使用すると、結果は不確定です。

次の関数は `-ll` オペランドを通してアクセス可能な `lex` ライブラリ内にだけ出てきます。そのためこれらの関数は移植性のあるアプリケーションによって再定義可能です。

<code>int yywrap(void)</code>	ファイルの終わりで <code>yylex</code> によって呼び出されます。デフォルトの <code>yywrap</code> は常に 1 を返します。アプリケーションが <code>yylex</code> に別の入力ソースで処理を継続させたい場合、アプリケーションは関数 <code>yywrap</code> を組み込みます。これは別のファイルと外部変数 <code>FILE *yyin</code> を関連付け、ゼロの値を返します。
<code>int main(int argc, char *argv[])</code>	字句解析のために <code>yylex</code> を呼び出してから、終了します。ユーザーコードにはアプリケーションに固有な動作を実行する <code>main</code> を組み込んだり、必要なら <code>yylex</code> を呼び出したたりできます。

移植性のあるアプリケーションによって確実に再定義可能なのは `libl.a` の関数だけだという理由によって、前述の関数は2つのグループに分割されています。

`lex` によって生成される名前で、`input`、`unput`、`main` を除くすべての外部およびスタティック名には、接頭辞 `yy` または `YY` が付きます。

使用法

`lex` 内のルールセクションではアクションのない ERE は受け入れられないことが、移植性のあるアプリケーションに警告されますが、`lex` がエラーとして検出する必要はありません。これはコンパイルまたは実行時エラーを引き起こすことがあります。

`input` の目的は、字句分析に関して、入力ストリームから文字を取り出し破棄することです。コメントの先頭を検出したら、コメント全体を破棄するという使い方が一般的です。

`lex` ユーティリティーは、`lex` ソースコードや生成された字句アナライザにおける正規表現の扱いが完全には国際化されていません。字句アナライザの実行時に、指定された環境に応じて `lex` ソース内の正規表現を解析することが理想とされますが、現在の `lex` テクノロジではこれは不可能です。さらに `lex` によって生成される字句アナライザの特徴は、ロケール固有なことが多い入力言語の字句要件に密接に結びついている必要があります。(たとえば、フランス語に使用するアナライザを作成しても、自動的にその他の言語の処理に役立つことはありません)。

使用例

例1 `lex` の使用

次の例は、Pascal に似た構文用の簡単なスキャナを実装する `lex` プログラムです。

```
%{
/* need this for the call to atof() below */
```

例1 lex の使用 (続き)

```
#include <math.h>
/* need this for printf(), fopen() and stdin below */
#include <stdio.h>
%}

DIGIT    [0-9]
ID       [a-z][a-z0-9]*
%%

{DIGIT}+  {
            printf("An integer: %s (%d)\n", yytext,
                atoi(yytext));
        }

{DIGIT}+"."{DIGIT}* {
            printf("A float: %s (%g)\n", yytext,
                atof(yytext));
        }

if|then|begin|end|procedure|function {
            printf("A keyword: %s\n", yytext);
        }

{ID}      printf("An identifier: %s\n", yytext);

"+"|"-"|"*"|"|"  printf("An operator: %s\n", yytext);

"{[^}\n]*}"      /* eat up one-line comments */

[ \t\n]+        /* eat up white space */

.              printf("Unrecognized character: %s\n", yytext);

%%

int main(int argc, char *argv[ ])
{
    ++argv, --argc; /* skip over program name */
    if (argc > 0)
        yyin = fopen(argv[0], "r");
    else
        yyin = stdin;

    yylex();
}
```

環境変数 `lex` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目 [yacc\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [regex\(5\)](#), [standards\(5\)](#)

注意事項 .l ファイルの `yyback()`、`yywrap()`、`yylock()` などのルーチンが外部 C 関数となる場合には、C++ プログラムをコンパイルするコマンド行で `__EXTERN_C__` マクロを定義する必要があります。例:

```
example% CC -D__EXTERN_C__ ... file
```

名前 `lgrpinfo` – 近傍性グループに関する情報の表示

形式 `lgrpinfo [-aceGLmrt] [-u unit] [-C | -P] lgrp ...`

`lgrpinfo -h`

`lgrpinfo -I [-c] [-G] [-C | -P] lgrp ...`

`lgrpinfo [-T] [-aceGLmr] [-u unit]`

`lgrpinfo -d device_path`

機能説明 `lgrpinfo` は、近傍性グループ (`lgroup`) の階層とその内容に関する情報を出力します。

`lgroup` は、一連の CPU およびメモリーに類似するハードウェアデバイスを表します。これらの間には少し距離 (遅延) がある場合があります。システム内のすべての `lgroup` が、`lgroup ID` と呼ばれる一意の整数によって識別されます。

`lgroup` は、もっとも近いリソースを見つけやすくするために階層にまとめられます。リーフ `lgroup` にはそれぞれ、互いにもっとも近い (ローカルの) リソースのセットが含まれます。階層内の各親 `lgroup` には、その子 `lgroup` のリソースに加えて、その次にもっとも近いリソースが含まれます。最後に、`root lgroup` には、最大待ち時間以内にあるドメイン内のすべてのリソースが含まれます。

ユニフォームメモリーアクセス (UMA) マシンは単純に、ルート `lgroup` によって表されます。非ユニフォームメモリーアクセス (NUMA) マシンは、対応する近傍性レベルを示す `lgroup` の階層によって表されます。たとえば、2つの待ち時間 (ローカルとリモート) を持つ NUMA マシンには、リーフとルートの2つのレベルから成る `lgroup` 階層があります。

すべてのアプリケーションスレッドは、「ホーム」`lgroup` を割り当てられます。システムは、スレッドに CPU またはメモリーリソースを割り当てる必要がある場合、そのスレッドのホーム `lgroup` から `lgroup` 階層を検索して、そのスレッドのホームにもっとも近い使用可能なリソースを探します。詳細は、[plgrp\(1\)](#) を参照してください。

引数がない場合、`lgrpinfo` は、システム内のすべての `lgroup` に関する一般情報を表示します。コマンド行でいずれかの `lgroup ID` が指定されている場合、このコマンドは、指定された `lgroup` に関する情報のみを表示します。表示される `lgroup` の種類や、各 `lgroup` に対して表示される正確な情報は、さまざまなオプションによって制御されます。

`lgroup` は、コマンド行で `lgroup ID` として、または特定のキーワードを使用して指定できます。「オペランド」の項を参照してください。

オプション オプションは組み合わせることができ、オプションを指定する順序は重要ではありません。小文字オプションによって、`lgroup` に関するどのような情報が表示されるかが選択されます。

引数なしで `lgrpinfo` を呼び出すことは、次の記述と同等です。

```
lgrpinfo -c -e -l -m -r -t all
```

サポートしているオプションは、次のとおりです。

- a トポロジ、メモリー、読み込み、および応答時間情報を出力します。
このオプションは次の短縮形です。

```
lgrpinfo -t -c -e -m -r -l -L
```


(ただし、`-T` も指定した場合を除く。)`-T` が指定されている場合、`-t` オプションは含まれません。
- c CPU 情報を出力します。
これはデフォルトです。
- C リスト内の各 `lgroup` をその子に置き換えます。
このオプションは、`-P` および `-T` オプションとともに使用することはできません。引数が指定されていない場合、このオプションはデフォルトで、表示される `lgroup` に適用されます。
- d *device_path* 指定された入出力デバイスにもっとも近い `lgroup` の ID を表示します。*device_path* は、デバイスパスを表す文字列です。
- e `lgroup` の平均負荷率を表示します。`lgroup` の平均負荷率は、リーフ `lgroup` の場合にのみ表示されます。
これはデフォルトです。
- G `lgroup` 階層の OS ビューを出力します。
デフォルトでは、`lgroup` 階層の呼び出し側のビューのうち、呼び出し側が使用できる内容のみ (たとえば、呼び出し側のプロセッサセット内の CPU のみ) が含まれているものが表示されます。オペレーティングシステムと呼び出し側のビューについては、[lgrp_init\(3LGRP\)](#) を参照してください。
- h 短いヘルプメッセージを出力して終了します。
- I 一致する ID のみを出力します。
このオプションはスクリプト用で、`-c`、`-G` および `-C`、または `-P` とともに使用できます。`-c` が指定されている場合は、一致するすべての `lgroup` に含まれている CPU のリストを表示します。それ以外の場合は、一致する `lgroup` の ID が表示されます。「使用例」を参照してください。

引数が指定されていない場合、このオプションは表示される `lgroup` (デフォルトでは、すべての `lgroup`) に適用されます。

-l `lgroup` の待ち時間に関する情報を表示します。

各 `lgroup` について指定された待ち時間の値はオペレーティングシステムによって定義されており、プラットフォームに固有です。この値は、実行中のシステム上の `lgroup` の相対的な比較にのみ使用できます。この値は、ハードウェアデバイス間の実際の応答時間を表しているとは限らず、異なるプラットフォーム間では当てはまらない場合もあります。

-L `lgroup` の待ち時間テーブルを表示します。 `lgroup` の待ち時間テーブルには、各 `lgroup` からほかの各 `lgroup` (それ自体を含む) への相対的な待ち時間が表示されます。

-m メモリー情報を出力します。

`-u` オプションも指定されている場合を除き、メモリーサイズは測定単位が調整され、0 から 1023 までの整数で生成されます。この数値の小数部分は 10 未満の値に対してのみ表示されます。この動作は、`ls(1)` または `df(1M)` の `-h` オプションを使用して読みやすい形式で表示する場合と同様です。

これはデフォルトです。

-P リスト内の各 `lgroup` をその親に置き換えます。

このオプションは、`-c` および `-T` オプションとともに使用することはできません。引数が指定されていない場合、このオプションは表示される `lgroup` (デフォルトでは、すべての `lgroup`) に適用されます。

-r `lgroup` のリソースに関する情報を表示します。

リソースは、各メンバー `lgroup` に CPU およびメモリーリソースが直接含まれている、`lgroup` のセットによって表されます。`-T` も指定されている場合は、中間 `lgroup` のリソースに関する情報のみが表示されます。

-t `lgroup` のトポロジに関する情報を表示します。

これはデフォルトです。

-T システムの `lgroup` トポロジをツリーとしてグラフィカルに表示します。このオプションは `-a`、`-c`、`-e`、`-G`、`-l`、`-L`、`-m`、`-r`、および `-u` オプションとともにのみ使用できます。`-r` とともに使用された場合は、中間 `lgroup` の `lgroup` リソースのみを表示します。`-T` が

-a とともに使用されると、-t オプションは省略されます。それが唯一の lgroup でないかぎり、root lgroup の情報は表示されません。

-uunits メモリー単位を指定します。単位は、b、k、m、g、t、p、または e(それぞれ、バイト、キロバイト、メガバイト、ギガバイト、テラバイト、ペタバイト、またはエクサバイトを表す)にしてください。この数値の小数部分は 10 未満の値に対してのみ表示されます。この動作は、**ls(1)** または **df(1M)** の -h オプションを使用して読みやすい形式で表示する場合と同様です。

オペランド 次のオペランドがサポートされています。

lgrp lgroup は、次のいずれかのキーワードを使用して、コマンド行で lgroup ID として指定できます。

all	すべての lgroup。 これはデフォルトです。
intermediate	すべての中間 lgroup。中間 lgroup とは、親と子を持つ lgroup のことです。
leaves	すべてのリーフ lgroup。リーフ lgroup とは、lgroup 階層内に子を持たない lgroup のことです。
root	ルート lgroup。ルート lgroup には、最大待ち時間以内にあるドメイン内のすべてのリソースが含まれており、親 lgroup は存在しません。

無効な lgroup が指定されている場合、lgrpinfo コマンドは無効な ID を示す標準エラーに関するメッセージを表示し、コマンド行で指定されたほかの lgroup の処理を続行します。指定されたどの lgroup も有効でない場合、lgrpinfo は 2 の終了ステータスで終了します。

使用例 例1 lgroup に関する情報の出力

次の例では、システム内の lgroup に関する一般情報を表示します。

この例では、システムは AMD Opteron CPU を 2 つ搭載したマシンで、1 つの CPU と 2G バイトのメモリーを持つノードが 2 つあります。これらの各ノードは、リーフ lgroup によって表されます。ルート lgroup には、マシン内のすべてのリソースが含まれます。

```
$ lgrpinfo
lgroup 0 (root):
  Children: 1 2
  CPUs: 0 1
  Memory: installed 4.0G, allocated 2.2G, free 1.8G
```

例1 lgroupに関する情報の出力 (続き)

```

    Lgroup resources: 1 2 (CPU); 1 2 (memory)
    Latency: 83
lgroup 1 (leaf):
    Children: none, Parent: 0
    CPU: 0
    Memory: installed 2.0G, allocated 1.2G, free 788M
    Lgroup resources: 1 (CPU); 1 (memory)
    Load: 0.793
    Latency: 56
lgroup 2 (leaf):
    Children: none, Parent: 0
    CPU: 1
    Memory: installed 2.0G, allocated 1017M, free 1.0G
    Lgroup resources: 2 (CPU); 2 (memory)
    Load: 0.817
    Latency: 56

```

例2 lgroupのトポロジの出力

次の例では、AMD Opteron CPU を4つ備えたマシンのlgroup トポロジツリーを出力します。

```

$ lgrpinfo -T
0
|-- 5
|  '-- 1
|-- 6
|  '-- 2
|-- 7
|  '-- 3
'-- 8
    '-- 4

```

例3 lgroupのトポロジの出力

次の例では、AMD Opteron CPU を2つ備えたマシンのlgroup トポロジツリー、リソース、メモリー、およびCPU 情報を出力します。

```

$ lgrpinfo -Ta
0
|-- 1
|  CPU: 0
|  Memory: installed 2.0G, allocated 1.2G, free 790M
|  Load: 0.274
|  Latency: 56
'-- 2
    CPU: 1

```

例3 lgroup のトポロジの出力 (続き)

```
Memory: installed 2.0G, allocated 1019M, free 1.0G
Load: 0.937
Latency: 56
```

Lgroup latencies:

```
-----
| 0 1 2
-----
0 | 83 83 83
1 | 83 56 83
2 | 83 83 56
-----
```

例4 lgroupID の出力

次の例では、ルート lgroup の子の lgroup ID を出力します。

```
$ lgrpinfo -I -C root
1 2
```

例5 CPUID の出力

次の例では、lgroup 1 内のすべての CPU の CPU ID を出力します。

```
$ lgrpinfo -c -I 1
0
```

例6 lgroup の応答時間に関する情報の出力

次の例では、lgroup の応答時間に関する情報を出力します。

```
$ lgrpinfo -l
lgroup 0 (root):
    Latency: 83
lgroup 1 (leaf):
    Latency: 56
lgroup 2 (leaf):
    Latency: 5
```

例7 指定されたデバイスにもっとも近い lgroup の ID の表示

次の例では、lgroup 2 および 6 が指定されたデバイスにもっとも近いことを示しています。

```
$ lgrpinfo -d /dev/dsk/c9t0d0s0
lgroup IDs : 2 6
```

- 終了ステータス
- 次の終了ステータスが返されます。
- 0

正常終了。
- 1

システムから `lgroup` の情報を取得できません。
- 2

指定された `lgroup` または `device_path` がすべて無効です。
- 3

構文が正しくありません。
- 属性
- 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

- 人間が読める形式の出力は「不確実」です。
- 関連項目
- [ls\(1\)](#), [plgrp\(1\)](#), [pmap\(1\)](#), [proc\(1\)](#), [ps\(1\)](#), [df\(1M\)](#), [prstat\(1M\)](#), [lgrp_init\(3LGRP\)](#), [liblgrp\(3LIB\)](#), [lgrp_cpus\(3LGRP\)](#), [lgrp_resources\(3LGRP\)](#), [proc\(4\)](#), [attributes\(5\)](#)

名前	limit, ulimit, unlimited – 現在のシェルとそのシェルから起動されたプロセスで利用できるシステムリソースの制限値を設定または取得
形式	/usr/bin/ulimit [-f] [<i>blocks</i>]
sh	ulimit [- [HS] [a cdfnstv]] ulimit [- [HS] [c d f n s t v]] <i>limit</i>
csch	limit [-h] [<i>resource</i> [<i>limit</i>]] unlimit [-h] [<i>resource</i>]
ksh88	ulimit [-HSacdfnstv] [<i>limit</i>]
ksh	ulimit [-HSacdfmnpstv] [<i>limit</i>]

機能説明

/usr/bin/ulimit	ulimit ユーティリティーは、シェルとその子プロセスに課せられる書き込み時のファイルサイズの制限値を、設定または報告します (ファイルサイズに関係なく読み取りは可能です)。適切な特権をもつプロセスのみ制限値を上げることができます。
sh	Bourne シェルの組み込み関数 <code>ulimit</code> は、リソースの強い制限値また弱い制限値を表示または設定します。これらの制限値については getrlimit(2) の説明を参照してください。 <i>limit</i> を省略すると、<code>ulimit</code> は指定されている制限値を表示します。制限値は一度にいくつでも表示できます。-a オプションは制限値すべてを表示します。 <i>limit</i> 引数を指定すると、<code>ulimit</code> は指定されたフラグに対応する制限値をその引数の値に設定します。文字列 <code>unlimited</code> は、現在の制限値 (ある場合) の削除を要求します。ユーザーはだれでも、強い制限値以下の値に弱い制限値を設定できます。ユーザーはだれでも、強い制限値を下げるできます。適切な特権を持ったユーザーだけが、強い制限値を上げたり、削除したりできます。getrlimit(2) を参照してください。 -H オプションは強い制限値を表し、-s オプションは弱い制限値を表します。どちらのオプションも指定しない場合、<code>ulimit</code> は両方の制限値を設定し、弱い制限値を表示します。 次のオプションは、制限値を表示または設定するリソースを指定します。オプションを一切指定しないと、ファイルサイズ限界値を表示または設定します。 -c 最大コアファイルサイズ (512 バイトブロック単位) -d データセグメントまたはヒープの最大サイズ (K バイト単位) -f 最大ファイルサイズ (512 バイトブロック単位) -n 最大ファイル記述子に 1 を加えたもの

- s スタックセグメントの最大サイズ (K バイト単位)
- t 最大 CPU 時間 (秒単位)
- v 仮想メモリーの最大サイズ (K バイト単位)

csh

C シェルの組み込み関数 `limit` は、現在のプロセス、またはそれが生成したすべてのプロセスについて、各プロセスが指定された *resource* を *limit* 以上消費しないよう制限します。文字列 `unlimited` は、現在の制限値 (ある場合) の削除を要求します。*limit* を省略すると、現在の制限値を出力します。*resource* を省略すると、すべての制限値を表示します。

- h 現在の制限値ではなく強い制限値を使用します。強い制限値は現在の制限値を制限します。強い制限値を上げることができるのは特権ユーザーだけです。

resource として指定できるものは次のとおりです。

<code>cputime</code>	プロセス当たりの最大 CPU 使用時間 (秒)
<code>filesize</code>	最大の単一ファイル容量。ファイルシステムのサイズと機能に制限されます。 df(1M) を参照してください。
<code>datasize</code>	K バイト単位のプロセスのヒープの最大サイズ。
<code>stacksize</code>	プロセスの最大スタックサイズ。デフォルトのスタックサイズは 2^{13} です。
<code>coredumpsize</code>	最大コアダンプのファイルサイズ。ファイルシステムのサイズに制限されます。
<code>descriptors</code>	ファイル記述子の最大数。システムで利用可能な最大制限値を調べるには sysdef(1M) コマンドを実行してください。 <code>sysdef</code> によって報告される値は 16 進数ですが、 bc(1) コマンドを使って 10 進数に変換できます。
<code>memorysize</code>	仮想記憶の最大サイズ

limit は数値で、次の単位を付加して指定することもできます。

<code>nh</code>	(<code>cputime</code> の) 時間
<code>nk</code>	<i>n</i> キロバイト。これは <code>cputime</code> を除くすべての値のデフォルト単位です。
<code>nm</code>	<i>n</i> メガバイトまたは (<code>cputime</code> の) 分
<code>mm:ss</code>	(<code>cputime</code> の) 分と秒

`unlimit` は *resource* に関する制限値を削除します。*resource* が指定されないと、すべてのリソースの制限値が削除されます。リソース名の一覧については、前述の `limit` コマンドの説明を参照してください。

- h 対応する強い制限値を削除します。これは特権ユーザーだけしか実行できません。

ksh88

Korn シェルの組み込み関数 `ulimit` は、リソースの制限を表示または設定します。使用可能なリソースの制限を次に説明します。システムによっては、次に挙げたすべての資源の制限を提供していないこともあります。`limit` を指定すると、指定したリソースの制限値が設定されます。`limit` の値は、各リソースに対応した単位(後述)の数値、または `unlimited` という文字列です。文字列 `unlimited` は、現在の制限値(ある場合)の削除を要求します。`-H` と `-S` の両フラグは、リソースに対して強い制限値と弱い制限値のどちらを設定するかを表します。強い制限値は、いったん設定したらあとで値を上げることはできません。弱い制限値は、強い制限値を超えない範囲で値を上げることが可能です。`-H` も `-S` も省略すると、指定した制限値が強い制限と弱い制限の両方に適用されます。`limit` 引数を省略すると、現在のリソース制限値が表示されます。このとき、`-H` が指定された場合を除き、表示されるのは弱い制限値です。複数のリソースを指定すると、値の前に制限するリソース名と単位とが表示されます。

- a 現在のリソース制限値をすべて表示します。
- c コアダンプのサイズをブロック (512 バイト) 単位で表します。
- d データ領域のサイズを K バイト単位で表します。
- f 子プロセスが書き込むファイルのサイズをブロック (512 バイト) 単位で表します。読み込むファイルのサイズに制限はありません。
- n ファイル記述子数に 1 を加えた値を表します。
- s スタック領域のサイズを K バイト単位で表します。
- t 各プロセスが使用する秒数 (CPU 時間) を表します。
- v 仮想記憶のサイズを K バイト単位で表します。

オプションをすべて省略すると、`-f` が指定されたものとみなします。

シェルのメモリーパラメータ

`heapsize`、`datasize`、および `stacksize` パラメータは調整可能なシステムパラメータではありません。これらを制御できるのは、シェルの起動ファイルで設定される強い制限値、または、システム規模の弱い制限値だけです。現在のバージョンの Solaris OS の場合、後者は 2^{13} バイトです。

ksh

`ulimit` は、リソースの制限値を設定または表示します。これらの制限値は、現在のプロセスと、リソースの制限値の設定後に作成された各子プロセスに適用されます。`limit` を指定した場合は、リソースの制限値が設定され、それ以外の場合は、その現在の値が標準出力に表示されます。

通常、リソースの制限値を上げるには特権が必要です。一部のシステムでは、ユーザーがリソースの制限値を下げて、あとで上げることができます。これらは、弱い制限値と呼ばれます。いったん強い制限値が設定されると、リソースを増やすことはできません。

異なるシステムで異なるリソースを指定でき、一部のシステムではリソースの制限値をどの程度上げられるかが制限されています。

limit の値は、各リソースで示されているリソースの単位によって異なります。また、*limit* に「unlimited」を指定して、そのリソースについては制限なしであることを示すことができます。

`-H` または `-s` を指定しない場合、一覧表示には `-s` が使用され、リソースの設定には `-s` と `-H` の両方が使用されます。

いずれのリソースも指定しない場合、デフォルトは `-f` です。

`ksh` における `ulimit` では、次のオプションを使用できます。

- `-a` 現在のリソースの制限値をすべて表示します。
- `-b`
- `--sbsize` ソケットバッファースizeをバイト単位で指定します。
- `-c`
- `--core` コアファイルサイズをブロック単位で指定します。
- `-d`
- `--data` データサイズをKバイト単位で指定します。
- `-f`
- `--fsize` ファイルサイズをブロック単位で指定します。
- `-H` 強い制限値を表示または設定します。
- `-L`
- `--locks` ファイルのロック数を指定します。
- `-l`
- `--memlock` ロックされたアドレス空間をKバイト単位で指定します。
- `-M`
- `--as` アドレス空間の制限値をKバイト単位で指定します。
- `-n`
- `--nofile` 開かれているファイルの数を指定します。
- `-p`
- `--pipe` パイプバッファースizeをバイト単位で指定します。


```

-m
--rss      常駐の設定サイズをKバイト単位で指定します。
-S         弱い制限値を表示または設定します。

-s
--stack    スタックサイズをKバイト単位で指定します。

-T
--threads  スレッド数を指定します。

-t
--cpu      CPU時間を秒単位で指定します。

-u
--nproc    プロセス数を指定します。

-v
--vmem     プロセスサイズをKバイト単位で指定します。

```

オプション

/usr/bin/ulimit では次のオプションがサポートされています。

```

-f      ファイルサイズの制限をブロック単位で設定 (blocks を指定しない場合は報告) します。これはデフォルト値です。 -

```

オペランド

/usr/bin/ulimit では次のオペランドがサポートされています。

```

blocks   新しくファイルサイズの制限として使用する 512 バイトごとのブロック数。

```

使用例

/usr/bin/ulimit

例1 スタックサイズを制限する

次の例は、スタックサイズを 512K バイトに制限します。

```

example% ulimit -s 512
example% ulimit -a
time(seconds)      unlimited
file(blocks)       100
data(kbytes)       523256
stack(kbytes)      512
coredump(blocks)   200
nofiles(descriptors) 64
memory(kbytes)     unlimited

```

sh/ksh88

例2 ファイル記述子の数を制限する

次のコマンドは、ファイル記述子の数を 12 に制限します。

例2 ファイル記述子の数を制限する (続き)

```
example$ ulimit -n 12
example$ ulimit -a
time(seconds)          unlimited
file(blocks)           41943
data(kbytes)           523256
stack(kbytes)          8192
coredump(blocks)       200
nofiles(descriptors)   12
vmemory(kbytes)        unlimited
```

csh

例3 コアダンプファイルのサイズを制限する

次のコマンドは、コアダンプファイルのサイズを 0K バイトに制限します。

```
example% limit coredumpsize 0
example% limit
cputime          unlimited
filesize        unlimited
datasize        523256 kbytes
stacksize       8192 kbytes
coredumpsize    0 kbytes
descriptors     64
memorysize      unlimited
```

例4 コアファイルサイズの制限を削除する

次のコマンドは、コアファイルサイズの前述の制限を削除します。

```
example% unlimited coredumpsize
example% limit
cputime          unlimited
filesize        unlimited
datasize        523256 kbytes
stacksize       8192 kbytes
coredumpsize    unlimited
descriptors     64
memorysize      unlimited
```

環境変数

ulimit の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

ulimit により、次の終了値が返されます。

0 正常終了。

>0 要求した制限値が大きいため拒否された、またはエラーが発生しました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/ulimit、csh、ksh88、sh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

ksh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	不確実

関連項目 [bc\(1\)](#), [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [df\(1M\)](#), [su\(1M\)](#), [swap\(1M\)](#), [sysdef\(1M\)](#), [getrlimit\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 [prctl\(1\)](#) などの Solaris のほかのリソース制限機能と一緒に [ulimit](#) を使用する場
合、予期しない結果が起こりうることに注意してください。 [resource_controls\(5\)](#)
を参照してください。

[resource_controls\(5\)](#) に説明されているプロジェクトリソース制御とともに
[prctl\(1\)](#) を使用して、有効な制限をいつでも正確に観察してください。

名前	line – read one line
形式	line
機能説明	The line utility copies one line (up to and including a new-line) from the standard input and writes it on the standard output. It returns an exit status of 1 on EOF and always prints at least a new-line. It is often used within shell files to read from the user's terminal.
終了ステータス	Exit status is: 0 Successful completion >0 End-of-file on input.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [sh\(1\)](#), [read\(2\)](#), [attributes\(5\)](#)

名前	list_devices – 割り当て可能なデバイスの一覧表示
形式	list_devices [-s] [-U uid] [-z zonename] [-a [-w]] -l -n -u [device] [-l -n -u] -c dev-class list_devices [-s] -d dev-type
機能説明	list_devices ユーティリティーは、システム内の割り当て可能なデバイスを指定された条件に従って一覧表示します。 デバイス、およびそのデバイスに関連付けられたすべてのデバイス特殊ファイルが一覧表示されます。デバイスの引数は省略可能で、引数がない場合はすべての関連デバイスが一覧表示されます。dev-classがある場合は、指定された dev-class に属するデバイスが一覧表示されます。デフォルトの dev-class はありません。
オプション	サポートしているオプションは、次のとおりです。 -l [-c dev-class device] 現在のプロセスに割り当て可能なデバイスに関連付けられたデバイス特殊ファイルのパス名を一覧表示します。 dev-class が指定されている場合、指定されたデバイスクラスのすべてのデバイスに関連付けられたファイルのみを一覧表示します。 device が指定されている場合、その指定されたデバイスに関連付けられたファイルのみを一覧表示します。 -n [-c dev-class device] 現在のプロセスに割り当て可能な (ただし、現時点では割り当てられていない) デバイスに関連付けられたデバイス特殊ファイルのパス名を一覧表示します。 dev-class が指定されている場合、指定されたデバイスクラスのすべてのデバイスに関連付けられたファイルのみを一覧表示します。 device が指定されている場合、その指定されたデバイスに関連付けられたファイルのみを一覧表示します。 -s サイレント。診断出力をすべて抑止します。 -u [-c dev-class device] 現在のプロセスの所有者に割り当てられているデバイスに関連付けられたデバイス特殊ファイルのパス名を一覧表示します。 dev-class が指定されている場合、指定されたデバイスクラスのすべてのデバイスに関連付けられたファイルのみを一覧表示します。 device が指定されている場合、その指定されたデバイスに関連付けられたファイルのみを一覧表示します。

-U uid `list_devices` 操作の実行中は、ユーザー ID `uid` を現在のプロセスの実際のユーザー ID の代わりに使用します。 `solaris.device.revoke` 承認を保持するユーザーだけがこのオプションを使用できます。

Trusted Extensions を使用したシステムの構成時には、次のオプションがサポートされます。

-a 承認、クリーニングプログラム、デバイスに関連付けられたラベルなどの属性を一覧表示します。

この一覧は、次の形式のように、デバイスごとに 1 行で `key=value` のペアをセミコロン (;) で区切って表示されます。

```
device=device-name;type=device-type;\
auths=auths;clean=device-exec;\
device-attributes;\
files=device-list
```

ここで、`device-attributes` は、`device_allocate(4)` の `reserved1` フィールドの内容です。このフィールドはコロン (:) で区切られます。

これらの属性とその形式については、`device_allocate(4)` を参照してください。

-a 出力には次のキーがあります。

auths	承認の一覧を指定します。値は <code>auths</code> です (<code>device_allocate(4)</code> で説明されています)。
clean	デバイスクリーンスクリプトを指定します。値は <code>device-exec</code> です (<code>device_allocate(4)</code> で説明されています)。
device	デバイス名を指定します。値は <code>device-name</code> です (<code>device_allocate(4)</code> で説明されています)。
files	デバイスのファイルパスを指定します。値は <code>device-list</code> です (<code>device_maps(4)</code> で説明されています)。
type	デバイスタイプを指定します。値は <code>device-type</code> です (<code>device_allocate(4)</code> で説明されています)。

-d デバイス割り当てによって管理されているデバイスタイプの、システム指定のデフォルト属性を表示します。`dev-type` が指定されている場合、そのデバイスタイプのためのデフォルト属性が一覧表示されます。

- w** このオプションを **-a** とともに使用すると、デバイスの現在の所有者を、キーと値のペア (*owner=value*) として一覧表示できます。*value* はデバイスの現在の所有者の *uid* です。デバイスが割り当てられていない場合、値は */FREE* となります。デバイスがエラー状態の場合、値は */ERROR* となります。このオプションも診断出力をすべて抑止します。
- z zonename** **-l** オプションとともに指定されている場合、ラベル範囲が *zonename* のラベルを含む未割り当てのデバイス、および割り当て済みのうち *zonename* と同じラベルで割り当てられているデバイスを一覧表示します。
- n** オプションとともに指定されている場合、ラベル範囲が *zonename* のラベルを含む未割り当てのデバイスのみを一覧表示します。
- u** オプションとともに指定されている場合、*zonename* と同じラベルで割り当て済みのデバイスのみを一覧表示します。

使用例

例1 すべてのデバイスの一覧表示

次の例では、呼び出し側が割り当てに使用できるすべてのデバイスを一覧表示します。

```
% list_devices -l
device: audio type: audio \
files: /dev/audio /dev/audiocctl /dev/sound/0 /dev/sound/0ctl
```

例2 すべてのデバイスの属性の一覧表示

次の例では、Trusted Extensions で構成されているシステム上で、呼び出し側が割り当てに使用できるすべてのデバイスの属性を一覧表示します。

```
% list_devices -al
device=audio1;type=audio;\
auths=solaris.device.allocate;\
clean=/etc/security/lib/audio_clean;\
minlabel=admin_low:maxlabel=admin_high;\
files=/dev/audio1 /dev/audiocctl /dev/sound/1 /dev/sound/1ctl
```

例3 デバイス所有者を含む属性の一覧表示

次の例では、Trusted Extensions で構成されているシステム上で、ユーザーに割り当てに使用できるすべてのデバイスの所有者を含む属性を一覧表示します。

```
% list_devices -auw
device=audio2;type=audio;auths=solaris.device.allocate;\
clean=/etc/security/lib/audio_clean;\
minlabel=admin_low:maxlabel=admin_high:zone=public;\
```

例3 デバイス所有者を含む属性の一覧表示 (続き)

```
owner=1234;\nfiles=/dev/audio2 /dev/audio2ctl /dev/sound/2 /dev/sound/2ctl
```

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

20

指定したデバイスのエントリが存在しません。

ほかの値

エラーが発生した。

ファイル

/etc/security/device_allocate

/etc/security/device_maps

/etc/security/dev/*

/usr/security/lib/*

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

呼び出しは不確実です。オプションは不確実です。-a および -w オプションからの出力は不確実です。ほかのすべての出力は非インタフェースです。

関連項目

[allocate\(1\)](#), [deallocate\(1\)](#), [device_allocate\(1M\)](#), [dminfo\(1M\)](#), [mkdevalloc\(1M\)](#), [mkdevmaps\(1M\)](#), [device_allocate\(4\)](#), [device_maps\(4\)](#), [attributes\(5\)](#)

Controlling Access to Devices

注意事項

このマニュアルページで説明している機能は、Solaris 監査が有効な場合にのみ使用できます。

このマニュアルページで説明している機能は、[device_allocate\(1M\)](#) サービスが有効な場合にのみ使用できます。

Trusted Extensions で構成されているシステムでは、この機能はデフォルトで有効です。

/etc/security/dev、[mkdevalloc\(1M\)](#)、および [mkdevmaps\(1M\)](#) は、Solaris オペレーティング環境の将来のリリースではサポートされない可能性があります。

名前	listusers - ユーザーログイン情報の一覧表示
形式	listusers [-g groups] [-l logins]
機能説明	このコマンドは、オプションを指定しないで実行すると、ログインでソートしたすべてのユーザーログインを一覧表示します。/etc/nsswitch.conf で指定したシステムのパスワードデータベースから、ログイン ID およびアカウントフィールドの値が出力されます。
オプション	サポートしているオプションは、次のとおりです。 -g groups グループに属しているすべてのユーザーログインを、ログインでソートして一覧表示します。複数のグループをコンマ区切りのリストとして指定できます。 -l logins ユーザーログインまたは logins で指定されたログインをログインでソートして一覧表示します。複数のログインをコンマ区切りのリストとして指定できます。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [nsswitch.conf\(4\)](#), [attributes\(5\)](#)

注意事項 ユーザーログインは、100 以上の UID を持つログインです。

 -l および -g オプションは組み合わせることができます。ユーザーログインは、選択されている複数のグループに属していても、1 回のみ表示されます。

名前	llc2_autoconfig – LLC2 構成ファイルの生成
形式	/usr/lib/llc2/llc2_autoconfig [-f]
機能説明	llc2_autoconfig ユーティリティーは、LLC2 構成ファイル (/etc/llc2/default/llc2.*) を生成するために使用されます。/etc/llc2_default/ 内に構成ファイルが存在しない場合は、システム内のすべての使用可能なインタフェースを検出し、対応するデフォルト構成ファイルを生成します。 /etc/llc2_default/ 内に既存の構成ファイルが存在する場合は、それらのファイルで定義されているインタフェースがまだ存在するかどうかを確認します。システム内に存在しない場合は、これらのファイル内の llc2_on を 0 に設定します。そのあと、システム内に新しいインタフェースが存在するかどうかを検出します。存在する場合は、それらのインタフェースの構成ファイルを生成します。
オプション	次のオプションがサポートされています。 -f /etc/llc2/default/ 内のすべての構成ファイルを消去します。次に、システム内のすべての使用可能なインタフェースを検出し、対応するデフォルト構成ファイルを生成します。このオプションは慎重に使用してください。
ファイル	/etc/llc2/default/llc2.* LLC2 構成ファイル
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/network/llc2

関連項目 [llc2_config\(1\)](#), [llc2\(4\)](#), [attributes\(5\)](#), [llc2\(7D\)](#)

名前	llc2_config - LLC2 インタフェースのパラメータの構成
形式	<code>/usr/lib/llc2/llc2_config</code> <code>[-P -U -d -q -i ppa -r ppa]</code>
機能説明	llc2_config ユーティリティは、LLC2 サブシステムを起動/停止したり、LLC2 インタフェースのパラメータを構成したりするために使用されます。
オプション	サポートしているオプションは、次のとおりです。

- d デバッグモードをオンに設定にします。追加のデバッグ情報は表示されません。
- ippa ファイル `/etc/llc2/default/llc2.ppa` を使用して、対応するインタフェースを初期化します。
- P すべての `/etc/llc2/default/llc2.*` 構成ファイルを読み取り、そのファイルで定義されているデバイスを開いて、LLC2 がこれらのデバイスを使用するために必要なストリームを設定します。これを実行する前、llc2_config -q は何も表示しません。
- q LLC2 サブシステムに照会します。LLC2 モジュールの下で使用可能なすべての PPA (物理接続点) に対して、次の例のような情報が表示されます。

PPA	State	ID	MACAddr	Type	MaxSDU	MinSDU	Mode
0	up	0000	0800208a217e	ethernet	1500	0	3

次に、表示されるフィールドについて説明します。

PPA	インタフェースの相対的な論理位置。
状態	インタフェースの状態。
up	インタフェースは初期化され、動作しています。
down	インタフェースは LLC2 ドライバによって「検出され」、ブート時の診断情報を渡して、初期化を待っています。
bad	インタフェースは LLC2 ドライバに認識されていますが、ブート時に実行された整合性検査の1つまたは複数に失敗しました。これには、割り込み要求や共有メモリー競合の検出、またはレベル0の診断の実行中に検出された障害が含まれる可能性があります。
ID	インタフェース ID。
MACAddr	インタフェースで現在有効な MAC アドレス。
Type	MAC のタイプ。現在サポートされているタイプには次のものがあります。

	csma/cd	10 メガビット Ethernet
	ethernet	Ethernet タイプデバイス
	tkn-ring	4/16 メガビットトークンリング
	fddi	100M ビット光ファイバ分散データインタフェース
MaxSDU	このインタフェース上で転送される最大サービスデータユニットサイズ。	
Mode	このインタフェースでサポートされるサービスモード。このフィールドは、 <code>/usr/include/sys/dlpi.h</code> でも定義されている、サポートされるモードのビット単位の論理和で構成されています。	
-r <i>ppa</i>	対応するインタフェースの初期化を解除します。このオプションを使用したあとに <code>-i</code> オプションを使用することによって、インタフェースに関連付けられたパラメータを変更できます。	
-U	LLC2 サブシステムによって使用されたすべてのストリームを破棄します。これは <code>-P</code> オプションの逆です。これが実行されたあと、 <code>llc2_config -q</code> は何も表示しません。	

ファイル

`/etc/llc2/default/llc2.*` LLC2 構成ファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	<code>system/network/llc2</code>

関連項目

[llc2_autoconfig\(1\)](#), [llc2\(4\)](#), [attributes\(5\)](#), [llc2\(7D\)](#)

名前	llc2_stats - LLC2 ステーション、SAP、および接続の統計情報
形式	llc2_stats <i>ppa</i> [-r] [-s <i>sap</i>] [-c <i>connection</i>]
機能説明	llc2_stats コマンドは、LLC2 ドライバのホストベース論理リンク制御クラス 2 コンポーネントから統計情報を取得するために使用されます。統計情報は、ステーション、SAP (サービスアクセスポイント)、および接続コンポーネント用に保持されます。
オプション	サポートしているオプションは、次のとおりです。 -c <i>connection</i> 目的の接続を指定します。その値は、先頭に 0x のない 16 進数表記で入力されます。 -r 指定されたカウンタを読み取ったあと、そのカウンタを 0 にリセットします。このオプションは、<i>root</i> ユーザーがこのコマンドを実行している場合にのみ有効です。 -ssap この要求の SAP を指定します。これは、先頭に 0x のない 16 進数表記で表されたシングルバイト値です。たとえば、NetBIOS sap (240 (0xf0)) は、-s f0 として入力されます。
オペランド	次のオペランドを指定できます。 <i>ppa</i> アダプタを指定するために使用される論理的な数字。PPA (物理接続点) は、最初の引数である必要があります。
使用例	例1 ステーションの統計情報 次のコマンドは、PPA 4 のステーションの統計情報を表示します。例のあとに、各フィールドの簡単な説明が示されています。 <pre>example% /usr/lib/llc2/llc2_stats 4</pre> <pre>Station values received: ppa = 0x00000004 clearFlag = 0x00 # of saps (hex) = 0x0002 saps (hex) = 02 aa state = 0x01 nullSapXidCmdRcvd = 0x00000000 nullSapXidRspSent = 0x00000000 nullSapTestCmdRcvd = 0x00000000 nullSapTestRspSent = 0x00000000 outOfState = 0x00000000 allocFail = 0x00000000 protocolError = 0x00000000</pre> 各フィールドは次のように説明されます。 <i>ppa</i> アダプタを指定するために使用される論理的な数字。

例1 ステーションの統計情報 (続き)

clearFlag	このフラグは、読み取られたあとに統計情報が0にリセットされるか(1に設定される)、または統計情報が読み取り専用であるか(0に設定される)を示します。
# of saps	現在、このステーション上でバインドされているSAPの数。
saps	LLCとそれに隣接するレイヤーの間にあるステーションのサービスアクセスポイント(SAP)の論理インタフェース値の配列。
state	ステーションコンポーネントの現在の状態を示す数字(0=停止、1=稼働中)。
nullSapXidCmdRcvd	NULLのSAPアドレス(sap=0x00)に対して受信されたXIDコマンド Protocol Data Unit (PDU)の数。
nullSapXidRspSent	NULLのSAPアドレスに対して受信されたXIDコマンドPDUに応答して送信されたXID応答PDUの数。
nullSapTestCmdRcvd	NULLのSAPアドレスに対して受信されたTESTコマンドPDUの数。
nullSapTestRspSent	NULLのSAPアドレスに対して受信されたTESTコマンドPDUに応答して送信されたTEST応答PDUの数。
outOfState	無効な状態で受信されたイベントの数。
allocFail	バッファ割り当ての失敗の数。
protocolError	LLCプロトコルエラー、つまり、不正な形式のPDUの受信や、フレームYが予期されていたときのフレームXの受信の数。

例2 SAPの統計情報

上の表示には、0x02と0xaaの2つのアクティブなSAPがあります。次の例では、SAP 02の統計情報を取得するためのコマンドと、表示されている各フィールドの簡単な説明を示しています。

example% /usr/lib/llc2/llc2_stats 4 -s 02

```
Sap values received:
ppa                = 0x00000004  clearFlag = 0x00
sap                 = 0x02
state               = 0x01
# of cons (hex)     = 0x0000000a
connections (hex)   = 0000 0001 0002 0003 0004 0005 0006 0007 0008 0009
```

例2 SAP の統計情報 (続き)

```

xidCmdSent      = 0x00000000
xidCmdRcvd      = 0x00000000
xidRspSent      = 0x00000000
xidRspRcvd      = 0x00000000
testCmdSent     = 0x00000000
testCmdRcvd     = 0x00000000
testRspSent     = 0x00000000
testRspRcvd     = 0x00000000
uiSent          = 0x00000000
uiRcvd          = 0x00000000
outOfState      = 0x00000000
allocFail       = 0x00000000
protocolError    = 0x00000000

```

各フィールドは次のように説明されます。

ppa	アダプタを指定するために使用される論理的な数字。
clearFlag	このフラグは、読み取られたあとに統計情報が0にリセットされるか(1に設定される)、または統計情報が読み取り専用であるか(0に設定される)を示します。
sap	ステーションの指定されたサービスアクセスポイント(SAP)の論理インタフェース値。
state	SAPコンポーネントの現在の状態を示す数字(0=非アクティブ、1=アクティブ)。
# of cons	このSAP上のアクティブな接続の数。
connections	アクティブな接続インデックスの配列。
xidCmdSent	送信されたXIDコマンドPDUの数(発信元のSAP=このsap)。
xidCmdRcvd	受信されたXIDコマンドPDUの数(送信先のSAP=このsap)。
xidRspSent	送信されたXID応答PDUの数(発信元のSAP=このsap)。
xidRspRcvd	受信されたXID応答PDUの数(発信元のSAP=このsap)。
testCmdSent	送信されたTESTコマンドPDUの数(発信元のSAP=このsap)。
testCmdRcvd	受信されたTESTコマンドPDUの数(送信先のSAP=このsap)。
testRspSent	送信されたTEST応答PDUの数(発信元のSAP=このsap)。
testRspRcvd	受信されたTEST応答PDUの数(発信元のSAP=このsap)。
uiSent	送信された番号なしの情報フレームの数。
uiRcvd	受信された番号なしの情報フレームの数。

例2 SAPの統計情報 (続き)

outOfState	無効な状態で受信されたイベントの数。
allocFail	バッファ割り当ての失敗の数。
protocolError	LLC プロトコルエラー、つまり、不正な形式の PDU の受信や、フレーム Y が予期されていたときのフレーム X の受信の数。

例3 接続の統計情報

この SAP には、10 個の確立された接続が関連付けられています。接続 1 の統計情報を取得するには、次のコマンドを入力します。

```
example% /usr/lib/llc2/llc2_stats 4 -s 2 -c 1
Connection values received:
ppa          = 0x0004  clearFlag    = 0x00
sap          = 0x02    con          = 0x0001  sid          = 0x0201
stateOldest  = 0x00    stateOlder   = 0x00    stateOld     = 0x01
state        = 0x08
dl_nodeaddr  = 0x0080d84008c2      dl_sap       = 0x04
flag         = 0x50    dataFlag    = 0x00    timerOn      = 0x18
vs           = 0x29    vr = 0x1e      nrRcvd       = 0x29  k = 0x14
retryCount   = 0x0000  numToBeAcked = 0x0000  numToResend  = 0x0000
macOutSave   = 0x0000  macOutDump   = 0x0000
iSent        = 0x0ba9  iRcvd        = 0x001e
frmrSent     = 0x0000  frmrRcvd     = 0x0000
rrSent       = 0x016a  rrRcvd       = 0x00c1
rnrSent      = 0x0000  rnrRcvd      = 0x06fb
rejSent      = 0x0000  rejRcvd      = 0x0000
sabmeSent    = 0x0000  sabmeRcvd    = 0x0001
uaSent       = 0x0001  uaRcvd       = 0x0000  discSent     = 0x0000
outOfState   = 0x0000  allocFail    = 0x0000  protocolError = 0x0000
localBusy    = 0x0000  remoteBusy   = 0x00b5  maxRetryFail = 0x0000
ackTimerExp  = 0x0000  pollTimerExp = 0x0000  rejTimerExp  = 0x0000
remBusyTimerExp = 0x0000
inactTimerExp = 0x0000
sendAckTimerExp = 0x0000
```

ppa	アダプタを指定するために使用される論理的な数字。
clearFlag	このフラグは、読み取られたあとに統計情報が0にリセットされるか(1に設定される)、または統計情報が読み取り専用であるか(0に設定される)を示します。
sap	ステーションの指定されたサービスアクセスポイント(SAP)の論理インタフェース値。
con	SAPの指定された接続インデックス値。

例3 接続の統計情報 (続き)

stateOldest	stateOlder の前の接続コンポーネントの状態を表す数字。
stateOlder	stateOld の前の接続コンポーネントの状態を表す数字。
stateOld	state の前の接続コンポーネントの状態を表す数字。
state	接続コンポーネントの最新の状態を表す数字。表 1 を参照してください。
sid	SAP(上位バイト)と接続インデックス(下位バイト)で構成されたステーション識別子。
dl_nodeaddr	データリンクノードアドレス。これは、送信先ノードの MAC アドレスです。
dl_sap	送信先ノードの SAP。
flag	接続コンポーネントの処理フラグ。表 3 を参照してください。
dataFlag	受信された I フレーム PDU からのデータユニットのステータスを表す数字 (0 = 破棄されていない、1 = 破棄済み、2 = REJ PDU が未処理のステータスでビジネスステータスになった)。
timerOn	タイマー動作フラグを表す数字であり、各ビットがこの接続のアクティブなタイマーを表します。タイマーの定義については、表 2 を参照してください。
vs	送信する次の I フレーム PDU のシーケンス番号。
vr	受信される次の I フレーム PDU の予期されるシーケンス番号。
nrRcvd	リモートノードによって確認された最後に送信された I フレーム PDU のシーケンス番号プラス 1。
k	転送ウィンドウサイズ。
retryCount	retryCount は、タイマーの期限切れが発生した場合は常に増分されます。これらのタイマーは、アウトバウンドフレームを保護します。
numToBeAcked	肯定応答を待っているアウトバウンド I フレームの数。
numToResend	再転送されるアウトバウンド I フレームの数。
macOutSave	使用されなくなりました。
macOutDump	使用されなくなりました。
iSent	送信された I フレームの数。

例3 接続の統計情報 (続き)

iRcvd	受信されたIフレームの数。
frmrSent	送信されたフレーム拒否 PDU (FRMR) の数。
frmrRcvd	受信されたフレーム拒否 PDU (FRMR) の数。
rrSent	送信された受信側準備完了 PDU (RR) の数。
rrRcvd	受信された受信側準備完了 PDU (RR) の数。
rnrSent	送信された受信側準備未完了 PDU (RNR) の数。
rnrRcvd	受信された受信側準備未完了 PDU (RNR) の数。
rejSent	送信された拒否 PDU (REJ) の数。
rejRcvd	受信された拒否 PDU (REJ) の数。
sabmeSent	送信された拡張非同期平衡モード設定 PDU (SABME) の数。
sabmeRcvd	受信された拡張非同期平衡モード設定 PDU (SABME) の数。
uaSent	送信された番号なしの肯定応答 PDU (UA) の数。
uaRcvd	受信された番号なしの肯定応答 PDU (UA) の数。
discSent	送信された切り離し PDU (DISC) の数。
outOfState	無効な状態で受信されたイベントの数。
allocFail	バッファ割り当ての失敗の数。
protocolError	LLC プロトコルエラー、つまり、不正な形式の PDU の受信や、フレーム Y が予期されていたときのフレーム X の受信の数。
localBusy	このコンポーネントがローカルビジー状態にあり、I フレームを受け入れることができなかった回数。
remoteBusy	リモート接続コンポーネントがビジー状態にあり、I フレームを受け入れることができなかった回数。
maxRetryFail	maxRetry に達したために発生した失敗の数。
ackTimerExp	肯定応答タイマーの期限切れの数。
pollTimerExp	ポーリングタイマーの期限切れの数。
rejTimerExp	拒否タイマーの期限切れの数。
remBusyTimerExp	リモートビジータイマーの期限切れの数。
inactTimerExp	非活動タイマーの期限切れの数。

例3 接続の統計情報 (続き)

sendAckTimerExp 送信肯定応答タイマーの期限切れの数。

表 1: LLC2 の状態	
ステーション	
~~DOWN	0x00
~~UP	0x01
SAP	
~~INACTIVE	0x00
~~ACTIVE	0x01
接続	
~~ADM	0x00
~~CONN	0x01
~~RESET_WAIT	0x02
~~RESET_CHECK	0x03
~~SETUP	0x04
~~RESET	0x05
~~D_CONN	0x06
~~ERROR	0x07
~~NORMAL	0x08
~~BUSY	0x09
~~REJECT	0x0a
~~AWAIT	0x0b
~~AWAIT_BUSY	0x0c
~~AWAIT_REJECT	0x0d

表 2: timersOn	
肯定応答	0x80
ポーリング	0x40

表 2: timersOn	
拒否	0x20
リモートビジー	0x10
非活動	0x08
肯定応答送信	0x04

表 3: LLC2 のフラグ	
P_FLAG	0x80
F_FLAG	0x40
S_FLAG	0x20
REMOTE_BUSY	0x10
RESEND_PENDING	0x08

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/network/llc2

ファイル /dev/llc2 クローンデバイス

関連項目 [attributes\(5\)](#)

注意事項 LLC2 のコンポーネント、状態、およびフラグについての詳細は、国際標準化機構 (ISO) のドキュメント「ISO 8802-2: 1994, Section 7」を参照してください。

名前	ln - ファイルへのハードリンクまたはシンボリックリンクの作成
形式	<pre> /usr/bin/ln [-fns] source_file [target] /usr/bin/ln [-fns] source_file... target /usr/xpg4/bin/ln [-fs] source_file [target] /usr/xpg4/bin/ln [-fs] source_file... target </pre>
機能説明	前述の2つの形式のうち、第1の形式の <code>ln</code> ユーティリティーは、<code>source_file</code> が示すファイル用に新しいディレクトリエントリ(リンク)を、<code>target</code> が示す宛先パスに作成します。<code>target</code> が省略されると、カレントディレクトリ内にリンクを作成します。最終オペランドが既存のディレクトリを指していなければ、この第1の形式の <code>ln</code> 呼び出しと見なされます。オペランドの数が3つ以上で、最終オペランドが既存のディレクトリではない場合、エラーとなります。 第2の形式の <code>ln</code> ユーティリティーは、<code>source_file</code> オペランドで示された各ファイル用に新しいディレクトリを、<code>target</code> が示す既存のディレクトリの宛先パスに作成します。 <code>ln</code> ユーティリティーは、ハードリンクとシンボリックリンクの両方を作成できます。ハードリンクはファイルへのポインタで、元のディレクトリエントリと区別されません。ファイルに加えられる変更は、そのファイルの参照にどの名前が使用されたかにかかわらず有効です。ハードリンクは、複数のファイルシステムにまたがることはできません。また、ディレクトリも参照することはできません。 デフォルトでは、<code>ln</code> はハードリンクを作成します。<code>source_file</code> は <code>target</code> にリンクされます。<code>target</code> がディレクトリの場合は、<code>target</code> の中に <code>source_file</code> という名前の別のファイルが作成され、元の <code>source_file</code> にリンクされます。 <code>target</code> がファイルで <code>-f</code> オプションが指定されていない場合は、<code>ln</code> は標準エラーに診断メッセージを書き込み、現在の <code>source_file</code> には何も行わず、ほかの残りの <code>source_file</code> に処理を進めます。 シンボリックリンクはファイルへの間接的なポインタで、そのディレクトリエントリにはリンク先のファイルの名前が書かれています。シンボリックリンクはファイルシステムをまたぐことができ、またディレクトリを参照することもできます。 <code>target</code> のアクセス権は <code>-l</code> を付けた <code>ls(1)</code> コマンドで表示されるアクセス権とは、異なる場合があります。<code>target</code> のアクセス権を表示する場合は、<code>ls -lL</code> を使用してください。詳細については、<code>stat(2)</code> を参照してください。 <code>/usr/bin/ln</code> <code>/usr/bin/ln</code> は、<code>target</code> のモードが書き込み禁止であった場合、そのモード (<code>chmod(1)</code> を参照) を出力し、応答を要求し、標準入力から1行を読み取ります。応答が肯定である場合、アクセス権が与えられていればリンクが行われ、それ以外の場合、コマンドは終了します。

/usr/xpg4/bin/ln

ハードリンクを作成するときや、リンク元のファイル自体がシンボリックリンクであるとき、リンク先は、シンボリックリンクのオブジェクト自体 (*source_file*) へのハードリンクではなく、シンボリックリンクによって参照されているファイルへのハードリンクになります。

オプション

次のオプションは、/usr/bin/ln と /usr/xpg4/bin/ln で指定できます。

-f *target* のモードで書き込みが禁止されていても、ユーザーに問い合わせることをしないリンクファイル。標準入力端末でない場合、デフォルトはこの設定になっています。

-s シンボリックリンクを作成します。

-s オプションに2つの引数を指定する場合、*target* に指定できるのは、既存のディレクトリまたは存在しないファイルです。*target* がすでに存在し、ディレクトリでもない場合は、エラーが返されます。ファイル名 *source_file* には、あらゆるパス名を指定でき、それは既存のものである必要はありません。既存のパス名にする場合は、ファイルとディレクトリのいずれも指定でき、また、*target* とは異なるファイルシステムにあるものを指定することもできます。*target* が既存のディレクトリである場合は、ディレクトリ *target* の中に *source_file* または *source_file* の最後のコンポーネントの名前の付いたファイルが作成されます。このファイルは、*source_file* を参照するシンボリックリンクです。*target* が存在しない場合は、*target* という名前のファイルが作成され、*source_file* を参照するシンボリックリンクになります。

-s オプションに3つ以上の引数を指定する場合は、*target* は既存のディレクトリでなければなりません。既存のディレクトリでない場合は、エラーが返されます。*source_file* ごとに、*target* の中に *source_file* の最後のコンポーネントの名前の付いたリンクが作成されます。新しい *source_file* は、それぞれ、元の *source_file* へのシンボリックリンクです。ファイルとターゲットは、異なるファイルシステムにあってもかまいません。

/usr/bin/ln

次のオプションは、/usr/bin/ln でのみ指定できます。

-n *target* が既存のファイルである場合、診断メッセージを標準エラーに書き込んで、残りの *source_file* に処理を進めます。-f オプションを使用すると、このオプションは無効になります。これは /usr/bin/ln と /usr/xpg4/bin/ln のデフォルトの動作であるため、指定しても無視されます。

オペランド

次のオペランドがサポートされています。

source_file リンクされるファイルのパス名。これは、通常ファイル、特殊ファイルのどちらでもかまいません。-s オプションが指定された場合には、*source_file* にはディレクトリを指定することもできます。

target 新しいディレクトリエントリのパス名、または新しいディレクトリエントリが作られる既存ディレクトリのパス名。

- 使用法** ファイルが2G バイト (2³¹ バイト) 以上ある場合の `ln` の動作については、[largefile\(5\)](#) を参照してください。
- 環境変数** `ln` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
- 終了ステータス** 次の終了ステータスが返されます。
- 0 指定されたファイルはすべて正常にリンクされました。
 - >0 エラーが発生した。
- 属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/ln

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

/usr/xpg4/bin/ln

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [chmod\(1\)](#), [ls\(1\)](#), [stat\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 ディレクトリへのシンボリックリンクは、予想とは異なった動作をする場合があります。シンボリックリンクに対して [ls\(1\)](#) を実行すると、指定したディレクトリ中のファイルが表示されますが、一方 `ls -l` はリンク自体の情報を表示します。

```
example% ln -s dir link
example% ls link
file1 file2 file3 file4
example% ls -l link
lrwxrwxrwx  1 user          7 Jan 11 23:27 link -> dir
```

/usr/bin/sh または /usr/bin/csh を使用して、シンボリックリンクを介したディレクトリに変更すると ([cd\(1\)](#) 参照)、ファイルシステム中の指定位置に移ることになります。つまり新たな作業ディレクトリの親は、シンボリックリンクの親ではなく、指定されたディレクトリの親になります。これは、/usr/bin/ksh または /usr/xpg4/bin/sh から、-P オプションを付けて `cd` を実行するときにも発生します。次に示す例で、最終的な作業ディレクトリは /home/user/linktest ではなく /usr である点に注意してください。

```
example% pwd
/home/user/linktest
example% ln -s /usr/tmp symlink
example% cd symlink
example% cd . .
example% pwd
/usr
```

C シェルのユーザーは、cd の代わりに C シェルの組み込みコマンドである pushd や popd を使えば、ディレクトリ移動がこのように複雑になるのを回避できます。

名前 ln – make hard or symbolic links to files

形式 `/usr/ucb/ln [-fs] filename [linkname]`
`/usr/ucb/ln [-fs] pathname... directory`

機能説明 The `/usr/ucb/ln` utility creates an additional directory entry, called a link, to a file or directory. Any number of links can be assigned to a file. The number of links does not affect other file attributes such as size, protections, data, etc.

filename is the name of the original file or directory. *linkname* is the new name to associate with the file or filename. If *linkname* is omitted, the last component of *filename* is used as the name of the link.

If the last argument is the name of a directory, symbolic links are made in that directory for each *pathname* argument; `/usr/ucb/ln` uses the last component of each *pathname* as the name of each link in the named *directory*.

A hard link (the default) is a standard directory entry just like the one made when the file was created. Hard links can only be made to existing files. Hard links cannot be made across file systems (disk partitions, mounted file systems). To remove a file, all hard links to it must be removed, including the name by which it was first created; removing the last hard link releases the inode associated with the file.

A symbolic link, made with the `-s` option, is a special directory entry that points to another named file. Symbolic links can span file systems and point to directories. In fact, you can create a symbolic link that points to a file that is currently absent from the file system; removing the file that it points to does not affect or alter the symbolic link itself.

A symbolic link to a directory behaves differently than you might expect in certain cases. While an `ls(1)` on such a link displays the files in the pointed-to directory, an `'ls -l'` displays information about the link itself:

```
example% /usr/ucb/ln -s dir link
example% ls link
file1 file2 file3 file4
example% ls -l link
lrwxrwxrwx 1 user          7 Jan 11 23:27 link -> dir
```

When you use `cd(1)` to change to a directory through a symbolic link, you wind up in the pointed-to location within the file system. This means that the parent of the new working directory is not the parent of the symbolic link, but rather, the parent of the pointed-to directory. For instance, in the following case the final working directory is `/usr` and not `/home/user/linktest`.

```
example% pwd
/home/user/linktest
example% /usr/ucb/ln -s /var/tmp symlink
example% cd symlink
```

```
example% cd . .
example% pwd
/usr
```

C shell user's can avoid any resulting navigation problems by using the pushd and popd built-in commands instead of cd.

オプション

- f Force a hard link to a directory. This option is only available to the super-user, and should be used with extreme caution.
- s Create a symbolic link or links.

使用法

See [largefile\(5\)](#) for the description of the behavior of ln when encountering files greater than or equal to 2 Gbyte (2³¹ bytes).

使用例

例1 The /usr/ucb/ln command
The commands below illustrate the effects of the different forms of the /usr/ucb/ln command:

```
example% /usr/ucb/ln file link
example% ls -F file link
file  link
example% /usr/ucb/ln -s file symlink
example% ls -F file symlink
file  symlink@
example% ls -li file link symlink
10606 -rw-r--r--  2 user          0 Jan 12 00:06 file
10606 -rw-r--r--  2 user          0 Jan 12 00:06 link
10607 lrwxrwxrwx  1 user          4 Jan 12 00:06 symlink -> file
example% /usr/ucb/ln -s nonesuch devoid
example% ls -F devoid
devoid@
example% cat devoid
devoid: No such file or directory
example% /usr/ucb/ln -s /proto/bin/* /tmp/bin
example% ls -F /proto/bin /tmp/bin
/proto/bin:
x*      y*      z*

/tmp/bin:
x@      y@      z@
```

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目	cp(1) , ls(1) , mv(1) , rm(1) , link(2) , readlink(2) , stat(2) , symlink(2) , attributes(5) , largefile(5)
注意事項	When the last argument is a directory, simple basenames should not be used for <i>pathname</i> arguments. If a basename is used, the resulting symbolic link points to itself: <pre>example% /usr/ucb/ln -s file /tmp example% ls -l /tmp/file lrwxrwxrwx 1 user 4 Jan 12 00:16 /tmp/file -> file example% cat /tmp/file /tmp/file: Too many levels of symbolic links</pre> To avoid this problem, use full pathnames, or prepend a reference to the <code>PWD</code> variable to files in the working directory: <pre>example% rm /tmp/file example% /usr/ucb/ln -s \$PWD/file /tmp lrwxrwxrwx 1 user 4 Jan 12 00:16 /tmp/file -> /home/user/subdir/file</pre>

名前	loadkeys, dumpkeys – load and dump keyboard translation tables
形式	loadkeys [<i>filename</i>] dumpkeys
機能説明	loadkeys reads the file specified by <i>filename</i>, and modifies the keyboard streams module's translation tables. If no file is specified, loadkeys loads the file: /usr/share/lib/keytables/type_<i>tt</i>/layout_<i>dd</i>, where <i>tt</i> is the value returned by the KIOCTYPE ioctl, and <i>dd</i> is the value returned by the KIOCLAYOUT ioctl (see kb(7M)). These keytable files specify only the entries that change between the specified layout and the default layout for the particular keyboard type. On self-identifying keyboards, the value returned by the KIOCLAYOUT ioctl is set from the DIP switches. dumpkeys writes the current contents of the keyboard streams module's translation tables, in the format specified by keytables(4), to the standard output.
ファイル	/usr/share/lib/keytables/layout_ <i>dd</i> default keytable files
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed

関連項目 [kbd\(1\)](#), [keytables\(4\)](#), [attributes\(5\)](#), [kb\(7M\)](#), [usbkbm\(7M\)](#)

名前	locale – ロケール固有の情報の取得
形式	<pre>locale [-a -m]</pre> <pre>locale [-ck] name...</pre>
機能説明	<code>locale</code> ユーティリティは現在のロケール環境、または、すべての公開ロケールに関する情報を標準出力に書き込みます。このにおいて、「公開ロケール」とは、アプリケーションからアクセスできる、該当する実装上で利用可能なロケールのことです。 引数を指定しないで <code>locale</code> を呼び出すと、環境変数の設定値によって決定されたロケールカテゴリごとに、現在のロケール環境が表示されます。 オペランドを指定して <code>locale</code> を呼び出すと、次のように、ロケールカテゴリでキーワードに割り当てられている値を標準出力に書き込みます。 ■ キーワード名を指定すると、指定したキーワードとそのキーワードが含まれているカテゴリが選択される。 ■ カテゴリ名を指定すると、指定したカテゴリとそのカテゴリ内のすべてのキーワードが選択される。
オプション	サポートしているオプションは、次のとおりです。 -a 利用可能なすべての公開ロケールに関する情報を標準出力に書き込みます。利用可能なロケールには、POSIX ロケールを表す <code>POSIX</code> も含まれます。 -c 選択したロケールカテゴリの名前を標準出力に書き込みます。-c オプションは、複数のカテゴリを選択したときに(たとえば、複数のキーワード名やカテゴリ名を使用したとき)、表示を読みやすくするために使用します。このオプションは、-k オプションの指定に関係なく有効です。 -k 選択したキーワードの名前と値を標準出力に書き込みます。実装によっては、一部のキーワードで値が省略されることがあります。「オペランド」を参照してください。 -m 利用できる文字マッピング(charmap)の名前を標準出力に書き込みます。localedef(1) を参照してください。
オペランド	次のオペランドを指定できます。 <i>name</i> ロケールカテゴリの名前、ロケールカテゴリ内のキーワードの名前、または予約名 <code>charmap</code>。指定したカテゴリまたはキーワードが選択され、出力されます。1つの <i>name</i> が現在のロケールでロケールカテゴリ名とキーワード名の両方を表す場合、指定しないのと同じ結果になります。それ以外の場合、カテゴリ名とキーワード名は、どちらも <i>name</i> オペランドとして任意の順序で指定できます。

使用例

例1 localeユーティリティーの例

次の例では、ロケール環境変数は次のように設定されているものとします。

```
LANG=locale_x LC_COLLATE=locale_y
```

locale は次の出力を生成します。

```
LANG=locale_x
LC_CTYPE="locale_x"
LC_NUMERIC="locale_x"
LC_TIME="locale_x"
LC_COLLATE=locale_y
LC_MONETARY="locale_x"
LC_MESSAGES="locale_x"
LC_ALL=
```

コマンド

```
LC_ALL=POSIX locale -ck decimal_point
```

このコマンドは、次の出力を生成します。

```
LC_NUMERIC
decimal_point="."
```

次の例は、locale を使用して、ユーザーの応答が肯定応答であるかどうかを調べるアプリケーションの一部を示します。

```
if printf "%s\n" "$response" | /usr/xpg4/bin/grep -Eq\
    "${(locale yesexpr)"
then
    affirmative processing goes here
else
    non-affirmative processing goes here
fi
```

環境変数

LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATHについては、[environ\(5\)](#)を参照してください。

LANG、LC_*、およびNLSPATH環境変数には、出力する現在のロケール環境が指定されていなければなりません。これらの環境変数は、-a オプションが指定されていない場合に使用されます。

終了ステータス

次の終了ステータスが返されます。

- 0 要求したすべての情報が見つかり、正常に出力されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/locale
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [localedef\(1\)](#), [attributes\(5\)](#), [charmap\(5\)](#), [environ\(5\)](#), [locale\(5\)](#), [locale_alias\(5\)](#), [standards\(5\)](#)

注意事項 LC_CTYPE または LC_CTYPE カテゴリのキーワードを指定すると、`0x00-0x7f` の範囲内の値だけが書き込まれます。

LC_COLLATE または LC_COLLATE カテゴリのキーワードを指定すると、実際の値は書き込まれません。

`locale -a` の出力で示されるロケール名は正規ロケール名に限定されます。使用できる、またサポートされているロケール名の別名については、[locale_alias\(5\)](#) を参照してください。

名前 localedef – define locale environment

形式 localedef [-c] [-C *compiler_options*] [-f *charmap*]
 [-i *sourcefile*] [-L *linker_options*] [-m *model*]
 [-u *code_set_name*] [-W *cc, arg*] [-x *extensions_file*] *localename*

機能説明

The `localedef` utility converts source definitions for locale categories into a format usable by the functions and utilities whose operational behavior is determined by the setting of the locale environment variables; see [environ\(5\)](#).

The utility reads source definitions for one or more locale categories belonging to the same locale from the file named in the `-i` option (if specified) or from standard input.

Each category source definition is identified by the corresponding environment variable name and terminated by an `END category-name` statement. The following categories are supported.

LC_CTYPE	Defines character classification and case conversion.
LC_COLLATE	Defines collation rules.
LC_MONETARY	Defines the format and symbols used in formatting of monetary information.
LC_NUMERIC	Defines the decimal delimiter, grouping and grouping symbol for non-monetary numeric editing.
LC_TIME	Defines the format and content of date and time information.
LC_MESSAGES	Defines the format and values of affirmative and negative responses.

オプション

The following options are supported:

<code>-c</code>	Creates permanent output even if warning messages have been issued.
<code>-C compiler_options</code>	Passes the <i>compiler_options</i> to the C compiler (<code>cc</code>). If more than one option is specified, then the options must be enclosed in quotes (" "). This is an old option. Use the <code>-W cc,arg</code> option instead.
<code>-f charmap</code>	Specifies the pathname of a file containing a mapping of character symbols and collating element symbols to actual character encodings. This option must be specified if symbolic names (other than collating symbols defined in a <code>collating-symbol</code> keyword) are used. If the <code>-f</code> option is not present, the default character mapping will be used.
<code>-i sourcefile</code>	The path name of a file containing the source definitions. If this option is not present, source definitions will be read from standard input.

- L *linker_options*** Passes the *linker_options* to the C compiler (cc) that follows the C source filename. If more than one option is specified, then the options must be enclosed in quotes (" ").
- This is an old option. Use the **-W cc,arg** option instead.
- m *model*** Specifies whether `localedef` will generate a 64-bit or a 32-bit locale object.
- Specify *model* as `ilp32` to generate a 32-bit locale object. Specify `lp64` to generate a 64-bit locale object. If the **-m** option is not specified, `localedef` generates a 32-bit locale object. And if no other options than **-c**, **-f**, and **-i** options are specified and if the system running `localedef` supports the 64-bit environment, `localedef` additionally generates a 64-bit locale object.
- u *code_set_name*** Specifies the name of a codeset used as the target mapping of character symbols and collating element symbols whose encoding values are defined in terms of the ISO/IEC 10646-1: 2000 standard position constant values. See NOTES.
- W cc,arg** Passes *arg* options to the C compiler. Each argument must be separated from the preceding by only a comma. A comma can be part of an argument by escaping it with an immediately preceding backslash character; the backslash is removed from the resulting argument.
- Use this option instead of the **-C** and **-L** options.
- x *extensions_file*** Specifies the name of an extension file where various `localedef` options are listed. See [locale\(5\)](#).

オペランド

The following operand is supported:

- localename*** Identifies the locale. If the name contains one or more slash characters, *localename* will be interpreted as a path name where the created locale definitions will be stored. This capability may be restricted to users with appropriate privileges. (As a consequence of specifying one *localename*, although several categories can be processed in one execution, only categories belonging to the same locale can be processed.)

Output

`localedef` creates a temporary C source file that represents the locale's data. `localedef` then calls the C compiler to compile this C source file into a shared object.

If the **-m ilp32** option is specified, `localedef` calls the C compiler for generating 32-bit objects and generates a 32-bit locale object. If the **-m lp64** option is specified, `localedef` calls the C compiler for generating 64-bit objects and generates a 64-bit locale object.

If the `-m` option is not specified, `localedef` calls the C compiler for generating 32-bit objects and generates a 32-bit locale object. If no other options than `-c`, `-f`, and `-i` options are specified and if the system running `localedef` supports the 64-bit environment, `localedef` additionally calls the C compiler for generating 64-bit objects and generates a 64-bit locale object.

If no option to the C compiler is explicitly specified using the `-W`, `-C`, or `-L` options, `localedef` calls the C compiler with appropriate C compiler options to generate a locale object or objects.

If the `-m ilp32` option is specified, `localedef` generates a 32-bit locale object named:

localename.so.version_number

If the `-m lp64` option is specified, `localedef` generates a 64-bit locale object named:

localename.so.version_number

If the `-m` option is not specified, `localedef` generates a 32-bit locale object named:

localename.so.version_number

and, if appropriate, generates a 64-bit locale object named:

64-bit_architecture_name/localename.so.version_number

The shared object for the 32-bit environment must be moved to:

/usr/lib/locale/localename/localename.so.version_number

The shared object for the 64-bit environment on SPARC must be moved to:

/usr/lib/locale/localename/sparcv9/localename.so.version_number

The shared object for the 64-bit environment on AMD64 must be moved to:

/usr/lib/locale/<localename>/amd64/<localename>.so.<version_number>

`localedef` also generates a text file named *localename* that is used for information only.

環境変数

See [environ\(5\)](#) for definitions of the following environment variables that affect the execution of `localedef`: `LANG`, `LC_ALL`, `LC_COLLATE`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

終了ステータス

The following exit values are returned:

- 0 No errors occurred and the locales were successfully created.
- 1 Warnings occurred and the locales were successfully created.

- 2 The locale specification exceeded implementation limits or the coded character set or sets used were not supported by the implementation, and no locale was created.
- 3 The capability to create new locales is not supported by the implementation.
- >3 Warnings or errors occurred and no output was created.

If an error is detected, no permanent output will be created.

ファイル

```
/usr/lib/localedef/extensions/generic_eucbc.x
    Describes what a generic EUC locale uses in the system. This file is used by default.

/usr/lib/localedef/extensions/single_byte.x
    Describes a generic single-byte file used in the system.

/usr/lib/locale/localename/localename.so.version_number
    The shared object for the 32-bit environment.

/usr/lib/locale/localename/sparcv9/localename.so.version_number
    The shared object for the 64-bit environment on SPARC.

/usr/lib/locale/<localename>/amd64/<localename>.so.<version_number>
    The shared object for the 64-bit environment on AMD64.
```

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[locale\(1\)](#), [iconv_open\(3C\)](#), [nl_langinfo\(3C\)](#), [strftime\(3C\)](#), [attributes\(5\)](#), [charmap\(5\)](#), [environ\(5\)](#), [extensions\(5\)](#), [locale\(5\)](#), [standards\(5\)](#)

警告

If warnings occur, permanent output will be created if the `-c` option was specified. The following conditions will cause warning messages to be issued:

- If a symbolic name not found in the *charmap* file is used for the descriptions of the LC_CTYPE or LC_COLLATE categories (for other categories, this will be an error conditions).
- If optional keywords not supported by the implementation are present in the source.

注意事項

When the `-u` option is used, the *code_set_name* option-argument is interpreted as a name of a codeset to which the ISO/IEC 10646-1: 2000 standard position constant values are converted. Both the ISO/IEC 10646-1: 2000 standard position constant values and other formats (decimal, hexadecimal, or octal) are valid as encoding values within the *charmap* file. The codeset can be any codeset that is supported by the [iconv_open\(3C\)](#) function on the system.

When conflicts occur between the charmap specification of *code_set_name*, *mb_cur_max*, or *mb_cur_min* and the corresponding value for the codeset represented by the *-u* option-argument *code_set_name*, the `localedef` utility fails as an error.

When conflicts occur between the charmap encoding values specified for symbolic names of characters of the portable character set and the character encoding values defined by the US-ASCII, the result is unspecified.

If a non-printable character in the charmap has a width specified that is not `-1`, `localedef` generates a warning.

名前	logger – add entries to the system log
形式	logger [-i] [-f <i>file</i>] [-p <i>priority</i>] [-t <i>tag</i>] [<i>message</i>] ...
機能説明	The logger command provides a method for adding one-line entries to the system log file from the command line. One or more <i>message</i> arguments can be given on the command line, in which case each is logged immediately. If this is unspecified, either the file indicated with -f or the standard input is added to the log. Otherwise, a <i>file</i> can be specified, in which case each line in the file is logged. If neither is specified, logger reads and logs messages on a line-by-line basis from the standard input.
オプション	The following options are supported: -f <i>file</i> Uses the contents of <i>file</i> as the message to log. -i Logs the process ID of the logger process with each line. -p <i>priority</i> Enters the message with the specified <i>priority</i>. The message priority can be specified numerically, or as a <i>facility.level</i> pair. For example, '-p local3.info' assigns the message priority to the info level in the local3 facility. The default priority is user.notice. -t <i>tag</i> Marks each line added to the log with the specified <i>tag</i>.
オペランド	The following operand is supported: <i>message</i> One of the string arguments whose contents are concatenated together, in the order specified, separated by single space characters.
使用例	例1 Examples of the logger command The following example: <pre>example% logger System rebooted</pre> logs the message 'System rebooted' to the default priority level notice to be treated by syslogd as are other messages to the facility user. The next example: <pre>example% logger -p local0.notice -t HOSTIDM -f /dev/idmc</pre> reads from the file /dev/idmc and logs each line in that file as a message with the tag 'HOSTIDM' at priority level notice to be treated by syslogd as are other messages to the facility local0.
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of logger: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.
終了ステータス	The following exit values are returned: 0 Successful completion.

>0 An error occurred.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[mailx\(1\)](#), [write\(1\)](#), [syslogd\(1M\)](#), [syslog\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	logger – add entries to the system log
形式	<code>/usr/ucb/logger [-f <i>filename</i>] [-i] [-p <i>priority</i>] [-t <i>tag</i>] mm [<i>message</i>]...</code>
機能説明	The logger utility provides a method for adding one-line entries to the system log file from the command line. One or more <i>message</i> arguments can be given on the command line, in which case each is logged immediately. If <i>message</i> is unspecified, either the file indicated with <code>-f</code> or the standard input is added to the log. Otherwise, a <i>filename</i> can be specified, in which case each line in the file is logged. If neither is specified, logger reads and logs messages on a line-by-line basis from the standard input.
オプション	The following options are supported: <code>-i</code> Log the process ID of the logger process with each line. <code>-f <i>filename</i></code> Use the contents of <i>filename</i> as the message to log. <code>-p <i>priority</i></code> Enter the message with the specified <i>priority</i>. The message priority can be specified numerically, or as a <i>facility.level</i> pair. For example, '<code>-p local3.info</code>' assigns the message priority to the info level in the local3 facility. The default priority is <code>user.notice</code>. <code>-t <i>tag</i></code> Mark each line added to the log with the specified <i>tag</i>.
使用例	例1 Logging a message The command: <pre>example% logger System rebooted</pre> will log the message 'System rebooted' to the facility at priority notice to be treated by syslogd as other messages to the facility notice are. 例2 Logging messages from a file The command: <pre>example% logger -p local0.notice -t HOSTIDM -f /dev/idmc</pre> will read from the file <code>/dev/idmc</code> and will log each line in that file as a message with the tag 'HOSTIDM' at priority notice to be treated by syslogd as other messages to the facility local0 are.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [syslogd\(1M\)](#), [syslog\(3C\)](#), [attributes\(5\)](#)

名前	login – システムへのサインオン
形式	<pre>login [-p] [-d device] [-R repository] [-s service] [-t terminal] [-u identity] [-U ruser] [-h hostname [terminal] -r hostname] [name [environ]...]</pre>
機能説明	login コマンドは、各端末セッションの開始時に使用するコマンドです。このコマンドを使用して、システムにユーザーの識別情報を提示します。login は、あるユーザーが <code>exit</code> コマンドを実行してログインシェルを終了させたあと、次のユーザーが最初に接続を確立するときに、システムによって呼び出されます。 スーパーユーザー以外は、コマンドとして login を呼び出すことはできません。 login がコマンドとして呼び出された場合には、初期コマンドインタプリタが置き換えられる必要があります。この方法で login を呼び出すには、 <pre>exec login</pre> 初期シェルから次のように入力します。C シェルと Korn シェルには、それぞれ独自の login の組み込みコマンドがあります。login の組み込みコマンドとその使用法については、ksh(1)、ksh88(1)、および csh(1) を参照してください。 引数としてユーザー名を指定しないと、login は、ユーザー名とユーザーパスワード (必要な場合) の入力を求めます。可能な場合、パスワードの入力中はエコーが抑止されます。このため、セッションの書き込み記録にパスワードが残ってしまうことはありません。 ログイン手続きの際に入力を誤ると、次のメッセージが出力され、 <pre>Login incorrect</pre> 新しいログインプロンプトが表示されます。ログイン手続きに 5 回失敗すると、これがすべて <code>/var/adm/loginlog</code> ファイルに記録され (ファイルが存在しない場合もある)、TTY 回線が切断されます。 パスワードの有効期限を設定する機能が有効になっている場合に、パスワードが有効期限を過ぎてしまうと、ユーザーはパスワードの変更を求められます (詳細は、passwd(1) を参照)。このとき、パスワードの記録場所を特定するために、<code>/etc/nsswitch.conf</code> ファイルが参照されます (nsswitch.conf(4) を参照)。パスワードの更新は、次の 5 つの形式でのみ実行できます。 ■ <code>passwd: files</code> ■ <code>passwd: files nis</code> ■ <code>passwd: compat (==> files nis)</code> いずれの形式にも合っていない場合、passwd(1) は失敗するため、システムにログインできません。一定時間内に正常にログインできなかった場合は、メッセージで通知されることなく接続が解除されることがあります。

正常にログインすると、アカウンティングファイルが更新されます。デバイスの所有者、グループ、アクセス権が `/etc/logindevperm` ファイルの内容に従って設定され、最終ログイン時間が表示されます ([logindevperm\(4\)](#) を参照)。

ユーザー ID、グループ ID、補助グループリスト、作業用ディレクトリは初期化され、コマンドインタプリタが起動されます。

基本環境は、次のように初期設定されます。

```
HOME=your-login-directory
LOGNAME=your-login-name
PATH=/usr/bin:
SHELL=last-field-of-passwd-entry
MAIL=/var/mail/
TZ=timezone-specification
```

Bourne シェルや Korn シェルでログインする場合は、それぞれ `/etc/profile` と `$HOME/.profile` が実行されます (ファイルが存在しない場合もある)。

ksh Korn シェルの場合は、対話型シェルが、`/etc/ksh.kshrc` と、ENV 環境変数で指定されたファイルを順に実行します。`$ENV` が設定されていない場合、これはデフォルトで `$HOME/.kshrc` になります。ksh および `/usr/xpg4/bin/sh` Korn シェルの場合、対話型シェルが、`$ENV` (デフォルトなし) によって指定されたファイルを実行します。

一方、C シェルでログインする場合は、`/etc/.login`、`$HOME/.cshrc`、`$HOME/.login` が実行されます。デフォルトの `/etc/profile` ファイルと `/etc/.login` ファイルは、ディスク容量の確認 ([quota\(1M\)](#) 参照)、`/etc/motd` の表示、およびメールの確認を行います。`$HOME/.hushlogin` ファイルが存在する場合、メッセージは何も表示されません。コマンドインタプリタ名は、`-(ダッシュ)` のあとにインタプリタのパス名の最後の構成要素を付けたものになります (例: `-sh`)。

パスワードファイルの `login-shell` フィールド ([passwd\(4\)](#) を参照) が空であれば、デフォルト時のコマンドインタプリタ `/usr/bin/sh` が使用されます。このフィールドがアスタリスク (*) の場合は、指定のディレクトリがルートディレクトリになります。この時点で、独自のルート構造を持つ新しいレベルで `login` が再実行されます。

実行時または `login` がログイン名を要求したときに `login` に引数を追加すると、環境を拡張したり変更したりできます。この引数は、`xxx` または `xxx=yyy` という形式で指定します。`=` (等号) を使用しない形式の引数は、次のように環境設定内に配置します。

```
Ln=xxx
```

このとき、`n` は 0 から始まる通し番号になります。この数値は、新しい変数名が必要になるたびに大きくなります。`=` (等号) を使用した形式の変数は、そのまま環境設定に取り込まれます。すでに設定が環境に取り込まれている場合に新しい変数を指定すると、前の値が新しい設定値に置き換えられます。

ただし、例外が2つあります。変数 `PATH` と変数 `SHELL` は変更できません。これは、制限付きのシェル環境にログインしているユーザーが、子プロセスとして制限のない新しいシェルを生成するのを防ぐためです。`login` は、単純な単一文字引用規則を認識します。文字の前に `\` (バックスラッシュ) を付けることにより、スペース文字やタブなどの文字も使用できます。

`login` に `-p` フラグを付けることによって、現在の環境を引き渡すこともできます。このフラグにより、現在定義されているすべての環境変数が、可能であれば新しい環境に引き渡されます。ただし、このオプションを指定しても、前述の環境変数の制限を回避することはできません。変数が両方の方法で引き渡された場合、`login` の引数に指定した環境変数が優先されます。

ルートによるリモートログインを可能にするには、`/etc/default/login` ファイルを編集し `CONSOLE=/dev/console` エントリの前に `#` (ポンド記号) を挿入してください。「ファイル」を参照してください。

セキュリティー ファイル (`passwd(4)` および `shadow(4)`) ネームサービスまたは `ldap` ネームサービスのアカウントの場合、`enableShadowUpdate true` を構成すると、ログイン試行が構成済みの値以上連続して失敗した場合にアカウントを自動的にロックするように構成できます。`ldapclient(1M)`、`user_attr(4)`、`policy.conf(4)`、および `pam_unix_auth(5)` を参照してください。

`login` コマンドは `pam(3PAM)` を使って、認証、アカウント管理、セッション管理、パスワード管理を行います。PAM 構成ポリシーは、`/etc/pam.conf` または `/etc/pam.d/login` に記述されており、`login` で使用するモジュールを指定します。次に、`pam.conf` ファイルの抜粋を示します。UNIX 認証、アカウント管理、セッション管理モジュールを使用する `login` コマンドのエントリが指定されています。

```
login auth      required pam_authtok_get.so.1
login auth      required pam_dhkeys.so.1
login auth      required pam_unix_auth.so.1
login auth      required pam_dial_auth.so.1
```

```
login account   requisite pam_roles.so.1
login account   required pam_unix_account.so.1
```

```
login session   required pam_unix_session.so.1
```

`/etc/pam.d/` 内の同等の PAM 構成では、`/etc/pam.d/login` 内に次のエントリが含まれます:

```
auth      required pam_authtok_get.so.1
auth      required pam_dhkeys.so.1
auth      required pam_unix_auth.so.1
auth      required pam_dial_auth.so.1
account   requisite pam_roles.so.1
account   required pam_unix_account.so.1
```

```
session required pam_unix_session.so.1
```

通常、`/etc/pam.conf` 内のパスワード管理スタックは次のようになります:

```
other password required pam_dhkeys.so.1
other password requisite pam_authtok_get.so.1
other password requisite pam_authtok_check.so.1
other password required pam_authtok_store.so.1
```

`/etc/pam.conf` および `/etc/pam.d/service` に PAM サービスのエントリがない場合は、`/etc/pam.conf` 内の「その他の」サービスのエントリが使用されます。`/etc/pam.conf` に「その他の」サービスのエントリが一切存在しない場合、`/etc/pam.d/other` 内のエントリが使用されます。認証モジュールが複数含まれている場合は、複数のパスワードが必要になることがあります。

`rlogind` や `telnetd` で `login` が呼び出されるときに、PAM が使用するサービス名はそれぞれ `rlogin`、`telnet` になります。

オプション

サポートしているオプションは、次のとおりです。

`-d device`

`login` は、デバイスオプション `device` を受け付けます。`device` は、`login` が動作する TTY ポートのパス名とみなされます。デバイスオプションを使用すると、`login` が `ttynam(3C)` を呼び出す必要がなくなるため、`login` のパフォーマンスの向上が期待できます。`-d` オプションは、ユーザー ID と実効ユーザー ID が `root` であるユーザーのみが使用できます。その他から `-d` が使用されると、`login` は何もせずに終了します。

`-h hostname [ terminal ]`

[in.telnetd\(1M\)](#) がリモートホストと端末タイプに関する情報を渡すために使用します。

`-h` オプションの 2 番目の引数としての端末タイプをハイフン (-) で始めてはいけません。

`-p`

ログインシェルに環境変数を渡すために使用します。

`-r hostname`

[in.rlogind\(1M\)](#) がリモートホストに関する情報を渡すために使用します。

`-R repository`

使用する PAM リポジトリを指定します。PAM リポジトリを使用するには、`-u` オプションを使用して(下記を参照)、PAM に「identity」情報を伝える必要があります。「identity」情報を伝えない場合、その PAM リポジトリは使用されません。

-s service

使用する PAM サービス名を指定します。通常、この引数は必ずしも必要ではなく、代替の PAM サービス名を指定するときだけに使用します。たとえば、Kerberos 認証 telnet プロセスの場合は「ktelnet」です。

-u identity

認証の対象となるユーザーに関連する「identity」文字列を指定します。通常、そのユーザーの UNIX ログイン名とは異なります。Kerberos 認証ログインセッションの場合、そのユーザーに関連する Kerberos 主体名になります。

-U ruser

rlogin 接続のリモート側でログインするユーザー名を指定します。[in.rlogind\(1M\)](#) が Kerberos 認証モードで動作している場合、このデーモンは、login を呼び出す前に、端末とリモートユーザー名の情報を処理します。したがって、このコマンド行パラメータを使用して、「ruser」データを指定する必要があります。通常、Kerberos 認証されていない rlogin では、リモートユーザーの情報は login デーモンがクライアントから読み取ります。

終了ステータス 次の終了ステータスが返されます。

0

正常終了。

0 以外

エラー。

ファイル

`$HOME/.cshrc`

各 csh の初期設定コマンド。

`$HOME/.hushlogin`

ログインメッセージの抑制。

`$HOME/.kshrc`

`$ENV` が未設定の場合の、対話型の ksh ユーザーのコマンドで、`/etc/ksh.kshrc` のあとに実行される。

`$HOME/.login`

csh ユーザーのログインコマンド。

`$HOME/.profile`

sh および ksh ユーザーのログインコマンド。

`$HOME/.rhosts`

信頼できるホスト名とユーザー名の組み合わせのプライベートリスト。

`/etc/.login`

システム全体の csh のログインコマンド。

`/etc/issue`

項目またはプロジェクトの識別情報。

`/etc/ksh.kshrc`

対話型 ksh のシステム全体のコマンド。

`/etc/logindevperm`

ログインベースのデバイスアクセス権。

`/etc/motd`

本日のメッセージ。

`/etc/nologin`

マシンのシャットダウン中にログインしようとするユーザーへのメッセージ。

`/etc/passwd`

パスワードファイル。

`/etc/profile`

システム全体の sh と ksh のログインコマンド。

`/etc/shadow`

ユーザーの暗号化パスワードのリスト。

`/usr/bin/sh`

ユーザーのデフォルトのコマンドインタプリタ。

`/var/adm/lastlog`

最後にログインした時間。

`/var/adm/loginlog`

失敗したログイン試行の記録。

`/var/adm/utmpx`

アカウンティング。

`/var/adm/wtmp`

アカウンティング。

`/var/mail/your-name`

ユーザー *your-name* 用のメールボックス。

`/etc/default/login`

このファイル内で次のフラグのデフォルト値を設定可能。このファイルにはデフォルト値がコメントとして指定されている (たとえば、`TIMEZONE=EST5EDT`)。

TIMEZONE

シェルの TZ 環境変数を設定 ([environ\(5\)](#) を参照)。

HZ

シェルの HZ 環境変数を設定。

ULIMIT

ログインのファイルサイズの制限を設定。設定はディスクブロック単位で、デフォルトはゼロ (制限なし)。

CONSOLE

設定されていれば、ルートはそのデバイスにのみログイン可能。ただし、[rsh\(1\)](#) によるリモートコマンドの実行は可能。ルートでログインできるようにするには、この行をコメントにします。

PASSREQ

ログインに NULL 以外のパスワードが必要かどうかを指定。

ALTSHELL

ログイン時に SHELL 環境変数を設定するかどうかを指定。

PATH

初期シェルの PATH 変数を設定。

SUPATH

ルート用初期シェルの PATH 変数を設定。

TIMEOUT

ログインセッションを終了するまでの待ち時間を秒単位 (0 から 900 の間) で設定。

UMASK

初期シェルのファイル作成モードマスクを設定。[umask\(1\)](#) を参照。

SYSLOG

LOG_NOTICE レベルでのルートによるログインすべてと LOG_CRIT レベルでの失敗したログイン試行を記録するのに [syslog\(3C\)](#) の LOG_AUTH を使うかどうかを指定。

DISABLETIME

存在し、かつ、ゼロより大きい場合、ログイン試行が RETRIES 回だけ失敗したあと、あるいは、PAM フレームワークが PAM_ABORT を戻したあとに login が待つ秒数。デフォルトは 20 秒。最小は 0 秒。最大はない。

SLEEPTIME

存在する場合、ログイン失敗メッセージを画面に表示するまでに待つ秒数を設定。これは PAM_ABORT 以外のログイン障害用。RETRIES に到達していない場合、あるいは、PAM フレームワークが PAM_MAXTRIES 回だけ戻されている場合、別のログイン試行が許可されます。デフォルトは 4 秒。最小は 0 秒。最大は 5 秒。

[su\(1M\)](#) と [sulogin\(1M\)](#) は両方とも SLEEPTIME の値の影響を受けます。

RETRIES

ログインを再試行する回数を設定 ([pam\(3PAM\)](#) を参照)。デフォルトは 5。最大の再試行回数は 15。自動ロックが構成されている場合 (前述の「セキュリティ」節を参照)、そのアカウントはロックされ、login は終了します。自動ロックが構成されていない場合、そのアカウントはロックされずに、login が終了します。

SYSLOG_FAILED_LOGINS

何回ログインに失敗したら、[syslog\(3C\)](#) LOG_NOTICE 機能によってログインに失敗したことを通知するメッセージが記録されるかを指定します。たとえば、変数の値が0に設定されている場合、loginは失敗したログインをすべて記録します。

/etc/default/login に記載されているフラグのうち、[sshd\(1M\)](#) では次のものが使われます:

- TIMEZONE
- HZ
- ALTSHELL
- PATH
- SUPATH
- CONSOLE
- PASSREQ
- UMASK
- ULIMIT
- RETRIES
- SYSLOG_AFTER_FAILED_LOGINS

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [csh\(1\)](#), [exit\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [mail\(1\)](#), [mailx\(1\)](#), [newgrp\(1\)](#), [passwd\(1\)](#), [rlogin\(1\)](#), [rsh\(1\)](#), [sh\(1\)](#), [shell_builtins\(1\)](#), [telnet\(1\)](#), [umask\(1\)](#), [in.rlogind\(1M\)](#), [in.telnetd\(1M\)](#), [user_attr\(4\)](#) and [policy.conf\(4\)](#), [logins\(1M\)](#), [quota\(1M\)](#), [sshd\(1M\)](#), [su\(1M\)](#), [sulogin\(1M\)](#), [syslogd\(1M\)](#), [useradd\(1M\)](#), [userdel\(1M\)](#), [pam\(3PAM\)](#), [rcmd\(3SOCKET\)](#), [syslog\(3C\)](#), [ttyname\(3C\)](#), [auth_attr\(4\)](#), [exec_attr\(4\)](#), [hosts.equiv\(4\)](#), [issue\(4\)](#), [logindevperm\(4\)](#), [loginlog\(4\)](#), [nologin\(4\)](#), [nsswitch.conf\(4\)](#), [pam.conf\(4\)](#), [passwd\(4\)](#), [policy.conf\(4\)](#), [profile\(4\)](#), [shadow\(4\)](#), [user_attr\(4\)](#), [utmpx\(4\)](#), [wtmpx\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [pam_unix_account\(5\)](#), [pam_unix_auth\(5\)](#), [pam_unix_session\(5\)](#), [pam_authtok_check\(5\)](#), [pam_authtok_get\(5\)](#), [pam_authtok_store\(5\)](#), [pam_dhkeys\(5\)](#), [pam_passwd_auth\(5\)](#), [termio\(7I\)](#)

診断 Login incorrect
ユーザー名またはパスワードが一致しません。

Not on system console
ルートでのログインが拒否されました。/etc/default/login の CONSOLE 設定を確認してください。

No directory! Logging in with home=/

[passwd\(4\)](#) データベースに指定されているユーザーのホームディレクトリが見つからないか、アクセス権が正しくありません。システム管理者に問い合わせてください。

No shell

[passwd\(4\)](#) データベースに指定されているシェルを実行できません。システム管理者に問い合わせてください。

NO LOGINS: System going down in *N* minutes

マシンがシャットダウンの処理中であり、ログインが無効になっています。

警告

ユーザー ID が 76695844 よりも大きいユーザーには、パスワードの有効期限が適用されません。また、このユーザーの最終ログイン時間も記録されません。

ルートによるログインを不能にするために `CONSOLE` を使用する場合は、ルートによるリモートコマンドの実行も不能にする必要があります。詳細については、[rsh\(1\)](#)、[rcmd\(3SOCKET\)](#)、および [hosts.equiv\(4\)](#) を参照してください。

名前	logname – ユーザーのログイン名の応答
形式	/usr/bin/logname
機能説明	logname は、ユーザーのログイン名を標準出力に書き込みます。ログイン名は getlogin(3C) 関数によって返される文字列です。getlogin() が正常に文字列を返さない場合、呼び出し側プロセスの実際のユーザー ID に対応する名前が代わりに使用されます。
環境変数	logname の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
終了ステータス	次のエラー値が返されます。 0 正常終了。 >0 エラーが発生した。
ファイル	/etc/profile ログイン時のユーザー環境 /var/adm/utmpx ユーザーとアカウントिंगの情報
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [env\(1\)](#), [login\(1\)](#), [getlogin\(3C\)](#), [utmpx\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 logout – ログインのセッションから抜け出すためのシェル組み込み関数

形式

cs logout

機能説明

cs ログインシェルを終了します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [cs\(1\)](#), [login\(1\)](#), [attributes\(5\)](#)

名前	look – find words in the system dictionary or lines in a sorted list
形式	<code>/usr/bin/look [-d] [-f] [-tc] string [filename]</code>
機能説明	The look command consults a sorted <i>filename</i> and prints all lines that begin with <i>string</i>. If no <i>filename</i> is specified, look uses <code>/usr/share/lib/dict/words</code> with collating sequence <code>-df</code>. look limits the length of a word to search for to 256 characters.
オプション	<code>-d</code> Dictionary order. Only letters, digits, TAB and SPACE characters are used in comparisons. <code>-f</code> Fold case. Upper case letters are not distinguished from lower case in comparisons. <code>-tc</code> Set termination character. All characters to the right of <i>c</i> in <i>string</i> are ignored.
ファイル	<code>/usr/share/lib/dict/words</code> spelling list
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/extended-system-utilities

関連項目 [grep\(1\)](#), [sort\(1\)](#), [attributes\(5\)](#)

名前	lookbib – find references in a bibliographic database				
形式	lookbib <i>database</i>				
機能説明	A bibliographic reference is a set of lines, constituting fields of bibliographic information. Each field starts on a line beginning with a '%', followed by a key-letter, then a blank, and finally the contents of the field, which may continue until the next line starting with '%'. The lookbib utility uses an inverted index made by <code>indxbib</code> to find sets of bibliographic references. It reads keywords typed after the '>' prompt on the terminal, and retrieves records containing all these keywords. If nothing matches, nothing is returned except another '>' prompt. It is possible to search multiple databases, as long as they have a common index made by indxbib(1). In that case, only the first argument given to <code>indxbib</code> is specified to <code>lookbib</code>. If <code>lookbib</code> does not find the index files (the <code>.i[abc]</code> files), it looks for a reference file with the same name as the argument, without the suffixes. It creates a file with a <code>.ig</code> suffix, suitable for use with <code>fgrep</code> (see grep(1)). <code>lookbib</code> then uses this <code>fgrep</code> file to find references. This method is simpler to use, but the <code>.ig</code> file is slower to use than the <code>.i[abc]</code> files, and does not allow the use of multiple reference files.				
ファイル	<code>x.ia</code> <code>x.ib</code> <code>x.ic</code> index files <code>x.ig</code> reference file				
属性	See attributes(5) for descriptions of the following attributes:				
	<table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>text/doctools</td></tr> </tbody> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	text/doctools
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	text/doctools				
関連項目	addbib(1) , grep(1) , indxbib(1) , refer(1) , roffbib(1) , sortbib(1) , attributes(5)				
使用上の留意点	Probably all dates should be indexed, since many disciplines refer to literature written in the 1800s or earlier.				

名前

形式

機能説明

ファイル

属性

関連項目

注意事項

lorder – find ordering relation for an object or library archive

lorder *filename*...

The input is one or more object or library archive *filenames* (see [ar\(1\)](#)). The standard output is a list of pairs of object file or archive member names; the first file of the pair refers to external identifiers defined in the second. The output may be processed by [tsort\(1\)](#) to find an ordering of a library suitable for one-pass access by ld. Note that the link editor ld is capable of multiple passes over an archive in the portable archive format (see [ar.h\(3HEAD\)](#)) and does not require that lorder be used when building an archive. The usage of the lorder command may, however, allow for a more efficient access of the archive during the link edit process.

The following example builds a new library from existing .o files.

```
ar -cr library `lorder *.o | tsort`
```

TMPDIR/*symref temporary files

TMPDIR/*symdef temporary files

TMPDIR usually /var/tmp but can be redefined by setting the environment variable TMPDIR. See `tempnam()` in [tmpnam\(3C\)](#).

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities

[ar\(1\)](#), [ld\(1\)](#), [tsort\(1\)](#), [tmpnam\(3C\)](#), [ar.h\(3HEAD\)](#), [attributes\(5\)](#)

lorder will accept as input any object or archive file, regardless of its suffix, provided there is more than one input file. If there is but a single input file, its suffix must be .o.

The length of the filename for TMPDIR is limited to whatever sed allows.

名前 ls - ディレクトリの内容を一覧表示

形式

```

/usr/bin/ls [-aAbcCdEfFghHiklLmnopqrRsStuUvwVx1@]
            [-/ c | -/v] [% atime | ctime | mtime | all]
            [--block-size size] [--color[=when]] [--file-type]
            [--si] [--time-style style] [file]...

/usr/xpg4/bin/ls [-aAbcCdEfFghHiklLmnopqrRsStuUvwVx1@]
                [-/ c | -/v] [% atime | ctime | mtime | all]
                [--block-size size] [--color[=when]] [--file-type]
                [--si] [--time-style style] [file]...

/usr/xpg6/bin/ls [-aAbcCdEfFghHiklLmnopqrRsStuUvwVx1@]
                [-/ c | -/v] [% atime | ctime | mtime | all]
                [--block-size size] [--color[=when]] [--file-type]
                [--si] [--time-style style] [file]...

```

機能説明 *file* オペランドがディレクトリの場合、ls は、そのディレクトリの内容を出力します。*file* が通常ファイルの場合、ls は、そのファイル名と要求されたほかの情報を出力します。デフォルトでは、アルファベット順にソートして出力します。引数をまったく指定しないと、現在のディレクトリ(.)の内容を出力します。引数を複数指定すると、引数は最初に適切にソートされますが、ファイル引数がディレクトリとその内容より先に処理されます。

出力形式には、主に3種類あります。端末に対するデフォルトの出力形式は、複数カラムで、ソートしたエントリを縦方向に並べて表示します。**-1** オプションを指定すると、単一のカラムで出力します。**-m** オプションを指定すると、ストリーム出力形式で表示します。**-C**、**-x**、**-m** オプションの出力形式を決定するために、ls は環境変数 **COLUMNS** を使用して、1 行に出力できる文字数を調べます。この環境変数が設定されていない場合は、環境変数 **TERM** に基づいて、[terminfo\(4\)](#) データベースが使用され、カラム数が決定されます。この情報が得られない場合は、カラム数は 80 カラムと見なされます。**-w** オプションを使用する場合、引数はその他のカラム幅よりも優先されます。

-e、**-E**、**-g**、**-l**、**-n**、**-o**、**-v**、**-V**、**-@** のいずれかのオプションが有効になっている場合、出力されるモードは 11 文字から成ります。最初の文字は次のいずれかです。

d エントリがディレクトリの場合

D エントリが **door** の場合

l エントリがシンボリックリンクの場合

b エントリがブロック型特殊ファイルの場合

- c エントリが文字型特殊ファイルの場合
- p エントリがFIFO(または「名前付きパイプ」)特殊ファイルの場合。
- P エントリがイベントポートの場合
- S エントリがAF_UNIXアドレスファミリのソケットの場合。
- エントリが通常ファイルの場合

残りの9文字は、3文字ずつの3組に分けられます。最初の組は、所有者へのアクセス権を示します。次の組は、同一のグループに属するほかのユーザーへのアクセス権を示します。最後の組は、その他のユーザーへのアクセス権を示します。各組においては、3文字がそれぞれ、ファイルの読み取り、書き込み、実行権を示します。ディレクトリの場合は、実行権は、指定したファイルの有無についてディレクトリを検索するアクセス権を意味します。アクセス権に続く文字は、ACL、拡張属性のいずれかを示します。ファイルに拡張属性が関連付けられており、かつ `-@` オプションが有効になっている場合、この文字は `@` になります。それ以外の場合、意味のあるACLがファイルに関連付けられている場合にはこの文字はプラス記号 `(+)` 文字になり、それ以外の場合はスペース文字になります。

`-/` または `-%` (あるいはその両方) が有効になっている場合、ファイルシステムで拡張システム属性がサポートされていれば、拡張システム属性が出力されます。次のように表示されます。

```
$ls -/ c file
-rw-r--r--  1 root    root          0 May 10 14:17 file
              {AHRSadim-u}

$ls -/ v file
-rw-r--r--  1 root    root          0 May 10 14:17 file
              {archive,hidden,readonly,system,appendonly\
              nodump,immutable, av_modified,\
              noav_quarantined,nounlink}

$ls -l -% all file
-rw-r--r--  1 root    root          0 May 10 14:17 file
              timestamp: atime    Jun 25 12:56:44 2007
              timestamp: ctime    May 10 14:20:23 2007
              timestamp: mtime    May 10 14:17:56 2007
              timestamp: crtime    May 10 14:17:56 2007
```

詳細は、`-/` および `-%` オプションの説明を参照してください。

`ls -l` (ロング形式出力) は、POSIX ロケールで次のように出力します。


```
-rwxrwxrwx+ 1 smith dev 10876 May 16 9:42 part2
```

右から左へ見ていくと、現在のディレクトリには `part2` というファイルが1つあるのがわかります。次に、ファイルの内容を最後に修正したのは 5 月 16 日午前 9 時 42 分です。ファイルには、10,876 文字 (バイト) が含まれています。ファイルの所有者またはユーザーは、グループ `dev` (おそらく *development* を示す) に属しており、ログイン名は `smith` です。この例で 1 と示されている数字は、`part2` というファイルへのリンクの数を示します ([cp\(1\)](#) を参照)。プラス記号は、ファイルに関連する ACL があることを示します。-@ オプションが指定されている場合、拡張属性が存在していると ACL は無効になり、プラス記号はアット記号 (@) に置き換えられます。さらにダッシュと文字によって、ユーザー、グループ、その他のユーザーが `part2` に対する読み取り、書き込み、実行権を持っていることが示されています。

実行権を表す `x` は、各組の 3 番目の位置に置かれます。3 番目の位置に `-` がある場合は、実行権を与えられていないことを示します。

各アクセス権の意味は、次のとおりです。

- `r`
ファイルが読み取り可能
- `w`
ファイルが書き込み可能
- `x`
ファイルが実行可能
- `-`
指定されたアクセス権は与えられていない
- `s`
セットユーザー ID ビットまたはセットグループ ID ビットがオンで、対応するユーザーまたはグループ実行権ビットもオン
- `S`
不定ビット状態 (セットユーザー ID ビットまたはセットグループ ID ビットがオンで、ユーザーまたはグループの実行権ビットがオフ)。グループのアクセス権の場合、通常ファイル以外のファイルだけに適用される
- `t`
1000 (8 進数) ビット、つまり、スティッキービットがオン ([chmod\(1\)](#) を参照) で、実行権ビットがオン
- `T`
1000 ビットがオンで、実行権ビットがオフ (不定ビット状態)
- `l`
アクセス中に強制ロックが発生する (通常ファイルについて、セットグループ ID ビットがオンで、グループ実行権ビットがオフ)

/usr/bin/ls

/usr/xpg4/bin/ls およ
び /usr/xpg6/bin/ls

L

アクセス中に強制ロックが発生する (通常ファイルについて、セットグループ ID ビットがオンで、グループ実行権ビットがオフ)

ユーザーやグループのアクセス権では、3 番目の位置に x または - 以外の文字が置かれる場合があります。セット ID ビットの状態が、ユーザーまたはグループのいずれのものであるかを参照して、この位置に s または S が置かれることもあります。実行中に、ユーザーのユーザー ID をファイルのユーザー ID と同じと見なすこの機能は、たとえば、スーパーユーザーとして起動したユーザーが、ログイン中に普通にログインした場合のユーザー ID を必要とするときなどに使用されます。

グループへのアクセス権においては、3 番目の位置に l が置かれることがあります。l は、強制的なファイルおよびレコードのロックを示しています。これはアクセス中にその他のファイルに対する読み取りまたは書き込み権をロックするファイルの機能があることを示しています。

その他のユーザーへのアクセス権においては、3 番目の位置に t または T が置かれることがあります。これらは、スティッキービットと実行権の状態を示しています。

オプション

サポートしているオプションは、次のとおりです。

/usr/bin/ls、/usr/xpg4/bin/ls および /usr/xpg6/bin/ls のオプションは、ls の 3 つのすべてのバージョンで指定できます。

-a

--all

すべてのエントリを出力します。つまり、ドット (.) で始まるエントリも出力します。

-A

--almost-all

ドット (.) で始まるものも含めたすべてのエントリを一覧表示しますが、作業用ディレクトリ (.) および親ディレクトリ (..) は例外です。

-b

--escape

出力できない文字を 8 進 \ddd の出力形式で強制的に出力します。

-B

--ignore-backups

チルド (~) で終わるファイルを表示しません。

-C

(-t オプションによる) ソートや (-l または -n オプションによる) 出力において、i ノードを最後に修正した時間 (ファイル作成、モード変更など) を使用します。

-C

複数カラム形式の出力で、エントリを縦方向にソートします。これは、デフォルトの出力形式です。

- d**
引数がディレクトリの場合、(内容ではなく)その名前だけを出力します。**-l**とともに使用すれば、ディレクトリのステータスを知ることができます。
- e**
-lと似ていますが、時間を秒まで表示します。また、時間の形式は、年数にかかわらず、すべてのファイルに対して1つだけで、その形式はPOSIX ロケールの場合、*mmm dd hh:mm:ss yyyy* です。
- E**
-lと似ていますが、時間をナノ秒まで表示します。また、時間の形式は、年数にかかわらず、すべてのファイルに対して、*yyyy-mm-ddhh:mm:ss.nnnnnnnnn* (ISO 8601 形式) の1つだけです。
- また、このオプションを指定すると、UTCからのオフセットをISO 8601 標準形式 (*+hhmm* または *-hhmm*) で表示します。オフセットが確認できない場合は何も表示しません。オフセットは、現在のタイムゾーンにおけるファイルの表示日時の時点で有効な標準オフセットまたは代替オフセットを反映しています。
- f**
強制的に、各引数をディレクトリと解釈し、各引数のディレクトリで見つかる名前を出力します。このオプションは、**-l**、**-t**、**-s**、**-S**、および **-r** を無視し、**-a** が指定されたものと見なします。エントリは、ディレクトリ内での順序で出力されます。
- F**
- classify**
特定のファイルタイプの後ろに、ファイルタイプを示すシンボルを追加します。次のシンボルが使用されます。
- / ディレクトリ
 - > Door ファイル
 - | 名前付きパイプ (FIFO)
 - @ シンボリックリンク
 - = ソケット
 - * 実行可能ファイル
- g**
所有者名が出力されない点を除いて、**-l** と同じです。

-h
--human-readable
すべてのサイズは縮小され、読みやすい形式で表示されます。たとえば、14K、234M、2.7G、3.0T などのようになります。縮尺は、1024 を除数として行われます。末尾の **--si** または **-h** オプションは、使用される除数を決定します。

-H
--dereference-command-line
引数がディレクトリを参照するシンボリックリンクの場合、リンク自身ではなく、そのリンクが参照するディレクトリのファイル情報およびファイルタイプを評価します。ただし、表示される名前はリンクの名前であり、参照されるディレクトリの名前ではありません。

-i
--inode
各ファイルについて、i ノード番号を第 1 カラムに出力します。

-k
すべてのサイズがキロバイト単位で出力されます。**--block-size=1024** と同等です。

-l
各ファイルについて、モード、ACL 表示、リンクの数、所有者名、グループ名、サイズ(バイト単位)、最終修正時間をロング形式で出力します。ファイルが特殊ファイルの場合、サイズフィールドにはメジャーデバイス番号とマイナーデバイス番号が入ります。最終修正時間が 6 か月よりも前である場合には、POSIX ロケールでは「月・日・年」の形式で表示されます。LC_TIME ロケールカテゴリが POSIX ロケールに設定されていない場合には、異なった形式で時間フィールドが表示されます。最終修正時間が 6 か月以内である場合には、「月・日・時」の形式で表示されます。ファイルがシンボリックリンクの場合は、ファイル名を出力したあとに、→ に続いて参照されるファイルのパス名が出力されます。

-L
--dereference
引数がシンボリックリンクの場合、リンク自身ではなく、そのリンクが参照するファイルまたはディレクトリのファイル情報およびファイルタイプを評価します。しかし、表示される名前はリンクの名前であり、参照されるファイルまたはディレクトリの名前ではありません。

-m
ストリーム形式出力を指定します。ファイルはコンマで区切られ横方向に出力されます。

-n
--numeric-uid-gid
所有者名とグループ名の代わりに、所有者のユーザー ID とグループのグループ ID が出力されることを除き、**-l** と同じです。

- o
- no-group
グループ名が出力されない点を除き、-l と同じです。
- p
ファイルがディレクトリの場合、各ファイル名のあとにスラッシュ (/) を付けます。
- q
- hide-control-chars
ファイル名中の出力できない文字を疑問符記号 (?) として出力します。
- r
- reverse
必要に応じて、アルファベットの逆順、古い順、またはファイルサイズの小さい順で取得するために、ソート順序を逆にします。
- R
- recursive
サブディレクトリの内容を再帰的に出力します。
- s
- size
各エントリについて、間接ブロックを含むブロック数を出力します
- S
ファイルサイズ (逆順) でソートします。ファイルサイズが同じ場合は、単に名前だけではなくファイル名 (アルファベット順) でソートします。
- t
ファイル名ではなく、タイムスタンプ (新しい順) でソートします。デフォルトは最終修正時間です -c、-u、および -% を参照してください。
- u
(-t オプションによる) ソートまたは (-l オプションによる) 出力において、最終修正時間の代わりに、最終アクセス時間を使用します。
- U
出力はソートされません。
- v
- l とほぼ同じですが、-l 出力のほかに冗長な ACL 情報が表示される点が異なります。ACL を持たないファイルやディレクトリの場合でも、ACL 情報が表示されます。
- V
-l とほぼ同じですが、-l 出力のあとにコンパクトな ACL 情報が表示される点が異なります。

-v オプションを適用できるのは、Solaris の ZFS ファイルシステムなど、NFSv4 ACL をサポートするファイルシステムに対してだけです。

ACL の表示形式は次のとおりです。

entry_type : *permissions* : *inheritance_flags* : *access_type*

entry_type に表示されるのは、次のいずれかです。

user:username

username に対する追加のユーザーアクセス。

group:groupname

グループ *groupname* に対する追加のグループアクセス。

owner@

ファイルの所有者。

group@

ファイルのグループ所有者。

everyone@

すべてのユーザーのアクセス。ファイルの所有者とグループ所有者も含みます。これは、POSIX の「other」クラスと等価ではありません。

-v、-V のいずれかのオプションを使用した場合、NFSv4 ACL モデルでサポートされている次のアクセス権が表示されます。

read_data (r)

ファイルのデータを読み取る権限。

list_directory (r)

ディレクトリの内容を表示するためのアクセス権。

write_data (w)

ファイルのデータを変更する権限。ファイルのオフセット範囲内の任意の場所。

add_file (w)

ディレクトリに新しいファイルを追加するためのアクセス権。

append_data (p)

ファイルのデータを変更する機能。ただし、変更できるのは EOF 以降のみです。

add_subdirectory (p)

ディレクトリにサブディレクトリを作成する権限。

read_xattr (R)

ファイルの拡張属性を読み取る機能。

write_xattr (W)

拡張属性を作成したり拡張属性ディレクトリに書き込んだりする機能。

`execute(x)`

ファイルを実行する権限。

`read_attributes(a)`

ファイルの基本属性 (ACL 以外) を読み取る機能。

`write_attributes(A)`

ファイルの基本属性 (ACL 以外) を変更するためのアクセス権。

`delete(d)`

ファイルを削除するためのアクセス権。

`delete_child(D)`

ディレクトリ内のファイルを削除する権限。

`read_acl(c)`

ファイルの ACL を読み取る権限。

`write_acl(C)`

ファイルの ACL を書き込む権限。

`write_owner(o)`

ファイルの所有者を変更する権限。

`synchronize(s)`

同期付きの読み取り権と書き込み権を使ってサーバー上のファイルにローカルアクセスする権限。

-

アクセス権が許可されていない。

-v、-V のいずれかのオプションを使用した場合、NFSv4 ACL モデルでサポートされている次の継承フラグが表示されます。

`file_inherit(f)`

新たに作成されたすべてのファイルに継承します。

`dir_inherit(d)`

新たに作成されたすべてのディレクトリに継承します。

`inherit_only(i)`

ディレクトリに対して指定された場合、そのディレクトリには適用せず、新たに作成されたファイルやディレクトリだけに適用します。このフラグを使用する際には、`file_inherit` または `dir_inherit`、あるいはその両方も指定する必要があります。

`no_propagate(n)`

ディレクトリ内のオブジェクトには ACL エントリを継承しますが、レベルが 1 つ下がった時点でその継承を停止すべきであることを示します。このフラグを使用する際には、`file_inherit` または `dir_inherit`、あるいはその両方も指定する必要があります。

successful_access (S)

正常にアクセスしたときに、アラームまたは監査記録を開始するかどうかを指定します。監査/アラーム ACE タイプと同時に使用されます。

failed_access (F)

アクセスに失敗したときに、アラームまたは監査記録を開始するかどうかを指定します。監査/アラーム ACE タイプと同時に使用されます。

inherited (I)

ACE が継承されました。

-

アクセス権は付与されていません。

access_type に表示されるのは、次のいずれかのタイプです。

alarm アラームをトリガーするアクセス権を指定するアクセス権フィールド。

allow 許可されたアクセス権を表すアクセス権フィールド。

audit 監査するアクセス権を指定するアクセス権フィールド。

deny 拒否されたアクセス権を表すアクセス権フィールド。

例:

```
$ ls -dV /sandbox/dir.1
drwxr-xr-x+ 2 root      root          2 Jan 17 15:09 dir.1
      user:marks:r-----:fd-----:allow
      owner@:-----:-----:deny
      owner@:rwxp---A-W-Co:-----:allow
      group@:-w-p-----:-----:deny
      group@:r-x-----:-----:allow
      everyone@:-w-p---A-W-Co:-----:deny
      everyone@:r-x---a-R-c-s:-----:allow

$
|||||+ inherited access
|||||+ failed access
|||||+--success access
|||||+-- no propagate
|||||+--- inherit only
|||||+---- directory inherit
|||||+---- file inherit
|||||
|||||+ sync
|||||+-- change owner
|||||+-- write ACL
|||||+--- read ACL
|||||+---- write extended attributes
```



```

|||||||+----- read extended attributes
|||||||+----- write attributes
||||||+----- read attributes
|||||+----- delete child
||||+----- delete
|||+----- append
||+----- execute
|+----- write data
|+----- read data

```

-w cols

--width cols

カラム幅を強制的に *cols* にする複数カラム形式出力。

-x

複数カラム形式出力で、エントリを縦方向ではなく横方向にソートします。

-l

1 行の出力につき 1 つのエントリを出力します。

-@

-l と似ていますが、拡張属性が存在していると ACL は無効になります。拡張属性を持つファイルのアクセス権ビットの後ろにはアット記号 (@) が表示されます。

-/

-/ オプションでは、**c** (コンパクトモード) と **v** (詳細モード) の 2 つのオプション引数がサポートされています。**-l** と同様に、長い形式のリストを表示します。さらに、基盤となるファイルシステムで拡張システム属性が完全にサポートされている場合は、ファイルに関連付けられた拡張システム属性も表示されます。

appendonly

オフセット EOF でのみファイルを変更できます。EOF 以外の場所でファイルを変更しようとする、EPERM エラーが発生します。

archive

最後にバックアップが作成されたときから、ファイルが変更されたかどうかを指定します。ファイルの変更時間 (mtime) が変更されるたびに、**archive** 属性が設定されます。

av_modified

ファイルの内容やサイズが変更されたりファイルの名前が変更されたりするたびに、ZFS はウイルス対策属性を設定します。

av_quarantined

ウイルス対策ソフトウェアがファイルに検疫済みのマークを付けます。

crttime

ファイル作成時のタイムスタンプ。

hidden

ファイルを非表示とマークを付けます。

immutable

ファイルの内容が変更されないようにします。アクセス時間の更新を除く、すべてのメタデータの変更も防ぎます。ディレクトリにある場合は、ディレクトリ内のファイルの削除および作成もできません。**immutable**とマークが付けられたファイルやディレクトリの内容を変更しようとする、**EPERM**エラーが発生します。**immutable**とマークが付けられたファイルの属性(アクセス時間、および適切な特権がある場合 **immutable** を除く)を変更しようすると、**EPERM**エラーが発生します。

nodump

Solaris システムには、この属性専用のセマンティクスはありません。

nounlink

ファイルが削除されないようにします。ディレクトリでは、この属性はディレクトリの内容への変更も防ぎます。つまり、ディレクトリ内のファイルの削除または名前の変更はできません。**nounlink**とマークが付けられたファイルおよびディレクトリのリンクを解除したり、名前を変更したりしようとすると、**errno EPERM**が返されます。

readonly

ファイルに **readonly** とマークを付けます。ファイルが **readonly** とマークが付けられると、ファイルの内容データを変更できなくなります。ファイルのその他のメタデータは変更できます。

sparse

この属性はユーザーおよびアプリケーションで使用可能であり、ファイルが疎であるものとして解釈できることを示します。ファイルが実際に疎であるかどうかは示さず、Solaris オペレーティングシステムに特別なセマンティクスはありません。ファイルの長さがゼロに切り詰められると、**sparse** 属性はクリアされます。

system

Solaris システムには、この属性専用のセマンティクスはありません。

コンパクトモード (-/c) で使用される表示文字は、次のとおりです。

Attribute Name	Display
archive	A
hidden	H
readonly	R
system	S
appendonly	a
nodump	d
immutable	i
av_modified	m
av_quarantined	q

```
sparse          s
nounlink        u
```

詳細モード (*/v*) の表示では、属性の完全名が設定されていれば使用され、設定されていなければ接頭辞として 'no' が付いた名前が使用されます。

属性名 `crttime` およびその他のすべてのタイムスタンプは、オプション `-%` に各タイムスタンプオプション引数および `all` オプション引数を付けて処理されます。表示位置は次のとおりです: 詳細モード (*-/v*) の表示では、属性の完全名が設定されていれば使用され、設定されていなければ接頭辞として `no` が付いた名前が使用されます。属性名 `crttime` およびその他のすべてのタイムスタンプは、オプション `-%` に各タイムスタンプオプション引数および `all` オプション引数を付けて処理されます。

表示位置は次のとおりです。

```
{|}|}|}|}|}|}
|}|}|}|}|}|}|+ s (sparse)
|}|}|}|}|}|}|+ O (offline)
|}|}|}|}|}|}|+ u (nounlink)
|}|}|}|}|}|}|+ q (av_quarantined)
|}|}|}|}|}|}|+ m (av_modified)
|}|}|}|}|}|}|+ i (immutable)
|}|}|}|}|}|}|+ d (nodump)
|}|}|}|}|}|}|+ a (appendonly)
|}|}|}|}|}|}|+ S (system)
|}|}|}|}|}|}|+ R (readonly)
|}|}|}|}|}|}|+ H (hidden)
|}|}|}|}|}|}|+ A (archive)
```

```
-% atime | crttime | ctime | mtime | all
```

`atime`

`-u` と同等です。

`crttime`

ソートまたは出力にファイルの作成時間を使用します。

`ctime`

`-c` と同等です。

`mtime`

ソートまたは出力にファイル内容の最終変更時間を使用します。

拡張システム属性がサポートされていない場合、ユーザーがファイルに対する `read` 特権を持っていない場合、または `crttime` 拡張属性が設定されていない場合、`crttime` は `mtime` の同義語と見なされます。

オプション引数 `-all` が指定されている場合、`-atime`、`-ctime`、`-mtime`、および拡張システム属性をサポートするファイルシステムでは `-crttime` (作成時間) を含むすべ

ての使用可能なタイムスタンプが出力されます。オプション `-%all` を指定しても、タイムスタンプが長形式で表示されることには変わりなく、ソートも影響を受けません。

`--block-size size`

サイズの倍数でサイズを表示します。サイズは接尾辞 `YyZzEePpTtGgMmKk` のいずれかを付けて指定できます。さらに、`B` を末尾に配置すると、2 乗ではなく 10 乗を指定できます。たとえば、次のように指定します。`10mB` は `100000000` バイトのブロックを意味し、`10m` は `10*2^20` `-- 10485760` `--` バイトのブロックを意味します。これは、`-h` オプションと同時に指定できません。

`--color [=when]`

`--colour [=when]`

色対応の端末で色を使用してファイル名を表示します。`when` は、色出力を表示するタイミングを決定するオプション引数です。

`when` に指定可能な値は、次のとおりです。

`always`

`yes`

`force`

常に、色を使用します。

`auto`

`tty`

`if-tty`

端末が存在する場合に、色を使用します。

`no`

`never`

`none`

色を使用しません。これはデフォルトです。

出力色を制御する方法については、このマニュアルページの色出力セクションを参照してください。

`--file-type`

`-F` オプションと同様に、タイプに従ってファイルの後ろに接尾辞を表示しますが、実行可能ファイルに `*` が付加される点が異なります。

`--si`

`-h` オプションと同様に、人間が読みやすい形式でサイズを表示しますが、値が 1024 ではなく 1000 で繰り返し除算される点が異なります。末尾の `--si` または `-h` オプションは、使用される除数を決定します。

`--time-style style`

指定された形式を使用して時間を表示します。拡張属性 (`-%`) で表示される時間は、この影響を受けません。

style に指定可能な値は、次のとおりです。

full-iso

-E と同等です。

long-iso

すべてのファイルを YYYY-MM-DD HH:MM 形式で表示します。

iso

古いファイルを YYYY-MM-DD 形式、新しいファイルを MM-DD HH:MM 形式で表示します。

locale

古いファイルと新しいファイルでデフォルトのロケール形式を使用します。これはデフォルトです。

+FORMAT

カスタム形式を使用します。値は、[strftime\(3C\)](#) で説明したものと同じです。文字列に NEWLINE が表示される場合、1 番目の行が古いファイルに使用され、2 番目の行が新しいファイルに使用されます。それ以外の場合は、指定された形式がすべてのファイルで使用されます。

/usr/bin/ls

-F

ディレクトリの末尾にスラッシュ (/) を、door の末尾に大記号 (>) を、実行可能ファイルの末尾にアスタリスク (*) を、FIFO の末尾に縦線 (|) を、シンボリックリンクの末尾に「at」記号 (@) を、AF_UNIX アドレスファミリソケットの末尾に等号 (=) を、それぞれ付加します。これらは、オペランドとして指定したシンボリックリンクのあとに表示されます。

--file-type

実行可能ファイルを除いて、-F と同様にエントリにマークします。実行可能ファイルはマークされません。これらは、オペランドとして指定したシンボリックリンクのあとに表示されます。

次のオプションは、同時に指定できません。このような矛盾する組み合わせでオプションを指定してもエラーにはなりません。-c と -l (小文字のエル)、-m と -l (小文字のエル)、-x と -l (小文字のエル)、-@ と -l (小文字のエル)。-l オプションは、各ペアに指定されたその他のオプションよりも優先されます。

次のオプションは、同時に指定できません。このような矛盾するグループでオプションを指定してもエラーにはなりません。-c と -l (数字の l)、-H と -L、-c と -u、-e と -E、-t と -S。特定のタイムスタンプを指定する最後のオプション (-c、-u、-% atime、-% ctime、-% ctime、および -% mtime) は、ソートまたは長形式リストで使用するタイムスタンプを決定します。最後のオプション (-t、-S、または -u) は、ソート動作を決定します。

/usr/xpg4/bin/ls

-F

ディレクトリの末尾にスラッシュ (/) を、door の末尾に大記号 (>) を、実行可能ファイルの末尾にアスタリスク (*) を、FIFO の末尾に縦線 (|) を、シンボリックリンクの末尾に「at」記号 (@) を、AF_UNIX アドレスファミリソケットの末尾に等号 (=) を、それぞれ付加します。これらは、オペランドとして指定したシンボリックリンクのあとに表示されます。

--file-type

実行可能ファイルを除いて、-F と同様にエントリにマークします。実行可能ファイルはマークされません。これらは、オペランドとして指定したシンボリックリンクのあとに表示されます。

次のオプションは、同時に指定できません。このような矛盾するグループでオプションを指定してもエラーにはなりません。-c と -l (小文字のエル)、-m と -l (小文字のエル)、-x と -l (小文字のエル)、-@ と -l (小文字のエル)、-c と -l (数字の 1)、-H と -L、-c と -u、-e と -E、-t と -S と -U。特定のタイムスタンプを指定する最後のオプション (-c、-u、-% atime、-% ctime、-% ctime、および -% mtime) は、ソートまたは長形式リストで使用されるタイムスタンプを決定します。最後のオプション (-t、-S、または -U) は、ソート動作を決定します。

/usr/xpg6/bin/ls

-F

ディレクトリの末尾にスラッシュ (/) を、door の末尾に大記号 (>) を、実行可能ファイルの末尾にアスタリスク (*) を、FIFO の末尾に縦線 (|) を、シンボリックリンクの末尾に「at」記号 (@) を、AF_UNIX アドレスファミリソケットの末尾に等号 (=) を、それぞれ付加します。これらは、-H オプションまたは -L オプションを指定しないかぎり、オペランドとして指定したシンボリックリンクの後に表示されません。

--file-type

実行可能ファイルを除いて、-F と同様にエントリにマークします。実行可能ファイルはマークされません。これらは、-H オプションまたは -L オプションを指定しないかぎり、オペランドとして指定したシンボリックリンクの後に表示されません。

次のオプションは、同時に指定できません。このような矛盾する組み合わせでオプションを指定してもエラーにはなりません。-c と -l (小文字のエル)、m と -l (小文字のエル)、-x と -l (小文字のエル)、-@ と -l (小文字のエル)、-c と -l (数字の 1)、-H と -L、-c と -u、および -e と -E、-t と -S と -U。特定のタイムスタンプを指定する最後のオプション (-c、-u、-% atime、-% ctime、-% ctime、および -% mtime) は、ソートまたは長形式リストで使用されるタイムスタンプを決定します。最後のオプション (-t、-S、または -U) は、ソート動作を決定します。

オペランド

次のオペランドを指定できます。

file

内容を印刷するファイルのパス名。ここで指定したファイルが見つからない場合、標準エラー出力に診断メッセージが書き込まれます。

色出力

色出力が有効になっている場合、環境変数 `LS_COLORS` がチェックされます。これが存在する場合、ファイル名の表示に使用される色が、その内容を使用して制御されます。設定されていない場合、デフォルトの色一覧が使用されます。`LS_COLORS` の形式は、属性仕様をコロンで区切ったリストです。各属性仕様の形式は、次のとおりです。

filespec=attr[;attr...]

filespec は、**.SUFFIX* (**.jar* や **.z* など) の形式、または次のファイルタイプのいずれかです。

no	通常ファイル
fi	標準ファイル
di	ディレクトリ
ln	シンボリックリンク
pi	FIFO (名前付きパイプ)
so	ソケット
do	Door ファイル
bd	ブロック型デバイス
cd	文字デバイス
ex	実行ビット (user、group、または other) セット
po	イベントポート
st	スティキービットセット
or	孤立したシンボリックリンク
sg	setgid バイナリ
su	setuid バイナリ
ow	その他のユーザ書き込み可能
tw	スティキービットとその他のユーザー書き込み可能

attr は色をセミコロンで区切ったリストであり、最終出力の色を決定する際に組み合わせる属性を表示します。*attr* 値は、任意に組み合わせて指定できます。指定可能な *attr* 値は、次のとおりです。

00	すべての属性をオフにします (デフォルト端末の色)。
01	テキストを太字で表示します。
04	テキストに下線を付けて表示します。

08 隠しテキストを使用して表示します。

39 前景をデフォルト端末の色に設定します。

49 前景をデフォルト端末の色に設定します。

ファイル仕様には、1つ以上の属性を指定する必要があります。

もっとも限定された一致を選択し、ファイル接尾辞から開始して、ファイルタイプの一致が見つかるまで継続すると、適切な色コードが選択されます。no (通常ファイル) タイプは任意のファイルに一致します。

使用法

ファイルが2G バイト (2^{31} バイト) 以上ある場合の `ls` の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 ファイルのアクセス権の表示

次の例は、特定のファイルに関する詳細情報を表示する方法を示したものです。

```
% ls -l file.1
-rw-r--r--  1 gozer    staff      206663 Mar 14 10:15 file.1
```

上記のアクセス権文字列 (-rw-r--r--) から、ファイルの所有者が読み取り権と書き込み権を、所有者のグループが読み取り権を、その他のユーザーが読み取り権を、それぞれ持っていることがわかります。

次の例は、特定のディレクトリに関する詳細情報を表示する方法を示したものです。

```
% ls -ld test.dir
drwxr-xr-x  2 gozer    staff      2 Mar 14 10:17 test.dir
```

前述のアクセス権文字列 (drwxr-xr-x) から、ディレクトリの所有者が読み取り権、書き込み権、および検索権を、所有者のグループが読み取り権と検索権を、その他のユーザーが読み取り権と検索権を、それぞれ持っていることがわかります。

ファイルのアクセス権を表示する別の例を、次に示します。

```
% ls -l file.2
-rw-rwl---  1 gozer    staff      206663 Mar 14 10:47 file.2
```

前述のアクセス権文字列 (-rw-rwl---) から、ファイルの所有者が読み取り権と書き込み権を、所有者のグループが読み取り権と書き込み権を、それぞれ持っていることがわかります。また、アクセス中にファイルがロックされる可能性のあることもわかります。

例2 ファイルやディレクトリの ACL 情報を表示する

次の例は、特定の ZFS ファイルの冗長な ACL 情報を表示する方法を示したものです。

```
% ls -v file.1
-rw-r--r--  1 marks    staff      206663 Mar 14 10:15 file.1
0:owner@:execute:deny
1:owner@:read_data/write_data/append_data/write_xattr/write_attributes
  /write_acl/write_owner:allow
2:group@:write_data/append_data/execute:deny
```

例2 ファイルやディレクトリのACL情報を表示する (続き)

```
3:group@:read_data:allow
4:everyone@:write_data/append_data/write_xattr/execute/write_attributes
  /write_acl/write_owner:deny
5:everyone@:read_data/read_xattr/read_attributes/read_acl/synchronize
  :allow
```

次の例は、特定の ZFS ディレクトリのコンパクトな ACL 情報を表示する方法を示したものです。

```
% ls -dV test.dir
drwxr-xr-x  2 marks      staff          2 Mar 14 10:17 test.dir
owner@:-----:-----:deny
owner@:rwxp---A-W-Co:-----:allow
group@:-w-p-----:-----:deny
group@:r-x-----:-----:allow
everyone@:-w-p---A-W-Co:-----:deny
everyone@:r-x---a-R-c--s:-----:allow
```

次の例は、UFS ファイルの ACL 情報を一覧表示する際の `ls -v` の動作を示したものです。

```
$ ls -v file.3
-rw-r--r--  1 root      root          2703 Mar 14 10:59 file.3
  0:user::rw-
  1:group::r--          #effective:r--
  2:mask:r--
  3:other:r--
```

例3 全ファイルの名前を出力

次の例は、現在のディレクトリ中にあるすべてのファイルの名前を出力します。通常は名前が出力されない、先頭にドット `.` が付いているファイルの名前も出力します。

```
example% ls -a
```

例4 ファイルの情報を出力する

次の例では、ファイルの情報を提供します。

```
example% ls -aisn
```

このコマンドは、ドットで始まるものを含む、すべてのファイルについて (a)、*i* 番号 — ファイルに対応する *i* ノードのメモリーアドレス — をカラムの左側に (i)、ファイルのサイズ (ブロック単位) を *i* 番号の右のカラムに (s) 出力します。さ

例4 ファイルの情報を出力する (続き)

らに、コマンドによる出力形式は、ファイルに関連する UID 番号 (ユーザー名の代わり) と GID 番号 (グループ名の代わり) が出力されるロング出力形式の数値バージョン (n) になります。

ディレクトリ内のファイルのサイズを出力する際、間接ブロックを含むブロックの総数を出力します。

例5 拡張システム属性の情報の提供

```
example% ls -/ c file      (extended system attribute in compact mode)
-rw-r--r--  1 root    root          0 May 10 14:17 file
               {AHRSadim-u}
```

この例では、av_quarantined は設定されません。

```
example% ls -/ v file (extended system attribute in verbose mode)
-rw-r--r--  1 root    root          0 May 10 14:17 file
               {archive,hidden,readonly,system,appendonly\
               nodump,immutable,av_modified,\
               noav_quarantined,nounlink}
```

```
example% ls -/ v file      (no extended system attribute)
-rw-r--r--  1 root    staff         0 May 16 14:48 file
               {}
```

```
example% ls -/ c file      (extended system attribute
                           supported file system)

-rw-r--r--  1 root staff          3 Jun  4 22:04 file
               {A-----m--}
```

拡張システム属性がサポートされるファイルでは、archive および av_modified 属性はデフォルトで設定されます。

```
example% ls -/ c -%crttime file

-rw-r--r--  root    root          0 May 10 14:17 file
               {AHRSadim-u}
```

この例では、作成時間としてタイムスタンプが表示されます。

```
example% ls -l -%all file
-rw-r--r--  1 root    root          0 May 10 14:17  file
               timestamp: atime   Jun 14 08:47:37 2007
               timestamp: ctime   May 10 14:20:23 2007
               timestamp: mtime   May 10 14:17:56 2007
```

timestamp: crtime May 10 14:17:56 2007

example% ls -%crtime -tl file*

```
-rw-r--r--  1 foo      staff      3 Jun  4 22:04 file1
-rw-r--r--  1 root     root       0 May 10 14:17 file
-rw-r--r--  1 foo      staff      0 May  9 13:49 file.1
```

この例では、ファイルは作成時間でソートされます。

環境変数

ls の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_TIME、LC_MESSAGES、NLSPATH、およびTZ。

COLUMNS	複数のテキストカラムからなる出力を生成する場合に、ユーザーが希望するカラム幅の値を指定します。この変数の値が10進整数を示す文字列の場合、ls は書き出すパス名テキストカラムの数を、その幅の値を元に計算します (-c の説明を参照)。COLUMNS が設定されていない、または値が不適当な場合には、カラム幅として 80 が用いられます。どのディレクトリのファイル名を出力する場合でも、選択されたカラム幅は一定です。複数テキストカラムを出力する際、ファイル名は長すぎても切り捨てられません。
LS_COLORS	色出力の出力時に使用される色スキームを決定します。これを設定せずに色出力が指定されている場合は、デフォルトのスキームが使用されます。TERM が設定されていない場合、色出力は使用されません。
TERM	端末のタイプを決定します。この変数が未設定または NULL の場合は、--color オプションの値に関係なく、色出力が生成されません。

終了ステータス 0 情報はすべて正常に書き込まれました。
 >0 エラーが発生した。

ファイル /etc/group ls -l および ls -g 用のグループ ID
 /etc/passwd ls -l および ls -o 用のユーザー ID
 /usr/share/lib/terminfo/?./* 端末情報データベース

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/ls

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	下記を参照。

-A、-b、-e、-E、-h、-S、U

-v、-V、-@、-/、-%、--all、--almost-all、--block-size、--classify、--color、--colour
 よび--time-styleを除くすべてのオプションについては、[standards\(5\)](#)を参照してください。

/usr/xpg4/bin/ls

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	下記を参照。

-A、-b、-e、-E、-h、-S、U

-v、-V、-@、-/、-%、--all、--almost-all、--block-size、--classify、--color、--colour
 よび--time-styleを除くすべてのオプションについては、[standards\(5\)](#)を参照してください。

/usr/xpg6/bin/ls

属性タイプ	属性値
使用条件	system/xopen/xcu6
CSI	有効
インタフェースの安定性	確実
標準	下記を参照。

-A、-b、-e、-E、-h、-S、U

-v、-V、-@、-/、-%、--all、--almost-all、--block-size、--classify、--color、--colour
 よび--time-styleを除くすべてのオプションについては、[standards\(5\)](#)を参照してください。

関連項目	chmod(1) , cp(1) , setfacl(1) , fgetattr(3C) , strftime(3C) , terminfo(4) , acl(5) , attributes(5) , environ(5) , fsattr(5) , largefile(5) , standards(5)
注意事項	ファイル名に出力できない文字があると、カラムの出力オプションが正しく処理されないことがあります。 ハードリンクされたファイルがある場合、総ブロック数は正しく表示されません。 ls の出力のソート順序はロケールの影響を受け、LC_COLLATE 環境変数で変更できます。たとえば、LC_COLLATE が C の場合、ファイルを表示する順序は、ファイル名がドット (.) で始まるファイル、次にファイル名が大文字で始まるファイル、その後にファイル名が小文字で始まるファイルという順番になります。一方、LC_COLLATE が en_US.ISO8859-1 の場合には、ソート順序を決定する際に、先行するドットや大文字と小文字の区別は無視されます。

名前	ls – list the contents of a directory
形式	<code>/usr/ucb/ls [-aAcCdfFgiLLqRstu1] file...</code>
機能説明	For each <i>filename</i> that is a directory, <code>ls</code> lists the contents of the directory; for each <i>filename</i> that is a file, <code>ls</code> repeats its name and any other information requested. By default, the output is sorted alphabetically. When no argument is given, the current directory is listed. When several arguments are given, the arguments are first sorted appropriately, but file arguments are processed before directories and their contents.
Permissions Field	The mode printed under the <code>-l</code> option contains 10 characters interpreted as follows. If the first character is: - d Entry is a directory. - D Entry is a door. - b Entry is a block-type special file. - c Entry is a character-type special file. - l Entry is a symbolic link. - p Entry is a FIFO (also known as “named pipe”) special file. - s Entry is an AF_UNIX address family socket. – Entry is a plain file. The next 9 characters are interpreted as three sets of three bits each. The first set refers to owner permissions; the next refers to permissions to others in the same user-group; and the last refers to all others. Within each set, the three characters indicate permission respectively to read, to write, or to execute the file as a program. For a directory, “execute” permission is interpreted to mean permission to search the directory. The permissions are indicated as follows: - r The file is readable. - w The file is writable. - x The file is executable. – The indicated permission is not granted. The group-execute permission character is given as <code>s</code> if the file has the set-group-id bit set; likewise the owner-execute permission character is given as <code>s</code> if the file has the set-user-id bit set. The last character of the mode (normally <code>x</code> or <code>–</code>) is <code>t</code> <code>rue</code> if the 1000 bit of the mode is on. See chmod(1) for the meaning of this mode. The indications of set-ID and 1000 bits of the mode are capitalized (<code>S</code> and <code>T</code>, respectively) if the corresponding execute permission is <i>not</i> set.

A plus sign (+) appended to the list of permissions indicates that an ACL is associated with the file.

When the sizes of the files in a directory are listed, a total count of blocks, including indirect blocks, is printed.

オプション

The following options are supported:

- a Lists all entries; in the absence of this option, entries whose names begin with a '.' are *not* listed (except for the privileged user, for whom `ls` normally prints even files that begin with a '.').
- A Same as -a, except that '.' and './' are not listed.
- c Uses time of last edit (or last mode change) for sorting or printing.
- C Forces multi-column output, with entries sorted down the columns; for `ls`, this is the default when output is to a terminal.
- d If argument is a directory, lists only its name (not its contents); often used with -l to get the status of a directory.
- f Forces each argument to be interpreted as a directory and lists the name found in each slot. This option turns off -l, -t, -s, and -r, and turns on -a; the order is the order in which entries appear in the directory.
- F Marks directories with a trailing slash (/), doors with a trailing greater-than sign (>), executable files with a trailing asterisk (*), FIFOs with a trailing vertical bar (|), symbolic links with a trailing at-sign (@), and AF_UNIX address family sockets with a trailing equals sign (=).
- g For `ls`, shows the group ownership of the file in a long output.
- i For each file, prints the i-node number in the first column of the report.
- l Lists in long format, giving mode, ACL indication, number of links, owner, size in bytes, and time of last modification for each file. If the file is a special file the size field will instead contain the major and minor device numbers. If the time of last modification is greater than six months ago, it is shown in the format '*month date year*'; files modified within six months show '*month date time*'. If the file is a symbolic link, the pathname of the linked-to file is printed preceded by '->'.
 - L If argument is a symbolic link, lists the file or directory the link references rather than the link itself.
- q Displays non-graphic characters in filenames as the character ?; for `ls`, this is the default when output is to a terminal.
- r Reverses the order of sort to get reverse alphabetic or oldest first as appropriate.
- R Recursively lists subdirectories encountered.

- s Indicate the total number of file system blocks consumed by each file displayed.
- t Sorts by time modified (latest first) instead of by name.
- u Uses time of last access instead of last modification for sorting (with the -t option) and/or printing (with the -l option).
- 1 Forces one entry per line output format; this is the default when output is not to a terminal.

オペランド The following operand is supported:

file A path name of a file to be listed. If the file specified is not found, a diagnostic message is output on standard error.

使用法 See [largefile\(5\)](#) for the description of the behavior of `ls` when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).

ファイル /etc/group to get group ID for '`ls -g`'
 /etc/passwd to get user IDs for '`ls -l`' and '`ls -o`'

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [ls\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

注意事項 NEWLINE and TAB are considered printing characters in filenames.

The output device is assumed to be 80 columns wide.

The option setting based on whether the output is a teletype is undesirable as '`ls -s`' is much different than '`ls -s | lpr`'. On the other hand, not doing this setting would make old shell scripts which used `ls` almost certain losers.

Unprintable characters in file names can confuse the columnar output options.

名前	m4 – macro processor
形式	<pre>/usr/bin/m4 [-e] [-s] [-B int] [-H int] [-S int] [-T int] [-Dname [=val]] ... [-U name] ... [file]... /usr/xpg4/bin/m4 [-e] [-s] [-B int] [-H int] [-S int] [-T int] [-Dname [...=val]] [-U name] ... [file]...</pre>
機能説明	The m4 utility is a macro processor intended as a front end for C, assembler, and other languages. Each of the argument files is processed in order. If there are no files, or if a file is –, the standard input is read. The processed text is written on the standard output. <i>Note:</i> m4 cannot include more than nine nested files and writes a diagnostic message if that number is exceeded.
Macro Syntax	Macro calls have the form: <pre>name(arg1, arg2, ..., argn)</pre> The open parenthesis character, (, must immediately follow the name of the macro. If the name of a defined macro is not followed by a (, it is deemed to be a call of that macro with no arguments. Potential macro names consist of alphanumeric characters and underscore (_), where the first character is not a digit. Leading unquoted blanks, TABs, and NEWLINES are ignored while collecting arguments. Left and right single quotes are used to quote strings. The value of a quoted string is the string stripped of the quotes.
Macro Processing	When a macro name is recognized, its arguments are collected by searching for a matching right parenthesis. If fewer arguments are supplied than are in the macro definition, the trailing arguments are taken to be NULL. Macro evaluation proceeds normally during the collection of the arguments, and any commas or right parentheses that happen to turn up within the value of a nested call are as effective as those in the original input text. After argument collection, the value of the macro is pushed back onto the input stream and rescanned.
オプション	The options and their effects are as follows: -Bint Changes the size of the push-back and argument collection buffers from the default of 4,096. Values of size less than or equal to zero are ignored and the default value is used. -e Operates interactively. Interrupts are ignored and the output is unbuffered. -Hint Changes the size of the symbol table hash array from the default of 199. For better performance, the size should be prime. Values of size less than or equal to zero are ignored and the default value is used. -s Enables line sync output for the C preprocessor (#line...) -Sint Changes the size of the call stack from the default of 100 slots. Macros take three slots, and non-macro arguments take one. Values of size less than or equal to zero are ignored and the default value is used.

-Tint Changes the size of the token buffer from the default of 512 bytes. Values of size less than or equal to zero are ignored and the default value is used.

To be effective, the above flags must appear before any file names and before any **-D** or **-U** flags:

-D name[=val] Defines *name* to *val* or to NULL in *val*'s absence.

-Uname Undefines *name*.

オペランド

The following operand is supported:

file A path name of a text file to be processed. If no *file* is given, or if it is **-**, the standard input is read.

使用法

The **m4** utility makes available the following built-in macros. These macros can be redefined, but once this is done the original meaning is lost. Their values are NULL unless otherwise stated.

changequote	Change quote symbols to the first and second arguments. The symbols can be up to five characters long. changequote without arguments restores the original values (that is, ' ').
changecom	Change left and right comment markers from the default # and NEWLINE . With no arguments, the comment mechanism is effectively disabled. With one argument, the left marker becomes the argument and the right marker becomes NEWLINE . With two arguments, both markers are affected. Comment markers can be up to five characters long.
decr	Returns the value of its argument decremented by 1.
define	The second argument is installed as the value of the macro whose name is the first argument. Each occurrence of <i>\$n</i> in the replacement text, where <i>n</i> is a digit, is replaced by the <i>n</i> -th argument. Argument 0 is the name of the macro; missing arguments are replaced by the null string; \$# is replaced by the number of arguments; \$* is replaced by a list of all the arguments separated by commas; \$@ is like \$* , but each argument is quoted (with the current quotes).
defn	Returns the quoted definition of its argument(s). It is useful for renaming macros, especially built-ins.
divert	m4 maintains 10 output streams, numbered 0-9. The final output is the concatenation of the streams in numerical order. Initially stream 0 is the current stream. The divert macro changes the current output stream to its (digit-string) argument. Output diverted to a stream other than 0 through 9 is discarded.
divnum	Returns the value of the current output stream.
dnl	Reads and discards characters up to and including the next NEWLINE .

<code>dumpdef</code>	Prints current names and definitions, for the named items, or for all if no arguments are given.
<code>errprint</code>	Prints its argument on the diagnostic output file.
<code>ifdef</code>	If the first argument is defined, the value is the second argument, otherwise the third. If there is no third argument, the value is NULL. The word <code>unix</code> is predefined.
<code>ifelse</code>	This macro has three or more arguments. If the first argument is the same string as the second, then the value is the third argument. If not, and if there are more than four arguments, the process is repeated with arguments 4, 5, 6 and 7. Otherwise, the value is either the fourth string, or, if it is not present, NULL.
<code>include</code>	Returns the contents of the file named in the argument.
<code>incr</code>	Returns the value of its argument incremented by 1. The value of the argument is calculated by interpreting an initial digit-string as a decimal number.
<code>index</code>	Returns the position in its first argument where the second argument begins (zero origin), or <code>-1</code> if the second argument does not occur.
<code>len</code>	Returns the number of characters in its argument.
<code>m4exit</code>	This macro causes immediate exit from m4. Argument 1, if given, is the exit code; the default is 0.
<code>m4wrap</code>	Argument 1 is pushed back at final EOF. Example: <code>m4wrap('cleanup()')</code>
<code>maketemp</code>	Fills in a string of "X" characters in its argument with the current process ID.
<code>popdef</code>	Removes current definition of its argument(s), exposing the previous one, if any.
<code>pushdef</code>	Like <code>define</code> , but saves any previous definition.
<code>shift</code>	Returns all but its first argument. The other arguments are quoted and pushed back with commas in between. The quoting nullifies the effect of the extra scan that is subsequently be performed.
<code>sinclude</code>	This macro is identical to <code>include</code> , except that it says nothing if the file is inaccessible.
<code>substr</code>	Returns a substring of its first argument. The second argument is a zero origin number selecting the first character; the third argument indicates the length of the substring. A missing third argument is taken to be large enough to extend to the end of the first string.

	<code>syscmd</code>	This macro executes the command given in the first argument. No value is returned.
	<code>sysval</code>	This macro is the return code from the last call to <code>syscmd</code> .
	<code>translit</code>	Transliterates the characters in its first argument from the set given by the second argument to the set given by the third. No abbreviations are permitted.
	<code>traceon</code>	This macro with no arguments, turns on tracing for all macros (including built-ins). Otherwise, turns on tracing for named macros.
	<code>traceoff</code>	Turns off trace globally and for any macros specified.
	<code>undefine</code>	Removes the definition of the macro named in its argument.
	<code>undivert</code>	This macro causes immediate output of text from diversions named as arguments, or all diversions if no argument. Text can be undiverted into another diversion. Undiverting discards the diverted text.
<code>/usr/bin/m4</code>	<code>eval</code>	Evaluates its argument as an arithmetic expression, using 32-bit signed-integer arithmetic. The following operators are supported: parentheses, unary <code>-</code> , unary <code>+</code> , <code>!</code> , <code>~</code> , <code>*</code> , <code>/</code> , <code>%</code> , <code>+</code> , <code>-</code> , relationals, bitwise <code>&</code> , <code> </code> , <code>&&</code> , and <code> </code> . Octal and hex numbers can be specified as in C. The second argument specifies the radix for the result; the default is 10. The third argument can be used to specify the minimum number of digits in the result.
<code>/usr/xpg4/bin/m4</code>	<code>eval</code>	Evaluates its argument as an arithmetic expression, using 32-bit signed-integer arithmetic. The following operators are supported: parentheses, unary <code>-</code> , unary <code>+</code> , <code>!</code> , <code>~</code> , <code>*</code> , <code>/</code> , <code>%</code> , <code>+</code> , <code>-</code> , <code><<</code> , <code>>></code> , relationals, bitwise <code>&</code> , <code> </code> , <code>&&</code> , and <code> </code> . Precedence and associativity are as in C. Octal and hex numbers can also be specified as in C. The second argument specifies the radix for the result; the default is 10. The third argument can be used to specify the minimum number of digits in the result.

使用例

例 1 Examples of m4 files

If the file `m4src` contains the lines:

```
The value of 'VER' is "VER".
    ifdef('VER', "VER" is defined to be VER., VER is not defined.)
    ifelse(VER, 1, "VER" is 'VER'.)
    ifelse(VER, 2, "VER" is 'VER'., "VER" is not 2.)
end
```

then the command:

```
m4 m4src
```

or the command:

```
m4 -U VER m4src
```

例1 Examples of m4 files (続き)

produces the output:

```
The value of VER is "VER".
    VER is not defined.

    VER is not 2.
end
```

The command:

```
m4 -D VER m4src
```

produces the output:

```
The value of VER is "".
    VER is defined to be .

    VER is not 2.
end
```

The command:

```
m4 -D VER=1 m4src
```

produces the output:

```
The value of VER is "1".
    VER is defined to be 1.
    VER is 1.
    VER is not 2.
end
```

The command:

```
m4 -D VER=2 m4src
```

produces the output:

```
The value of VER is "2".
    VER is defined to be 2.

    VER is 2.
end
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of m4: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

終了ステータス The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred

If the `m4exit` macro is used, the exit value can be specified by the input file.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/m4	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	system/core-os

/usr/xpg4/bin/m4	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	system/xopen/xcu4
	Interface Stability	Committed
	Standard	See standards(5) .

関連項目 [as\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 mac – 入力メッセージ認証コードの計算

形式 /usr/bin/mac -l

 /usr/bin/mac [-v] -a *algorithm*
 [-k *keyfile* | -K *key_label* [-T *token_spec*]] [*file*]...

機能説明 mac ユーティリティーは、指定のアルゴリズムを使用して、特定の1つ以上のファイルまたはstdinのメッセージ認証コード (MAC) を計算します。

複数のファイルが指定された場合、出力の各行は単一ファイルの MAC となります。

オプション サポートしているオプションは、次のとおりです。

-a *algorithm* 暗号化または復号化プロセスで使用するアルゴリズムの名前を指定します。詳細については、「使用法」の「アルゴリズム」を参照してください。注: 一般的な長さの MAC を生成するアルゴリズムはサポートされていません。

-k *keyfile* 暗号化アルゴリズムのキー値を含むファイルを指定します。PKCS#11 仕様で規定されているように、各アルゴリズムには固有のキー素材要件があります。-k が指定されていない場合、mac は、[getpassphrase\(3C\)](#) を使用してキー素材の入力を要求します。

キーファイルの生成については、[pktool\(1\)](#)、[dd\(1M\)](#)、または『[Oracle Solaris 11.1 の管理: セキュリティーサービス](#)』を参照してください。

-K *key_label* PKCS#11 トークン内で対称トークンキーのラベルを指定します。

-l システムで利用可能なアルゴリズムの一覧を表示します。この一覧は、暗号化フレームワークの構成によって変わります。キーサイズはビット単位で表示されます。

-T *token_spec* -K が指定されている場合に、デフォルトのソフトトークンオブジェクトストア以外の PKCS#11 トークンを指定します。

token_spec は次の形式になります。

token_name [:*manuf_id* [:*serial_no*]]

トークンラベルの末尾にスペースがある場合でも、このオプションではユーザーがスペースを入力する必要はありません。

トークン識別文字列はコロンで区切ります。いずれかの部分に文字としてコロン(:)を含める場合、そのコロンをバックslashで(\)でエスケープする必要があります。コロン(:)がな

い場合、文字列全体が(最大 32 文字)がトークンラベルと見なされます。コロンが(:)1つだけある場合、文字列はトークンラベルと作成者となります。

-v 詳細な情報を表示します。

使用法

アルゴリズム サポートされているアルゴリズムは、**-l** オプションで表示されます。これらのアルゴリズムは、暗号化フレームワークによって提供されます。サポートされている各アルゴリズムは、もっともよく使用され、もっとも制限が少ない特定のアルゴリズムタイプの別名です。たとえば、`md5_hmac` は、`CKM_MD5_HMAC` の別名です。

これらの別名は **-a** オプションとともに使用され、大文字と小文字が区別されま

パスフレーズ **-k** オプションが暗号化および復号化タスクで使用されない場合、ユーザーはパスフレーズの入力を求められます。パスフレーズは、PKCS #5 で指定されている PBKDF2 アルゴリズムを使用して処理され、より安全なキーに変換されます。

使用例

例1 使用できるアルゴリズムの一覧表示

次の例では、使用できるアルゴリズムが一覧表示されます。

```
example$ mac -l
Algorithm      Keysize:  Min   Max
-----
des_mac                64    64
sha1_hmac              8   512
md5_hmac               8   512
sha224_hmac            8   512
sha256_hmac            8   512
sha384_hmac            8  1024
sha512_hmac            8  1024
```

例2 メッセージ認証コードの取得

次の例では、ファイル用のメッセージ認証コードを取得します。

```
example$ mac -v -k mykey -a sha1_hmac /export/foo
sha1_hmac (/export/foo) = 913ced311df10f1708d9848641ca8992f4718057
```

例3 トークンキー付きのメッセージ認証コードの取得

次の例では、ソフトトークンキーストア内で汎用トークンキー付きのメッセージ認証コードを取得します。汎用トークンキーは `pktool(1)` で生成できます。

```
encrypt -v -a sha1_hmac -K my_generic_key \
-T "Sun Software PKCS#11 softtoken" /export/foo
Enter pin for Sun Software PKCS#11 softtoken:
```

例3 トークンキー付きのメッセージ認証コードの取得 (続き)

```
sha1_hmac (/etc/foo) = c2ba5c38458c092a68940081240d22b670182968
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [digest\(1\)](#), [pktool\(1\)](#), [dd\(1M\)](#), [getpassphrase\(3C\)](#), [libpkcs11\(3LIB\)](#), [attributes\(5\)](#), [pkcs11_softtoken\(5\)](#)

『Oracle Solaris 11.1 の管理: セキュリティーサービス』

RSA PKCS#11 v2.20 and RSA PKCS#5 v2.0, <http://www.rsasecurity.com>

名前	mach – 現在のホストのプロセッサタイプの表示
形式	mach
機能説明	mach コマンドは、現在のホストのプロセッサタイプを表示します。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [arch\(1\)](#), [uname\(1\)](#), [attributes\(5\)](#)

注意事項 mach および `uname -p` は同等の値を返します。そのため独立ソフトウェアベンダー (ISV) やその他のベンダーでプロセッサタイプを確認する必要がある場合、mach コマンドではなく `uname` を `-p` オプションとともに使用することをお勧めします。mach コマンドは以前のリリースとの互換性確保のために提供されており、通常は使用しないようにしてください。

名前	machid, sun, i386, i486, sparc – プロセッサタイプの真理値の取得				
形式	sun i386 sparc				
機能説明	次のコマンドは、コマンド名が示す命令セットを使用する場合、真の値 (終了コードは 0) を返します。 sun Sun システムであれば真。 i386 iAPX386 プロセッサを使用するコンピュータであれば真。 sparc SPARC ファミリのプロセッサを使用するコンピュータであれば真。 該当しないコマンドは偽 (0 以外) の値を返します。これらのコマンドは、可搬性を高めるために、メイクファイル (make(1S) を参照) およびシェルスクリプト (sh(1) を参照) 内でよく使用されます。				
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。				
	<table><tr><th>属性タイプ</th><th>属性値</th></tr><tr><td>使用条件</td><td>system/core-os</td></tr></table>	属性タイプ	属性値	使用条件	system/core-os
属性タイプ	属性値				
使用条件	system/core-os				
関連項目	make(1S) , sh(1) , test(1) , true(1) , uname(1) , attributes(5)				
注意事項	machid ファミリのコマンドは廃止されました。代わりに <code>uname -p</code> および <code>uname -m</code> を使用してください。				

名前	madv.so.1 – madv ライブラリ
形式	/usr/lib/madv.so.1
機能説明	madv.so.1 共有オブジェクトは、起動されたプロセス (1 つまたは複数) とその子孫の VM アドバイスを選択的に構成する手段を提供します。madv.so.1 を有効にするには、次の文字列が 1 つ以上の MADV 環境変数とともに環境内に存在する必要があります (ld.so.1(1) を参照)。 LD_PRELOAD=\$LD_PRELOAD:madv.so.1
環境変数	LD_PRELOAD リストに madv.so.1 共有オブジェクトを指定すると、次の環境変数が madv 共有オブジェクトに読み込まれ、指定されたアドバイスをどの生成済みプロセスに適用するのかが決定されます。 MADV=advice MADV には、プロセスのアドレス空間内のすべてのヒープ、共有メモリー、および mmap 領域に使用される VM アドバイスを指定します。この情報は生成されたすべてのプロセスに適用されます。 <i>advice</i> の値は、madvise(3C) でメモリアクセスパターンを指定するために使用する <sys/mman.h> の値と対応しています。 normal random sequential access_lwp access_many access_many_pset access_default MADVCFGFILE=config-file <i>config-file</i> は、次の形式の madv 構成エントリを 1 つ以上含むテキストファイルです。 <i>exec-name exec-args:advice-opts</i> <i>config-file</i> で指定されたアドバイスは、MADV 環境変数によって指定された情報よりも優先されます。MADVCFGFILE が設定されていない場合は、/etc/madv.conf ファイルからアドバイスが取得されます (存在している場合)。 <i>exec-name</i> には、アプリケーションまたは実行可能ファイルの名前を指定します。対応するアドバイスは、ファイル内で最初に見つかった <i>exec-name</i> と一致する新規作成プロセス (getexecname(3C) を参照) 用に設定されます。

exec-name には、フルパス名、基本名、またはパターン文字列を指定できます。パターンマッチングについては、[sh\(1\)](#) の「ファイル名の生成」を参照してください。

exec-args は、引数と照合する任意指定のパターン文字列です。*exec-args* が指定されていないか、または *exec-name* の引数内に存在する場合にのみ、アドバイスは設定されます。

advice-opts は、次のようなさまざまなメモリー領域に対するアドバイスを指定したコンマ区切りのリストです。

madv=advice プロセスのアドレス空間のすべてのヒープ、共有メモリー、および *mmap* 領域に適用されます。

heap=advice ヒープは、*brk* 領域 ([brk\(2\)](#)) を参照) として定義されます。既存のヒープにも、将来割り当てられる追加ヒープメモリーにも適用されます。

shm=advice

ism=advice

dism=advice

任意のフラグ、*SHM_SHARE_MMU* フラグ、または *SHM_PAGEABLE* フラグを使用して接続された共有メモリーセグメント ([shmat\(2\)](#)) を参照)。*ism* および *dism* オプションは、*shm* オプションよりも優先されます。

map=advice

mapshared=advice

mapprivate=advice

mapanon=advice

[mmap\(2\)](#) の任意のフラグ、*MAP_SHARED* フラグ、*MAP_PRIVATE* フラグ、または *MAP_ANON* フラグを使用して確立されたマッピング。 *mapshared*、*mapprivate*、および *mapanon* オプションは、*map* オプションよりも優先されます。 *mapanon* オプションは、*mapshared* および *mapprivate* よりも優先されます。

MADVERRFILE=*pathname* デフォルトでは、エラーメッセージは **syslog(3C)** の **LOG_ERR** レベルと **LOG_USER** 機能を使用してログに記録されます。MADVRRFILE に有効な *pathname* (**/dev/stderr** など) が含まれている場合は、エラーメッセージがその場所に記録されます。

使用例

例1 アドバイスをすべての ISM セグメントに適用する

次の構成では、**/usr/bin/foo** アプリケーションのすべての ISM セグメントにアドバイスを適用します。

```
example$ LD_PRELOAD=$LD_PRELOAD:madv.so.1
example$ MADVCFGFILE=madvcfg
example$ export LD_PRELOAD MADVCFGFILE
example$ cat $MADVCFGFILE
/usr/bin/foo:ism=access_lwp
```

例2 例外付きですべてのアプリケーションにアドバイスを設定する

次の構成では、**ls** を除くすべてのアプリケーションにアドバイスを設定します。

```
example$ LD_PRELOAD=$LD_PRELOAD:madv.so.1
example$ MADV=access_many
example$ MADVCFGFILE=madvcfg
example$ export LD_PRELOAD MADV MADVCFGFILE
example$ cat $MADVCFGFILE
ls:
```

例3 優先順位規則 (例2 の続き)

MADVCFGFILE は MADV よりも優先されるため、最後の **madv** 構成エントリの *exec-name* に「*」(すべてに一致するパターン)を指定することは、MADV を設定することと同等です。次の例は、例2 と同等です。

```
example$ LD_PRELOAD=$LD_PRELOAD:madv.so.1
example$ MADVCFGFILE=madvcfg
example$ export LD_PRELOAD MADVCFGFILE
example$ cat $MADVCFGFILE
ls:
*:madv=access_many
```

例4 領域ごとに異なるアドバイスを適用する

次の構成では、**foo** で始まる実行可能ファイル名を持つ一連のアプリケーションについて、あるタイプのアドバイスを **mmap** 領域に、別のタイプのアドバイスをヒープ領域と共有メモリ領域に適用します。

```
example$ LD_PRELOAD=$LD_PRELOAD:madv.so.1
example$ MADVCFGFILE=madvcfg
example$ export LD_PRELOAD MADVCFGFILE
```

例4 領域ごとに異なるアドバイスを適用する (続き)

```
example$ cat $MADVCFGFILE
foo*:madv=access_many,heap=sequential,shm=access_lwp
```

例5 アドバイスの選択的な適用

次の構成では、名前がoraで始まり、引数としてora1が指定されたアプリケーションのヒープにアドバイスを適用します。

```
example$ LD_PRELOAD=$LD_PRELOAD:madv.so.1
example$ MADVCFGFILE=madvcfg
example$ export LD_PRELOAD MADVCFGFILE
example$ cat $MADVCFGFILE
ora* ora1:heap=access_many
```

ファイル /etc/madv.conf 構成ファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/extended-system-utilities)
インタフェースの安定性	不確実

関連項目 [cat\(1\)](#), [ld.so.1\(1\)](#), [proc\(1\)](#), [sh\(1\)](#), [brk\(2\)](#), [exec\(2\)](#), [fork\(2\)](#), [mmap\(2\)](#), [mencntl\(2\)](#), [shmat\(2\)](#), [getexecname\(3C\)](#), [madvise\(3C\)](#), [syslog\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)

注意事項 アドバイスは継承されます。子プロセスには親と同じアドバイスが適用されます。exec() ([exec\(2\)](#) を参照) では、madv 共有オブジェクトで別のアドバイスが構成されていないかぎり、アドバイスはデフォルトのシステムアドバイスに設定されます。

アドバイスは、ユーザープログラムによって作成された mmap 領域にのみ適用されます。実行時リンカーまたは直接システムコールを行うシステムライブラリ (libthread によるスレッドスタックの割り当てなど) によって確立された領域は影響を受けません。

名前 mail, rmail – メールの読み取りまたはユーザーへのメールの送信

形式

メールの送信 mail [-tw] [-m *message_type*] *recipient*...

rmail [-tw] [-m *message_type*] *recipient*...

メールの読み取り mail [-ehpPqr] [-f *file*]

デバッグ mail [-x *debug_level*] [*other_mail_options*] *recipient*...

機能説明

recipient は通常、ドメイン形式 ("*user@machine*") のアドレス、または [login\(1\)](#) が認識するユーザー名です。*recipient* を指定すると、mail はメッセージを送信中であるとみなします。このコマンドは、標準入力からファイルの終端 (Control-d) まで、あるいは端末デバイスから読み取っている場合は、ピリオドだけの行に至るまで読み取りを続けます。ファイルの終端またはピリオドだけの行のいずれかを受信すると、mail は各 *recipient* に対してレターをメールファイルに追加します。

レターは、最初にヘッダー行が数行、次に空行、さらに本文が続きます。レターのヘッダー行セクションは、1 つまたは複数の UNIX ポストマークから構成されます。

From *sender* *date_and_time* [*remote from remote_system_name*]

この後に、次の形式の 1 つまたは複数の標準メッセージヘッダー行が続きます。

keyword-name: [*printable text*]

ここで、*keyword-name* は、コロン (':') 以外の表示可能なスペース以外の文字から構成されます。MIME-version: ヘッダー行は、メッセージが RFC 2045 の記述に従ってフォーマットされていることを示します。本文のバイト数を示す Content-Length: ヘッダー行は、レターがメッセージ内容のないヘッダー行だけから構成されていないかぎり、常に存在します。Content-Type: ヘッダー行は本文 (テキスト、アプリケーション/オクテットストリームなど) のタイプを示し、常に存在します。ただし、レターがメッセージ内容のないヘッダー行だけから構成されている場合は例外です。ヘッダー行は、その次の行がスペースで始まっている場合、次の行に継続できます。

オプション

メールの送信 次のコマンド行引数は、メール送信に影響を与えます。

-m *message_type* Message-Type: 行を、*message_type* の値とともにメッセージヘッダーに追加します。

-t 対象となる *recipient* のそれぞれについて、To: 行を、メッセージヘッダーに追加します。

-w リモート送信プログラムの完了を待たずに、レターをリモート受信者に送信します。

レターが配送できないことがわかった場合、そのレターは障害の原因と性質を示す診断とともに送信者に返されます。入力中に `mail` が中断された場合、メッセージを `dead.letter` というファイルにセーブするので、後から編集と再送信を実行することもできます。`dead.letter` は常に追加されるので、以前の内容はすべて保存されます。`dead.letter` の追加(または作成)は、まず、現在のディレクトリで行われます。これが失敗すると、ユーザーのログインディレクトリに `dead.letter` を追加(または作成)しようとします。2度目の試行も失敗した場合は、`dead.letter` の処理は一切行われません。

`rmail` は、メールの送信だけ可能です。`uucp(1C)` は `rmail` をセキュリティー通知として使用します。メールメッセージを生成するアプリケーションプログラムは、メッセージの送信や配送、またはその両方において、必ず、`mail` ではなく `rmail` を起動する必要があります。

ローカルシステムに `Basic Networking Utilities` をインストールしてある場合、ローカルシステムで利用可能な送信メカニズムに応じて、さまざまな方法でリモートシステム上の受信者にメールを送信できます。もっとも広く利用されているアドレス指定機構は、ドメイン形式と UUCP 形式の2つです。

ドメイン形式のアドレス指定 受信者名に '@' とドメイン (およびサブドメインもあり得る) 情報を追加してリモート受信者を指定します (たとえば、`user@sf.att.com`)。 (ローカルシステム上でどの規則が利用可能かについての詳細は、ローカルシステム管理者に相談する必要があります)。

UUCP 形式のアドレス指定 リモート受信者は、`sysa!user` のように、受信者名にリモートシステム名と感嘆符を接頭辞として付けることによって指定します。`cs(1)` がデフォルトシェルの場合、`sysa\!user` を使用するようになしてください。一連のシステム名を感嘆符で区切れば、拡張したネットワークを介してレターを出力できます (たとえば、`sysa!sysb!sysc!user` または `sysa\!sysb\!sysc\!user`)。

メールの読み取り 次のコマンド行引数は、メールの読み取りに影響を与えます。

-e メールがあるかどうかをテストします。`mail` は何も表示しません。

ユーザーにメールがある場合、0 の終了コードを返します。メールがない場合、1 の終了コードを返します。

-E **-e** と似ていますが、新しいメールがあるかどうかをテストします。

ユーザーに新しいメールがある場合、0 の終了コードを返します。新しいメールがない場合、1 の終了コードを返します。既読のメールがある場合、2 の終了コードを返します。

- h 最初に、最新のメッセージではなく、ヘッダーのウィンドウを表示します。ディスプレイの後に ? プロンプトを表示します。
- p 配置を示すプロンプトなしに、すべてのメッセージを表示します。
- P デフォルトの選択したヘッダー行を表示する代わりに、ヘッダー行すべてを表示してから、すべてのメッセージを表示します。
- q 割り込みを受けたあと、mail は終了します。通常、割り込みによって発生することは、表示中のメッセージの終了だけです。
- r 先入れ先出し方式でメッセージを表示します。
- f *file* デフォルトのメールファイルの代わりに、mail は、*file* (たとえば、mbox) を使用します。

mail は、コマンド行引数の影響を特に受けないかぎり、後入れ先出し方式でユーザーのメールメッセージを表示します。メッセージ表示のデフォルトのモードは、至急必要なヘッダー行だけを表示することです。これらには、UNIX の From ポストマークおよび >From ポストマーク、From: ヘッダー行、Date: ヘッダー行、Subject: ヘッダー行、および Content-Length: ヘッダー行、ならびに To:、Cc:、Bcc: などの受信者ヘッダー行が含まれますが、これに限定されるわけではありません。ヘッダー行を表示したあと、mail は、メッセージに表示できない文字がないかぎり、その内容 (本体) を表示します。表示できない文字がある場合、mail は、メッセージの内容がバイナリである旨の警告文を発行し、その内容を表示しません (これは、p コマンドで無効にできます)。

各メッセージにおいて、ユーザーは ? というプロンプトを受け、標準入力から行が読み取られます。メッセージの配置を判別するときは、次のコマンドを利用できます。

- # 現在のメッセージの数を表示します。
- 直前のメッセージを表示します。
- <復帰改行>、+、または n 次のメッセージを表示します。
- !*command* シェルにエスケープして、*command* を実行します。
- a mail セッション中に到着したメッセージを表示します。
- d または dp 現在のメッセージを削除し、次のメッセージを表示します。

<code>d n</code>	<code>n</code> 番のメッセージを削除します。次のメッセージを表示しません。
<code>dq</code>	メッセージを削除し、 <code>mail</code> を終了します。
<code>h</code>	現在のメッセージを中心としてヘッダーのウィンドウを表示します。
<code>hn</code>	<code>n</code> 番のメッセージを中心としてヘッダーのウィンドウを表示します。
<code>h a</code>	ユーザーのメールファイルにすべてのメッセージのヘッダーを表示します。
<code>h d</code>	削除予定のメッセージのヘッダーを表示します。
<code>m [persons]</code>	指定された <i>persons</i> に現在のメッセージを送信 (および削除) します。
<code>n</code>	<code>n</code> 番のメッセージを表示します。
<code>p</code>	現在のメッセージを再度表示し、バイナリ (つまり、表示不能) の内容の表示を無効にします。
<code>P</code>	デフォルトの省略モードを無効にし、現在のメッセージを再表示して、ヘッダー行をすべて表示します。
<code>q</code> または <code>CTRL-D</code>	削除を解除したメールをメールファイルに戻し、 <code>mail</code> を終了します。
<code>r [users]</code>	送信者とほかの <i>users</i> に応答してから、メッセージを削除します。
<code>s [files]</code>	メッセージを、指定された <i>files</i> (デフォルトは <code>mbox</code>) にメッセージの内容をヘッダー行なしでセーブし、メッセージを削除します。
<code>u [n]</code>	<code>n</code> 番のメッセージの削除を解除します (デフォルトは直前に読み取ったメッセージ)。
<code>w [files]</code>	指定された <i>files</i> (デフォルトは <code>mbox</code>) にメッセージの内容をヘッダー行なしでセーブし、メッセージを削除します。
<code>x</code>	メールすべてを変更を加えずにメールファイルに戻し、 <code>mail</code> を終了します。
<code>y [files]</code>	<code>-w</code> オプションと同じ。
<code>?</code>	コマンドサマリーを表示します。

通常、ユーザーがログインする際にメールが存在すれば、それが通知されます。また、mail を使用中に新しいメールが到着した場合でも通知されます。

`chmod(1)` を使用し、2 種類の方法でメールファイルのアクセス権を処理すれば、mail の機能を変更できます。ファイルのその他のアクセス権を、読み取り書き込みとも可能 (0666)、読み取り専用 (0664)、または読み取り書き込みとも不可 (0660) とすることによって、プライバシをさまざまに調整できます。デフォルト (0660 モード) 以外に変更すると、空の場合でもファイルは保存され、希望のアクセス権が永久的になります (管理者は mailcnfg の DEL_EMPTY_MAILFILE オプションを使用すれば、このファイル保存を無効にできます)。

メールファイルのグループ ID を mail として新しいメッセージを配送可能にし、メールファイルをグループ mail で書き込み可能にする必要があります。

デバッグ

次のコマンド行引数によって、mail は、デバッグ情報を提供します。

`-x debug_level` mail は、デバッグ情報の入ったトレースファイルを作成します。

`-x` オプションを指定すると、mail は /tmp/MLDBGprocess_id という名前のファイルを作成します。このファイルには、mail が現在のメッセージを処理した手順に関するデバッグ情報が入っています。`debug_level` の絶対値はデバッグ情報の長さを制御します。0 はデバッグなしを意味します。`debug_level` が 0 より大きい場合、デバッグファイルが保持されるのは、mail のメッセージ処理中に問題が発生した場合だけです。`debug_level` が 0 未満の場合、常にデバッグファイルを保持します。`-x` で `debug_level` を指定すると、/etc/mail/mailcnfg 内の DEBUG の指定すべてが無効になります。`-x` オプションの提供する情報は難解であり、役立つのはシステム管理者だけだと思います。

配送通知

メールの通知には、いくつかの形式があります。次の行をメッセージヘッダーに挿入することによって実現します。

Transport-Options: [/options]

Default-Options: [/options]

>To: recipient [/options]

ここで、"/options" には、次のうちの 1 つまたは複数が使用できます。

/delivery	メッセージが recipient のメールボックスに正常に配送されたことを送信者に通知します。
/nodelivery	配送が成功したことを送信者に通知しません。
/ignore	配送が失敗したことを送信者に通知しません。
/return	メール配送が失敗したかどうかを送信者に通知します。送信者に失敗メッセージを返します。

	/report /returnと同じ。ただし、元のメッセージは返しません。
	デフォルトは /nodelivery/return です。矛盾のあるオプションを使用すると、最初のオプションを認識し、その後の矛盾する条件を無視します。
オペランド	メール送信には次のオペランドがあります。 recipient ドメイン形式 ("user@machine") のアドレス、または login(1) が認識するユーザーログイン名
使用法	ファイルが 2G バイト (2³¹ バイト) 以上ある場合の mail と rmail の動作については、 largefile(5) を参照してください。
環境変数	mail の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。 LC_CTYPE、 LC_MESSAGES、 および NLSPATH。 TZ 日付と時間の文字列とともに使用するタイムゾーンを指定する
終了ステータス	次の終了ステータスが返されます。 0 そのユーザー宛のメールがあり、処理は正常に終了しました。 1 ユーザー宛のメールがなかった、または初期化時にエラーが発生しました。 >1 初期化のあとでエラーが発生しました。
ファイル	dead.letter メールできなかったテキスト /etc/passwd 送信者の識別および recipient の発見用 \$HOME/mbox セーブされたメール \$MAIL メールファイルのパス名を含む変数 /tmp/MLDBG* デバッグ・トレースファイル /var/mail/*.lock メールディレクトリのロック /var/mail/:saved 一時ファイルを保持し、システムクラッシュ時のデータ損失を防ぐためのディレクトリ /var/mail/user ユーザーに送られたメール (ユーザーに届いたメール)。すなわち、デフォルトのメールファイル var/tmp/ma* 一時ファイル
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目	<code>chmod(1)</code> , <code>csh(1)</code> , <code>login(1)</code> , <code>mailx(1)</code> , <code>uucp(1C)</code> , <code>uuencode(1C)</code> , <code>vacation(1)</code> , <code>write(1)</code> , <code>attributes(5)</code> , <code>environ(5)</code> , <code>largefile(5)</code> 『Solaris Advanced User's Guide』
注意事項	前述の「配送通知」のセクションで説明したように、ヘッダ行による解釈と結果的な処置が発生するのは、配送 (または失敗) が行われたシステム上に、このバージョンの <code>mail</code> がインストールされている場合だけです。 <code>mail</code> の初期のバージョンはどのタイプの配送通知もサポートしません。 条件によっては、ロックファイルの削除が失敗することがあります。 割り込み後、次のメッセージが表示されないことがあります。表示を強制するときは、<code>p</code> を入力します。

名前	mail, Mail – interactive message processing system	
形式	/usr/ucb/mail ...	
	/usr/ucb/Mail ...	
機能説明	/usr/ucb/mail and /usr/ucb/Mail are provided as links to /usr/bin/mailx. See mailx(1) for more information on the usage of these commands.	
属性	See attributes(5) for descriptions of the following attributes:	
/usr/ucb/mail	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	compatibility/ucb
/usr/ucb/Mail	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	compatibility/ucb
関連項目	mailx(1) , attributes(5)	

名前	mailcompat – provide SunOS compatibility for Solaris mailbox format
機能説明	mailcompat is a program to provide SunOS 4.x compatability for the Solaris mailbox format. You would typically run mailcompat to be able to read mail on a workstation running SunOS 4.x when your mail server is running Solaris. Enabling mailcompat creates an entry in your .forward file, if it exists. If this file does not exist, mailcompat will create it. Disabling mailcompat will remove the entry from the .forward file, and if this was the only entry, will remove the entire file. To execute mailcompat, log onto the Solaris mail server and enter mailcompat on the command line. Answer the queries provided by the program.
使用法	See largefile(5) for the description of the behavior of mailcompat when encountering files greater than or equal to 2 Gbyte (2^{31} bytes). Depending on the security settings in your system, mailcompat can fail to create a .forward file. If you encounter such a failure, take the steps outlined below. As root user or user with comparable privileges, enter: <pre>% sendmail -bv \$USER</pre> You receive one of the following two messages. The string fwtest is for example purposes only. <pre>fwtest... deliverable: mailer local, user fwtest</pre> <pre>" /usr/bin/mailcompat fwtest"... deliverable: mailer prog,\ user " /usr/bin/mailcompat fwtest"</pre> The first message, with the local designation, indicates that mailcompat was not able to create the .forward file. The second message, with the prog designation, means that mailcompat succeeded in creating .forward. As a further check, invoke: <pre>% check-permissions \$USER</pre> You should get the message: <pre>No unsafe directories found.</pre> If you get a message different from this, it might indicate a reason for mailcompat failure. You should correct any conditions pointed out by the message before retrying mailcompat.
使用例	例1 Examples of the mailcompat feature. The following example enables the mailcompat feature for the user john. <pre>example% mailcompat</pre> This program can be used to store your mail in a format that you can read with SunOS 4.X based mail readers

例1 Examples of the mailcompat feature. (続き)

To enable the mailcompat feature a ".forward" file is created.
Would you like to enable the mailcompat feature? Y
Mailcompat feature ENABLED.Run mailcompat with no arguments to remove it
example%

The following example disables the mailcompat feature for the user john.

example% **mailcompat**
This program can be used to store your mail in a format
that you can read with SunOS 4.X based mail readers
You have a .forward file in your home directory containing:
 "/usr/bin/mailcompat johns"
Would you like to remove it and disable the mailcompat feature? y
Back to normal reception of mail.
example%

ファイル ~/.forward list of recipients for forwarding messages

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [mailx\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

名前	mailp, digestp, filep, newsp, filofaxp, franklinp, timemanp, timesysp – プリンタ記述言語 (PDL) プリティープリントフィルタへの mp テキストへのフロントエンド
形式	<pre>mailp [options] filename... newsp [options] filename... digestp [options] filename... filep [options] filename... filofaxp [options] filename... franklinp [options] filename... timemanp [options] filename... timesysp [options] filename...</pre>
機能説明	mailp ユーティリティーは、mp(1) プログラムへのフロントエンドです。異なる名前を使用して、さまざまな mp オプションを提供します。 mailp メールメッセージを印刷します。 newsp USENET のニュース記事を印刷します。 digestp USENET ダイジェストファイルを印刷します。 filep 通常の ASCII ファイルを印刷します。 filofaxp Filofax のパーソナル・オーガナイザ・フォーマットで印刷します。 franklinp Franklin のパーソナル・オーガナイザ・フォーマットで印刷します。 timemanp Time Manager のパーソナル・オーガナイザ・フォーマットで印刷します。 timesysp Time/System International のパーソナル・オーガナイザ・フォーマットで印刷します。 mailp(およびそれに関連するプログラム) は、各 <i>filename</i> を順に読み取り、内容の清書バージョンを生成します。ファイル名引数が指定されていない場合、mailp は標準入力を読み取ります。 mailp は、次の 2 つの方法で動作します。-d オプションを指定した場合には、X 印刷サーバーのクライアントとして動作し、出力先プリンタの PDL を生成してスプールします。-d または -p オプションを指定した場合には、PostScript(tm) の出力を生成してスプールします。 オプション サポートしているオプションは、次のとおりです。 -d <i>printer</i> 指定したプリンタへ出力を送信します。それ以外の場合は、PRINTER 環境変数で指定されたプリンタへ出力を送信します。

- D 出力先プリンタ用に PDL を生成して、そのプリンタ用にスプールします。
- F メールの上部のヘッダーに、メールの送信先ではなく、送信元が印刷されます。自分専用のプリンタを持っているユーザー向けのオプションです。
- h バナーは印刷されません。通常バナーシートに表示される情報が、mp バナーでの出力です。
- l 横モードで出力します。紙 1 枚にテキストの 2 ページ分が印刷されます。
- P printer -d オプションを指定した場合と同じです。
- s subject subject を、印刷する新しいサブジェクトとして使用します。コマンド行で指定した通常の ASCII ファイルを印刷する場合は、サブジェクトはデフォルトでその ASCII ファイル名となります。

オペランド 次のオペランドを指定できます。

filename 読み取るファイルの名前。

環境変数 -d、-D、-P オプションのいずれも指定されていない場合は、mailp は、PRINTER 環境変数を使用して mp(1) プログラムからの出力の送り先プリンタを決定します。この PRINTER 環境変数が見つからない場合、デフォルトで PostScript(tm) プリンタに送られます。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	print/mp

関連項目 mp(1), attributes(5)

注意事項 -P オプションは、mp(1) で使用された場合は出力先プリンタに直接 PDL をスプールしますが、mailp で使用される場合には、下位互換性を保つために PostScript(tm) を生成します。

名前	mailq – print the mail queue
形式	<code>/usr/bin/mailq [-Ac] [-q <i>subarg</i>] [-v]</code>
機能説明	The mailq utility displays a summary of the mail messages queued for future delivery. The first line displayed for each mail message shows the internal identifier used on this host for the message, the size of the message in bytes, the date and time the message was accepted into the queue, and the envelope sender of the message. The second line of the display shows the error message that caused this message to be retained in the queue. This line will not be displayed if the message is being processed for the first time. The mailq utility used to be identical to sendmail -bp. Now it checks for the authorization attribute, solaris.mail.mailq. If the check for the invoking user succeeds, sendmail -bp is executed with the remaining argument vector. Otherwise, an error message is printed. This authorization attribute is by default enabled for all users. It can be disabled by modifying the Basic Solaris User entry in prof_attr(4).
オプション	The following options are supported: -Ac Like sendmail(1M), this flag tells mailq to use submit.cf rather than sendmail.cf even if the operation mode does not indicate an initial mail submission. This will result in the client queue <code>/var/spool/clientmqueue</code> being displayed rather than the default server queue <code>/var/spool/mqueue</code>. -qp[<i>time</i>] Similar to -qtime, except that instead of periodically forking a child to process the queue, sendmail forks a single persistent child for each queue that alternates between processing the queue and sleeping. The sleep time is given as the argument. The sleep time default is 1 second. The process will always sleep at least 5 seconds if the queue was empty in the previous queue run. -qf Processes saved messages in the queue once and does not fork(), but runs in the foreground. -qG <i>name</i> Processes jobs in the queue group called <i>name</i> only. -q[!]I <i>substr</i> Limits processed jobs to those containing <i>substr</i> as a substring of the queue id, or not when ! is specified. -q[!]R <i>substr</i> Limits processed jobs to those containing <i>substr</i> as a substring of one of the recipients, or not when ! is specified. -q[!]S <i>substr</i> Limits processed jobs to those containing <i>substr</i> as a substring of the sender, or not when ! is specified. -v Prints verbose information. This adds the priority of the message and a single character indicator (+ or blank) indicating whether a warning message has been sent on the first line of the message. Additionally, extra lines may be intermixed with the recipients that indicate the "controlling user"

information. This shows who will own any programs that are executed on behalf of this message and the name of the alias this command is expanded from, if any.

終了ステータス **0** Successful completion.
 >0 An error occurred.

ファイル /etc/security/prof_attr local source for execution profile attributes
 /var/spool/mqueue default server queue
 /var/spool/clientmqueue client queue

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/smtp/sendmail

関連項目 [sendmail\(1M\)](#), [prof_attr\(4\)](#), [attributes\(5\)](#)

名前	mailstats – print statistics collected by sendmail																		
形式	mailstats [-o] [-c] [-C <i>configfile</i>] [-f <i>statisticsfile</i>] [-p] [-P]																		
機能説明	The mailstats utility prints out the statistics collected by the sendmail(1M) program on mailer usage. These statistics are collected if the file indicated by the StatusFile configuration option of sendmail (defined in /etc/mail/sendmail.cf) exists. The default statistics file is /etc/mail/statistics. To enable mailstats, you must, as root, touch /etc/mail/statistics. See the StatusFile processing option in sendmail(1M). mailstats first prints the time that the statistics file was created and the last time it was modified. Then, the statistics for each mailer are displayed on a single line, each with the following whitespace-separated fields: <table> <tr> <td>M</td><td>The mailer number.</td></tr> <tr> <td>msgsfr</td><td>Number of messages from the mailer.</td></tr> <tr> <td>bytes_from</td><td>Kbytes from the mailer.</td></tr> <tr> <td>msgsto</td><td>Number of messages to the mailer.</td></tr> <tr> <td>bytes_to</td><td>Kbytes to the mailer.</td></tr> <tr> <td>msgsrej</td><td>Number of messages rejected by the mailer.</td></tr> <tr> <td>msgsdisc</td><td>Number of messages discarded by the mailer.</td></tr> <tr> <td>msgsqur</td><td>Number of messages quarantined by the mailer.</td></tr> <tr> <td>Mailer</td><td>The name of the mailer.</td></tr> </table> The display of statistics described above is followed by a separation line containing only equal sign (=) characters. After the separation line, a line preceded with a “T” and totaling the values for all of the mailers is displayed. This is followed by another line preceded with a “C” that lists the number of TCP connections. To reinitialize the statistics file once a night, add an entry to root's crontab(1): <pre>mailstats -p > /dev/null</pre>	M	The mailer number.	msgsfr	Number of messages from the mailer.	bytes_from	Kbytes from the mailer.	msgsto	Number of messages to the mailer.	bytes_to	Kbytes to the mailer.	msgsrej	Number of messages rejected by the mailer.	msgsdisc	Number of messages discarded by the mailer.	msgsqur	Number of messages quarantined by the mailer.	Mailer	The name of the mailer.
M	The mailer number.																		
msgsfr	Number of messages from the mailer.																		
bytes_from	Kbytes from the mailer.																		
msgsto	Number of messages to the mailer.																		
bytes_to	Kbytes to the mailer.																		
msgsrej	Number of messages rejected by the mailer.																		
msgsdisc	Number of messages discarded by the mailer.																		
msgsqur	Number of messages quarantined by the mailer.																		
Mailer	The name of the mailer.																		
オプション	The following options are supported: <table> <tr> <td>-c</td><td>Try to use submit.cf instead of the default sendmail configuration file.</td></tr> <tr> <td>-C <i>configfile</i></td><td>Specify a sendmail configuration file.</td></tr> <tr> <td>-f <i>statisticsfile</i></td><td>Specify a sendmail statistics file.</td></tr> <tr> <td>-o</td><td>Do not display the name of the mailer in the output.</td></tr> </table>	-c	Try to use submit.cf instead of the default sendmail configuration file.	-C <i>configfile</i>	Specify a sendmail configuration file.	-f <i>statisticsfile</i>	Specify a sendmail statistics file.	-o	Do not display the name of the mailer in the output.										
-c	Try to use submit.cf instead of the default sendmail configuration file.																		
-C <i>configfile</i>	Specify a sendmail configuration file.																		
-f <i>statisticsfile</i>	Specify a sendmail statistics file.																		
-o	Do not display the name of the mailer in the output.																		

-P Output information in program-readable mode without clearing statistics.

ファイル	/dev/null	Zero-lined file
	/etc/mail/statistics	Default sendmail statistics file
	/etc/mail/sendmail.cf	Default sendmail configuration file

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/smtp/sendmail
Interface Stability	The output is uncommitted.

名前	mailx – interactive message processing system
形式	<pre>mailx [-BdeHiInNURvV~] [-f [file +folder]] [-T file] [-u user] mailx [-BdFintUv~] [-b bcc] [-c cc] [-h number] [-r address] [-s subject] recipient... /usr/ucb/mail ... /usr/ucb/Mail ...</pre>
機能説明	The mail utilities listed above provide a comfortable, flexible environment for sending and receiving mail messages electronically. When reading mail, the mail utilities provide commands to facilitate saving, deleting, and responding to messages. When sending mail, the mail utilities allow editing, reviewing and other modification of the message as it is entered. Incoming mail is stored in a standard file for each user, called the mailbox for that user. When the mail utilities are called to read messages, the mailbox is the default place to find them. As messages are read, they are marked to be moved to a secondary file for storage, unless specific action is taken, so that the messages need not be seen again. This secondary file is called the mbox and is normally located in the user's HOME directory (see MBOX in ENVIRONMENT VARIABLES for a description of this file). Messages can be saved in other secondary files named by the user. Messages remain in a secondary file until forcibly removed. The user can access a secondary file by using the -f option. Messages in the secondary file can then be read or otherwise processed using the same Commands as in the primary mailbox. This gives rise within these pages to the notion of a current mailbox.
オプション	On the command line options start with a dash (–). Any other arguments are taken to be destinations (recipients). If no recipients are specified, mailx attempts to read messages from the mailbox. -B Do not buffer standard input or standard output. -b <i>bcc</i> Set the blind carbon copy list to <i>bcc</i>. <i>bcc</i> should be enclosed in quotes if it contains more than one name. -c <i>cc</i> Set the carbon copy list to <i>cc</i>. <i>cc</i> should be enclosed in quotes if it contains more than one name. -d Turn on debugging output. (Neither particularly interesting nor recommended.) -e Test for the presence of mail. mailx prints nothing and exits with a successful return code if there is mail to read. -F Record the message in a file named after the first recipient. Overrides the record variable, if set (see Internal Variables).

- f [*file*] Read messages from *file* instead of mailbox. If no *file* is specified, the mbox is used.
- f [+*folder*] Use the file *folder* in the folder directory (same as the folder command). The name of this directory is listed in the folder variable.
- H Print header summary only.
- h *number* The number of network “hops” made so far. This is provided for network software to avoid infinite delivery loops. This option and its argument are passed to the delivery program.
- I Include the newsgroup and article-id header lines when printing mail messages. This option requires the -f option to be specified.
- i Ignore interrupts. See also ignore in Internal Variables.
- N Do not print initial header summary.
- n Do not initialize from the system default mailx.rc or Mail.rc file. See USAGE.
- r *address* Use *address* as the return address when invoking the delivery program. All tilde commands are disabled. This option and its argument is passed to the delivery program.
- s *subject* Set the Subject header field to *subject*. *subject* should be enclosed in quotes if it contains embedded white space.
- T *file* Message-id and article-id header lines are recorded in *file* after the message is read. This option also sets the -I option.
- t Scan the input for To:, Cc:, and Bcc: fields. Any recipients on the command line will be ignored.
- U Convert UUCP-style addresses to internet standards. Overrides the conv environment variable.
- u *user* Read *user*'s mailbox. This is only effective if *user*'s mailbox is not read protected.
- V Print the mailx version number and exit.
- v Pass the -v flag to [sendmail\(1M\)](#).
- ~ Interpret tilde escapes in the input even if not reading from a tty.

オペランド

The following operands are supported:

recipient Addressee of message.

使用法

Starting Mail

At startup time, mailx executes the system startup file `/etc/mail/mailx.rc`. If invoked as `mail` or `Mail`, the system startup file `/etc/mail/Mail.rc` is used instead.

The system startup file sets up initial display options and alias lists and assigns values to some internal variables. These variables are flags and valued parameters which are set and cleared using the `set` and `unset` commands. See [Internal Variables](#).

With the following exceptions, regular commands are legal inside startup files: `!`, `Copy`, `edit`, `followup`, `Followup`, `hold`, `mail`, `preserve`, `reply`, `Reply`, `shell`, and `visual`. An error in the startup file causes the remaining lines in the file to be ignored.

After executing the system startup file, the mail utilities execute the optional personal startup file `$HOME/.mailrc`, wherein the user can override the values of the internal variables as set by the system startup file.

If the `-n` option is specified, however, the mail utilities do not execute the system startup file.

Many system administrators include the commands

```
set appenddeadletter
unset replyall
unset pipeignore
```

in the system startup files (to be compatible with past Solaris behavior), but this does not meet standards requirements for mailx. To get standard behavior for mailx, users should use the `-n` option or include the following commands in a personal startup file:

```
unset appenddeadletter
set replyall
set pipeignore
```

When reading mail, the mail utilities are in *command mode*. A header summary of the first several messages is displayed, followed by a prompt indicating the mail utilities can accept regular commands (see [Commands](#) below). When sending mail, the mail utilities are in *input mode*. If no subject is specified on the command line, and the `asksub` variable is set, a prompt for the subject is printed.

As the message is typed, the mail utilities read the message and store it in a temporary file. Commands may be entered by beginning a line with the tilde (`~`) escape character followed by a single command letter and optional arguments. See [Tilde Escapes](#) for a summary of these commands.

Reading Mail

Each message is assigned a sequential number, and there is at any time the notion of a current message, marked by a right angle bracket (`>`) in the header summary. Many commands take

an optional list of messages (*message-list*) to operate on. In most cases, the current message is set to the highest-numbered message in the list after the command is finished executing.

The default for *message-list* is the current message. A *message-list* is a list of message identifiers separated by spaces, which may include:

<i>n</i>	Message number <i>n</i> .										
.	The current message.										
^	The first undeleted message.										
\$	The last message.										
*	All messages.										
+	The next undeleted message.										
—	The previous undeleted message.										
<i>n–m</i>	An inclusive range of message numbers.										
<i>user</i>	All messages from <i>user</i> .										
<i>/string</i>	All messages with <i>string</i> in the Subject line (case ignored).										
<i>:c</i>	All messages of type <i>c</i> , where <i>c</i> is one of: <table><tr><td>d</td><td>deleted messages</td></tr><tr><td>n</td><td>new messages</td></tr><tr><td>o</td><td>old messages</td></tr><tr><td>r</td><td>read messages</td></tr><tr><td>u</td><td>unread messages</td></tr></table>	d	deleted messages	n	new messages	o	old messages	r	read messages	u	unread messages
d	deleted messages										
n	new messages										
o	old messages										
r	read messages										
u	unread messages										

Notice that the context of the command determines whether this type of message specification makes sense.

Other arguments are usually arbitrary strings whose usage depends on the command involved. Filenames, where expected, are expanded using the normal shell conventions (see [sh\(1\)](#)). Special characters are recognized by certain commands and are documented with the commands below.

Sending Mail

Recipients listed on the command line may be of three types: login names, shell commands, or alias groups. Login names may be any network address, including mixed network addressing. If mail is found to be undeliverable, an attempt is made to return it to the sender's mailbox. If the recipient name begins with a pipe symbol (|), the rest of the name is taken to be a shell command to pipe the message through. This provides an automatic interface with any

program that reads the standard input. Groups are set by the `alias` command (see `Commands` below) or in a system startup file (for example, `$HOME/.mailrc`). Aliases are lists of recipients of any type.

Forwarding Mail To forward a specific message, include it in a message to the desired recipients with the `~f` or `~m` tilde escapes. See `Tilde Escapes` below. To forward mail automatically, add a comma-separated list of addresses for additional recipients to the `.forward` file in your home directory. This is different from the format of the `alias` command, which takes a space-separated list instead. *Note:* Forwarding addresses must be valid, or the messages will “bounce.” You cannot, for instance, reroute your mail to a new host by forwarding it to your new address if it is not yet listed in the NIS aliases domain.

Commands Regular commands are of the form

[*command*] [*message-list*] [*arguments*]

In *input mode*, commands are recognized by the escape character, tilde (~), and lines not treated as commands are taken as input for the message. If no command is specified in *command mode*, next is assumed. The following is a complete list of mailx commands:

<code>!shell-command</code>	Escape to the shell. See SHELL in ENVIRONMENT VARIABLES.
<code># comment</code>	NULL command (comment). Useful in mailrc files.
<code>=</code>	Print the current message number.
<code>?</code>	Prints a summary of commands.
<code>alias alias name . . .</code> <code>group alias name . . .</code>	Declare an alias for the given names. The names are substituted when <code>alias</code> is used as a recipient. Useful in the mailrc file. With no arguments, the command displays the list of defined aliases.
<code>alternates name . . .</code>	Declare a list of alternate names for your login. When responding to a message, these names are removed from the list of recipients for the response. With no arguments, print the current list of alternate names. See also <code>allnet</code> in Internal Variables.

`cd` [*directory*]
`chdir` [*directory*]

Change directory. If *directory* is not specified, `$HOME` is used.

`copy` [*file*]
`copy` [*message-list*] *file*

Copy messages to the file without marking the messages as saved. Otherwise equivalent to the `save` command.

`Copy` [*message-list*]

Save the specified messages in a file whose name is derived from the author of the message to be saved, without marking the messages as saved. Otherwise equivalent to the `Save` command.

`delete` [*message-list*]

Delete messages from the mailbox. If `autoprint` is set, the next message after the last one deleted is printed (see `Internal Variables`).

`discard` [*header-field*. . .]
`ignore` [*header-field*. . .]

Suppress printing of the specified header fields when displaying messages on the screen. Examples of header fields to ignore are `Status` and `Received`. The fields are included when the message is saved, unless the `alwaysignore` variable is set. The `More`, `Page`, `Print`, and `Type` commands override this command. If no header is specified, the current list of header fields being ignored is printed. See also the `undiscard` and `unignore` commands.

`dp` [*message-list*]
`dt` [*message-list*]

Delete the specified messages from the mailbox and print the next message after the last one deleted. Roughly equivalent to a `delete` command followed by a `print` command.

`echo` *string*. . .

Echo the given strings (like `echo(1)`).

`edit` [*message-list*]

Edit the given messages. Each message is placed in a temporary file and the

	program named by the EDITOR variable is invoked to edit it (see ENVIRONMENT VARIABLES). Default editor is ed(1) .
exit xit	Exit from mailx, without changing the mailbox. No messages are saved in the mbox (see also quit).
field [<i>message-list</i>] header-file	Display the value of the header field in the specified message.
file [<i>file</i>] folder [<i>file</i>]	Quit from the current file of messages and read in the specified file. Several special characters are recognized when used as file names: % the current mailbox. %<i>user</i> the mailbox for <i>user</i>. # the previous mail file. & the current mbox. +<i>file</i> The named file in the <i>folder</i> directory (listed in the <i>folder</i> variable).
	With no arguments, print the name of the current mail file, and the number of messages and characters it contains.
folders	Print the names of the files in the directory set by the folder variable (see Internal Variables).
Followup [<i>message</i>]	Respond to a message, recording the response in a file whose name is derived from the author of the message. Overrides the record variable, if set. If the replyall variable is set, the actions of Followup and followup are reversed. See also the followup, Save, and Copy commands and outfolder in Internal Variables, and the Starting Mail section in USAGE above.

followup [*message-list*]

Respond to the first message in the *message-list*, sending the message to the author of each message in the *message-list*. The subject line is taken from the first message and the response is recorded in a file whose name is derived from the author of the first message. If the `replyall` variable is set, the actions of `followup` and `Followup` are reversed. See also the `Followup`, `Save`, and `Copy` commands and `outfolder` in *Internal Variables*, and the *Starting Mail* section in *USAGE* above.

from [*message-list*]

Print the header summary for the specified messages. If no messages are specified, print the header summary for the current message.

group *alias name* . . .
alias *alias name* . . .

Declare an alias for the given names. The names are substituted when `alias` is used as a recipient. Useful in the `mailrc` file.

headers [*message*]

Print the page of headers which includes the message specified. The `screen` variable sets the number of headers per page (see *Internal Variables*). See also the `z` command.

help

Print a summary of commands.

hold [*message-list*]

preserve [*message-list*]

Hold the specified messages in the mailbox.

if *s* | *r* | *t*
 mail-commands
else
 mail-commands
endif

Conditional execution, where *s* executes following *mail-commands*, up to an `else` or `endif`, if the program is in *send* mode, *r* causes the *mail-commands* to be

	executed only in <i>receive</i> mode, and <i>t</i> causes the <i>mail-commands</i> to be executed only if <i>mailx</i> is being run from a terminal. Useful in the <i>mailrc</i> file.
<i>inc</i>	Incorporate messages that arrive while you are reading the system mailbox. The new messages are added to the message list in the current <i>mail</i> session. This command does not commit changes made during the session, and prior messages are not renumbered.
<i>ignore</i> [<i>header-field</i> ...] <i>discard</i> [<i>header-field</i> ...]	Suppress printing of the specified header fields when displaying messages on the screen. Examples of header fields to ignore are <i>Status</i> and <i>Cc</i> . All fields are included when the message is saved. The <i>More</i> , <i>Page</i> , <i>Print</i> and <i>Type</i> commands override this command. If no header is specified, the current list of header fields being ignored is printed. See also the <i>undiscard</i> and <i>unignore</i> commands.
<i>list</i>	Print all commands available. No explanation is given.
<i>load</i>	[<i>message</i>] <i>file</i> The specified message is replaced by the message in the named file. <i>file</i> should contain a single mail message including mail headers (as saved by the <i>save</i> command).
<i>mail recipient</i> ...	Mail a message to the specified recipients.
<i>Mail recipient</i>	Mail a message to the specified recipients, and record it in a file whose name is derived from the author of the message. Overrides the <i>record</i> variable, if set. See also the <i>Save</i> and <i>Copy</i> commands and <i>outfolder</i> in <i>Internal Variables</i> .

`mbox` [*message-list*]

Arrange for the given messages to end up in the standard mbox save file when mailx terminates normally. See MBOX in ENVIRONMENT VARIABLES for a description of this file. See also the exit and quit commands.

`more` [*message-list*]

`page` [*message-list*]

Print the specified messages. If `crt` is set, the messages longer than the number of lines specified by the `crt` variable are paged through the command specified by the `PAGER` variable. The default command is `pg(1)` or if the `bsdcompat` variable is set, the default is `more(1)`. See ENVIRONMENT VARIABLES. Same as the print and type commands.

`More` [*message-list*]

`Page` [*message-list*]

Print the specified messages on the screen, including all header fields. Overrides suppression of fields by the ignore command. Same as the Print and Type commands.

`new` [*message-list*]

`New` [*message-list*]

`unread` [*message-list*]

`Unread`

[*message-list*] Take a message list and mark each message as *not* having been read.

`next` [*message*]

Go to the next message matching *message*. If message is not supplied, this command finds the next message that was not deleted or saved. A *message-list* may be specified, but in this case the first valid message in the list is the only one used. This is useful for jumping to the next message from a specific user, since the name would be taken as a command in the absence of a real command. See

```
pipe [message-list] [shell-command]
| [message-list] [shell-command]
```

the discussion of *message-list* above for a description of possible message specifications.

Pipe the message through the given *shell-command*. The message is treated as if it were read. If no arguments are given, the current message is piped through the command specified by the value of the `cmd` variable. If the `page` variable is set, a form feed character is inserted after each message (see *Internal Variables*).

```
preserve [message-list]
hold [message-list]
```

Preserve the specified messages in the mailbox.

```
print [message-list]
type [message-list]
```

Print the specified messages. If `crt` is set, the messages longer than the number of lines specified by the `crt` variable are paged through the command specified by the `PAGER` variable. The default command is [pg\(1\)](#) or if the `bsdcompat` variable is set, the default is [more\(1\)](#). See *ENVIRONMENT VARIABLES*. Same as the `more` and `page` commands.

```
Print [message-list]
Type [message-list]
```

Print the specified messages on the screen, including all header fields. Overrides suppression of fields by the `ignore` command. Same as the `More` and `Page` commands.

```
put [file]
put [message-list] file
```

Save the specified message in the given file. Use the same conventions as the `print` command for which header fields are ignored.

Put [*file*]

Put [*message-list*] *file*

Save the specified message in the given file. Overrides suppression of fields by the ignore command.

quit

Exit from mailx, storing messages that were read in mbox and unread messages in the mailbox. Messages that have been explicitly saved in a file are deleted unless the keepsave variable is set.

reply [*message-list*]

respond [*message-list*]

replysender [*message-list*]

Send a response to the author of each message in the *message-list*. The subject line is taken from the first message. If record is set to a file, a copy of the reply is added to that file. If the replyall variable is set, the actions of Reply/Respond and reply/respond are reversed. The replysender command is not affected by the replyall variable, but sends each reply only to the sender of each message. See the Starting Mail section in USAGE above.

Reply [*message*]

Respond [*message*]

replyall [*message*]

Reply to the specified message, including all other recipients of that message. If the variable record is set to a file, a copy of the reply added to that file. If the replyall variable is set, the actions of Reply/Respond and reply/respond are reversed. The replyall command is not affected by the replyall variable, but always sends the reply to all recipients of the message. See the Starting Mail section in USAGE above.

retain

Add the list of header fields named to the *retained list*. Only the header fields in the retain list are shown on your terminal when you print a message. All

Save [*message-list*]

save [*file*]
save [*message-list*] *file*

set
set *variable*
set *variable*=*string*
set *variable*=*number*

other header fields are suppressed. The set of retained fields specified by the retain command overrides any list of ignored fields specified by the ignore command. The Type and Print commands can be used to print a message in its entirety. If retain is executed with no arguments, it lists the current set of retained fields.

Save the specified messages in a file whose name is derived from the author of the first message. The name of the file is taken to be the author's name with all network addressing stripped off. See also the Copy, followup, and Followup commands and outfolder in Internal Variables.

Save the specified messages in the given file. The file is created if it does not exist. The file defaults to mbox. The message is deleted from the mailbox when mailx terminates unless keepsave is set (see also Internal Variables and the exit and quit commands).

Define a *variable*. To assign a *value* to *variable*, separate the variable name from the value by an '=' (there must be no space before or after the '='). A variable may be given a null, string, or numeric *value*. To embed SPACE characters within a *value*, enclose it in quotes.

With no arguments, set displays all defined variables and any values they might have. See Internal Variables for a description of all predefined mail variables.

shell	Invoke an interactive shell. See also SHELL in ENVIRONMENT VARIABLES.
size [<i>message-list</i>]	Print the size in characters of the specified messages.
source <i>file</i>	Read commands from the given file and return to command mode.
top [<i>message-list</i>]	Print the top few lines of the specified messages. If the top <code>lines</code> variable is set, it is taken as the number of lines to print (see Internal Variables). The default is 5.
touch [<i>message-list</i>]	Touch the specified messages. If any message in <i>message-list</i> is not specifically saved in a file, it is placed in the mbox, or the file specified in the MBOX environment variable, upon normal termination. See exit and quit.
Type [<i>message-list</i>] Print [<i>message-list</i>]	Print the specified messages on the screen, including all header fields. Overrides suppression of fields by the ignore command.
type [<i>message-list</i>] print [<i>message-list</i>]	Print the specified messages. If <code>crt</code> is set, the messages longer than the number of lines specified by the <code>crt</code> variable are paged through the command specified by the <code>PAGER</code> variable. The default command is pg(1) . See ENVIRONMENT VARIABLES.
unalias [<i>alias</i>] ... ungroup [<i>alias</i>] ...	Remove the definitions of the specified aliases.
undelete [<i>message-list</i>]	Restore the specified deleted messages. Will only restore messages deleted in the current mail session. If <code>autoprint</code> is set, the last message of those restored is printed (see Internal Variables).

<code>undiscard [header-field ...]</code> <code>unignore [header-field ...]</code>	Remove the specified header fields from the list being ignored. If no header fields are specified, all header fields are removed from the list being ignored.
<code>unretain [header-field ...]</code>	Remove the specified header fields from the list being retained. If no header fields are specified, all header fields are removed from the list being retained.
<code>unread [message-list]</code> <code>Unread [message-list]</code> Same as the new command.	
<code>unset variable ...</code>	Erase the specified variables. If the variable was imported from the environment (that is, an environment variable or exported shell variable), it cannot be unset from within <code>mailx</code> .
<code>version</code>	Print the current version and release date of the <code>mailx</code> utility.
<code>visual [message-list]</code>	Edit the given messages with a screen editor. Each messages is placed in a temporary file and the program named by the <code>VISUAL</code> variable is invoked to edit it (see <code>ENVIRONMENT VARIABLES</code>). Notice that the default visual editor is <code>vi</code> .
<code>write [message-list] file</code>	Write the given messages on the specified file, minus the header and trailing blank line. Otherwise equivalent to the <code>save</code> command.
<code>xit</code> <code>exit</code>	Exit from <code>mailx</code> , without changing the <code>mailbox</code> . No messages are saved in the <code>mbox</code> (see also <code>quit</code>).
<code>z[+ -]</code>	Scroll the header display forward or backward one screen—full. The number of headers displayed is set by the <code>screen</code> variable (see <code>Internal Variables</code>).

Tilde Escapes

The following tilde escape commands can be used when composing mail to send. These may be entered only from *input mode*, by beginning a line with the tilde escape character (~). See escape in Internal Variables for changing this special character. The escape character can be entered as text by typing it twice.

~ ! <i>shell-command</i>	Escape to the shell. If present, run <i>shell-command</i> .
~.	Simulate end of file (terminate message input).
~ : <i>mail-command</i>	
~_ <i>mail-command</i>	Perform the command-level request. Valid only when sending a message while reading mail.
~?	Print a summary of tilde escapes.
~A	Insert the autograph string Sign into the message (see Internal Variables).
~a	Insert the autograph string sign into the message (see Internal Variables).
~b <i>name</i> ...	Add the <i>names</i> to the blind carbon copy (Bcc) list. This is like the carbon copy (Cc) list, except that the names in the Bcc list are not shown in the header of the mail message.
~c <i>name</i> ...	Add the <i>names</i> to the carbon copy (Cc) list.
~d	Read in the dead-letter file. See DEAD in ENVIRONMENT VARIABLES for a description of this file.
~e	Invoke the editor on the partial message. See also EDITOR in ENVIRONMENT VARIABLES.
~f [<i>message-list</i>]	Forward the specified message, or the current message being read. Valid only when sending a message while reading mail. The messages are inserted into the message without alteration (as opposed to the ~m escape).
~F [<i>message-list</i>]	Forward the specified message, or the current message being read, including all header fields. Overrides the suppression of fields by the ignore command.
~h	Prompt for Subject line and To, Cc, and Bcc lists. If the field is displayed with an initial value, it may be edited as if you had just typed it.
~i <i>variable</i>	Insert the value of the named variable into the text of the message. For example, ~A is equivalent to '~i Sign.' Environment variables set and exported in the shell are also accessible by ~i.

<code>~m [message-list]</code>	Insert the listed messages, or the current message being read into the letter. Valid only when sending a message while reading mail. The text of the message is shifted to the right, and the string contained in the <code>indentprefix</code> variable is inserted as the leftmost characters of each line. If <code>indentprefix</code> is not set, a TAB character is inserted into each line.
<code>~M [message-list]</code>	Insert the listed messages, or the current message being read, including the header fields, into the letter. Valid only when sending a message while reading mail. The text of the message is shifted to the right, and the string contained in the <code>indentprefix</code> variable is inserted as the leftmost characters of each line. If <code>indentprefix</code> is not set, a TAB character is inserted into each line. Overrides the suppression of fields by the <code>ignore</code> command.
<code>~p</code>	Print the message being entered.
<code>~q</code>	Quit from input mode by simulating an interrupt. If the body of the message is not null, the partial message is saved in <code>dead-letter</code> . See <code>DEAD</code> in <code>ENVIRONMENT VARIABLES</code> for a description of this file.
<code>~R</code>	Mark message for return receipt.
<code>~r file</code>	Read in the specified file. If the argument begins with an exclamation point (!), the rest of the string is taken as an arbitrary shell command and is executed, with the standard output inserted into the message.
<code>~< file</code>	
<code>~< ! shell-command</code>	
<code>~s string ...</code>	Set the subject line to <i>string</i> .
<code>~t name ...</code>	Add the given <i>names</i> to the To list.
<code>~v</code>	Invoke a preferred screen editor on the partial message. The default visual editor is <code>vi(1)</code> . See also <code>VISUAL</code> in <code>ENVIRONMENT VARIABLES</code> .
<code>~w file</code>	Write the message into the given file, without the header.
<code>~x</code>	Exit as with <code>~q</code> except the message is not saved in <code>dead-letter</code> .
<code>~ shell-command</code>	Pipe the body of the message through the given <i>shell-command</i> . If the <i>shell-command</i> returns a successful exit status, the output of the command replaces the message.

Internal Variables

The following variables are internal variables. They may be imported from the execution environment or set using the `set` command at any time. The `unset` command may be used to erase variables.

allnet	All network names whose last component (login name) match are treated as identical. This causes the <i>message-list</i> message specifications to behave similarly. Disabled by default. See also the <code>alternates</code> command and the <code>metoo</code> and <code>fuzzymatch</code> variables.
alwaysignore	Ignore header fields with <code>ignore</code> everywhere, not just during print or type. Affects the <code>save</code> , <code>Save</code> , <code>copy</code> , <code>Copy</code> , <code>top</code> , <code>pipe</code> , and <code>write</code> commands, and the <code>~m</code> and <code>~f</code> tilde escapes. Enabled by default.
append	Upon termination, append messages to the end of the <code>mbox</code> file instead of prepending them. Although disabled by default, <code>append</code> is set in the system startup file (which can be suppressed with the <code>-n</code> command line option).
appenddeadletter	Append to the deadletter file rather than overwrite it. Although disabled by default, <code>appenddeadletter</code> is frequently set in the system startup file. See <code>Starting Mail</code> in <code>USAGE</code> above.
askbcc	Prompt for the Bcc list after the Subject is entered if it is not specified on the command line with the <code>-b</code> option. Disabled by default.
askcc	Prompt for the Cc list after the Subject is entered if it is not specified on the command line with the <code>-c</code> option. Disabled by default.
asksub	Prompt for subject if it is not specified on the command line with the <code>-s</code> option. Enabled by default.
autoinc	Automatically incorporate new messages into the current session as they arrive. This has an affect similar to issuing the <code>inc</code> command every time the command prompt is displayed. Disabled by default, but <code>autoinc</code> is set in the default system startup file for <code>mailx</code> ; it is not set for <code>/usr/ucb/mail</code> or <code>/usr/ucb/Mail</code> .
autoprint	Enable automatic printing of messages after delete and undelete commands. Disabled by default.
bang	Enable the special-casing of exclamation points (!) in shell escape command lines as in vi(1) . Disabled by default.
bsdcompat	Set automatically if <code>mailx</code> is invoked as <code>mail</code> or <code>Mail</code> . Causes <code>mailx</code> to use <code>/etc/mail/Mail.rc</code> as the system startup file. Changes the default pager to more(1) .

<code>cmd=shell-command</code>	Set the default command for the pipe command. No default value.
<code>conv=conversion</code>	Convert uucp addresses to the specified address style, which can be either: <code>internet</code> This requires a mail delivery program conforming to the RFC822 standard for electronic mail addressing. <code>optimize</code> Remove loops in <code>uucp(1C)</code> address paths (typically generated by the <code>reply</code> command). No rerouting is performed; <code>mail</code> has no knowledge of UUCP routes or connections. Conversion is disabled by default. See also <code>sendmail(1M)</code> and the <code>-U</code> command-line option.
<code>crt[=number]</code>	Pipe messages having more than <i>number</i> lines through the command specified by the value of the <code>PAGER</code> variable (<code>pg(1)</code> or <code>more(1)</code> by default). If <i>number</i> is not specified, the current window size is used. Disabled by default.
<code>debug</code>	Enable verbose diagnostics for debugging. Messages are not delivered. Disabled by default.
<code>dot</code>	Take a period on a line by itself, or EOF during input from a terminal as end-of-file. Disabled by default, but <code>dot</code> is set in the system startup file (which can be suppressed with the <code>-n</code> command line option).
<code>fcc</code>	By default, <code>mailx</code> will treat any address containing a slash (/) character as a local send to file address. By unsetting this option, this behavior is disabled. Enabled by default.
<code>flipr</code>	Reverse the effect of the <code>followup/Followup</code> and <code>reply/Reply</code> command pairs. If both <code>flipr</code> and <code>replyall</code> are set, the effect is as if neither was set.
<code>from</code>	Extract the author listed in the header summary from the <code>From:</code> header instead of the UNIX <code>From</code> line. Enabled by default.
<code>fuzzymatch</code>	The <code>from</code> command searches for messages from the indicated sender. By default, the full sender address must be specified. By setting this option, only a sub-string of the sender address need be specified. Disabled by default.

escape= <i>c</i>	Substitute <i>c</i> for the ~ escape character. Takes effect with next message sent.
folder= <i>directory</i>	The directory for saving standard mail files. User-specified file names beginning with a plus (+) are expanded by preceding the file name with this directory name to obtain the real file name. If <i>directory</i> does not start with a slash (/), \$HOME is prepended to it. There is no default for the folder variable. See also out folder below.
header	Enable printing of the header summary when entering mailx. Enabled by default.
hold	Preserve all messages that are read in the mailbox instead of putting them in the standard mbox save file. Disabled by default.
ignore	Ignore interrupts while entering messages. Handy for noisy dial-up lines. Disabled by default.
ignoreeof	Ignore end-of-file during message input. Input must be terminated by a period (.) on a line by itself or by the ~. command. See also dot above. Disabled by default.
indentprefix= <i>string</i>	When indentprefix is set, <i>string</i> is used to mark indented lines from messages included with ~m. The default is a TAB character.
keep	When the mailbox is empty, truncate it to zero length instead of removing it. Disabled by default.
iprompt= <i>string</i>	The specified prompt string is displayed before each line on input is requested when sending a message.
keepsave	Keep messages that have been saved in other files in the mailbox instead of deleting them. Disabled by default.
makeremote	When replying to all recipients of a message, if an address does not include a machine name, it is assumed to be relative to the sender of the message. Normally not needed when dealing with hosts that support RFC822.
metoo	If your login appears as a recipient, do not delete it from the list. Disabled by default.
mustbang	Force all mail addresses to be in bang format.
onehop	When responding to a message that was originally sent to several recipients, the other recipient addresses are normally forced to be relative to the originating author's machine for the response. This flag disables alteration of the recipients'

	addresses, improving efficiency in a network where all machines can send directly to all other machines (that is, one hop away). Disabled by default.
<code>outfolder</code>	Locate the files used to record outgoing messages in the directory specified by the <code>folder</code> variable unless the path name is absolute. Disabled by default. See <code>folder</code> above and the <code>Save</code> , <code>Copy</code> , <code>followup</code> , and <code>Followup</code> commands.
<code>page</code>	Used with the <code>pipe</code> command to insert a form feed after each message sent through the pipe. Disabled by default.
<code>pipeignore</code>	Omit ignored header when outputting to the <code>pipe</code> command. Although disabled by default, <code>pipeignore</code> is frequently set in the system startup file. See <code>Starting Mail</code> in <code>USAGE</code> above.
<code>postmark</code>	Your real name to be included in the <code>From</code> line of messages you send. By default this is derived from the <code>comment</code> field in your <code>passwd(4)</code> file entry.
<code>prompt=string</code>	Set the <i>command mode</i> prompt to <i>string</i> . Default is “ <code>? </code> ”, unless the <code>bsdcompat</code> variable is set, then the default is “ <code>& </code> ”.
<code>quiet</code>	Refrain from printing the opening message and version when entering <code>mailx</code> . Disabled by default.
<code>record=file</code>	Record all outgoing mail in <i>file</i> . Disabled by default. See also <code>outfolder</code> above.
<code>replyall</code>	Reverse the effect of the <code>reply</code> and <code>Reply</code> and <code>followup</code> and <code>Followup</code> commands. Although set by default, <code>replyall</code> is frequently unset in the system startup file. See <code>flipr</code> and <code>Starting Mail</code> in <code>USAGE</code> above.
<code>returnaddr=string</code>	The default sender address is that of the current user. This variable can be used to set the sender address to any arbitrary value. Set with caution.
<code>save</code>	Enable saving of messages in dead-letter on interrupt or delivery error. See <code>DEAD</code> for a description of this file. Enabled by default.
<code>screen=number</code>	Sets the number of lines in a screen-full of headers for the <code>headers</code> command. <i>number</i> must be a positive number. The default is set according to baud rate or window size. With a baud rate less than 1200, <i>number</i> defaults to 5, if baud rate is

exactly 1200, it defaults to 10. If you are in a window, *number* defaults to the default window size minus 4. Otherwise, the default is 20.

<code>sendmail=shell-command</code>	Alternate command for delivering messages. <i>Note:</i> In addition to the expected list of recipients, <code>mail</code> also passes the <code>-i</code> and <code>-m</code> , flags to the command. Since these flags are not appropriate to other commands, you may have to use a shell script that strips them from the arguments list before invoking the desired command. Default is <code>/usr/bin/rmail</code> .
<code>sendwait</code>	Wait for background mailer to finish before returning. Disabled by default.
<code>showname</code>	Causes the message header display to show the sender's real name (if known) rather than their mail address. Disabled by default, but <code>showname</code> is set in the <code>/etc/mail/mailx.rc</code> system startup file for <code>mailx</code> .
<code>showto</code>	When displaying the header summary and the message is from you, print the recipient's name instead of the author's name.
<code>sign=string</code>	The variable inserted into the text of a message when the <code>~a</code> (autograph) command is given. No default (see also <code>~i</code> in Tilde Escapes).
<code>Sign=string</code>	The variable inserted into the text of a message when the <code>~A</code> command is given. No default (see also <code>~i</code> in Tilde Escapes).
<code>toplines=number</code>	The number of lines of header to print with the <code>top</code> command. Default is 5.
<code>verbose</code>	Invoke sendmail(1M) with the <code>-v</code> flag.
<code>translate</code>	The name of a program to translate mail addresses. The program receives mail addresses as arguments. The program produces, on the standard output, lines containing the following data, in this order: ■ the postmark for the sender (see the <code>postmark</code> variable) ■ translated mail addresses, one per line, corresponding to the program's arguments. Each translated address will replace the corresponding address in the mail message being sent. ■ a line containing only <code>y</code> or <code>n</code>. if the line contains <code>y</code> the user will be asked to confirm that the message should be sent.

The translate program will be invoked for each mail message to be sent. If the program exits with a non-zero exit status, or fails to produce enough output, the message is not sent.

Large File Behavior	See largefile(5) for the description of the behavior of mailx when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of mailx: HOME, LANG, LC_CTYPE, LC_TIME, LC_MESSAGES, NLSPATH, and TERM.
DEAD	The name of the file in which to save partial letters in case of untimely interrupt. Default is \$HOME/dead.letter.
EDITOR	The command to run when the edit or ~e command is used. Default is ed(1) .
LISTER	The command (and options) to use when listing the contents of the folder directory. The default is ls(1) .
MAIL	The name of the initial mailbox file to read (in lieu of the standard system mailbox). The default is /var/mail/username .
MAILRC	The name of the startup file. Default is \$HOME/.mailrc.
MAILX_HEAD	The specified string is included at the beginning of the body of each message that is sent.
MAILX_TAIL	The specified string is included at the end of the body of each message that is sent.
MBOX	The name of the file to save messages which have been read. The exit command overrides this function, as does saving the message explicitly in another file. Default is \$HOME/mbox.
PAGER	The command to use as a filter for paginating output. This can also be used to specify the options to be used. Default is pg(1) , or if the bsdcompat variable is set, the default is more(1) . See Internal Variables.
SHELL	The name of a preferred command interpreter. Default is sh(1) .
VISUAL	The name of a preferred screen editor. Default is vi(1) .
終了ステータス	When the -e option is specified, the following exit values are returned: 0 Mail was found. >0 Mail was not found or an error occurred. Otherwise, the following exit values are returned: 0 Successful completion. Notice that this status implies that all messages were <i>sent</i>, but it gives no assurances that any of them were actually <i>delivered</i>.

	>0	An error occurred
ファイル	\$HOME/.mailrc	personal startup file
	\$HOME/mbox	secondary storage file
	\$HOME/.Maillock	lock file to prevent multiple writers of system mailbox
	/etc/mail/mailx.rc	optional system startup file for mailx only
	/etc/mail/Mail.rc	BSD compatibility system-wide startup file for /usr/ucb/mail and /usr/ucb/Mail
	/tmp/R[emqsx]*	temporary files
	/usr/share/lib/mailx/mailx.help*	help message files
	/var/mail/*	post office directory

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [biff\(1B\)](#), [echo\(1\)](#), [ed\(1\)](#), [ex\(1\)](#), [fmt\(1\)](#), [ls\(1\)](#), [mail\(1\)](#), [mail\(1B\)](#), [mailcompat\(1\)](#), [more\(1\)](#), [pg\(1\)](#), [sh\(1\)](#), [uucp\(1C\)](#), [vacation\(1\)](#), [vi\(1\)](#), [newaliases\(1M\)](#), [sendmail\(1M\)](#), [aliases\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 Where *shell-command* is shown as valid, arguments are not always allowed. Experimentation is recommended.

Internal variables imported from the execution environment cannot be unset.

The full internet addressing is not fully supported by mailx. The new standards need some time to settle down.

Replies do not always generate correct return addresses. Try resending the errant reply with onehop set.

mailx does not lock your record file. So, if you use a record file and send two or more messages simultaneously, lines from the messages may be interleaved in the record file.

The format for the alias command is a space-separated list of recipients, while the format for an alias in either the . forward or /etc/aliases is a comma-separated list.

To read mail on a workstation running Solaris 1.x when your mail server is running Solaris 2.x, first execute the `mailcompat(1)` program.

名前	make – maintain, update, and regenerate related programs and files
形式	<pre> /usr/bin/make [-d] [-dd] [-D] [-DD] [-e] [-i] [-k] [-n] [-p] [-P] [-q] [-r] [-s] [-S] [-t] [-u] [-w] [-V] [-f <i>makefile</i>]... [-K <i>statefile</i>]... [<i>target</i>]... [<i>macro</i> = <i>value</i>...] /usr/xpg4/bin/make [-d] [-dd] [-D] [-DD] [-e] [-i] [-k] [-n] [-p] [-P] [-q] [-r] [-s] [-S] [-t] [-u] [-w] [-V] [-f <i>makefile</i>]... [<i>target</i>]... [<i>macro</i> = <i>value</i>...] </pre>
機能説明	The make utility executes a list of shell commands associated with each <i>target</i>, typically to create or update a file of the same name. <i>makefile</i> contains entries that describe how to bring a target up to date with respect to those on which it depends, which are called <i>dependencies</i>. Since each dependency is a target, it can have dependencies of its own. Targets, dependencies, and sub-dependencies comprise a tree structure that make traces when deciding whether or not to rebuild a <i>target</i>. The make utility recursively checks each <i>target</i> against its dependencies, beginning with the first target entry in <i>makefile</i> if no <i>target</i> argument is supplied on the command line. If, after processing all of its dependencies, a target file is found either to be missing, or to be older than any of its dependencies, make rebuilds it. Optionally with this version of make, a target can be treated as out-of-date when the commands used to generate it have changed since the last time the target was built. To build a given target, make executes the list of commands, called a <i>rule</i>. This rule can be listed explicitly in the target's makefile entry, or it can be supplied implicitly by make. If no <i>target</i> is specified on the command line, make uses the first target defined in <i>makefile</i>. If a <i>target</i> has no makefile entry, or if its entry has no rule, make attempts to derive a rule by each of the following methods, in turn, until a suitable rule is found. Each method is described under 使用法 below. ■ Pattern matching rules. ■ Implicit rules, read in from a user-supplied makefile. ■ Standard implicit rules (also known as suffix rules), typically read in from the file <code>/usr/share/lib/make/make.rules</code>. ■ SCCS retrieval. make retrieves the most recent version from the SCCS history file (if any). See the description of the <code>.SCCS_GET</code>: special-function target for details. ■ The rule from the <code>.DEFAULT</code>: target entry, if there is such an entry in the makefile. If there is no makefile entry for a <i>target</i>, if no rule can be derived for building it, and if no file by that name is present, make issues an error message and halts.

オプション

The following options are supported:

- d Displays the reasons why *make* chooses to rebuild a target. *make* displays any and all dependencies that are newer. In addition, *make* displays options read in from the `MAKEFLAGS` environment variable.
- dd Displays the dependency check and processing in vast detail.
- D Displays the text of the makefiles read in.
- DD Displays the text of the makefiles, `make.rules` file, the state file, and all hidden-dependency reports.
- e Environment variables override assignments within makefiles.
- f *makefile* Uses the description file *makefile*. A – as the *makefile* argument denotes the standard input. The contents of *makefile*, when present, override the standard set of implicit rules and predefined macros. When more than one -f *makefile* argument pair appears, *make* uses the concatenation of those files, in order of appearance.

When no *makefile* is specified, `/usr/bin/make` tries the following in sequence, except when in POSIX mode (see `.POSIX` in 使用法):

- If there is a file named `makefile` in the working directory, *make* uses that file. If, however, there is an SCCS history file (`SCCS/s.makefile`) which is newer, *make* attempts to retrieve and use the most recent version.
- In the absence of the above file(s), if a file named `Makefile` is present in the working directory, *make* attempts to use it. If there is an SCCS history file (`SCCS/s.Makefile`) that is newer, *make* attempts to retrieve and use the most recent version.

When no *makefile* is specified, `/usr/bin/make` in POSIX mode and `/usr/xpg4/bin/make` try the following files in sequence:

- `./makefile`, `./Makefile`
- `s.makefile`, `SCCS/s.makefile`
- `s.Makefile`, `SCCS/s.Makefile`

- i Ignores error codes returned by commands. Equivalent to the special-function target `.IGNORE:.`
- k When a nonzero error status is returned by a rule, or when *make* cannot find a rule, abandons work on the current target, but continues with other dependency branches that do not depend on it.
- K *statefile* Uses the state file *statefile*. A – as the *statefile* argument denotes the standard input. The contents of *statefile*, when present, override the standard set of implicit rules and predefined macros. When more than one -K *statefile*

argument pair appears, make uses the concatenation of those files, in order of appearance. (See also `.KEEP_STATE` and `.KEEP_STATE_FILE` in the [Special-Function Targets](#) section).

- n No execution mode. Prints commands, but does not execute them. Even lines beginning with an @ are printed. However, if a command line contains a reference to the `$(MAKE)` macro, that line is always executed (see the discussion of `MAKEFLAGS` in [Reading Makefiles and the Environment](#)). When in POSIX mode, lines beginning with a “+” are executed.
- p Prints out the complete set of macro definitions and target descriptions.
- P Merely reports dependencies, rather than building them.
- q Question mode. make returns a zero or nonzero status code depending on whether or not the target file is up to date. When in POSIX mode, lines beginning with a “+” are executed.
- r Does not read in the default makefile `/usr/share/lib/make/make.rules`.
- s Silent mode. Does not print command lines before executing them. Equivalent to the special-function target `.SILENT:`.
- S Undoes the effect of the `-k` option. Stops processing when a non-zero exit status is returned by a command.
- t Touches the target files (bringing them up to date) rather than performing their rules. *Warning:* This can be *dangerous* when files are maintained by more than one person. When the `.KEEP_STATE:` target appears in the makefile, this option updates the state file just as if the rules had been performed. When in POSIX mode, lines beginning with a “+” are executed.
- u Unconditional build of targets. Even if a target is up to date, it is rebuilt. This might be useful for rebuilding all targets without cleaning.
- V Puts make into SysV mode. Refer to [sysV-make\(1\)](#) for respective details.
- w Print a message containing the working directory before and after other processing. This can be useful for tracking down errors from complicated nests of recursive make commands.
- x Puts make into the specified compatibility mode. The following compatibility modes are supported:
 1. Compatibility with POSIX:
`-x SUN_MAKE_COMPAT_MODE=POSIX`
 2. Compatibility with SUN make:
`-x SUN_MAKE_COMPAT_MODE=SUN`

3. Compatibility with GNU make (partially supported):

-x SUN_MAKE_COMPAT_MODE=GNU

4. Compatibility with /usr/lib/svr4.make:

-x SUN_MAKE_COMPAT_MODE=SVR4

オペランド

The following operands are supported:

target Target names, as defined in [使用法](#).

macro=value Macro definition. This definition overrides any regular definition for the specified macro within the makefile itself, or in the environment. However, this definition can still be overridden by conditional macro assignments.

使用法

The usage of make is described below:

Reading Makefiles and the Environment

When make first starts, it reads the MAKEFLAGS environment variable to obtain any of the following options specified present in its value: -d, -D, -e, -i, -k, -n, -p, -q, -r, -s, -S, or -t. Due to the implementation of POSIX.2 (see [POSIX.2\(5\)](#)), the MAKEFLAGS values contains a leading - character. The make utility then reads the command line for additional options, which also take effect.

Next, make reads in a default makefile that typically contains predefined macro definitions, target entries for implicit rules, and additional rules, such as the rule for retrieving SCCS files. If present, make uses the file `make.rules` in the current directory; otherwise it reads the file `/usr/share/lib/make/make.rules`, which contains the standard definitions and rules. Use the directive:

```
include /usr/share/lib/make/make.rules
```

in your local `make.rules` file to include them.

Next, make imports variables from the environment (unless the -e option is in effect), and treats them as defined macros. Because make uses the most recent definition it encounters, a macro definition in the makefile normally overrides an environment variable of the same name. When -e is in effect, however, environment variables are read *after* all makefiles have been read. In that case, the environment variables take precedence over definitions in the makefile.

Next, make reads any makefiles you specify with -f, or one of `makefile` or `Makefile` as described above and then the state file, in the local directory if it exists. If the makefile contains a `.KEEP_STATE_FILE` target, then it reads the state file that follows the target. Refer to special target `.KEEP_STATE_FILE` for details.

Next (after reading the environment if -e is in effect), make reads in any macro definitions supplied as command line arguments. These override macro definitions in the makefile and the environment both, but only for the make command itself.

make exports environment variables, using the most recently defined value. Macro definitions supplied on the command line are not normally exported, unless the macro is also an environment variable.

make does not export macros defined in the makefile. If an environment variable is set, and a macro with the same name is defined on the command line, make exports its value as defined on the command line. Unless -e is in effect, macro definitions within the makefile take precedence over those imported from the environment.

The macros MAKEFLAGS, MAKE, SHELL, HOST_ARCH, HOST_MACH, and TARGET_MACH are special cases. See [Special-Purpose Macros](#) below for details.

Makefile Target Entries A target entry has the following format:

```
target [:::] [dependency] ... [; command] ... [command] ...
```

The first line contains the name of a target, or a space-separated list of target names, terminated with a colon or double colon. If a list of targets is given, this is equivalent to having a separate entry of the same form for each target. The colon(s) can be followed by a *dependency*, or a dependency list. make checks this list before building the target. The dependency list can be terminated with a semicolon (;), which in turn can be followed by a single Bourne shell command. Subsequent lines in the target entry begin with a TAB and contain Bourne shell commands. These commands comprise the rule for building the target.

Shell commands can be continued across input lines by escaping the NEWLINE with a backslash (\). The continuing line must also start with a TAB.

To rebuild a target, make expands macros, strips off initial TAB characters and either executes the command directly (if it contains no shell metacharacters), or passes each command line to a Bourne shell for execution.

The first *non-empty* line that does not begin with a TAB or # begins another target or macro definition.

Special Characters Special characters are defined below.

<i>Global</i>	#	Start a comment. The comment ends at the next NEWLINE. If the # follows the TAB in a command line, that line is passed to the shell (which also treats # as the start of a comment).
	include <i>filename</i>	If the word include appears as the first seven letters of a line and is followed by a SPACE or TAB, the string that follows is taken as a filename to interpolate at that line. include files can be nested to a depth of no more than about 16. If <i>filename</i> is a macro reference, it is expanded. If <i>filename</i> is surrounded by double quotes, make searches for a <i>filename</i> with relation to current

makefile path. If not, make is supposed to find it with relation to *path* where make was launched.

Targets and Dependencies

:	Target list terminator. Words following the colon are added to the dependency list for the target or targets. If a target is named in more than one colon-terminated target entry, the dependencies for all its entries are added to form that target's complete dependency list.
::	Target terminator for alternate dependencies. When used in place of a : the double-colon allows a target to be checked and updated with respect to alternate dependency lists. When the target is out-of-date with respect to dependencies listed in the first alternate, it is built according to the rule for that entry. When out-of-date with respect to dependencies in another alternate, it is built according the rule in that other entry. Implicit rules do not apply to double-colon targets; you must supply a rule for each entry. If no dependencies are specified, the rule is always performed.
<i>target</i> [+ <i>target</i> . . .] :	Target group. The rule in the target entry builds all the indicated targets as a group. It is normally performed only once per make run, but is checked for command dependencies every time a target in the group is encountered in the dependency scan.
%	Pattern matching wild card metacharacter. Like the * shell wild card, % matches any string of zero or more characters in a target name or dependency, in the target portion of a conditional macro definition, or within a pattern replacement macro reference. Notice that only one % can appear in a target, dependency-name, or pattern-replacement macro reference.
./ <i>pathname</i>	make ignores the leading ./ characters from targets with names given as pathnames relative to “dot,” the working directory.

Macros

=	Macro definition. The word to the left of this character is the macro name; words to the right comprise its value. Leading and trailing white space characters are stripped from the value. A word break following the = is implied.
\$	Macro reference. The following character, or the parenthesized or bracketed string, is interpreted as a macro reference; make expands the reference (including the \$) by replacing it with the macro's value.
()	Macro-reference name delimiters. A parenthesized or bracketed word appended to a \$ is taken as the name of the macro being referred to. Without the delimiters, make recognizes only the first character as the macro name.
{ }	
\$\$	A reference to the dollar-sign macro, the value of which is the character \$. Used to pass variable expressions beginning with \$ to the shell, to refer to environment

variables which are expanded by the shell, or to delay processing of dynamic macros within the dependency list of a target, until that target is actually processed.

- `\$` Escaped dollar-sign character. Interpreted as a literal dollar sign within a rule.
- `+=` When used in place of `=`, appends a string to a macro definition (must be surrounded by white space, unlike `=`).
- `:=` Conditional macro assignment. When preceded by a list of targets with explicit target entries, the macro definition that follows takes effect when processing only those targets, and their dependencies.
- `:sh =` Define the value of a macro to be the output of a command (see [Command Substitutions](#) below).
- `:sh` In a macro reference, execute the command stored in the macro, and replace the reference with the output of that command (see [Command Substitutions](#) below).

- Rules*
- `+` `make` always executes the commands preceded by a “+”, even when `-n` is specified.
 - `—` `make` ignores any nonzero error code returned by a command line for which the first non-TAB character is a `—`. This character is not passed to the shell as part of the command line. `make` normally terminates when a command returns nonzero status, unless the `-i` or `-k` options, or the `.IGNORE:` special-function target is in effect.
 - `@` If the first non-TAB character is a `@`, `make` does not print the command line before executing it. This character is not passed to the shell.
 - `?` Escape command-dependency checking. Command lines starting with this character are not subject to command dependency checking.
 - `!` Force command-dependency checking. Command-dependency checking is applied to command lines for which it would otherwise be suppressed. This checking is normally suppressed for lines that contain references to the `?` dynamic macro (for example, `$?`).
- When any combination of `+`, `—`, `@`, `?`, or `!` appear as the first characters after the TAB, all that are present apply. None are passed to the shell.

- Special-Function Targets*
- When incorporated in a makefile, the following target names perform special-functions:
- `.DEFAULT:` If it has an entry in the makefile, the rule for this target is used to process a target when there is no other entry for it, no rule for building it, and no SCCS history file from which to retrieve a current version. `make` ignores any dependencies for this target.
 - `.DONE:` If defined in the makefile, `make` processes this target and its dependencies after all other targets are built. This target is also performed when `make` halts with an error, unless the `.FAILED` target is defined.

<code>.FAILED:</code>	This target, along with its dependencies, is performed instead of <code>.DONE</code> when defined in the makefile and <code>make</code> halts with an error.
<code>.GET_POSIX:</code>	This target contains the rule for retrieving the current version of an SCCS file from its history file in the current working directory. <code>make</code> uses this rule when it is running in POSIX mode.
<code>.IGNORE:</code>	Ignore errors. When this target appears in the makefile, <code>make</code> ignores non-zero error codes returned from commands. When used in POSIX mode, <code>.IGNORE</code> could be followed by target names only, for which the errors is ignored.
<code>.INIT:</code>	If defined in the makefile, this target and its dependencies are built before any other targets are processed.
<code>.KEEP_STATE:</code>	If this target is in effect, <code>make</code> updates the state file, <code>.make.state</code> , in the current directory. This target also activates command dependencies, and hidden dependency checks. If either the <code>.KEEP_STATE:</code> target appears in the makefile, or the environment variable <code>KEEP_STATE</code> is set (<code>setenv KEEP_STATE</code>), <code>make</code> rebuilds everything in order to collect dependency information, even if all the targets were up to date due to previous <code>make</code> runs. See also the 環境変数 section. This target has no effect if used in POSIX mode.
<code>.KEEP_STATE_FILE:</code>	This target has no effect if used in POSIX mode. This target implies <code>.KEEP_STATE</code> . If the target is followed by a filename, <code>make</code> uses it as the state file. If the target is followed by a directory name, <code>make</code> looks for a <code>.make.state</code> file in that directory. If the target is not followed by any name, <code>make</code> looks for <code>.make.state</code> file in the current working directory.
<code>.MAKE_VERSION:</code>	A target-entry of the form: <code>.MAKE_VERSION: VERSION-number</code> enables version checking. If the version of <code>make</code> differs from the version indicated by a string like <code>VERSION-1.0</code> , <code>make</code> issues a warning message.
<code>.NO_PARALLEL:</code>	Currently, this target has no effect, it is, however, reserved for future use.
<code>.PARALLEL:</code>	Currently of no effect, but reserved for future use.
<code>.POSIX:</code>	This target enables POSIX mode.
<code>.PRECIOUS:</code>	List of files not to delete. <code>make</code> does not remove any of the files listed as dependencies for this target when interrupted. <code>make</code> normally

removes the current target when it receives an interrupt. When used in POSIX mode, if the target is not followed by a list of files, all the file are assumed precious.

.SCCS_GET: This target contains the rule for retrieving the current version of an SCCS file from its history file. To suppress automatic retrieval, add an entry for this target with an empty rule to your makefile.

.SCCS_GET_POSIX: This target contains the rule for retrieving the current version of an SCCS file from its history file. make uses this rule when it is running in POSIX mode.

.SILENT: Run silently. When this target appears in the makefile, make does not echo commands before executing them. When used in POSIX mode, it could be followed by target names, and only those are executed silently.

.SUFFIXES: The suffixes list for selecting implicit rules (see [The Suffixes List](#)).

.WAIT: Currently of no effect, but reserved for future use.

Clearing Special Targets

In this version of make, you can clear the definition of the following special targets by supplying entries for them with no dependencies and no rule:

.DEFAULT, **.SCCS_GET**, and **.SUFFIXES**

Command Dependencies

When the **.KEEP_STATE:** target is effective, make checks the command for building a target against the state file. If the command has changed since the last make run, make rebuilds the target.

Hidden Dependencies

When the **.KEEP_STATE:** target is effective, make reads reports from [cpp\(1\)](#) and other compilation processors for any “hidden” files, such as `#include` files. If the target is out of date with respect to any of these files, make rebuilds it.

Macros

Entries of the form

macro=value

define macros. *macro* is the name of the macro, and *value*, which consists of all characters up to a comment character or unescaped NEWLINE, is the value. make strips both leading and trailing white space in accepting the value.

Subsequent references to the macro, of the forms `$(name)` or `${name}` are replaced by *value*. The parentheses or brackets can be omitted in a reference to a macro with a single-character name.

Macro references can contain references to other macros, in which case nested references are expanded first.

Suffix Replacement Macro References

Substitutions within macros can be made as follows:

```
$(name:string1=string2)
```

where *string1* is either a suffix, or a word to be replaced in the macro definition, and *string2* is the replacement suffix or word. Words in a macro value are separated by SPACE, TAB, and escaped NEWLINE characters.

Pattern Replacement Macro References

Pattern matching replacements can also be applied to macros, with a reference of the form:

```
$(name: op%os= np%ns)
```

where *op* is the existing (old) prefix and *os* is the existing (old) suffix, *np* and *ns* are the new prefix and new suffix, respectively, and the pattern matched by % (a string of zero or more characters), is carried forward from the value being replaced. For example:

```
PROGRAM=fabricate
DEBUG= $(PROGRAM:%=tmp/%-g)
```

sets the value of DEBUG to tmp/fabricate-g.

Notice that pattern replacement macro references cannot be used in the dependency list of a pattern matching rule; the % characters are not evaluated independently. Also, any number of % metacharacters can appear after the equal-sign.

Appending to a Macro

Words can be appended to macro values as follows:

```
macro += word . . .
```

Special-Purpose Macros

When the MAKEFLAGS variable is present in the environment, make takes options from it, in combination with options entered on the command line. make retains this combined value as the MAKEFLAGS macro, and exports it automatically to each command or shell it invokes.

Notice that flags passed by way of MAKEFLAGS are only displayed when the -d, or -dd options are in effect.

The MAKE macro is another special case. It has the value make by default, and temporarily overrides the -n option for any line in which it is referred to. This allows nested invocations of make written as:

```
$(MAKE) . . .
```

to run recursively, with the -n flag in effect for all commands but make. This lets you use make -n to test an entire hierarchy of makefiles.

For compatibility with the 4.2 BSD make, the MFLAGS macro is set from the MAKEFLAGS variable by prepending a -. MFLAGS is not exported automatically.

The SHELL macro, when set to a single-word value such as `/usr/bin/csh`, indicates the name of an alternate shell to use. The default is `/bin/sh`. Notice that `make` executes commands that contain no shell metacharacters itself. Built-in commands, such as `dirs` in the C shell, are not recognized unless the command line includes a metacharacter (for instance, a semicolon). This macro is neither imported from, nor exported to the environment, regardless of `-e`. To be sure it is set properly, you must define this macro within every makefile that requires it.

The syntax of the VPATH macro is:

```
VPATH = [ pathname [ : pathname ] ... ]
```

VPATH specifies a list of directories to search for the files, which are targets or dependencies, when `make` is executed. VPATH is also used in order to search for the `include` files mentioned in the particular makefile.

When processing a target or a dependency or an include directive, `make` checks the existence of the file with the same name in the current directory. If the file is found to be missing, `make` searches for this file in the list of directories presented in VPATH (like the PATH variable in the shell). Unlike the PATH variable, VPATH is used in order to search for the files with relative pathnames. When `make` attempts to apply implicit rules to the target, it also searches for the dependency files using VPATH.

When the file is found using VPATH, internal macros `$(D)`, `$(F)`, `$(S)`, `$(*)`, and their alternative forms (with D or F appended) are set in accordance with the name derived from VPATH. For instance, if the target `subdir/foo.o` is found in the directory `/aaa/bbb` using VPATH, then the value of the internal macro `$(D)` for this target is `/aaa/bbb/subdir/foo.o`.

If a target or a dependency file is found using VPATH, then any occurrences of the word that is the same as the target name in the subsequent rules are replaced with the actual name of the target derived from VPATH.

For example:

```
VPATH=./subdir
file.o : file.c
        cc -c file.c -o file.o
```

If `file.c` is found in `./subdir`, then the command

```
cc -c ./subdir/file.c -o file.o
```

are executed.

The following macros are provided for use with cross-compilation:

HOST_ARCH	The processor type of the host system. By default, this is the output of the <code>mach(1)</code> command, prepended with <code>-</code> . Under normal circumstances, this value should never be altered by the user.

HOST_MACH	The machine architecture of the host system. By default, this is the output of the arch(1) command, prepended with <code>-</code> . Under normal circumstances, this value should never be altered by the user.
TARGET_ARCH	The processor type of the target system. By default, the output of <code>mach</code> , prepended with <code>-</code> .

Dynamic Macros

There are several dynamically maintained macros that are useful as abbreviations within rules. They are shown here as references; if you were to define them, `make` would simply override the definition.

<code>\$*</code>	The basename of the current target, derived as if selected for use with an implicit rule.
<code>\$<</code>	The name of a dependency file, derived as if selected for use with an implicit rule.
<code>\$@</code>	The name of the current target. This is the only dynamic macro whose value is strictly determined when used in a dependency list. (In which case it takes the form <code>\$\$@</code> .)
<code>\$?</code>	The list of dependencies that are newer than the target. Command-dependency checking is automatically suppressed for lines that contain this macro, just as if the command had been prefixed with a <code>?</code> . See the description of <code>?</code> , under Special Character Rules above. You can force this check with the <code>!</code> command-line prefix.
<code>\$\$</code>	The name of the library member being processed. (See Library Maintenance below.)

To refer to the `$@` dynamic macro within a dependency list, precede the reference with an additional `$` character (as in, `$$@`). Because `make` assigns `$<` and `$*` as it would for implicit rules (according to the suffixes list and the directory contents), they can be unreliable when used within explicit target entries.

These macros can be modified to apply either to the filename part, or the directory part of the strings they stand for, by adding an upper case `F` or `D`, respectively (and enclosing the resulting name in parentheses or braces). Thus, `$(@D)` refers to the directory part of the string `$@`; if there is no directory part, `.` is assigned. `$(@F)` refers to the filename part.

Conditional Macro Definitions

A macro definition of the form:

```
target-list := macro = value
```

indicates that when processing any of the targets listed *and their dependencies*, `macro` is to be set to the `value` supplied. Notice that if a conditional macro is referred to in a dependency list, the `$` must be delayed (use `$$` instead). Also, `target-list` can contain a `%` pattern, in which case the macro is conditionally defined for all targets encountered that match the pattern. A pattern replacement reference can be used within the `value`.

You can temporarily append to a macros value with a conditional definition of the form:

```
target-list := macro += value
```

Predefined Macros make supplies the macros shown in the table that follows for compilers and their options, host architectures, and other commands. Unless these macros are read in as environment variables, their values are not exported by make. If you run make with any of these set in the environment, it is a good idea to add commentary to the makefile to indicate what value each is expected to take. If -r is in effect, make does not read the default makefile (./make.rules or /usr/share/lib/make/make.rules) in which these macro definitions are supplied.

Table of Predefined Macros		
Use	Macro	Default Value
Library Archives	AR	ar
	ARFLAGS	rv
Assembler Commands	AS	as
	ASFLAGS	
	COMPILE.s	\$(AS) \$(ASFLAGS)
	COMPILE.S	\$(CC) \$(ASFLAGS) \$(CPPFLAGS) -c
C Compiler Commands	CC	cc
	CFLAGS	
	CPPFLAGS	
	COMPILE.c	\$(CC) \$(CFLAGS) \$(CPPFLAGS) -c
	LINK.c	\$(CC) \$(CFLAGS) \$(CPPFLAGS) \$(LDFLAGS)
C++ Compiler Commands	CCC	CC
	CCFLAGS	CFLAGS
	CPPFLAGS	
	COMPILE.cc	\$(CCC) \$(CCFLAGS) \$(CPPFLAGS) -c
	LINK.cc	\$(CCC) \$(CCFLAGS) \$(CPPFLAGS) \$(LDFLAGS)
	COMPILE.C	\$(CCC) \$(CCFLAGS) \$(CPPFLAGS) -c
	LINK.C	\$(CCC) \$(CCFLAGS) \$(CPPFLAGS) \$(LDFLAGS)

Table of Predefined Macros		
Use	Macro	Default Value
FORTRAN 77 Compiler Commands	FC	f77
	FFLAGS	
	COMPILE.f	\$(FC) \$(FFLAGS) -c
	LINK.f	\$(FC) \$(FFLAGS) \$(LDFLAGS)
	COMPILE.F	\$(FC) \$(FFLAGS) \$(CPPFLAGS) -c
	LINK.F	\$(FC) \$(FFLAGS) \$(CPPFLAGS) \$(LDFLAGS)
FORTRAN 90 Compiler Commands	FC	f90
	F90FLAGS	
	COMPILE.f90	\$(F90C) \$(F90FLAGS) -c
	LINK.f90	\$(F90C) \$(F90FLAGS) \$(LDFLAGS)
	COMPILE.ftn	\$(F90C) \$(F90FLAGS) \$(CPPFLAGS) -c
	LINK.ftn	\$(F90C) \$(F90FLAGS) \$(CPPFLAGS) \$(LDFLAGS)
Link Editor Command	LD	ld
	LDFLAGS	
lex Command	LEX	lex
	LFLAGS	
	LEX.l	\$(LEX) \$(LFLAGS) -t
lint Command	LINT	lint
	LINTFLAGS	
	LINT.c	\$(LINT) \$(LINTFLAGS) \$(CPPFLAGS)
Modula 2 Commands	M2C	m2c
	M2FLAGS	

Table of Predefined Macros		
Use	Macro	Default Value
	MODFLAGS DEFFLAGS COMPILE.def COMPILE.mod	 \$(M2C) \$(M2FLAGS) \$(DEFFLAGS) \$(M2C) \$(M2FLAGS) \$(MODFLAGS)
Pascal Compiler Commands	PC PFLAGS COMPILE.p LINK.p	pc \$(PC) \$(PFLAGS) \$(CPPFLAGS) -c \$(PC) \$(PFLAGS) \$(CPPFLAGS) \$(LDFLAGS)
Ratfor Compilation Commands	RFLAGS COMPILE.r LINK.r	 \$(FC) \$(FFLAGS) \$(RFLAGS) -c \$(FC) \$(FFLAGS) \$(RFLAGS) \$(LDFLAGS)
rm Command	RM	rm -f
sccs Command	SCCSFLAGS SCCSGETFLAGS	 -s
yacc Command	YACC YFLAGS YACC.y	yacc \$(YACC) \$(YFLAGS)
Suffixes List	SUFFIXES	.o .c .c~ .cc .cc~ .y .y~ .l .l~ .s .s~ .sh .sh~ .S .S~ .ln .h .h~ .f .f~ .F .F~ .mod .mod~ .sym .def .def~ .p .p~ .r .r~ .cps .cps~ .C .C~ .Y .Y~ .L .L .f90 .f90~ .ftn .ftn~

Implicit Rules When a target has no entry in the makefile, make attempts to determine its class (if any) and apply the rule for that class. An implicit rule describes how to build any target of a given class, from an associated dependency file. The class of a target can be determined either by a pattern, or by a suffix; the corresponding dependency file (with the same basename) from which such a target might be built. In addition to a predefined set of implicit rules, make allows you to define your own, either by pattern, or by suffix.

Pattern Matching Rules

A target entry of the form:

tp%ts: dp%ds
rule

is a pattern matching rule, in which *tp* is a target prefix, *ts* is a target suffix, *dp* is a dependency prefix, and *ds* is a dependency suffix (any of which can be null). The % stands for a basename of zero or more characters that is matched in the target, and is used to construct the name of a dependency. When make encounters a match in its search for an implicit rule, it uses the rule in that target entry to build the target from the dependency file. Pattern-matching implicit rules typically make use of the \$@ and \$< dynamic macros as placeholders for the target and dependency names. Other, regular dependencies can occur in the dependency list; however, none of the regular dependencies can contain %. An entry of the form:

tp%ts: [dependency...] dp%ds [dependency...]
rule

is a valid pattern matching rule.

Suffix Rules

When no pattern matching rule applies, make checks the target name to see if it ends with a suffix in the known suffixes list. If so, make checks for any suffix rules, as well as a dependency file with same root and another recognized suffix, from which to build it.

The target entry for a suffix rule takes the form:

DsTs: rule

where *Ts* is the suffix of the target, *Ds* is the suffix of the dependency file, and *rule* is the rule for building a target in the class. Both *Ds* and *Ts* must appear in the suffixes list. (A suffix need not begin with a . to be recognized.)

A suffix rule with only one suffix describes how to build a target having a null (or no) suffix from a dependency file with the indicated suffix. For instance, the .c rule could be used to build an executable program named file from a C source file named file.c. If a target with a null suffix has an explicit dependency, make omits the search for a suffix rule.

Table of Standard Implicit (Suffix) Rules for Assembly Files	
Implicit Rule Name	Command Line
.s.o	\$(COMPILE.s) -o \$@ \$<

Table of Standard Implicit (Suffix) Rules for Assembly Files	
Implicit Rule Name	Command Line
.s.a	\$(COMPILE.s) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.s~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.s \$(COMPILE.s) -o \$@ \$*.s
.S.o	\$(COMPILE.S) -o \$@ \$<
.S.a	\$(COMPILE.S) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.S~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.S \$(COMPILE.S) -o \$@ \$*.S
.S~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.S \$(COMPILE.S) -o \$% \$*.S \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%

Table of Standard Implicit (Suffix) Rules for C Files	
Implicit Rule Name	Command Line
.c	\$(LINK.c) -o \$@ \$< \$(LDLIBS)
.c.ln	\$(LINT.c) \$(OUTPUT_OPTION) -i \$<

<i>Table of Standard Implicit (Suffix) Rules for C Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.c.o</code>	<code>\$(COMPILE.c) \$(OUTPUT_OPTION) \$<</code>
<code>.c.a</code>	<code>\$(COMPILE.c) -o \$% \$<</code> <code>\$(AR) \$(ARFLAGS) \$@ \$%</code> <code>\$(RM) \$%</code>
<code>.c~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.c</code> <code>\$(CC) \$(CFLAGS) \$(LDFLAGS) -o \$@ \$*.c</code>
<code>.c~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.c</code> <code>\$(CC) \$(CFLAGS) -c \$*.c</code>
<code>.c~.ln</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.c</code> <code>\$(LINT.c) \$(OUTPUT_OPTION) -c \$*.c</code>
<code>.c~.a</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.c</code> <code>\$(COMPILE.c) -o \$% \$*.c</code> <code>\$(AR) \$(ARFLAGS) \$@ \$%</code> <code>\$(RM) \$%</code>

<i>Table of Standard Implicit (Suffix) Rules for C++ Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.cc</code>	<code>\$(LINK.cc) -o \$@ \$< \$(LDLIBS)</code>
<code>.cc.o</code>	<code>\$(COMPILE.cc) \$(OUTPUT_OPTION) \$<</code>

Table of Standard Implicit (Suffix) Rules for C++ Files	
Implicit Rule Name	Command Line
.cc.a	\$(COMPILE.cc) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.cc~	\$(GET) \$(GFLAGS) -p \$< > \$*.cc \$(LINK.cc) -o \$@ \$*.cc \$(LDLIBS)
.cc.o	\$(COMPILE.cc) \$(OUTPUT_OPTION) \$<
.cc~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.cc \$(COMPILE.cc) \$(OUTPUT_OPTION) \$*.cc
.cc.a	\$(COMPILE.cc) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.cc~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.cc \$(COMPILE.cc) -o \$% \$*.cc \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.C	\$(LINK.C) -o \$@ \$< \$(LDLIBS)
.C~	\$(GET) \$(GFLAGS) -p \$< > \$*.C \$(LINK.C) -o \$@ \$*.C \$(LDLIBS)
.C.o	\$(COMPILE.C) \$(OUTPUT_OPTION) \$<

Table of Standard Implicit (Suffix) Rules for C++ Files	
Implicit Rule Name	Command Line
.C~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.C \$(COMPILE.C) \$(OUTPUT_OPTION) \$*.C
.C.a	\$(COMPILE.C) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.C~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.C \$(COMPILE.C) -o \$% \$*.C \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%

Table of Standard Implicit (Suffix) Rules for FORTRAN 77 Files	
Implicit Rule Name	Command Line
.f	\$(LINK.f) -o \$@ \$< \$(LDLIBS)
.f.o	\$(COMPILE.f) \$(OUTPUT_OPTION) \$<
.f.a	\$(COMPILE.f) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.f	\$(LINK.f) -o \$@ \$< \$(LDLIBS)
.f~	
	\$(GET) \$(GFLAGS) -p \$< > \$*.f \$(FC) \$(FFLAGS) \$(LDFFLAGS) -o \$@ \$*.f

Table of Standard Implicit (Suffix) Rules for FORTRAN 77 Files	
Implicit Rule Name	Command Line
.f~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.f \$(FC) \$(FFLAGS) -c \$*.f
.f~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.f \$(COMPILE.f) -o \$% \$*.f \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.F	\$(LINK.F) -o \$@ \$< \$(LDLIBS)
.F.o	\$(COMPILE.F) \$(OUTPUT_OPTION) \$<
.F.a	\$(COMPILE.F) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.F~	\$(GET) \$(GFLAGS) -p \$< > \$*.F \$(FC) \$(FFLAGS) \$(LDFLAGS) -o \$@ \$*.F
.F~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.F \$(FC) \$(FFLAGS) -c \$*.F
.F~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.F \$(COMPILE.F) -o \$% \$*.F \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%

<i>Table of Standard Implicit (Suffix) Rules for FORTRAN 90 Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.f90</code>	<code>\$(LINK.f90) -o \$\$ \$< \$(LDLIBS)</code>
<code>.f90~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.f90</code> <code>\$(LINK.f90) -o \$\$ \$*.f90 \$(LDLIBS)</code>
<code>.f90.o</code>	<code>\$(COMPILE.f90) \$(OUTPUT_OPTION) \$<</code>
<code>.f90~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.f90</code> <code>\$(COMPILE.f90) \$(OUTPUT_OPTION) \$*.f90</code>
<code>.f90.a</code>	<code>\$(COMPILE.f90) -o \$\$ \$<</code> <code>\$(AR) \$(ARFLAGS) \$\$ \$%</code> <code>\$(RM) \$%</code>
<code>.f90~.a</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.f90</code> <code>\$(COMPILE.f90) -o \$\$ \$*.f90</code> <code>\$(AR) \$(ARFLAGS) \$\$ \$%</code> <code>\$(RM) \$%</code>
<code>.ftn</code>	<code>\$(LINK.ftn) -o \$\$ \$< \$(LDLIBS)</code>
<code>.ftn~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.ftn</code> <code>\$(LINK.ftn) -o \$\$ \$*.ftn \$(LDLIBS)</code>
<code>.ftn.o</code>	<code>\$(COMPILE.ftn) \$(OUTPUT_OPTION) \$<</code>
<code>.ftn~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.ftn</code>

Table of Standard Implicit (Suffix) Rules for FORTRAN 90 Files	
Implicit Rule Name	Command Line
	\$(COMPILE.ftn) \$(OUTPUT_OPTION) \$*.ftn
.ftn.a	\$(COMPILE.ftn) -o \$% \$< \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%
.ftn~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.ftn \$(COMPILE.ftn) -o \$% \$*.ftn \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%

Table of Standard Implicit (Suffix) Rules for lex Files	
Implicit Rule Name	Command Line
.l	\$(RM) \$*.c \$(LEX.l) \$< > \$*.c \$(LINK.c) -o \$@ \$*.c \$(LDLIBS) \$(RM) \$*.c
.l.c	\$(RM) \$@ \$(LEX.l) \$< > \$@
.l.ln	\$(RM) \$*.c \$(LEX.l) \$< > \$*.c \$(LINT.c) -o \$@ -i \$*.c \$(RM) \$*.c
.l.o	\$(RM) \$*.c

Table of Standard Implicit (Suffix) Rules for lex Files	
Implicit Rule Name	Command Line
	<code>\$(LEX.l) \$< > \$*.c</code> <code>\$(COMPILE.c) -o \$@ \$*.c</code> <code>\$(RM) \$*.c</code>
<code>.l~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.l</code> <code>\$(LEX) \$(LFLAGS) \$*.l</code> <code>\$(CC) \$(CFLAGS) -c lex.yy.c</code> <code>rm -f lex.yy.c</code> <code>mv lex.yy.c \$@</code>
<code>.l~.c</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.l</code> <code>\$(LEX) \$(LFLAGS) \$*.l</code> <code>mv lex.yy.c \$@</code>
<code>.l~.ln</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.l</code> <code>\$(RM) \$*.c</code> <code>\$(LEX.l) \$*.l > \$*.c</code> <code>\$(LINT.c) -o \$@ -i \$*.c</code> <code>\$(RM) \$*.c</code>
<code>.l~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.l</code> <code>\$(LEX) \$(LFLAGS) \$*.l</code> <code>\$(CC) \$(CFLAGS) -c lex.yy.c</code> <code>rm -f lex.yy.c</code> <code>mv lex.yy.c \$@</code>

Table of Standard Implicit (Suffix) Rules for Modula 2 Files	
Implicit Rule Name	Command Line
.mod	\$(COMPILE.mod) -o \$@ -e \$@ \$<
.mod.o	\$(COMPILE.mod) -o \$@ \$<
.def.sym	\$(COMPILE.def) -o \$@ \$<
.def~.sym	\$(GET) \$(GFLAGS) -p \$< > \$*.def \$(COMPILE.def) -o\$@ \$*.def
.mod~	\$(GET) \$(GFLAGS) -p \$< > \$*.mod \$(COMPILE.mod) -o \$@ -e \$@ \$*.mod
.mod~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.mod \$(COMPILE.mod) -o \$@ \$*.mod
.mod~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.mod \$(COMPILE.mod) -o \$\$ \$*.mod \$(AR) \$(ARFLAGS) \$@ \$% \$(RM) \$%

Table of Standard Implicit (Suffix) Rules for NeWS Files	
Implicit Rule Name	Command Line
.cps.h	cps \$*.cps
.cps~.h	\$(GET) \$(GFLAGS) -p \$< > \$*.cps \$(CPS) \$(CPSFLAGS) \$*.cps

<i>Table of Standard Implicit (Suffix) Rules for Pascal Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.p</code>	<code>\$(LINK.p) -o \$@ \$< \$(LDLIBS)</code>
<code>.p.o</code>	<code>\$(COMPILE.p) \$(OUTPUT_OPTION) \$<</code>
<code>.p~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.p</code> <code>\$(LINK.p) -o \$@ \$*.p \$(LDLIBS)</code>
<code>.p~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.p</code> <code>\$(COMPILE.p) \$(OUTPUT_OPTION) \$*.p</code>
<code>.p~.a</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.p</code> <code>\$(COMPILE.p) -o \$% \$*.p</code> <code>\$(AR) \$(ARFLAGS) \$@ \$%</code> <code>\$(RM) \$%</code>

<i>Table of Standard Implicit (Suffix) Rules for Ratfor Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.r</code>	<code>\$(LINK.r) -o \$@ \$< \$(LDLIBS)</code>
<code>.r.o</code>	<code>\$(COMPILE.r) \$(OUTPUT_OPTION) \$<</code>
<code>.r.a</code>	<code>\$(COMPILE.r) -o \$% \$<</code> <code>\$(AR) \$(ARFLAGS) \$@ \$%</code> <code>\$(RM) \$%</code>
<code>.r~</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.r</code> <code>\$(LINK.r) -o \$@ \$*.r \$(LDLIBS)</code>

Table of Standard Implicit (Suffix) Rules for Ratfor Files	
Implicit Rule Name	Command Line
.r~.o	\$(GET) \$(GFLAGS) -p \$< > \$*.r \$(COMPILE.r) \$(OUTPUT_OPTION) \$*.r
.r~.a	\$(GET) \$(GFLAGS) -p \$< > \$*.r \$(COMPILE.r) -o %% \$*.r \$(AR) \$(ARFLAGS) \$@ %% \$(RM) %%

Table of Standard Implicit (Suffix) Rules for SCCS Files	
Implicit Rule Name	Command Line
.SCCS_GET	sccs \$(SCCSFLAGS) get \$(SCCSGETFLAGS) \$@ -G\$@
.SCCS_GET_POSIX	sccs \$(SCCSFLAGS) get \$(SCCSGETFLAGS) \$@
.GET_POSIX	\$(GET) \$(GFLAGS) s.\$@

Table of Standard Implicit (Suffix) Rules for Shell Scripts	
Implicit Rule Name	Command Line
.sh	cat \$< >\$@ chmod +x \$@
.sh~	\$(GET) \$(GFLAGS) -p \$< > \$*.sh cp \$*.sh \$@ chmod a+x \$@

<i>Table of Standard Implicit (Suffix) Rules for yacc Files</i>	
<i>Implicit Rule Name</i>	<i>Command Line</i>
<code>.y</code>	\$(YACC.y) \$< \$(LINK.c) -o \$@ y.tab.c \$(LDLIBS) \$(RM) y.tab.c
<code>.y.c</code>	\$(YACC.y) \$< mv y.tab.c \$@
<code>.y.ln</code>	\$(YACC.y) \$< \$(LINT.c) -o \$@ -i y.tab.c \$(RM) y.tab.c
<code>.y.o</code>	\$(YACC.y) \$< \$(COMPILE.c) -o \$@ y.tab.c \$(RM) y.tab.c
<code>.y~</code>	\$(GET) \$(GFLAGS) -p \$< > \$*.y \$(YACC) \$(YFLAGS) \$*.y \$(COMPILE.c) -o \$@ y.tab.c \$(RM) y.tab.c
<code>.y~.c</code>	\$(GET) \$(GFLAGS) -p \$< > \$*.y \$(YACC) \$(YFLAGS) \$*.y mv y.tab.c \$@
<code>.y~.ln</code>	\$(GET) \$(GFLAGS) -p \$< > \$*.y \$(YACC.y) \$*.y \$(LINT.c) -o \$@ -i y.tab.c

Table of Standard Implicit (Suffix) Rules for yacc Files	
Implicit Rule Name	Command Line
	<code>\$(RM) y.tab.c</code>
<code>.y~.o</code>	<code>\$(GET) \$(GFLAGS) -p \$< > \$*.y</code> <code>\$(YACC) \$(YFLAGS) \$*.y</code> <code>\$(CC) \$(CFLAGS) -c y.tab.c</code> <code>rm -f y.tab.c</code> <code>mv y.tab.o \$@</code>

make reads in the standard set of implicit rules from the file `/usr/share/lib/make/make.rules`, unless `-r` is in effect, or there is a `make.rules` file in the local directory that does not include that file.

The Suffixes List

The suffixes list is given as the list of dependencies for the `.SUFFIXES:` special-function target. The default list is contained in the `SUFFIXES` macro (See *Table of Predefined Macros* for the standard list of suffixes). You can define additional `.SUFFIXES:` targets; a `.SUFFIXES` target with no dependencies clears the list of suffixes. Order is significant within the list; make selects a rule that corresponds to the target's suffix and the first dependency-file suffix found in the list. To place suffixes at the head of the list, clear the list and replace it with the new suffixes, followed by the default list:

```
.SUFFIXES:
.SUFFIXES: suffixes $(SUFFIXES)
```

A tilde (~) indicates that if a dependency file with the indicated suffix (minus the ~) is under SCCS its most recent version should be retrieved, if necessary, before the target is processed.

Library Maintenance

A target name of the form:

```
lib(member ...)
```

refers to a member, or a space-separated list of members, in an `ar(1)` library.

The dependency of the library member on the corresponding file must be given as an explicit entry in the makefile. This can be handled by a pattern matching rule of the form:

```
lib(%s): %s
```

where `.s` is the suffix of the member; this suffix is typically `.o` for object libraries.

A target name of the form:

```
lib( (symbol) )
```

refers to the member of a randomized object library that defines the entry point named *symbol*.

Command Execution Command lines are executed one at a time, *each by its own process or shell*. Shell commands, notably `cd`, are ineffectual across an unescaped `NEWLINE` in the makefile. A line is printed (after macro expansion) just before being executed. This is suppressed if it starts with a `@`, if there is a `.SILENT` entry in the makefile, or if `make` is run with the `-s` option. Although the `-n` option specifies printing without execution, lines containing the macro `$(MAKE)` are executed regardless, and lines containing the `@` special character are printed. The `-t` (touch) option updates the modification date of a file without executing any rules. This can be dangerous when sources are maintained by more than one person.

`make` invokes the shell with the `-e` (exit-on-errors) argument. Thus, with semicolon-separated command sequences, execution of the later commands depends on the success of the former. This behavior can be overridden by starting the command line with a `-`, or by writing a shell script that returns a non-zero status only as it finds appropriate.

Bourne Shell Constructs To use the Bourne shell `if` control structure for branching, use a command line of the form:

```
if expression ; \
then command ; \
    ... ; \
else command; \
    ... ; \
fi
```

Although composed of several input lines, the escaped `NEWLINE` characters insure that `make` treats them all as one (shell) command line.

To use the Bourne shell `for` control structure for loops, use a command line of the form:

```
for var in list ; \
do command; \
    ... ; \done
```

To refer to a shell variable, use a double-dollar-sign (`$$`). This prevents expansion of the dollar-sign by `make`.

Command Substitutions To incorporate the standard output of a shell command in a macro, use a definition of the form:

```
MACRO :sh =command
```

The command is executed only once, standard error output is discarded, and `NEWLINE` characters are replaced with `SPACES`. If the command has a non-zero exit status, `make` halts with an error.

To capture the output of a shell command in a macro reference, use a reference of the form:

```
$(MACRO :sh)
```

where *MACRO* is the name of a macro containing a valid Bourne shell command line. In this case, the command is executed whenever the reference is evaluated. As with shell command substitutions, the reference is replaced with the standard output of the command. If the command has a non-zero exit status, make halts with an error.

In contrast to commands in rules, the command is not subject for macro substitution; therefore, a dollar sign (\$) need not be replaced with a double dollar sign (\$\$).

Signals

INT, SIGTERM, and QUIT signals received from the keyboard halt make and remove the target file being processed unless that target is in the dependency list for .PRECIOUS:.

Compatibility with GNU make

The compatibility mode with GNU make changes Oracle Solaris make's behavior with respect to the dynamic macro \$<. By default the Oracle Solaris make treats this macro as the name of a dependency file, derived as if selected for use with an implicit rule.

GNU make treats this macro as the name of a dependency, even if it is not a file. If the -x SUN_MAKE_COMPAT_MODE=GNU option is passed to the Oracle Solaris make, it behaves as GNU make in this particular case.

使用例

例1 Defining dependencies

This makefile says that pgm depends on two files a.o and b.o, and that they in turn depend on their corresponding source files (a.c and b.c) along with a common file incl.h:

```
pgm: a.o b.o
    $(LINK.c) -o $@a.o b.o
a.o: incl.h a.c
    cc -c a.c
b.o: incl.h b.c
    cc -c b.c
```

例2 Using implicit rules

The following makefile uses implicit rules to express the same dependencies:

```
pgm: a.o b.o
    cc a.o b.o -o pgm
a.o b.o: incl.h
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of make: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

KEEP_STATE This environment variable has the same effect as the .KEEP_STATE: special-function target. It enables command dependencies, hidden dependencies and writing of the state file.

USE_SVR4_MAKE This environment variable causes make to invoke the generic System V version of make (/usr/lib/avr4.make). See [sysV-make\(1\)](#).

MAKEFLAGS

This variable is interpreted as a character string representing a series of option characters to be used as the default options. The implementation accepts both of the following formats (but need not accept them when intermixed):

1. The characters are option letters without the leading hyphens or blank character separation used on a command line.
2. The characters are formatted in a manner similar to a portion of the `make` command line: options are preceded by hyphens and blank-character-separated. The `macro=name` macro definition operands can also be included. The difference between the contents of `MAKEFLAGS` and the command line is that the contents of the variable is not subjected to the word expansions associated with parsing the command line values. See [wordexp\(3C\)](#).

When the command-line options `-f` or `-p` are used, they take effect regardless of whether they also appear in `MAKEFLAGS`. If they otherwise appear in `MAKEFLAGS`, the result is undefined.

The `MAKEFLAGS` variable is accessed from the environment before the makefile is read. At that time, all of the options (except `-f` and `-p`) and command-line macros not already included in `MAKEFLAGS` are added to the `MAKEFLAGS` macro. The `MAKEFLAGS` macro is passed into the environment as an environment variable for all child processes. If the `MAKEFLAGS` macro is subsequently set by the makefile, it replaces the `MAKEFLAGS` variable currently found in the environment.

PROJECTDIR

Provides a directory to be used to search for SCCS files not found in the current directory. In all of the following cases, the search for SCCS files is made in the directory `SCCS` in the identified directory. If the value of `PROJECTDIR` begins with a slash, it shall be considered an absolute pathname. Otherwise, the value of `PROJECTDIR` is treated as a user name and that user's initial working directory shall be examined for a subdirectory `src` or `source`. If such a directory is found, it shall be used. Otherwise, the value is used as a relative pathname.

If `PROJECTDIR` is not set or has a null value, the search for SCCS files shall be made in the directory `SCCS` in the current directory. The setting of `PROJECTDIR` affects all files listed in the remainder of this utility description for files with a component named `SCCS`.

`SUN_MAKE_COMPAT_MODE` Causes make to change behavior according to the specified compatibility mode. Examples:

`SUN_MAKE_COMPAT_MODE="POSIX"`
Support POSIX makefiles and compatibility with
`/usr/xpg4/bin/make`

`SUN_MAKE_COMPAT_MODE="SUN"`
Support Sun makefiles and compatibility with Oracle Solaris
`/usr/bin/make`

`SUN_MAKE_COMPAT_MODE="GNU"`
Support GNU makefiles and GNU make behavior (partially supported)

`SUN_MAKE_COMPAT_MODE="SVR4"`
Support SVR4 makefiles and compatibility with
`/usr/lib/svr4.make`

終了ステータス When the -q option is specified, the make utility exits with one of the following values:

- 0 Successful completion.
- 1 The target was not up-to-date.
- >1 An error occurred.

When the -q option is not specified, the make utility exits with one of the following values:

- 0 Successful completion
- >0 An error occurred

ファイル

<code>makefile</code>	
<code>Makefile</code>	current version(s) of make description file
<code>s.makefile</code>	
<code>s.Makefile</code>	SCCS history files for the above makefile(s) in the current directory
<code>SCCS/s.makefile</code>	
<code>SCCS/s.Makefile</code>	SCCS history files for the above makefile(s)
<code>make.rules</code>	default file for user-defined targets, macros, and implicit rules
<code>/usr/share/lib/make/make.rules</code>	makefile for standard implicit rules and macros (not read if <code>make.rules</code> is)
<code>.make.state</code>	state file in the local directory

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/make

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

/usr/xpg4/bin/make

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/xopen/xcu4
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[ar\(1\)](#), [arch\(1\)](#), [cd\(1\)](#), [cpp\(1\)](#), [lex\(1\)](#), [mach\(1\)](#), [sccs-get\(1\)](#), [sh\(1\)](#), [sysV-make\(1\)](#), [yacc\(1\)](#), [wordexp\(3C\)](#), [passwd\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [POSIX.2\(5\)](#), [standards\(5\)](#)

『Solaris Advanced User's Guide』

診断

Don't know how to make target *target*

There is no makefile entry for *target*, and none of make's implicit rules apply (there is no dependency file with a suffix in the suffixes list, or the target's suffix is not in the list).

*** *target* removed.

make was interrupted while building *target*. Rather than leaving a partially-completed version that is newer than its dependencies, make removes the file named *target*.

*** *target* not removed.

make was interrupted while building *target* and *target* was not present in the directory.

*** *target* could not be removed, *reason*

make was interrupted while building *target*, which was not removed for the indicated reason.

Read of include file *file* failed

The makefile indicated in an include directive was not found, or was inaccessible.

Loop detected when expanding macro value *macro*'

A reference to the macro being defined was found in the definition.

Could not write state file *file*

You used the .KEEP_STATE: target, but do not have write permission on the state file.

***Error code *n*

The previous shell command returned a nonzero error code.

*** *signal message*

The previous shell command was aborted due to a signal. If – core dumped appears after the message, a core file was created.

Conditional macro conflict encountered

Displayed only when `-d` is in effect, this message indicates that two or more parallel targets currently being processed depend on a target which is built differently for each by virtue of conditional macros. Since the target cannot simultaneously satisfy both dependency relationships, it is conflicted.

使用上の留意点

Some commands return nonzero status inappropriately; to overcome this difficulty, prefix the offending command line in the rule with a `—`.

Filenames with the characters `=`, `:`, or `@`, do not work.

You cannot build `file.o` from `lib(file.o)`.

Options supplied by `MAKEFLAGS` should be reported for nested `make` commands. Use the `-d` option to find out what options the nested command picks up from `MAKEFLAGS`.

This version of `make` is incompatible in certain respects with previous versions:

- The `-d` option output is much briefer in this version. `—dd` now produces the equivalent voluminous output.
- `make` attempts to derive values for the dynamic macros `$*`, `$<`, and `$?`, while processing explicit targets. It uses the same method as for implicit rules; in some cases this can lead either to unexpected values, or to an empty value being assigned. (Actually, this was true for earlier versions as well, even though the documentation stated otherwise.)
- `make` no longer searches for SCCS history (`s.`) files.
- Suffix replacement in macro references are now applied after the macro is expanded.

There is no guarantee that makefiles created for this version of `make` works with earlier versions.

If there is no `make.rules` file in the current directory, and the file `/usr/share/lib/make/make.rules` is missing, `make` stops before processing any targets. To force `make` to run anyway, create an empty `make.rules` file in the current directory.

Once a dependency is made, `make` assumes the dependency file is present for the remainder of the run. If a rule subsequently removes that file and future targets depend on its existence, unexpected errors can result.

When hidden dependency checking is in effect, the `$?` macro's value includes the names of hidden dependencies. This can lead to improper filename arguments to commands when `$?` is used in a rule.

Pattern replacement macro references cannot be used in the dependency list of a pattern matching rule.

Unlike previous versions, this version of `make` strips a leading `./` from the value of the `$$` dynamic macro.

With automatic SCCS retrieval, this version of `make` does not support tilde suffix rules.

The only dynamic macro whose value is strictly determined when used in a dependency list is `$$` (takes the form `$$@`).

`make` invokes the shell with the `-e` argument. This cannot be inferred from the syntax of the rule alone.

名前 makekey – 暗号化鍵の生成

形式 /usr/lib/makekey

機能説明 makekey は、鍵空間の検索に必要な時間を増大させることで、その鍵に依存する暗号化方式の有効性を高めます。8 バイトが鍵 (入力の最初の 8 バイト) として読み取られ、次に 2 バイトがソルト (入力の最後の 2 バイト) として読み取られます。入力に応じて出力が変わるため、出力の計算が難しくなります (数分の 1 秒を要します)。

入力の最初の 8 バイト (入力鍵) は、任意の ASCII 文字列にできます。最後の 2 バイト (ソルト) は、数字、., /、大文字、および小文字を組み合わせて選択することをお勧めします。ソルト文字は、出力の最初の 2 文字として繰り返されます。出力の残りの 11 文字は、ソルトと同じセットから選択され、出力鍵を構成します。

変換の実行により、基本的に次のような動作が行われます。ソルトが使用され、4,096 個の暗号機のいずれかが選択されます。これらの暗号機はすべて米規格基準局の DES アルゴリズムに基づいていますが、その解読方法は 4,096 とおり存在します。入力鍵が鍵として使用され、定数文字列を暗号機に入力する処理が何度も繰り返されます。算出された 64 ビットが、結果として得られる出力鍵の 66 ビットに分配されます。

makekey は、暗号化を実行するプログラムの用に提供されています。通常、その入力と出力はパイプです。

関連項目 [ed\(1\)](#), [vi\(1\)](#), [passwd\(4\)](#)

注意事項 makekey で生成される結果は、入力が端末で実行されたか、ファイルからリダイレクトされたかによって異なります。

名前	man - マニュアルページの表示
形式	<pre>man [-] [-adFlrt] [-M path] [-T macro-package] [-s section] name... man [-M path] [-s section] -k query... man [-M path] -f file... man [-M path] [-s section] -K query...</pre>
機能説明	man コマンドは、リファレンスマニュアルから得た情報を表示します。 <i>name</i> で選択された完全なマニュアルページか、 <i>query</i> (-k または -K) と関連するファイルの名前 (-f) のどちらかで選択された 1 行から成るサマリー情報を表示します。マニュアルページが見つからない場合、man はエラーメッセージを出力します。
ソース形式	リファレンスマニュアルのページは、nroff (nroff(1)) を参照) または SGML (Standard Generalized Markup Language) タグ (sgml(5)) を参照) のどちらかのタグがマークアップされています。man コマンドはそのマークアップの種類を認識して、適宜にファイルを処理します。マークアップの種類によって分けられたディレクトリに、個々のソースファイルが置かれています。
マニュアルページの位置	オンラインのリファレンスマニュアルのページのディレクトリは、通常 /usr/share/man ディレクトリにあります。nroff のソースは、/usr/share/man/man* ディレクトリにあります。SGML のソースは、/usr/share/man/sman* ディレクトリにあります。各ディレクトリは、マニュアルのセクションごとに対応しています。これらのディレクトリは必須ではないため、ホスト上にインストールされているとは限りません。存在していなければ、ほかのホストから /usr/share/man をマウントしてください。 事前整形された最新バージョンが cat* または fmt* ディレクトリ中にあれば、man は単にそれを表示または印刷します。バージョンが古かったり存在しない場合には、man は再フォーマット処理を行ってから出力し、cat* または fmt* が書き込み可の場合に事前整形されたバージョンを格納します。インデックスファイルは更新されません。catman(1M) を参照してください。事前整形されたバージョン用のディレクトリが提供されていない場合、man は要求に応じてページを再フォーマットし、表示処理中、フォーマットされたテキストを一時ファイルに収納します。 標準出力が端末ではない場合、または -f フラグが指定された場合には、man は cat(1) を通じてテキストを出力します。それ以外の場合、man はページごとの出力や下線付けを画面上で行うために more(1) を通じて出力します。
クエリー文字列	-k または -K オプションを使用して、マニュアルページで、<i>query</i> (1 つ以上の語または語句) を検索できます。インデックスファイルベースの全文テキスト検索、ステミング、および部分一致がサポートされています。インデックスファイルの生成方法については、catman(1M) および man(5) を参照してください。 英語の場合、たとえば cat という語幹でステミングを実行すると、cats、catlike、catty などの文字列が検索されます。stem でステミングを実行

すると、`stemmer`、`stemming`、および `stemmed` が検索されます。ステミングのアルゴリズムによって、`fishing`、`fished`、`fish`、および `fisher` という単語が `fish` という語幹に変換されます。

マッチングでは、大文字と小文字が区別されます。ステミングは、英語のマニュアルページでのみ使用できます。

一致したマニュアルページは、クエリーの一致のスコアを基準として昇順にソートされて表示されます。

Oracle Solaris のマニュアルページは、名前、形式、機能説明などのセクションに分かれています。ユーザーは、`-k` オプションで詳しく説明しているように、検索範囲を1つのセクションに限定できます。

オプション

サポートしているオプションは、次のとおりです。

`-a`

MANPATH 検索パス中で *name* に一致したすべてのマニュアルページを出力します。複数個あるときは、見つかった順序で出力します。

`-d`

デバッグモード。セクションを示す引数に対応する値、検索に用いた方法、および `man` が検索を行なったパスを表示します。

`-f file ...`

`man` は、指定されたいずれかの *file* に関連するマニュアルページを見つけようとします。各 *file* から先頭のパス名部分を取り除き、結果となるベース名または名前を含む1行のサマリーを出力します。

このオプションは、インデックスファイルを使用します。インデックスファイルの生成方法については、[catman\(1M\)](#) および [man\(5\)](#) を参照してください。

`-F`

インデックス照合ファイルを使う代わりに、MANPATH で指定した全ディレクトリまたは `man.cf` ファイルを使って検索を行うよう `man` に指示します。このオプションは、インデックスファイルの内容が最新状態にない場合に便利な指定で、`man` コマンドのデフォルトの動作です。そのため、通常はこのオプションを起動する必要はありません。参考のためにここに記述しています。

`-k query...`

インデックスファイルから1行のサマリーを出力します。

インデックスファイルの生成方法については、`-k` オプションを参照してください。インデックスファイルがない場合、マニュアルページファイルが直接照合されるため、インデックスファイルが存在する場合に比べて応答時間が遅くなります。

-K query...

指定したクエリーをインデックスファイルから検索します。インデックスファイルがない場合、マニュアルページに対して直接検索が実行されるため、検索速度が非常に遅くなります。

クエリーオプション引数で左から1番目のテキストとしてセクション名を追加し、その最後にコロンの(:)を付けて「セクション名:クエリー」のようになると、そのクエリー文字列の検索は指定したセクションに対してのみ実行されます。指定したセクション名が存在しない場合は、ユーザーでサポートされているセクション名がすべてリストされます。

-f、-k、および-Kによって使用される /usr/share/man および /usr/gnu/share/man 内のインデックスファイルは、これらのディレクトリ内のマニュアルページがインストールまたは更新され、これらを配布するパッケージが、[pkg\(5\)](#) で指定されるようにファイルに `restart_fmri=svc:/application/man-index:default` のタグを付けたときに自動的に生成されます。これらは `svcadm restart application/man-index` を手動で実行するか、[catman\(1M\)](#) に -w を付けて実行しても生成できます。

-l

検索パス内にある、*name* に一致するマニュアルページをすべて一覧表示します。

-M path

マニュアルページ用に別の検索パスを指定します。*path* には、マニュアルページのディレクトリサブツリーを含んだディレクトリの名前をコロンの区切りで記述します。たとえば *path* の値として /usr/share/man:/usr/local/man と指定すると、man は最初に標準のディレクトリを検索し、次に /usr/local/man を検索して *name* を探します。-f、-k または -K オプションとともに使用する場合は、-M オプションを最初に記述する必要があります。*path* 中の各ディレクトリ中には、各セクションにつき1つの *man** または *sman** という名のサブディレクトリが存在していると見なされます。このオプションは、MANPATH 環境変数より優先されます。

-r

マニュアルページを再フォーマットするだけで、結果の表示は行いません。この指定は、`man - -t name` と同等です。

-s section...

man で検索するマニュアルのセクションを指定します。*section* 引数に対応するディレクトリだけを検索して *name* を探します。*section* の値は1桁の数字で、検索対象のセクション名を伴う場合もあります(たとえば、"3lib")。または、*section* を1つの単語(たとえば、local、new、old、public)にすることもできます。また、*section* は1文字の場合もあります。複数のセクションを指定するには、各セクションをコンマで区切ります。このオプションは、MANPATH 環境変数や man.cf ファイルより優先されます。man で検索を実行する方法の説明については、後続の「検索パス」を参照してください。

-t

man は、指定されたマニュアルページが、適切なラスター出力デバイス用に troff されるようにします (**troff(1)** を参照)。- と -t の両オプションがともに指定された場合、man は指定された各 *name* の troff バージョンを (必要であれば) 更新しますが、結果の出力は行いません。

-T *macro-package*

マニュアルページをフォーマットする際、/usr/share/lib/tmac/an に定義されている標準の -man マクロの代わりに、*macro-package* が示すパッケージを使用します。デフォルトの検索パスの順序に関する詳しい説明については、後述の「使用法」の「検索パス」を参照してください。

オペランド

次のオペランドを指定できます。

name

標準ユーティリティまたはキーワードの名前。

使用法

man の使用方法是次のとおりです。

マニュアルページ
のセクション

リファレンスマニュアルのエントリは、セクションに編成されています。セクションの名前は、通常 1 桁の主セクション名と、そのあとに必要に応じて続く通常 1 つ以上の文字であるサブセクション名から成ります。主セクション名 (例: "9") を、そのセクションのサブセクション名 (例: "9e", "9f", "9s" など) の省略名として使用することはできません。サブセクションを検索する場合は、man -s で個別に指定する必要があります。各セクションに属するマニュアルページ群は、共通のカテゴリに属しており、その中でさらに細かく分類するためにサブセクションがあります。このリリースで使用されている分類の説明については、intro のマニュアルページを参照してください。

次に、マニュアルページの各セクションと、そこに含まれている情報について簡単に説明します。

- セクション 1 では、このオペレーティングシステムで使用できるコマンドをアルファベット順に説明しています。
- セクション 1M では、主にシステムの保守や管理に使用するコマンドを、アルファベット順に説明しています。
- セクション 2 では、すべてのシステムコールについて説明しています。これらのコールのほとんどに、1 つ以上のエラーの戻り値があります。エラー状態は、ほかの場合には返されない戻り値によって示されます。
- セクション 3 では、さまざまなライブラリにある関数のうち、セクション 2 で説明されている、UNIX システムのプリミティブを直接呼び出す関数以外について説明しています。
- セクション 4 では、さまざまなファイルの形式の概要について説明しています。C 言語の構造体によるファイル形式の宣言も適宜示します。
- セクション 5 には、文字セット表などのその他のドキュメントが含まれています。

- セクション7では、特定のハードウェア周辺装置やデバイスドライバを参照するさまざまな特殊ファイルについて説明しています。また、STREAMS ソフトウェアのドライバやモジュール、STREAMS で一般的な一連のシステムコールについても説明しています。
- セクション9E では、開発者がデバイスドライバに含めることのできる DDI (デバイスドライバインタフェース)/DKI (ドライバ/カーネルインタフェース)、DDI のみ、および DKI のみのエントリポイントルーチンについて説明しています。
- このセクションでは、デバイスドライバで使用可能なカーネル関数について説明します。
- セクション9S では、ドライバとカーネルの間で情報を共有するためにドライバによって使用されるデータ構造について説明しています。

検索パス

指定されたある *name* を検索する前に、man はまず候補となりうるディレクトリとセクションのリストを構築します。man は環境変数 MANPATH が示すディレクトリ群の中で *name* を探します。

MANPATH が存在しない場合、man は最初に PATH 要素の最後のコンポーネントに man を代入することによって、PATH 環境変数に基づいた検索パスを作成します。/sbin、/usr/ucb、/usr/xpg4/bin などのディレクトリの独自の特性に対応するために、特別なプロビジョニングが追加されています。ファイル引数に / 文字が含まれる場合は、検索パスを作成するために PATH 要素の代わりに引数の *dirname* 部分が使用されます。

マニュアルページのディレクトリ内において、man は次に示すセクションだけを次の順序で検索します。

- コマンド行上で -s オプションにより指定されたセクション群
- MANPATH 環境変数に埋め込まれているセクション群
- MANPATH 環境変数が示す各ディレクトリの man.cf ファイル中に指定されているセクション群

前述の指定がいずれも存在しない場合には、man はマニュアルページパス中の各ディレクトリを検索し、最初に見つかったマニュアルページだけを表示します。

man.cf ファイルの形式は次のとおりです。

```
MANSECTS=section[,section]...
```

文字 '#' で始まる行および空行は注釈と見なされ、無視されます。MANPATH が示す各ディレクトリには、そのディレクトリのデフォルトの検索順序を指定するマニュアルページ設定ファイルを含めることができます。

マニュアルページのフォーマット

マニュアルページは `nroff(1)` または `sgml(5)` でマークアップされています。nroff のマニュアルページは、-man マクロパッケージにより用意されている `nroff(1)` または

[troff\(1\)](#)によって処理されます。マクロの使用法については、[man\(5\)](#)を参照してください。SGMLタグがついたマニュアルページは、SGMLパーサーによって処理され、フォーマッタに引き渡されます。

nroff マニュアル
ページの前処理

nroff マニュアルページをフォーマットする際、man は、先頭行を検査して特殊な処理が必要かどうかを判断します。先頭行が

```
'\" X
```

という形式の文字列 (X と $"$ の間は1つのスペース文字) であり、次に示す文字の任意の組み合わせで構成されている場合、man は入力テキストを対応するプリプロセッサを介して [troff\(1\)](#) または [nroff\(1\)](#) に送ります。

```
e
```

[eqn\(1\)](#)、または nroff の場合は neqn

```
r
```

[refer\(1\)](#)

```
t
```

[tbl\(1\)](#)

```
v
```

[vgrind\(1\)](#)

eqn または neqn は、呼び出されると自動的にファイル `/usr/pub/eqnchar` を読み取ります ([eqnchar\(5\)](#) を参照)。[nroff\(1\)](#) が呼び出されると、[col\(1\)](#) が自動的に使用されます。

ほかの nroff マ
ニュアルページへ
の参照

nroff のマニュアルページの先頭行が、次のパターンで別のマニュアルページを参照している場合、

```
.so man*/sourcefile
```

man は、現在のファイルの代わりにその参照されているファイルを処理します。この参照は、マニュアルページのディレクトリサブツリーのルートに対して相対的なパス名として表現されている必要があります。

2 行目以降の行が `.so` で開始されている場合、man はそれを無視し、[troff\(1\)](#) または [nroff\(1\)](#) は通常の方法で要求を処理します。

SGML マニュアル
ページの処理

マニュアルページは、ファイル中に `<!DOCTYPE` という文字列が存在することによって、SGML でマークアップされていると特定されます。ファイルに `SHADOW_PAGE` という文字列も含まれている場合、そのファイルは別のマニュアルページの内容を参照します。この参照は、テキストが含まれているマニュアルページへのファイル実体の参照によって行われます。これは、nroff フォーマットのマニュアルページで使用する `.so` のメカニズムに類似しています。

環境変数

`man` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLS_PATH。

MANPATH

コロンで区切られたディレクトリのリスト。各ディレクトリは、コンマで区切られたセクションのリストのあとに続いて指定できます。この環境変数に値を設定すると、デフォルトのディレクトリ検索パスである `/usr/share/man` と、デフォルトのセクション検索パスである `man.cf` を無効にします。また、`-M` フラグはデフォルトのディレクトリ検索パスを、`-s` フラグはデフォルトのセクション検索パスを、それぞれ無効にします。

PAGER

`man` の出力を対話モードで画面に送るために使用するプログラム。この環境変数が設定されていない場合は、`'more -s'` が使用されます。[more\(1\)](#) を参照してください。

TCAT

`troff` で処理されたマニュアルページを表示するために使用するプログラムの名前。

TROFF

`-t` オプションが指定された場合に用いるフォーマッタの名前。この環境変数が設定されていない場合は、[troff\(1\)](#) が使用されます。

使用例

例1 テキストバージョンのマニュアルページの作成

次の例では、[pipe\(2\)](#) のマニュアルページを ASCII テキストで作成します。

```
% man pipe.2 | col -x -b > pipe.text
```

これは、`man -t` を使用してマニュアルページをデフォルトのプリンタに送信する代わりに、テキストファイルバージョンのマニュアルページを作成します。

例2 1つ以上の用語に一致するマニュアルページのリストの取得

次の例では、`zfs` または `create` という語に一致するマニュアルページのリストを取得します:

```
% man -K zfs create
```

例3 1つ以上の語句に一致するマニュアルページのリストの取得

次の例では、引用符で囲まれた語句 `"zfs create"` または `"storage pool"` に一致するマニュアルページのリストを取得します。

```
% man -K 'zfs create' "storage pool"
```

例4 セクション内で語または語句に一致するマニュアルページのリストの取得

次の例では、関連項目セクションで `zfs` という語が含まれるマニュアルページのリストを取得します:

例4 セクション内で語または語句に一致するマニュアルページのリストの取得 (続き)

```
% man -K see also: zfs
```

次の例では、使用例セクションで“zfs create”という語句が含まれるマニュアルページのリストを取得します:

```
% man -K examples: "zfs create"
```

終了ステータス 次の終了ステータスが返されます。

```
0
  正常終了。
```

```
>0
  エラーが発生した。
```

ファイル

```
/usr/share/man
  標準マニュアルページのディレクトリサブツリーのルート
```

```
/usr/share/man/man?/*
  書式なし nroff のマニュアルのエントリ
```

```
/usr/share/man/man_index/*
  目次とキーワードのデータベース。
```

次のファイルが生成されます。

- /usr/share/man/man-index/man.idx
- /usr/share/man/man-index/man.dic
- /usr/share/man/man-index/man.req
- /usr/share/man/man-index/man.pos
- /usr/share/man/man-index/man.doc

```
/usr/share/man/sman?/*
  書式なし SGML のマニュアルのエントリ
```

```
/usr/share/man/cat?/*
  nroff 処理後のマニュアルのエントリ
```

```
/usr/share/man/fmt?/*
  troff 処理後のマニュアルのエントリ
```

```
/usr/share/lib/tmac/an
  標準-man マクロパッケージ
```

```
/usr/share/lib/sgml/locale/C/dtd/*
  SGML ドキュメント型定義ファイル
```

```
/usr/share/lib/sgml/locale/C/solbook/*
  SGML のスタイルシートと実体ファイルの定義ディレクトリ
```

/usr/share/lib/pub/eqnchar
eqn と neqn の標準定義

man.cf

セクションごとのデフォルトの検索順序

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools
CSI	有効。「注意事項」を参照。
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目

[apropos\(1\)](#), [cat\(1\)](#), [col\(1\)](#), [eqn\(1\)](#), [more\(1\)](#), [nroff\(1\)](#), [refer\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [vgrind\(1\)](#), [whatis\(1\)](#), [catman\(1M\)](#), [attributes\(5\)](#), [environ\(5\)](#), [eqnchar\(5\)](#), [man\(5\)](#), [sgml\(5\)](#), [standards\(5\)](#)

注意事項

-f、-k、および -K オプションは、[man\(5\)](#) で指定される SMF サービスによって自動的に生成されるか、または [catman\(1M\)](#) で -w オプションを指定して手動で生成するインデックスファイルを使用します。

windex データベースファイルは使用されなくなりました。windex データベースファイルは新しいインデックスファイルに置き換えられました。

man コマンドは CSI に対応しています。ただし、man コマンドによって呼び出される一部のユーティリティー (troff、eqn、neqn、refer、tbl、および vgrind) は、CSI 対応が検証されていません。このため、-t オプションを指定した man コマンドは、EUC 以外のデータを処理できません。また、eqn、neqn、refer、tbl、または vgrind を通した特殊な処理が必要なマニュアルページを表示するための man コマンドの使用を CSI 対応にすることもできません。

使用上の留意点

マニュアルは、写真植字機または ASCII 端末のどちらかで再現可能であると想定されています。ただし、端末上では (たとえば、フォントの変更で示される) 一部の情報が失われます。

一部のダム端末は、e ([eqn\(1\)](#) を参照) 前処理フラグによって生成される縦方向の動きを処理できません。これらの端末での歪んだ出力を防ぐために、e フラグを使用するときには t フラグも使用して、暗黙的に [col\(1\)](#) を呼び出してください。この回避方法には、上付きおよび下付きの添字が、それらを表示できる端末でも削除されるという欠点があります。[eqn\(1\)](#) の出力によって正しく動作しなくなった端末は、Control-q でクリアーしてください。

名前	mconnect – SMTP メールサーバーソケットへの接続
形式	mconnect [-p <i>port</i>] [-r] [<i>hostname</i>]
機能説明	mconnect ユーティリティーは、特定のホスト上のメールサーバーに接続し、ほかのすべてのメールソフトウェアとは別個にそのメールサーバーをテストできるようにします。ホストを指定しなかった場合は、ローカルホストに接続します。この接続では、サーバーが Simple Mail Transfer Protocol (SMTP) 通信を実行します。終了するには、quit コマンドを入力します。EOF を入力すると、ファイルの終わりがサーバーに送信されます。割り込みが発生すると、ただちに接続が閉じられて終了します。
オプション	サポートしているオプションは、次のとおりです。 -p <i>port</i> デフォルトの SMTP ポート (25 番) に代わるポート番号を次の引数として指定します。 -r Raw モード: デフォルトのラインバッファリングと入力処理を無効にします。これは、ポート番号 25 に対して telnet(1) と同じような効果を与えます。
オペランド	次のオペランドを指定できます。 <i>hostname</i> 特定のホストの名前。
使用法	mconnect コマンドは IPv6 に対応しています。 ip6(7P) を参照してください。
ファイル	/etc/mail/sendmail.hf SMTP コマンドのヘルプファイル
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	service/network/smtp/sendmail

関連項目	telnet(1) , sendmail(1M) , attributes(5) , ip6(7P)
	Postel, Jonathan B., RFC 821, Simple Mail Transfer Protocol, Information Sciences Institute, University of Southern California, August 1982.

名前	mcs – manipulate the comment section of an object file
形式	mcs [-cdHpVz] [-A <i>file</i>] [-a <i>string</i>] [-n <i>name</i>] <i>file</i> ...
機能説明	The <code>mcs</code> command is used to display, compress, or append content to comment sections in an ELF object file. It can also be used to delete comment or non-comment sections, and to zero <code>SHT_PROGBITS</code> sections. Unless otherwise specified, the section named <code>.comment</code> is manipulated. <code>mcs</code> cannot modify or delete a section that is contained within a segment. If the input file is an archive (see ar.h(3HEAD)), the archive is treated as a set of individual files. For example, if the <code>-a</code> option is specified, the string is appended to the comment section of each object file in the archive; if the archive member is not an object file, then it is left unchanged. <code>mcs</code> must be given one or more of the options described below. It applies each option, in the order given, to each file. For append operations, if the object does not already contain a section with the specified name, <code>mcs</code> will create a new empty section with that name before performing the append operation.
オプション	The following options are supported: -A <i>file</i> Appends the contents of the given file to the comment section of the object files. -a <i>string</i> Appends <i>string</i> to the comment section of the object files. -c Compresses the contents of the comment section of the object files. All duplicate entries are removed. The ordering of the remaining entries is not disturbed. -d Deletes the specified section from the object files. -H When -p is used, suppress the output of the name of the file, as well as any blank lines normally inserted between the output for each file. -n <i>name</i> Specifies the name of the section to access if other than <code>.comment</code>. By default, <code>mcs</code> deals with the section named <code>.comment</code>. <code>mcs</code> can take multiple <code>-n</code> options to allow for specification of multiple sections. <code>mcs</code> always manipulations sections assuming that they have the format of a comment section. -p Prints the contents of the comment section on the standard output. Unless used with -H, each section printed is prefixed with the name of the file from which it was extracted, using the format <i>file[member_name]</i>: for archive files and <i>file</i>: for other files. -V Prints a message on standard error giving information about the version of <code>mcs</code>. -z Replaces any <code>SHT_PROGBITS</code> sections with zeros while retaining the original attributes of the sections.

使用例

例 1 Printing a file's comment section

The following entry

```
example% mcs -p elf.file
```

prints the comment section of the file `elf.file`.

例 2 Appending a string to a comment section

The following entry

```
example% mcs -a xyz elf.file
```

appends string `xyz` to `elf.file`'s comment section.

例 3 Stripping a specified non-allocable section

Although used primarily with comment sections, `mcs` can operate on any non-allocable section. In contrast to the `strip` command, which removes a predefined selection of non-allocable sections, `mcs` can be used to delete a specific section. The following entry

```
example% mcs -d -n .annotate elf.file
```

removes the section named `.annotate` from the file `elf.file`.

ファイル

`/tmp/mcs*` temporary files

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities
Interface Stability	Committed

関連項目

[ar\(1\)](#), [as\(1\)](#), [ld\(1\)](#), [strip\(1\)](#), [ar.h\(3HEAD\)](#), [elf\(3ELF\)](#), [a.out\(4\)](#), [attributes\(5\)](#)

注意事項

When `mcs` deletes a section using the `-d` option, it tries to bind together sections of type `SHT_REL` and target sections pointed to by the `sh_info` section header field. If one is to be deleted, `mcs` attempts to delete the other of the pair.

The `-z` option removes the contents of `SHT_PROGBITS` sections while retaining the original ELF structure of the object. The need for use of the `-z` option is limited. However, the option can be used to deliver an object file when the contents of `SHT_PROGBITS` sections are not relevant.

名前 mdb - モジュラーデバッガ

形式 mdb [-fkmwyAFKMSUW] [±o option] [-p pid] [-s distance]
 [-I path] [-L path] [-P prompt] [-R root]
 [-V dis-version] [object [core] | core | suffix]

機能説明

はじめに mdb ユーティリティーは、稼働中のオペレーティングシステム、オペレーティングシステムのクラッシュダンプ、ユーザープロセス、ユーザープロセスのコアダンプ、およびオブジェクトファイルの低レベルのデバッグと編集を行うための拡張可能なユーティリティーです。mdb の機能の詳細については、マニュアル『[Oracle Solaris Modular Debugger Guide](#)』を参照してください。

デバッグングとは、欠陥を取り除くために、ソフトウェアプログラムの実行と状態を分析するプロセスのことです。従来のデバッグ用のツールは、実行制御の機能を備えたもので、それによって、プログラマは制御された環境でプログラムを実行し直したり、プログラムデータの現在の状態を表示したり、プログラム開発に使用するソース言語の表現を評価したりできます。

残念ながら、これらの手法は、バグが再現しない、プログラムの状態が大規模で分散しているオペレーティングシステムなどの複雑なソフトウェアシステムのデバッグには適さないことがよくあります。また、高度に最適化され、デバッグ情報が削除されている、それ自体が低レベルのデバッグ用ツールであるプログラムや、開発者が事後情報にしかアクセスできない顧客環境にも適しません。

mdb では、前述のようなプログラムや状況をデバッグするために、徹底的にカスタマイズできます。mdb に含まれている動的なモジュール機能を使用して、プログラム固有の分析を行う場合に、プログラマ独自のデバッグングコマンドを実行できます。プログラムの実行時、事後分析時などさまざまな状況で、各 mdb モジュールをプログラム検査に使用できます。

定義 ターゲットとは、デバッガによって検査されるプログラムのことです。mdb は現在、次の種類のターゲットをサポートしています。ユーザープロセス、ユーザープロセスのコアファイル、稼働中のオペレーティングシステム (/dev/kmem および /dev/ksyms 経由)、オペレーティングシステムのクラッシュダンプ、オペレーティングシステムのクラッシュダンプ内に記録されたユーザープロセスイメージ、ELF オブジェクトファイル、および raw バイナリファイル。各ターゲットは、1 つ以上のアドレス空間、1 つ以上のシンボルテーブル、ロードオブジェクトセット、後述のデバッガコマンドを使って検査できるスレッドセットなど、プロパティの標準セットをエクスポートします。

デバッガコマンド (mdb 用語法では、*dcmd* と表記し、「ディーコマンド」と読む) は、デバッガルーチンで、現ターゲットのどのプロパティにもアクセスできます。mdb は、標準入力からコマンドを構文解析し、次に対応する *dcmd* を実行します。各 *dcmd* は、文字列や数値引数のリストも受け取ることができます(後述の構文の説明を参照)。あとで説明しますが、mdb には、常に使用可能な組み込み *dcmd*

セットが入っています。独自の dcmd を記述して、mdb そのものの機能を拡張することもできます (『[Oracle Solaris Modular Debugger Guide](#)』を参照)。

walker は、特定のプログラムデータ構造体の要素を調べたり、繰り返し調べたりする方法を記述するルーチンセットです。walker は、dcmd や mdb そのものからデータ構造体の実装状態をカプセル化します。walker は、対話処理でも使用でき、ほかの dcmd や walker を構築するためのプリミティブとしても使用できます。dcmd の場合と同様に、独自の walker をデバッグモジュールの一部として実装することにより、mdb を拡張できます。

デバッグモジュール (dmod と表記し、「ディーモッド」と読む) は、動的に読み込まれるライブラリで、一連の dcmd と walker が含まれています。初期設定の状態では、mdb は、ターゲット内に存在するロードオブジェクトに対応する dmod を読み込もうとします。その後、mdb を実行している間はいつでも、dmod の読み込みや読み込み解除ができます。mdb には、Solaris カーネルをデバッグするための標準 dmod セットが付属しています。独自のデバッグモジュールの開発に関する詳細は、『[Oracle Solaris Modular Debugger Guide](#)』を参照してください。

「マクロファイル」とは、実行するコマンドセットが入っているテキストファイルのことです。一般的に、マクロファイルは、単純データ構造体の表示プロセスを自動化するときに使用されます。mdb には、[adb\(1\)](#) 用に記述されたマクロファイルの実行に関して完全な下位互換性があり、Solaris インストールには、エディタツールで使える Solaris カーネルのデバッグ用マクロファイルセットが含まれています。

構文

デバッグは、標準入力からコマンドを処理します。端末からの標準入力の場合、mdb では端末編集機能が使用できます。また、mdb は、マクロファイルからのコマンドや dcmd パイプラインからのコマンドも処理できます。これについては後述します。言語構文は、ターゲット内のメモリアドレスに代表されるような、式の値を計算し、次に dcmd をそのアドレスに適用するという構想に基づいて設計されています。現在のアドレス位置はドットとして参照され、その値は「.」を使って参照されます。

メタキャラクタには、次のような文字があります。

```
[ ] | ! / \ ? = > $ : ;
      NEWLINE SPACE TAB
```

ブランク (blank) は、タブ (TAB) またはスペース文字 (SPACE) のことです。ワード (word) とは、1 つまたは複数の引用符なしのメタキャラクタで区切られた文字列のことです。ただし、コンテキストによっては単に区切り記号として機能するメタキャラクタもあるので、これについては後述します。識別子とは、文字、下線、またはピリオドで始まる、一連の文字列、数字、下線、ピリオド、逆引用符のことです。識別子は、シンボル名、変数、dcmd、walker として使用されます。コマンドは、復帰改行文字やセミコロン (;) で区切ります。

dcmd は、次のようなワードまたはメタキャラクタで表されます。

```
/ \ ? = > $character :character ::identifier
```

メタキャラクタで設定された dcmd や、\$ か : を接頭辞として1つ持つ dcmd は、組み込み演算子として提供され、レガシー [adb\(1\)](#) ユーティリティのコマンドセットとの互換性を備えています。dcmd が解析されると、/、\、?、=、>、\$、: の各文字は、引数リストが終了するまでメタキャラクタとして認識されなくなります。

単純コマンドとは、空白文字で区切られた、0 個以上の一連のワードが後ろに続く dcmd のことです。これらのワードは、呼び出される dcmd に引数として渡されます。ただし、後述の「クォート」と「算術展開」で特に指定されているワードは例外です。各 dcmd は、処理の成功、失敗、または無効な引数を受け取ったことを示す終了ステータスを返します。

パイプライン (pipeline) は、パイプ (|) で区切られた1つ以上の単純コマンドの並びです。シェルの場合とは異なり、mdb のパイプラインの dcmd は、別々のプロセスとしては実行されません。mdb では、パイプラインが構文解析されたあとに、それぞれの dcmd が左から右へと順に呼び出されます。各 dcmd の出力は、処理されたあとに格納されます(後述の「dcmd パイプライン」を参照)。左側の dcmd 処理が終了すると、その出力はパイプライン内の次の dcmd への入力として使用されます。いずれかの dcmd が終了ステータスとして正常終了を返さない場合、そのパイプラインは強制終了します。

式 (expression) は一連のワードで表され、64 ビットの符号なし整数を計算するために評価されます。ワードは、後述の「算術展開」に示す規則を使用して評価されます。

コマンド

コマンドは、次のいずれかです。

pipeline [! word...][;]

単純コマンドまたはパイプラインは、接尾辞として感嘆符 (!) を付けることもできます。この場合、デバッグは [pipe\(2\)](#) を開いたあと、mdb パイプライン内の最後の dcmd の標準出力を、\$SHELL -c の実行により作成された外部プロセスへ送ります。c オプションのあとには、感嘆符 (!) の後ろのワードを結合して生成される文字列が続きます。詳細については、後述の「シェルエスケープ」を参照してください。

expression pipeline [! word...][;]

単純コマンドまたはパイプラインは、接頭辞として式を付けることもできます。この場合、パイプラインが実行される前に、ドットの値 (「.」で表される変数) が式の値に設定されます。

expression , expression pipeline [! word...][;]

単純コマンドまたはパイプラインは、接頭辞として式を2つ付けることもできます。最初の式は新しいドット値を判定するために評価され、2 番目の式はパイプライン内の最初の dcmd の繰り返し回数を判定するために評価されます。この場合、dcmd は、判定された回数繰り返し実行し、そのあとにパイプライン内の次の dcmd を実行します。繰り返し回数は、パイプライン内の最初の dcmd にだけ適用されます。

, expression pipeline [! word...][;]

式の値に応じて、パイプライン内の最初の dcmd が繰り返されます。ただし、先頭の式が省略されている場合、ドットは変更されません。

expression [! word...][;]

コマンドは、算術式だけで構成される場合があります。この場合、式が評価されたあと、その値にドット変数が設定されます。次に、直前の dcmd と引数が新しいドット値を使用して実行されます。

expression, expression [! word...][;]

コマンドは、ドット式と繰り返し回数式だけで構成される場合があります。この場合、最初の式の値がドットに設定されたあと、2 番目の式で指定された回数、直前の dcmd と引数が繰り返し実行されます。

, expression [! word...][;]

直前の dcmd と引数が、繰り返し回数式の値で指定された回数、繰り返し実行されます。ただし、先頭の式が省略されている場合は、ドットは変更されません。

! word...[;]

コマンドが感嘆符 (!) で始まっている場合、どの dcmd も実行されません。つまり、デバッグは感嘆符 (!) の後ろに指定されたワードを結合して、その文字列を \$SHELL -c の後ろに指定して実行するだけです。

Comments

// を付けると、その後の復帰改行文字まで、ワードや文字列がすべて無視されます。

演算機能の拡張

mdb コマンドの前に、開始アドレスを表すオプション式や、開始アドレスと繰り返し回数を表すオプション式がある場合は、演算機能が拡張されます。また、dcmd に使用する数値引数を計算する場合にも、演算機能が拡張されます。演算式は、ドル記号の後ろに角括弧で囲んだ引数リスト (*\$[expression]*) で表され、その式の値に置き換えられます。

式には、次の特殊ワードのどれかを使用できます。

integer

特定の整数値。整数値は、接頭辞として *0i* または *0I* を付けると 2 進数値、*0o* または *0O* を付けると 8 進数値、*0t* または *0T* を付けると 10 進数値、*0x* または *0X* を付けると 16 進数値 (デフォルト) を表します。

0[tT][0-9]+.[0-9]+

指定された 10 進数の浮動小数点値。IEEE 倍精度浮動小数点表示に変換されます。

'ccccccc'

各文字を ASCII 値に等しいバイトに変換することによって計算された整数値。最大 8 文字までを文字定数に指定できます。文字列は、最下位のバイトから始まって、右から左へと逆方向に整数の中へ格納されます。

<identifier

識別子 (*identifier*) によって指定される変数値。

identifier

識別子 (*identifier*) によって指定されるシンボル値。

(*expression*)

式 (*expression*) の値。

.

ドット値。

&

dcmd を実行するために使用される最新のドット値。

+

現在のインクリメントによって増分されるドット値。

^

現在のインクリメントによって減分されるドット値。

インクリメントとは、最後にフォーマット dcmd によって読み込まれる合計バイトを格納する大域変数のことです。インクリメントの詳細については、後述の「フォーマット dcmd」を参照してください。

単項演算子は右結合で、2 項演算子よりも優先順位は高くなります。単項演算子を次に示します。

#expression

論理否定。

~expression

ビット単位の補数。

-expression

整数否定。

%expression

ターゲットの仮想アドレス空間内の仮想アドレス式に対応するオブジェクトファイル位置でのポインタサイズの数値。

%[csil]/expression

ターゲットの仮想アドレス空間内の仮想アドレス式に対応するオブジェクトファイル位置での char サイズ、short サイズ、int サイズ、または long サイズの数値。

%[1248]/expression

ターゲットの仮想アドレス空間内の仮想アドレス式に対応するオブジェクトファイル位置での 1 バイト、2 バイト、4 バイト、または 8 バイトの数値。

**expression*

ターゲットの仮想アドレス空間内の仮想アドレス式でのポインタサイズの数値。

**/[csil]/expression*

ターゲットの仮想アドレス空間内の仮想アドレス式での char サイズ、short サイズ、int サイズ、または long サイズの数量値。

**/[1248]/expression*

ターゲットの仮想アドレス空間内の仮想アドレス式での 1 バイト、2 バイト、4 バイト、または 8 バイトの数量値。

2 項演算子は左結合で、単項演算子よりも優先順位は低くなります。次に 2 項演算子を、優先度の高いものから順に示します。

*

整数の乗算。

%

整数の除算。

#

左辺を右辺の最小の倍数に切り上げる。

+

整数の加算。

-

整数の減算。

<<

ビット単位で左へシフト。

>>

ビット単位で右へシフト。

==

等しい。

!=

異なる。

&

ビット単位の論理積。

^

ビット単位の排他的論理和。

|

ビット単位の論理和。

クォート

「構文」で説明したように、各メタキャラクタは引用符で囲まないとワードを終了します。mdb を使用して各文字を特別な意味のない文字そのものとして解釈させるには、それらを単一引用符 (' ') または二重引用符 (" ") で囲めば、文字列として引

用できます。単一引用符を、単一引用符で囲んで表示させることはできません。二重引用符内では、mdb は C プログラミング言語の文字エスケープシーケンスを認識します。

シェルエスケープ ! 文字を使用して、mdb コマンドとユーザーのシェル間のパイプラインを作成できます。\$SHELL 環境変数が設定されている場合、mdb はそのプログラムをシェルエスケープのためにフォーク (fork) したり、実行 (exec) したりします。変数が設定されていない場合は、/bin/sh コマンドが使用されます。シェルは、-c オプション付きで呼び出されます。c オプションのあとには、感嘆符 (!) の後ろのワードを結合して生成される文字列が続きます。! 文字は、ほかのどのメタキャラクタよりも高い優先度を持っています。ただし、セミコロン (;) と復帰改行文字は例外です。シェルエスケープが検出されたら、次のセミコロンまたは復帰改行文字までの残りの文字列は、そのままシェルへ渡されます。シェルの出力コマンドを mdb dcmd へパイプすることはできません。シェルエスケープによって実行されたコマンドは、その出力を mdb へは送らずに、直接端末へ送ります。

変数 変数は、変数名、対応する整数値、および一連の属性を持ちます。変数名は、一連の文字列、数字、下線、ピリオドなどで表されます。変数には、>dcmd や ::typeset dcmd を使用して値を割り当てることができます。また、その属性は、::typeset dcmd を使用して変更できます。各変数の値は、64 ビットの符号なし整数として表されます。変数は、1 つまたは複数の属性を持つことができます。たとえば、読み取り専用 (ユーザーによって変更されない)、固定表示 (ユーザーによって設定解除されない)、タグ (ユーザー定義のインジケータ) などです。

次の変数の属性は固定表示として定義されています。

- 0 /、\、?、= の dcmd を使用して出力された最新の値。
- 9 \$<dcmd とともに使用された最新のカウンタ。
- b データセクションの基底仮想アドレス。
- d データセクションのバイトサイズ。
- e エントリポイントの仮想アドレス。
- m ターゲットのプライマリオブジェクトファイルの初期バイト (マジックナンバー)。オブジェクトファイルがまだ読み出されていない場合はゼロ。
- t テキストセクションのバイトサイズ。

hits

一致したソフトウェアイベント指定子が一致した回数。後述の「イベントコールバック」を参照してください。

thread

現在の代表スレッドのスレッド識別子。この識別子の値は、現在のターゲットが使用しているスレッド化モデルによって変わります。後述の「スレッドのサポート」を参照してください。

さらに、mdb カーネルとプロセスターゲットは、代表スレッドのレジスタセットの現在値を設定変数としてエクスポートします。これらの変数の名前は、ターゲットのプラットフォームや命令セットのアーキテクチャーによって決まります。

シンボルの名前解決

前述の「構文」で説明しているように、式コンテキスト内にあるシンボル識別子は該当するシンボルの値を求めるために評価します。一般的に、この値は、ターゲットの仮想アドレス空間内のシンボルと関連付けられる、記憶領域の仮想アドレスを表します。ターゲットは、プライマリ実行可能シンボルテーブル、プライマリ動的シンボルテーブル、実行時リンクエディタシンボルテーブル、およびロードオブジェクト(ユーザープロセスの共有ライブラリや Solaris カーネルのカーネルモジュールなど)の各番号の標準シンボルテーブルと動的シンボルテーブルを含む(ただし、これらに限定されません)複数のシンボルテーブルをサポートできます。一般的に、ターゲットは、最初にプライマリ実行可能シンボルテーブルを検索し、次にほかの1つまたは複数のシンボルテーブルを検索します。ELF シンボルテーブルには、外部シンボル、大域シンボル、静的シンボルなどへのエントリだけが含まれます。自動シンボルは、mdb によって処理されるシンボルテーブルにはありません。

さらに、mdb が提供する専用のユーザー定義シンボルテーブルは、ほかのどのターゲットシンボルテーブルよりも先に検索されます。専用シンボルテーブルは、最初は空の状態ですが、`::nmadd dcmd` や `::nmde1 dcmd` を使用して操作できます。`::nm -P` オプションは、専用のシンボルテーブルの内容を表示するために使用されます。専用のシンボルテーブルによって、元のプログラムでは抜け落ちていたプログラム機能やデータのシンボル定義を作成できます。次からは、mdb がシンボル名をアドレスに変換したり、アドレスをもっとも近くのシンボルへ変換したりするときにはいつでも、これらの定義が使用可能となります。

ターゲットには複数のシンボルテーブルが含まれていて、各シンボルテーブルには複数のオブジェクトファイルからシンボルを入れることができるので、同じ名前で見異なるシンボルが存在することもあります。このような場合に、プログラマが希望するシンボル値を得られるように、mdb はシンボル名適用範囲演算子として、逆引用符(`()`)を使用します。プログラマは、シンボル名の解釈に使用する範囲を次のように指定できます。つまり、`object'name`、`file'name`、または `object'file'name` です。オブジェクトの識別子は、ロードオブジェクトの名前を参照します。ファイル識別子は、ソースファイルのベース名を参照します。ソースファイルは、指定されたオブジェクトのシンボルテーブル内に `STT_FILE` 型のシンボルを持っています。オブジェクト識別子の解釈は、ターゲットタイプによって決まります。

mdb カーネルターゲットでは、オブジェクトが、読み込まれたカーネルモジュールのベース名を指定すると考えられます。たとえば、次のシンボル名を考えてみましょう。

```
specfs'_init
```

これは、specfs カーネルモジュール内の `_init` シンボルの値を求めるために評価します。

mdb プロセスタargetでは、オブジェクトが、実行可能な名前、または読み込まれた共用ライブラリの名前を指定すると考えられます。この場合、次の形式のどれかが使用されます。

1. 完全一致 (つまり、フルパス名): `/usr/lib/libc.so.1`
2. ベース名に完全に一致: `libc.so.1`
3. ベース名の冒頭から接尾辞の「.」まで一致: `libc.so` または `libc`
4. リテラル文字列 `a.out` は、実行可能な名前の別名として受け入れられます。

プロセスタargetも前述の4つの形式を受け入れることができますが、この場合、接頭辞としてオプションのリンクマップ ID (lmid) が付きます。lmid 接頭辞は、LM、リンクマップ ID (16 進数)、および逆引用符の順番で構成されます。たとえば、次のシンボル名を考えてみましょう。

```
LM0'libc.so.1'_init
```

これは、`libc.so.1` ライブラリにある `_init` シンボルの値として評価して、この値がリンクマップ 0 (LM_ID_BASE) 上に読み込まれます。同じライブラリが複数のリンクマップ上に読み込まれている場合、シンボル名の重複を解決するためにリンクマップ指定子が必要になることもあります。リンクマップの詳細については、『[リンカーとライブラリ](#)』および [dlopen\(3C\)](#) を参照してください。リンクマップ識別子を表示するには、`showlmid` オプションの設定に従ってシンボルを出力します(「オプション」を参照)。

シンボルと 16 進整数値で名前が重複した場合、mdb は、最初にあいまいなトークンをシンボルとして評価し、次に整数値として評価しようとします。たとえば、`f` というトークンが、デフォルトの 16 進数では 10 進整数の 15 を表し、同時にターゲットのシンボルテーブル内の `f` という名前の大域変数を表す場合もあります。あいまいな名前を持つシンボルが存在するときには、明示的な `0x` または `0X` の接頭辞を用いることによって、整数値を明確に指定できます。

dcmd と walker の名前解決

前述のように、mdb の各 dmod は、一連の dcmd と walker を提供します。dcmd と walker は、2 つの異なる大域名前空間でトラックされます。また、mdb も、各 dmod に関連付けられた dcmd と walker の名前空間をトラックし続けます。1 つの dmod 内で、dcmd や walker に同じ名前を付けることはできません。このような名前の重複がある dmod は、読み込みに失敗します。異なる dmod から提供された dcmd や walker 間での名前の重複は、大域名前空間では許されます。名前の重複がある場合、その特定の読み込まれる名前を持つ dcmd または walker のうち、最初のものが大域名前空間で優先権を与えられます。ほかの定義は、読み取り順にリストに保存

されます。逆引用符 (') は、ほかの定義を選択するための参照範囲演算子として、`dcmd` や `walker` の名前に使用されます。たとえば、`dmod m1` と `m2` が、それぞれ `dcmd d` を提供する場合に、`m1` の方が `m2` よりも先に読み込まれたときには、次のようになります。

```
::d
  m1 の d 定義を実行します。
```

```
::m1'd
  m1 の d 定義を実行します。
```

```
::m2'd
  m2 の d 定義を実行します。
```

現時点で `m1` モジュールが読み込まれていない場合は、大域定義リスト上の次の `dcmd` である `m2'd` が、大域定義として使用されます。`dcmd` や `walker` の現在の定義は、次に示すように、`::which dcmd` を使用して定義できます。大域定義リストは、`::which -v` を使用して表示できます。

dcmd パイプライン `dcmd` は、縦棒演算子 (|) を使ってパイプラインの中へ入れることができます。パイプラインの目的は、一般的に仮想アドレスのような値のリストを、1つの `dcmd` や `walker` から次の `dcmd` や `walker` へと渡していくことです。パイプラインステージは、あるデータ構造体タイプのポインタを、それに対応するデータ構造体のポインタへと対応付けるために使用します。その目的は、アドレスリストをソートしたり、あるプロパティを持つ構造体のアドレスを選択したりすることです。

`mdb` は、パイプライン内の各 `dcmd` を左から右へと順番に実行します。現在のドット値、またはコマンドの開始時に明示的な式によって指定された値を使って、もっとも左にある `dcmd` が実行されます。縦棒演算子 (|) を見つけると、`mdb` は、その左側までの `dcmd` 出力、`mdb` 構文解析部、および空の値リストとの間に、パイプすなわち共用バッファを作成します。`dcmd` を実行するにしたがって、その標準出力はパイプの中に配置され、次に構文解析部によって使用され、評価されます。それは、あたかも `mdb` が標準出力からデータを読み込んでいるように見えます。各行には、終端に復帰改行文字またはセミコロン (;) を持つ算術式が含まれます。その算術式の値は、パイプに関連付けられた値のリストに追加されます。構文エラーが発見されると、そのパイプラインは異常終了します。

縦棒演算子 (|) の左側までの `dcmd` が完了すると、そのパイプに関連付けられた値のリストは、縦棒演算子 (|) の右側の `dcmd` を呼び出すために使用されます。リストの各値については、ドットにその値が設定されたあと、右側の `dcmd` が実行されます。パイプラインのもっとも右にある `dcmd` だけが、その出力を標準出力に表示します。パイプライン内のいずれかの `dcmd` が標準エラー出力を生じた場合は、それらのメッセージを直接標準エラーに出力するので、パイプラインの一部としては処理されません。

シグナル処理	デバッグは、PIPE および QUIT シグナルを無視します。INT シグナルの場合、現在実行中のコマンドが中止されます。ILL、TRAP、EMT、FPE、BUS、および SEGV のシグナルの場合、デバッグは中断され、特別な処理を行います。これらのシグナルが非同期的に生成された場合 (つまり、kill(2) を使用して別のプロセスから配信された場合)、mdb はシグナルをそのデフォルトの設定に復元して、コアダンプを生成します。しかし、これらのシグナルがデバッグプロセス自身によって同期的に生成され、外部的に読み込まれた dmod から dcmd が現在実行されており、さらに、標準入力が端末である場合、ユーザーは mdb が提供するメニューを使用して、強制的にコアダンプを生成するか、コアダンプを生成せずに終了するか、停止してデバッグに接続するか、あるいは、そのまま再開するかを選択できます。再開オプションを選択した場合、すべてのアクティブなコマンドは中止され、障害が発生したときに dcmd がアクティブであった dmod を読み込み解除します。この後、ユーザーは dmod を読み込み直すことができます。dcmd にバグがある場合に対して、再開オプションは制限付きの保護機能を提供します。再開オプションの危険性については、後述の「警告」にある「エラー回復メカニズムの使用」を参照してください。
コマンド行の再入力	端末デバイスから入力した最後の HISTSIZE (デフォルトは 128) 個のコマンドのテキストはメモリーに保存されます。次に説明するインライン編集機能が提供するキーマッピングを使用すると、この履歴リストから以前入力したコマンドを検索および取得できます。
インライン編集機能	標準入力が端末デバイスである場合、コマンド行を編集するために、mdb が提供するいくつかのシンプルな emacs スタイルの機能を使用できます。編集モードの search、previous、および next コマンドを使用すると、履歴リストにアクセスできます。検索するときに一致するのは、パターンではなく、文字列だけです。次に示すリストにおいて、制御記号はキャレット文字 (^) とそれに続く大文字の英字で表記されます。エスケープシーケンスは M- とそれに続く文字で表記されます。たとえば、M-f (「メタエフ」と呼ぶ) を入力するには、まず <ESC> を押して、次に「f」を押すか、あるいは Meta キーをサポートしているキーボード上では、まず Meta キーを押して、次に「f」を押します。コマンド行を発行および実行するには、復帰改行文字 (RETURN または NEWLINE) を使用します。編集コマンドは次のとおりです。 ^F カーソルを 1 文字だけ前方 (右) に移動します。 M-f カーソルを 1 単語だけ前方に移動します。 ^B カーソルを 1 文字だけ後方 (左) に移動します。 M-b カーソルを 1 単語だけ後方に移動します。 ^A カーソルを行の先頭に移動します。

^E

カーソルを行の末尾に移動します。

^D

カーソルのある行が空でない場合、カーソル位置の文字を削除します。カーソルのある行が空である場合、**^D**はEOFを意味し、デバッガは終了します。

M-^H

(メタ-バックスペース)直前の単語を削除します。

^K

カーソルから行の末尾までを削除します。

^L

画面を消去して、現在の行を出力し直します。

^T

現在の文字と次の文字を入れ換えます。

^N

履歴リストから次のコマンドを取得します。**^N**を入力するたびに、さらに次のコマンドが取得されます。

^P

履歴リストから前のコマンドを取得します。**^P**を入力するたびに、さらに前のコマンドが取得されます。

^R*[string]*

履歴リストから文字列を含むコマンドを後方に検索します。文字列は復帰改行文字 (RETURN または NEWLINE) で終了する必要があります。文字列を省略した場合、前回入力した文字列を含むコマンドを検索します。

編集モードはまた、次のようなユーザー定義シーケンスも編集コマンドとして解釈します。ユーザー定義シーケンスを読み取ったり、変更したりするには、[stty\(1\)](#) コマンドを使用します。

erase

ユーザー定義の消去文字 (通常は **^H** または **^?**)。前の1文字を削除します。

intr

ユーザー定義の割り込み文字 (通常は **^C**)。現在のコマンドを中断して、新しいプロンプトを出力します。

kill

ユーザー定義の強制終了文字 (通常は **^U**)。現在のコマンド行全体を強制終了します。

quit

ユーザー定義の終了文字 (通常は **^**)。デバッガを終了します。

suspend

ユーザー定義の中断文字 (通常は ^Z)。デバッガを中断します。

werase

ユーザー定義の単語消去文字 (通常は ^W)。直前の単語を消去します。

矢印キーのある拡張キーパッドをサポートするキーボードでは、mdb は次のキーストロークを編集コマンドとして解釈します。

up-arrow

履歴リストから前のコマンドを取得します (^P と同じ)。

down-arrow

履歴リストから次のコマンドを取得します (^N と同じ)。

left-arrow

カーソルを 1 文字だけ後方に移動します (^B と同じ)。

right-arrow

カーソルを 1 文字だけ前方に移動します (^F と同じ)。

出力ページャー

mdb は、組み込み出力ページャーを備えています。出力ページャーを使用できるのは、デバッガの標準出力が端末デバイスである場合だけです。コマンドを実行するたびに、mdb は 1 画面分の出力を生成してから中断し、次のようなページャーのプロンプトを表示します。

```
>> More [<space>, <cr>, q, n, c, a] ?
```

出力ページャーは次のようなキーシーケンスを認識します。

スペース

次の 1 画面分の出力を表示します。

a, A

現在のトップレベルのコマンドを中止して、プロンプトに戻ります。

c, C

現在のトップレベルのコマンドが完了するまで、画面ごとに中断するのではなく、出力を表示し続けます。

n, N, 復帰改行文字 (NEWLINE または RETURN)

次の 1 行分の出力を表示します。

q, Q, ^C, ^

現在の dcmd を終了 (中止) します。

フォーマット dcmd

/、\、?、=などのメタキャラクタを使用して、特別な出力書式の dcmd を表します。各 dcmd では、1 つまたは複数の書式制御文字を含む引数リスト、繰り返し回数、または引用文字列を使用できます。書式制御文字は、次の表に示すように、ASCII 文字の一種です。書式制御文字を使用して、ターゲットからデータを読み取り、フォーマットします。繰り返し回数は、書式制御文字の前に位置する正の

整数で、基数は、常に 10 進数として解釈されます。また、繰り返し回数は、先頭にドル記号を付けた角括弧で囲まれた式 ($\$[]$) として指定される場合もあります。文字列の引数は、二重引用符 (" ") で囲みます。フォーマット引数の間には、空白は不要です。

フォーマット dcmd は、次のとおりです。

/

ドットで指定される仮想アドレスで始まるターゲットの仮想アドレス空間からデータを表示します。

\

ドットで指定される物理アドレスで始まるターゲットの物理アドレス空間からデータを表示します。

?

ドットで指定される仮想アドレスに対応するオブジェクトファイル位置で始まるターゲットのプライマリオブジェクトファイルからデータを表示します。

=

指定されたデータ書式のそれぞれにおいて、ドット値そのものを表示します。したがって、= dcmd は、基底間の変換と計算を行うときに便利です。

また、mdb は、ドットのほかに、インクリメントと呼ばれる大域値も絶えずトラッキングしています。インクリメントは、ドットと、最後のフォーマット dcmd によって読み込まれるすべてのデータが後に続くアドレスとの距離を表します。たとえば、フォーマット dcmd を、A というアドレスに等しいドットで実行した結果、4 バイトの整数が出力された場合、この dcmd が終了したあとには、ドットはまだ A ですが、インクリメントは 4 に設定されています。前述の「算術展開」で説明したように、ここでは、正符号 (+) は、 $A + 4$ の値を出すための評価をします。その後、正符号は、次に続く dcmd 用のデータオブジェクトのアドレスにドットを設定し直します。

次の表に示すように、ほとんどの書式制御文字は、データ書式のサイズに対応するバイトの数だけ、インクリメントの値を増分します。書式制御文字表は、`::formats dcmd` を使用して、mdb の内部から表示できます。書式制御文字は、次のとおりです。

+	カウンタの数だけドットを増分する (変数サイズ)
-	カウンタの数だけドットを減分する (変数サイズ)
B	16 進数 int (1 バイト)
C	C の文字表記法を使う文字 (1 バイト)
D	10 進数の符号付き int (4 バイト)
E	10 進数の符号なし long long (8 バイト)

F	double (8 バイト)
G	8 進数の符号なし long long (8 バイト)
H	スワップバイトと short (4 バイト)
I	アドレスと分解命令 (変数サイズ)
J	16 進数 long long (8 バイト)
K	16 進数 uintptr_t (4 または 8 バイト)
N	改行
O	8 進数の符号なし int (4 バイト)
P	シンボル (4 または 8 バイト)
Q	8 進数の符号付き int (4 バイト)
R	2 進数 int (8 バイト)
S	C の文字列表記法を使った文字列 (変数サイズ)
T	水平タブ
U	10 進数の符号なし int (4 バイト)
V	10 進数の符号なし int (1 バイト)
W	デフォルト基数の符号なし int (4 バイト)
X	16 進数 int (4 バイト)
Y	復号化される time32_t (4 バイト)
Z	16 進数 long long (8 バイト)
^	インクリメント * カウントの数だけドットを減分する (変数サイズ)
a	symbol+offset としてのドット
b	8 進数の符号なし int (1 バイト)
c	文字 (1 バイト)
d	10 進数の符号付き short (2 バイト)
e	10 進数の符号付き long long (8 バイト)
f	float (4 バイト)
g	8 進数の符号付き long long (8 バイト)
h	スワップバイト (2 バイト)
i	命令の分解 (変数サイズ)

n	改行
o	8進数の符号なし short (2 バイト)
p	シンボル (4 または 8 バイト)
q	8進数の符号付き short (2 バイト)
r	余白
s	raw 文字列 (変数サイズ)
t	水平タブ
u	10進数の符号なし short (2 バイト)
v	10進数の符号付き int (1 バイト)
w	符号なしのデフォルト基数 short (2 バイト)
x	16進数 short (2 バイト)
y	復号化される time64_t (8 バイト)

h、\、および?のフォーマット dcmd を使用して、ターゲットの仮想アドレス空間、物理アドレス空間、またはオブジェクトファイルに書き込みを行うことができます。この場合には、次の修飾子の 1 つを最初の書式制御文字として指定し、次に、即値またはドル記号のあとの角括弧に囲まれた式 (\$[1]) で表されるワードのリストを指定します。

書き込み修飾子は、次のとおりです。

- v 各式の値の最下位バイトを、ドットで指定された位置から始まるターゲットに書き込む。
- w 各式の値の最下位 2 バイトを、ドットで指定された位置から始まるターゲットに書き込む。
- W 各式の値の最下位 4 バイトを、ドットで指定された位置から始まるターゲットに書き込む。
- Z 各式の値の 8 バイトすべてを、ドットで指定された位置から始まるターゲットに書き込む。

h、\、および?のフォーマット dcmd を使用して、ターゲットの仮想アドレス空間、物理アドレス空間、およびオブジェクトファイル内の特定の整数値をそれぞれ検索できます。この場合には、次の修飾子の 1 つを最初の書式制御文字として指定し、次に、値とオプションマスクを指定します。各値とマスクは、即値またはドル

記号の後の角括弧に囲まれた式として指定されます。値だけが指定されている場合、mdb は、適当なサイズの整数値を読み取り、一致する値が含まれるアドレスのところで終了します。また、V という値と、M というマスクが指定されている場合、mdb は、適当なサイズの整数値を読み取り、 $(X \& M) == V$ となるような値 X が含まれるアドレスのところで終了します。dcmd が終了すると、ドットは、一致した値が含まれるアドレスに更新されます。一致する値が見つからなかった場合、ドットは、最後に読み込まれたアドレスに残されます。

検索修飾子は、次のとおりです。

- I 指定された 2 バイトの値を検索する。
- L 指定された 4 バイトの値を検索する。
- M 指定された 8 バイトの値を検索する。

ユーザーターゲットでも、カーネルターゲットでも、アドレス空間は、一般的に不連続セグメントセットで構成されています。対応するセグメントを持たないアドレスから読み込むことはできません。セグメント内で一致するものが検索されない場合には、検索は強制的に終了します。

実行制御

mdb には、動作中のプログラムの実行を制御および追跡する機能があります。現在、ユーザープロセスターゲットだけが実行制御をサポートしています。mdb はシンプルな実行制御モデルを提供します。ターゲットプロセスを起動するには、`::run` を使用してデバッガ内から起動するか、`:A`、`::attach`、または `-p` コマンド行オプション (後述の説明を参照) を使用して mdb を既存のプロセスに接続します。ユーザーは追跡対象「ソフトウェアイベント」のリストを指定できます。追跡対象イベントがターゲットプロセス内で発生するごとに、ターゲット内のすべてのスレッドが停止して、そのイベントをトリガーしたスレッドが代表スレッドとして選択され、デバッガに制御が戻ります。ターゲットプログラムが「running (動作中)」と設定されると、ユーザー定義割り込み文字 (通常は ^c) を入力して、非同期的にデバッガに制御を戻すことができます。

「ソフトウェアイベント」とは、デバッガが監視しているターゲットプログラムにおける状態遷移のことです。たとえば、デバッガは、プログラムカウンタレジスタが特定の値 (ブレイクポイント) に変更されたり、特定のシグナルが送られたりするのを監視できます。

「ソフトウェアイベント指定子」とは、デバッガがどのようなイベントを監視するかをターゲットプログラムに指示するときに使用するソフトウェアイベントのクラスのことです。ソフトウェアイベント指定子のリストを表示するには、`::events` dcmd を使用します。各ソフトウェアイベント指定子には標準プロパティセットが関連付けられています (後述の「`::events`」を参照)。

デバッグはさまざまなソフトウェアイベント(たとえば、ブレイクポイント、ウォッチポイント、シグナル、マシン障害、およびシステムコールなど)を監視できます。`::bp`、`::fltp`、`::sigbp`、`::sysbp`、または`::wp`を使用すると、新しいソフトウェアイベント指定子を作成できます。各指定子には、コールバック(コマンド行に入力された場合と同じように動作する mdb コマンド文字列)とプロパティーセットが関連付けられています(後述の説明を参照)。異なるコールバックとプロパティーを持つのであれば、任意の数の指定子を同じイベントに作成できます。追跡イベントの現在のリストや対応するイベント指定子のプロパティーを表示するには、`::events dcmd`を使用します。イベント指定子のプロパティーの定義については、後述の「`::events dcmd`」と「`::evset dcmd`」を参照してください。

実行制御組み込み `dcmd`(後述の説明を参照)はいつでも利用できますが、実行制御がサポートされないターゲットに適用しようとする、「このターゲットはサポートされません」というエラーメッセージが表示されます。デバッグの実行制御による `exec`、`attach`、`release`、およびジョブ制御との対話の詳細については、後述の「注意事項」を参照してください。

イベントコールバック

`::evset dcmd` とイベント追跡用の `dcmd` を使用すると、(-c オプションを使用して)、イベント指定子ごとにイベントコールバックを関連付けることができます。「イベントコールバック」とは、ターゲットにおいて対応するイベントが発生したときに実行される mdb コマンドを表す文字列のことです。このようなコマンドは、コマンドプロンプトに入力された場合と同じように実行されます。各コールバックが実行される前に、`dot` 変数には代表スレッドのプログラムカウンタの値が設定され、「`hits`」変数には該当する指定子が一致した回数(現在の一致も含む)が設定されます。

ターゲットを継続するための1つまたは複数のコマンド(たとえば、`::cont` や `::step`)が、イベントコールバック自身に含まれる場合、このようなコマンドはターゲットをすぐに継続したり、ターゲットがもう一度停止するまで待機したりしません。その代わりに、このようなターゲットを継続する `dcmd` は「現在、継続操作が中断されている」ということをイベントコールバック内部に示して、すぐに戻ります。したがって、複数の `dcmd` がイベントコールバックに含まれている場合、ステップまたは継続用の `dcmd` は最後に指定する必要があります。「すべて」のイベントコールバックを実行したあと、一致した「すべて」のイベントコールバックが継続を要求している場合、ターゲットはすぐに実行を再開します。要求された継続操作が競合する場合、どの種類の操作を継続するかは、優先順位がもっとも高い継続操作が決定します。優先順位は(高いものから)、「ステップ(step)」、「ステップオーバー(step-over)または次(next)」、「ステップアウト(step-out)」、そして「継続(continue)」の順番になります。

スレッドのサポート

mdb は、ターゲットに関連する各スレッドのスタックとレジスタを調査する機能を提供します。永続的な「スレッド」変数には、現在の代表スレッド識別子が入っています。スレッド識別子の書式はターゲットによって異なります。`::regs dcmd` と `::fpregs dcmd` を使用すると、代表スレッドのレジスタセットや、別のスレッドのレジスタセットが現在利用できる場合はそのレジスタセットを調査できます。さら

に、代表スレッドのレジスタセットは名前付き変数セットとしてエクスポートされます。1つまたは複数のレジスタの値を変更するには、対応する名前付き変数に `> dcmd` を適用します。

`mdb` カーネルターゲットは、対応する内部スレッド構造体の仮想アドレスを指定されたスレッドの識別子としてエクスポートします。Solaris カーネル内のスレッドのデバッグに関するサポートの詳細は、『[Oracle Solaris Modular Debugger Guide](#)』を参照してください。`mdb` プロセスタargetは、ネイティブな `lwp_*` インタフェース、`/usr/lib/libthread.so`、または `/usr/lib/lwp/libthread.so` を使用するマルチスレッド化されたユーザープロセスの検査に対する適切なサポートを提供します。動作中のユーザープロセスをデバッグするとき、`mdb` はシングルスレッド化されたプロセスが `libthread` を `dlopen` または `close` するかどうかを検出して、動作中のユーザープロセスのスレッド化モデルのビューを自動的に調整します。プロセスターゲットスレッド識別子は、アプリケーションが使用するスレッド化モデルに基づいて、代表スレッドの `lwpid_t`、`thread_t`、または `pthread_t` に対応付けられます。

`mdb` がユーザープロセスターゲットをデバッグしているとき、コンパイラによってサポートされるスレッドに局所的な記憶領域をターゲットが利用している場合、`mdb` は自動的に、スレッドに局所的な記憶領域を参照するシンボル名を、現在の代表スレッドに対応する記憶領域のアドレスで評価します。`::tls` 組み込み `dcmd` を使用すると、代表スレッド以外のスレッドのシンボルの値も表示できます。

組み込み dcmd

`mdb` は、常に定義されている組み込み `dcmd` セットを備えています。これらの `dcmd` のなかには、特定のターゲットだけに適用されるものもあります。`dcmd` が現在のターゲットに適用できない場合、その `dcmd` は停止し、「コマンドが現在のターゲットに適用されません」という内容のメッセージが表示されます。`mdb` は、多くの場合、レガシー [adb\(1\)](#) `dcmd` の名前に対応するニモニック (`::identifier`) を提供します。たとえば、`::quit` は、`$q` に相当します。[adb\(1\)](#) の使用経験を持つプログラマや、簡略符号あるいは難解なコマンドを認識するプログラマは、組み込みコマンドの `$` や `:` 形式の方を好むかもしれません。一方、`mdb` に慣れていないプログラマは、より詳細でわかりやすい `::` 形式を好むでしょう。次に組み込みコマンドを、アルファベット順に説明します。`$` または `:` の形式に `::identifier` に対応するものがある場合、`::identifier` 形式の下に `$` または `:` の形式を示します。組み込み `dcmd` は、次のとおりです。

`> variable-name`

`>/modifier/variable-name`

指定された名前の変数にドット値を割り当てます。変数が読み取り専用の場合には、変更できません。`>` の後ろに `//` で囲まれた修飾子がある場合、ドット値は割り当ての一部として変更されます。修飾子は、次のとおりです。

`c`
符号なし `char` の量 (1 バイト)

`s`
符号なし `short` の量 (2 バイト)

i
符号なし int の量 (4 バイト)

l
符号なし long の量 (32 ビットでは 4 バイト、64 ビットでは 8 バイト)

これらの演算子はキャストを実行しません。代わりに、指定された数の下位バイト (リトルエンディアンアーキテクチャーの場合) または上位バイト (ビッグエンディアンアーキテクチャーの場合) を取得します。修飾子には下位互換性があります。ただし、mdb の `*/modifier/` および `%/modifier/` 構文を使用する必要があります。

`$<macro-name`

指定したマクロファイルからコマンドを読み取り、実行します。ファイル名は、絶対パスまたは相対パスとして与えられます。ファイル名に「/」が含まれない場合は単純名です。単純名の場合、mdb は、マクロファイル組み込みパス内でそのファイル名を検索します。現時点で別のマクロファイルが処理されている場合、そのファイルは閉じられ、代わりに新しいファイルが処理されます。

`$<<macro-name`

`$<` と同様に、指定されたマクロファイルからコマンドを読み取って実行しますが、現在開いているマクロファイルは閉じません。

`$?`

ターゲットがユーザープロセスまたはコアファイルの場合、まず、ターゲットのプロセス ID と現在のシグナルを出力して、次に、代表スレッドの汎用レジスタセットを出力します。

`[address] $C [count]`

C スタックのバックトレースを、スタックフレームポインタの情報も含めて出力します。この `dcmd` の前に明示的な `address` がある場合には、その仮想記憶アドレスから始まるバックトレースを表示します。その他の場合には、代表スレッドのスタックを表示します。オプションのカウント値が引数として指定されている場合には、出力の各スタックフレームに対して、`count` 引数で指定された数の引数だけが表示されます。

`[base] $d`

デフォルトの出力基数を取得または設定します。この `dcmd` の前に明示的な式がある場合には、デフォルトの出力基数は、指定された `base` に設定されます。その他の場合には、現在の基数が 10 進数で出力されます。デフォルトの基数は 16 (16 進数) です。

`$e`

既知の外部すなわち大域的なオブジェクト型シンボルや関数シンボルのリスト、そのシンボルの値、およびターゲットの仮想アドレス空間内の対応位置に格納される最初の 4 バイト (32 ビット mdb) または 8 バイト (64 ビット mdb) のリストを出力します。::nm `dcmd` には、シンボルテーブルの表示用にさらに柔軟なオプションが用意されています。

\$P prompt-string

指定された *prompt-string* にプロンプトを設定します。デフォルトのプロンプトは、「>」です。::set -P または -P コマンド行オプションを使用しても、プロンプトは設定できます。

distance \$s

アドレスからシンボル名へ変換するための、シンボルマッチングディスタンスを取得または設定します。シンボルマッチングディスタンスのモードについては、「オプション」の -s コマンド行オプションで説明します。::set -s オプションを使用しても、シンボルマッチングディスタンスは変更できます。距離が指定されない場合には、現在の設定が表示されます。

\$v

指定された変数のうち、ゼロ以外の値を持つ変数のリストを出力します。::vars dcmd を使用すると、変数の一覧表示にはほかのオプションを付けることができます。

width \$w

出力のページ幅を指定された値に設定します。通常は、mdb が端末に幅の照会をしてサイズを変更するので、このコマンドは必要ありません。

\$W

ターゲットを書き込み用にもう一度開きます (-w オプションをコマンド行に指定して mdb を実行する場合と同じ)。::set -w オプションを使用しても、書き込みモードを有効にできます。

[pid] ::attach [core | pid]

[pid] :A [core | pid]

ユーザープロセスターゲットが動作中の場合には、指定されたプロセス ID またはコアファイルに接続して、デバッグします。コアファイルのパス名は、文字列引数として指定されます。プロセス ID は、この dcmd の前で、文字列引数として、または式の値として指定されます。デフォルトは 16 進数であることを忘れないでください。したがって、pgrep(1) や ps(1) を使用して得た 10 進数のプロセス ID (PID) を式として指定する場合には、その先頭に「0t」を付けてください。

[address] ::bp [-/-dDesT] [-c cmd] [-n count] sym ...

address :b [cmd ...]

指定された場所にブレークポイントを設定します。::bp dcmd は、指定されたアドレスまたはシンボル (dcmd の前にある明示的な式で指定されたオプションのアドレスを含む) ごと、そして、dcmd の後ろにある文字列または即値ごとにブレークポイントを設定します。引数には、指定された特定の仮想アドレスを示すシンボル名または即値を指定できます。シンボル名を指定した場合、ターゲットプロセス内でまだ評価できないシンボルを参照できます。つまり、まだ読み込んでいないオブジェクト内のオブジェクト名や関数名を参照できます。この場合、ターゲット内のブレークポイントは延期され、指定された名前に一致するオブジェクトが読み込まれるまでアクティブ(有効)になりません。オブジェクトが読み込まれると、このブレークポイントは自動的に有効になります。共用ライ

ブラリ内に定義されたシンボル上のブレークポイントを設定する場合、アドレスは実際のシンボル定義ではなく、対応する PLT (Procedure Linkage Table) エントリを参照する可能性があるため、必ず、アドレス式ではなく、シンボル名を使用する必要があります。PLT エントリ上で設定されたブレークポイントは、その PLT エントリがあとで実際のシンボル定義に解釈されると、実行時リンクエディタによって上書きされることがあります。-d、-D、-e、-s、-t、-T、-c、および -n オプションは、`::evset dcmd` で使用するのと同じです (後述の説明を参照)。`dcmd` の `:b` 形式を使用した場合、ブレークポイントは `dcmd` の前にある式で指定した仮想アドレスだけに設定されます。`:b dcmd` の後ろにある引数は連結され、コールバック文字列となります。この文字列にメタキャラクタが含まれる場合、文字列を引用符で囲む必要があります。

`::cat filename ...`

ファイルを連結および表示します。各ファイル名は、相対パス名または絶対パス名として指定できます。ファイルの内容は、標準出力に表示されますが、出力ページャには渡されません。この `dcmd` は、`|` 演算子とともに使用するようになっています。したがって、プログラマは外部ファイルに格納されたアドレスリストを使用してパイプラインを処理できます。

`::cont [ SIG ]`

`:c [ SIG ]`

デバッグを中断し、ターゲットプログラムを継続し、指定されたソフトウェアイベントが発生したあとにターゲットプログラムが終了または停止するまで待機します。`-o nostop` オプションを有効にしてデバッグを動作中のプログラムに接続しており、ターゲットがすでに動作している場合、この `dcmd` は単に、指定されたソフトウェアイベントが発生したあとにターゲットプログラムが終了または停止するまで待機します。オプションのシグナルの名前または番号

([signal.h\(3HEAD\)](#) を参照) が引数として指定された場合、このシグナルはターゲットの実行を再開するための一部として、すぐにターゲットに送られます。SIGINT シグナルを追跡している場合、ユーザー定義割り込み文字 (通常は `^c`) を入力すると、非同期的にデバッグに制御を戻すことができます。この SIGINT シグナルは自動的に消去され、次回ターゲットが継続される場合、ターゲットにはこのシグナルは送られません。現在、ターゲットプログラムが動作していない場合、`::cont` は、`::run` を実行した場合と同じように、新しいプログラムの実行を起動します。

`address ::context`

`address $p`

指定されたプロセスへのコンテキストスイッチ。コンテキストスイッチの操作は、カーネルターゲットを使用している場合にだけ有効です。プロセスのコンテキストを指定するには、カーネルの仮想アドレス空間において、そのプロセスの `proc` 構造体の `address` を使用します。特別なコンテキストアドレス「0」は、カーネルそのもののコンテキストを表すときに使用されます。カーネルページだけの場合とは対照的に、指定されたユーザープロセスの物理メモリーページがクラッシュダンプに含まれる場合、クラッシュダンプを検査するときに `mdb` が実行できるのはコンテキストスイッチだけです。[dumppadm\(1M\)](#)

を使用すると、すべてのページまたは現在のユーザープロセスのページをダンプできるようにカーネルクラッシュダンプ機能を構成できます。::status dcmd を使用すると、現在のクラッシュダンプの内容を表示できます。

ユーザーがカーネルターゲットからコンテキストスイッチを要求した場合には、mdb は指定されたユーザープロセスに相当する新しいターゲットを作成します。スイッチが発生すると、新しいターゲットは、自身の dcmd を大域レベルに置きます。したがって、このとき、/dcmd が、ユーザープロセスの仮想アドレス空間からデータをフォーマットして表示したり、::mappings dcmd が、ユーザープロセスのアドレス空間でマッピングを表示したりするようになります。0::context を実行すると、カーネルターゲットを復元できます。

::dcmds

使用可能な dcmd を一覧表示し、各 dcmd の簡単な説明を出力します。

[address] ::delete [id|all]

[address] :d [id|all]

指定された ID 番号のイベント指定子を削除します。id 番号引数はデフォルトで 10 進数として解釈されます。オプションのアドレスを dcmd の前に指定した場合、指定された仮想アドレスに関連するすべてのイベント指定子(たとえば、そのアドレスに影響するすべてのブレークポイントまたはウォッチポイント)が削除されます。特別な引数「all」を指定した場合、すべてのイベントが削除されます。ただしスティッキー (T フラグ) がマークされたイベントは除きます。::events dcmd はイベント指定子の現在のリストを表示します。

[address] ::dis [-fw] [-n count] [address]

最後の引数または現在のドット値によって指定されたアドレス、またはそのアドレス周辺から、逆アセンブルします。アドレスが既知の関数の最初の部分に一致した場合は、関数全体が逆アセンブルされます。その他の場合は、指定されたアドレスの前後の命令の「ウィンドウ」が、コンテキストの理解のために出力されます。デフォルトでは、命令はターゲットの仮想アドレス空間から読み取られます。-f オプションを指定した場合は、命令はターゲットのオブジェクトファイルから読み取られます。デバッガが現在、動作中のプロセス、コアファイル、またはクラッシュダンプに接続されていない場合、-f オプションはデフォルトで有効になります。また、アドレスが既知の関数の最初の部分に一致した場合でも、-w オプションを指定すると、「ウィンドウ」を強制的に開くモードに設定できます。デフォルトでは、ウィンドウのサイズは命令 10 個分です。-n オプションを使用すれば、命令の数を明示的に指定できます。

::disasms

使用可能な逆アセンブラのモードを一覧表示します。ターゲットが初期化されている場合には、mdb は適切な逆アセンブラモードを選択しようとします。また、::dismode dcmd を使用して、ユーザーは、初期モードを一覧表のどれかに変更できます。

::dismode [mode]

\$V [mode]

逆アセンブラモードを受け取るか、設定します。引数が指定されていないと、現在の逆アセンブラモードを出力します。*mode* 引数が指定されている場合には、逆アセンブラを指定されたモードに切り替えます。また、`::disasms dcmd` を使用して、逆アセンブラのリストを表示できます。

`::dmods [-l] [module-name]`

読み込まれたデバッグモジュールを一覧表示します。`-l` オプションが指定されていると、各 *dmod* に関連付けられた *dcmd* や *walker* の一覧がその *dmod* 名の下に出力されます。特定の *dmod* の名前を追加の引数として指定すれば、出力はその *dmod* に限定されます。

`[address] ::dump [-eqrstu] [-f] [-p]`

`[-g bytes] [-w paragraphs]`

ドットによって指定されたアドレスを含む、16 バイトで割り当てられた記憶領域のメモリーダンプを 16 進数の ASCII 形式で出力します。`::dump` に繰り返し回数を指定すると、ダンプする繰り返し数としてではなく、ダンプするバイト数として解釈されます。また、`::dump dcmd` は、次のオプションも認識します。

`-e`

エンディアンに合わせて調整します。`-e` オプションでは、4 バイトのワードを使用できます。`-g` オプションを使用して、デフォルトのワードサイズを変更できます。

`-f`

ターゲットの仮想アドレス空間からではなく、指定された仮想アドレスに対応するオブジェクトファイルの位置からデータを読み取ります。デバッガが現在、動作中のプロセス、コアファイル、またはクラッシュダンプに接続されていない場合、`-f` オプションはデフォルトで有効になります。

`-g bytes`

バイトをバイトのグループで表示します。デフォルトの *group* サイズは 4 バイトです。*group* サイズは行幅を分割する 2 のべき乗にする必要があります。

`-p`

address を、仮想アドレスではなく、ターゲットのアドレス空間内の物理的なアドレス位置として解釈します。

`-q`

データの ASCII 形式の復号化を出力しません。

`-r`

各行に、絶対的なアドレスではなく、開始アドレスからの相対的な行数を与えます。このオプションを指定すると、`-u` オプションも暗黙的に指定されます。

`-s`

繰り返しの行を省略します。

-t
行全体を読み取って出力するのではなく、指定されたアドレスの内容だけを読み取って表示します。

-u
段落の境界に配列するのではなく、配列せずに出力します。

-w *paragraphs*
1行あたり 16 バイトの段落で段落を表示します。デフォルトの段落数は 1 です。-w の最大値は 16 です。

::echo [*string* | *value* ...]
空白文字で区切られ、復帰改行文字で終わる引数を標準出力に出力します。\$[] で囲まれた式は値に評価されて、デフォルトで出力されます。

::eval *command*
指定された文字列をコマンドとして評価し、実行します。コマンドがメタキャラクタやスペースを含む場合は、引用符や二重引用符で囲みます。

::events [-av]

\$b [-av]
ソフトウェアイベント指定子のリストを表示します。各イベント指定子には一意の ID 番号が割り当てられ、この ID 番号を使用すると、あとでイベント指定子を削除または変更できます。デバッガは、それ自体の内部イベントで追跡を有効にすることもできます。それらのイベントは、-a オプションが指定されている場合にのみ表示されます。-v オプションを指定した場合、たとえば、指定子がアクティブになっていない理由などの、より詳細な情報が表示されます。次に、一部の出力例を示します。

```
> ::events
      ID S TA HT LM Description                               Action
-----
[ 1 ] - T   1  0 stop on SIGINT                               -
[ 2 ] - T   0  0 stop on SIGQUIT                             -
[ 3 ] - T   0  0 stop on SIGILL                               -
...
[ 11] - T   0  0 stop on SIGXCPU                              -
[ 12] - T   0  0 stop on SIGXFSZ                              -
[ 13] -    2  0 stop at libc'printf                          ::echo printf
>
```

次に、各カラムの意味を説明します。::help events を使用しても、この情報のサマリーを表示できます。

ID
イベント指定子の識別子。この識別子は、指定子が有効である場合は角括弧「[]」で囲まれ、指定子が無効である場合は丸括弧「()」で囲まれ、指定された指定子に一致するイベントが発生したあとにターゲットプログラムが現在停止している場合は山括弧「< >」で囲まれます。

S

イベント指定子の状態。この状態は次のシンボルのうちの1つです。

-

イベント指定子はアイドル状態です。どのターゲットプログラムも動作していないとき、すべての指定子はアイドル状態になります。ターゲットプログラムが動作しているとき、評価できない指定子はアイドル状態になることがあります。(たとえば、共用オブジェクトがまだ読み込まれておらず、共用オブジェクト内のブレークポイントが延期されている場合)。

+

イベント指定子はアクティブです。ターゲットが継続されると、この種類のイベントがデバッガによって検出されます。

*

イベント指定子は作動準備されてます。この状態は、ターゲットが現在動作しており、この種類のイベントが監視されていることを意味します。この状態を表示できるのは、`-o nostop` オプションを使用してデバッガを動作中のプログラムに接続している場合だけです。

!

オペレーティングシステムのエラーのために、イベント指定子は作動準備されていません。`::events -v` オプションを使用すると、監視が失敗した理由についてより詳細な情報が表示されます。

TA

Temporary、Sticky、および Automatic というイベント指定子のプロパティ。次のシンボルのうちの1つまたは複数が表示されます。

t

このイベント指定子は一時的であり、次回ターゲットが停止すると、一致するかどうかにかかわらず削除されます。

T

このイベント指定子はスティッキーであり、`::delete all` または `:z` では削除されません。このイベント指定子を削除するには、その ID を明示的に `::delete` に指定します。

d

このイベント識別子は、ヒット回数がヒット制限に到達すると自動的に無効になります。

D

このイベント識別子は、ヒット回数がヒット制限に到達すると自動的に削除されます。

s

このターゲットは、ヒット回数がヒット制限に到達すると自動的に停止します。

HT

現在のヒット回数。このカラムには、当該イベント指定子が作成されたあとに、対応するソフトウェアイベントがターゲット内で発生した回数が表示されます。

LM

現在のヒット制限。このカラムには、自動無効、自動削除または自動停止が発生するときのヒット回数の制限が表示されます。このような動作を構成するには、`::evset dcmd` を使用します (後述の説明を参照)。

「説明」

指定された指定子に一致するソフトウェアイベントの種類の説明。

処置

対応するソフトウェアイベントが発生したときに実行されるコールバック文字列。このコールバックは、コマンドプロンプトに入力された場合と同じように実行されます。

`[id] ::evset [-/-dDestT] [-c cmd] [-n count] id ...`

1つまたは複数のソフトウェアイベント指定子のプロパティを変更します。プロパティは、`dcmd` の前にあるオプションの式と `dcmd` の後ろにあるオプションの引数リストによって識別される指定子ごとに設定されます。基数が明示的に指定されない限り、引数リストは10進数の整数のリストとして解釈されます。`::evset dcmd` は次のオプションを認識します。

-d

ヒット回数がヒット制限に到達すると、イベント識別子を無効にします。`-d` 形式のオプションを指定した場合、この動作は無効になります。イベント指定子が無効になると、もう一度有効になるまで、デバッガは対応する監視をすべて削除し、対応するソフトウェアイベントを無視します。`-n` オプションを指定しない場合、指定子はすぐに無効になります。

-D

ヒット回数がヒット制限に到達すると、イベント識別子を削除します。`-D` 形式のオプションを指定した場合、この動作は無効になります。`-D` オプションは `-d` オプションよりも優先されます。ヒット制限を構成するには、`-n` オプションを使用します。

-e

イベント指定子を有効にします。`-e` 形式のオプションを指定した場合、この動作は無効になります。

-s

ヒット回数がヒット制限に到達すると、ターゲットプログラムを停止します。`-s` 形式のオプションを指定した場合、この動作は無効になります。`-s` オプションを指定すると、デバッガは指定子のコールバックが実行されるたびに (ただし、 N 回目の実行を除く (N は指定子のヒット制限の現在の値))、`::cont` が発行されたかのように動作します。`-s` オプションは `-D` オプションと `-d` オプションよりも優先されます。

-t

イベント指定子を「一時的」であるとマークします。一時的な指定子は、次回ターゲットが停止すると、指定した指定子に対応するソフトウェアイベントの結果によって停止したかどうかに関わらず、自動的に削除されます。-t 形式のオプションを指定した場合、「一時的」マーカーは削除されます。-t オプションは -T オプションよりも優先されます。

-T

イベント指定子を「スティッキー」であるとマークします。スティッキー指定子は、`::delete all` や `:z` では削除されません。対応する指定子の ID を `::delete` への明示的な引数として指定することによって削除できます。-T 形式のオプションを指定した場合、「スティッキー」プロパティは削除されます。イベント指定子のデフォルトセットには初めから「スティッキー」のマークが付いています。

-c

対応するソフトウェアイベントがターゲットプログラム内で発生するたびに、指定された *cmd* 文字列を実行します。現在のコールバック文字列を表示するには、`::events` を使用します。

-n

ヒット制限の現在の値を *count* に設定します。ヒット制限を現在設定せずに、-n オプションと一緒に -s または D オプションを指定していない場合、ヒット制限は 1 に設定されます。

`::help evset` を使用しても、この情報のサマリーを表示できます。

`::files`

\$f

既知のソースファイルの一覧、すなわち、種々のターゲットシンボルテーブルの中にある STT_FILE 型のシンボルを出力します。

`[flt] ::fltbp [-/-dDestT] [-c cmd] [-n count] flt ...`

指定されたマシン障害を追跡します。障害を特定するには、*dcmd* の前にオプションの障害番号を指定するか、*dcmd* の後ろに障害の名前または番号のリストを指定します (<sys/fault.h> を参照)。-d、-D、-e、-s、-t、-T、-c、および -n オプションは、`::evset dcmd` で使用するのと同じです。

`[thread] ::fpregs`

`[thread] $x, $X, $y, $Y`

代表スレッドの浮動小数点レジスタセットを出力します。スレッドを指定すると、そのスレッドの浮動小数点レジスタが表示されます。スレッド式は、前述の「スレッドのサポート」で説明したスレッド識別子の 1 つである必要があります。

::formats

/、\、?、=などのフォーマット *dcmd* とともに使用する、利用可能な出力書式制御文字の一覧を表示します。フォーマットとその使用法については、前述の「フォーマット *dcmd*」で説明しています。

::grep command

指定されたコマンド文字列を評価したあと、新しいドット値がゼロ以外の場合には、古いドット値を出力します。*command* にスペースやメタキャラクタが含まれる場合は、必ず引用符で囲んでください。パイプライン内で **::grep dcmd** を使用すると、アドレスリストをフィルタ処理できます。

::help [dcmd-name]

引数がない場合には、**::help dcmd** は、*mdb* で使用可能なヘルプ機能の概要を簡潔に出力します。*dcmd-name* が指定されている場合には、*mdb* は、その *dcmd* の使用法の概略を出力します。

signal :i

ターゲットが動作中のユーザープロセスである場合、指定されたシグナルを無視して、透過的にターゲットにそのシグナルを送ります。指定されたシグナルの送り先を追跡しているイベント指定子はすべて、追跡イベントのリストから削除されます。デフォルトでは、無視されるシグナルは、デフォルトでプロセスがコアをダンプする原因となるシグナルセット ([signal.h\(3HEAD\)](#) を参照) の補集合に初期化されます (ただし、デフォルトで追跡される SIGINT を除く)。

\$i

デバッガが無視して、ターゲットが直接処理するシグナルのリストを表示します。追跡対象シグナルについてのより詳細な情報を取得するには、**::events dcmd** を使用します。

[address] ::if [-p] {type member tests | [type] [at off] test}

テストを評価し、テストが真である場合に古いドット値を出力します。

-p

仮想アドレスではなく、物理アドレスを使用します。

2種類のテストがあります。最初のものを使用すると、構造体やユニオンのメンバーの値をテストできます。2番目のものでは、ドットからのオフセットをテストできます。次に、これらのテストを AND または OR を使って結合して、より複雑なテストを構築できます。

最初の例では、構造体のメンバーをテストします:

```
::if "struct foo" namep <> 0
```

構造体 *foo* 内の要素 *namep* が 0 でない場合、この文は古いドット値を出力します。

```
::if "struct foo" namep <> 0 AND namep->name <> 0 AND  
namep->name streq "bar"
```


foo 内の namep が指し示す構造体内の要素 name が 0 でなく、文字列 bar を含む文字列を指し示す場合、前の文は古いドット値を出力します。

```
::if "struct foo" name <> 0 AND name streq "bar" and value = 0x123
```

前の文で出力が行なわれるのは、名前が bar で value が 0x123 である場合だけです。

2 番目の形式では、指定したオフセットのタイプが指定した値を保持するかどうかをテストします。次に例を示します。

```
::if uint_t at 0x34 = 0x123
```

これは、uint_t のオフセット 0x34 の値が 0x123 である場合に出力します。ここでも、AND や OR を使用して、これをほかのテストと組み合わせることができます。この形式では、::if には、シンボリックタイプ情報がない場合でも使用可能な内部の組み込みタイプの数が含まれます。これらのタイプを次に示します:

```
char, uchar, short, ushort, int, uint, long, ulong,
longlong, ulonglong, pointer, addr
```

タイプが addr の場合、そのアドレスの値ではなく実際のアドレスがテストされます。このため、次のステートメント:

```
::if addr at 0x34 = 0x100034
```

は、ドット + 0x34 が 0x100034 である場合にのみ、ドットの値を報告します。一方:

```
::if pointer at 0x34 = 0x100034
```

は、値がドット + 0x34 の場合にのみ出力します。

より実用性の高い、次の文:

```
::if "char *" at 0x34 streq "foo"
```

は、オフセット 0x34 で指し示す文字列が foo の場合に、古いドット値を出力します。

実行可能なテストは次のとおりです:

=	- True if the values are equal.
<>	- True if the values are not equal.
<	- True if the value is less than the right hand value.
<=	- True if the value is less than or equal to the right hand value.
>	- True if the value is greater than the right hand value.
>=	- True if the value is greater than or equal to the right hand value.
&	- True if the values ANDed together are non zero.
^	- True if the values XORed together are non zero.
streq	- True if the strings exactly match.


```

strneq      - True if the strings don't match.
strcaseeq   - True if the strings match case insensitively.
strncaseeq  - True if the strings do not match case insensitively.
strleneq    - True if the string is this length.
strlenneg   - True if the string is not this length.
strlengt    - True if the string is longer than this.
strlenlt    - True if the string is shorter than this.
strleng     - True if the string is this long or longer.
strlenle    - True if the string is this long or shorter

```

テストの右側には、次のいずれかを指定できます:

- 構造体またはユニオン内の要素。
- シンボルからの値。
- 絶対値。
- `<var` — 変数 `var` の値。
- `$expr` - mdb 式 `expr` の評価後のドット値。これは、二重引用符 ("") を使って囲む必要があります。例:

```
"$<var=J"
```

は、変数 `var` の値を取得します。

```
::kill
```

```
:k
```

ターゲットが動作中のユーザープロセスである場合、強制的にターゲットを終了します。また、ターゲットが本来デバッガで `::run` を使用して作成されていた場合、デバッガを終了すると、ターゲットは強制的に終了されます。

```
$l
```

ターゲットがユーザープロセスである場合、代表スレッドの LWPID を出力します。

```
$L
```

ターゲットがユーザープロセスである場合、ターゲット内にある各 LWP の LWPID を出力します。

```
[address] ::list [-b back_member][-p][-L] [type] member [variable-name]
```

リンクリストデータ構造体の要素を調べて、リスト内の各要素のアドレスを出力します。リスト内の最初の要素のアドレスは、オプションのアドレスを使用して指定できます。その他の場合には、リストは現在のドット値から始まるものとみなされます。mdb が適切なサイズのオブジェクトから読み取ることができるように、`type` パラメータは C 言語の構造体または共用体を指定する必要があります、リスト内の要素の型を記述するのに使用されます。`::list` でタイプを変別できる場合は、`type` を省略できます。`member` パラメータは、リスト内の次の要素へのポインタを含む、`type` のメンバーを指定するのに使用されます。`::list dcmd` が反復処理を続行するのは、NULL ポインタに遭遇するか、最初の要素に再度達する(循環リスト)か、ループ検出が有効(-L)でかつループが検出されるか、バックメ

ンバーチェックが有効 (-b) であるか、不良バックポインタが検出されるか、または要素の読み取り時にエラーが発生するまでです。オプションの *variable-name* が指定されている場合には、mdb がパイプラインの次のステージを呼び出すときに walk の各ステップが返す値に、指定変数が割り当てられます。

-b *back_member*

back_member がリストの直前のメンバーを指し示すことを確認します。リストの最初のメンバーには、NULL バックポインタの保持が許可されます。

-L

リスト内のループをチェックします。リスト内のすべての要素を 1 回出力します。

-p

仮想アドレスではなく、物理アドレスから読み取ります。

::list dcmd を使用できるのは、mdb で使用するように設計されているシンボリックデバッグ情報を含むオブジェクトに対してのみです。詳細は、後述の「注意事項」の「シンボリックデバッグ情報」を参照してください。

::load [-s] *module-name*

指定された dmod をロードします。モジュール名は、絶対パスまたは相対パスとして指定します。*module-name* が単純名、つまり「/」を含んでいない場合には、mdb はモジュールライブラリパス内で検索します。モジュールの名前に重複があった場合には、そのモジュールは読み込まれません。その場合は、まず既存のモジュール名を読み込み解除する必要があります。-s オプションを指定すると、mdb はモジュールを発見または読み込めなくても何も出力せず、エラーメッセージも表示しません。

::log [-d | [-e] *filename*]

\$> [*filename*]

出力ログを有効にしたり、無効にしたりします。mdb は、相互ログ機能を提供しているので、まだユーザーとの対話処理が行われているときにも、入力コマンドと標準出力の両方が同じファイルに記録できます。-e オプションでファイルを指定すると、指定したファイルへのログの書き込みが有効になり、ファイル名を指定しない場合、前回のログファイルへの書き込みが再度有効になります。-d オプションは、ログを無効にします。また、\$> dcmd を使用する場合、ファイル名引数が指定されているときには、ログが有効になります。その他の場合、ログは無効になります。指定されたログファイルがすでに存在する場合、mdb は新しいログ出力をそのファイルに追加します。

::map *command*

文字列引数として指定される *command* を使用して、ドット値を対応する値へ割り当ててから新しい値を出力します。*command* にスペースやメタキャラクタが含まれる場合は、必ず引用符で囲んでください。::map dcmd をパイプライン内で使用すると、アドレスのリストを新しいアドレスリストに変換できます。

```
[ address ] :: mappings [ name ]
```

```
[ address ] $m [ name ]
```

ターゲットの仮想アドレス空間内の各割り当てを、アドレス、サイズ、それぞれの割り当て記述などを含めて一覧表示します。*address* が *dcmd* の前にある場合、*mdb* は指定されたアドレスを含む割り当てだけを表示します。文字列 *name* の引数を指定した場合、*mdb* はその説明と一致する割り当てだけを表示します。

```
::next [ SIG ]
```

```
::e [ SIG ]
```

ターゲットプログラムを 1 命令だけ実行しますが、サブルーチンの呼び出しまでは進めます。オプションのシグナルの名前または番号 ([signal.h\(3HEAD\)](#) を参照) が引数として指定された場合、このシグナルはターゲットの実行を再開するための一部として、すぐにターゲットに送られます。どのターゲットプログラムも動作していない場合、*::next* は、*::run* を実行したかのように新しいプログラムを起動し、最初の命令で停止します。

```
[ address ] :: nm [ -DPdghnopuvx ] [ -t types ]
```

```
[ -f format ] [ object ]
```

現在のターゲットに関連付けられたシンボルテーブルを出力します。オプションの *address* を *dcmd* の前に指定した場合、*address* に対応するシンボル用のシンボルテーブルエントリだけが表示されます。*object* を指定すると、この読み込みオブジェクト用のシンボルテーブルだけが表示されます。また、*::nm dcmd* は、次のオプションも認識します。

-D

.symtab の代わりに *.dynsym* (動的シンボルテーブル) を出力します。

-P

.symtab の代わりに専用シンボルテーブルを出力します。

-d

値とサイズフィールドを 10 進数で出力します。

-g

大域シンボルだけを出力します。

-h

ヘッダー行を抑制します。

-n

名前順にシンボルをソートします。

-o

値とサイズフィールドを 8 進数で出力します。

-p

シンボルを、一連の *::nmadd* コマンドとして出力します。このオプションは *-P* とともに使用して、マクロファイルを作成できます。その後、*\$<* コマンドを用いて、このマクロファイルをデバッガに読み込みます。

- u
未定義のシンボルだけを出力します。
- v
値順にシンボルをソートします。
- x
値とサイズフィールドを 16 進数で出力します。
- t *type[,type ...]*
指定された型のシンボルだけを出力します。有効な型引数文字列は次のとおりです。
 - noty
STT_NOTYPE
 - objt
STT_OBJECT
 - func
STT_FUNC
 - sect
STT_SECTION
 - file
STT_FILE
 - comm
STT_COMMON
 - tls
STT_TLS
 - regi
STT_SPARC_REGISTER
- f *format[,format ...]*
指定されたシンボル情報だけを出力します。有効な書式化引数文字列は次のとおりです。
 - ndx
シンボルテーブルのインデックス
 - val
シンボル値
 - size
サイズ (単位はバイト)
 - type
シンボルの型

bind
結合

oth
その他

shndx
セクションのインデックス

name
シンボル名

ctype
シンボルの C 言語の型 (既知の場合)

obj
シンボルを定義するオブジェクト

value *::nmadd* [-fo] [-e *end*] [-s *size*] *name*

指定されたシンボルの名前を、専用シンボルテーブルへ追加します。mdb は、構成可能な専用シンボルテーブルを用意しています。前述の「シンボルの名前解決」で説明したように、この専用テーブルは、ターゲットのシンボルテーブル内に置くことができます。また、*::nmadd dcmd* は、次のオプションも認識します。

-e
シンボルのサイズを *end-value* に設定します。

-f
シンボルのタイプを **STT_FUNC** に設定します。

-o
シンボルのタイプを **STT_OBJECT** に設定します。

-s
シンボルのサイズを *size* に設定します。

::nmdel *name*

指定されたシンボルの名前を専用シンボルテーブルから削除します。

::objects [-v]

既知の読み込みオブジェクトのプライマリ割り当て (通常はテキストセクション) に対応するマッピングだけを表示して、そのターゲットの仮想アドレス空間の割り当てを出力します。-v オプションは、各ロードオブジェクトのバージョンを表示します。すべてのロードオブジェクトのバージョン情報が得られるわけではありません。バージョン情報のないロードオブジェクトは、バージョンが「不明」のオブジェクトとして -v オプションの出力に表示されます。

::offsetof member

指定された型のメンバーのオフセットを出力します。型は、C 言語の構造体の名前にします。メンバーが指定されない場合は、この型の全メンバーが報告されます。

オフセットはバイトで出力されます。ただし、メンバーがビットフィールドの場合、オフセットはビットで出力されます。分かりやすくするために、出力の末尾には常に適切な単位が付きます。型名には、前述の「シンボルの名前解決」で説明した逆引用符 (') 有効範囲規則を使用できます。::offsetof dcmd は、mdb で使用されるように設計されたシンボルデバッグ情報を含むオブジェクトだけに使用できます。詳細は、後述の「注意事項」の「シンボルデバッグ情報」を参照してください。

```
address ::print [-aCdiLptx] [-c lim]
[-l lim] [type [member ...]]
```

指定された仮想 *address* にあるデータ構造体を、指定された *type* 情報を使用して出力します。*type* パラメータは C 言語の構造体、共用体、列挙型、基本的な整数型、あるいは、これらの型へのポインタを指定できます。型名にスペース文字が含まれる場合 (たとえば、「struct foo」)、単一引用符または二重引用符で囲む必要があります。型名には、前述の「シンボルの名前解決」で説明した逆引用符 (') 有効範囲規則を使用できます。*type* が構造化された型である場合、::print dcmd は構造体または共用体の各メンバーを再帰的に出力します。*type* 引数を指定せず、かつ、静的または大域的な STT_OBJECT シンボルがアドレスに一致する場合、::print は自動的に適切な型を推測します。*type* 引数を指定した場合は、その後に *member* 式のリストをオプションで指定できます。その場合は、指定された *type* のメンバーまたはサブメンバーだけが表示されます。*type* にその他の構造化された型が含まれる場合、区切り記号のピリオド ('.') で区切ったメンバー名のリストを生成することで、各メンバー文字列はサブ構造体要素を参照できます。::print dcmd は、mdb で使用されるように設計されたシンボルデバッグ情報を含むオブジェクトだけに使用できます。詳細は、後述の「注意事項」の「シンボルデバッグ情報」を参照してください。データ構造体を表示したあと、::print はドットを *type* のサイズ分 (バイト) だけインクリメントします。

-a オプションを指定すると、各メンバーのアドレスが表示されます。-p オプションを指定すると、::print はアドレスを、仮想メモリアドレスではなく、物理メモリアドレスとして解釈します。-t オプションを指定すると、各メンバーの型が表示されます。-d または -x オプションを指定した場合は、すべての整数が 10 進数 (-d) または 16 進数 (-x) で表示されます。デフォルトでは、発見的方法を使用して、値が 10 進数で表示されるか 16 進数で表示されるかを判別します。文字列として読み取りまたは表示される文字配列内の文字数は -c オプションで制限できます。-c オプションを指定すると、文字数は制限されません。読み取りまたは表示される標準配列内の要素数は -l オプションで制限できます。-L オプションを指定すると、文字数は制限されず、配列内のすべての要素が表示されます。-c と -l オプションのデフォルト値を変更するには、「オプション」で説明する ::set または -o コマンド行オプションを使用します。

-i オプションを指定した場合、アドレス値は出力する即値として解釈されます。値を解釈する型を指定する必要があります。型が64ビットより小さい場合、即値は型のサイズである場合と同じように解釈されます。-i オプションを-p オプションとともに使用することはできません。-a オプションを指定した場合、表示されるアドレスはゼロから始まるバイトオフセットです。

::quit

\$q

デバッガを終了します。

[*thread*] ::regs

[*thread*] \$r

代表スレッドの汎用レジスタセットを出力します。スレッドを指定すると、そのスレッドの汎用レジスタセットが表示されます。スレッド式は、前述の「スレッドのサポート」で説明したスレッド識別子の1つである必要があります。

::release [-a]

:R [-a]

以前に接続されたプロセスまたはコアファイルを解放します。-a オプションを指定すると、プロセスは解放され、停止および中断されたままになります。このようなプロセスを継続するには `prun(1)` を使用し (`proc(1)` を参照)、再開するには `mdb` などのデバッガを適用します。デフォルトでは、解放されたプロセスは、`mdb` の `::run` で作成された場合には強制的に終了され、`mdb` の -p オプション、`::attach` または `:A dcmd` で接続されていた場合には解放および実行状態に設定されます。

::run [*args...*]

:r [*args...*]

指定された引数を使用して、新しいターゲットプログラムの実行を起動し、接続します。引数はシェルによって解釈されません。デバッガがすでに動作中のプログラムを検査している場合、`::release` が実行された場合と同じように、最初にこのプログラムから切断します。

::set [-wF] [-/ -o *option*] [-s *distance*] [-I *path*]

[-L *path*] [-P *prompt*]

デバッガの種々のプロパティを取得または設定します。いずれのオプションも指定されていない場合には、デバッガのプロパティの現在の設定が表示されます。`::set dcmd` は次のオプションを認識します。

-F

その次のユーザープロセスで、`::attach` が適用されているプロセスを強制的に引き継ぎます (-F オプションをコマンド行に指定して `mdb` を実行する場合と同じ)。

-I

マクロファイルを検出するためのデフォルトパスを設定します。パス引数は特殊トークンを使用できます。「オプション」の -I コマンド行オプションの説明を参照してください。

-L

デバッガモジュールを検出するためのデフォルトパスを設定します。パス引数は特殊トークンを使用できます。「オプション」の -I コマンド行オプションの説明を参照してください。

-O

指定したデバッガオプションを有効にします。-o 書式が使用されている場合には、そのデバッガオプションを無効にします。オプションの文字列については、-o コマンド行オプションとともに「オプション」で説明されています。

-P

コマンドプロンプトを、指定されたプロンプト文字列に設定します。

-S

シンボルマッチングディスタンスを指定された距離に設定します。詳細は、「オプション」の -s コマンド行オプションの説明を参照してください。

-w

ターゲットを書き込み用にもう一度開きます (-w オプションをコマンド行に指定して mdb を実行する場合と同じ)。

::showrev [-pv]

ハードウェアおよびソフトウェアの変更情報を表示します。オプションを指定しないと、一般的なシステム情報が表示されます。-v オプションは、すべてのロードオブジェクトのバージョン情報を表示します。-p オプションは、パッチの一部としてシステムにインストールされたロードオブジェクトのバージョン情報だけを表示します。すべてのロードオブジェクトのバージョン情報が得られるわけではありません。バージョン情報のないロードオブジェクトは、-p オプションの出力から省略され、バージョンが「不明」のオブジェクトとして -v オプションの出力に表示されます。

[signal] ::sigbp [-/-dDestT] [-c cmd] [-n count] SIG...

[signal] :t [-/-dDestT] [-c cmd] [-n count] SIG...

指定されたシグナルの送り先を追跡します。シグナルを特定するには、dcmd の前にオプションのシグナル番号を指定するか、dcmd の後ろにシグナルの名前または番号のリストを指定します (signal.h(3HEAD) を参

照)。-d、-D、-e、-s、-t、-T、-c、および -n オプションは、::evset dcmd で使用するのと同じです。最初に、デフォルトでプロセスがコアをダンプする原因となるシグナルセット (signal.h(3HEAD) を参照) と SIGINT が追跡されます。

::sizeof [-s size] [-r min max] [type [member ...]]

オプションは、次のとおりです:

-s size

このサイズのエントリの型を示します。

-r min max

サイズ範囲のエントリの型を示します。

指定された型のサイズをバイトで表示します。*type* パラメータはC言語の構造体、共用体、枚举型、基本的な整数型、あるいは、これらの型へのポインタを指定できます。型名には、前述の「シンボルの名前解決」で説明した逆引用符(⁂)有効範囲演算子を使用できます。*member* は、標準のC言語構文で、配列インデックス作成演算子 “[*index*]”、構造体メンバー演算子 “.” を使用して指定できます。

::sizeof dcmd は、mdb で使用されるように設計されたシンボルデバッグ情報を含むオブジェクトだけに使用できます。詳細は、後述の「注意事項」の「シンボルデバッグ情報」を参照してください。

```
[ address ] ::stack [ count ]
```

```
[ address ] $c [ count ]
```

C スタックのバックトレースを出力します。この dcmd の前に明示的な *address* がある場合には、その仮想記憶アドレスから始まるバックトレースを表示します。その他の場合には、代表スレッドのスタックを表示します。オプションのカウント値が引数として指定されている場合には、出力の各スタックフレームに対して、*count* 引数で指定された数の引数だけが表示されます。

```
::status
```

現在のターゲットに関連した情報のサマリーを出力します。

```
::step [ over | out ] [ SIG ]
```

```
:s [ SIG ]
```

```
:u [ SIG ]
```

ターゲットプログラムを 1 命令だけ実行します。オプションのシグナルの名前または番号 ([signal.h\(3HEAD\)](#) を参照) が引数として指定された場合、このシグナルはターゲットの実行を再開するための一部として、すぐにターゲットに送られます。*over* 引数を指定した場合、::step はサブルーチンの呼び出しまで進めます。::step *over* 引数は ::next dcmd と同じです。オプションの *out* 引数を指定した場合、代表スレッドが現在の関数から戻ってくるまで、ターゲットプログラムは実行を継続します。どのターゲットプログラムも動作していない場合、::step *out* は、::run を実行したかのように新しいプログラムを起動し、最初の命令で停止します。:s dcmd は ::step と同じです。:u dcmd は ::step *out* と同じです。

```
[ syscall ] ::sysbp [ -/-dDestT ] [ -io ] [ -c cmd ]
```

```
[ -n count ] syscall...
```

エントリを追跡するか、指定されたシステムコールから終了します。システムコールを特定するには、dcmd の前にオプションのシステムコール番号を指定するか、dcmd の後ろにシステムコールの名前または番号のリストを指定します (<sys/syscall.h> を参照)。-i オプションを指定した場合 (デフォルト)、イベント指定子はシステムコールごとにカーネルに入るトリガーとなります。-o オプションを指定した場合、イベント指定子はカーネルから終了するトリガーとなります。-d、-D、-e、-s、-t、-T、-c、および -n オプションは、::evset dcmd で使用するのと同じです。

thread::tls symbol

指定されたスレッドのコンテキストにおいて、指定された TLS (Thread-Local Storage) シンボル用の記憶領域のアドレスを出力します。スレッド式は、前述の「スレッドのサポート」で説明したスレッド識別子の1つである必要があります。シンボル名には、前述の「シンボルの名前解決」で説明している任意の有効範囲規則を使用できます。

::typeset [-/-t] variable-name...

指定された変数に属性を設定します。1つまたは複数の名前が指定されている場合は、それらを定義して、ドット値に設定します。-t オプションを指定すると、各変数に関連付けられたユーザー定義のタグが設定されます。-t オプションを指定すると、そのタグは削除されます。変数名が何も指定されていない場合には、変数のリストとその値を出力します。

::unload module-name

指定された dmod を読み込み解除します。::dmods dcmd を使用すると、動作中の dmod のリストを出力できます。組み込みモジュールは、読み込み解除できません。使用中のモジュール、すなわち現在実行中の dcmd を提供しているモジュールは、解除できません。

::unset variable-name...

定義された変数リストから、指定された変数の設定を解除、すなわち削除します。mdb によってエクスポートされている変数の中には、永続的と指定されていて、ユーザーが削除できないものがあります。

::vars [-npt]

指定された変数の一覧を表示します。-n オプションを指定すると、その出力は、ゼロ以外の変数に限定されます。-p を指定すると、変数は \$<dcmd を使用して、デバッガの再処理に適切な形式で出力されます。このオプションを使用すると、変数をマクロファイルに記録しておき、あとでこれらの値を復元できます。-t を指定すると、タグ付き変数だけが出力されます。変数にタグを付けるには、::typeset dcmd の -t オプションを使用します。

::version

デバッガのバージョン番号を出力します。

address::vtop [-a as]

可能な場合、指定された仮想アドレスに対する物理アドレスのマッピングを出力します。::vtop dcmd を利用できるのは、カーネルターゲットを検査しているとき、あるいは、::context dcmd を実行したあとで、カーネルクラッシュダンプ内のユーザープロセスを検査しているときだけです。

カーネルコンテキストからカーネルターゲットを検査しているとき、-a オプションを使用すると、仮想アドレスから物理アドレスへの変換で使われる代替アドレス空間構造体のアドレス (as) を指定できます。デフォルトでは、この変換にはカーネルのアドレス空間が使用されます。ダンプに含まれるのがカーネルページだけの場合でも、このオプションはアクティブなアドレス空間に対して利用できます。

```
[ address ] ::walk walker-name [ variable-name ]
```

指定された walker を使用して、データ構造体の要素を調べます。::walkers dcmd を使用すると、使用可能な walker を一覧表示できます。walker は、大域的なデータ構造体について動作する場合もあり、開始アドレスを必要としないものがあります。たとえば、カーネル内の proc 構造体のリストを調べる場合などです。その他の walker は、アドレスが明示的に指定されている固有のデータ構造体上で動作します。たとえば、アドレス空間でポインタを指定して、セグメントのリストを調べる場合です。対話処理で使用される場合、::walk dcmd は、データ構造体内の各要素のアドレスをデフォルト形式で出力します。また、この dcmd は、パイプラインにアドレスリストを提供するときにも使用できます。walker 名には、前述の dcmd and Walker Name Resolution で説明した逆引用符 (') 有効範囲規則を使用できます。オプションの *variable-name* が指定されている場合には、mdb がパイプラインの次のステージを呼び出すときに walk の各ステップが返す値に、指定変数が割り当てられます。

```
::walkers
```

使用可能な walker の一覧と、各 walker の簡潔な説明を出力します。

```
::whence [ -v ] name ...
```

```
::which [ -v ] name ...
```

指定された dcmd と walker をエクスポートする dmod を出力します。これらの dcmd を使用すると、指定された dcmd または walker の大域定義を現在提供しているのはどの dmod かを判断できます。大域的な名前解決の詳細については、前述の「dcmd と walker の名前解決」のセクションを参照してください。-v オプションを指定すると、各 dcmd や walker の代替定義を優先順に出力します。

```
addr [ ,len ] ::wp [ -/-dDestT ] [ -rwx ] [ -c cmd ]
```

```
[ -n count ]
```

```
addr [ ,len ] :a [ cmd ... ]
```

```
addr [ ,len ] :p [ cmd ... ]
```

```
addr [ ,len ] :w [ cmd ... ]
```

指定された場所にウォッチポイントを設定します。監視される領域の長さをバイト数で設定するには、dcmd の前に繰り返し回数を指定します。長さを明示的に設定しない場合、デフォルトは1バイトです。::wp dcmd を使用すると、読み取り (-r オプション)、書き込み (-w オプション)、または実行 (-x オプション) のアクセス権の任意の組み合わせでのトリガーが可能となるようにウォッチポイントを構成できます。-d、-D、-e、-s、-t、-T、-c、および -n オプション

は、::evset dcmd で使用すると同じです。:a dcmd は、指定されたアドレスに読み取り権のウォッチポイントを設定します。:p dcmd は、指定されたアドレスに実行権のウォッチポイントを設定します。:w dcmd は、指定されたアドレスに書き込み権のウォッチポイントを設定します。:a、:p、および :w dcmd の後ろにある引数は連結され、コールバック文字列となります。この文字列にメタキャラクタが含まれる場合、文字列を引用符で囲む必要があります。

```
::xdata
```

現在のターゲットによってエクスポートされた外部データバッファを一覧表示します。外部データバッファは、現在のターゲットに関連付けられた情報を示

します。この情報は、標準ターゲット機能ではアクセスできないもので、アドレス空間、シンボルテーブル、レジスタセットなどが含まれています。これらのバッファは、`dcmd` による使用が可能です。詳細については、『[Oracle Solaris Modular Debugger Guide](#)』を参照してください。

:Z

すべてのイベント指定子を追跡対象ソフトウェアイベントのリストから削除します。`::delete` を使用しても、イベント指定子は削除できます。

オプション

サポートしているオプションは、次のとおりです。

-A

`mdb` モジュールの自動読み込みを無効にします。デフォルトでは、`mdb` は、ユーザープロセスまたはコアファイルのアクティブな共用ライブラリに対応しているデバッガモジュール、または稼働中のオペレーティングシステムかオペレーティングシステムのクラッシュダンプにある読み込み済みのカーネルモジュールに対応しているデバッガモジュールを読み込もうとします。

-f

強制的に `raw` ファイルデバッグモードに入ります。デフォルトでは、`mdb` は、オブジェクトとコアファイルのオペランドがユーザーの実行可能ファイルとコアダンプを参照しているのか、または1組のオペレーティングシステムのクラッシュダンプファイルを参照しているのかを判断しようとします。ファイルのタイプを推測できない場合、デバッガはデフォルトでそのファイルをプレーンなバイナリデータとして調査します。`-f` オプションを使用すると、`mdb` は引数を調査すべき `raw` ファイルセットとして解釈します。

-F

必要に応じて、指定されたユーザープロセスに強制的に接続します。デフォルトでは、`mdb` は、すでに `truss(1)` など別のデバッグ用ツールの制御下にあるユーザープロセスへの接続を拒否します。`-F` オプションを指定すると、`mdb` はこれらのプロセスに接続します。これによって、`mdb` とプロセスを制御しようとしているほかのツールとの間で予期しない相互作用が発生する可能性があります。

-I *path*

マクロファイルを検出するためのデフォルトのパスを設定します。マクロファイルは、`$<` または `$<< dcmd` を使用して読み取ります。このときのパスは、一連のディレクトリ名をコロン (:) 文字で区切ったものです。`-I include` パスと `-L library` パス (以降を参照) には、次のトークンを含めることができます。

%i

現在の命令セットアーキテクチャー (ISA) の名前 (「`sparc`」、「`sparcv9`」、または「`i386`」) に展開されます。

%o

変更対象のパスの古い値まで展開されます。これは、既存のパスの前または後ろにディレクトリを追加するときには有用です。

%p

現在のプラットフォーム文字列 (`uname -i` またはプロセスのコアファイルあるいはクラッシュダンプに格納されているプラットフォーム文字列) に展開されます。

%r

ルートディレクトリのパス名まで展開されます。-R オプションを使用すると、代替ルートディレクトリを指定できます。-R オプションを指定しないと、ルートディレクトリは mdb 実行可能ファイル自体へのパスから動的に決定されます。たとえば、`/bin/mdb` を実行した場合、ルートディレクトリは `/` です。`/net/hostname/bin/mdb` を実行した場合、ルートディレクトリは `/net/hostname` となります。

%t

現在のターゲット名まで展開されます。これはリテラル文字列「`proc`」(ユーザープロセスまたはユーザープロセスのコアファイル)、「`kvm`」(カーネルクラッシュダンプまたは稼働中のオペレーティングシステム)、または「`raw`」(raw ファイル)のいずれかです。

32 ビットの mdb に対するデフォルトのインクルードパスは、次のとおりです。

```
%r/usr/platform/%p/lib/adb:%r/usr/lib/adb
```

64 ビットの mdb に対するデフォルトのインクルードパスは、次のとおりです。

```
%r/usr/platform/%p/lib/adb/%i:%r/usr/lib/adb/%i
```

-k

強制的にカーネルデバッグモードにします。デフォルトでは、mdb は、オブジェクトとコアファイルのオペランドがユーザーの実行可能ファイルとコアダンプを参照しているのか、または 1 組のオペレーティングシステムのクラッシュダンプファイルを参照しているのかを判断しようとします。-k オプションを指定すると、mdb は、これらのファイルがオペレーティングシステムのクラッシュダンプファイルであるとみなします。オブジェクトまたはコアオペランドを指定せずに -k オプションを指定すると、mdb は、オブジェクトファイルを `/dev/ksyms` に、コアファイルを `/dev/kmem` にデフォルト設定します。`/dev/kmem` に読み取りアクセスできるのはグループ `sys` だけです。書き込みアクセスにはすべての特権が必要です。

-K

`kmdb` をロードし、実行中のオペレーティングシステムカーネルを停止して、`kmdb` デバッガプロンプトを表示します。このオプションは、システムコンソールでしか使用してはいけません。それに続く `kmdb` プロンプトがシステムコンソールに表示されます。

-L *path*

デバッガモジュールを検索するためのデフォルトのパスを設定します。モジュールは起動時に自動的に読み込まれるか、または `::load dcmd` を使用して読

み込まれます。このときのパスは、一連のディレクトリ名をコロン(:)文字で区切ったものです。-L ライブラリパスには、前述の -I オプションで示したトークンも含めることができます。

-m

カーネルモジュールシンボルのデマンドローディングを無効にします。デフォルトでは、mdb は読み込まれたカーネルモジュールのリストを処理し、モジュールごとにシンボルテーブルのデマンドローディングを実行します。-m オプションを指定すると、mdb はカーネルモジュールのリストを処理したり、モジュールごとにシンボルテーブルを提供したりしなくなります。結果として、アクティブなカーネルモジュールに対応する mdb モジュールは起動時に読み込まれません。

-M

すべてのカーネルモジュールシンボルを事前に読み込みます。デフォルトでは、mdb はカーネルモジュールシンボルのデマンドローディングを実行します。アドレスがそのモジュールのテキストであるとき、またはデータセクションが参照されているとき、モジュールのシンボルテーブルが完全に読み取られます。-M オプションを指定すると、mdb は起動時にすべてのカーネルモジュールのシンボルテーブルを完全に読み込みます。

-o *option*

指定したデバッガオプションを有効にします。-o 形式のオプションを使用した場合は、指定したオプションが無効になります。次に掲載しているものを除いて、各オプションともデフォルトでは無効になっています。mdb は次のオプション引数を認識します。

adb

[adb\(1\)](#) の互換性をより厳密にします。プロンプトは空の文字列に設定され、出力ページャーなどの多数の mdb 機能が無効になる。

array_mem_limit=*limit*

:::print が表示する配列のメンバー数に対してデフォルトの制限を設定します。*limit* が特別なトークン none である場合、デフォルトで配列のすべてのメンバーが表示される。

array_str_limit=*limit*

:::print が char 型の配列を出力するときに ASCII 文字列として表示する文字数に対してデフォルトの制限を設定します。*limit* が特別なトークン none である場合、デフォルトで char 配列全体が string として表示される。

follow_exec_mode=*mode*

[exec\(2\)](#) システムコールに従って動作するようにデバッガを設定します。*mode* は次の名前付き定数の 1 つである必要があります。

ask

stdout が端末デバイスである場合、デバッガは [exec\(2\)](#) システムコールが返ったあとに停止して、exec または stop のどちらに従うかをユーザーに尋ねる。stdout が端末デバイスではない場合、ask モードはデフォルトで stop になる。

follow

デバッガは自動的にターゲットプロセスを継続し、新しい実行可能ファイルに基づいて、ターゲットプロセスのマッピングとシンボルテーブルをすべてリセットし、`exec`に従う。`follow`の動作の詳細については、後述の「注意事項」の「`exec`との対話」を参照。

stop

`exec` システムコールから戻ったあと、デバッガは `exec` システムコールに従うことを停止します。`stop` の動作の詳細については、後述の「注意事項」の「`exec`との対話」を参照。

follow_fork_mode=mode

`fork(2)`、`fork1(2)`、または `vfork(2)` システムコールに従って動作するようにデバッガを設定します。`mode` は次の名前付き定数の 1 つである必要があります。

ask

`stdout` が端末デバイスである場合、デバッガは `fork(2)` システムコールが返ったあとに停止して、親プロセスまたは子プロセスのどちらに従うかをユーザーに尋ねる。`stdout` が端末デバイスではない場合、`ask` モードはデフォルトで `parent` になる。

parent

デバッガは親プロセスに従い、子プロセスから切断して子プロセスが動作するように設定します。

child

デバッガは子プロセスに従い、親プロセスから切断して親プロセスが動作するように設定します。

ignoreeof

端末に EOF シーケンス (^D) が入力されても、デバッガは終了しません。終了するには `::quit dcmd` を使用する必要がある

nostop

`-p` オプションを指定したか、`::attach` または `:A dcmd` を適用した場合、ユーザープロセスと接続しているときには、そのユーザープロセスを停止しません。`nostop` の動作の詳細については、後述の「注意事項」の「プロセスの接続と解放」を参照。

pager

出力ページャーを有効にする (デフォルト)。

repeatlast

復帰改行文字がコマンドとして端末に入力された場合、`mdb` は前のコマンドを現在のドットの値で繰り返す。`-o adb` を指定した場合、このオプションも自動的に指定されている

showlmid

mdb は **LM_ID_BASE** と **LM_ID_LDSO** 以外のリンクマップを使用するユーザーアプリケーションでシンボルの命名と識別をサポートする(「シンボルの名前解決」を参照)。**LM_ID_BASE** または **LM_ID_LDSO** 以外のリンクマップ上のシンボルは **LMlmid'library'symbol** のように表示される(このとき、**lmid** はデフォルトの出力基数(16進数)のリンクマップID)。**showlmid** オプションを有効にすると、すべてのシンボルとオブジェクト(**LM_ID_BASE** と **LM_ID_LDSO** に関連するものも含む)のリンクマップID有効範囲を表示するように **mdb** を構成できる。オブジェクトファイル名を扱う組み込み **dcmd** (**::nm**、**::mappings**、**\$m**、**::objects** など)は、前述の **showlmid** の値に従ってリンクマップIDを表示します。

-p pid

指定されたプロセスIDに接続し、そのプロセスを停止します。**mdb** は、**/proc/pid/object/a.out** ファイルを実行可能ファイルのパス名として使用します。

-P prompt

コマンドプロンプトを設定します。デフォルトのプロンプトは、「>」です。

-R root

パス名の展開に合わせてルートディレクトリを設定します。デフォルトでは、ルートディレクトリは **mdb** 実行可能ファイル自体のパス名から導かれます。ルートディレクトリは、パス名の展開の際に **%r** トークンと置き換えられます。

-s distance

アドレスからシンボル名への変換用のシンボルマッチングディスタンスを、指定した **distance** に設定します。デフォルトでは、**mdb** はこの距離をゼロに設定し、スマートマッチングモードを有効にします。ELFシンボルテーブルのエントリには値 **V** とサイズ **S** が含まれ、関数またはデータオブジェクトのサイズがバイト単位で示されます。スマートモードでは、**mdb** は、**A** が **[V, V+S)** の範囲にある場合、アドレス **A** と与えられたシンボルとを一致させます。ゼロ以外の距離を指定した場合も同じアルゴリズムが使用されますが、前述の式に **S** を指定した場合、常に絶対距離が指定され、シンボルのサイズは無視されます。

-S

ユーザーの **~/.mdbrc** ファイルの処理を抑制します。デフォルトでは、**mdb** は、**\$HOME** で定義されているユーザーのホームディレクトリにマクロファイル **.mdbrc** があれば、それを読み取って処理します。**-S** オプションを指定すると、このファイルは読み取られません。

-u

強制的にユーザーデバッグモードにします。デフォルトでは、**mdb** は、オブジェクトとコアファイルのオペランドがユーザーの実行可能ファイルとコアダンプを参照しているのか、または1組のオペレーティングシステムのクラッシュダ

ンプファイルを参照しているのかを判断しようとしています。-u オプションを使用すると、mdb は、これらのファイルがオペレーティングシステムのクラッシュダンプファイルではないとみなします。

-U

kmdb が読み込まれた場合は、それを読み込み解除します。kmdb が使用中でない場合は、それを読み込み解除し、カーネルデバッガで使用されたメモリを解放してオペレーティングシステムが利用できるようにします。

-V *version*

逆アセンブラのバージョンを設定します。デフォルトでは、mdb は、デバッグターゲットに対する適切な逆アセンブラのバージョンを判断しようとしています。-v オプションを使用すると、逆アセンブラを明示的に設定できます。::disasms dcmd によって、使用可能な逆アセンブラのバージョンが一覧表示されます。

-w

指定したオブジェクトとコアファイルを書き込み用に開きます。

-W

入出力デバイスに割り当てられるメモリアドレスへのアクセスを許可します。デフォルトでは、mdb はそのようなアクセスを許可しません。これは、多くのデバイスには無効なソフトウェア操作に対するハードウェア保護機能がないためです。このオプションは、デバイスドライバのデバッグ時にのみ、注意して使用してください。

-y

tty モードに対する明示的な端末初期化シーケンスを送信します。端末には、たとえば cmdtool(1) など、tty モードに切り換えるために明示的な初期化シーケンスを必要とするものがあります。この初期化シーケンスがないと、mdb からスタンドアウトモードなどの端末機能を使用できない場合があります。

オペランド

次のオペランドがサポートされています。

object

調査する ELF 書式のオブジェクトファイルを指定します。mdb が調査および編集できるのは、ELF 書式の実行可能ファイル (ET_EXEC)、ELF 動的ライブラリファイル (ET_DYN)、ELF 再配置可能オブジェクトファイル (ET_REL)、およびオペレーティングシステムのシンボルテーブル (unix.X ファイル) です。

core

ELF プロセスコアファイル (ET_CORE) またはオペレーティングシステムのクラッシュダンプ (vmcore.X ファイル) を指定します。ELF コアファイルオペランド (core) を指定する場合、対応するオブジェクトファイルを指定しないと、mdb はいくつかの異なるアルゴリズムを使用して、コアファイルを生成した実行可能ファイルの名前を推測しようとしています。このような実行可能ファイルが見つからなかった場合、mdb は動作し続けますが、いくつかのシンボル情報が利用できなくなっている可能性があります。

suffix

オペレーティングシステムのクラッシュダンプファイルのペアを表す数値の接尾辞を指定します。たとえば、接尾辞が「3」である場合、mdb は「unix.3」や「vmcore.3」などの名前のファイルを調査すると推測します。これらのファイルは存在しないが、「vmdump.3」が存在する場合は、ダンプファイルを圧縮解除するには最初に `savecore -f vmdump.3` を実行する必要があることを示すメッセージが出力されます。同じ名前の実際のファイルが現在のディレクトリに存在する場合、この数字文字列は接尾辞としては解釈されません。

使用法

mdb は、大規模なファイルも認識しつつ、すべての入力ファイル(スクリプト、オブジェクトファイル、コアファイル、raw データファイルなど)を処理します。2G バイト (2^{31} バイト) 以上の大規模なファイルの処理の詳細は、[largefile\(5\)](#) を参照してください。

終了ステータス

次の終了ステータスが返されます。

- 0
デバッグは正常に実行を終了しました。
- 1
致命的なエラーが発生しました。
- 2
無効なコマンド行オプションが指定された。

環境変数

HISTSIZE

この変数は、コマンド履歴リストの最大の長さを決定するために使用されます。この変数が存在しない場合、デフォルトの長さは 128 です。

HOME

この変数は、ユーザーのホームディレクトリ (`.mdbrc` ファイルが存在する場所) のパス名を決定するために使用されます。この変数が存在しない場合、`.mdbrc` は処理されません。

SHELL

この変数は、! メタキャラクタを使用して要求されたシェルエスケープを処理するシェルのパス名を決定するために使用されます。この変数が存在しない場合、`/bin/sh` が使用されます。

ファイル

`$HOME/.mdbrc`

ユーザーの mdb 初期設定ファイル。`.mdbrc` ファイルが存在する場合、そのファイルが処理されるのは、デバッグターゲットが初期化されたあとですが、モジュールの自動読み込みが実行される前、またはコマンドが標準入力から読み込まれる前です。

`/dev/kmem`

カーネルの仮想記憶イメージデバイス。このデバイス固有のファイルは、稼働中のオペレーティングシステムを検査するときにコアファイルとして使用されます。

/dev/ksyms

カーネルのシンボルテーブルデバイス。このデバイス固有のファイルは、稼働中のオペレーティングシステムを検査するときにオブジェクトファイルとして使用されます。

/proc/pid/*

ユーザプロセスを検査および制御するときに読み込まれるプロセス情報ファイル。

/usr/lib/adb**/usr/platform/platform-name/lib/adb**

\$<および\$<<dcmdで読み込まれるマクロファイルのデフォルトディレクトリ。*platform-name*はプラットフォームの名前で、コアファイルまたはクラッシュダンプの情報から、あるいは現在のマシンから、`uname -i`を使用する場合と同じように得られます ([uname\(1\)](#) を参照)。

/usr/lib/mdb**/usr/platform/platform-name/lib/mdb**

::load dcmd を使用して読み込まれるデバッガモジュールのデフォルトディレクトリ。*platform-name*はプラットフォームの名前で、コアファイルまたはクラッシュダンプの情報から、あるいは現在のマシンから、`uname -i`を使用する場合と同じように得られます ([uname\(1\)](#) を参照)。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/debug/mdb
インタフェースの安定性	確実

関連項目

[adb\(1\)](#), [cmdtool\(1\)](#), [gcore\(1\)](#), [proc\(1\)](#), [pgrep\(1\)](#), [ps\(1\)](#), [stty\(1\)](#), [truss\(1\)](#), [uname\(1\)](#), [coreadm\(1M\)](#), [dumpadm\(1M\)](#), [largefile\(5\)](#), [savecore\(1M\)](#), [exec\(2\)](#), [fork\(2\)](#), [_lwp_self\(2\)](#), [pipe\(2\)](#), [vfork\(2\)](#), [dlopen\(3C\)](#), [elf\(3ELF\)](#), [libc_db\(3LIB\)](#), [libkvm\(3LIB\)](#), [libthread\(3LIB\)](#), [signal\(3C\)](#), [signal.h\(3HEAD\)](#), [thr_self\(3C\)](#), [core\(4\)](#), [proc\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [threads\(5\)](#), [ksyms\(7D\)](#), [mem\(7D\)](#)

『リンカーとライブラリ』

『Oracle Solaris Modular Debugger Guide』

警告

エラー回復メカニ
ズムの使用

デバッガとその dmod は同じアドレス空間内で動作するので、dmod にバグがあると、mdb がコアをダンプしたり、誤動作したりする可能性があります。mdb の resume 機能 (前述の「シグナル処理」を参照) はこのような状況に対して、制限付きで回復メカニズムを提供します。しかし、dmod 自身の状態やデバッガの大域的な状態だけでは、mdb は該当する dmod が壊れているかどうかを最終的には判断でき

ません。したがって、`resume` 操作は安全であるとは保証されず、また、その後のデバッグのクラッシュを防ぐこともできません。`resume` によるもっとも安全な対処方法は、重要なデバッグ情報を保存し、デバッグを停止し再起動します。

動作中のオペレーティングシステムのデバッグによる修正

動作中のオペレーティングシステムのアドレス空間をデバッグを使用して修正する(書き込む)ことは非常に危険であり、ユーザーがカーネルデータの構造を破損するとシステムパニックが発生する可能性があります。

注意事項

プロセスコアファイルの調査に関する制限

`mdb` は、Solaris 2.6 より前の Solaris のリリースによって生成されたプロセスコアファイルを検査するためのサポートを提供しません。Solaris 9 のリリースまたはそれ以前のリリースによって生成されたコアファイルをデバッグする場合は、シンボル情報が使用できない可能性があります。これらのコアファイルにはテキストセクションと読み取り専用データが存在しないため、プロセスがコアをダンプした時点でプロセスに存在したデータとシンボル情報が一致しない場合があります。Solaris 9 以降のリリースでは、テキストセクションと読み取り専用データはデフォルトでコアファイルに含まれます。ユーザーは、[coreadm\(1M\)](#) を使用して、その情報をコアファイルから除外するようにプロセスを構成できます。これにより、それらのコアファイルの `mdb` によって出力される情報は、プロセスがコアをダンプした時点で存在したデータと一致しくなくなります。x86 版の Solaris システムから生成されたコアファイルは SPARC 版の Solaris システムでは調査できず、その逆もできません。

クラッシュダンプファイルの調査に関する制限

Solaris 7 以前のリリースで生成されたクラッシュダンプを調査するには、対応するオペレーティングシステムのリリース用の `libkvm` が必要です。あるオペレーティングシステムのリリースで生成されたクラッシュダンプを別のオペレーティングシステムのリリースで `dmod` を使用して調査する場合、カーネルの実装によっては、いくつかの `dcmd` や `walker` が適切に動作しない可能性があります。この状況を検出すると、`mdb` は警告メッセージを発行します。x86 版の Solaris システムから生成されたクラッシュダンプは SPARC 版の Solaris システムでは調査できず、その逆もできません。

32 ビットと 64 ビットのデバッグ間の関係

`mdb` は 32 ビットと 64 ビットの両方のプログラムのデバッグをサポートします。ターゲットのプログラムを調査して、そのデータモデルを決定したあと、`mdb` は必要に応じて自動的に、ターゲットと同じデータモデルを持つ `mdb` バイナリを実行し直します。このアプローチによって、読み込まれたモジュールがプライマリターゲットと同じデータモデルを使用するので、デバッグモジュールを作成するタスクが簡単になります。64 ビットのターゲットプログラムをデバッグできるのは 64 ビットのデバッグだけです。64 ビットのデバッグを使用できるのは 64 ビットのオペレーティング環境が動作しているシステム上だけです。

64 ビットプロセスを `exec` する 32 ビットプロセスをデバッグする場合、またはその逆の場合、デバッグを実行し直す必要があることもあります。このような状況に対応する方法の詳細については、後述の「`exec` との対話」を参照してください。

exec との対話

制御されているプロセスが `exec(2)` を正常に実行するとき、デバッガの動作は `::set -o follow_exec_mode` オプションで制御できます (前述の説明を参照)。デバッガと対象プロセスのデータモデルが同じ場合、`exec` のあとで、`mdb` が自動的にターゲットを継続するか、デバッガのプロンプトに戻るかは、`stop` モードか `follow` モードかによって決定されます。デバッガと対象プロセスのデータモデルが異なる場合、`follow` モードでは、`mdb` は自動的に `mdb` バイナリを適切なデータモデルで再度 `exec` して、プロセスに接続し直すので、`exec` から戻っても、プロセスは停止したままになります。このような再実行後、すべてのデバッガの状態が保存されるわけではありません。

32 ビットの対象プロセスが 64 ビットのプログラムを実行した場合、`stop` モードでは、コマンドプロンプトに戻りますが、この時点で 64 ビットデータモデルを使用しているので、対象プロセスを検査できません。デバッグを再開するには、`::release -a dcmd` を実行して、`mdb` を終了し、そして、`mdb -p pid` を実行して、64 ビットデバッガを対象プロセスに接続し直します。

64 ビットの対象プロセスが 32 ビットのプログラムを実行した場合、`stop` モードでは、コマンドプロンプトに戻りますが、新しいプロセスを検査する機能はかなり制限されます。組み込み `dcmd` はすべてマニュアルどおりに機能しますが、読み込み可能な `dcmd` は構造体のデータモデル変換を実行しないので、マニュアルどおりに機能しません。デバッグ機能を完全に復元するには、前述のように、デバッガを解放し、プロセスに接続し直す必要があります。

ジョブ制御との対話

ジョブ制御によって停止されている (つまり、`SIGTSTP`、`SIGTTIN`、または `SIGTTOU` で停止されている) プロセスにデバッガが接続されている場合、継続用の `dcmd` で継続しようとしても、そのプロセスは再度「動作中 (running)」に設定できません。対象プロセスが同じセッションのメンバーである場合 (つまり、`mdb` と同じ制御端末を共用している場合)、`mdb` は関連するプロセスグループをフォアグラウンドに持ってきて、`SIGCONT` でプロセスを継続し、そのプロセスをジョブ制御の停止から再開しようとしています。このようなプロセスから `mdb` を切断すると、`mdb` はそのプロセスグループをバックグラウンドに戻してから終了します。対象プロセスが同じセッションのメンバーでない場合、`mdb` はそのプロセスを安全にフォアグラウンドに持ってくるできないので、`mdb` はデバッガに関してはそのプロセスを継続しますが、そのプロセスはジョブ制御によって停止されているままになります。この場合、`mdb` は警告を出力するので、ユーザーは適切なシェルから `fg` コマンドを実行して、プロセスを再開する必要があります。

プロセスの接続と解放

`mdb` が動作中のプロセスに接続するとき、プロセスは停止し、継続用の `dcmd` の 1 つが適用されるまで、あるいは、デバッガが終了するまで停止したままになります。`-p` オプションでデバッガをプロセスに接続する前、あるいは、`::attach` または `:A` コマンドを発行する前に、`-o nostop` オプションを有効にしている場合、`mdb` はプロセスに接続しますが、プロセスを停止しません。プロセスが動作している間、(結果の整合性は失われるのにもかかわらず) プロセスは通常どおりに監視でき、ブレークポイントや追跡用のフラグも有効にできます。プロセスが動作している間に `:c` または `::cont dcmd` を実行した場合、デバッガはプロセスが停止するまで

待機します。どの追跡対象ソフトウェアイベントも発生しない場合、:cまたは::contのあと、割り込み文字(通常は^c)を入力すると、プロセスを強制的に停止し、デバッガに制御を戻すことができます。

:R、::release、:r、::run、\$q、または::quit dcmdを実行した場合、あるいは、EOFまたはシグナルなどの結果としてデバッガが終了した場合、mdbは現在動作中のプロセスを(もしあれば)解放します。プロセスが本来デバッガで:rまたは::runを使用して作成されていた場合、そのプロセスは解放されると、SIGKILLが送信された場合と同じように強制的に終了されます。プロセスがmdbに接続される前にすでに動作していた場合、そのプロセスは解放されると、もう一度「動作中(running)」に設定されます。プロセスを解放して停止および放棄したままにするには、::release -a オプションを使用します。

シンボルデバッグ 情報

::list、::offsetof、::print、および::sizeof dcmdでは、1つ以上のロードオブジェクトにmdbでの使用に適した圧縮シンボルデバッグ情報が含まれている必要があります。現時点ではこの情報を利用できるのは、特定のSolarisカーネルモジュールだけです。

開発者向けの情報

『[Oracle Solaris Modular Debugger Guide](#)』には、mdbの機能に関する詳細な説明、およびデバッガモジュールの開発者向け情報が記載されています。

ヘッダーファイル<sys/mdb_modapi.h>にはMDBモジュールAPIにある関数のプロトタイプが含まれており、/source/demo/mdb-examples パッケージにはディレクトリ/usr/demo/mdbにあるサンプルモジュールのソースコードが含まれています。

名前	mesg – permit or deny messages
形式	mesg [-n -y n y]
機能説明	The mesg utility will control whether other users are allowed to send messages via write(1) , talk(1) , or other utilities to a terminal device. The terminal device affected is determined by searching for the first terminal in the sequence of devices associated with standard input, standard output, and standard error, respectively. With no arguments, mesg reports the current state without changing it. Processes with appropriate privileges may be able to send messages to the terminal independent of the current state.
オプション	The following options are supported: -n n Denies permission to other users to send message to the terminal. See write(1). -y y Grants permission to other users to send messages to the terminal.
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of mesg: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.
終了ステータス	The following exit values are returned: 0 if messages are receivable. 1 if messages are not receivable. 2 on error.
ファイル	/dev/tty* terminal devices /dev/pts/* terminal devices
属性	See attributes(5) for descriptions of the following attributes:
関連項目	talk(1) , write(1) , attributes(5) , environ(5) , standards(5)

名前	mkdir - ディレクトリの作成
形式	<code>/usr/bin/mkdir [-m <i>mode</i>] [-p] <i>dir</i>...</code>
機能説明	mkdir コマンドは、777 モード（ファイル生成マスク umask(1) で変更可能）で、指定されたディレクトリを作成します。 ディレクトリ内の標準エントリ（たとえば、そのディレクトリ自身を表す "." ファイルや、その親を表す "."）は自動的に作成されます。mkdir は、これらのエントリに名前を指定することはできません。ディレクトリを作成するためには、親ディレクトリの書き込み権が必要です。 新しいディレクトリのオーナー ID とグループ ID はそれぞれ、プロセスの実効ユーザー ID と実効グループ ID に設定されます。mkdir は mkdir(2) システムコールを呼び出します。
setgid と mkdir	新たに作成されたディレクトリの setgid ビットを変更するには、mkdir を実行後に、<code>chmod g+s</code> または <code>chmod g-s</code> を使用する必要があります。 setgid ビットの値は、親ディレクトリのものが受け継がれます。
オプション	サポートしているオプションは、次のとおりです。 <code>-m <i>mode</i></code> 新しいディレクトリにモードを指定します。モードの選択項目は chmod(1) で表示できます。 <code>-p</code> このオプションを指定すると、mkdir はまず、存在しない親ディレクトリをすべて作成することによって <i>dir</i> を作成します。中間ディレクトリに与えられるモードの値は、777 とファイル生成マスクに設定されたビットの値の差です。ただしこの差は、少なくとも 300（ユーザーの書き込み権と実行権）でなければなりません。
オペランド	次のオペランドを指定できます。 <i>dir</i> 生成されるディレクトリのパス名。
使用法	ファイルが 2G バイト (2^{31} バイト) 以上ある場合の mkdir の動作については、 largefile(5) を参照してください。
使用例	例1 mkdir の使用 次の例を見てください。 <pre>example% mkdir -p ltr/jd/jan</pre> この例では、ltr/jd/jan というサブディレクトリ構造を作成します。
環境変数	mkdir の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 指定されたディレクトリはすべて正常に作成された、または(-p オプション指定時) 指定された各ディレクトリは存在していたかもしくは正常に作成されました。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [chmod\(1\)](#), [rm\(1\)](#), [sh\(1\)](#), [umask\(1\)](#), [Intro\(2\)](#), [mkdir\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前 mkmsgs - gettext でアクセスできるメッセージファイルの作成

形式 mkmsgs [-o] [-i locale] inputstrings msgfile

機能説明 mkmsgs ユーティリティーは、テキスト検索ツール ([gettext\(1\)](#)、[srchtxt\(1\)](#)、[exstr\(1\)](#)、および [gettext\(3C\)](#) を参照) を使用してアクセスできる、テキスト文字列のファイルを作成するために使用されます。入力の対象は、特定のロケール ([setlocale\(3C\)](#) を参照) 用のテキスト文字列のファイルです。出力先は、[gettext\(1\)](#) と [gettext\(3C\)](#) の両方で検索できる形式のテキスト文字列のファイルです。-i オプションを使用すると、作成されたファイルを /usr/lib/locale/locale/LC_MESSAGES ディレクトリにインストールできます (locale は、テキスト文字列が記述されている言語に対応する)。

inputstrings 引数には、もとなるテキスト文字列を含んだファイル名を指定します。*msgfile* は、mkmsgs が、[gettext\(1\)](#) や [gettext\(3C\)](#) で読み取り可能な形式の文字列を書き込む出力ファイルの名前です。この *msgfile* 名は、長さが 14 文字以内であること、\0 (NULL) を含んでいないこと、スラッシュ (/) またはコロン (:) に該当する ASCII コードを含んでいないことが必要です。

入力ファイルの内容は、特定のロケールに対応したテキスト文字列がいくつか集まったものです。文字列と文字列とは復帰改行文字で区切られます。表示不可能な文字は、エスケープシーケンスとしてアルファベットで表す必要があります。メッセージは *inputstrings* から順番に読み込まれ、変換され、*msgfile* に書き込まれます。空のメッセージを *msgfile* 中に生成する場合は、*inputstrings* 中の適切な位置に空の行を入れておいてください。

文字列の内容を変更する場合は、単に *inputstrings* ファイルを編集してください。新たな文字列は、ファイルの最後尾にだけ追加できます。追加を行ったら、新たに *msgfile* を作成して正しい場所に登録しておく必要があります。この手順を誤ると、検索機能使用時に誤った文字列が検索されて、ソフトウェアの互換性が損なわれます。

オプション サポートしているオプションは、次のとおりです。

-o *msgfile* がすでに存在していれば上書きします。

-i locale *msgfile* を /usr/lib/locale/locale/LC_MESSAGES ディレクトリにインストールします。このディレクトリ中でファイルを生成したり上書きしたりできるのは、スーパーユーザー、およびグループ bin のメンバーだけです。/usr/lib/locale の下のディレクトリは、存在していなければ生成されます。

使用例 例1 mkmsgs コマンドの使用例

C.str という名の入力メッセージソースファイルの例です。

```
File %s:\t cannot be opened\n
%s: Bad directory\n
.
```

例1 mkmsgs コマンドの使用例 (続き)

```
.  
.  
write error\  
.  
.
```

例2 ファイルにテキスト文字列を作成するために C.str から入力文字列を使用する例
次のコマンドを実行すると、c.str から文字列が読み取られ、適切な形式の文字列
に変換され、現在のディレクトリ中の ux というファイルに書き込まれます。

```
example% mkmsgs C.str UX
```

例3 ファイルにテキスト文字列を作成するために FR.str から入力文字列を使用する例
次のコマンドは、FR.str からテキスト文字列を読み取り、適切な形式に変換
し、/usr/lib/locale/fr/LC_MESSAGES というディレクトリ中の ux というファイルに書き
込みます。

```
example% mkmsgs -i fr FR.str UX
```

これらのテキスト文字列は、環境変数 LC_MESSAGES=fr が設定されていて、「機能説
明」節の冒頭に一覧表示されているテキスト検索ツールのいずれかが呼び出された
場合にアクセスされます。

ファイル
属性

/usr/lib/locale/locale/LC_MESSAGES/* mkmsgs が生成するメッセージファイル
属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/locale

関連項目

[exstr\(1\)](#), [gettext\(1\)](#), [srchtxt\(1\)](#), [gettext\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#)

名前 mkstr – create an error message file by massaging C source files

形式 /usr/ucb/mkstr [-] *messagefile prefix filename...*

機能説明

The `mkstr` utility creates files of error messages. You can use `mkstr` to make programs with large numbers of error diagnostics much smaller, and to reduce system overhead in running the program — as the error messages do not have to be constantly swapped in and out.

`mkstr` processes each of the specified *filenames*, placing a massaged version of the input file in a file with a name consisting of the specified *prefix* and the original source file name. A typical example of using `mkstr` would be:

```
mkstr pistrings processed *.c
```

This command would cause all the error messages from the C source files in the current directory to be placed in the file `pistrings` and processed copies of the source for these files to be placed in files whose names are prefixed with *processed*.

To process the error messages in the source to the message file, `mkstr` keys on the string `'error('` in the input stream. Each time it occurs, the C string starting at the `'` is placed in the message file followed by a null character and a NEWLINE character; the null character terminates the message so it can be easily used when retrieved, the NEWLINE character makes it possible to sensibly `cat` the error message file to see its contents. The massaged copy of the input file then contains a `lseek` pointer into the file which can be used to retrieve the message, that is:

```
char efilename[ ] = "/usr/lib/pi_strings";
int efil = -1;

error(a1, a2, a3, a4)
{
    char
    buf[256];
    if (efil < 0) {

        efil = open(efilename, 0);
        if (efil < 0) {
oops:
            perror (efilename);
            exit (1);
        }
    }
    if (lseek(efil, (long) a1, 0) || read(efil, buf, 256) <= 0)
        goto oops;
    printf(buf, a2, a3, a4);
}
```

オプション – Place error messages at the end of the specified message file for recompiling part of a large mkstred program.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [xstr\(1\)](#), [attributes\(5\)](#)

名前	mktemp – make temporary filename
形式	<code>mktemp [-dtqu] [--directory] [--quiet] [--dry-run] [-p <i>directory</i>] [--suffix=<i>suff</i>] [--tmpdir[=<i>dir</i>]] [<i>template</i>]</code>
機能説明	The <code>mktemp</code> utility makes a temporary filename. To do this, <code>mktemp</code> takes the specified filename template and overwrites a portion of it to create a unique filename. See OPERANDS. The template is converted to a path name using the <code>mktemp(3C)</code>, <code>mkdtemp(3C)</code>, <code>mkstemp(3C)</code>, and <code>mkstemp(3C)</code> library functions. If <code>mktemp</code> can successfully generate a unique filename, the file (or directory) is created with file permissions such that it is only readable and writable by its owner (unless the <code>-u</code> flag is given) and the filename is printed to standard output. <code>mktemp</code> allows shell scripts to safely use temporary files. Traditionally, many shell scripts take the name of the program with the PID as a suffix and used that as a temporary filename. This kind of naming scheme is predictable and the race condition it creates is easy for an attacker to win. A safer, though still inferior approach is to make a temporary directory using the same naming scheme. While this guarantees that a temporary file is not subverted, it still allows a simple denial of service attack. Use <code>mktemp</code> instead.
オプション	The following options are supported: <code>-d, --directory</code> Make a directory instead of a file. <code>-p <i>directory</i></code> Use the specified directory as a prefix when generating the temporary filename. The directory is overridden by the user's <code>TMPDIR</code> environment variable if it is set. This option implies the <code>-t</code> flag. <code>-q, --quiet</code> Fail silently if an error occurs. This is useful if a script does not want error output to go to standard error. <code>--suffix=<i>suff</i></code> Append <i>suff</i> to the template. <i>suff</i> must not contain a slash (/). This option is implied when the template ends in characters other than replaceable X characters, and does not remove the requirement to include such X characters. <code>--tmpdir[=<i>dir</i>]</code> 'template' is relative to <i>dir</i>. If <i>dir</i> is not specified, the user's environment variable <code>TMPDIR</code> is used if set, else <code>/tmp</code> is used. This option does not support a template with an absolute name and unlike with the use of <code>-t</code>, a template may contain slashes. <code>-t</code> Generate a path rooted in a temporary directory. This directory is chosen as follows: If the user's <code>TMPDIR</code> environment variable is set, the directory contained therein is used.

Otherwise, if the `-p` flag was given the specified directory is used. If none of the above apply, `/tmp` is used. In this mode, the template (if specified) should be a directory component (as opposed to a full path) and thus should not contain any forward slashes.

`-u, --dry-run`

Operate in unsafe mode. The temp file is unlinked before `mktemp` exits. This is slightly better than `mktemp(3C)`, but still introduces a race condition. Use of this option is discouraged.

オペランド

The following operands are supported:

template *template* can be any filename with three to six Xs included in it, for example `/tmp/tfile.XXXXXX`.

If *template* is not specified, a default of `tmp.XXXXXX` is used and the `-t` flag is implied.

If *template* has multiple series of Xs, the final series is used for the replacement text, unless `--suffix` specifies otherwise.

If *template* has characters after the final set of Xs, then the `--suffix` option is implied to be all such characters, unless explicitly specified.

使用例

例 1 Using mktemp

The following example illustrates a simple use of `mktemp` in a `sh(1)` script. In this example, the script quits if it cannot get a safe temporary file.

```
TMPFILE='mktemp /tmp/example.XXXXXX'
if [ -z "$TMPFILE" ]; then exit 1; fi
echo "program output" >> $TMPFILE
```

例 2 Using mktemp to Support TMPDIR

The following example uses `mktemp` to support for a user's `TMPDIR` environment variable:

```
TMPFILE='mktemp -t example.XXXXXX'
if [ -z "$TMPFILE" ]; then exit 1; fi
echo "program output" >> $TMPFILE
```

例 3 Using mktemp Without Specifying the Name of the Temporary File

The following example uses `mktemp` without specifying the name of the temporary file. In this case the `-t` flag is implied.

```
TMPFILE='mktemp'
if [ -z "$TMPFILE" ]; then exit 1; fi
echo "program output" >> $TMPFILE
```

例 4 Using mktemp with a Default Temporary Directory Other than /tmp

The following example creates the temporary file in /extra/tmp unless the user's TMPDIR environment variable specifies otherwise:

```
TMPFILE='mktemp -p /extra/tmp example.XXXXX'
if [ -z "$TMPFILE" ]; then exit 1; fi
echo "program output" >> $TMPFILE
```

例 5 Using mktemp to Remove a File

The following example attempts to create two temporary files. If creation of the second temporary file fails, mktemp removes the first file before exiting:

```
TMP1='mktemp -t example.1.XXXXXX'
if [ -z "$TMP1" ]; then exit 1; fi
TMP2='mktemp -t example.2.XXXXXX'
if [ -z "$TMP2" ]; then
    rm -f $TMP1
    exit 1
fi
```

例 6 Using mktemp

The following example does not exit if mktemp is unable to create the file. That part of the script has been protected.

```
TMPFILE='mktemp -q -t example.XXXXXX'
if [ ! -z "$TMPFILE" ]
then
    # Safe to use $TMPFILE in this block
    echo data > $TMPFILE
    ...
    rm -f $TMPFILE
fi
```

例 7 Using mktemp with Suffix Option

The following command illustrates the use of the suffix option. The effect of this command is to create the temporary file ex.q5N.SUFF.

```
# mktemp --suffix=.SUFF ex.XXXXXX
ex.q5Ngid.SUFF
```

例 8 Using Suffix and Tmpdir Options

The following command illustrates the use of the suffix and tmpdir options.

```
# mktemp --tmpdir=$HOME --suffix=.bar foo.XXXXXX
/root/foo.7Za0_N.bar
```


例 9 Using Directory and Suffix Options

The following command uses both the directory and suffix options.

```
# mktemp --directory --suffix=.bar foo.XXXXXX
foo.GSa03d.bar
# ls -l
drwx----- 2 root      staff      512 Mar 19 2012 foo.GSa0.bar
```

例 10 Supporting a Template with Non-Trailing Xs

The following command shows the use of the directory option with non-trailing X characters. In this command, the `--suffix=suff` option is implied, where `bar` is used as the suffix.

```
# mktemp XXfooXXXXXXbar
XXfoaFY0N6bar
```

例 11 Using the Quiet and Tmpdir Options

The following command illustrates the use of the quiet and tmpdir options.

```
# mktemp --quiet --tmpdir=/tmp foo
[No diagnostic message is returned]
```

例 12 Using mktemp with Multiple Options

The following command combines the use of the dry-run, tmpdir, and suffix options.

```
# mktemp --dry-run --tmpdir=$HOME --suffix=SUFF
/root/tmp.qdaGc0SUFF
# ls -l /root/tmp.qdaGc0SUFF
/root/tmp.qdaGc0SUFF: No such file or directory
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `mktemp` with the `-t` option: `TMPDIR`.

TMPDIR Name a directory used for creating temporary files to override system default; used by `mktemp`.

終了ステータス

The following exit values are returned:

- 0 Successful completion.
- 1 An error occurred.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed

関連項目 [sh\(1\)](#), [mkdtemp\(3C\)](#), [mkstemp\(3C\)](#), [mkstemp\(3C\)](#), [mktemp\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 The `mktemp` utility appeared in OpenBSD 2.1. The Solaris implementation uses only as many 'Xs' as are significant for [mktemp\(3C\)](#), [mkstemp\(3C\)](#), and [mkstemp\(3C\)](#).

名前	moe - 最適なパス名展開のマニフェストの生成
形式	moe [-c] [-32 -64] [-s -v] <i>path</i>
機能説明	moe ユーティリティは、実行時リンカーの予約トークンを含む最適なパス名展開のマニフェストを生成します。これらのトークンを使用すると、動的なオブジェクト内の依存関係、フィルタ対象、および実行パスを定義できます。これらのトークンを実行時に展開することにより、そのマシン上で最適に実行されるオブジェクトと検索パスを柔軟に選択できます。ld.so.1(1) を参照してください。 たとえば、\$HWCAP トークンを使用すると、フィルタと依存関係を表すことができます。このトークンを実行時に解釈すると、オブジェクトファミリーを得ることができます。これらのオブジェクトを分析することにより、プロセスで読み込む場合のオブジェクトの適用レベルが決定されます。オブジェクトは、各オブジェクトの実行時に必要なハードウェア機能に基づいてソートされます。moe は、現在のプラットフォーム上で実行するのに最適なオブジェクトの名前を返します。 moe は、指定されたパスを RTLD_FIRST フラグとともに dlopen(3C) に渡すことによってパス名を分析します。したがって、予約トークンの展開は、実行中のプロセスで展開が発生したときに ld.so.1 によって実行されます。dlopen() を呼び出すことで複数のオブジェクトを分析できますが、RTLD_FIRST フラグを指定すると最適なオブジェクトだけが処理されます。 デフォルトでは、moe は指定した <i>path</i> を 2 回分析します。1 回目の分析では、32 ビットオブジェクトが検索されます。2 回目の分析では、64 ビットオブジェクトが検索されます (該当する場合)。通常、32 ビットオブジェクトと 64 ビットオブジェクトは別々のディレクトリに隔離されています。これらのディレクトリには、多くの場合、ディレクトリに格納されているオブジェクトのクラスを反映した名前が付けられています。moe の複数回の受け渡しにより、32 ビットオブジェクトと 64 ビットオブジェクトが同じディレクトリに含まれるインスタンスが取得されます。複数の受け渡しにより、指定したパス名ではディレクトリに含まれる可能性のあるオブジェクトのクラスがわからない場合にも柔軟に対応できます。 実行時リンカーによって実行される予約トークンの展開の詳細については、『リンカーとライブラリ』を参照してください。
オプション	サポートしているオプションは、次のとおりです。 -32 32 ビットオブジェクトのみを分析します。 -64 64 ビットオブジェクトのみを分析します。 -c 各パス名の前にオブジェクトのクラスを付加します。 -s サイレント。最適な名前やエラー診断が表示されません。エラーの戻り値のみが使用できます。このオプションは、-32 オプションまたは -64 オプションと一緒に指定したときのみ有効です。-s オプションは、-v オプションと同時に使用することはできません。

-v 冗長。最適な展開名を決定できない場合に、エラー診断を標準エラーに書き込みます。-v オプションは、-s オプションと同時に使用することはできません。

オペランド 次のオペランドを指定できます。

path 展開するパス名。

使用例 次の例では、moe を使用して /usr/lib/libc ディレクトリ内のオブジェクトの最適な展開を表示しています。このディレクトリには、さまざまなハードウェア機能を使用するために作成された Intel オブジェクトファミリーが含まれています。

```
% moe '/usr/lib/libc/$HWCAP'
/usr/lib/libc/libc_hwcap.so.1
```

-c オプションを使用すると、最適なオブジェクトのクラスを明確にできます。

```
% moe -c '/usr/lib/libc/$HWCAP'
32-bit: /usr/lib/libc/libc_hwcap.so.1
```

次の例では、moe を使用して /opt/ISV/cpu ディレクトリ階層下のオブジェクトの最適な展開を表示しています。これらのディレクトリには、さまざまなプラットフォーム用に作成された SPARC オブジェクトファミリーが含まれています。

```
% moe -c -64 '/opt/ISV/$ISALIST/isa.so.1'
64-bit: /opt/ISV/sparcv9/isa.so.1
```

-v を使用すると、最適な名前が返されないインスタンスを診断できます。直前のパス名を 32 ビットオブジェクトとして検査すると、次のような診断が生成されます。

```
% moe -c -v -32 '/opt/ISV/$ISALIST/isa.so.1'
32-bit: /opt/ISV/sparcv9/isa.so.1: wrong ELF class: ELFCLASS64
```

終了ステータス -32 オプションまたは -64 オプションを有効にした場合、最適な展開に成功すると 0 が返され、失敗すると 0 以外の値が返されます。-32 オプションまたは -64 オプションを有効にしなかった場合、戻り値は常に 0 です。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/linker
インタフェースの安定性	確実

関連項目 [ld.so.1\(1\)](#), [optisa\(1\)](#), [isalist\(1\)](#), [dlmopen\(3C\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

名前	more, page – テキストファイルの表示またはページング
形式	<pre> /usr/bin/more [-cdfllrsuw] [-lines] [+ <i>linenumber</i>] [+/<i> pattern</i>] [<i>file</i>]... /usr/bin/page [-cdfllrsuw] [-lines] [+ <i>linenumber</i>] [+/<i> pattern</i>] [<i>file</i>]... /usr/xpg4/bin/more [-cdeisu] [-n <i>number</i>] [-p <i>command</i>] [-t <i>tagstring</i>] [<i>file</i>]... /usr/xpg4/bin/more [-cdeisu] [-n <i>number</i>] [+ <i>command</i>] [-t <i>tagstring</i>] [<i>file</i>]...</pre>
機能説明	more ユーティリティーは、端末の画面上にテキストファイルの内容を 1 画面ずつ表示するフィルタです。通常、画面がいっぱいになると休止します。そのとき /usr/bin/more は --More-- を、 /usr/xpg4/bin/more は <i>file</i> を、いずれも画面の最下段に表示します。more がパイプではなくファイルから読み取る場合、それまでに表示された文字のパーセンテージもあわせて表示します。 more ユーティリティーは、RETURN 文字が入力されると、もう 1 行スクロールアップします。SPACE 文字が入力されると、次の画面を表示します。その他のコマンドは、次で説明します。 page ユーティリティーは、テキストの次の画面を表示する前に画面をクリアします。page は、more とは異なり、前画面の最後の行を現画面の最初に重複して表示することはありません。 more ユーティリティーは、端末を NOECHO モードに設定し、連続して出力できるようにします。通常、入力するコマンドは端末上に表示されません。ただし、/ コマンドと ! コマンドの場合を除きます。 /usr/bin/more ユーティリティーは、指定された最後のファイルを表示して終了します。/usr/xpg4/bin/more は、指定された最後のファイルの最終行で、コマンド入力を要求するプロンプトを発行します。 標準出力が端末の場合、more は cat(1) とまったく同じように動作しますが、複数のファイルが指定された場合、ファイルの前にヘッダーが表示されます。
オプション	/usr/bin/more と /usr/xpg4/bin/more の両方で次のオプションが使用できます。 -c 表示前に画面をクリアします。より高速に表示するため、画面をスクロールせずに描画し直します。端末が行の終端までクリアする機能を持たない場合、このオプションは無視されます。 -d 認識できないコマンドが使用された場合、端末ベルを鳴らさずにエラーメッセージを表示します。これは経験の少ないユーザーに役立ちます。 -s 圧縮。複数の空行を単一の空行と置換します。これは、nroff(1) を画面表示させる場合に役立ちます。
/usr/bin/more	次のオプションは /usr/bin/more でのみ指定できます。

- f 長い行を折り返しません。**nroff(1)**出力を**ul(1)**によりパイプ連結したとき生成されるような非表示文字、またはエスケープシーケンスが行にあるときに役立ちます。
- l **FORMFEED** 文字 (Control-I) を改ページと見なしません。**-l**を使用しない場合、**more** は休止して、**^L** 文字 (Control-I) の入った行のあとのコマンドを受け入れます。また、ファイルが **FORMFEED** で始まっている場合、ファイルを表示する前に画面をクリアします。
- r 通常、**more** は解釈できない制御文字を無視します。**-r** オプションにより、これらの文字は **^C** として表示されます。ここで、**C** はこのような任意の制御文字を表します。
- u 下線引きエスケープシーケンスを生成しません。通常、**more** は、**nroff(1)** が生成するような下線引きを、端末に適切な方法で処理します。端末が下線引きを実行できるか、あるいは強調モードを備えている場合、**more** はテキストファイルで呼び出されたものに対応するエスケープシーケンスを提供します。
- w 通常、**more** はその出力の端末まで来ると終了します。しかし、**-w** を用いると、**more** は終了する前にプロンプトを出し、キーが押されるのを待ちます。
- lines デフォルト (端末画面内の行の数から 2 を引いた数) の代わりに、指定された数の *lines* を行数分画面に表示します。
- +linenumber *linenumber* から表示します。
- +/*pattern* *pattern* という正規表現の入った行より上の 2 行から表示します。注: エディタとは異なり、この形式は **^** で終了してはいけません。スラッシュで終了すると、その後のスラッシュは検索パターンの文字と見なされます。

/usr/xpg4/bin/more

次のオプションは /usr/xpg4/bin/more でのみ指定できます。

- e 引数リストの最後のファイルの最終行を出力すると、ただちに終了します。
- i 検索時に、大文字と小文字を区別せずにパターンマッチングを行います。
- n *number* 1 画面当たりの行数を指定します。*number* 引数の値は正の整数です。**-n** オプションの値は、環境から得られる値に優先して用いられます。
- p *command*
- +*command* 検査対象の各ファイルに対し、最初に *command* 引数中の **more** コマンドを実行します。それが行番号や正規表現検索など位置決め用のコ

マンドであれば、コマンドの最終結果を表すように現在の位置を設定します。ファイルの中間行は書き出しません。例として次の2つのコマンドを見てください。

```
more -p 1000j file
more -p 1000G file
```

これらは、現在の位置を行番号 1000 として表示を開始する、という同じ動作をします。しかも、j がファイルの検査中に呼び出されていれば書き出して画面から消したであろうと思われる行は飛ばします。位置決めコマンドが正常終了でなければ、ファイルの先頭行が現在の位置となります。

- t *tagstring* *tagstring* 引数で指定したタグを持つファイルの内容を 1 画面分表示します。詳細については [ctags\(1\)](#) ユーティリティの説明を参照してください。
- u バックスペース文字を印刷可能文字として扱い、^H (Control-h) と表示します。このとき、ある種の端末では下線付きまたは強調モードテキストで出力するような特殊な処理や、下線を付ける処理は行いません。さらにこのオプションが指定されると、行の終わりのキャリッジリターンを無視しません。

-t *tagstring* と -p *command* (または旧式の +*command*) オプションの両方が指定された場合、-t *tagstring* が先に処理されます。

使用法

環境 `more` は、端末の [terminfo\(4\)](#) エントリを使用して、そのディスプレイ特性を判別します。

`more` は、`MORE` という環境変数に設定済みオプションがないか調べます。たとえば、デフォルトで -c モードを使用してファイルをページングするときは、この環境変数の値を -c に設定します(通常、この環境変数を設定するコマンドシーケンスは `.login` または `.profile` ファイルに格納されます)。

コマンド コマンドはただちに有効になります。コマンドが *file*、*command*、*tagstring*、*pattern* のいずれかを必要としないかぎり、キャリッジリターンを入力する必要はありません。コマンド文字自体を指定しないかぎり、行抹消文字を入力すれば、入力中の数値引数を取り消すことができます。さらに、消去文字を入力すれば、'-More-- (xx%)' または *file* メッセージを再表示できます。

次のコマンドにおいて、*i* は数値引数です(デフォルトでは 1)。

`iSPACE` 次の画面を表示します。*i* を指定すると、*i* 行追加表示します。

`iRETURN` 次の行を表示します。*i* が指定されていればその行数分を表示します

<i>ib</i>	
<i>i^B</i>	(Control-b) 画面を <i>i</i> 個、逆にスキップして次の画面を表示します。
<i>id</i>	
<i>i^D</i>	(Control-d) 画面を半画面分または <i>i</i> 行分、順方向にスクロールします。 <i>i</i> が指定されていれば、その値が以降の <i>d</i> および <i>u</i> コマンド用のデフォルトとなります。
<i>if</i>	画面を <i>i</i> 個スキップして次の画面を表示します。
<i>h</i>	ヘルプ。more のすべてのコマンドの説明を表示します。
<i>^L</i>	(Control-l) 画面の再表示。
<i>in</i>	直前に入力した <i>pattern</i> の <i>i</i> 番目の一致を検索します。
<i>q</i>	
<i>Q</i>	more を終了します。
<i>is</i>	<i>i</i> 行スキップしてから 1 画面分を表示します。
<i>v</i>	現在のファイルの現在行で、vi エディタに入ります。
<i>iz</i>	SPACE と同じ。ただし <i>i</i> を指定した場合、その値が画面当たりの行数の新しいデフォルト値になります。
<i>=</i>	現在の行番号を表示します。
<i>i/pattern</i>	<i>pattern</i> という正規表現の <i>i</i> 番目の一致を順方向に検索します。 <i>pattern</i> という正規表現の <i>i</i> 番目の一致またはパイプの終端、どちらか先に見つかった方を含む行の前 2 行から画面を表示します。more がファイルを表示中であり、一致がない場合、そのファイルにおける位置は変更されません。正規表現は、消去文字と抹消文字を使用して編集できます。第 1 カラムを越えて消去すると、検索コマンドが取り消されます。
<i>!command</i>	シェルを起動し、 <i>command</i> を実行します。% および ! という文字を <i>command</i> 内で使用すると、それぞれ、現在のファイル名および直前のシェルコマンドに置換されます。現在のファイル名がない場合、% は展開されません。これらの文字の前にバックスラッシュを追加して、展開をエスケープしてください。
<i>:f</i>	現在のファイル名と行番号を表示します。
<i>i:n</i>	コマンド行に指定された <i>i</i> 番目後のファイル名、または <i>i</i> が範囲外の場合はリスト内の最後のファイル名までスキップします。
<i>i:p</i>	コマンド行に指定された <i>i</i> 番目前のファイル名、または <i>i</i> が範囲外の場合はリスト内の最初のファイル名までスキップします。ファイル内に

`more` を位置決めしている間に指定すると、ファイルの最初に移動します。`more` がパイプから読み取り中の場合、`more` は単に端末ベルを鳴らすだけです。

:q
:Q `more` を終了します (q または Q と同じ)。

/usr/bin/more 次のコマンドは /usr/bin/more でのみ使用できます。

’ 単一引用符。直前の検索が開始された点に移動します。現在のファイルで検索を実行していない場合、ファイルの最初に移動します。

. ドット。直前のコマンドを繰り返します。

^ \ テキストの一部表示を停止します。`more` は出力の送信を停止し、通常の --More-- プロンプトを表示します。一部の出力は結果的に失われることがあります。

/usr/xpg4/bin/more 次のコマンドは /usr/xpg4/bin/more でのみ使用できます。

i^F (Control-f) 画面を *i* 個スキップして次の 1 画面分を表示します (if と同じ)。

^G (Control-g) 現在の行番号を表示します (= と同じ)。

ig ファイル中の行番号 *i* に進みます。デフォルトは先頭行です。

iG ファイル中の行番号 *i* に進みます。デフォルトは最終行です。

ij 次の行を表示します。*i* が指定されていればその行数分を表示します (iRETURN と同じ)。

ik 画面を逆方向にスクロールします。*i* 指定時はその行数、省略時は 1 行です。

mletter *letter* で示す名前で現在の位置をマークします。

N 逆方向に検索します。

r 画面を再表示します。

R 画面を再表示し、バッファー内に入力があればそれを破棄します。

iu

i^U (Control-u) 画面を逆方向にスクロールします。*i* 指定時はその行数、省略時は半画面分です。*i* が指定されていれば、その値が以降の d および u コマンド用のデフォルトとなります。

ZZ `more` を終了します (q と同じ)。

:e *file* 新たなファイルを検査 (表示) します。*file* を省略すると、現在のファイルが再表示されます。

	<code>:t tagstring</code>	<code>tagstring</code> 引数が示すタグの位置へ進み、タグを含む行が現在の位置となるように画面をスクロールします。詳しくは <code>ctags</code> ユーティリティの説明を参照してください。
	<code>'letter</code>	以前に <code>letter</code> という名を付けてマークした位置に戻ります。
	<code>''</code>	最新の 1 画面分を超える移動を行なったときの元の位置に戻ります。デフォルトはファイルの先頭です。
	<code>i?[!]pattern</code>	ファイルを逆方向に検索し、 <code>pattern</code> を含んでいる <code>i</code> 番目の行を見つけます。 <code>!</code> は、 <code>pattern</code> を含んでいない <code>i</code> 番目の行の検索です。
	<code>i/!pattern</code>	ファイルを順方向に検索し、 <code>pattern</code> を含んでいない <code>i</code> 番目の行を見つけます。
	<code>![command]</code>	シェルまたは指定したコマンドを呼び出します。
大規模ファイルの動作		ファイルが 2G バイト (2^{31} バイト) 以上ある場合の <code>more</code> と <code>page</code> の動作については、 largefile(5) を参照してください。
環境変数		<code>more</code> の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_COLLATE (/usr/xpg4/bin/more のみ)、LC_CTYPE、LC_MESSAGES、NLSPATH、および TERM。
/usr/xpg4/bin/more		次の環境変数も /usr/xpg4/bin/more の実行に影響を与えます。
	COLUMNS	画面の水平方向のサイズとして、システムが選択した値の代わりに用いる値を指定します。
	EDITOR	エディタを選択する際に <code>v</code> コマンドを使用します。
	LINES	画面の垂直方向のサイズとして、システムが選択した値の代わりに用いる値を指定します。画面当たりの行数を決める際、 <code>-n</code> オプションの値が <code>LINES</code> の値に優先します。
	MORE	前述の「オプション」節で説明した、オプションを指定する文字列です。コマンド行に記述する場合と同様に、オプションとオプションの間は空白文字で区切り、個々のオプションは <code>-</code> で始まらなければなりません。 <code>MORE</code> で指定したオプションのあとで、コマンド行のオプションが処理されます。つまり、コマンド行が次に示すように記述されていると見なされます。 <code>more \$MORE options operands</code>
終了ステータス		次の終了ステータスが返されます。
	<code>0</code>	正常終了。
	<code>>0</code>	エラーが発生した。
ファイル	/usr/lib/more.help	/usr/bin/more と /usr/bin/page のためだけのヘルプファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/more /usr/bin/page	属性タイプ	属性値
	使用条件	system/core-os
	CSI	未対応

/usr/xpg4/bin/more	属性タイプ	属性値
	使用条件	system/xopen/xcu4
	CSI	有効
	インタフェースの安定性	确实
	標準	standards(5) を参照してください。

関連項目 [cat\(1\)](#), [csh\(1\)](#), [ctags\(1\)](#), [man\(1\)](#), [nroff\(1\)](#), [script\(1\)](#), [sh\(1\)](#), [ul\(1\)](#), [terminfo\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

/usr/bin/more [regcomp\(3C\)](#)
/usr/bin/page
/usr/xpg4/bin/more [regex\(5\)](#)

注意事項

/usr/bin/more 逆方向のスキップを大規模なファイルに対して行うと遅くなります。

/usr/xpg4/bin/more 端末が正しく設定されていないと、このユーティリティーも正しく動作しません。

名前	mp - テキストからプリンタ記述言語 (PDL) へのプリティープリントフィルタ
形式	mp [-A4] [-C] [-D <i>target_printer_name</i>] [-F] [-L <i>localename</i>] [-P <i>target_spool_printer</i>] [-PS] [-US] [-a] [-
機能説明	mp プログラムは、-D または -P オプションを指定せずに呼び出された場合は、各 <i>filename</i> 順に読み取ってその内容の清書バージョンを PostScript™ フォーマットで作成し、標準出力に送ります。ファイル名引数が指定されていない場合は、mp は標準入力を読み取ります。端末からの標準入力の場合、EOF シグナル (通常では Ctrl-D) で入力を終了します。 -D および -P オプションを指定するには、引数として出力先プリンタ名が必要です。このプリンタ名を指定すると、出力先プリンタのプリンタ記述言語 (PDL) が生成されます。-D オプションを指定すると、PDL は標準出力に出力されます。-P オプションを指定すると、PDL はプリンタ用に直接スプールされます。このどちらのオプションも指定されていない場合は、mp はデフォルトの PostScript の出力を生成します。 mp プログラムは、Solaris でサポートされているさまざまなロケールのテキストを受け入れ、指定されたロケールに応じて出力を生成します。また、適切なテキストレイアウトで出力します。たとえば、mp は双方向テキストレンダリングや CTL (Complex Text Layout)などをサポートします。 mp の入力フォーマットとして、メール、ニュース記事、通常の ASCII ファイル、メールフォルダ全体、ダイジェストのすべてを利用できます。出力フォーマットの各ページの上部と下部には、バナー情報を含む、グレーのひし形、またはひし形と同じ寸法の輪郭が含まれます。
オプション	サポートしているオプションは、次のとおりです。 -a ニュース記事としてファイルをフォーマットします。上部のバナーには "Article from newsgroup" というテキストが書かれています。newsgroup は、"Newsgroups:" 行で最初に見つかった news グループになります。 -A4 A4 用紙サイズ (8.26 x 11.69 インチ) を使用します。 -c <i>chars</i> ユーザーの /etc/passwd エントリの gecost フィールドから抽出できる最大文字数です。デフォルトは 18 です。 -C "\nFrom" を使用して新しいメールメッセージの開始を示すのではなく、mp が Content-Length: メールヘッダーの値を探してそれを使用します。Content-Length で次の "\nFrom" に行かない場合は、その値は正しくありません。mp はメールフォルダ内で後退して次の "\nFrom" を探します。 -d ダイジェストとしてファイルをフォーマットします。 -D <i>target_printer_name</i> 出力先プリンタ用の PDL を生成し、X 印刷サーバーへの接続を要求します。<i>target_printer_name</i>

は、`printer_name@machine[:display_number]` または簡単に `printer_name` で指定します。最初の書式の場合、`mp` は出力先プリンタを `printer_name` として、X 印刷サーバーのディスプレイ `machine[:display_number]` に接続しようとしています。

- e ELM メールが中間ファイルフォーマットにフロントエンドすると仮定します。ELM 内部からのメッセージの印刷 ("p" コマンドを使う)、特にタグ付きメッセージの印刷の場合に使用されます。このオプションは、ELM オプションの設定で指定する必要があります。
- ff Filofax のパーソナルオーガナイザによって使用されるファイルをフォーマットします。
- fp Franklin Planner のパーソナルオーガナイザによって使用されるファイルをフォーマットします。
- F メールの上部のヘッダーに、メールの送信先ではなく、送信元が印刷されます。自分専用のプリンタを持っているユーザー向けのオプションです。
- l 横モードで出力します。紙 1 枚にテキストの 2 ページ分が印刷されます。
- ll 横モードで出力します。紙 1 枚にテキストの 1 ページ分が印刷されます。このオプションは、1 行が通常より長いファイルを印刷する場合に便利です。
- L *localename* 印刷するファイルのロケールを指定します。このオプションを省略すると、`mp` は `MP_LANG` 環境変数のロケールを使用します。`MP_LANG` 環境変数も設定されていない場合は、`LANG` 環境変数を使用します。`LANG` 環境変数も設定されていない場合は、`mp` は現在動作中のロケールを特定しようと試みます。特定できない場合、`mp` は C ロケールで動作中であると見なします。
- m メールフォルダとしてファイルをフォーマットし、複数のメッセージを印刷します。
- M 対象ロケール用の `prolog.ps` が存在する場合でも、`mp` が `mp.conf` ファイルを使用して印刷します。このオプションは、英語版 PostScript™ プリンタで日本語を印刷する場合に便利です。
- n グレーの線と、その前後の情報を、ヘッダーとフッターから取り除きます。このオプションは、`lp filename` コマンドのような出力にする場合に使用します。

- o 通常の ASCII ファイルとしてファイルをフォーマットします。
- p *prologue* 指定した *prologue* ファイルを PostScript/Xprt prologue ファイルとして使用します。それ以前に定義したファイル名は無効になります。この *prologue* ファイルによって、印刷されるファイルの形式が指定されます。PostScript では、*prologue* ファイルの拡張子は .ps です。Xprt クライアント (-D オプションが指定されている場合) では、このファイルの拡張子は .xpr です。これらのファイルの定義については、後述の「提供される prologue ファイル」を参照してください。
- P *target_spool_printer* 出力先プリンタ用に PDL をスプールします。標準出力に送られる出力はありません。X 印刷サーバーへの接続を要求します。*target_spool_printer* は、*printer_name@machine[:display_number]* または簡単に *printer_name* で指定します。最初の書式の場合、mp は出力先プリンタを *printer_name* として、ディスプレイ *machine[:display_number]* に接続しようとします。
- PS メールやダイジェストメッセージが PostScript をメッセージのテキストとしてだけ持っている場合、通常は印刷されないでそのまま通過してしまいがちですが、このオプションを指定すると、PostScript もテキストとして印刷されます。
- s *subject* *subject* を、印刷する新しいサブジェクトとして使用します。コマンド行で指定した通常の ASCII ファイルを印刷する場合は、サブジェクトはデフォルトでその ASCII ファイル名となります。
- tm Time Manager のパーソナルオーガナイザで使用するファイルをフォーマットします。
- ts Time/System International のパーソナルオーガナイザで使用するファイルをフォーマットします。
- US 用紙サイズ US (8.5 x 11 インチ) を使用します。これは、デフォルトの用紙サイズです。
- u *config_file_path* デフォルトの構成ファイル `/usr/lib/lp/locale/locale_name/mp/mp.conf` の代わりに使用するファイルを指定します。ファイル名は絶対パスで指定してください。
- v このリリースの mp のバージョン番号を表示します。

<code>-w words</code>	ユーザーの <code>/etc/passwd</code> エントリの <code>gecos</code> フィールドから抽出できる最大文字数です。デフォルトは3です。
<code>-z point_size</code>	<code>point_size</code> に指定したサイズでテキストを印刷します。デフォルトのサイズは、縦モードでは12ポイントで、横モードでは9ポイントです。
<code>-?</code>	<code>mp</code> の使用方法を表示します。 <code>cs(1)</code> を使用している場合は <code>? 文字</code> をエスケープする必要があります。

オペランド 次のオペランドを指定できます。

`filename` 読み取るファイルの名前。

使用例 `mp` 印刷フィルタは、ユーザーのマシンにインストールされているすべてのロケールでファイルを印刷するのに使用できます。

例1 日本語のテキストファイルの印刷

`euc` コードセットでエンコードされた日本語テキストファイルは、次のように入力すると、日本語以外のすべての PostScript プリンタで印刷できます。

```
example% mp -L ja_JP.eucJP -M ja_JP_eucJP.txt | lp
```

ここで、`-L` オプションがロケールを指定し、`-M` オプションがデフォルトの `prolog.ps` ファイルの代わりに `mp.conf` 構成ファイルと呼び出します。`ja_JP.eucJP` ロケールの場合、`/usr/lib/lp/locale/ja_JP.eucJP/mp/mp.conf` および `/usr/openwin/lib/locale/ja_JP.eucJP/print/prolog.ps` ファイルの両方が存在します。そのため、`-M` オプションはデフォルトの `prolog.ps` ファイルの優先権をオーバーライドする働きがあります。構成ファイルとして `mp.conf` を使用すると、すべての PostScript プリンタに印刷できます。

`-L` オプションで指定されたロケールのエンコーディング、および印刷されるテキストファイルのエンコーディングは、同一である必要があります。前述の日本語のファイルの例では、テキストファイルが Shift-JIS でエンコードされている場合、ロケール `ja_JP.PCK` は SJIS でエンコードされているので、次のコマンドを使用してください。

```
example% mp -L ja_JP.PCK -M SJIS.txt | lp
```

例2 Xprt モードでの実行

X 印刷サーバーのデーモン (`/usr/openwin/bin/Xprt`) がネットワークのいずれかのシステムで実行されている場合、`mp` は次のようにして起動できます。Xprt でサポートされている任意のページ記述言語 (PDL) で出力が可能です (`display_number` のデフォルト値は 2100 です)。

```
example% setenv XPSERVERLIST "machine1[:display_number1] \
machine2[:display_number2] machine3[:display_number3]"
```

例2 Xprt モードでの実行 (続き)

または

```
example% setenv XPDISPLAY machine_name[:display_number]
```

オプション `-D printer_name[@machine[:display_number]]` または `-P printer_name[@machine[:display_number]]` を指定すると、最優先権が与えられ、mp は `printer_name` で、`machine[:display_number]` 上で実行する Xprt に接続しようとしてします。これらのオプションを指定しない場合、デフォルトの `display_number` 値は 2100 です。これが失敗すると、`printer_name` は次のロジックから入手する Xprt ディスプレイで再試行します。コマンド行で `-D printer_name` または `-P printer_name` だけを入力すると、次のことも有効になります。

mp は XPSERVERLIST で、スペースで区切られた Xprt サーバーのリストを確認します。`printer_name` 属性をサポートする 1 つのサーバーが見つかるまで続きます。1 つも見つからない場合、mp は XPDISPLAY 環境変数を確認します。これは `machine[:display_number]` という書式です。これも設定されていない、または無効である場合、mp はデフォルトのディスプレイ :2100 に接続しようとしてします。これも成功しない場合、mp はエラーメッセージを表示して終了します。

XPSERVERLIST または XPDISPLAY が設定されている場合、データを出力先プリンタにパイプするには、次のように入力します。

```
example% mp -D printer_name -L ja_JP.eucJP \
-M ja_JP_eucJP.txt | lp -d printer_name
```

Xprt クライアントモードで動作していて直接スプールするには、`-P` オプションを使用します。

```
example% mp -P printer_name -L ja_JP.eucJP -M ja_JP_eucJP.txt
```

例3 ヘッダーおよびフッターを無効にする

mp のヘッダーおよびフッターを無効にするには、`-n` オプションを使用します。

```
example% mp -n mytext.txt | lp
```

例4 長いテキスト行を印刷する

横モードで 80 カラムより長いテキストファイルを印刷するには、`-ll` オプションを使用します。

```
example% mp -ll mytext.txt | lp
```

例5 印刷用ポイントサイズを指定する

任意のポイントサイズを指定するには、`-z` オプションを使用します。次では、20 ポイントを指定します。

例5 印刷用ポイントサイズを指定する (続き)

```
example% mp -z 20 mytext.txt | lp
```

環境変数

XPSEVERLIST -D または -P の引数は、*printer_name@machine[:display_number]* という書式で、XPSEVERLIST は *machine[:display_number]* が *printer_name* をサポートしない場合のみ使用されます。

XPSEVERLIST には、プリンタを接続する Xprt がスペースで区切られたリストが含まれています。mp は指定されたプリンタをサポートできる Xprt サーバーが見つかるまでリストを順番に調べます。mp が接続するディスプレイを最初に見つけたところで、インスタンスが終了します。XPSEVERLIST が設定されていない場合、代わりに環境変数 XPDISPLAY が使用されます。

XPDISPLAY -D または -P オプションが指定され、かつ、環境内で *printer_name* 引数だけが指定され、XPSEVERLIST 変数が設定されていない場合は、クライアントに接続するために X 印刷サーバーを実行する *machine[:display_number]* を決定するのに XPSEVERLIST 変数が使用されます。XPDISPLAY も設定されていない場合は、印刷サーバー起動スクリプトが、クライアントを実行しているマシンのポート 2100 で Xprt サーバーを起動します。ジョブが終わると、スクリプトは印刷サーバーを終了します。XPDISPLAY が設定されている場合、mp クライアントは XPDISPLAY で実行中の印刷サーバーと通信しようとしています。この場合、印刷サーバーが稼働していなくても、起動しようとはしません。

MP_PROLOGUE ページフォーマットファイル(.xpr または .ps)を保存するディレクトリを決定します。これらのファイルは、ページ装飾、各物理ページの論理ページ番号、横フォーマットか縦フォーマットかななどを決定します。MP_PROLOGUE がない場合には、デフォルトディレクトリは、`/usr/lib/lp/locale/C/mp` になります。

MP_LANG
LANG

-D または -P オプションのどちらも指定されていない場合、prologue ファイルが印刷される出力の先頭に付加されます。prologue ファイルとは、`/usr/openwin/lib/locale/localename/print/prolog.ps` または `/usr/lib/lp/locale/localename/mp/prolog.ps` を指します。*localename* は、環境変数 MP_LANG または LANG の値です。この両方の変数が存在する場合は、下位互換性を保つために `/usr/openwin/lib/locale/localename/print/prolog.ps` ファイルが優先されます。どちらかのファイルが存在せず、かつ、-D オプションが指定されていない場合は、印刷に必要なプロローグ情報の代わりに、該当ロケールの構成ファイル `/usr/lib/lp/locale/localename/mp/mp.conf` が構成情報として使用

されます。prolog.ps が存在する場合、下位互換性を保つために mp.conf は使用されなくなります。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

1 エラーが発生した。

提供される **prologue** ファイル

次の **prologue** ファイルが提供されています。拡張子 **.ps** を持つファイルは PostScript 出力用のもので、拡張子 **.xpr** を持つファイルは印刷サーバークライアント用のものです。**.xpr** ファイルは 300dpi のプリンタ用に生成されますが、ほかの解像度にも適応可能です。

mp.common.ps このディレクトリ中のその他すべての .ps ファイル用の共通 prologue ファイル。

mp.pro.ps
mp.pro.xpr デフォルトで使用されます。

mp.pro.ff.ps
mp.pro.ff.xpr -ff オプションが有効なときに使用されます。

mp.pro.fp.ps
mp.pro.fp.xpr -fp オプションが有効なときに使用されます。

mp.pro.tm.ps
mp.pro.tm.xpr -tm オプションが有効なときに使用されます。

mp.pro.ts.ps
mp.pro.ts.xpr -ts オプションが有効なときに使用されます。

mp.pro.alt.ps	
mp.pro.alt.xpr	デフォルトの <code>prologue</code> ファイルの代わりに使用可能なファイル。下部バナーの右隅にページ番号を出力します。

mp.pro.l.ps	
mp.pro.l.xpr	横フォーマット用の prologue ファイル

mp.pro.ll.ps	
mp.pro.ll.xpr	1行が通常より長いファイルを印刷する場合、横 フォーマット用の prologue ファイル

mp.pro.altl.ps	
mp.pro.altl.xpr	前述の代わりに使用可能な横フォーマット用の prologue ファイル

ファイル `.cshrc`
`csh(1)` の初期化ファイル

```
.mailrc
    mail(1) の初期化ファイル

/usr/bin/mp
    実行可能ファイル

/usr/lib/lp/locale/C/mp/mp.conf
    デフォルトの構成ファイル

/usr/lib/lp/locale/C/mp/mp.common.ps
    このディレクトリ中のその他すべての .ps ファイル用の共通 prologue ファイル。
    .xpr ファイル用ではありません。

/usr/lib/lp/locale/C/mp/mp.pro.ps
/usr/lib/lp/locale/C/mp/mp.pro.xpr
    メール印刷用のデフォルトの PostScript prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.l.ps
/usr/lib/lp/locale/C/mp/mp.pro.l.xpr
    横フォーマット用のデフォルトの prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.ll.ps
/usr/lib/lp/locale/C/mp/mp.pro.ll.xpr
    紙 1 枚にテキストの 1 ページ分を印刷する、横フォーマット用のデフォルトの
    prologue ファイル。1 行が通常より長いファイルを印刷する場合に便利です。

/usr/lib/lp/locale/C/mp/mp.pro.altl.ps
/usr/lib/lp/locale/C/mp/mp.pro.altl.xpr
    横フォーマット用の代替の prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.alt.ps
/usr/lib/lp/locale/C/mp/mp.pro.alt.xpr
    代替のデフォルトの prologue ファイル。各ページの右下にページ番号を挿入。

/usr/lib/lp/locale/C/mp/mp.pro.ff.ps
/usr/lib/lp/locale/C/mp/mp.pro.ff.xpr
    Filofax フォーマット用のデフォルトの prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.fp.ps
/usr/lib/lp/locale/C/mp/mp.pro.fp.xpr
    Franklin Planner フォーマット用のデフォルトの prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.tm.ps
/usr/lib/lp/locale/C/mp/mp.pro.tm.xpr
    Time Manager フォーマット用のデフォルトの prologue ファイル

/usr/lib/lp/locale/C/mp/mp.pro.ts.ps
/usr/lib/lp/locale/C/mp/mp.pro.ts.xpr
    Time/System International フォーマット用のデフォルトの prologue ファイル
```

`/usr/openwin/lib/locale/localename/print/prolog.ps`
`/usr/lib/lp/locale/localename/mp/prolog.ps`
mp.conf の代わりに使用されるデフォルトのロケール固有 prologue ファイル (詳細は環境変数の説明を参照)。

mp.conf と .xpr ファイルの構造およびフォーマットについては、『[国際化対応言語環境の利用ガイド](#)』に記述されています。プリンタ常駐フォントなどほかのフォントを使用する必要がある場合、あるいは出力形式を変更する場合には、この文書を参照してください。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	print/mp

関連項目 [csh\(1\)](#), [mail\(1\)](#), [mailtool\(1\)](#), [attributes\(5\)](#)

『[国際化対応言語環境の利用ガイド](#)』

名前	mpss.so.1 – 優先ページサイズを設定するための共有オブジェクト		
形式	mpss.so.1		
機能説明	mpss.so.1 共有オブジェクトは、起動されたプロセスとその子孫のスタック、ヒープ、あるいはその両方の優先ページサイズを選択的に構成する手段を提供します。mpss.so.1 を有効にするには、次の文字列が 1 つ以上の MPSS (Multiple Page Size Support) 環境変数とともに環境内に存在する必要があります (ld.so.1(1) を参照)。 LD_PRELOAD=\$LD_PRELOAD:mpss.so.1		
環境変数	mpss.so.1 共有オブジェクトは、事前に読み込まれると次の環境変数を読み取り、その環境変数に固有のものである可能性がある優先ページサイズ要件とプロセスを判定します。 MPSSHEAP=<i>size</i> MPSSSTACK=<i>size</i>	MPSSHEAP および MPSSSTACK には、それぞれヒープおよびスタックの優先ページサイズを指定します。指定したページサイズは、作成されたすべてのプロセスに適用されます。 <i>size</i> は、サポートされるページサイズ (pagesize(1) を参照) または 0 である必要があります。後者の場合、システムは適切なページサイズを選択します (memcntl(2) を参照)。 <i>size</i> は、K (K バイト)、M (M バイト)、G (G バイト)、または T (T バイト) で修飾できます。	MPSSCFGFILE=<i>config-file</i> <i>config-file</i> は、次の形式の mpss 構成エントリを 1 つ以上含むテキストファイルです。 <i>exec-name exec-args:heap-size:stack-size</i> <i>exec-name</i> には、アプリケーションまたは実行可能ファイルの名前を指定します。対応する優先ページサイズは、ファイル内にある最初の <i>exec-name</i> に一致する新しく作成されたプロセス (getexecname(3C) を参照) のために設定されます。 <i>exec-name</i> には、フルパス名、基本名、またはパターン文字列を指定できます。パターンマッチングについては、sh(1) の「ファイル名の生成」を参照してください。 <i>exec-args</i> は、引数と照合する任意指定のパターン文字列です。<i>exec-args</i> が指定されていないか、または <i>exec-args</i> が <i>exec-name</i> の引数内に存在する場合にのみ、優先ページサイズは設定されます。

heap-size、*stack-size*、あるいはその両方を指定しないと、対応する優先ページサイズは設定されません。

MPSSCFGFILE は、MPSSHEAP および MPSSSTACK よりも優先されます。MPSSCFGFILE が設定されていない場合は、`/etc/mpss.conf` から優先ページサイズが取得されます (存在している場合)。

MPSSERRFILE=*pathname* デフォルトでは、エラーメッセージは `syslog(3C)` の LOG_ERR レベルと LOG_USER 機能を使用してログに記録されます。MPSSERRFILE に有効な *pathname* (`/dev/stderr` など) が含まれている場合は、代わりにエラーメッセージがそこに記録されます。

使用例

例1 MPSSCFGFILE を使用した優先ページサイズの構成

次の Bourne シェルコマンド (`sh(1)` を参照) は、MPSSCFGFILE 環境変数を使用して、優先ページサイズを `foo` で始まる実行可能ファイル名を持つ選択された一連のアプリケーションに構成します。MPSS 構成ファイル `mpsscfg` は、`vi(1)` などのテキストエディタを使用して以前に作成されていると見なされます。`cat(1)` コマンドは、内容をダンプしているだけです。

```
example$ LD_PRELOAD=$LD_PRELOAD:mpss.so.1
example$ MPSSCFGFILE=mpsscfg
example$ export LD_PRELOAD MPSSCFGFILE
example$ cat $MPSSCFGFILE
foo*:512K:64K
```

アプリケーションが起動されたあと、`pmap (proc(1))` を使用して、構成された実際のページサイズを表示できます。

```
example$ foobar &
example$ pmap -s 'pgrep foobar'
```

(`pmap` の出力に表示された) ページサイズが希望のサイズに構成されていない場合は、MPSS 構成ファイルまたは環境変数のエラーが原因である可能性があります。エラーログ (デフォルトでは `/var/adm/messages`) でエラーを確認してください。

エラーが見つからない場合は、リソースまたは境界合わせの制約が原因である可能性があります。「注意事項」のセクションを参照してください。

例2 MPSSHEAP と MPSSSTACK を使用した優先ページサイズの構成

次の Bourne シェルコマンドは、MPSSHEAP および MPSSSTACK 環境変数を使用して、すべてのアプリケーションに対して 512K のヒープの優先ページサイズと 64K のスタックの優先ページサイズを構成しています。

例2 MPSSHEAPとMPSSSTACKを使用した優先ページサイズの構成 (続き)

```
example$ LD_PRELOAD=$LD_PRELOAD:mpss.so.1
example$ MPSSHEAP=512K
example$ MPSSSTACK=64K
example$ export LD_PRELOAD MPSSHEAP MPSSSTACK
```

例3 優先順位ルール(例2の続き)

MPSSCFGFILE による優先ページサイズの構成は、MPSSHEAP およびMPSSSTACK よりも優先されます。例2のコマンドのあとに次のコマンドを追加すると、構成ファイル内のアプリケーション、ls コマンド、および引数として ora1 が指定された ora で始まるすべてのアプリケーションを除き、すべてのアプリケーションで 512K のヒープの優先ページサイズと 64K のスタックの優先ページサイズが構成されます。

```
example$ MPSSCFGFILE=mpsscfg2
example$ export MPSSCFGFILE
example$ cat $MPSSCFGFILE
ls::
ora* ora1:4m:4m
```

- ファイル /usr/lib/ld/map.bssalign bss 整列用のテンプレートリンカーマップファイル (mapfile)(「注意事項」を参照)。
/etc/mpss.conf 構成ファイル
- 属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/extended-system-utilities
インタフェースの安定性	確実

- 関連項目 [cat\(1\)](#), [ld\(1\)](#), [ld.so.1\(1\)](#), [pagesize\(1\)](#), [ppgsz\(1\)](#), [proc\(1\)](#), [sh\(1\)](#), [vi\(1\)](#), [exec\(2\)](#), [fork\(2\)](#), [memcntl\(2\)](#), [getexecname\(3C\)](#), [getpagesize\(3C\)](#), [syslog\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)
- 注意事項 ヒープおよびスタックの優先ページサイズは継承されます。子プロセスには親と同じ優先ページサイズが適用されます。[exec\(2\)](#) では、優先ページサイズが mpss 共有オブジェクトを使用して構成されていないかぎり、優先ページサイズがデフォルトのシステムページサイズに戻されます。

また、[ppgsz\(1\)](#) (proc ツール) を使用して、スタックまたはヒープ、あるいはその両方の優先ページサイズを設定することもできます。ただし、名前の一致に基づいて子孫のページサイズを選択して構成することはできません。

[ppgsz\(1\)](#) の「注意事項」も参照してください。

名前 msgcc – C language message catalog compiler

形式 msgcc [-M-*option*] [cc-options*option*] *file*...

機能説明 msgcc is a C language message catalog compiler. It accepts cc style options and arguments.

A [msgcpp\(1\)](#) .mso file is generated for each input .c file. If the -c option is not specified then a [gencat\(1\)](#) format .msg file is generated from the input .mso and .msg files. If -c is not specified then a .msg suffix is appended to the -o file if it doesn't already have a suffix. The default output is a.out.msg if -c and -o are not specified.

If -M-new is not specified then messages are merged with those in the pre-existing -o file.

オプション The following options are supported:

- cc-options* Specify cc style options and arguments.
- M-option* Set a msgcc option.
- Specify option as one of the following:
- | | |
|------------------------|---|
| mkmsgs | The -o file is assumed to be in mkmsgs(1) format. |
| new | Create a new -o file. |
| preserve | Messages in the -o file that are not in new .msg file arguments are preserved. The default is to either reuse the message numbers with new message text that is similar to the old or to delete the message text, leaving an unused message number. |
| set= <i>number</i> | Set the message set number to <i>number</i> . The default is 1. |
| similar= <i>number</i> | The message text similarity message threshold. The similarity measure between old and new message text is:
$100 * (2 * \text{gzip}(\text{old} + \text{new}) \backslash \\ / (\text{gzip}(\text{old}) + \text{gzip}(\text{new})) - 1)$ where $\text{gzip}(x)$ is the size of text x when compressed by gzip. The default threshold is <code>__similar__</code> . A threshold of 0 turns off message replacement, but unused old messages are still deleted. Use -M-preserve to preserve all old messages. |
| verbose | Trace similar message replacements on the standard error. |

オペランド The following operands are supported:

file Specifies the name of the file on which msgcc operates.

終了ステータス 0 Successful completion.
 >0 An error occurred.

使用例

例1 Using msgcc

The following example uses msgcc to extract localizable strings from the file `hello.c`, marked using `ERROR_dictionary()`, writes them to the file `hello.mso`, and creates a gencat format `xxx.msg` file:

```
example% cat hello.c

#include <stdio.h>
#include <stdlib.h>

/*
 * dummy macro to avoid including
 * libast headers
 */
#define ERROR_dictionary(x) x

int main(int ac, char *av[])
{
    puts( ERROR_dictionary("hello world") );
    return( EXIT_SUCCESS );
}

example% msgcc -o xxx -D__STDC__ -D__i386 hello.c

example% cat hello.mso
str "hello world"

example% cat xxx.msg
$ xxx message catalog
$translation msgcc 2007-09-25
$set 1
$quote "
1 "hello world"
```

Authors

Glenn Fowler, gsf@research.att.com

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/astdev
Interface Stability	Volatile

関連項目 [cpp\(1\)](#), [gencat\(1\)](#), [mkmsgs\(1\)](#), [msggen\(1\)](#), [msgcpp\(1\)](#), [msgcvt\(1\)](#), [attributes\(5\)](#)

名前	msgcpp – C language message catalog preprocessor
形式	msgcpp [-ACEHMPVX] [-D <i>name</i> [= <i>value</i>]] [-I <i>directory</i>] [-U <i>name</i>] [-T[<i>length</i>]] [-Y <i>directory</i>] [<i>input</i> [<i>output</i>]]
機能説明	msgcpp is a C language message catalog preprocessor. It accepts cpp(1) style options and arguments. msgcpp preprocesses an input C source file and emits keyed lines to the output, usually for further processing by msgcc(1). msgcc output is in the gencat(1) syntax. Candidate message text is determined by arguments to the last <code><error.h></code> and <code><option.h></code> functions. The msgcpp keyed output lines are: <i>cmd command</i> <i>command</i> is a candidate for --??keys option string generation. This is triggered by <code>b_command(int argc, in the input.</code> <i>def name string</i> <i>name</i> is a candidate variable with <i>string</i> value string. <i>str string</i> <i>string</i> should be entered into the catalog. <i>var name</i> If <i>def name</i> occurs then its string value should be entered into the catalog. The input source file is preprocessed with the <code>pp:allpossible</code> option on. This enables non-C semantics. All source should first be compiled error-free with a real compiler before running msgcpp. The following changes are enabled for the top level files. Included file behavior is not affected. 1. All <code>#if</code>, <code>#ifdef</code> and <code>#ifndef</code> branches are enabled. 2. The first definition for a macro is retained, even when subsequent <code>#define</code> statements would normally redefine the macro. <code>#undef</code> must be used to redefine a macro. 3. Macro calls with an improper number of arguments are silently ignored. 4. <code>#include</code> on non-existent headers are silently ignored. 5. Invalid C source characters are silently ignored. <code>msgcat.h</code> is included if it exists. This file may contain macro definitions for functions that translate string arguments. If <code>foo</code> is a function that translates its string arguments then include the line <code>#define foo _TRANSLATE_</code> in <code>msgcat.h</code>, or specify the option <code>-Dfoo=_TRANSLATE_</code>. If <code>bar</code> is a function that translates string arguments if the first argument is <code>stderr</code>, then use either <code>#define bar _STDIO_</code> or <code>-Dbar=_STDIO_</code>. The macro <code>_BLD_msgcat</code> is defined to be 1. As an alternative to <code>msgcat.h</code>, <code>_TRANSLATE_</code> definitions could be placed inside <code>#ifdef _BLD_msgcat ... #endif</code>.
オプション	The following options are supported: -A --assert=<i>assertion</i> Enter the assertion using <code>#assert</code> for system V compatibility. -C --comments Pass comments to the output.

Comments are omitted by default.

-D

--define=*name*[=*value*]

Define the macro *name* to have *value*. This is the only portable way to pass options through cc to [cpp\(1\)](#).

- If *=value* is omitted, *value* is assumed to be 1.
- If *name* begins with :, then it is interpreted as a libpp #pragma pp: statement.
- If *name* begins with %, it is interpreted as a libpp # directive statement.
- If *name* begins with a - or a +, it is interpreted as a libpp option.

- turns the option on, + turns it off.

- Most options have a #pragma counterpart that is listed with the option definition.

-D-C

pp:compatibility

Preprocess for K&R C compatibility.

-D-Dlevel

pp:debug level *level*

Set the debug trace level.

Specify *level* as a number greater than or equal to 0. Higher levels produce more output. Levels higher than 3 can only be enabled in the -g compiled versions.

-D-Fname

Set the main input file name to *name*. This only affects the error messages and the line sync output.

-D-H

pp:hosted

All directories are hosted. Compatibility warning messages from the hosted directory headers are suppressed.

-D-I

pp:cdir

All directories contain C headers. This option is only used only with -D-+.

-D-K

pp:keyargs

- Enable the non-standard *name=value* macro argument mode.
- D-L[id]**
pp:lineid [id]
Set the line sync directive id to *id*. If *id* is not specified, set to null.
- D-M**
pp:nomultiple
Disable multiple include detection.
- D-P**
pp:passthrough
Enable the non-standard passthrough mode. This can be useful for processing non-C input.
- D-Q**
pp:dump
Dump macro definitions to the output so that the output may be passed through cpp again. This is used for generating precompiled headers.
- D-R**
pp:transition
Enable the transition preprocessing mode. This is used for compilers that cannot make up their semantics between K&R and ISO C.
- D-S**
pp:strict
Enable strict preprocessing semantics and warnings. This works with any mode (compatibility, transition, or the default ISO).
- D-Ttest**
pp:test test
Enable implementation specific test code according to *test*.
- D-W**
pp:warn
Enable pedantic warnings in non-hosted files.
- D-X[cc]**
Preprocess for the *cc* compiler, which must be an executable path or an executable on \$PATH.
- D-Z**
pp:pool
Enable pool mode.

-D-d	List canonicalized <code>#define</code> statements for non-predefined macros in the output.
-D-m	List canonicalized <code>#define</code> statements for all macros. All other output is disabled.
-D-+	
pp:plusplus	Preprocess for the C++ dialect.
-E	
--preprocess	Ignored; for compatibility with very old compilers.
-H	
--include-reference	Emit <code>#include</code> file paths on the standard error, one per line, indented to show nesting.
-I	
--include[= <i>directory</i>]	Append <i>directory</i> to the list of directories searched for <code>#include</code> files.
	If <i>directory</i> is <code>-</code> :
	1. <code>-I</code> directories before <code>-I-</code> are searched only for <code>"..."</code> include files 2. <code>-I</code> directories after <code>-I-</code> are searched for <code>"..."</code> and <code><"..."></code> include files 3. the <code>directory .</code> is searched only if it is explicitly specified by an <code>-I</code> option
-I-C <i>directory</i>	
pp:cdir <i>directory</i>	Mark <i>directory</i> as a C header directory. This option is used with <code>pp:plusplus</code> .
-I-D[<i>file</i>]	Read the default probe definitions from <i>file</i> , or ignore the default definitions if <i>file</i> is omitted.
-I-H <i>directory</i>	
pp:hostdir <i>directory</i>	Mark <i>directory</i> as a hosted directory. Headers from hosted directories have compatibility warnings disabled.
-I-I <i>header</i>	
pp:ignore <i>header</i>	Add <i>header</i> to the list of ignored headers.

	-I -Mfile <i>file</i> contains a sequence of header [= "map"] lines, where header is either <name> or "name", and "map" is an explicit binding for header. header is ignored if "map" is omitted. -I -Rfile Include <i>file</i> but do not emit text or line syncs. -I -Sdirectory Add <i>directory</i> to the default standard include directory list. -I -Tfile Include <i>file</i> and emit text to the output file. The option value can be omitted.
-M	
--dependencies	Generate make(1S) dependencies. This option is not needed with nmake. The -M option can be followed by optional flags to change the dependency output styles. The following optional flags are supported: D Generate dependencies in a separate .d file. Preprocessed output is still written to output, or the standard output if output is omitted. G Also generate missing dependencies. M Only generate local header dependencies. Hosted headers are omitted. Hosted headers are determined by the -I -H option and the --pp:hosted and pp:hostdir pragmas. No special distinction is made between the "" and <> include styles.
-P	
--sync	Emit line syncs. Line sync is turned on by default. -P means --nosync.
-T[length]	If not gcc, truncate identifiers to <i>length</i> characters for compatibility with old AT&T compilers.
-U	
--undefine=name	Remove the definition for the macro <i>name</i>.
-V	
--version	Emit the libpp version.

-X
--argmode Enable *name=value* macro arguments for ease1 compatibility.
-Y
--standard=*directory* Add *directory* to the list searched for #include <...> files.

オペランド

The following operands are supported:

input Specifies C source file to preprocess.

output Specifies output file.

終了ステータス

0 Successful completion.

>0 An error occurred.

使用例

例1 Using msgcpp to Extract Localizable Strings

The following example uses msgcpp to extract localizable strings from the file `hello.c`, marked using the `ERROR_dictionary()`, and writes them to the file `hello.mso`:

```
example% cat hello.c
```

```
#include <stdio.h>
#include <stdlib.h>
```

```
/*
 * dummy macro to avoid including
 * libast headers
 */
#define ERROR_dictionary(x) x

int main(int ac, char *av[])
{
    puts( ERROR_dictionary("hello world") );
    puts( ERROR_dictionary("hello all") );
    return( EXIT_SUCCESS );
}
```

```
example% msgcpp -D__STDC__ -D__i386 hello.c hello.mso
```

```
example% cat hello.mso
str "hello world"
str "hello all"
```

Authors

Glenn Fowler, gsf@research.att.com

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/astdev
Interface Stability	Volatile

関連項目

[cpp\(1\)](#), [gencat\(1\)](#), [msgcc\(1\)](#), [msgcvt\(1\)](#), [msggen\(1\)](#), [make\(1S\)](#), [attributes\(5\)](#)

Kernighan, Brian W. and Ritchie, Dennis M., The C Programming Language, Prentice Hall, 1988.

名前	msgfmt – create a message object from a message file																						
形式	<pre>msgfmt [-D <i>dir</i> --directory=<i>dir</i>] [-f --use-fuzzy] [-g] [-o <i>output-file</i> --output-file=<i>output-file</i>] [-s] [--strict] [-v] [--verbose] <i>filename.po...</i></pre>																						
機能説明	The msgfmt utility creates message object files from portable object files (<i>filename.po</i>), without changing the portable object files. The .po file contains messages displayed to users by system commands or by application programs. .po files can be edited. The messages in these files can be rewritten in any language supported by the system. The xgettext(1) command can be used to create .po files from script or programs. msgfmt interprets data as characters according to the current setting of the LC_CTYPE locale category or according to the codeset specified in the .po file.																						
オプション	The following options are supported: <table> <tr> <td>-D <i>dir</i></td><td></td></tr> <tr> <td>--directory=<i>dir</i></td><td>Adds <i>dir</i> to the list for input files search.</td></tr> <tr> <td>-f</td><td></td></tr> <tr> <td>--use-fuzzy</td><td>Uses fuzzy entries in output. If this option is not specified, fuzzy entries are not included into the output. These options are ignored if Solaris message catalogs are processed.</td></tr> <tr> <td>-g</td><td>Directs the utility to generate the GNU-compatible message catalog file. This option cannot be specified with the -s option.</td></tr> <tr> <td>-o <i>output-file</i></td><td></td></tr> <tr> <td>--output=<i>output-file</i></td><td>Specifies the output file name as <i>output-file</i>. All domain directives and duplicate msgids in the .po file are ignored.</td></tr> <tr> <td>-s</td><td>Directs the utility to generate the Solaris message catalog file. This option cannot be specified with the -g option.</td></tr> <tr> <td>--strict</td><td>Directs the utility to append the suffix .mo to the generating message object file name if it doesn't have this suffix. This option is ignored if Solaris message catalogs are processed.</td></tr> <tr> <td>-v</td><td></td></tr> <tr> <td>--verbose</td><td>Verbose. Lists duplicate message identifiers if Solaris message catalog files are processed. Message strings are not redefined.</td></tr> </table> If GNU-compatible message files are processed, this option detects and diagnoses input file anomalies which might represent translation errors. The msgid and msgstr strings are studied and	-D <i>dir</i>		--directory= <i>dir</i>	Adds <i>dir</i> to the list for input files search.	-f		--use-fuzzy	Uses fuzzy entries in output. If this option is not specified, fuzzy entries are not included into the output. These options are ignored if Solaris message catalogs are processed.	-g	Directs the utility to generate the GNU-compatible message catalog file. This option cannot be specified with the -s option.	-o <i>output-file</i>		--output= <i>output-file</i>	Specifies the output file name as <i>output-file</i> . All domain directives and duplicate msgids in the .po file are ignored.	-s	Directs the utility to generate the Solaris message catalog file. This option cannot be specified with the -g option.	--strict	Directs the utility to append the suffix .mo to the generating message object file name if it doesn't have this suffix. This option is ignored if Solaris message catalogs are processed.	-v		--verbose	Verbose. Lists duplicate message identifiers if Solaris message catalog files are processed. Message strings are not redefined.
-D <i>dir</i>																							
--directory= <i>dir</i>	Adds <i>dir</i> to the list for input files search.																						
-f																							
--use-fuzzy	Uses fuzzy entries in output. If this option is not specified, fuzzy entries are not included into the output. These options are ignored if Solaris message catalogs are processed.																						
-g	Directs the utility to generate the GNU-compatible message catalog file. This option cannot be specified with the -s option.																						
-o <i>output-file</i>																							
--output= <i>output-file</i>	Specifies the output file name as <i>output-file</i> . All domain directives and duplicate msgids in the .po file are ignored.																						
-s	Directs the utility to generate the Solaris message catalog file. This option cannot be specified with the -g option.																						
--strict	Directs the utility to append the suffix .mo to the generating message object file name if it doesn't have this suffix. This option is ignored if Solaris message catalogs are processed.																						
-v																							
--verbose	Verbose. Lists duplicate message identifiers if Solaris message catalog files are processed. Message strings are not redefined.																						

compared. It is considered abnormal if one string starts or ends with a newline while the other does not. Also, if the string represents a format string used in a printf-like function, both strings should have the same number of % format specifiers, with matching types. If the flag `c` - format appears in the special comment '#' for this entry, a check is performed.

使用法

The format of portable object files (.po files) is defined as follows. Each .po file contains one or more lines, with each line containing either a comment or a statement. Comments start the line with a pound sign (#) and end with the newline character. All comments (except special comments described later) and empty lines are ignored. The format of a statement is:

directive *value*

Each *directive* starts at the beginning of the line and is separated from *value* by white space (such as one or more space or tab characters). *value* consists of one or more quoted strings separated by white space. Use any of the following types of directives for the Solaris message file:

```
domain domainname
msgid message_identifier
msgstr message_string
```

For a GNU-compatible message file, use any of the following types of directives:

```
domain domainname
msgid message_identifier
msgid_plural untranslated_string_plural
msgstr message_string
msgstr[n] message_string
```

The behavior of the domain directive is affected by the options used. See OPTIONS for the behavior when the `-o` or `--output-file` options are specified. If the `-o` or `--output-file` options are not specified, the behavior of the domain directive is as follows:

- All msgids from the beginning of each .po file to the first domain directive are put into a default message object file. The default message object file is named `messages.mo`, if the Solaris message catalog file format is used to generate the message object file or if the `--strict` option is specified. Otherwise, the default message object file is named `messages`.
- When msgfmt encounters a domain *domainname* directive in the .po file, all following msgids until the next domain directive are put into the message object file, named `domainname.mo`, if the Solaris message catalog file format is used to generate the message object file or if the `--strict` option is specified. Otherwise, the msgids are put into the message object file named `domainname`.
- Duplicate msgids are defined in the scope of each domain. That is, a msgid is considered a duplicate only if the identical msgid exists in the same domain.

- All duplicate msgids are ignored.

The `msgid` directive specifies the value of a message identifier associated with the directive that follows it. The `msgid_plural` directive specifies the plural form message specified to the plural message handling functions `ngettext()`, `dngettext()`, or `dcngettext()`. The *message_identifier* string identifies a target string to be used at retrieval time. Each statement containing a `msgid` directive must be followed by a statement containing a `msgstr` directive or `msgstr[n]` directives.

The `msgstr` directive specifies the target string associated with the *message_identifier* string declared in the immediately preceding `msgid` directive.

The directive `msgstr[n]` (where $n = 0, 1, 2, \dots$) specifies the target string to be used with plural form handling functions `ngettext()`, `dngettext()`, and `dcngettext()`.

Message strings can contain the escape sequences `\n` for newline, `\t` for tab, `\v` for vertical tab, `\b` for backspace, `\r` for carriage return, `\f` for formfeed, `\\` for backslash, `\"` for double quote, `\b01`;a for alarm, `\ddd` for octal bit pattern, and `\xDD` for hexadecimal bit pattern.

Comments for a GNU-compatible message file should be in one of the following formats (the `msgfmt` utility will ignore these comments when processing Solaris message files):

```
# translator-comments
# . automatic-comments
#: reference. .
#, flag
```

The `#:` comments indicate the location of the `msgid` string in the source files in *filename:line* format. The `#`, `#.`, and `#:` comments are informative only and are silently ignored by the `msgfmt` utility. The `#,` comments require one or more flags separated by the comma character. The following *flags* can be specified:

table {
fuzzy	This flag can be inserted by the translator. It shows that the `msgstr` string might not be a correct translation (anymore). Only the translator can judge if the translation requires further modification or is acceptable as is. Once satisfied with the translation, the translator removes this fuzzy flag. If this flag is specified, the `msgfmt` utility will not generate the entry for the immediately following `msgid` in the output message catalog.
c-format	
no-c-format	The `c-format` flag indicates that the `msgid` string is used as a format string by `printf`-like functions. In case the `c-format` flag is given for a string, the `msgfmt` utility does some more tests to check the validity of the translation.

In the GNU-compatible message file, the `msgid` entry with empty string (`""`) is called the header entry and treated specially. If the message string for the header entry contains `nplurals=value`, the value indicates the number of plural forms. For example, if `nplurals=4`,

there are four plural forms. If `nplurals` is defined, the same line should contain `plural=expression`, separated by a semicolon character. The *expression* is a C language expression to determine which version of `msgstr[n]` is to be used based on the value of *n*, the last argument of `ngettext()`, `dngettext()`, or `dcngettext()`. For example,

```
nplurals=2; plural= n == 1 ? 0 : 1
```

indicates that there are two plural forms in the language. `msgstr[0]` is used if `n == 1`, otherwise `msgstr[1]` is used. For another example:

```
nplurals=3; plural= n == 1 ? 0 : n == 2 ? 1 : 2
```

indicates that there are three plural forms in the language. `msgstr[0]` is used if `n == 1`, `msgstr[1]` is used if `n == 2`, otherwise `msgstr[2]` is used.

If the header entry contains a `charset=codeset` string, the *codeset* is used to indicate the codeset to be used to encode the message strings. If the output string's codeset is different from the message string's codeset, codeset conversion from the message string's codeset to the output string's codeset will be performed upon the call of `gettext()`, `dgettext()`, `dcgettext()`, `ngettext()`, `dngettext()`, and `dcngettext()` for the GNU-compatible message catalogs. The output string's codeset is determined by the current locale's codeset (the return value of `nl_langinfo(CODESET)`) by default, and can be changed by the call of `bind_textdomain_codeset()`.

Message catalog file format

The `msgfmt` utility can generate the message object both in Solaris message catalog file format and in GNU-compatible message catalog file format. If the `-s` option is specified and the input file is a Solaris `.po` file, the `msgfmt` utility generates the message object in Solaris message catalog file format. If the `-g` option is specified and the input file is a GNU `.po` file, the `msgfmt` utility generates the message object in GNU-compatible message catalog file format. If neither the `-s` nor `-g` option is specified, the `msgfmt` utility determines the message catalog file format as follows:

- If the `.po` file contains a valid GNU header entry (having an empty string for `msgid`), the `msgfmt` utility uses the GNU-compatible message catalog file format.
- Otherwise, the `msgfmt` utility uses the Solaris message catalog file format.

If the `msgfmt` utility determined that the Solaris message catalog file format is used, as above, but found the `.po` file contains directives that are specific to the GNU-compatible message catalog file format, such as `msgid_plural` and `msgstr[n]`, the `msgfmt` utility handles those directives as invalid specifications.

使用例

例 1 Creating message objects from message files

In this example, `module1.po` and `module2.po` are portable message objects files.

```
example% cat module1.po
# default domain "messages.mo"
```

例1 Creating message objects from message files (続き)

```

msgid "msg 1"
msgstr "msg 1 translation"
#
domain "help_domain"
msgid "help 2"
msgstr "help 2 translation"
#
domain "error_domain"
msgid "error 3"
msgstr "error 3 translation"
example% cat module2.po
# default domain "messages.mo"
msgid "mesg 4"
msgstr "mesg 4 translation"
#
domain "error_domain"
msgid "error 5"
msgstr "error 5 translation"
#
domain "window_domain"
msgid "window 6"
msgstr "window 6 translation"

```

The following command will produce the output files `messages.mo`, `help_domain.mo`, and `error_domain.mo` in Solaris message catalog file format:

```
example% msgfmt module1.po
```

The following command will produce the output files `messages.mo`, `help_domain.mo`, `error_domain.mo`, and `window_domain.mo` in Solaris message catalog file format:

```
example% msgfmt module1.po module2.po
```

The following command will produce the output file `hello.mo` in Solaris message catalog file format:

```
example% msgfmt -o hello.mo module1.po module2.po
```

環境変数

See [environ\(5\)](#) for descriptions of the following environmental variables that affect the execution of `msgfmt`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/locale

ATTRIBUTE TYPE	ATTRIBUTE VALUE
CSI	Enabled

関連項目 [xgettext\(1\)](#), [gettext\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 Installing message catalogs under the C locale is pointless, since they are ignored for the sake of efficiency.

名前	msggen – generate a machine independent formatted message catalog
形式	msggen [-fls] <i>catfile</i> [<i>msgfile</i>]
機能説明	msggen merges the message text source file <i>msgfile</i> into a machine independent formatted message catalog <i>catfile</i>. The file <i>catfile</i> is created if it does not already exist. If <i>catfile</i> does exist, its messages are included in the new <i>catfile</i>. If set and message numbers collide, the new message text defined in <i>msgfile</i> replaces the old message text currently contained in <i>catfile</i>. Non-ASCII characters must be UTF-8 encoded. iconv(1) can be used to convert to/from UTF-8.
オプション	The following options are supported: -f --format List the printf(3C) format signature for each message in <i>catfile</i>. A format signature is one line containing one character for each format specification: - c char - d double - D long double - f float - h short - i int - j long long - l long - p void* - s string - t ptrdiff_t - z size_t ? unknown -l --list List <i>catfile</i> in UTF-8 msgfile form. -s --set Convert the <i>catfile</i> to a message set number and print the number on the standard output.

オペランド

The following operands are supported:

catfile Machine independent formatted message catalog file.

msgfile Message text source file.

使用法

Message text source files are in [gencat\(1\)](#) format, defined as follows. The fields of a message text source line are separated by a single blank character. Any other blank characters are considered to be part of the subsequent field. The `NL_*` constants are defined in one or both of `<limits.h>` and `<nl_types.h>`.

\$comment

A line beginning with a `$` followed by a blank character is treated as a comment.

\$delset n comment

This line deletes message set *n* from an existing message catalog. *n* denotes the set number `[1, NL_SETMAX]`. Any text following the set number is treated as a comment.

\$quote c

This line specifies an optional quote character *c*, which can be used to surround message-text so that trailing spaces or empty messages are visible in a message source line. By default, or if an empty `$quote` directive is supplied, no quoting of message-text is recognized.

\$set n comment

This line specifies the set identifier of the following messages until the next `$set` or end-of-file (EOF) appears. *n* denotes the set identifier, which is defined as a number in the range `[1, NL_SETMAX]`. Set numbers need not be contiguous. Any text following the set identifier is treated as a comment. If no `$set` directive is specified in a message text source file, all messages are located in message set 1.

\$translation identification YYYY-MM-DD[,...]

Append translation information to the message catalog header. Only the newest date for a given identification is retained in the catalog. Multiple translation lines are combined into a single, comma-separated list.

m message-text

m denotes the message identifier, which is defined as a number in the range `[1, NL_MSGMAX]`. The message-text is stored in the message catalogue with the set identifier specified by the last `$set` directive, and with message identifier *m*. If the message-text is empty, and a blank character field separator is present, an empty string is stored in the message catalogue. If a message source line has a message number, but neither a field separator nor message-text, the existing message with that number (if any) is deleted from the catalogue. Message identifiers need not be contiguous. There are no *message-text* length restrictions.

終了ステータス 0 Successful completion.
 >0 One or more specified jobs does not exist.

使用例 例 1 Using msggen
The following example generates a message catalog xxx from the message file xxx.msg:
example% msggen xxx xxx.msg

Authors Glenn Fowler, gsf@research.att.com

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/astdev
Interface Stability	Volatile

関連項目 [gencat\(1\)](#), [iconv\(1\)](#), [msgcc\(1\)](#), [printf\(3C\)](#), [attributes\(5\)](#)

名前 msgget – get a message from a message catalog

形式 `msgget locale [command:]catalog [set.]number [text]`

機能説明 msgget gets the message corresponding to the parameters. See OPERANDS.

オペランド The following operands are supported:

<i>catalog</i>	Specifies the message catalog name.
<i>command</i>	Specifies command-specific message.
<i>locale</i>	Specifies the locale. If <i>locale</i> is - then the current locale is used.
<i>[set].number</i>	Identifies the message by message number and an optional message set. If specified as - , the message set and number are determined by looking up text in the corresponding C locale message catalog.
<i>text</i>	Specifies the text of the message to be output upon error.

終了ステータス

0	Successful completion.
>0	An error occurred.

使用例

例 1 Getting a Message in the Current Locale

The following example gets msg 1 in the current locale from message catalog hello:

```
example% msgget - hello 1
hello world
```

Authors Glenn Fowler, gsf@research.att.com

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/astdev
Interface Stability	Volatile

関連項目 [iconv\(1\)](#), [msgcc\(1\)](#), [msggen\(1\)](#), [attributes\(5\)](#)

名前	mt – 磁気テープの制御																		
形式	mt [-f <i>tapename</i>] <i>command</i> ... [<i>count</i>]																		
機能説明	mt ユーティリティーは、指定されたコマンドを磁気テープドライブに送信します。 <i>-f tapename</i> が指定されていない場合は、環境変数 TAPE が使用されます。 TAPE が存在しない場合、mt はデバイス /dev/rmt/0n を使用します。																		
オプション	サポートしているオプションは、次のとおりです。 <i>-f tapename</i> raw テープデバイスを指定します。																		
オペランド	次のオペランドがサポートされています。 <table> <tr> <td><i>count</i></td><td>要求された操作を実行する回数です。mtは、デフォルトでは要求された <i>command</i> を 1 回だけ実行します。<i>count</i> を指定することにより、複数の <i>command</i> 操作を実行できます。</td></tr> <tr> <td><i>command</i></td><td>磁気テープドライブに送信できる、次の使用可能なコマンドがサポートされています。コマンド名の一部 (ただし、コマンドとして一意に認識できる文字数) だけの指定でもかまいません。 <table> <tr> <td><i>asf</i></td><td><i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、<i>fsf count</i> を指定した場合と同等です。</td></tr> <tr> <td><i>bsf</i></td><td><i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。</td></tr> <tr> <td><i>bsr</i></td><td><i>count</i> で指定された数のレコード分だけ後退します。</td></tr> <tr> <td><i>bssf</i></td><td>要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、<i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。</td></tr> <tr> <td><i>eof</i></td><td></td></tr> <tr> <td><i>weof</i></td><td>テープ上の現在の位置に、<i>count</i> で指定された数の EOF マークを書き込みます。</td></tr> <tr> <td><i>fsf</i></td><td><i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。</td></tr> </table> </td></tr> </table>	<i>count</i>	要求された操作を実行する回数です。mtは、デフォルトでは要求された <i>command</i> を 1 回だけ実行します。 <i>count</i> を指定することにより、複数の <i>command</i> 操作を実行できます。	<i>command</i>	磁気テープドライブに送信できる、次の使用可能なコマンドがサポートされています。コマンド名の一部 (ただし、コマンドとして一意に認識できる文字数) だけの指定でもかまいません。 <table> <tr> <td><i>asf</i></td><td><i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、<i>fsf count</i> を指定した場合と同等です。</td></tr> <tr> <td><i>bsf</i></td><td><i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。</td></tr> <tr> <td><i>bsr</i></td><td><i>count</i> で指定された数のレコード分だけ後退します。</td></tr> <tr> <td><i>bssf</i></td><td>要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、<i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。</td></tr> <tr> <td><i>eof</i></td><td></td></tr> <tr> <td><i>weof</i></td><td>テープ上の現在の位置に、<i>count</i> で指定された数の EOF マークを書き込みます。</td></tr> <tr> <td><i>fsf</i></td><td><i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。</td></tr> </table>	<i>asf</i>	<i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、 <i>fsf count</i> を指定した場合と同等です。	<i>bsf</i>	<i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。	<i>bsr</i>	<i>count</i> で指定された数のレコード分だけ後退します。	<i>bssf</i>	要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、 <i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。	<i>eof</i>		<i>weof</i>	テープ上の現在の位置に、 <i>count</i> で指定された数の EOF マークを書き込みます。	<i>fsf</i>	<i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。
<i>count</i>	要求された操作を実行する回数です。mtは、デフォルトでは要求された <i>command</i> を 1 回だけ実行します。 <i>count</i> を指定することにより、複数の <i>command</i> 操作を実行できます。																		
<i>command</i>	磁気テープドライブに送信できる、次の使用可能なコマンドがサポートされています。コマンド名の一部 (ただし、コマンドとして一意に認識できる文字数) だけの指定でもかまいません。 <table> <tr> <td><i>asf</i></td><td><i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、<i>fsf count</i> を指定した場合と同等です。</td></tr> <tr> <td><i>bsf</i></td><td><i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。</td></tr> <tr> <td><i>bsr</i></td><td><i>count</i> で指定された数のレコード分だけ後退します。</td></tr> <tr> <td><i>bssf</i></td><td>要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、<i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。</td></tr> <tr> <td><i>eof</i></td><td></td></tr> <tr> <td><i>weof</i></td><td>テープ上の現在の位置に、<i>count</i> で指定された数の EOF マークを書き込みます。</td></tr> <tr> <td><i>fsf</i></td><td><i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。</td></tr> </table>	<i>asf</i>	<i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、 <i>fsf count</i> を指定した場合と同等です。	<i>bsf</i>	<i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。	<i>bsr</i>	<i>count</i> で指定された数のレコード分だけ後退します。	<i>bssf</i>	要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、 <i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。	<i>eof</i>		<i>weof</i>	テープ上の現在の位置に、 <i>count</i> で指定された数の EOF マークを書き込みます。	<i>fsf</i>	<i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。				
<i>asf</i>	<i>count</i> ファイル番号への絶対位置移動を指定します。このコマンドは、rewind後に、 <i>fsf count</i> を指定した場合と同等です。																		
<i>bsf</i>	<i>count</i> で指定された数の EOF 分だけ後退します。テープの位置は、EOF マークのテープの先頭側になります。																		
<i>bsr</i>	<i>count</i> で指定された数のレコード分だけ後退します。																		
<i>bssf</i>	要求されたシーケンシャルファイルマークの数だけ後退します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、 <i>bssf 4</i> は 4 つのシーケンシャルファイルマークがある 1 番目の場所まで後方に検索して、4 番目のファイルマークの BOP 側に移動します。 一部のドライブでは、このコマンドはサポートされていません。																		
<i>eof</i>																			
<i>weof</i>	テープ上の現在の位置に、 <i>count</i> で指定された数の EOF マークを書き込みます。																		
<i>fsf</i>	<i>count</i> で指定された数の EOF 分だけ進みます。テープの位置は、ファイルの第 1 ブロックの位置になります。																		

<code>fsr</code>	<code>count</code> で指定された数のレコード分だけ進みます。
<code>fssf</code>	要求されたシーケンシャルファイルマークの数だけ前進します。シーケンシャルファイルマークとは、ほかのどのような種類のブロックもファイルマーク間に挿入せずに、次々とファイルマークが記録される場所です。番号引数は、進むシーケンシャルファイルマークの数を指定します。たとえば、 <code>fssf 4</code> は、4 つのシーケンシャルファイルマークがある 1 番目の場所まで前方検索して、4 番目のファイルマークの後ろに移動します。 一部のドライブでは、このコマンドはサポートされていません。
<code>load</code>	ドライブのロードとスレッドの現在のメディアを要求します。一部のドライブではサポートされていません。
<code>lock</code>	メディアの取り外しを防ぎます。
<code>nbsf</code>	<code>count</code> で指定された数のファイル分だけ後退します。テープの位置は、ファイルの第 1 ブロックの位置になります。これは、 <code>count+1</code> 回 <code>bsf</code> を行い、1 回 <code>fsf</code> を行う場合と同等です。
<code>seek</code>	要求されたテープの論理位置に移動します。
<code>tell</code>	現在のテープの論理位置を取得して出力します。
<code>unlock</code>	メディアの取り外しを許可します。

`count` が次のいずれかのコマンドで指定された場合、`count` は無視され、コマンドが 1 度だけ実行されます。

<code>config</code>	現在のドライブ構成をドライバから読み取って、 <code>st.conf</code> 形式で表示します。各フィールドの定義とその意味については、 st(7D) を参照してください。
<code>eom</code>	テープ上の記録済み部分の末尾まで移動します。このコマンドは、すでに書き込んであるテープにファイルを追加するのに便利です。
<code>erase</code>	テープ全体を消去します。

テープドライブによっては、テープの特定の部分だけを消去できるオプション設定が存在するものもあります。正しい設定、つまり、テープ全体を消去する設定を選択するようにしてください。テープの消去は、装置またはテープ、もしくはその両方に

よっては時間がかかります。時間の詳細については、装置についているマニュアルを参照してください。

forcereserve 別のイニシエータによって発行された SCSI II の予約を破棄しようとします。このコマンドが完了したとき、ドライブは現在のイニシエータのために予約されていませんが、使用することはできます。このコマンドは、スーパーユーザー特権を持つユーザーのみが実行できます。

offline
rewoffl テープを巻き戻し、適切であれば、テープをアンロードし、ドライブ装置をオフラインにします。

release クローズ時にリリースするデフォルトの動作を確定し直します。

reserve デバイスをクローズしたあともテープドライブを予約したままにします。ドライブをリリースするときには、明示的に行う必要があります。

retension カートリッジテープを完全に巻き戻し、次にリールの終わりまで進め、テープの初めまで戻して、テープの張りをなめらかにします。

rewind テープを巻き戻します。

status テープデバイスのステータスに関する情報を表示します。

ステータス情報には、ドライブで報告されたセンスキー、最後の操作の残りと再試行、ファイル番号で報告された現在のテープ位置、およびファイルの先頭からのブロック数が含まれることがあります。ドライブで WORM メディアがロードされたことが報告される場合もあります。

- 終了ステータス
- 0 操作はすべて正常に終了しました。
 - 1 コマンドは認識されなかった。または **mt** は指定された磁気テープドライブをオープンできなかった。
 - 2 操作は失敗しました。

ファイル `/dev/rmt/*` 磁気テープインタフェース

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	SUNWcsu

関連項目 [tar\(1\)](#), [tcopy\(1\)](#), [ar.h\(3HEAD\)](#), [attributes\(5\)](#), [mtio\(7I\)](#), [st\(7D\)](#)

使用上の留意点 すべてのデバイスがすべてのオプションをサポートしているわけではありません。いくつかのオプションは、ハードウェアに依存します。該当するデバイスのマニュアルページを参照してください。

mt は、アーキテクチャーに依存します。異機種間の操作 (つまり、SPARC からほかの機種、またはその逆) はサポートされていません。

名前	mv – ファイルの移動
形式	<pre> /usr/bin/mv [-fi] source target_file /usr/bin/mv [-fi] source... target_dir /usr/xpg4/bin/mv [-fi] source target_file /usr/xpg4/bin/mv [-fi] source... target_dir </pre>
機能説明	第1の形式のmvユーティリティーは、<i>source</i> オペランドが示すファイルを <i>target_file</i> が示す宛先に移動します。<i>source</i> と <i>target_file</i> に同じ名前を指定することはできません。<i>target_file</i> が存在しない場合、mv は <i>target_file</i> という名前のファイルを作成します。<i>target_file</i> が存在していれば、その内容は上書きされます。最終オペランドが既存のディレクトリを示していないとき、mv はこの第1形式であると見なします。 第2の形式のmv は、<i>source</i> オペランドで指定された各ファイルを <i>target_dir</i> オペランドが示す既存ディレクトリ中のファイルに移動します。各 <i>source</i> 用の宛先パス名は、宛先ディレクトリ名のあとにスラッシュ(/)と <i>source</i> の最終パス名部分を付加したものとなります。オペランドが既存のディレクトリを示しているとき、mv はこの第2形式であると見なします。 mv は、<i>target_file</i> のモードが書き込み禁止であると判断すると、モードを表示し(chmod(2)を参照)、応答を要求して、標準入力を1行読み取ります。応答が肯定である場合、使用可能ならばmv が実行されます。それ以外の場合、このコマンドは終了します。<i>target</i> が ACLに関連付けられている場合、モードはアクセス権を完全には表示しません。<i>source</i> の親ディレクトリが書き込み可能でスティッキー・ビットセットを持っている場合、次の条件のうち1つ以上が真である必要があります。 ■ ユーザーはファイルを所有している必要がある ■ ユーザーはディレクトリを所有している必要がある ■ ファイルはユーザーが書き込み可能である必要がある ■ ユーザーは特権ユーザーである必要がある <i>source</i> がファイルで <i>target_file</i> が複数のリンクを持っている別のファイルへのリンクである場合、ほかのリンクは残存し、<i>target_file</i> は新しいファイルになります。 <i>source</i> と <i>target_file/target_dir</i> が異なるファイルシステム上にある場合、mv はソースファイルをコピーし、もとのファイルまたはディレクトリを削除します。ほかのファイルへのハードリンクはすべてなくなります。mv は、ソースファイルの特性(ファイルの所有者IDやグループID、アクセス権モード、修正時間やアクセス時間、および適用可能であればACLと拡張属性)も対象ファイルに引き継ごうとします。シンボリックリンクに対しては、mv はリンク自体の所有者IDやグループIDだけを保存します。 所有者IDやグループIDを保存できなかった場合、mv は対象ファイルの <code>S_ISUID</code> ビットと <code>S_ISGID</code> ビットをクリアします。これらのビットをクリアできない場合、mv は標準エラー出力に診断メッセージを出力しますが、終了ステータスには影響を及ぼしません。ターゲットのファイルシステムが拡張属性をサポートしていな

い場合、mv は拡張属性を保持できません。/usr/xpg4/bin/mv が、ファイルの特性を引き継ごうとして失敗したその他すべてのエラーについて、標準エラー出力に診断メッセージを出力します。終了コードには影響しません。

ソースファイルの特性を保存するには、ユーザーが適切なファイルのアクセス権を持っている必要があります。つまり、スーパーユーザーになるか、あるいは対象ファイルと同じ所有者 ID を持つことが必要です。

オプション サポートしているオプションは、次のとおりです。

- f mv は、既存の *target* に書き込む場合でもプロンプトを出さずにファイルを移動します。標準入力端末でない場合これがデフォルトになることに注意してください。
- i mv は、移動により既存の *target* が上書きされる場合は、必ず、確認のプロンプトを出します。これは、入力が端末から送られているかどうかに関係なく実行されます。確認のプロンプトでエラーが発生した場合は、ユーザーによる否定応答と同等です。肯定を応答すれば、移動処理は続行されます。その他の応答では、mv は *target* を上書きしません。

/usr/bin/mv -f と -i のオプションを両方指定してもエラーとは見なされません。-f オプションは、-i オプションよりも優先されます。

/usr/xpg4/bin/mv -f と -i のオプションを両方指定してもエラーとは見なされません。最後に指定したオプションが mv の動作を決定します。

オペランド 次のオペランドがサポートされています。

source 移動するファイルまたはディレクトリのパス名
target_file 移動するファイルまたはディレクトリの新たなパス名
target_dir ファイルの移動先となる既存ディレクトリのパス名

使用法 ファイルが 2G バイト (2^{31} バイト) 以上ある場合の mv の動作については、[largefile\(5\)](#) を参照してください。

環境変数 mv の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの `yesexpr` キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、`yesexpr` に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、`yesexpr` に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#) を参照してください。

終了ステータス 次の終了ステータスが返されます。

0 入力ファイルはすべて正常に移されました。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/mv

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

/usr/xpg4/bin/mv

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	標準

関連項目 [cp\(1\)](#), [cpio\(1\)](#), [ln\(1\)](#), [rm\(1\)](#), [setfacl\(1\)](#), [chmod\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 -- によって、ユーザーはコマンド行オプションの終端を明示的にマーク付けできるので、mv は - で始まるファイル名引数を認識できるようになります。BSD 移行のサポートとして、mv は - を -- の同義語として受け入れます。この導入補助は、将来のリリースでは提供されない可能性があります。

名前	nawk - パターン走査およびパターン処理の言語
形式	<pre>/usr/bin/nawk [-F ERE] [-v assignment] 'program' -f progfile... [argument]... /usr/xpg4/bin/awk [-F ERE] [-v assignment]... 'program' -f progfile... [argument]...</pre>
機能説明	<code>/usr/bin/nawk</code> と <code>/usr/xpg4/bin/awk</code> の両ユーティリティーは、テキストデータ処理専用の <code>nawk</code> プログラミング言語で記述されたプログラムを実行します。<code>nawk</code> プログラムは、一連のパターンとそれに対応するアクション(動作)から構成されます。<code>program</code> を指定する文字列は、シェルによって解釈されないように単一引用符 (') で囲む必要があります。一連のパターン-アクション文は、<code>program</code> としてコマンド行に指定することも、<code>-f progfile</code> オプションで指定する1つ以上のファイル内に指定することもできます。パターンに一致する入力を読み取られると、そのパターンに関連付けられたアクションが実行されます。 入力は一連のレコードとして解釈されます。1レコードはデフォルトで1行ですが、RS 組み込み変数を使用すれば変更できます。入力の各レコードは、<code>program</code> 内の各パターンと照合されます。パターンが一致するごとに、関連付けられたアクションが実行されます。 <code>nawk</code> ユーティリティーは、各入力レコードを一連のフィールドとして解釈します。デフォルトでは、フィールドは空白以外の文字列です。デフォルトのスペースフィールド区切り文字(スペースまたはタブ、あるいはその両方)は、FS 組み込み変数または <code>-F ERE</code> オプションを使って変更できます。<code>nawk</code> ユーティリティーは、レコードの最初のフィールドを \$1、2番目のフィールドを \$2、(以下同様)と見なします。記号 \$0 はレコード全体を指します。その他のフィールドを設定すると、\$0 が再評価されます。\$0 を割り当てると、すべてのフィールドと NF 組み込み変数の値がリセットされます。
オプション	サポートしているオプションは、次のとおりです。 <code>-F ERE</code> 入力を読み取られる前に、入力フィールド区切り文字が拡張正規表現 <code>ERE</code> であることを定義します(文字も可)。 <code>-f progfile</code> <code>nawk</code> プログラムが含まれているファイル <code>progfile</code> のパス名を指定します。このオプションを複数指定した場合は、<code>progfile</code> として指定されたファイルが指定された順序で連結されて <code>nawk</code> プログラムになります。<code>nawk</code> プログラムは、1つの引数としてコマンド行に指定することもできます。 <code>-v assignment</code> <code>assignment</code> 引数は、<code>assignment</code> オペランドと同じ形式にする必要があります。代入の形式は <code>var=value</code> です。<code>var</code> は後述する変数の1つです。指定された代入は、BEGIN パターンがあればそれに関連付けられたアクションも含め、<code>nawk</code> プログラムの実行前に行われます。このオプションは複数回指定できます。

オペランド

次のオペランドがサポートされています。

- program* -f オプションを省略すると、nawk の最初のオペランドは nawk プログラムのテキストになります。アプリケーションは nawk への単一引数として *program* オペランドを提供します。テキストが改行文字で終わっていない場合、nawk はテキストが改行文字で終わっているものとして解釈します。
- argument* 次の 2 つのタイプの *argument* は混在可能です。
- file* 読み取られる入力が含まれているファイルのパス名。プログラム内の一連のパターンと照合されます。*file* オペランドを 1 つも指定しない場合、または *file* オペランドに - を指定した場合には、標準入力 が用いられます。
- assignment* 下線または移植性のある文字セットの英字で始まり、そのあとに下線、移植性のある文字セットの数字と英字、= 記号が続くオペランドは、パス名ではなく変数の代入を指定するものです。= 記号より前の文字は nawk 変数の名前を表します。この名前が nawk の予約語である場合、動作は未定義です。等号よりあとの文字は、nawk プログラム内に二重引用符 (") で囲まれているもの、つまり STRING トークンとして解釈されます。ただし、最後の文字がエスケープされていないバックスラッシュの場合、それはシーケンス「\」の最初の文字ではなく、バックスラッシュという文字そのものとして解釈されます。変数には、この STRING トークンの値が代入されます。値が数値列と考えられる場合、変数にはその数値が代入されます。このような変数代入は、そのあとに *file* 指定があればその処理の直前に実行されます。そのため、最初の *file* 引数より前の代入は、BEGIN アクション(ある場合)のあとに実行され、最後の *file* 引数よりあとの代入は END アクション(ある場合)の前に実行されます。*file* 引数がない場合は、標準入力の処理の前に代入が実行されます。

入力ファイル

nawk プログラムへの入力ファイルとして、次のソースからのものがあります。

- *file* オペランドまたはそれと同等のもの (nawk 変数 ARGV と ARGC を変更して得られる)
- 標準入力 (*file* オペランドがない場合)
- getline 関数の引数

これらはテキストファイルである必要があります。これらのファイルについて、変数 RS が復帰改行文字以外の値に設定されるかどうかによって、システムでは指定

された区切り文字で終わるレコードを {LINE_MAX} バイトまでサポートしたり、もっと長いレコードをサポートすることができます。

-f *progfile* を指定する場合、*progfile* オプション引数のそれぞれが示すファイルは、nawk プログラムが含まれているテキストファイルである必要があります。

標準入力を使用されるのは、*file* オペランドが指定されない場合、または - の場合だけです。

拡張機能説明

nawk プログラムは、次の形式のパターンとアクションのペアから構成されます。

```
pattern { action }
```

パターンまたはアクション (中括弧を含む) のどちらかを省略することもできます。パターン-アクション文は、セミコロンまたは復帰改行文字で区切られます。

パターンを省略すると、入力のどのレコードとも一致します。アクションを省略すると、入力内の一致するレコードが標準出力に書き込まれます。

nawk プログラムの実行は、まずすべての BEGIN パターンに関連付けられたアクションをプログラム内の順序で実行することによって開始されます。次に、レコード区切り文字 (デフォルトでは復帰改行文字) が検出されるまで、各 *file* オペランド (ファイルを指定しない場合は標準入力) が処理され、ファイルからデータが読み取られ、現在の FS 値を使用して現在のレコードがフィールドに分割され、プログラム内の順序で各パターンが評価され、現在のレコードに一致する各パターンに関連付けられたアクションが実行されます。後続のパターンが評価される前に、一致するパターンのアクションが実行されます。最後に、すべての END パターンに関連付けられたアクションがプログラム内の順序で実行されます。

nawk 内の式

式は、パターンとアクションで使用される計算を記述します。次の表では、有効な式演算をグループごとに、優先度の高い順に並べてあります。優先度が同じ演算子は上下の横線で囲まれています。構文があいまいな式は、優先度の高い演算子が優先度の低い演算子よりも先に評価されます。この表の *expr*、*expr1*、*expr2*、および *expr3* は任意の式を表し、*lvalue* は代入される要素 (つまり、代入演算子の左側にあるもの) を表します。

構文	「名前」	結果の型	結合規則
(<i>expr</i>)	グループ	<i>expr</i> の型	該当せず
<i>\$expr</i>	フィールド参照	string	該当せず
<i>++ lvalue</i>	前置インクリメント	数値	該当せず
<i>-- lvalue</i>	前置デクリメント	数値	該当せず
<i>lvalue ++</i>	後置インクリメント	数値	該当せず

構文	「名前」	結果の型	結合規則
<i>lvalue</i> —	後置デクリメント	数値	該当せず
<i>expr</i> ^ <i>expr</i>	べき乗	数値	右
! <i>expr</i>	論理否定	数値	該当せず
+ <i>expr</i>	単項プラス	数値	該当せず
− <i>expr</i>	単項マイナス	数値	該当せず
<i>expr</i> * <i>expr</i>	乗算	数値	左
<i>expr</i> / <i>expr</i>	除算	数値	左
<i>expr</i> % <i>expr</i>	剰余演算	数値	左
<i>expr</i> + <i>expr</i>	加算	数値	左
<i>expr</i> − <i>expr</i>	減算	数値	左
<i>expr</i> <i>expr</i>	文字列連結	string	左
<i>expr</i> < <i>expr</i>	より小さい	数値	なし
<i>expr</i> <= <i>expr</i>	小さいか等しい	数値	なし
<i>expr</i> != <i>expr</i>	等しくない	数値	なし
<i>expr</i> == <i>expr</i>	等しい	数値	なし
<i>expr</i> > <i>expr</i>	より大きい	数値	なし
<i>expr</i> >= <i>expr</i>	大きいか等しい	数値	なし
<i>expr</i> ~ <i>expr</i>	ERE 一致	数値	なし
<i>expr</i> !~ <i>expr</i>	ERE 不一致	数値	なし
<i>expr</i> in <i>array</i>	配列のメンバー	数値	左
(<i>index</i>) in <i>array</i>	多次元配列 メンバー	数値	左
<i>expr</i> && <i>expr</i>	論理積	数値	左
<i>expr</i> <i>expr</i>	論理和	数値	左
<i>expr</i> 1 ? <i>expr</i> 2 : <i>expr</i> 3	条件式	選ばれた式の型 <i>expr</i> 2 または <i>expr</i> 3	右
<i>lvalue</i> ^= <i>expr</i>	べき乗 代入	数値	右

構文	「名前」	結果の型	結合規則
<i>lvalue</i> %= <i>expr</i>	剰余演算代入	数値	右
<i>lvalue</i> *= <i>expr</i>	乗算	数値	右
	代入		
<i>lvalue</i> /= <i>expr</i>	除算代入	数値	右
<i>lvalue</i> += <i>expr</i>	加算代入	数値	右
<i>lvalue</i> -= <i>expr</i>	減算代入	数値	右
<i>lvalue</i> = <i>expr</i>	代入	<i>expr</i> の型	右

各式には、1つの文字列値、数値、またはその両方が入ります。すでに述べた特殊なコンテキストを除き、式の値は、その式が使われるコンテキストに必要な型に暗黙のうちに変換されます。文字列値は、次の呼び出しと同等のものによって数値に変換されます。

```
setlocale(LC_NUMERIC, "");
numeric_value = atof(string_value);
```

数値 (正確には整数値) は、文字列 `%d` が `fmt` 引数として使用され、変換される数値が最初で唯一の `expr` 引数として使用される `sprintf` 関数の呼び出しと同等のものによって文字列に変換されます。その他の数値は、変数 `CONVFMT` の値が `fmt` 引数として使用され、変換される数値が最初で唯一の `expr` 引数として使用される `sprintf` 関数の呼び出しと同等のものによって文字列に変換されます。

次の場合、文字列値は数値列と見なされます。

- 1. 先行および末尾の空白文字が無視されます。
- 2. 無視されない最初の文字が `+` または `-` の場合、それは無視されます。
- 3. 無視されない残りの文字が構文上 `NUMBER` トークンとして認識される場合、文字列は数値列と見なされます。

前述の手順で `-` 文字が無視されると、数値列の値は、認識された `NUMBER` トークンの数値の正負を逆にした値になります。それ以外の場合、数値列の数値は認識された `NUMBER` トークンの数値になります。文字列が数値列かどうかは問題になるのは、このセクションでその用語が使用されているコンテキストにおいてのみです。

ブール型のコンテキストで式が使用されるとき、式に数値が含まれている場合、値ゼロは偽として扱われ、その他の値は真として扱われます。それ以外の場合、`NULL` 文字列の文字列値は偽として扱われ、その他の値は真として扱われます。ブール型のコンテキストは、次のいずれかです。

- 条件式の最初の部分式。
- 論理否定、論理積、または論理和によって演算される式。

- for 文の 2 番目の式。
- if 文の式。
- while 文または do . . . while 文の while 句の式。
- パターンとして使用される式 (Overall Program Structure 内のように)。

nawk 言語は、数字や文字列を格納するための配列を提供します。配列を宣言する必要はありません。配列は最初は空で、サイズは動的に変わります。連想配列機能の一種を提供する添字 (要素識別子) は文字列です。配列名とそれに続く角括弧内の添字は、構文で説明しているように、*lvalue* や式として使用できます。添字のない配列名を使用できるのは、次のコンテキストにおいてのみです。

- 関数定義または関数呼び出し内のパラメータ。
- キーワード `in` のあとに続く NAME トークン。

有効な配列インデックスは、一部のプログラミング言語で多次元配列にインデックスが付けられるのと同様に、コンマで区切られた 1 つ以上の式から構成されます。nawk 配列は実際には 1 次元なので、SUBSEP 変数の値で区切られている個々の式の文字列値を連結することで、そのようなコンマで区切られたリストが 1 つの文字列に変換されます。

したがって、次の 2 つのインデックス処理は等しい結果となります。

```
var[expr1, expr2, ... exprn]
var[expr1 SUBSEP expr2 SUBSEP ... SUBSEP exprn]
```

`in` 演算子が付いた多次元インデックスは、括弧で囲む必要があります。`in` 演算子は、特定の配列要素の有無を調べ、無い場合には要素を作成しません。存在しない配列要素へのその他の参照を行うと、要素は自動的に作成されます。

変数と特殊変数

nawk プログラムでは、変数を参照することで変数を使用できます。関数パラメータ以外は、明示的に宣言されません。初期化されていないスカラー変数と配列要素には、ゼロの数値と空の文字列の文字列値の両方が入っています。

フィールド変数は、`$` とそれに続く数字または数値式によって指定されます。負でない整数以外のものに評価されるフィールド番号 *expression* の結果は不確定です。この状況では、初期化されていない変数や文字列値は数値に変換される必要があります。新しいフィールド変数は、値を割り当てることによって作成されます。存在しないフィールド (つまり、`$NF` のあとのフィールド) を参照すると、NULL 文字列が生成されます。ただし、存在しないフィールドに割り当てると (たとえば、`$(NF+2) = 5`)、`NF` の値が増分され、値として NULL 文字列を持つ中間フィールドが作成され、`$0` の値が再計算され、`OFs` の値でフィールドが区切られます。各フィールド変数には、作成時に文字列値が入ります。現在のロケールの小数点文字がピリオド文字に変更された文字列は数値列と見なされ (前述の「nawk 内の式」を参照)、フィールド変数にも数値列の値が入ります。

`/usr/bin/nawk`、`/usr/xpg4/bin/nawk`、`/usr/bin/awk`、`/usr/xpg4/bin/awk`では次の特殊な変数を設定します。これらの変数は、`/usr/bin/nawk`と`/usr/xpg4/bin/awk`の両方でサポートされます。

ARGC	ARGV 配列内の要素の数。
ARGV	オプションと <i>program</i> 引数を除いたコマンド行引数の配列。ゼロから ARGC - 1 までの番号が付けられます。 ARGV 内の引数は変更や追加が可能です。ARGC は変更できます。各入力ファイルが終わると、nawk は、現在の ARGC - 1 の値まで、ARGV の次の NULL 以外の要素を次の入力ファイルの名前として扱います。ARGV の要素を NULL に設定すると、入力ファイルとして扱われなくなります。名前- は標準入力を示します。引数が <i>assignment</i> オペランドの形式と一致すると、その引数は <i>file</i> 引数ではなく代入として扱われます。
ENVIRON	変数 ENVIRON は環境値を表す配列です。配列のインデックスは環境変数の名前から成る文字列で、各配列要素の値はその変数の値から成る文字列です。環境変数の値が数値列と見なされる場合は、配列要素にもその数値が入ります。 nawk の動作が環境変数の影響を受ける (nawk が <code>system</code> 関数を使って実行するコマンド、および <code>print</code> 文、 <code>printf</code> 文、または <code>getline</code> 関数を使ってパイプライン経由で実行するコマンドの環境を含む) 場合はいずれも、nawk の実行開始時点の環境が使われます。
FILENAME	現在の入力ファイルのパス名。BEGIN アクション内では、この値は未定義です。END アクション内では、この値は最後に処理された入力ファイルの名前です。
FNR	現在のファイル内の現在のレコードの序数。BEGIN アクション内では、この値はゼロです。END アクション内では、この値は最後に処理されたファイル内の最後に処理されたレコードの番号です。
FS	入力フィールド区切り文字の正規表現。デフォルトはスペース文字です。
NF	現在のレコード内のフィールドの数。BEGIN アクション内では、 <i>var</i> 引数のない <code>getline</code> 関数とその前に実行されていないかぎり、NF の使用は未定義です。END アクション内では、END アクションに入る前に後続のリダイレクトされた <i>var</i> 引数のない <code>getline</code> 関数が実行されていないかぎり、NF は最後に読み取られたレコードの値を保持します。
NR	入力の開始からの現在のレコードの序数。BEGIN アクション内では、この値はゼロです。END アクション内では、この値は最後に処理されたレコードの番号です。

OFMT	出力文 "%.6g" 内でデフォルトとして数字を文字列に変換するための printf 形式。OFMT の値が浮動小数点形式の仕様でない場合、変換の結果は不確定です。
OFS	print 文出力フィールド区切り文字。デフォルトはスペース文字です。
ORS	print 出力レコード区切り文字。デフォルトは復帰改行文字です。
LENGTH	match 関数によって照合される文字列の長さ。
RS	RS の文字列値の最初の文字は入力レコード区切り文字(デフォルトは復帰改行文字)です。RS に複数の文字が含まれている場合、結果は不確定です。RS が NULL の場合、レコードは一連の 1 つ以上の空白行で区切られます。先頭または末尾に空白行があっても、入力の先頭または最後に空のレコードが生成されることはありません。また、FS の値に関係なく、フィールド区切り文字は常に復帰改行文字になります。
RSTART	match 関数によって照合される文字列の開始位置。1 から順に番号が付いています。これは常に match 関数の戻り値と同等です。
SUBSEP	多次元配列用の添字区切り文字列。デフォルト値は \034 です。

/usr/xpg4/bin/awk 次の変数は、/usr/xpg4/bin/awk でのみサポートされます。

CONVFMT 数字を文字列に変換するための printf 形式 (OFMT が使用される出力文を除く)。デフォルトは %.6g です。

正規表現 /usr/xpg4/bin/nawk ユーティリティーは、拡張正規表現 (ERE) ([regex\(5\)](#) を参照) を利用しています。ただし、ERE 内で特殊文字をエスケープするときは C 言語規約を使用できます。つまり、\\、\a、\b、\f、\n、\r、\t、\v、および次の表に示したエスケープシーケンスを使用できます。これらのエスケープシーケンスは、括弧の内でも外でも認識されます。なお、レコードは復帰改行文字で区切る必要がなく、文字列定数には復帰改行文字を含めることができるため、nawk ERE では \n シーケンスも有効です。正規表現内にスラッシュ文字を使用する場合は、次の表に示すようにエスケープ処理を行う必要があります。

エスケープシーケンス	説明	意味
\"	バックスラッシュ引用符	引用符文字
\/	バックスラッシュスラッシュ	スラッシュ文字

<code>\ddd</code>	バックスラッシュ文字のあとに1から3桁の8進文字(01234567)の最長シーケンスが続きます。すべての桁が0である(つまり、NULL文字を表している)場合、動作は未定義です。	1から3桁の8進整数で文字がエンコードされます。複数バイト文字には、バイトごとに先行\を含む、複数の連結されたエスケープシーケンスが必要です。
<code>\ c</code>	バックスラッシュ文字のあとにこの表で説明されていない文字または特殊文字が続きます(\、\a、\b、\f、\n、\r、\t、\v)。	未定義

2つの正規表現照合演算子`~`および`!`のいずれかを使用して、正規表現を特定のフィールドまたは文字列と照合できます。これらの演算子は、オペランドの右側を正規表現として解釈し、オペランドの左側を文字列として解釈します。正規表現が文字列と一致すると、`~`式は値1に評価され、`!`式は値0に評価されます。正規表現が文字列と一致しないと、`~`式は値0に評価され、`!`式は値1に評価されます。右側のオペランドが字句トークンERE以外の式であれば、前述のエスケープ規約も含め、式の文字列値は拡張正規表現として解釈されます。文字列リテラル(字句トークンSTRING)の値を判断するときも、これと同じエスケープ規約が適用され、この状況で文字列リテラルが使用されるときにもう一度適用されます。

EREトークンが、`~`または`!`演算子の右辺以外で使われている場合、または後述の組み込み関数の引数の1つとして使われている場合、結果の式の値は次のものと等価になります。

`$0 ~ /ere/`

`gsub`、`match`、`sub`関数の`ere`引数と、`split`関数の`fs`引数(「文字列関数」を参照)は、拡張正規表現として解釈されます。これらの引数はEREトークンまたは任意の式のどちらでもよく、`~`または`!`演算子の右辺と同じ方法で解釈されます。

`-FERE`オプションを使うか、式を含んだ文字列を組み込み変数FSに割り当てることで、フィールドの分割に拡張正規表現を使用できます。FS変数のデフォルト値は1つのスペース文字です。FSの動作について次に説明します。

1. FSが1つの文字の場合

- FSがスペース文字であれば、先行および末尾のスペース文字をスキップします。フィールドは1つ以上のスペース文字で区切られます。
- FSが空白以外の文字`c`であれば、フィールドは個々の`c`によって区切られません。

2. それ以外の場合、FSの文字列値は拡張正規表現と見なされます。拡張正規表現と一致するシーケンスの発生により、フィールドが区切られます。

`gsub`、`match`、`split`、および`sub`組み込み関数の場合を除き、正規表現照合は入力レコードに基づいて行われます。つまり、レコード区切り文字(変数RSの値の最初の文字、デフォルトは復帰改行文字)は式に組み込まず、レコード区切り文字と一

致する式はありません。レコード区切り文字が復帰改行文字以外であれば、式に組み込まれた復帰改行文字を照合させることができます。この4つの組み込み関数では、正規表現照合がテキスト文字列に基づいて行われます。そのため、どの文字(復帰改行文字やレコード区切り文字を含む)もパターンに組み込み、適切なパターンを任意の文字と照合させることができます。しかし、`nawk`のすべての正規表現照合では、パターン、入力レコード、またはテキスト文字列に1つ以上のNULL文字を使用すると、結果が未定義になります。

- パターン** パターンは、有効な式、コンマで区切られた2つの式で指定された範囲、または2つの特殊パターン `BEGIN` と `END` のいずれかです。
- 特殊パターン** `nawk` ユーティリティでは、2つの特殊パターン `BEGIN` と `END` を認識します。各 `BEGIN` パターンは一度照合され、最初の入力レコードが読み取られる前(以前の `BEGIN` アクションで `getline` 関数が使用された場合を除く)とコマンド行割り当てが行われる前に、関連付けられたアクションが実行されます。各 `END` パターンは一度照合され、最後の入力レコードが読み取られたあとに、関連付けられたアクションが実行されます。この2つのパターンにはアクションが関連付けられています。
- `BEGIN` と `END` は、他のパターンとは組み合わせられません。複数の `BEGIN` および `END` パターンを使用できます。`BEGIN` パターンに関連付けられたアクションは、プログラム内に指定された順序で実行されます。これは `END` アクションの場合も同じです。プログラム内では、`END` パターンが `BEGIN` パターンより前にあってもかまいません。
- `nawk` プログラムが `BEGIN` パターンのアクションだけで構成されていて、`BEGIN` アクションに `getline` 関数が含まれていない場合、最後の `BEGIN` アクションの最後の文が実行されると、`nawk` は入力を読み取らずに終了します。`nawk` プログラムが `END` パターンのアクションだけ、または `BEGIN` および `END` パターンのアクションだけで構成されている場合、`END` アクションの文が実行される前に入力を読み取られます。
- 式パターン** 式パターンは、ブール型のコンテキストの式として評価されます。結果が真の場合、パターンは一致すると見なされ、関連付けられたアクションがあればそれが実行されます。結果が偽の場合、アクションは実行されません。
- パターンの範囲** パターンの範囲は、コンマで区切られた2つの式で構成されます。この場合、最初の式の一致から2番目の式の一致までの間にあるすべてのレコードに対してアクションが実行されます。この時点で、入力レコードの先頭から一致した範囲の最後まで、パターンの範囲を繰り返すことができます。
- アクション** アクションは一連の文です。使用できる文は次のうちのいずれかです。

```
if ( expression ) statement [ else statement ]
while ( expression ) statement
do statement while ( expression )
for ( expression ; expression ; expression ) statement
for ( var in array ) statement
delete array[subscript] #delete an array element
```



```

break
continue
{ [ statement ] . . . }
expression          # commonly variable = expression
print [ expression-list ] [ >expression ]
printf format [ ,expression-list ] [ >expression ]
next                  # skip remaining patterns on this input line
exit [expr] # skip the rest of the input; exit status is expr
return [expr]

```

1つの文を、括弧で囲まれた文のリストに置き換えることができます。文は、復帰改行文字またはセミコロンで区切られ、指定された順序で実行されます。

next 文は、現在の入力レコードの以降の処理をすべて破棄します。BEGIN または END アクション内で *next* が現れたり、呼び出されたりすると、動作は未定義です。

exit 文は、すべての END アクションをプログラムソース内の順序で呼び出した後、以降の入力を読み取らずにプログラムを終了させます。END アクション内の *exit* 文は、以降の END アクションを実行せずにプログラムを終了させます。*exit* 文内に式が指定されている場合は、その数値が *nawk* の終了ステータスになります。ただし、エラーが検出されたり、式を含んだ *exit* 文が実行される場合を除きます。

出力文

print 文も *printf* 文もデフォルトでは標準出力に書き込みます。次のように、*output_redirection* が指定されていれば、その宛先に出力が書き込まれます。

```
> expression>> expression | expression
```

どの場合でも、*expression* が評価されて、書き込み先のフルパス名 (> または >> の場合) または実行されるコマンド (| の場合) として使用される文字列が生成されます。最初の2つの形式を使うと、その名前のファイルが現在開いていなければ開かれ、必要ならファイルが作成されます。最初の形式を使うと、ファイルは切り捨てられます。次に、出力がファイルに付加されます。ファイルが開いているかぎり、式が同じ文字列値に評価される以降の呼び出しは、単に出力をファイルに付加していきます。同じ文字列値に評価される式で *close* 関数が呼び出されるまで、ファイルは開いたままです。

3番目の形式は、コマンドの入力にパイプされたストリームに出力を書き込みます。コマンド名として *expression* の値を持つストリームが現在開いていない場合は、ストリームが作成されます。作成されたストリームは、*command* 引数として *expression* の値、*mode* 引数として *w* の値を指定した [popen\(3C\)](#) 関数の呼び出しによって作成されるものと同等です。ストリームが開いているかぎり、*expression* が同じ文字列値に評価される以降の呼び出しは、出力を既存のストリームに書き込みます。同じ文字列値に評価される式で *close* 関数が呼び出されるまで、ストリームは開いたままです。*close* 関数が呼び出されると、*pclose* 関数が呼び出されたかのようにストリームが閉じます。

これらの出力文は、文法において非終端記号 `expr_list`、`print_expr_list`、または `print_expr_list_opt` によって参照される、コンマで区切られた *expressions* のリストを取ります。このリストはここでは式リストとして参照され、各メンバーは式引数として参照されます。

`print` 文は、現在の出力フィールド区切り文字 (前述の変数 `OFS` を参照) で区切られ、出力レコード区切り文字 (前述の変数 `ORS` を参照) で終わる、指定された出力ストリームに各式引数の値を書き込みます。式引数はすべて文字列として扱われ、必要があれば変換されます。ただし、`CONVFMT` の値ではなく、`OFMT` の `printf` 形式が使用されます。空の式リストは、入力レコード全体 (`$0`) を意味します。

`printf` 文は、本書でファイル形式の説明に使用しているファイル形式表記と同様の表記に基づいて出力を生成します。出力は、文字列 `format` として最初の式引数、`arg1` から `argn` までの文字列として後続の式引数で指定されたとおりに生成されます。ただし、次のような例外があります。

1. *format* はグラフィック表現ではなく、実際の文字列です。そのため、空の文字位置を含めることはできません。*format* 文字列内のスペース文字は、変換仕様の *flag* 以外のコンテキストでは、通常の文字として扱われ、出力にコピーされます。
2. 文字セットにデルタ文字が含まれていて、その文字が *format* 文字列に現れた場合は、通常の文字として扱われ、出力にコピーされます。
3. バックスラッシュ文字で始まる *escape sequences* は、通常の文字のシーケンスとして扱われ、出力にコピーされます。なお、このようなシーケンスは、リテラル文字列に現れたときに `nawk` によって字句が解析されますが、`printf` 文では特に処理されません。
4. *field width* または *precision* は、数値列ではなく * 文字として指定できます。この場合、式リストの次の引数を取り出され、その数値がフィールド幅または精度と見なされます。
5. システムでは、*format* 文字列で指定されていない空白文字を `d` または `u` 変換仕様の出力の先頭や末尾には付けません。
6. システムでは、*format* 文字列で指定されていない先行ゼロを `o` 変換仕様の出力には付けません。
7. `c` 変換仕様の場合、引数が数値であれば、エンコーディングとしてその値を持つ文字が出力されます。値がゼロの場合、または文字セットのどの文字にもエンコードされていない場合、動作は未定義です。引数が数値を持たない場合、文字列値の最初の文字が出力されます。文字列に文字が含まれていない場合、動作は未定義です。
8. 引数が必要な変換仕様のそれぞれについて、次の式引数が評価されます。`c` 変換以外は、変換仕様に応じて値が適切な型に変換されます。
9. *format* 文字列内のすべての変換仕様を満たすだけの十分な式引数がない場合、動作は未定義です。

10. *format* 文字列内の文字シーケンスが % 文字で始まるが、有効な変換仕様を形成していない場合、動作は未定義です。

`print` と `printf` はどちらも、少なくとも `{LINE_MAX}` バイトを出力できます。

関数 `nawk` 言語は、算術関数、文字列関数、入出力関数、一般関数など、さまざまな組み込み関数を備えています。

算術関数 `int` を除き、算術関数は ISO C 標準に基づいています。エラーが返される、または動作が未定義であると ISO C 標準で指定している場合、動作は未定義です。文法上は引数や括弧のない組み込み関数が認められていますが、次のリストで引数や括弧が省略可能 ([] 括弧で表示) と示されていないかぎり、省略した場合の結果は未定義です。

<code>atan2(y,x)</code>	<code>y/x</code> の逆正接を返します。
<code>cos(x)</code>	<code>x</code> をラジアン単位とし、 <code>x</code> の余弦を返します。
<code>sin(x)</code>	<code>x</code> をラジアン単位とし、 <code>x</code> の正弦を返します。
<code>exp(x)</code>	<code>x</code> の指数関数を返します。
<code>log(x)</code>	<code>x</code> の自然対数を返します。
<code>sqrt(x)</code>	<code>x</code> の平方根を返します。
<code>int(x)</code>	引数を切り捨てて、整数にします。つまり <code>x</code> が 0 よりも大きい場合は、0 に向かって切り捨てます。
<code>rand()</code>	$0 \leq n < 1$ の範囲の乱数 <code>n</code> を返します。
<code>srand([expr])</code>	<code>rand</code> のシード値を <code>expr</code> に設定します。 <code>expr</code> が省略された場合は、時間を使用します。直前のシード値が返されます。

文字列関数 次のリストの文字列関数がサポートされています。文法上は引数や括弧のない組み込み関数が認められていますが、次のリストで引数や括弧が省略可能 ([] 括弧で表示) と示されていないかぎり、省略した場合の結果は未定義です。

<code>gsub(ere,repl[, in])</code>	後述の <code>sub</code> と同様の機能ですが、相違点は (ed ユーティリティーの一括置換のように) 指定されたときに、 <code>\$0</code> または <code>in</code> 引数内に出現する正規表現をすべて置換することです。
<code>index(s,t)</code>	文字列 <code>s</code> の中で文字列 <code>t</code> が最初に出現する位置 (最初の文字を 1 として文字数を数えたもの) を返します。出現しない場合はゼロを返します。
<code>length([([s])])</code>	引数を文字列として解釈し、その長さ (文字数) を返します。引数が指定されていない場合は、レコード <code>\$0</code> 全体の長さを返します。

<code>match(s,ere)</code>	文字列 <i>s</i> の中で拡張正規表現 <i>ere</i> が出現する位置 (最初の文字を 1 として文字数を数えたもの) を返します。出現しない場合はゼロを返します。RSTART は開始位置 (返される値と同じ) に設定され、一致しない場合はゼロに設定されます。RLENGTH は一致した文字列の長さに設定され、一致しない場合は -1 に設定されます。
<code>split(s,a[,fs])</code>	文字列 <i>s</i> を配列要素 <i>a</i> [1]、 <i>a</i> [2]、...、 <i>a</i> [<i>n</i>] に分割し、値 <i>n</i> を返します。この分割は、拡張正規表現 <i>fs</i> によって行われ、 <i>fs</i> が指定されていない場合はフィールドセパレータ FS によって行われます。各配列要素は、作成されたときに文字列値を持ちます。配列要素に割り当てられた文字列が、現在のロケールの小数点文字がピリオド文字に変更されていて、数値列と見なされる場合は、配列要素もその数値列の数値を持つことになります。 <i>fs</i> の値が NULL 文字列の場合、結果は不確定です。
<code>sprintf(fmt,expr,expr,...)</code>	<i>fmt</i> で指定した printf 形式に従って式を書式設定し、結果として得られた文字列を返します。
<code>sub(ere,repl[, in])</code>	文字列 <i>in</i> 内で最初に見つかった拡張正規表現 <i>ERE</i> のインスタンスを文字列 <i>repl</i> に置き換え、置換した数を返します。文字列 <i>repl</i> 内に出現するアンパサンド (&) は、正規表現に一致する <i>in</i> 内の文字列に置換されます。アンパサンドの前にバックスラッシュ (\) を付けると、リテラルのアンパサンド文字として解釈されます。バックスラッシュが連続で 2 つ現れると、単一のリテラルバックスラッシュ文字として解釈されます。そのほかに現れるバックスラッシュ (ほかの文字の前など) は、リテラルバックスラッシュ文字として処理されます。 <i>repl</i> が文字列リテラルの場合、アンパサンド文字の処理は字句処理 (バックスラッシュによるエスケープシーケンスの字句処理など) のあとに発生します。 <i>in</i> を指定し、それが <i>lvalue</i> 以外の場合、動作は未定義です。 <i>in</i> を省略した場合、nawk では代わりに現在のレコード (\$0) が使用されます。
<code>substr(s,m[, n])</code>	文字列 <i>s</i> 内の <i>m</i> 文字目から始まる最大 <i>n</i> 文字の長さの部分文字列を返します。 <i>n</i> がない場合、部分文字列の長さは文字列 <i>s</i> の長さによって制限されます。
<code>tolower(s)</code>	文字列 <i>s</i> に基づいて文字列を返します。現在のロケールの LC_CTYPE カテゴリによって tolower マッピングを持つように指定されている <i>s</i> 内の各大文字が、そ

のマッピングで指定されている小文字に置換されます。*s*内のほかの文字は変更されません。

`toupper(s)`

文字列*s*に基づいて文字列を返します。現在のロケールの LC_CTYPE カテゴリによって `toupper` マッピングを持つように指定されている *s* 内の各小文字が、そのマッピングで指定されている大文字に置換されます。*s*内のほかの文字は変更されません。

パラメータとして *ERE* を取る前述の関数はすべて、次に定義されているように、正規表現であるパターン値または文字列値の式を受け付けます。

入出力関数と一般関数

入出力関数と一般関数は次のとおりです。

`close(expression)`

`print` または `printf` 文で開かれたファイルまたはパイプ、あるいは同じ文字列値の *expression* を指定した `getline` 呼び出しで開かれたファイルまたはパイプを閉じます。正常に閉じられた場合、関数は `0` を返します。それ以外の場合は、ゼロ以外を返します。

`expression|getline[var]`

コマンドの出力からパイプされたストリームから、入力レコードを読み取ります。コマンド名として *expression* の値を持つストリームが現在開いていない場合は、ストリームが作成されます。作成されたストリームは、`command` 引数として *expression* の値、`mode` 引数として *r* の値を指定した `popen` 関数の呼び出しによって作成されるものと同等です。ストリームが開いているかぎり、*expression* が同じ文字列値に評価される以降の呼び出しは、ファイルから後続のレコードを読み取ります。同じ文字列値に評価される式で `close` 関数が呼び出されるまで、ストリームは開いたままです。`close` 関数が呼び出されると、`pclose` 関数が呼び出されたかのようにストリームが閉じます。*var* がいない場合、`$0` および `NF` が設定されます。それ以外の場合、*var* が設定されます。

| の左側 (`getline` を含む式の先頭まで) に括弧で囲まれている演算子 (連結を含む) がある場合、`getline` 演算子の構造があいまいになる可能性があります。演算子 `$` のコンテキストでは、| は `$` よりも優先度が低いかにように動作します。その他の演算子を評価した結果は不確定であるため、移植性のあるアプリケーションでは、このような表記をすべて正しく括弧で囲む必要があります。

`getline`

`$0` に、現入力ファイルの次の入力レコードを設定します。この形式の `getline` では、`NF`、`NR`、および `FNR` 変数を設定します。

<code>getline var</code>	変数 <i>var</i> に、現在の入力ファイルの次のレコードを設定します。この形式の <code>getline</code> では、 <code>FNR</code> および <code>NR</code> 変数を設定します。
<code>getline [var] < expression</code>	指定されたファイルから次の入力レコードを読み取ります。 <i>expression</i> が評価されて、フルパス名として使用される文字列が生成されます。その名前のファイルが現在開いていない場合は開きます。ストリームが開いているかぎり、 <i>expression</i> が同じ文字列値に評価される以降の呼び出しは、ファイルから後続のレコードを読み取ります。同じ文字列値に評価される式で <code>close</code> 関数が呼び出されるまで、ファイルは開いたままです。 <i>var</i> がいない場合、 <code>\$0</code> および <code>NF</code> が設定されます。それ以外の場合、 <i>var</i> が設定されます。 < の右側 (<code>getline</code> を含む式の終わりまで) に括弧で囲まれていない 2 項演算子 (連結を含む) がある場合、 <code>getline</code> 演算子の構造があいまいになる可能性があります。このような構造を評価した結果は不確定であるため、移植性のあるアプリケーションでは、このような表記をすべて正しく括弧で囲む必要があります。
<code>system(expression)</code>	<i>expression</i> で指定されたコマンドを、 <code>system(3C)</code> 関数と同等の方法で実行し、コマンドの終了ステータスを返します。

`getline` 関数はどの形式でも、正常終了時には 1、ファイルの終わりに達すると 0、エラーが発生時には -1 を返します。

ファイルやパイプラインの名前として文字列が使用されている場所では、文字列のテキストをまったく同じにする必要があります。「同じ文字列値」とは「等価文字列」を意味し、スペース文字が違っただけでも異なるファイルを表すことになります。

ユーザー定義関数 `nawk` 言語ではユーザー定義関数も使用できます。ユーザー定義関数は次のように定義できます。

```
function name(args, . . .) { statements }
```

関数は、`nawk` プログラム内のどこからでも参照でき、特に、その定義より前に使用できます。関数の有効範囲はグローバルです。

関数の引数は、スカラーまたは配列のどちらでもかまいません。スカラーを使用する関数に引数として配列名が渡された場合、または配列を使用する関数に引数としてスカラー式が渡された場合、その動作は未定義です。関数の引数は、スカラーの場合は値によって、配列名の場合は参照によって渡されます。引数名は関数に対し

てローカルであり、その他のすべての変数名はグローバルです。引数名と関数名または特殊な `nawk` 変数名に同じ名前は使用されません。グローバルな有効範囲を持つ変数名と関数名に同じ名前を使用しないでください。同じ有効範囲内では、スカラー変数と配列に同じ名前を使用しないでください。

関数定義内のパラメータの数は、関数呼び出しのときのパラメータの数と一致する必要はありません。ローカル変数として、余分な仮パラメータを使用できます。関数定義内の引数よりも少ない引数が関数呼び出しの際に指定された場合、関数本体でスカラーとして使用されている余分なパラメータは、NULL 文字列の文字列値とゼロの数値で初期化されます。また、関数本体で配列として使用されている余分なパラメータは、空の配列として初期化されます。関数定義内の引数よりも多い引数が関数呼び出しの際に指定された場合、その動作は未定義です。

関数の呼び出し時に、関数名と左括弧の間にスペースを入れることはできません。関数呼び出しは入れ子にすることができ、再帰呼び出しも可能です。入れ子または再帰関数呼び出しから戻っても、参照によって渡された配列パラメータ以外、呼び出し側関数のパラメータのどの値も変わりません。`return` 文を使用すれば、値を返すことができます。`return` 文が関数定義の外で使用された場合、その動作は未定義です。

関数定義において、左括弧より前と右括弧よりあとの復帰改行文字は任意指定です。関数定義は、パターンとアクションのペアが許可されている場所であればプログラム内のどこに置いてかまいません。

使用法

`index`、`length`、`match`、および `substr` 関数と ISO C 標準の同様の関数とを混同しないようにしてください。`nawk` バージョンでは文字を扱い、ISO C 標準ではバイトを扱います。

連結演算は、明示的な演算子ではなく隣接する式によって表現されるので、評価の正しい優先度を適用するため、括弧を使う必要があることがよくあります。

ファイルが 2G バイト (2^{31} バイト) 以上ある場合の `nawk` の動作については、[largefile\(5\)](#) を参照してください。

使用例

`sh` を使用するアプリケーションの場合、コマンド行に指定する `nawk` プログラムは、単一引用符で囲んで指定する (たとえば、`'program'`) のがもっとも簡単な方法です。`nawk` プログラムには一般に、二重引用符をはじめとする、シェルに対して特殊な意味を持つ文字が含まれていることが多いためです。`nawk` プログラムに単一引用符文字が含まれている場合、プログラムの大部分を単一引用符で囲んで文字列として指定し、引用符付きの単一引用符文字とシェルによって連結するのが通常もっとも簡単な方法です。例:

```
nawk '/"/ { print "quote:", $0 }'
```

これは、標準入力から読み取った単一引用符文字を含むすべての行に `quote:` という接頭辞を付けて出力します。

簡単な nawk プログラムの例を次に示します。

例1 第3フィールドの値が5より大きい入力行をすべて標準出力に書き出します。

```
$3 > 5
```

例2 10行ごとに書き出します。

```
(NR % 10) == 0
```

例3 部分文字列が正規表現と一致する行を書き出します。

```
/(G|D)(2[0-9][[:alpha:]]*)/
```

例4 部分文字列として、文字GまたはDのあとに一連の数字と文字を伴う行を出力します。
この例では、文字クラス digit と alpha を使って、言語に依存しない数字と英字をそれぞれ照合します。

```
/(G|D)([[:digit:]][[:alpha:]]*)/
```

例5 第2フィールドは正規表現と一致するが第4フィールドは一致しない、という行を出力します。

```
$2 ~ /xyz/ && $4 !~ /xyz/
```

例6 第2フィールドにバックスラッシュが含まれている行を出力します。

```
$2 ~ /\
```

例7 第2フィールドにバックスラッシュが含まれている行を出力します (別の方法)。

ここではバックスラッシュによるエスケープを2度解釈しています。1度は文字列の字句処理時、もう1度は正規表現の処理時です。

```
$2 ~ "\\\"
```

例8 各行の最後から2番目のフィールドと最後のフィールドをコロンで区切って出力します。

```
{OFS=":";print $(NF-1), $NF}
```

例9 各行の行番号とフィールドの数を出力します。

行番号、コロン、フィールド数を表す3つの文字列が連結され、1つの文字列として標準出力に書き込まれます。

```
{print NR ":" NF}
```

例10 長さが72文字を超えている行を出力します。

```
{length($0) > 72}
```

例11 最初の2つのフィールドを逆の順序で、OFSで区切って出力します。

```
{ print $2, $1 }
```

例12 前述と同様ですが、入力フィールドはコンマ、またはスペース文字とタブ文字、あるいはその両方で区切られています。

```
BEGIN { FS = ",[\\t]*|[\\t]+" }  
      { print $2, $1 }
```

例13 先頭のカラムの値を合計し、その合計値と平均値を出力します。

```
{s += $1 }  
END {print "sum is ", s, " average is", s/NR}
```

例14 フィールドの内容を逆の順序で、1行につき1フィールドずつ出力します。1入力行につき多数の出力行が発生します。

```
{ for (i = NF; i > 0; --i) print $i }
```

例15 「start」と「stop」という2つの文字列の間にあるすべての行を出力します。

```
/start/, /stop/
```

例16 先頭フィールドの値が直前の行の先頭フィールドと異なっている行をすべて出力します。

```
$1 != prev { print; prev = $1 }
```

例17 echo コマンドの動作をシミュレートします。

```
BEGIN {  
    for (i = 1; i < ARGV; ++i)  
        printf "%s%s", ARGV[i], i==ARGC-1?"\n":""  
    }  
}
```

例18 PATH 環境変数に含まれているパス接頭辞を1行につき1つずつ出力します。

```
BEGIN {  
    n = split (ENVIRON["PATH"], path, ":")  
    for (i = 1; i <= n; ++i)  
        print path[i]  
    }  
}
```

例19 「input」ファイルの内容を、ページ番号5を開始ページとして出力します。

input という名前のファイルがあり、次の形式のページヘッダーが含まれているとします。

Page#

また、program という名前のファイルの内容は次のとおりとします。

例 19 「input」 ファイルの内容を、ページ番号 5 を開始ページとして出力します。 (続き)

```
/Page/{ $2 = n++; }
{ print }
```

ここで、次に示すコマンドを実行します。

```
nawk -f program n=5 input
```

これにより、input ファイルが、ページ番号 5 を開始ページとして出力されます。

環境変数

実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

LC_NUMERIC 数値入力の解釈、数値と文字列との変換、数値出力のフォーマットに用いる、小数点文字を決定します。awk プログラム (コマンド行引数で指定される代入も含む) の処理で認識される小数点文字は、ロケールに関係なくピリオド (POSIX ロケールの小数点文字) です。

終了ステータス

次の終了ステータスが返されます。

- 0 指定されたファイルはすべて正常に処理されました。
- >0 エラーが発生した。

プログラム内で exit 式を使って終了ステータスを変更することができます。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/nawk

属性タイプ	属性値
使用条件	system/core-os

/usr/xpg4/bin/awk

属性タイプ	属性値
使用条件	system/xopen/xcu4

関連項目

[awk\(1\)](#)、[ed\(1\)](#)、[egrep\(1\)](#)、[grep\(1\)](#)、[lex\(1\)](#)、[sed\(1\)](#)、[popen\(3C\)](#)、[printf\(3C\)](#)、[system\(3C\)](#)、[at](#)

Aho, A.V., B. W. Kernighan, P. J. Weinberger 共著、『The AWK Programming Language』、Addison-Wesley 発行、1988 年。

診断

file オペランドが指定され、指定されたファイルにアクセスできない場合、nawk は標準エラーに診断メッセージを書き込んだあと、それ以上何もせずに終了します。

program オペランドまたは proffile オペランドで指定されたプログラムが有効な nawk プログラム (「拡張機能説明」で指定されている) でない場合、その動作は未定義です。

注意事項

入力行にスペースが含まれる場合、出力時に保証されません。

数値と文字の間の明示的な変換は行われません。式が数値として扱われるようにするには、式に 0 を加算します。式が文字列として扱われるようにするには、式に NULL 文字列 ("") を連結します。

名前	nc, netcat – 任意の TCP および UDP の接続と待機
形式	<pre>nc -h nc [-46dnrtuvz] [-i interval] [-P proxy_username] [-p port] [-s source_ip_address] [-T dspc] [-w timeout] [-X proxy_protocol] [-x proxy_address[:port]] [-L timeout] [-e program] [-b bufsize] [-q timeout] [-m bytes] [-I bufsize] [-O bufsize] hostname port_list nc -l [-46DdEFnrtuvzZ] [-i interval] [-T dspc] [-e program] [-b bufsize] [-q timeout] [-R address/port[/proto]] [-m bytes] [-L timeout] [-I bufsize] [-O bufsize] [hostname] port nc -l [-46DEFdnrtuvzZ] [-i interval] [-T dspc] [-e program] [-b bufsize] [-q timeout] [-R address/port[/proto]] [-m bytes] [-L timeout] [-I bufsize] [-O bufsize] -p port [hostname] nc -U [-Ddtvz] [-i interval] [-w timeout] [-e program] [-b bufsize] [-q timeout] [-m bytes] path nc -Ul [-46DdktvZ] [-i interval] [-e program] [-b bufsize] [-q timeout] [-R address/port[/proto]] [-m bytes] path</pre>
機能説明	nc(または netcat) ユーティリティーは、TCP または UDP に関連付けられたさまざまなタスクに使用できます。nc は、TCP 接続を開き、UDP パケットを送信し、任意の TCP および UDP ポートで待機し、ポートスキャンを実行し、IPv4 と IPv6 の両方に対応します。telnet(1) とは異なり、nc は簡単にスクリプト化でき、エラーメッセージを標準出力に送信するのではなく標準エラーに分割します。 nc コマンドは、多くの場合、次のタスクに使用されます。 ■ 単純な TCP プロキシ ■ シェルスクリプトベースの HTTP クライアントおよびサーバー ■ ネットワークデーモンのテスト ■ ssh(1) 用の SOCKS またはプロキシコマンド (ProxyCommand) nc コマンドは、netcat として実行して同じオプションを使用することもできます。
オプション	サポートしているオプションは、次のとおりです。 -4 nc が IPv4 アドレスのみを使用するよう強制します。 -6 nc が IPv6 アドレスのみを使用するよう強制します。 -b <i>bufsize</i> 読み取り操作のバッファサイズを指定します。 デフォルト値は 1024 バイトです。

- D
ソケット上でのデバッグを有効にします。
- d
stdinからの読み取りを行いません。
- E
排他バインドを使用して、TCP または UDP ソケットを待機します。

-l オプションを指定せずにこのオプションを使用すると、エラーになります。

このオプションは、-u オプションと同時に使用しても何の効果もありません。
- e *program*
接続の受け入れまたは実行後に外部プログラムを実行します。実行前に、stdin, stdout, stderr がネットワーク記述子にリダイレクトされます。このオプションでは、1つのポートのみを使用できます。

このオプションを -R、-k、または -i オプションと同時に使用すると、エラーになります。
- F
stdin で EOF を表示したあと、書き込みのためにネットワークソケットを閉じないでください。
- h
nc のヘルプを表示します。
- I *bufsize*
受信する (入力) ソケットのバッファサイズを設定します。

このオプションは、-u オプションと同時に使用しても何の効果もありません。
- i *interval*
テキスト行の送受信間隔 (*interval*) の遅延時間を指定します。

間隔は適切な小数で指定します (秒単位)。

このオプションを使用すると複数のポートへの接続で遅延が発生するため、ポートスキャンモードにも影響が及びます。
- k
現在の接続が閉じられたら、nc が別の接続を待機するよう強制します。

-l オプションを指定せずにこのオプションを使用すると、エラーになります。

このオプションを -e オプションと同時に使用すると、エラーになります。
- L *timeout*
閉じる操作を遅らせます。ネットワーク記述子が閉じられたあとにメッセージが送信されるのを、指定した時間 (秒) だけ待機します。

-l

リモートホストへの接続を開始せずに、着信する接続を待機します。

このオプションを -s または -z オプションと同時に使用すると、エラーになります。

-l オプションをワイルドカードソケット (IP アドレスとホスト名の指定はなし) とともに使用し、-4/-6 オプションを指定しない場合は、IPv4 接続と IPv6 接続の両方が受け入れられます。

-m *byte_count*

少なくとも *byte_count* バイトを受信したあとに終了します。-l オプションとともに使用すると、*byte_count* はクライアントから受信したバイト数と比較されます。

byte_count は 0 より大きく INT_MAX より小さい値である必要があります。

-N *file*

ファイルをパターンで指定して UDP ポートをスキャンします。このファイルの内容は、放出された各 UDP パケットのペイロードとして使用されます。

-u オプションと -z オプションを指定せずにこのオプションを使用すると、エラーになります。

-n

アドレス、ホスト名、またはポートに対して名前およびサービスの検索を行いません。

このオプションを使用すると、*hostname* 引数と *port* 引数は数値に制限されます。

-v オプションとともに使用すると、引数に対する制限に加えて、すべてのアドレスとポートが数値形式で出力されます。このオプションは、-u オプションと同時に使用しても何の効果もありません。

-O *bufsize*

送信 (出力) ソケットのバッファサイズを設定します。

このオプションは、-u オプションと同時に使用しても何の効果もありません。

-P *proxy_username*

認証を要求するプロキシサーバーに渡されるユーザー名 (*proxy_username*) を指定します。*proxy_username* を指定しない場合、認証は行われません。現時点では、HTTP CONNECT プロキシのプロキシ認証のみがサポートされています。

このオプションを -l オプションと同時に使用すると、エラーになります。

-p *port*

-l オプションを指定せずに使用する場合は、nc が使用する発信元ポートを特権の制限と使用条件に従って指定します。-l オプションと同時に使用する場合は、待機ポートを設定します。

このオプションと -l オプションの組み合わせは、グローバルなポート引数が指定されていない場合にのみ使用できます。

-q *timeout*

stdin で EOF を受信したあとに、指定した秒数待機してから終了します。

-R *addr/port[/proto]*

指定した *host* と *port* へのポートリダイレクトを実行します。

接続が受け入れられると、nc はリモートの *host/port* に接続し、クライアントとリモートホスト間ですべてのデータを受け渡しします。リダイレクト指定の *proto* (プロトコル) 部分は、tcp と udp のどちらかにできます。*proto* が指定されていない場合、redirector はサーバーと同じプロトコルを使用します。

このオプションを -z オプションと同時に使用すると、エラーになります。

-r

port_list 引数で指定されたすべてのポートの中から着信先ポートを (順番にではなく) ランダムに選択します。

このオプションを -l オプションと同時に使用すると、エラーになります。

-s *source_ip_address*

パケットの送信に使用するインタフェースの IP を指定します。

このオプションを -l オプションと同時に使用すると、エラーになります。

-T *dscp*

接続用の差別化サービスコードポイントを指定します。

IPv4 の場合、IP Type of Service (ToS) IP ヘッダーフィールドが指定されます。この引数の有効な値は文字列トークンで、lowdelay、throughput、reliability、または 0x の後に 8 ビットの 16 進数値が付いた値になります。

IPv6 (トラフィッククラス) の場合、16 進数値のみが使用できます。

-t

RFC 854 の DO および WILL 要求に対して、nc が RFC 854 の DON'T および WON'T 応答を送信します。これにより、nc を使用して telnet セッションをスクリプト化できるようになります。

-U

UNIX ドメインソケットの使用を指定します。-l を指定せずにこのオプションを指定すると、nc は AF_UNIX クライアントになります。このオプションを -l オプションと同時に指定すると、AF_UNIX サーバーが作成されます。

このオプションを使用する場合は、ホスト名やポートではなく、有効な UNIX ドメインパスからなる単独の引数を nc に指定する必要があります。

-u

デフォルトのオプションである TCP の代わりに UDP を使用します。

-v

詳細出力を指定します。

-w *timeout*

接続と stdin のアイドル状態が *timeout* 秒を超えた場合に、メッセージを表示せずに接続を閉じます。

デフォルトはタイムアウトなしです。

クライアントモードで接続確立フェーズにある場合、またはサーバーモードで接続を待機している場合、このオプションの効果はありません。

-X *proxy_protocol*

プロキシサーバーとの通信時に、指定したプロトコルを使用します。サポートされるプロトコルは、4 (SOCKS v.4)、5 (SOCKS v.5)、および connect (HTTP プロキシ) です。プロトコルを指定しなかった場合は、SOCKS v. 5 が使用されます。

このオプションを -l オプションと同時に使用すると、エラーになります。

-x *proxy_address[:port]*

proxy_address および *port* で指定されたプロキシを使用して、*hostname* への接続を要求します。*port* を指定しなかった場合は、プロキシプロトコルのウェルノウンポート (SOCKS の場合は 1080、HTTP の場合は 3128) が使用されます。

このオプションを -l オプションと同時に使用すると、エラーになります。

このオプションは、IPv6 アドレスの数値表現では機能しません。

-Z

待機モードで、SO_ALLZONES ソケットオプションを使用して、すべてのゾーンでアドレス/ポートにバインドします。

このオプションでは、SYS_NET_CONFIG 権限が必要です。

-Z

ポートスキャンを実行します。TCP ポート (デフォルト) の場合、データが送信されずに接続スキャン (3 方向のフルハンドシェイク) が試行されます。UDP (-u) の場合、空の UDP パケットがデフォルトで送信されます。UDP ペイロードを指定するには、-N オプションを使用できます。

UDP スキャンモードでは、否定応答 (ICMP 着信先ポート未達メッセージ) が受信されない場合は、ポートが開かれているものと推定します。このモードでは、-w オプションで設定したタイムアウトを使用して、リモートノードからの ICMP

メッセージまたはデータを待機します。-vを指定すると、すべての受信データが16進数バイト値としてstderrにダンプされます。

ほとんどのオペレーティングシステムが、入力パケットに対するICMPメッセージの送信に速度制限を課しているため、UDP スキャンを実行するときに-i オプションを使用しないと、結果が信頼性の低いものになります。

このオプションを-l オプションと同時に使用すると、エラーになります。

オペランド

次のオペランドがサポートされています。

hostname ホスト名を指定します。

hostname には、数値 IP アドレスまたは記号ホスト名 (-n オプションを指定していない場合のみ) を指定できます。

一般に、*hostname* を指定する必要があるのは、-l オプションを指定しない場合か、-u を使用しない場合です (その場合、この引数はパスになります)。*hostname* 引数を -l オプションを同時に指定した場合、*port* 引数も指定する必要があり、nc はそのアドレスとポートにバインドしようとし、*hostname* 引数を -l オプションと同時に指定しなかった場合、nc は指定されたポートのワイルドカードソケットで待機しようとし、

path パス名を指定します。

port

port_list ポートを指定します。

port_list には、1つの整数、整数の範囲、あるいはその両方の組み合わせを指定できます。範囲は *nn-mm* の形式で指定します。*port_list* には1つ以上のメンバーを指定する必要があり、複数のポートまたは範囲をコンマで区切って指定できます。

一般に、着信先ポートを指定する必要があるのは、-u オプションを指定しない場合です。そのオプションを指定した場合は、*hostname* の代わりに Unix ドメインソケットのパスを指定する必要があります。

複数のポートが含まれているポート一覧を -e オプションと同時に使用すると、エラーになります。

使用法

クライアント/サーバーモデル

nc を使用すると、非常に基本的なクライアント/サーバーモデルをきわめて簡単に構築できます。1つ目のコンソールで、特定のポートで接続を待機する nc を起動します。次に例を示します。

```
$ nc -l 1234
```

これは、ポート 1234 で接続を待機します。2 つ目のコンソール (または 2 つ目のマシン) で、nc が待機しているマシンとポートに接続します。

```
$ nc 127.0.0.1 1234
```

これで、ポート間の接続が確立されます。2 つ目のコンソールで入力した内容が、1 つ目のコンソールで入力した内容に連結されます (その逆も同様)。接続が設定されると、nc はどちら側がサーバーとして使用され、どちら側がクライアントとして使用されているかを区別しません。接続を終了するには EOF (Ctrl-D) を使用します。

データ転送

前のセクションの例を応用すると、基本的なデータ転送モデルを作成できます。接続の一方の側に入力された情報がもう一方の側に出力されるため、入力と出力を簡単に取り込んでファイル転送をエミュレートできます。

最初に、nc を使用して特定のポートで待機し、出力をファイルに取り込みます。

```
$ nc -l 1234 > filename.out
```

2 つ目のマシンを使用して、待機中の nc プロセスに接続し、転送するファイルを入力します。

```
$ nc host.example.com 1234 < filename.in
```

ファイルの転送が完了すると、接続は自動的に終了します。

サーバーとの通信

ユーザーインタフェースを使用せずに手動でサーバーと通信したほうが便利な場合があります。この方法は、トラブルシューティングで、クライアントが実行したコマンドに対応してサーバーがどのようなデータを送信しているのかを確認する必要があります。

たとえば、Web サイトのホームページを取得するには、次のコマンドを使用します。

```
$ echo -n "GET / HTTP/1.0\r\n\r\n" | nc host.example.com 80
```

これにより、Web サーバーが送信したヘッダーも表示されます。表示された内容は、[sed\(1\)](#) などのツールを使用すると、必要に応じてフィルタ処理できます。

サーバーが求める要求形式がわかっている場合は、より複雑な例を作成できます。もう 1 つの例として、電子メールを SMTP サーバーに送信するには、次を使用します。

```
$ nc localhost 25 << EOF
HELO host.example.com
MAIL FROM: <user@host.example.com>
RCTP TO: <user2@host.example.com>
DATA
Body of email.
.
```

```
QUIT
EOF
```

ポートスキャン

ターゲットマシン上のどのポートが開き、サービスを実行しているかがわかると便利な場合があります。-z フラグを使用すると、接続を開始せずに、開いているポートを報告するよう nc に指示できます。

上記の例では、

```
$ nc -z host.example.com 20-30
Connection to host.example.com 22 port [tcp/ssh] succeeded!
Connection to host.example.com 25 port [tcp/smtp] succeeded!
```

検索するポートの範囲をポート 20-30 に限定するように指定されています。

また、実行されているサーバーソフトウェアの種類やバージョンがわかると便利な場合もあります。この情報は、多くの場合、グリーティングバナーに含まれています。これらを取得するには、まず接続を確立し、バナーを取得したら接続を切断する必要があります。この操作は、-w フラグを使用して短いタイムアウトを指定することで実行できますが、サーバーに QUIT コマンドを発行することで実行できる場合もあります。

```
$ echo "QUIT" | nc host.example.com 20-30
SSH-2.0-Sun_SSH_1.1
Protocol mismatch.
220 host.example.com IMS SMTP Receiver Version 0.84 Ready
```

inetd の機能

考えられる用途の 1 つに、[inetd\(1M\)](#) を使った簡単なサービスの作成があります。

次の例では、ホスト realwww 上で TCP ポート 8080 からポート 80 へのリダイレクトを作成しています。

```
# cat << EOF >> /etc/services
wwwredir    8080/tcp    # WWW redirect
EOF
# cat << EOF > /tmp/wwwredir.conf
wwwredir stream tcp nowait nobody /usr/bin/nc /usr/bin/nc -w 3 realwww 80
EOF
# inetconv -i /tmp/wwwredir.conf
wwwredir -> /var/svc/manifest/network/wwwredir-tcp.xml
Importing wwwredir-tcp.xml ...Done
# inetadm -l wwwredir/tcp
SCOPE    NAME=VALUE
name="wwwredir"
endpoint_type="stream"
proto="tcp"
isrpc=FALSE
wait=FALSE
exec="/usr/bin/nc -w 3 realwww 80"
```



```

arg0="/usr/bin/nc"
user="nobody"
default bind_addr=""
default bind_fail_max=-1
default bind_fail_interval=-1
default max_con_rate=-1
default max_copies=-1
default con_rate_offline=-1
default failrate_cnt=40
default failrate_interval=60
default inherit_env=TRUE
default tcp_trace=TRUE
default tcp_wrappers=FALSE

```

特権

特権ポート番号にバインドするには、nc に `net_privaddr` 特権を付与する必要があります。Solaris Trusted Extensions が構成され、nc が待機するポートがマルチレベルポートとして構成されている場合は、`net_bindmlp` 特権も必要です。

特権をユーザーまたは役割に直接割り当てるには、[user_attr\(4\)](#) にアカウントのデフォルトの特権セットを指定します。ただし、このユーザーまたは役割が起動したアプリケーションにもこれらの特権が追加されます。nc が起動されたときにだけ特権 ([privileges\(5\)](#)) を付与するには、[rbac\(5\)](#) 権利プロファイルを作成して割り当てることをお勧めします。詳細は、「使用例」を参照してください。

使用例

例1 ncを使用する

ポート 3141 を発信元ポートとして使用し、タイムアウトを 5 秒に設定して、`host.example.com` のポート 42 への TCP 接続を開きます。

```
$ nc -p 3141 -w 5 host.example.com 42
```

`host.example.com` のポート 53 への UDP 接続を開きます。

```
$ nc -u host.example.com 53
```

接続のローカル側の IP として `10.1.2.3` を使用し、`host.example.com` のポート 42 への TCP 接続を開きます。

```
$ nc -s 10.1.2.3 host.example.com 42
```

ポートとポート範囲の一覧を使用して、さまざまなポートでポートスキャンを実行します。

```
$ nc -z host.example.com 21-25,53,80,110-120,443
```

UNIX ドメインソケットを作成して、待機します。

```
$ nc -lU /var/tmp/dsocket
```

関連付けられたポート 8888 を使用して UDP ソケットを作成し、待機します。

例1 ncを使用する (続き)

```
$ nc -u -l -p 8888
```

これは次と同じです。

```
$ nc -u -l 8888
```

関連付けられたポート 2222 を使用して TCP ソケットを作成して待機し、アドレス 127.0.0.1 にのみバインドします。

```
$ nc -l 127.0.0.1 2222
```

遅延オプションを設定してタイムアウトを 0 に設定すると、TCP ポートに接続し、データを送信し、(従来の TCP クロージングハンドシェークではなく) TCP RST セグメントを使用して接続を終了します。

```
$ echo "foo" | nc -L 0 host.example.com 22
```

ローカルポート 4545 からホスト host.example.com 上のポート 22 へのポートリダイレクトを実行します。

```
$ nc -R host.example.com/22 -l 4545
```

これにより、[ssh\(1\)](#) クライアントを実行し、上記のコマンドを実行している host redir.example.com を使用して host.example.com に接続できるようになります。

```
$ ssh -oStrictHostKeyChecking=no -p 4545 redir.example.com
```

TCP ポート上で nc を待機させ、TCP データストリームを UDP に変換することもできます (またはその逆も可能です)。

```
$ nc -R host.example.com/53/udp -l 4666
```

10.2.3.4、ポート 8080 の HTTP プロキシを使用して host.example.com のポート 42 に接続します。この例は [ssh\(1\)](#) で使用することもできます。詳細は、[ssh_config\(4\)](#) の ProxyCommand 指令を参照してください。

```
$ nc -x10.2.3.4:8080 -Xconnect host.example.com 42
```

同じ例で、今度はユーザー名 ruser を使用してプロキシ認証を有効にします (プロキシが認証を要求する場合)。

```
$ nc -x10.2.3.4:8080 -Xconnect -Pruser host.example.com 42
```

基本的な UDP ポートスキャンは、次のように効率的に実行できます。

```
$ nc -z -w 3 -u -i 0.5 host.example.com 11-100
```

例1 ncを使用する (続き)

2つの各ポート間で0.5秒間一時停止し(これにより、ICMPメッセージの速度制限が回避されます)、最大で3秒間返信を待機します。返信がない場合、そのポートは開かれている可能性があります。

多くの特権を持つユーザーまたは役割(デフォルトの root アカウントなど)として最小限の特権セットでncを実行するには、[ppriv\(1\)](#)も使用して起動します。たとえば、特権ポートにバインドするための特権だけで実行するように制限します。

```
$ ppriv -e -sA=basic,!file_link_any,!proc_exec,!proc_fork,\
!proc_info,!proc_session,net_privaddr nc -l 42
```

ユーザーまたは役割が `net_privaddr` 特権だけでncを実行できるようにするには、権利プロファイルを作成する必要があります。

```
/etc/security/exec_attr
Netcat privileged:solaris:cmd:::/usr/bin/nc:prvs=net_privaddr
```

```
/etc/security/prof_attr
Netcat privileged:::Allow nc to bind to privileged ports:help=None.html
```

[user_attr\(4\)](#)を使用してこの権利プロファイルを割り当てると、そのユーザーまたは役割は、任意のポートで待機できるncを実行できるようになります。ユーザーまたは役割がncを使用して特定のポートだけで待機できるようにするには、権利プロファイルにラッパースクリプトを指定するようにしてください。

```
/etc/security/exec_attr
Netcat restricted:solaris:cmd:::/usr/bin/nc-restricted:prvs=net_privaddr
```

```
/etc/security/prof_attr
Netcat restricted:::Allow nc to bind to privileged ports:help=None.html
```

また、許可されるオプションを制限するシェルスクリプトを作成する必要があります。たとえば、ポート42-64(42と64は除く)へのバインドのみを許可するものです。

```
/usr/bin/nc-restricted:

#!/bin/sh
[ $# -eq 1 ] && [ $1 -gt 42 -a $1 -lt 64 ] && /usr/bin/nc -l -p "$1"
```

これにより、ユーザーまたは役割がプロファイルシェルからラッパースクリプトを使用してncを起動すると、追加の特権が付与されます。[pfsh\(1\)](#)、[pfksh\(1\)](#)、[pfcsh\(1\)](#)、および[pfexec\(1\)](#)を参照してください。

ncを直接起動したり、`pfexec`またはプロファイルシェルを使用せずに起動したりすると、追加の特権を使用してncが実行されません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/netcat
インタフェースの安定性	下記を参照。

パッケージ名は「確実」です。-4、-6、-l、-n、-p、-u、-wの各オプションとその引数(存在する場合)のコマンド行構文は「確実」です。*name* および *port* リスト引数は「確実」です。ポート範囲の構文は「不確実」です。ほかのすべてのコマンド行オプションとその引数に関しては、インタフェースの安定性レベルは「不確実」です。

関連項目 [cat\(1\)](#), [pfcsh\(1\)](#), [pfexec\(1\)](#), [pfksh\(1\)](#), [pfsh\(1\)](#), [ppriv\(1\)](#), [sed\(1\)](#), [ssh\(1\)](#), [telnet\(1\)](#), [inetadm\(1M\)](#), [inetconv\(1M\)](#), [inetd\(1M\)](#), [ssh_config\(4\)](#), [user_attr\(4\)](#), [attributes\(5\)](#), [privileges\(5\)](#), [rbac\(5\)](#)

著者 nc の最初の実装は Hobbit (hobbit@avian.org) によって作成されました。

IPv6 サポートに伴う nc の改訂は、Eric Jackson (ericj@monkey.org) によって行われました。

注意事項 nc のインスタンスが(アドレスファミリ指定にかかわらず)ワイルドカードソケット上で待機している場合、別の nc プロセスを実在する IP アドレスにバインドしてこのアドレスへの接続を受け入れることもできます。たとえば、以下のプロセスを実行している場合、

```
$ nc -4 -l 5656
```

特定の IP アドレスおよび同じポート上で待機している別の nc プロセスを実行できます。

```
$ nc -4 -l 10.20.30.40 5656
```

アドレス 10.20.30.40 およびポート 5656 への TCP 接続はあとのプロセスによって受け入れられ、ポート 5656 および異なるアドレスへのすべての TCP 接続は前のプロセスによって受け入れられます。

また、IPv4 ワイルドカードソケットにバインドすることによって、(アドレスファミリを指定せずに)ワイルドカードソケット上で待機するプロセスから IPv4 接続を流用できます。この動作および上記の動作を無効にするには、-E オプションを使用します。

名前	ncab2clf – バイナリログファイルの共通ログファイル形式への変換
形式	<code>/usr/bin/ncab2clf [-Dhv] [-i input-file] [-o output-file] [-b size] [-n number] [-s datetime]</code>
機能説明	ncab2clf コマンドは、Solaris Network Cache and Accelerator (NCA) によって生成されたログファイルをバイナリ形式から共通ログファイル (CLF) 形式に変換するために使用します。 <i>input-file</i> を指定しない場合、b2clf は stdin を使用します。 <i>output-file</i> を指定しない場合、出力は stdout に送られます。
オプション	-b バイナリログファイルのブロック化(Kバイト単位)を指定します。デフォルトは64Kバイトです。 -D ダイレクト I/O を無効にすることを指定します。 -h 使用法を表示します。 -i <i>input-file</i> 入力ファイルを指定します。 -n <i>number</i> <i>number</i> 個の CLF レコードを出力します。 -o <i>output-file</i> 出力ファイルを指定します。 -s <i>datetime</i> <i>datetime</i> で指定した日時より前のレコードをスキップします。日時は、CLF 形式または touch(1) ユーティリティーで指定された形式で指定できます。CLF 形式が優先されるため、ncab2clf はまず <i>datetime</i> を CLF とみなして解析します。 -v 詳細な出力を表示します。
使用例	例1 バイナリファイルを共通ログファイル形式に変換する 次の例では、バイナリファイル <code>/var//logs/.blf</code> を共通ログファイル形式の <code>/var//logs/.clf</code> ファイルに変換しています。 <pre>example% ncab2clf -D -i /var/nca/logs/nca.blf -o /var/nca/logs/nca.clf</pre> 例2 複数のログファイルを変換する 次のスクリプトを使用すると、複数のログファイルを変換できます。「*」で指定されるディレクトリには、ログファイルだけが含まれている必要があります。 <pre>#!/bin/ksh for filename in * do ncab2clf -D < \$filename > \$filename.clf done</pre>

例3 raw デバイスで -s と -n を使用する

次の例は、raw デバイスで `ncab2clf` を使用方法を示しています。-n オプションを使用しない場合、デフォルトでファイルの先頭から末尾までのすべてのレコードが変換されます。-s で指定する日時は、次のように CLF 形式にします。

```
example% ncab2clf -s '10/Apr/2001:09:23:13' -n 100 < /dev/dsk/c2t1d0s6
```

終了ステータス 次の終了ステータスが返されます。

- 0 ファイルが正常に変換されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/network/http-cache-accelerator
インタフェースの安定性	確実

関連項目 [ncakmod\(1\)](#), [nca.if\(4\)](#), [ncakmod.conf\(4\)](#), [ncalogd.conf\(4\)](#), [attributes\(5\)](#), [nca\(7d\)](#)

『Solaris のシステム管理 (IP サービス)』

注意事項 NCA によって生成されるバイナリログファイルは、サイズが非常に大きくなる場合があります。サイズの大きなバイナリファイルを変換するときは、-b オプションを使用すると `ncab2clf` コマンドのパフォーマンスが向上します。

 ダイレクト I/O は、書き込まれるデータがそれほど大きな単位にならない場合に有効です。ただし、-b オプションを使用して大きな単位でログファイルを変換する場合は、-D オプションを使用してダイレクト I/O を無効にするようにしてください。

名前	ncakmod – NCA カーネルモジュールの起動または停止
形式	/etc/init.d/ncakmod start stop
機能説明	ncakmod は、Solaris Network Cache and Accelerator (NCA) カーネルモジュールを起動または停止するために使用します。 コマンド行に start オプションを指定すると、nca.if ファイルに一覧表示されているすべての物理インタフェースで NCA カーネルモジュールが有効になります。stop オプションを指定して ncakmod コマンドを起動すると、NCA カーネルモジュールは次のメッセージを出力します。 To stop NCA, please set the status configuration parameter to disable in ncakmod.conf and then reboot your system. See the ncakmod.conf(4) manual page for more information. システム上の NCA を正常に停止するには、最初に ncakmod.conf(4) ファイルを編集してステータスフィールドを「disable」に設定してから、システムをリブートする必要があります。
オプション	start NCA カーネルモジュールを起動します。 stop NCA 機能の現在の停止方法を表示します。
使用例	例1 NCA 機能を起動および停止する NCA 機能を起動するには、次のコマンドを使用します。 example% /etc/init.d/ncakmod start
ファイル	/etc/init.d/ncakmod NCA カーネルモジュールの起動スクリプト。 /etc/nca/ncakmod.conf NCA カーネルモジュールの構成オプションを指定します。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/network/http-cache-accelerator
インタフェースの安定性	確実

関連項目	ncab2clf(1) , ncad_addr(4) , nca.if(4) , ncakmod.conf(4) , ncalogd.conf(4) , attributes(5) , nca(7d)

名前	newform - テキストファイル形式の変更
形式	newform [-s] [-itabspec] [-otabspec] [-bn] [-en] [-pn] [-an] [-f] [-cchar] [-ln] [filename]...
機能説明	newform は <i>filename</i> 引数で指定されたファイルから行を読み取り、形式を変更して標準出力に書き込みます。ファイル名の指定が省略された場合には、標準入力から読み取ります。形式の変更は、コマンド行オプションの指定に従って行います。 -s を除き、コマンド行オプションは任意の順序で指定でき、繰り返し指定が可能で、省略可能な <i>filename</i> を間に入れることもできます。オプションは記述された順序で処理されます。つまり、“-e15 -l60” という指定と “-l60 -e15” という指定とでは、得られる結果は異なります。複数のファイル名が指定された場合、オプションはそのすべてに適用されます。
オプション	サポートしているオプションは、次のとおりです。 -s 各行において、先頭から最初のタブまでの文字を切り捨て、切り捨てた部分の先頭から最大 8 文字を各行の終わりに付加します。この切り捨てた部分 (最初のタブ文字は含まない) の長さが 8 文字を超える場合には、8 文字目は * で表され、以降の文字はすべて捨てられます。最初のタブは常に捨てられます。 タブを含んでいない行を持つファイルに対してこのオプションを指定すると、エラーメッセージが出力されプログラムは終了します。切り捨てられた部分の文字は、その行に対して有効なほかのオプションがすべて処理されるまで内部的に保持されます。その後、行の最後尾にその部分が付加されます。 例として、先頭に数値、次に複数のタブ、さらにその後にテキストが書かれている行からなるファイルが存在していると仮定します。このファイルを変換して、2 番目以降のタブをスペースに置き換えたものから始まり、その後にテキスト部分を置き、短ければ 72 カラムまでスペースを詰め (または長い場合には 72 カラムまでで切り捨て)、先頭にあった数値を 73 カラムから置く、という形式にする場合には、次のように指定します。 <pre>newform -s -i -l -a -e filename</pre> -itabspec 入力タブ指定。入力行に含まれるタブのそれぞれを、tabspec 引数の指定に従って展開します。tabspec の値としては、tabs(1) で説明されているすべての形式が指定できます。また - という値も指定可能で、この場合 newform は、標準入力から読み取るデータの先頭行にタブの処理方法が記述されていると見なします (fspec(4) を参照)。tabspec 引数を省略すると、デフォルト値の -8 が用いられます。tabspec の値が -0 のとき、タブは存在しないことを意味します。このときタブが検出されると、-1 が指定されていたものとして処理されます。

- otabspec** 出力タブ指定。tabspec 引数の指定に従って、スペースをタブに置き換えます。tabspec に指定できる値は、前述の **-itabspec** オプションと同じです。tabspec 引数を省略すると、デフォルト値の **-8** が用いられます。tabspec の値が **-0** のとき、スペースはタブに変換されずにそのまま出力されます。
- bn** 行の長さが有効な長さ (**-ln** オプションの説明を参照) を超えている場合、先頭から *n* 文字分を破棄します。デフォルトとして行の長さが有効な長さと等しくなるように、必要な数の文字を切り捨てます (デフォルト値は、*n* なしで **-b** オプションを指定した場合に使用される)。このオプションの便利な使い方として、COBOL プログラムの各行の行番号を削除する例を次に示します。
- ```
newform -l1 -b7 filename
```
- en** **-bn** が先頭部分を破棄するのに対し、この **en** は終端部分を破棄します。それ以外は **bn** と同じです。
- pn** 行の長さが有効な長さに満たないとき、行の先頭に付加する文字の数を指定します (文字の種類は **-cchar** オプションの説明を参照)。n なしで **p** オプションを指定すると、デフォルトとして行の長さが有効な長さと等しくなるように必要な数の文字を付加します。
- an** **-pn** が先頭部分に文字を付加するのに対し、この **an** は終端部分に文字を付加します。それ以外は **pn** と同じです。
- f** 最初の出力行として、タブ指定形式を表す行を出力します。その内容は **-o** オプションの指定に対応しており、**o** が複数個指定されていれば最後のものに対応しています。**-o** が 1 つも指定されていなければ、デフォルトのタブ形式の **-8** が出力されます。
- cchar** **pn** や **an** オプションで付加する文字として、**char** が示す文字を使用します。**char** のデフォルト値はスペースです。
- ln** 行の有効な長さを *n* 文字とします。*n* なしで **-l** オプションを指定すると、デフォルトとして 72 が用いられます。**-l** そのものを省略すると、デフォルトとして 80 文字が有効な長さとなります。なお、個々のタブ文字およびバックスペース文字は、それぞれ 1 文字と数えられます。ただし **-i** オプションを使って、タブをいくつかのスペース文字に展開することも可能です。
- b** オプションを有効に使用するために、行の有効な長さをファイル中のどの行の長さよりも短い値に設定する場合、**-l1** と指定してください。

オペランド 次のオペランドを指定できます。

*filename* 入力ファイル名

- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

1

エラーが発生しました。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [csplit\(1\)](#), [tabs\(1\)](#), [fspec\(4\)](#), [attributes\(5\)](#)

- 診断

いずれの診断メッセージも致命的なエラーを表します。

usage: . . .

newform コマンド行に不正なオプションが指定されている

"not -s format"

タブが含まれていない行を検出した

"can't open file"

ファイルがオープンできない

"internal line too long"

内部作業バッファ中で行を展開したら、長さが 512 文字を超えた

"tabspec in error"

タブ指定の形式が誤り。またはタブ位置の記述が昇順になっていない

"tabspec indirection illegal"

ファイル(または標準入力)から読み取られた *tabspec* に、ほかのファイル(または標準入力)を参照する *tabspec* が含まれていた

- 注意事項

通常 newform は、物理的な文字だけを認識します。ただし -i および -o に関しては、各行を適切な論理カラム位置に合わせて出力するために、バックスペース文字も認識します。

-i または -o- により *tabspec* を標準入力から読み取るように指定した場合でも、newform はユーザーに対してプロンプトを表示しません。

-f オプションが指定され、最後の -o オプションが -o- と記述されていて、その前に -o- または -i- という指定が記述されている場合、タブ指定形式を示す出力行の内容は不正確になります。
- 1282

マニュアルページセクション 1: ユーザーコマンド • 最終更新日 1997 年 7 月 21 日

名前 newgrp – 新たなグループへのログイン

形式

コマンド /usr/bin/newgrp [-l -L] [group]

sh 組み込み newgrp [argument]

ksh88 組み込み \*newgrp [argument]

ksh 組み込み +newgrp [argument]

機能説明

コマンド newgrp コマンドは、ユーザーの実および実効グループ ID を変更してユーザーを新たなグループへログインさせます。ユーザーはログイン状態を保ち、またカレントディレクトリも変わりません。newgrp を実行すると、常にシェルは新たなものに入れ替えられます。これはコマンドが、グループ名不明のためにエラーで終了した場合でも同様です。

エクスポートされない変数は、NULL またはデフォルト値にリセットされます。エクスポートされた変数は、その値を保持します。PS1、PS2、PATH、MAIL、HOME のようなシステム変数は、システムやユーザーによりエクスポートされないかぎり、デフォルト値にリセットされます。たとえば、ユーザーが主プロンプト文字列として、デフォルトの \$ の代わりに PS1 を使っていて、PS1 をエクスポートしなかった場合、ユーザーの PS1 はデフォルト文字列の \$ に設定されます。これは newgrp がエラーで終了したとしても同様です。なおシェルコマンドの export (sh(1) と set(1) を参照) は、変数をエクスポートするための方法で、これを使うと、新たなシェルを呼び出す際に割り当て済みの変数値を保持できます。

オペランドとオプションをすべて省略した場合、newgrp は、ユーザーのグループ ID (実 ID と実効 ID) を、ユーザーのパスワードファイルエントリに指定されているグループに戻します。この方法を使うと、いったん newgrp コマンド実行により変更した状態を、元に戻すことができます。

グループがパスワードを持っていて、ユーザーがそのグループのメンバーである旨が /etc/group ファイルに定義されていない場合、パスワードの入力が要求されます。グループ用にパスワードを生成する唯一の方法は、まず passwd(1) を実行し、さらに /etc/shadow から /etc/group へパスワードを「カット & ペースト」することです。グループ用パスワードは、今日ではあまり使われません。

sh 組み込み exec newgrp argument を実行した場合と同等です。argument は newgrp コマンドのオプションまたはオペランド、もしくはその両方を表します。

ksh88 組み込み exec /bin/newgrp argument を実行した場合と同等です。argument は newgrp コマンドのオプションまたはオペランド、もしくはその両方を表します。

このマニュアルページでは、1つまたは2つの\*(アスタリスク)が先頭に付加されている **ksh88(1)** コマンドは、次のような特殊な扱いを受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、\*\*を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

#### ksh 組み込み

`exec /bin/newgrp argument` を実行した場合と同等です。*argument* は `newgrp` コマンドのオプションまたはオペランド、もしくはその両方を表します。

このマニュアルページでは、1つまたは2つの+(プラス符号)が先頭に付加されている **ksh(1)** コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

#### オプション

次のオプションがサポートされています。

- l|-- 環境を、ユーザーが実際に新グループのメンバーとしてログインした場合に得られるものに変更します。

#### オペランド

次のオペランドがサポートされています。

*group*      グループデータベースから得られるグループ名、または負でない数値のグループ ID。実グループ ID と実効グループ ID として新たに用いるグループ ID を指定します。*group* が負でない数値で、グループ名としてグループデータベースに存在している (`getgrnam(3C)` を参照) 場合、そのグループ名に対応した数値グループ ID が新たなグループ ID として用いられます。

*argument*    `sh` と `ksh88` でのみ使用。`newgrp` コマンドのオプションまたはオペランド、もしくはその両方。

- 環境変数

newgrp の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。
- 終了ステータス

newgrp が新たなシェル実行環境を生成できた場合、グループ ID の変更が正常終了したかどうかにかかわらず、終了ステータスはシェルの終了ステータスと同じになります。環境が生成できなかったとき、次の終了値が返されます。  
  
>0 エラーが発生した。
- ファイル

/etc/group システムグループファイル  
/etc/passwd システムパスワードファイル
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/newgrp、ksh88、

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

ksh

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 不確実            |

関連項目

[login\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [set\(1\)](#), [sh\(1\)](#), [Intro\(3\)](#), [getgrnam\(3C\)](#), [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | newtask – タスクの新規作成と任意のプロジェクト変更                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 形式    | newtask [-p <i>project</i> ] [-v] [-c <i>pid</i>   [-F]] [ <i>command</i> ...]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明  | <p>newtask コマンドは、ユーザーのデフォルトシェルまたは指定されたコマンドを実行し、実行したコマンドを指定されたプロジェクトが所有する新しいタスクに配置します。ユーザーのデフォルトシェルは、passwd データベースに指定されたシェルであり、getpwnam() を使用して特定されます。</p> <p>また、newtask では、すでに実行されているプロセスを使用すると、新しく作成したタスクを入れることができます。新しいタスク用のプロジェクトもこの形式のコマンドで指定できます。これは、非常に重要で再起動できないプロセスを新しいプロジェクトに配置するのに適しています。</p> <p>また、拡張アカウンティングがアクティブな場合に newtask コマンドを使用すると、タスクのアカウンティングレコードを作成して、先行するシステムタスクの完了をマークすることもできます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-c <i>pid</i></b>    実行中のプロセスを新しく作成したタスクに入れます。また、-p オプションを使用すると、新しいタスク用のプロジェクトを指定できます。コマンドを起動したユーザーは、プロセスを所有しているか、スーパーユーザー特権を持っている必要があります。</p> <p>プロジェクトを変更する場合は、プロセスの所有者が指定のプロジェクトのメンバーであるか、コマンドを起動したユーザーがスーパーユーザー特権を持っている必要があります。実行中のプロセスのプロジェクトが変更されると、そのプールバインディングとリソース制御が新しいプロジェクトの構成に合わせて変更されます。プロジェクトのエントリで明示的に指定されなかった制御は保持されます。</p> <p>このオプションは、-F および -l オプションと同時に使用できません。</p> <p><b>-F</b>    ファイナライズタスクを作成します。ファイナライズタスク内では、以降の newtask または <a href="#">settaskid(2)</a> の起動は失敗します。ファイナライズタスクは、一部のサイトでリソース消費の帰属を簡略化する際に役立ちます。</p> <p><b>-l</b>    環境を、ユーザーが実際に新しいプロジェクトのメンバーとして再度ログインした場合に得られる環境に変更します。</p> <p><b>-p</b>    新しいタスクのプロジェクト ID を特定のプロジェクト名に関連付けられた ID に変更します。コマンドを正常に実行するには、起動したユーザーが、要求するプロジェクトの有効なメンバーであるか、スーパーユーザー特権を持っている必要があります。プロジェクト名を指定しなかった場合、新しいタスクは、起動したユーザーの現在のプロジェクト内で開始されます。</p> |

**-v** 詳細: 新しいシステムタスクの開始時にシステムタスク ID を表示します。

**オペランド** 次のオペランドがサポートされています。

**project** 作成したタスクのリソース使用量をチャージするプロジェクト。要求するプロジェクトは、[nsswitch.conf\(4\)](#) で定義されたプロジェクトデータベースで定義されている必要があります。

**command** 新しいタスクとして実行するコマンド。コマンドを指定しなかった場合、ユーザーのログインシェルが起動されます。ログインシェルが使用できない場合は、`/bin/sh` が起動されます。

## 使用例

**例1** 新しいシェルを作成する

次の例では、`canada` プロジェクトに新しいシェルを作成し、タスク ID を表示しています。

```
example$ id -p
uid=565(gh) gid=10(staff) projid=10(default)
example$ newtask -v -p canada
38
example$ id -p
uid=565(gh) gid=10(staff) projid=82(canada)
```

**例2** `date` コマンドを実行する

次の例では、`russia` プロジェクトで `date` コマンドを実行しています。

```
example$ newtask -p russia date
Tue Aug 31 11:12:10 PDT 1999
```

**例3** 既存プロセスのプロジェクトを変更する

次の例では、`pid` が 9999 の既存プロセスのプロジェクトを `russia` に変更しています。

```
example$ newtask -c 9999 -p russia
```

**終了ステータス** 次の終了ステータスが返されます。

- 0 正常終了。
- 1 実行中に致命的なエラーが発生した。
- 2 無効なコマンド行オプションが指定された。

## ファイル

`/etc/project` このマシンの有効なプロジェクト定義を含むローカルデータベース。

`/proc/pid/*` プロセス情報および制御ファイル。

属性

属性についての詳細は、[attributes\(5\)](#)を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目

[proc\(1\)](#), [id\(1M\)](#), [poolbind\(1M\)](#), [execvp\(2\)](#), [setrctl\(2\)](#), [settaskid\(2\)](#),  
[setproject\(3PROJECT\)](#), [nsswitch.conf\(4\)](#), [proc\(4\)](#), [project\(4\)](#), [attributes\(5\)](#)



|                    |                                                                                                                                                                                                                                                                                                                                                                                                                               |
|--------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前                 | nice – コマンドを変更されたスケジューリング優先順位で実行                                                                                                                                                                                                                                                                                                                                                                                              |
| 形式                 | <pre>/usr/bin/nice [-increment   -n increment] command [argument]...</pre> <pre>/usr/xpg4/bin/nice [-increment   -n increment] command [argument]...</pre>                                                                                                                                                                                                                                                                    |
| csch組み込みコマンド       | nice [-increment   +increment] [command]                                                                                                                                                                                                                                                                                                                                                                                      |
| 機能説明               | <p>nice ユーティリティーは、異なるシステムのスケジューリング優先順位で動作するように command を実行します。priocntl(1) コマンドは、スケジューラ機能とのより汎用的なインタフェースです。</p> <p>起動するプロセス (通常は、ユーザーのシェル) は、nice コマンドをサポートするスケジューリングクラスで実行されていなければなりません。</p> <p>C シェル (csh(1) 参照) を使用している場合、コマンドのフルパス名を指定する必要があります。指定しないと、csh 組み込みコマンドの nice が実行されます。後述の「csh 組み込みコマンド」を参照してください。</p>                                                                                                          |
| /usr/bin/nice      | nice が引数のあるコマンドを実行する場合、デフォルトのシェル /usr/bin/sh が使われます (sh(1) 参照)。                                                                                                                                                                                                                                                                                                                                                               |
| /usr/xpg4/bin/nice | nice が引数のあるコマンドを実行する場合は、/usr/xpg4/bin/sh が使用されます (ksh88(1) を参照)。                                                                                                                                                                                                                                                                                                                                                              |
| csch組み込みコマンド       | nice という csh の組み込みコマンドもあります。こちらの動作は、ここで説明した nice とは異なります。詳しくは csh(1) を参照してください。                                                                                                                                                                                                                                                                                                                                               |
| オプション              | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-increment   -n increment</b>    <i>increment</i> は正または負の 10 進整数です。このオプションは、ユーティリティーバージョンの実行時に、nice() 関数に <i>increment</i> オプション引数 (数値) を指定して呼び出した場合と同じ効果を発揮します。nice(2) を参照してください。nice() のエラー (EINVAL 以外) は無視されます。このオプションを指定しない場合、増分は 10 であると想定されます。</p> <p>スーパーユーザーは、たとえば <b>-10</b> のように、負数の <i>increment</i> を使用することにより、通常よりも高い優先順位でコマンドを実行できます。特権を持たないユーザーが負数の増分値を指定した場合は無視されます。</p> |
| オペランド              | <p>次のオペランドがサポートされています。</p> <p><b>command</b>    呼び出すコマンドの名前。command に組み込みコマンド (shell_builtins(1) を参照) を指定した場合、処理の結果は保証されません。</p>                                                                                                                                                                                                                                                                                              |

*argument*     *command* を呼び出す際に引数として与える文字列。

**環境変数**     *nice* の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、PATH、およびNLSPATH。

**終了ステータス**     *command* で指定したコマンドが呼び出された場合、そのコマンドの終了ステータスが *nice* の終了ステータスとなります。呼び出されなかった場合には、*nice* は次の終了ステータスを返します。

- 1-125     エラーが発生した。
- 126        *command* で示すコマンドユーティリティーは見つかったが呼び出すことができなかった。
- 127        コマンドが見つからなかった。

**属性**     属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/nice

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin/nice

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/xopen/xcu4                       |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

**関連項目**     [csh\(1\)](#), [ksh88\(1\)](#), [nohup\(1\)](#), [priocntl\(1\)](#), [sh\(1\)](#), [shell\\_builtins\(1\)](#), [nice\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前

nl – line numbering filter

形式

```
/usr/bin/nl [-p] [-b [type]] [-d [delim]] [-f [type]]
 [-h [type]] [-i [incr]] [-l [num]] [-n [format]]
 [-s [sep]] [-w [width]] [-v [startnum]] [file]

/usr/xpg4/bin/nl [-p] [-b type] [-d delim] [-f type]
 [-h type] [-i incr] [-l num] [-n format] [-s sep]
 [-w width] [-v startnum] [file]
```

機能説明

The `nl` utility reads lines from the named *file*, or the standard input if no *file* is named, and reproduces the lines on the standard output. Lines are numbered on the left in accordance with the command options in effect.

`nl` views the text it reads in terms of logical pages. Line numbering is reset at the start of each logical page. A logical page consists of a header, a body, and a footer section. Empty sections are valid. Different line numbering options are independently available for header, body, and footer. For example, `-bt` (the default) numbers non-blank lines in the body section and does not number any lines in the header and footer sections.

The start of logical page sections are signaled by input lines containing nothing but the following delimiter character(s):

| Line contents | Start Of |
|---------------|----------|
| \:\::         | header   |
| \::           | body     |
| \:            | footer   |

Unless optioned otherwise, `nl` assumes the text being read is in a single logical page body.

オプション

Command options may appear in any order and may be intermingled with an optional file name. Only one file may be named. The specified default is used when the option is not entered on the command line. `/usr/xpg4/bin/nl` options require option arguments. A SPACE character *may* separate options from option arguments. `/usr/bin/nl` options *may* have option arguments. If option-arguments of `/usr/bin/nl` options are not specified, these options result in the default. The supported options are:

- btype            Specifies which logical page body lines are to be numbered. Recognized *types* and their meanings are:
  - a            number all lines
  - t            number all non-empty lines.
  - n            no line numbering

*pexp* number only lines that contain the regular expression specified in *exp*. See NOTES below.

Default *type* for logical page body is *t* (text lines numbered).

*-ftype* Same as *-btype* except for footer. Default *type* for logical page footer is *n* (no lines numbered).

*-ddelim* The two delimiter characters specifying the start of a logical page section may be changed from the default characters (*\ :*) to two user-specified characters. If only one character is entered, the second character remains the default character (*:*). No space should appear between the *-d* and the delimiter characters. To enter a backslash, use two backslashes.

*-htype* Same as *-btype* except for header. Default *type* for logical page header is *n* (no lines numbered).

*-incr* *incr* is the increment value used to number logical page lines. Default *incr* is 1.

*-lnum* *num* is the number of blank lines to be considered as one. For example, *-l2* results in only the second adjacent blank being numbered (if the appropriate *-ha*, *-ba*, and/or *-fa* option is set). Default *num* is 1.

*-nformat* *format* is the line numbering format. Recognized values are:

*ln* left justified, leading zeroes suppressed

*rn* right justified, leading zeroes suppressed

*rz* right justified, leading zeroes kept

Default *format* is *rn* (right justified).

*-p* Do not restart numbering at logical page delimiters.

*-ssep* *sep* is the character(s) used in separating the line number and the corresponding text line. Default *sep* is a TAB.

*-vstartnum* *startnum* is the initial value used to number logical page lines. Default *startnum* is 1.

*-width* *width* is the number of characters to be used for the line number. Default *width* is 6.

## オペランド

The following operand is supported:

*file* A path name of a text file to be line-numbered.

**使用例**                    例 1   An example of the nl command

The command:

```
example% nl -v10 -i10 -d!+ filename1
```

will cause the first line of the page body to be numbered 10, the second line of the page body to be numbered 20, the third 30, and so forth. The logical page delimiters are !+.

**環境変数**                    See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of nl: LANG, LC\_ALL, LC\_COLLATE, LC\_CTYPE, LC\_MESSAGES, and NLSPATH.

**終了ステータス**            The following exit values are returned:

0            Successful completion.

>0          An error occurred.

**ファイル**                    /usr/lib/locale/locale/LC\_COLLATE/CollTable      Collation table generated by localedef

                              /usr/lib/locale/locale/LC\_COLLATE/coll.so      Shared object containing string transformation library routines

**属性**                        See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/nl

| ATTRIBUTE TYPE | ATTRIBUTE VALUE |
|----------------|-----------------|
| Availability   | system/core-os  |

/usr/xpg4/bin/nl

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE                    |
|---------------------|------------------------------------|
| Availability        | system/xopen/xcu4                  |
| Interface Stability | Committed                          |
| Standard            | See <a href="#">standards(5)</a> . |

**関連項目**                    [pr\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [regex\(5\)](#), [regexp\(5\)](#), [standards\(5\)](#)

**注意事項**                    Internationalized Regular Expressions are used in the POSIX and "C" locales. In other locales, Internationalized Regular Expressions are used if the following two conditions are met:

- /usr/lib/locale/locale/LC\_COLLATE/CollTable is present.
- /usr/lib/locale/locale/LC\_COLLATE/coll.so is not present.

Otherwise, Simple Regular Expressions are used.

Internationalized Regular Expressions are explained on [regex\(5\)](#). Simple Regular Expressions are explained on [regexp\(5\)](#).

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | nm - オブジェクトファイルのネームリストを表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 形式    | <pre>/usr/bin/nm [-ACDhLlnPpRrsTVv] [-efox] [-g   -u] [-t format] file...  /usr/xpg4/bin/nm [-ACDhLlnPpRrsTVv] [-efox] [-g   -u] [-t format] file...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明  | <p>nm ユーティリティーは、<i>file</i> オペランドで指定された ELF オブジェクトファイルのそれぞれに対して、シンボルテーブルを表示します。</p> <p>正しいファイルが指定されたにもかかわらず、そのファイルのシンボリック情報が存在しないとき、nm ユーティリティーはそのことを報告します。これはエラーとは見なされません。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| オプション | <p>次のオプションを使って、nm の出力内容を制御できます。</p> <ul style="list-style-type: none"><li>-A 各行に、オブジェクトの完全パス名とライブラリ名を出力します。</li><li>-C C++ シンボル名を、表示する前に複合化します。</li><li>-D SHT_DYNSYM シンボル情報を表示します。これは <code>ld.so.1</code> によって使用されるシンボルテーブルで、ストリップされた動的実行可能ファイルにも存在します。-D が指定されていない場合、デフォルトの動作では SHT_SYMTAB シンボル情報が表示されます。</li><li>-e (「注意事項」を参照)</li><li>-f (「注意事項」を参照)</li><li>-g 外部(グローバル)シンボル情報だけを出力します。</li><li>-h 見出しのデータを表示しません。</li><li>-L SHT_SUNW_LDYNSYM シンボル情報を表示します。このシンボルテーブルには、ローカル関数シンボルが含まれます。SHT_SUNW_LDYNSYM シンボルテーブルは、取り除かれた動的実行可能ファイルにも存在します。これらのシンボルによって、SHT_DYNSYM シンボルテーブルで見つかるグローバルシンボルが増加します。-L が指定されていない場合、デフォルトの動作では SHT_SYMTAB シンボル情報が表示されます。</li><li>-l WEAK と GLOBAL のシンボルを、WEAK シンボルのキー文字のあとに * を付加することにより区別します。</li><li>-n 外部シンボルを、名前順にソートして出力します。</li><li>-o シンボルの値とサイズを、10 進数ではなく 8 進数で出力します (-t o と同等)。</li><li>-p 簡単に解析できる、簡易形式の出力を生成します。各シンボル名の前には、シンボルの値 (未定義なら空白) と、シンボルの状態を示す文字が表示されます。その文字とは次のいずれかです。</li></ul> |

- A 絶対シンボル
- B bss (初期化されていないデータスペース) シンボル
- C COMMON シンボル
- D データオブジェクトシンボル
- F ファイルシンボル
- N タイプのないシンボル
- L スレッドローカル記憶領域シンボル
- R レジスタシンボル
- S セクションシンボル
- T テキストシンボル
- U 未定義

シンボルのバインディング属性の意味は次のとおりです。

- LOCAL このキー文字は小文字
- WEAK このキー文字は大文字。また `-l` 修飾子が指定されていると、大文字のキー文字の後に `*` が付加される
- GLOBAL このキー文字は大文字。

- P 後述の「標準出力」の項で述べるような、移植性のある出力形式で情報を出力します。
- r 各出力行に、オブジェクトファイルまたはアーカイブ名を付加します。
- R オブジェクトファイルとシンボル名の前に、アーカイブ名 (もし存在していれば) を出力します。-r オプションが同時に指定されている場合には、この R オプションは無視されます。
- s セクションインデックスの代わりにセクション名を出力します。
- t *format* 個々の数値を指定された形式で出力します。出力形式は、オプション引数の *format* により 1 文字で指定します。
  - d オフセットを 10 進数で出力 (デフォルト)
  - o オフセットを 8 進数で出力
  - x オフセットを 16 進数で出力
- T 「注意事項」を参照してください。

- /usr/bin/nm

-u

未定義のシンボルだけを出力します。
- /usr/xpg4/bin/nm

-u

未定義の各シンボルについての一覧を出力します。後述の「出力」を参照してください。
- v

外部シンボルを、値の順にソートして出力します。
- V

実行した nm コマンドのバージョンを標準エラー出力に書き出します。
- x

シンボルの値とサイズを、10 進数ではなく 16 進数で出力します (-tx と同等)。

オプションはいくつでも指定できます。複数個指定する場合、その順序は問いません。また、コマンド行上のどこに記述しても構いません。矛盾する組み合わせでオプションを指定すると(たとえば -v と -n または、-o と -x)、最初のオプションが有効となり、2 番目は無視されて警告メッセージが出力されます(例外は -R オプションを参照)。

オペランド

次のオペランドを指定できます。

file

オブジェクトファイル、実行可能ファイル、またはオブジェクトファイルライブラリのパス名

出力

このセクションでは、nm ユーティリティの出力オプションについて説明します。

標準出力

各シンボルに関して次の情報を出力します。

|       |                                                                                                                                                                                                                                                                                                                                                     |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Index | シンボルのインデックスです (インデックスは大括弧 [] で囲んで表示されます)。                                                                                                                                                                                                                                                                                                           |
| Value | シンボルの値で、次のいずれかです。 <div><div>■</div>再配置可能ファイル中で定義されているシンボルのセクションオフセット</div> <div><div>■</div>セクションインデックスが SHN_COMMON であるシンボルの境界合わせ上の制約</div> <div><div>■</div>実行可能および動的ライブラリファイル中の仮想アドレス</div>                                                                                                                                                       |
| Size  | 対応するオブジェクトのサイズで、単位はバイトです。                                                                                                                                                                                                                                                                                                                           |
| 「タイプ」 | シンボルのタイプで、次のいずれかです。 <div><div>NOTYPE</div><div>タイプは指定されていない</div></div> <div><div>OBJECT</div><div>配列や変数のようなデータオブジェクト</div></div> <div><div>FUNC</div><div>関数あるいはほかの実行可能コード</div></div> <div><div>REGI</div><div>レジスタシンボル (SPARC の場合のみ)</div></div> <div><div>SECTION</div><div>セクションシンボル</div></div> <div><div>FILE</div><div>ソースファイル名</div></div> |



|             |        |                                                                                    |
|-------------|--------|------------------------------------------------------------------------------------|
|             | COMMON | 初期化されていない共通ブロック                                                                    |
|             | TLS    | スレッドローカル記憶領域に関連する変数                                                                |
| Bind        |        | シンボルのバインディング属性で、次のいずれかです。                                                          |
|             | LOCAL  | このシンボルの有効範囲は、その定義を含んでいるオブジェクトファイルに限定されている                                          |
|             | GLOBAL | このシンボルは、結合されているすべてのオブジェクトファイルから見る事ができる                                             |
|             | WEAK   | このシンボルは、実質的にはグローバルシンボルだが、GLOBAL よりも優先順位は低い                                         |
| Other       |        | <sys/elf.h> で定義されている、いずれかの STV_ シンボル可視性の値に対応する整数。                                  |
| Shndx       |        | シンボル定義のもととなる、セクションヘッダーテーブルインデックスです。ただし例外として、3つの特殊な値のいずれかが表示されることがあります。次の特殊な値があります。 |
|             | ABS    | シンボルの値は再配置しても変わらないことを表します。                                                         |
|             | COMMON | 未割付ブロックを表し、値は境界合わせ用の制約を表す                                                          |
|             | UNDEF  | 未定義のシンボルを表す                                                                        |
| Name        |        | シンボル名です。                                                                           |
| Object Name |        | -A オプションが指定されているとき、オブジェクトまたはライブラリの名前を表します。                                         |

-p オプションが指定された場合には、前述の情報が次に説明する移植性のある形式で出力されます。この形式には3種類あり、-td、-to、-tx のどれが指定されたかによって使い分けられます。

"%s%s %s %d %d\n", library/object name, name, type, value, size

"%s%s %s %o %o\n", library/object name, name, type, value, size

"%s%s %s %x %x\n", library/object name, name, type, value, size

ここで、type の形式は -p オプションの説明と同じであり、library/object name の形式は次のとおりです。

- -A が指定されていなければ <library/object name> は空の文字列です。
- -A が指定され、対応する file オペランドがライブラリ名を示していない場合は、次のようになります。

"%s: ", file

- -Aが指定され、対応する *file* オペランドがライブラリ名を示している場合には、次のようになります。なお <*object file*> は、該当するシンボルを含んでいるライブラリのオブジェクトファイル名です。

`"%s[%s]:"`, *file*, *object file*

-A が省略されたとき、*file* オペランドが複数指定されているか、あるいは1つだけ指定された *file* オペランドがライブラリを表していれば、nm は、後続のシンボルを含むオブジェクトを識別する行を、そのシンボルを含む行の前に出力します。出力形式は次のとおりです。

- 対応する *file* オペランドがライブラリ名を示していないとき

`"%s:\n"`, *file*

- 対応する *file* オペランドがライブラリ名を示しているとき (なお <*object file*> は、後続のシンボルを含むライブラリ中のファイル名)

`"%s[%s]:\n"`, *file*, *object file*

-P が指定され、-t が省略された場合には、出力形式は -tx が指定された場合と同一になります。

**環境変数** nm の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_COLLATE、LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。

**終了ステータス** 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

**属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| /usr/bin/nm | 属性タイプ | 属性値                                |
|-------------|-------|------------------------------------|
|             | 使用条件  | developer/base-developer-utilities |

| /usr/xpg4/bin/nm | 属性タイプ       | 属性値               |
|------------------|-------------|-------------------|
|                  | 使用条件        | system/xopen/xcu4 |
|                  | インタフェースの安定性 | 確実                |

**関連項目** [ar\(1\)](#), [as\(1\)](#), [dump\(1\)](#), [ld\(1\)](#), [ld.so.1\(1\)](#), [ar.h\(3HEAD\)](#), [a.out\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

**注意事項** 次のオプションは、オブジェクトファイルの形式が変わってしまったため現在ではあまり使用されていません。将来のリリースで削除されることになります。

- e 外部シンボルと静的シンボルだけを出力します。現在、シンボルテーブルには、この2つしか含まれていません。自動シンボルは、シンボルテーブル内には存在しなくなり、cc -g で生成されるデバッグ情報に出力されます。この情報は、[dump\(1\)](#) を使って検査できます。
- f 完全な出力を生成します。なお .text や .data などの冗長シンボルは、現在はもう存在していません。したがってこの完全出力の生成は、デフォルト出力の生成と同じ意味になります。
- T デフォルトで、nm はシンボルの名前全体を出力します。シンボル名の出力カラムの位置が最終カラムに移動したので、オーバーフローの心配はなく、シンボル名を途中で切り捨てる必要はなくなりました。

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | nohup – ハングアップおよび停止の影響を受けないコマンドの実行                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 形式   | <pre>/usr/bin/nohup command [argument]...<br/>/usr/bin/nohup -p [-Fa] pid [pid]...<br/>/usr/bin/nohup -g [-Fa] gpid [gpid]...<br/>/usr/xpg4/bin/nohup command [argument]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 機能説明 | <p>nohup ユーティリティーは、<i>command</i> で示すコマンドを、指定された引数とともに呼び出します。<i>command</i> が呼び出されたとき、nohup は、そのプロセスが SIGHUP シグナルを無視するように設定します。</p> <p>-p または -g フラグを指定して実行すると、nohup は、プロセス ID またはグループ ID のリストで指定された実行中のプロセスが、ハングアップおよび停止の影響を受けないように設定します。</p> <p>指定したコマンドが実行に時間を要する場合、ユーザーがコマンド終了前にログアウトする場合は、nohup ユーティリティーを使うと便利です。シェルが終了するときに、システムはその子プロセスに SIGHUP シグナルを送信します。それにより、デフォルトでは子プロセスが終了します。停止、実行中、バックグラウンドにあるすべてのジョブは、その呼び出しが nohup コマンドによるものだった場合、もしくはプロセスがプログラム上 SIGHUP を無視するように選択されていた場合、SIGHUP を無視して続行します。</p> <p>nohup ユーティリティーは、プロセスが SIGHUP を無視するようにしますが、これらのプロセスをほかのシグナルから保護するわけではありません。最近のシェルはログアウト時に SIGHUP 以外のシグナルを送信する場合があるため、制御しているシェルが終了したときに、/usr/bin/nohup の下で実行されているジョブが強制終了されることがあります。</p> <p>/usr/bin/nohup                      /usr/bin/nohup によって実行されるプロセスは、SIGHUP (ハングアップ) および SIGQUIT (中止) シグナルを受け付けません。</p> <p>/usr/bin/nohup -p [-Fa]            ID に指定されたプロセスは、SIGHUP (ハングアップ) および SIGQUIT (中止) シグナルを受け付けません。また、制御端末への出力は、すべて nohup.out ファイルにリダイレクトされます。-F を指定すると、nohup は、各プロセスに制御を強制します。-a を指定すると、プロセスが SIGHUP (ハングアップ) および SIGQUIT (中止) シグナルのどちらかに対してハンドラをインストールした場合でも、これらのシグナル設定を変更します。</p> <p>/usr/bin/nohup -g [-Fa]            ID によって指定したプロセスと同じグループに属するすべてのプロセスは、SIGHUP (ハングアップ) および SIGQUIT (中止) シグナルを受け付けません。また、制御</p> |

端末への出力は、すべて `nohup.out` ファイルにリダイレクトされます。-F を指定すると、nohup は、各プロセスに制御を強制します。-a を指定すると、プロセスが `SIGHUP` (ハングアップ) および `SIGQUIT` (中止) シグナルのどちらかに対してハンドラをインストールした場合でも、これらのシグナル設定を変更します。

`/usr/xpg4/bin/nohup`

`/usr/xpg4/bin/nohup` によって実行されるプロセスは、`SIGHUP` シグナルを受け付けません。

nohup ユーティリティーは、プロセスを `SIGTERM` (終了) シグナルに影響されないようにはしないため、プロセス自身またはシェルが `SIGTERM` に影響されないようにしていないかぎり、プロセスは `SIGTERM` シグナルを受信します。

`nohup.out` が現在のディレクトリに書き込み可能でない場合、`$HOME/nohup.out` に出力先が変更 (リダイレクト) されます。ファイルが作成された場合、そのファイルは読み取り権と書き込み権 (600) を持ちます。[chmod\(1\)](#) を参照してください。標準エラーは、端末の場合出力先を標準出力に変更され、その他の場合は変更されません。nohup が実行するプロセスの優先度は変更されません。

## オプション

サポートしているオプションは、次のとおりです。

- a    どのような場合でも、対象プロセスの設定を変更します。このオプションは、-p または -g と一緒に使用した場合にのみ有効です。
- F    強制。別のプロセスが制御していても、対象のプロセスを捕捉します。このオプションは、-p または -g と一緒に使用した場合にのみ有効です。
- g    リストしたプロセスグループに対して処理を実行します。-p オプションと同時に使用できません。
- p    リストしたプロセスに対して処理を実行します。-g オプションと同時に使用できません。

## オペランド

次のオペランドがサポートされています。

|                |                                                                                        |
|----------------|----------------------------------------------------------------------------------------|
| <i>pid</i>     | nohup -p が使用する、10 進数のプロセス ID。                                                          |
| <i>pgid</i>    | nohup -g が使用する、10 進数のプロセスグループ ID。                                                      |
| <i>command</i> | 呼び出すコマンドの名前。このオペランドに <a href="#">shell builtins(1)</a> ユーティリティーを指定した場合の処理結果は定義されていません。 |

*argument*      *command* オペランドのコマンドを呼び出す際に引数として与える文字列。

**使用法**            -F フラグを使用するときは注意してください。2つの制御プロセスを1つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の `proc` ツールの適用時に何も実行していない場合のみ、安全性が確保されます。

**使用例**            例1 nohup をパイプラインまたはコマンドリストに適用する

nohup をパイプラインまたはコマンドのリストに適用した方が望ましい場合がよくあります。これは、シェルスクリプトと呼ばれる、パイプラインおよびコマンドリストを単一のファイルに格納する方法でしか実現されません。そうすれば、次を実行できます。

```
example$ nohup sh file
```

これで、nohup は *file* 内のすべてに適用されます。 *file* というシェルスクリプトを頻繁に実行する予定の場合、 *file* 実行権を指定すれば、sh の入力が少なくて済みます。

アンパサンドを追加すると、 *file* の内容がバックグラウンドで実行され、割り込みも無視されます ([sh\(1\)](#) を参照)。

```
example$ nohup file &
```

例2 nohup -p をプロセスに適用する

```
example$ long_running_command &
example$ nohup -p 'pgrep long_running_command'
```

例3 nohup -g をプロセスグループに適用する

```
example$ make &
example$ ps -o sid -p $$
 SID
 81079
example$ nohup -g 'pgrep -s 81079 make'
```

**環境変数**            nohup の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、NLSPATH、および PATH。

**HOME**            ユーザーのホームディレクトリのパス名を定義します。出力ファイル nohup.out が現在のディレクトリ内に生成できない場合、nohup コマンドは HOME が示すディレクトリを使ってファイルを作成します。

**終了ステータス**    次の終了ステータスが返されます。

- 126     *command* で示すコマンドユーティリティーは見つかったが呼び出すことができなかった。
- 127     nohup 中でエラーが発生した、または *command* で示すコマンドが見つからなかった。

それ以外の場合、*command* オペランドの終了値が nohup の終了値になります。

ファイル

nohup.out

標準出力が端末で現在のディレクトリが書き込み可能な場合に、nohup 実行 (結果) の出力用に使われるファイル

\$HOME/nohup.out

標準出力が端末で現在のディレクトリが書き込み可能でない場合に、nohup 実行 (結果) の出力用に使われるファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/nohup

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin/nohup

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/xopen/xcu4                       |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目

[bash\(1\)](#), [batch\(1\)](#), [chmod\(1\)](#), [csh\(1\)](#), [disown\(1\)](#), [ksh88\(1\)](#), [nice\(1\)](#), [pgrep\(1\)](#), [proc\(1\)](#), [ps\(1\)](#), [sh\(1\)](#), [shell\\_builtins\(1\)](#), [setpgrp\(1\)](#), [signal\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

警告

ログアウトをしようとしたときにログインシェルとして Korn シェル ([ksh88\(1\)](#)) を実行し、nohup されたジョブを実行していた場合、次のような警告メッセージが表示されます。

You have jobs running.

ログアウトを有効にするために、ログアウトを 2 回行う必要があります。ただし、バックグラウンドジョブは実行し続けます。

注意事項

nohup という C シェル ([csh\(1\)](#)) 組み込みコマンドは、終了シグナル SIGHUP の影響が及ばないようにします。しかし、出力先を nohup.out に変更しません。csh (1) を使用すると、&' で実行するコマンドは、バックグラウンドに居る間は、自動的に HUP シグナルの影響を受けなくなります。

nohup は、コマンドシーケンスを認識しません。次のコマンドを見てください。

```
example$ nohup command1; command2
```

このコマンドでは、nohup ユーティリティーが適用するのは `command1` だけです。次のコマンドを見てください。

```
example$ nohup (command1; command2)
```

このコマンドは構文的に正しくありません。



|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | nroff - ディスプレイまたはラインプリンタ用に文書をフォーマット                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 形式    | nroff [-ehiq] [-mname] [-nN] [-opagelist] [-raN] [-sN]<br>[-Tname] [-uN] [filename...]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 機能説明  | <p>nroff ユーティリティーは、指定された <i>filename</i> 内のテキストをタイプライタ的なデバイス用にフォーマットします。troff(1) の説明も参照してください。</p> <p><i>filename</i> 引数が指定されていない場合、nroff は標準入力を読み取ります。1つの - からなる引数は、標準入力に対応したファイル名であるとみなされます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| オプション | <p>以下のオプションがサポートされています。各オプションは、ファイルの前でさえあれば、任意の順序で指定できます。</p> <p>-e            端末の解像度に従い、語と語の間のスペースを均等にして左右両端をそろえた形式でテキストを出力します。</p> <p>-h            横方向に空白を出力する際、空白文字とタブ文字を併用します。これにより出力の速度を上げ、出力文字数を減らすことができます。タブ文字の位置は、文字 8 個分の幅ごとに設定されています。</p> <p>-i            指定した入力ファイルをすべて読み込んだあとで標準入力を読み取ります。</p> <p>-q            .rd 要求から読み込まれた出力を出力しません。</p> <p>-mname       指定した入力ファイルの前に、マクロファイル /usr/share/lib/tmac/tmac.name を付加します。</p> <p>-nN           生成した最初のページのページ番号を <i>N</i> とします。</p> <p>-opagelist   <i>pagelist</i> で示したページだけを印刷します。このリストには、複数のページ番号またはページ番号の範囲 (またはその両方) をコンマで区切って記述します。範囲として <i>N-M</i> と記述すればページ番号 <i>N</i> から <i>M</i> までが出力され、リストの先頭に -<i>N</i> と記述すれば先頭ページからページ番号 <i>N</i> までが出力され、最後に <i>N-</i> と記述すればページ番号 <i>N</i> から最終ページまでが出力されます。</p> <p>-raN          レジスタ設定。1 文字の引数 <i>a</i> で示すレジスタに、値 <i>N</i> を設定します。</p> <p>-sN           <i>N</i> ページごとの停止。nroff は、用紙のローディングや取り替えを可能にするために <i>N</i> ページ (<i>N</i> のデフォルト値は 1) ごとに停止し、復帰改行 (NEWLINE) を受信すると再開します。</p> <p>-Tname       <i>name</i> で示したデバイス用に出力を準備します。<i>name</i> としては次の値が使用できます。</p> <p>37           Teletype Corporation Model 37 型端末。この値がデフォルトです。</p> |

|            |                                                                                   |
|------------|-----------------------------------------------------------------------------------|
| lp   tn300 | GE — 「半改行」機能を持たないラインプリンタまたは端末                                                     |
| 300        | DASI-300                                                                          |
| 300-12     | DASI-300 — 12 ピッチ                                                                 |
| 300S       | DASI-300S                                                                         |
| 300S-12    | DASI-300S                                                                         |
| 382        | DASI-382 (fancy DTC 382).                                                         |
| 450        | DASI-450 (Diablo Hyterm)                                                          |
| 450-12     | DASI-450 (Diablo Hyterm) —12 ピッチ                                                  |
| 832        | AJ 832                                                                            |
| -uN        | 位置 3 にマウントされているフォントのボールド係数を <i>N</i> に設定します。 <i>N</i> を指定しない場合、ボールド係数は 0 に設定されます。 |

オペランド      次のオペランドを指定できます。

*filename*      nroff によって処理されるテキストを含むファイル。

使用例      例1    マクロパッケージを使ってフォーマット

本コマンドはマクロパッケージ `-me` を使って `users.guide` をフォーマットし、出力時には 4 ページごとに印刷をいったん停止します。

example% **nroff -s4 -me users.guide**

環境変数      nroff の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。

|      |                                          |                       |
|------|------------------------------------------|-----------------------|
| ファイル | <code>/usr/tmp/trtmp*</code>             | 一時ファイル (後述の「注意事項」を参照) |
|      | <code>/usr/share/lib/tmac/tmac.*</code>  | 標準マクロファイル             |
|      | <code>/usr/share/lib/nterm/*</code>      | nroff 用の端末駆動テーブル      |
|      | <code>/usr/share/lib/nterm/README</code> | 端末記述ファイルのインデックス       |

属性      属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値           |
|-------|---------------|
| 使用条件  | text/doctools |
| CSI   | 有効            |

関連項目 [checknr\(1\)](#), [col\(1\)](#), [eqn\(1\)](#), [man\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [me\(5\)](#), [ms\(5\)](#), [term\(5\)](#)

注意事項 現在、`/usr/tmp` は `/var/tmp` へのシンボリックリンクです。

以前のドキュメントでは、数値レジスタ `yr` を現在の年の末尾の 2 桁であると説明していましたが、これは正しくありません。`yr` は、実際には 1900 年以降の年数を示します。2099 年までの現在の年の末尾の 2 桁を正しく取得するには、文字列レジスタ `yy` の次の定義をドキュメントに含めることで、その後 2 桁の年の表示に使用することができます。`yy` の代わりに、ほかの任意の 1 文字または 2 文字のレジスタ名を使用できます。

```
.\" definition of new string register yy--last two digits of year
.\" use yr (# of years since 1900) if it is < 100
.ie \n(yr<100 .ds yy \n(yr
.el \{
 .\" else, subtract 100 from yr, store in ny
.nr ny \n(yr-100
.ie \n(ny>9 \{
 .\" use ny if it is two digits
.ds yy \n(ny
.\" remove temporary number register ny
.rr ny \}
.el \{.ds yy 0
.\" if ny is one digit, append it to 0
.as yy \n(ny
.rr ny \} \}
```

|             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |      |    |           |    |           |    |          |    |
|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|----|-----------|----|-----------|----|----------|----|
| 名前          | od - 8進ダンブ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |      |    |           |    |           |    |          |    |
| 形式          | <pre>/usr/bin/od [-bcCDdFfOoSsvXx] [-] [file] [offset_string] /usr/bin/od [-bcCDdFfOoSsvXx] [-A address_base] [-j skip] [-N count] [-t type_string]... [-] [file]... /usr/xpg4/bin/od [-bcCDdFfOoSsvXx] [file] [offset_string] /usr/xpg4/bin/od [-bcCDdFfOoSsvXx] [-A address_base] [-j skip] [-N count] [-t type_string]... [file]...</pre>                                                                                                                                                          |      |    |           |    |           |    |          |    |
| 機能説明        | <p>od コマンドは、指定された入力ファイルを順番に標準出力にコピーします。このとき、-t または -bcCDdFfOoSsvXx オプションで指定された出力タイプに従って入力データを変換します。出力タイプ指定が省略された場合には、-t o2 が指定されたものと見なします。-bcCDdFfOoSstvXx オプションを複数回記述して、複数のタイプを指定することも可能です。その場合、記述した順序に従って出力行がタイプごとに書き込まれます。file の指定がない場合は、標準入力を使用します。[offset_string] オペランドと -A、-j、-N、-t の各オプションは、互いに排他的の関係にあります。ここでは、次の用語を使用しています。</p> <p>ワード                      16 ビットのユニットを表します。マシンのワードサイズとは無関係です。</p> <p>ロングワード                32 ビットのユニットを表します。</p> <p>ダブルロングワード        64 ビットのユニットを表します。</p> |      |    |           |    |           |    |          |    |
| オプション       | <p>サポートしているオプションは、次のとおりです。</p> <p>-A address_base            入力オフセット値の基数を指定します。address_base 引数は次に述べる文字のいずれかでなければなりません。文字 d、o、x は、それぞれ 10 進、8 進、16 進を表します。文字 n は、オフセットは出力しない旨を表します。-An を指定しないかぎり、出力行の先頭には、次に書き込まれるバイトの入力オフセット値 (入力ファイルでの累積値) が付加されます。また、入力データがすべて処理されたあとは、最後に出力されたバイトの次のバイトのオフセット値が表示されます。-Aaddress_base オプションも [offset_string] オペランドも指定されない場合、入力オフセット値は 8 進数で表示されます。</p> <p>-b                          バイトを 8 進数で解釈します。この指定は -t o1 と同等の意味です。</p>                                     |      |    |           |    |           |    |          |    |
| /usr/bin/od | <p>-c                          シングルバイト文字を表示します。非図形文字のなかには、C 言語のエスケープで表示されるものもあります。</p> <table><tr><td>null</td><td>\0</td></tr><tr><td>backspace</td><td>\b</td></tr><tr><td>form-feed</td><td>\f</td></tr><tr><td>new-line</td><td>\n</td></tr></table>                                                                                                                                                                                                                                              | null | \0 | backspace | \b | form-feed | \f | new-line | \n |
| null        | \0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |      |    |           |    |           |    |          |    |
| backspace   | \b                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |      |    |           |    |           |    |          |    |
| form-feed   | \f                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |      |    |           |    |           |    |          |    |
| new-line    | \n                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |      |    |           |    |           |    |          |    |

```

return \r
tab \t

```

その他の文字は、3桁の8進数で表示されます。例:

```

echo "hello world" | od -c
0000000 h e l l o w o r l d \n
0000014

```

/usr/xpg4/bin/od

**-c** LC\_CTYPE ロケール カテゴリの現在の設定に基づき、バイトをシングルバイト文字または複数バイト文字で解釈します。印刷可能な複数バイト文字は、文字の最初のバイトに対応する領域に書き込まれます。残りのバイトに対応する領域には、文字が継続することを示す\*\*が書き込まれます。非図形文字は、**-c** オプションを使用したときと同じように解釈されます。

**-C** LC\_CTYPE ロケール カテゴリの現在の設定に基づき、バイトをシングルバイト文字または複数バイト文字で解釈します。印刷可能な複数バイト文字は、文字の最初のバイトに対応する領域に書き込まれます。残りのバイトに対応する領域には、文字が継続することを示す\*\*が書き込まれます。非図形文字のなかには、C 言語のエスケープで表示されるものもあります。

```

null \0
backspace \b
form-feed \f
new-line \n
return \r
tab \t

```

その他の非図形文字に関しては、各バイトにつき3桁の8進数が表示されます。

**-d** ワードを符号なし10進数で解釈します。この指定は **-t u2** と同等の意味です。

**-D** ロングワードを符号なし10進数で解釈します。この指定は **-t u4** と同等の意味です。

**-f** ロングワードを浮動小数点で解釈します。この指定は **-t f4** と同等の意味です。

**-F** ダブルロングワードを拡張精度で解釈します。この指定は **-t f8** と同等の意味です。

**-j skip** 入力データの先頭の *skip* 個のバイトをスキップします。od コマンドによる読み取りまたはシークは、連結された入力ファイルにおいて *skip* バイトを超えた地点から行われます。入力データの合計

長が *skip* バイトに満たない場合、*od* コマンドは標準エラー出力に診断メッセージを書き出し、ゼロ以外の終了ステータスコードで処理を終了します。

デフォルトでは *skip* 引数は 10 進数と解釈されます。先頭に *0x* または *0X* が付加されていれば、オフセット値は 16 進数と解釈されます。また先頭が *0* のときは、8 進数と解釈されます。また数値の後に文字 *b*、*k*、または *m* を付加すると、それぞれ 512、1024、または 1 048 576 バイトの倍数と見なされます。*skip* 値が 16 進数のとき、最後に *b* が付加されていてもそれは 16 進の数字と見なされます。アドレス表示は *0000000* から始まります。その基数が *skip* 引数の基数によって示されることはありません。

- N *count*      *count* が示すバイト数を超えない範囲で入力データをフォーマットします。デフォルトでは *count* は 10 進数と解釈されます。先頭に *0x* または *0X* が付加されていれば、*count* は 16 進数と解釈されます。また先頭が *0* のときは、8 進数と解釈されます。入力データが (-*jskip* 指定時はスキップ完了後) *count* バイト分存在しない場合でも、エラーとは見なされません。*od* コマンドは、存在している分の入力データをフォーマットします。表示されるアドレスの基数が *count* 引数の基数によって表されることはありません。
- o              ワードを 8 進数で解釈します。この指定は -*t o2* と同等の意味です。
- O              ロングワードを符号なし 8 進数で解釈します。この指定は -*t o4* と同等の意味です。
- s              ワードを符号付き 10 進数で解釈します。この指定は -*t d2* と同等の意味です。
- S              ロングワードを符号付き 10 進数で解釈します。この指定は -*t d4* と同等の意味です。
- t *type\_string*      出力タイプを指定します。*type\_string* オプション引数は、入力データを書き出す際に用いる出力タイプを表す文字列でなければなりません。文字列は、出力タイプを表す次の文字で構成されていなければいけません。
  - a      「名前を与えられた文字」バイトを名前を与えられた文字として解釈します。このタイプ指定では、各バイトのうち最下位の 7 ビットだけが用いられます。次の表に示す値を持つバイトが、文字に対応した名前を使って出力されます。

*od* における名前を与えられた文字

| Value | Name |
|-------|------|
|-------|------|

```

\000 nul
\001 soh
\002 stx
\003 etx
\004 eot
\005 enq
\006 ack
\007 bel
\010 bs
\011 ht
\012 lf
\013 vt
\014 ff
\015 cr
\016 so
\017 si
\020 dle
\021 dc1
\022 dc2
\023 dc3
\024 dc4
\025 nak
\026 syn
\027 etb
\030 can
\031 em
\032 sub
\033 esc
\034 fs
\035 gs
\036 rs
\037 us
\040 sp
\177 del

```

- c 「文字」を表します。LC\_CTYPE ロケール カテゴリの現在の設定に基づき、バイトをシングルバイト文字または複数バイト文字で解釈します。印刷可能な複数バイト文字は、文字の最初のバイトに対応する領域に書き込まれます。残りのバイトに対応する領域には、文字が継続することを示す \*\* が書き込まれます。次の非図形文字は、C 言語のエスケープで表示されます。`\0`、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v`。その他の非図形文字に関しては、各バイトにつき 3 桁の 8 進数が表示されます。

タイプ指定文字である `d`、`f`、`o`、`u`、`x` の後には、出力タイプが発生するたびに変換するバイト数を示す符号なし 10 進数を記述することもできます

- `f` 「浮動小数点数」を表します。後に文字 `F`、`D`、または `L` を付加して、タイプが `float`、`double`、`long double` の項目に対して変換を行う旨を表すこともできます。
- `d`、`o`、`u`、`x` それぞれ符号付き 10 進数、8 進数、符号なし 10 進数、16 進数を表します。後に文字 `C`、`S`、`I`、または `L` を付加して、タイプが `char`、`short`、`int`、または `long` の項目に対して変更を行う旨を表すこともできます。

1 つの *type\_string* 中に複数のタイプを続けて指定できます。また `-t` オプションを複数個指定することが可能です。その場合、タイプ指定文字を記述した順序に従って、出力行がタイプごとに書き込まれます。

- `-v` すべてのデータを表示します (冗長)。`-v` オプションを省略すると、直前の出力行に等しい (バイトオフセットを除く) 行は、すべてアスタリスク (\*) だけから成る行に置き換えられます。
- `-x` ワードを 16 進数で解釈します。この指定は `-t x2` と同等の意味です。
- `-X` ロングワードを 16 進数で解釈します。この指定は `-t x4` と同等の意味です。

## オペランド

`/usr/bin/od`

次のオペランドは、`/usr/bin/od` でのみ指定されます。

- `-` 指定されたすべてのファイルに加えて、標準入力を使用します。このオペランドを省略すると、*file* オペランドを指定しなかった場合だけ、標準入力を使用されます。
- file* 読み出すファイルのパス名。*file* を指定しないと、標準入力を読み込まれます。オペランドの数が 2 以内であり、`-A`、`-j`、`-N`、`-t` オプションがすべて省略され、さらに次の項目のいずれかが真である場合には、
  1. 最後のオペランドの最初の文字が正の記号 (+) である。
  2. 2 番目のオペランドの最初の文字が数値である。
  3. 2 番目のオペランドの最初の文字が `x` であり、2 番目のオペランドの 2 番目の文字が小文字の 16 進数文字または数字である。



4. 2 番目のオペランドが"x"である。

5. 2 番目のオペランドが"."である。

対応するオペランドはファイルのオペランドではなく、オフセット値指定と見なされます。

-N count オプションを省略した場合、表示はファイルの終わりに到達するまで続けられます。

```
[+][0] offset [.] [b|B]
[+][0][offset] [.]
[+][0x|x][offset]
[+][0x|x] offset[B]
```

*offset\_string* オペランドは、ファイル中のダンプを始める位置のバイトオフセット値を指定します。デフォルトでは、オフセット値は8進数のバイトで解釈され表示されます。*offset* が"0"で始まる場合には、オフセット値は8進数で解釈されます。*offset* が"x"または"0x"で始まる場合には、オフセット値は16進数で解釈され、"b"を付加した場合には、16進数の数字であるとみなされます。"."を付加すると、オフセット値は10進数で解釈されます。"b"または"B"を付加すると、オフセット値は512 バイト単位で解釈されます。*file* 引数を省略する場合には、オフセット値の先頭に(+)を付加しなければなりません。アドレスの表示は、指定されたオフセットから始まります。アドレスの基数は、指定してあればオフセットの基数と同じになります。指定していない場合は8進数になります。10進数は8進数をオーバーライドし、同じオフセットオペランドに16進数と10進数の両方の変換を指定するとエラーになります。

/usr/xpg4/bin/od

次のオペランドは、/usr/xpg4/bin/od でのみ指定できます。

*file*

最初の2つの項目のどちらかが真でなければならないことを除いては、/usr/bin/od と同じ。

```
[+] [0] offset [.] [b|B]
+ [offset] [.]
[+] [0x|x][offset]
[+] [0x|x] offset [B]
+x [offset]
+xoffset [B]
```

オフセット値の記述は /usr/bin/od と同じです。

環境変数

od の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、LC\_NUMERIC、およびNLSPATH。

終了ステータス    次の終了ステータスが返されます。

- 0      正常終了。
- >0    エラーが発生した。

属性                属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/od

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin/od

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/xopen/xcu4                       |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目            [sed\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | on – execute a command on a remote system with the local environment                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式   | on [-i] [-d] [-n] <i>host command [argument] ...</i>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 機能説明 | <p>The on program is used to execute commands on another system, in an environment similar to that invoking the program. All environment variables are passed and the current working directory is preserved. To preserve the working directory, the working file system must be either already mounted on the host or be exported to it. Relative path names will only work if they are within the current file system. Absolute path names may cause problems.</p> <p>The standard input is connected to the standard input of the remote command. The standard output and the standard error from the remote command are sent to the corresponding files for the on command.</p> <p>Note that the on program requires that the <a href="#">rpc.rxd(1M)</a> service be running on the remote machine. By default, <code>rpc.rxd</code> is present but not running on an Oracle Solaris system. Because of its better security, <a href="#">ssh(1)</a> is the preferred method of invoking commands on remote machines.</p> |

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| オプション | <p>The following options are supported:</p> <ul style="list-style-type: none"> <li>-d     Debug mode. Prints out some messages as work is being done.</li> <li>-i     Interactive mode. Uses remote echoing and special character processing. This option is needed for programs that expect to be talking to a terminal. All terminal modes and window size changes are propagated.</li> <li>-n     No Input. This option causes the remote program to get EOF when it reads from the standard input, instead of passing the standard input from the standard input of the on program. For example, -n is necessary when running commands in the background with job control.</li> </ul> |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

属性     See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE        |
|----------------|------------------------|
| Availability   | system/file-system/nfs |

関連項目     [chkey\(1\)](#), [rlogin\(1\)](#), [rsh\(1\)](#), [ssh\(1\)](#), [telnet\(1\)](#), [rpc.rxd\(1M\)](#), [attributes\(5\)](#)

|    |                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 診断 | <p>unknown host                      Host name not found.</p> <p>cannot connect to server        Host down or not running the server.</p> <p>can't find                        Problem finding the working directory.</p> <p>can't locate mount point        Problem finding current file system.</p> <p>RPC: Authentication error        The server requires DES authentication and you do not have a secret key registered with</p> |
|----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

keyserv. Perhaps you logged in without a password. Try to keylogin. If that fails, try to set your publickey with chkey.

on server: RPC: can't encode arguments

The 10240 byte limit for arguments to be encoded and passed from the sending to the receiving system has been exceeded.

Other diagnostic messages may be passed back from the server.

#### 使用上の留意点

When the working directory is remote mounted over NFS, a Control-Z hangs the window.

Root cannot use on.

|      |                                                                                                                                                                                                            |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | optisa – さまざまな命令セットから最適に使用できるものを決定                                                                                                                                                                         |
| 形式   | optisa <i>instruction_set</i> ...                                                                                                                                                                          |
| 機能説明 | optisa は、コマンドで指定した <i>instruction_set</i> のうち、このマシンで最適に機能するものを表示します。この場合、「最適」は <a href="#">isalist(1)</a> で返される命令セット名の順序に従って定義されます。 <i>instruction_set</i> に使用可能な値は、 <a href="#">isalist(5)</a> に表示されています。 |
| 属性   | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                              |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

|         |                                                                                                                                                         |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| 終了ステータス | 次の終了ステータスが返されます。<br>0 指定した <i>instruction_set</i> 値のいずれかがこのコマンドによって表示されます。<br>1 出力がありません。つまり、このマシンは optisa コマンドで指定したどの <i>instruction_set</i> も使用できません。 |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------|

関連項目 [isalist\(1\)](#), [uname\(1\)](#), [attributes\(5\)](#), [isalist\(5\)](#)

注意事項 特定のマシン上で、特定のプログラムの複数のバイナリバージョンのうちどれを使用すべきかを決定する場合は、`uname -p` や `uname -m` ([uname\(1\)](#) を参照) よりも、optisa を使用することをお勧めします。

名前 pack, pcat, unpack – ファイルの圧縮および復元

形式 pack [-f/] [-] file...

pcat file...

unpack [-/] file...

## 機能説明

pack コマンドは、指定されたファイルを圧縮した形で保存します。可能であれば (そして便利ならば)、file という入力ファイルは、file と同じアクセスモード、同じアクセスの日付や変更した日付、同じ所有者を持つ file.z という圧縮されたファイルに置き換えられます。pack の実行が正常に終了すると、file は削除されます。

圧縮の総量は、入力ファイルのサイズおよび文字度数分布によります。デコード用ツリーがそれぞれの .z ファイルの最初の部分を形成しているので、3 ブロック未満のファイルを圧縮することは、あまり意味がありません。ただし、プリンタプロットや図形の場合に起こるように、文字度数分布が非常に偏っているものは例外です。

一般的にテキストファイルを元のサイズの 60% から 75% に圧縮します。大きい文字セットを使用し、文字分布が一様なロードモジュールは、元のサイズのおよそ 90% にしか圧縮されません。

pack ユーティリティは、圧縮できなかったファイルの数を示す値を返します。この数値が 255 を超えた場合は、255 が返されます。

次の場合は、圧縮が発生しません。

- ファイルがすでに圧縮されている場合
- ファイル名が長すぎて、接尾辞 .z を付加できない場合
- ファイルにリンクがある場合
- ファイルがディレクトリの場合
- ファイルをオープンできない場合
- ファイルが空の場合
- 圧縮によってディスクブロックを減らせない場合
- file.z というファイルがすでに存在している場合
- .z ファイルを作成することができない場合
- 処理中に、入出力エラーが発生した場合

ファイル名の最後のセグメントは、.z 拡張子を付加する余裕を確保するために十分に短くする必要があります。ディレクトリは圧縮できません。

pcat コマンドは、[cat\(1\)](#) が通常ファイルに対して行うことを、圧縮したファイルに対して行います。ただし pcat をフィルタとして使うことはできません。指定したファイルは、復元され、標準出力に書き込まれます。

pcat は、復元できなかったファイルの数を返します。次の場合にはエラーとなります。

- ファイルがオープンできなかった
- ファイルが、pack の出力ファイルと認識できなかった

## unpack

unpack コマンドは、pack で作成したファイルを復元します。コマンドで指定したファイル file に対して、file.z というファイル (あるいは file が .z で終わる場合は単に file) を検索します。このファイルが圧縮されたファイルである場合は、復元したファイルに置き換えます。新たなファイル名は .z 接頭辞が取り除かれ、アクセスモード、アクセス日付や変更日付および所有者名は圧縮されたファイルと同じです。

unpack は、復元できなかったファイルの数を示す値を返します。pcat で述べた理由のほかに、次のような場合には復元できません。

- 復元されたときのファイル名がすでに存在する場合
- 復元したファイルを作成できない場合

## オプション

pack では次のオプションがサポートされています。

-f file の強制圧縮。ディレクトリ全体を圧縮するのに有効です。ただし、中には圧縮しても小さくならないファイルがあります。unpack または pcat は、圧縮したファイルを元の形式に復元できます。

pack および unpack では次のオプションがサポートされています。

-/ 圧縮または復元時に、ソースファイルに関連付けられた ACL および拡張システム属性がターゲットファイルにコピーされます。ACL または拡張システム属性をコピーできない場合は、元のファイルが保持され、診断メッセージが標準エラー出力に書き込まれ、最終的な終了ステータスがゼロ以外になります。

## オペランド

次のオペランドがサポートされています。

- file pack、unpack、または pcat するファイルのパス名。file には拡張子 .z を指定しても省略してもかまいません。
- pack は、1 バイトごとに Huffman (最小冗長度) コードを使用します。- 引数を使用すると、それぞれのバイトの使用回数、相対頻度およびバイトのコードを標準出力に出力するように内部フラグが設定されます。file の代わりに - を追加すると、内部フラグを設定し、リセットします。

## 使用法

ファイルが 2G バイト ( $2^{31}$  バイト) 以上ある場合の pack、pcat、unpack の動作については、[largefile\(5\)](#) を参照してください。

- 使用例

例1 圧縮ファイルを表示する

file.z という名前の圧縮ファイルを見るには、次のようにします。

```
example%pcat file.z
```

あるいは、単に次のようにします。

```
example%pcat file
```

例2 圧縮ファイルの復元コピーを作成する

file.z という名前の圧縮ファイルの復元コピー nnn を file.z を破壊せずに作成するには、次を使用します。

```
example%pcat file>nnn
```
- 環境変数

pack、pcat、およびunpackの実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。
- 終了ステータス

次の終了ステータスが返されます。

0      正常終了。

>0    エラーが発生した。圧縮または復元できなかったファイルの数が返される。失敗の数が255を超えた場合は、255が返されます。
- 属性

属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

- 関連項目

[cat\(1\)](#), [compress\(1\)](#), [zcat\(1\)](#), [fgetattr\(3C\)](#), [fsetattr\(3C\)](#)[attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#)



|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | packagemanager – Image Packaging System の GUI                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 形式    | <pre> /usr/bin/packagemanager [options]  /usr/bin/packagemanager [-h   --help]                         [-i   --info-install file] [-U   --update-all]                         [-R   --image-dir dir]  /usr/bin/packagemanager [file] </pre>                                                                                                                                                                                                                                                |
| 機能説明  | <p>packagemanager は、pkg(5) (Image Packaging System ソフトウェア) のグラフィカル ユーザーインタフェースです。</p> <p>パッケージマネージャーを使用すると、次のタスクを実行できます。</p> <ul style="list-style-type: none"> <li>■ パッケージの検索、インストール、および削除</li> <li>■ パブリッシャーの追加、削除、および変更</li> <li>■ ブート環境の作成、削除、および管理</li> </ul> <p><i>file</i> オペランドが指定されていて、その接尾辞が .p5i である場合、packagemanager は Web インストールモードで起動し、1 つ以上のパブリッシャーおよびパブリッシャーごとに数多くのパッケージが追加されます。</p>                                                                                       |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <pre> -h --help     使用方法に関するメッセージを表示します。  -i file --info-install file     .p5i ファイルを指定して、packagemanager を Web インストールモードで実行します。file の接尾辞は .p5i である必要があります。  -R dir --image-dir dir     自動的に検出されたイメージではなく、dir をルートとするイメージに対して処理を行います。  -U --update-all     利用可能な更新があるすべてのインストール済みパッケージを更新します。 </pre> <p>注 – package/pkg、package/pkg/package-manager、または package/pkg/update-manager を更新する必要がある場合、packagemanager は最初にこれらのパッケージを更新してから、残りの更新を実行するために再起動されます。</p> |

オペランド      *file*      Web インストールファイル。このファイルの接尾辞は .p5i である必要があります。Web インストールについての詳細は、パッケージマネージャーのオンラインヘルプを参照してください。

## 使用例

例1 現在のイメージで操作する

現在のイメージで `packagemanager` を起動します。

```
$ packagemanager
```

例2 指定されたイメージで操作する

/aux0/example\_root に格納されたイメージで `packagemanager` を起動します。

```
$ packagemanager -R /aux0/example_root
```

例3 Web インストールモードで起動する

Web インストールモードで `packagemanager` を起動します。

```
$ packagemanager ~/test.p5i
```

終了ステータス      次の終了ステータスが返されます。

- 0      すべてが動作しました。
- 1      エラーが発生した。
- 2      無効なコマンド行オプションが指定された。

## ファイル

`pkg(5)` イメージはより大きなファイルシステム内に任意に配置できるため、トークン `$IMAGE_ROOT` を使用して相対パスが区別されます。標準のシステムインストールでは、`$IMAGE_ROOT` は `/` と同等です。

`$IMAGE_ROOT/var/pkg`  
完全または部分的なイメージのメタデータディレクトリ。

`$IMAGE_ROOT/.org.opensolaris,pkg`  
ユーザーイメージのメタデータディレクトリ。

特定のイメージのメタデータ内にある特定のファイルおよびディレクトリには、修復時や復旧時に役立つ情報が含まれています。トークン `$IMAGE_META` は、メタデータが含まれる最上位ディレクトリを参照するために使用されます。通常、`$IMAGE_META` は前述の 2 つのパスのいずれかです。

`$IMAGE_META/gui-cache`  
プログラムの起動やパブリッシャーの切り替えの速度を上げるために `packagemanager` で保持されるキャッシュ済みメタデータの場所。

`$IMAGE_META` ディレクトリ階層内のその他のパスは非公開であり、変更される可能性があります。

属性 次の属性については、`attributes(5)` を参照してください。

| 属性タイプ       | 属性値                         |
|-------------|-----------------------------|
| 使用条件        | package/pkg/package-manager |
| インタフェースの安定性 | 不確実                         |

関連項目 [pm-updatemanager\(1\)](#), [pkg\(1\)](#), [pkg\(5\)](#)

パッケージマネージャのオンラインヘルプ

<http://hub.opensolaris.org/bin/view/Project+pkg/>

注意事項 `packagemanager` は、イメージのファイルおよびディレクトリで操作するために十分な権限で起動する必要があります。

|       |                                                                                                                              |
|-------|------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pagesize – メモリーのページサイズの表示                                                                                                    |
| 形式    | /usr/bin/pagsize [-a]                                                                                                        |
| 機能説明  | pagesize ユーティリティーは、 <a href="#">getpagesize(3C)</a> から返されたメモリーのデフォルトのページサイズをバイト単位で出力します。このプログラムは、移植性の高いシェルスクリプトを作成する際に役立ちます。 |
| オプション | 次のオプションがサポートされています。<br><br>-a    システムでサポート可能なすべてのハードウェアアドレス変換サイズを出力します。                                                      |
| 属性    | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目    [pggsz\(1\)](#), [getpagesize\(3C\)](#), [getpagesizes\(3C\)](#), [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pargs – プロセス引数、環境変数、または補助ベクトルの表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式    | pargs [-aceFlx] [pid   core]...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 機能説明  | <p>pargs ユーティリティーは、ターゲットプロセスまたはプロセスコアファイルを検査して、引数、環境変数と値、またはプロセス補助ベクトルを表示します。</p> <p>pargs は、表示できない文字をエスケープ 8 進数として、\xxx の形式で出力します。ただし、その文字が <a href="#">formats(5)</a> の「Escape Sequences」のセクションで指定されているいずれかの文字である場合は、そのセクションで指定されているとおりに表示します。</p> <p>pargs は、ターゲットプロセスのロケールに応じて動作しようとし、ターゲットプロセスと pargs プロセスが共通の文字エンコーディングを共有していない場合、pargs は <a href="#">iconv(3C)</a> 機能を使用して、抽出された文字列を表示可能な形式で生成しようとしています。そのような変換が不可能な場合、文字列は 7 ビット ASCII で表示されます。</p>                                                                                                                                                          |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-a    argv[] に含まれているプロセス引数を表示します (デフォルト)。</li> <li>-c    ターゲットのロケールに関係なく、ターゲットプロセス内の文字列を 7 ビット ASCII でエンコードされた文字列と同様に処理します。<a href="#">iconv(3C)</a> の使用は無効になります。</li> <li>-e    _environ シンボル、または /proc/pid/psinfo の pr_envp によって指定される、プロセスの環境変数と値を表示します。</li> <li>-F    強制。別のプロセスが制御していても、ターゲットプロセスを捕捉します。</li> <li>-l    引数を単一のコマンド行で表示します。コマンド行は、/bin/sh による解釈に適した方法で表示されます。引数の中に表示できない文字が含まれている場合、またはターゲットプロセスが別のロケールを使用している場合は、警告メッセージが表示されます。結果として表示されるコマンド行は、/bin/sh によって正しく解釈されない場合があります。</li> <li>-x    プロセス補助ベクトルを表示します。</li> </ul> |
| オペランド | <p>次のオペランドがサポートされています。</p> <p>pid    プロセス ID リスト。</p> <p>core   プロセスコアファイル。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 使用法   | <p>-F フラグを使用するときは注意してください。2 つの制御プロセスを 1 つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の proc ツールの適用時に何も実行していない場合のみ、安全性が確保されます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

0 以外

エラーが発生した (該当するプロセスがない、アクセス権が拒否された、オプションが無効など)。
- ファイル

/proc/pid/\*    プロセス情報および制御ファイル。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 確実             |

関連項目

[proc\(1\)](#), [iconv\(3C\)](#), [proc\(4\)](#), [ascii\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#), [formats\(5\)](#)

名前 passwd – ログインパスワードおよびパスワード属性の変更

形式

```
passwd [-r files | -r ldap | -r nis] [name]

passwd [-r files] [-egh] [name]

passwd [-r files] -s [-a]

passwd [-r files] -s [name]

passwd [-r files] [-d | -l | -u | -N] [-f] [-n min]
 [-w warn] [-x max] name

passwd -r ldap [-egh] [name]

passwd [-r ldap] -s [-a]

passwd [-r ldap] -s [name]

passwd -r ldap [-d | -l | -u | -N] [-f] [-n min]
 [-w warn] [-x max] name

passwd -r nis [-egh] [name]
```

機能説明

passwd コマンドは、パスワードを変更したり、あるいはユーザーのログイン *name* に関連するパスワードの属性を表示したりします。さらに、承認されたユーザーは passwd を使用して、すべてのログイン *name* に関連するパスワードおよび属性を登録もしくは変更できます。

パスワードを変更する場合、passwd は、古いパスワードがあれば、それを入力するよう要求します。次に、新しいパスワードを2回入力するように要求します。古いパスワードを入力すると、passwd は、そのパスワードが十分に時間経過しているかをチェックします。時間経過が不十分である場合、passwd は終了します。詳細は、[pwconv\(1M\)](#) および [shadow\(4\)](#) を参照してください。

pwconv コマンドは、`/etc/passwd` の情報をもとに `/etc/shadow` を作成し、更新します。pwconv は、`/etc/passwd` のパスワードフィールドで「x」という特殊な値を探します。この「x」という値は、ユーザー用のパスワードがすでに `/etc/shadow` にあり、修正するべきではないということを表します。

時間経過が十分である場合、新しいパスワードの構造が要求に合うかどうかチェックが行われます。新しいパスワードを2回入力した時点で、2つの新しいパスワードが比較されます。2つのパスワードが同じでない場合は、新しいパスワードに対して、最大2回までプロンプトが繰り返されます。

パスワードは、次の要求に合うように作らなければなりません。

- 各パスワードは、PASSLENGTH 文字以上でなければなりません。PASSLENGTH は、`/etc/default/passwd` に定義され、6 に設定されています。PASSLENGTH を 8 文字より多く設定するには、8 文字より多くの文字をサポートするアルゴリズムを [policy.conf\(4\)](#) に構成する必要があります。

- 各パスワードは、`/etc/default/passwd` に指定されている複雑さの制約を満たす必要があります。
- 各パスワードは、`/etc/default/passwd` に指定されている構成済み辞書のメンバーと同じであってははいけません。
- パスワード履歴チェックをサポートしているネームサービス用のアカウントで旧パスワード履歴が定義されている場合、新しいパスワードは、旧パスワード履歴に含まれるものであってはいけません。

前述の条件がすべて満たされた場合、デフォルトでは `passwd` コマンドは `/etc/nsswitch.conf` を参照してパスワード更新を実行すべきレポジトリ (記録場所) を決めます。具体的には `passwd` と `passwd_compat` の両エントリを検索します。これらのエントリに対応したリソースつまりレポジトリが更新されます。ただし、サポートされるパスワード更新設定は、次の場合に制限されます。いずれの形式にも合っていない場合はシステムにログインできません。使用可能なパスワード更新定義形式は次のとおりです。

- `passwd: files`
- `passwd: files ldap`
- `passwd: files nis`
- `passwd: compat (==> files nis)`
- `passwd: compat (==> files ldap)`  
`passwd_compat: ldap`

`ad` キーワードを上記のリストの `passwd` 構成のいずれかに追加できます。ただし、Active Directory (AD) ユーザーのパスワードを変更するときは、`passwd` コマンドを使用できません。パスワードの更新操作中に `ad` キーワードが `passwd` エントリで見つかった場合は、無視されます。

パスワードテーブルを所有するネットワーク管理者は、パスワードのどの属性をも変更できます。また、LDAP シャドー情報を更新するために構成された管理者も、任意のパスワード属性を変更できます。[ldapclient\(1M\)](#) を参照してください。

ユーザーのパスワードが、ローカル `files` エントリ内のほかに、いずれかのネームサービス内に格納されている場合、`passwd` コマンドはその両方を更新します。ネームサービス内のパスワードとローカル `files` エントリ内のパスワードは異なっている可能性があります。特定のパスワードリポジトリを変更するには、`passwd -r` を使用してください。

`passwd` は、承認されたユーザーに古いパスワードを入力するよう要求しません。

LDAP が有効になっている場合、ネイティブ LDAP クライアントシステム上の承認されたユーザーは、古い LDAP パスワードを要求されずにパスワードを変更できます。



デフォルトでは、ほかのユーザーのパスワードを変更することを承認されているユーザーでも、構成済みパスワードポリシーに従う必要があります。[pam\\_authok\\_check\(5\)](#)を参照してください。

通常、引数なしで入力された `passwd` は現在のユーザーのパスワードを変更します。ユーザーがログインしたあとに、役割または別のユーザーになるために [su\(1M\)](#) を呼び出した場合、`passwd` は、役割または新しいユーザーのパスワードではなく、元のユーザーのパスワードを変更します。

`-s` 引数は、承認されたユーザーに制限されます。

表示の書式は、次のようになります。

```
name status mm/dd/yy min max warn
```

パスワードの有効期限の情報が存在しない場合は、次のようになります。

```
name status
```

各情報の意味は次のとおりです。

*name*

ユーザーのログイン ID。

*status*

*name* のパスワードステータス。

*status* フィールドは次の値をとることができます。

LK

このアカウントがロックされていることを表します。「セキュリティ」を参照してください。

NL

このアカウントではログインできないことを表します。「セキュリティ」を参照してください。

NP

パスワードがないため、非認証でアクセス可能であることを表します。

PS

パスワードがあることを表します。

UN

パスワードフィールドのデータが不明です。認識可能なハッシュパスワードや上記のエントリのどちらでもありません。有効なパスワードハッシュについては、[crypt\(3C\)](#)を参照してください。

UP

このアカウントはまだ管理者によって有効化されていないため、使用できません。「セキュリティ」を参照してください。

*mm/dd/yy*

*name* のパスワードが最近変更された日付(すべてのパスワードの日付は、グリニッジ標準時(ユニバーサル時間)を使用して決定されます。したがって、時差があるところでは最大1日ずれることがあるので注意してください)。

*min*

*name* に対するパスワード変更に最低限必要な日数。MINWEEKS は /etc/default/passwd にあり、NULL に設定されています。

*max*

パスワードが *name* に対して有効である最大日数。MAXWEEKS は /etc/default/passwd にあり、NULL に設定されています。

*warn*

パスワードの有効期限切れが近いことを示す警告を *name* が受ける日を、*max* に相対的な日数で表します。

## セキュリティ

パスワードを変更するとき、passwd は [pam\(3PAM\)](#) を使用します。PAM を呼び出すとき、passwd は passwd というサービス名を使用し、パスワードを変更するための認証とパスワードとして auth というサービスモジュールタイプを使用します。

アカウントのロック (-l オプション) を使用すると、パスワードに基づくログインの使用と遅延実行 ([at\(1\)](#)、[batch\(1\)](#)、[cron\(1M\)](#) など) を禁止します。-N オプションを使用すると、パスワードに基づくログインを禁止しますが、遅延実行は許可します。

ロックされてパスワードがないアカウントやログインできないアカウントでは、ステータスをアクティブなパスワードに直接変更できません。-d を参照してください。ロックされる前にパスワードを持っていた、ロックされたアカウントでパスワードを変更すると、アカウントをロック解除せずにパスワードが変更されます。アカウントのロック解除については、-u を参照してください。承認された管理者は、アカウントにパスワードを付与することによって、有効化されていない状態でアカウントを有効化できます。

アカウントは非アクティブになったあとでロックされることがあります。このようなアカウントをロック解除するには、-u または -f オプションを使用します。-u の場合、パスワードは変更されません。-f を使用すると強制的にパスワードが変更されます。

## オプション

サポートしているオプションは、次のとおりです。

-a

すべてのエントリのパスワードの属性を表示します。-s オプションとともにのみ使用し、*name* を指定してはいけません。files および ldap リポジトリの場合、承認されたユーザーに制限されます。

-e

ログインシェルを変更します。通常のユーザーは自分のシェル情報を変更でき、承認されたユーザーはすべてのユーザーの情報を変更できます。シェルの選

択は、[getusershell\(3C\)](#)によって制限されています。ユーザーが現在 [getusershell](#) によって許可されていないシェルを保持している場合、承認されたユーザーはシェルを変更できます。

-g

[gecos \(finger\)](#) 情報を変更します。通常のユーザーは自分の [gecos](#) 情報を変更でき、承認されたユーザーはすべてのユーザーの情報を変更できます。

-h

ホームディレクトリを変更します。

-r

処理の対象とするレポジトリ (記録場所) を指定します。サポートされているレポジトリは、[files](#)、[ldap](#)、または [nis](#) です。

-s *name*

*name* が示すログイン名用のパスワード属性を表示します。file および [ldap](#) レポジトリの場合、これは承認されたユーザーに対してのみ機能します。また [nis](#) の場合には、パスワードの有効期限を設定する (エージング) 機能がサポートされていないので、この指定はどのユーザーも使用できません。

このオプションの出力 (このオプションのみ) は、「確実」で構文解析可能です。形式は *username* のあとにスペースが続き、そのあとに次のコードのいずれかが続きます。

これを解析するコードが不明なコードに対して柔軟性を持つように、今後新しいコードが追加される可能性があります。既存のすべてのコードの長さは2文字ですが、コードが常にこの長さであるとは限りません。

現在のステータスコードは次のとおりです。

LK

UNIX 認証でアカウントがロックされています。passwd -l が実行されたか、認証の失敗回数が、許容される構成済みの最大値に到達したためアカウントが自動的にロックされました。[policy.conf\(4\)](#) および [user\\_attr\(4\)](#) を参照してください。

NL

このアカウントではログインできないことを表します。passwd -N が実行されました。

NP

アカウントはパスワードを持っていません。passwd -d が実行されました。

PS

アカウントは有効なパスワードを持っている可能性が高いです。

UN

パスワードフィールドのデータが不明です。認識可能なハッシュパスワードや上記のエントリのどちらでもありません。有効なパスワードハッシュについては、[crypt\(3C\)](#)を参照してください。

UP

このアカウントはまだ管理者によって有効化されていないため、使用できません。「セキュリティ」を参照してください。

承認された  
ユーザーのオプ  
ション

既存のアカウントをロックおよびロック解除できるように、管理者に User Security プロファイルを付与する必要があります。このプロファイルには、新しく作成されたアカウントを有効化する機能、パスワードの有効期限オプションを設定する機能、およびパスワード属性を表示する機能も用意されています。次のリストには、さまざまな操作を実行するために必要な承認を示します。

承認されたユーザーのみが、次のオプションを使用できます。

-d

*name* に対するパスワードを削除します。ログイン *name* に対するパスワードを入力するように要求することはありません。この指定は、files および ldap リポジトリにのみ適用できます。

[login\(1\)](#) オプション PASSREQ=YES が構成されている場合、アカウントにはログインできません。PASSREQ=YES はデフォルト設定です。

-f

*name* のパスワードの期限を切ることによって、次のログインセッションでパスワードを強制的にユーザーに変更させます。このオプションは、非アクティブが原因でロックされたアカウントをロック解除する場合に便利です。

-l

すでにロックされているか、ログインできないアカウントである場合を除いて、*name* に対するアカウントをロックします。アカウントのロックを解除する方法については、-d オプション、または -u オプションを参照してください。

-N

*name* に対するパスワードエントリを、ログインで使用できないが、アカウントをロックしない値にします。この値を削除したり、ログインできるようにパスワードを設定したりする方法については、-d オプションを参照してください。

-n *min*

*name* の最小のフィールドを設定します。*min* フィールドには、*name* に対するパスワードを変更するために最低限必要な日数が設定されます。*min* が *max* より大きい場合は、ユーザーはパスワードを変更できません。このオプションは、必ず -x オプションとともに使用してください。ただし、*max* が -1 に設定されている場合は例外です (最低経過日数を設定する機能がオフ)。この場合は、*min* を設定する必要はありません。

**-u**

エントリ *name* に対してロックされたパスワードをロック解除します。ロックされたパスワードを削除したり、ログインできるようにパスワードを設定したりする方法については、**-d** オプションを参照してください。**-u** オプションは、非アクティブが原因でロックされたアカウントをロック解除する場合に便利です。

**-w *warn***

*name* の警告フィールドを設定します。*warn* フィールドには、パスワードの有効期限が切れる何日前に、期限切れが近いことを示す警告をユーザーが受けるか、その日数が設定されます。このオプションは、パスワードの有効期限が切れている場合は無効です。

**-x *max***

*name* の最大のフィールドを設定します。*max* フィールドには、パスワードが *name* に対して有効である最大の日数が設定されます。*max* が -1 に設定された場合は、*name* の有効期限を設定する機能はただちにオフになります。

## オペランド

次のオペランドを指定できます。

*name*

ユーザーのログイン名

## 環境変数

**LC\_\* 変数** (LC\_CTYPE、LC\_MESSAGES、LC\_TIME、LC\_COLLATE、LC\_NUMERIC、および LC\_MONETARY) ([environ\(5\)](#) を参照) のいずれも環境に設定されていなければ、それぞれ対応するロケールのカテゴリにおける **passwd** の動作は、環境変数 **LANG** によって決定されます。LC\_ALL が設定されていれば、その内容が **LANG** 変数やその他の LC\_\* 変数より優先されます。前述の変数が環境にまったく設定されていなければ、**c** ロケール (米国スタイル) が **passwd** の動作を決定します。

**LC\_CTYPE**

**passwd** の文字の処理方法を決定します。LC\_CTYPE に有効な値が設定されていると、**passwd** は、そのロケールにあった文字を含むテキストやファイル名を表示および処理できます。**passwd** は拡張 UNIX コード (EUC) も表示および処理できます。この場合、文字は 1 バイト幅、2 バイト幅、3 バイト幅のいずれも使用できます。また、**passwd** は 1、2、またはそれ以上のカラム幅の EUC 文字も処理することができます。**c** ロケールにおいては、ISO 8859-1 の文字だけが有効です。

**LC\_MESSAGES**

診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、肯定・否定の応答形式も決定します。**c** ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。

## 終了ステータス

**passwd** は、処理終了時に次のいずれかの値を返します。

0

成功。

- 1     アクセス権が与えられていません。
- 2     オプションの組み合わせが無効です。
- 3     予想できない失敗。パスワードファイルは変更されません。
- 4     予想できない失敗。パスワードファイルがありません。
- 5     パスワードファイルは使用中です。あとで行なってください。
- 6     オプションに対する引数が無効です。
- 7     有効期限のオプションが無効です。
- 8     メモリーがありません。
- 9     システムエラー。
- 10    アカウントが期限切れです。
- 11    パスワード情報は変更されませんでした。

## ファイル

/etc/default/passwd

デフォルト値は /etc/default/passwd 中の次のフラグで設定することができます。例: MAXWEEKS=26

### DICTIONDBDIR

生成された辞書データベースが格納されるディレクトリ。デフォルトは /var/passwd。

DICTIONLIST と DICTIONDBDIR のどちらも指定しない場合、辞書チェックは実行されません。

### DICTIONLIST

コンマ区切りの辞書ファイルのリスト。たとえば、`DICTIONLIST=file1, file2, file3`。各辞書ファイルは複数行から構成され、各行は単語と NEWLINE 文字から構成されます (/usr/share/lib/dict/words と同様)。完全パス名を指定する必要があります。これらのファイルの単語は1つのデータベースにマージされ、パスワードが辞書の単語に基づいているかどうかを判断します。

DICTIONLIST と DICTIONDBDIR のどちらも指定しない場合、辞書チェックは実行されません。

辞書データベースを事前構築するには、[mkpwdict\(1M\)](#) を参照してください。

#### HISTORY

ユーザーごとに保存しておく旧パスワード履歴の最大数。HISTORY 値をゼロ (0) に設定するか、HISTORY フラグを削除した場合、任意のユーザーが次回パスワードを変更するときに、すべてのユーザーの旧パスワード履歴が破棄されます。デフォルトでは、HISTORY フラグを定義しません。最大値は 26 です。現在のところ、この機能は、files ネームサービス (ローカルの [passwd\(4\)/shadow\(4\)](#)) で定義されたユーザーアカウントにのみ実施されます。

#### MAXREPEATS

同じ文字を続けることができる最大数を定義します。MAXREPEATS を設定しないか、ゼロ (0) に設定した場合、デフォルトでは、このチェックは実行されません。

#### MAXWEEKS

パスワードが有効な最大期間

#### MINALPHA

パスワードに必要な英字の最小数。MINALPHA を設定しない場合、デフォルトは 2 です。

#### MINDIFF

古いパスワードと新しいパスワードで異なる必要がある文字の最小数。MINDIFF を設定しない場合、デフォルトは 3 です。

#### MINDIGIT

パスワードに必要な数字の最小数。MINDIGIT を設定しないか、ゼロ (0) に設定した場合、デフォルトでは、このチェックは実行されません。MINNONALPHA を指定した場合、MINDIGIT は指定できません。

#### MINLOWER

パスワードに必要な英小文字の最小数。MINLOWER を設定しないか、ゼロ (0) に設定した場合、デフォルトでは、このチェックは実行されません。

#### MINNONALPHA

パスワードに必要な英字以外の文字 (数字または特殊文字) の最小数。MINNONALPHA を設定しない場合、デフォルトは 1 です。MINDIGIT または MINSPECIAL を指定した場合、MINNONALPHA は指定できません。

#### MINWEEKS

パスワード変更に最低限必要な期間



**MINSPECIAL**  
パスワードに必要な特殊文字 (英字と数字以外の文字) の最小数。MINSPECIAL を設定しないか、ゼロ (0) に指定した場合、デフォルトでは、このチェックは実行されません。MINNONALPHA を指定した場合、MINSPECIAL は指定できません。

**MINUPPER**  
パスワードに必要な英大文字の最小数。MINUPPER を設定しないか、ゼロ (0) に指定した場合、デフォルトでは、このチェックは実行されません。

**NAMECHECK**  
ログイン名のチェックを有効または無効にします。デフォルトでは、ログイン名チェックは実行されます。NAMECHECK を no (大文字小文字の区別なし) に設定した場合、このチェックは実行されません。

**PASSLENGTH**  
パスワードに最低限必要な文字数

**WARNWEEKS**  
パスワードの有効期間が切れる前の警告期間

**WHITESPACE**  
パスワード内でスペース文字を許可するかどうかを指定します。指定できる値は、YES と NO です。WHITESPACE が設定されていないか、または YES に設定されている場合、スペース文字は許可されます。

**/etc/oshadow**  
passwd および pwconv が実際のシャドウファイルを更新するときに使用する一時ファイル。

**/etc/passwd**  
パスワードファイル。

**/etc/shadow**  
シャドウパスワードファイル。

**/etc/shells**  
シェルデータベース。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| CSI         | 有効             |
| インタフェースの安定性 | 下記を参照。         |

人間が読める形式の出力は「不確実」です。オプションは「確実」です。



## 関連項目

[at\(1\)](#), [batch\(1\)](#), [finger\(1\)](#), [login\(1\)](#), [cron\(1M\)](#), [domainname\(1M\)](#), [eeprom\(1M\)](#), [id\(1M\)](#), [ldapclient\(1M\)](#), [mkpwdict\(1M\)](#), [pwconv\(1M\)](#), [su\(1M\)](#), [useradd\(1M\)](#), [userdel\(1M\)](#), [usermod\(1M\)](#), [crypt\(3C\)](#), [getpwnam\(3C\)](#), [getspnam\(3C\)](#), [getusershell\(3C\)](#), [pam\(3PAM\)](#), [loginlog\(4\)](#), [nsswitch.conf\(4\)](#), [pam.conf\(4\)](#), [passwd\(4\)](#), [policy.conf\(4\)](#), [shadow\(4\)](#), [shells\(4\)](#), [user\\_attr\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [pam\\_authtok\\_check\(5\)](#), [pam\\_authtok\\_get\(5\)](#), [pam\\_authtok\\_store\(5\)](#), [pam\\_dhkeys\(5\)](#), [pam\\_ldap\(5\)](#), [pam\\_unix\\_account\(5\)](#), [pam\\_unix\\_auth\(5\)](#), [pam\\_unix\\_session\(5\)](#), [crypt\\_unix\(5\)](#)

## 注意事項

`yppasswd` コマンドは、`passwd` のラッパーです。`yppasswd` の使用は推奨されていません。`passwd -r repository_name` を使用することをお勧めします。

`files` および `ldap` リポジトリ内のパスワードを変更すると、失敗したログインのカウン트가クリアされます。

パスワードを変更すると、非アクティブ期間の長さのために無効になったアカウントが再度有効になります。

入力端末処理は、キーシーケンスの一部を解釈し、その部分の情報を `passwd` コマンドに渡さない可能性があります。

パスワードを持っていないアカウント (ステータスコード NP) は、ログインできない可能性があります。[login\(1\)](#) PASSREQ オプションを参照してください。

さまざまなオプションを実行するために必要な承認:

|    |                                                         |                           |
|----|---------------------------------------------------------|---------------------------|
| -d | delete password                                         | solaris.passwd.assign     |
| -N | set nologin                                             | solaris.passwd.assign     |
|    | change any passwd                                       | solaris.passwd.assign     |
| -l | lock account                                            | solaris.account.setpolicy |
| -u | unlock account                                          | solaris.account.setpolicy |
| -n | set min field for name                                  | solaris.account.setpolicy |
| -w | set warn field for name                                 | solaris.account.setpolicy |
| -x | set max field for name                                  | solaris.account.setpolicy |
| -f | forces password expiration                              | solaris.account.setpolicy |
| -s | display password attributes                             | solaris.account.setpolicy |
| -a | display password attributes for all entries             | solaris.account.setpolicy |
| -e | change login shell                                      | solaris.user.manage       |
| -g | change gecost information                               | solaris.user.manage       |
| -h | change home directory                                   | solaris.user.manage       |
|    | set a newly created account's passwd for the first time | solaris.account.activate  |

[crypt\\_unix\(5\)](#) を除くすべてのパスワードハッシュアルゴリズムでは、パスワードの長さの最大値は 255 です。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | paste - 複数のファイルの対応する行および以降の行のマージ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 形式    | /usr/bin/paste [options] [file...]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 機能説明  | <p>paste ユーティリティーは、指定された入力ファイルの対応する行を連結し、その結果行を標準出力に書き出します。</p> <p>paste のデフォルト動作は、入力ファイルの対応する行を連結します。最後の入力ファイルからの行以外の各行で、NEWLINE 文字を TAB 文字に置き換えます。</p> <p>1 つまたは複数の入力ファイル (すべての入力ファイルではない) で EOF (ファイルの終わり) 状態が検出された場合、-s オプションが指定されている場合を除き、paste は EOF が検出されたファイルから空行を読み込んだかのように動作します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-d list</b> list にバックスラッシュ文字 (\) がある場合を除き、list 中の各文字は区切り文字を指定する要素となります。list 中にバックスラッシュ文字がある場合は、後述するように、そのバックスラッシュ文字とそれに続くいくつかの文字が区切り文字を指定する要素となります。これらの要素により、入力行の復帰改行文字を置き換える文字として、デフォルトのタブ文字に代わって使用する区切り文字を指定します。list 中の要素は循環して使われます。つまりリストの最終要素に到達したら、次は先頭の要素に戻ります。</p> <p>-s オプションが指定されると、次のような動作が行われます。</p> <ul style="list-style-type: none"><li>■ ファイル中の最後の NEWLINE 文字は変更されません。</li><li>■ 各 file オペランドが処理されたあと、区切り文字はリストの先頭要素に戻ります。</li></ul> <p>s オプションが省略されたときは、次のような動作となります。</p> <ul style="list-style-type: none"><li>■ 最後の file で指定されるファイル中の NEWLINE 文字は変更されません。</li><li>■ 各ファイルからの 1 つの行が処理されるたびに、区切り文字は list の先頭要素に戻ります。</li></ul> <p>list 中にバックスラッシュ文字がある場合、それとそれに続く文字が使用され、次の区切り文字を表します。</p> <p>\n 復帰改行文字</p> <p>\t TAB 文字。</p> <p>\\ バックスラッシュ文字</p> <p>\0 空の文字列 (NULL 文字ではない)。0 の直後の文字が x か X のとき、または LC_CTYPE digit キーワードで定義されたいずれかの文字</p> |

のとき、結果は予測できません。

バックスラッシュのあとにその他の文字が続く場合、結果は予測できません。

**-s**      コマンド行で指定された順序で、各ファイルにおいてすべての行を連結します。各入力ファイルで、最終行を除くすべての行の NEWLINE 文字が、**-d** オプションで指定されないかぎり、TAB 文字に置き換えられます。

**オペランド**      次のオペランドを指定できます。

*file*      入力ファイルのパス名。1 つまたは複数のファイルで指定されると、標準入力を使用されます。標準入力は、ドット (.) のインスタンスごとに、循環して一度に 1 行ずつ読み込まれます。実装は、少なくとも 12 の *file* オペランドのペーストをサポートしています。

**使用法**      ファイルが 2G バイト (2<sup>31</sup> バイト) 以上ある場合の **paste** の動作については、[largefile\(5\)](#) を参照してください。

**使用例**

**例1** 1 カラムでのディレクトリの表示

次の例では、ディレクトリを 1 カラムで表示します。

```
example% ls | paste -d" " -
```

**例2** 4 カラムでのディレクトリの表示

次の例では、ディレクトリを 4 カラムで表示します。

```
example% ls | paste - - - -
```

**例3** ファイル内での 2 行ずつの連結

次の例では、ファイル内の行を 2 行ずつ連結します。

```
example% paste -s -d"\ t\ n" file
```

**環境変数**      **paste** の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、および NLSPATH。

**終了ステータス**      次の終了ステータスが返されます。

**0**      正常終了。

**>0**      エラーが発生した。

**属性**      属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目

[cut\(1\)](#), [grep\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | patch – apply changes to files                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式    | <pre>patch [-bLNR] [-c   -e   -n   -u] [-d <i>dir</i>] [-D <i>define</i>]       [-i <i>patchfile</i>] [-o <i>outfile</i>] [-p <i>num</i>] [-r <i>rejectfile</i>]       [<i>file</i>]</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 機能説明  | <p>The patch command reads a source (patch) file containing any of the three forms of difference (diff) listings produced by the <a href="#">diff(1)</a> command (normal, context or in the style of <a href="#">ed(1)</a>) and apply those differences to a file. By default, patch reads from the standard input.</p> <p>patch attempts to determine the type of the diff listing, unless overruled by a -c, -e, or -n option.</p> <p>If the patch file contains more than one patch, patch attempts to apply each of them as if they came from separate patch files. (In this case the name of the patch file must be determinable for each diff listing.)</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| オプション | <p>The following options are supported:</p> <ul style="list-style-type: none"> <li>-b            Saves a copy of the original contents of each modified file, before the differences are applied, in a file of the same name with the suffix <code>.orig</code> appended to it. If the file already exists, it is overwritten. If multiple patches are applied to the same file, the <code>.orig</code> file is written only for the first patch. When the -o <i>outfile</i> option is also specified, <i>file.orig</i> is not created but, if <i>outfile</i> already exists, <i>outfile.orig</i> is created.</li> <li>-c            Interprets the patch file as a context difference (the output of the command diff when the -c or -C options are specified).</li> <li>-d <i>dir</i>       Changes the current directory to <i>dir</i> before processing as described in EXTENDED DESCRIPTION.</li> <li>-D <i>define</i>   Marks changes with the C preprocessor construct: <pre> #ifdef <i>define</i> . . . #endif</pre> <p>The option-argument <i>define</i> is used as the differentiating symbol.</p> </li> <li>-e            Interprets the patch file as an ed script, rather than a diff script.</li> <li>-i <i>patchfile</i> Reads the patch information from the file named by the path name <i>patchfile</i>, rather than the standard input.</li> <li>-l            (The letter ell.) Causes any sequence of blank characters in the difference script to match any sequence of blank characters in the input file. Other characters is matched exactly.</li> <li>-n            Interprets the script as a normal difference.</li> </ul> |

- N Ignores patches where the differences have already been applied to the file; by default, already-applied patches are rejected.
- o *outfile* Instead of modifying the files (specified by the *file* operand or the difference listings) directly, writes a copy of the file referenced by each patch, with the appropriate differences applied, to *outfile*. Multiple patches for a single file is applied to the intermediate versions of the file created by any previous patches, and results in multiple, concatenated versions of the file being written to *outfile*.
- p *num* For all path names in the patch file that indicate the names of files to be patched, deletes *num* path name components from the beginning of each path name. If the path name in the patch file is absolute, any leading slashes are considered the first component (that is, -p 1 removes the leading slashes). Specifying -p 0 causes the full path name to be used. If -p is not specified, only the basename (the final path name component) is used.
- R Reverses the sense of the patch script. That is, assumes that the difference script was created from the new version to the old version. The -R option cannot be used with ed scripts. patch attempts to reverse each portion of the script before applying it. Rejected differences is saved in swapped format. If this option is not specified, and until a portion of the patch file is successfully applied, patch attempts to apply each portion in its reversed sense as well as in its normal sense. If the attempt is successful, the user is prompted to determine if the -R option should be set.
- r *rejectfile* Overrides the default reject file name. In the default case, the reject file has the same name as the output file, with the suffix .rej appended to it. See Patch Application.
- u Interprets the patch file as a unified context difference, that is, the output of the command diff when the -u or -U options are specified.

## オペランド

The following operand is supported:

*file* A path name of a file to patch.

## 使用法

The -R option does not work with ed scripts because there is too little information to reconstruct the reverse operation.

The -p option makes it possible to customize a patch file to local user directory structures without manually editing the patch file. For example, if the file name in the patch file was `/curds/whey/src/blurfl/blurfl.c`:

- Setting -p 0 gives the entire path name unmodified.
- Setting -p 1 gives:

`curds/whey/src/blurfl/blurfl.c`

- Without the leading slash, `-p 4` gives:

```
blurfl/blurfl.c
```

- Not specifying `-p` at all gives:

```
blurfl.c
```

When using `-b` in some file system implementations, the saving of a `.orig` file might produce unwanted results. In the case of 12-, 13-, or 14-character file names, on file systems supporting 14-character maximum file names, the `.orig` file overwrites the new file.

## 環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `patch`: `LANG`, `LC_ALL`, `LC_COLLATE`, `LC_CTYPE`, `LC_MESSAGES`, `LC_TIME`, and `NLSPATH`.

Affirmative responses are processed using the extended regular expression defined for the `yesexpr` keyword in the `LC_MESSAGES` category of the user's locale. The locale specified in the `LC_COLLATE` category defines the behavior of ranges, equivalence classes, and multi-character collating elements used in the expression defined for `yesexpr`. The locale specified in `LC_CTYPE` determines the locale for interpretation of sequences of bytes of text data a characters, the behavior of character classes used in the expression defined for the `yesexpr`. See [locale\(5\)](#).

## Output Files

The output of `patch` the save files (`.orig` suffixes) and the reject files (`.rej` suffixes) are text files.

## 拡張機能説明

A patch file can contain patching instructions for more than one file. File names are determined as specified in [Patch Determination](#). When the `-b` option is specified, for each patched file, the original is saved in a file of the same name with the suffix `.orig` appended to it.

For each patched file, a reject file can also be created as noted in [Patch Application](#). In the absence of an `-r` option, the name of this file is formed by appending the suffix `.rej` to the original file name.

## Patch File Format

The patch file must contain zero or more lines of header information followed by one or more patches. Each patch must contain zero or more lines of file name identification in the format produced by `diff -c`, and one or more sets of `diff` output, which are customarily called hunks.

`patch` recognizes the following expression in the header information:

Index: *pathname*     The file to be patched is named *pathname*.

If all lines (including headers) within a patch begin with the same leading sequence of blank characters, `patch` removes this sequence before proceeding. Within each patch, if the type of difference is context, `patch` recognizes the following expressions:

- \* \* \* *filename timestamp*      The patches arose from *filename*.
- — — *filename timestamp*      The patches should be applied to *filename*.

Each hunk within a patch must be the `diff` output to change a line range within the original file. The line numbers for successive hunks within a patch must occur in ascending order.

#### File Name Determination

If no *file* operand is specified, `patch` performs the following steps to obtain a path name:

1. If the patch contains the strings `***` and `— — —`, `patch` strips components from the beginning of each path name (depending on the presence or value of the `-p` option), then tests for the existence of both files in the current directory (or directory specified with the `-d` option).
2. If both files exist, `patch` assumes that no path name can be obtained from this step. If the header information contains a line with the string `Index :`, `patch` strips components from the beginning of the path name (depending on `-p`), then tests for the existence of this file in the current directory (or directory specified with the `-d` option).
3. If an SCCS directory exists in the current directory, `patch` attempts to perform a `get -e SCCS/s .filename` command to retrieve an editable version of the file.
4. If no path name can be obtained by applying the previous steps, or if the path names obtained do not exist, `patch` writes a prompt to standard output and request a file name interactively from standard input.

#### Patch Application

If the `-c`, `-e`, `-n`, or `-u` option is present, `patch` interprets information within each hunk as a context difference, an `ed` difference, a normal difference, or a unified context difference, respectively. In the absence of any of these options, `patch` determines the type of difference based on the format of information within the hunk.

For each hunk, `patch` begins to search for the place to apply the patch at the line number at the beginning of the hunk, plus or minus any offset used in applying the previous hunk. If lines matching the hunk context are not found, `patch` scans both forwards and backwards at least 1000 bytes for a set of lines that match the hunk context.

If no such place is found and it is a context difference, then another scan takes place, ignoring the first and last line of context. If that fails, the first two and last two lines of context is ignored and another scan is made. Implementations can search more extensively for installation locations.

If no location can be found, `patch` appends the hunk to the reject file. The rejected hunk is written in context-difference format regardless of the format of the patch file. If the input was a normal or `ed`-style difference, the reject file can contain differences with zero lines of context. The line numbers on the hunks in the reject file can be different from the line numbers in the patch file since they reflect the approximate locations for the failed hunks in the new file rather than the old one.



If the type of patch is an ed diff, the implementation can accomplish the patching by invoking the ed command.

終了ステータス    The following exit values are returned:

- 0      Successful completion.
- 1      One or more lines were written to a reject file.
- >1    An error occurred.

属性                See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE                    |
|---------------------|------------------------------------|
| Availability        | system/core-os                     |
| Interface Stability | Committed                          |
| Standard            | See <a href="#">standards(5)</a> . |

関連項目            [ed\(1\)](#), [diff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pathchk – check path names                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 形式    | pathchk [-p] <i>path</i> ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 機能説明  | <p>The pathchk command will check that one or more path names are valid (that is, they could be used to access or create a file without causing syntax errors) and portable (that is, no filename truncation will result). More extensive portability checks are provided by the -p option.</p> <p>By default, pathchk will check each component of each <i>path</i> operand based on the underlying file system. A diagnostic will be written for each <i>path</i> operand that:</p> <ul style="list-style-type: none"><li>▪ is longer than PATH_MAX bytes.</li><li>▪ contains any component longer than NAME_MAX bytes in its containing directory</li><li>▪ contains any component in a directory that is not searchable</li><li>▪ contains any character in any component that is not valid in its containing directory.</li></ul> <p>The format of the diagnostic message is not specified, but will indicate the error detected and the corresponding <i>path</i> operand.</p> <p>It will not be considered an error if one or more components of a <i>path</i> operand do not exist as long as a file matching the path name specified by the missing components could be created that does not violate any of the checks specified above.</p> |
| オプション | <p>The following option is supported:</p> <p>-p    Instead of performing checks based on the underlying file system, write a diagnostic for each <i>path</i> operand that:</p> <ul style="list-style-type: none"><li>▪ is longer than _POSIX_PATH_MAX bytes</li><li>▪ contains any component longer than _POSIX_NAME_MAX bytes</li><li>▪ contains any character in any component that is not in the portable filename character set.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| オペランド | <p>The following operand is supported:</p> <p><i>path</i>    A path to be checked.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 使用法   | See <a href="#">largefile(5)</a> for the description of the behavior of pathchk when encountering files greater than or equal to 2 Gbyte ( 2 <sup>31</sup> bytes).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 使用例   | <p>例 1 Using the pathchk command</p> <p>To verify that all paths in an imported data interchange archive are legitimate and unambiguous on the current system:</p> <pre>example% pax -f archive   sed -e '/ == ./s///'   xargs pathchk if [ \$? -eq 0 ] then</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

例1 Using the pathchk command (続き)

```

 pax -r -f archive
else
 echo Investigate problems before importing files.
 exit 1
fi

```

To verify that all files in the current directory hierarchy could be moved to any system conforming to the X/Open specification that also supports the [pax\(1\)](#) command:

```

example% find . -print | xargs pathchk -p
if [$? -eq 0]
then
 pax -w -f archive .
else
 echo Portable archive cannot be created.
 exit 1
fi

```

To verify that a user-supplied path names a readable file and that the application can create a file extending the given path without truncation and without overwriting any existing file:

```

example% case $- in
 (C) reset="";;
 *) reset="set +C"
 set -C;;
esac
test -r "$path" && pathchk "$path.out" &&
rm "$path.out" > "$path.out"
if [$? -ne 0]; then
 printf "%s: %s not found or %s.out fails \
creation checks.\n" $0 "$path" "$path"
 $reset # reset the noclobber option in case a trap
 # on EXIT depends on it
 exit 1
fi
$reset
PROCESSING < "$path" > "$path.out"

```

The following assumptions are made in this example:

1. PROCESSING represents the code that will be used by the application to use \$path once it is verified that \$path.out will work as intended.
2. The state of the noclobber option is unknown when this code is invoked and should be set on exit to the state it was in when this code was invoked. (The reset variable is used in this example to restore the initial state.)

例 1 Using the pathchk command (続き)

3. Note the usage of:

`rm "$path.out" > "$path.out"`

- a. The pathchk command has already verified, at this point, that `$path.out` will not be truncated.
- b. With the `noclobber` option set, the shell will verify that `$path.out` does not already exist before invoking `rm`.
- c. If the shell succeeded in creating `$path.out`, `rm` will remove it so that the application can create the file again in the PROCESSING step.
- d. If the PROCESSING step wants the file to exist already when it is invoked, the:

`rm "$path.out" > "$path.out"`

should be replaced with:

`> "$path.out"`

which will verify that the file did not already exist, but leave `$path.out` in place for use by PROCESSING.

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of pathchk: LANG, LC\_ALL, LC\_CTYPE, LC\_MESSAGES, and NLSPATH.

終了ステータス The following exit values are returned:

- `0` All *path* operands passed all of the checks.
- `>0` An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE                    |
|---------------------|------------------------------------|
| Availability        | system/core-os                     |
| Interface Stability | Committed                          |
| Standard            | See <a href="#">standards(5)</a> . |

関連項目 [pax\(1\)](#), [test\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前                  | pax – portable archive interchange                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式                  | <pre> pax [-cdnv] [-H   -L] [-f <i>archive</i>] [-o <i>options</i>]...     [-s <i>replstr</i>]... [<i>pattern</i>]...  pax -r [-cdikuv@/] [-H   -L] [-f <i>archive</i>] [-o <i>options</i>]...     [-p <i>string</i>]... [-s <i>replstr</i>]... [<i>pattern</i>]...  pax -w [-dituvX@/] [-H   -L] [-b <i>blocksize</i>] [-a]     [-f <i>archive</i>] [-o <i>options</i>]... [-s <i>replstr</i>]...     [-x <i>format</i>] [<i>file</i>]...  pax -r -w [-diklntuvX@/] [-H   -L] [-o <i>options</i>]...     [-p <i>string</i>]... [-s <i>replstr</i>]... [<i>file</i>]... <i>directory</i> </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 機能説明                | pax reads, writes, and writes lists of the members of archive files and copies directory hierarchies. A variety of archive formats are supported. See the <i>-x format</i> option.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Modes of Operations | <p>The action to be taken depends on the presence of the <i>-r</i> and <i>-w</i> options. The four combinations of <i>-r</i> and <i>-w</i> are referred to as the four modes of operation: list, read, write, and copy modes, corresponding respectively to the four forms shown in the SYNOPSIS.</p> <p><b>list</b>      In list mode, that is, when neither <i>-r</i> nor <i>-w</i> are specified, pax writes the names of the members of the archive file read from the standard input, with path names matching the specified patterns, to standard output. If a named file has extended attributes, the extended attributes are also listed. If a named file is of type directory, the file hierarchy rooted at that file is listed as well.</p> <p><b>read</b>      In read mode, that is, when <i>-r</i> is specified, but <i>-w</i> is not, pax extracts the members of the archive file read from the standard input, with path names matching the specified patterns. If an extracted file is of type directory, the file hierarchy rooted at that file is extracted as well. The extracted files are created performing path name resolution with the directory in which pax was invoked as the current working directory.</p> <p style="padding-left: 40px;">If an attempt is made to extract a directory when the directory already exists, this is not considered an error. If an attempt is made to extract a FIFO when the FIFO already exists, this is not considered an error.</p> <p style="padding-left: 40px;">The ownership, access and modification times, and file mode of the restored files are discussed under the <i>-p</i> option.</p> <p><b>write</b>      In write mode, that is, when <i>-w</i> is specified, but <i>-r</i> is not, pax writes the contents of the <i>file</i> operands to the standard output in an archive format. If no <i>file</i> operands are specified, a list of files to copy, one per line, are read from the standard input. A file of type directory includes all of the files in the file hierarchy rooted at the file.</p> <p><b>copy</b>      In copy mode, that is, when both <i>-r</i> and <i>-w</i> are specified, pax copies the <i>file</i> operands to the destination directory.</p> |

If no *file* operands are specified, a list of files to copy, one per line, are read from the standard input. A file of type directory includes all of the files in the file hierarchy rooted at the file.

The effect of the copy is as if the copied files were written to an archive file and then subsequently extracted, except that there can be hard links between the original and the copied files. If the destination directory is a subdirectory of one of the files to be copied, the results are unspecified. It is an error if *directory* does not exist, is not writable by the user, or is not a directory.

In read or copy modes, if intermediate directories are necessary to extract an archive member, pax performs actions equivalent to the `mkdir(2)` function, called with the following arguments:

- The intermediate directory used as the *path* argument.
- The octal value of 777 or *rwX* (read, write, and execute permissions) as the *mode* argument (see `chmod(1)`).

If any specified *pattern* or *file* operands are not matched by at least one file or archive member, pax writes a diagnostic message to standard error for each one that did not match and exits with a non-zero exit status.

The supported archive formats are automatically detected on input. The default output archive format is `tar(1)`.

A single archive can span multiple files. pax determines what file to read or write as the next file.

If the selected archive format supports the specification of linked files, it is an error if these files cannot be linked when the archive is extracted, except if the files to be linked are symbolic links and the system is not capable of making hard links to symbolic links. In that case, separate copies of the symbolic link are created instead. Any of the various names in the archive that represent a file can be used to select the file for extraction. For archive formats that do not store file contents with each name that causes a hard link, if the file that contains the data is not extracted during this pax session, either the data is restored from the original file, or a diagnostic message is displayed with the name of a file that can be used to extract the data. In traversing directories, pax detects infinite loops, that is, entering a previously visited directory that is an ancestor of the last file visited. When it detects an infinite loop, pax writes a diagnostic message to standard error and terminates.

## オプション

The following options are supported:

- a                    Appends files to the end of the archive. This option does not work for some archive devices, such as 1/4-inch streaming tapes and 8mm tapes.

- 
- b *blocksize*      Blocks the output at a positive decimal integer number of bytes per write to the archive file. Devices and archive formats can impose restrictions on blocking. Blocking is automatically determined on input. Portable applications must not specify a *blocksize* value larger than 32256. Default blocking when creating archives depends on the archive format. See the -x option below.
  - c                    Matches all file or archive members except those specified by the *pattern* or *file* operands.
  - d                    Causes files of type directory being copied or archived or archive members of type directory being extracted or listed to match only the file or archive member itself and not the file hierarchy rooted at the file.
  - f *archive*          Specifies the path name of the input or output archive, overriding the default standard input (in list or read modes) or standard output (write mode).
  - H                    If a symbolic link referencing a file of type directory is specified on the command line, pax archives the file hierarchy rooted in the file referenced by the link, using the name of the link as the root of the file hierarchy. Otherwise, if a symbolic link referencing a file of any other file type which pax can normally archive is specified on the command line, then pax archives the file referenced by the link, using the name of the link. The default behavior is to archive the symbolic link itself.
  - i                    Interactively renames files or archive members. For each archive member matching a *pattern* operand or file matching a *file* operand, a prompt is written to the file /dev/tty. The prompt contains the name of the file or archive member. A line is then read from /dev/tty. If this line is blank, the file or archive member is skipped. If this line consists of a single period, the file or archive member is processed with no modification to its name. Otherwise, its name is replaced with the contents of the line. pax immediately exits with a non-zero exit status if end-of-file is encountered when reading a response or if /dev/tty cannot be opened for reading and writing.  
  
The results of extracting a hard link to a file that has been renamed during extraction are unspecified.
  - k                    Prevents the overwriting of existing files.
  - l                    Links files. In copy mode, hard links are made between the source and destination file hierarchies whenever possible. If specified in conjunction with -H or -L, when a symbolic link is encountered, the hard link created in the destination file hierarchy is to the file referenced by the symbolic link. If specified when neither -H nor -L is specified, when a symbolic link is encountered, the implementation creates a hard link to the symbolic link in the source file hierarchy or copies the symbolic link to the destination.

- L            If a symbolic link referencing a file of type directory is specified on the command line or encountered during the traversal of a file hierarchy, pax archives the file hierarchy rooted in the file referenced by the link, using the name of the link as the root of the file hierarchy. Otherwise, if a symbolic link referencing a file of any other file type which pax can normally archive is specified on the command line or encountered during the traversal of a file hierarchy, pax archives the file referenced by the link, using the name of the link. The default behavior is to archive the symbolic link itself.
- n            Selects the first archive member that matches each *pattern* operand. No more than one archive member is matched for each pattern, although members of type directory still match the file hierarchy rooted at that file.
- o *options*   Provides information to the implementation to modify the algorithm for extracting or writing files. The value of options consists of one or more comma-separated keywords of the form:

*keyword*[[:]=*value*][,*keyword*[[:]=*value*], ...]

Some keywords apply only to certain file formats, as indicated with each description. Use of keywords that are inapplicable to the file format being processed produces undefined results.

Keywords in the *options* argument must be a string that would be a valid portable filename.

Keywords are not expected to be filenames, merely to follow the same character composition rules as portable filenames.

Keywords can be preceded with white space. The *value* field consists of zero or more characters. Within *value*, the application precedes any literal comma with a backslash, which is ignored, but preserves the comma as part of *value*. A comma as the final character, or a comma followed solely by white space as the final characters, in *options* is ignored. Multiple -o options can be specified. If keywords given to these multiple -o options conflict, the keywords and values appearing later in command line sequence take precedence and the earlier ones are silently ignored. The following keyword values of *options* are supported for the file formats as indicated:

*delete=pattern*

This keyword is applicable only to the -x pax format. When used in write or copy mode, pax omits from extended header records that it produces any keywords matching the string pattern. When used in read or list mode, pax ignores any keywords matching the string pattern in the extended header records. In both cases, matching is performed using the pattern matching notation. For example:

**-o delete=security.\***



would suppress security-related information.

When multiple `-o delete=pattern` options are specified, the patterns are additive. All keywords matching the specified string patterns are omitted from extended header records that `pax` produces.

`exthdr.name=string`

This keyword is applicable only to the `-x pax` format. This keyword allows user control over the name that is written into the `ustar` header blocks for the extended header. The name is the contents of *string*, after the following character substitutions have been made:

- `%d`    The directory name of the file, equivalent to the result of the *dirname* utility on the translated path name.
- `%f`    The filename of the file, equivalent to the result of the *basename* utility on the translated path name.
- `%p`    The process ID of the `pax` process.
- `%%`    A `'%'` character.

Any other `'%'` characters in *string* produce undefined results.

If no `-o exthdr.name=string` is specified, `pax` uses the following default value:

```
%d/PaxHeaders.%p/%f
```

`globexthdr.name=string`

This keyword is applicable only to the `-x pax` format. When used in `write` or `copy` mode with the appropriate options, `pax` creates global extended header records with `ustar` header blocks that are treated as regular files by previous versions of `pax`. This keyword allows user control over the name that is written into the `ustar` header blocks for global extended header records. The name is the contents of *string*, after the following character substitutions have been made:

- `%n`    An integer that represents the sequence number of the global extended header record in the archive, starting at 1.
- `%p`    The process ID of the `pax` process.
- `%%`    A `'%'` character.

Any other `'%'` characters in *string* produce undefined results.

If no `-o globexthdr.name=string` is specified, `pax` uses the following default value:

```
$TMPDIR/GlobalHead.%p.%n
```

where `$TMPDIR` represents the value of the `TMPDIR` environment variable. If `TMPDIR` is not set, `pax` uses `/tmp`.

`invalid=action`

This keyword is applicable only to the `-x pax` format. This keyword allows user control over the action `pax` takes upon encountering values in an extended header record that, in `read` or `copy` mode, are invalid in the destination hierarchy or, in `list` mode, cannot be written in the codeset and current locale of the implementation. The following are invalid values that are recognized by `pax`:

- In `read` or `copy` mode, a filename or link name that contains character encodings invalid in the destination hierarchy. For example, the name can contain embedded NULs.
- In `read` or `copy` mode, a filename or link name that is longer than the maximum allowed in the destination hierarchy, for either a path name component or the entire path name.
- In `list` mode, any character string value (filename, link name, user name, and so on) that cannot be written in the codeset and current locale of the implementation.

The following mutually-exclusive values of the `action` argument are supported:

|                     |                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>bypass</code> | In <code>read</code> or <code>copy</code> mode, <code>pax</code> bypasses the file, causing no change to the destination hierarchy. In <code>list</code> mode, <code>pax</code> writes all requested valid values for the file, but its method for writing invalid values is unspecified.                                                                             |
| <code>rename</code> | In <code>read</code> or <code>copy</code> mode, <code>pax</code> acts as if the <code>-i</code> option were in effect for each file with invalid filename or link name values, allowing the user to provide a replacement name interactively. In <code>list</code> mode, <code>pax</code> behaves identically to the <code>bypass</code> action.                      |
| <code>UTF-8</code>  | <code>pax</code> uses the actual UTF-8 encoding for the name when it is used in <code>read</code> , <code>copy</code> , or <code>list</code> mode and a filename, link name, owner name, or any other field in an extended header record cannot be translated from the <code>pax UTF-8</code> codeset format to the codeset and current locale of the implementation. |
| <code>write</code>  | In <code>read</code> or <code>copy</code> mode, <code>pax</code> writes the file, translating the name, regardless of whether this can overwrite an existing file with a valid name. In <code>list</code> mode, <code>pax</code> behaves identically to the <code>bypass</code> action.                                                                               |

If no `-o invalid=` option is specified, `pax` acts as if `-o invalid=bypass` were specified. Any overwriting of existing files that can be allowed by the

-o *invalid=actions* are subject to permission (-p) and modification time (-u) restrictions, and are suppressed if the -k option is also specified.

#### linkdata

This keyword is applicable only to the -x pax format. In write mode, pax writes the contents of a file to the archive even when that file is merely a hard link to a file whose contents have already been written to the archive.

#### listopt=*format*

This keyword specifies the output format of the table of contents produced when the -v option is specified in list mode. (See List Mode Format Specifications below.) To avoid ambiguity, the *listopt=format* is the only or final *keyword=value* pair in an -o option-argument. All characters in the remainder of the option-argument are considered to be part of the format string. When multiple -o *listopt=format* options are specified, the format strings are considered to be a single, concatenated string, evaluated in command line order.

#### times

This keyword is applicable only to the -x pax and -x xustar formats. When used in write or copy mode, pax includes *atime* and *mtime* extended header records for each file.

In addition to these keywords, if the -x pax format is specified, any of the keywords and values, including implementation extensions, can be used in -o option-arguments, in either of two modes:

**keyword=*value*** When used in write or copy mode, these keyword/value pairs are included at the beginning of the archive as *typeflag g* global extended header records. When used in read or list mode, these keyword/value pairs act as if they had been at the beginning of the archive as *typeflag g* global extended header records.

**keyword:=*value*** When used in write or copy mode, these keyword/value pairs are included as records at the beginning of a *typeflag x* extended header for each file. This is equivalent to the equal-sign form except that it creates no *typeflag g* global extended header records. When used in read or list mode, these keyword/value pairs act as if they were included as records at the end of each extended header. Thus, they override any global or file-specific extended header record keywords of the same names. For example, in the command:

```
pax -r -o "
gname:=mygroup,
" <archive
```

the group name is forced to a new value for all files read from the archive.

**-p *string*** Specifies one or more file characteristic options (privileges). The *string* option-argument must be a string specifying file characteristics to be retained or discarded on extraction. The string consists of the specification characters a, e, m, o, and p. Multiple characteristics can be concatenated within the same string and multiple -p options can be specified. The meaning of the specification characters is as follows:

- a Does not preserve file access times.
- e Preserves the user ID, group ID, file mode bits, access time, and modification time.
- m Does not preserve file modification times.
- o Preserves the user ID and group ID.
- p Preserves the file mode bits.

In the preceding list, preserve indicates that an attribute stored in the archive is given to the extracted file, subject to the permissions of the invoking process. Otherwise, the attribute is determined as part of the normal file creation action. The access and modification times of the file is preserved unless otherwise specified with the -p option or not stored in the archive. All attributes that are not preserved are determined as part of the normal file creation action.

If neither the e nor the o specification character is specified, or the user ID and group ID are not preserved for any reason, pax does not set the setuid and setgid bits of the file mode.

If the preservation of any of these items fails for any reason, pax writes a diagnostic message to standard error. Failure to preserve these items affects the final exit status, but does not cause the extracted file to be deleted.

If file-characteristic letters in any of the *string* option-arguments are duplicated or conflict with each other, the ones given last take precedence. For example, if -p eme is specified, file modification times are preserved.

**-r** Reads an archive file from standard input.

**-s *replstr*** Modifies file or archive member names named by *pattern* or *file* operands according to the substitution expression *replstr*, which is based on the [ed\(1\)](#) s (substitution) utility, using the regular expression syntax of [regex\(5\)](#). The concepts of “address” and “line” are meaningless in the context of the pax command, and must not be supplied. The format is:

`-s /old/new/ [gp]`

where, as in `ed`, *old* is a basic regular expression and *new* can contain an ampersand (&), a `\n` backreference, where *n* is a digit, or subexpression matching. The *old* string is also permitted to contain newlines.

Any non-null character can be used as a delimiter (/ shown here). Multiple `-s` expressions can be specified. The expressions are applied in the order specified, terminating with the first successful substitution. The optional trailing `g` is as defined in the `ed` command. The optional trailing `p` causes successful substitutions to be written to standard error. File or archive member names that substitute to the empty string are ignored when reading and writing archives.

`-t` When reading files from the file system, and if the user has the permissions required by `utime()` to do so, sets the access time of each file read to the access time that it had before being read by `pax`.

`-u` Ignores files that are older (having a less recent file modification time) than a pre-existing file or archive member with the same name.

`read mode` An archive member with the same name as a file in the file system is extracted if the archive member is newer than the file.

`write mode` An archive file member with the same name as a file in the file system is superseded if the file is newer than the archive member. If option `-a` is also specified, this is accomplished by appending to the archive. Otherwise, it is unspecified whether this is accomplished by actual replacement in the archive or by appending to the archive.

`copy mode` The file in the destination hierarchy is replaced by the file in the source hierarchy or by a link to the file in the source hierarchy if the file in the source hierarchy is newer.

`-v` In `list` mode, produces a verbose table of contents (see `Standard Output`). Otherwise, writes archive member path names and extended attributes to standard error (see `Standard Error`).

`-w` Writes files to the standard output in the specified archive format.

`-x format` Specifies the output archive format. The `pax` utility recognizes the following formats:

`cpio` The extended `cpio(1)` interchange format. See IEEE Std 1003.1–2001. The default *blocksize* for this format for character special archive files is 5120. Implementations support all *blocksize* values less than or equal to 32256 that are multiples of 512.

This archive format allows files with UIDs and GIDs up to 262143 to be stored in the archive. Files with UIDs and GIDs greater than this value are archived with the UID and GID of 60001.

**pax** The pax interchange format. See IEEE Std 1003.1–2001. The default *blocksize* for this format for character special archive files is 5120. Implementations support all *blocksize* values less than or equal to 32256 that are multiples of 512.

Similar to *ustar*. Also allows archiving and extracting files whose size is greater than 8GB; whose UID, GID, *devmajor*, or *devminor* values are greater than 2097151; whose path (including filename) is greater than 255 characters; or whose *linkname* is greater than 100 characters.

**ustar** The extended [tar\(1\)](#) interchange format. See the IEEE 1003.1(1990) specifications. The default *blocksize* for this format for character special archive files is 10240. Implementations support all *blocksize* values less than or equal to 32256 that are multiples of 512.

This archive format allows files with UIDs and GIDs up to 2097151 to be stored in the archive. Files with UIDs and GIDs greater than this value are archived with the UID and GID of 60001.

**xustar** Similar to *ustar*. Also allows archiving and extracting files whose size is greater than 8GB; whose UID, GID, *devmajor*, or *devminor* values are greater than 2097151; whose path (including filename) is greater than 255 characters; or whose *linkname* is greater than 100 characters. This option should not be used if the archive is to be extracted by an archiver that cannot handle the larger values.

Any attempt to append to an archive file in a format different from the existing archive format causes pax to exit immediately with a non-zero exit status.

In copy mode, if no -x format is specified, pax behaves as if -x pax were specified.

- X When traversing the file hierarchy specified by a path name, pax does not descend into directories that have a different device ID (*st\_dev*, see [stat\(2\)](#)).
- @ Includes extended attributes in the archive. pax does not place extended attributes in the archive by default.

When traversing the file hierarchy specified by a path name, pax descends into the attribute directory for any file with extended attributes. Extended attributes go into the archive as special files.

When this flag is used during file extraction, any extended attributes associated with a file being extracted are also extracted. Extended attribute files can only be extracted from an archive as part of a normal file extract. Attempts to explicitly extract attribute records are ignored.

- / Includes extended system attributes in the archive. pax does not place extended system attributes in the archive by default.

When traversing the file hierarchy specified by a path name, pax descends into the attribute directory for any file with extended attributes. Extended attributes go into the archive as special files. When this flag is used during file extraction, any extended attributes associated with a file being extracted are also extracted. Extended attribute files can only be extracted from an archive as part of a normal file extract. Attempts to explicitly extract attribute records are ignored.

Specifying more than one of the mutually-exclusive options -H and -L is not considered an error. The last option specified determines the behavior of the utility.

The options that operate on the names of files or archive members (-c, -i, -n, -s, -u and -v) interact as follows.

In read mode, the archive members are selected based on the user-specified *pattern* operands as modified by the -c, -n and -u options. Then, any -s and -i options modify, in that order, the names of the selected files. The -v option writes names resulting from these modifications.

In write mode, the files are selected based on the user-specified path names as modified by the -n and -u options. Then, any -s and -i options modify, in that order, the names of these selected files. The -v option writes names resulting from these modifications.

If both the -u and -n options are specified, pax does not consider a file selected unless it is newer than the file to which it is compared.

#### List Mode Format Specifications

In list mode with the -o listopt=*format* option, the format argument is applied for each selected file. pax appends a NEWLINE to the listopt output for each selected file. The *format* argument is used as the format string with the following exceptions. (See [printf\(1\)](#) for the first five exceptions.)

1. A SPACE character in the format string, in any context other than a flag of a conversion specification, is treated as an ordinary character that is copied to the output.
2. A ' ' character in the format string is treated as a ' ' character, not as a SPACE.

3. In addition to the escape sequences described in the [formats\(5\)](#) manual page, (`\\`, `\a`, `\b`, `\f`, `\n`, `\r`, `\t`, `\v`), `\ddd`, where *ddd* is a one-, two-, or three-digit octal number, is written as a byte with the numeric value specified by the octal number.
4. Output from the `d` or `u` conversion specifiers is not preceded or followed with BLANKs not specified by the format operand.
5. Output from the `o` conversion specifier is not preceded with zeros that are not specified by the format operand.
6. The sequence (*keyword*) can occur before a format conversion specifier. The conversion argument is defined by the value of *keyword*. The following keywords are supported (see IEEE Std 1003.1–2001):
  - Any of the Field Name entries in ustar Header Block and Octet-Oriented cpio Archive Entry. The implementation supports the cpio keywords without the leading `c_` in addition to the form required by Values for cpio `c_ mode` Field.
  - Any keyword defined for the extended header in pax Extended Header.
  - Any keyword provided as an implementation-defined extension within the extended header defined in pax Extended Header.

For example, the sequence “`%(charset)s`” is the string value of the name of the character set in the extended header.

The result of the keyword conversion argument is the value from the applicable header field or extended header, without any trailing NULs.

All keyword values used as conversion arguments are translated from the UTF-8 encoding to the character set appropriate for the local file system, user database, and so on, as applicable.

7. An additional conversion specifier character, `T`, is used to specify time formats. The `T` conversion specifier character can be preceded by the sequence (*keyword=**subformat*), where *subformat* is a date format as defined by *date* operands. The default *keyword* is *mtime* and the default *subformat* is:
 

```
%b %e %H:%M %Y
```
8. An additional conversion specifier character, `M`, is used to specify the file mode string as defined in `ls` Standard Output. If (*keyword*) is omitted, the mode keyword is used. For example, `%.1M` writes the single character corresponding to the *entry type* field of the `ls -l` command.
9. An additional conversion specifier character, `D`, is used to specify the device for block or special files, if applicable, in an implementation-defined format. If not applicable, and (*keyword*) is specified, then this conversion is equivalent to `%(keyword)u`. If not applicable, and (*keyword*) is omitted, then this conversion is equivalent to SPACE.



10. An additional conversion specifier character, F, is used to specify a path name. The F conversion character can be preceded by a sequence of comma-separated keywords:

*(keyword[,keyword] ... )*

The values for all the keywords that are non-null are concatenated, each separated by a '/'. The default is (path) if the keyword path is defined. Otherwise, the default is (*prefix*,name).

11. An additional conversion specifier character, L, is used to specify a symbolic link expansion. If the current file is a symbolic link, then %L expands to:

*"%s -> %s", value of keyword, contents of link*

Otherwise, the %L conversion specification is the equivalent of %F.

## オペランド

The following operands are supported:

*directory*     The destination directory path name for copy mode.

*file*            A path name of a file to be copied or archived.

*pattern*        A pattern matching one or more path names of archive members. A pattern must conform to the pattern matching notation found on the [fnmatch\(5\)](#) manual page. The default, if no *pattern* is specified, is to select all members in the archive.

## Output

Output formats are discussed below:

### Standard Output

In write mode, if -f is not specified, the standard output is the archive formatted according to one of the formats described below. See -x format for a list of supported formats.

In list mode, when the -o listopt=*format* option has been specified, the selected archive members are written to standard output using the format described above under List Mode Format Specifications. In list mode without the -o listopt=*format* option, the table of contents of the selected archive members are written to standard output using the following format:

*"%s\n", pathname*

If the -v option is specified in list mode, the table of contents of the selected archive members are written to standard output using the following formats:

- For path names representing hard links to previous members of the archive:

*"%s == %s\n", <ls -l listing, linkname*

- For all other path names:

*"%s\n", <ls -l listing>*

where *<ls -l listing>* is the format specified by the ls command with the -l option. When writing path names in this format, it is unspecified what is written for fields for which the

underlying archive format does not have the correct information, although the correct number of blank-character-separated fields is written.

In `list` mode, standard output is not buffered more than a line at a time.

**Standard Error** If `-v` is specified in `read`, `write` or `copy` modes, `pax` writes the path names it processes to the standard error output using the following format:

`"%s\n", pathname`

These path names are written as soon as processing is begun on the file or archive member, and are flushed to standard error. The trailing NEWLINE character, which is not buffered, is written when the file has been read or written.

If the `-s` option is specified, and the replacement string has a trailing `p`, substitutions are written to standard error in the following format:

`"%s >> %s\n", <original pathname>, <new pathname>`

In all operating modes of `pax`, optional messages of unspecified format concerning the input archive format and volume number, the number of files, blocks, volumes, and media parts as well as other diagnostic messages can be written to standard error.

In all formats, for both standard output and standard error, it is unspecified how non-printable characters in path names or link names are written.

When `pax` is in `read` mode or `list` mode, using the `-x pax` archive format, and a file name, link name, owner name, or any other field in an extended header record cannot be translated from the `pax` UTF-8 codeset format to the codeset and current locale of the implementation, `pax` writes a diagnostic message to standard error, processes the file as described for the `-o invalid=option`, and then processes the next file in the archive.

**Output Files** In `read` mode, the extracted output files are of the archived file type. In `copy` mode, the copied output files are the type of the file being copied. In either mode, existing files in the destination hierarchy are overwritten only when all permission (`-p`), modification time (`-u`), and invalid-value (`-o invalid=`) tests allow it. In `write` mode, the output file named by the `-f` option-argument is a file formatted according to one of the specifications in IEEE Std 1003.1–2001.

**エラー** If `pax` cannot create a file or a link when reading an archive, or cannot find a file when writing an archive, or cannot preserve the user ID, group ID, or file mode when the `-p` option is specified, a diagnostic message is written to standard error and a non-zero exit status is returned, but processing continues. In the case where `pax` cannot create a link to a file, `pax` does not, by default, create a second copy of the file.

If the extraction of a file from an archive is prematurely terminated by a signal or error, `pax` can have only partially extracted the file or, if the `-n` option was not specified, can have

extracted a file of the same name as that specified by the user, but which is not the file the user wanted. Additionally, the file modes of extracted directories can have additional bits from the read, write, execute mask set as well as incorrect modification and access times.

## 使用法

The `-p` (privileges) option was invented to reconcile differences between historical [tar\(1\)](#) and [cpio\(1\)](#) implementations. In particular, the two utilities use `-m` in diametrically opposed ways. The `-p` option also provides a consistent means of extending the ways in which future file attributes can be addressed, such as for enhanced security systems or high-performance files. Although it can seem complex, there are really two modes that are most commonly used:

- p e     Preserve everything. This would be used by the historical superuser, someone with all the appropriate privileges, to preserve all aspects of the files as they are recorded in the archive. The `e` flag is the sum of `o` and `p`, and other implementation-dependent attributes.
- p p     Preserve the file mode bits. This would be used by the user with regular privileges who wished to preserve aspects of the file other than the ownership. The file times are preserved by default, but two other flags are offered to disable these and use the time of extraction.

The one path name per line format of standard input precludes path names containing newlines. Although such path names violate the portable filename guidelines, they can exist and their presence can inhibit usage of `pax` within shell scripts. This problem is inherited from historical archive programs. The problem can be avoided by listing file name arguments on the command line instead of on standard input.

It is almost certain that appropriate privileges are required for `pax` to accomplish parts of this. Specifically, creating files of type block special or character special, restoring file access times unless the files are owned by the user (the `-t` option), or preserving file owner, group, and mode (the `-p` option) all probably require appropriate privileges.

In read mode, implementations are permitted to overwrite files when the archive has multiple members with the same name. This can fail if permissions on the first version of the file do not permit it to be overwritten.

When using the `-x xustar` and `-x -pax` archive formats, if the underlying file system reports that the file being archived contains holes, the Solaris `pax` utility records the presence of holes in an extended header record when the file is archived. If this extended header record is associated with a file in the archive, those holes are recreated whenever that file is extracted from the archive. See the `SEEK_DATA` and `SEEK_HOLE` whence values in [lseek\(2\)](#). In all other cases, any NUL (`\0`) characters found in the archive is written to the file when it is extracted.

See [largefile\(5\)](#) for the description of the behavior of `pax` when encountering files greater than or equal to 2 Gbyte ( $2^{31}$  bytes).

**Standard Input** In write mode, the standard input is used only if no *file* operands are specified. It is a text file containing a list of path names, one per line, without leading or trailing blanks. In list and read modes, if -f is not specified, the standard input is an archive file. Otherwise, the standard input is not used.

**Input Files** The input file named by the *archive* option-argument, or standard input when the archive is read from there, is a file formatted according to one of the formats described below. See **Extended Description**. The file /dev/tty is used to write prompts and read responses.

## 使用例

例1 Copying the Contents of the Current Directory

The following command:

```
example% pax -w -f /dev/rmt1m .
```

copies the contents of the current directory to tape drive 1, medium density. This assumes historical System V device naming procedures. The historical BSD device name would be /dev/rmt9.

例2 Copying the Directory Hierarchy

The following commands:

```
example% mkdir newdir
example% pax -rw olddir newdir
```

copy the olddir directory hierarchy to newdir.

例3 Reading an Archive Extracted Relative to the Current Directory

The following command:

```
example% pax -r -s ',^/*usr/*,,', -f a.pax
```

reads the archive a.pax, with all files rooted in /usr in the archive extracted relative to the current directory.

例4 Overriding the Default Output Description

Using the option:

```
-o listopt="%M %(atime)T %(size)D %(name)s"
```

overrides the default output description in Standard Output and instead writes:

```
-rw-rw- - - Jan 12 15:53 2003 1492 /usr/foo/bar
```

Using the options:

```
-o listopt='%L\t%(size)D\n%.7' \
-o listopt='(name)s\n%(atime)T\n%T'
```

## 例 4 Overriding the Default Output Description (続き)

overrides the default output description in standard output and instead writes:

```
usr/foo/bar -> /tmp 1492
/usr/foo
Jan 12 15:53 1991
Jan 31 15:53 2003
```

## 環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of pax: LANG, LC\_ALL, LC\_CTYPE, LC\_MESSAGES, LC\_TIME, and NLS\_PATH.

|            |                                                                                                                                                                                                                                                                                                                                                |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LC_COLLATE | Determine the locale for the behaviour of ranges, equivalence classes, and multi-character collating elements used in the pattern matching expressions for the <i>pattern</i> operand, the basic regular expression for the -s option, and the extended regular expression defined for the yesexpr locale keyword in the LC_MESSAGES category. |
| TMPDIR     | Determine the path name that provides part of the default global extended header record file, as described for the -o globexthdr= keyword as described in the OPTIONS section.                                                                                                                                                                 |
| TZ         | Determine the timezone used to calculate date and time strings when the -v option is specified. If TZ is unset or null, an unspecified default timezone is used.                                                                                                                                                                               |

## 終了ステータス

The following exit values are returned:

- 0 All files were processed successfully.
- >0 An error occurred.

## 拡張機能説明

pax Interchange  
Format

A pax archive tape or file produced in the -xpax format contains a series of blocks. The physical layout of the archive is identical to the *ustar* format described in *ustar Interchange Format*. Each file archived is represented by the following sequence:

- An optional header block with extended header records. This header block is of the form 27403 with a *typeflag* value of x or g. The extended header records is included as the data for this header block.
- A header block that describes the file. Any fields in the preceding optional extended header overrides the associated fields in this header block for this file.
- Zero or more blocks that contain the contents of the file.

At the end of the archive file there are two 512-byte blocks filled with binary zeroes, interpreted as an end-of-archive indicator.

The following is a schematic of an example archive with global extended header records and two actual files in pax format archive. In the example, the second file in the archive has no extended header preceding it, presumably because it has no need for extended attributes.

| Description                            | Block                              |
|----------------------------------------|------------------------------------|
| Global Extended Header                 | ustar Header [ <i>typeflag</i> =g] |
|                                        | Global Extended Header Data        |
| File 1: Extended Header is included    | ustar Header [ <i>typeflag</i> =x] |
|                                        | Extended Header Data               |
|                                        | [ <i>typeflag</i> =0]              |
|                                        | ustar Header Data for File 1       |
| File 2: No Extended Header is included | ustar Header [ <i>typeflag</i> =0] |
|                                        | Data for File2                     |
| End of Archive Indicator               | Block of binary zeros              |
|                                        | Block of binary zeros              |

pax Header Block

The pax header block is identical to the ustar header block described in ustar Interchange Format except that two additional *typeflag* values are defined:

- g Represents global extended header records for the following files in the archive. The format of these extended header records are as described in pax Extended Header. Each value affects all subsequent files that do not override that value in their own extended header record and until another global extended header record is reached that provides another value for the same field. The *typeflag* g global headers should not be used with interchange media that could suffer partial data loss in transporting the archive.
- x Represents extended header records for the following file in the archive (which has its own ustar header block). The format of these extended header records is as described in pax Extended Header.

For both of these types, the *size* field is the size of the extended header records in octets. The other fields in the header block are not meaningful to this version of pax. However, if this archive is read by pax conforming to a previous version of ISO POSIX-2:1993 Standard, the header block fields are used to create a regular file that contains the extended header records as data. Therefore, header block field values should be selected to provide reasonable file access to this regular file.

A further difference from the ustar header block is that data blocks for files of *typeflag* 1 (the digit one) (hard link) might be included, which means that the *size* field can be greater than zero. Archives created by pax -o linkdata includes these data blocks with the hard links.

**pax Extended Header**     A pax extended header contains values that are inappropriate for the `ustar` header block because of limitations in that format: fields requiring a character encoding other than that described in the ISO/IEC 646: 1991 standard, fields representing file attributes not described in the `ustar` header, and fields whose format or length do not fit the requirements of the `ustar` header. The values in an extended header add attributes to the specified file or files or override values in the specified header blocks, as indicated in the following list of keywords. See the description of the `typeflag` header block.

An extended header consists of one or more records, each constructed as follows:

```
"%d %s=%s\n", length, keyword, value
```

The extended header records are encoded according to the ISO/IEC 10646-1: 2000 standard (UTF-8). *length*, BLANK, equals sign (=), and NEWLINE are limited to the portable character set, as encoded in UTF-8. *keyword* and *value* can be any UTF-8 characters. *length* is the decimal length of the extended header record in octets, including the trailing NEWLINE.

*keyword* is one of the entries from the following list or a keyword provided as an implementation extension. Keywords consisting entirely of lowercase letters, digits, and periods are reserved for future standardization. A keyword does not include an equals sign.

In the following list, the notation of `file(s)` or `block(s)` are used to acknowledge that a keyword affects the specified single file after a `typeflag x` extended header, but possibly multiple files after `typeflag g`. Any requirements in the list for pax to include a record when in write or copy mode applies only when such a record has not already been provided through the use of the `-o` option. When used in copy mode, pax behaves as if an archive had been created with applicable extended header records and then extracted.

- atime**                     The file access time for the specified files, equivalent to the value of the `st_atime` member of the `stat` structure for a file, as described by the [stat\(2\)](#) function. The access time (atime) is restored if the process has the appropriate privilege required to do so. The format of the *value* is as described in `pax Extended Header File Times`.
- charset**                     The name of the character set used to encode the data in the specified files. The entries in the following table are defined to refer to known standards; additional names can be agreed on between the originator and recipient.

| <i>value</i>       | Formal Standard     |
|--------------------|---------------------|
| ISO-IR 646 1990    | ISO/IEC646:1990     |
| ISO-IR 8859 1 1998 | ISO/IEC8859-1:1998  |
| ISO-IR 8859 2 1999 | ISO/IEC 8859-2:1999 |
| ISO-IR 8859 3 1999 | ISO/IEC 8859-3:1999 |

|                         |                              |
|-------------------------|------------------------------|
| ISO-IR 8859 4 1999      | ISO/IEC8859-4:1998           |
| ISO-IR 8859 5 1999      | ISO/IEC8859-5-1999           |
| ISO-IR 8859 6 1999      | ISO/IEC8859-6-1999           |
| ISO-IR 8859 7 1987      | ISO/IEC8859-7:1987           |
| ISO-IR 8859 8 1999      | ISO/IEC8859-8:1999           |
| ISO-IR 8859 9 1999      | ISO/IEC8859-9:1999           |
| ISO-IR 8859 10 1998     | ISO/IEC8859-10:1999          |
| ISO-IR 8859 13 1998     | ISO/IEC8859-13:1998          |
| ISO-IR 8859 14 1998     | ISO/IEC8859-14:1998          |
| ISO-IR 8859 15 1999     | ISO/IEC8859-15:1999          |
| ISO-IR 10646 2000       | ISO/IEC 10646:2000           |
| ISO-IR 10646 2000 UTF-8 | ISO/IEC 10646,UTF-8 encoding |
| BINARY                  | None                         |

The encoding is included in an extended header for information only; when pax is used as described in IEEE Std 1003.1-200x, it does not translate the file data into any other encoding. The BINARY entry indicates unencoded binary data. When used in write or copy mode, it is implementation-defined whether pax includes a charset extended header record for a file.

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| comment | A series of characters used as a comment. All characters in the <i>value</i> field are ignored by pax.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| gid     | The group ID of the group that owns the file, expressed as a decimal number using digits from the ISO/IEC 646: 1991 standard. This record overrides the gid field in the specified header blocks. When used in write or copy mode, pax includes a gid extended header record for each file whose group ID is greater than 2097151 (octal 7777777).                                                                                                                                                                                                                                                |
| gname   | The group of the files, formatted as a group name in the group database. This record overrides the gid and <i>gname</i> fields in the specified header blocks, and any gid extended header record. When used in read, copy, or list mode, pax translates the name from the UTF-8 encoding in the header record to the character set appropriate for the group database on the receiving system. If any of the UTF-8 characters cannot be translated, and if the -o invalid=UTF-8 option is not specified, the results are implementation-defined. When used in write or copy mode, pax includes a |



---

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     | <i>gname</i> extended header record for each file whose group name cannot be represented entirely with the letters and digits of the portable character set.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <i>linkpath</i>     | The pathname of a link being created to another file, of any type, previously archived. This record overrides the <i>linkname</i> field in the specified ustar header blocks. The specified ustar header block determines the type of link created. If <i>typeflag</i> of the specified header block is 1, it is a hard link. If <i>typeflag</i> is 2, it is a symbolic link and the <i>linkpath</i> value is the contents of the symbolic link. pax translates the name of the link (contents of the symbolic link) from the UTF-8 encoding to the character set appropriate for the local file system. When used in write or copy mode, pax includes a <i>linkpath</i> extended header record for each link whose pathname cannot be represented entirely with the members of the portable character set other than NULL. |
| <i>mtime</i>        | The pathname of a link being created to another file, of any type, previously archived. This record overrides the <i>linkname</i> field in the specified ustar header blocks. The specified ustar header block determines the type of link created. If <i>typeflag</i> of the specified header block is 1, it is a hard link. If <i>typeflag</i> is 2, it is a symbolic link and the <i>linkpath</i> value is the contents of the symbolic link. pax translates the name of the link (contents of the symbolic link) from the UTF-8 encoding to the character set appropriate for the local file system. When used in write or copy mode, pax includes a <i>linkpath</i> extended header record for each link whose pathname cannot be represented entirely with the members of the portable character set other than NULL. |
| <i>path</i>         | The pathname of the specified files. This record overrides the name and <i>prefix</i> fields in the specified header blocks. pax translates the pathname of the file from the UTF-8 encoding to the character set appropriate for the local file system. When used in write or copy mode, pax includes a <i>path</i> extended header record for each file whose pathname cannot be represented entirely with the members of the portable character set other than NULL.                                                                                                                                                                                                                                                                                                                                                     |
| <i>realtime.any</i> | The keywords prefixed by <i>realtime</i> are reserved for future standardization.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>security.any</i> | The keywords prefixed by <i>security</i> are reserved for future standardization.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <i>size</i>         | The size of the file in octets, expressed as a decimal number using digits from the ISO/IEC 646: 1991 standard. This record overrides the <i>size</i> field in the specified header blocks. When used in write or copy mode, pax includes a <i>size</i> extended header record for each file with a <i>size</i> value greater than 8589934591 (octal 7777777777).                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

|              |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>uid</i>   | The user ID of the file owner, expressed as a decimal number using digits from the ISO/IEC 646:1991 standard. This record overrides the <i>uid</i> field in the following header block(s). When used in write or copy mode, pax includes a <i>uid</i> extended header record for each file whose owner ID is greater than 2097151 (octal 7777777).                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <i>uname</i> | The owner of the specified files, formatted as a user name in the user database. This record overrides the <i>uid</i> and <i>uname</i> fields in the specified header blocks, and any <i>uid</i> extended header record. When used in read, copy, or list mode, pax translates the name from the UTF-8 encoding in the header record to the character set appropriate for the user database on the receiving system. If any of the UTF-8 characters cannot be translated, and if the <i>-o invalid=UTF-8</i> option is not specified, the results are implementation-defined. When used in write or copy mode, pax includes a <i>uname</i> extended header record for each file whose user name cannot be represented entirely with the letters and digits of the portable character set. |

If the *value* field is zero length, it deletes any header block field, previously entered extended header value, or global extended header value of the same name.

If a keyword in an extended header record (or in an *-o* option-argument) overrides or deletes a corresponding field in the *ustar* header block, pax ignores the contents of that header block field.

Unlike the *ustar* header block fields, *NULLs* does not delimit values; all characters within the *value* field are considered data for the field.

#### pax Extended Header Keyword Precedence

This section describes the precedence in which the various header records and fields and command line options are selected to apply to a file in the archive. When pax is used in read or list modes, it determines a file attribute in the following sequence:

1. If *-o delete=keyword-prefix* is used, the affected attributes is determined from step 7, if applicable, or ignored otherwise.
2. If *-o keyword:=* is used, the affected attributes is ignored.
3. If *-o keyword=value* is used, the affected attribute is assigned the value.
4. If there is a *typeflag x* extended header record, the affected attribute is assigned the value. When extended header records conflict, the last one given in the header takes precedence.
5. If *-o keyword=value* is used, the affected attribute is assigned the value.
6. If there is a *typeflag g* global extended header record, the affected attribute is assigned the value. When global extended header records conflict, the last one given in the global header takes precedence.
7. Otherwise, the attribute is determined from the *ustar* header block.

pax Extended Header File Times

pax writes an *mtime* record for each file in write or copy modes if the file's modification time cannot be represented exactly in the *ustar* header logical record described in *ustar* Interchange Format. This can occur if the time is out of *ustar* range, or if the file system of the underlying implementation supports non-integer time granularities and the time is not an integer. All of these time records are formatted as a decimal representation of the time in seconds since the Epoch. If a period ( . ) decimal point character is present, the digits to the right of the point represents the units of a sub-second timing granularity, where the first digit is tenths of a second and each subsequent digit is a tenth of the previous digit. In read or copy mode, pax truncates the time of a file to the greatest value that is not greater than the input header file time. In write or copy mode, pax outputs a time exactly if it can be represented exactly as a decimal number, and otherwise generates only enough digits so that the same time is recovered if the file is extracted on a system whose underlying implementation supports the same time granularity.

ustar Interchange Format

A *ustar* archive tape or file contains a series of logical records. Each logical record is a fixed-size logical record of 512 octets. Although this format can be thought of as being stored on 9-track industry-standard 12.7mm (0.5 in) magnetic tape, other types of transportable media are not excluded. Each file archived is represented by a header logical record that describes the file, followed by zero or more logical records that give the contents of the file. At the end of the archive file there are two 512-octet logical records filled with binary zeros, interpreted as an end-of-archive indicator.

The logical records can be grouped for physical I/O operations, as described under the *-bblocksize* and *-x* *ustar* options. Each group of logical records can be written with a single operation equivalent to the `write(2)` function. On magnetic tape, the result of this write is a single tape physical block. The last physical block always is the full size, so logical records after the two zero logical records can contain undefined data.

The header logical record is structured as shown in the following table. All lengths and offsets are in decimal.

表 1   ustar Header Block

| Field Name    | Octet Offset | Length (in Octets) |
|---------------|--------------|--------------------|
| <i>name</i>   | 0            | 100                |
| <i>mode</i>   | 100          | 8                  |
| <i>uid</i>    | 108          | 8                  |
| <i>gid</i>    | 116          | 8                  |
| <i>size</i>   | 124          | 12                 |
| <i>mtime</i>  | 136          | 12                 |
| <i>chksum</i> | 148          | 8                  |

表 1   ustar Header Block      (続き)

| Field Name      | Octet Offset | Length (in Octets) |
|-----------------|--------------|--------------------|
| <i>typeflag</i> | 156          | 1                  |
| <i>linkname</i> | 157          | 100                |
| <i>magic</i>    | 257          | 6                  |
| <i>version</i>  | 263          | 2                  |
| <i>uname</i>    | 265          | 32                 |
| <i>gname</i>    | 297          | 32                 |
| <i>devmajor</i> | 329          | 8                  |
| <i>devminor</i> | 337          | 8                  |
| <i>prefix</i>   | 345          | 155                |

All characters in the header logical record is represented in the coded character set of the ISO/IEC 646: 1991 standard. For maximum portability between implementations, names should be selected from characters represented by the portable filename character set as octets with the most significant bit zero. If an implementation supports the use of characters outside of slash and the portable filename character set in names for files, users, and groups, one or more implementation-defined encodings of these characters are provided for interchange purposes.

pax never creates filenames on the local system that cannot be accessed using the procedures described in IEEE Std 1003.1-200x. If a filename is found on the medium that would create an invalid filename, it is implementation-defined whether the data from the file is stored on the file hierarchy and under what name it is stored. pax can choose to ignore these files as long as it produces an error indicating that the file is being ignored. Each field within the header logical record is contiguous; that is, there is no padding used.

Each field within the header logical record is contiguous. There is no padding used. Each character on the archive medium is stored contiguously.

The fields *magic*, *uname* and *gname* are character strings, each of which is terminated by a NULL character. The fields *name*, *linkname*, and *prefix* are NULL-terminated character strings except when all characters in the array contain non-NULL characters including the last character. The *version* field is two octets containing the characters 00 (zero-zero) The *typeflag* contains a single character. All other fields are leading zero-filled octal numbers using digits from the ISO/IEC 646:1991 standard IRV. Each numeric field is terminated by one or more SPACE or NULL characters.

Each character on the archive medium is stored contiguously. The fields *magic*, *uname*, and *gname* are character strings each terminated by a NULL character.

*name*, *linkname*, and *prefix* are NULL-terminated character strings except when all characters in the array contain non-NULL characters including the last character. The *version* field is two octets containing the characters 00 (zero-zero). The *typeflag* contains a single character. All other fields are leading zero-filled octal numbers using digits from the ISO/IEC 646: 1991 standard IRV. Each numeric field is terminated by one or more spaces or NULL characters.

The *name* and the *prefix* fields produce the pathname of the file. A new pathname is formed, if *prefix* is not an empty string (its first character is not NULL), by concatenating *prefix* (up to the first NULL character), a slash character, and *name*; otherwise, *name* is used alone. In either case, *name* is terminated at the first NULL character. If *prefix* begins with a NULL character, it is ignored. In this manner, pathnames of at most 256 characters can be supported. If a pathname does not fit in the space provided, *pax* notifies the user of the error, and does not store any part of the file-header or data-on the medium.

The *linkname* field does not use the *prefix* to produce a pathname. As such, a *linkname* is limited to 100 characters. If the name does not fit in the space provided, *pax* notifies the user of the error, and does not attempt to store the link on the medium. The *mode* field provides 12 bits encoded in the ISO/IEC 646: 1991 standard octal digit representation. The encoded bits represent the following values in the *ustar mode* field:

| Bit Value | IEE Std 1003.1–2001 Bit | Description                                    |
|-----------|-------------------------|------------------------------------------------|
| 04000     | S_ISUID                 | Set UID on execution                           |
| 02000     | S_ISGID                 | Set GID on execution                           |
| 01000     | <i>reserved</i>         | Reserved for future standardization            |
| 00400     | S_IRUSR                 | Read permission for file owner class           |
| 00200     | S_IWUSR                 | Write permission for file owner class          |
| 00100     | S_IXUSR                 | Execute/search permission for file owner class |
| 00040     | S_IRGRP                 | Read permission for file group class           |
| 00020     | S_IWGRP                 | Write permission for file group class          |
| 00010     | S_IXGRP                 | Execute/search permission for file group class |
| 00004     | S_IROTH                 | Read permission for file other class           |
| 00002     | S_IWOTH                 | Write permission for file other class          |
| 00001     | S_IXOTH                 | Execute/search permission for file other class |

When appropriate privilege is required to set one of these mode bits, and the user restoring the files from the archive does not have the appropriate privilege, the mode bits for which the user

does not have appropriate privilege are ignored. Some of the mode bits in the archive format are not mentioned elsewhere in volume IEEE Std 1003.1-200x. If the implementation does not support those bits, they can be ignored.

The *uid* and *gid* fields are the user and group ID of the owner and group of the file, respectively.

The *size* field is the size of the file in octets. If the *typeflag* field is set to specify a file to be of type 1 (a link) or 2 (a symbolic link), the *size* field is specified as zero. If the *typeflag* field is set to specify a file of type 5 (directory), the *size* field is interpreted as described under the definition of that record type. No data logical records are stored for types 1, 2, or 5. If the *typeflag* field is set to 3 (character special file), 4 (block special file), or 6 (FIFO), the meaning of the *size* field is unspecified by volume IEEE Std 1003.1-200x, and no data logical records is stored on the medium. Additionally, for type 6, the *size* field is ignored when reading. If the *typeflag* field is set to any other value, the number of logical records written following the header is  $(size+511)/512$ , ignoring any fraction in the result of the division.

The *mtime* field is the modification time of the file at the time it was archived. It is the ISO/IEC 646: 1991 standard representation of the octal value of the modification time obtained from the `stat()` function.

The *chksum* field is the ISO/IEC 646: 1991 standard IRV representation of the octal value of the simple sum of all octets in the header logical record. Each octet in the header is treated as an unsigned value. These values are added to an unsigned integer, initialized to zero, the precision of which is not less than 17 bits. When calculating the checksum, the *chksum* field is treated as if it were all spaces.

The *typeflag* field specifies the type of file archived. If a particular implementation does not recognize the type, or the user does not have appropriate privilege to create that type, the file is extracted as if it were a regular file if the file type is defined to have a meaning for the *size* field that could cause data logical records to be written on the medium. If conversion to a regular file occurs, pax produces an error indicating that the conversion took place. All of the *typeflag* fields are coded in the ISO/IEC 646: 1991 standard IRV:

- |   |                                                                                                                                                                                                                                                                                                                                                       |
|---|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0 | Represents a regular file. For backward compatibility, a <i>typeflag</i> value of binary zero ('0') should be recognized as meaning a regular file when extracting files from the archive. Archives written with this version of the archive file format create regular files with a <i>typeflag</i> value of the ISO/IEC 646: 1991 standard IRV '0'. |
| 1 | Represents a file linked to another file, of any type, previously archived. Such files are identified by each file having the same device and file serial number. The linked-to name is specified in the <i>linkname</i> field with a NULL-character terminator if it is less than 100 octets in length.                                              |
| 2 | Represents a symbolic link. The contents of the symbolic link are stored in the <i>linkname</i> field.                                                                                                                                                                                                                                                |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 3, 4          | Represents character special files and block special files respectively. In this case the <i>devmajor</i> and <i>devminor</i> fields contain information defining the device, the format of which is unspecified by volume IEEE Std 1003.1-200x. Implementations can map the device specifications to their own local specification or can ignore the entry.                                                                                                                                                                           |
| 5             | Specifies a directory or subdirectory. On systems where disk allocation is performed on a directory basis, the <i>size</i> field contain the maximum number of octets (which can be rounded to the nearest disk block allocation unit) that the directory can hold. A <i>size</i> field of zero indicates no such limiting. Systems that do not support limiting in this manner should ignore the <i>size</i> field.                                                                                                                   |
| 6             | Specifies a FIFO special file. The archiving of a FIFO file archives the existence of this file and not its contents.                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 7             | Reserved to represent a file to which an implementation has associated some high- performance attribute. Implementations without such extensions should treat this file as a regular file (type 0).                                                                                                                                                                                                                                                                                                                                    |
| A - Z         | The letters A through Z inclusive are reserved for custom implementations. All other values are reserved for future versions of IEEE Std 1003.1-200x.                                                                                                                                                                                                                                                                                                                                                                                  |
| SUN.devmajor  | <p>A Solaris extension to pax extended header keywords. Specifies the major device number of the file.</p> <p>When used in write or copy mode and the xustar or pax format (see -x format) was specified, pax includes a SUN.devmajor extended header record for each file whose major device number is too large to fit in 8 octets.</p>                                                                                                                                                                                              |
| SUN.devminor  | <p>A Solaris extension to pax extended header keywords. Specifies the minor device number of the file.</p> <p>When used in write or copy mode and the xustar or pax format (see -x format) is specified, pax includes a SUN.devminor extended header record for each file whose minor device number is too large to fit in 8 octets.</p>                                                                                                                                                                                               |
| SUN.holesdata | <p>A Solaris extension to pax extended header keywords. Specifies the data and hole pairs for a sparse file.</p> <p>In write or copy modes and when the xustar or pax format (see -x format) is specified, pax includes a SUN.holesdata extended header record if the underlying file system supports the detection of files with holes (see <a href="#">fpathconf(2)</a>) and reports that there is at least one hole in the file being archived. <i>value</i> consists of two or more consecutive entries of the following form:</p> |

`SPACEdata_offsetSPACEhole_offset`

where the data and hole offsets are the long values returned by passing `SEEK_DATA` and `SEEK_HOLE` to `lseek(2)`, respectively. For example, the following entry is an example of the `SUN.holesdata` entry in the extended header for a file with data offsets at bytes 0, 24576, and 49152, and hole offsets at bytes 8192, 32768, and 49159: `49 SUN.holesdata= 0 8192 24576 32768 49152 49159:`

`49 SUN.holesdata= 0 8192 24576 32768 49152 49159`

When extracting a file from an archive in read or copy modes, if a `SUN.holesdata = pair` is found in the extended header for the file, then the file is restored with the holes identified using this data. For example, for the `SUN.holesdata` provided in the example above, bytes from 0 to 8192 are restored as data, a hole is created up to the next data position (24576), bytes 24576 to 32768 is restored as data, and so forth.

- X A Solaris custom `typeflag` implementation which specifies an `xstar` format (see `-x` format) extended header. The `typeflag 'x'` extended header is treated as a `ustar typeflag 'x'` extended header.
- E A Solaris custom `typeflag` implementation which specifies an extended attributes header. See `fsattr(5)`.

Attempts to archive a socket using `ustar` interchange format produce a diagnostic message. Handling of other file types is implementation-defined.

The *magic* field is the specification that this archive was output in this archive format. If this field contains `ustar` (the five characters from the ISO/IEC 646: 1991 standard IRV shown followed by `NULL`), the *uname* and *gname* fields contain the ISO/IEC 646: 1991 standard IRV representation of the owner and group of the file, respectively (truncated to fit, if necessary). When the file is restored by a privileged, protection-preserving version of the utility, the user and group databases are scanned for these names. If found, the user and group IDs contained within these files are used rather than the values contained within the *uid* and *gid* fields.

cpio Interchange  
Format

The octet-oriented `cpio` archive format are a series of entries, each comprising a header that describes the file, name of the file, and contents of the file.

An archive can be recorded as a series of fixed-size blocks of octets. This blocking is be used only to make physical I/O more efficient. The last group of blocks are always at the full size.

For the octet-oriented `cpio` archive format, the individual entry information are in the order indicated and described by the following table: Octet-Oriented `cpio` Archive Entry. See the `cpio.h` header for additional details.



| Header Field Name   | Length (in Octets) | Interpreted as  |
|---------------------|--------------------|-----------------|
| <i>c_magic</i>      | 6                  | Octal number    |
| <i>c_dev</i>        | 6                  | Octal number    |
| <i>c_ino</i>        | 6                  | Octal number    |
| <i>c_mode</i>       | 6                  | Octal number    |
| <i>c_uid</i>        | 6                  | Octal number    |
| <i>c_gid</i>        | 6                  | Octal number    |
| <i>c_nlink</i>      | 6                  | Octal number    |
| <i>c_rdev</i>       | 6                  | Octal number    |
| <i>c_mtime</i>      | 11                 | Octal number    |
| <i>c_namesize</i>   | 6                  | Octal number    |
| <i>c_filesize</i>   | 11                 | Octal number    |
| Filename Field Name | Length             | Interpreted as  |
| <i>c_name</i>       | <i>c_namesize</i>  | Pathname string |
| Filename Field Name | Length             | Interpreted as  |
| <i>c_filedata</i>   | <i>c_filesize</i>  | Data            |

cpio Header

For each file in the archive, a header as defined previously written. The information in the header fields is written as streams of the ISO/IEC 646: 1991 standard characters interpreted as octal numbers. The octal numbers are extended to the necessary length by appending the ISO/IEC 646: 1991 standard IRV zeros at the most-significant-digit end of the number. The result is written to the most-significant digit of the stream of octets first. The fields are interpreted as follows:

|                    |                                                                                                                                                                                                                                                  |
|--------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>c_magic</i>     | Identifies the archive as being a transportable archive by containing the identifying value "070707".                                                                                                                                            |
| <i>c_dev,c_ino</i> | Contains values that uniquely identify the file within the archive (that is, no files contain the same pair of <i>c_dev</i> and <i>c_ino</i> values unless they are links to the same file). The values are determined in an unspecified manner. |
| <i>c_mode</i>      | Contains the file type and access permissions as defined in the following table.<br><br>Directories, FIFOs, symbolic links, and regular files are supported on a system conforming to volume IEEE Std 1003.1-200x; additional values defined     |

previously are reserved for compatibility with existing systems. Additional file types can be supported. Such files should not be written to archives intended to be transported to other systems.

| File Permissions Name | Value  | Indicates |
|-----------------------|--------|-----------|
| C_IRUSR               | 000400 | by owner  |
| C_IWUSR               | 000200 | by owner  |
| C_IXUSR               | 000100 | by owner  |
| C_IRGRP               | 000040 | by group  |
| CW_IWFGP              | 000020 | by group  |
| CW_IXGRP              | 000010 | by group  |
| CW_IROTH              | 000004 | by others |
| CW_IWOTH              | 000002 | by others |
| CW_IXOTH              | 000001 | by others |
| CW_ISUID              | 004000 | Set uid   |
| W_ISGID               | 002000 | Set gid   |
| W_ISVTX               | 001000 | Reserved  |

  

| File Type Name | Value   | Indicates              |
|----------------|---------|------------------------|
| C_ISDIR        | 040000  | Directory              |
| C_ISFIFO       | 010000  | FIFO                   |
| C_ISREG        | 0100000 | Regular file           |
| C_ISLNK        | 0120000 | Symbolic link          |
| C_ISBLK        | 060000  | Block special file     |
| C_ISCHR        | 020000  | Character special file |
| C_ISSOCK       | 0140000 | Socket                 |
| C_ISCTG        | 0110000 | Reserved               |

- c\_uid* Contains the user ID of the owner.
- c\_gid* Contains the group ID of the group

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>c_nlink</i>    | Contains a number greater than or equal to the number of links in the archive referencing the file. If the <i>-a</i> option is used to append to a <i>cpio</i> archive, <i>pax</i> does need not to account for the files in the existing part of the archive when calculating the <i>c_nlink</i> values for the appended part of the archive. It does also need not alter the <i>c_nlink</i> values in the existing part of the archive if additional files with the same <i>c_dev</i> and <i>c_ino</i> values are appended to the archive. |
| <i>c_rdev</i>     | Contains implementation-defined information for character or block special files.                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>c_mtime</i>    | Contains the latest time of modification of the file at the time the archive was created.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <i>c_namesize</i> | Contains the length of the pathname, including the terminating NULL character.                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <i>c_filesize</i> | Contains the length of the file in octets. This is the length of the data section following the header structure.                                                                                                                                                                                                                                                                                                                                                                                                                            |

*cpio* Filename      The *c\_name* field contains the pathname of the file. The length of this field in octets is the value of *c\_namesize*. If a filename is found on the medium that would create an invalid pathname, it is implementation-defined whether the data from the file is stored on the file hierarchy and under what name it is stored. All characters are represented in the ISO/IEC 646: 1991 standard IRV. For maximum portability between implementations, names should be selected from characters represented by the portable filename character set as octets with the most significant bit zero. If an implementation supports the use of characters outside the portable filename character set in names for files, users, and groups, one or more implementation-defined encodings of these characters are provided for interchange purposes. *pax* does not create filenames on the local system that cannot be accessed by way of the procedures described in volume IEEE Std 1003.1-200x. If a filename is found on the medium that would create an invalid filename, it is implementation-defined whether the data from the file is stored on the local file system and under what name it is stored. *pax* can choose to ignore these files as long as it produces an error indicating that the file is being ignored.

*cpio* File Data      Following *c\_name*, there is *c\_filesize* octets of data. Interpretation of such data occurs in a manner dependent on the file. If *c\_filesize* is zero, no data is contained in *c\_filedata*. When restoring from an archive:

- If the user does not have the appropriate privilege to create a file of the specified type, *pax* ignores the entry and writes an error message to standard error.
- Only regular files have data to be restored. Presuming a regular file meets any selection criteria that might be imposed on the format-reading utility by the user, such data is restored.
- If a user does not have appropriate privilege to set a particular *mode* flag, the flag is ignored. Some of the *mode* flags in the archive format are not mentioned in volume IEEE Std 1003.1-200x. If the implementation does not support those flags, they can be ignored.

cpio Special Entries      FIFO special files, directories, and the trailer are recorded with `c_filesize` equal to zero. For other special files, `c_filesize` is unspecified in volume IEEE Std 1003.1-200x. The header for the next file entry in the archive are written directly after the last octet of the file entry preceding it. A header denoting the filename trailer indicates the end of the archive; the contents of octets in the last block of the archive following such a header are undefined.

属性      See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE                    |
|---------------------|------------------------------------|
| Availability        | system/core-os                     |
| Interface Stability | Committed                          |
| Standard            | See <a href="#">standards(5)</a> . |

関連項目      [chmod\(1\)](#), [cpio\(1\)](#), [ed\(1\)](#), [printf\(1\)](#), [tar\(1\)](#), [mkdir\(2\)](#), [lseek\(2\)](#), [stat\(2\)](#), [write\(2\)](#), [archives.h\(3HEAD\)](#), [attributes\(5\)](#), [environ\(5\)](#), [fnmatch\(5\)](#), [formats\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [standards\(5\)](#)

IEEE Std 1003.1-200x, ISO/IEC 646: 1991, ISO POSIX-2:1993 Standard

|         |                                                                                                                                       |                                        |
|---------|---------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
| 名前      | perl – 抽出および出力を行う言語 (Practical Extraction and Report Language)                                                                        |                                        |
| 形式      | perl [-sTuU] [-hv] [-V [: <i>configvar</i> ]] [-cw] [-d [: <i>debugger</i> ]] [-D [ <i>number/list</i> ]] [-pna] [-F <i>pattern</i> ] |                                        |
| 機能説明    | 調べやすくするために、Perl のマニュアルは以下のセクションに分割されています。                                                                                             |                                        |
|         | 概要                                                                                                                                    |                                        |
|         | perl                                                                                                                                  | Perl overview (this section)           |
|         | perlintro                                                                                                                             | Perl introduction for beginners        |
|         | perltoc                                                                                                                               | Perl documentation table of contents   |
|         | チュートリアル                                                                                                                               |                                        |
| チュートリアル | perlreftut                                                                                                                            | Perl references short introduction     |
|         | perldsc                                                                                                                               | Perl data structures intro             |
|         | perllo1                                                                                                                               | Perl data structures: arrays of arrays |
|         | perlrequick                                                                                                                           | Perl regular expressions quick start   |
|         | perlretut                                                                                                                             | Perl regular expressions tutorial      |
|         | perlboot                                                                                                                              | Perl 00 tutorial for beginners         |
|         | perltoot                                                                                                                              | Perl 00 tutorial, part 1               |
|         | perltooc                                                                                                                              | Perl 00 tutorial, part 2               |
|         | perlbot                                                                                                                               | Perl 00 tricks and examples            |
|         | perlstyle                                                                                                                             | Perl style guide                       |
|         | perlcheat                                                                                                                             | Perl cheat sheet                       |
|         | perltrap                                                                                                                              | Perl traps for the unwary              |
|         | perldebtut                                                                                                                            | Perl debugging tutorial                |
|         | perlfaq                                                                                                                               | Perl frequently asked questions        |
|         | perlfaq1                                                                                                                              | General Questions About Perl           |
|         | perlfaq2                                                                                                                              | Obtaining and Learning about Perl      |
|         | perlfaq3                                                                                                                              | Programming Tools                      |
|         | perlfaq4                                                                                                                              | Data Manipulation                      |
|         | perlfaq5                                                                                                                              | Files and Formats                      |
|         | perlfaq6                                                                                                                              | Regexes                                |
|         | perlfaq7                                                                                                                              | Perl Language Issues                   |
|         | perlfaq8                                                                                                                              | System Interaction                     |
|         | perlfaq9                                                                                                                              | Networking                             |
|         | リファレンスマニュアル                                                                                                                           |                                        |
|         | perlsyn                                                                                                                               | Perl syntax                            |
|         | perldata                                                                                                                              | Perl data structures                   |
|         | perlop                                                                                                                                | Perl operators and precedence          |
|         | perlsub                                                                                                                               | Perl subroutines                       |
|         | perlfunc                                                                                                                              | Perl built-in functions                |
|         | perlopenut                                                                                                                            | Perl open() tutorial                   |
|         | perlpacktut                                                                                                                           | Perl pack() and unpack() tutorial      |

|                |                                                       |
|----------------|-------------------------------------------------------|
| perlpod        | Perl plain old documentation                          |
| perlpodspec    | Perl plain old documentation format specification     |
| perlrun        | Perl execution and options                            |
| perldiag       | Perl diagnostic messages                              |
| perllexwarn    | Perl warnings and their control                       |
| perldebug      | Perl debugging                                        |
| perlvar        | Perl predefined variables                             |
| perlre         | Perl regular expressions, the rest of the story       |
| perlref        | Perl regular expressions quick reference              |
| perlref        | Perl references, the rest of the story                |
| perlform       | Perl formats                                          |
| perlobj        | Perl objects                                          |
| perltye        | Perl objects hidden behind simple variables           |
| perlshmfilter  | Perl DBM filters                                      |
| perlipc        | Perl interprocess communication                       |
| perlfork       | Perl fork() information                               |
| perlnumber     | Perl number semantics                                 |
| perlthrtut     | Perl threads tutorial                                 |
| perlthrtut     | Old Perl threads tutorial                             |
| perlport       | Perl portability guide                                |
| perllocale     | Perl locale support                                   |
| perluniintro   | Perl Unicode introduction                             |
| perlunicode    | Perl Unicode support                                  |
| perlebcdic     | Considerations for running Perl on EBCDIC platforms   |
| perlsec        | Perl security                                         |
| perlmod        | Perl modules: how they work                           |
| perlmodlib     | Perl modules: how to write and use                    |
| perlmodstyle   | Perl modules: how to write modules with style         |
| perlmodinstall | Perl modules: how to install from CPAN                |
| perlnewmod     | Perl modules: preparing a new module for distribution |
| perlutil       | utilities packaged with the Perl distribution         |
| perlcompile    | Perl compiler suite intro                             |
| perlfilter     | Perl source filters                                   |

## 内部関数とC言語インタフェース

|             |                                                        |
|-------------|--------------------------------------------------------|
| perlembed   | Perl ways to embed perl in your C or C++ application   |
| perldebbugs | Perl debugging guts and tips                           |
| perlxsut    | Perl XS tutorial                                       |
| perlxs      | Perl XS application programming interface              |
| perlclib    | Internal replacements for standard C library functions |
| perlguys    | Perl internal functions for those doing extensions     |
| perlcall    | Perl calling conventions from C                        |

|            |                                                 |
|------------|-------------------------------------------------|
| perlapi    | Perl API listing (autogenerated)                |
| perlintern | Perl internal functions (autogenerated)         |
| perljol    | C API for Perl's implementation of IO in Layers |
| perlpio    | Perl internal IO abstraction interface          |
| perlhack   | Perl hackers guide                              |

## その他

|               |                                          |
|---------------|------------------------------------------|
| perlbook      | Perl book information                    |
| perltodo      | Perl things to do                        |
| perldoc       | Look up Perl documentation in Pod format |
| perlhists     | Perl history records                     |
| perldelta     | Perl changes since previous version      |
| perl583delta  | Perl changes in version 5.8.3            |
| perl582delta  | Perl changes in version 5.8.2            |
| perl581delta  | Perl changes in version 5.8.1            |
| perl58delta   | Perl changes in version 5.8.0            |
| perl573delta  | Perl changes in version 5.7.3            |
| perl572delta  | Perl changes in version 5.7.2            |
| perl571delta  | Perl changes in version 5.7.1            |
| perl570delta  | Perl changes in version 5.7.0            |
| perl561delta  | Perl changes in version 5.6.1            |
| perl56delta   | Perl changes in version 5.6              |
| perl5005delta | Perl changes in version 5.005            |
| perl5004delta | Perl changes in version 5.004            |
| perlartistic  | Perl Artistic License                    |
| perlgpl       | GNU General Public License               |

## 言語固有

|         |                                         |
|---------|-----------------------------------------|
| perlcnc | Perl for Simplified Chinese (in EUC-CN) |
| perljp  | Perl for Japanese (in EUC-JP)           |
| perlko  | Perl for Korean (in EUC-KR)             |
| perltw  | Perl for Traditional Chinese (in Big5)  |

## プラットフォーム固有

|             |                        |
|-------------|------------------------|
| perlsolaris | Perl notes for Solaris |
|-------------|------------------------|

## プラットフォーム固有

初めて Perl を学習する場合は、perlintro から始めるべきです。このドキュメントは、初心者のための概要であり、残りの Perl 拡張ドキュメントに進むための背景情報も提供します。調べやすくするため、Perl のマニュアルはいくつかのセクションに分割されています。

前述のマニュアルページは /usr/perl5/man/ ディレクトリにインストールされています。

前述のほかにも、Perl モジュールに関するマニュアルページがあります。これらも `/usr/perl5/man` ディレクトリにあります。マニュアルページの一部は標準で Perl に提供されています。ユーザーが任意でインストールするサードパーティのモジュールに関するマニュアルページも、同じディレクトリに置かれます。

Perl のドキュメントを表示するには、[man\(1\)](#) の `MANPATH` 環境変数に `/usr/perl5/man` を含めます。Perl 関連のマニュアルページに対して [catman\(1M\)](#) を実行することはサポートされていません。その他の Solaris 固有の情報については、後述の「注意事項」のセクションを参照してください。

Perl に関する情報は、`/usr/perl5/bin/perldoc` スクリプトを使用して参照することもできます。

プログラム中に何か問題があるときにその場所を特定できない場合は、`-w` スイッチを使用してください。多くの場合に問題箇所を特定することができます。

Perl は、テキストファイルを読み取り、そこから情報を抽出して、その情報をもとにして報告を出力する言語です。また、システム管理タスクを行うのにも適した言語です。Perl は、美しいこと (小規模、上品、最小限) よりも実用性 (使いやすい、効率、完全) を目指して作成されています。

Perl は (筆者の意見では)、C、`sed`、`awk`、および `sh` の長所を組み合わせで作成されているので、これらの言語に慣れている方には、Perl を使用することはそれほど難しくないでしょう。( `csh`、`Pascal`、および `BASIC-PLUS` の特徴も入っているという意見もあります。) Perl の式の構文はかなりの部分で、C の式の構文に対応しています。多くの UNIX ユーティリティとは異なり、Perl にはデータのサイズに制限がありません。つまり、メモリーが十分にあれば、Perl ではファイル全体を 1 つの文字列の中に入れてしまうこともできます。再帰の深さにも制限がありません。パフォーマンスの低下を防ぐために、ハッシュテーブル (連想配列と呼ばれることもある) のサイズが必要に応じて自動的に大きくなります。Perl は、高度なパターンマッチング技術を使用して、大量のデータを高速に読み取ることができます。Perl はテキストファイルを扱うことを用途として最適化されていますが、バイナリデータを扱ったり、`dbm` ファイルをハッシュテーブルのようにして扱うこともできます。データフローのトレースメカニズムによって、多くの不注意によるセキュリティホールが防止されているので、Perl の `setuid` スクリプトは C プログラムよりも安全です。

通常なら `sed`、`awk`、または `sh` を使用して記述するような処理をさせたい場合に、`sed`、`awk`、`sh` では不可能または処理速度をもっと上げたい、けれども C で記述するほどではないというとき、Perl を使用するのが適しています。また、`sed` および `awk` のスクリプトを Perl スクリプトに変換するプログラムもあります。

さらに情報は続きます。

1993 年に誕生して以来 ([perlhist](#) を参照)、Perl はバージョン 5 で、ほぼ全体が改訂され、次のような特長が追加されています。



- 多数のモジュールを使用した、モジュール性と再利用性。perlmod、perlmodlib、およびperlmodinstallを参照してください。
- 組み込みと拡張性。perlembed、perlxsut、perlxs、perlcall、perlguts、およびxsubppを参照してください。
- 独自のマジック変数の活用(複数のDBM同時実装を含む)。perl tie および AnyDBM\_File を参照してください。
- サブルーチンのオーバーライド、自動ロード、プロトタイプ化が可能。perlsubを参照してください。
- 任意の多重データ構造体および匿名の関数。perlreftut、perlref、perldsc、およびperlloolを参照してください。
- オブジェクト指向プログラミング。perlobj、perlboot、perltoot、perltooc、およびperlbotを参照してください。
- 軽量プロセス(スレッド)のサポート。perlthrtut および Thread を参照してください。
- Unicode、国際化、およびローカリゼーションのサポート。perluniintro、perllocale、およびLocale::Maketextを参照してください。
- 字句のスコープ。perlsubを参照してください。
- 正規表現の拡張。perlre および perl op を参照してください。
- 統合エディタの採用による、デバッガの機能拡張および対話型のPerl環境。perldebtut、perldebug および perldeb guts を参照してください。
- POSIX 1003.1 準拠。POSIX を参照してください。

以上です。

## 環境変数

Solarisに含まれているPerlは、/usr/localではなく、/usr/perl5にインストールされます。これは、ユーザーがデフォルトの/usr/localにインストールする別バージョンのPerlと衝突しないようにするためです

追加でインストールするモジュールは、/usr/perl5/site\_perl/5.8.4ディレクトリに置かれます。/usr/perl5/vendor\_perlディレクトリには、SMIが提供するモジュールが置かれます。

perldoc および perlbug などのPerlスクリプトは、/usr/perl5/binディレクトリにあります。これらのスクリプトを使用する場合は、PATH環境変数に/usr/perl5/binを追加してください。

perlrun も参照してください。

著者 Larry Wall 他

これから Perl をアプリケーションに使用しようとしているほかの人たちのために、自分の Perl に関する成功事例や経験談を提供していただける方、あるいは、Larry を初めとする Perl 開発者たちに簡単なお礼を述べたい方は、perl-thanks@perl.org まで電子メールを送信してください。

ファイル "@INC" Perl ライブラリの場所

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                                                                                                 |
|-------------|---------------------------------------------------------------------------------------------------------------------|
| 使用条件        | runtime/perl-584, runtime/perl-584/extra, runtime/perl-584/manual, SUNWpl5u, SUNWpl5v, SUNWpl5p, SUNWpl5m<br>下記を参照。 |
| インタフェースの安定性 | 下記を参照。                                                                                                              |

Perl は、ほぼすべての Unix 系プラットフォームを含めて、ほとんどのオペレーティングシステムで使用できます。リストについては、perlport の「Supported Platforms」を参照してください。

スクリプトインタフェース、XSUB インタフェース、およびディレクトリレイアウトは「確実」です。パイナリインタフェースは「不確実」です。

関連項目 a2p awk から perl への変換プログラム  
s2p sed から perl への変換プログラム  
<http://www.perl.com> Perl のホームページ  
<http://www.perl.com/CPAN> 総合的な Perl アーカイブ  
<http://www.perl.org> Perl Mongers (Perl ユーザーグループ)

診断 use warnings プラグマ (および -w スイッチ) を使用すると、診断メッセージが表示されます。

Perl の診断メッセージについての説明は、perldiag を参照してください。use diagnostics プラグマを使用すると、警告メッセージおよびエラーメッセージがより詳細に出力されます。

コンパイル時のエラーメッセージには、エラー箇所の行番号、確認が必要な次のトークンまたはトークンタイプが示されます。複数の -e スイッチによって Perl に渡されるスクリプトでは、各 -e スイッチが 1 行として数えられます。

setuid スクリプトには、「Insecure dependency」などのエラーメッセージとして示されるような制限事項があります。詳細は perlsec を参照してください。

常に -w スイッチを使用することをお勧めします。

## 注意事項

インタープリタ自体は 32 ビットアプリケーション (LP32) ですが、Perl 5.8.4 は大規模ファイルの処理が可能で 64 ビットの整数を使用するように作られています。詳しい構成情報を参照するには、perl V および perlbug dv を使用してください。

gcc を使用して CPAN からアドオンモジュールを構築およびインストールするには、/usr/perl5/5.8.4/bin/perlgcc スクリプトを使用します。詳細については、perlgcc(1) を参照してください。

独自のバージョンの Perl を作成してインストールする場合でも、/usr/perl5 にある 5.8.4 バージョンの Perl は削除しないでください。このバージョンの Perl は、複数のシステムユーティリティで使用されています。Perl のパッケージ名は、以下のとおりです。

|                 |                             |
|-----------------|-----------------------------|
| SUNWperl584core | Perl 5.8.4 (Core files)     |
| SUNWperl584usr  | Perl 5.8.4 (Non-core files) |
| SUNWperl584man  | Perl 5.8.4 (Manual pages)   |

Solaris 10 には、Solaris 9 に付属していたバージョン 5.6.1 の Perl も付属しています。システムのアップグレードは行いますが、Perl 5.6.1 をデフォルトの Perl バージョンとして引き続き使用する場合、その方法についての詳細は、perlsolaris のマニュアルページを参照してください。Perl 5.6.1 は将来のリリースで削除されるため、可能な限り、Perl 5.8.4 にアップグレードしてください。

Perl のモットーは、「やり方は何通りもある」です。ほかの方法を考えることは、読者にとってよい練習となるでしょう。

プログラマにとっての主な 3 つの美德は、「怠慢」、「短気」、そして「傲慢」です。理由については「ラクダ本」を参照してください。

## 使用上の留意点

-w スイッチを使用することは、必須事項ではありません。

Perl は、型キャストを行う atof() や、浮動小数点を出力する sprintf() など、さまざまな処理のマシン上での定義に影響されます。

標準入出力が、特定のストリーム上の読み取りと書き込みの間にシークまたは EOF を必要とする場合、Perl がそれを実行します。ただしこれは sysread() および syswrite() には適用されません。

組み込みデータ型にサイズ制限 (メモリーサイズを除く) はありませんが、若干の制限事項があります。変数名は 251 文字未満にする必要があります。診断メッセージ

に示される行番号は内部で short 型整数として格納されるので、表示できる行番号は 65535 行目までです。これ以上の行番号は通常 wraparound による影響を受けません。

発見したバグについて、電子メールで <perlbug@perl.org> 宛に報告することができます。このとき、使用している環境の構成情報 (Perl のソースツリーにある `myconfig` コマンドまたは `perl V` の実行結果) も必ずお知らせください。Perl のコンパイルに成功した場合、`utils/` サブディレクトリにある `perlbug` スクリプトを使用すると、簡単にバグレポートをメールで送信できます。

実は、Perl は「Pathologically Eclectic Rubbish Lister」の略なのですが、私がこう述べたことはだれにも言わないようにしてください。

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pfexec, pfbash, pfcsh, pfksh, pfsh, pftcsh, pfzsh – プロファイル内のコマンドの実行                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 形式      | <pre> /usr/bin/pfexec <i>command</i>  /usr/bin/pfexec -P <i>privspec</i> <i>command</i> [ <i>arg</i> ]...  /usr/bin/pfsh [ <i>options</i> ] [ <i>argument</i> ]...  /usr/bin/pfcsh [ <i>options</i> ] [ <i>argument</i> ]...  /usr/bin/pfksh [ <i>options</i> ] [ <i>argument</i> ]...</pre>                                                                                                                                                                                                                                                                                                                                                           |
| 機能説明    | <p>pfexec プログラムは PRIV_PFEXEC プロセスフラグを設定し、現プロセスをプロファイルシェルとしてマークします。次に、指定されたコマンドを実行します。カーネルは <a href="#">exec_attr(4)</a> データベースに照会し、適切な属性を使用して実行します。</p> <p>プロファイルは、<a href="#">user_attr(4)</a> データベース内の該当するユーザーのエントリで指定された順序で検索されます。同じコマンドが複数のプロファイルに存在する場合、プロファイルシェルは最初に一致したエントリを使用します。</p> <p>2 番目の形式 <code>pfexec -P <i>privspec</i></code> を使用すると、ユーザーは <a href="#">prof_attr(4)</a> 内の該当するユーザーのプロファイルに与えられた追加の特権を取得できます。コマンド行での特権の指定は、<a href="#">priv_str_to_set(3C)</a> を使用して解析されます。その結果得られた特権と、該当するユーザーのすべてのプロファイルについて <a href="#">prof_attr(4)</a> 内の <code>privs</code> キーワードで指定されている特権の和集合との間で、積集合をとった上で、その結果がコマンドの実行前に継承可能セットに追加されます。</p> |
| 使用法     | <p>pfexec は、特定のユーザーまたはグループ ID など、定義済みのプロセス属性でコマンドを実行するために使用します。</p> <p>プロファイルシェルの使用法についての詳細は、<a href="#">sh(1)</a>、<a href="#">csh(1)</a>、および <a href="#">ksh(1)</a> のマニュアルページを参照してください。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 使用例     | <p>例1 追加ユーザー特権の取得</p> <pre>example% pfexec -P all chown user file</pre> <p>このコマンドは、現在のユーザーに割り当てられているすべての特権 (必ずしもすべての特権であるとは限らない) で <code>chown user file</code> を実行します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <ul style="list-style-type: none"> <li>0 正常終了。</li> <li>1 エラーが発生した。</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [bash\(1\)](#), [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [profiles\(1\)](#), [sh\(1\)](#), [tcsh\(1\)](#), [zsh\(1\)](#), [exec\\_attr\(4\)](#), [prof\\_attr\(4\)](#), [user\\_attr\(4\)](#), [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pg - CRT 用のファイル閲覧フィルタ                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 形式    | pg [-number] [-p string] [-cefnrs] [+ linenumber] [+/- pattern /] [filename]...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 機能説明  | <p>pg コマンドは、<i>filename</i> を CRT 上で 1 度に 1 画面分調べることができるフィルタです。ユーザーが復帰改行を入力すると、他のページが表示されます。他の機能については次に示します。</p> <p>このコマンドは、ユーザーがすでに通過したものを元に戻って見直すことができるという点で、今までのページングコマンドと異なります。この方法については、以下で説明します。</p> <p>端末の属性を判定するために、pg は、<a href="#">terminfo(4)</a> データベースを走査し、環境変数 TERM によって指定された端末タイプを求めます。TERM が定義されていない場合、端末タイプは dumb と見なされます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| オプション | <p><b>-number</b> pg がデフォルトの代わりに使用するウィンドウのサイズ(行数)を指定するための整数(24 行を表示する端末では、デフォルトのウィンドウサイズは 23)。</p> <p><b>-p string</b> pg は、<i>string</i> をプロンプトとして使用します。プロンプトの文字列に %d がある場合は、プロンプトの中の %d は、プロンプトが出されたときの現在のページ番号と置き換えられます。デフォルトのプロンプト文字列は ":" です。</p> <p><b>-c</b> 各ページを表示する前にカーソルをホームポジションに戻し、画面をクリアします。<a href="#">terminfo(4)</a> データベースに、使用する端末タイプ用の clear_screen が定義されていない場合、このオプションは無視されます。</p> <p><b>-e</b> pg は、各ファイルの終りで停止しません。</p> <p><b>-f</b> 通常、pg は、スクリーン幅よりも長い行を分割します。しかし、表示されているテキストの文字シーケンスのなかには、好ましくない結果を生じるものもあります(たとえば、下線のためのエスケープシーケンス)。-f オプションは、pg が行を分割しないようにします。</p> <p><b>-n</b> 通常、コマンドは &lt;newline&gt; 文字で終了しなければなりません。このオプションを指定すると、コマンド文字を入力するとすぐにコマンドは自動的に終了します。</p> <p><b>-r</b> 制限モード。シェルエスケープは却下されます。pg は、エラーメッセージを出力しますが、終了しません。</p> <p><b>-s</b> pg は、標準出力モード(通常は、反転映像)ですべてのメッセージおよびプロンプトを出力します。</p> <p><b>+linenumber</b> <i>linenumber</i> から表示します。</p> |

`+/pattern/` 正規表現パターンを含む最初の行から開始します。

オペランド 次のオペランドがサポートされています。

`filename` 表示するテキストファイルのパス名。`filename` を省略するか、あるいは `-` を指定すると、標準入力と見なされます。

## 使用法

コマンド `pg` 入力を待っているときに打ち込むことができる応答は、次の3つのカテゴリに分割できます。閲覧の続行、検索、および閲覧環境の変更です。

閲覧を続けるためのコマンドには、前に `address` が来ます。これは、次のテキストが表示される場所を示す任意の符号付き数です。この `address` は、コマンドによって、ページか行のいずれかに解釈されます。符号付き `address` は、現在のページまたは行からの相対的な場所を指定し、符号なし `address` は、ファイルの先頭からの絶対アドレスを指定します。いずれのコマンドにも、アドレスが指定されていない場合に使用されるデフォルトのアドレスがあります。

閲覧のためのコマンドとそのデフォルトは、次のとおりです。

(+1)<newline> または <blank> 1 ページ表示します。アドレスは、ページ単位で指定されます。

(+1)l 相対アドレスとともに使うと、`pg` は指定された行数だけ画面を順方向または逆方向にスクロールをシミュレートします。絶対アドレスとともに使うと、このコマンドは、指定された行から始まる1画面を出力します。

(+1)d または ^D 順方向または逆方向に半画面のスクロールをシミュレートします。

if テキストの *i* 個のスクリーンをスキップします。

iz <newline> と同じですが、*i* がある場合、*i* がデフォルトの新しい1画面の行数になります。

次の閲覧コマンドに `address` は必要ありません。

. または ^L ピリオドを単独で打ち込むと、テキストの現在のページが再表示されます。

\$ ファイルの中の最後のフルウィンドウを表示します。入力がパイプである場合は、注意してください。

以下のコマンドは、テキスト中のテキストパターンの検索に利用できます。正規表現については、[regex\(5\)](#) のマニュアルページを参照してください。`-n` オプションを指定する場合でも、必ず <newline> で終了しなければなりません。



*i/pattern/* *i* 回目の *pattern* の出現を順方向に検索します (デフォルトは *i*=1)。検索は、現在のページの直後から、現在のファイルが終了するまで続きます。循環はしません。

*i^pattern^*

*i?pattern?* *i* 回目の *pattern* の出現を逆方向に検索します (デフォルトは *i*=1)。検索は、現在のページの直前から、現在のファイルの初めまで続きます。循環はしません。^ 表記は、? の取り扱いが適切でない Adds 100 端末に有効です。

通常 **pg** は検索の後で、発見された行を画面の最上段に表示します。検索コマンドに **m** または **b** を追加することによって、それ以降ウィンドウの中央または最下段で見つかった行を残しておくことができます。接尾辞 **t** を使用して、元の状態に戻すことができます。

**pg** のユーザーは、以下のコマンドを使用して、閲覧の環境を変更することができます。

**in** コマンド行の中の *i* 個後のファイルの閲覧を始めます。*i* は、符号なしの数で、デフォルトは 1 です。

**ip** コマンド行の中の *i* 個前のファイルの閲覧を始めます。*i* は、符号なしの数で、デフォルトは 1 です。

**iw** テキストの他のウィンドウを表示します。*i* が存在する場合は、ウィンドウサイズを *i* に設定します。

**s filename** 指定されたファイルに入力を保存します。現在閲覧されているファイルのみが保存されます。**s** と *filename* の間のスペースは任意です。**-n** オプションを指定する場合でも、このコマンドは、必ず **<newline>** で終了しなければなりません。

**h** 利用可能なコマンドの一覧を簡略化して表示します。

**q** または **Q** **pg** を終了します。

**!command** *command* を、SHELL 環境変数に指定されているシェルに引き渡しします。SHELL 環境変数に値が指定されていない場合は、デフォルトのシェルを使用します。**-n** オプションを指定する場合でも、このコマンドは、必ず **<newline>** で終了しなければなりません。

出力が端末に送信されるときはいつでも、ユーザーは、中止キー (通常は CTRL-C) または割り込み (ブレーク) キーを打つことができます。これによって、**pg** は出力の送信を停止し、プロンプトを表示します。またユーザーは、前述のコマンドの 1 つを通常の方法で入力することができます。残念ながら、この場合、出力がなくなる場合があります。なぜなら、終了シグナルが発生すると、端末の出力待ちキューの文字がフラッシュされるからです。

標準出力が端末ではない場合は、pg は [cat\(1\)](#) と同じような働きをします。ただし、ファイルが2つ以上ある場合、各ファイルの前にヘッダーが出力されます。

大規模ファイルの動作      ファイルが2G バイト (2<sup>31</sup> バイト) 以上ある場合の pg の動作については、[largefile\(5\)](#) を参照してください。

使用例      例1 pg を使用してシステム情報を読み取る  
次のコマンド行は、pg を使用してシステム情報を読み取ります。  
  
example% news | pg -p "(Page %d):"

環境変数      pg の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC\_CTYPE、LC\_MESSAGES、およびNLSPATH。  
  
以下の環境変数も pg の実行に影響を与えます。

- COLUMNS      水平方向の画面サイズを決定します。この変数が設定されていない、あるいは NULL に設定されているときは、TERM の値、ウィンドウサイズ、またはボーレート、もしくはこれらの値の組み合わせにより、画面サイズ計算用の端末タイプが表されます。
- LINES      画面に表示する行数を決定します。この変数が設定されていない、あるいは NULL に設定されているときは、TERM の値、ウィンドウサイズ、またはボーレート、もしくはこれらの値の組み合わせにより、画面サイズ計算用の端末タイプが表されます。
- SHELL      ! コマンド用に実行するコマンドインタプリタの名前を決定します。
- TERM      端末の属性を決定します。またオプションとして、TERM の値に基づいた、システムに依存するデータベースの検索を試みます。情報が何も存在しない場合には、カーソルによるアドレス指定が不可能な端末と見なされます。

終了ステータス      次の終了ステータスが返されます。  
  
0      正常終了。  
>0      エラーが発生した。

ファイル      /tmp/pg\*      入力がパイプからの場合の一時ファイル  
                  /usr/share/lib/terminfo/?./\*      端末情報データベース

属性      属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

| 属性タイプ | 属性値 |
|-------|-----|
| CSI   | 有効  |

**関連項目**

[cat\(1\)](#), [grep\(1\)](#), [more\(1\)](#), [terminfo\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#)

**注意事項**

端末入力を待っている間に、pg は BREAK、CTRL-C、および CTRL-\ に応答して実行を終了します。しかし、プロンプトの出ている間であれば、これらのシグナルは、pg の現在のタスクに割り込み、プロンプトモードに戻ります。割り込みによってパイプラインの中の他のコマンドが終了してしまう可能性があるので、パイプから入力を読み取るときには注意が必要です。

区切り記号 /、^、または ? は、検索コマンドから省略することができます。

区切り記号タブが 8 カラムごとに設定されていないと、好ましくない結果が生じることがあります。

端末 I/O オプションを変更する他のコマンドとともに pg をフィルタとして使用すると、端末設定が正しく復元されないことがあります。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pgrep, pkill – 名前またはほかの属性によるプロセスの検出またはシグナル送信                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 形式    | pgrep [-flvx] [-n   -o] [-d <i>delim</i> ] [-P <i>ppidlist</i> ] [-g <i>pgrplist</i> ] [-s <i>sidlist</i> ] [-u <i>euidlist</i> ] [-U <i>uidlist</i> ] [-G <i>gidlist</i> ]<br>pkill [-signal] [-fvx] [-n   -o] [-P <i>ppidlist</i> ] [-g <i>pgrplist</i> ] [-s <i>sidlist</i> ] [-u <i>euidlist</i> ] [-U <i>uidlist</i> ] [-G <i>gidlist</i> ]                                                                                                                                                                                                                                                                                                      |
| 機能説明  | <p>pgrep ユーティリティーは、システム上のアクティブなプロセスを調べて、コマンド行に指定した条件を満たす属性を持つプロセスのプロセスIDを報告します。各プロセスIDは10進数値として出力され、プロセスIDは区切り文字(デフォルトは復帰改行)で区切られます。コマンド行で属性オプションに値を指定するときは、複数の値をコンマで区切って指定できます。たとえば、</p> <p><b>pgrep -G other,daemon</b></p> <p>この例は、実グループIDがotherまたはdaemonであるプロセスを検出します。条件となるオプションを複数指定すると、pgrepは、条件オプションの論理積に一致する属性を持つプロセスを検出します。たとえば、</p> <p><b>pgrep -G other,daemon -U root,daemon</b></p> <p>この例は、次の属性を持つプロセスを検出します。</p> <p>(実グループIDがotherまたはdaemonであり)かつ<br/>(実ユーザーIDがrootまたはdaemonである)</p> <p>pkillの機能はpgrepの機能と同等ですが、検出したプロセスのプロセスIDを出力するのではなく、<a href="#">kill(1)</a>と同様に、各プロセスにシグナルを送信します。シグナルの名前または番号はpkillへの最初のコマンド行オプションとして指定できます。</p> |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-c <i>ctidlist</i></b> 指定されたリスト内にプロセス契約IDがあるプロセスだけに一致します。</p> <p><b>-d <i>delim</i></b> 一致したプロセスID間に置かれる区切り文字を指定します。<b>-d</b>オプションを指定しない場合、デフォルトの区切り文字は復帰改行です。<b>-d</b>オプションはpgrepへのオプションとして指定した場合にのみ有効です。</p> <p><b>-f</b> 正規表現 <i>pattern</i> の照合は、プロセスの引数を含む文字列全体(/proc/<i>nnnnn</i>/psinfo ファイルのpr_psargs フィールドから取得)に対して行われます。<b>-f</b>オプションを指定しないと、正規表現 <i>pattern</i> の照合は、実行可能ファイルの名前(/proc/<i>nnnnn</i>/psinfo ファイルのpr_fname フィールドから取得)に対してのみ行われます。</p> <p><b>-g <i>pgrplist</i></b> 指定したリスト内にプロセスグループIDがあるプロセスだけに一致します。リストにグループ0がある場合、これはpgrepまたはpkillプロセスのプロセスグループIDとして解釈されます。</p>                     |

- G *gidlist* 指定したリスト内に実グループ ID があるプロセスだけに一致します。各グループ ID はグループ名または数値のグループ ID のどちらでも指定できます。
- J *projidlist* 指定したリスト内にプロジェクト ID があるプロセスだけに一致します。各プロジェクト ID はプロジェクト名または数値のプロジェクト ID のどちらでも指定できます。
- l 詳細出力形式。一致したプロセスの名前と ID を出力します。-f オプションが指定されているかどうかによって (前述の説明を参照)、プロセス名は `pr_psargs` または `pr_fname` フィールドから取得されます。-l オプションは `pgrep` へのオプションとして指定した場合にのみ有効です。
- n 指定した条件すべてを満たすプロセスの中で、最新の (最後に作成された) プロセスだけに一致します。このオプションは、-o オプションと同時に使用できません。
- o 指定した条件すべてを満たすプロセスの中で、もっとも古い (最初に作成された) プロセスだけに一致します。このオプションは、-n オプションと同時に使用できません。
- P *ppidlist* 指定したリスト内に親プロセス ID があるプロセスだけに一致します。
- s *sidlist* 指定したリスト内にセッション ID があるプロセスだけに一致します。リストのセッション ID 0 がある場合、これは `pgrep` または `pskill` プロセスのセッション ID として解釈されます。
- t *termlist* 指定したリストの端末に関連するプロセスだけに一致します。各端末は、`/dev` 内にある端末へのパス名 (つまり、`/dev/` 以降のパス名) として指定します。たとえば、`term/a` や `pts/0` などです。
- T *taskidlist* 指定したリストのタスク ID があるプロセスだけに一致します。リストにタスク ID 0 がある場合、これは `pgrep` または `pskill` プロセスのタスク ID として解釈されます。
- u *euidlist* 指定したリストの実効ユーザー ID があるプロセスだけに一致します。各ユーザー ID はログイン名または数値のユーザー ID のどちらでも指定できます。
- U *uidlist* 指定したリストの実ユーザー ID があるプロセスだけに一致します。各ユーザー ID はログイン名または数値のユーザー ID のどちらでも指定できます。
- v 一致の意味を反転します。指定した条件を「満たさない」すべてのプロセスに一致します。

- x 指定した *pattern* に引数の文字列または実行可能ファイル名が完全に一致するプロセスだけに一致します。「完全に一致する」というのは、プロセス引数の文字列または実行可能ファイルの名前のすべての文字が *pattern* に一致することを意味します。
- z *zoneidlist* 指定されたリスト内にゾーン ID があるプロセスだけに一致します。各ゾーン ID は、ゾーン名または数値のゾーン ID のどちらでも指定できます。このオプションは、グローバルゾーンで実行する場合にのみ便利です。kill ユーティリティーを使用して、ほかのゾーン内のプロセスにシグナルを送信する場合、送信元のプロセスは {PRIV\_PROC\_ZONE} 特権を表明しておく必要があります ([privileges\(5\)](#) を参照)。
- signal 一致した各プロセスに送信するシグナルを指定します。シグナルを指定しないと、デフォルトで SIGTERM が送信されます。signal は [signal.h\(3HEAD\)](#) で定義されているシンボル名 (ただし SIG 接頭辞は付けない) または対応するシグナル番号 (10 進数) のどちらでも指定できます。-signal オプションは kill への最初のオプションとして指定した場合にのみ有効です。

オペランド 次のオペランドを指定できます。

*pattern* 実行可能ファイルの名前またはプロセス引数の文字列全体と照合させる拡張正規表現 (ERE) パターンを指定します。ERE 構文の詳細は、[regex\(5\)](#) を参照してください。

## 使用例

例1 プロセス ID を取得する  
sendmail のプロセス ID を取得します。

```
example% pgrep -x -u root sendmail
283
```

例2 プロセスを強制終了する  
最後に作成された xterm を強制終了します。

```
example% pkill -n xterm
```

終了ステータス 次の終了ステータスが返されます。

- 0 1 つまたは複数のプロセスが一致しました。
- 1 どのプロセスも一致しなかった。
- 2 無効なコマンド行オプションが指定された。
- 3 致命的なエラーが発生しました。

ファイル            `/proc/nnnnn/psinfo`      プロセス情報ファイル

属性                属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目           [kill\(1\)](#), [proc\(1\)](#), [ps\(1\)](#), [truss\(1\)](#), [kill\(2\)](#), [signal.h\(3HEAD\)](#), [proc\(4\)](#), [attributes\(5\)](#), [privileges\(5\)](#), [regex\(5\)](#), [zones\(5\)](#)

注意事項           どちらのユーティリティーも `/proc/nnnnn/psinfo` ファイルの `pr_fname` または `pr_psargs` フィールドに対して ERE で *pattern* 引数を照合することができます。これらの文字列の長さ制限は `<sys/procfs.h>` に定義されています。現在の制限よりも長い文字列に一致する可能性があるパターンを使用すると、予期したプロセスと一致しないことがあります。

*pattern* 引数に ERE のメタキャラクタが含まれており、そのメタキャラクタがシェルのメタキャラクタでもある場合、そのメタキャラクタを適切なシェル引用符で囲む必要があります。

終了したプロセスは、`pgrep` と `pkill` のいずれでも検出することはできません。

現在の `pgrep` または `pkill` プロセスは、自身を照合候補とすることはありません。

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pkcs11_inspect – 証明書の内容の出力                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 形式      | /usr/lib/pam_pkcs11/pkcs11_inspect [debug] [config_file=filename]                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明    | <p>pkcs11_inspect は、pam_pkcs11 ライブラリインフラストラクチャーを使用して、証明書の内容を取得および表示します。</p> <p>pkcs11_inspect は、<a href="#">pam_pkcs11(5)</a> PAM モジュールと同じ構成ファイルおよび引数を使用します。また、定義済みのマッパーモジュール (<a href="#">ms_mapper</a> など) を読み込み、それを使用して証明書に必要なエントリ (<a href="#">ms UPN</a> エントリなど) があるかどうかを調べます。</p> <p>マッパーモジュールは、証明書内で適切なエントリを見つけると、それを UTF-8 に変換して stdout に出力します。</p>                                                                                        |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <p><code>config_file=filename</code>      構成ファイルを設定します。デフォルト値は <code>/etc/security/pam_pkcs11/pam_pkcs11.conf</code> です。</p> <p><code>debug</code>                      デバッグ出力を有効にします。</p> <p><a href="#">pam_pkcs11(5)</a> と同じ構成ファイルが使用されるため、<code>pam_pkcs11</code> のすべてのオプションを使用できます。一部のオプションは PAM 以外の環境では意味がないため、無視されます。一部のマッパーオプション (<code>mapfile</code>、<code>ignorecase</code>) も証明書の内容に影響しないため、無視されます。</p> |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0      正常終了。</p> <p>      pkcs11_inspect は、マッパーに関して見つかった証明書の内容のすべてを stdout に出力します。</p> <p>1      エラーが発生した。</p>                                                                                                                                                                                                                                                                                                          |
| 使用例     | <p>例1 pkcs_inspect を使用する</p> <p>次の例では、オプションを指定せずに pkcs_inspect コマンドを実行しています。</p> <pre>% pkcs11_inspect</pre> <p>例2 オプションを指定して pkcs_inspect を使用する</p> <p>次の例では、オプションを指定して pkcs_inspect コマンドを実行しています。</p> <pre>% pkcs11_inspect debug config_file=\${HOME}/.pam_pkcs11.conf</pre>                                                                                                                                                                    |
| ファイル    | <code>/etc/security/pam_pkcs11/pam_pkcs11.conf</code>                                                                                                                                                                                                                                                                                                                                                                                              |



著者 Juan Antonio Martinez ([jonsito@teleline.es](mailto:jonsito@teleline.es))

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                    |
|-------------|----------------------------------------|
| 使用条件        | library/security/pam/module/pam-pkcs11 |
| インタフェースの安定性 | 不確実                                    |

関連項目 [pklogin\\_finder\(1\)](#), [attributes\(5\)](#), [pam\\_pkcs11\(5\)](#)

PAM-PKCS11 User Manual, [http://www.opensc-project.org/pam\\_pkcs11](http://www.opensc-project.org/pam_pkcs11)

名前 pkg – Image Packaging System 取得クライアント

形式

```
/usr/bin/pkg [options] command [cmd_options] [operands]

/usr/bin/pkg refresh [-q] [--full] [publisher ...]

/usr/bin/pkg install [-nvq] [-C n] [-g path_or_uri ...]
 [--accept] [--licenses] [--no-be-activate] [--no-index]
 [--no-refresh] [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 [--reject pkg_fmri_pattern ...] pkg_fmri_pattern ...

/usr/bin/pkg uninstall [-nvq] [-C n] [--no-be-activate]
 [--no-index] [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 pkg_fmri_pattern ...

/usr/bin/pkg update [-fnvq] [-C n] [-g path_or_uri ...]
 [--accept] [--licenses] [--no-be-activate] [--no-index]
 [--no-refresh] [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 [--reject pkg_fmri_pattern ...] [pkg_fmri_pattern ...]

/usr/bin/pkg list [-Hafnsuv] [-g path_or_uri ...]
 [--no-refresh] [pkg_fmri_pattern ...]

/usr/bin/pkg info [-lr] [-g path_or_uri ...] [--license]
 [pkg_fmri_pattern ...]

/usr/bin/pkg contents [-Hmr] [-a attribute=pattern ...]
 [-g path_or_uri ...] [-o attribute ...] [-s sort_key]
 [-t action_name ...] [pkg_fmri_pattern ...]

/usr/bin/pkg search [-HIaflpr] [-o attribute ...]
 [-s repo_uri] query

/usr/bin/pkg verify [-Hqv] [pkg_fmri_pattern ...]

/usr/bin/pkg fix [--accept] [--licenses]
 [pkg_fmri_pattern ...]

/usr/bin/pkg revert [-nv] [--no-be-activate]
 [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 (--tagged tag-name ... | path-to-file ...)

/usr/bin/pkg mediator [-aH] [-F format] [mediator ...]

usr/bin/pkg set-mediator [-nv] [-I implementation]
 [-V version] [--no-be-activate]
 [--no-backup-be | --require-backup-be]
```

```

 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 mediator ...

/usr/bin/pkg unset-mediator [-nvIV] [--no-be-activate]
 [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 mediator ...

/usr/bin/pkg variant [-H] [variant.variant_name ...]

/usr/bin/pkg change-variant [-nvq] [-C n] [-g path_or_uri ...]
 [--accept] [--licenses] [--no-be-activate]
 [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 variant_name=value ...

/usr/bin/pkg facet [-H] [facet_name ...]

/usr/bin/pkg change-facet [-nvq] [-C n] [-g path_or_uri ...]
 [--accept] [--licenses] [--no-be-activate]
 [--no-backup-be | --require-backup-be]
 [--backup-be-name name]
 [--deny-new-be | --require-new-be] [--be-name name]
 facet_name=[True|False|None] ...

/usr/bin/pkg avoid [pkg_fmri_pattern ...]

/usr/bin/pkg unavoid [pkg_fmri_pattern ...]

/usr/bin/pkg freeze [-n] [-c reason] [pkg_fmri_pattern] ...

/usr/bin/pkg unfreeze [-n] [pkg_name_pattern] ...

/usr/bin/pkg property [-H] [propname ...]

/usr/bin/pkg set-property propname propvalue

/usr/bin/pkg add-property-value propname propvalue

/usr/bin/pkg remove-property-value propname propvalue

/usr/bin/pkg unset-property propname ...

/usr/bin/pkg publisher [-HPn] [-F format] [publisher ...]

/usr/bin/pkg set-publisher [-Ped] [-k ssl_key] [-c ssl_cert]
 [-g origin_to_add | --add-origin origin_to_add ...]
 [-G origin_to_remove | --remove-origin origin_to_remove ...]
 [-m mirror_to_add | --add-mirror mirror_to_add ...]
 [-M mirror_to_remove | --remove-mirror mirror_to_remove ...]
 [--enable] [--disable] [--no-refresh] [--reset-uuid]
 [--non-sticky] [--sticky] [--search-after publisher]
 [--search-before publisher] [--search-first]

```

```

[--approve-ca-cert path_to_CA]
[--revoke-ca-cert hash_of_CA_to_remove]
[--unset-ca-cert hash_of_CA_to_remove]
[--set-property name_of_property=value]
[--add-property-value name_of_property=value_to_add]
[--remove-property-value name_of_property=value_to_remove]
[--unset-property name_of_property_to_delete]
[--proxy proxy_to_use] publisher

/usr/bin/pkg set-publisher -p repo_uri [-Ped]
[-k ssl_key] [-c ssl_cert] [--non-sticky] [--sticky]
[--search-after publisher] [--search-before publisher]
[--search-first] [--approve-ca-cert path_to_CA]
[--revoke-ca-cert hash_of_CA_to_remove]
[--unset-ca-cert hash_of_CA_to_remove]
[--set-property name_of_property=value]
[--add-property-value name_of_property=value_to_add]
[--remove-property-value name_of_property=value_to_remove]
[--unset-property name_of_property_to_delete]
[--proxy proxy_to_use] [publisher]

/usr/bin/pkg unset-publisher publisher ...

/usr/bin/pkg history [-HnL] [-t [time | time-time],...]
[-o column,...] [-n number]

/usr/bin/pkg purge-history

/usr/bin/pkg rebuild-index

/usr/bin/pkg update-format

/usr/bin/pkg version

/usr/bin/pkg help

/usr/bin/pkg image-create [-FPUfz] [--force]
[--full | --partial | --user] [--zone]
[-k ssl_key] [-c ssl_cert] [--no-refresh]
[--variant variant_name=value ...]
[-g path_or_uri | --origin path_or_uri ...]
[-m uri | --mirror uri ...]
[--set-property name_of_property=value]
[--facet facet_name=(True|False) ...]
[(-p | --publisher) [name=]repo_uri] dir

```

## 機能説明

pkg は Image Packaging System 用の取得クライアントです。有効な構成では、pkg を呼び出すことにより、パッケージをインストールする場所を作成し、イメージを呼び出し、パッケージをイメージにインストールすることができます。パッケージはパブリッシャーによって公開され、パブリッシャーはそれらのパッケージを1つ以上のリポジトリで、またはパッケージアーカイブで入手可能にすることができます。

す。pkg はパブリッシャーのリポジトリまたはパッケージアーカイブからパッケージを取得し、パッケージをイメージにインストールします。

パブリッシャーの名前によって、人、人のグループ、または組織が1つ以上のパッケージのソースとして識別されます。パブリッシャーの名前の競合を避け、パブリッシャーを識別しやすくするために、パッケージを公開するエンティティを表すドメイン名をパブリッシャーの名前として使用することがベストプラクティスです。

リポジトリは、クライアントがパッケージの内容(プログラムやドキュメントなど、パッケージの内部に含まれるファイル)とメタデータ(パッケージの名前や説明など、パッケージについての情報)を公開および取得できる場所です。たとえば、example.org という名前のパブリッシャーは、http://example.org/repository という URI にリポジトリを配置することができます。

pkg では、パッケージをアンインストールしたり、(利用可能なパッケージの一覧などの)パブリッシャーメタデータをリフレッシュしたり、イメージにインストールされたパッケージを検証したり、イメージからさまざまなトークンを照会したりすることもできます。これらのクエリーは pkg(5) リポジトリで構成することもできます。

イメージには3つの種類があります。完全なシステムを提供できるフルイメージ、フルイメージ(親イメージ)にリンクされるがそれ自身では完全なシステムを提供しない部分イメージ、およびユーザーイメージです。

## オプション

サポートしているオプションは、次のとおりです。

-?

--help

使用方法に関するメッセージを表示します。

-R *dir*

--image-dir *dir*

*dir* をルートとするイメージを操作します。ディレクトリが指定されなかったか、または環境に基づいて決定される場合、デフォルトは / です。詳細は、「環境変数」のセクションを参照してください。

## サブコマンド

サポートされているサブコマンドは次のとおりです。

pkg refresh [-q] [- -full] [*publisher* ...]

クライアントの利用可能パッケージの一覧およびすべての発行元の発行元メタデータを更新します。

*publisher*

クライアントの利用可能パッケージの一覧、および指定された発行元の発行元メタデータのみを更新します。

-q

リクエストされた操作の実行中、進捗状況メッセージを表示しません。

--full

増分更新を試みる代わりにすべての発行元メタデータを強制的に完全取得し、操作中に使用されるすべてのプロキシでキャッシュデータを無視するようにリクエストします。このオプションはトラブルシューティング目的に用意されており、通常時は使用しないでください。

```
pkg install [-nvq] [-C n] [-g path_or_uri ...] [--accept] [--licenses] [--no-be-activate]
[--no-index] [--no-refresh] [--no-backup-be | --require-backup-be]
[--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name]
[--reject pkg_fmri_pattern ...] pkg_fmri_pattern ...
```

指定されたパッケージをインストールし、イメージにインストールされたパッケージで許容されている *pkg\_fmri\_pattern* と一致する最新バージョンにパッケージを更新します。パッケージの最新バージョンを明示的に要求するには、*pkg\_fmri\_pattern* のバージョン部分に *latest* を使用します。たとえば、*vim@latest* のように指定します。

パッケージは、発行元検索順およびスティッキネスに基づいて選択されます。検索順およびスティッキネスについては、*pkg publisher* および *pkg set-publisher* コマンドを参照してください。*pkg\_fmri\_pattern* が発行元を指定しない場合、一致するパッケージを提供する最初の発行元がインストール元として使用されます。該当する発行元がこのイメージ内にインストール可能なパッケージのバージョンを提供しない場合、インストール操作は失敗します。このイメージにインストール可能なパッケージのバージョンを提供する発行元を確認するには、*pkg list -a* コマンドを使用します。

複数の *pkg\_fmri\_pattern* が指定され、かつ指定されたパッケージのいずれかをこのイメージにインストールできない場合、指定されたパッケージは一切インストールされません。

インストールプロセスの間に、一部の設定ファイルの名前変更または置換が行われる場合があります。どのファイルを保持するかをパッケージシステムが決定する方法と、パッケージ操作中にファイルが保持されるしくみについては、*pkg(5)* のマニュアルページのファイルアクションに関する項目を参照してください。

パッケージが回避リストにある場合は、インストールすると回避リストから削除されます。

-g path\_or\_uri

指定されたパッケージリポジトリまたはアーカイブを、パッケージデータの取得元になるイメージ内のソースリストに一時的に追加します。クライアント SSL 証明書が必要なりポジトリは、このオプションとともに使用できません。このオプションは複数回指定できます。

どのバージョンのパッケージを使用するかを決定するときは、イメージ内に構成されているが、指定された *path\_or\_uri* ソースに見つからない発行元が優先されます。インストールするパッケージのバージョンが、イメージ内で構成されている発行元および *path\_or\_uri* ソースから提供されている場合、クラ

イアントはそのパッケージの内容を *path\_or\_uri* ソースから取得します。インストールまたは更新のあとに、発行元によって提供され、イメージ内で構成されていないパッケージがある場合は、起点なしでイメージ構成に追加されます。イメージ内で構成されている発行元を表示するには、`pkg publisher` コマンドを使用します。

**-n**

パッケージの変更は行わずに試しに操作を実行します。

**-v**

リクエストされた操作の実行中に詳細な進捗状況メッセージを出力し、詳細な計画情報(ファセット、メディアータ、バリエーションの変更など)を表示します。このオプションを複数回指定して、表示される計画情報の量を増やすことができます。

**-q**

リクエストされた操作の実行中、進捗状況メッセージを表示しません。

**-C *n***

同時に更新する子イメージの数を指定します。子イメージ(通常はゾーン)内への再帰時に、最大 *n* 個の子イメージを同時に更新します。更新する子イメージのデフォルト数は、1 です。*n* が 0 または負数の場合は、すべての子イメージが同時に更新されます。「環境変数」セクションの「`PKG_CONCURRENCY`」も参照してください。

**--accept**

更新またはインストールされるパッケージのライセンス条項に同意することを示します。このオプションを指定しないと、パッケージのライセンスに同意が必要になった場合、インストール操作は失敗します。

**--licenses**

この操作の一環としてインストールまたは更新されるパッケージのすべてのライセンスを表示します。

**--no-backup-be**

バックアップブート環境を作成しません。

**--no-be-activate**

ブート環境が作成される場合に、それを次回ブート時にアクティブなブート環境として設定しません。詳細は、`beadm(1M)` のマニュアルページを参照してください。

**--no-index**

操作が正常に完了したあとに検索インデックスを更新しません。

**--no-refresh**

入手可能パッケージやその他のメタデータの最新リストを取得するために、イメージの発行元用のリポジトリへのアクセスを試みません。

**--backup-be-name *name***

指定された引数を使って、作成されたバックアップブート環境に名前を付けます。**--backup-be-name** を使用すると **--require-backup-be** が暗黙的に指定されます。`beadm(1M)` のマニュアルページも参照してください。

**--be-name *name***

新たに作成されたブート環境の名前を、指定された引数になるように変更します。**--be-name** の使用は、暗黙的に **--require-new-be** を示します。`beadm(1M)` のマニュアルページも参照してください。

**--require-backup-be**

新しいブート環境が作成されない場合に、常にバックアップブート環境を作成します。このオプションを指定しないと、イメージポリシーに基づいてバックアップブート環境が作成されます。バックアップブート環境がいつ自動的に作成されるかについての説明は、次の「イメージプロパティ」の `be-policy` を参照してください。

**--require-new-be**

常に新しいブート環境を作成します。このオプションを指定しないと、イメージポリシーに基づいてブート環境が作成されます。ブート環境がいつ自動的に作成されるかについての説明は、次の「イメージプロパティ」の `be-policy` を参照してください。このオプションを **--require-backup-be** と組み合わせることはできません。

**--deny-new-be**

新しいブート環境を作成しません。新しいブート環境が必要な場合、この操作は実行されません。

**--reject *pkg\_fmri\_pattern***

指定されたパターンと一致する名前を持つパッケージはインストールされません。一致するパッケージがすでにインストールされている場合、それらはこの操作の一環として削除されます。グループ依存関係のターゲットである拒否対象パッケージは回避リストに登録されます。

```
pkg uninstall [-nvq] [-C n] [--no-be-activate] [--no-index] [--no-backup-be |
--require-backup-be] [--backup-be-name name] [--deny-new-be | --require-new-be]
[--be-name name] pkg_fmri_pattern ...
```

`pkg_fmri_pattern` に一致するインストール済みパッケージを削除します。

パッケージがグループ依存関係の対象である場合、パッケージをアンインストールするとそのパッケージは回避リストに登録されます。後述する `avoid` サブコマンドを参照してください。

オプションの説明については、前述の `install` コマンドを参照してください。

コマンド出力で、新しいブート環境が作成されたことを示すメッセージに注意してください。新しいブート環境を作成してアクティブにすると、それが次回リ



ブート時のデフォルトのブート環境になります。ブート環境の管理については、`beadm(1M)` のマニュアルページを参照してください。

```
pkg update [-f nvq] [-C n] [-g path_or_uri ...] [--accept] [--licenses] [--no-be-activate]
[--no-index] [--no-refresh] [--no-backup-be | --require-backup-be]
[--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name]
[--reject pkg_fmri_pattern ...] [pkg_fmri_pattern ...]
```

現在のイメージ内のすべてのインストール済みパッケージを、インストール済みパッケージと発行元構成によりシステムに適用される制約で許可される最新のバージョンに更新します。

#### *pkg\_fmri\_pattern*

現在のイメージにインストール済みの、指定されたパッケージのみを更新します。指定された *pkg\_fmri\_pattern* パターンの1つがアスタリスク (\*) である場合、現在のイメージ内のすべてのインストール済みパッケージを前述のとおり更新します。

パッケージの最新バージョンを明示的に要求するには、*pkg\_fmri\_pattern* のバージョン部分に `latest` を使用します。たとえば、`vim@latest` のように指定します。

すでにインストールされているものより古い、または新しいバージョンを指定することで、特定のパッケージのインプレースダウングレードまたはアップグレードを実行できます。パッケージの名前変更または廃止の境界をまたがった特定パッケージの更新はサポートされていません。

保持される構成ファイルのうち、ダウングレードされるパッケージの一部であり、元のバージョンがインストールされたあとに変更されたファイルは、拡張子 `.update` を使用して名前が変更されます。どのファイルを保持するかをパッケージシステムが決定する方法と、パッケージアップグレード中にファイルが保持されるしくみについては、`pkg(5)` のマニュアルページのファイルアクションに関する項目を参照してください。

複数の *pkg\_fmri\_pattern* が指定され、かつ指定されたパッケージのいずれかをこのイメージに更新できない場合、指定されたパッケージは一切更新されません。

#### *-f*

すべてのインストール済みパッケージを更新するときにクライアントに対する最新状態チェックを実行しません。

その他のすべてのオプションについては、前述の `install` コマンドを参照してください。

コマンド出力で、新しいブート環境が作成されたことを示すメッセージに注意してください。新しいブート環境を作成してアクティブにすると、それが次

回、`--no-be-activate` オプションを指定しない場合のリブート時のデフォルトブート環境になります。ブート環境の管理については、`beadm(1M)` のマニュアルページを参照してください。

`pkg list [-Hafnsuv] [-g path_or_uri ...] [--no-refresh] [pkg_fmri_pattern ...]`

バージョンやインストール状態などの情報を含む、現在のイメージ内のインストール済みの全パッケージのリストを表示します。デフォルトでは、異なるアーキテクチャーまたはゾーンタイプのパッケージバリエーションは除外されません。通常の実出力は3列形式です。

| NAME (PUBLISHER)       | VERSION                | IFO |
|------------------------|------------------------|-----|
| system/core-os         | 0.5.11-0.175.0.0.0.2.1 | i-- |
| x11/wm/fvwm (fvwm.org) | 2.6.5                  | i-- |

最初の列にはパッケージの名前が表示されます。パッケージのインストール元(または、インストールされていない場合は提供元)であるパブリッシャーがパブリッシャー検索順で先頭でない場合、パッケージ名のあとに一覧表示されるパブリッシャー名は括弧で囲まれています。2番目の列にはパッケージのリリースバージョンとブランチバージョンが表示されます。リリースバージョンとブランチバージョンについて、およびバリエーションについては、`pkg(5)` のマニュアルページを参照してください。

最後の列には、パッケージのステータスを示す一連のフラグが表示されます。

- I列の `i` は、パッケージがインストールされていることを示します。
- F列の `f` は、パッケージが凍結されていることを示します。
- o列の `o` は、パッケージが廃止されていることを示します。o列の `r` は、パッケージの名前が変更されたことを示します(廃止の形態の1つです)。

`pkg_fmri_pattern`

指定したパッケージのみを一覧表示します。

`-H`

一覧からヘッダーを省略します。

`-a`

インストール済みパッケージと、インストールのために入手可能なパッケージの最新バージョンを一覧表示します。インストール済みの `incorporation` によって、またイメージのバリエーションによって許可されている場合、そのパッケージはインストールのために入手可能であるとみなされます。1つ以上のパターンを指定した場合、指定されたパターンに一致し、インストール済みの `incorporation` およびイメージのバリエーションによって許可されている最新バージョンが一覧表示されます。`-a` を指定しない場合、インストール済みパッケージのみを一覧表示します。

`-af`

`incorporation` の制約またはインストール状態に関係なく、すべてのバリエーションについてすべてのパッケージのすべてのバージョンを一覧表示します。これ

らのオプションを使用するとき、パッケージの最新バージョンを明示的に一覧表示するには、*pkg\_fmri\_pattern* のバージョン部分に *latest* を使用します。たとえば、*vim@latest* のように指定します。

#### **-g path\_or\_uri**

指定されたパッケージリポジトリまたはアーカイブを、操作のためのパッケージデータのソースとして使用します。クライアント SSL 証明書が必要なりポジトリは、このオプションとともに使用できません。このオプションは複数回指定できます。-n を指定しない場合、-g を使用すると -a が暗黙的に指定されます。

#### **-n**

インストール状態に関係なく、すべての既知のパッケージの最新バージョンを表示します。

#### **-s**

パッケージ名とサマリーを示す 1 行の短縮形式を表示します。このオプションは -a、-n、-u、または -v とともに使用できます。

#### **-u**

新しいバージョンが入手可能なパッケージのみを一覧表示します。このオプションは -g とともに使用できません。

#### **-v**

発行元と完全バージョンを含む、フルパッケージ FMRI をすべて最初の列に表示します (VERSION 列は消えます)。このオプションは -a、-n、または -u とともに使用できます。

#### **--no-refresh**

入手可能パッケージやその他のメタデータの最新リストを取得するために、イメージの発行元用のリポジトリへのアクセスを試みません。

**pkg info [-lr] [-g path\_or\_uri ...] [--license] [pkg\_fmri\_pattern ...]**

現在のイメージにインストールされているすべてのパッケージについての情報を、人間が判読できる形式で表示します。

#### **pkg\_fmri\_pattern**

指定したパッケージの情報のみを表示します。

#### **-g path\_or\_uri**

指定されたパッケージリポジトリまたはアーカイブを、操作のためのパッケージデータのソースとして使用します。クライアント SSL 証明書が必要なりポジトリは、このオプションとともに使用できません。このオプションは複数回指定できます。-g を使用すると -r が暗黙的に指定されます。

#### **-l**

インストール済みのパッケージの情報だけを表示します。これはデフォルトです。

-r

最新の入手可能バージョンに基づいてパッケージを照合し、イメージの構成済み発行元のリポジトリから、現在インストールされていないパッケージの情報を取得します (必要な場合)。このオプションを使用するときは、少なくとも1つのパッケージを指定する必要があります。-rを指定しない場合、デフォルトでインストール済みパッケージのみが表示されます。

--license

パッケージのライセンステキストを表示します。このオプションは -l または -r と組み合わせることができます。

pkg contents [-Hmr] [-a attribute=pattern ...] [-g path\_or\_uri ...] [-o attribute,...] [-s sort\_key] [-t action\_name ...] [pkg\_fmri\_pattern ...]

イメージ内のすべてのインストール済みパッケージの内容 (アクション属性) を表示します。オプションが指定されていない場合、現在のイメージにインストールされているアクションの path 属性の値をアルファベット順で並べ替えて表示します。アクションとそれらの属性については、pkg(5) のマニュアルページのアクションに関する項目を参照してください。下の疑似属性名のリストも参照してください。

*pkg\_fmri\_pattern*

指定したパッケージの内容のみを表示します。

-H

出力からヘッダーを省略します。

-a attribute=pattern

名前がオプション引数で指定され、値がオプション引数 (属性名と等号に続く部分) の (glob) パターンに一致する属性を持つアクションに出力を限定します。複数の -a オプションを指定した場合、それらのいずれかに一致するアクションが表示されます。

-g path\_or\_uri

指定されたパッケージリポジトリまたはアーカイブからこのイメージにインストール可能なパッケージの情報を表示します。クライアント SSL 証明書が必要なリポジトリは、このオプションとともに使用できません。インストール可能なパッケージには、現在インストールされているパッケージと、バリエーションやファセットの制限などのこのイメージへのインストールの条件を満たすその他のパッケージが含まれます。このオプションは複数回指定できます。-g を使用すると -r が暗黙的に指定されます。

-m

このイメージにインストールできないアクションを含めて、指定されたパッケージのすべてのアクションのすべての属性を表示します。

-o attribute

指定された属性を、リストの先頭属性の値に従って並べ替えて表示します。-o オプションは複数回指定できます。または、属性名をコンマで区切る

ことにより、1つの `-o` オプションの引数として複数の属性を指定できます。要求された属性を持つアクションのみが表示されます。

`-r`

このイメージ内で構成された発行元のリポジトリからこのイメージにインストール可能なパッケージの最新バージョンの情報を表示します。インストール可能なパッケージには、現在インストールされているパッケージと、バリエーションやファセットの制限などのこのイメージへのインストールの条件を満たすその他のパッケージが含まれます。このオプションを使用するときは、少なくとも1つのパッケージを指定する必要があります。

`-s sort_key`

指定されたアクション属性で、アクションをソートします。このオプションを指定しない場合、デフォルトではパスによって、または `-o` オプションで最初に指定された属性によってソートします。 `-s` オプションは複数回指定できます。

`-t action_name`

指定されたアクションのみを一覧表示します。複数のアクションをコンマ区切りリストで指定できます。 `action_name` の値は、 `pkg(5)` のマニュアルページの「アクション」内に一覧表示されているアクションのいずれかで、 `file`、 `directory`、 `driver`、 `depend`、 `set` などです。このオプションは複数回指定できます。

利便性のために、いくつかの特殊な疑似属性名を使用できます。

`action.hash`

アクションがペイロードを伝送する場合、アクションのハッシュの値です。

`action.key`

アクションのキー属性の値です。たとえば、 `file` アクションの場合、鍵属性はファイルのパスです。キー属性のないアクションもあります。

`action.name`

アクションの名前です。たとえば、ファイルアクションの場合、これは `file` です。

`action.raw`

一致するアクションのすべての属性。

`pkg.fmri`

アクションを包含しているパッケージのフル形式 FMRI (たとえば、 `pkg://solaris/group/feature/amp@0.5.11,5.11-0.175.0.0.0.2.1:20120705T153434Z`) です。

`pkg.name`

アクションを包含しているパッケージの名前 (たとえば、 `web/amp`) です。

`pkg.publisher`

アクションを包含しているパッケージの発行元 (たとえば、 `solaris`) です。

**pkg.shortfmri**

アクションを包含しているパッケージのショート形式 FMRI  
(pkg://solaris/group/feature/amp@0.5.11,5.11-0.175 など) です。

関連するサブコマンドは **contents** および **search** であり、どちらもパッケージの内容についてシステムをクエリーします。**contents** サブコマンドは、1つまたは複数のインストールされているか、インストール可能なパッケージ内のアクションを、指定されたオプションに基づいて出力をフィルタ処理して表示します。**search** サブコマンドは逆方向からクエリーを行い、ユーザーが指定したトークンを含むすべてのパッケージの名前を表示します。

各サブコマンドで実行できるクエリーの一部は、他方でも実行できます。サブコマンドの選択は慎重に行ってください。クエリーによっては、もう一方のほうがより自然に実行できる場合があります。

**pkg search [-HIafldr] [-o attribute,...] [-s repo\_uri] query**

*query* の一致を検索し、結果を表示します。次の *query* の説明を参照してください。

**-H**

出力からヘッダーを省略します。

**-I**

大文字/小文字を区別する検索を使用します。

**-a**

検索を実行して、一致するアクションに関する情報を表示します。これはデフォルトです。

**-f**

パッケージのバージョンに関係なく、すべての結果を表示します。**search** はデフォルトで、現在インストールされているバージョンよりも古いパッケージ、および現在の **incorporation** によって除外されているパッケージバージョンからの結果を取り除きます。

**-l**

イメージのインストール済みパッケージを検索します。

**-l** と **-r** (または **-s**) の両方を同時に指定できます。この場合、ローカル検索とリモート検索の両方が実行されます。

**-p**

一部のアクションが各クエリー条件に一致するパッケージを表示します。このオプションを使用することは、クエリーの各条件を山括弧 (<>) で囲むことと等価です。<> 演算子の詳細は、次の *query* を参照してください。

**-r**

イメージの発行元に対応するリポジトリを検索します。これはデフォルトです。

`-l` と `-r` (または `-s`) の両方を同時に指定できます。この場合、ローカル検索とリモート検索の両方が実行されます。

#### `-o attribute`

結果の列を制御します。`-o` オプションは複数回指定できます。または、属性名をコンマで区切ることで、1つの `-o` オプションの引数として複数の属性を指定できます。前述した疑似属性に加えて、検索結果用の次の属性が定義されています。

#### `search.match`

検索クエリーに一致した文字列。

#### `search.match_type`

検索クエリーに一致した文字列を含む属性。

#### `-s repo_uri`

指定された URI に位置する `pkg(5)` リポジトリを検索します。これは複数回指定できます。パッケージアーカイブはサポートされていません。

#### `query`

デフォルトでは、`query` は完全一致する一連の条件として解釈されます。`?` および `*` 文字を `glob(3C)` 形式のワイルドカードとして使用でき、より柔軟なクエリー一致が可能になります。

単純なトークン一致およびワイルドカード検索に加えて、より複雑なクエリー言語がサポートされています。単一引用符または二重引用符 (`'` または `"`) を使用することにより、語句を検索できます。`pkg` が実際に `'` または `"` を認識するように、必ずシェルを考慮に入れてください。

AND と OR を使用する論理検索がサポートされています。

どのトークンがインデックス化されるかはアクションに依存しますが、コンテンツハッシュとパス名を含めることができます。アクションとそれらの属性については、`pkg(5)` のマニュアルページのアクションに関する項目を参照してください。前述の `pkg contents` および `-o` 内の疑似属性名のリストも参照してください。

構造化されたクエリーは、次の構文でサポートされます:

`pkg_name:action_name:index:token`

`action_name` の値は、`pkg(5)` マニュアルページの「アクション」に一覧表示されているアクションのいずれかです。`index` は、アクションの属性です。`index` の値は、`token` と一致している必要があります。

すべてのアクション属性が検索可能なわけではありません。たとえば、`mode` は `file` アクションの属性ですが、`mode` は `index` の有効な値ではありません。



*index* の一部の値は、アクション属性ではなく、ほかの属性から派生した値です。たとえば、パスの最後のコンポーネントを取ることで、*index* を、どのアクションの属性でもないが、file または dir アクションの *path* 属性から派生した *basename* にできます。

アクションタイプが異なると、有効な *index* 値も異なります。このドキュメントには、使用可能な値がすべて含まれているわけではありません。より有用な *index* 値には、ファイルシステムアクションの *basename* および *path*、*depend* アクションの依存性タイプ (たとえば、*require*、*optional*、*group*)、および *driver* アクションの *driver\_name* および *alias* などがあります。

特殊な *index* 値の 1 つに、*set* アクションの *name* 属性の値があります。この場合、*token* が、指定された *name* 属性に対応する *value* 属性の値と照合されます。たとえば、次の検索により、Development/Databases または System/Databases のいずれかに分類されるパッケージを検索します。「使用例」セクションで、SMF サービスを検索する例を参照してください。

```
$ pkg search info.classification:databases
```

構造化クエリー内の欠落したフィールドは、暗黙的にワイルドカード化されます。*basename:pkg* の検索は、*basename* の *index* を持ち、次の部分出力に示すような *token* *pkg* に一致するすべてのパッケージ内のすべてのアクションに一致します:

```
$ pkg search basename:pkg
```

| INDEX    | ACTION | VALUE         | PACKAGE                                  |
|----------|--------|---------------|------------------------------------------|
| basename | dir    | usr/share/pkg | pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1  |
| basename | dir    | var/sadm/pkg  | pkg:/package/svr4@0.5.11-0.175.0.0.0.2.1 |
| basename | dir    | var/spool/pkg | pkg:/package/svr4@0.5.11-0.175.0.0.0.2.1 |
| basename | file   | usr/bin/pkg   | pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1  |

次の完全な出力に示すように、別のフィールドを追加すると検索が絞り込まれます:

```
$ pkg search file:basename:pkg
```

| INDEX    | ACTION | VALUE       | PACKAGE                                 |
|----------|--------|-------------|-----------------------------------------|
| basename | file   | usr/bin/pkg | pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 |

*pkg\_name* および *token* フィールドでは明示的ワイルドカードがサポートされません。*action\_name* および *index* は、正確に一致する必要があります。

ファイルと依存関係の検索例については、「使用例」セクションを参照してください。

アクションをこれらのアクションを含むパッケージに変換するには、次の部分出力に示すように、< を使用します:

```
$ pkg search \<pkg\>
```

| PACKAGE                                  | PUBLISHER |
|------------------------------------------|-----------|
| pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1  | solaris   |
| pkg:/package/svr4@0.5.11-0.175.0.0.0.2.1 | solaris   |



-a オプションを使用する場合(デフォルト)、token を検索した結果は token に一致するアクションについての情報である一方で、<token> を検索した結果は、token に一致するアクションを包含しているパッケージの一覧です。

`pkg verify [-Hqv] [pkg_fmri_pattern ...]`

現在のイメージ内で、インストールされているすべてのパッケージのインストールを検証します。関連する発行元の現在の署名ポリシーが `ignore` でない場合、各パッケージの署名がポリシーに基づいて検証されます。署名ポリシーが適用されるしくみについては、後述する「イメージプロパティ」の `signature-policy` で説明します。

*pkg\_fmri\_pattern*

現在のイメージ内で、指定されたインストール済みパッケージのインストールのみを検証します。

-H

検証の出力からヘッダーを省略します。

-q

何も出力しませんが、致命的なエラーがある場合はエラーを返します。

-v

パッケージに関する情報メッセージを含めます。

`pkg fix [--accept] [--licenses] [pkg_fmri_pattern ...]`

`pkg verify` で報告されたエラーをすべて修正します。インストール済みパッケージの内容は、独自の内容解析に基づいて検証されるため、ほかのプログラムの場合とは異なる結果が返されることがあります。

*pkg\_fmri\_pattern*

現在のイメージ内で、指定されたインストール済みパッケージに対する `pkg verify` により報告されたエラーを修正します。

--accept

更新またはインストールされるパッケージのライセンス条項に同意することを示します。このオプションを指定しないと、パッケージのライセンスに同意が必要になった場合、操作は失敗します。

--licenses

この操作の一環としてインストールまたは更新されるパッケージのすべてのライセンスを表示します。

`pkg revert [-nv] [-no-be-activate] [--no-backup-be | --require-backup-be ]`  
`[--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name]`  
`(--tagged tag-name ... | path-to-file ...)`

`pkg(5)` パッケージにより配布されたパッケージを、配布時の状態に戻します。ファイルの所有権および保護も復元されます。

注意—一部の編集可能ファイルをデフォルト値に戻すと、システムがブート不可になったり、その他の異常動作の原因になったりする可能性があります。

--tagged *tag-name*

*tag-name* でタグ付けされたすべてのファイルを元に戻します。

*path-to-file*

指定されたファイルを元に戻します。

その他のすべてのオプションについては、前述の `install` コマンドを参照してください。

`pkg mediator [-aH] [-F format] [mediator ...]`

現在選択されているバージョンやすべてのメディエータの実装を表示します。

*mediator*

現在選択されているバージョンや指定されたメディエータの実装のみを表示します。

-a

現在インストールされているパッケージに設定可能なメディエーションを一覧表示します。

-F

代替出力形式を指定します。現時点では `tsv` (タブ区切り値) のみが有効です。

-H

一覧からヘッダーを省略します。

`pkg set-mediator [-nv] [-I implementation] [-V version] [--no-be-activate]`

`[--no-backup-be | --require-backup-be] [--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name] mediator ...`

現在のイメージ内の指定されたメディエータのバージョンと実装を設定します。

-I *implementation*

使用するメディエート対象インタフェースの実装を設定します。デフォルトでは、バージョンが指定されない場合、すべての実装バージョンが許可されます。バージョンなしで実装を指定するには、アット記号 (@) を付加します。

-V *version*

使用するメディエート対象インタフェースのバージョンを設定します。

指定されたメディエータのバージョンと実装のどちらかまたは両方が現在入手できない場合、指定されたメディエータを使用するリンクはすべて削除されます。

その他のすべてのオプションについては、前述の `install` コマンドを参照してください。

`pkg unset-mediator [-nvIV] [--no-be-activate] [--no-backup-be |`

`--require-backup-be] [--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name] mediator ...`

指定されたメディエータのバージョンと実装をシステムデフォルトに戻します。

-I

メディエート対象インタフェースの実装のみを元に戻します。

-V

メディエート対象インタフェースのバージョンのみを元に戻します。

その他のすべてのオプションについては、前述の `install` コマンドを参照してください。

`pkg variant [-H] [variant.variant_name ...]`

このイメージに設定されているすべてのバリエーションの現在値を表示します。バリエーションの詳細については、`pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

*variant.variant\_name*

このイメージで設定されている、指定されたバリエーションのみの現在値を表示します。

-H

一覧からヘッダーを省略します。

`pkg change-variant [-nvq] [-C n] [-g path_or_uri ...] [--accept] [--licenses] [--no-be-activate] [--no-backup-be | --require-backup-be] [--backup-be-name name] [--deny-new-be | --require-new-be] [--be-name name] variant_name= value ...`

現在のイメージに設定され、指定されているバリエーションの値を変更します。

バリエーションの値を変更すると、パッケージの内容が削除、更新、またはインストールされることがあります。バリエーションの値を変更すると、新しいイメージ構成を満たすために、パッケージ全体がインストール、更新、または削除されることもあります。バリエーションの詳細については、`pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

オプションの説明については、前述の `install` コマンドを参照してください。

`pkg facet [-H] [facet_name ...]`

`pkg change-facet` コマンドを使用してこのイメージに明示的に設定されているすべてのファセットの現在の値を表示します。ファセットの詳細については、`pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

*facet\_name*

このイメージに設定されている、指定されたファセットのみの現在値を表示します。

-H

一覧からヘッダーを省略します。

```
pkg change-facet [-nvq] [-C n] [-g path_or_uri ...] [--accept] [--licenses]
[--no-be-activate] [--no-backup-be | --require-backup-be] [--backup-be-name name]
[--deny-new-be | --require-new-be] [--be-name name] facet_name=[True|False|None] ...
```

現在のイメージに設定され、指定されたファセットの値を変更します。

ファセットは `True` または `False` に設定できます。ファセットを `None` に設定すると、`True` のデフォルト値がそのファセットに適用されるため、ファセットに依存するすべてのアクションがインストールされます。アクションについては、`pkg(5)` のマニュアルページのアクションに関する項目を参照してください。

ファセットの値を変更すると、パッケージの内容が削除、更新、またはインストールされることがあります。ファセットの値を変更すると、新しいイメージ構成を満たすために、パッケージ全体がインストール、更新、または削除されることもあります。ファセットの詳細については、`pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

オプションの説明については、前述の `install` コマンドを参照してください。

```
pkg avoid [pkg_fmri_pattern ...]
```

回避対象の各パッケージを、そのパッケージにグループ依存関係を持つパッケージとともに表示します。

回避リストに登録されているパッケージは、要求された依存関係を満たすために必要であればインストールされます。その依存関係が削除された場合、パッケージはアンインストールされます。

```
pkg_fmri_pattern
```

指定されたパターンに現在一致するパッケージ名を回避リストに登録することにより、それらがグループ依存関係のターゲットである場合にそれらを回避します。現在インストールされていないパッケージのみを回避できます。パッケージが現在グループ依存関係のターゲットである場合、パッケージをアンインストールするとそのパッケージは回避リストに登録されます。

```
pkg unavoid [pkg_fmri_pattern ...]
```

回避対象のパッケージのリストを表示します。

```
pkg_fmri_pattern
```

指定されたパッケージを回避リストから削除します。回避リストに登録されており、インストール済みパッケージのグループ依存関係に一致するパッケージは、このサブコマンドを使用して削除できません。グループ依存性に一致するパッケージを回避リストから削除するには、パッケージをインストールします。

```
pkg freeze [-n] [-c reason] [pkg_fmri_pattern] ...
```

現在凍結されているパッケージについての情報 (パッケージ名、バージョン、パッケージがいつ凍結されたか、パッケージの凍結に関連付けられた理由があればその理由) が表示されます。

パッケージを凍結しても、そのパッケージを削除できなくなるわけではありません。パッケージが削除される場合に警告は表示されません。

#### *pkg\_fmri\_pattern*

指定されたパッケージを指定されたバージョンに凍結します。バージョンを指定しない場合、パッケージがインストールされている必要があります、そのインストール済みバージョンで凍結されます。すでに凍結されているパッケージを凍結すると、新しく指定されたバージョンによって凍結バージョンが置き換えられます。

凍結されているパッケージをインストールまたは更新するときは、凍結された時点のバージョンと一致するバージョンである必要があります。たとえば、パッケージが 1.2 で凍結された場合、1.2.1、1.2.9、1.2.0.0.1 などのバージョンに更新することはできます。そのパッケージは 1.3 または 1.1 で終了することはできません。 *pkg\_fmri\_pattern* で提供されるパブリッシャーは、一致するパッケージを検索するために使用されます。ただし、パブリッシャー情報は凍結の一環として記録されません。パッケージは発行元ではなくバージョンのみに関して凍結されます。

#### *-c reason*

凍結されるパッケージとともに理由を記録します。凍結が原因でインストールまたは更新に失敗する場合、その理由が示されます。

#### *-n*

凍結操作を試しに実行し、凍結されるパッケージの一覧を表示しますが、実際にはどのパッケージも凍結しません。

#### *pkg unfreeze [-n] [pkg\_name\_pattern] ...*

現在凍結されているパッケージについての情報 (パッケージ名、バージョン、パッケージがいつ凍結されたか、パッケージの凍結に関連付けられた理由があればその理由) が表示されます。

#### *pkg\_fmri\_pattern*

凍結によって適用される制約を、指定されたパッケージから削除します。バージョンを提供しても無視されます。

#### *-n*

凍結解除操作を試しに実行し、凍結解除されるパッケージの一覧を表示しますが、実際にはどのパッケージも凍結解除しません。

#### *pkg property [-H] [propname ...]*

すべてのイメージプロパティの名前と値を表示します。イメージプロパティの説明については、下の「イメージプロパティ」を参照してください。

#### *propname*

指定されたプロパティのみの名前と値を表示します。

#### *-H*

一覧からヘッダーを省略します。

**pkg set-property *propname propvalue***

既存のイメージプロパティーを更新するか、または新しいイメージプロパティーを追加します。

**pkg add-property-value *propname propvalue***

既存のイメージプロパティーに値を追加するか、または新しいイメージプロパティーを追加します。

**pkg remove-property-value *propname propvalue***

既存のイメージプロパティーから値を削除します。

**pkg unset-property *propname ...***

既存のイメージプロパティーを削除します。

**pkg publisher [-HPn] [-F *format*] [*publisher ...*]**

すべての発行元、それらの起点 URI、およびミラーの一覧を、検索の優先順に従って表示します。

*publisher*

指定された発行元のみの詳細な構成を表示します。

**-H**

一覧からヘッダーを省略します。

**-P**

発行元検索順の先頭の発行元のみを表示します。

**-n**

有効な発行元のみを表示します。

**-F**

代替出力形式を指定します。現時点では tsv (タブ区切り値) のみが有効です。

**pkg set-publisher [-Ped] [-k *ssl\_key*] [-c *ssl\_cert*] [-g *origin\_to\_add* | - --add-origin *origin\_to\_add ...*] [-G *origin\_to\_remove* | - --remove-origin *origin\_to\_remove ...*] [-m *mirror\_to\_add* | - --add-mirror *mirror\_to\_add ...*] [-M *mirror\_to\_remove* | - --remove-mirror *mirror\_to\_remove ...*] [--enable] [--disable] [--no-refresh] [--reset-uuid] [--non-sticky] [--sticky] [--search-after *publisher*] [--search-before *publisher*] [--search-first] [--approve-ca-cert *path\_to\_CA*] [--revoke-ca-cert *hash\_of\_CA\_to\_remove*] [--unset-ca-cert *hash\_of\_CA\_to\_remove*] [--set-property *name\_of\_property=value*] [--add-property-value *name\_of\_property=value\_to\_add*] [--remove-property-value *name\_of\_property=value\_to\_remove*] [--unset-property *name\_of\_property\_to\_delete*] [--proxy *proxy\_to\_use*] *publisher***

既存の発行元を更新するか、発行元を追加します。検索順に影響するオプションを指定しない場合、新しいパブリッシャーは検索順の末尾に付加され、最後に検索されます。

**-P**

**--search-first**

指定された発行元を、検索順の先頭に設定します。新しいパッケージをインストールするとき、このパブリッシャーが最初に検索されます。インストール済みパッケージの更新は、そのパブリッシャーが *sticky* であるかぎり、そのパッケージを最初に提供した同じパブリッシャーから取得されます。

**--non-sticky**

この発行元よりも上位にランクされる発行元が、この発行元から最初にインストールされたパッケージに更新を提供できます。

**--sticky**

この発行元からインストールされたパッケージへの更新も、この発行元から取得する必要があります。これはデフォルトの動作です。

**--search-before *publisher***

追加または変更される発行元が、このオプションで指定された発行元の前に検索されるように、発行元検索順序を変えます。

**--search-after *publisher***

追加または変更される発行元が、このオプションで指定された発行元のあとに検索されるように、発行元検索順序を変えます。

**--approve-ca-cert *path\_to\_CA***

指定された証明書を信頼できる CA 証明書として追加します。ユーザーが承認した CA 証明書の PEM 表現のハッシュは、`pkg publisher` コマンドの詳細出力に一覧表示されます。

**--revoke-ca-cert *hash\_of\_CA\_to\_remove***

指定された PEM 表現のハッシュを持つ証明書を失効済みとして扱います。ユーザーが失効させた CA 証明書のハッシュは、`pkg publisher` コマンドの詳細出力に一覧表示されます。

**--unset-ca-cert *hash\_of\_CA\_to\_remove***

指定されたハッシュを持つ証明書を、承認済み証明書のリストおよび失効済み証明書のリストから削除します。

**--set-property *name\_of\_property=value***

既存の発行元プロパティーを更新するか、または新しい発行元プロパティーを追加します。

**--add-property-value *name\_of\_property=value\_to\_add***

既存の発行元プロパティーに値を追加するか、または新しい発行元プロパティーを追加します。

**--remove-property-value *name\_of\_property=value\_to\_remove***

既存の発行元プロパティーから値を削除します。

**--unset-property *name\_of\_property\_to\_delete***

既存の発行元プロパティーを削除します。



`-k ssl_key`  
クライアント SSL 鍵を指定します。

`-c ssl_cert`  
クライアント SSL 証明書を指定します。

`-g origin_to_add`  
`--add-origin origin_to_add`  
指定された URI またはパスを、特定の発行元の起点として追加します。これはパッケージのリポジトリまたはアーカイブの場所になしてください。

`-G origin_to_remove`  
`--remove-origin origin_to_remove`  
指定された発行元の起点のリストから URI またはパスを削除します。特殊値 \* を使用して、すべての起点を削除することができます。

`--no-refresh`  
入手可能パッケージやその他のメタデータの最新リストを取得するために、イメージの発行元用のリポジトリへのアクセスを試みません。

`--reset-uuid`  
このイメージをその発行元に対して識別する新しい一意識別子を選択します。

`-m mirror_to_add`  
`--add-mirror mirror_to_add`  
URI を指定された発行元のミラーとして追加します。

`-M mirror_to_remove`  
`--remove-mirror mirror_to_remove`  
指定された発行元のミラーのリストから URI を削除します。特殊値 \* を使用して、すべてのミラーを削除することができます。

`-e`  
`--enable`  
発行元を有効にします。

`-d`  
`--disable`  
発行元を無効にします。無効にされたパブリッシャーは、パッケージリストの生成時に、または特定のパッケージ操作 (インストール、アンインストール、および更新) で使用されません。ただし、無効なパブリッシャーのプロパティを設定または表示することはできます。発行元が 1 つだけの場合は、無効にすることはできません。

`--proxy proxy_to_use`  
指定した Web プロキシ URI を使用して、指定した起点 (-g) またはミラー (-m) のコンテンツを取得します。プロキシ値は、発行元の構成の一部として格納されます。実行時に、`$http_proxy` または関連する環境変数は、このプロキシ



設定をオーバーライドします。受け入れられる環境変数名のリストについては、curl(1) のマニュアルページを参照してください。

```
pkg set-publisher -p repo_uri [-Ped] [-k ssl_key] [-c ssl_cert] [- -non-sticky] [--sticky]
[--search-after publisher] [--search-before publisher] [--search-first]
[--approve-ca-cert path_to_CA] [--revoke-ca-cert hash_of_CA_to_remove]
[--unset-ca-cert hash_of_CA_to_remove] [--set-property name_of_property=value]
[--add-property-value name_of_property=value_to_add]
[--remove-property-value name_of_property=value_to_remove] [--unset-property
name_of_property_to_delete] [--proxy proxy_to_use] [publisher]
repo_uri リポジトリ URI から発行元構成情報を取得します。
```

発行元オペランドがこの set-publisher サブコマンドに指定されている場合、その発行元のみが追加または更新されます。発行元を指定しない場合、*repo\_uri* 内のすべての発行元が必要に応じて追加または更新されます。

オプションの説明については、前述の set-publisher コマンドを参照してください。-p と同時に使用する場合、-P、--search-first、--search-before、および --search-after オプションは追加された発行元のみにも適用され、更新された発行元には適用されません。

-p オプション

は、-g、--add-origin、-G、--remove-origin、-m、--add-mirror、-M、--remove-mirror、または --reset-uuid オプションと組み合わせることはできません。

```
pkg unset-publisher publisher ...
```

指定された発行元に関連付けられた構成を削除します。

```
pkg history [-Hn] [-t [time | time-time],...] [-o column,...] [-n number]
```

該当するイメージのコマンド履歴を表示します。

-H

一覧からヘッダーを省略します。

-t *time*

-t *time-time*

%Y-%m-%dT%H:%M:%S 形式のタイムスタンプのコンマ区切りリストでログレコードを表示します (strftime(3C) を参照)。日時の範囲を指定するには、開始と終了のタイムスタンプの間にハイフン (-) を使用します。キーワード now は、現在の日時の別名として使用できます。指定されたタイムスタンプに、重複したタイムスタンプまたは重複する日付範囲が含まれる場合、重複した各履歴イベントの 1 つのインスタンスのみが出力されます。

-l

長い形式でログレコードを表示します。これには標準形式に加えて、コマンドの結果、コマンドが完了した日時、使用されたクライアントのバージョンと名前、操作を実行したユーザーの名前、およびコマンドの実行中に発生したすべてのエラーが含まれます。

- N  
リリースノートのテキストを表示します。
- n *number*  
最新のものから順に指定された数のエントリのみを表示します。
- o *column*  
列名を指定するコンマ区切りリストを使用して出力を表示します。有効な列名は次のとおりです。
  - be  
この操作が開始されたブート環境の名前。
  - be\_uuid  
この操作が開始されたブート環境の UUID。
  - client  
クライアントの名前。
  - client\_ver  
クライアントのバージョン。
  - command  
この操作のために使用されたコマンド行。
  - finish  
この操作が完了した日時。
  - id  
この操作を開始したユーザー ID。
  - new\_be  
この操作によって作成された新しいブート環境。
  - new\_be\_uuid  
この操作によって作成された新しいブート環境の UUID。
  - operation  
操作の名前。
  - outcome  
この操作の結果のサマリー。
  - reason  
この操作の結果に関する追加情報。
  - release\_note  
この操作によってリリースノートが生成されたかどうかを示します。
  - snapshot  
この操作中に作成されたスナップショット。これは、操作が正常に完了したあとにスナップショットが自動削除されなかった場合にのみ記録されます。

**start**

この操作が開始した日時。

**time**

この操作の実行にかかった合計時間。1 秒未満の操作については 0:00:00 と表示されます。

**user**

この操作を開始したユーザー名。

**command** または **reason** 列を指定する場合、出力フィールドの区切りを維持するためには、それらの列が **-o** リストの最終項目である必要があります。同じ **history** コマンドでこれら 2 つの列を表示することはできません。

ブート環境がシステムに存在しなくなった場合、**be** または **new\_be** の値のあとにアスタリスク (\*) が表示されます。

**be** および **new\_be** の値は、**be\_uuid** または **new\_be\_uuid** フィールドを使用して現在のブート環境名を検索することによって取得されます。その後、ブート環境の名前が変更されたあとにその環境が削除された場合、**be** および **new\_be** に表示される値は、**pkg** 操作の時点で記録された値です。

**pkg purge-history**

既存の履歴情報をすべて削除します。

**pkg rebuild-index**

**pkg search** によって使用されるインデックスを再構築します。これは復旧操作であり、一般的に使用することは想定されていません。

**pkg update-format**

イメージの形式を現在のバージョンに更新します。この操作が完了したあとは、古いバージョンの **pkg(5)** システムと組み合わせてイメージを使用することはできなくなります。

**pkg version**

**pkg** のバージョンを識別する一意な文字列を表示します。この文字列は、バージョン間で何らかの方法で比較可能であることは保証されていません。

**pkg help**

使用方法に関するメッセージを表示します。

**pkg image-create** [-FPUz] [-f|--force] [--full|--partial|--user] [--zone] [-k *ssl\_key*] [-c *ssl\_cert*] [--no-refresh] [--variant *variant\_name=value* ...] [-g *path\_or\_uri*] [--origin *path\_or\_uri* ...] [-m *uri*] [--mirror *uri* ...] [--set-property *name\_of\_property=value*] [--facet *facet\_name*=(True|False) ...] [(-p|--publisher) [*name*=] *repo\_uri*] *dir*  
*dir* によって指定された場所に、パッケージ操作に適したイメージを作成します。デフォルトのイメージタイプはユーザーであり、**-U** (**--user**) オプションによって指定されます。イメージタイプはフルイメージ (**--F** または **--full**)、または指定された *dir* パスを包含するフルイメージにリンクされた部分イメージ (**-P**

または `--partial`) に設定できます。 `-g` または `--origin` を使用して追加の起点を指定できます。 `-m` または `--mirror` を使用して追加のミラーを指定できます。

パッケージリポジトリの URI は、 `-p` または `--publisher` オプションを使用して提供する必要があります。パブリッシャーの名前も提供した場合、イメージの作成時にそのパブリッシャーのみが追加されます。パブリッシャーの名前を提供しない場合、指定されたりポジトリによって認識されているすべてのパブリッシャーがイメージに追加されます。このパブリッシャーに関連付けられたカタログは、初期作成操作に続いて取得が試みられます。

クライアント SSL 認証を使用するパブリッシャーの場合、クライアント鍵およびクライアント証明書は `-c` および `-k` オプションを通して登録できます。この鍵と証明書は、イメージ作成中に追加されるすべてのパブリッシャーのために使用されます。

イメージが非大域ゾーンコンテキストの内部で実行される予定の場合、 `-z` (`--zone`) オプションを使用して適切なバリエーションを設定できます。

`-f`

`--force`

既存のイメージ上にイメージを強制的に作成します。このオプションは慎重に使用してください。

`--no-refresh`

入手可能パッケージやその他のメタデータの最新リストを取得するために、イメージの発行元用のリポジトリへのアクセスを試みません。

`--variant variant_name= value`

指定されたバリエーションを指定された値に設定します。バリエーションの詳細については、 `pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

`--facet facet_name=( True|False)`

指定されたファセットを指定された値に設定します。ファセットの詳細については、 `pkg(5)` のマニュアルページのファセットとバリエーションに関する項目を参照してください。

`--set-property name_of_property= value`

指定されたイメージプロパティを指定された値に設定します。イメージプロパティの説明については、下の「イメージプロパティ」を参照してください。

## イメージプロパティ

次のプロパティはイメージの特性を定義します。これらのプロパティは、イメージの目的、内容、および動作に関する情報を格納します。イメージ内のこれらのプロパティの現在の値を表示するには、 `pkg property` コマンドを使用します。これらのプロパティの値を変更するには、 `pkg set-property` コマンドおよび `pkg unset-property` コマンドを使用します。

**be-policy**

(文字列) パッケージ操作中にいつブート環境が作成されるかを指定します。次の値が許可されます。

**default**

デフォルトブート環境作成ポリシー `create-backup` を適用します。

**always-new**

次のブート時にアクティブに設定されている新しいブート環境でパッケージ操作を実行するため、すべてのパッケージ操作に対してリブートを必要とします。明示的にリクエストされないかぎり、バックアップブート環境は作成されません。

このポリシーはもっとも安全ですが、リブートしないとパッケージを追加できないため、ほとんどのサイトの要求よりも厳格です。

**create-backup**

リブートを必要とするパッケージ操作で、新しいブート環境が作成され、次のブート時にアクティブに設定されます。パッケージが変更されるか、カーネルに影響を及ぼす可能性のある内容がインストールされて、ライブブート環境がこの操作の影響を受ける場合、バックアップブート環境は作成されますが、アクティブには設定されません。バックアップブート環境を明示的にリクエストすることもできます。

このポリシーは、新しくインストールされたソフトウェアによりシステムが不安定になっている場合にのみ潜在的に危険です。この可能性はありますが、比較的まれです。

**when-required**

リブートを必要とするパッケージ操作で、新しいブート環境が作成され、次のブート時にアクティブに設定されます。明示的にリクエストされないかぎり、バックアップブート環境は作成されません。

ライブブート環境へのパッケージ変更によりそれ以上の変更が不可能になる場合、フォールバック可能な最近のブート環境が存在しない可能性があるため、このポリシーには最大の危険が伴います。

**ca-path**

(文字列) SSL 操作用の CA 証明書が格納されたディレクトリを指すパス名。このディレクトリの形式は、ベースとなる SSL 実装に固有です。信頼できる CA 証明書のために別の場所を使用するには、別のディレクトリを指すようにこの値を変更します。CA ディレクトリの要件については、`SSL_CTX_load_verify_locations(3openssl)` の `CApath` に関する項目を参照してください。

デフォルト値: `/etc/openssl/certs`

**check-certificate-revocation**

(ブール型) True に設定されている場合、パッケージクライアントは、署名検証のために使用される証明書の CRL 配布ポイントへのアクセスを試み、発行時よりもあとに証明書が失効していないかどうかを調べます。

デフォルト値: False

**flush-content-cache-on-success**

(ブール型) True に設定されている場合、パッケージクライアントは、インストールまたは更新操作の完了時にその内容キャッシュ内のファイルを削除します。更新操作の場合、内容はソースブート環境からのみ削除されます。出力先ブート環境でパッケージ操作が次に発生したとき、このオプションが変更されていなければ、パッケージクライアントはその内容キャッシュをフラッシュします。

このプロパティーを使用して、ディスク容量の限られたシステムで内容キャッシュを小さく保つことができます。このプロパティーを使用すると、操作が完了するまでの時間が長くなる可能性があります。

デフォルト値: True

**mirror-discovery**

(ブール型) このプロパティーは、mDNS および DNS-SD を使用してリンクローカル内容ミラーを検出するようにクライアントに命令します。このプロパティーを True に設定すると、クライアントはミラーを動的に検出し、そのミラーからパッケージ内容のダウンロードを試みます。mDNS を介してその内容を通知するミラーの実行方法については、pkg.depotd(1M) のマニュアルページを参照してください。

デフォルト値: False

**send-uuid**

(ブール型) ネットワーク操作の実行時にイメージの汎用一意識別子 (UUID) を送信します。ユーザーはこのオプションを無効にできますが、一部のネットワークリポジトリは UUID を供給しないクライアントとのやり取りを拒否する場合があります。

デフォルト値: True

**signature-policy**

(文字列) イメージ内のパッケージのインストール、更新、修正、または検証時にマニフェストに対してどのチェックが実行されるかを決定します。パッケージに適用される最終的なポリシーは、イメージポリシーと発行元ポリシーの組み合わせに依存します。この組み合わせの厳格さは、少なくとも、この2つのポリシーが個別に適用された場合の厳格な方と同じです。デフォルトでは、パッケージクライアントは証明書が失効済みかどうかをチェックしません。そのようなチェック (クライアントから外部 Web サイトへのアクセスが必要

な場合がある)を有効にするには、`check-certificate-revocation` イメージプロパティを `True` に設定します。次の値が許可されます。

#### `ignore`

すべてのマニフェストの署名を無視します。

#### `verify`

署名が含まれているすべてのマニフェストが有効に署名されていることを確認しますが、インストール済みパッケージがすべて署名されている必要はありません。これがデフォルト値です。

#### `require-signatures`

新しくインストールされたすべてのパッケージに、有効な署名が少なくとも 1 つ含まれている必要があります。インストール済みパッケージに有効な署名が含まれていない場合は、`pkg fix` および `pkg verify` コマンドでも警告が表示されます。

#### `require-names`

`require-signatures` と同じ要件に従いますが、`signature-required-names` プロパティで一覧表示される文字列が、署名の信頼のチェーンを検証するために使用される証明書の共通名としても表示される必要があります。

#### `signature-required-names`

(文字列のリスト) パッケージの署名の検証中に、証明書の共通名として表示される必要のある名前の一覧です。

#### `trust-anchor-directory`

(文字列) イメージの信頼アンカーを格納するディレクトリのパス名です。このパスはイメージに対して相対的です。デフォルト値は `ignore` です。

#### `use-system-repo`

(ブール型) このプロパティではシステムリポジトリを、イメージおよびパブリッシャー構成のソースとして、および提供されたパブリッシャーと通信するためのプロキシとしてイメージで使用するべきかどうかを指定します。デフォルト値は `False` です。システムリポジトリについては、`pkg.sysrepo(1M)` のマニュアルページを参照してください。

### パブリッシャープロパティ

次のプロパティは、特定の発行元の署名ポリシーを定義します。同じ名前のイメージプロパティはイメージの署名ポリシーを定義します。特定の発行元のこれらのプロパティの現在の値を表示するには、`pkg publisher publisher_name` コマンドを使用します。これらの発行元の署名ポリシープロパティの値を変更するには、`pkg set-publisher` コマンドの `--set-property` オプションと `--unset-property` オプションを使用します。

#### `signature-policy`

(文字列) このプロパティの機能は、特定のパブリッシャーからのパッケージのみに適用されることを除いて、同じ名前のイメージプロパティと同じです。



**signature-required-names**

(文字列のリスト) このプロパティの機能は、特定のパブリッシャーからのパッケージのみに適用されることを除いて、同じ名前のイメージプロパティと同じです。

**使用例**

例1 パブリッシャーを構成してイメージを作成する

パブリッシャーを `example.com` として新しいフルイメージを作成し、`/aux0/example_root` に格納します。

```
$ pkg image-create -F -p example.com=http://pkg.example.com:10000 \
/aux0/example_root
```

例2 追加の起点とミラーを指定してイメージを作成する

パブリッシャーを `example.com` として新しいフルイメージを作成し、1つのミラーと2つの起点を追加し、`/aux0/example_root` に格納します。

```
$ pkg image-create -F -p example.com=http://pkg.example.com:10000 \
-g http://alternatel.example.com:10000/ \
-g http://alternate2.example.com:10000/ \
-m http://mirror.example.com:10000/ \
/aux0/example_root
```

例3 パブリッシャーを構成せずにイメージを作成する

パブリッシャーを構成せずに、新しいフルイメージを `/aux0/example_root` に作成します。

```
$ pkg image-create -F /aux0/example_root
```

例4 パッケージのインストール

`widget` パッケージの最新バージョンを現在のイメージにインストールします。

```
$ pkg install application/widget
```

例5 パッケージの指定された内容を一覧表示する

`system/file-system/zfs` パッケージの内容を一覧表示します。アクション名、ファイルのモード(定義されている場合)、サイズ(定義されている場合)、パス、およびターゲット(リンクの場合)を表示します。すべてのアクションで利用可能な `action.name` 属性を指定するとすべてのアクションの行が表示されますが、ここでは望ましくないため、`dir`、`file`、`link`、および `hardlink` の各タイプにアクションを限定します。

```
$ pkg contents -t dir,file,link,hardlink \
-o action.name,mode,pkg.size,path,target system/file-system/zfs
ACTION.NAME MODE PKG.SIZE PATH TARGET
dir 0755 etc
dir 0755 etc/fs
```



例5 パッケージの指定された内容を一覧表示する (続き)

```

dir 0755 etc/fs/zfs
link etc/fs/zfs/mount ../../usr/sbin/zfs
link etc/fs/zfs/umount ../../usr/sbin/zfs
dir 0755 etc/zfs
dir 0755 kernel
dir 0755 kernel/drv
dir 0755 kernel/drv/amd64
file 0755 1706744 kernel/drv/amd64/zfs
file 0644 980 kernel/drv/zfs.conf
dir 0755 kernel/fs
dir 0755 kernel/fs/amd64
hardlink kernel/fs/amd64/zfs ../../kernel/drv/amd64/zfs
...

```

例6 2つのパッケージの指定された内容を一覧表示する

web/browser/firefox および mail/thunderbird の内容を一覧表示します。path 属性の末尾が .desktop または .png であるアクションのパッケージ名属性およびパス属性のみに表示を限定します。

```

$ pkg contents -o pkg.name,path -a path=*.desktop \
-a path=*.png web/browser/firefox mail/thunderbird
PKG.NAME PATH
web/browser/firefox usr/share/applications/firefox.desktop
mail/thunderbird usr/share/applications/thunderbird.desktop
web/browser/firefox usr/share/pixmaps/firefox-icon.png
mail/thunderbird usr/share/pixmaps/thunderbird-icon.png
...

```

例7 パッケージを検索する

パッケージデータベースからトークン bge を検索します。

```

$ pkg search bge
INDEX ACTION VALUE PACKAGE
driver_name driver bge pkg:/driver/network/ethernet/bge@0.5
basename file kernel/drv/sparcv9/bge pkg:/driver/network/ethernet/bge@0.5
basename file kernel/drv/amd64/bge pkg:/driver/network/ethernet/bge@0.5
basename file platform/sun4v/kernel/drv/sparcv9/bge pkg:/system/kernel/platform@0.5.11-0
pkg.fmri set solaris/driver/network/bge pkg:/driver/network/bge@0.5.11-0.173
pkg.fmri set solaris/driver/network/ethernet/bge pkg:/driver/network/ethernet/bge@0.5

```

このトークンはパッケージ driver/network/bge 内に、/kernel/drv/arch/bge を表すファイルアクションのベース名として、およびドライバ名として存在します。

## 例8 ファイルの検索

先頭のスラッシュ文字を含むファイルのフルパス名を指定して、ファイルを提供するパッケージを検索します。

```
$ pkg search -o path,pkg.name -l /usr/bin/vim
PATH PKG.NAME
usr/bin/vim editor/vim/vim-core
```

*action\_name* に *file* を指定し、*index* に *path* または *basename* を指定し、*token* に完全または部分的なファイル名を指定することにより、ファイルおよびそのファイルを提供するパッケージを検索します。

```
$ pkg search -o path,pkg.name -l file:basename:vim
PATH PKG.NAME
usr/bin/vim editor/vim/vim-core
```

## 例9 ファイルとディレクトリの検索

*index* に *path* または *basename* を、*token* に完全または部分的なファイル名を指定することにより、ファイルとディレクトリ、およびそれらを提供するパッケージを検索します。使用するシェルによっては、ワイルドカードをエスケープする必要があります。

```
$ pkg search -o path,pkg.name -l path:*/vim
PATH PKG.NAME
usr/bin/vim editor/vim/vim-core
usr/share/vim editor/vim
usr/share/vim editor/vim/vim-core
$ pkg search -o path,pkg.name -l basename:vim
PATH PKG.NAME
usr/share/vim editor/vim
usr/share/vim editor/vim/vim-core
usr/bin/vim editor/vim/vim-core
```

## 例10 どのパッケージがどのSMFサービスを提供するかを表示する

構造化された検索内で *index* に値 `org.opensolaris.smf.fmri` を指定し、*token* に検索対象のサービスの名前を指定することにより、特定のSMFサービスを提供するのはどのパッケージかを表示します。値 `org.opensolaris.smf.fmri` は、`set` アクションの属性の名前です。サービスの名前では、「:」を確実にエスケープしてください。

たとえば、使用可能なHTTPサーバーを表示するには、*token* に値 `svc:/network/http` を指定します。

```
$ pkg search 'org.opensolaris.smf.fmri:svc:/network/http*'
INDEX ACTION VALUE PACKAGE
org.opensolaris.smf.fmri set svc:/network/http pkg:/web/server/lighttpd-14@1.4.23-0.1
org.opensolaris.smf.fmri set svc:/network/http pkg:/web/proxy/privoxy@3.0.17-0.175.0.
org.opensolaris.smf.fmri set svc:/network/http pkg:/web/proxy/squid@3.1.18-0.175.0.0.
```

例10 どのパッケージがどのSMFサービスを提供するかを表示する (続き)

```
org.opensolaris.smf.fmri set svc:/network/http pkg:/web/java-servlet/tomcat@6.0
org.opensolaris.smf.fmri set svc:/network/http pkg:/web/server/apache-22@2.2.22
org.opensolaris.smf.fmri set svc:/network/http:apache22 pkg:/web/server/apache-22@2.2.22
org.opensolaris.smf.fmri set svc:/network/http:lighttpd14 pkg:/web/server/lighttpd-14@1.4.
org.opensolaris.smf.fmri set svc:/network/http:privoxy pkg:/web/proxy/privoxy@3.0.17-0.
org.opensolaris.smf.fmri set svc:/network/http:squid pkg:/web/proxy/squid@3.1.18-0.17
org.opensolaris.smf.fmri set svc:/network/http:tomcat6 pkg:/web/java-servlet/tomcat@6.0
```

例11 指定されたパッケージに依存するパッケージを検索する  
package/pkg に依存するインストール済みパッケージを検索します。

```
$ pkg search -l depend::package/pkg
INDEX ACTION VALUE PACKAGE
incorporate depend package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/consolidation/ips/ips-incorporate
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/package/pkg/package-manager@0.5.11
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/system/library/install@0.5.11
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/package/pkg/update-manager@0.5.11
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/system/library/boot-management@0.5.11
require depend package/pkg pkg:/system/zones/brand/brand-solaris
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/install/distribution-construct@0.5.11
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/system/boot-environment-utilities@0.5.11
require depend pkg:/package/pkg@0.5.11-0.175.0.0.0.2.1 pkg:/package/pkg/system-repository@0.5.11
```

例12 依存関係を検索する  
インストール済みパッケージ内のすべての incorporate 依存関係を検索します。

```
$ pkg search -l depend:incorporate:
INDEX ACTION VALUE PACKAGE
incorporate depend pkg:/BRMbnx@0.5.11-0.175.0.0.0.2.1 pkg:/consolidation/osnet/osnet-incorporate
incorporate depend pkg:/BRMbnxe@0.5.11-0.175.0.0.0.2.1 pkg:/consolidation/osnet/osnet-incorporate
...
```

例13 発行元を追加  
新しいパブリッシャー example.com を追加し、リポジトリの場所を  
http://www.example.com/repo に設定します。

```
$ pkg set-publisher -g http://www.example.com/repo example.com
```

例14 鍵と証明書を指定してパブリッシャーを追加する  
新しいパブリッシャー example.com を追加し、セキュリティー保護されたりポジトリの場所を https://secure.example.com/repo に設定し、鍵および証明書をディレクトリ /root/creds に格納します。

```
$ pkg set-publisher -k /root/creds/example.key \
-c /root/creds/example.cert -g https://secure.example.com/repo \
example.com
```

例15 パブリッシャーを追加して自動構成する

自動設定を使用して新しいパブリッシャーを追加し、リポジトリの場所を `/export/repo` に設定します。

```
$ pkg set-publisher -p /export/repo
```

例16 パブリッシャーを追加して手動構成する

手動設定を使用して新しいパブリッシャー `example.com` を追加し、リポジトリの場所を `/export/repo/example.com` に設定します。

```
$ pkg set-publisher -g /export/repo example.com
```

例17 発行元を追加してプロキシを構成する

起点 `http://server/repo` とプロキシ `http://webcache:8080` を使用して新しい発行元 `mypub` を追加します。

```
$ pkg set-publisher -g http://server/repo \
--proxy http://webcache:8080 mypub
```

例18 すべての署名付きパッケージを検証する

すべての署名付きパッケージを検証するようにイメージを構成します。

```
$ pkg set-property signature-policy verify
```

例19 すべてのパッケージで署名を必須にする

すべてのパッケージで署名を必須とするように、また、いずれかの証明書で文字列 `example.com` が共通名として信頼チェーンに出現することを必須とするようにイメージを構成します。

```
$ pkg set-property signature-policy require-names example.com
```

例20 指定したパブリッシャーからのすべてのパッケージで署名を必須にする

パブリッシャー `example.com` からインストールされるすべてのパッケージで署名を必須とするようにイメージを構成します。

```
$ pkg set-publisher --set-property signature-policy=require-signatures \
example.com
```

例21 信頼チェーンで指定された文字列を必須にする

有効とみなされるために署名の信頼チェーンに存在する必要がある文字列 `foo` をイメージの共通名の一覧に追加します。

```
$ pkg add-property-value signature-require-names foo
```

例22 指定されたパブリッシャーの信頼チェーンから文字列を削除する

パブリッシャー example.com の署名を検証するために存在する必要がある文字列 foo を共通名の一覧から削除します。

```
$ pkg set-publisher --remove-property-value signature-require-names=foo \
example.com
```

例23 信頼できるCA証明書を追加する

/tmp/example\_file.pem に格納された証明書を、パブリッシャー example.com の信頼できるCA証明書として追加します。

```
$ pkg set-publisher --approve-ca-cert /tmp/example_file.pem \
example.com
```

例24 証明書を失効させる

a12345 というハッシュを持つ、パブリッシャー example.com 用の証明書を失効させ、example.com からのパッケージの署名をその証明書で検証しないようにします。

```
$ pkg set-publisher --revoke-ca-cert a12345 example.com
```

例25 証明書を抹消する

証明書 a12345 がユーザーによって追加または失効させられたことを pkg から抹消します。

```
$ pkg set-publisher --unset-ca-cert a12345 example.com
```

例26 パッケージをダウングレードする

インストール済みパッケージ foo@1.1 を古いバージョンにダウングレードします。

```
$ pkg update foo@1.0
```

例27 競合するパッケージインストールを切り替える

2つのパッケージが競合している場合に、どちらのパッケージがインストールされるかを切り替えます。パッケージ A がパッケージ B または C のどちらかに依存し、B および C は相互排他であるとします。A と B がインストールされる場合に、次のコマンドを使用して、A をアンインストールせずに B の代わりに C を使用するように切り替えます。

```
$ pkg install --reject B C
```

例28 パッケージアーカイブ内のパッケージを一覧表示する

パッケージアーカイブ内のすべてのパッケージのすべてのバージョンを一覧表示します。

```
$ pkg list -f -g /my/archive.p5p
```

例29 パッケージリポジトリ内のパッケージを一覧表示する

リポジトリ内のすべてのパッケージのすべてのバージョンを一覧表示します。

```
$ pkg list -f -g http://example.com:10000
```

例30 パッケージアーカイブ内のパッケージについての情報を表示する

パッケージアーカイブ内のパッケージの最新バージョンのパッケージ情報を表示します。パッケージが現在インストールされていなくてもかまいません。

```
$ pkg info -g /my/archive.p5p pkg_name
```

例31 パッケージアーカイブ内のパッケージの内容を表示する

パッケージアーカイブ内のパッケージの内容を表示します。パッケージは現在インストールされていません。

```
$ pkg contents -g /my/archive.p5p pkg_name
```

例32 パブリッシャーの起点とミラーをすべて削除する

パブリッシャーのすべての起点およびミラーを削除し、新しい起点を追加します。

```
$ pkg set-publisher -G '*' -M '*' -g http://example.com:10000 \
example.com
```

## 環境変数

### PKG\_IMAGE

パッケージ操作に使用するイメージが含まれるディレクトリ。-Rを指定した場合は無視されます。

### PKG\_CLIENT\_CONNECT\_TIMEOUT

トランスポート操作中に接続しようとするのを待機する秒数(試行ごと)。これが経過するとクライアントは操作を中止します。値0は無制限に待機することを意味します。

デフォルト値: 60

### PKG\_CLIENT\_LOWSPEED\_TIMEOUT

トランスポート操作中に `lowspeed` 制限(1024 バイト/秒)を下回ってられる秒数。これが経過するとクライアントは操作を中止します。値0は、操作を中止しないことを意味します。

デフォルト値: 30

### PKG\_CLIENT\_MAX\_CONSECUTIVE\_ERROR

一時的なトランスポートエラーの最大数。これを上回るとクライアントは操作を中止します。値0は、操作を中止しないことを意味します。

デフォルト値: 4

**PKG\_CLIENT\_MAX\_REDIRECT**

トランスポート操作中に許可される HTTP または HTTPS リダイレクトの最大数。これを上回ると接続が中止されます。値 0 は、操作を中止しないことを意味します。

デフォルト値: 5

**PKG\_CONCURRENCY**

同時に更新する子イメージの数。-c オプションが指定されている場合には、無視されます。

子イメージ (通常はゾーン) 内への再帰時に、最大 \$PKG\_CONCURRENCY 個の子イメージを同時に更新します。\$PKG\_CONCURRENCY が 0 または負数の場合は、すべての子イメージが同時に更新されます。

デフォルト値: 1

**PKG\_CLIENT\_MAX\_TIMEOUT**

ホストあたりのトランスポート試行の最大数。これを上回るとクライアントは操作を中止します。値 0 は、操作を中止しないことを意味します。

デフォルト値: 4

**http\_proxy、https\_proxy**

HTTP または HTTPS プロキシサーバー。

終了ステータス 次の終了ステータスが返されます。

- 0 コマンドが成功しました。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。
- 3 複数の操作が要求されましたが、それらの一部のみが成功しました。
- 4 変更が行われませんでした - 何もしません。
- 5 要求された操作はライブイメージでは実行できません。
- 6 インストールまたは更新中のパッケージのライセンスが受け入れられなかったため、要求された操作を完了できません。
- 7 イメージは現在別のプロセスによって使用されているため、変更できません。
- 99 予期しない例外が発生しました。

ファイル

pkg(5) イメージは、より大きなファイルシステム内の任意の場所に置くことができます。次のファイル説明で、トークン \$IMAGE\_ROOT は相対パスを区別するために使用されています。一般的なシステムインストールでは、\$IMAGE\_ROOT は / と等価です。

`$IMAGE_ROOT/var/pkg`  
完全または部分的なイメージのメタデータディレクトリ。

`$IMAGE_ROOT/.org.opensolaris,pkg`  
ユーザーイメージのメタデータディレクトリ。

特定のイメージのメタデータ内のファイルやディレクトリによっては、修復および復旧中に使用される情報が含まれている可能性があります。トークン `$IMAGE_META` は、メタデータが含まれる最上位ディレクトリを参照します。`$IMAGE_META` は通常、上に示した2つのパスのいずれかです。

`$IMAGE_META/lost+found`  
パッケージ操作中に移動された、競合するディレクトリおよびファイルの場所。削除されたディレクトリのパッケージ化されない内容の場所。

`$IMAGE_META/publisher`  
パブリッシャーごとに1つのディレクトリが含まれます。各ディレクトリにはパブリッシャー固有のメタデータが格納されます。

`$IMAGE_META` ディレクトリ階層内のほかのパスは非公開であり、変更される可能性があります。

属性 次の属性については、`attributes(5)` を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目 [pkgsend\(1\)](#), [pkg.depotd\(1M\)](#), [glob\(3C\)](#), [pkg\(5\)](#), [beadm\(1M\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>



名前 pkgdepend – Image Packaging System 依存関係アナライザ

形式 /usr/bin/pkgdepend [*options*] *command* [*cmd\_options*] [*operands*]

```
/usr/bin/pkgdepend generate [-IMm] -d dir [-d dir]
 [-D name=value] [-k path] manifest_file
```

```
/usr/bin/pkgdepend resolve [-moSv] [-d output_dir]
 [-s suffix] manifest_file ...
```

## 機能説明

pkgdepend コマンドは、パッケージの依存関係を生成および解決します。パッケージは、ほかのパッケージのファイルに依存することがあります。pkgdepend コマンドは通常、ファイル依存関係の生成と、ファイルからパッケージへの解決という2つのパスで使用されます。

**generate** サブコマンドはパッケージの内容を検査し、そのパッケージにどのような外部ファイルが必要かを判定します。

**resolve** サブコマンドは、**generate** ステップからファイルのリストを取得し、これらの依存ファイルを含むパッケージの名前を判定するためにパッケージの参照セットを検索します。依存ファイルが検索されるパッケージの参照セットは、パブリッシャーのシステムに現在インストールされているパッケージです。

依存関係情報のソースとして、提供されたファイルの次のいくつかのコンポーネントが使用されます。

**ELF** 提供されたファイル内の ELF ヘッダーが依存関係情報のために分析され、**-k** および **-D** オプションが指定されていると取得された情報が変更されます。ELF 依存関係についての詳細は、**ldd** および『[リンカーとライブラリ](#)』を参照してください。

**スクリプト** インタプリタを参照している **#!** 行を含むシェルスクリプトによって、そのインタプリタを提供しているパッケージへの依存関係が生成されます。

**Python** Python スクリプトはまず、スクリプトとして分析されます。さらに、そのスクリプトでインポートが宣言されていると、それらも依存関係情報のソースとして機能することがあります。

**ハードリンク** マニフェスト内のハードリンクによって、リンクターゲットを提供しているパッケージへの依存関係が生成されます。

**SMF** **require\_all** 依存関係を含む提供された SMF サービスマニフェストによって、これらの FMRI を提供する SMF マニフェストを提供しているパッケージへの依存関係が生成されます。

オプション サポートしているオプションは、次のとおりです。

**-?**

**--help**

使用方法に関するメッセージを表示します。

**-R *dir***

**--image-dir *dir***

*dir* をルートとするイメージを操作します。ディレクトリが指定されなかったか、または環境に基づいて決定される場合、デフォルトは `/` です。詳細は、「環境変数」のセクションを参照してください。

## サブコマンド

サポートされているサブコマンドは次のとおりです。

**pkgdepend generate [-IMm] -d *dir* [-d *dir*] [-D *name=value*] [-k *path*] *manifest\_file***  
*manifest\_file* で指定されたマニフェストのファイルへの依存関係を生成します。

**-I**

*manifest\_file* 内で満たされている依存関係を表示します。-I と pkgdepend resolve の結果を使用しないでください。

**-M**

分析できなかったファイルタイプのリストを表示します。

**-m**

元のマニフェストを繰り返してから、検出された依存関係をすべて追加します。

**-d *dir***

マニフェストのファイルを検索するディレクトリのリストに *dir* を追加します。

**-D *name=value***

ELF ファイル依存関係のための実行パス内のトークン *name* を展開する方法として *value* を追加します。

**-k *path***

カーネルモジュールを検索する実行パスのリストに *path* を追加します。-k オプションを使用すると、デフォルトのパス (`/kernel` と `/usr/kernel`) が削除されます。

-k オプションで指定されるような実行パスは、アクションまたはマニフェスト属性 `pkg.depend.runpath` を使用して、アクションごとまたはマニフェストごとにも指定できます。pkg.depend.runpath 属性の値は、使用するパスのコロンで区切られた文字列です。マニフェストまたはアクション内にいずれかの `pkg.depend.runpath` 属性を設定すると、-k オプションで指定されたパスがオーバーライドされます。

pkg.depend.runpath 属性値の 1 つのコンポーネントとして、特殊なトークン `$PKGDEPEND_RUNPATH` を使用すると、分析対象のファイルのための標準のシステム実行パスを含めることができます。

場合によっては、依存関係が自動的に生成されることを回避したいことがあります。たとえば、あるパッケージによって、一連のモジュールをインポートするサンプルの Python スクリプトが提供される場合、そのサンプルスクリプトに

よってインポートされるこれらのモジュールは、サンプルスクリプトを提供しているパッケージの依存関係ではありません。指定されたファイルに対する依存関係の生成を回避するには、アクションまたはマニフェスト属性 `pkg.depend.bypass-generate` を使用します。

`pkg.depend.bypass-generate` 値は、ファイル名に一致する Python の正規表現です。これらの正規表現は、ファイルパスの先頭と最後に暗黙的に固定されます。次の例で指定されている値は `this/that` に一致しますが、`something/this/that/the/other` には一致しません。

```
pkg.depend.bypass-generate=this/that
```

Python 正規表現の構文については、コマンド `pydoc re` を使用するか、または <http://docs.python.org/dev/howto/regex.html> にあるより完全なドキュメントを参照してください。

`pkgdepend generate` 入力マニフェストに SMF マニフェストファイルが含まれる場合、それらの SMF マニフェストファイルによって宣言されている SMF サービスまたはインスタンスが `pkgdepend` の出力に含まれます。これらの SMF サービスまたはインスタンスは、`org.opensolaris.smf.fmri` の名前で `set` アクションの形式で含まれます。

```
pkgdepend resolve [-moSv] [-d output_dir] [-s suffix] manifest_file ...
```

ファイルへの依存関係を、これらのファイルを提供するパッケージへの依存関係に変換します。依存関係は、まずコマンド行に指定されたマニフェストに基づいて解決され、次にシステムにインストールされているパッケージに基づいて解決されます。デフォルトでは、各マニフェストの依存関係は `manifest_file.res` という名前のファイルに格納されます。

**-m**

解決された依存関係を追加する前に、`generate` ステップによって生成された依存関係をすべて削除して、マニフェストを繰り返します。

**-o**

結果を標準出力に書き込みます。このオプションは、人による使用を目的にしています。この出力をファイルに追加すると、無効なマニフェストが生成されることがあります。マニフェスト処理のパイプラインでは、`-o` の代わりに `-d` または `-s` オプションを使用することを強くお勧めします。

**-d output\_dir**

指定された各マニフェストの解決された依存関係を `output_dir` 内の個別のファイルに書き込みます。デフォルトでは、各ファイルには、そのファイルに書き込まれた依存関係のソースだったマニフェストと同じベース名が付けられます。

**-s suffix**

出力ファイルごとに、解決された依存関係のソースだったファイルのベース名に `suffix` を追加します。`suffix` が `.suffix` でない場合、ピリオド (.) が `suffix` の前に付加されます。

-S

コマンド行で指定されたマニフェストに対してのみ解決し、システムにインストールされているマニフェストに対しては解決しません。

-v

追加のパッケージ依存関係デバッグ用のメタデータを含めます。

## 使用例

### 例1 依存関係を生成する

foo(このコンテンツディレクトリは ./bar/baz 内に存在する)に書き込まれているマニフェストの依存関係を生成し、その結果を foo.fdeps に格納します。

```
$ pkgdepend generate -d ./bar/baz foo > foo.fdeps
```

### 例2 依存関係を解決する

foo.fdeps と bar.fdeps における互いに対するファイル依存関係と、システムに現在インストールされているパッケージに対するファイル依存関係を解決します。

```
$ pkgdepend resolve foo.fdeps bar.fdeps
```

```
$ ls *.res
```

```
foo.fdeps.res bar.fdeps.res
```

### 例3 2つのマニフェストの依存関係を生成および解決する

2つのマニフェスト(foo と bar) へのファイル依存関係を生成し、すべての情報を元のマニフェスト内に保持します。次に、これらのファイル依存関係を解決し、結果のマニフェストを ./res 内に格納します。これらの結果のマニフェストは、pkgsend publish で使用できます。

```
$ pkgdepend generate -d /proto/foo -m foo > ./deps/foo
```

```
$ pkgdepend generate -d /proto/bar -m bar > ./deps/bar
```

```
$ pkgdepend resolve -m -d ./res ./deps/foo ./deps/bar
```

```
$ ls ./res
```

```
foo bar
```

### 例4 ELF ファイル依存関係のためのトークンに値を追加する

コンテンツディレクトリが / 内に存在する foo に書き込まれているマニフェストの依存関係を生成するときに、ELF ファイルの実行パス内のすべての PLATFORM トークンを sun4v と sun4u に置き換えます。

```
$ pkgdepend generate -d / -D 'PLATFORM=sun4v' -D 'PLATFORM=sun4u' foo
```

### 例5 カーネルモジュールディレクトリを指定する

コンテンツディレクトリが / 内に存在する foo に書き込まれているマニフェストの依存関係を生成するときに、カーネルモジュールを検索するディレクトリとして /kmod を指定します。

```
$ pkgdepend generate -d / -k /kmod foo
```

例6 依存関係の生成をバイパスする

指定された Python スクリプトの標準の Python 実行パスに `opt/python` を追加し、`opt/python/foo/file.py` として提供されるファイルの `test` という名前のすべての Python モジュールに対する依存関係の生成をバイパスします。

`usr/lib/python2.6/vendor-packages/xdg` で提供されたすべてのファイルに対する依存関係の生成を回避します。

```
$ cat manifest.py
set name=pkg.fmri value=pkg:/mypackage@1.0,1.0
set name=pkg.summary value="My test package"
dir path=opt mode=0755 group=sys owner=root
dir path=opt/python mode=0755 group=sys owner=root
dir path=opt/python/foo mode=0755 group=sys owner=root
file NOHASH path=opt/python/__init__.py mode=0644 group=sys owner=root
file NOHASH path=opt/python/foo/__init__.py mode=0644 group=sys owner=root
#
Add runpath and bypass-generate attributes:
#
file NOHASH path=opt/python/foo/file.py mode=0644 group=sys owner=root \
 pkg.depend.bypass-generate=.*test.py.* \
 pkg.depend.bypass-generate=.*testmodule.so \
 pkg.depend.bypass-generate=.*test.so \
 pkg.depend.bypass-generate=usr/lib/python2.6/vendor-packages/xdg/. * \
 pkg.depend.runpath=$PKGDEPEND_RUNPATH:/opt/python
```

```
$ pkgdepend generate -d proto manifest.py
```

環境変数

PKG\_IMAGE      パッケージ操作に使用するイメージを含むディレクトリを指定します。  
-Rが指定されている場合は、この値は無視されます。

終了ステータス

次の終了ステータスが返されます。

- 0      すべてが動作しました。
- 1      エラーが発生した。
- 2      無効なコマンド行オプションが指定された。
- 99     予期しない例外が発生しました。

属性

次の属性については、`attributes(5)` を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目

[pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgdiff - パッケージマニフェストの比較                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式    | <pre>/usr/bin/pkgdiff [-i attribute ...] [-o attribute]                   [-v name=value ...] file1 file2</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 機能説明  | <p>pkgdiff は 2 つのパッケージマニフェストを比較し、それらの違いを報告します。pkgdiff は、比較の前に、各マニフェストとアクションを一貫した順序にソートします。</p> <p>出力の形式は次のとおりです。</p> <p>+ <i>complete_action</i><br/>このアクションは <i>file2</i> に含まれていますが、<i>file1</i> には含まれていません。</p> <p>- <i>complete_action</i><br/>このアクションは <i>file1</i> に含まれていますが、<i>file2</i> には含まれていません。</p> <p><i>actionname keyvalue</i> [<i>variant values, if any</i>]</p> <p>- <i>attribute1=value1</i><br/>この <i>attribute,value</i> は <i>file1</i> に含まれていますが、<i>file2</i> には含まれていません。</p> <p>+ <i>attribute2=value2</i><br/>この <i>attribute,value</i> は <i>file2</i> に含まれていますが、<i>file1</i> には含まれていません。</p> <p>バリエントは異なるが、タイプとキー属性値が同じアクションは、比較の目的では別のアクションとして扱われます。そのため、属性を変更したアクションは属性変更としてではなく、完全な形式で表示されます。</p> |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>--help<br/>使用方法に関するメッセージを表示します。</p> <p>-i <i>attribute</i><br/>比較中に <i>attribute</i> が存在した場合は、無視します。-i <i>hash</i> により、ファイルのハッシュ値を無視できます。このオプションを -o オプションとともに使用することはできません。このオプションは繰り返すことができます。</p> <p>-o <i>attribute</i><br/><i>attribute</i> の違いのみを報告します。このオプションを -i オプションとともに使用することはできません。このオプションは、アクション時に <i>attribute</i> に影響を与えないアクション変更をすべて省略します。</p> <p>-v <i>name=value</i><br/>このバリエント値の違いのみを計算します。たとえば、arch=sparc の違いのみを計算します。このバリエントタグは、比較前にすべてのアクションについて削除されます。バリエントあたり 1 つの値のみを指定できます。このオプションは、異なるバリエントに対して繰り返すことができます。</p>                                                                                                                                                 |

終了ステータス 次の終了ステータスが返されます。

0 違いは見つかりませんでした。

1 違いが見つかりました。

>1 エラーが発生した。

99 予期しない例外が発生しました。

属性 次の属性については、attributes(5)を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目 [pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>



|         |                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pkgfmt - パッケージマニフェストの整形                                                                                                                                                                                                                                                                                                                                                                         |
| 形式      | /usr/bin/pkgfmt [-c -d -u] [ <i>package-manifest-file</i> ]                                                                                                                                                                                                                                                                                                                                     |
| 機能説明    | <p>-c または -d オプションが指定されない pkgfmt は、80 文字での行の折り返し、タイプによるアクションのソート、属性のソートなどを含む、一貫した方法でパッケージマニフェストを整形します。アクションに解析されない行(マクロ、コメント、変換など)は、ソートされた順序では表示されません。</p> <p>引数が指定されない場合、pkgfmt は stdin を EOF まで読み取ったあと、整形されたマニフェストを stdout に書き込みます。コマンド行で指定されたマニフェストはすべて、その場所で整形されます。</p> <p>-c オプションが指定された pkgfmt は、マニフェストが pkgfmt 形式で整形されているかどうかをチェックします。-d オプションを指定すると、ファイルが正しく整形されていない場合にその違いが表示されます。</p> |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <p>-?            使用方法に関するメッセージを表示します。</p> <p>--help       使用方法に関するメッセージを表示します。</p> <p>-c            マニフェストが pkgfmt 形式で整形されているかどうかをチェックします。</p> <p>-d            統合された形式で整形されたバージョンとのマニフェストの違いを表示します。</p> <p>-u            80 文字で行を折り返しません。このオプションは、パッケージマニフェストに従来のテキスト処理ツールを適用する場合に有効です。</p>                                                                     |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0            コマンドが成功しました。</p> <p>1            -c または -d オプションが指定され、1 つ以上のマニフェストが pkgfmt の正常な形式でないか、またはエラーが発生しました。</p> <p>2            無効なコマンド行オプションが指定された。</p> <p>99          予期しない例外が発生しました。</p>                                                                                                                                                                    |
| 属性      | 次の属性については、attributes(5) を参照してください。                                                                                                                                                                                                                                                                                                                                                              |

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目

[pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkginfo - ソフトウェアパッケージ情報の表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| 形式    | <pre>pkginfo [-q   -x   -l] [-p   -i] [-r] [-a arch]         [-v version] [-c category]... [pkginst]...  pkginfo -d device [-R root_path] [-q   -x   -l] [-a arch]         [-v version] [-c category]... [pkginst]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 機能説明  | <p>pkginfo は、システムにインストールされているソフトウェアパッケージの情報を表示します (最初の形式)。あるいは、特定のデバイスまたはディレクトリ上にあるソフトウェアパッケージの情報を表示します (2 番目の形式)。</p> <p>オプションを指定しないと、pkginfo は、プライマリカテゴリ、パッケージのインスタンス、および完全または部分的にインストールされているすべてのパッケージの名前を一覧表示します。これらの情報はパッケージごとに 1 行に表示されます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| オプション | <p>-d オプションと組み合わせて使用すると、-p オプションと -i オプションは無効になります。</p> <p>-q オプション、-x オプション、および -l オプションを同時に指定することはできません。</p> <p>-a arch<br/>パッケージのアーキテクチャを arch として指定します。</p> <p>-c category<br/>category に一致するパッケージを表示します。カテゴリは <a href="#">pkginfo(4)</a> ファイルの CATEGORY パラメータで定義されます。複数のカテゴリが設定されている場合、リスト内の 1 つのカテゴリとだけ一致する必要があります。照合に大文字と小文字の区別はありません。</p> <p>-d device<br/>ソフトウェアが存在するデバイス device を定義します。device はディレクトリの絶対パス名、または、テープ、着脱式ディスクなどの識別子のどちらでもかまいません。spool という特殊なトークンを使用すると、デフォルトのインストールスプールディレクトリ (/var/spool/pkg) を指定することができます。</p> <p>-i<br/>完全にインストールされているパッケージの情報だけを表示します。</p> <p>-l<br/>詳細な出力形式を指定します。つまり、指定されたパッケージについて利用できる情報をすべて表示します。</p> <p>-p<br/>部分的にインストールされているパッケージの情報だけを表示します。</p> <p>-q<br/>情報を表示しません。このオプションは、パッケージがインストールされているかどうかをプログラム内で検査するときに使用します。</p> |

- r  
再配置可能なパッケージのインストールのベースディレクトリを一覧表示します。
- R *root\_path*  
*root\_path* として使用するディレクトリのフルパス名を定義します。すべてのファイル(パッケージシステム情報ファイルを含む)は *root\_path* から始まるディレクトリツリーに再配置されます。
- v *version*  
パッケージのバージョンを *version* として指定します。バージョンは [pkginfo\(4\)](#) ファイルの `VERSION` パラメータで定義されます。互換性があるすべてのバージョンを要求するには、バージョン名の前にチルド(-)を付けます。バージョンの比較では、複数のスペースは1つのスペースに置き換えられます。
- x  
抽出されたパッケージ情報の一覧を示します。一覧には、パッケージの省略名、名前、アーキテクチャ(もしあれば)、およびバージョン(もしあれば)が含まれます。

オペランド

*pkginst*  
インスタンスによるパッケージの指定。インスタンスはパッケージの省略形または特定のインスタンス(たとえば、*inst.1*や*inst.2*)のどちらでもかまいません。パッケージのすべてのインスタンスを要求するには、*inst.\**を使用します。シェルによっては、アスタリスク文字(\*)が特殊な意味を持つことがあり、\*をエスケープしなければならない場合があります。Cシェルでは、「\*」は単一引用符(')で囲むか、またはバックスラッシュ(\)を前に付ける必要があります。

終了ステータス

- 0  
正常終了。
- >0  
エラーが発生した。

ファイル

*/var/spool/pkg*  
デフォルトのインストールスプールディレクトリ

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目

[pkgtrans\(1\)](#), [pkgadd\(1M\)](#), [pkgask\(1M\)](#), [pkgchk\(1M\)](#), [pkgrm\(1M\)](#), [pkginfo\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

『Application Packaging Developer’s Guide』

---

注意事項

パッケージコマンドは、[largefile\(5\)](#) を認識します。これらのコマンドは、2G バイトより大きなファイルを、より小さなファイルと同様に処理します。現在の実装では、[pkgadd\(1M\)](#)、[pkgtrans\(1\)](#)、およびその他のパッケージコマンドは最大 4G バイトのデータストリームを処理できます。

名前 pkglint – Image Packaging System パッケージ lint

形式 /usr/bin/pkglint [-c *cache\_dir*] [-r *repo\_uri*] [-p *regex*]  
[-f *config\_file*] [-b *build\_no*] [-v]  
[-l *lint\_uri*] | *manifest* ...  
  
/usr/bin/pkglint -L [-v]

機能説明 pkglint は、必要に応じて別のリポジトリを参照しながら、1 つ以上のパッケージマニフェストに対して一連のチェックを実行します。

pkglint は、パッケージ公開前の、パッケージ作成処理中に使用するべきです。pkglint は、マニフェストに対して、pkgsend または pkg.depotd の通常動作中に実行するには負荷が大きすぎる可能性のある徹底的なテストを実行します。pkglint のチェックには、重複したアクション、欠落した属性、および異常なファイルアクセス権に対するテストが含まれます。

lint のためのマニフェストは、コマンド行で、スペースで区切られたローカルファイルのリストとして渡すことができます。または、リポジトリからマニフェストを取得できます。

リポジトリからマニフェストを取得する場合、pkglint は最初の実行で、指定されたキャッシュディレクトリ内に pkg(5) ユーザーイメージを作成して設定します。-r オプションが指定されている場合は、参照リポジトリのために *cache\_dir/ref\_image* という名前のユーザーイメージが作成されます。-l オプションが指定されている場合は、lint リポジトリのために *cache\_dir/lint\_image* という名前のユーザーイメージが作成されます。これらのイメージに内容はインストールされません。これらのイメージは、リポジトリからマニフェストを取得するために pkglint でのみ使用されます。

pkglint の以降の呼び出しではキャッシュディレクトリを再利用できるため、-r または -l 引数はすべて省略できます。

pkglint では、キャッシュディレクトリ内のパブリッシャーを構成するためのサポートが制限されています。これらのイメージに対して、より複雑な発行元の構成を実行するには、pkg を使用します。

pkglint では、パッケージ作成者は、特定のマニフェストまたはアクションに対するチェックをバイパスできます。True に設定された属性 pkg.linted を含むマニフェストまたはアクションでは、そのマニフェストまたはアクションに対する lint 出力は生成されません。

pkglint チェックの名前の部分文字列を使用して、よりきめ細かな pkg.linted 設定を行うことができます。たとえば、pkg.linted.check.id が True に設定されていると、特定のマニフェストまたはアクションに対する check.id という名前を持つすべてのチェックがバイパスされます。

pkglint の動作は、pkglintrc ファイルを指定することによって構成できます。デフォルトでは、pkglint は、/usr/share/lib/pkg/pkglintrc および \$HOME/.pkglintrc 内の構成オプションを検索します。別の構成ファイルを指定するには、-f オプションを使用します。

lint の実行中に検出されたエラーまたは警告はすべて stderr に出力されます。

## オプション

サポートしているオプションは、次のとおりです。

-h

--help

使用方法に関するメッセージを表示します。

-b *build\_no*

lint および参照リポジトリからの lint 中に使用されるパッケージのリストを絞り込むために使用されるビルド番号を指定します。-b オプションが指定されていない場合は、最新バージョンのパッケージが使用されます。version.pattern 設定プロパティも参照してください。

-c *cache\_dir*

lint および参照リポジトリからのパッケージのメタデータをキャッシュするために使用されるローカルディレクトリを指定します。

-l *lint\_uri*

lint リポジトリの場所を表す URI を指定します。HTTP ベースとファイルシステムベースの両方の公開がサポートされています。-l を指定する場合は、-c も指定する必要があります。

-L

既知の lint チェックと除外された lint チェックを一覧表示してから終了します。各チェックの短縮名と説明を表示します。-v フラグと組み合わせる場合は、説明の代わりに、このチェックを実装するメソッドを表示します。

-f *config\_file*

*config\_file* 構成ファイルを使用して pkglint セッションを構成します。

-p *regexp*

lint リポジトリからチェックされるパッケージのリストを絞り込むために使用される正規表現を指定します。参照リポジトリからはすべてのマニフェストがロードされ(-b が指定されている場合は、その値に一致すると仮定)、このパターンは無視されます。

-r *repo\_uri*

参照リポジトリの場所を表す URI を指定します。-r を指定する場合は、-c も指定する必要があります。

-v

pkglint を冗長モードで実行し、構成ファイル内の log\_level 設定をすべて上書きします。

## ファイル

pkglintrc 構成ファイルは、次のキーと値の引数を取ります。

`log_level`

lint メッセージを発行する最小のレベル。このレベルより低い lint メッセージは破棄されます。デフォルト値は **INFO** です。

ログレベルを最小のレベルからもっとも重大なレベルの順に示すと、**DEBUG**、**INFO**、**WARNING**、**ERROR**、**CRITICAL** になります。

`do_pub_checks`

**True** の場合は、公開されたパッケージにとってのみ意味がある可能性のあるチェックを実行します。デフォルト値は **True** です。

`pkglint.ext.*`

pkglint のプラグインメカニズムを使用すると、実行時に lint モジュールを追加できます。pkglint.ext. で始まるキーはすべて、完全に指定された Python モジュールである必要のある値を取ります。詳細は、「開発者」のセクションを参照してください。

`pkglint.exclude`

実行されるチェックのセットから省略する、完全に指定された Python モジュール、クラス、または関数名のスペースで区切られたリスト。

`use_progress_tracker`

**True** の場合は、lint の実行中にマニフェストに対する処理を繰り返すときに進捗トラッカーを使用します。デフォルト値は **True** です。

`version.pattern`

lint の対象となるビルド番号を指定するときに使用されるバージョンのパターン (-b)。構成ファイルで指定されていない場合、-b オプションはパターン `*.5.11-0.` を使用します。これは、ブランチ接頭辞が 0 である 5.11 ビルドのすべてのコンポーネントに一致します。

## 開発者

pkglint によって実行されるチェックのセットを拡張するに

は、`pkg.lint.base.Checker` とそのサブクラス `ManifestChecker`

、`ActionChecker`、および `ContentChecker` をサブクラス化します。これらのクラスを含む Python モジュール名を、構成ファイル内の新しい `pkglint.ext.` キーに追加します。

これらの新しいサブクラスのインスタンスは、起動時に pkglint によって作成されます。lint セッション中に、特殊なキーワード引数 `pkglint_id` を持つ、各サブクラスの内部のメソッドが呼び出されます。これらのメソッドには、スーパークラス内の対応する `check()` メソッドと同じ署名が含まれているべきです。また、これらのメソッドには、`pkglint -L` によって出力される説明として使用される `pkglint_desc` 属性も割り当てられてはいけません。

パラメータは `Checker` サブクラスから使用できます。これにより、これらのサブクラスは自身の動作を調整できます。推奨されるパラメータの命名規則は、`pkglint_id.name` です。パラメータ値は構成ファイル内に格納するか、または



`LintEngine.get_param()` メソッドを使用して取得されるマニフェストまたはアクションでアクセスすることができます。マニフェストからパラメータにアクセスする場合は、`pkglint` パラメータが既存のどのアクションまたはマニフェスト値とも重複しないようにするために、キー名に接頭辞 `pkg.lint` が付加されます。

## 使用例

例1 特定のリポジトリに対する最初の実行

特定のリポジトリに対する `pkglint` セッションのはじめての実行。

```
$ pkglint -c /space/cache -r http://localhost:10000 mymanifest.mf
```

例2 同じリポジトリに対するそれ以降の実行

例1 で使用されている同じリポジトリに対するそれ以降の実行。

```
$ pkglint -c /space/cache mymanifest-fixed.mf
```

例3 絞り込まれたマニフェストセットでの `lint` リポジトリの使用

`lint` リポジトリでの `pkglint` セッションの実行と、チェックするマニフェストのサブセットの指定。

```
$ pkglint -c /space/othercache -l http://localhost:10000 \
-p '*.firefox.*'
```

例4 ビルドの指定

冗長モードでの特定のビルドに対する `pkglint` セッションの実行。

```
$ pkglint -c /space/cache -r http://localhost:10000 \
-l http://localhost:12000 -b 147 -v
```

例5 構成ファイルの変更

新しい `lint` モジュールを含む構成ファイル (一部のチェックを除外)。

```
$ cat ~/.pkglintrc
[pkglint]

log_level = DEBUG
log_level = INFO

pkglint.ext.mycheck = org.timf.mychecks
pkglint.ext.opensolaris = pkg.lint.opensolaris
pkglint.exclude: pkg.lint.opensolaris.OpenSolarisActionChecker
pkg.lint.pkglint.PkgActionChecker.unusual_perms pkg.lint.pkglint.PkgManifestChecker
pkg.lint.opensolaris.OpenSolarisManifestChecker
```

終了ステータス 次の終了ステータスが返されます。

- 0 コマンドが成功しました。
- 1 1 つ以上の `lint` チェックが出力を発行しました。

- 2 無効なコマンド行オプションが指定された。
- 99 予期しない例外が発生しました。

属性 次の属性については、attributes(5)を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目 [pkg\(1\)](#), [pkg.depotd\(1M\)](#), [pkgsend\(1\)](#), [pkg\(5\)](#)  
<http://hub.opensolaris.org/bin/view/Project+pkg/>

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgmerge – Image Packaging System パッケージマージユーティリティー                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 形式    | <pre> /usr/bin/pkgmerge [-n] -d <i>dest_repo</i>                   -s <i>variant=value[,...],src_repo ...</i>                   [<i>pkg_fmri_pattern ...</i>] </pre>                                                                                                                                                                                                                                                                                                                                                                                   |
| 機能説明  | <p>pkgmerge は、複数バリエーションのパッケージを作成するためのパッケージ公開ツールです。このツールはそのために、名前とバージョン(タイムスタンプは除外)が同じパッケージをマージし、特定のソースの指定されたバリエーション名および値とマージされるバージョン内で一意であるアクションをタグ付けしたあと、新しいパッケージをターゲットリポジトリに公開します。各ソースのすべてのパッケージの最新バージョンのみが使用されます。</p> <p>アクションの属性 <code>pkg.merge.blend</code> がマージされるバリエーションの名前に設定されている場合は、アクションの最終的な出力に追加されたバリエーションタグが含まれないようにするために、そのアクションはマージの前にほかのマニフェストにコピーされます。属性 <code>pkg.merge.blend</code> 自体は、出力マニフェストですべてのアクションから削除されることに注意してください。この属性は、複数のパスマージに異なる値を使用して繰り返すことができます。</p> <p>入力マニフェスト内の同じパスに入る同一でないアクションがあると、pkgmerge はエラーで終了します。</p> |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-?</p> <p>--help<br/>使用方法に関するメッセージを表示します。</p> <p>-d <i>dest_repo</i><br/>マージされたパッケージを公開する先のターゲットリポジトリのファイルシステムのパスまたは URI を指定します。このターゲットリポジトリはすでに存在している必要があります。新しいリポジトリは、pkgrepo を使用して作成できます。</p> <p>-n<br/>ターゲットリポジトリを変更することなく、試験実行を行います。</p> <p>-s <i>variant= value[,...],src_repo</i><br/>このソースのパッケージに使用するバリエーション名および値、そのあとに、パッケージを取得する元のソースリポジトリまたはパッケージアーカイブのファイルシステムのパスまたは URI を指定します。複数のバリエーションをコマンドで区切って指定できます。すべてのソースに対して同じバリエーションを指定する必要があります。このオプションは複数回指定できます。</p>                                       |
| 環境変数  | <p>次の環境変数がサポートされています。</p> <p>TMPDIR    プログラム実行中に一時データが格納されるディレクトリの絶対パス。設定されていない場合、一時データはデフォルトで /var/tmp に格納</p>                                                                                                                                                                                                                                                                                                                                                                                                                                        |

されます。

## 使用例

例1 バリエント名および値を指定する

指定されたソース内に見つかった各パッケージを、取得元のソースに指定された特定のバリエント名および値でタグ付けします。

```
$ pkgmerge -s arch=sparc,http://src.example.com \
-d http://dest.example.com
```

サンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
dir group=sys mode=0755 owner=root path=usr
```

操作のあとのサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
set name=variant.arch value=sparc
dir group=sys mode=0755 owner=root path=usr
```

例2 パッケージをマージおよび公開する

特定のソースの最新バージョンの各パッケージをマージし、新しいパッケージをターゲットリポジトリに公開します。

```
$ pkgmerge -s arch=sparc,http://src1.example.com \
-s arch=i386,http://src2.example.com \
-d /path/to/target/repository
```

ソース 1 (SPARC) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T121410Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

ソース 2 (i386) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

マージされたパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
set name=variant.arch value=sparc value=i386
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=sparc
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=i386
dir group=sys mode=0755 owner=root path=usr
```

例3 i386 システムと SPARC システムのデバッグおよびデバッグ以外のパッケージをマージする

i386 システムと SPARC システムの一連のデバッグおよびデバッグ以外のリポジトリ内の最新バージョンの各パッケージをマージします。

```
$ pkgmerge -s arch=sparc,debug=false,/repo/sparc-nondebug \
-s arch=sparc,debug=true,/repo/sparc-debug \
-s arch=i386,debug=false,/repo/i386-nondebug \
-s arch=i386,debug=true,/repo/i386-debug \
-d /path/to/target/repository
```

ソース 1 (SPARC デバッグ以外) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T121410Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

ソース 2 (SPARC デバッグ) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T121411Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

ソース 3 (i386 デバッグ以外) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

ソース 4 (i386 デバッグ) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163428Z
file id mode=0555 owner=root group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

マージされたパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163428Z
set name=variant.arch value=sparc value=i386
set name=variant.debug value=false value=true
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=sparc variant.debug=false
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=sparc variant.debug=true
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=i386 variant.debug=false
file id mode=0555 owner=root group=bin path=usr/bin/foo variant.arch=i386 variant.debug=true
dir group=sys mode=0755 owner=root path=usr
```

例4 pkg.merge.blend を使用してマージする

pkg.merge.blend 属性を使用して、競合しない2つのアーキテクチャーのパッケージをマージします。

例4 pkg.merge.blendを使用してマージする (続き)

```
$ pkgmerge -s arch=sparc,http://src1.example.com \
-s arch=i386,http://src2.example.com \
-d /path/to/target/repository
```

ソース 1 (SPARC) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T121410Z
file 1d5eac1aab628317f9c088d21e4afda9c754bb76 mode=0555 owner=root \
 group=bin path=usr/bin/sparc/foo pkg.merge.blend=arch
file d285ada5f3cae14ea00e97a8d99bd3e357caadc0 mode=0555 owner=root \
 group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

ソース 2 (i386) のサンプルパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
file a285ada5f3cae14ea00e97a8d99bd3e357cb0dca mode=0555 owner=root \
 group=bin path=usr/bin/i386/foo pkg.merge.blend=arch
file d285ada5f3cae14ea00e97a8d99bd3e357caadc0 mode=0555 owner=root \
 group=bin path=usr/bin/foo
dir group=sys mode=0755 owner=root path=usr
```

マージされたパッケージ:

```
set name=pkg.fmri value=pkg://example.com/foo@5.11,5.11-0.200:20381001T163427Z
set name=variant.arch value=sparc value=i386
file d285ada5f3cae14ea00e97a8d99bd3e357caadc0 mode=0555 owner=root \
 group=bin path=usr/bin/foo
file a285ada5f3cae14ea00e97a8d99bd3e357cb0dca mode=0555 owner=root \
 group=bin path=usr/bin/i386/foo
file 1d5eac1aab628317f9c088d21e4afda9c754bb76 mode=0555 owner=root \
 group=bin path=usr/bin/sparc/foo
dir group=sys mode=0755 owner=root path=usr
```

終了ステータス 次の終了ステータスが返されます。

- 0 コマンドが成功しました。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。
- 99 予期しない例外が発生しました。

属性 次の属性については、attributes(5)を参照してください。

---

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

## 関連項目

[pkgrepo\(1\)](#), [pkg\(5\)](#)<http://hub.opensolaris.org/bin/view/Project+pkg/>

名前 pkgmk – インストール可能なパッケージの生成

形式 pkgmk [-o] [-a *arch*] [-b *base\_src\_dir*] [-d *device*]  
[-f *prototype*] [-l *limit*] [-p *pstamp*] [-r *root\_path*]  
[-v *version*] [*variable=value*]... [*pkginst*]

機能説明 pkgmk ユーティリティーは、[pkgadd\(1M\)](#) コマンドへの入力として使用されるインストール可能なパッケージを生成します。パッケージの内容はディレクトリ構造形式です。

このコマンドは、パッケージの [prototype\(4\)](#) ファイルを入力として使用し、[pkgmap\(4\)](#) ファイルを作成します。prototype ファイルの各エントリの内容は、適切な出力場所にコピーされます。内容に関する情報 (チェックサム、ファイルサイズ、変更日付) が算出され、prototype ファイルに指定された属性情報とともに [pkgmap](#) ファイルに格納されます。

pkgmk は、次の条件に従って [prototype\(4\)](#) ファイルに一覧表示されているファイルを検索します。注: パッケージに含めるファイル場所が [prototype](#) ファイルで明示的に指定されている場合、検索に関する次の説明は適用されません。

1. -b オプションも -r オプションも指定しなかった場合、[prototype\(4\)](#) ファイルに一覧表示されている各ファイルパスのファイル名部分は [prototype\(4\)](#) ファイルと同じディレクトリ内にあるとみなされます。
2. -b で (先頭に「/」が付かない) 相対パスを指定した場合、[prototype\(4\)](#) ファイルの相対ファイルパスの前に *base\_src\_dir* が付加されます。この結果得られたパスが *root\_path* ディレクトリ内で検索されます。*root\_path* を指定しなかった場合、デフォルトの「/」が使用されます。
3. -b で (先頭に「/」を付けた) 絶対パスを指定した場合、[prototype\(4\)](#) ファイルの相対パスの前に *base\_src\_dir* が付加され、その結果がファイルの場所になります。*root\_path* は検索されません。
4. -r を指定した場合、[prototype\(4\)](#) ファイルの完全なファイルパスが使用されます。相対パスの前には *base\_src\_dir* が付加されます。*base\_src\_dir* を指定しなかった場合は、デフォルトの "" が使用されます。この結果得られたパスが *root\_path* の各ディレクトリ内で検索されます。

"[pkgproto a/relative/path](#)" または "[pkgproto a/relative/path=install/path](#)" を使用して [prototype](#) ファイルを作成した場合、[pkgmk](#) がソースファイルの場所を正しく特定できるように、-r *root\_path* オプションを使用して [a/relative/path](#) の場所を指定するようにしてください。

[pkgmk](#) などのパッケージコマンドは [largefile\(5\)](#) を認識します。これらのコマンドは、2G バイトより大きなファイルを、より小さなファイルと同様に処理します。現在の実装では、[pkgadd\(1M\)](#)、[pkgtrans\(1\)](#)、およびその他のパッケージコマンドは最大 4G バイトのデータストリームを処理できます。



|                        |                                                                                                                                                                                                                                        |
|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| オプション                  | サポートしているオプションは、次のとおりです。                                                                                                                                                                                                                |
| <b>-a arch</b>         | <a href="#">pkginfo(4)</a> ファイルに指定されたアーキテクチャー情報を <i>arch</i> でオーバーライドします。                                                                                                                                                              |
| <b>-b base_src_dir</b> | 指定された <i>base_src_dir</i> を先頭に付加して、ソースマシン上の再配置可能オブジェクトの場所を特定します。このオプションは、 <i>prototype</i> ファイル内のすべてのオブジェクトを検索する場合に使用します。 <i>pkgmk</i> は、 <i>/base_src_dir</i> でオブジェクトを検索したり、 <b>-b</b> オプションと <b>-r</b> オプションを使用してオブジェクトの場所を特定したりします。 |
| <b>-d device</b>       | <i>device</i> 上にパッケージを作成します。 <i>device</i> はディレクトリの絶対パス名、または着脱式ディスクの識別子のどちらでもかまいません。デフォルトのデバイスはインストールスプールディレクトリ ( <i>/var/spool/pkg</i> ) です。                                                                                          |
| <b>-f prototype</b>    | コマンドへの入力として <i>prototype</i> ファイルを使用します。デフォルトの <i>prototype</i> ファイル名は <i>[Pp]rototype</i> です。                                                                                                                                         |
| <b>-l limit</b>        | <i>limit</i> に、出力デバイスの最大サイズを512バイトブロック単位で指定します。デフォルトでは、出力ファイルがディレクトリまたはマウント可能なデバイスである場合、 <i>pkgmk</i> は <b>df(1M)</b> コマンドを使用して出力デバイスの使用可能な容量を動的に計算します。このオプションを <a href="#">pkgtrans(1)</a> と組み合わせて使用すると、パッケージをデータストリーム形式で作成できます。      |
| <b>-o</b>              | 同じインスタンスを上書きします。パッケージインスタンスがすでに存在する場合は上書きされます。                                                                                                                                                                                         |
| <b>-p pstamp</b>       | <a href="#">pkginfo(4)</a> ファイル内の製品スタンプ定義を <i>pstamp</i> でオーバーライドします。                                                                                                                                                                  |
| <b>-r root_path</b>    | 指定された <i>root_path</i> にソースパス名を付加したものを使用して、ソースマシン上のオブジェクトの場所を特定します。パス要素の区切り文字にはコンマ (,) を使用します。このオプションを指定した場合、指定したディレクトリごとに完全な宛先パスが検索されます。 <b>-b</b> も <b>-r</b> も指定しなかった場合は、現在のディレクトリ内のリーフファイル名が検索されます。                                |
| <b>-v version</b>      | <a href="#">pkginfo(4)</a> ファイルに指定されたバージョン情報を <i>version</i> でオーバーライドします。                                                                                                                                                              |
| <b>variable=value</b>  | 指定された変数をパッケージング環境に設定します。変数指定の定義については、 <a href="#">prototype(4)</a> を参照してください。                                                                                                                                                          |
| オペランド                  | 次のオペランドを指定できます。                                                                                                                                                                                                                        |

*pkginst* インスタンスによるパッケージの指定。インスタンスはパッケージの省略形または特定のインスタンス (たとえば、*inst.1* や *inst.2*) のどちらでもかまいません。パッケージのすべてのインスタンスを要求するには、*inst.\** を使用します。シェルによっては、アスタリスク文字 (\*) が特殊な意味を持つことがあり、\* をエスケープしなければならない場合があります。C シェルでは、\* は単一引用符 (') で囲むか、またはバックスラッシュ (\) を前に付ける必要があります。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [pkgparam\(1\)](#), [pkgproto\(1\)](#), [pkgtrans\(1\)](#), [uname\(1\)](#), [df\(1M\)](#), [pkgadd\(1M\)](#), [pkginfo\(4\)](#), [pkgmap\(4\)](#), [prototype\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

『Application Packaging Developer’s Guide』

注意事項 アーキテクチャー情報は、コマンド行で *-a* オプションを使用して指定するか、または [prototype\(4\)](#) ファイルに入力します。アーキテクチャー情報を指定しなかった場合は、*uname -m* の出力が使用されます ([uname\(1\)](#) を参照)。

バージョン情報は、コマンド行で *-v* オプションを使用してを指定するか、または [pkginfo\(4\)](#) ファイルに入力します。バージョン情報を指定しなかった場合は、現在の日付に基づくデフォルトが指定されます。

アーキテクチャーとバージョンのどちらについても、コマンド行の定義が [prototype\(4\)](#) の定義よりも優先されます。

ゾーン関連パラメータを次のような無効な組み合わせで使用すると、*pkgmk* は失敗します。

1. *SUNW\_PKG\_ALLZONES* と *SUNW\_PKG\_THISZONE* の両方を *TRUE* に設定します。
2. *SUNW\_PKG\_HOLLOW* を *TRUE* に設定し、*SUNW\_PKG\_ALLZONES* を *FALSE* に設定します。
3. パッケージに要求スクリプトを含め、*SUNW\_PKG\_THISZONE* を *TRUE* に設定します。

これらのパラメータの詳細については、[pkginfo\(4\)](#) を参照してください。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgmogrify – Image Packaging System マニフェスト変換ツール                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 形式    | <pre> /usr/bin/pkgmogrify [-vi] [-I includedir ...]                   [-D macro=value ...] [-O outputfile]                   [-P printfile] [inputfile ...] </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 機能説明  | <p>pkgmogrify は、パッケージマニフェストのプログラムによる編集を可能にして、ソフトウェアの構築やパッケージの公開を自動化するときに必要な標準的な変換を簡略化します。</p> <p>pkgmogrify は、次の機能を提供します。</p> <ul style="list-style-type: none"> <li>■ マクロ置換。単一のマニフェストを各種アーキテクチャーやプラットフォーム間で簡単に共有できます。</li> <li>■ ほかのマニフェスト、または標準のコンポーネントや変換などのマニフェストフラグメントの取り込み。</li> <li>■ パッケージアクションの変換。アクション属性の変更、削除、または追加が含まれます。</li> </ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-?</p> <p>--help<br/>使用方法に関するメッセージを表示します。</p> <p>-D <i>macro</i>= <i>value</i><br/><i>macro</i> を値 <i>value</i> とともにマクロとして定義します。マクロは、入力ファイル内で \$(<i>macro</i>) として指定されます。置換は、それ以上の変換が見つからなくなるまで繰り返されます。一般的な語法には次のものがあります。</p> <ul style="list-style-type: none"> <li>■ 行の先頭にあるアーキテクチャー固有のタグを使用した、ほかのアーキテクチャー上のマニフェスト内の行の除外。</li> </ul> <pre>\$(sparc_ONLY)file ...</pre> <p>SPARC アーキテクチャーを処理する場合、このマクロは空の文字列に設定されます。ほかのアーキテクチャーを処理する場合、このマクロはコマンド行で # に設定されるため、このアクションが現在のアーキテクチャー上のマニフェストから除外されます。</p> <ul style="list-style-type: none"> <li>■ パス名のプラットフォーム固有の部分の指定。実行可能ファイルおよびライブラリのための 64 ビットアーキテクチャーディレクトリの名前などがあります。</li> </ul> <pre>file NOHASH path=usr/bin/\$(ARCH64)/cputrack ...</pre> <p>これらのマクロは、コマンド行で目的の値に設定するようにしてください。定義済みのマクロ値は存在しません。</p> |

**-I *includedir***

指定されたディレクトリを、コマンド行で指定されたファイルと、埋め込みのインクルード指令の両方の検索パスに追加します。

**-O *outputfile***

マニフェスト出力を指定されたファイルに書き込みます。このファイルは、エラーが発生した場合や、変換指令によって強制的な中止操作が実行された場合は書き込まれません。デフォルトでは、マニフェスト出力は標準出力に書き込まれます。

**-P *printfile***

変換指令の出力操作から得られた出力を指定されたファイルに書き込みます。このファイルは、エラーが発生した場合や、変換指令によって強制的な中止操作が実行された場合は書き込まれません。デフォルトでは、この出力は標準出力に書き込まれます。

**-i**

ファイル内のインクルード指令を無視します。コマンド行 (または標準入力) で指定されたファイルのみが処理されます。

**-v**

出力マニフェストに変換の効果を示すコメントを書き込みます。この情報がデバッグに役立つことがあります。

**埋め込みの指令**

マニフェストファイルでは、インクルード指令と変換指令という2つのタイプの指令がサポートされています。

インクルード指令の形式は次のとおりです。

```
<include file>
```

この指令により、pkgmogrifyは*file*という名前のファイルを、最初に現在のディレクトリで、次に -I オプションで指定されたディレクトリで検索します。見つかった場合は、ファイルの内容がマニフェストの、この指令が置かれている場所に挿入されます。見つからなかった場合、pkgmogrifyはエラーで終了します。

変換指令の形式は次のとおりです。

```
<transform matching-criteria -> operation>
```

これらの指令は、すべての入力メモリ内に読み取られるまで累積されたあと、各指令が検出された順序でアクションに適用されます。

一致条件の形式は次のとおりです。

```
[action-name ...] [attribute=<value-regex> ...]
```

指定された *action-name* の少なくとも1つが一致する必要があります。指定されたすべての *attribute* が一致する必要があります。アクションの名前と属性が、pkg(5) マニュアルページの「アクション」に一覧表示されます。使用されている正規表現

の構文はPythonの構文です。Python正規表現の構文については、コマンド `pydoc re` を使用するか、または <http://docs.python.org/dev/howto/regex.html> にあるより完全なドキュメントを参照してください。正規表現は先頭に固定され、末尾には固定されません。そのため、ファイルを拡張子で照合する正規表現には先頭に `.` を含める必要があり、さらに末尾には `$` を含めるようにしてください。

複数の条件をスペースで区切って指定できます。

次の操作が使用できます。

|                      |                                                                                                                                                                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>add</code>     | 属性に値を追加します。この操作は2つの引数を取ります。最初の引数は属性の名前であり、2番目の引数は値です。                                                                                                                                                                                               |
| <code>default</code> | 属性の値がまだ存在しない場合は、その値を設定します。この操作は <code>add</code> 操作と同じく2つの引数を取ります。                                                                                                                                                                                  |
| <code>delete</code>  | 属性値を削除します。この操作は2つの引数を取ります。最初の引数は属性の名前です。2番目の引数は、削除される属性値を照合するための正規表現です。アクションを照合するために使用される正規表現とは異なり、この表現は固定されません。                                                                                                                                    |
| <code>drop</code>    | このアクションを破棄します。                                                                                                                                                                                                                                      |
| <code>edit</code>    | アクションの属性を変更します。この操作には3つの引数を指定します。最初の引数は属性の名前であり、2番目の引数は属性値を照合するための正規表現です。3番目の引数は、正規表現で一致した値の部分に置き換えられる置換文字列です。アクションを照合するために使用される正規表現とは異なり、この表現は固定されません。正規表現でグループが定義されている場合は、置換文字列内で通常の正規表現の後方参照 (形式は <code>\1</code> 、 <code>\2</code> など) を使用できます。 |
| <code>emit</code>    | マニフェスト出力ストリームに行を出力します。これは、有効なアクション文字列、空 (空行が生成されます)、またはコメント ( <code>#</code> のあとに任意のテキストが続きます) である必要があります。                                                                                                                                          |
| <code>exit</code>    | マニフェスト処理を終了します。マニフェストは出力されず、 <code>print</code> 操作は適用されません。引数が1つ指定されている場合、その引数は整数である必要があり、終了コードとして使用されます。デフォルト値は0です。引数が2つ指定されている場合、最初の引数は終了コードであり、2番目の引数は標準エラー出力に出力されるメッセージです。                                                                      |
| <code>print</code>   | <code>-P</code> で指定された出力ファイルにメッセージを出力します。                                                                                                                                                                                                           |
| <code>set</code>     | 属性の値を設定します。この操作は <code>add</code> 操作と同じく2つの引数を取ります。                                                                                                                                                                                                 |

`delete` と `drop` を除くすべての操作は、出力ストリームにその内容が出力される (省略可能である可能性がある) 引数を取ります。これらの文字列には、3種類の特殊な

トークンを含めることができます。これらのトークンにより、各アクションの固定された変換に基づかない情報を出力に含めることが可能になります。

最初の種類の置換を使用すると、パーセント記号に続けて属性の名前を丸括弧に入れて指定することによって、操作で現在のアクションの属性の値を参照できます。たとえば、%(alias) は、アクションの alias 属性の値を参照します。

いくつかの合成属性が存在します。次の2つは pkgmogrify に固有です

- pkg.manifest.filename は、アクションが見つかったファイルの名前を参照します。
- pkg.manifest.lineno は、アクションが見つかった行を参照します。

次の3つの合成属性は、pkg で使用されるものに似ています:

- action.hash は、アクションにペイロードが含まれている場合、そのアクションのハッシュ値を参照します。ペイロードを含むアクションの場合、set 操作は、action.hash 属性を操作することによってアクションのハッシュを変更できます。
- action.key は、キー属性の値を参照します。
- action.name は、アクションの名前を参照します。

値を要求された属性が存在しない場合、pkgmogrify はエラーで終了します。エラーで終了することを回避するには、属性名のあとに ;notfound= と、属性値の代わりに置換する値を指定します。たとえば、%(alias;notfound='no alias') は、属性 alias の値が存在する場合はその値を出力し、それ以外の場合は no alias を出力します。

値を要求された属性が複数値の場合は、各値がスペースで区切られて出力されます。notfound トークンと同様に、トークン prefix、suffix、および sep を使用してこの動作を変更できます。prefix で指定された文字列は各値の先頭に追加され、suffix で指定された文字列は各値の末尾に追加され、sep はある値の接尾辞とその次の値の接頭辞の間に配置されます。

アクション属性と同様に、pkgmogrify 指令では %{pkg.fmri} のように、丸括弧の代わりに中括弧を使用してパッケージ属性を参照できます。変換指令が適用される時点で、この属性は set アクションで定義されている必要があります。そうしないと、この属性は上で説明した notfound として扱われます。処理が、パッケージを記述しているマニフェストファイルの最後に達すると、属性は次のパッケージのためにクリアされます。

これは、パッケージ属性をアクション属性であるかのように参照するためだけでなく、これらの属性を照合したり、一時的に変更したりするためにも有効です。したがって、これらの状況では、合成アクション名 pkg を (pkgmogrify のコンテキストでのみ) 使用できます。



pkgmogrify がコマンド行で指定されたマニフェストの読み取りを完了し、そのマニフェストで pkg.fmri パッケージ属性が定義されている場合、pkgmogrify はこの合成アクション pkg を作成します。属性はパッケージの属性になります。その後、<transform> 指令は、ほかのアクションに照合できるのと同様に、このアクションに照合できます。

pkg アクションに対する操作は、メモリー内でのみ実行されるために、出力されたマニフェストに直接影響を与えないという点で特殊です。たとえば、add、default、または set 操作を使用して pkg アクションに属性を設定しようとしても、照合するほかの <transform> 指令では使用可能であるにもかかわらず、set アクションがマニフェストに追加されません。pkg アクションに対して emit を実行しようとする、エラーが発生します。パッケージ属性を追加するには、代わりに set アクションに対して emit を実行します。

3 番目の種類の置換は後方参照です。この置換は edit 操作で使用可能なものとは異なり、-> の左側にある変換一致に列挙されているグループへの参照です。これらは、一致条件にある順序で、%<1>、%<2> などによって指定されます。

処理の順序は次のとおりです。

1. 入力ファイルから行が読み取られます。
2. マクロが適用されます。
3. <include ...> および <transform> 指令が処理されることにより、追加のファイルが検出され、読み取られます。
4. すべての入力が累積されたあと、入力内の各行がアクションに変換され、すべての変換が適用されます。
5. 処理が正常に完了すると、出力が書き込まれます。

## 使用例

例1 SMF マニフェストにタグを追加する

サービス管理機能 (SMF) マニフェストにタグを追加して、パッケージがライブシステムにインストールされたときにこれらのマニフェストがインポートされるようにします。

```
<transform file path=(var|lib)/svc/manifest/*.xml -> \
 add restart_fmri svc:/system/manifest-import:default>
```

例2 ファイルを移動する

ファイルを usr/sfw/bin から usr/bin に移動します。

```
<transform file -> edit path usr/sfw/bin usr/bin>
```

**例3** リブートの必要性を指定する

reboot-needed タグを、/kernel の下にある、.conf ファイル以外のファイルに追加します。この例では、入力ファイルに存在する順序で各アクションに変換が適用されることを利用しています。

```
<transform file path=kernel/*. * -> set reboot-needed true>
<transform file path=kernel/*. *.conf -> delete reboot-needed . *>
```

これはまた、正規表現を使用した1つの変換規則でも実行できます。

**例4** FMRI属性を依存アクションに変換する

パッケージ属性 pkg.fmri を incorporation の一部になるように depend アクションに変換します。

```
<transform set name=pkg.fmri -> \
 emit depend type=incorporate fmri=%(value)>
<transform set name=pkg.fmri -> drop>
```

**例5** バグ番号のリストを出力する

引用符で囲まれた、接頭辞付きのバグ番号のコンマ区切りリストを出力します。

```
set name=bugs value=12345 value=54321 value=13579 value=97531
<transform set name=bugs -> \
 print %(value;sep=",";prefix="bug='";suffix="'")>
```

**例6** 欠落した属性を許可する

属性がない場合でも、安全にメッセージを出力します。

```
<transform driver -> print Found aliases: %(alias;notfound=<none>)>
```

**例7** デフォルト値を設定する

所有者、グループ、およびアクセス権のデフォルト値を設定します。

```
<transform file dir -> default owner root>
<transform file dir -> default group bin>
<transform file -> default mode 0444>
<transform dir -> default mode 0755>
```

**例8** 廃止としてマークされていないパッケージに依存関係を追加する

廃止としてマークされていないすべてのパッケージについて、そのパッケージを提供する統合のための incorporation への依存関係を追加します。この一連の変換は、マニフェストが読み取られたあとに実行される必要があります。そうしないと、依存関係が常に出力されます。pkg アクションの変更には永続的な効果がないため、pkg.obsolete=false に一致する属性をクリーンアップする必要はありません。



例8 廃止としてマークされていないパッケージに依存関係を追加する (続き)

```
<transform pkg -> default pkg.obsolete false>
<transform pkg pkg.obsolete=false -> emit depend \
 fmri=consolidation/$(CONS)/$(CONS)-incorporation type=require>
```

例9 エラーが検出された場合は終了してメッセージを出力する  
廃止された属性がマニフェスト内に検出された場合は、エラーメッセージを出力します。

```
<transform file dir link hardlink opensolaris.zone=.* -> \
 exit 1 The opensolaris.zone attribute is obsolete.>
```

例10 適切なロケールファセットを設定する  
対象のパス名に適したロケールファセットを設定します。

```
<transform dir file link hardlink path=.* /locale/([^\s/]+).* -> \
 default facet.locale.%<1> true>
```

終了ステータス 次の終了ステータスが返されます。

- 0     すべてが動作しました。
- 1     予期しない不具合が発生しました。
- 2     無効なコマンド行オプションが指定された。
- 99    予期しない処理エラー。

ファイル     /usr/share/pkg/transforms  
              このディレクトリには、ファセット、アクチュエータ、およびその他の属性を設定するために役立つ変換を含むファイルが入っています。

属性         次の属性については、attributes(5) を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目     [pkg\(1\)](#), [pkg\(5\)](#)  
<http://hub.opensolaris.org/bin/view/Project+pkg/>

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------|----------------------------------|---------------------|-----------------------------------------------------------------------------------------------------------------------|-----------|---------------------------|
| 名前                  | pkgparam - パッケージパラメータ値の表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| 形式                  | <pre>pkgparam [-v] [-d device] [-R root_path] pkginst [param]...<br/>pkgparam -f filename [-v] [param]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| 機能説明                | <p>pkgparam は、コマンド行で要求された 1 つ以上のパラメータに関連付けられた値を表示します。この値は、<i>pkginst</i> の <a href="#">pkginfo(4)</a> ファイルまたは <i>-f</i> オプションで指定された特定のファイルから検索されます。</p> <p>1 行に 1 つのパラメータ値が表示されます。<i>-v</i> オプションを使用しない場合は、パラメータの値だけが表示されます。このオプションを使用すると、コマンドの出力は次の形式になります。</p> <pre>parameter1='value1'<br/>parameter2='value2'<br/>parameter3='value3'</pre> <p>コマンド行にパラメータを指定しなかった場合は、パッケージに関連付けられたすべてのパラメータの値が表示されます。</p>                                                                                                                                                                          |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| オプション               | <p>このコマンドのオプションと引数は次のとおりです。</p> <table><tr><td><i>-d device</i></td><td><i>pkginst</i> を保存するデバイスを指定します。ディレクトリパス名、または、テープかりムーバブルディスクの識別子 (/var/tmp、/, /dev/dsk/c1d0s0 など) を指定できます。spool という特殊なトークンを使用すると、デフォルトのインストールスプールディレクトリ (/var/spool/pkg) を指定できます。</td></tr><tr><td><i>-f filename</i></td><td><i>filename</i> からパラメータ値を読み取ります。</td></tr><tr><td><i>-R root_path</i></td><td><i>root_path</i> として使用するサブディレクトリのフルパス名を定義します。すべてのファイル (パッケージシステム情報ファイルを含む) は <i>root_path</i> から始まるディレクトリツリーに再配置されます。</td></tr><tr><td><i>-v</i></td><td>冗長モード。パラメータの名前とその値を表示します。</td></tr></table> | <i>-d device</i> | <i>pkginst</i> を保存するデバイスを指定します。ディレクトリパス名、または、テープかりムーバブルディスクの識別子 (/var/tmp、/, /dev/dsk/c1d0s0 など) を指定できます。spool という特殊なトークンを使用すると、デフォルトのインストールスプールディレクトリ (/var/spool/pkg) を指定できます。 | <i>-f filename</i> | <i>filename</i> からパラメータ値を読み取ります。 | <i>-R root_path</i> | <i>root_path</i> として使用するサブディレクトリのフルパス名を定義します。すべてのファイル (パッケージシステム情報ファイルを含む) は <i>root_path</i> から始まるディレクトリツリーに再配置されます。 | <i>-v</i> | 冗長モード。パラメータの名前とその値を表示します。 |
| <i>-d device</i>    | <i>pkginst</i> を保存するデバイスを指定します。ディレクトリパス名、または、テープかりムーバブルディスクの識別子 (/var/tmp、/, /dev/dsk/c1d0s0 など) を指定できます。spool という特殊なトークンを使用すると、デフォルトのインストールスプールディレクトリ (/var/spool/pkg) を指定できます。                                                                                                                                                                                                                                                                                                                                                                                                        |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| <i>-f filename</i>  | <i>filename</i> からパラメータ値を読み取ります。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| <i>-R root_path</i> | <i>root_path</i> として使用するサブディレクトリのフルパス名を定義します。すべてのファイル (パッケージシステム情報ファイルを含む) は <i>root_path</i> から始まるディレクトリツリーに再配置されます。                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| <i>-v</i>           | 冗長モード。パラメータの名前とその値を表示します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| オペランド               | <table><tr><td><i>pkginst</i></td><td>パラメータ値を表示する特定のパッケージインスタンスを定義します。</td></tr><tr><td><i>param</i></td><td>値を表示する特定のパラメータを定義します。</td></tr></table>                                                                                                                                                                                                                                                                                                                                                                                                                                    | <i>pkginst</i>   | パラメータ値を表示する特定のパッケージインスタンスを定義します。                                                                                                                                                 | <i>param</i>       | 値を表示する特定のパラメータを定義します。            |                     |                                                                                                                       |           |                           |
| <i>pkginst</i>      | パラメータ値を表示する特定のパッケージインスタンスを定義します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| <i>param</i>        | 値を表示する特定のパラメータを定義します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| エラー                 | 指定されたパッケージからパラメータ情報を取得できない場合、コマンドは 0 以外のステータスで終了します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| 終了ステータス             | <table><tr><td>0</td><td>正常終了。</td></tr><tr><td>&gt;0</td><td>エラーが発生した。</td></tr></table>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | 0                | 正常終了。                                                                                                                                                                            | >0                 | エラーが発生した。                        |                     |                                                                                                                       |           |                           |
| 0                   | 正常終了。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| >0                  | エラーが発生した。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |
| 属性                  | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                  |                                                                                                                                                                                  |                    |                                  |                     |                                                                                                                       |           |                           |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [pkgmk\(1\)](#), [pkgproto\(1\)](#), [pkgtrans\(1\)](#), [pkgadd\(1M\)](#), [pkginfo\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

『[Application Packaging Developer's Guide](#)』

注意事項 `-f` オプションを使用すると、パラメータ値の抽出元ファイルを指定できます。このファイルは、[pkginfo\(4\)](#) ファイルと同じ形式でなければなりません。たとえば、このようなファイルをパッケージの開発中に作成しておき、この段階でソフトウェアをテストするときに使用できます。

パッケージコマンドは、[largefile\(5\)](#) を認識します。これらのコマンドは、2G バイトより大きなファイルを、より小さなファイルと同様に処理します。現在の実装では、[pkgadd\(1M\)](#)、[pkgtrans\(1\)](#)、およびその他のパッケージコマンドは最大 4G バイトのデータストリームを処理できます。

**名前** pkgproto – pkgmk コマンドに入力する prototype ファイルエントリの生成

**形式** pkgproto [-i] [-c *class*] [*path1*]  
pkgproto [-i] [-c *class*] [*path1=path2*]...

**機能説明** pkgproto は、指定されたパスを調べ、[pkgmk\(1\)](#) コマンドへの入力として使用できる [prototype\(4\)](#) ファイルのエントリを生成します。

コマンド行にパスを指定しなかった場合、標準入力が入力のパスの一覧とみなされます。コマンド行に指定したパス名がディレクトリである場合は、ディレクトリの内容が検索されます。ただし、stdin から入力を読み取る場合は、パス名として指定したディレクトリは検索されません。

pkgproto などのパッケージコマンドは [largefile\(5\)](#) を認識します。これらのコマンドは、2G バイトより大きなファイルを、より小さなファイルと同様に処理します。現在の実装では、[pkgadd\(1M\)](#)、[pkgtrans\(1\)](#)、およびその他のパッケージコマンドは最大 4G バイトのデータストリームを処理できます。

**オプション** -i シンボリックリンクを無視し、そのパスを `ftype=s` (シンボリックリンク) ではなく `ftype=f` (ファイル) として記録します。

-c *class* すべてのパスのクラスを *class* にマップします。

**オペランド** *path1* オブジェクトが配置されているパス名。

*path2* 出力時に *path1* から置き換えられるパス名。

**使用例** 例1 基本的な使用方法

次の例は、pkgproto の一般的な使用方法と生成される出力の一覧の一部を示しています。

```
example% pkgproto /bin=bin /usr/bin=usrbin /etc=etc
f none bin/sed=/bin/sed 0775 bin bin
f none bin/sh=/bin/sh 0755 bin daemon
f none bin/sort=/bin/sort 0755 bin bin
f none usrbin/sdb=/usr/bin/sdb 0775 bin bin
f none usrbin/shl=/usr/bin/shl 4755 bin bin
d none etc/master.d 0755 root daemon
f none etc/master.d/kernel=/etc/master.d/kernel 0644 root daemon
f none etc/rc=/etc/rc 0744 root daemon
```

例2 パイプラインで pkgproto を使用する

次のコマンドは、find コマンドの出力を受け入れる pkgproto を示します。

```
example% find / -type d -print | pkgproto
d none / 755 root root
d none /bin 755 bin bin
d none /usr 755 root root
d none /usr/bin 775 bin bin
```

例2 パイプラインで pkgproto を使用する (続き)

```
d none /etc 755 root root
d none /tmp 777 root root
```

終了ステータス 0 正常終了。  
>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [pkgmk\(1\)](#), [pkgparam\(1\)](#), [pkgtrans\(1\)](#), [pkgadd\(1M\)](#), [prototype\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#)

『Application Packaging Developer’s Guide』

注意事項 デフォルトでは、pkgproto は、検出されたあらゆるシンボリックリンクに対してシンボリックリンクエントリ (ftype=s) を作成します。-i オプションを使用すると、pkgproto は、シンボリックリンクに対してファイルエントリ (ftype=f) を作成します。v (揮発性)、e (編集可能)、x (排他的ディレクトリ) などのファイルタイプを割り当てるには、[prototype\(4\)](#) ファイルを編集する必要があります。pkgproto は、リンクされたファイルを検出します。複数のファイルが同時にリンクされている場合は、最初に検出されたパスがそのリンクのソースとみなされます。

デフォルトでは、pkgproto は prototype のエントリを標準出力に出力します。ただし、出力を [pkgmk\(1\)](#) コマンドへの入力として使用するには、その出力をファイルに保存するようにしてください (ファイル名を Prototype または prototype にすると便利です)。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgrecv – Image Packaging System 内容取得ユーティリティー                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 形式    | <pre>/usr/bin/pkgrecv [-s <i>src_uri</i>] [-a] [-d (<i>path</i> <i>dest_uri</i>)]<br/>[-c <i>cache_dir</i>] [-kr] [-m <i>match</i>] [-n] [--raw]<br/>[--key <i>keyfile</i> --cert <i>certfile</i>] (<i>fmri</i> <i>pattern</i>) ...<br/><br/>/usr/bin/pkgrecv [-s <i>src_uri</i>] --newest</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 機能説明  | <p>pkgrecv を使用すると、ユーザーは pkg(5) リポジトリまたはパッケージアーカイブからパッケージを取得できます。また、pkgrecv はオプションで、取得されたパッケージを別のパッケージリポジトリに再公開したり、アーカイブしたりすることもできます。デフォルトでは、パッケージは pkg、pkg.depotd、およびパッケージ公開ツールでの使用に適したパッケージリポジトリ形式で取得されます。</p> <p>pkgrecv 操作後に、リポジトリに対して pkgrepo refresh または pkgrepo rebuild を実行し、検索インデックスを構築します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p>-h<br/>使用方法に関するメッセージを表示します。</p> <p>-a<br/>取得されたパッケージデータを、-d で指定された場所にある pkg(5) アーカイブ内に格納します。このファイルがすでに存在してはいけません。このオプションは、ファイルシステムベースの出力先の場合にのみ使用できます。必須ではありませんが、.p5p のファイル拡張子 (たとえば、archive.p5p) を使用することを強くお勧めします。このオプションを --raw と組み合わせることはできません。</p> <p>-c <i>cache_dir</i><br/>ダウンロードされた内容をキャッシュするために使用されるディレクトリのパスを指定します。このディレクトリが指定されていない場合は、クライアントによってキャッシュディレクトリが自動的に選択されます。ダウンロードが中断されたときに、キャッシュディレクトリが自動的に選択されていた場合は、このオプションを使用してダウンロードを再開します。一時的なデータストレージに使用される場所を設定する方法についての詳細は、下の「環境変数」のセクションを参照してください。</p> <p>-d (<i>path</i> <i>dest_uri</i>)<br/>パッケージを再公開する先のターゲットのファイルシステムのパスまたは URI を指定します。-a が指定されている場合、ターゲットはまだ存在しない新しいパッケージアーカイブです。指定されていない場合、ターゲットはすでに存在するパッケージリポジトリである必要があります。新しいリポジトリは、pkgrepo を使用して作成できます。</p> |

- k  
取得されたパッケージの内容を圧縮されたままにします。このオプションは、再公開時には無視されます。圧縮されたパッケージの内容を `pkgsend` で使用しないようにしてください。
- m *match*  
次の値を使用して、マッチング動作を制御します：
  - all-timestamps*  
最新のタイムスタンプだけでなく、一致するすべてのタイムスタンプを含めます (*all-versions* が暗黙的に指定されます)。
  - all-versions*  
最新のバージョンだけでなく、一致するすべてのバージョンを含めます。
- n  
変更を行うことなく、試験実行を行います。
- r  
提供されたパッケージリストのすべての依存関係を再帰的に取得します。
- s *src\_uri*  
パッケージデータ受信先の `pkg(5)` リポジトリまたはパッケージアーカイブの場所を表す URI を指定します。
- cert *certfile*  
HTTPS リポジトリからのパッケージ取得に使用するクライアント SSL 証明書ファイルを指定します。
- key *keyfile*  
HTTPS リポジトリからのパッケージ取得に使用するクライアント SSL キーファイルを指定します。
- newest  
指定されたりポジトリから使用可能な最新バージョンのパッケージを一覧表示し、終了します (-s を除くその他のオプションはすべて無視されます)。
- raw  
*raw* パッケージデータを取得し、-d で指定された場所に、幹およびバージョンごとの一連のディレクトリ構造内に格納します。このオプションは、ファイルシステムベースの出力先の場合にのみ使用できます。このパッケージデータを使用すると、一般にはファイルの内容を修正するか、または追加のパッケージメタデータを提供することによって、パッケージを便利に変更して再公開することができます。このオプションを -a と組み合わせることはできません。

## 使用例

例1 最新のパッケージを一覧表示する

`test` という名前のシステム上のリポジトリから使用可能な最新のパッケージを一覧表示します。

例1 最新のパッケージを一覧表示する (続き)

```
$ pkgrecv -s http://test --newest
pkg://solaris/system/library/c++-runtime@0.5.11,5.11-0.175.0.0.0.2.1:20120921T190358Z
pkg://solaris/system/library/freetype-2@2.4.8,5.11-0.175.1.0.0.7.1234:20120109T215840Z
pkg://solaris/system/library/math@0.5.11,5.11-0.175.0.0.0.2.1:20120921T190432Z
```

例2 raw パッケージデータを取得する

例1 から c++-runtime パッケージを pkgsend publish で使用するために適した形式で受け取ります。

```
$ pkgrecv -s http://test \
-d /local/repo --raw \
c++-runtime@0.5.11,5.11-0.175.0.0.0.2.1:20120921T190358Z
Processing packages for publisher solaris ...
Retrieving and evaluating 1 package(s)...
PROCESS ITEMS GET (MB) SEND (MB)
Completed 1/1 3.5/3.5 0.0/0.0
$ ls /local/repo
pkg5.repository publisher system%2Flibrary%2Fc%2B%2B-runtime
```

例3 システムから依存関係を取得する

test という名前のシステムから、パッケージ editor/vim とそのすべての依存関係を受け取ります。

```
$ pkgrecv -s http://test -d /local/repo -r editor/vim
```

例4 すべてのバージョンを取得する

test という名前のシステムから、パッケージ editor/vim のすべてのバージョンを受け取ります。

```
$ pkgrecv -s http://test -d /local/repo -m all-versions \
editor/vim
Processing packages for publisher solaris ...
Retrieving and evaluating 2 package(s)...
PROCESS ITEMS GET (MB) SEND (MB)
Completed 2/2 16.7/16.7 44.9/44.9
```

例5 すべてのバージョンを取得し、リモートから再公開する

test という名前のシステムから、パッケージ library/zlib のすべてのバージョンを受け取り、それを remote という名前のシステム上のリモートリポジトリに再公開します。

```
$ pkgrecv -s http://test -d http://remote:10000 \
-m all-versions library/zlib
```



例6 リポジトリから依存関係を取得する

/export/repo にあるリポジトリから、パッケージ editor/gnu-emacs とそのすべての依存関係を受け取ります。

```
$ pkgrev -s /export/repo -d /local/repo -r editor/gnu-emacs
```

例7 追加パッケージおよび変更された内容の取得

まだ存在しないすべてのパッケージおよびすべての変更済みの内容を、<http://pkg.oracle.com/solaris/release/> にあるリポジトリから /export/repoSolaris11 にあるリポジトリに受信します。

```
$ pkgrev -s http://pkg.oracle.com/solaris/release/ \
-d /export/repoSolaris11 -m all-timestamps '*'
```

まだ存在しないすべてのパッケージおよびすべての変更済みの内容を、<http://pkg.oracle.com/solaris/support/> にあるセキュアなリポジトリから /export/repoSolaris11 にあるリポジトリに受信します。

```
$ pkgrev -s http://pkg.oracle.com/solaris/support/ \
-d /export/repoSolaris11 -m all-timestamps \
--key /var/pkg/ssl/Oracle_Solaris_11_Support.key.pem \
--cert /var/pkg/ssl/Oracle_Solaris_11_Support.certificate.pem '*'
```

例8 パッケージアーカイブを作成する

<http://example.com:10000> にあるリポジトリから、パッケージ editor/gnu-emacs とそのすべての依存関係を含むパッケージアーカイブを作成します。

```
$ pkgrev -s http://example.com:10000 -d /my/emacs.p5p -a \
-r editor/gnu-emacs
```

例9 パッケージをアーカイブからリポジトリにコピーする

パッケージアーカイブ内のすべてのパッケージを /export/repo にある既存のリポジトリにコピーします。

```
$ pkgrev -s /my/archive.p5p -d /export/repo '*'
```

## 環境変数

次の環境変数がサポートされています。

|          |                                                                                  |
|----------|----------------------------------------------------------------------------------|
| PKG_DEST | 取得されたパッケージを保存する先のディレクトリのパス、またはパッケージがコピーされるリポジトリまたはパッケージアーカイブのファイルシステムのパスまたは URI。 |
| PKG_SRC  | パッケージ取得先の pkg(5) リポジトリまたはパッケージアーカイブの場所を表す URI またはファイルシステムのパス。                    |
| TMPDIR   | プログラム実行中に一時データが格納されるディレクトリの絶対パス。設定されていない場合、一時データはデフォルトで /var/tmp に格納されます。        |

- 終了ステータス
- 次の終了ステータスが返されます。
- 0

コマンドが成功しました。
- 1

エラーが発生した。
- 2

無効なコマンド行オプションが指定された。
- 3

複数の操作が要求されましたが、それらの一部のみが成功しました。
- 99

予期しない例外が発生しました。

属性

次の属性については、`attributes(5)` を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目

[pkgrepo\(1\)](#), [pkgsend\(1\)](#), [pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>

名前 pkgrepo – Image Packaging System リポジトリ管理ユーティリティー

形式

```

/usr/bin/pkgrepo create [--version ver] uri_or_path

/usr/bin/pkgrepo add-publisher -s repo_uri_or_path publisher ...

/usr/bin/pkgrepo get [-F format] [-H] [-p publisher ...]
 -s repo_uri_or_path [section/property ...]

/usr/bin/pkgrepo info [-F format] [-H] [-p publisher ...]
 -s repo_uri_or_path

/usr/bin/pkgrepo list [-F format] [-H] [-p publisher ...]
 -s repo_uri_or_path [pkg_fmri_pattern ...]

/usr/bin/pkgrepo rebuild [-p publisher ...]
 -s repo_uri_or_path [--no-catalog] [--no-index]

/usr/bin/pkgrepo refresh [-p publisher ...]
 -s repo_uri_or_path [--no-catalog] [--no-index]

/usr/bin/pkgrepo remove [-n] [-p publisher ...]
 -s repo_uri_or_path pkg_fmri_pattern ...

/usr/bin/pkgrepo set [-p publisher] -s repo_uri_or_path
 section/property=[value]

/usr/bin/pkgrepo set [-p publisher] -s repo_uri_or_path
 section/property=(value) ...

/usr/bin/pkgrepo help

/usr/bin/pkgrepo version

```

機能説明

pkgrepo を使用すると、pkg(5) パッケージリポジトリの作成および管理を行うことができます。パッケージリポジトリは、pkg や, pkgsend または pkgrecv などの公開クライアントがパッケージデータを格納したり取得したりできるようにするための、定義済みの一連のディレクトリおよびファイルです。さらに、パッケージリポジトリへのネットワークベースのアクセスが必要な場合、pkg.depotd は、クライアントにパッケージデータを格納したり取得したりするためのリポジトリへのアクセスを提供できます。

オプション サポートしているオプションは、次のとおりです。

```

-?
--help

```

使用方法に関するメッセージを表示します。

サブコマンド サポートされているサブコマンドは次のとおりです。

```

pkgrepo create [--version ver] uri_or_path

```

指定された場所に pkg(5) リポジトリを作成します。

このサブコマンドは、ファイルシステムベースのリポジトリでのみ使用できます。

#### --version

指定されたバージョンと互換性がある形式でリポジトリを作成します。デフォルトでは、バージョン4のリポジトリが作成されます。サポートされるバージョンは次のとおりです。

- 3    1つのパブリッシャー、カタログバージョン1、および検索バージョン1でのパッケージの格納をサポートします。
- 4    複数のパブリッシャー、カタログバージョン1、および検索バージョン1でのパッケージの格納をサポートします。

**pkgrepo add-publisher -s *repo\_uri\_or\_path* *publisher* ...**

指定された発行元をリポジトリに追加します。新しいパブリッシャーにパッケージや内容は含まれていません。

このサブコマンドは、バージョン4のファイルシステムベースのリポジトリでのみ使用できます。

**pkgrepo get [-F *format*] [-H] [-p *publisher* ...] -s *repo\_uri\_or\_path* [ *section/property* ...]**

リポジトリまたはその発行元のプロパティ情報を表示します。

デフォルトでは、各プロパティとその値が個別の行に出力されます。空のASCII文字列値は、二重引用符(")のペアで表されます。ASCII文字列値内の次の Bourne シェルのメタキャラクタと、改行、スペース、およびタブは、バックスラッシュ文字(\)でエスケープする必要があります。

; & ( ) | ^ < > \ " ' ,

発行元とリポジトリプロパティの表示例については、「使用例」セクションを参照してください。

プロパティのリスト、および各プロパティの目的と値については、下の **set** サブコマンドを参照してください。

#### -F *format*

代替出力形式を指定します。*format* の値は、**tsv** (Tab Separated Values)、**json** (単一行としての JavaScript Object Notation)、または **json-formatted** (読みやすい形式にされた JavaScript Object Notation) にできます。

#### -H

一覧からヘッダーを省略します。

#### -p *publisher*

指定された発行元のプロパティ情報を表示します。特殊な値 **all** により、すべてのパブリッシャーのプロパティが表示されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***

指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

***section/property***

*publisher/prefix* や *repository/version* などの、指定されたプロパティの値のみを表示します。プロパティの完全な一覧については、**set** サブコマンドを参照してください。

**pkgrepo info [-F *format*] [-H] [-p *publisher ...*] -s *repo\_uri\_or\_path***

リポジトリで認識されているパッケージ発行元のリストを表示します。このリストには、パブリッシャーごとのパッケージの数、パブリッシャーのパッケージデータが最後に更新された日時、およびパブリッシャーのパッケージデータのステータス (現在処理されているかどうかなど) が含まれます。

**-F *format***

代替出力形式を指定します。*format* の値は、*tsv* (Tab Separated Values)、*json* (単一行としての JavaScript Object Notation)、または *json-formatted* (読みやすい形式にされた JavaScript Object Notation) にできます。

**-H**

一覧からヘッダーを省略します。

**-p *publisher***

指定された発行元のデータのみを表示します。指定されていない場合は、すべてのパブリッシャーのデータが表示されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***

指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

**pkgrepo list [-F *format*] [-H] [-p *publisher ...*] -s *repo\_uri\_or\_path* [*pkg\_fmri\_pattern ...*]**

指定された *pkg\_fmri\_pattern* にマッチする *repo\_uri\_or\_path* リポジトリのパッケージを一覧表示します。パターンが指定されない場合、リポジトリのすべてのパッケージが一覧表示されます。

デフォルトの出力では、最初の列にパッケージのパブリッシャーの名前が含まれます。2 番目の列にはパッケージの名前が含まれます。3 番目の列には、パッケージのステータスを示すフラグが含まれます。ステータス列の *o* の値は、パッケージが廃止されていることを示します。ステータス列の *r* の値は、パッケージの名前が変更されたことを示します (廃止の形態の 1 つです)。4 番目の列には、パッケージのリリースおよびブランチのバージョンが含まれます。リリースバージョンとブランチバージョンについては、**pkg(5)** を参照してください。

**-F *format***

代替出力形式を指定します。*format* の値は、*tsv* (Tab Separated Values)、*json* (単一行としての JavaScript Object Notation)、または *json-formatted* (読みやすい形式にされた JavaScript Object Notation) にできます。

**-H**

一覧からヘッダーを省略します。

**-p *publisher***

指定された発行元のパッケージのみを表示します。指定されていない場合は、すべてのパブリッシャーのパッケージが表示されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***

指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

**pkgrepo rebuild [-p *publisher* ...] -s *repo\_uri\_or\_path* [- -no-catalog] [--no-index]**

リポジトリ内に見つかったすべてのカタログ、検索、およびその他のキャッシュされた情報を破棄し、それをリポジトリの現在の内容に基づいて再作成します。

**-p *publisher***

指定された発行元についてのみ操作を実行します。指定されていない場合や、特殊な値 *all* が指定されている場合は、すべてのパブリッシャーについて操作が実行されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***

指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

**--no-catalog**

パッケージデータを再構築しません。

**--no-index**

検索インデックスを再構築しません。

**pkgrepo refresh [-p *publisher* ...] -s *repo\_uri\_or\_path* [- -no-catalog] [--no-index]**

リポジトリ内に見つかった新しいパッケージをすべてカタログ化し、すべての検索インデックスを更新します。これは、遅延公開 (*pkgsend* の *--no-catalog* または *--no-index* オプション) で使用されることを目的にしています。

**-p *publisher***

指定された発行元についてのみ操作を実行します。指定されていない場合や、特殊な値 *all* が指定されている場合は、すべてのパブリッシャーについて操作が実行されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***

指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

**--no-catalog**  
新しいパッケージを追加しません。

**--no-index**  
検索インデックスを更新しません。

**pkgrepo remove [-n] [-p *publisher* ...] -s *repo\_uri\_or\_path* *pkg\_fmri\_pattern* ...**  
リポジトリから、指定されたパターンに一致するパッケージ(これらのパッケージが参照している、ほかのどのパッケージでも使用されていないすべてのファイルを含む)を削除します。

注-関連するパブリッシャーのすべての検索インデックスデータが削除されます。

このサブコマンドは、ファイルシステムベースのリポジトリでのみ使用できます。

注意-この操作は元に戻せません。また、ほかのクライアントがそのリポジトリにアクセスしている間に使用すべきではありません。使用すると、それらのクライアントが取得操作中に失敗する可能性があります。

**-n**  
パッケージの変更は行わずに試しに操作を実行します。終了する前に、削除されるパッケージのリストが表示されます。

**-p *publisher***  
指定された発行元の一致するパッケージのみを削除します。指定されていない場合は、すべてのパブリッシャーの一致するパッケージがすべて削除されます。このオプションは複数回指定できます。

**-s *repo\_uri\_or\_path***  
指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

**pkgrepo set [-p *publisher*] -s *repo\_uri\_or\_path* *section/property*=[*value*] ...**  
**set [-p *publisher*] -s *repo\_uri\_or\_path* *section/property*=(*[value]*) ...**  
リポジトリまたは発行元の指定されたプロパティの値を設定します。

このサブコマンドは、ファイルシステムベースのリポジトリでのみ使用できます。

**-p *publisher***  
指定された発行元のプロパティデータのみを設定します。パブリッシャーがまだ存在しない場合は、追加されます。特殊な値 **all** を使用すると、すべての発行元のプロパティを設定できます。

**-s *repo\_uri\_or\_path***  
指定された URI またはファイルシステムのパスにあるリポジトリを操作します。

プロパティーと値は、次のいずれかの形式を使用して指定できます。

*section/property=*

プロパティー値をクリアします。

*section/property= value*

プロパティー値を指定された値に置き換えます。

*section/property=( value1 value2 valueN)*

プロパティー値を値のリストに置き換えます。

リポジトリバージョン3および4の場合は、リポジトリの次のプロパティーを設定できます。

**publisher/prefix** デフォルトのパブリッシャーの名前を表す文字列。最初の文字はa-z、A-Z、または0-9である必要があります。文字列の残りの部分には、文字0-9、-、.、a-z、およびA-Zのみを含めることができます。この値は、複数のパブリッシャーのパッケージが存在するとき、またはパッケージがリポジトリに公開されているが、パブリッシャーが指定されていないときに使用されるべきパブリッシャーを示します。

リポジトリバージョン3および4の場合は、リポジトリ内の個々のパブリッシャーの次のプロパティーを設定できます。

**publisher/alias** リポジトリの構成データを使用してパブリッシャーを追加するときにクライアントが使用すべきデフォルトの別名を表す文字列。最初の文字はa-z、A-Z、または0-9である必要があります。文字列の残りの部分には、文字0-9、-、.、a-z、およびA-Zのみを含めることができます。

**repository/collection\_type** このリポジトリで提供されるパッケージのタイプを示す値 **core** または **supplemental** を持つことができます。

**core** タイプは、このリポジトリにはリポジトリ内のパッケージによって宣言されたすべての依存関係が含まれていることを示します。**core** タイプは主に、オペレーティングシステムリポジトリに使用されます。

**supplemental** タイプは、このリポジトリには、別のリポジトリ内に配置されているパッケージに依存するか、またはそれらの



|                                          |                                                                             |
|------------------------------------------|-----------------------------------------------------------------------------|
|                                          | パッケージとともに使用されるためのパッケージが含まれていることを示します。                                       |
| <code>repository/description</code>      | リポジトリの目的と内容を説明した標準テキストの段落。                                                  |
| <code>repository/detailed_url</code>     | リポジトリに関する追加情報を提供するドキュメント (Web ページなど) の場所を表す URI。                            |
| <code>repository/legal_uris</code>       | リポジトリに関する追加の使用条件情報を提供するドキュメントの場所 (URI) のリスト。                                |
| <code>repository/mirrors</code>          | リポジトリのパッケージ内容のコピーを含むが、パッケージのメタデータは含まないリポジトリの場所 (URI) のリスト。                  |
| <code>repository/name</code>             | リポジトリの名前を含む標準テキスト文字列。                                                       |
| <code>repository/origins</code>          | リポジトリのパッケージのメタデータと内容の完全なコピーを含むリポジトリの場所 (URI) のリスト。                          |
| <code>repository/refresh_seconds</code>  | クライアントが、更新されたパッケージデータがあるかどうかを調べるためにリポジトリをチェックする際に、各更新チェックのあとに待つべき秒数を表す整数値。  |
| <code>repository/registration_uri</code> | リポジトリへのアクセスのための資格を取得するために使用する必要のあるリソースの場所を表す URI。この 1 つの例に、登録 Web ページがあります。 |
| <code>repository/related_uris</code>     | ユーザーが関心を持っている可能性があるパッケージを含むリポジトリの場所 (URI) のリスト。                             |

ここでは文書化されていないが、`get` サブコマンドの出力に一覧表示されるプロパティは内部使用のために予約されているため、設定しないようにしてください。

#### `pkgrepo help`

使用方法に関するメッセージを表示します。

#### `pkgrepo version`

`pkg(5)` システムのバージョンを識別する一意の文字列を表示します。`version` 操作によって生成される値はソート可能ではないため、等しいかどうかを超えて安全に比較することはできません。

## 使用例

例1 パッケージリポジトリを作成する

```
$ pkgrepo create /my/repository
```

例2 情報を表示する

パブリッシャーのサマリーと、リポジトリ内のパッケージの数を表示します。

```
$ pkgrepo info -s /my/repository
PUBLISHER PACKAGES STATUS UPDATED
example.com 5 online 2011-07-22T18:09:09.769106Z
$ pkgrepo info -s http://pkg.oracle.com/solaris/release/
PUBLISHER PACKAGES STATUS UPDATED
solaris 3941 online 2010-11-12T19:24:25.967246Z
```

例3 カタログと検索データを再構築する

リポジトリのカタログと検索データを再構築します。

```
$ pkgrepo rebuild -s /my/repository
```

例4 カタログと検索データを再表示する

リポジトリのカタログと検索データを再表示します。

```
$ pkgrepo refresh -s /my/repository
$ pkgrepo refresh -s http://example.com/repository
```

例5 すべてのリポジトリプロパティーを表示する

```
$ pkgrepo get -s /export/repoSolaris11
SECTION PROPERTY VALUE
publisher prefix solaris
repository description Local\ copy\ of\ the\ Oracle\ Solaris\ 11\ repository
repository name Oracle\ Solaris\ 11
repository version 4
$ pkgrepo get -s http://pkg.oracle.com/solaris/release/
SECTION PROPERTY VALUE
deployment content s11_11-11
deployment pubdate 20111102T222051Z
publisher prefix solaris
repository version 4
```

例6 すべてのパブリッシャープロパティーを表示する

```
$ pkgrepo get -s http://pkg.oracle.com/solaris/release/ -p all
PUBLISHER SECTION PROPERTY VALUE
solaris publisher alias
solaris publisher prefix solaris
solaris repository collection-type core
solaris repository description This\ repository\ serves\ the\ Oracle\
Solaris\ 11\ Package\ repository.
```

例6 すべてのパブリッシャープロパティを表示する (続き)

```
solaris repository legal-uris ()
solaris repository mirrors (http://pkg-cdn1.oracle.com/solaris.release/)
solaris repository name Oracle\ Solaris\ 11\ Package\ Repository
solaris repository origins ()
solaris repository refresh-seconds
solaris repository registration-uri ""
solaris repository related-uris ()
```

例7 デフォルトのパブリッシャーを設定する

```
$ pkgrepo set -s /my/repository publisher/prefix=example.com
```

例8 パブリッシャープロパティを設定する

```
$ pkgrepo set -s /my/repository -p example.com \
repository/origins=http://example.com/repository
```

例9 新しいパブリッシャーをリポジトリに追加する

```
$ pkgrepo add-publisher -s /my/repository example.com
```

終了ステータス 次の終了ステータスが返されます。

- 0 コマンドが成功しました。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。
- 3 複数の操作が要求されましたが、それらの一部のみが成功しました。
- 4 変更は行われませんでした。処理がありません。
- 99 予期しない例外が発生しました。

属性 次の属性については、attributes(5)を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目 [pkg\(1\)](#)、[pkgrecv\(1\)](#)、[pkgsend\(1\)](#)、[pkg.depotd\(1M\)](#)、[pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>

|        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前     | pkgsend – Image Packaging System の公開クライアント                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 形式     | <pre> /usr/bin/pkgsend [options] command [cmd_options] [operands]  /usr/bin/pkgsend generate [-T pattern] [--target file] source ...  /usr/bin/pkgsend publish [-b bundle ...] [-d source ...] [-s repo_uri_or_path] [-T pattern] [--no-cata</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明   | <p>pkgsend では、パッケージのマニフェストを使用して、新しいパッケージと新しいパッケージのバージョンをイメージパッケージングリポジトリに公開できます。リポジトリを作成または管理するには、pkgrepo(1)を参照してください。既存のリポジトリ内でパッケージからパッケージアーカイブを作成するには、pkgrecv(1)を参照してください。パッケージのマニフェストの詳細は、pkg(5)を参照してください。</p> <p>pkgsend 操作後に、リポジトリに対して pkgrepo refresh または pkgrepo rebuild を実行し、検索インデックスを構築します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| オプション  | <p>サポートしているオプションは、次のとおりです。</p> <pre> -? --help </pre> <p>使用方法に関するメッセージを表示します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| サブコマンド | <p>サポートされているサブコマンドは次のとおりです。</p> <p>pkgsend generate [-T pattern] [--target file] source ...</p> <p>各 source (SVR4 パッケージ、ディレクトリ、tar ファイルなど)を読み取り、その source を標準出力に表示するマニフェストを生成します。出力マニフェストで、file および dir アクションの所有者は root に設定され、グループは bin に設定されています。</p> <p>これで、出力されたマニフェストに注釈を加え、pkgdepend を使用して依存関係を追加または分析し、pkglint を使用してその正当性を検証してから publish サブコマンドに渡すことができますようになります。</p> <p>サポートされているソースを次に示します。</p> <ul style="list-style-type: none"> <li>■ ファイルシステム形式の SVR4 パッケージ</li> <li>■ データストリーム形式の SVR4 パッケージ</li> <li>■ tar ファイル</li> <li>■ ディレクトリ</li> </ul> <p>ソース内のファイルのベース名が -T で指定されたパターンに一致している場合、ファイルのタイムスタンプがそのファイルのアクションに追加されます。pattern は、次のシェルマッチング規則を使用します。</p> <p>*       すべてと一致します。</p> <p>?       任意の単一文字と一致します。</p> <p>[seq]   seq 内にある任意の文字と一致します。</p> |

![seq] *seq* 内にない文字と一致します。

指定されたソースがディレクトリ内にある場合、単一のiノードに対して複数のパス名があると、ファイルのアクションをハードリンクのアクションと明確に区別できません。通常、ファイルシステム調査で最初に見つかったものがファイルとして扱われ、残りのものがハードリンクとして扱われます。この処理は、ファイルシステムの実装に応じて自由に設定できます。ファイルとして扱うパス名を指定するには、各パス名を引数として `--target` オプションに渡します。このオプションはほかの種類のソースには影響しません。これは、ソースにはパス名がファイルであるかハードリンクであるかを示す機能があるためです。

SVR4 パッケージがソースとして提供されている場合、pkgsend は、クラスアクションスクリプトを持つファイルが存在せず、プリインストールスクリプト、ポストインストールスクリプト、削除前スクリプト、または削除後スクリプトが存在することを確認します。manifest クラスを使用してインストールされた SMF マニフェストがある場合、例外が作成されます。BASEDIR は、すべての再配置可能パスから削除されます。

SVR4 DESC パラメータは、pkg.description 値に変換されます。SVR4 NAME パラメータは、pkg.summary 値に変換されます。

```
pkgsend publish [-b bundle ...] [-d source ...] [-s repo_uri_or_path] [-T pattern]
[--no-catalog] [manifest ...]
```

指定されたパッケージマニフェストを使用するパッケージをターゲットパッケージリポジトリに公開します。これにより、指定されたソースからそのパッケージ用のファイルが取得されます。複数のマニフェストが指定されている場合、それらのマニフェストは指定された順序で追加されます。マニフェストが指定されていない場合、stdin からマニフェストが読み取られます。

指定しない場合、pkgsend publish はビルドバージョンをパッケージ FMRI に追加します。publish ツールは、タイムスタンプも (UTC の現在時間) をパッケージ FMRI に追加します。パッケージ FMRI のバージョン文字列については、pkg (5) のマニュアルページを参照してください。

#### **-b bundle**

マニフェスト内でファイルを検索するときに、指定されたバンドルを検索するソースの一覧に追加します。バンドルは、tar ファイルや SVR4 パッケージなどのソースです。このオプションが複数回指定されている場合、ソースはコマンド行に表示される順序で検索されます。`-b` と `-d` の両方が指定されている場合、`-d` のソースが最初に検索されます。サポートされているバンドルとその使用方法の詳細は、前述の `generate` サブコマンドを参照してください。

#### **-d source**

マニフェスト内でファイルを検索するときに、指定されたディレクトリを検索するソースの一覧に追加します。このオプションが複数回指定されている

場合、ソースはコマンド行に表示される順序で検索されます。サポートされているソースとその使用方法の詳細は、上記の `generate` サブコマンドを参照してください。

**-s *repo\_uri\_or\_path***

特定の URI またはファイルシステムパスに存在しているリポジトリにパッケージが公開されます。公開についての制限事項と推奨事項の詳細は、次の「注意事項」のセクションを参照してください。また、「環境変数」のセクションも参照してください。

**--no-catalog**

パッケージを発行元のカatalogに追加しません。パブリッシャーのカatalogの更新は連続で実行されるため、複数のパッケージを一度に公開する場合には常にこのオプションを使用することを推奨します。公開が完了すると、`pkgrepo` の `refresh` サブコマンドを使用して、新しいパッケージを各発行元のカatalogに追加できます。

-T オプションの説明については、前述の `generate` サブコマンドを参照してください。

## 環境変数

**PKG\_REPO**     公開先リポジトリのパスまたは URI です。

## 使用例

### 例1 パッケージの生成と公開

`pkgsend generate` を使用してパッケージを作成し、そのパッケージを公開します。

```
$ pkgsend generate /path/to/proto > /path/to/manifests/foo.p5m
```

`example.com` パブリッシャーのパッケージ FMRI を、`foo.p5m` の先頭に追加します。

```
set name=pkg.fmri value=pkg://example.com/foo@1.0
```

結果として生成されるマニフェストは、次のようになります。

```
set name=pkg.fmri value=pkg://example.com/foo@1.0
dir group=sys mode=0755 owner=root path=usr
dir group=bin mode=0755 owner=root path=usr/bin
file usr/bin/foo group=bin mode=0555 owner=root path=usr/bin/foo

$ pkgsend publish -s http://example.com:10000 -d /path/to/proto \
/path/to/manifests/foo.p5m
```

### 例2 簡易パッケージの作成と公開

次の行を含むパブリッシャー `example.com` に対してマニフェストを作成します。

```
set name=pkg.fmri value=pkg://example.com/foo@1.0-1
file /exdir/foo mode=0555 owner=root group=bin path=usr/bin/foo
```

パッケージを公開します。

```
$ pkgsend publish -s http://example.com:10000 -d /exdir
```

例3 既存のマニフェストの使用  
ファイルシステムベースの公開と既存のマニフェストを使用してパッケージを公開します。

```
$ pkgsend publish -s /tmp/example_repo -d /tmp/pkg_files \
/tmp/pkg_manifest
```

終了ステータス 次の終了ステータスが返されます。

- 0 コマンドが成功しました。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。
- 99 予期しない例外が発生しました。

属性 次の属性については、attributes(5)を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目 [pkgdepend\(1\)](#)、[pkgrepo\(1\)](#)、[pkg.depotd\(1M\)](#)、[pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project/pkg/>

注意事項 公開プロトコルの制限事項により、サイズが 128MB を超えるパッケージファイルを個別に公開する場合には、ファイルシステムベースの公開を使用する必要があります。ファイルシステムベースの公開は、リポジトリのアクセス制御が必要な場合にも推奨されます。

ファイルシステムベースの公開を使用する場合、公開が完了して Web インタフェースまたは検索応答で変更が反映されたあとに、公開先リポジトリを提供している pkg.depotd プロセスを再開する必要があります。詳細は、pkg.depotd(1M)を参照してください。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgsign – Image Packaging System の署名ユーティリティー                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 形式    | <pre>/usr/bin/pkgsign [-a <i>hash_algorithm</i>]<br/>                  [-c <i>path_to_signing_certificate</i>]<br/>                  [-i <i>path_to_intermediate_cert</i>] ...<br/>                  [-k <i>path_to_private_key</i>] [-n] -s <i>path_or_uri</i><br/>                  [--help] [--no-index] [--no-catalog]<br/>                  (<i>fmri pattern</i>) ...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 機能説明  | pkgsign は、指定されたキーと証明書を使用して署名アクションを追加することにより、リポジトリ内で所定の FMRI のマニフェストを直接更新します。変更されたパッケージは、元のタイムスタンプを保持します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>--help</b><br/>使用方法に関するメッセージを表示します。</p> <p><b>-a <i>hash_algorithm</i></b><br/>デフォルトの署名アルゴリズムではなく、<i>hash_algorithm</i> 署名アルゴリズムが使用されます。デフォルトの署名アルゴリズムは <i>rsa-sha256</i> です。サポートされている署名アルゴリズムは、<i>rsa-sha256</i>、<i>rsa-sha384</i>、<i>rsa-sha512</i>、<i>sha256</i>、<i>sha384</i>、<i>sha512</i> です。ハッシュアルゴリズムを指定するだけの署名アルゴリズムを使用すると、署名の値は、パッケージのマニフェストのハッシュになります。<i>rsa</i> とハッシュアルゴリズムを指定する署名アルゴリズムを使用すると、署名の値は、提供された非公開鍵で署名が追加されたマニフェストのハッシュになります (<i>-c</i> オプションと <i>-k</i> オプションを参照)。</p> <p><b>-c <i>path_to_signing_certificate</i></b><br/>証明書 <i>path_to_signing_certificate</i> を、このアクションで署名の値を検証するとき使用する証明書として追加します。<i>-c</i> オプションは、<i>-k</i> オプションでのみ使用されます。</p> <p><b>-i <i>path_to_intermediate_cert</i></b><br/>証明書 <i>path_to_intermediate_cert</i> を、<i>-c</i> への引数として指定された証明書 <i>path_to_signing_certificate</i> を検証するとき使用する証明書として追加します。複数の証明書を追加するには、<i>-i</i> を複数回指定します。</p> <p><b>-k <i>path_to_private_key</i></b><br/><i>path_to_private_key</i> に格納された非公開鍵を使用してマニフェストに署名を追加します。<i>-k</i> オプションは、<i>-c</i> オプションでのみ使用されます。<i>-k</i> が設定されていない場合、署名の値はマニフェストのハッシュになります。</p> <p><b>-n</b><br/>評価実行を行い、リポジトリは一切変更しません。</p> <p><b>-s <i>path_or_uri</i></b><br/><i>path_or_uri</i> にあるリポジトリ内のパッケージに署名を追加します。</p> |



- `--no-index`  
署名付きマニフェストが再公開されたあとに、リポジトリの検索インデックスを更新しません。
- `--no-catalog`  
署名付きマニフェストが再公開されたあとに、リポジトリのカタログを更新しません。

使用例

- 例1 マニフェストのハッシュ値を使用した署名の追加  
マニフェストのハッシュ値を使用して、`http://localhost:10000` に公開されたパッケージに署名を追加します。多くの場合、これはテストに役立ちます。  
  
\$ **pkgsign -s http://localhost:10000 -a sha256 \**  
**example\_pkg@1.0,5.11-0:20100626T030108Z**
- 例2 キーと証明書を使用した署名の追加  
`rsa-sha384` を使用して、`/foo/bar` 内のファイルリポジトリに公開されたパッケージに署名を追加し、マニフェストをハッシュ化して署名を追加します。署名キーは `/key/usr2.key` に、関連の証明書は `/key/usr2.cert` に、その証明書の検証に必要な証明書は `/icerts/usr1.cert` にあります。  
  
\$ **pkgsign -s file:///foo/bar/ -a rsa-sha384 \**  
**-k /key/usr2.key -c /key/usr2.cert -i /icerts/usr1.cert \**  
**example\_pkg@1.0,5.11-0:20100626T031341Z**

終了ステータス

- 次の終了ステータスが返されます。
- 0 コマンドが成功しました。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。
- 3 複数の操作が要求されましたが、それらの一部のみが成功しました。
- 99 予期しない例外が発生しました。

属性

次の属性については、`attributes(5)` を参照してください。

| 属性タイプ       | 属性値         |
|-------------|-------------|
| 使用条件        | package/pkg |
| インタフェースの安定性 | 不確実         |

関連項目

[pkg\(1\)](#), [pkgrecv\(1\)](#), [pkgsend\(1\)](#), [pkgrepo\(1\)](#), [pkg\(5\)](#)  
  
<http://hub.opensolaris.org/bin/view/Project+pkg/>

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pkgtrans – パッケージ形式の変換                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 形式    | pkgtrans [-inosg] [-k <i>keystore</i> ] [-a <i>alias</i> ] [-P <i>passwd</i> ] <i>device1 device2</i> [ <i>pkginst</i> ]...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明  | <p>pkgtrans ユーティリティーは、インストール可能なパッケージをある形式から別の形式に変換します。次のような変換が行われます。</p> <ul style="list-style-type: none"><li>■ ファイルシステム形式からデータストリームへ</li><li>■ ファイルシステム形式から署名付きのデータストリームへ</li><li>■ データストリームからファイルシステム形式へ</li><li>■ あるファイルシステム形式から別のファイルシステム形式へ</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| オプション | <p>このコマンドのオプションと引数は次のとおりです。</p> <p>-a <i>alias</i>      friendlyName が示す別名に関連付けられた公開鍵証明書と対応する秘密鍵を使用します。詳細は、KEYSTORE LOCATIONS and KEYSTORE AND CERTIFICATE FORMATS in <a href="#">pkgadd(1M)</a>を参照してください。</p> <p>-g              結果として得られたデータストリームに署名します。</p> <p>-i              <a href="#">pkginfo(4)</a> ファイルと <a href="#">pkgmap(4)</a> ファイルのみをコピーします。</p> <p>-k <i>keystore</i>    キーストアを使用して、署名の生成に使用する秘密鍵を取得します。指定しなかった場合は、-a で指定した秘密鍵を見つけるためにデフォルトの場所が検索されます。別名を指定せず、キーストアに複数の鍵が存在する場合は、pkgtrans が異常終了します。検索の場所と形式の詳細については、KEYSTORE LOCATIONS and KEYSTORE AND CERTIFICATE FORMATS in <a href="#">pkgadd(1M)</a>を参照してください。</p> <p>                root 以外のユーザーが実行した場合、証明書検索のデフォルトベースディレクトリは ~/.pkg/security になります。ここで、~ は pkgtrans を起動したユーザーのホームディレクトリです。</p> <p>-n              このパッケージのインスタンスがすでに存在する場合は、宛先デバイス上にパッケージの新しいインスタンスを作成します。作成するインスタンスの最大数は、<a href="#">pkginfo(4)</a> ファイルの MAXINST 変数で指定されます。</p> <p>-o              宛先デバイス上の同じインスタンスを上書きします。パッケージインスタンスがすでに存在する場合、そのインスタンスは上書きされます。</p> <p>-P <i>passwd</i>      キーストアの復号化に使用するパスワードを指定します。このオプションの引数の構文の詳細については、PASS PHRASE ARGUMENTS in <a href="#">pkgadd(1M)</a>を参照してください。</p> <p>-s              パッケージをファイルシステムではなくデータストリームとして <i>device2</i> に書き込むよう指定します。デフォルトの動作では、両方の形式をサポートするデバイスにはファイルシステム形式で書き込みま</p> |

す。

|       |                |                                                                                                                                                                                                                      |
|-------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| オペランド | <i>device1</i> | ソースデバイスを指定します。このデバイス上の1つ以上のパッケージを変換して、 <i>device2</i> に格納します。下記の「デバイス識別子」を参照してください。                                                                                                                                  |
|       | <i>device2</i> | 宛先デバイスを指定します。変換されたパッケージはこのデバイスに格納されます。下記の「デバイス識別子」を参照してください。                                                                                                                                                         |
|       | <i>pkginst</i> | <i>device1</i> 上の変換対象となるパッケージインスタンスを1つ以上指定します。トークン <code>all</code> を使用すると、すべてのパッケージを指定できます。 <code>pkginst.*</code> を使用すると、1つのパッケージのすべてのインスタンスを指定できます。パッケージを定義しなかった場合は、デバイス上のすべてのパッケージがプロンプトに表示され、変換対象を選択するよう求められます。 |

シェルによっては、アスタリスク文字 (\*) が特殊な意味を持つことがあり、\* をエスケープしなければならない場合があります。C シェルの場合、\* は単一引用符 (') で囲むか、またはバックスラッシュ (\) を前に付ける必要があります。

**デバイス識別子**     `pkgtrans`、`pkgadd(1M)`、`pkgchk(1M)` などのパッケージツールでは、パッケージが格納されているデバイスを指定することによってパッケージの場所を指定できます。次の一覧は、パッケージを格納して取り出すことができるデバイスのタイプです。ただし、ソースデバイスと宛先デバイスを同じデバイスにすることはできません。

|        |                                                                                                                                                                                                                                                  |
|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| デバイス   | デバイス識別子をデバイスとして指定することで、パッケージを文字デバイスまたはブロック型デバイスに格納できます。このデバイスタイプのよくある例は、取り外し可能磁気テープの <code>/dev/rmt/0</code> です。 <code>pkgtrans</code> では、通常のファイルシステムのファイルをストリーム形式で生成することもできます。これは、文字デバイスや Web サーバーへの格納、または <code>pkgadd(1M)</code> への入力に適しています。 |
| ディレクトリ | ファイルシステムのディレクトリへの絶対パスを指定することにより、パッケージをディレクトリに格納できます。パッケージの内容は、指定したディレクトリ内のディレクトリに格納されます。パッケージのディレクトリ名は、 <code>pkginfo(4)</code> ファイルでの PKG の指定と同じにする必要があります。このタイプのデバイス指定の例は、 <code>/export/packages</code> です。                                   |

## 使用例

例1 `/tmp` 上のパッケージを変換する

次の例では、`/tmp` 上のパッケージ `pkg1` および `pkg2` をデータストリーム形式に変換しています。

```
example% pkgtrans -s /tmp /tmp/datastream.pkg pkg1 pkg2
```

例2 署名付きパッケージを作成する

次の例では、pkg1 および pkg2 から署名付きパッケージを作成し、\$PASS 環境変数からパスワードを読み取っています。

```
example% pkgtrans -sg -k /tmp/keystore.p12 -a foo \
-p env:PASS /tmp /tmp/signedpkg pkg1 pkg2
```

例3 パッケージデータストリームを変換する

次の例では、パッケージのデータストリームをファイルシステム形式のパッケージに変換しています。

```
example% pkgtrans /tmp/pkg1.pkg ~/tmp pkg1
```

- 環境変数
- pkginfo(4) ファイルに MAXINST 変数が設定され、パッケージインスタンスの最大数が宣言されています。
- 終了ステータス
- 0 正常終了。  
>0 エラーが発生した。
- 属性
- 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

| 属性タイプ       | 属性値          |
|-------------|--------------|
| 使用条件        | package/svr4 |
| インタフェースの安定性 | 確実           |

関連項目

pkginfo(1), pkgmk(1), pkgparam(1), pkgproto(1), installf(1M), pkgadd(1M), pkgask(1M), pkgrm(1M), removef(1M), pkginfo(4), pkgmap(4), attributes(5), largefile(5)

『Application Packaging Developer’s Guide』

- 注意事項
- デフォルトでは、パッケージのインスタンスが宛先デバイスにすでに存在する場合、pkgtrans はそのパッケージのどのインスタンスも変換しません。-n オプションを使用すると、このパッケージのインスタンスがすでに存在する場合に新しいインスタンスが作成されます。-o オプションを使用すると、このパッケージのインスタンスがすでに存在する場合にそのインスタンスが上書きされます。宛先デバイスがデータストリームである場合は、どちらのオプションも使用できません。

パッケージコマンドは、largefile(5) を認識します。これらのコマンドは、2G バイトより大きなファイルを、より小さなファイルと同様に処理します。現在の実装では、pkgadd(1M)、pkgtrans1、およびその他のパッケージコマンドは最大 4G バイトのデータストリームを処理できます。

|         |                                                                                                                                                                                                                                                                                                                                                                                               |
|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pklogin_finder – 証明書のユーザーへのマッピング                                                                                                                                                                                                                                                                                                                                                              |
| 形式      | /usr/lib/pam_pkcs11/pklogin_finder [debug] [config_file=filename]                                                                                                                                                                                                                                                                                                                             |
| 機能説明    | <p>pklogin_finder は、pam_pkcs11 ライブラリインフラストラクチャーを使用して、PKCS#11 で規定された証明書を対話形式でユーザーにマップします。</p> <p>pklogin_finder は、pam_pkcs11(5) PAM モジュールと同じ構成ファイルと引数を使用します。また、定義済みのマッパーモジュールを読み込み、見つかった証明書とユーザーログイン間のマップを探し出します。</p>                                                                                                                                                                          |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <p><code>config_file=filename</code>      構成ファイルを設定します。</p> <p>デフォルト値は /etc/security/pam_pkcs11/pam_pkcs11.conf です。</p> <p><code>debug</code>      デバッグ出力を有効にします。</p> <p>デフォルトはデバッグなしです。</p> <p>pam_pkcs11(5) と同じ構成ファイルが使用されるため、pam_pkcs11 のすべてのオプションを使用できます。一部のオプションは PAM 以外の環境では意味がないため、無視されます。一部のマッパーオプション (mapfile、ignorecase) も証明書の内容に影響しないため、無視されます。</p> |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0      正常終了。</p> <p>pkcs11_inspect は、ログイン名を stdout に出力して終了します。</p> <p>1      エラーが発生した。</p> <p>ユーザーマッピングエラーが見つかった。</p> <p>2      エラーが発生した。</p> <p>一致するユーザーが見つからなかった。</p>                                                                                                                                                                                            |

|     |                                                                                                                                                                                                                                                                                     |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 使用例 | <p>例1 pklogin_finder を使用する</p> <p>次の例では、オプションを指定せずに pklogin_finder コマンドを実行しています。</p> <pre>% pkcs11_inspect</pre> <p>例2 オプションを指定して pklogin_finder を使用する</p> <p>次の例では、オプション指定して pkcs_finder コマンドを実行しています。</p> <pre>% pklogin_finder debug config_file=\${HOME}/.pam_pkcs11.conf</pre> |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

ファイル `/etc/security/pam_pkcs11/pam_pkcs11.conf`

著者 Juan Antonio Martinez (jonsito@teleline.es)

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                    |
|-------------|----------------------------------------|
| 使用条件        | library/security/pam/module/pam-pkcs11 |
| インタフェースの安定性 | 不確実                                    |

関連項目 `pkcs11_inspect(1)`, `attributes(5)`, `pam_pkcs11(5)`

PAM-PKCS11 User Manual, [http://www.opensc-project.org/pam\\_pkcs11](http://www.opensc-project.org/pam_pkcs11)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pktool – 証明書と鍵の管理                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 形式    | pktool [-f <i>option_file</i> ] [-i] <i>subcommand subcommand_options</i> ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明  | <p>pktool コマンドを使用すると、PKCS#11 トークン (暗号化フレームワーク)、Netscape Security Services (NSS) トークン、および OpenSSL の標準的なファイルベースのキーストアを含む、複数のキーストア内の証明書と鍵を管理できます。</p> <p>pktool は、証明書失効リスト (Certificate Revocation List、CRL) の一覧表示、削除、およびインポートもサポートしています。pktool は、CRL の作成、CRL への署名の追加、および CRL のエクスポートはサポートしていません。PKCS#11 キーストアの CRL サポートはファイルベースです。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| オプション | <p>次のコマンドオプションがサポートされています。</p> <p><b>-f <i>option_file</i></b><br/>       コマンド行にオプションを入力する代わりに、ファイルにオプションを設定できます。</p> <p>pktool では多数のサブコマンドと関連オプションをコマンド行で指定する可能性があるため、ユーザーの利便性を高めるためにこのオプションが用意されています。</p> <p><i>option_file</i> は、1 行に 1 つのオプションまたは値のペアという形式になります。</p> <p><i>option_file</i> の例を次に示します。</p> <pre>list keystore=nss dir=/export/foo objtype=key</pre> <p><b>-i</b><br/>       gencert サブコマンドおよび gencsr サブコマンドの subject-DN を対話形式で指定できます。-i を指定すると、subject-DN を作成するためのデータを入力するよう求めるプロンプトが表示されます。</p> <p>-i オプションの使用例を次に示します。</p> <pre>Country Name (2 letter code) [US]:US State or Province Name (full name) [Some-State]:CA Locality Name (eg, city) []:Menlo Park Organization Name (eg, company):Sun Microsystems Inc. Organizational Unit Name (eg, section):OPG Common Name (eg, YOUR name):John Smith Email Address []: john.smith@sun.com</pre> <p>その結果作成される subject-DN は次のとおりです。</p> |

```
"C=US, ST=CA, L=Menlo Park, O=Sun Microsystems Inc.,\
OU=OPG, emailAddress=john.smith@sun.com, \
CN=John Smith"
```

サブコマンド サポートされているサブコマンドは次のとおりです。

#### delete

delete サブコマンドの書式は次のとおりです。

```
pktool delete [token=token[:manuf[:serial]]]
 [objtype=private|public|both]
 [label=object-label]

pktool delete keystore=pkcs11
 objtype=cert[:public | private | both]]
 [token=token[:manuf[:serial]]]
 [label=cert-label]
 [serial=hex-serial-number]
 [issuer=issuer-DN]
 [subject=subject-DN]

pktool delete keystore=nss
 objtype=cert
 [subject=subject-DN]
 [issuer=issuer-DN]
 [serial=hex-serial-number]
 [nickname=cert-nickname]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]

pktool delete keystore=nss
 objtype=crl
 [nickname=cert-nickname]
 [subject=subject-DN]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]

pktool delete keystore=pkcs11
 objtype=key[:public | private | both]]
 [token=token[:manuf[:serial]]]
 [label=key-label]

pktool delete keystore=pkcs11
 objtype=crl
 infile=input-fn
```



```
pktool delete keystore=file
 objtype=cert
 [infile=input-fn]
 [dir=directory-path]
 [serial=hex-serial-number]
 [issuer=issuer-DN]
 [subject=subject-DN]
```

```
pktool delete keystore=file
 objtype=key
 [infile=input-fn]
 [dir=directory-path]
```

```
pktool delete keystore=file
 objtype=crl
 infile=input-fn
```

証明書、鍵、または証明書失効リスト (CRL) を削除します。

PKCS#11 トークンから個人証明書または秘密鍵を削除する場合は、正しい個人識別番号 (Personal Identification Number、PIN) を入力して PKCS#11 に対する認証を行うよう求めるプロンプトが表示されます。

#### download

download サブコマンドの書式は次のとおりです。

```
pktool download url=url_str
 [objtype=crl|cert]
 [http_proxy=proxy_str]
 [outfile=output-fn]
 [dir=directory-path]
```

指定された URL の場所から CRL ファイルまたは証明書ファイルをダウンロードします。ファイルが正常にダウンロードされると、ダウンロードした CRL または証明書ファイルの有効性が確認されます。CRL または証明書の期限が切れている場合、download は警告を表示します。

#### export

export サブコマンドの書式は次のとおりです。

```
pktool export [token=token[:manuf[:serial]]]
 outfile=output-fn
```

```
pktool export keystore=pkcs11
 outfile=output-fn
 [objtype=cert|key]
 [label=label]
 [subject=subject-DN]
 [issuer=issuer-DN]
```

```
[serial=hex-serial-number]
[outformat=pem|der|pkcs12|raw]
[token=token[:manuf[:serial]]]
```

```
pktool export keystore=nss
 outfile=output-fn
 [subject=subject-DN]
 [issuer=issuer-DN]
 [serial=hex-serial-number]
 [nickname=cert-nickname]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]
 [outformat=pem|der|pkcs12]
```

```
pktool export keystore=file
 certfile=cert-input-fn
 keyfile=key-input-fn
 outfile=output-pkcs12-fn
```

PKCS#11 トークンの内容、あるいは NSS トークンまたはファイルベースのキーストアに含まれる証明書を、指定されたファイルに保存します。

## gencert

gencert サブコマンドの書式は次のとおりです。

```
pktool gencert [-i] keystore=nss
 label=cert-nickname
 subject=subject-DN
 serial=hex_serial_number
 [altname=[critical:]subjectAltName,subjectAltName...]
 [keyusage=[critical:]usage,usage...]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]
 [keytype=rsa | ec [curve=ECC Curve Name] \
 [hash= md5 | sha1 | sha224 | sha256 | sha384 | sha512]
 [keytype=dsa [hash=sha1 | sha224]
 [keylen=key-size]
 [trust=trust-value]
 [eku=[critical:]EKU_name,...]
 [listcurves]
 [lifetime=number-hour|number-day|number-year]

pktool gencert [-i] [keystore=pkcs11]
 label=key/cert-label
 subject=subject-DN
 serial=hex_serial_number
 [altname=[critical:]subjectAltName,subjectAltName...]
```

```

[keyusage=[critical:]usage,usage...]
[token=token[:manuf[:serial]]]
[keytype=rsa | ec [curve=ECC Curve Name] \
 [hash=md5 | sha1 | sha224 | sha256 | sha384 | sha512]]
[keytype=dsa [hash=sha1 | sha224 | sha256]]
[keylen=key-size]
[eku=[critical:]EKU_name,...]
[listcurves]
[lifetime=number-hour|number-day|number-year]

```

```

pktool gencert [-i] keystore=file
 outcert=cert-fn
 outkey=key-fn
 subject=subject-DN
 serial=hex_serial_number
 [altname=[critical:]subjectAltName,subjectAltName...]
 [keyusage=[critical:]usage,usage...]
 [format=der|pem]
 [keytype=rsa [hash=md5 | sha1 | sha224 | sha256 | sha384 | sha512]]
 [keytype=dsa [hash=sha1 | sha224 | sha256]]
 [keylen=key-size]
 [eku=[critical:]EKU_name,...]
 [lifetime=number-hour|number-day|number-year]

```

自己署名付き証明書を生成し、証明書とそれに関連付けられた秘密鍵を指定のキーストアにインストールします。

`gencert` を実行すると、トークンベースのキーストア用の PIN を入力するように求めるプロンプトが表示されます。

## gencsr

`gencsr` サブコマンドの書式は次のとおりです。

```

pktool gencsr [-i] keystore=nss
 nickname=key-nickname
 outcsr=csr-fn
 subject=subject-DN
 [altname=[critical:]subjectAltName,subjectAltName...]
 [keyusage=[critical:]usage,usage...]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]
 [keytype=rsa | ec [curve=ECC Curve Name] \
 [hash= md5 | sha1 | sha224 | sha256 | sha384 | sha512]]
 [keytype=dsa [hash=sha1 | sha224]]
 [keylen=key-size]
 [format=pem|der]
 [eku=[critical:]EKU_name,...]
 [listcurves]

```

```

pktool gencsr [-i] keystore=pkcs11
 label=key-label
 outcsr=csr-fn
 subject=subject-DN
 [altname=[critical:]subjectAltName,subjectAltName...]
 [keyusage=[critical:]usage,usage...]
 [token=token[:manuf[:serial]]]
 [keytype=rsa | ec [curve=ECC Curve Name] \
 [hash=md5 | sha1 | sha224 | sha256 | sha384 | sha512]]
 [keylen=key-size]
 [format=pem|der]
 [eku=[critical:]EKU_name,...]
 [listcurves]

pktool gencsr [-i] keystore=file
 outcsr=csr-fn
 outkey=key-fn
 subject=subject-DN
 [altname=[critical:]subjectAltName,subjectAltName...]
 [keyusage=[critical:]usage,usage...]
 [dir=directory-path]
 [keytype=rsa [hash=md5 | sha1 | sha224 | sha256 | sha384 | sha512]]
 [keytype=dsa [hash=sha1 | sha224 | sha256]]
 [keylen=key-size]
 [format=pem|der]
 [eku=[critical:]EKU_name,...]

```

PKCS#10 の証明書署名要求 (CSR) ファイルを作成します。この CSR を承認局 (Certifying Authority、CA) に送信すると、署名を取得できます。gencsr サブコマンドを実行すると、トークンベースのキーストア用の PIN を入力するように求めるプロンプトが表示されます。

## genkey

genkey サブコマンドの書式は次のとおりです。

```

pktool genkey [keystore=pkcs11]
 label=key-label
 [keytype=aes|arcfour|des|3des|generic]
 [keylen=key-size (for aes, arcfour, or \
 generic keytypes only)]
 [token=token[:manuf[:serial]]]
 [sensitive=y|n]
 [extractable=y|n]
 [print=y|n]

pktool genkey keystore=nss
 label=key-label
 [keytype=aes|arcfour|des|3des|generic]

```

```

[keylen=key-size (for aes, arcfour, or \
 generic keytypes only)]
[token=token[:manuf[:serial]]]
[dir=directory-path]
[prefix=DBprefix]

```

```

pktool genkey keystore=file
outkey=key-fn
[keytype=aes|arcfour|des|3des|generic]
[keylen=key-size (for aes, arcfour, \
 or generic keytypes only)]
[print=y|n]

```

指定されたキーストアに对称鍵を生成します。genkey サブコマンドを実行すると、トークンベースのキーストア用の PIN を入力するように求めるプロンプトが表示されます。

### genkeypair

genkeypair サブコマンドの書式は次のとおりです。

```

pktool genkeypair keystore=nss
label=key-nickname
[token=token[:manuf[:serial]]]
[dir=directory-path]
[prefix=DBprefix]
[keytype=rsa|dsa|ec [curve=ECC Curve Name]]
[keylen=key-size]
[listcurves]

```

```

pktool genkeypair [keystore=pkcs11]
label=key-label
[token=token[:manuf[:serial]]]
[keytype=rsa|dsa|ec [curve=ECC Curve Name]]
[keylen=key-size]
[listcurves]

```

```

pktool genkeypair keystore=file
outkey=key_filename
[format=der|pem]
[keytype=rsa|dsa]
[keylen=key-size]

```

### import

import サブコマンドの書式は次のとおりです。

```

pktool import [token=token>[:manuf>[:serial>]]]
infile=input-fn

```

```

pktool import [keystore=pkcs11]
infile=input-fn

```

```
label=object-label
[keytype=aes|arcfour|des|3des|generic]
[sensitive=y|n]
[extractable=y|n]
[token=token[:manuf[:serial]]]
[objtype=cert|key]
```

```
pktool import keystore=pkcs11
objtype=crl
infile=input-fn
outcrl=output-crl-fn
outformat=pem|der
```

```
pktool import keystore=nss
objtype=cert
infile=input-fn
label=cert-label
[token=token[:manuf[:serial]]]
[dir=directory-path]
[prefix=DBprefix]
[trust=trust-value]
```

```
pktool import keystore=nss
objtype=crl
infile=input-fn
[verifycrl=y|n]
[token=token[:manuf[:serial]]]
[dir=directory-path]
[prefix=DBprefix]
```

```
pktool import keystore=file
infile=input-fn
outkey=output-key-fn
outcert=output-key-fn
[outformat=pem|der]
```

```
pktool import keystore=file
objtype=crl
infile=input-fn
outcrl=output-crl-fn
outformat=pem|der
```

証明書、鍵、またはCRLを、指定の入力ファイルから指定のキーストアに読み込みます。

inittoken

inittoken サブコマンドの書式は次のとおりです。

```
pktool inittoken [slotid=slot number]
[currlabel=token[:manuf[:serial]]]
```

```
[newlabel=new token label]
```

このコマンドは、C\_InitToken API を使用して PKCS#11 トークンを初期化します。トークンを検索するには、トークンのデフォルトラベルを指定することをお勧めします。オプションで、newlabel 引数を使用すると、トークンに新しいラベルを割り当てることができます。newlabel がいない場合、トークンラベルは変更されません。このコマンドを続行するために、セキュリティ責任者 (Security Officer, SO) の PIN を入力するよう求めるプロンプトが表示されます。

## list

list サブコマンドの書式は次のとおりです。

```
pktool list [token=token[:manuf[:serial]]]
 [objtype=private|public|both]
 [label=label]

pktool list [keystore=pkcs11]
 [objtype=cert[:public | private | both]]
 [token=token[:manuf[:serial]]]
 [label=cert-label]
 [serial=hex-serial-number]
 [issuer=issuer-DN]
 [subject=subject-DN]

pktool list [keystore=pkcs11]
 objtype=key[:public | private | both]]
 [token=token[:manuf[:serial]]]
 [label=key-label]
 [keyvalue=y|n]

pktool list keystore=pkcs11
 objtype=crl
 infile=input-fn

pktool list keystore=nss
 objtype=cert
 [subject=subject-DN]
 [issuer=issuer-DN]
 [serial=hex-serial-number]
 [nickname=cert-nickname]
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]

pktool list keystore=nss
 objtype=key
 [token=token[:manuf[:serial]]]
 [dir=directory-path]
 [prefix=DBprefix]
```

```
pktool list keystore=file
 objtype=cert
 [infile=input-fn]
 [dir=directory-path]
 [serial=hex-serial-number]
 [issuer=issuer-DN]
 [subject=subject-DN]
```

```
pktool list keystore=file
 objtype=key
 [infile=input-fn]
 [dir=directory-path]
 [keyvalue=y|n]
```

証明書、鍵、または証明書失効リスト (CRL) を一覧表示します。PKCS#11 トークンの個人証明書または秘密鍵を表示すると、正しい PIN を入力して PKCS#11 トークンに対する認証を行うよう求めるプロンプトが表示されます。

#### setpin

setpin サブコマンドの書式は次のとおりです。

```
pktool setpin keystore=nss
 [token=token]
 [dir=directory-path]
 [prefix=DBprefix]
```

```
pktool setpin [keystore=pkcs11]
 [token=token[:manuf[:serial]]]
 [usertype=user | so]
```

PKCS#11 または NSS トークンに対してユーザーを認証するために使用されるパスフレーズを変更します。パスフレーズには、NULL 以外の 1-256 文字の長さの任意の文字列を指定できます。

古いパスフレーズがある場合、setpin は、そのパスフレーズを入力するよう求めるプロンプトを表示します。古いパスフレーズが一致すると、pktool は、新しいパスフレーズを 2 回入力するよう求めるプロンプトを表示します。2 回入力した新しいパスフレーズが一致すると、それがトークンの最新のパスフレーズになります。

Sun Software PKCS#11 ソフトトークンキーストア (デフォルト) の場合は、setpin コマンドを使用してオブジェクトストアのパスフレーズを変更するときに、デフォルトのパスフレーズ `changeme` を古いパスフレーズとして使用する必要があります。この操作は、パスフレーズを初期化して、新しく作成したトークンオブジェクトストアに設定するために必要です。



PKCS#11 ベースのトークンに対して `usertype=so` オプションを指定すると、通常ユーザーの PIN ではなく、セキュリティ責任者 (Security Officer、SO) ユーザーの PIN が変更されます。デフォルトでは、`usertype` は `user` とみなされます。

### signcsr

signcsr サブコマンドの書式は次のとおりです。

```
signcsr keystore=pkcs11
 signkey=label (label of key to use for signing)
 csr=CSR_filename
 serial=serial_number_hex_string_for_final_certificate
 outcert=filename_for_final_certificate
 issuer=issuer-DN
 [store=y|n] (store the new cert in NSS DB, default=n)
 [outlabel=certificate label]
 [format=pem|der] (certificate output format)
 [subject=subject-DN] (override the CSR subject name)
 [altname=subjectAltName,subjectAltName...] (add subjectAltName)
 [keyusage=[critical:]usage,...] (add key usage bits)
 [eku=[critical:]EKU_Name,...] (add Extended Key Usage)
 [lifetime=number-hour|number-day|number-year]
 [token=token[:manuf[:serial]]]

signcsr keystore=file
 signkey=filename
 csr=CSR_filename
 serial=serial_number_hex_string_for_final_certificate
 outcert=filename_for_final_certificate
 issuer=issuer-DN
 [format=pem|der] (certificate output format)
 [subject=subject-DN] (override the CSR subject name)
 [altname=subjectAltName,subjectAltName...] (add a subjectAltName)
 [keyusage=[critical:]usage,...] (add key usage bits)
 [lifetime=number-hour|number-day|number-year]
 [eku=[critical:]EKU_Name,...] (add Extended Key Usage)

signcsr keystore=nss
 signkey=label (label of key to use for signing)
 csr=CSR_filename
 serial=serial_number_hex_string_for_final_certificate
 outcert=filename_for_final_certificate
 issuer=issuer-DN
 [store=y|n] (store the new cert in NSS DB, default=n)
 [outlabel=certificate label]
 [format=pem|der] (certificate output format)
 [subject=subject-DN] (override the CSR subject name)
 [altname=subjectAltName,subjectAltName...] (add a subjectAltName)
 [keyusage=[critical:]usage,...] (add key usage bits)
 [eku=[critical:]EKU_Name,...] (add Extended Key Usage)
 [lifetime=number-hour|number-day|number-year]
```

```
[token=token[:manuf[:serial]]]
[dir=directory-path]
[prefix=DBprefix]
```

## tokens

tokens サブコマンドの書式は次のとおりです。

```
pktool tokens
```

tokens サブコマンドは、表示可能なすべての PKCS#11 トークンを一覧表示します。

-?

このサブコマンドの書式は次のとおりです。

```
pktool -?
pktool --help
```

-? オプションを使用すると、使用方法やヘルプ情報を表示できます。--help の機能は -? と同じです。

## 使用法

pktool サブコマンドは次のオプションをサポートします。

altname=[critical:]*subjectAltName,subjectAltName...*

証明書の被認証者代替名。altname オプションのあとに指定する引数は、タグ=値の形式にします。有効なタグは、IP、DNS、EMAIL、URI、DN、KRB、UPN、および RID です。altname 文字列の直前に critical: というワードを付加すると、SubjectAltName 拡張が critical としてマークされます。

クリティカルなフラグは、すべての altname で共有されるため、最初の被認証者代替名の先頭に配置してください。

altname タグ (IP、EMAIL、DN など) の区切り記号は、コンマ (,) です。DN タイプ内のコンポーネントの区切り記号は、セミコロン (;) です。

例 1: *subjectAltName* 拡張に IP アドレスを追加する: altname="IP=1.2.3.4" 例 2: *subjectAltName* 拡張に電子メールアドレスを追加し、critical としてマークする: altname="critical:EMAIL=first.last@company.com"

例 3: 複数の被認証者代替名 IP アドレス、メールアドレス、および識別名を *subjectAltName* 拡張に追加し、クリティカルのマークを付けます。

```
altname="critical:IP=1.2.3.4,EMAIL=first.last@company.com,\
DN=C=US;O=Oracle;OU=Security;CN=John Smith"
```

currlabel=token label

このオプションは、inittoken コマンドとのみ使用できます。これは、初期化するデフォルトトークンを見つけるために使用します。使用するトークン名の書式についての詳細は、token オプションを参照してください。

`curve=Elliptic_Curve_Name`

このオプションは、X.509 証明書または証明書署名要求を生成する場合、または楕円曲線の鍵ペアを生成する場合に使用される楕円曲線パラメータを指定します。

次の曲線名がサポートされています。

```
secp112r1, secp112r2, secp128r1, secp128r2, secp160k1
secp160r1, secp160r2, secp192k1, secp192r1, secp224k1
secp224r1, secp256k1, secp256r1, secp384r1, secp521r1
sect113r1, sect113r2, sect131r1, sect131r2, sect163k1
sect163r1, sect163r2, sect193r1, sect193r2, sect233k1
sect233r1, sect239k1, sect283k1, sect283r1, sect409k1
sect409r1, sect571k1, sect571r1, c2pnb163v1, c2pnb163v2
c2pnb163v3, c2pnb176v1, c2tnb191v1, c2tnb191v2, c2tnb191v3
c2pnb208w1, c2tnb239v1, c2tnb239v2, c2tnb239v3, c2pnb272w1
c2pnb304w1, c2tnb359v1, c2pnb368w1, c2tnb431r1, prime192v2
prime192v3
```

曲線名の一覧は、`gencert`、`gencsr`、または `genkeypair` サブコマンドで `listcurves` オプションを使用した場合にも表示されます。

`dir=directory_path`

要求したオブジェクトが格納されている NSS データベースディレクトリまたは OpenSSL キーストアディレクトリを指定します。

`eku=[critical:]EKU_Name,[critical:]EKU_Name, ...]`

証明書または証明書要求に追加する X.509v3 拡張の拡張鍵用途の値を指定します。

`EKU_Name` に

は、`serverAuth`、`clientAuth`、`codeSigning`、`emailProtection`、`ipsecEndSystem`、`ipsecTransport`、または `scLogon` のいずれかを指定します。

例:

```
eku=KPClientAuth,clientAuth
```

`extractable=y | n`

PKCS#11 トークン内に生成される対称鍵を抽出可能にするかどうかを指定します。有効な値は `y` および `n` です。デフォルト値は `y` です。

`format=pem | der | pkcs12`

`gencert` サブコマンドの場合、このオプションは OpenSSL などファイルベースのキーストアにのみ適用されます。このオプションは、作成される鍵ファイルまたは証明書ファイルの出力形式を指定するために使用します。有効な書式は `pem` または `der` です。デフォルトの書式は `pem` です。

gencsr サブコマンドの場合、このオプションは CSR ファイルの出力エンコード形式を指定します。有効な書式は pem または der です。デフォルトの書式は pem です。

hash=md5 | sha1 | sha224 | sha256 | sha384 | sha512

gencert および gencsr サブコマンドの場合、このオプションを使用すると、呼び出し側は X.509 証明書の署名を生成するために使用するハッシュアルゴリズムを指定できます。このオプションは、NSS または PKCS#11 のキーストアを使用して EC または RSA ベースの証明書を作成するときに使用できます。OpenSSL のファイルベースのキーストアを使用する場合、楕円曲線はサポートされません。

infile=input-fn

list および delete サブコマンドで objtype=cert および keystore=file の場合、これらのサブコマンドの証明書ファイル名を指定します。import サブコマンドの場合、インポートするファイル名を指定します。list、delete、および import サブコマンドで objtype=crl の場合、これらのサブコマンドの入力 CRL ファイル名を指定します。

issuer=issuer-DN

証明書の発行者を指定します。

keylen=key-size

生成する秘密鍵または対称鍵のサイズ(ビット数)を指定します。

gencert および gencsr サブコマンドの場合、デフォルトの鍵長は 1024 ビットです。

genkey サブコマンドの場合、AES アルゴリズムを使用して生成する対称鍵の最小ビット数と最大ビット数は、それぞれ 128 と 256 です。ARCFOUR アルゴリズムを使用する場合の最小ビット数と最大ビット数は、それぞれ 8 と 2048 です。一般的な秘密鍵の場合、最小ビット数は 8 で最大ビット数は任意です。AES、ARCFOUR、または一般的な秘密鍵のデフォルトの鍵長は 128 です。DES 鍵または 3DES 鍵の場合、鍵長は固定であるため、このオプションを指定しても無視されます。

keystore=nss | pkcs11 | file

基本となるキーストアのタイプ(NSS トークン、PKCS#11 トークン、またはファイルベースのプラグイン)を指定します。

keytype=rsa | dsa | ec | aes | arcfour | des | 3des | generic

生成する秘密鍵または対称鍵のタイプを指定します。

gencert および gencsr サブコマンドの場合、有効な秘密鍵のタイプは rsa、ec、または dsa です。デフォルトの鍵のタイプは rsa です。

genkey サブコマンドの場合、有効な対称鍵のタイプは aes、arcfour、des、3des、または generic です。デフォルトの鍵のタイプは aes です。

```
keyusage=[critical:]usage,usage,usage,...
```

Key Usage strings:

```
* digitalSignature
* nonRepudiation
* keyEncipherment
* dataEncipherment
* keyAgreement
* keyCertSign
* cRLSign
* encipherOnly
* decipherOnly
```

例 1: cert (または csr) を使用して証明書または CRL 以外のデータを署名および検証するため (digitalSignature)、および暗号化鍵以外のデータを暗号化および復号化するため (dataEncipherment) に使用できるように KeyUsage を設定する:

```
keyusage=digitalSignature,dataEncipherment
```

例 2: 上記 (例 1) と同じであるが、さらに critical ビットを設定する:

```
keyusage=critical:digitalSignature,dataEncipherment
```

keyvalue=y | n

このオプションは、ファイルベースおよび PKCS#11 鍵ストア内の対称鍵の鍵値を 16 進数形式で表示します。有効な値は y および n です。デフォルト値は n です。

label=key-label | cert-label

gencert サブコマンドの場合、このオプションは、PKCS#11 トークン内の秘密鍵と自己署名証明書のラベルを指定します。

gencsr サブコマンドの場合、このオプションは、PKCS#11 トークン内の秘密鍵のラベルを指定します。

list サブコマンドの場合、このオプションは、PKCS#11 トークン内の X.509 証明書 (objtype=key の場合) または秘密鍵 (objtype=cert の場合) のラベルを指定して、一覧を絞り込みます。

delete サブコマンドの場合、このオプションは、X.509 証明書 (objtype=key の場合) または秘密鍵 (objtype=cert の場合) のラベルを指定して、指定したオブジェクトを PKCS#11 トークンから削除します。

listcurves

サポートされている楕円曲線名の一覧を表示します。このオプションは、gencert、gencsr、または genkeypair サブコマンドでのみ使用できます。

lifetime=number-hour | number-day | number-year

証明書の有効期間を指定します。証明書の有効期間

は、number-hour、number-day、または number-year で指定できます。指定できる書式は 1 つだけです。デフォルトは 1-year です。たとえば、このオプションは、lifetime=1-hour、lifetime=2-day、lifetime=3-year のようになります。

**newlabel=token label**

このオプションは、**inittoken** コマンドとのみ使用できます。これは、初期化するトークンに割り当てられたラベルを変更するために使用します。使用するトークン名の書式についての詳細は、**token** オプションを参照してください。

**nickname=cert-nickname**

**gencert** サブコマンドの場合、このオプションを使用して、NSS キーストアの証明書のニックネームを指定する必要があります。

**list** サブコマンドの場合、このオプションは、内容を表示する NSS トークン内の証明書のニックネームを指定します。**delete** サブコマンドの場合、NSS トークンから CRL を削除するには、このオプションを使用して発行者の証明書のニックネームを指定します。**delete** サブコマンドで NSS トークンから証明書を削除する場合、このオプションは証明書のニックネームを指定します。**import** サブコマンドの場合、指定した入力ファイルを NSS トークンにインポートするには、このオプションを使用して、生成される証明書のニックネームを指定する必要があります。

**objtype=cert | key | crl**

オブジェクトのクラス (**cert**、**key**、または **crl**) を指定します。**download** サブコマンドの場合、このオプションを指定しないと、デフォルトの **crl** が使用されます。

**objtype=public | private | both**

オブジェクトのタイプ (非公開オブジェクトまたは公開オブジェクト、あるいはその両方) を指定します。このオプションは、**list** および **delete** サブコマンドで **objtype=key** と指定した場合の PKCS#11 トークンにのみ適用されます。デフォルト値は **public** です。

**list** サブコマンドの場合、**label** オプションをこのオプションと組み合わせると、鍵のリストをさらに絞り込むことができます。**delete** サブコマンドの場合、このオプションを使用すると、削除する鍵を公開鍵または秘密鍵だけに絞り込むことができます。また、**label** オプションを省略すると、公開または非公開、あるいはその両方のタイプの鍵をすべて削除対象として指定できます。**objtype** パラメータに指定する **public**、**private**、および **both** は、以前のバージョンの **pktool** コマンドとの互換性を維持するために PKCS#11 キーストアにのみ適用されます。

**outcert=cert-fn**

書き込み先の出力証明書ファイル名を指定します。このオプションは、OpenSSL などファイルベースのプラグインに対して指定する必要があります。このオプションとともに、**outkey=key-fn** オプションも指定する必要があります。

**outcrl=output-crl-fn**

書き込み先の出力 CRL ファイル名を指定します。

**outcsr=csr-fn**

書き込み先の出力 CSR ファイル名を指定します。

**outfile=output-fn**

**export** サブコマンドの場合、このオプションは、作成される出力ファイル名を指定します。**import** サブコマンドの場合、このオプションは、証明書または CRL の出力ファイル名を指定します。OpenSSL などファイルベースのプラグインにのみ適用されます。**download** サブコマンドの場合、このオプションを指定しないと、ダウンロードされたファイルの名前は URL 文字列のベース名になります。

**outformat=pem | der | pkcs12**

**import** サブコマンドの場合、このオプションは、指定した PKCS#12 ファイルからファイルベースのプラグインに抽出される証明書または鍵の出力形式を指定します。有効な値は **pem** または **der** です。デフォルトは **pem** です。CRL を CRL ファイルベースのキーストアにインポートする場合、このオプションは、CRL の出力形式を指定します。有効な値は **pem** または **der** です。デフォルトは **der** です。**export** サブコマンドの場合、このオプションは、作成される指定の出力ファイルの形式を指定します。サポートされている書式は、**pem**、**der**、または **pkcs12** です。デフォルトは **pkcs12** です。

**outkey=key-fn**

書き込み先の出力秘密鍵ファイル名を指定します。このオプションは、ファイルベースのキーストアを使用する場合にのみ必要です。

**prefix=DBprefix**

NSS データベースの接頭辞を指定します。このオプションは NSS トークンにのみ適用されます。

**print=y | n**

このオプションは **genkey** サブコマンドで使用され、PKCS11 キーストアとファイルベースのキーストアに適用されます。**print=y** の場合、**genkey** サブコマンドは、生成された鍵の鍵値を 16 進数で 1 行に出力します。デフォルト値は **n** です。PKCS11 キーストアの場合、**sensitive=y** または **extractable=n** と指定して対称鍵を作成すると、**print** オプションを **y** に設定していても鍵値は表示されません。鍵は作成されますが、「cannot reveal the key value」という警告が表示されます。

**sensitive=y | n**

PKCS#11 トークン内に生成される対称鍵を機密扱いするかどうかを指定します。有効な値は **y** および **n** です。デフォルト値は **n** です。

**serial=hex-serial-number**

証明書に対して一意のシリアル番号を指定します。シリアル番号は 16 進数値で指定する必要があります。例: **0x0102030405060708090a0b0c0d0e0f**

**subject=subject-DN**

証明書または証明書要求に対して特定の証明書所有者を指定します。**subject=** の設定例を次に示します。



```
subject=O=Sun Microsystems Inc., \
OU=Solaris Security Technologies Group, \
L=Ashburn, ST=VA, C=US, CN=John Smith
```

`token=token[:manuf[:serial]]`

トークンラベルの末尾にスペースがある場合でも、このオプションではユーザーがスペースを入力する必要はありません。

コロンで区切られたトークン識別文字列です (`token:manuf:serial`)。いずれかの部分に文字として `:` を含める場合、バックスラッシュ (`\`) を使用してエスケープする必要があります。 `:` がない場合、文字列全体 (最大 32 文字) がトークンラベルとみなされます。 `:` が 1 つだけある場合、文字列はトークンラベルと作成者となります。 `keystore=nss` と指定した場合、このオプションを指定しないと、デフォルトで NSS 内部トークンが使用されます。 `keystore=pkcs11` と指定した場合、このオプションを指定しないと、デフォルトで `pkcs11_softtoken` が使用されます。

`trust=trust-value`

証明書の信頼属性を指定します。これは NSS 証明書専用であり、標準の NSS 構文が適用されます。

`usertype=user | so`

`setpin` コマンドの実行対象となるユーザーのタイプを指定します。デフォルトは標準ユーザーですが、`so` を指定すると、トークンのセキュリティー責任者の PIN を設定できます。

`url=url_string`

CRL または証明書ファイルをダウンロードするための URL を指定します。

`verifycrl=y | n`

CRL を NSS キーストアにインポートする場合、このオプションは、CRL の検証を実行するかどうかを指定します。有効な値は `y` および `n` です。デフォルト値は `n` です。

`http_proxy=proxy_str`

プロキシサーバーのホスト名とポート番号を指定します。書式は `http://hostname[:port]` または `hostname[:port]` です。このオプションを指定しないと、`download` サブコマンドは `http_proxy` 環境変数をチェックします。コマンド行オプションが環境変数よりも優先されます。

## 使用例

例1 自己署名証明書の生成

次の例では、証明書を作成し、コマンドで指定したキーストアに格納しています。

```
$ pktool gencert keystore=nss nickname=WebServerCert \
 subject="O=Sun Microsystems Inc., OU=Solaris Security Technologies Group, \
 L=Ashburn, ST=VA, C=US, CN=John Smith" dir=/etc/certs \
 keytype=rsa keylen=2048 hash=sha512
```



例2 証明書署名要求を生成する

次の例では、CSR を作成し、コマンドで指定したキーストアに格納しています。

```
$ pktool gencsr keystore=nss subject="O=Sun Microsystems Inc., \
OU=Solaris Security Technologies Group, L=Ashburn, ST=VA, C=US, \
CN=John Smith" keytype=rsa keylen=2048 hash=sha256 outcsr=csr.dat
```

例3 証明書をインポートする

次の例では、証明書オブジェクトを指定の入力ファイルからコマンドで指定したキーストアにインポートしています。

```
$ pktool import keystore=nss objtype=cert infile=mycert.pem \
nickname=mycert
```

終了ステータス 次の終了ステータスが返されます。

```
0
 正常終了。
>0
 エラーが発生した。
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 確実             |

関連項目 [attributes\(5\)](#), [pkcs11\\_softtoken\(5\)](#)

RSA PKCS#11 v2.11 <http://www.rsasecurity.com>

RSA PKCS#12 v1.0 <http://www.rsasecurity.com>

SECG Recommended Elliptic Curve Domain Parameters <http://www.secg.org>

|         |                                                                                                            |
|---------|------------------------------------------------------------------------------------------------------------|
| 名前      | plabel – プロセスのラベルの取得                                                                                       |
| 形式      | /usr/bin/plabel [-sS] [pid...]                                                                             |
| 機能説明    | plabel は、プロセスのラベルを取得するプロセスツールコマンドです。pid を指定しない場合は、plabel コマンドのラベルが表示されます。オプションを指定しない場合、ラベルはデフォルト形式で表示されます。 |
| オプション   | -s pid に関連付けられているラベルを短形式で表示します。<br>-S pid に関連付けられているラベル長形式で表示します。                                          |
| 終了ステータス | plabel は、終了時に次のいずれかの値を返します。<br>0 正常終了。<br>1 使用方法の誤りのためエラー終了しました。<br>2 ラベルを変換できなかった。<br>3 メモリーを割り当てられなかった。  |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                              |

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/trusted |
| インタフェースの安定性 | 下記を参照。         |

plabel ユーティリティは「確実」です。出力は非インタフェースです。

関連項目 [proc\(1\)](#), [getlabel\(3TSOL\)](#), [attributes\(5\)](#)

注意事項 このマニュアルページに記載されている機能を利用できるのは、システムが Trusted Extensions を使用して構成されている場合のみです。

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | plgrp – スレッドのホーム lgroup と lgroup アフィニティーの監視と変更                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 形式   | <pre> plgrp [-F] [-h] pid   core [/lwps] ...  plgrp [-F] -i pid[/lwps] ...  plgrp [-F] -a lgroup_list pid[/lwps] ...  plgrp [-F] [-I default   none   future]         -H lgroup_list pid[/lwps] ...  plgrp [-F] [-I default   none   future] -H lgroup_list -e command [arguments]  plgrp [-F] [-I default   none   future]         -A lgroup_list/none   weak   strong [,...] pid[/lwps] ...  plgrp [-F] [-I default   none   future]         -A lgroup_list/none   weak   strong [,...] -e command [arguments]  plgrp [-F] -I default   none   future pid[/lwps]  plgrp [-F] -I default   none   future -e command [arguments] ... </pre>                                                                                                                                                                                                                                                                      |
| 機能説明 | <p>plgrp は、1つ以上のプロセス、スレッド、または LWP のホーム lgroup と lgroup アフィニティーを表示および設定します。</p> <p>lgroup は、一連の CPU およびメモリーに類似するハードウェアデバイスを表します。これらの間には少し距離(遅延)がある場合があります。システムの各 lgroup は、一意の lgroup ID によって識別されます。lgroup は、もっとも近いリソースを見つけやすくするために階層にまとめられます。lgroups と lgroup 階層の詳細については、lgrpinfo(1) を参照してください。</p> <p>デフォルトでは、各スレッドには、作成時にホーム lgroup が割り当てられます。CPU リソースまたはメモリーリソースをスレッドに割り当てる必要がある場合、システムは、スレッドのホームにもっとも近い位置にある使用可能リソースを探すために、スレッドのホーム lgroup から順番に lgroup 階層を検索します。</p> <p>通常、スレッドのホーム lgroup は、スレッドがもっともアフィニティーを持つ lgroup です。初期状態では、システムは各スレッドのホーム lgroup を選択しますが、その lgroup に対するスレッドのアフィニティーは none に設定されたままにします。スレッドがホームではないプロセッサセットに、より強い lgroup のアフィニティーを設定した場合、スレッドが CPU に関連付けられていないかぎり、スレッドのホームはその lgroup に変更されます。ホーム lgroup のアフィニティー(もしあれば)が削除された(none に設定)場合、スレッドのホームは、そのプロセッサセットで2番目に高いアフィニティーを持つ lgroup に変更できます。</p> <p>lgroup アフィニティーの各種レベルとそのセマンティクスの詳細については、lgrp_affinity_set(3LGRP) を参照してください。</p> |

## 使用法

**lggroup** の指定 *lggroup\_list* は、次のものを 1 つ以上含むコンマで区切られたリストです。

- *lggroup\_ID*
- Range of *lggroup\_ID*s specified as  
  <start *lggroup\_ID*>-<end *lggroup\_ID*>
- all
- root
- leaves

all キーワードは、システム内のすべての *lggroup* ID を表します。root キーワードは、ルート *lggroup* の ID を表します。leaves キーワードは、すべての leaf *lggroup*、つまり、子を持たない *lggroup* の ID を表します。

**plgrp** は、スペース文字で区切られた 1 つ以上のプロセスまたはスレッドを引数に取ります。プロセスとスレッドは、**proc(1)** ツールと同様の形式で指定できます。プロセス ID は、整数の *pid* または */proc/pid* として指定できます。*/proc/pid* を使用する場合、プロセスを指定するためにシェル展開を使用できます。たとえば、システム内のすべてのプロセスを指定するには */proc/\** を使用します。プロセス ID が単独で指定された場合、**plgrp** の引数としてプロセスのすべてのスレッドが含まれます。

スレッドは、*pid/lwpid* のようにプロセス ID とスレッド ID で明示的に指定できます。プロセスの複数のスレッドは、ハイフン(-)とコンマ(,)を使用すると、一度に選択できます。たとえば、*pid/1,2,7-9* は、プロセス ID が *pid* であるプロセスのスレッド 1、2、7、8、および 9 を指定しています。

**オプション** サポートしているオプションは、次のとおりです。

- a *lggroup\_list* 指定された *lggroup\_list* の指定されたプロセスまたはスレッドの *lggroup* のアフィニティーを表示します。
- A *lggroup\_list*/none|weak|strong[,...] 指定された *lggroup\_list* の指定されたプロセスまたはスレッドのアフィニティーを設定します。  
  
*lggroup*/アフィニティーによる割り当ての、コンマで区切られたリストを指定すると、一度に複数のアフィニティーを設定できます。
- F 別のプロセスが制御していても、ターゲットプロセスを捕捉して強制実行します。-F フラグを使用するときは注意してください。2 つの制御プロセスを 1 つの犠牲プロセスに課すと混乱することがあ

ります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止していて、この `proc` ツールの適用中に何も実行していない場合のみ、安全性が確保されます。詳細は、「警告」を参照してください。

`-e`

新しいプロセスを作成し、そのプロセスに `plgrp` を適用し、指定されたコマンドと引数を実行します。

`-h`

指定されたプロセスまたはスレッド、あるいはその両方のホーム `lgroup` を取得します。オプションが何も指定されていない場合は、これがデフォルトになります。

`-H lgroup_list`

指定されたプロセスおよびスレッドのホーム `lgroup` を設定します。

これは、スレッドのホームを変更する対象の `lgroup` に、強力なアフィニティーを設定します。複数の `lgroup` が指定された場合、`plgrp` はラウンドロビン方式でスレッドのホームを `lgroup` に設定を試みます。

`-i`

指定されたプロセスまたはスレッドの `lgroup` のアフィニティーの継承を表示します。

`-I default | none | future`

指定されたプロセスまたはスレッドの `lgroup` のアフィニティーの継承を設定します。

## オペランド

次のオペランドがサポートされています。

`lwps` スレッドを指定します。「使用法」を参照してください。

`pid` プロセス ID を指定します。「使用法」を参照してください。

## 使用例

例1 シェルのホーム `lgroup` の取得

次の例では、シェルのホーム `lgroup` を取得します。

```
% plgrp $$
PID/LWPID HOME
3401/1 1
```

**例2** 複数のスレッドのホーム lgroup に対するルート lgroup の設定

次の例では、複数のスレッドのホーム lgroup にルート lgroup を設定します。

```
% plgrp -H root 'pgrep firefox'
PID/LWPID HOME
918/1 1 => 0
934/1 2 => 0
934/2 1 => 0
934/3 2 => 0
934/625 1 => 0
934/626 2 => 0
934/624 2 => 0
934/623 2 => 0
934/630 1 => 0
```

**例3** 複数スレッドのホーム lgroup としてルートの lgroup で plgrp を実行

次の例では、複数スレッドのホーム lgroup としてルートで firefox を実行します。

```
% plgrp -H root -e /usr/bin/firefox
```

**例4** lgroup 0-2 の2つのスレッドのアフィニティーの取得

次の例では、lgroup 1-2 の2つのスレッドのアフィニティーを取得します。

```
% plgrp -a 0-2 101398/1 101337/1
PID/LWPID HOME AFFINITY
101398/1 1 0-2/none
101337/1 1 0-2/none
```

**例5** lgroup のアフィニティーの設定

次の例では、lgroup のアフィニティーを設定します。

```
% plgrp -A 0/weak,1/none,2/strong 101398
PID/LWPID HOME AFFINITY
101398/1 1 => 2 0,2/none => 2/strong,0/weak
```

**終了ステータス** 次の終了ステータスが返されます。

- 0 正常終了。
- 1 構文エラー。変更は行われていません。
- 2 致命的ではないエラーまたは中断。変更が行われた箇所がある可能性があります。

**属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

コマンド構文と出力形式は「不確実」です。

#### 関連項目

[lgrpinfo\(1\)](#), [madv.so.1\(1\)](#), [pmadvise\(1\)](#), [pmap\(1\)](#), [proc\(1\)](#), [ps\(1\)](#), [prstat\(1M\)](#),  
[lgrp\\_affinity\\_get\(3LGRP\)](#), [lgrp\\_affinity\\_set\(3LGRP\)](#),  
[lgrp\\_affinity\\_inherit\\_get\(3LGRP\)](#), [lgrp\\_affinity\\_inherit\\_set\(3LGRP\)](#),  
[lgrp\\_home\(3LGRP\)](#), [liblgrp\(3LIB\)](#), [proc\(4\)](#), [attributes\(5\)](#)

#### 警告

[proc\(1\)](#) ツールと同様に、[plgrp](#) ユーティリティは、任意のオプションを指定して呼び出した場合、ターゲットプロセスの検査中にターゲットプロセスを停止して結果を報告します。

状況によっては、プロセスのデッドロックが発生する可能性があります。停止中のプロセスは何も実行できません。本稼働環境で頻繁に使用されるプロセスを短時間でも停止すると、これらのプロセスが深刻なボトルネックになったりハングアップしたりして、ユーザーがプロセスを使用できなくなることがあります。このため、本稼働環境で UNIX プロセスを停止することは避けてください。[proc\(1\)](#) を参照してください。

このツールによって停止されたプロセスは、次のように識別できる場合があります。次のコマンドを発行して:

```
/usr/bin/ps -efll
```

出力の最初の列で T を探します。ただし、[sched](#) などの特定のプロセスは、ほとんどの場合デフォルトで T ステータスを表示する可能性があります。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | plimit - 実行中のプロセスのリソース制限の取得または設定                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 形式    | <pre>plimit [-km] pid...</pre> <pre>plimit {-cdfnstv} <i>soft,hard</i>... pid...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 機能説明  | <p><i>cdfnstv</i> オプションが1つ以上指定された場合、<i>plimit</i> は、プロセス ID リスト <i>pid</i> によって識別されるプロセスの指定されたリソースのソフト (現在値) 制限およびハード (最大値) 制限を設定します。それ以外の場合、<i>plimit</i> は、プロセス ID リスト <i>pid</i> によって識別されるプロセスのリソース制限を報告します。</p> <p>プロセスのリソース制限を取得または設定できるのは、プロセスの所有者またはスーパーユーザーのみです。強い制限値を増やすことができるのは、スーパーユーザーのみです。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-k ファイルサイズを 512 バイトブロックではなくキロバイト (1024 バイト) で出力に表示します。</li> <li>-m ファイルサイズとメモリーサイズをメガバイト (1024*1024 バイト) で出力に表示します。</li> </ul> <p>残りのオプションは、指定されたリソース制限を変更するために使用します。これらは次の形式の引数を受け入れます。</p> <p><i>soft,hard</i></p> <p>ここで、<i>soft</i> は弱い (現在値) 制限、<i>hard</i> は強い (最大値) 制限を指定します。強い制限を指定しない場合は、コンマを省略できます。弱い制限が空の文字列である場合は、強い制限のみが設定されます。各制限は、リテラル文字列 <i>unlimited</i> または数値で、次の単位を付加して指定することもできます。</p> <p><i>nk</i>            <i>n</i> キロバイト</p> <p><i>nm</i>            <i>n</i> メガバイト (CPU 時間の場合は、分)</p> <p><i>nh</i>            <i>n</i> 時間 (CPU 時間の場合のみ)</p> <p><i>mm:ss</i>        分と秒 (CPU 時間の場合のみ)</p> <p>弱い制限は強い制限を超えることができません。</p> <ul style="list-style-type: none"> <li>-c <i>soft,hard</i> コアファイルサイズの制限を設定します (デフォルトの単位は 512 バイトブロック)。</li> <li>-d <i>soft,hard</i> データセグメント (ヒープ) サイズの制限を設定します (デフォルトの単位はキロバイト)。</li> <li>-f <i>soft,hard</i> ファイルサイズの制限を設定します (デフォルトの単位は 512 バイトブロック)。</li> </ul> |



- n *soft,hard*    ファイル記述子の制限を設定します (デフォルトの単位はありません)。
- s *soft,hard*    スタックセグメントのサイズの制限を設定します (デフォルトの単位はキロバイト)。
- t *soft,hard*    CPU 時間の制限を設定します (デフォルトの単位は秒)。
- v *soft,hard*    仮想メモリーサイズの制限を設定します (デフォルトの単位はキロバイト)。

オペランド      次のオペランドがサポートされています。

pid      プロセス ID リスト。

終了ステータス    plimit は、成功した場合はゼロ、失敗した場合 (該当するプロセスがない、アクセス権が拒否された、オプションが無効など) はゼロ以外の終了値を返します。

ファイル          /proc/pid/\*      プロセス情報および制御ファイル

属性              属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目          [ulimit\(1\)](#), [proc\(1\)](#), [getrlimit\(2\)](#), [setrlimit\(2\)](#), [proc\(4\)](#), [attributes\(5\)](#),

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pmadvise – プロセスへのメモリーに関するアドバイスの適用                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 形式    | pmadvise -o option[,option] [-F] [-l] [-v] pid...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 機能説明  | <p>pmadvise は、指定されたプロセスでのメモリーの使用方法に関するアドバイスを <a href="#">madvise(3C)</a> を使用して適用します。</p> <p>pmadvise では、特定の時点の特定のサブレンジにアドバイスを適用できません。pmadvise が <a href="#">madv.so.1(1)</a> と異なるのは、<a href="#">madv.so.1(1)</a> はターゲットプログラムを実行している間、指定されたタイプのすべてのセグメントにアドバイスを常時適用する点です。</p>                                                                                                                                                                                                                                                                                                                                                                                                                |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"><li>-F 別のプロセスが制御していても、ターゲットプロセスを捕捉して強制実行します。</li><li>-F オプションを使用する場合は注意してください。 <a href="#">proc(1)</a> を参照してください。</li><li>-l 未解決の動的リンカーマップ名を表示します。</li><li>-o 適用するアドバイスは次の形式で指定します。</li></ul> <pre>private=advice shared=advice heap=advice stack=advice address[:length]=advice</pre> <p>ここで、advice には次のいずれかを指定できます。</p> <pre>normal random sequential willneed dontneed free access_lwp access_many access_many_pset access_default</pre> <p>address と length は、デバイスに適用するサブレンジを指定するために設定できます。デフォルトで、address は 16 進数、length はバイト数になります。</p> <p>length が指定されておらず、開始アドレスがセグメントの先頭を指している場合、アドバイスはそのセグメントに適用されます。length に</p> |

は、K、M、G、T、P、またはEを付けて、それぞれキロバイト、メガバイト、ギガバイト、テラバイト、ペタバイト、またはエクサバイトを単位として指定できます。

- v 詳細な情報を出力します。[pmap\(1\)](#)と同様に、どのアドバイスがどこに適用されるかを出力に表示します。これは、指定した領域(たとえば、非公開、共有など)にアドバイスを適用する場合に、アドバイスが実際に適用された場所についてのフィードバックを取得するときに役に立ちます。

`pmap` は、すべての有効なオプションの処理を試みます。不正なアドレスの範囲が指定された場合は、エラーメッセージが出力され、問題のあるオプションがスキップされます。構文エラーがある場合、`pmap` は、どのオプションも処理せずに終了し、使用法に関するメッセージを出力します。

競合するアドバイスが1つの領域に指定された場合、優先順位はもっとも限定されているアドバイスからもっとも限定されていない(つまり、もっとも一般的な)アドバイスの順になります。言い換えると、特定のアドレスの範囲に指定されたアドバイスは、ヒープおよびスタックへのアドバイスより優先され、ヒープおよびスタックへのアドバイスは、非公開メモリーおよび共有メモリーへのアドバイスより優先されます。

また、次の各グループに含まれるアドバイスは、同じグループ内のほかのアドバイス規則とは互いに排他的です。

```
MADV_NORMAL, MADV_RANDOM, MADV_SEQUENTIAL
MADV_WILLNEED, MADV_DONTNEED, MADV_FREE
MADV_ACCESS_DEFAULT, MADV_ACCESS_LWP, MADV_ACCESS_MANY
```

## オペランド

次のオペランドがサポートされています。

*pid* プロセス ID。

## 使用例

例1 指定されたアドレスのセグメントにアドバイスを適用する

次の例では、指定されたアドレスのセグメントにアドバイスを適用します。

```
% pmap $$
100666: tcsh
00010000 312K r-x-- /usr/bin/tcsh
0006C000 48K rwx-- /usr/bin/tcsh
00078000 536K rwx-- [heap]
FF100000 856K r-x-- /lib/libc.so.1
FF1E6000 32K rwx-- /lib/libc.so.1
FF1EE000 8K rwx-- /lib/libc.so.1
FF230000 168K r-x-- /lib/libcurses.so.1
FF26A000 32K rwx-- /lib/libcurses.so.1
FF272000 8K rwx-- /lib/libcurses.so.1
FF280000 576K r-x-- /lib/libnsl.so.1
```

例1 指定されたアドレスのセグメントにアドバイスを適用する (続き)

```
FF310000 40K rwx-- /lib/libnsl.so.1
FF31A000 24K rwx-- /lib/libnsl.so.1
FF364000 8K rwx-- [anon]
FF370000 48K r-x-- /lib/libsocket.so.1
FF38C000 8K rwx-- /lib/libsocket.so.1
FF3B0000 176K r-x-- /lib/ld.so.1
FF3EC000 8K rwx-- /lib/ld.so.1
FF3EE000 8K rwx-- /lib/ld.so.1
FFBE6000 104K rw--- [stack]
%
% pmadvise -o 78000=access_lwp $$
%
%
```

例2 -v オプションを使用する  
次の例では、pmadvise の詳細な出力を表示します。

```
% pmadvise -o heap=access_lwp,stack=access_default -v $$
1720: -sh
00010000 88K r-x-- /usr/sbin/sh
00036000 8K rwx-- /usr/sbin/sh
00038000 16K rwx-- [heap] <= access_lwp
FF250000 24K r-x-- /lib/libgen.so.1
FF266000 8K rwx-- /lib/libgen.so.1
FF272000 8K rwx-- [anon]
FF280000 840K r-x-- /lib/libc.so.1
FF362000 32K rwx-- /lib/libc.so.1
FF36A000 16K rwx-- /lib/libc.so.1
FF390000 64K rwx-- [anon]
FF3B0000 168K r-x-- /lib/ld.so.1
FF3EA000 8K rwx-- /lib/ld.so.1
FF3EC000 8K rwx-- /lib/ld.so.1
FFBFE000 8K rw--- [stack] <= access_default
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 0 以外 エラーが発生した。

ファイル /proc/\* プロセスファイル  
/usr/prob/lib/\* proc ツールサポートファイル

## 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

コマンドの構文は「確実」です。出力形式は「不確実」です。

## 関連項目

[madv.so.1\(1\)](#), [pmap\(1\)](#), [proc\(1\)](#), [madvise\(3C\)](#), [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pmap - プロセスのアドレス空間に関する情報の表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 形式    | <pre>/usr/bin/pmap [-rslF] [-A address_range] [pid   core]...<br/>/usr/bin/pmap -L [-rslF] [-A address_range] [pid] ...<br/>/usr/bin/pmap -x [-aslF] [-A address_range] [pid   core]...<br/>/usr/bin/pmap -S [-alF] [-A address_range] [pid   core]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 機能説明  | pmap ユーティリティーは、プロセスのアドレス空間に関する情報を表示します。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-a</b> 共有マッピングの匿名およびスワップ予約を表示します。</p> <p><b>-A address_range</b> 表示するアドレス空間のサブレンジを指定します。<i>address_range</i> は、次のいずれかの形式で指定します。</p> <p><i>start_addr</i><br/>単一のアドレスを指定すると、そのアドレスを含むセグメント (-L オプションが指定されている場合は、ページ) を出力するように制限されます。指定されたアドレスがセグメントの開始アドレスに対応している場合は、-L オプションが指定されていても、常にセグメント全体が出力に含められます。</p> <p><i>start_addr,</i><br/>アドレスの後ろにコンマが付いていて、終了アドレスがない場合は、指定されたアドレスを含むセグメント以降のすべてのセグメント (-L オプションが指定されている場合は、ページ) を出力するように制限されます。</p> <p><i>start_addr,end_addr</i><br/>開始アドレスと終了アドレスによって指定されたアドレスの範囲の場合は、開始アドレスを含むセグメントまたはページから、終了アドレスを含むセグメントまたはページまでのすべてのセグメント (-L オプションが指定されている場合は、ページ) を出力するように制限されます。</p> <p><i>,end_addr</i><br/>アドレスの範囲がコンマから始まり、開始アドレスがない場合は、最初のセグメントから、指定されたアドレスを含むセグメント (-L オプションが指定されている場合は、ページ) までのすべてのセグメント (-L オプションが指定されている場合は、ページ) を出力するように制限されます。</p> <p><b>-F</b> 強制。別のプロセスが制御していても、ターゲットプロセスを捕捉します。</p> |

「使用法」を参照してください。

- l 未解決の動的リンカーマップ名を表示します。
- L 仮想メモリーを戻す物理メモリーが含まれている lgroup を出力します。
- r プロセスの予約済みアドレスを表示します。
- s HAT ページサイズ情報を表示します。
- S マッピングごとのスワップ予約情報を表示します。詳細は、「使用法」を参照してください。
- x マッピングごとの追加情報を表示します。詳細は、「使用法」を参照してください。

## 使用法

pmap ユーティリティーは、プロセスのアドレス空間に関する情報を表示します。

### プロセスマッピング

```
/usr/bin/pmap [-rslF] [-A address_range] [pid | core] ...
```

デフォルトでは、pmap は、プロセスにマップされる仮想アドレスの順序で、すべてのマッピングを表示します。マッピングサイズ、フラグ、およびマップされたオブジェクト名が表示されます。

-A オプションを使用すると、指定されたアドレスの範囲への出力を制限できます。指定されたアドレスは、セグメント境界まで切り上げまたは切り下げられ、出力にはそれらのアドレスの範囲内のセグメントが含まれます。

### プロセス Lgroup マッピング

```
/usr/bin/pmap -L [-rslF] [-A address_range] pid ...
```

-L オプションを使用すると、指定された仮想メモリーを戻す物理メモリーを含む lgroup を判別できます。-A オプションと一緒に使用すると、指定されたアドレスは、ページ境界まで切り上げまたは切り下げられ、出力はそれらのアドレスの範囲内のページに制限されます。

これは、[plgrp\(1\)](#) と組み合わせて使用すると、目的のスレッドのホーム lgroup がメモリーが配置されている場所と同じかどうか、およびスレッドにメモリーの局所性があるかどうかを確認できます。[lgrpinfo\(1\)](#) コマンドもこの pmap オプションと一緒に使用すると役に立ちます。これは、メモリーが配分される lgroup についての詳細な情報が得られる lgroup 階層、コンテンツ、および特性や、それらの lgroup 同士の関係およびその他の lgroup との関係を表示します。

また、スレッドとメモリーの配置は、[plgrp\(1\)](#)、[pmadvise\(1\)](#)、または [madv.so.1\(1\)](#) を使用して変更できます。

### プロセスの匿名またはロックされたマッピングの詳細

```
/usr/bin/pmap -x [-aslF] [-A address_range] [pid | core] ...
```

-x オプションは、マッピングごとの追加情報を表示します。このオプションを指定すると、各マッピングのサイズ、常駐物理メモリー (RSS) の量、匿名メモリーの量、およびロックされたメモリーの量が表示されます。この中には、このプロセスが原因でカーネルアドレス空間によって取得される匿名メモリーは含まれません。

### スワップ予約

```
/usr/bin/pmap -S [-aLF] [-A address_range] [pid | core] ...
```

-S オプションは、マッピングごとのスワップ予約情報を表示します。

-F フラグを使用するときは注意してください。2つの制御プロセスを1つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバugga) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の proc ツールの適用時に何も実行していない場合のみ、安全性が確保されます。

### 表示書式

--s または --L オプションが指定されている場合を除き、プロセス内の各マッピングごとに1行が出力されます。-s オプションを指定すると、各ハードウェア変換ページサイズの連続するマッピングごとに1行が出力されます。-L オプションを指定すると、同じ lgroup に属する連続するマッピングごとに1行が出力されます。-L オプションと -s オプションの両方を指定すると、同じ lgroup に属する各ハードウェア変換ページサイズの連続するマッピングごとに1行が出力されます。列ヘッダーは以降の括弧の中に示されています。

|                  |                                                   |
|------------------|---------------------------------------------------|
| 仮想アドレス (Address) | 出力の最初の列は、各マッピングの開始仮想アドレスを表しています。仮想アドレスは昇順で表示されます。 |
|------------------|---------------------------------------------------|

|                     |                         |
|---------------------|-------------------------|
| 仮想マッピングサイズ (Kbytes) | 各マッピングの仮想サイズ (K バイト単位)。 |
|---------------------|-------------------------|

|                |                                                      |
|----------------|------------------------------------------------------|
| 常駐物理メモリー (RSS) | ほかのアドレス空間と共有されるものを含む、マッピングごとに常駐する物理メモリーの量 (K バイト単位)。 |
|----------------|------------------------------------------------------|

|               |                                                                                                           |
|---------------|-----------------------------------------------------------------------------------------------------------|
| 匿名メモリー (Anon) | 指定されたマッピングに関連付けられている匿名メモリーの (システムページサイズを使用してカウントされた) ページ数。-a オプションを指定しないかぎり、ほかのアドレス空間と共有される匿名メモリーは含まれません。 |
|---------------|-----------------------------------------------------------------------------------------------------------|

匿名メモリーは、プロセスのヒープおよびスタックに関して、また MAP\_PRIVATE (mmap(2) を参照) でマップされたマッピングを含む「書き込み時コピー」ページに関して報告されます。

|              |                                              |
|--------------|----------------------------------------------|
| ロック (Locked) | マッピング内でロックされたページ数。典型的な例として、mlock() でロックされたメモ |
|--------------|----------------------------------------------|



## アクセス権/フラグ (Mode)

リーや、SHM\_SHARE\_MMU で作成された System V 共有メモリーがあります。

仮想メモリーのアクセス権がマッピングごとに表示されます。有効なアクセス権は次のとおりです。

- r: 当該プロセスによってマッピングを読み取ることができます。
- w: 当該プロセスによってマッピングを書き込むことができます。
- x: 当該プロセスによってマッピング内に存在する命令を実行できます。

各マッピングの追加情報を示すフラグが表示される場合があります。

- s: 監視されたアドレス空間での変更がマップされたファイルに反映され、マッピングを共有するほかのすべてのプロセスから認識できるように、マッピングが共有されます。
- R: このマッピング用のスワップ空間が予約されていません。MAP\_NORESERVE で作成されたマッピングと System V ISM 共有メモリーマッピングは、スワップ空間を予約しません。
- \*: マッピングのデータがコアファイル内に存在しません(コアファイルに適用される場合にのみ該当)。コアファイルコンテンツの構成については、[coreadm\(1M\)](#) を参照してください。

## Lgroup (Lgrp)

指定されたマッピングを戻す物理メモリーが含まれている lgroup。

## マッピング名 (Mapped File)

各マッピングの説明名称。表示されるマッピング名の主要なタイプは次のとおりです。

- マップされたファイル: プロセスとファイル間のマッピングでは、pmap コマンドはマッピングごとにファイル名の解決を試みます。ファイル名を解決できない場合、pmap はファイルが含まれているデバイスのメ

ジャー番号とマイナー番号、およびファイルのファイルシステム i ノード番号を表示します。

- 匿名メモリー: ファイルシステム内のどの名前付きオブジェクトまたはファイルにも関連しないメモリーは [ anon ] として報告されます。

pmap コマンドは、既知のいくつかの匿名メモリーマッピングについては、その一般的な名前を表示します。

|                            |                                            |
|----------------------------|--------------------------------------------|
| [ heap ]                   | マッピングはプロセスヒープです。                           |
| [ stack ]                  | マッピングはメインスタックです。                           |
| [ stack tid= <i>n</i> ]    | マッピングはスレッド <i>n</i> のスタックです。               |
| [ altstack tid= <i>n</i> ] | マッピングは、スレッド <i>n</i> の代替シグナルスタックとして使用されます。 |

マッピングの共通名が不明な場合、pmap はマッピング名として [ anon ] を表示します。

- System V 共有メモリー: System V 共有メモリーシステムコールを使用して作成されたマッピングは、次に示す名前でも報告されます。

|                       |                                                                                                                             |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------|
| shmid= <i>n</i> :     | マッピングは、System V 共有メモリーマッピングです。マッピングの作成に使用された共有メモリー識別子が報告されます。                                                               |
| ism shmid= <i>n</i> : | マッピングは、System V 共有メモリーの「Intimate Shared Memory」バリエーションです。ISM マッピングは、shmat(2) ( <a href="#">shmop(2)</a> ) を参照) に従って、設定されている |

SHM\_SHARE\_MMU フラグを使って作成されます。

dism shmidx=n: マッピングは、ISM のページング可能バリエーションです。ページング可能な ISM は、shmat(2) (shmop(2) を参照) に従って、設定されている SHM\_PAGEABLE フラグを使って作成されます。

- その他: フレームバッファなどのデバイスを含む、その他のオブジェクトのマッピング。その他のマップされたオブジェクトのマッピング名は表示されません。

ページサイズ (Pgsz)

このマッピングのハードウェアアドレス変換に使用されるページサイズ (K バイト単位)。詳細は、memcntl(2) を参照してください。

スワップ空間 (Swap)

このマッピング用に予約されているスワップ空間の量 (K バイト単位)。つまり、コマンド swap -s で表示される予約可能スワップ空間の合計使用可能プールから差し引かれるスワップ空間。swap(1M) を参照してください。

## 使用例

### 例1 プロセスマッピングの表示

デフォルトでは、pmap はターゲットプロセスのアドレス空間内のマッピングごとに 1 行を表示します。次の例では、一般的な Bourne シェルのアドレス空間を表示します。

```
example$ pmap 102905
102905: sh
00010000 192K r-x-- /usr/bin/ksh
00040000 8K rwx-- /usr/bin/ksh
00042000 40K rwx-- [heap]
FF180000 664K r-x-- /usr/lib/libc.so.1
FF236000 24K rwx-- /usr/lib/libc.so.1
FF23C000 8K rwx-- /usr/lib/libc.so.1
FF250000 8K rwx-- [anon]
FF260000 16K r-x-- /usr/lib/en_US.ISO8859-1.so.2
FF272000 16K rwx-- /usr/lib/en_US.ISO8859-1.so.2
FF280000 560K r-x-- /usr/lib/libnsl.so.1
FF31C000 32K rwx-- /usr/lib/libnsl.so.1
FF324000 32K rwx-- /usr/lib/libnsl.so.1
FF350000 16K r-x-- /usr/lib/libmp.so.2
```

例1 プロセスマッピングの表示 (続き)

```
FF364000 8K rwx-- /usr/lib/libmp.so.2
FF380000 40K r-x-- /usr/lib/libsocket.so.1
FF39A000 8K rwx-- /usr/lib/libsocket.so.1
FF3A0000 8K r-x-- /usr/lib/libdl.so.1
FF3B0000 8K rwx-- [anon]
FF3C0000 152K r-x-- /usr/lib/ld.so.1
FF3F6000 8K rwx-- /usr/lib/ld.so.1
FFBFC000 16K rw--- [stack]
total 1864
```

例2 メモリー割り当てとマッピングタイプの表示

-x オプションを使用すると、マッピングごとにメモリー割り当てとマッピングタイプに関する情報を表示できます。常駐メモリー、非共有匿名メモリー、およびロックされたメモリーの量がマッピングごとに表示されます。

```
example$ pmap -x 102908
102908: sh
Address Kbytes RSS Anon Locked Mode Mapped File
00010000 88 88 - - r-x-- sh
00036000 8 8 8 - rwx-- sh
00038000 16 16 16 - rwx-- [heap]
FF260000 16 16 - - r-x-- en_US.ISO8859-1.so.2
FF272000 16 16 - - rwx-- en_US.ISO8859-1.so.2
FF280000 664 624 - - r-x-- libc.so.1
FF336000 32 32 8 - rwx-- libc.so.1
FF380000 24 24 - - r-x-- libgen.so.1
FF396000 8 8 - - rwx-- libgen.so.1
FF3A0000 8 8 - - r-x-- libdl.so.1
FF3B0000 8 8 8 - rwx-- [anon]
FF3C0000 152 152 - - r-x-- ld.so.1
FF3F6000 8 8 8 - rwx-- ld.so.1
FFBFE000 8 8 8 - rw--- [stack]

total Kb 1056 1016 56 -
```

各マッピングの常駐メモリーと匿名メモリーの数字を使用することで、プロセスの各追加インスタンスによって使用される増分メモリーの量を見積もることができます。

前述の例では、Bourne シェルには 1032K バイトの常駐メモリーがあります。ただし、シェルによって使用される大量の物理メモリーは、シェルのほかのインスタンスと共有されます。シェルの別の同一インスタンスは、可能であればほかのシェル

## 例2 メモリー割り当てとマッピングタイプの表示 (続き)

と物理メモリーを共有し、非共有部分には匿名メモリーを割り当てます。前述の例では、2つめ以降の各 Bourne シェルは約 56K バイトの追加物理メモリーを使用します。

異なるマッピングタイプを含むプロセスの出力形式を、より複雑な例で示します。この例では、マッピングは次のとおりです。

```
0001000: Executable text, mapped from 'maps' program

0002000: Executable data, mapped from 'maps' program

0002200: Program heap

0300000: A mapped file, mapped MAP_SHARED
0400000: A mapped file, mapped MAP_PRIVATE

0500000: A mapped file, mapped MAP_PRIVATE | MAP_NORESERVE

0600000: Anonymous memory, created by mapping /dev/zero

0700000: Anonymous memory, created by mapping /dev/zero
 with MAP_NORESERVE

0800000: A DISM shared memory mapping, created with SHM_PAGEABLE
 with 8MB locked via mlock(2)

0900000: A DISM shared memory mapping, created with SHM_PAGEABLE,
 with 4MB of its pages touched.

0A00000: A DISM shared memory mapping, created with SHM_PAGEABLE,
 with none of its pages touched.

0B00000: An ISM shared memory mapping, created with SHM_SHARE_MMU
```

```
example$ pmap -x 15492
```

```
15492: ./maps
```

| Address  | Kbytes | RSS   | Anon  | Locked | Mode   | Mapped File         |
|----------|--------|-------|-------|--------|--------|---------------------|
| 00010000 | 8      | 8     | -     | -      | r-x--  | maps                |
| 00020000 | 8      | 8     | 8     | -      | rw-x-- | maps                |
| 00022000 | 20344  | 16248 | 16248 | -      | rw-x-- | [ heap ]            |
| 03000000 | 1024   | 1024  | -     | -      | rw-s-  | dev:0,2 ino:4628487 |
| 04000000 | 1024   | 1024  | 512   | -      | rw---  | dev:0,2 ino:4628487 |
| 05000000 | 1024   | 1024  | 512   | -      | rw--R  | dev:0,2 ino:4628487 |
| 06000000 | 1024   | 1024  | 1024  | -      | rw---  | [ anon ]            |
| 07000000 | 512    | 512   | 512   | -      | rw--R  | [ anon ]            |

例2 メモリー割り当てとマッピングタイプの表示 (続き)

|          |       |       |       |       |       |                   |
|----------|-------|-------|-------|-------|-------|-------------------|
| 08000000 | 8192  | 8192  | -     | 8192  | rwxs- | [ dism shmid=0x5] |
| 09000000 | 8192  | 4096  | -     | -     | rwxs- | [ dism shmid=0x4] |
| 0A000000 | 8192  | 8192  | -     | 8192  | rwxsR | [ ism shmid=0x2 ] |
| 0B000000 | 8192  | 8192  | -     | 8192  | rwxsR | [ ism shmid=0x3 ] |
| FF280000 | 680   | 672   | -     | -     | r-x-- | libc.so.1         |
| FF33A000 | 32    | 32    | 32    | -     | rwX-- | libc.so.1         |
| FF3A0000 | 8     | 8     | -     | -     | r-x-- | libdl.so.1        |
| FF3B0000 | 8     | 8     | 8     | -     | rwX-- | [ anon ]          |
| FF3C0000 | 152   | 152   | -     | -     | r-x-- | ld.so.1           |
| FF3F6000 | 8     | 8     | 8     | -     | rwX-- | ld.so.1           |
| FFBFA000 | 24    | 24    | 24    | -     | rwX-- | [ stack ]         |
| -----    |       |       |       |       |       |                   |
| total Kb | 50456 | 42256 | 18888 | 16384 |       |                   |

例3 ページサイズ情報の表示

-s オプションを使用すると、アドレス空間の各部分のハードウェア変換ページサイズを表示できます。(Solaris の複数ページサイズのサポートについては、[memcntl\(2\)](#) を参照してください。)

次の例では、マッピングの大部分が8Kバイトのページサイズを使用していますが、ヒープは4Mバイトのページサイズを使用していることがわかります。

同じページサイズの常駐ページの非連続領域は、別々のマッピングとして報告されます。次の例では、libc.so テキストの一部しか常駐していないため、libc.so ライブラリは複数の別々のマッピングとして報告されます。

```
example$ pmap -xs 15492
15492: ./maps
 Address Kbytes RSS Anon Locked Pgsz Mode Mapped File
00010000 8 8 - - 8K r-x-- maps
00020000 8 8 8 - 8K rwX-- maps
00022000 3960 3960 3960 - 8K rwX-- [heap]
00400000 8192 8192 8192 - 4M rwX-- [heap]
00C00000 4096 - - - - rwX-- [heap]
01000000 4096 4096 4096 - 4M rwX-- [heap]
03000000 1024 1024 - - 8K rw-s- dev:0,2 ino:4628487
04000000 512 512 512 - 8K rw--- dev:0,2 ino:4628487
04080000 512 512 - - - rw--- dev:0,2 ino:4628487
05000000 512 512 512 - 8K rw--R dev:0,2 ino:4628487
05080000 512 512 - - - rw--R dev:0,2 ino:4628487
06000000 1024 1024 1024 - 8K rw--- [anon]
07000000 512 512 512 - 8K rw--R [anon]
08000000 8192 8192 - 8192 - rwxs- [dism shmid=0x5]
09000000 4096 4096 - - 8K rwxs- [dism shmid=0x4]
0A000000 4096 - - - - rwxs- [dism shmid=0x2]
```

## 例3 ページサイズ情報の表示 (続き)

```

0B000000 8192 8192 - 8192 4M rwxSR [ism shmid=0x3]
FF280000 136 136 - - 8K r-x-- libc.so.1
FF2A2000 120 120 - - - r-x-- libc.so.1
FF2C0000 128 128 - - 8K r-x-- libc.so.1
FF2E0000 200 200 - - - r-x-- libc.so.1
FF312000 48 48 - - 8K r-x-- libc.so.1
FF31E000 48 40 - - - r-x-- libc.so.1
FF33A000 32 32 32 - 8K rwx-- libc.so.1
FF3A0000 8 8 - - 8K r-x-- libdl.so.1
FF3B0000 8 8 8 - 8K rwx-- [anon]
FF3C0000 152 152 - - 8K r-x-- ld.so.1
FF3F6000 8 8 8 - 8K rwx-- ld.so.1
FFBFA000 24 24 24 - 8K rwx-- [stack]

total Kb 50456 42256 18888 16384

```

## 例4 スワップ予約の表示

-s オプションを使用すると、プロセスのスワップ予約の内容を表示できます。予約されたスワップ空間の量がプロセス内のマッピングごとに表示されます。共有マッピングのスワップ予約はシステム全体で1度しか行われないため、ゼロとして報告されます。

```
example$ pmap -S 15492
```

```
15492: ./maps
```

| Address  | Kbytes | Swap Mode   | Mapped File         |
|----------|--------|-------------|---------------------|
| 00010000 | 8      | - r-x--     | maps                |
| 00020000 | 8      | 8 rwx--     | maps                |
| 00022000 | 20344  | 20344 rwx-- | [ heap ]            |
| 03000000 | 1024   | - rw-s-     | dev:0,2 ino:4628487 |
| 04000000 | 1024   | 1024 rw--   | dev:0,2 ino:4628487 |
| 05000000 | 1024   | 512 rw--R   | dev:0,2 ino:4628487 |
| 06000000 | 1024   | 1024 rw--   | [ anon ]            |
| 07000000 | 512    | 512 rw--R   | [ anon ]            |
| 08000000 | 8192   | - rwxS-     | [ dism shmid=0x5]   |
| 09000000 | 8192   | - rwxS-     | [ dism shmid=0x4]   |
| 0A000000 | 8192   | - rwxS-     | [ dism shmid=0x2]   |
| 0B000000 | 8192   | - rwxSR     | [ ism shmid=0x3]    |
| FF280000 | 680    | - r-x--     | libc.so.1           |
| FF33A000 | 32     | 32 rwx--    | libc.so.1           |
| FF3A0000 | 8      | - r-x--     | libdl.so.1          |
| FF3B0000 | 8      | 8 rwx--     | [ anon ]            |
| FF3C0000 | 152    | - r-x--     | ld.so.1             |
| FF3F6000 | 8      | 8 rwx--     | ld.so.1             |
| FFBFA000 | 24     | 24 rwx--    | [ stack ]           |

## 例4 スワップ予約の表示 (続き)

```

total Kb 50456 23496
```

スワップ予約情報を使用すると、各追加プロセスによって使用される仮想スワップの量を見積もることができます。各プロセスは、大域仮想スワッププールから仮想スワップを消費します。大域スワップ予約は、[swap\(1M\)](#) コマンドの「avail」フィールドによって報告されます。

## 例5 マルチスレッドプロセスでのスタックのラベル付け

```
example$ pmap 121969
121969: ./stacks
00010000 8K r-x-- /tmp/stacks
00020000 8K rwx-- /tmp/stacks
FE8FA000 8K rwx-R [stack tid=11]
FE9FA000 8K rwx-R [stack tid=10]
FEAFA000 8K rwx-R [stack tid=9]
FEBFA000 8K rwx-R [stack tid=8]
FECFA000 8K rwx-R [stack tid=7]
FEDFA000 8K rwx-R [stack tid=6]
FEEFA000 8K rwx-R [stack tid=5]
FEFFA000 8K rwx-R [stack tid=4]
FF0FA000 8K rwx-R [stack tid=3]
FF1FA000 8K rwx-R [stack tid=2]
FF200000 64K rw--- [altstack tid=8]
FF220000 64K rw--- [altstack tid=4]
FF240000 112K rw--- [anon]
FF260000 16K rw--- [anon]
FF280000 672K r-x-- /usr/lib/libc.so.1
FF338000 24K rwx-- /usr/lib/libc.so.1
FF33E000 8K rwx-- /usr/lib/libc.so.1
FF35A000 8K rwx-s [anon]
FF360000 104K r-x-- /usr/lib/libthread.so.1
FF38A000 8K rwx-- /usr/lib/libthread.so.1
FF38C000 8K rwx-- /usr/lib/libthread.so.1
FF3A0000 8K r-x-- /usr/lib/libdl.so.1
FF3B0000 8K rwx-- [anon]
FF3C0000 152K r-x-- /usr/lib/ld.so.1
FF3F6000 8K rwx-- /usr/lib/ld.so.1
FFBFA000 24K rwx-- [stack]
total 1384
```

## 例6 lgroup メモリーの割り当ての表示

次の例では、lgroup メモリーの割り当てをマッピングごとに表示します。



## 例6 lgroup メモリーの割り当ての表示 (続き)

```

example$ pmap -L 'pgrep nscd'
100095: /usr/sbin/nscd
00010000 8K r-x-- 2 /usr/sbin/nscd
00012000 48K r-x-- 1 /usr/sbin/nscd
0002E000 8K rwx-- 2 /usr/sbin/nscd
00030000 16K rwx-- 2 [heap]
00034000 8K rwx-- 1 [heap]
.
.
.
FD80A000 24K rwx-- 2 [anon]
FD820000 8K r-x-- 2 /lib/libmd5.so.1
FD840000 16K r-x-- 1 /lib/libmp.so.2
FD860000 8K r-x-- 2 /usr/lib/straddr.so.2
FD872000 8K rwx-- 1 /usr/lib/straddr.so.2
FD97A000 8K rw--R 1 [stack tid=24]
FD990000 8K r-x-- 2 /lib/nss_nis.so.1
FD992000 16K r-x-- 1 /lib/nss_nis.so.1
FD9A6000 8K rwx-- 1 /lib/nss_nis.so.1
FD9C0000 8K rwx-- 2 [anon]
FD9D0000 8K r-x-- 2 /lib/nss_files.so.1
FD9D2000 16K r-x-- 1 /lib/nss_files.so.1
FD9E6000 8K rwx-- 2 /lib/nss_files.so.1
FDAFA000 8K rw--R 2 [stack tid=23]
FDBFA000 8K rw--R 1 [stack tid=22]
FDCFA000 8K rw--R 1 [stack tid=21]
FDDFA000 8K rw--R 1 [stack tid=20]
.
.
.
FEFFA000 8K rw--R 1 [stack tid=2]
FF000000 8K rwx-- 2 [anon]
FF004000 16K rwx-- 1 [anon]
FF00A000 16K rwx-- 1 [anon]
.
.
.
FF3EE000 8K rwx-- 2 /lib/ld.so.1
FFBFE000 8K rw-- 2 [stack]
total 2968K

```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 0 以外 エラーが発生しました。

ファイル            /proc/\*            プロセスファイル  
                     /usr/proc/lib/\*    proc ツールサポートファイル

属性                属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

コマンドの構文は「確実」です。-L オプションおよび出力形式は「不確実」です。

関連項目            [ldd\(1\)](#), [lgrpinfo\(1\)](#), [madv.so.1\(1\)](#), [mdb\(1\)](#), [plgrp\(1\)](#), [pmadvise\(1\)](#), [proc\(1\)](#), [ps\(1\)](#),  
[coreadm\(1M\)](#), [prstat\(1M\)](#), [swap\(1M\)](#), [mmap\(2\)](#), [memcntl\(2\)](#), [meminfo\(2\)](#), [shmop\(2\)](#),  
[dlopen\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)

|         |                                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pm-updatemanager – パッケージを更新するためのアプリケーション                                                                                                                                                                                                                                                                                                                                                                                 |
| 形式      | <pre> /usr/bin/pm-updatemanager [<i>options</i>] /usr/bin/pm-updatemanager [-h   --help] [-d   --debug]                         [-R <i>dir</i>   --image-dir <i>dir</i>] </pre>                                                                                                                                                                                                                                          |
| 機能説明    | <p>pm-updatemanager は、システムにインストールされたパッケージで利用可能な更新の有無を確認し、インストールします。</p> <p>注 – package/pkg、package/pkg/package-manager、または package/pkg/update-manager パッケージを更新する必要がある場合、pm-updatemanager は、最初にこれらのパッケージを更新してから、残りの更新の実行を再開します。</p>                                                                                                                                                                                       |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <pre> -h --help     使用方法に関するメッセージを表示します。  -d --debug     デバッグモードで pm-updatemanager を実行します。  -R <i>dir</i> --image-dir <i>dir</i>     自動的に検出されたイメージではなく、<i>dir</i> をルートとするイメージに対して処理を行います。 </pre>                                                                                                                                                                                          |
| 使用例     | <p>例1 現在のイメージの更新</p> <p>現在のイメージ上で pm-updatemanager を呼び出します。現在のイメージにインストールされたパッケージで利用可能なすべての更新の有無を確認し、インストールします。</p> <pre>\$ /usr/lib/pm-launch pm-updatemanager</pre> <p>これは、デスクトップのメニューオプションで「システム」&gt;「管理」&gt;「Update Manager」を選択したときに呼び出されるコマンドと同じです。</p> <p>例2 指定されたイメージの更新</p> <p>/aux0/example_root に格納されている pm-updatemanager を呼び出します。</p> <pre>\$ /usr/lib/pm-launch pm-updatemanager -R /aux0/example_root</pre> |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <pre> 0    すべてが動作しました。 1    エラーが発生した。 </pre>                                                                                                                                                                                                                                                                                                                                                     |

2 無効なコマンド行オプションが指定された。

属性 次の属性については、attributes(5)を参照してください。

| 属性タイプ       | 属性値                        |
|-------------|----------------------------|
| 使用条件        | package/pkg/update-manager |
| インタフェースの安定性 | 不確実                        |

関連項目 [packagemanager\(1\)](#), [pkg\(1\)](#), [pkg\(5\)](#)

<http://hub.opensolaris.org/bin/view/Project+pkg/>

注意事項 所有権がないイメージに対して処理を行う場合、必要な権限を使用して pm-updatemanager を呼び出す必要があります。通常、このような状況では、 /usr/lib/pm-launch を使用して pm-updatemanager を呼び出します。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | ppgsz - スタック、ヒープ、およびその他の匿名セグメントの優先ページサイズの設定                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 形式    | /usr/bin/ppgsz [-F] -o option[,option] cmd   -p pid...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 機能説明  | ppgsz ユーティリティーは、ターゲットプロセス (起動された <i>cmd</i> またはプロセス ID リストにあるプロセス) のスタック、ヒープ、およびその他の匿名セグメントの優先ページサイズを設定します。ppgsz は、ページサイズを変更する間、ターゲットプロセスを停止します。memcntl(2) を参照してください。                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <p><b>-F</b> 強制。別のプロセスによって制御されている場合でも、ターゲットプロセスに優先ページサイズオプションを設定します。<b>-F</b> フラグを使用するときは注意してください。proc(1) を参照してください。</p> <p><b>-o option[,option]</b> option は次のとおりです。</p> <p><b>heap=size</b> このオプションは、ターゲットプロセスのヒープの優先ページサイズを指定します。heap は、bss (初期化されていないデータ) と bss の直後に続く brk 領域として定義されます (brk(2) を参照)。ヒープの優先ページサイズは、既存のヒープと将来割り当てられる追加ヒープメモリに設定されます。「注意事項」を参照してください。</p> <p><b>stack=size</b> このオプションは、ターゲットプロセスのスタックの優先ページサイズを指定します。スタックの優先ページサイズは、既存のスタック、および拡張されて新しく割り当てられるスタックの部分に設定されます。</p> <p><b>anon=size</b> このオプションは、ターゲットプロセスの既存のすべての MAP_PRIVATE 匿名セグメントの優先ページサイズを指定します。これらは、heap および stack 以外のセグメントであり、指定されたサイズの割り当てられた 1 ページに少なくとも適合する十分な大きさがあります。十分に大きいセグメントの場合、優先ページサイズは、セグメント内のサイズに合わせた最初のアドレスから始まるように設定されます。anon 優先ページサイズは、将来作成される MAP_PRIVATE 匿名セグメントには適用されません。mmap(2) の MAP_ANON を参照してください。</p> |

匿名メモリーとは、ファイルシステムでファイルと直接関連付けられない MAP\_PRIVATE ページのことを指しています。ppgsz コマンドは、[memcntl\(2\)](#) を使用して、匿名セグメントの優先ページサイズを設定します。memcntl(2) の MC\_HAT\_ADVISE を参照してください。

上記のオプションを少なくとも1つ指定する必要があります。

size は、サポートされるページサイズ ([pagesize\(1\)](#) を参照) または 0 である必要があり、後者の場合、システムは適切なページサイズを選択します。[memcntl\(2\)](#) を参照してください。

デフォルトでは size はバイトであり、8 進数 (0)、10 進数、または 16 進数 (0x) で指定できます。数値は、K、M、G、または T を付けると、それぞれキロバイト、メガバイト、ギガバイト、またはテラバイトで指定できます。4194304、0x400000、4096K、0x1000K、および 4M は、異なる方法で 4M バイトを指定しています。

#### -p pid

-p オプションの後ろのプロセス ID (pid) リストにあるターゲットプロセスに優先ページサイズオプションを設定します。pid リストは、/proc ディレクトリ内の名前で作成することもできます。ページサイズを設定できるのは、プロセスの所有者またはスーパーユーザーのみです。

-p が指定されていない場合は、cmd が解釈されます。ppgsz は、cmd を起動して、新しいプロセスにページサイズオプションを適用します。

ヒープおよびスタックの優先ページサイズは継承されます。ppgsz が完了したあとに、起動されたプロセスまたは pid リスト内のターゲットプロセスから作成された子プロセス ([fork\(2\)](#) を参照) は、ヒープとスタックの優先ページサイズを継承します。[exec\(2\)](#) では、すべてのセグメントの優先ページサイズがデフォルトのシステムページサイズに戻されます ([getpagesize\(3C\)](#) を参照)。その他のすべての匿名セグメントの優先ページサイズは、起動されたプロセスまたはターゲットプロセスの子には継承されません。

## 使用例

### 例1 ヒープおよびスタックの優先ページサイズの設定

次の例では、ora で始まるコマンドを実行している ora 所有のすべてのプロセスのヒープの優先ページサイズを 4M に、スタックの優先ページサイズを 512K に設定します。

例1 ヒープおよびスタックの優先ページサイズの設定 (続き)

```
example% ppgsz -o heap=4M,stack=512K -p 'pgrep -u ora '^ora''
```

例2 匿名セグメントの優先ページサイズを設定する

次の例では、プロセス ID 953 の既存の該当する匿名セグメントの優先ページサイズを 512K に設定します。

```
example% ppgsz -o anon=512k -p 953
```

**終了ステータス** *cmd* が指定されて正常に呼び出された ([exec\(2\)](#) を参照) 場合、ppgsz の終了ステータスは *cmd* の終了ステータスになります。それ以外の場合は、ppgsz は次のいずれかの値で終了します。

- 0 プロセス ID リストにあるプロセスに優先ページサイズを正常に設定されました。
- 125 ppgsz でエラーが発生しました。エラーには、無効な引数、無効なページサイズの指定、プロセス ID リストまたは *cmd* の 1 つ以上のプロセスに対する優先ページサイズの設定の失敗などが含まれます。
- 126 *cmd* が見つかりましたが、呼び出せませんでした。
- 127 *cmd* が見つかりませんでした。

**ファイル** */proc/\** プロセスファイル。  
*/usr/lib/ld/map.bssalign* bss 整列用のテンプレートリンカーマップファイル (mapfile) (「注意事項」を参照)。

**属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                              |
|-------------|----------------------------------|
| 使用条件        | system/extended-system-utilities |
| インタフェースの安定性 | 確実                               |

**関連項目** [ld\(1\)](#), [mpss.so.1\(1\)](#), [pagesize\(1\)](#), [pgrep\(1\)](#), [pmap\(1\)](#), [proc\(1\)](#), [brk\(2\)](#), [exec\(2\)](#), [fork\(2\)](#), [mемcntl\(2\)](#), [mmap\(2\)](#), [sbrk\(2\)](#), [getpagesize\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

**注意事項** リソースの制約があるため、優先ページサイズを設定しても、ターゲットプロセスは必ずしも優先ページサイズを取得すると保証されるわけではありません。ターゲットプロセスのヒープとスタックの実際のページサイズを表示するには、[pmap\(1\)](#) を使用します (pmap -s オプションを参照)。

大きいページは、大きいページサイズの倍数のアドレスにマップする必要があります。通常、ヒープは大きいページに割り当てられないため、(最初に大きいページが割り当てられるアドレスより前の)ヒープの開始位置は、システムメモリーのページサイズでマップされます。[getpagesize\(3C\)](#)を参照してください。

大きなページサイズでマップされるヒープを提供するために、`bss` セグメント宣言の指令を含むリンカー ([ld\(1\)](#)) の `mapfile` を使用してアプリケーションを構築できます。この指令、および `/usr/lib/ld/map.bssalign` で提供されるテンプレート `mapfile` の詳細は、『[リンカーとライブラリ](#)』の「`Mapfile` オプション」セクションを参照してください。割り当ての指定はマシンによって異なることがあり、異なるハードウェアプラットフォームでは利点が損なわれる場合があるため、注意してください。将来のリリースでは、基本となる最適なページサイズをより柔軟に要求する方法が進展する可能性があります。

また、[mpss.so.1\(1\)](#) (事前にロード可能な共有オブジェクト) を使用して、スタックまたはヒープ、あるいはその両方の優先ページサイズを設定することもできます。



|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | ppriv – プロセス特権セットおよび属性の検査または変更                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 形式    | <pre> /usr/bin/ppriv -e [-D   -N] [-M] [-s spec] [-X   -r rule] command [arg]...  /usr/bin/ppriv [-vn] [-S] [-D   -N] [-s spec] [-X   -r rule]     [pid   core]...  /usr/bin/ppriv -l [-vn] [privilege-specification   extended-policy]...</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 機能説明  | <p>ppriv コマンドの最初の呼び出しでは、コマンド行の引数に従って変更された特権セットとフラグが指定された <i>command</i> が実行されます。</p> <p>2 番目の呼び出しでは、実行中のプロセスとコアファイルの特権の状態を検査または変更します。</p> <p>3 番目の呼び出しでは、定義されている特権および指定された特権または特権セットの指定に関する情報を一覧表示します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-D 指定されたプロセスまたはコマンドの特権のデバッグを有効にします。</li> <li>-e 残りの引数をコマンド行として解釈し、指定された特権の属性およびセットでコマンド行を実行します。</li> <li>-l 現在定義されている特権をすべて標準出力に書き込みます。</li> <li>-M システムが Trusted Extensions で構成されている場合、このオプションを使用すると、NET_MAC_AWARE および NET_MAC_AWARE_INHERIT プロセス属性が有効になります。</li> </ul> <p>これらの属性と net_mac_aware 特権を持つプロセスは、より低いレベルのリモートピアと通信できます。</p> <ul style="list-style-type: none"> <li>-n ポート番号とユーザーを数字として表示します。通常、ppriv は、ポート番号とユーザーをシンボルとして表示します。このオプションは、拡張ポリシーを表示する場合にのみ適用できます。</li> <li>-N 指定されたプロセスまたはコマンドの特権のデバッグを無効にします。</li> <li>-s spec スペース文字を含まない [AEILP][+ -=]privsetspec という形式の指定である spec に従ってプロセスの特権セットを変更します。</li> </ul> <p>AEILP 変更する特権セットを示す 1 文字以上の英字を示します。これらは大文字と小文字の区別がなく、たとえば、a または A のいずれもすべての特権セットを示します。</p> <p>特権セットの単一文字省略名の定義については、<a href="#">privileges(5)</a> を参照してください。</p> |

**+=** リストされた特権を *privsetspec* で指定されたセットに、それぞれ追加(+)、削除(-)、または割り当て(=)する修飾子を示します。

*privsetspec* [priv\\_str\\_to\\_set\(3C\)](#) で説明されている、コンマで区切られた特権セットの指定 (*priv1*, *priv2* など) を示します。

個々のセットにそれぞれ1つの割り当てを行うか、任意の数の追加および削除を行うかのいずれかであれば、同じセットを複数の **-s** オプションで変更することは可能です。つまり、単一のセットに対する割り当てと、追加または削除は相互に排他的です。

**-r rule** 拡張ポリシーをインストールします。 [privileges\(5\)](#) を参照してください。

複数のルールを指定できます。新しいルールは、既存のポリシーに追加されます。既存のポリシーを置き換えるには、まず **-x** を使用してそのポリシーを削除し、次に **-r** を使用して新しいポリシーを追加します。

**-S** 短い。可能なかぎり短い出力文字列でセットを報告します。デフォルトは移植性のある出力形式です。 [priv\\_str\\_to\\_set\(3C\)](#) を参照してください。

**-X** 拡張ポリシーを無効にします。

**-v** 冗長。特権名を使用して特権セットを報告します。

## 使用法

**ppriv** ユーティリティーは、プロセスとコアファイルを検査し、それらの特権セットを出力または変更します。

**ppriv** は、特権のデバッグを有効または無効にして、あるいは呼び出すプロセスよりも少ない特権でコマンドを実行できます。

サブプロセスを実行する場合、変更できるセットは **L** と **I** のみです。 **ppriv** が **P=E=I** で起動された場合、特権を削除できるのは **L** と **I** からのみです。

**ppriv** は、プロセスから特権を削除する場合、または別のプロセスに特権を委譲する場合にも使用できます。プロセスを制御するには、 **ppriv** ユーティリティーの実効セットは、制御されるプロセスの **E**、**I**、および **P** の上位集合である必要があります。ユーティリティーの制限セットは、ターゲットの制限セットの上位集合である必要があります。ターゲットプロセスのユーザー ID が一致しない場合は、ユーティリティーの実効セットで **{PRIV\_PROC\_OWNER}** 特権を表明する必要があります。制御されるプロセスが値 **0** のユーザー ID を持つ場合、さらに制限が存在する可能性があります。 [privileges\(5\)](#) を参照してください。

## 使用例

例1 現在のシェルに対するプロセスの特権の取得

次の例では、現在のシェルに対するプロセスの特権を取得します。

### 例1 現在のシェルに対するプロセスの特権の取得 (続き)

```
example$ ppriv $$
387: -sh
flags = <none>
 E: basic
 I: basic
 P: basic
 L: all
```

### 例2 シェルの継承可能な実効セットからの特権の削除

次の例では、シェルの継承可能な実効セットから特権を削除します。

```
example$ ppriv -s EI-proc_session $$
```

サブプロセスはまだ親シェルを検査できますが、親に影響を及ぼすことはできません。親は ppriv の子プロセスよりも多い特権を Permitted セットに持っているためです。

```
example$ truss -p $$
truss: permission denied: 387
```

```
example$ ppriv $$
387: -sh
flags = <none>
 E: basic,!proc_session
 I: basic,!proc_session
 P: basic
 L: all
```

### 例3 特権のデバッグを使用したプロセスの実行

次の例では、特権のデバッグを有効にしてプロセスを実行します。

```
example$ ppriv -e -D cat /etc/shadow
cat[418]: missing privilege "file_dac_read" (euid = 21782),
 needed at ufs_access+0x3c
cat: cannot open /etc/shadow
```

特権のデバッグのエラーメッセージは、現在のプロセスの制御端末に送信されます。needed at アドレス指定は、カーネル実装のアーティファクトであり、ソフトウェアの更新後にいつでも変更できます。

システムコール番号は、/etc/name\_to\_sysnum を使用するとシステムコールにマップできます。

**例4** 現在のゾーンで使用可能な特権の一覧表示

次の例では、現在のゾーンで使用可能な特権を一覧表示します ([zones\(5\)](#) を参照)。大域ゾーンで実行すると、定義されているすべての特権が一覧表示されます。

```
example$ ppriv -l zone
... listing of all privileges elided ...
```

**例5** 特権に対応したプロセスの検査

次の例では、特権に対応したプロセスを検査します。

```
example$ ppriv -S 'pgrep rpcbind'
```

```
928: /usr/sbin/rpcbind
flags = PRIV_AWARE
E: net_privaddr,proc_fork,sys_nfs
I: none
P: net_privaddr,proc_fork,sys_nfs
L: none
```

フラグの説明については、[setpflags\(2\)](#) を参照してください。

**例6** 拡張ポリシーの下でのプロセスの実行

次の例では、拡張ポリシーの下でプロセスを実行します。

```
example$ ppriv -r '{file_write}:/home/casper/.mozilla/*' \
-r '{file_write}:/tmp/*,{proc_exec}:/usr/*' -e firefox
```

[privileges\(5\)](#) を参照してください。

**例7** 起動されているプロセスの検査

次の例では、例6で起動されたプロセスを検査します。

```
example$ ppriv 101272
101272: /usr/lib/firefox/firefox-bin
flags = PRIV_XPOLICY
Extended policies:
{file_write}:/home/casper/.mozilla/*
{file_write}:/tmp/*
{proc_exec}:/usr/*
E: basic,!file_write,!proc_exec
I: basic,!file_write,!proc_exec
P: basic,!file_write,!proc_exec
L: all
```

- 終了ステータス

次の終了ステータスが返されます。  
0            正常終了。  
0 以外      エラーが発生しました。
- ファイル

/proc/\*                      プロセスファイル  
/etc/name\_to\_sysnum       システムコール名と番号のマッピング
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

- 呼び出しは「確実」です。出力は「不確実」です。
- 関連項目

[gcore\(1\)](#), [truss\(1\)](#), [setpflags\(2\)](#), [priv\\_str\\_to\\_set\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#), [privileges\(5\)](#), [zones\(5\)](#)

名前 pr - ファイルの出力

形式

```
/usr/bin/pr [+ page] [-column] [-adFmrt] [-e [char] [gap]] [-h header] [-i [char] [gap]] [-l lines] [
/usr/xpg4/bin/pr [+ page] [-column | -c column] [-adFmrt] [-e [char] [gap]] [-h header] [-i [char] [
```

機能説明

pr ユーティリティーは、印刷とページ設定用のフィルタです。複数のファイルを指定すると、その各々を読み込み、書式化して、標準出力に書き出します。デフォルトでは、入力データを 66 行からなるページに分割します。各ページには以下のものが含まれます。

- 5 行からなるヘッダー。ページ番号、日付、時間、ファイルのパス名を含む
- 5 行の空行からなるトレーラ

標準出力が端末に割り当てられているとき、診断メッセージは pr ユーティリティーの処理が完了してから出力されます。

複数カラム出力を示すオプションを指定すると、テキストカラムは等しい幅で出力されます。この幅に収まらない入力行は切り捨てられます。デフォルトでは、出力テキストカラムは 1 つ以上の空白文字で区切られます。

オプション

以下のオプションがサポートされています。なおオプションの説明において、*column*、*lines*、*offset*、*page*、*width* は正の 10 進整数を表し、*gap* は負でない 10 進整数を表します。オプション引数の中には、省略可能なものがあります。また直前のオプション文字と分離して独立した引数として指定できないものがあります。具体例としては、*-s* オプションと引数とを離して記述することはできません。また *-e*、*-i*、*-n* の各オプションは、2 つの引数を記述するのであれば、それらをオプション文字と離すことはできません。

次のオプションは、*/usr/bin/pr* と */usr/xpg4/bin/pr* の両方で使用できます。

- +page* 書式化された入力の、*page* で示すページ番号から出力を開始します。
- column* *column* で示す数 (デフォルトは 1) のカラムからなる複数カラム出力を生成します。これらのカラムには、入力ファイルからテキストを受け取った順序でデータが書き込まれます。このオプションは *-m* オプションと同時に使うことはできません。複数カラム出力を指定すると、*-e* と *-i* の両オプションも指定されたと見なされます。同一行数となるようなテキストカラムの生成を指定したか否かに関わらず、テキストカラムがページの長さを超えることはありません (*-l* オプションの説明を参照)。このオプションを *-t* とともに指定した場合、最少の行数で出力データを書き出します。
- a* 出力ページ内において、連鎖的にカラムが印刷されるよう、*-column* オプション指定による出力を加工します。た

例えば、*column* の値が2 のとき、最初の入力行が1 行目のカラム1 に、2 番目の入力行が1 行目のカラム2 に出力された後、続く3 番目の行は2 行目のカラム1 に置かれます。

- d 入力内に復帰改行文字が現れるたびに、そのあとにもう1 つ改行文字を挿入して、ダブルスペース (2 倍の改行幅) で出力を生成します。
- e [ *char* ][ *gap* ] 入力内の各タブ文字を、式  $n * gap + 1$  を満たす次のカラム位置に展開します。*n* は正の整数です。*gap* に 0 を指定するか、*gap* を省略すると、デフォルトとして8 が用いられます。入力内のタブ文字はすべて、適切な数のスペース文字に展開されます。数字でない文字 *char* を指定すると、それは入力タブ文字と見なされます。
- f 改ページの際に、デフォルトである一連の復帰改行文字の代わりに、用紙送り文字を使用します。標準出力が端末に割り当てられているとき、最初のページの出力を始める前に一時停止します。
- h *header* ページヘッダーの見出しとして、*file* オペランドの値の代わりに *header* で示す文字列を使用します。
- l *lines* ページの長さを、デフォルトの 66 の代わりに *lines* で示す行数に設定します。*lines* の値がヘッダーとトレーラの長さの合計行数以下の場合、*pr* は -t オプションが有効な場合と同じように、ヘッダーとトレーラの出力を抑止します。
- m ファイルをマージします。*pr* は、*file* で指定された複数のファイルから1 行ずつを読み込み、同じ幅のテキストカラムにそれらの行の内容を書き込んで標準出力を生成します。最大9 個のファイルのマージがサポートされています。
- n [ *char* ][ *width* ] 行番号の桁数を *width* で示す値とします。デフォルトは5 です。出力上において行番号は、デフォルト出力の場合には各テキストカラムの先頭の *width* 個のカラムを、-m 出力の場合には各行の先頭の *width* 個のカラムを占めます。数字でない文字 *char* を指定すると、行番号と後続の出力を隔てるために行番号に付加されます。*char* のデフォルトはタブ文字です。
- o *offset* 各出力行の前にオフセットの<スペース>が置かれます。-o オプションを省略すると、デフォルトのオフ

セット値は0となります。ここで指定するオフセット値は、出力行の幅に追加されます(後述する `-w` オプションの説明を参照)。

- p 標準出力が端末に割り当てられているとき、ページの出力を始める前に一時停止します。pr は警告 (ALERT) 文字を標準エラー出力に送り、`/dev/tty` 上で復帰改行文字が読み込まれるのを待ちます。
- r ファイルをオープンできなくても、それに関する診断レポートを出力しません。
- s [*char*] テキストカラムを区切る文字として、一連のスペース文字の代わりに、*char* で示す1文字を使用します。*char* のデフォルトはタブ文字です。
- t 各ページに通常出力される5行のヘッダーと5行のトレーラを出力しません。各ファイルの最終行を出力した後、そのページの最終行まで行送りをせず、出力を停止します。
- w *width* 行の幅を *width* で示す数のカラム位置に設定します。この指定は複数カラム出力に対してのみ有効です。このオプションを省略したときのデフォルトの行幅は、`-s` オプションも省略されていれば72となり、`--s` オプションが指定されていれば512となります。-

シングルカラム出力の場合には、入力行の内容が出力幅の制限によって切り捨てられることはありません。

`/usr/bin/pr` 次のオプションは、`/usr/bin/pr` でのみ使用できます。

- F 入力ファイルの行を折り返します。`-a` または `-m` オプション指定による複数カラムモード使用時は、現在のカラム幅に合うように行を折り返します。それ以外の場合は、現在の行幅(80カラム)に合うように行を折り返します。
- i [*char*] [*gap*] 出力上で1つ以上の連続したスペース文字が  $gap+1$ 、 $2*gap+1$ 、 $3*gap+1$  のカラム位置に到達した場合、それらのスペース文字をタブ文字で置き換えます。*gap* を省略するか、0を指定した場合、デフォルトとして8カラムごとにタブ位置が設定されていると見なされます。数字でない文字 *char* を指定すると、それが出力タブ文字として使用されます。

`/usr/xpg4/bin/pr` 次のオプションは、`/usr/xpg4/bin/pr` でのみ使用できます。



- F

改ページの際に、デフォルトである一連の復帰改行文字の代わりに、用紙送り文字を使用します。
- i [ char ][ gap ]

出力上で2つ以上の連続したスペース文字が  $gap+1$ 、 $2*gap+1$ 、 $3*gap+1$  のカラム位置に到達した場合、それらのスペース文字をタブ文字で置き換えます。 $gap$  を省略するか、 $0$  を指定した場合、デフォルトとして8カラムごとにタブ位置が設定されていると見なされます。数字でない文字  $char$  を指定すると、それが出力タブ文字として使用されます。

オペランド

次のオペランドを指定できます。

file

内容を印刷するファイルのパス名。このオペランドを省略するか、`-`を指定すると、標準入力を用いられます。

使用例

例1

現在のディレクトリ中の全ファイルの番号付き一覧を出力する

example% `ls -a | pr -n -h "Files in $(pwd)."`

例2

ファイルをカラムに出力する

この例では、`file1` および `file2` を、`file list` というヘッダーを付けた、3カラムからなる2倍の改行幅のリストに出力します。

example% `pr -3d -h "file list" file1 file2`

例3

拡張したカラムタブを使用してファイルを書き出す

次の例では、タブ位置をカラム10、19、28に設定して、`file1` を `file2` に書き出します。

example% `pr -e9 -t <file1 >file2`

環境変数

`pr` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、LC\_TIME、TZ、およびNLSPATH。

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/pr

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

/usr/xpg4/bin/pr

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/xopen/xcu4                       |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目

[expand\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

|         |                                                                                                                                                                                                                                                                                                                                                                         |
|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | praliases – display system mail aliases                                                                                                                                                                                                                                                                                                                                 |
| 形式      | praliases [-C <i>configfile</i> ] [-f <i>aliasfile</i> ] [ <i>key</i> ]                                                                                                                                                                                                                                                                                                 |
| 機能説明    | The <code>praliases</code> utility displays system mail aliases. When no key is given, <code>praliases</code> displays the current system aliases, one per line, in no particular order. The form is <code>key: value</code> . If a key is given, only that key is looked up and the appropriate <code>key: value</code> is displayed if found.                         |
| オプション   | <p>The following options are supported:</p> <ul style="list-style-type: none"> <li>-C <i>configfile</i> Specifies a sendmail configuration file.</li> <li>-f <i>aliasfile</i> Reads the specified file <i>aliasfile</i> instead of the default sendmail system aliases file.</li> </ul>                                                                                 |
| オペランド   | <p>The following operands are supported:</p> <ul style="list-style-type: none"> <li><i>key</i> A specific alias key to look up.</li> </ul>                                                                                                                                                                                                                              |
| 終了ステータス | <p>The following exit values are returned:</p> <ul style="list-style-type: none"> <li>0 Successful operation.</li> <li>&gt;0 An error occurred.</li> </ul>                                                                                                                                                                                                              |
| ファイル    | <ul style="list-style-type: none"> <li>/etc/mail/aliases Default sendmail system aliases file</li> <li>/etc/mail/aliases.db Database versions of the /etc/mail/aliases file</li> <li>/etc/mail/aliases.dir</li> <li>/etc/mail/aliases.pag Database versions of the /etc/mail/aliases file</li> <li>/etc/mail/sendmail.cf Default sendmail configuration file</li> </ul> |
| 属性      | See <a href="#">attributes(5)</a> for descriptions of the following attributes:                                                                                                                                                                                                                                                                                         |
| 関連項目    | <a href="#">mailq(1)</a> , <a href="#">newaliases(1M)</a> , <a href="#">sendmail(1M)</a> , <a href="#">attributes(5)</a>                                                                                                                                                                                                                                                |

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | prctl - 実行中のプロセス、タスク、およびプロジェクトのリソース制御の取得または設定                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| 形式    | prctl [-P] [-t [basic   privileged   system]] [-n name [-srx] [-v value] [-e   -d action] [-p pid]]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 機能説明  | <p>prctl ユーティリティーを使用すると、システム上のアクティブなプロセス、タスク、またはプロジェクトに関連付けられているリソース制御に対して、確認や変更を行うことができます。これを使えば、指定されたエンティティー上の基本制限、特権付き制限、および現在の使用状況にアクセスできます。</p> <p>Solaris オペレーティングシステムの現行リリースでサポートされているリソース制御については、<a href="#">resource_controls(5)</a> を参照してください。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| オプション | <p>-s、-r、-x、-v、-d、-e のいずれのオプションも指定されなかった場合、その呼び出しは取得処理とみなされます。それ以外の場合、それは変更処理とみなされます。</p> <p>サポートしているオプションは、次のとおりです。</p> <div><div>-d   -e action</div><div>-v、-t、および -p で指定されたリソース制御値に対し、指定された <i>action</i> を無効 (-d) または有効 (-e) にします。-v、-t、-p のいずれかのオプションが指定されなかった場合、すべての値、特権、または受取人 pid に一致します。たとえば、-v のみが指定された場合、それに一致する値を持つ最初のリソース制御が変更されます。その際、すべての特権と受取人 pid に一致します。一致するリソース制御値が見つからなかった場合には、あたかも -s が指定されたかのように、新しい値が追加されます。</div></div> <p>アクション:</p> <div><div>all</div><div>このアクションは -d でのみ使用できます。これは、すべてのアクションを無効化します。これは、deny 大域フラグを持つリソース制御値では失敗します。</div></div> <div><div>deny</div><div>プロセス、タスク、プロジェクト、またはゾーンからのリソース要求がリソース制御値を超えた場合に、リソース制御がその要求を拒否しようとすることを示します。no-deny 大域フラグを持つリソース制御では、deny アク</div></div> |

|               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | <p>ションは有効にできません。deny 大域フラグを持つリソース制御では、deny アクションは無効にできません。</p>                                                                                                                                                                                                                                                                                                                                                                                                                |
| signal        | <p>このアクションは -d でのみ使用できます。これは、signal アクションを無効にします。</p>                                                                                                                                                                                                                                                                                                                                                                                                                         |
| signal=signum | <p>signal=signum アクションの signum は、シグナル番号 (またはシグナルの文字列表現) です。no-local-action 大域フラグを持つリソース制御上に signal アクションを設定すると、失敗します。送信可能なシグナルは限られています。詳細については、「注意事項」を参照してください。</p>                                                                                                                                                                                                                                                                                                           |
| -i idtype     | <p>ID オペランドのタイプを指定します。有効な idtype は、process、task、project、または zone です。また、pid、taskid、projid、および zoneid も使用できます。-i オプションが省略された場合のデフォルトの ID タイプは、process です。</p> <p>変更処理の場合、ID オペランドが所属するエンティティが、ターゲットエンティティになります。たとえば、-i process を指定して特定のプロジェクトリソース制御を設定した場合、指定された各プロセス引数が所属するプロジェクト上にそのリソース制御が設定されます。</p> <p>取得処理の場合、ID オペランドが所属するすべてのエンティティのリソース制御が、一覧表示されます。たとえば、-i task taskid と指定した場合、そのタスクのタスクリソース制御、プロジェクトリソース制御、およびゾーンリソース制御に加え、そのタスクが所属するプロジェクトとゾーンのリソース制御も一覧表示されます。</p> |
| -n name       | <p>取得または設定するリソース制御の名前を指定します。name が指定されなかった場合、すべてのリソース制御が取得されます。</p>                                                                                                                                                                                                                                                                                                                                                                                                           |

|                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>-p pid</code>                       | <p><code>-s</code>、<code>-r</code>、<code>-x</code>、<code>-d</code>、<code>-e</code>のいずれかを使ってタスク、プロジェクト、またはゾーンの基本リソース制御値を操作する場合、<code>-p</code>を使って受取人 <i>pid</i> を指定することができます。特定のタスク、プロジェクト、またはゾーン上で1つ以上の新しい基本リソース制御を設定する際に、<code>-i idtype</code> オプションの引数が <code>process</code> でない場合には <code>-p</code> オプションが必要となります。</p>                                                                                                                                                                                                                           |
| <code>-P</code>                           | <p>リソース制御値をスペース文字で区切られた形式で表示します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <code>-r</code>                           | <p>最初のリソース制御値 (<code>-t privilege</code> に一致するもの) を、<code>-v</code> オプションで指定された新しい値で置き換えます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>-s</code>                           | <p>新しいリソース制御値を設定します。</p> <p>このオプションは、必ず <code>-v</code> とともに指定してください。</p> <p>ユーザーが <code>-t</code> オプションを指定しなかった場合、基本特権が使用されます。基本のタスク資源制御、プロセス資源制御、またはゾーン資源制御を設定する場合、<code>-p</code> が必要になります。<code>-e</code> または <code>-d</code> も併せて指定された場合、新しい <code>rctl</code> へのアクションの設定も行われます。</p> <p><code>-v</code> が指定され、かつ <code>-e</code>、<code>-d</code>、<code>-r</code>、<code>-x</code> のいずれも指定されなかった場合、以前のリリースとの互換性を維持する目的でこのオプションが暗黙のうちに仮定されます。</p> <p>リソース制御値の設定時に大きな値を表現するために使用可能な単位修飾子や倍率については、<a href="#">resource_controls(5)</a> を参照してください。</p> |
| <code>-t [basic privileged system]</code> | <p>設定するリソース制御タイプを指定します。「lowerable」フラグが設定されたリソース制御でない限り、特権付きリソース制御を変更できるのは、<code>root</code> と同等の特権を持つユーザー (または <code>setuid</code> プログラム) による呼び出しだけです。<code>RCTL_GLOBAL_LOWERABLE</code> フラグについては、<a href="#">rctlblk_set_value(3C)</a> を参照してください。タイプが指定されなかった場合、<code>basic</code> が仮定されます。取得処理では、タイプが指定され</p>                                                                                                                                                                                                                                       |

なかった場合、`system`を含むすべてのリソース制御タイプの値が表示されます。

`-v value` 設定処理用として、リソース制御の値を指定します。`value`が指定されなかった場合、指定されたタイプのうち、最小の値を持つリソース制御に対して、変更(削除、アクションの有効化、またはアクションの無効化)が実行されます。

リソース制御値の設定時に大きな値を表現するために使用可能な単位修飾子や倍率については、[resource\\_controls\(5\)](#)を参照してください。

`-x` 指定されたリソース制御値を削除します。削除オプションが指定されなかった場合のprctlのデフォルト操作は、値や特権の一致したリソース制御値を変更するか、指定された特権を使って新しい値を挿入することです。一致条件の詳細については、[setrctl\(2\)](#)を参照してください。

`-d`、`-e`、`-v`、`-x`のいずれのオプションも指定されなかった場合、その呼び出しは取得処理とみなされます。

オペランド 次のオペランドを指定できます。

`id` 問い合わせるエンティティー(`process`、`task`、`project`、`zone`のいずれか)のID。呼び出し元のユーザーの資格が特権付きでなく、かつ問い合わせ対象のエンティティーが所有している資格が異なっている場合、処理が失敗します。`id`が指定されなかった場合、エラーメッセージが返されます。

使用例 例1 現在のリソース制御設定の表示

次の例では、現在のシェルが所属するタスクの現在のリソース制御設定を表示しています。

```
example$ ps -o taskid -p $$
TASKID
8
example$ prctl -i task 8
136150: /bin/ksh
NAME PRIVILEGE VALUE FLAG ACTION RECIPIENT
task.max-cpu-time
 usage 8s
 system 18.4Es inf none -
task.max-lwps
 usage 39
 system 2.15G max deny -
project.max-contracts
```

例1 現在のリソース制御設定の表示 (続き)

|                           |        |     |      |   |
|---------------------------|--------|-----|------|---|
| privileged                | 10.0K  | -   | deny | - |
| project.max-locked-memory |        |     |      |   |
| usage                     | 0B     |     |      |   |
| privileged                | 508MB  | -   | deny | - |
| project.max-port-ids      |        |     |      |   |
| privileged                | 8.19K  | -   | deny | - |
| project.max-shm-memory    |        |     |      |   |
| privileged                | 508MB  | -   | deny | - |
| project.max-shm-ids       |        |     |      |   |
| privileged                | 128    | -   | deny | - |
| project.max-msg-ids       |        |     |      |   |
| privileged                | 128    | -   | deny | - |
| project.max-sem-ids       |        |     |      |   |
| privileged                | 128    | -   | deny | - |
| project.max-crypto-memory |        |     |      |   |
| usage                     | 0B     |     |      |   |
| privileged                | 508MB  | -   | deny | - |
| project.max-tasks         |        |     |      |   |
| usage                     | 2      |     |      |   |
| system                    | 2.15G  | max | deny | - |
| project.max-lwps          |        |     |      |   |
| usage                     | 39     |     |      |   |
| system                    | 2.15G  | max | deny | - |
| project.cpu-shares        |        |     |      |   |
| usage                     | 1      |     |      |   |
| privileged                | 1      | -   | none | - |
| zone.max-shm-memory       |        |     |      |   |
| system                    | 16.0EB | max | deny | - |
| zone.max-shm-ids          |        |     |      |   |
| system                    | 16.8M  | max | deny | - |
| zone.max-sem-ids          |        |     |      |   |
| system                    | 16.8M  | max | deny | - |
| zone.max-msg-ids          |        |     |      |   |
| system                    | 16.8M  | max | deny | - |
| zone.max-lwps             |        |     |      |   |
| system                    | 2.15G  | max | deny | - |
| zone.cpu-shares           |        |     |      |   |
| privileged                | 1      | -   | none | - |
| zone.max-locked-memory    |        |     |      |   |
| usage                     | 0B     |     |      |   |
| privileged                | 508MB  | -   | deny | - |

例2 特定の制御の値の表示、置換、および確認

次の例では、ある既存プロジェクト上の特定の制御の値を表示、置換、および確認しています。



## 例2 特定の制御の値の表示、置換、および確認 (続き)

```
example# prctl -n project.cpu-shares -i project group.staff
project: 10: group.staff
NAME PRIVILEGE VALUE FLAG ACTION RECIPIENT
project.cpu-shares
 usage 1
 privileged 1 - none -
 system 65.5K max none -
```

```
example# prctl -n project.cpu-shares -v 10 -r -i project group.staff
example# prctl -n project.cpu-shares -i project group.staff
project: 10: group.staff
NAME PRIVILEGE VALUE FLAG ACTION RECIPIENT
project.cpu-shares
 usage 10
 privileged 10 - none -
 system 65.5K max none -
```

## 例3 リソースの調整

次の例では、project.max-locked-memory リソースを使用しています。

最初に、id -p を使用して、現在のシェルが所属しているプロジェクトを検出します。

```
/home/garfield> id -p
uid=77880(garfield) gid=10(staff) projid=10(group.staff)
```

変更を行う前に、ターゲットプロジェクトを使用してリソースの制限値を識別します。

```
/home/garfield> prctl -n project.max-locked-memory -i project \
group.staff
project 10: group.staff
project.max-locked-memory
 privileged 256MB - deny -
 system 16.0EB max deny -
```

current limit is 256 Megabytes.

次に、ターゲットプロジェクトに対して、project.max-locked-memory の制限を 300M バイトに調整します。

```
prctl -n project.max-locked-memory -v 300M -r -i project group.staff
```

変更後のリソースの制限値が、新しい値 300M バイトを示しています。

## 例3 リソースの調整 (続き)

```
prctl -n project.max-locked-memory -i project group.staff
project 10:group.staff
project.max-locked-memory
usage 200MG
privileged 300MB - deny
system 16.0EB max deny
```

## 例4 プロジェクトのCPUキャップを変更する

prctl コマンドは、project.cpu-cap リソース制御 ([resource\\_controls\(5\)](#) を参照) を使用して、プロジェクトのCPUキャップを設定したり変更したりすることができます。( /etc/project ファイルでも同じリソース制御を使用できます。 [project\(4\)](#) を参照) 次のコマンドは、user.smith を3つのCPUに制限するようにCPUキャップを変更します。

```
prctl -r -t privileged -n project.cpu-cap -v 300 -i project user.smith
```

上で使用されている prctl -r オプションは、プロジェクトまたはゾーンのCPUキャップを動的に変更するために使用されます。たとえば、次のコマンドは、前のコマンドで設定されたキャップを80%に変更します。

```
prctl -r -t privileged -n project.cpu-cap -v 80 -i project user.smith
```

CPUキャップを削除するには、次のように入力します。

```
prctl -x -n project.cpu-cap $$
```

## 例5 ゾーンのCPUキャップを変更する

prctl コマンドは、zone.cpu-cap リソース制御 ([resource\\_controls\(5\)](#) を参照) を使用して、ゾーンのCPUキャップを設定したり変更したりすることができます。( [zonecfg\(1M\)](#) コマンドでも同じリソース制御を操作できます。) 次のコマンドは、大域ゾーンをCPUの80%に制限するようにCPUキャップを変更します。

```
prctl -t privileged -n zone.cpu-cap -v 80 -i zone global
```

キャップを50%に削減するには、次のコマンドを使用します。

```
prctl -r -t privileged -n zone.cpu-cap -v 50 -i zone global
```

終了ステータス 次の終了ステータスが返されます。

- 0 成功。
- 1 致命的なエラーが発生しました。
- 2 無効なコマンド行オプションが指定された。

ファイル

/proc/pid/\*      プロセス情報および制御ファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

コマンド行の構文は「確実」です。人間が読める形式の出力は「不確実」です。機械可読な出力は「確実」です。

関連項目

[rctldm\(1M\)](#), [zonecfg\(1M\)](#), [setrctl\(2\)](#), [rctlblk\\_get\\_local\\_action\(3C\)](#), [project\(4\)](#), [attributes\(5\)](#), [resource\\_controls\(5\)](#)

注意事項

ローカルアクションを許可するリソース制御ブロック上に設定可能な有効なシグナルは、SIGABRT、SIGXRES、SIGHUP、SIGSTOP、SIGTERM、およびSIGKILLです。これに加え、CPU時間関連の制御はSIGXCPUシグナルを発行でき、ファイルサイズ関連の制御はSIGXFSZシグナルを送信できます。

|         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | preap - すでに存在しないプロセスの親による取得の強制                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| 形式      | preap [-F] pid...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 機能説明    | <p>すでに存在しない (またはゾンビ) プロセスとは、終了ステータスが親によってまだ取得されていないプロセスです。終了ステータスは、<a href="#">wait(3C)</a>、<a href="#">waitid(2)</a>、または <a href="#">waitpid(3C)</a> システムコールによって取得されます。通常のシステム運用では、ゾンビが発生する可能性はありますが、一般的には短時間存在するのみです。これは、親が一部の子またはすべての子の終了ステータスを取得せずに終了した場合に発生することがあります。この場合、それらの子の親は <code>PID 1</code> に設定されます。このようなプロセスを定期的に取得する <a href="#">init(1M)</a> を参照してください。</p> <p>中途な状態の親プロセスは長時間終了できず、ゾンビがシステムに残ります。オペレーティングシステムは、プロセスが存在しなくなる前にプロセスのほとんどすべてのコンポーネントを破棄するので、通常、そのような存在しないプロセスはシステム動作に影響しません。ただし、それらは少量のシステムメモリーを消費します。</p> <p><i>pid</i> がすでに存在しないプロセスを表す場合、preap は、<i>pid</i> によって指定されたプロセスの親が <i>pid</i> に対して <a href="#">waitid(3C)</a> を実行するように強制します。</p> <p>preap は、次の場合に、親によってまもなく取得される可能性がある子プロセスを、管理者が無分別に取得させることを防止しようとします。</p> <ul style="list-style-type: none"><li>■ プロセスは <a href="#">init(1M)</a> の子である。</li><li>■ 親プロセスが停止していて、子の実行が再度許可されるのを待機している可能性があります。</li><li>■ プロセスが存在しなくなってから 1 分間経過していない。</li></ul> |
| オプション   | <p>次のオプションがサポートされています。</p> <p>-F    安全検査を無効にして、親が子を取得するように強制します。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| オペランド   | <p>次のオペランドを指定できます。</p> <p><i>pid</i>    プロセス ID リスト。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 使用法     | <p>-F フラグを使用するときは注意してください。2 つの制御プロセスを 1 つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の <code>proc</code> ツールの適用時に何も実行していない場合のみ、安全性が確保されます。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 終了ステータス | <p>次の終了値が preap によって返されます。これは、終了させられた各ターゲットプロセスの終了値を出力しています。</p> <p>0            正常に処理されました。</p> <p>0 以外      失敗 (該当するプロセスがない、アクセス権が拒否された、オプションが無効など)。</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値                              |
|-------|----------------------------------|
| 使用条件  | system/extended-system-utilities |

関連項目 [proc\(1\)](#), [init\(1M\)](#), [waitid\(2\)](#), [wait\(3C\)](#), [waitpid\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#)

警告 `preap` は、存在しないプロセスが親プロセスによって取得されていないことを管理者または開発者が確認した場合にのみ、慎重に適用してください。そうしないと、`preap` を適用したときに、予期しない動作で親プロセスに障害が発生する可能性があります。

名前 `print` – 画面またはウィンドウに文字を出力するシェル組み込み関数

## 形式

`/usr/bin/print` `print [-CRenprsv] [-f format] [-u fd] [string...]`

`ksh88` `print [-Rnprsu [n]] [arg]...`

`ksh` `print [-CRenprsv] [-f format] [-u fd] [string...]`

## 機能説明

`ksh88` シェルの出力メカニズムです。オプションを省略した場合、または `-` オプションを指定した場合は、[echo\(1\)](#) で述べるように標準出力上に引数を表示します。`-` オプションを指定した場合は、`-` で始まる場合でも、これに続くものはすべて引数として処理されます。

`/usr/bin/print`、`ksh` デフォルトでは、`print` は各文字列オペランドを標準出力に書き込み、復帰改行文字を付加します。

`-r`、`-R`、または `-f` オプションを指定しないかぎり、各文字列オペランドの各 `\` 文字は特別に次のように処理されます。

- `\a` 警告文字
- `\b` バックスペース文字
- `\c` 復帰改行を付加せずに出力を終了します。残りの文字列オペランドは無視されます。
- `\E` エスケープ文字 (ASCII 8 進数 033)
- `\f` 用紙送り文字
- `\n` 復帰改行文字
- `\t` タブ文字
- `\v` 垂直タブ文字
- `\\` バックスラッシュ文字
- `\0x` ASCII コードを 1、2、または 3 桁の 8 進数 *x* で表した 8 ビット文字。

## オプション

`ksh88` では次のオプションがサポートされています。

- `-n` 復帰改行 (`new-line`) の出力を抑制します。
- `-r-R` (`raw` モード) `echo` のエスケープ規則を無視します。`-R` オプションは、`-n` を除く後続の引数およびオプションすべてを表示します。

- p 標準出力の代わりに |& で生成されたプロセスのパイプ上に引数を出力します。
- s 標準出力の代わりに 履歴ファイル上に引数を書き込みます。
- u [*n*] 出力を格納するファイル記述子番号を、1 桁の数値 *n* で指定します。デフォルトは 1 です。

/usr/bin/print、ksh /usr/man/print および ksh では、次のオプションがサポートされています。

- e -f を指定しないかぎり、上記で説明したとおりに、各文字列オペランドで \シーケンスを処理します。これはデフォルトの動作です。  
  
-e と -r の両方が指定された場合は、最後に指定されたオプションが使用されます。
- f *format* 書式文字列 *format* を使用して文字列引数を書き込み、復帰改行文字を付加しません。書式を指定する方法については、[printf\(1\)](#) を参照してください。  
  
-f オプションが指定されていて、文字列オペランドが書式指定子よりも多い場合は、書式文字列は最初から再処理されます。文字列オペランドが書式指定子よりも少ない場合は、最初の不要な書式指定子が検出された時点で出力が終了します。
- n 復帰改行文字を出力に付加しません。
- p 標準出力の代わりに、現在の並行プロセスに書き込みます。
- r
- R 上記で説明したとおりに、各文字列オペランドで \シーケンスを処理しません。  
  
-e と -r の両方が指定された場合は、最後に指定されたオプションが使用されます。
- s 標準出力の代わりに、シェル履歴ファイルのエントリとして出力を書き込みます。
- u *fd* 標準出力の代わりに、ファイル記述子番号 *fd* に書き込みます。デフォルト値は 1 です。
- v 各文字列を変数名として処理し、値を %B 形式で書き込みます。-f オプションと同時に使用できません。
- C 各文字列を変数名として処理し、値を %#B 形式で書き込みます。-f オプションと同時に使用できません。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。

>0 書き込み用出力ファイルが開かない。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [echo\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [printf\(1\)](#), [attributes\(5\)](#)



**名前** printenv – display environment variables currently set

**形式** /usr/ucb/printenv [*variable*]

**機能説明** printenv prints out the values of the variables in the environment. If a *variable* is specified, only its value is printed.

**属性** See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE   |
|----------------|-------------------|
| Availability   | compatibility/ucb |

**関連項目** [csh\(1\)](#), [echo\(1\)](#), [sh\(1\)](#), [stty\(1\)](#), [tset\(1B\)](#), [attributes\(5\)](#), [environ\(5\)](#)

**診断** If a *variable* is specified and it is not defined in the environment, printenv returns an exit status of 1.

名前                    printf – write formatted output

形式

/usr/bin/printf        printf *format* [*argument*]...

ksh                    printf *format* [*string*...]

機能説明

/usr/bin/printf        The printf utility writes each string operand to standard output using *format* to control the output format.

オペランド

/usr/bin/printf        The following operands are supported by /usr/bin/printf:

- format*                A string describing the format to use to write the remaining operands. The *format* operand is used as the *format* string described on the [formats\(5\)](#) manual page, with the following exceptions:
- A SPACE character in the format string, in any context other than a flag of a conversion specification, is treated as an ordinary character that is copied to the output.
  - A character in the format string is treated as a character, not as a SPACE character.
  - In addition to the escape sequences described on the [formats\(5\)](#) manual page (`\\`, `\a`, `\b`, `\f`, `\n`, `\r`, `\t`, `\v`), `\ddd`, where *ddd* is a one-, two- or three-digit octal number, is written as a byte with the numeric value specified by the octal number.
  - The program does not precede or follow output from the *d* or *u* conversion specifications with blank characters not specified by the *format* operand.
  - The program does not precede output from the *o* conversion specification with zeros not specified by the *format* operand.
  - An additional conversion character, *b*, is supported as follows. The argument is taken to be a string that can contain backslash-escape sequences. The following backslash-escape sequences are supported:
    - the escape sequences listed on the [formats\(5\)](#) manual page (`\\`, `\a`, `\b`, `\f`, `\n`, `\r`, `\t`, `\v`), which are converted to the characters they represent
    - `\0ddd`, where *ddd* is a zero-, one-, two- or three-digit octal number that is converted to a byte with the numeric value specified by the octal number
    - `\c`, which is written and causes printf to ignore any remaining characters in the string operand containing it, any remaining string operands and any additional characters in the *format* operand.

The interpretation of a backslash followed by any other sequence of characters is unspecified.

Bytes from the converted string are written until the end of the string or the number of bytes indicated by the precision specification is reached. If the precision is omitted, it is taken to be infinite, so all bytes up to the end of the converted string are written. For each specification that consumes an argument, the next argument operand is evaluated and converted to the appropriate type for the conversion as specified below. The *format* operand is reused as often as necessary to satisfy the argument operands. Any extra *c* or *s* conversion specifications are evaluated as if a null string argument were supplied; other extra conversion specifications are evaluated as if a zero argument were supplied. If the *format* operand contains no conversion specifications and *argument* operands are present, the results are unspecified. If a character sequence in the *format* operand begins with a *%* character, but does not form a valid conversion specification, the behavior is unspecified.

*argument* The strings to be written to standard output, under the control of *format*. The *argument* operands are treated as strings if the corresponding conversion character is *b*, *c* or *s*. Otherwise, it is evaluated as a C constant, as described by the ISO C standard, with the following extensions:

- A leading plus or minus sign is allowed.
- If the leading character is a single- or double-quote, the value is the numeric value in the underlying codeset of the character following the single- or double-quote.

If an argument operand cannot be completely converted into an internal value appropriate to the corresponding conversion specification, a diagnostic message is written to standard error and the utility does not exit with a zero exit status, but continues processing any remaining operands and writes the value accumulated at the time the error was detected to standard output.

*ksh* The *format* operands support the full range of ANSI C/C99/XPG6 formatting specifiers as well as additional specifiers:

*%b* Each character in the string operand is processed specially, as follows:

- \a* Alert character.
- \b* Backspace character.
- \c* Terminate output without appending NEWLINE. The remaining string operands are ignored.
- \E* Escape character (ASCII octal 033).
- \f* FORM FEED character.

|                  |                                                                                                                                                                                                             |
|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>\n</code>  | NEWLINE character.                                                                                                                                                                                          |
| <code>\t</code>  | TAB character.                                                                                                                                                                                              |
| <code>\v</code>  | Vertical tab character.                                                                                                                                                                                     |
| <code>\\</code>  | Backslash character.                                                                                                                                                                                        |
| <code>\0x</code> | The 8-bit character whose ASCII code is the 1-, 2-, or 3-digit octal number <i>x</i> .                                                                                                                      |
| <code>%B</code>  | Treat the argument as a variable name and output the value without converting it to a string. This is most useful for variables of type <code>-b</code> .                                                   |
| <code>%H</code>  | Output string with characters <code>&lt;</code> , <code>&amp;</code> , <code>&gt;</code> , <code>"</code> , and non-printable characters, properly escaped for use in HTML and XML documents.               |
| <code>%P</code>  | Treat <i>string</i> as an extended regular expression and convert it to a shell pattern.                                                                                                                    |
| <code>%q</code>  | Output <i>string</i> quoted in a manner that it can be read in by the shell to get back the same string. However, empty strings resulting from missing string operands are not quoted.                      |
| <code>%R</code>  | Treat <i>string</i> as an shell pattern expression and convert it to an extended regular expression.                                                                                                        |
| <code>%T</code>  | Treat <i>string</i> as a date/time string and format it. The <code>T</code> can be preceded by ( <i>dformat</i> ), where <i>dformat</i> is a date format as defined by the <a href="#">date(1)</a> command. |
| <code>%Z</code>  | Output a byte whose value is 0.                                                                                                                                                                             |

When performing conversions of *string* to satisfy a numeric format specifier, if the first character of *string* is `"` or `'`, the value is the numeric value in the underlying code set of the character following the `"` or `'`. Otherwise, *string* is treated like a shell arithmetic expression and evaluated.

If a *string* operand cannot be completely converted into a value appropriate for that format specifier, an error occurs, but remaining *string* operands continue to be processed.

In addition to the format specifier extensions, the following extensions of ANSI C/C99/XPG6 are permitted in format specifiers:

- The escape sequences `\E` and `\e` expand to the escape character which is octal 033 in ASCII.
- The escape sequence `\cx` expands to CTRL-*x*.
- The escape sequence `\C[.name.]` expands to the collating element *name*.
- The escape sequence `\x{hex}` expands to the character corresponding to the hexadecimal value *hex*.
- The format modifier flag `=` can be used to center a field to a specified width. When the output is a terminal, the character width is used rather than the number of bytes.

- Each of the integral format specifiers can have a third modifier after width and precision that specifies the base of the conversion from 2 to 64. In this case, the # modifier causes *base#* to be prepended to the value.
- The # modifier can be used with the d specifier when no base is specified to cause the output to be written in units of 1000 with a suffix of one of k M G T P E.
- The # modifier can be used with the i specifier to cause the output to be written in units of 1024 with a suffix of one of Ki Mi Gi Ti Pi Ei.

If there are more *string* operands than format specifiers, the format string is reprocessed from the beginning. If there are fewer *string* operands than format specifiers, then *string* specifiers are treated as if empty strings were supplied, numeric conversions are treated as if 0 was supplied, and time conversions are treated as if now was supplied.

/usr/bin/printf is equivalent to ksh's printf built-in and print -f, which allows additional options to be specified.

## 使用法

/usr/bin/printf

The printf utility, like the [printf\(3C\)](#) function on which it is based, makes no special provision for dealing with multi-byte characters when using the %c conversion specification. Applications should be extremely cautious using either of these features when there are multi-byte characters in the character set.

Field widths and precisions cannot be specified as \*.

The %b conversion specification is not part of the ISO C standard; it has been added here as a portable way to process backslash escapes expanded in string operands as provided by the echo utility. See also the USAGE section of the [echo\(1\)](#) manual page for ways to use printf as a replacement for all of the traditional versions of the echo utility.

If an argument cannot be parsed correctly for the corresponding conversion specification, the printf utility reports an error. Thus, overflow and extraneous characters at the end of an argument being used for a numeric conversion are to be reported as errors.

It is not considered an error if an argument operand is not completely used for a c or s conversion or if a string operand's first or second character is used to get the numeric value of a character.

## 使用例

/usr/bin/printf  
Examples

例1 Printing a Series of Prompts

The following example alerts the user, then prints and reads a series of prompts:

```
example% printf "\aPlease fill in the following: \nName: "
read name
```

例 1    Printing a Series of Prompts      (続き)

```
printf "Phone number: "
read phone
```

例 2    Printing a Table of Calculations

The following example prints a table of calculations. It reads out a list of right and wrong answers from a file, calculates the percentage correctly, and prints them out. The numbers are right-justified and separated by a single tab character. The percentage is written to one decimal place of accuracy:

```
example% while read right wrong ; do
 percent=$(echo "scale=1;($right*100)/($right+$wrong)" | bc)
 printf "%2d right\t%2d wrong\t(%%)\n" \
 $right $wrong $percent
done < database_file
```

例 3    Printing number strings

The command:

```
example% printf "%5d%4d\n" 1 21 321 4321 54321
```

produces:

```
1 21
3214321
54321 0
```

The *format* operand is used three times to print all of the given strings and that a 0 was supplied by `printf` to satisfy the last `%4d` conversion specification.

例 4    Tabulating Conversion Errors

The following example tabulates conversion errors.

The `printf` utility tells the user when conversion errors are detected while producing numeric output. These results would be expected on an implementation with 32-bit twos-complement integers when `%d` is specified as the *format* operand:

| Arguments   | Standard    | Diagnostic                             |
|-------------|-------------|----------------------------------------|
| 5a          | 5           | printf: 5a not completely converted    |
| 9999999999  | 2147483647  | printf: 9999999999: Results too large  |
| -9999999999 | -2147483648 | printf: -9999999999: Results too large |
| ABC         | 0           | printf: ABC expected numeric value     |

The value shown on standard output is what would be expected as the return value from the function `strtol(3C)`. A similar correspondence exists between `%u` and `strtoul(3C)`, and `%e`, `%f` and `%g` and `strtod(3C)`.

例 5    Printing Output for a Specific Locale

The following example prints output for a specific locale. In a locale using the ISO/IEC 646:1991 standard as the underlying codeset, the command:

```
example% printf "%d\n" 3 +3 -3 \'3 \' +3 \' -3"
```

produces:

|    |                                                                             |
|----|-----------------------------------------------------------------------------|
| 3  | Numeric value of constant 3                                                 |
| 3  | Numeric value of constant 3                                                 |
| -3 | Numeric value of constant -3                                                |
| 51 | Numeric value of the character '3' in the ISO/IEC 646:1991 standard codeset |
| 43 | Numeric value of the character '+' in the ISO/IEC 646:1991 standard codeset |
| 45 | Numeric value of the character '-' in the SO/IEC 646:1991 standard codeset  |

In a locale with multi-byte characters, the value of a character is intended to be the value of the equivalent of the `wchar_t` representation of the character.

If an argument operand cannot be completely converted into an internal value appropriate to the corresponding conversion specification, a diagnostic message is written to standard error and the utility does exit with a zero exit status, but continues processing any remaining operands and writes the value accumulated at the time the error was detected to standard output.

ksh Examples

The following examples illustrate the use of the ksh93 version of `printf`.

例 6    Alternative floating point representation 1

The `printf` utility supports an alternative floating point representation (see `printf(3C)` entry for the `"%a"/"%A"`), which allows the output of floating-point values in a format that avoids the usual base16 to base10 rounding errors.

```
example% printf "%a\n" 2 3.1 NaN
```

produces:

```
0x1.00000000000000000000000000000000p+01
0x1.8cccccccccccccccccccccccccdp+01
nan
```

**例 7** Alternative floating point representation 2

The following example shows two different representations of the same floating-point value.

```
example% x=2 ; printf "%f == %a\n" x x
```

produces:

```
2.000000 == 0x1.00000000000000000000000000000000p+01
```

**例 8** Output of unicode values

The following command will print the EURO unicode symbol (code-point 0x20ac).

```
example% LC_ALL=en_US.UTF-8 printf "\u[20ac]\n"
```

produces:

```
<euro>
```

where <euro> represents the EURO currency symbol character.

**例 9** Convert unicode character to unicode code-point value

The following command will print the hexadecimal value of a given character.

```
example% export LC_ALL=en_US.UTF-8
```

```
example% printf "%x\n" "'<euro>"
```

where <euro> represents the EURO currency symbol character (code-point 0x20ac).

produces:

```
20ac
```

**例 10** Print the numeric value of an ASCII character

```
example% printf "%d\n" "'A"
```

produces:

```
65
```

**例 11** Print the language-independent date and time format

To print the language-independent date and time format, the following statement could be used:

```
example% printf "format" weekday month day hour min
```

For example,

```
$ printf format "Sunday" "July" 3 10 2
```

For American usage, format could be the string:



例 11 Print the language-independent date and time format (続き)

```
"%s, %s %d, %d:%.2d\n"
```

producing the message:

```
Sunday, July 3, 10:02
```

Whereas for EU usage, format could be the string:

```
"%1$s, %3$d. %2$s, %4$d:%5$.2d\n"
```

Note that the '\$' characters must be properly escaped, such as

```
"%1\\$s, %3\\$d. %2\\$s, %4\\$d:%5\\$.2d\n"
```

producing the message:

```
Sunday, 3. July, 10:02
```

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of printf: LANG, LC\_ALL, LC\_CTYPE, LC\_MESSAGES, LC\_NUMERIC, and NLSPATH.

終了ステータス The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/printf

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE                    |
|---------------------|------------------------------------|
| Availability        | text/locale                        |
| CSI                 | Enabled                            |
| Interface Stability | Committed                          |
| Standard            | See <a href="#">standards(5)</a> . |

ksh

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE |
|---------------------|-----------------|
| Availability        | system/core-os  |
| Interface Stability | Uncommitted     |

関連項目 [awk\(1\)](#), [bc\(1\)](#), [date\(1\)](#), [echo\(1\)](#), [ksh\(1\)](#), [printf\(3C\)](#), [strtod\(3C\)](#), [strtol\(3C\)](#), [strtoul\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [formats\(5\)](#), [standards\(5\)](#)

**注意事項**

Using format specifiers (characters following '%') which are not listed in the [printf\(3C\)](#) or this manual page will result in undefined behavior.

Using escape sequences (the character following a backslash ('\')) which are not listed in the [printf\(3C\)](#) or this manual page will result in undefined behavior.

Floating-point values follow C99, XPG6 and IEEE 754 standard behavior and can handle values the same way as the platform's `|long double|` datatype.

Floating-point values handle the sign separately which allows signs for values like NaN (for example, `-nan`), Infinite (for example, `-inf`) and zero (for example, `-0.0`).

|      |                                                                                                                                                                                                                                                                                                               |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前   | priocntl - 指定したプロセスおよび LWP のスケジューリングパラメータの表示または設定                                                                                                                                                                                                                                                             |
| 形式   | <pre>priocntl -l</pre> <pre>priocntl -d [-i <i>idtype</i>] [<i>idlist</i>]</pre> <pre>priocntl -s [-c <i>class</i>] [<i>class-specific options</i>]   [-i <i>idtype</i>] [<i>idlist</i>]</pre> <pre>priocntl -e [-c <i>class</i>] [<i>class-specific options</i>] <i>command</i>   [<i>argument(s)</i>]</pre> |
| 機能説明 | <p>priocntl コマンドは、指定したプロセスまたは LWP のスケジューリングパラメータを表示または設定します。また、システムのプロセススケジューラの現在の設定情報を表示したり、指定したスケジューリングパラメータを使ってコマンドを実行したりする目的にも使用できます。</p>                                                                                                                                                                 |

プロセスと LWP はいくつかのクラスに分けられます。各クラスにはそれぞれ異なったスケジューリング方針が適用されます。現在サポートされているクラスは、リアルタイムクラス、タイムシェアリングクラス、対話型クラス、フェアシェアクラス、および固定優先順位クラスです。各クラスの特徴およびクラス固有のオプションについては、後述の「リアルタイムクラス」、「タイムシェアリングクラス」、「対話型クラス」、「フェアシェアクラス」、「固定優先順位クラス」のセクションで説明します。ユーザーは、正しいアクセス権を持っていれば、priocntl コマンドを使って、稼動中のプロセスまたは LWP のクラスや対応するスケジューリングパラメータを変更できます。

デフォルトの設定では、実行可能状態にあるリアルタイムプロセスまたは LWP がほかのプロセスに先だって実行されます。したがって、リアルタイムプロセスまたは LWP の使い方を誤ると、システムのパフォーマンスが著しく損なわれてしまう恐れがあります。

priocntl コマンドに *idlist* 引数を指定する場合は、コマンド行の最後に記述してください。また *idlist* 中の要素は、スペースで区切る必要があります。*idlist* 引数を省略するときは、*idtype* 引数として

pid、ppid、pgid、sid、taskid、class、uid、gid、projid、または zoneid を使って、priocntl コマンド自体のプロセス ID、親プロセス ID、プロセスグループ ID、セッション ID、タスク ID、クラス、ユーザー ID、グループ ID、プロジェクト ID、またはゾーン ID をそれぞれ指定します。

## コマンド

```
priocntl -d [-i idtype] [idlist]
```

これは、*idtype* および *idlist* 引数で指定したプロセス群のクラスとクラス固有のスケジューリングパラメータを表示します。

## コマンド

```
priocntl -s [-c class] [class-specific options] \
 [-i idtype] [idlist]
```

これは、指定したプロセスのクラスおよびクラス固有のパラメータを、コマンド行で指定した値に設定します。-c *class* オプションが、設定するクラスを表します。現在サポートされている *class* 引数値は、リアルタイムクラス用の RT、タイムシェアリングクラス用の TS、対話型クラス用の IA、フェアシェア用の FSS、固定優先順位用の FX のいずれかです。

各クラス固有のパラメータは、対応するクラス固有のオプションを使って指定します。詳しくは各クラスの項で後述します。-c *class* オプションを省略する場合には、*idtype* と *idlist* を使って、同じクラスに属する一群のプロセスまたは LWP を指定してください。そうしないとエラーが発生します。クラス固有のオプションをすべて省略すると、このプロセスのクラス固有パラメータの値は、-c *class* で指定したクラスのデフォルト値に設定されます。なお、-c *class* オプションも省略した場合には、そのプロセスの現在のクラスのデフォルト値に設定されます。

priocntl を使ってプロセスまたは LWP のスケジューリングパラメータを変更するには、priocntl を実行したユーザーの実ユーザー ID または実効ユーザー ID (それぞれのグループ ID) がこのプロセスまたは LWP の実ユーザー ID または実効ユーザー ID (それぞれのグループ ID) と一致しているか、ユーザーの実効ユーザー ID がスーパーユーザーを示している必要があります。アクセス権に関するこの要件は、全クラスに適用される最低限の必要条件です。クラスによっては、プロセスを設定する場合やクラス固有のスケジューリングパラメータを設定する場合に、他のアクセス権に関する要件が追加されることもあります。

*idtype* と *idlist* の両引数を使って (LWP のリストを指定して、または指定しないで) いくつかのプロセスを指定すると、priocntl はそれらのプロセスに対して処理を実行します。ただし、処理する順序は実装状態により異なります。エラーを検出した場合、そのエラーの種類により、以降のプロセスの処理を続行できる場合と続行できない場合とがあります。

検出したエラーがアクセス権に関連するものであれば、priocntl はエラーメッセージを出力しますが、設定されたプロセスのうちユーザーが適正なアクセス権を持っているものについては、パラメータをリセットして処理を続行します。エラーがアクセス権に関連したものでなければ、エラーメッセージを表示してただちに実行を終了します。

特殊なシステムプロセス (たとえばスワッププロセス) の実行をスケジューリングする目的のため、sys という特殊なスケジューリングクラスが存在しています。どのプロセスに対しても、クラスを sys に変更することはできません。また、*idtype* や *idlist* 引数で sys クラスに属するプロセスまたは LWP を指定した場合、priocntl はそれらを無視します。たとえば *idtype* として uid を指定し *idlist* に 0 (ゼロ) を指定すると、UID の値が 0 の全プロセスのうち sys クラスに属するものを除いたプロセス、および (-s オプションを使ってパラメータを変更する場合には) init プロセスが処理の対象となります。

init プロセス (プロセス ID は 1) は特殊なケースです。priocntl コマンドを使って init プロセスのクラスやその他のスケジューリングパラメータを変更するためには、*idtype* として *pid* を指定し、*idlist* として 1 だけを指定してください。init プロセスは、システム上に存在するどのクラスに割り当てても構いませんが、ほとんどの場合タイムシェアリングクラスを選択するのが適切です。ほかのクラスに割り当てると、システムに悪影響を及ぼすことがあります。詳細は『[Oracle Solaris の管理: 一般的なタスク](#)』を参照してください。

## コマンド

```
priocntl -e [-c class] [class-specific options] command \
 [argument...]
```

このコマンドは、*command* で示すコマンドを、指定したクラスとスケジューリングパラメータを使って実行します。*arguments* は、*command* に対する引数です。*-c class* オプションを省略すると、指定したコマンドはユーザーの現クラスで実行されます。

## オプション

サポートしているオプションは、次のとおりです。

### -c class

設定するクラスを指定します。現在サポートされている *class* 引数値は、リアルタイムクラス用の RT、タイムシェアリングクラス用の TS、対話型クラス用の IA、フェアシェア用の FSS、固定優先順位用の FX のいずれかです。指定したクラスが構成されていない場合には、自動的に構成されます。

### -d

設定したプロセスに対応したスケジューリングパラメータを表示します。

### -e

指定したコマンドを、指定したプロセスに対応したスケジューリングパラメータを使って実行します。

### -i idtype

このオプションは、*idlist* 引数 (もし存在していれば) とともに、priocntl コマンドの処理対象となる 1 つ以上のプロセスまたは LWP を指定します。*idlist* がどのように解釈されるかは、*idtype* の値により異なります。*-d* オプションまたは *-s* オプションを指定するときに、*-i idtype* オプションを指定しない場合、デフォルトの *idtype* として *pid* が想定されます。

*idtype* として指定可能な値、およびそれぞれの値に対応した *idlist* の解釈は以下のとおりです。

### -i all

priocntl コマンドの処理対象は、存在しているすべてのプロセスです。この場合、*idlist* 引数は指定できません。指定しても無視されます。また次に述べるアクセス権上の制限事項は、この *all* 指定の場合でも有効です。

**-i ctid**

*idlist* は、プロセスの契約 ID のリストです。priocntl コマンドの処理対象は、リスト内の ID と同一のプロセスの契約 ID を持つすべてのプロセスです。

**-i class**

*idlist* は単一のクラス名です (リアルタイム クラスなら RT、タイムシェアリング クラスなら TS、対話型クラスなら IA、フェアシェアクラスなら FSS、固定優先順位クラスなら FX)。priocntl コマンドの処理対象は、指定されたクラスに属するすべてのプロセスです。

**-i gid**

*idlist* は、グループ ID のリストです。priocntl コマンドは、このリスト内の ID に等しい実効グループ ID を持つすべてのプロセスに適用されます。

**-i pgid**

*idlist* はプロセスグループ ID のリストです。priocntl コマンドは、指定されたプロセスグループ内のすべてのプロセスに適用されます。

**-i pid[/lwps]**

*idlist* はプロセス ID のリストであり、各 ID のあとにはスラッシュ (/) と、コマンドで区切られた LWP ID のリストが続く可能性があります。LWP ID の範囲は、範囲内の最初と最後の項目をハイフン (-) で区切って示すことができます。

**-i ppid**

*idlist* は親プロセス ID のリストです。priocntl コマンドは、親プロセス ID がこのリスト内にあるすべてのプロセスに適用されます。

**-i projid**

*idlist* はプロジェクト ID のリストです。priocntl コマンドは、このリスト内の ID に等しい実効プロジェクト ID を持つすべてのプロセスに適用されます。

**-i sid**

*idlist* は、セッション ID のリストです。priocntl コマンドの処理対象は、指定されたセッション中のすべてのプロセスです。

**-i taskid**

*idlist* は、タスク ID のリストです。priocntl コマンドの処理対象は、指定されたタスクのすべてのプロセスです。

**-i uid**

*idlist* は、ユーザー ID のリストです。priocntl コマンドは、このリスト内の ID に等しい実効ユーザー ID を持つすべてのプロセスに適用されます。

**-i zoneid**

*idlist* はゾーン ID のリストです。priocntl コマンドは、このリスト内の ID に等しい実効ゾーン ID を持つすべてのプロセスに適用されます。

-l

現在システム上に定義されているクラスの一覧を、各クラス固有の情報とともに表示します。クラス固有情報の出力形式は「使用方法」の節で説明します。

-s

指定したプロセスに対応したスケジューリングパラメータを設定します。

リアルタイムクラスのパラメータを設定する場合には、以下のクラス固有のオプションを指定できます。

-p *rtpri*

指定したプロセスおよびLWPのリアルタイム優先順位を、*rtpri* に設定します。

-t *tqntm* [-r *res*]

指定したプロセスのタイムクアラムを、*tqntm* に設定します。タイムクアラムの単位を指定することも可能です(詳しくは後述)。

-q *tqsig*

指定したプロセスおよびLWPのリアルタイムのタイムクアラムシグナルを *tqsig* に設定します。

タイムシェアリングクラスのパラメータを設定する場合には、以下のクラス固有のオプションを指定できます。

-m *tsuprilim*

指定したプロセスおよびLWPのユーザー優先順位の制限値を、*tsuprilim* に設定します。

-p *tsupri*

指定したプロセスおよびLWPのユーザー優先順位を、*tsupri* に設定します。

対話型クラスのパラメータを設定する場合には、以下のクラス固有のオプションを指定できます。

-m *iauprilim*

指定したプロセスおよびLWPのユーザー優先順位の制限値を、*iauprilim* に設定します。

-p *iaupri*

指定したプロセスおよびLWPのユーザー優先順位を、*iaupri* に設定します。

フェアシェアクラスのパラメータを設定する場合には、以下のクラス固有のオプションを指定できます。

-m *fssuprilim*

指定したプロセスおよびLWPのユーザー優先順位の制限値を、*fssuprilim* に設定します。

-p *fssupri*

指定したプロセスおよびLWPのユーザー優先順位を、*fssupri* に設定します。

固定優先順位クラスのパラメータを設定する場合には、以下のクラス固有のオプションを指定できます。

**-m *fxuprilim***

指定したプロセスおよびLWPのユーザー優先順位の制限値を、*fxuprilim* に設定します。

**-p *fxupri***

指定したプロセスおよびLWPのユーザー優先順位を、*fxupri* に設定します。

**-t *tqntm***

**[-r *res*]** 指定したプロセスおよびLWPのタイムクアंटムを、*tqntm* に設定します。タイムクアंटムの単位を指定することも可能です (詳しくは後述)。

## 使用法

### リアルタイムクラス

リアルタイムクラスは、高速でかつ決定力のある応答を必要とし、スケジューリング方針に関してユーザーやアプリケーションに対する絶対的な制御権を必要とするようなプロセスに対して、固定された高い優先順位を持つスケジューリング方針を与えるものです。システム中にリアルタイムクラスが定義されている場合には、システム上の最上位のスケジューリング優先順位群を制御できる唯一のクラスである必要があります。これにより、実行可能状態にあるリアルタイムプロセスは、ほかのクラスに属するあらゆるプロセスよりも先にCPUのサービスを受けられることが保証できます。

リアルタイムクラスには、一連のリアルタイム優先順位値 (*rtpri*) が与えられていて、そのクラスに属するプロセスに割り当てることができます。リアルタイム優先順位値の範囲は0から*x*までで、この*x*の値はリアルタイムクラスのスケジューラを構成済みのシステムごとに設定可能であり、次のコマンドを使って表示することができます。

```
priocntl -l
```

リアルタイムクラスのスケジューリング方針とは、固定優先順位を割り当てることです。つまり、ユーザーやアプリケーション側からのプロセスの*rtpri* 値を変更する明示的な要求を実行しない限り、リアルタイムプロセスのスケジューリング優先順位は変わりません。

リアルタイムクラスのプロセスに関して、*rtpri* の値はプロセスのスケジューリング優先順位と実用上同じです。リアルタイムクラスのあるプロセスの、同じクラス内の他のプロセスに対する相対的なスケジューリング優先順位値は、*rtpri* 値により決定されます。*rtpri* 値は数値で示され、大きい数が高い優先順位を表します。リアルタイムクラスはシステム中で最も高いスケジューリング優先順位群を制御しているので、実行可能状態にあるリアルタイムプロセスのうち最高の*rtpri* 値を持つプロセスがシステム中の他のあらゆるプロセスに先だって選ばれる、ということが保証されています。



`priocntl` は、優先順位に関する制御権だけでなく、リアルタイムクラス中のプロセスに割り当てられるタイムクアンタムを制御する権利も提供しています。このタイムクアンタム値は、プロセスがリソースやイベントの待ち状態(休眠状態)に陥らないという前提で、プロセスの実行時間の最大値を指定するものです。なお、あるプロセスの稼動中に、より高い優先順位を持つ他のプロセスが実行可能状態になった場合、現在稼動中のプロセスは割り当てられたタイムクアンタムに達していません。もしも取り上げられてしまうことがあります。

## コマンド

```
priocntl -d [-i idtype] [idlist]
```

このコマンドは、*idtype* と *idlist* で指定した一群のリアルタイムプロセスの個々のリアルタイム優先順位、時間(単位はミリ秒)、およびタイムクアンタムシグナル値を表示します。

リアルタイムクラス用の `priocntl -s` または `priocntl -e` コマンドには、`-p`、`-t` [`-r`]、および `-q` を任意の組み合わせで指定できます。どれかを省略した場合、当該プロセスが現在リアルタイムであれば、省略されたオプションに対応したパラメータには影響ありません。あるプロセスをほかのクラスからリアルタイムクラスに変更するような `priocntl` コマンド中でいずれかのオプションを省略すると、そのオプションに対応したパラメータはデフォルト値に設定されます。*rtpri* のデフォルト値は 0 です。タイムクアンタムのデフォルトは、*rtpri* の値およびシステムの構成により異なります ([rt\\_dptbl\(4\)](#) 参照)。

`-t tqntm` オプションを使用する際、タイムクアンタムの単位を `-r res` オプションを使って指定することもできます。単位のデフォルトはミリ秒です。*res* には、1 から 1,000,000,000 までの正の整数を指定してください。この場合、使用される単位は *res* 分の 1 秒となります。たとえば、`-t 10 -r 100` と指定すると、単位は 100 分の 1 秒となり、タイムクアンタム長として指定されている 10 は、10/100 つまり 10 分の 1 秒を意味することになります。ナノ秒などの細かい単位も指定可能ですが、実際にはシステムクロックの精度に合うようにタイムクアンタム長は丸められてしまいます。タイムクアンタムとして 0 を指定した場合、およびシステムに固有の最大クアンタムを超える 大きな値を指定した場合には、エラーとなります。

リアルタイム時間シグナルを使用すると、実行中のリアルタイムプロセスに自身の時間消費量を知らせることができます。リアルタイム時間シグナルはリアルタイムプロセスを監視し、その実行時間が上限に達した時点で、構成されているシグナルをリアルタイムプロセスに送信します。時間シグナル *tqsig* のデフォルト値 (0) は、シグナルを送信しないことを示します。*tqsig* に正の値を指定すると、その値ごとに設定されているシグナルが送信されます。[kill\(1\)](#) などのシグナルを扱うコマンドの場合と同様に、`-q tqsig` オプションを指定して XCPU や KILL のようなシンボルを使用したシグナル名も扱うことができます。

プロセスのクラスを他のものからリアルタイムクラスに変更するには、`priocntl` コマンドを実行するユーザーがスーパーユーザー特権を持つ必要があります。また、リアルタイムプロセスの *rtpri* 値やタイムクアンタムを変更するには、`priocntl`

コマンドを実行するユーザーはスーパーユーザーであるか、あるいは現在リアルタイムクラスにいて(リアルタイムプロセスとして稼動中のシェル)その実ユーザー ID または実効ユーザー ID が処理対象のプロセスの実ユーザー ID または実効ユーザー ID と一致している必要があります。

`fork(2)` や `exec(2)` などのシステムコールを使った場合、リアルタイム優先順位、タイムクアンタム、およびタイムクアンタムシグナルは、これらのシステムコールに受け継がれます。`exec(2)` システムコールで、ユーザー定義のシグナルハンドラを持つ時間シグナルが使用された場合、指定された時間の上限に達する前に、新しいイメージの中にユーザーが定義のシグナルハンドラがインストールされている必要があります。そうでない場合、予測できない動作が発生します。

#### タイムシェアリングクラス(TSクラス)

タイムシェアリングのスケジューリング方針とは、CPU の消費に関して異なった性質を持つプロセスに対して、公平で有効な CPU リソースの割り付けを行うことです。タイムシェアリング方針の目的は、対話型プロセスには速いレスポンスを提供し、CPU 消費型のジョブには高いスループットを提供し、さらにスケジューリングの制御権をユーザーやアプリケーションにもある程度与えることにあります。

タイムシェアリングクラスには、一連のタイムシェアリングユーザー優先順位値(`rtpri`)が与えられていて、そのクラスに属するプロセスに割り当てることが出来ます。ユーザー優先順位値の範囲は  $-x$  から  $+x$  までで、この  $x$  の値はシステムごとに設定可能であり、以下のコマンドを使って表示することができます。

```
prioctl -l
```

ユーザー優先順位の目的は、タイムシェアリングクラスのプロセスのスケジューリングを、ユーザーやアプリケーションがある程度制御できるようにすることです。タイムシェアリングクラスのプロセスの `tsupri` 値を増減させれば、そのプロセスのスケジューリング優先順位が上下します。ただし、高い `tsupri` 値を持つタイムシェアリングプロセスが、低い `tsupri` 値を持つプロセスよりも先に実行されるとは限りません。なぜなら `tsupri` 値は、タイムシェアリングプロセスのスケジューリング優先順位を決定する際の要素の 1 つにしかすぎないからです。システムは最近のシステム利用率などほかの要素を考慮の上、タイムシェアリングプロセスの内部スケジューリング優先順位を動的に調整します。

`prioctl -l` で表示される、システム全体でのユーザー優先順位上限のほかに、プロセスごとにユーザー優先順位上限(`tsuprilim`)があります。この値は、プロセスの `tsupri` として設定できる最大の値を示します。

#### コマンド

```
prioctl -d [-i idtype] [idlist]
```

このコマンドは、`idtype` と `idlist` で指定した一群のタイムシェアリングプロセスの個々のユーザー優先順位と優先順位制限値を表示します。

タイムシェアリングプロセスは、自身の(および同じユーザー ID を持つほかのプロセスの) `tsuprilim` 値を下げるすることができます。`tsuprilim` 値を上げることができるの

は、スーパーユーザー特権を持つプロセスだけです。あるプロセスのクラスをタイムシェアリングから他のクラスに変更する場合、*tsuprilim* の初期値を 0 より大きい値に設定するには、スーパーユーザー特権が必要となります。

どのタイムシェアリングプロセスも、自身の (および同じユーザー ID を持つ他のプロセスの) *tsupri* 値を設定できますが、そのプロセスの *tsuprilim* 値を超えることはできません。*tsuprilim* を超える値に *tsupri* を設定しようとした場合、または *tsupri* 未満の値に *tsuprilim* を設定しようとした場合には、結果として *tsupri* が *tsuprilim* と等しい値に設定されます。

タイムシェアリングクラス用の `prioctl -s` または `prioctl -e` コマンド中に、`-m` と `-p` の両オプションを任意の組み合わせで指定できます。どちらかを省略した場合、当該プロセスが現在タイムシェアリングであれば、省略されたオプションに対応したパラメータには通常影響ありません。ただし、`-p` オプションを省略し、`-m` を使って *tsuprilim* の値を *tsupri* 未満に設定しようとした場合は例外です。このような設定を試みると、*tsupri* は現在の *tsuprilim* の値に設定されます。あるプロセスを他のクラスからタイムシェアリングクラスに変更する際にいずれかのオプションを省略すると、そのオプションに対応したパラメータはデフォルト値に設定されます。*tsuprilim* のデフォルト値は 0 です。*tsupri* のデフォルト値は、現在設定されている *tsuprilim* の値です。

`fork(2)` や `exec(2)` などのシステムコールを使った場合、タイムシェアリングユーザー優先順位 およびユーザー優先順位の制限値は受け継がれます。

#### 対話型クラス

対話型のスケジューリング方針とは、CPU の消費に関して異なった性質を持つプロセスに対して、公平で有効な CPU リソースの割り付けを行い、さらに対話用に速いレスポンスを提供することです。対話型方針の目的は、対話型プロセスには速いレスポンスを提供し、CPU 消費型のジョブには高いスループットを提供することにあります。対話型クラスにある処理の優先順位は、タイムシェアリングクラスにある処理と同じように変更できます。ただし、修正された優先順位はユーザーとの対話の速い応答性を提供するために継続します。

対話型クラスのユーザー優先順位の制限値 *iauprilim* は *tsuprilim* と同等です。対話型クラスの、プロセスごとのユーザー優先順位 *iaupri* は *tsupri* と同等です。

*iamode* (対話型モード) のビットが設定されている対話型クラスのプロセスには、最優先順位を示す値 10 が割り当てられます。この値を持つプロセスは、優先順位の計算が行われるとき、つまり、プロセスの優先順位が再調整されるたびに、そのプロセスのユーザーモード優先順位に組み込まれます。この機能は、X ウィンドウシステムで使用されています。X ウィンドウシステムでは、このビットを現在アクティブなウィンドウ内で実行中のプロセスに割り当てて、そのプロセスにより高い優先順位を与えています。

#### 公平共有クラス (FSS クラス)

フェアシェアのスケジューリング方針とは、所有するプロセスの数に関係なく、各プロジェクト間に公平に CPU リソースを割り付けることです。プロジェクトには、CPU リソースの使用権を制御するための「割り当て数」が与えられます。一定

時間ごとに CPU リソースの使用量が通知されるため、他のプロジェクトより使用量の多いプロジェクトの使用権は縮小され、使用量の少ないプロジェクトの使用権は拡大されます。CPU 時間は、各プロジェクトが所有するプロセスの数とは無関係に、プロジェクトごとの使用権に応じてスケジュールされます。

FSS スケジューリングクラスは、タイムシェアリングスケジューラとの互換性を保持するために、プロセス単位のユーザー優先順位およびユーザー優先順位上限という概念を取り入れています。フェアシェアスケジューラは、ユーザー優先順位の全領域について、均等な効果を提供することを目的としています。負の *fxsupri* 値を持つプロセスには、通常より頻度の低いタイムスライスが配分され、正の *fxsupri* 値を持つプロセスには、通常より頻度の高いタイムスライスが配分されます。ユーザー優先順位が割り当て数に影響することはありません。つまり、プロセスの *fxsupri* 値を変更しても、そのプロセスの属するプロジェクト全体の CPU 使用量は変更されません。CPU 使用量に影響するものは、ほかのプロジェクトとの比較によって割り付けられる割り当て数の配分のみです。

フェアシェアクラス内のプロセスの優先順位は、タイムシェアリングクラスのプロセスの優先順位と同じ方法で変更できます。

#### 固定優先順位クラス(FXクラス)

固定優先順位クラスは、システムによってプロセスのスケジューリング優先順位が調整されることなく、ユーザーまたはアプリケーションがプロセスのスケジューリング優先順位を制御できる、優先順位が固定された事前定義のスケジューリング方針を提供します。

固定優先順位クラスは、デフォルトではタイムシェアリングクラスとスケジューリング優先順位の同じ範囲を共有しています。固定優先順位クラスには、ユーザー優先順位値 (*fxupri*) の範囲が与えられ、そのクラスに属するプロセスに割り当てることができます。ユーザー優先順位値の範囲は 0 から *x* までで、この *x* の値はリアルタイムクラスのスケジューラを構成済みのシステムごとに設定可能であり、以下のコマンドを使って表示することができます。

```
priocntl -l
```

ユーザー優先順位の目的は、固定優先順位クラスのプロセスのスケジューリングを、ユーザーやアプリケーションが制御できるようにすることにあります。固定優先順位クラスのプロセスに関して、*fxupri* の値はプロセスのスケジューリング優先順位と実用上同じです。固定優先順位のあるプロセスの、同じクラス内の他のプロセスに対する相対的なスケジューリング優先順位値は、*fxupri* 値により決定されます。*fxupri* 値は数値で示され、大きい数が高い優先順位を表します。

`priocntl -l` で表示される、システム全体でのユーザー優先順位上限のほかに、プロセスごとにユーザー優先順位上限 (*fxuprilim*) があります。この値は、プロセスの *fxupri* として設定できる最大の値を示します。

固定優先順位プロセスは、自身の (および同じユーザー ID を持つ他のプロセスの) *fxuprilim* 値を下げるすることができます。*fxuprilim* 値を上げることができるのは、スーパーユーザー特権を持つプロセスだけです。あるプロセスのクラスを固定

優先順位から他のクラスに変更する場合、*fxuprilm* の初期値を 0 より大きい値に設定するには、スーパーユーザー特権が必要となります。

どの固定優先順位プロセスも、自身の (および同じユーザー ID を持つ他のプロセスの) *fxupri* 値を設定できますが、そのプロセスの *fxuprilm* 値を超えることはできません。*fxuprilm* を超える値に *fxupri* を設定しようとした場合、または *fxupri* 未満の値に *fxuprilm* を設定しようとした場合には、結果として *fxupri* が *fxuprilm* と等しい値に設定されます。

*prioctl* は、優先順位に関する制御権だけでなく、固定優先順位クラス中のプロセスに割り当てられるタイムクォンタムを制御する権利も提供しています。このタイムクォンタム値は、プロセスがリソースやイベントの待ち状態 (休眠状態) に陥らないという前提で、プロセスの実行時間の最大値を指定するものです。なお、あるプロセスの稼動中に、より高い優先順位を持つ他のプロセスが実行可能状態になった場合、現在稼動中のプロセスは割り当てられたタイムクォンタムに達していなくても取り上げられてしまうことがあります。

固定優先順位クラス用の *prioctl -s* または *prioctl -e* コマンドには、*-m*、*-p*、および *-t* を任意の組み合わせで指定できます。どれかを省略した場合、当該プロセスが現在固定優先順位であれば、省略されたオプションに対応したパラメータには影響ありません。ただし、*-p* を省略して、*-m* オプションで *fxuprilm* の値を現在の *fxupri* の値より小さな値に設定した場合は例外です。この場合、*fxupri* は、設定された *fxuprilm* の値と同じ値になります。あるプロセスを他のクラスから固定優先順位クラスに変更するときにオプションを省略すると、そのオプションに対応したパラメータはデフォルト値に設定されます。*fxuprilm* のデフォルト値は 0 です。*fxupri* の値は、設定されている *fxuprilm* の値とデフォルトで同じになります。タイムクォンタムのデフォルトは、設定されている *fxupri* の値およびシステムの構成により異なります (*fx\_dptbl(4)* 参照)。

固定優先順位クラス内のプロセスのタイムクォンタムは、リアルタイムクラスのプロセスのタイムクォンタムと同じ方法で変更できます。

*fork(2)* や *exec(2)* などのシステムコールを使った場合、固定優先順位のユーザー優先順位、ユーザー優先順位上限、およびタイムクォンタムは、これらのシステムコールに受け継がれます。

## 使用例

リアルタイムクラスの例を示します。

### 例1 クラスを設定する

*idtype* と *idlist* で指定された非リアルタイムのプロセスのクラスをリアルタイムに変更し、リアルタイム優先順位値をデフォルトの 0 に設定します。現在リアルタイムクラスに属しているプロセスのリアルタイム優先順位値は変わりません。指定されたプロセスのタイムクォンタムは、すべて 1/10 秒に設定されます。

```
example% prioctl -s -c RT -t 1 -r 10 -i idtype idlist
```



**例2 リアルタイムでコマンドを実行する**

*command* で指定したコマンドをリアルタイムクラスで実行します。リアルタイム優先順位値は5で、タイムクアンタムは20ミリ秒となります。

```
example% priocntl -e -c RT -p 15 -t 20 command
```

**例3 時間シグナルを指定して、リアルタイムでコマンドを実行する**

*command* で指定したコマンドをリアルタイムクラスで実行します。リアルタイム優先順位値は11、タイムクアンタムは250ミリ秒で、リアルタイム時間シグナルはSIGXCPUとなります。

```
example% priocntl -e -c RT -p 11 -t 250 -q XCPU command
```

タイムシェアリングクラスの例を示します。

**例4 非タイムシェアリングプロセスのクラスを設定する**

*idtype* と *idlist* で指定した非タイムシェアリングのプロセスのクラスをタイムシェアリングに変更し、ユーザー優先順位値と優先順位制限値を0に設定します。現在タイムシェアリングクラスに属しているプロセスには影響を与えません。

```
example% priocntl -s -c TS -i idtype idlist
```

**例5 タイムシェアリングクラスでコマンドを実行する**

*command* で指定したコマンドを、*arguments* で指定した引数を使ってタイムシェアリングクラスで実行します。ユーザー優先順位の制限値は0で、ユーザー優先順位は-15となります。

```
example% priocntl -e -c TS -m 0 -p -15 command [arguments]
```

**例6 固定優先順位クラスでコマンドを実行する**

*command* で指定したコマンドを固定優先順位クラスで実行します。ユーザー優先順位値上限は20、ユーザー優先順位は10で、タイムクアンタムは250ミリ秒となります。

```
example% priocntl -e -c FX -m 20 -p 10 -t 250 command
```

**例7 特定のLWPの優先順位の変更**

次の例では、プロセス500のLWP5のユーザー優先順位制限に20、ユーザー優先順位に15を設定します。

```
example% priocntl -s -m 20 -p 15 500/5
```

**終了ステータス** 次の終了ステータスが返されます。

-d、-l、-s オプションの場合:

- 0 正常終了。
- 1 エラーが発生しました。

-e オプションの場合:

正常終了を表す、実行されたコマンドの終了ステータスが返されます。それ以外の場合は、次のステータスが返されます。

- 1 コマンドは指定された優先順位で実行できなかった。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| CSI   | 有効             |

関連項目 [kill\(1\)](#), [nice\(1\)](#), [ps\(1\)](#), [dispadm\(1M\)](#), [exec\(2\)](#), [fork\(2\)](#), [priocntl\(2\)](#), [fx\\_dptbl\(4\)](#), [process\(4\)](#), [rt\\_dptbl\(4\)](#), [attributes\(5\)](#), [zones\(5\)](#), [FSS\(7\)](#)

『Oracle Solaris の管理: 一般的なタスク』

診断 priocntl は次に示すエラーメッセージを出力します。

Process(es) not found  
指定されたプロセスが1つも存在しません。

Specified processes from different classes  
-s オプションを使ってパラメータを設定しようとしたが、-c class オプションが省略されていて、複数のクラスのプロセスが指定されていました。

Invalid option or argument  
指定されたオプションまたは引数の中に、認識できないものまたは不正なものがあります。

名前 proc, pflags, pcred, pldd, psig, pstack, pfiles, pwdx, pstop, prun, pwait, ptime – proc ツール

形式

```

/usr/bin/pflags [-r] pid | core [/lwp] ...
/usr/bin/pcred [pid | core]...
/usr/bin/pcred [-u user/uid] [-g group/gid] [-G grouplist] pid...
/usr/bin/pcred -l login pid...
/usr/bin/pldd [-Fl] [pid | core]...
/usr/bin/psig [-n] pid...
/usr/bin/pstack [-F] pid | core [/lwp] ...
/usr/bin/pfiles [-Fn] pid...
/usr/bin/pwdx pid...
/usr/bin/pstop pid[/lwp] ...
/usr/bin/prun pid[/lwp] ...
/usr/bin/pwait [-v] pid...
/usr/bin/ptime [-Fm] [-p] pid...
/usr/bin/ptime [-m]command [arg]...
```

## 機能説明

proc ツールは、/proc ([proc\(4\)](#) を参照) の機能を使用するユーティリティです。これらのツールのほとんどには、プロセス ID (*pid*) のリストを指定できます。プロセス ID を指定できるツールでは、/proc/*nnn* もプロセス ID として認識されるため、シェル展開 /proc/\* を使用するとシステム内のすべてのプロセスを指定できます。

一部の proc ツールは、コアファイルに適用することもできます ([core\(4\)](#) を参照)。コアファイルに適用されるツールは、プロセス ID またはコアファイルの名前あるいはその両方のリストを認識します。

一部の proc ツールは、個別のスレッドに対して動作できます。ユーザーは、選択した (つまり、/thread-id をプロセス ID またはコアに追加した) スレッドだけを検査できます。複数のスレッドを選択するには、区切り文字「-」と「,」を使用します。たとえば、「/1,2,7-9」は、スレッド 1、2、7、8、および 9 を検査します。

「警告」を参照してください。

**pflags**     /proc 追跡フラグ (保留および維持シグナル)、およびその他の /proc ステータス情報をプロセスごとまたは各プロセスの指定された lwp ごとに表示します。

**pcred**     各プロセスの資格情報 (実効 UID、実 UID、保存された UID、実効 GID、実 GID、保存された GID) を表示または設定します。



|                     |                                                                                                                                                                                                                                                                                                                                         |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>pldd</code>   | 各プロセスにリンクされている ( <code>dlopen(3C)</code> ) を使用して明示的に接続されている共有オブジェクトを含む) 動的ライブラリを表示します。 <code>ldd(1)</code> も参照してください。                                                                                                                                                                                                                   |
| <code>psig</code>   | 各プロセスのシグナルの動作とハンドラを表示します。 <code>signal.h(3HEAD)</code> を参照してください。                                                                                                                                                                                                                                                                       |
| <code>pstack</code> | 16 進数とシンボリックのスタックトレースをプロセスごとまたは各プロセスの指定された <code>lwp</code> ごとに出力します。                                                                                                                                                                                                                                                                   |
| <code>pfiles</code> | 各プロセスで開かれているすべてのファイルに関する <code>fstat(2)</code> および <code>fcntl(2)</code> 情報を報告します。ネットワークエンドポイントの場合は、ローカル (および接続されている場合はピア) のアドレス情報も提供されます。ソケットの場合は、ソケットタイプ、ソケットオプション、および送信と受信のバッファサイズも提供されます。また、 <code>/proc/pid/path</code> から情報を使用できる場合は、ファイルへのパスが報告されます。これは、ファイルを開くときに使用された名前と必ずしも同じではありません。詳細については、 <code>proc(4)</code> を参照してください。 |
| <code>pwdx</code>   | 各プロセスの現在の作業用ディレクトリを表示します。                                                                                                                                                                                                                                                                                                               |
| <code>pstop</code>  | 各プロセスまたは指定された <code>lwp</code> を停止します ( <code>PR_REQUESTED</code> 停止)。                                                                                                                                                                                                                                                                  |
| <code>prun</code>   | 各プロセスまたは指定された <code>lwp</code> を実行するように設定します ( <code>pstop</code> の逆)。                                                                                                                                                                                                                                                                  |
| <code>pwait</code>  | 指定したすべてのプロセスの終了を待機します。                                                                                                                                                                                                                                                                                                                  |
| <code>ptime</code>  | <code>time(1)</code> と同様に、コマンドの時間を計測しますが、再現性のある精度を得るためにマイクロスタートアカウンティングを使用します。 <code>time(1)</code> とは異なり、コマンドの子の時間は計測されません。<br><br><code>-p pid</code> バージョンが使用された場合、指定された <code>pid</code> の時間統計のスナップショットを表示します。                                                                                                                      |

## オプション

次の一般的なオプションがサポートされています。

- `-F` 強制。別のプロセスが制御していても、ターゲットプロセスを捕捉します。
- `-n` (`psig` および `pfiles` のみ) 非冗長モードを設定します。`psig` は、名前ではなく、シグナルハンドラアドレスを表示します。`pfiles` は、各ファイル記述子の冗長情報を表示しません。代わりに、`pfiles` は、プロセスが `fstat(2)` を各ファイル記述子に適用した場合に取得される情報に出力を制限します。
- `-r` (`pflags` のみ) プロセスが停止された場合、プロセスのマシンレジスタを表示します。
- `-v` (`pwait` のみ) 冗長。終了を標準出力に報告します。

一般的なオプションのほかに、`pcrred` は次のオプションをサポートします。

- g group/gid**    ターゲットプロセスの実グループ ID (GID)、実効グループ ID、保存されたグループ ID を、指定した値に設定します。
- G grouplist**    ターゲットプロセスの補助 GID を、指定したグループリストに設定します。補助グループは、グループ名 ID のコンマ区切りリストで指定します。空のリストは、ターゲットプロセスの補助グループリストを消去します。
- l login**    ターゲットプロセスの実 UID、実効 UID、保存された UID を、指定したログインの UID に設定します。ターゲットプロセスの実 GID、実効 GID、保存された GID を、指定したログインの GID に設定します。補助グループリストを、指定したログインの補助グループリストに設定します。
- u user/uid**    ターゲットプロセスの実ユーザー ID (UID)、実効ユーザー ID、保存されたユーザー ID を、指定した値に設定します。

一般的なオプションのほかに、**pldd** は次のオプションをサポートします。

- l**    未解決の動的リンカーマップ名を表示します。

一般的なオプションのほかに、**ptime** は次のオプションをサポートします。

- m**    マイクロステートアカウンティング統計をすべて表示します。

表示されるフィールドは次のとおりです。

- real**    時計時間。
- user**    ユーザーレベルの CPU 時間。
- sys**    システムコールの CPU 時間。
- trap**    その他のシステムトラップの CPU 時間。
- tflt**    テキストページフォルトのスリープ時間。
- dflt**    データページフォルトのスリープ時間。
- kflt**    カーネルページフォルトのスリープ時間。
- lock**    ユーザーロック待機のスリープ時間。
- slp**    その他すべてのスリープ時間。
- lat**    CPU 遅延 (待機) 時間。
- stop**    停止時間。

- p pid**    指定された *pid* の時間統計情報のスナップショットを表示します。

別のプロセスの資格情報を設定するには、[setuid\(2\)](#) で説明されている規則に従って、ユーザー ID およびグループ ID を指定した値に変更するための十分な特権がプロセスに必要です。また、ターゲットプロセスを制御するための十分な特権がプロセスに必要です。

## 使用法

次の `proc` ツールは、ターゲットプロセスの検査時および結果の報告時にターゲットプロセスを停止します。`pfiles`、`pldd`、および `pstack`。停止中のプロセスは何も実行できません。したがって、たとえば、X サーバーが、X サーバーの制御下のウィンドウで実行されているこれらのいずれかの `proc` ツールによって検査される場合、`proc` ツールはリフレッシュできないウィンドウに結果を表示しようとするため、ウィンドウシステム全体がデッドロックになる可能性があります。この場合、[ssh\(1\)](#) を使用して別のシステムからログインし、問題のある `proc` ツールを終了すると、デッドロックが解決されます。

「警告」を参照してください。

`-F` フラグを使用するときは注意してください。2つの制御プロセスを1つの犠牲プロセスに課すと混乱することがあります。プライマリ制御プロセス (通常はデバッグ) が犠牲プロセスを停止しており、プライマリ制御プロセスが当該の `proc` ツールの適用時に何も実行していない場合のみ、安全性が確保されます。

一部の `proc` ツールは、前述の形式で示されているように、コアファイルにも適用できます。コアファイルは、プロセスの状態のスナップショットで、シグナルでプロセスを終了する前にカーネルによって、または [gcore\(1\)](#) ユーティリティによって生成されます。一部の `proc` ツールでは、コアをダンプしたプロセスに対応する実行可能ファイルの名前またはプロセスに関連付けられている共有ライブラリの名前を得ることが必要になる場合があります。これらのファイルは、たとえば、[pstack\(1\)](#) のシンボルテーブル情報を提供するために必要です。その `proc` ツールで必要な実行可能ファイルまたは共有ライブラリを検出できない場合、一部のシンボル情報を表示できないことがあります。同様に、あるオペレーティングシステムリリースのコアファイルが別のオペレーティングシステムリリース上で検査される場合は、ランタイムリンクエディタデバッグインタフェース (`librtld_db`) を初期化できません。この場合、共有ライブラリのシンボル情報は使用できません。

終了ステータス 次の終了ステータスが返されます。

0            正常終了。  
0 以外      エラーが発生しました。

ファイル    `/proc/*`    プロセスファイル

属性        属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

人間が読める形式の出力は「不確実」です。オプションは「確実」です。

関連項目

[gcore\(1\)](#), [ldd\(1\)](#), [pargs\(1\)](#), [pgrep\(1\)](#), [pkill\(1\)](#), [plimit\(1\)](#), [pmap\(1\)](#), [preap\(1\)](#), [ps\(1\)](#), [ptree\(1\)](#), [ppgsz\(1\)](#), [pwd\(1\)](#), [rlogin\(1\)](#), [ssh\(1\)](#), [time\(1\)](#), [truss\(1\)](#), [wait\(1\)](#), [fcntl\(2\)](#), [fstat\(2\)](#), [setuid\(2\)](#), [dlopen\(3C\)](#), [signal.h\(3HEAD\)](#), [core\(4\)](#), [proc\(4\)](#), [process\(4\)](#), [attributes\(5\)](#), [zones\(5\)](#)

警告

次の `proc` ツールは、ターゲットプロセスの検査中および結果の報告中にターゲットプロセスを停止します。 `pfiles`、`pldd`、および `pstack`。ただし、`pstack` は、個々のスレッドに対して実行されていても、プロセス全体が停止します。

停止中のプロセスまたはスレッドは何も実行できません。本稼働環境で頻繁に使用されるプロセスまたはスレッドを短時間でも停止すると、これらのプロセスまたはスレッドが深刻なボトルネックになったりハングアップしたりする可能性があります。ユーザーがプロセスまたはスレッドを使用できなくなることがあります。一部のデータベースも異常終了する可能性があります。したがって、たとえば、前述の `proc` ツールを使用していずれかのデータベースプロセスまたはスレッドが追跡されるときに、大きな負荷のかかるデータベースサーバーはハングアップする可能性があります。このため、本稼働環境で UNIX プロセスまたはスレッドを停止することは避けてください。

これらのツールによって停止されているプロセスまたはスレッドは、`/usr/bin/ps -efll` を実行し、最初の列で「T」を検索すると、識別できます。ただし、「`sched`」などの特定のプロセスは、ほとんどの場合デフォルトで「T」ステータスを表示する可能性があります。

ネットワークファイルシステムでロックされたファイルに返されたプロセス ID には意味がない場合があります。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | prof – display profile data                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| 形式    | prof [-ChsVz] [-a   c   n   t] [-o   x] [-g   l] [-m <i>mdata</i> ]<br>[ <i>prog</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| 機能説明  | <p>The <code>prof</code> command interprets a profile file produced by the <code>monitor</code> function. The symbol table in the object file <i>prog</i> (<i>a.out</i> by default) is read and correlated with a profile file (<i>mon.out</i> by default). For each external text symbol the percentage of time spent executing between the address of that symbol and the address of the next is printed, together with the number of times that function was called and the average number of milliseconds per call.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| オプション | <p>The mutually exclusive options <code>-a</code>, <code>-c</code>, <code>-n</code>, and <code>-t</code> determine the type of sorting of the output lines:</p> <ul style="list-style-type: none"> <li>-a    Sort by increasing symbol address.</li> <li>-c    Sort by decreasing number of calls.</li> <li>-n    Sort lexically by symbol name.</li> <li>-t    Sort by decreasing percentage of total time (default).</li> </ul> <p>The mutually exclusive options <code>-o</code> and <code>-x</code> specify the printing of the address of each symbol monitored:</p> <ul style="list-style-type: none"> <li>-o    Print each symbol address (in octal) along with the symbol name.</li> <li>-x    Print each symbol address (in hexadecimal) along with the symbol name.</li> </ul> <p>The mutually exclusive options <code>-g</code> and <code>-l</code> control the type of symbols to be reported. The <code>-l</code> option must be used with care; it applies the time spent in a static function to the preceding (in memory) global function, instead of giving the static function a separate entry in the report. If all static functions are properly located, this feature can be very useful. If not, the resulting report may be misleading.</p> <p>Assume that A and B are global functions and only A calls static function S. If S is located immediately after A in the source code (that is, if S is properly located), then, with the <code>-l</code> option, the amount of time spent in A can easily be determined, including the time spent in S. If, however, both A and B call S, then, if the <code>-l</code> option is used, the report will be misleading; the time spent during B's call to S will be attributed to A, making it appear as if more time had been spent in A than really had. In this case, function S cannot be properly located.</p> <ul style="list-style-type: none"> <li>-g    List the time spent in static (non-global) functions separately. The <code>-g</code> option function is the opposite of the <code>-l</code> function.</li> <li>-l    Suppress printing statically declared functions. If this option is given, time spent executing in a static function is allocated to the closest global function loaded before the static function in the executable. This option is the default. It is the opposite of the <code>-g</code> function and should be used with care.</li> </ul> |

The following options may be used in any combination:

- C           Demangle C++ symbol names before printing them out.
- h           Suppress the heading normally printed on the report. This is useful if the report is to be processed further.
- m *mdata*   Use file *mdata* instead of *mon.out* as the input profile file.
- s           Print a summary of several of the monitoring parameters and statistics on the standard error output.
- V           Print *prof* version information on the standard error output.
- z           Include all symbols in the profile range, even if associated with zero number of calls and zero time.

A single function may be split into subfunctions for profiling by means of the MARK macro. See [prof\(5\)](#).

- 環境変数
- PROFDIR    The name of the file created by a profiled program is controlled by the environment variable PROFDIR. If PROFDIR is not set, *mon.out* is produced in the directory current when the program terminates. If PROFDIR=*string*, *string/pid.progname* is produced, where *progname* consists of *argv[0]* with any path prefix removed, and *pid* is the process ID of the program. If PROFDIR is set, but null, no profiling output is produced.
- ファイル
- mon.out    default profile file
- a.out       default namelist (object) file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE                    |
|----------------|------------------------------------|
| Availability   | developer/base-developer-utilities |

関連項目

[gprof\(1\)](#), [exit\(2\)](#), [pcsample\(2\)](#), [profil\(2\)](#), [malloc\(3C\)](#), [malloc\(3MALLOC\)](#), [monitor\(3C\)](#), [attributes\(5\)](#), [prof\(5\)](#)

注意事項

If the executable image has been stripped and does not have the *.symtab* symbol table, *gprof* reads the global dynamic symbol tables *.dynsym* and *.SUNW\_ldynsym*, if present. The symbols in the dynamic symbol tables are a subset of the symbols that are found in *.symtab*. The *.dynsym* symbol table contains the global symbols used by the runtime linker. *.SUNW\_ldynsym* augments the information in *.dynsym* with local function symbols. In the case where *.dynsym* is found and *.SUNW\_ldynsym* is not, only the information for the global symbols is available. Without local symbols, the behavior is as described for the *-a* option.

The times reported in successive identical runs may show variances because of varying cache-hit ratios that result from sharing the cache with other processes. Even if a program seems to be the only one using the machine, hidden background or asynchronous processes may blur the data. In rare cases, the clock ticks initiating recording of the program counter may *beat* with loops in a program, grossly distorting measurements. Call counts are always recorded precisely, however.

Only programs that call `exit` or return from `main` are guaranteed to produce a profile file, unless a final call to `monitor` is explicitly coded.

The times for static functions are attributed to the preceding external text symbol if the `-g` option is not used. However, the call counts for the preceding function are still correct; that is, the static function call counts are not added to the call counts of the external function.

If more than one of the options `-t`, `-c`, `-a`, and `-n` is specified, the last option specified is used and the user is warned.

`LD_LIBRARY_PATH` must not contain `/usr/lib` as a component when compiling a program for profiling. If `LD_LIBRARY_PATH` contains `/usr/lib`, the program will not be linked correctly with the profiling versions of the system libraries in `/usr/lib/libp`. See [gprof\(1\)](#).

Functions such as `mcount()`, `_mcount()`, `moncontrol()`, `_moncontrol()`, `monitor()`, and `_monitor()` may appear in the `prof` report. These functions are part of the profiling implementation and thus account for some amount of the runtime overhead. Since these functions are not present in an unprofiled application, time accumulated and call counts for these functions may be ignored when evaluating the performance of an application.

#### 64-bit profiling

64-bit profiling may be used freely with dynamically linked executables, and profiling information is collected for the shared objects if the objects are compiled for profiling. Care must be applied to interpret the profile output, since it is possible for symbols from different shared objects to have the same name. If duplicate names are seen in the profile output, it is better to use the `-s` (summary) option, which prefixes a module id before each symbol that is duplicated. The symbols can then be mapped to appropriate modules by looking at the modules information in the summary.

If the `-a` option is used with a dynamically linked executable, the sorting occurs on a per-shared-object basis. Since there is a high likelihood of symbols from differed shared objects to have the same value, this results in an output that is more understandable. A blank line separates the symbols from different shared objects, if the `-s` option is given.

#### 32-bit profiling

32-bit profiling may be used with dynamically linked executables, but care must be applied. In 32-bit profiling, shared objects cannot be profiled with `prof`. Thus, when a profiled, dynamically linked program is executed, only the *main* portion of the image is sampled. This means that all time spent outside of the *main* object, that is, time spent in a shared object, will not be included in the profile summary; the total time reported for the program may be less than the total time used by the program.

Because the time spent in a shared object cannot be accounted for, the use of shared objects should be minimized whenever a program is profiled with `prof`. If desired, the program should be linked to the profiled version of a library (or to the standard archive version if no profiling version is available), instead of the shared object to get profile information on the functions of a library. Versions of profiled libraries may be supplied with the system in the `/usr/lib/libp` directory. Refer to compiler driver documentation on profiling.

Consider an extreme case. A profiled program dynamically linked with the shared C library spends 100 units of time in some `libc` routine, say, `malloc()`. Suppose `malloc()` is called only from routine B and B consumes only 1 unit of time. Suppose further that routine A consumes 10 units of time, more than any other routine in the *main* (profiled) portion of the image. In this case, `prof` will conclude that most of the time is being spent in A and almost no time is being spent in B. From this it will be almost impossible to tell that the greatest improvement can be made by looking at routine B and not routine A. The value of the profiler in this case is severely degraded; the solution is to use archives as much as possible for profiling.



名前 profiles – 権利プロファイルの一覧表示および管理

形式 profiles [-l] [-a | *user ...*] [-S *repository*]  
 profiles -p *profiles* [-S *repository*]  
 profiles -p *profiles* [-S *repository*] *subcommand*  
 profiles -p *profiles* [-S *repository*] -f *command\_file*  
 profiles help

機能説明 profiles ユーティリティーは、ローカルファイルのネームサービスまたはLDAPネームサービスの [prof\\_attr\(4\)](#) または [exec\\_attr\(4\)](#) データベースの権利プロファイルの構成を作成および変更します。権利プロファイルの構成は、プロファイル名といくつかのプロパティーで作成されています。

次の形式の profiles サブコマンドは、対話形式で使用されます。

profiles -p *profile* [-S *repository*] [*subcommand*]

profiles コマンドは、自分または任意指定のユーザーあるいは役割名に割り当てられている権利プロファイルの名前を標準出力に書き込みます。プロファイルは、特定の機能の実行に必要なコマンドおよび承認を列挙するために使用されるバンドリングメカニズムです。リストには各実行可能ファイルとともに、実効ユーザー ID や実効グループ ID などのプロセス属性が表示されます。特権コマンドインタプリタによって起動されるときにこれらの属性を指定してプロセスが実行されます。[pfexec\(1\)](#) のマニュアルページを参照してください。プロファイルには、[prof\\_attr\(4\)](#) で定義されているほかのプロファイルを含めることができます。

複数のプロファイルを結合して適切なアクセス制御を作成することができます。プロファイルが割り当てられると、既存のセットに承認が追加されます。複数のプロファイルに同じコマンドが存在する場合、プロファイルの順序付けによって決定された、最初のプロファイルがプロセス属性の設定に使用されます。すべてのコマンドに一致するようにワイルドカードを指定できます。

特殊なプロファイル「Stop」を使用すると、以降のプロファイルの評価がショートカットされます。「Stop」プロファイル以降にあるプロファイルは、評価されることも、ほかのコマンドを見つけるために使用されることもありません。そのファイルに AUTH\_GRANTED として一覧表示されている承認および PROF\_GRANTED キーを持つプロファイルを使用すると、`/etc/security/policy.conf` に一覧表示されているプロファイルの一部を対象外にするためにこのプロファイルを使用できます。

プロファイルが解釈されると、プロファイルリストが [user\\_attr\(4\)](#) から読み込まれます。デフォルトプロファイルが `/etc/security/policy.conf` ([policy.conf\(4\)](#) を参照) で定義されている場合は、[user\\_attr\(4\)](#) から読み込まれたリストにデフォルトプロファイルのリストが追加されます。[prof\\_attr\(4\)](#) で一致するエントリは承認リストを、[exec\\_attr\(4\)](#) で一致するエントリはコマンドリストを提供します。

プロパティー -p オプションを指定して呼び出すと、指定されたプロファイルのプロパティーおよび関連付けられている実行可能ファイルのプロパティーを管理できます。ただし、システムの整合性を維持するため、Solaris によって管理されるプロファイルは、このコマンドを使用しても変更できません。そのようなプロファイルは、システムの更新中に pkg(1) コマンドを使用した場合のみ変更できます。

また、その他のプロファイルも、pkg(1) コマンドを使用すると、変更できないプロファイルとして必要に応じて配布できます。

特権のエスカレーションを防ぐため、プロパティー値はユーザーの承認に基づいて制限されます。管理者には、少なくとも Rights Management プロファイルが付与されている必要があります。また、委任承認によって管理されているセキュリティ関連のプロパティーを変更するには、管理者に Rights Delegation プロファイルが付与されている必要があります。詳細は、exec\_attr(4)、prof\_attr(4)、および次のサマリーを参照してください。

プロパティー値には、単純な文字列、またはコンマで区切られた単純な文字列のリストを指定できます。スペース文字を含む単純な文字列は、二重引用符で囲む必要があります。

profiles コマンドは、profile コンテキストと command コンテキストの両方で動作します。profile コンテキストは、初期状態であり、各種のプロファイルプロパティーを管理できます。次の表は、profile コンテキストのプロパティーをまとめたものです。

| Property Name | Value Type         | Required Authorizations             |
|---------------|--------------------|-------------------------------------|
| name          | simple             | none                                |
| auths         | list of simple     | solaris.auth.{assign/delegate}      |
| profiles      | list of simple     | solaris.profile.{assign/delegate}   |
| privs         | list of simple     | solaris.privilege.{assign/delegate} |
| limitpriv     | list of simple     | solaris.privilege.{assign/delegate} |
| defaultpriv   | list of simple     | solaris.privilege.{assign/delegate} |
| always_audit  | list of simple     | solaris.audit.assign                |
| never_audit   | list of simple     | solaris.audit.assign                |
| desc          | simple             | none                                |
| help          | simple             | none                                |
| pam_policy    | simple             | solaris.account.setpolicy           |
| cmd           | simple/new context | none                                |

コマンドのコンテキストは、cmd プロパティーを指定することによって入力します。command コンテキストでは、現在のコマンドのプロパティーを管理できます。

次の表は、command コンテキストのプロパティーをまとめたものです。

| Property Name | Value Type | Required Authorizations |
|---------------|------------|-------------------------|
| id            | simple     | none                    |

|            |                |                                     |
|------------|----------------|-------------------------------------|
| privs      | list of simple | solaris.privilege.{assign/delegate} |
| limitprivs | list of simple | solaris.privilege.{assign/delegate} |
| euid       | simple         | solaris.profile.cmd.setuid          |
| uid        | simple         | solaris.profile.cmd.setuid          |
| egid       | simple         | solaris.group.{assign/delegate}     |
| gid        | simple         | solaris.group.{assign/deleg         |

profile コンテキストのプロパティに指定可能な値を、次のリストに示します。次のリストで指定されているように、プロパティとその値の間には等号(=)が必要です。

#### always\_audit

常に監査するイベントクラスを指定する監査フラグ。ユーザーの [user\\_attr\(4\)](#) エントリ、または割り当てられているプロファイルの順序付けされたリストにある、このプロパティの最初に出現しているもののみが、ログインおよび su で適用されます。

#### auths

新しいプロファイルに追加される、コンマで区切られた 1 つ以上の承認。ワイルドカード文字 (\*) を承認名に使用する場合は、名前を二重引用符 (") で囲む必要があります。

#### cmd

実行可能ファイルへの完全指定パス、またはすべてのコマンドを指定するために使用されるアスタリスク (\*) 記号。パス名のファイル名コンポーネントを置き換えるアスタリスクは、特定のディレクトリのすべてのファイルを示します。

これは、`command` コンテキストを入力して、コマンドのセキュリティプロパティを管理するために使用される特殊なプロパティです。

これらの ID には、数値 ID または名前を使用できます。

#### id

このプロパティには、前の `cmd` プロパティによって指定された値が設定されますが、この値は変更できます。`select` サブコマンドと組み合わせて使用すると、その後の編集のために既存のコマンドのプロパティのクローンを作成できます。

#### pam\_policy

ユーザーに適用する PAM ポリシー。`pam_policy` は、[pam.conf\(4\)](#) 形式のファイルの絶対パス名、または `/etc/security/pam_policy` 内の [pam.conf\(4\)](#) 形式のファイルの名前にする必要があります。詳細は、[pam\\_user\\_policy\(5\)](#) を参照してください。

#### privs

実行可能プロセスの継承可能セットに適用する一連の特権。デフォルトは `basic` です。

**limitprivs**

実行可能プロセスの制限セットに適用する一連の特権。デフォルトはallです。

**euid**

コマンドで実行するプロセスの実効ユーザー ID。

**uid**

コマンドで実行するプロセスの実ユーザー ID。

**egid**

コマンドで実行するプロセスの実効グループ ID。

**gid**

コマンドで実行するプロセスの実グループ ID。

**defaultpriv**

ユーザーの一連のプロセスに割り当てられるデフォルトの一連の特権。ユーザーの **user\_attr(4)** エントリ、または割り当てられているプロファイルの順序付けされたリストにある、このプロパティの最初に出現しているもののみが、ログインおよび su で適用されます。

**desc**

新しいプロファイルの説明。テキストは引用符で囲む必要があります。

**help**

新しいプロファイルのヘルプファイル名。ヘルプファイルは、`/usr/lib/help/profiles/locale/<locale>` ディレクトリにコピーされます。ここで、`<locale>` は、ユーザーの言語ロケールの値です。何も指定されていない場合は `c` になります。このプロパティの指定は、`files` リポジトリでのみ使用できます。

**limitpriv**

ユーザーが **su(1M)** またはその他の方法で開始したユーザーまたはプロセスが取得できる特権セットの最大数。ユーザーの **user\_attr(4)** エントリ、または割り当てられているプロファイルの順序付けされたリストにある、このプロパティの最初に出現しているもののみが、ログインおよび su で適用されます。

**name**

プロファイルの名前。名前の初期値は、コマンド行で `-p` オプションを使用して指定します。名前を変更すると、現在のプロファイルのプロパティが、新しい名前が付けられたプロファイルに適用されます。このようにして、その後の編集のために既存のプロファイルのクローンを作成できます。名前には、既存のプロファイルと同じ名前は使用できません。

**never\_audit**

監査しないイベントクラスを指定する監査フラグ。ユーザーの **user\_attr(4)** エントリ、または割り当てられているプロファイルの順序付けされたリストにある、このプロパティの最初に出現しているもののみが、ログインおよび su で適用されます。

privs

[pfexec\(1\)](#) コマンドの **P** オプションを使用して指定できる一連の特権。

profiles

新しいプロファイルに追加される、コンマで区切られた 1 つ以上の補助プロファイル。

## オプション

サポートしているオプションは、次のとおりです。

**-a**

指定されたりポジトリのすべてのプロファイル名を一覧表示します。リポジトリが指定されていない場合は、[nsswitch.conf\(4\)](#) の `prof_attr` に構成されている内容に従います。

**-f *command\_file***

`profiles` のコマンドファイルの名前を指定します。*command\_file* は、`profiles` サブコマンドが 1 行に 1 つあるテキストファイルです。

**-l**

権利プロファイルに関する情報を表示し、コマンドと、ユーザーやグループ ID などそのコマンドの特殊なプロセス属性を一覧表示します。

**-p *profile***

プロファイル名を指定します。

**-S *repository***

有効なリポジトリは、`files` と `ldap` です。*repository* は、更新するネームサービスを指定します。デフォルトの *repository* は `files` です。

## サブコマンド

**-p** オプションを指定して呼び出すと、サブコマンドをコマンド行または対話形式で指定できます。セミコロンで区切られた複数のサブコマンドをコマンド行に指定するには、一連のサブコマンド全体を引用符で囲みます。サブコマンドが指定されていない場合は、対話型セッションが行われることを意味します。対話型セッションでは、TAB キーを使用することにより、サブコマンドの自動補完を呼び出すことができます。

**add** サブコマンドと **select** サブコマンドを使用すると、特定のコマンドを選択できます。選択した時点で、コンテキストがそのコマンドのコンテキストに変更されます。対話型セッションでは、`command` コンテキストは、プロンプト文字列のコマンドのベース名によって識別されます。**end** サブコマンドと **cancel** サブコマンドを使用すると、コマンドの指定を終了できます。指定を終了した時点で、コンテキストは `profile` コンテキストに戻ります。

破壊的な動作や作業内容の消失を伴う可能性のあるサブコマンドには、強制的にその処理を実行するために **-F** オプションが用意されています。端末デバイスから入力を行う場合、ユーザーは適切なタイミングで入力を要求されます。これは、サブコマンドが **-F** オプションなしで指定された場合に発生することがあります。それ以外の場合、処理は許可されず、診断メッセージが標準エラー出力に書き込まれます。

プロパティー値には、単純値、またはリストを受け入れるプロパティーの場合は、単純値のリストを指定できます。サポートされているサブコマンドは次のとおりです。

**add cmd=pathname**

profile コンテキストで、対象となるコマンドの指定を開始します。コンテキストは、command タイプに変更されます。

**add property-name=property-value**

指定した値を現在のプロパティー値に追加します。このサブコマンドを適用できるのは、リストを受け入れるプロパティーのみです。

**cancel**

コマンドの指定を終了し、コンテキストを profile にリセットします。指定途中のリソースをすべて破棄します。cancel を適用できるのは、command コンテキストのみです。

**clear property name**

プロパティーの値を消去します。

**commit**

現在の構成をメモリーから安定した記憶領域に確定します。変更を適用するには、構成を確定する必要があります。メモリー内構成を確定するまでは、revert サブコマンドを使って変更を取り消すことができます。profiles セッションが終了するときには、commit 処理が自動的に試みられます。構成が正しく設定されていないと確定できないので、この処理では自動的に構成が確認されます。

**delete [-F]**

指定されたプロファイルをメモリー内および安定した記憶領域から削除します。この操作は、対象のプロファイルが同じリポジトリの別のプロファイルにサブプロファイルとして含まれている場合は許可されません。代わりに、このプロファイルを含むプロファイルのリストが提供されます。それらのプロファイルからこのプロファイルを手動で削除してから、対象のプロファイルを削除する必要があります。この操作を強制的に実行する場合は、-F オプションを使用します。削除が許可される場合、処理はすぐに実行され、セッションは終了します。

**end**

コマンドの指定を終了します。このサブコマンドを適用できるのは、command コンテキストのみです。profiles コマンドは、現在のコマンドが完全に指定されていることをチェックします。完全に指定されている場合は、メモリー内構成に追加され(メモリー内構成を安定した記憶領域に保存する方法については commit を参照)、コンテキストは profile コンテキストに戻ります。指定が完全でない場合は、対応するエラーメッセージが通知されます。

**exit [-F]**

profiles のセッションを終了します。必要な場合は、commit 操作が自動的に試行されます。EOF 文字を使用して profiles を終了することもできます。この操作を強制的に実行する場合は、-F オプションを使用します。



**export** [-f *output-file*]

標準出力に構成を出力します。output-file に構成を出力する場合は、-f オプションを使用します。このオプションを使用すると、コマンドファイルオプションに適した形式で出力が作成されます。

**help** [*usage*] [*subcommands*] [*properties*] [<*subcommand.*>] [<*properties*>]

一般ヘルプまたは特定項目のヘルプを出力します。

**info** [*property-name*]

現在のプロファイルまたは指定されたプロパティに関する情報を表示します。

**remove** cmd=*fullpath*

指定されたコマンドをプロファイルから削除します。このサブコマンドは、profile コンテキストでのみ有効です。

**remove** [-F] cmd

プロファイルからすべてのコマンドを削除します。-F オプションを使用していない場合は、確認が求められます。このサブコマンドは、profile コンテキストでのみ有効です。

**remove** property-name=*property-value*

指定した値をプロパティから削除します。これを適用できるのは、リストを受け入れるプロパティのみです。

**revert** [-F]

構成を、最後に確定されたときの状態に戻します。この操作を強制的に実行する場合は、-F オプションを使用します。

**select** cmd=*fullpath*

変更を行うために、指定したパス名の条件に一致するコマンドを選択します。このサブコマンドを適用できるのは、profile コンテキストのみです。

**set** property-name=*property-value*

指定されたプロパティ名を、指定された値に設定します。profile コンテキストでのみ有効なプロパティ (たとえば、name や desc) と、command コンテキストでのみ有効なプロパティがあります。このサブコマンドは、profile コンテキストと command コンテキストの両方で適用できます。

**verify**

現在の構成が正しいかどうかを確認します。

- 必須のプロパティが指定されている。
- 各キーワードの値が有効である。
- ユーザーは値を指定することを承認されている。

## 使用例

例1 profiles コマンドを使用する

profiles コマンドの出力は次の形式です。

```
example% profiles tester01 tester02
tester01 : Audit Management, All Commands
```

**例1 profiles コマンドを使用する (続き)**

```
tester02 : Device Management, All Commands
example%
```

**例2 list オプションの使用**

```
example% profiles -l tester01 tester02
tester01 :
 Audit Management:
 /usr/sbin/audit euid=root
 /usr/sbin/auditconfig euid=root egid=sys
 All Commands:
 *
tester02 :
 Device Management:
 /usr/bin/allocate: euid=root
 /usr/bin/deallocate: euid=root
 All Commands
 *
example%
```

**例3 新しいプロファイルの作成**

次のコマンドは、新しい User Manager プロファイルを LDAP に作成します。新しいプロファイルの説明は「Manage users and groups」で、割り当てられる承認は solaris.user.manage です。割り当てられる補助プロファイルは Mail Management です。ヘルプファイル名は、RtUserMgmt.html です。

```
example% profiles -p "User Manager" -S ldap
profiles:User Manager> set desc="Manage users and groups"
profiles:User Manager> set help=RtUserMgmt.html
profiles:User Manager> set auths=solaris.user.manage
profiles:User Manager> set profiles="Mail Management"
profiles:User Manager> exit
```

**例4 現在の構成に関する情報の表示**

次のコマンドは、User Manager プロファイルに関する情報を表示します。

```
example% profiles -p "User Manager" -S ldap info
name=User Manager
desc=Manage users and groups
auths=solaris.user.manage
profiles=Mail Management
help=RtUserMgmt.html
```



**例5 プロファイルの削除**

次のコマンドは、LDAP から User Manager プロファイルを削除します。

```
example% profiles -p "User Manager" -S ldap delete -F
```

**例6 プロファイルの変更**

次のコマンドは、LDAP の User Manager プロファイルを変更します。新しいプロファイルの説明は「Manage world」、新しい承認の割り当ては solaris.user.\* 承認、および新しい補助プロファイルの割り当ては All です。

```
example% profiles -p "User Manager" -S ldap
profiles:User Manager> set desc="Manage world"
profiles:User Manager> set auths="solaris.user.*"
profiles:User Manager> set profiles=All
profiles:User Manager> exit
```

**例7 exec\_attr データベースエントリの作成**

次のコマンドは、LDAP の User Manager プロファイルに新しい exec\_attr のエントリを作成します。/usr/bin/cp というエントリが追加されます。コマンドには、実効ユーザー ID に 0、および実効グループ ID に 0 が指定されています。

```
example% profiles -p "User Manager" -S ldap
profiles:User Manager> add cmd=/usr/bin/cp
profiles:User Manager:cp> set euid=0
profiles:User Manager:cp> set egid=0
profiles:User Manager:cp> end
profiles:User Manager> exit
example%
```

**例8 exec\_attr データベースエントリの削除**

次の例では、LDAP から User Manager プロファイルの exec\_attr データベースエントリを削除します。コマンド /usr/bin/cp に指定されているエントリが削除されます。

```
example% profiles -p "User Manager" -S ldap
profiles:User Manager> remove cmd=/usr/bin/cp
profiles:User Manager> exit
example%
```

**例9 exec\_attr データベースエントリの変更**

次を使用して、LDAP の User Manager プロファイルの exec\_attr データベースエントリの属性を変更します。/usr/bin/cp エントリは、実ユーザー ID が 0、および実グループ ID が 0 で実行するように変更されます。

```
example% profiles -p "User Manager" -S ldap
profiles:User Manager> select cmd=/usr/bin/cp
profiles:User Manager:cp> clear euid
```

例9 exec\_attr データベースエントリの変更 (続き)

```
profiles:User Manager:cp> clear egid
profiles:User Manager:cp> set uid=0
profiles:User Manager:cp> set gid=0
profiles:User Manager:cp> end
profiles:User Manager> exit
example%
```

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

1 エラーが発生した。

ファイル /etc/security/exec\_attr  
/etc/security/prof\_attr  
/etc/user\_attr  
/etc/security/policy.conf

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |

関連項目 [auths\(1\)](#), [pfexec\(1\)](#), [pkg\(1\)](#), [roles\(1\)](#), [getprofattr\(3C\)](#), [auth\\_attr\(4\)](#), [exec\\_attr\(4\)](#), [nsswitch.conf\(4\)](#), [pam.conf\(4\)](#), [policy.conf\(4\)](#), [prof\\_attr\(4\)](#), [user\\_attr\(4\)](#), [audit\\_flags\(5\)](#), [attributes\(5\)](#), [pam\\_user\\_policy\(5\)](#), [privileges\(5\)](#)

|         |                                                                                                                                                                                             |
|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | projects – ユーザーのプロジェクトメンバーシップの出力                                                                                                                                                            |
| 形式      | projects [-dv] [ user]<br>projects -l [projectname [projectname]...]                                                                                                                        |
| 機能説明    | projects コマンドは、コマンドを呼び出したユーザーまたは任意で指定されたユーザーが所属するプロジェクトを標準出力に表示します。各ユーザーは、 <a href="#">project(4)</a> ファイルに指定された特定のプロジェクトセットに所属しています。それらのプロジェクトセットは、関連のある NIS マップおよび LDAP に指定されている可能性もあります。 |
| オプション   | サポートしているオプションは、次のとおりです。<br>-d デフォルトのプロジェクトだけを出力します。<br>-l 各プロジェクトのプロジェクト名に関する詳細な情報を出力します。プロジェクト名が指定されていない場合、すべてのプロジェクトに関する情報が出力されます。<br>-v プロジェクト名と一緒にプロジェクトの説明を出力します。                      |
| オペランド   | 次のオペランドを指定できます。<br>projectname 指定されたプロジェクトの情報を表示します。<br>user 指定されたユーザーのプロジェクトメンバーシップを表示します。                                                                                                 |
| 使用例     | 例1 指定されたユーザーのメンバーシップの表示<br>example\$ <b>projects paul</b><br>default beatles wings<br>example\$ <b>projects ringo</b><br>default beatles<br>example\$ <b>projects -d paul</b><br>beatles    |
| 終了ステータス | 次の終了ステータスが返されます。<br>0 正常終了。<br>1 実行中に致命的なエラーが発生した。<br>2 無効なコマンド行オプションが指定された。                                                                                                                |
| ファイル    | /etc/project このマシンの有効なプロジェクト定義を含むローカルデータベース。                                                                                                                                                |
| 属性      | 属性についての詳細は、 <a href="#">attributes(5)</a> を参照してください。                                                                                                                                        |

| 属性タイプ | 属性値            |
|-------|----------------|
| 使用条件  | system/core-os |
| 安定性   | 下記を参照。         |

呼び出しは「確実」です。人間が読める形式の出力は「不確実」です。

関連項目 [getdefaultproj\(3PROJECT\)](#), [getprojent\(3PROJECT\)](#), [project\(4\)](#), [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | ps - プロセスのステータス報告                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 形式    | <pre>ps [-aAcdefjHLLPyZ] [-g grplist] [-h lgrplist]     [-n namelist] [-o format]... [-p proclist]     [-s sidlist] [-t term] [-u uidlist] [-U uidlist]     [-G gidlist] [-z zonelist]</pre>                                                                                                                                                                                                                                                                                                                                                                         |
| 機能説明  | <p>ps コマンドは、活動中のプロセスに関する情報を出力します。オプションを指定しないと、ps は、「コマンドを実行したユーザー」と同じ実効ユーザー ID および同じ制御端末に関連するプロセスについての情報を出力します。出力される情報は、プロセス ID、端末識別子、累積実行時間およびコマンド名のみです。それ以外の場合、出力される情報はオプションによって制御されます。</p> <p>オプションのなかには、リストを引数として使うものがあります。リストの項目は、コンマで区切られるか、あるいは引用符で囲まれてコンマかスペースで区切られます。proclist および grplist には、数値を指定してください。</p> <p>ps コマンドは、ネイティブにコマンドを呼び出すか、ps(1B) で必要なコマンド構文を使用して呼び出すかを決定しようとします。後者の場合、ps コマンドは ps(1B) で説明されているのとまったく同じように動作します。</p>                                                                                                                        |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-a<br/>もっとも頻繁に要求されるすべてのプロセスに関する情報を表示します。ただし、プロセスのセッションリーダーおよび端末に関連していないプロセスは除きます。</li> <li>-A<br/>すべてのプロセスの情報を表示します。後述の -e と同じです。</li> <li>-c<br/>priocntl(1) で説明するように、スケジューラ特性を反映する形式で情報を出力します。以下で説明するように、-c オプションは、-f オプションと -l オプションの出力に影響します。</li> <li>-d<br/>セッションリーダーを除くすべてのプロセスに関する情報を表示します。</li> <li>-e<br/>現在実行中のすべてのプロセスに関する情報を表示します。</li> <li>-e オプションを指定すると、-z、-t、-u、-U、-g、-G、-p、-g、-s、および -a オプションは効果がありません。</li> <li>-f<br/>完全リストを作成します (完全リストにおけるカラムの意味については、以下を参照)。</li> </ul> |

**-g *grplist***

*grplist* にグループリーダーの ID 番号が指定されているプロセスの情報を表示します (グループリーダーとは、プロセス ID 番号がプロセスグループ ID 番号と同じプロセスのこと)。

**-G *gidlist***

*gidlist* に実グループの ID 番号が指定されているプロセスの情報を出力します。*gidlist* は、複数の ID を空白またはコンマで区切って記述するリストで、単一の引数として指定する必要があります。

**-h *lgrplist***

指定された *lgrplist* が存在するプロセスだけを表示します。*lgrplist* で指定されている無効なグループについては、何も表示されません。

**-H**

追加の列ヘッダー (LGRP) にプロセスのホーム *lgroup* を出力します。

**-j**

セッション ID およびプロセスグループ ID を出力します。

**-l**

長い形式のリストを作成します (下記を参照)。

**-L**

選択された各プロセス中の各軽量プロセス (*lwp*) についての情報を出力します (下記を参照)。

**-n *namelist***

*namelist* で示すシステムファイルをデフォルトの代わりに使用します。このオプションは、互換のために受け入れられますが無視されます。

**-o *format***

*format* が示す書式指定に従って出力情報を表示します。詳細は「表示書式」の節を参照してください。-o オプションは複数回指定することが可能です。その場合、複数の書式指定引数は、スペース文字で区切られて継続していると見なされます。

**-p *proclist***

プロセス ID 番号が *proclist* で指定されたプロセスの情報のみを表示します。

**-P**

追加のカラムヘッダー *PSR* (もしある場合) をつけて、プロセスまたは *lwp* が接続されたプロセッサ数を出力します。

**-s *sidlist***

ID が *sidlist* にあるすべてのセッションリーダーの情報を表示します。

**-t *term***

*term* に関連するプロセスの情報のみを表示します。端末識別子は、デバイスのファイル名および識別子として指定されます。たとえば、*term/a* や *pts/0* などです。

**-u uidlist**

実効ユーザー ID またはログイン名が *uidlist* で指定されたプロセスデータのみを表示します。リスト出力時に、ログイン名を表示する **-f** オプションをユーザーが指定しないかぎり、ユーザー ID は数字で出力されます。

**-U uidlist**

実グループ ID またはログイン名が *uidlist* に存在するプロセスの情報を表示します。*uidlist* は、複数の ID を空白またはコンマで区切って記述するリストで、単一の引数として指定する必要があります。

**-y**

長い形式のリストで (**-l**)、旧式の **F** と **ADDR** カラムが省略され、プロセスの常駐の設定サイズを示す **RSS** カラムが含まれます。**-y** オプションを指定すると、**RSS** と **SZ** (後述参照) の両方が、ページの代わりに **K** バイト単位で報告されます。

**-z zonelist**

指定されたゾーン内のプロセスだけを表示します。ゾーンは名前と ID のどちらでも指定できます。このオプションは、グローバルゾーンで実行する場合にのみ便利です。

**-Z**

プロセスに関連するゾーン名を、追加のカラムヘッダー **ZONE** の下に表示します。**ZONE** カラムの幅は 8 文字に制限されます。現在実行中のすべてのプロセスと関連ゾーン名に関する情報をすばやく確認するには、**ps -ez** を使用します。8 文字幅を超えるゾーン名を表示するには、

```
ps -eo zone,uid,pid,ppid,time,comm,...
```

を使用します。

`/usr/ucb/ps` コマンドでは、次のオプションが使用されます (**ps(1B)** を参照)。これらは `/usr/bin/ps` でサポートされており、後者が UCB の動作をエミュレートすることを可能にします。UCB オプションでは、ハイフンは使用しません。これらのオプションを、前述のオプションと混在させることはできません。

**r**

出力を実行中および実行可能なプロセスに制限します。

**S**

このプロセスおよび取得されるすべての子プロセスで使用される CPU 時間の累計を表示します。

**V**

仮想メモリーを含む出力のバージョンを表示します。これには、次で説明する **SIZE**、**%CPU**、**%MEM**、および **RSS** フィールドが含まれます。

**w**

80 カラムではなく 132 カラムのワイド出力形式を使用します。**-ww** とオプション文字が繰り返される場合、任意幅の出力が使用されます。この情報を使用し

て、長いコマンドをどれだけ出力するかが決定されます。注: ワイド出力オプションを表示できるのは、スーパーユーザーまたはこのプロセスを所有するユーザーだけです。

x

制御端末のないプロセスが含まれます。

*num*

プロセス番号を指定できます (この場合、出力はそのプロセスに限定されます)。このオプションは、最後に指定する必要があります。

前述の多くのオプションは、対象とするプロセスをリストに選択するために使用します。これらのオプションが指定された場合、**ps** はデフォルトリストを無視し、オプションの選択条件の論理和により示されるプロセスを選択します。

## 表示書式

-f オプションが指定された場合、**ps** は、ユーザーブロックを検索することによって、プロセスが作成されたときに指定されたコンド名および引数を特定しようとします。これができない場合には、-f オプションを指定しない場合のコマンド名が角括弧に囲まれて出力されます。

**ps** の実行結果として表示されるリストのカラムの見出しおよびカラムの意味を説明します。文字 **f** および **i** は、それに相当する見出しを作成するオプション (それぞれ完全形式または長い形式) を示します。**all** は、見出しを常に出力することを意味します。注: これら 2 つのオプションは、プロセスのどの情報が表示されるかを決定するだけで、どのプロセスが表示されるかということは決定しません。

**F (l)**

プロセスに関連するフラグ (16 進数表現の論理和)。これらのフラグは過去においては意味を持っていたが、現在では何の意味も持たない

**S (l)**

プロセスの状態

**O**

プロセスは、プロセッサ上で実行中

**S**

休止状態。プロセスは、イベントが完了するのを待っている

**R**

実行可能状態。プロセスは、実行待ち行列上にある

**T**

ジョブ制御シグナルにより、もしくはトレース状態にあるため、プロセスは停止されている

**W**

待機状態。プロセスは、CPU 使用率が CPU キャップの強制された制限値まで下がるのを待っている。



Z

ゾンビ状態。プロセスは終了していて、親プロセスは待っていない

UID (f,l)

プロセスの実効ユーザー ID (-f オプションではログイン名が出力)

PID (all)

プロセスのプロセス ID (このデータはプロセスを停止させるために必要)

PPID (f,l)

親プロセスのプロセス ID

C (f,l)

スケジューリングのためのプロセッサ利用率 (旧仕様) -c オプションを使用した場合は、出力されない

CLS (f,l)

スケジューリングクラス。-c オプションを使用した場合にのみ、出力される

PRI (l)

プロセスの優先順位。-c オプションを指定しないと、数が高いほど優先順位が低くなる。-c オプションを指定すると、数が高いほど優先順位が高いことを意味する

NI (l)

優先順位の計算に使用される nice 値。-c オプションを使用した場合は、出力されない特定のスケジューリングクラスのプロセスのみが、nice 値を持つ

ADDR (l)

プロセスのメモリーアドレス。すべての特権で実行中以外は 0 です。

SZ (l)

割り当てられたすべてのファイルとデバイスを含む、仮想メモリー内のプロセスの合計サイズ (ページ単位)。[pagesize\(1\)](#) を参照

WCHAN (l)

プロセスが休止状態のイベントのアドレス。すべての特権で実行される場合にのみ表示されます。それ以外の場合は 0 です。プロセスが休止中であるかどうかを確認するには、s 列をチェックします。

STIME (f)

時間、分、秒で示されるプロセスの開始時間 (ps が実行される 24 時間以上前に開始したプロセスは、月および日で示される)

TTY (all)

プロセスを制御している端末 (制御端末がない場合は、メッセージ? が出力される)

TIME (all)

プロセスの累積実行時間。

**LTIME (all)**

報告される lwp の実行時間。

**CMD (all)**

コマンド名 (-f オプションでは、完全なコマンド名と引数を最大 80 文字まで出力)

-j オプションを指定すると、次の 2 つのカラムも出力されます。

**PGID**

プロセスグループリーダーのプロセス ID

**SID**

セッションリーダーのプロセス ID

-L オプションを指定すると、次の 2 つのカラムも出力されます。

**LWP**

情報が出力されている lwp の lwp ID

**NLWP**

プロセス中の lwp 数 (-f も指定した場合)

-L オプションを指定すると、プロセス中の各 lwp が 1 行に 1 つずつ出力され、プロセスではなく lwp に対する時間のフィールド STIME と LTIME を示します。従来のシングルスレッドプロセスは 1 つの lwp だけを含みます。

すでに終了し、親プロセスを持ち、ただし、その親プロセスが待ち状態になっていないプロセスは、<defunct> と符号で示されます。

**-o format**

-o オプションを使用すると、出力書式をユーザーが指定することができます。

出力書式は、空白またはコンマで区切った引数のリストで指定します。各変数にはデフォルトのヘッダーがあります。等号と新しいヘッダーのテキストを追加することによって、デフォルトのヘッダーを上書きできます。引数の残りの文字は、ヘッダーテキストとして使用されます。コマンド行で指定した順番でフィールドが書き込まれ、カラムとして出力されます。フィールド幅は、少なくともヘッダーを表示できる幅(デフォルト値または指定値)がシステムによって選択されます。-o user= のようにヘッダーテキストが NULL である場合、フィールド幅は少なくともデフォルトのヘッダーテキストを表示できる幅になります。すべてのヘッダーテキストが NULL である場合、ヘッダー行が出力されません。

POSIX ロケールでは、次の名前が認識されます。

**user**

プロセスの実効ユーザー ID。テキストのユーザー ID を取得でき、フィールド幅が足りる場合は、テキストのユーザー ID になります。そうでない場合は、10 進数の ID になります。

**ruser**

プロセスの実ユーザー ID。テキストのユーザー ID を取得でき、フィールド幅が足りる場合は、テキストのユーザー ID になります。そうでない場合は、10 進数の ID になります。

**group**

プロセスの実効グループ ID。テキストのグループ ID を取得でき、フィールド幅が足りる場合は、テキストのグループ ID になります。そうでない場合は、10 進数の ID になります。

**rgroup**

プロセスの実グループ ID。テキストのグループ ID を取得でき、フィールド幅が足りる場合は、テキストのグループ ID になります。そうでない場合は、10 進数の ID になります。

**pid**

プロセス ID の 10 進数値。

**ppid**

親プロセス ID の 10 進数値。

**pgid**

プロセスグループ ID の 10 進値。

**pcpu**

最近のある期間において、使用できる CPU 時間に対して実際に使用された CPU 時間の割合 (単位: %)。ここでいう「最近」は、不特定の期間です。使用できる CPU 時間は不特定な方法で決まります。

**vsz**

仮想メモリー中のプロセスの合計サイズ (単位: キロバイト)。

**nice**

プロセスの、システムスケジューリング優先順位を表す 10 進数値。[nice\(1\)](#) を参照してください。

**etime**

POSIX ロケールにおける、プロセスが開始されてからの経過時間。書式は次のとおりです。

`[[dd-]hh:]mm:ss`

各情報の意味は次のとおりです。

*dd*

日

*hh*

時間

*mm*  
分

*ss*  
秒

*dd* フィールドの値は 10 進の整数です。*hh*、*mm*、*ss* フィールドは、ゼロによって左揃えされた 2 桁の 10 進整数です。

*time*

POSIX ロケールにおける、プロセスの累積 CPU 時間。書式は次のとおりです。

[*dd-*]*hh:mm:ss*

*dd*、*hh*、*mm*、*ss* フィールドは、*etime* と同様です。*etime* の説明を参照してください。

*tty*

プロセスの制御端末の名前。書式は **who(1)** コマンドによって使用されるものと同じです。

*comm*

実行されているコマンドの、文字列としての名前 (*argv[0]* の値)。

*args*

文字列としての、コマンドとそのすべての引数。実装によってフィールド幅までに切り捨てられることがあります。これは実装に依存します。この文字列が、コマンドが起動されたときにコマンドに渡された引数であるか、またはアプリケーションによって変更された引数であるかは、不特定です。アプリケーションは、引数リストを変更できて、その変更を *ps* の出力に反映することに依存しません。Solaris 実装では、この文字列は 80 文字以内に限定され、コマンドが起動されたときにコマンドに渡された引数となります。

Solaris 実装では、次の名前が認識されます。

*f*

プロセスに関連するフラグ (16 進数表現の論理和)。

*s*

プロセスの状態。

*c*

スケジューリングのためのプロセッサ利用率 (旧仕様)

*uid*

プロセスの実効ユーザー ID (10 進整数)

*ruid*

プロセスの実ユーザー ID (10 進整数)

**gid**  
プロセスの実効グループ ID (10 進整数)

**rgid**  
プロセスの実グループ ID (10 進整数)

**projid**  
プロセスのプロジェクト ID (10 進整数)

**project**  
プロセスのプロジェクト ID (取得できる場合はテキスト。そうでない場合は 10 進整数)

**zoneid**  
プロセスのゾーン ID 番号 (10 進整数)

**zone**  
プロセスのゾーン ID (取得できる場合はテキスト。そうでない場合は 10 進整数)

**sid**  
セッションリーダーのプロセス ID

**taskid**  
プロセスのタスク ID

**class**  
プロセスのスケジューリングクラス

**pri**  
プロセスの優先順位。(数値が大きいほど優先順位が高い)

**opri**  
プロセスの廃止優先順位 (数値が小さいほど優先順位が高い)

**lwp**  
lwp ID (10 進数)。1 行に 1 つの進行中の軽量プロセスが示されます。

**nlwp**  
軽量プロセスの数

**psr**  
プロセスまたは軽量プロセスがバインドされているプロセッサの数

**pset**  
プロセスまたは軽量プロセスがバインドされているプロセッサセットの ID

**addr**  
プロセスのメモリーアドレス

**osz**  
仮想メモリー中のプロセスの合計サイズ (単位: ページ)

- wchan  
プロセスが休止中のイベントのアドレス (- の場合、プロセスは動作中)
- stime  
プロセスの開始時間または開始日。空白なしで出力されます。
- rss  
プロセスの常駐の設定サイズ (単位: キロバイト)。ps が報告する rss 値は [proc\(4\)](#) が提供する概算です。したがって、実際の常駐の設定サイズを下回る可能性があります。容量を計画するときに、より正確な使用状況を知りたい場合は、[pmap\(1\)](#) -x を使用してください。
- pmem  
マシン上の物理メモリーに対する、プロセスの常駐の設定サイズの割合 (単位: %)
- fname  
プロセスの実行可能ファイルのベース名の先頭 8 バイト
- ctid  
プロセスがメンバーであるプロセス契約の契約 ID (10 進整数)
- lgrp  
プロセスのホーム lgroup。

comm および args だけに、空白文字を含めることができます。Solaris 実装の変数およびその他すべての名前には、空白文字を含めることができません。

POSIX ロケールで使用されるデフォルトヘッダーとそれに対応する書式指定子は、次のとおりです。

| 表記形式  | デフォルト   | 表記形式   | デフォルト  |
|-------|---------|--------|--------|
| 指定子   | ヘッダー    | 指定子    | ヘッダー   |
| args  | COMMAND | ppid   | PPID   |
| comm  | COMMAND | rgroup | RGROUP |
| etime | ELAPSED | ruser  | RUSER  |
| group | GROUP   | time   | 時間     |
| nice  | NI      | tty    | TT     |
| pcpu  | %CPU    | user   | USER   |
| pgid  | PGID    | vsz    | VSZ    |
| pid   | PID     |        |        |

Solaris 実装における書式指定子と対応するデフォルトヘッダーは、次のとおりです。

| 表記形式<br>指定子 | デフォルト<br>ヘッダー | 表記形式<br>指定子 | デフォルト<br>ヘッダー |
|-------------|---------------|-------------|---------------|
| addr        | ADDR          | projid      | PROJID        |
| c           | C             | project     | PROJECT       |
| class       | CLS           | psr         | PSR           |
| f           | F             | rgid        | RGID          |
| fname       | COMMAND       | rss         | RSS           |
| gid         | GID           | ruid        | RUID          |
| lgrp        | LGRP          | s           | S             |
| lwp         | LWP           | sid         | SID           |
| nlwp        | NLWP          | stime       | STIME         |
| opri        | PRI           | taskid      | TASKID        |
| osz         | SZ            | uid         | UID           |
| pmem        | %MEM          | wchan       | WCHAN         |
| pri         | PRI           | zone        | ZONE          |
| ctid        | CTID          | zoneid      | ZONEID        |

## 使用例

例1 ps コマンドの例  
次のコマンド

```
example% ps -o user,pid,ppid=MOM -o args
```

このコマンドによって、POSIX ロケールでは次の内容が出力されます。

```
USER PID MOM COMMAND
helene 34 12 ps -o uid,pid,ppid=MOM -o args
```

切り捨てが発生する可能性があるため、COMMAND の内容は必ずしも前述の例と一致しません。

## 環境変数

ps の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC\_ALL、LC\_CTYPE、LC\_MESSAGES、LC\_TIME、および NLSPATH。

### COLUMNS

画面の水平方向のサイズとして、システムが決定する値の代わりに使用する値を定義します。この値により、表示するテキストカラム数が決まります。

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。

ファイル

/dev/pts/\*

/dev/term/\*

端末 (tty) 名を検索するファイル

/etc/passwd

UID 情報を提供

/proc/\*

プロセス制御ファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| CSI         | 有効(「使用法」参照)                             |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

関連項目

[kill\(1\)](#), [lgrpinfo\(1\)](#), [nice\(1\)](#), [pagesize\(1\)](#), [pmap\(1\)](#), [prioctl\(1\)](#), [who\(1\)](#), [ps\(1B\)](#), [getty\(1M\)](#), [proc\(4\)](#), [ttysrch\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [resource\\_controls\(5\)](#), [standards\(5\)](#), [zones\(5\)](#)

注意事項

ps の実行中に状況が変化することがあります。ps が提供する情報は、ある瞬間だけに当てはまるものであり、ユーザーがそれを参照するときには正確ではないことがあります。すでに存在しないプロセスについてのデータなど、該当しなくなった情報が ps の出力結果に含まれていることもあります。

プロセスを選択するオプションを1つも指定しない場合、ps はその制御端末に関連するプロセスすべてについて報告します。制御端末がない場合には、ヘッダー以外には何も報告されません。

ps -ef および ps -o stime は、tty ログインセッションの実際の開始時間を報告せず、実際よりも少し早い時間、つまり getty が最後に tty 行に再生成された時間を報告します。

以前のリリースでは、プロセスまたはイベント (あるいはその両方) のカーネルメモリーアドレスが待機中でも ADDR および WCHAN フィールドに含まれることがありまし



た。現在では、すべての特権で実行されるプロセスで要求されないかぎり、これらのフィールドは常に 0 になっています。mdb 内で `::ps` and `::thread dcmds` を使用すると、値を取得できます。

`ps` はログイン名(ユーザー名)を除いて CSI 対応が可能です。

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | ps – display the status of current processes                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| 形式    | /usr/ucb/ps [-] [aceglnrSuUvwx] [-t <i>term</i> ] [ <i>num</i> ]                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 機能説明  | <p>The ps command displays information about processes. Normally, only those processes that are running with your effective user ID and are attached to a controlling terminal (see <a href="#">termio(7I)</a>) are shown. Additional categories of processes can be added to the display using various options. In particular, the -a option allows you to include processes that are not owned by you (that do not have your user ID), and the -x option allows you to include processes without controlling terminals. When you specify both -a and -x, you get processes owned by anyone, with or without a controlling terminal. The -r option restricts the list of processes printed to running and runnable processes.</p> <p>ps displays in tabular form the process ID, under PID; the controlling terminal (if any), under TT; the cpu time used by the process so far, including both user and system time, under TIME; the state of the process, under S; and finally, an indication of the COMMAND that is running.</p> <p>The state is given by a single letter from the following:</p> <ul style="list-style-type: none"><li>O Process is running on a processor.</li><li>S Sleeping. Process is waiting for an event to complete.</li><li>R Runnable. Process is on run queue.</li><li>Z Zombie state. Process terminated and parent not waiting.</li><li>T Traced. Process stopped by a signal because parent is tracing it.</li></ul> |
| オプション | <p>The following options must all be combined to form the first argument. The ps command accepts the arguments without the leading (-) for historical reasons</p> <ul style="list-style-type: none"><li>-a Includes information about processes owned by others.</li><li>-c Displays the command name rather than the command arguments.</li><li>-e Displays the environment as well as the arguments to the command.</li><li>-g Displays all processes. Without this option, ps only prints interesting processes. Processes are deemed to be uninteresting if they are process group leaders. This normally eliminates top-level command interpreters and processes waiting for users to login on free terminals.</li><li>-l Displays a long listing, with fields F, PPID, CP, PRI, NI, SZ, RSS, and WCHAN as described below.</li><li>-n Produces numerical output for some fields. In a user listing, the USER field is replaced by a UID field.</li><li>-r Restricts output to running and runnable processes.</li></ul>                                                                                                                                                                                                                                                                                                                                                                                                                            |

---

|                |                                                                                                                                                                                                                                                                                                                                                        |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| -S             | Displays accumulated CPU time used by this process and all of its reaped children.                                                                                                                                                                                                                                                                     |
| -t <i>term</i> | Lists only process data associated with the terminal, <i>term</i> . Terminal identifiers may be specified in one of two forms: the device's file name (for example, <code>tty04</code> or <code>term/14</code> ) or, if the device's file name starts with <code>tty</code> , just the digit identifier (for example, <code>04</code> ).               |
| -u             | Displays user-oriented output. This includes fields USER, %CPU, %MEM, SZ, RSS, and START as described below.                                                                                                                                                                                                                                           |
| -U             | Obsolete. This option no longer has any effect. It causes <code>ps</code> to exit without printing the process listing.                                                                                                                                                                                                                                |
| -v             | Displays a version of the output containing virtual memory. This includes fields SIZE, %CPU, %MEM, and RSS, described below.                                                                                                                                                                                                                           |
| -w             | Uses a wide output format, that is, 132 columns rather than 80. If the option letter is repeated, that is, <code>-ww</code> , this option uses arbitrarily wide output. This information is used to decide how much of long commands to print. <i>Note:</i> The wide output option can be viewed only by a superuser or the user who owns the process. |
| -x             | Includes processes with no controlling terminal.                                                                                                                                                                                                                                                                                                       |
| <i>num</i>     | A process number may be given, in which case the output is restricted to that process. This option must be supplied last.                                                                                                                                                                                                                              |

### Display Formats

Fields that are not common to all output formats:

|      |                                                                                                             |
|------|-------------------------------------------------------------------------------------------------------------|
| USER | Name of the owner of the process.                                                                           |
| %CPU | CPU use of the process. This is a decaying average over up to a minute of previous (real) time.             |
| NI   | Process scheduling increment (see <a href="#">getpriority(3C)</a> ).                                        |
| SIZE | The total size of the process in virtual memory, including all mapped files and devices, in kilobyte units. |
| SZ   | Same as SIZE.                                                                                               |
| RSS  | Real memory (resident set) size of the process, in kilobyte units.                                          |
| UID  | Numerical user-ID of process owner.                                                                         |
| PPID | Numerical ID of parent of process.                                                                          |
| CP   | Short-term CPU utilization factor (used in scheduling).                                                     |
| PRI  | The priority of the process (higher numbers mean lower priority).                                           |

- START

The starting time of the process, given in hours, minutes, and seconds. A process begun more than 24 hours before the ps inquiry is executed is given in months and days.
- WCHAN

The address of an event for which the process is sleeping (if blank, the process is running).
- %MEM

The ratio of the process's resident set size to the physical memory on the machine, expressed as a percentage.
- F

Flags (hexadecimal and additive) associated with the process. These flags are available for historical purposes; no meaning should be currently ascribed to them.

A process that has exited and has a parent, but has not yet been waited for by the parent, is marked <defunct> ; otherwise, ps tries to determine the command name and arguments given when the process was created by examining the user block.

ファイル

/dev/tty\*

/etc/passwd

UID information supplier

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE   |
|----------------|-------------------|
| Availability   | compatibility/ucb |

関連項目

[kill\(1\)](#), [ps\(1\)](#), [whodo\(1M\)](#), [getpriority\(3C\)](#), [proc\(4\)](#), [attributes\(5\)](#), [termio\(7I\)](#)

注意事項

Things can change while ps is running. The picture ps gives is only a close approximation to the current state. Some data printed for defunct processes is irrelevant.

|         |                                                                                                                                                                                                                                                                                                                                                    |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | ptree – プロセスツリーを表示する                                                                                                                                                                                                                                                                                                                               |
| 形式      | /usr/bin/ptree [-a] [-c] [-z zone] [pid   user]...                                                                                                                                                                                                                                                                                                 |
| 機能説明    | ptree ユーティリティーは、指定した <i>pid</i> または <i>user</i> を含むプロセスツリーを表示します。子プロセスは各親プロセスからインデントされて表示されます。すべて数字の引数はプロセス ID と解釈され、それ以外の場合はユーザーログイン名と見なされます。デフォルトはすべてのプロセスです。                                                                                                                                                                                    |
| オプション   | <p>サポートしているオプションは、次のとおりです。</p> <p>-a       すべて。プロセス 0 の子を含む、すべてのプロセスを表示します。</p> <p>-c       契約。親子関係に加えて、プロセス契約メンバーシップを表示します。<a href="#">process(4)</a> を参照してください。このオプションは、-a オプションの意味を含んでいます。</p> <p>-z zone   ゾーン。指定された <i>zone</i> 内のプロセスだけを表示します。各ゾーン ID は、ゾーン名または数値のゾーン ID のどちらでも指定できます。</p> <p>          このオプションは、グローバルゾーンで実行する場合にのみ便利です。</p> |
| オペランド   | <p>次のオペランドがサポートされています。</p> <p><i>pid</i>     プロセス ID またはプロセス ID のリスト。ptree では、<i>/proc/nnn</i> もプロセス ID として認識されるため、シェル展開 <i>/proc/*</i> を使用すると、システム内のすべてのプロセスを指定できます。</p> <p><i>user</i>     ユーザー名またはユーザー名のリスト。実効ユーザー ID が指定されたものと一致するプロセスが表示されます。</p>                                                                                             |
| 使用例     | <p>例1 ptree の使用</p> <p>次の例では、コマンド名 <i>ssh</i> に一致するプロセスのプロセスツリー (プロセス 0 の子を含む) を表示します。</p> <pre>\$ ptree -a 'pgrep ssh' 1      /usr/sbin/init 100909 /usr/lib/ssh/sshd 569150 /usr/lib/ssh/sshd 569157 /usr/lib/ssh/sshd 569159 -ksh 569171 bash 569173 /bin/ksh 569193 bash</pre>                                                                 |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0       正常終了。</p> <p>0 以外   エラーが発生しました。</p>                                                                                                                                                                                                                                                                             |

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ       | 属性値            |
|-------------|----------------|
| 使用条件        | system/core-os |
| インタフェースの安定性 | 下記を参照。         |

人間が読める形式の出力は「不確実」です。オプションは「確実」です。

[gcore\(1\)](#), [ldd\(1\)](#), [pargs\(1\)](#), [pgrep\(1\)](#), [pkill\(1\)](#), [plimit\(1\)](#), [pmap\(1\)](#), [preap\(1\)](#), [proc\(1\)](#), [ps\(1\)](#),  
[ppgzs\(1\)](#), [pwd\(1\)](#), [rlogin\(1\)](#), [time\(1\)](#), [truss\(1\)](#), [wait\(1\)](#), [fcntl\(2\)](#), [fstatt\(2\)](#), [setuid\(2\)](#),  
[dlopen\(3C\)](#), [signal.h\(3HEAD\)](#), [core\(4\)](#), [proc\(4\)](#), [process\(4\)](#), [attributes\(5\)](#), [zones\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | pvs – 動的オブジェクトの内部バージョン情報の表示                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| 形式    | pvs [-Cdlnorsv] [-I <i>index-expr</i> ] [-N <i>name</i> ] <i>file</i> ...                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| 機能説明  | <p>pvs ユーティリティーは、ELF ファイル内に含まれている内部バージョン情報を表示します。一般に、これらのファイルは動的な実行可能ファイル、共有オブジェクト、および場合によっては再配置可能なオブジェクトです。このバージョン情報は、2つのカテゴリのいずれかに分類できます。</p> <ul style="list-style-type: none"> <li>■ バージョン定義</li> <li>■ バージョン依存関係</li> </ul> <p>バージョン定義は、ELF ファイルによって使用可能になるインタフェースについて記述します。各バージョン定義は、このファイルによって提供される一連の大域シンボルに関連付けられています。バージョン定義は、リンカーで -M オプションと関連付けられた <i>mapfile</i> 指令を使用することによって、作成時にファイルに割り当てることができます。詳細は、『<a href="#">リンカーとライブラリ</a>』を参照してください。</p> <p>バージョン依存関係は、共有オブジェクトの依存関係があるバージョン定義での、動的オブジェクトのバインディングの要件について記述します。動的オブジェクトが共有オブジェクトとともに作成される場合、リンカーは共有オブジェクトが依存関係にあることを示す情報を動的オブジェクト内に記録します。この依存関係は実行時に満たされている必要があります。共有オブジェクトにバージョン定義も含まれている場合は、動的オブジェクトの大域シンボル要件を満たすそれらのバージョン定義も、作成される動的オブジェクトに記録されます。プロセスの初期化時に、プロセスを作成するために使用される動的オブジェクトのインタフェース要件を検証する手段として、実行時リンカーはバージョン依存関係を使用します。</p> |
| オプション | <p>以下のオプションがサポートされています。-d オプションまたは -r オプションのいずれも指定されていない場合は、両方が有効になります。</p> <ul style="list-style-type: none"> <li>-C                   Demangles C++ シンボル名を復号化します。</li> <li>-d                   バージョン定義情報を出力します。</li> <li>-I <i>index-expr</i>   特定のバージョンインデックスまたはインデックスの範囲で検索するバージョンを限定します。たとえば、オブジェクト内のインデックス 3 のバージョンを表示するには、次のように指定します。</li> </ul> <pre>example% pvs -I 3 filename</pre> <p><i>index-expr</i> には、上記の例に示したように、特定のバージョンを指定する負でない単一の整数値を指定できます。または、<i>index-expr</i> は、バージョンの範囲を示すコロン (:) で区切られた 2 つの負でない単一の整数値で構成することもできます。次の例では、ファイル内のバージョン 3、4、および 5 を表示します。</p> <pre>example% pvs -I 3:5 filename</pre>                                                                                                                                                                                    |

インデックスの範囲を指定する場合、2 番目の値を省略すると、ファイル内の最後の項目を意味することができます。たとえば、次の文は 10 番目から最後まですべてのバージョンを一覧表示します。

```
example% pvs -I 10: filename
```

マッチングオプション (-I、-N) の詳細な情報については、「マッチングオプション」を参照してください。

- l バージョン管理のためにバインディングが大域からローカルに縮小された、すべてのシンボルを出力します。規約上、これらのシンボルエントリは、`.symtab` セクションに配置され、出力ファイルを表す `FILE` シンボルと、出力ファイルを生成するために使用される最初の入力ファイルを表す `FILE` シンボルの間に置かれます。これらの縮小されたシンボルエントリには、加工されたバージョン定義 `_LOCAL` が割り当てられます。ファイルが取り除かれている (`strip(1)` を参照) 場合、またはシンボルエントリの規約が判別できない場合、縮小されたシンボルは出力されません。

-l オプションを使用すると、-s オプションが暗黙的に有効になります。

- n バージョン定義情報を正規化します。デフォルトでは、オブジェクト内のすべてのバージョン定義が表示されます。ただし、バージョン定義は別のバージョン定義を継承することがあります。正規化すると、各継承リストの先頭のみが表示されます。

- N *name* -d オプションとともに -N を使用すると、指定されたバージョン定義 *name* とそれを継承したバージョン定義の情報のみが出力されます。

-r オプションとともに -N を使用すると、指定された依存関係ファイル *name* の情報のみが出力されます。ファイル名の後ろでバージョンを括弧で囲んで指定することで、依存関係ファイルを特定のバージョンに限定できます。

```
example% pvs -N 'dependency (version)' filename
```

マッチングオプション (-I、-N) の詳細な情報については、「マッチングオプション」を参照してください。

- o バージョン定義を 1 行単位で出力します。デフォルトでは、ファイル、バージョン定義、およびシンボルの出力は、人間による検査を容易にするためにインデントされます。このオプションを使用すると、各出力行の前にファイル名とバージョン定義名が付けられ、自動化ツールで分析する場合、より便利になることがあります。
- r バージョン依存関係 (要件) 情報を出力します。



- s 各バージョン定義に関連付けられているシンボルを出力します。オブジェクトによって定義されているバージョンのデータシンボルには、データ項目のサイズ(バイト単位)が一緒に出力されます。
- v 詳細情報を出力します。弱いバージョン定義とバージョン定義の継承を示します。-N オプションおよび -d オプションとともに使用すると、基本バージョン定義の継承も表示されます。-s オプションとともに使用すると、バージョンシンボル定義も表示されます。

オペランド 次のオペランドがサポートされています。

*file* 内部バージョン情報を表示する ELF ファイル。

## 使用法

### マッチングオプション

-I オプションと -N オプションは、まとめてマッチングオプションと呼ばれます。これらのオプションは、インデックスまたは名前によって、検査するバージョンの範囲を絞り込むために使用されます。

任意の数およびタイプのマッチングオプションを組み合わせ、対象となる *pvs* の呼び出しで使用できます。この場合、*pvs* は使用されたすべてのマッチングオプションと一致する、すべてのバージョンの上位集合を表示します。この機能を使用すると、各項目の指定にもっとも簡単な形式を使用し、複雑な項目のグループを選択できます。

## 使用例

### 例1 バージョン定義の表示

次の例では、*libelf.so.1* のバージョン定義を表示します。

```
% pvs -d /lib/libelf.so.1
 libelf.so.1;
 SUNW_1.1
```

### 例2 1行単位の出力の作成

*mapfile* バージョン管理指令を作成するのに適した、正規化された1行単位の出力は、-n オプションと -o オプションを使用して作成できます。

```
% pvs -don /lib/libelf.so.1
/lib/libelf.so.1 - SUNW_1.1;
```

### 例3 バージョン要件の表示

次の例では、*ldd* と *pvs* のバージョン要件を表示します。

```
% pvs -r /usr/bin/ldd /usr/bin/pvs
/usr/bin/ldd:
 libelf.so.1 (SUNW_1.1);
 libc.so.1 (SUNW_1.1);
/usr/bin/pvs:
```

**例3 バージョン要件の表示 (続き)**

```
libelf.so.1 (SUNW_1.1);
libc.so.1 (SUNW_1.1);
```

**例4 依存関係シンボルのバージョンの判別**

次の例では、`ldd` コマンドによって実行時に `printf` 関数を見つけることが予期されている共有オブジェクトと、それが属するバージョンを表示します。

```
% pvs -ors /usr/bin/ldd | grep ' printf'
/usr/bin/ldd - libc.so.1 (SYSVABI_1.3): printf;
```

**例5 特定バージョンのすべての依存関係シンボルの判別**

`-N` オプションを使用すると、特定バージョンに属する依存関係のすべてのシンボルのリストを取得できます。`ldd` が `libc.so.1` のバージョン `SYSVABI_1.3` から見つけるシンボルを判別するには、次のコマンドを実行します。

```
% pvs -s -N 'libc.so.1 (SYSVABI_1.3)' /usr/bin/ldd
```

```
libc.so.1 (SYSVABI_1.3):
_exit;
strstr;
printf;
__fpstart;
strncmp;
lseek;
strcmp;
getopt;
execl;
close;
fflush;
wait;
strerror;
putenv;
sprintf;
getenv;
open;
perror;
fork;
strlen;
geteuid;
access;
setlocale;
atexit;
fprintf;
exit;
```

# 例5 特定バージョンのすべての依存関係シンボルの判別 (続き)

```
read;
malloc;
```

ldd によって使用される具体的なシンボルのリストは、Solaris のリリースによって異なる場合があります。

## 例6 基本として定義されているバージョンの、インデックスによる表示

規約上、オブジェクトによって定義されている基本大域バージョンは、そのオブジェクトの名前を持ちます。たとえば、pvs の基本バージョンには、'pvs' という名前が付けられます。オブジェクトの基本バージョンは、常にバージョンインデックス 1 です。したがって、-I オプションを使用すると、名前を指定することなく、オブジェクトの基本バージョンを表示できます。

```
% pvs -v -I 1 /usr/bin/pvs
pvs [BASE];
```

**終了ステータス** 要求されたバージョン情報が見つからなかった場合は、ゼロ以外の値が返されます。それ以外の場合は、値 0 が返されます。

次のいずれかが真である場合は、バージョン情報が見つからなかったと判定されます。

- -d オプションが指定され、バージョン定義が見つからない。
- -r オプションが指定され、バージョン要件が見つからない。
- -d オプションと -r オプションのいずれもが指定されず、バージョン定義またはバージョン要件が見つからない。

**属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値                                |
|-------|------------------------------------|
| 使用条件  | developer/base-developer-utilities |

**関連項目** [elfdump\(1\)](#), [ld\(1\)](#), [ldd\(1\)](#), [strip\(1\)](#), [elf\(3ELF\)](#), [attributes\(5\)](#)

『リンカーとライブラリ』

|         |                                                                                                                                                                                |
|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前      | pwd – 作業中のディレクトリの出力                                                                                                                                                            |
| 形式      | /usr/bin/pwd                                                                                                                                                                   |
| 機能説明    | <p>pwd は、現在の作業用 ディレクトリの絶対パス名を標準出力に書き出します。</p> <p>Bourne シェル (<a href="#">sh(1)</a>) と Korn シェル (<a href="#">ksh(1)</a> および <a href="#">ksh88(1)</a>) にも組み込み pwd コマンドがあります。</p> |
| 環境変数    | pwd の実行に影響を与える次の環境変数についての詳細は、 <a href="#">environ(5)</a> を参照してください。LANG、LC_ALL、LC_MESSAGES、および NLSPATH。                                                                        |
| 終了ステータス | <p>次の終了ステータスが返されます。</p> <p>0      正常終了。</p> <p>&gt;0    エラーが発生した。</p> <p>エラーが検出された場合、出力は標準出力に書き込まれず、診断メッセージが標準エラーに書かれます。終了ステータスは 0 にはなりません。</p>                                |
| 属性      | 属性についての詳細は、マニュアルページの <a href="#">attributes(5)</a> を参照してください。                                                                                                                  |

| 属性タイプ       | 属性値                                     |
|-------------|-----------------------------------------|
| 使用条件        | system/core-os                          |
| CSI         | 有効                                      |
| インタフェースの安定性 | 確実                                      |
| 標準          | <a href="#">standards(5)</a> を参照してください。 |

|      |                                                                                                                                                                                                                                   |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 関連項目 | <a href="#">cd(1)</a> , <a href="#">ksh(1)</a> , <a href="#">ksh88(1)</a> , <a href="#">sh(1)</a> , <a href="#">shell_builtins(1)</a> , <a href="#">attributes(5)</a> , <a href="#">environ(5)</a> , <a href="#">standards(5)</a> |
| 診断   | <p>pwd: cannot determine current directory!</p> <p>ネットワーク管理者に問い合わせてください。</p>                                                                                                                                                      |
| 注意事項 | 現在のディレクトリまたはその 1 つ上のディレクトリに移動する場合、pwd が正しく応答しないことがあります。 <a href="#">cd(1)</a> コマンドにフルパス名を付けて使用し、この状態を修正してください。                                                                                                                    |

|                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前                               | radadrngen – code generator                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| 形式                               | <pre> /usr/bin/radadrngen [-N] spec.xml  /usr/bin/radadrngen [-N] -c dir [-r [ -m ] [ -s ]] spec.xml  /usr/bin/radadrngen [-N] -j dir [-i] spec.xml  /usr/bin/radadrngen [-N] {-o docbook   docbook-man   man   rmdoc} spec.xml  /usr/bin/radadrngen -d baseline.xml spec.xml </pre>                                                                                                                                                                                                                                                                                                                                                  |
| 機能説明                             | <p>The radadrngen command is the ADR IDL processing tool. Its primary purpose is to generate language bindings for the rad server and the various rad client environments. It can also generate documentation, and audit changes to interfaces for consistency with their versions.</p> <p>All invocations validate the given <code>spec.xml</code> against the ADR schema.</p>                                                                                                                                                                                                                                                       |
| オプション                            | The following options are supported:                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Common Options                   | <p><b>-N</b></p> <p>Do not limit what types can be nullable. This restriction is in place to facilitate mapping to languages where certain types are not naturally nullable. This option relaxes this restriction for uses where it is both permissible and advantageous for all types to be nullable.</p>                                                                                                                                                                                                                                                                                                                            |
| C Definition Options             | <p><b>-c dir</b></p> <p>Generate C definitions for the types and interfaces defined in the input file. Files will be created under <i>dir</i>.</p> <p><b>-r</b></p> <p>Generate C server-side definitions that reference entry points using the prescribed entry-point naming scheme.</p> <p><b>-m</b></p> <p>Do not generate separate definitions for inherited interfaces. By default, new definitions are generated for inherited interfaces, replicating the definitions in the inherited interface.</p> <p><b>-s</b></p> <p>Generate C stubs for entry points referenced by the definitions created by the <b>-r</b> option.</p> |
| Java Definition Options          | <p><b>-j dir</b></p> <p>Generate Java interfaces for the types and interfaces defined in the input file. The Java source tree will be created under <i>dir</i>.</p> <p><b>-i</b></p> <p>Generate concrete implementation classes for the interfaces created for structured types.</p>                                                                                                                                                                                                                                                                                                                                                 |
| Documentation Generation Options | <p><b>-o format</b></p> <p>Generate documentation, in the given format, for the interface definition in <code>spec.xml</code>. Documentation can be improved by decorating the various parts of the interface definition with <i>summary</i> and <i>doc</i> elements.</p> <p>Valid formats:</p>                                                                                                                                                                                                                                                                                                                                       |

- docbook
  - standard docbook format
- docbook-man
  - man page content in docbook format, a precursor to man format, below
- man
  - a man page, in [nt]roff format
- rmdoc
  - spec.xml, with the *summary* and *doc* elements removed
- Interface Comparison Options
  - d baseline.xml
    - Compare the interfaces defined by baseline.xml and spec.xml and verify that the version of the modified interface is consistent with the differences found. An error message is displayed if the versions are inconsistent.

オペランド

The following operand must be specified on the command line:

spec.xml

The path to the API specification for which type and interface definitions must be generated.

ファイル

/usr/share/lib/xml/rng/radadr.rng.1

The core ADR RelaxNG schema definition.

/usr/share/lib/xml/rng/radadr-doc.rng.1

The ADR RelaxNG schema definition for documentation elements.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE      | ATTRIBUTE VALUE       |
|---------------------|-----------------------|
| Availability        | system/management/rad |
| Interface Stability | Private               |

関連項目

[rad\(1M\)](#), [attributes\(5\)](#)

名前                    `ranlib` – convert archives to random libraries

形式                    `ranlib archive`

機能説明                The `ranlib` utility was used in SunOS 4.x to add a table of contents to archive libraries, which converted each archive to a form that could be linked more rapidly. This is no longer needed, as the `ar(1)` command automatically provides all the functionality `ranlib` used to provide.

This script is provided as a convenience for software developers who need to maintain Makefiles that are portable across a variety of operating systems.

終了ステータス        `ranlib` has exit status 0.

属性                    See [attributes\(5\)](#) for descriptions of the following attributes:

| ATTRIBUTE TYPE | ATTRIBUTE VALUE                    |
|----------------|------------------------------------|
| Availability   | developer/base-developer-utilities |

関連項目                [ar\(1\)](#), [ar.h\(3HEAD\)](#), [attributes\(5\)](#)

|       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 名前    | rcapstat – リソース上限適用デーモンの統計の報告                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 形式    | rcapstat [-g] [-p   -z] [-T u   d ] [interval [count]]                                                                                                                                                                                                                                                                                                                                                                                                             |
| 機能説明  | <p>rcapstat コマンドは、<a href="#">rcapd(1M)</a> によって上限が設けられたゾーンまたはプロジェクトについて報告します。各報告には、プロジェクトまたはゾーンに関する統計、およびページング統計が含まれています。ページングとは、物理メモリーから(または物理メモリーへ) ページと呼ばれるメモリーの一部を再配置する処理を指します。rcapd は、もっとも使用されていないページをページアウトします。</p> <p>最初に発行する報告では、ページング統計はデーモンの起動以降の活動を示します。以後の報告では、前回の報告以降の活動を示します。</p> <p>報告は、<i>interval</i> 秒ごとに <i>count</i> によって指定されている回数まで発行されます。<i>count</i> が指定されていない場合は、停止されることなく発行され続けます。</p>                                                   |
| オプション | <p>サポートしているオプションは、次のとおりです。</p> <ul style="list-style-type: none"> <li>-g 大域統計。メモリー上限適用の最小メモリー使用率を報告します(<a href="#">rcapadm(1M)</a>を参照)。また、インストールされている物理メモリーに対する割合(パーセント)として現在のメモリー使用率を報告します。</li> <li>-p 上限が設けられたプロジェクトの統計を報告します。これは、オプションが指定されていない場合のデフォルトです。</li> <li>-T u   d タイムスタンプを表示します。<br/>時間の内部表現の出力表現に u を指定します。<a href="#">time(2)</a> を参照してください。標準の日付フォーマットに d を指定します。<a href="#">date(1)</a> を参照してください。</li> <li>-z 上限が設けられたゾーンの統計を報告します。</li> </ul> |
| 出力    | <p>rcapstat の報告の列見出しの定義、および報告の解釈方法についての情報を、次のリストに示します。</p> <ul style="list-style-type: none"> <li>id 上限が設けられたプロジェクトまたはゾーンのプロジェクト ID またはゾーン ID。</li> <li>project プロジェクト名。</li> <li>zone ゾーン名。</li> <li>nproc 前回の報告以降のプロジェクトまたはゾーンのプロセス数。</li> <li>vm ディスクまたはメモリーのスワップを予約するすべての匿名マッピングの合計。</li> <li>rss プロジェクトまたはゾーンのプロセス常駐の合計設定サイズ(RSS)。単位は、キロバイト(K)、メガバイト(M)、またはギガバイト(G)です。サイズには共有ページは含まれていません。</li> </ul>                                                         |



|              |                                                                                                                                                             |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>cap</b>   | プロジェクトまたはゾーンの RSS の上限。メモリーの上限を指定する方法については、 <code>rcapd(1M)</code> を参照してください。                                                                                |
| <b>at</b>    | <code>rcapd</code> がページアウトを試みるメモリーの合計容量。<br><br>ページングとは、物理メモリーから (または物理メモリーへ) ページと呼ばれるメモリーの一部を再配置する処理を指します。 <code>rcapd</code> は、もっとも使用されていないページをページアウトします。 |
| <b>avgat</b> | <code>rcapd</code> が各サンプリングサイクル中にページアウトを試みるメモリーの平均サイズ。 <code>rcapd</code> が RSS をサンプリングする頻度は、 <code>rcapadm(1M)</code> で設定できます。                             |
| <b>pg</b>    | <code>rcapd</code> が正常にページアウトするメモリーの合計容量の見積り。                                                                                                               |
| <b>avgpg</b> | <code>rcapd</code> が各サンプリングサイクル中に正常にページアウトするメモリーの平均サイズの見積り。 <code>rcapd</code> がプロセス RSS をサンプリングする頻度は <code>rcapadm</code> で設定できます。                         |

オペランド 次のオペランドがサポートされています。

|                 |                                                                                                                |
|-----------------|----------------------------------------------------------------------------------------------------------------|
| <b>interval</b> | 報告間隔を指定します (秒)。デフォルトの間隔は 5 秒です。                                                                                |
| <b>count</b>    | 報告を生成する回数を指定します。デフォルトでは、終了シグナルを受信するまで、あるいは、 <code>rcapd</code> プロセスが終了するまで、 <code>rcapstat</code> は統計を報告し続けます。 |

## 使用例

例1 `rcapstat` を使用した、上限とプロジェクト情報の報告

2 人のユーザーに関連付けられた 2 つのプロジェクトに、上限が定義されています。`user1` の上限は 50M バイト、`user2` の上限は 10M バイトです。

次のコマンドは、5 つの報告を 5 秒間のサンプリング間隔で生成します。

example# **rcapstat 5 5**

```

id project nproc vm rss cap at avgat pg avgpg
112270 user1 24 123M 35M 50M 50M 0K 3312K 0K
78194 user2 1 2368K 1856K 10M 0K 0K 0K 0K
id project nproc vm rss cap at avgat pg avgpg
112270 user1 24 123M 35M 50M 0K 0K 0K 0K
78194 user2 1 2368K 1856K 10M 0K 0K 0K 0K
id project nproc vm rss cap at avgat pg avgpg
112270 user1 24 123M 35M 50M 0K 0K 0K 0K
78194 user2 1 2368K 1928K 10M 0K 0K 0K 0K
id project nproc vm rss cap at avgat pg avgpg
112270 user1 24 123M 35M 50M 0K 0K 0K 0K
78194 user2 1 2368K 1928K 10M 0K 0K 0K 0K
```

例1 rcapstatを使用した、上限とプロジェクト情報の報告 (続き)

出力の最初の3行は1回目の報告です。ここには、2つのプロジェクトに関する上限とプロジェクトの情報、および rcapd 起動以降のページング統計が記載されています。at と pg の列において、user1 にはゼロより大きな値が入っており、user2 にはゼロが入っています。これは、1回目の報告の期間中、user1 は上限を超えたが、user2 は超えなかったことを意味します。

2回目以降の報告では、目立った活動はありません。

例2 rcapstatを使用した、プロジェクトのRSSの監視

```
example% rcapstat 5 5
```

| id     | project | nproc | vm   | rss | cap | at   | avgat | pg    | avgpg |
|--------|---------|-------|------|-----|-----|------|-------|-------|-------|
| 376565 | user1   | 57    | 209M | 46M | 10M | 440M | 220M  | 5528K | 2764K |
| 376565 | user1   | 57    | 209M | 44M | 10M | 394M | 131M  | 4912K | 1637K |
| 376565 | user1   | 56    | 207M | 43M | 10M | 440M | 147M  | 6048K | 2016K |
| 376565 | user1   | 56    | 207M | 42M | 10M | 522M | 174M  | 4368K | 1456K |
| 376565 | user1   | 56    | 207M | 44M | 10M | 482M | 161M  | 3376K | 1125K |

プロジェクト user1 には、物理メモリーの上限を超える RSS があります。pg 列の 0 以外の値が示しているとおおり、rcapd は、このプロジェクトのプロセスの物理メモリーの利用率を上限に合わせて下げようと、メモリーをページアウトし続けています。ただし、rcapd が成功していないのは、rss 値が相応の減少を示していないことでわかります。これは、アプリケーションの常駐メモリーがアクティブに使用されているために rcapd が作業用セットに影響を及ぼしていることを意味します。この状況では、作業用セットのサイズ (WSS) が減少するか、上限を上げるか、アプリケーションのメモリーアクセスパターンを変更するまで、高いページフォルト率とそれに関連する入出力がシステムで続きます。ページフォルトが発生するのは、新しいページを作成する必要があるとき、あるいは、システムがスワップデバイスからページをコピーする必要があるときです。

例3 プロジェクトの作業セットサイズの決定

この例は、例 1 の続きで、同じプロジェクトを使用します。

```
example% rcapstat 5 5
```

| id     | project | nproc | vm   | rss | cap  | at   | avgat | pg    | avgpg |
|--------|---------|-------|------|-----|------|------|-------|-------|-------|
| 376565 | user1   | 56    | 207M | 44M | 10M  | 381M | 191M  | 15M   | 7924K |
| 376565 | user1   | 56    | 207M | 46M | 10M  | 479M | 160M  | 2696K | 898K  |
| 376565 | user1   | 56    | 207M | 46M | 10M  | 424M | 141M  | 7280K | 2426K |
| 376565 | user1   | 56    | 207M | 43M | 10M  | 401M | 201M  | 4808K | 2404K |
| 376565 | user1   | 56    | 207M | 43M | 10M  | 456M | 152M  | 4800K | 1600K |
| 376565 | user1   | 56    | 207M | 44M | 10M  | 486M | 162M  | 4064K | 1354K |
| 376565 | user1   | 56    | 207M | 52M | 100M | 191M | 95M   | 1944K | 972K  |
| 376565 | user1   | 56    | 207M | 55M | 100M | 0K   | 0K    | 0K    | 0K    |
| 376565 | user1   | 56    | 207M | 56M | 100M | 0K   | 0K    | 0K    | 0K    |
| 376565 | user1   | 56    | 207M | 56M | 100M | 0K   | 0K    | 0K    | 0K    |

## 例3 プロジェクトの作業セットサイズの決定 (続き)

```

376565 user1 56 207M 56M 100M 0K 0K 0K 0K
376565 user1 56 207M 56M 100M 0K 0K 0K 0K

```

プロジェクトの上限を上げるか、上限適用の最小物理メモリー使用率を変更して、上限の適用を禁止することにより (rcapadm(1M) 参照)、常駐セットを作業用セットにすることができます。前の例で示したように、`rss` 列が安定してプロジェクトの WSS を表示するようになる可能性があります。WSS は、このプロジェクトのプロセスがページフォルトを絶え間なく起こさずに動作できる、上限の最小値です。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定されました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

| 属性タイプ | 属性値                                |
|-------|------------------------------------|
| 使用条件  | system/resource-mgmt/resource-caps |

関連項目 [rcapadm\(1M\)](#), [rcapd\(1M\)](#), [attributes\(5\)](#)

『Oracle Solaris のシステム管理 (資源管理)』の「リソース上限デーモンによる物理メモリーの制御」

注意事項 `rcapstat` に指定した間隔が、(`rcapadm(1M)` で) `rcapd` に指定した報告間隔よりも短い場合、いくつかの間隔に対する出力がゼロになることがあります。これは、`rcapd` は、`rcapadm` で指定された間隔ほど頻繁に統計を更新しないためです。この間隔は、`rcapstat` によって使用されるサンプリング間隔より正確ではなく、かつ無関係です。

名前 rcp – リモートファイルコピー

形式 `rcp [-p] [-a] [-K] [-x] [-PN | -PO] [-k realm] filename1 filename2`  
`rcp [-pr] [-a] [-K] [-x] [-PN | -PO] [-k realm] filename... directory`

機能説明 rcp コマンドは、マシン間でファイルをコピーします。*filename* 引数、*directory* 引数にリモートファイルを指定する場合は、次のような書式を使用します。

*hostname:path*

ローカルファイルを指定する場合は、コロン(:)を使用しないか、コロンの前にスラッシュ(/)を付けます。

*hostname* には、IPv4 または IPv6 のアドレス文字列を指定できます([inet\(7P\)](#)、[inet6\(7P\)](#)を参照)。IPv6 のアドレスにはコロンが含まれているので、*hostname* を角括弧で囲む必要があります。*hostname* を角括弧で囲まないと、最初のコロンが *hostname* と *path* の区切り文字と見なされてしまいます。たとえば、次のように指定します。

`[1080::8:800:200C:417A]:tmp/file`

また、フルパス名を指定しないと、*filename* はホスト(*hostname*)上のホームディレクトリから見た相対パスと解釈されます。リモートホスト上のパス名と解釈させるには、パス名(*path*)を \、"、' など囲みます。Kerberos 認証バージョンの rcp は IPv6 が無効であることに注意してください。

rcp コマンドの使用時、パスワードを入力する必要はありません。Kerberos 認証をコマンド行オプションで有効にした場合、rcp は Kerberos 認証を使用します。そうでない場合、現在のローカルユーザー名が *hostname* 上に存在し、[rsh\(1\)](#) でリモートコマンドを実行できる必要があります。

rcp セッションで Kerberos 認証を使用するには、次の任意の Kerberos 固有オプションを使用します。-a、-PN または -PO、-x、および -k *realm*。これらのオプションのうち、-a、-x、および -PN または -PO は、[krb5.conf\(4\)](#) の [appdefaults] セクションにも指定できます。これらのオプションの使用法と期待される動作については、次の「オプション」セクションを参照してください。Kerberos 承認を使用する場合、アカウントの承認は、[krb5\\_auth\\_rules\(5\)](#) の規則によって制御されます。Kerberos 承認が失敗した場合、-PO オプションをコマンド行に明示的に指定するか、[krb5.conf\(4\)](#) 経由で指定している場合にのみ、rhosts を使用する通常の rcp へのフォールバックが発生します。Kerberos 承認が成功した場合、パスワードの入力は求められずに、リモートコピーは成功します。-PN または -PO、-x、および -k *realm* オプションは、-a オプションの上位集合であることに注意してください。

rcp では、ソースファイルもターゲットファイルも現在のマシン上に存在しないような「サードパーティコピー」も処理できます。ホスト名は、次の形式になります。

*username@hostname:filename*

この場合、現在のローカルユーザー名でなく、*username* に指定されたユーザー名がリモートホスト上のユーザー名として使用されます。また、*rcp* は、リモートホストのインターネットドメインアドレス指定をサポートしており、次の形式で指定できます。

*username@host.domain:filename*

使用するユーザー名、ホスト名、ホストの所属するドメインを指定できます。*filename* にフルパス名を指定しないと、*username* に指定したリモートホスト上のユーザーのホームディレクトリから見た相対アドレスと解釈されます。

## オプション

サポートしているオプションは、次のとおりです。

- a           このオプションは、Kerberos 認証を明示的に有効にし、アクセス制御において *.k5login* ファイルを信用します。サーバー側での *in.rshd(1M)* による承認チェックが成功して、なおかつ、*.k5login* ファイルがアクセスを許可する場合、ユーザーは *rcp* 転送を実行することが許可されます。
- k *realm*   *rcp* は、*krb5.conf(4)* で定義されているリモートホストの領域ではなく、指定した *realm* 内のリモートホスト用のチケットを取得します。
- K *realm*   このオプションは、Kerberos 認証を明示的に無効にします。このオプションを使用すると、*krb5.conf(4)* の *autologin* 変数をオーバーライドできます。
- p           必要に応じて、各コピーに、オリジナルのファイルと同じ変更時間、アクセス時間、モード、ACL を設定します。
- PO  
-PN          Kerberos の「*rcmd*」プロトコルの新しいバージョン (-PN) または古いバージョン (-PO) を明示的に要求します。新しいプロトコルは、古いプロトコルで蔓延している多くのセキュリティの問題を回避し、より安全であると考えられますが、古い (MIT/SEAM) サービスとは相互運用できません。PO オプションで古いプロトコルを明示的に指定するか、*krb5.conf(4)* 経由で指定していないかぎり、デフォルトでは、新しいプロトコルが使用されます。古い「*rcmd*」プロトコルを使用しているときに Kerberos 承認が失敗した場合、Kerberos 承認を使用しない通常の *rcp* へのフォールバックが発生します。これは、新しい、より安全な「*rcmd*」プロトコルを使用している場合には当てはまりません。
- r           指定された *filename* のサブツリーをコピーします。この場合、コピー先はディレクトリでなければなりません。
- x           ホスト間で転送される情報を暗号化します。このコマンド自身は暗号化されずにリモートシステムに送信されることに注意してください。それ以降の転送はすべて暗号化されます。

使用法
 

ファイルサイズが2Gバイト(2<sup>31</sup>バイト)以上の場合の rcp の動作については、[largefile\(5\)](#)を参照してください。

rcp コマンドでは、IPv6を使用できます。[ip6\(7P\)](#)を参照してください。現在のところ、IPv6はKerberos V5 認証ではサポートされません。

Kerberos 承認を使用する rcp セッションの場合、各ユーザーは自分のホームディレクトリの .k5login ファイルに、専用の承認リストを持つことができます。このファイルの各行には、形式 *principal/instance@realm* の Kerberos 主体名が入っている必要があります。~/ .k5login ファイルが存在する場合、起点ユーザーが~/ .k5login ファイルに指定された主体の1人であると認証された場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。それ以外の場合は、*authenticated-principal-name* → *local-user-name* マッピング規則を使用して、起点ユーザーの認証された主体名をローカルアカウント名にマッピングできる場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。 .k5login ファイル(アクセス制御用)が処理されるのは、Kerberos 認証が行われるときだけです。

終了ステータス
 

次の終了ステータスが返されます。

0       すべてのファイルを正常にコピーしました。

>0     エラーが発生した。

終了コードの注意点に関する「注意事項」の項を参照してください。

ファイル
 

[\\$HOME/.profile](#)

[\\$HOME/.k5login](#)            アクセスを許可する Kerberos 主体を含むファイル

[/etc/krb5/krb5.conf](#)      Kerberos 構成ファイル

属性
 

属性についての詳細は、マニュアルページの [attributes\(5\)](#)を参照してください。

| 属性タイプ | 属性値                             |
|-------|---------------------------------|
| 使用条件  | service/network/network-clients |
| CSI   | 有効                              |

関連項目
 

[cpio\(1\)](#), [ftp\(1\)](#), [rlogin\(1\)](#), [rsh\(1\)](#), [setfacl\(1\)](#), [tar\(1\)](#), [tar\(1\)](#), [in.rshd\(1M\)](#), [hosts.equiv\(4\)](#), [krb5.conf\(4\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [krb5\\_auth\\_rules\(5\)](#), [inet\(7P\)](#), [inet6\(7P\)](#), [ip6\(7P\)](#)

注意事項
 

rcp の目的は、別々のホスト間でファイルをコピーすることです。このため、たとえば次のような指定で、あるファイルをそのファイル自体にコピーしようとする
 と、

```

example% rcp tmp/file myhost:/tmp/file

```

ファイルが壊れてしまいます。

コピーの対象がディレクトリではなくファイルである場合、`rcp` が正常に失敗しないことがあります。

また、リモートホスト上の `$HOME/.profile` ファイルに指定されているコマンドからの出力によって、処理が正しく行われなかったこともあります。

`rcp` でサードパーティコピーを行うには、ソースホストがリモートホスト上でコマンドを実行することを許可されていなければなりません。

`rcp` では、シンボリックリンクの処理が正しく行われません。シンボリックリンクや名前付きパイプが含まれているディレクトリをリモートコピーする場合は、`tar` または `cpio` コマンドを `rsh` コマンドにパイプして使用してください。詳細については、[tar\(1\)](#) と [cpio\(1\)](#) を参照してください。

リモートホストを指定する際のメタキャラクタを引用符で囲まないと、解釈不可を示すエラーメッセージが出力されます。

ACL をサポートしていないファイルシステムに ACL をコピーすると、`rcp` は失敗します。

`rcp` は、ユーザー名、ホスト名、ドメインの処理を除いて CSI 対応が可能です。

どちらか一方のリモートマシンで `Solaris` を実行していない場合に、`rcp` を使用してサードパーティコピーを実行するとき、その終了コードは信頼できません。つまり、終了コードが正常終了を示していても、エラーが発生している可能性があります。あるいは、終了コードが異常終了を示していても、コピーは完全に成功している可能性があります。



名前 read – 標準入力から 1 行を入力

## 形式

```
/usr/bin/read /usr/bin/read [-r] var...
sh read name...
csh set variable = $<
ksh88 read [-prsu [n]] [name ? prompt] [name]...
ksh read [-ACprs] [-d delim] [-n nsize] [-N nsize] [-t timeout]
 [-u unit] [vname?prompt] [vname...]
```

## 機能説明

/usr/bin/read read ユーティリティーは、標準入力から 1 行を読み込みます。

デフォルトでは、`-r` オプションが指定されない限り、バックスラッシュ文字 (`\`) をエスケープ文字と見なします。標準入力端末で、呼び出し側のシェルが対話型のとき、`read` は次の条件が存在していればプロンプトを表示し、継続行の入力を要求します。

- `-r` オプションが指定されておらず、バックスラッシュで終わる行をシェルが読み込んだとき。
- 復帰改行文字の入力後、here-document (各種シェルで `<<` を使用する機能) が終了しないとき。

シェル内と同じように、入力行はいくつかのフィールドに分割されます。先頭のフィールドは最初に指定された変数 `var` に、次のフィールドは 2 番目の変数 `var` に、というように割り当てられます。フィールドの総数より `var` オペランドの数が少ない場合、余ったフィールドおよびそれらの区切り文字はすべて最後に指定された `var` に割り当てられます。フィールドより `var` の方が多い場合には、余った `var` には空の文字列が設定されます。

`var` オペランドで指定された変数の設定は、現在のシェルの実行環境に影響を及ぼします。次のように、サブシェルや別のユーティリティー実行環境で呼び出された場合は、

```
(read foo)
nohup read ...
find . -exec read ... \;
```

これは呼び出し側の環境のシェル変数には影響しません。

標準入力はテキストファイルでなければなりません。



- sh** 標準入力から1行を読み取り、内部フィールド区切り文字のIFS (通常はスペースまたはタブ) を用いてワード境界を区切り、最初のワードを最初の *name* に、2番目のワードを2番目の *name* に、というように順次割り当てます。残ったワードは最後の *name* に割り当てます。\`\` に続いて復帰改行を入力すれば、行を継続できます。復帰改行以外の文字の前にバックスラッシュを付加すれば、その文字を引用できます。このバックスラッシュは、ワードが *name* に割り当てられる前に削除され、バックスラッシュの後に位置する文字は解釈されません。ファイルの終わりに到達した場合を除き、戻り値は0となります。
- cs** 次の表記を見てください。
- ```
set variable = $<
```
- この記述は、標準入力の1行を *variable* 値としてロードします (**cs**(1) 参照)。
- ksh88** シェルの入力メカニズムです。1つの行を読み取り、IFS が示す文字を、区切り文字として使用して行の内容をいくつかのフィールドに分割します。エスケープ文字 (`\`) は、次の文字の特別な意味または行の継続に関する意味を取り除くために使用します。`-r` で指定する raw モードでは、`,` および `\` 文字が持つこの特殊な意味は無視されます。第1フィールドを1番目の *name* に、第2フィールドを2番目の *name* に、という順番で割り当てていき、余ったフィールドがあれば最後の *name* に割り当てます。`-p` オプションは、シェルが `|&` を使用して生成したプロセスの入力パイプから入力行を取り出します。`-s` フラグは、入力をコマンドとして履歴ファイルに保存します。`-u` フラグは、読み取り元となるファイル記述子番号を1桁の数値 *n* で指定します。ファイル記述子は、`exec` という特殊コマンドで開くことができます。*n* のデフォルト値は0です。*name* を省略すると、REPLY の値をデフォルトとして使用します。入力ファイルが読み込み用にオープンされていない場合とEOFに到達した場合を除き、終了ステータスは0です。`-p` オプションが指定されていてファイルの終わりを検出すると、このプロセスをクリアし別のプロセスを作成可能にします。最初の引数が?を含んでいると、シェルが対話型のとき、このワードの残りを標準エラーに対するプロンプトとして使用します。ファイルの終わりに到達しないかぎり、終了ステータスは0です。
- ksh** `read` は、標準入力から行を読み取り、IFS 変数値の文字を区切り文字として使用して複数のフィールドに分割します。`-r` オプションを指定しないかぎり、エスケープ文字 (`\`) は、次の文字の特別な意味または行の継続に関する意味を取り除くために使用します。
- フィールドより変数の方が多く場合は、残りの変数には空の文字列が設定されます。フィールドより変数の数が少ない場合、余ったフィールドおよびそれらの区切り文字は最後に指定された変数に割り当てられます。`var` が指定されていない場合、変数 REPLY が使用されます。
- `var` にバイナリ属性があり、`-n` または `-N` が指定されている場合、読み取られたバイト数は直接 `var` に格納されます。
- 1番目の `var` の後ろに `?prompt` を指定した場合、標準入力端末またはパイプのときに、`read` は標準エラーにプロンプトを表示します。

オプション

`/usr/bin/read`, `ksh88` `/usr/bin/read` および `ksh88` では次のオプションがサポートされています。

`-r` バックスラッシュ文字を特別な文字としません。単なる入力行の一部として扱います。

`ksh` `ksh` では次のオプションがサポートされています。

`-A` `var` の設定を解除して、インデックス `0` で始まる行に各フィールドを含むインデックス付き配列を作成します。

`-C` `var` の設定を解除して、`var` を複合変数として読み取ります。

`-d delim` 行の終わりではなく、区切り記号 `delim` まで読み取ります。

`-n nsize` 最大で `nsize` バイトを読み取ります。バイナリフィールドのサイズはバイト単位です。

`-N nsize` 正確に `nsize` バイトを読み取ります。バイナリフィールドのサイズはバイト単位です。

`-p` 標準入力の代わりに、現在の並行プロセスから読み取ります。ファイルの終わりに達すると、`read` は別の並行プロセスを作成できるように並行プロセスを切断します。

`-r` 特に入力行を処理するときに、`\` を処理しません。

`-s` 入力のコピーをシェル履歴ファイルのエントリとして保存します。

`-t timeout` 端末またはパイプから読み取るときに、`timeout` を秒単位で指定します。

`-u fd` 標準入力の代わりに、ファイル記述子番号 `fd` から読み取ります。デフォルト値は `0` です。

`-v` 端末から読み取るときに、1 番目の変数値を表示して、それをデフォルト値として使用します。

オペランド 次のオペランドを指定できます。

`var` 存在している、あるいはしていないシェル変数の名前。

使用例

例1 `read` コマンドを使用する方法

次に示す `/usr/bin/read` の例は、入力ファイルの内容を、各行の先頭フィールドを最後尾に移動して出力するものです。

```
example% while read -r xx yy
do
    printf "%s %s\n" "$yy" "$xx"
done < input_file
```

環境変数 read の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

- IFS** フィールドを区切るのに使われている、内部フィールド区切り文字を定義します。
- PS2** 対話型シェルが標準エラー出力に出力するプロンプトの文字列を定義します。プロンプトが出力されるのは、-r オプションが指定されておらずバックスラッシュで終わる行を読み込んだとき、または復帰改行文字の入力後に here-document が終了しないときです。

終了ステータス 次の終了ステータスが返されます。

- 0** 正常終了。
- >0** ファイルの終わりを検出した、またはエラーが発生しました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/read, csh,
ksh88, sh

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

ksh

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	不確実

関連項目 [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [line\(1\)](#), [set\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前 readonly - 変数の値の再設定による書き換えを防ぐシェル組み込み関数

形式

sh readonly [name]...

ksh88 **readonly [name [= value]]...

 **readonly -p

ksh ++readonly [-p] [name [= value]]...

機能説明

sh 指定された各 *name* に「読み取り専用」のマークを付け、これらの名前が後続の割り当てでは変更できないようにします。引数を省略すると、読み取り専用と指定された名前がすべて一覧表示されます。

ksh88 指定された *name* を読み取り専用にします。あとの代入でこれらの名前を変更できないようにします。

-p を指定したとき、readonly は、すべての読み取り専用変数の名前と値を、次の形式で標準出力に書き出します。

```
"readonly %s=%s\n", name, value
```

これは、*name* が設定されている場合です。

```
"readonly $s\n", name
```

これは、*name* が設定されていない場合です。

コマンドとしてシェルに再入力できる形式で書き込まれるため、同じ読み取り専用の属性と値をシェル実行環境で再現するときに便利です。保存しておいた出力をシェルに再入力するときには、次のような制限があります。

1. 出力時に値が設定されていた変数には、読み取り専用の属性は設定されません。
2. 出力時に設定されていなかった変数には、保存しておいた出力をシェルに再入力するときには、値は設定されません。

このマニュアルページでは、1 つまたは 2 つのアスタリスク (**) が先頭に付加されている [ksh88\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は = 符号のあとに実行され、ワード分割とファイル名生成は実行されません。

ksh

`readonly` は、名前で指定された各変数に読み取り専用の属性を設定します。これにより、値を変更できなくなります。`=value` が指定されている場合は、変数が読み取り専用になる前に変数名が `value` に設定されます。

名前を指定しない場合は、すべての読み取り専用変数の名前と値が標準出力に書き込まれます。

引数でフィールド分割やパス名展開が実行されないように、`readonly` は宣言コマンドとしてシェルに組み込まれています。チルド展開は値に対して行われます。

`-p` 現在の読み取り専用変数を再作成するために、シェルへの入力として使用可能な `readonly` コマンドの形式で出力されます。

このマニュアルページでは、1つまたは2つの `+` (プラス符号) が先頭に付加されている [ksh\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、`++` を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は `=` 符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

終了ステータス

ksh

次の終了ステータスが返されます。

- `0` 正常終了。
- `>0` エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [typeset\(1\)](#), [attributes\(5\)](#)

名前	refer – expand and insert references from a bibliographic database
形式	refer [-ben] [-ar] [-cstring] [-kx] [-lm,n] [-p filename] [-skeys] filename...
機能説明	refer is a preprocessor for nroff(1), or troff(1), that finds and formats references. The input files (standard input by default) are copied to the standard output, except for lines between ‘. [’ and ‘.]’ command lines. Such lines are assumed to contain keywords as for lookbib(1), and are replaced by information from a bibliographic data base. The user can avoid the search, override fields from it, or add new fields. The reference data, from whatever source, is assigned to a set of troff strings. Macro packages such as ms(5) print the finished reference text from these strings. A flag is placed in the text at the point of reference. By default, the references are indicated by numbers. When refer is used with eqn(1), neqn, or tbl(1), refer should be used first in the sequence, to minimize the volume of data passed through pipes.
オプション	-b Bare mode — do not put any flags in text (neither numbers or labels). -e Accumulate references instead of leaving the references where encountered, until a sequence of the form: <pre>. [\$LIST\$.]</pre> is encountered, and then write out all references collected so far. Collapse references to the same source. -n Do not search the default file. -ar Reverse the first <i>r</i> author names (Jones, J. A. instead of J. A. Jones). If <i>r</i> is omitted, all author names are reversed. -cstring Capitalize (with SMALL CAPS) the fields whose key-letters are in <i>string</i>. -kx Instead of numbering references, use labels as specified in a reference data line beginning with the characters %<i>x</i>; By default, <i>x</i> is L. -lm,n Instead of numbering references, use labels from the senior author's last name and the year of publication. Only the first <i>m</i> letters of the last name and the last <i>n</i> digits of the date are used. If either of <i>m</i> or <i>n</i> is omitted, the entire name or date, respectively, is used. -p filename Take the next argument as a file of references to be searched. The default file is searched last. -skeys Sort references by fields whose key-letters are in the <i>keys</i> string, and permute reference numbers in the text accordingly. Using this option implies the -e option. The key-letters in <i>keys</i> may be followed by a number indicating how many such fields are used, with a + sign taken as a very large number. The

default is AD, which sorts on the senior author and date. To sort on all authors and then the date, for instance, use the options '-sA+T'.

ファイル /usr/lib/refer directory of programs
 /usr/lib/refer/papers directory of default publication lists and indexes

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

関連項目 [addbib\(1\)](#), [eqn\(1\)](#), [indxbib\(1\)](#), [lookbib\(1\)](#), [nroff\(1\)](#), [roffbib\(1\)](#), [sortbib\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#)

名前	regcmp - 正規表現のコンパイル				
形式	regcmp [-] filename...				
機能説明	regcmp コマンドは、regcmp 関数と同様の機能を果たします。これにより C プログラムでの regcmp 関数の呼び出しがほとんど必要なくなります。regcmp 関数をバイパスすることで、実行時間とプログラムサイズの両方を節約できます。regcmp コマンドは、filename 内の正規表現をコンパイルし、その結果を filename.i に出力します。				
オプション	- オプションが使用されている場合、その結果は filename.c に出力されます。filename 内のエントリの形式は、名前 (C 変数)、その後ろに 1 つ以上の空白文字をあけて、二重引用符に囲まれた 1 つ以上の正規表現が続きます。regcmp の出力は、C ソースコードです。コンパイルされた正規表現は、extern char ベクトルとして表されます。したがって、filename.i ファイルが C プログラムに #include されることも、あるいは filename.c ファイルがコンパイルされ、その後ロードされることもあります。regcmp 出力を使用する C プログラムでは、regex(abc, line) は abc という名の正規表現を line に適用します。診断は、自明です。				
使用例	例1 regcmp command コマンドの使用例 <pre>name "([A-Za-z][A-Za-z0-9_]*)\$0" telno "\({0,1}([2-9][01][1-9])\$0\) {0,1} *" "([2-9][0-9]{2})\$1[-]{0,1}" "([0-9]{4})\$2"</pre> 上記の telno に対する 3 つの引数はすべて、1 行に入力しなければなりません。 regcmp 出力を使用する C プログラムでは、 <pre>regex(telno, line, area, exch, rest)</pre> は telno という名の正規表現を line に適用します。				
環境変数	LC_* 環境変数の使用法の一般的な説明は environ(5) を参照してください。 <table><tr><td>LC_CTYPE</td><td>regcmp が文字を処理する方法を決定します。LC_CTYPE に有効な値が設定されていると、regcmp は、そのロケールに合った文字を含むテキストやファイル名を表示および処理できます。</td></tr><tr><td>LC_MESSAGES</td><td>診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、肯定・否定の応答形式も決定します。"c" ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。</td></tr></table>	LC_CTYPE	regcmp が文字を処理する方法を決定します。LC_CTYPE に有効な値が設定されていると、regcmp は、そのロケールに合った文字を含むテキストやファイル名を表示および処理できます。	LC_MESSAGES	診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、肯定・否定の応答形式も決定します。"c" ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。
LC_CTYPE	regcmp が文字を処理する方法を決定します。LC_CTYPE に有効な値が設定されていると、regcmp は、そのロケールに合った文字を含むテキストやファイル名を表示および処理できます。				
LC_MESSAGES	診断メッセージや情報メッセージの表示方法を決定します。また、メッセージの言語とスタイル、肯定・否定の応答形式も決定します。"c" ロケールでは、メッセージはプログラム自身が使用しているデフォルトの形 (通常、米語) で表示されます。				

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities
CSI	有効

関連項目 [regcmp\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#)

名前 renice – alter priority of running processes

形式 renice [-n *increment*] [-i *idtype*] *ID*...
renice [-n *increment*] [-g | -p | -u] *ID*...
renice *priority* [-p] *pid*... [-g *gid*]... [-p *pid*]...
[-u *user*]...
renice *priority* -g *gid*... [-g *gid*]... [-p *pid*]...
[-u *user*]...
renice *priority* -u *user*... [-g *gid*]... [-p *pid*]...
[-u *user*]...

機能説明 The renice command alters the scheduling priority of one or more running processes. By default, the processes to be affected are specified by their process IDs.

If the first operand is a number within the valid range of priorities (−20 to 20), renice will treat it as a *priority* (as in all but the first synopsis form). Otherwise, renice will treat it as an *ID* (as in the first synopsis form).

Altering Process Priority Users other than the privileged user may only alter the priority of processes they own, and can only monotonically increase their “nice value” within the range 0 to 19. This prevents overriding administrative fiats. The privileged user may alter the priority of any process and set the priority to any value in the range −20 to 19. Useful priorities are: 19 (the affected processes will run only when nothing else in the system wants to); 0 (the “base” scheduling priority);, and any negative value (to make things go very fast). 20 is an acceptable nice value, but will be rounded down to 19.

オプション renice supports the following option features:

- The first operand, *priority*, must precede the options and can have the appearance of a multi-digit option.
- The -g, -p, and -u options can each take multiple option-arguments.
- The *pid* option-argument can be used without its -p option.
- The -i option can be used to specify the *ID* type for the ID list. This is preferred in specifying *ID* type over the use of the -g | -p | -u syntax, which is now obsolete. See NOTES.

The following options are supported:

- g Interprets all operands or just the *gid* arguments as unsigned decimal integer process group IDs.
- i This option, together with the *ID* list arguments, specifies a class of processes to which the renice command is to apply. The interpretation of the ID list depends on the value of *idtype*. The valid *idtype* arguments are: pid, pgid, uid, gid, sid, taskid, projid, and zoneid.

- n *increment* Specifies how the system scheduling priority of the specified process or processes is to be adjusted. The *increment* option-argument is a positive or negative decimal integer that will be used to modify the system scheduling priority of the specified process or processes. Positive *increment* values cause a lower system scheduling priority. Negative *increment* values may require appropriate privileges and will cause a higher system scheduling priority.
- p Interprets all operands or just the *pid* arguments as unsigned decimal integer process IDs. The -p option is the default if no options are specified.
- u Interprets all operands or just the *user* argument as users. If a user exists with a user name equal to the operand, then the user ID of that user will be used in further processing. Otherwise, if the operand represents an unsigned decimal integer, it will be used as the numeric user ID of the user.

オペランド

The following operands are supported:

- ID* A process ID, process group ID, or user name/user ID, depending on the option selected.
- priority* The value specified is taken as the actual system scheduling priority, rather than as an increment to the existing system scheduling priority. Specifying a scheduling priority higher than that of the existing process may require appropriate privileges.

使用例

例 1 Adjusting the scheduling priority of process IDs

Adjust the system scheduling priority so that process IDs 987 and 32 would have a lower scheduling priority:

```
example% renice -n 5 -p 987 32
```

例 2 Adjusting the scheduling priority of group IDs

Adjust the system scheduling priority so that group IDs 324 and 76 would have a higher scheduling priority, if the user has the appropriate privileges to do so:

```
example% renice -n -4 -g 324 76
```

例 3 Adjusting the scheduling priority of a user ID and user name

Adjust the system scheduling priority so that numeric user ID 8 and user sas would have a lower scheduling priority:

```
example% renice -n 4 -u 8 sas
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of renice: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

- 終了ステータス
- The following exit values are returned:

0 Successful completion.
>0 An error occurred.
- ファイル
- /etc/passwd map user names to user IDs
- 属性
- See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[nice\(1\)](#), [passwd\(1\)](#), [priocntl\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

The `renice` syntax

```
renice [-n increment] [-i idtype] ID ...
```

is preferred over the old syntax

```
renice [-n increment] [-g | -p | -u] ID ...
```

which is now obsolete.

If you make the priority very negative, then the process cannot be interrupted.

To regain control you must make the priority greater than 0.

Users other than the privileged user cannot increase scheduling priorities of their own processes, even if they were the ones that decreased the priorities in the first place.

The `priocntl` command subsumes the function of `renice`.

名前	rlogin – リモートログイン
形式	<code>rlogin [-8EL] [-ec] [-A] [-K] [-x] [-PN -PO] [-f -F] [-a] [-l username] [-k realm] hostname</code>
機能説明	rlogin ユーティリティーは、使用している端末から <i>hostname</i> で指定したリモートマシンへのリモートログインセッションを確立します。ユーザーは、Kerberos V5 を使用して rlogin セッションを Kerberos 認証するかどうかを選択できます。また、転送されるデータも保護できます。 ホスト名のリストは <i>hosts</i> データベースに格納されています。このデータベースは <i>/etc/hosts</i> ファイル、ネットワーク情報サービス (NIS) の <i>hosts</i> マップ、インターネットドメインネームサーバーのいずれかに含まれています。このうちのいくつかに含まれている場合もあります。各ホストには 1 つの正式名(データベースエントリの最初の名前)があり、さらにいくつかのニックネームが存在することもあります。<i>hostname</i> には、正式ホスト名とニックネームのどちらでも指定できます。 ユーザーは、Kerberos V5 認証を使用して、rlogin セッションをセキュリティ保護することもできます。また、転送されるデータも暗号化できます。rlogin セッションで Kerberos 認証を使用するには、任意の Kerberos 固有オプションを使用します (-A、-PN または -PO、-x、-f または -F、および -k realm)。これらのオプションのうち、-A、-x、-PN または -PO、および -f または -F は、krb5.conf(4) の [appdefaults] セクションにも指定できます。これらのオプションの使用法と期待される動作については、次の「オプション」セクションを参照してください。Kerberos 承認を使用する場合、アカウントの承認は、krb5_auth_rules(5) の規則によって制御されます。この認証が失敗した場合、-PO オプションをコマンド行に明示的に指定するか、krb5.conf(4) 経由で指定している場合にのみ、rhosts を使用する通常の rlogin へのフォールバックが発生します。-PN または -PO、-x、-f または -F、および -k realm オプションは、-A オプションの上位集合です。 リモートの端末タイプは、環境変数 TERM を使用して、ローカルの端末タイプと同じになります。端末またはウィンドウのサイズもリモートシステムにコピーされます(ただし、サーバーがこのオプションをサポートしている場合)。また、サイズの変更も、リモートシステムに反映されます。処理内容は常にリモート側にエコーされるので、遅延が発生している場合以外は、リモートログイン処理であることを意識せずに操作できます。また Control-S や Control-Q を使ったフロー制御、割り込み時の入出力のフラッシュも正しく処理されます。
オプション	サポートしているオプションは、次のとおりです。 -8 ネットワーク間で、7 ビットデータではなく 8 ビットデータを送信します。 -a リモートマシンがパスワードをたずねるようにします。具体的には、ヌルのローカルユーザー名を送信します。 -A このオプションは、Kerberos 認証を明示的に有効にし、アクセス制御において .k5login ファイルを信用します。サーバー側での

[in.rlogind\(1M\)](#) による承認チェックが成功して、なおかつ、`.k5login` ファイルがアクセス権を許可する場合、ユーザーはパスワードを入力せずにログインすることが許可されます。

- ec リモートホストとの接続を解除する際のエスケープ文字として、標準の文字ではなく `c` を使用します。
- E すべての文字が、エスケープ文字として認識されなくなります。
- f ローカルの証明書 (Kerberos チケット許可チケット) のコピーをリモートシステムに転送します。この証明書は再転送可能ではありません。このチケット許可チケットを転送する必要があるのは、リモートホスト上で Kerberos 認証を使用しているほかのネットワークサービスで自分自身を認証する必要がある場合です。たとえば、リモートホスト上にある自分のホームディレクトリが Kerberos V5 経由で NFS マウントされている場合などです。この場合、ローカルの証明書を転送しなければ、自分のホームディレクトリにアクセスできません。このオプションは、`-F` オプションと互いに排他的の関係にあります。
- F ローカルの証明書 (Kerberos チケット許可チケット) のコピーをリモートシステムに転送します。この証明書は再転送可能です。`-f` オプションは、`-f` オプションが提供する機能の上位集合を提供します。たとえば、`-f` オプションの場合、リモートホストに接続したあとに、`/usr/bin/ftp`、`/usr/bin/telnet`、`/usr/bin/rlogin`、または `/usr/bin/rsh` を `-f` オプションまたは `-F` オプションで呼び出そうとしても失敗します。つまり、単一のネットワーク署名をほかのシステムにプッシュできません。このオプションは、`-f` オプションとは相互排他的です。
- k *realm* このオプションにより、`rlogin` は、[krb5.conf\(4\)](#) で定義されているリモートホストの領域ではなく、指定した *realm* 内のリモートホスト用のチケットを取得します。
- K このオプションは、Kerberos 認証を明示的に無効にします。このオプションを使用すると、[krb5.conf\(4\)](#) の `autologin` 変数をオーバーライドできます。
- l *username* リモートログイン時のユーザー名として別の名前 (*username*) を指定します。このオプションを使用しない場合、使用されるリモートユーザー名はローカルユーザー名と同じになります。
- L `litout` モードでの `rlogin` セッションの実行を許可します。
- PN Kerberos の「rcmd」プロトコルの新しいバージョン (`-PN`) または古いバージョン (`-PO`) を明示的に要求します。新しいプロトコルは、古いプロトコルにおける多くのセキュリティ問題を回避し、はるか

に安全であると考えられますが、古い (MIT/SEAM) サーバーとは相互運用できません。これらのオプションを使うか、[krb5.conf\(4\)](#) を使って明示的に指定していない限り、デフォルトでは、新しいプロトコルが使用されます。古い「rcmd」プロトコルを使用しているときに Kerberos 承認が失敗した場合、Kerberos 承認を使用しない通常の rlogin へのフォールバックが発生します。これは、新しい、より安全な「rcmd」プロトコルを使用しているときには当てはまりません。

-x rlogin セッションを通過するすべてのデータに DES 暗号化を有効にします。これによって、応答時間が低下し、CPU 使用率が上昇します。

エスケープシーケンス 端末から入力した行がチルド (~) で始まっている場合、その行はエスケープシーケンスとみなされます。-e オプションを使って、エスケープ文字をチルド以外に変更することもできます。

~. リモートホストとの接続を切断します。ローカルホストはリモート側に警告を出さずに切断を実行します。この点で通常のログアウトとは異なります。

~susp ログインセッションを中断しますが、ジョブ制御を持つシェルを使用している場合のみです。susp は「中断用の」文字で、通常は CTRL-Z です。[tty\(1\)](#) を参照してください。

~dsusp ログインの入力側で中断しますが、出力は引き続き表示できます (ジョブ制御を持つシェルを使用する場合のみ)。dsusp は「一時中断用の」文字であり、通常は CTRL-Y です。[tty\(1\)](#) を参照してください。

オペランド *hostname* rlogin がリモートログインセッションを確立するリモートマシンです。

使用法 Kerberos 承認を使用する rlogin セッションの場合、各ユーザーは自分のホームディレクトリの `.k5login` ファイルに、専用の承認リストを持つことができます。このファイルの各行には、形式 `principal/instance@realm` の Kerberos 主体名が入っている必要があります。~/.k5login ファイルが存在する場合、起点ユーザーが ~/.k5login ファイルに指定された主体の 1 人であると認証された場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。それ以外の場合、`authenticated-principal-name` → `local-user-name` マッピング規則を使用して、起点ユーザーの認証された主体名をローカルアカウント名にマッピングできる場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。`.k5login` ファイル (アクセス制御用) が処理されるのは、Kerberos 認証が行われるときだけです。

セキュリティ保護されていない rlogin セッションの場合、各リモートマシンで `/etc/hosts.equiv` という名のファイルを使用できます。このファイルには、そのマシンとユーザー名を共有する、信頼できるホスト名のリストが含まれています。ローカルマシン上とリモートマシン上でのユーザー名が同一のユーザーは、リモートマシンの `/etc/hosts.equiv` ファイルにリストされているマシンから、パス

ワードを入力せずに `rlogin` を実行できます。個々のユーザーは、このような同等名リストを個人用の `.rhosts` ファイルとして、自身のホームディレクトリに作成できます。このファイル中の各行には2つの名前、ホスト名とユーザー名が含まれ、両者はスペースで区切られます。リモートユーザーの `.rhosts` ファイルの `username` に指定されているユーザーが `hostname` に指定されているホストにログインしていれば、そのユーザーはリモートユーザーとしてリモートマシンにパスワードなしでログインできます。ローカルホスト名がリモートマシン上の `/etc/hosts.equiv` ファイル中に見つからず、ローカルのユーザー名とホスト名がリモートユーザーの `.rhosts` ファイル中に見つからない場合、リモートマシンからパスワードを要求されます。`/etc/hosts.equiv` または `.rhosts` ファイルに記録されているホスト名は、`hosts` データベースに登録されている正式なホスト名である必要があります。この両ファイル中には、ニックネームは指定できません。

セキュリティ上の理由から、`.rhosts` ファイルの所有者はリモートユーザーまたは `root` でなければなりません。

ファイル	<code>/etc/passwd</code>	ユーザーアカウントに関する情報が含まれています。
	<code>/usr/hosts/*</code>	コマンドの <code>hostname</code> バージョンです。
	<code>/etc/hosts.equiv</code>	ユーザー名を共有する、信頼できるホスト名が記述されています。
	<code>/etc/nologin</code>	マシンのシャットダウン中にログインしようとするユーザーへのメッセージ。
	<code>\$HOME/.rhosts</code>	信頼できるホスト名とユーザー名の組み合わせのプライベートリスト。
	<code>\$HOME/.k5login</code>	アクセスを許可する Kerberos 主体を含むファイル。
	<code>/etc/krb5/krb5.conf</code>	Kerberos 構成ファイル。
	<code>/etc/hosts</code>	ホストデータベース。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/network/network-clients

関連項目 [rsh\(1\)](#), [stty\(1\)](#), [tty\(1\)](#), [in.rlogind\(1M\)](#), [hosts\(4\)](#), [hosts.equiv\(4\)](#), [krb5.conf\(4\)](#), [nologin\(4\)](#), [attributes\(5\)](#), [krb5_auth_rules\(5\)](#)

診断 次のメッセージは、マシンがシャットダウン中で、ログインできないことを表します。

NO LOGINS: System going down in N minutes

注意事項

`hosts.equiv` に登録されているシステムのセキュリティは、少なくともローカルシステムのセキュリティと同レベルである必要があります。セキュリティレベルの低いシステムが `hosts.equiv` 中に 1 つでも存在していると、システム全体のセキュリティが損なわれる可能性があります。

ネットワーク情報サービス (NIS) は、以前は *Sun Yellow Pages* (YP) として知られていました。これらの機能は同等です。名前が異なっているだけです。

この実装では TCP ネットワークサービス以外は使用できません。

名前 rm, rmdir – ディレクトリのエントリの削除

形式 /usr/bin/rm [-f] [-i] *file*...
/usr/bin/rm -rR [-f] [-i] *dirname*... [*file*]...
/usr/xpg4/bin/rm [-fiRr] *file*...
/usr/bin/rmdir [-ps] *dirname*...

機能説明

/usr/bin/rm
/usr/xpg4/bin/rm

rm ユーティリティーは、*file* オペランドで示された各ディレクトリエントリを削除します。*file* に書き込み権がなく、標準入力が端末の場合、そのファイルに対するすべてのアクセス権 (8 進数) が出力され、後ろに疑問符がつけられます。これは、確認を求めるプロンプトです。回答が肯定の場合、ファイルが削除されます。それ以外の場合、ファイルはそのままです。

file がシンボリックリンクの場合、リンクは削除されますが、参照するファイルまたはディレクトリは削除されません。ユーザーがシンボリックリンクを削除する際、そのディレクトリに書き込み権があれば、シンボリックリンクに書き込み権は必要ありません。

複数のファイルが指定され、何らかの理由でそのうちの 1 つが削除できなかった場合、rm は診断メッセージを標準エラー出力に書き出し、そのファイルにはそれ以上の処理は行わず、残りの *file* を処理します。

標準入力が端末でない場合、rm は -f オプションが指定されている場合と同様に動作します。

/usr/bin/rmdir

rmdir ユーティリティーは、*dirname* オペランドで示されたディレクトリのエントリを削除します。*dirname* は空のディレクトリを指している必要があります。

ディレクトリは指定された順序で処理されます。したがって、あるディレクトリとそのディレクトリのサブディレクトリを 1 度の rmdir 呼び出しで削除したい場合、サブディレクトリを親ディレクトリよりも前に指定して先に削除し、親ディレクトリの削除のときにディレクトリが空になるようにしてください。

オプション 次のオプションは、/usr/bin/rm と /usr/xpg4/bin/rm で使用できます。

-r 引数リストにあるディレクトリおよびサブディレクトリを再帰的に削除します。ディレクトリ内は空にされ、削除されます。通常、ディレクトリにある書き込み保護された任意のファイルの削除には、ユーザーに確認を求めるプロンプトが出されます。ただし、-f オプションが使用されている場合、または標準入力が端末ではなく、しかも -i オプションが使用されていない場合、書き込み保護されたファイルはプロンプトなしで削除されます。

このオプションが使用されたシンボリックリンクは、シンボリックリンク先には及びません。

書き込み保護された空でないディレクトリを削除しようとする、このコマンドは常に (-f オプションが使用されている場合でも) 失敗に終わり、エラーメッセージが表示されます。

-R -r オプションと同じです。

/usr/bin/rm

次のオプションは、/usr/bin/rm でのみ使用できます。

- f ユーザーへの指示なしにディレクトリ内のファイル(書き込み保護されているファイルも含む)を削除します。しかし、書き込み保護されたディレクトリでは、(たとえどんなアクセス権があっても) ファイルは決して削除されず、それについてのメッセージは表示されません。
- i 対話型。このオプションを使用すると、rm はファイルを削除する前に、確認を求めるプロンプトを出します。これは -f オプションより優先され、標準入力端末でない場合でもその効力を持続します。

/usr/xpg4/bin/rm

次のオプションは、/usr/xpg4/bin/rm のみでサポートされています。

- f 確認を求めるプロンプトメッセージを出力しません。存在しないものがオペランドで指定されても、診断メッセージを出力したり終了ステータスを変更したりしません。前に -i オプションが指定されていれば、それを無視します。
- i 確認を求めるプロンプトを出力します。-f オプションが指定されていれば、それを無視します。

/usr/bin/rmdir

次のオプションは、/usr/bin/rmdir でのみ使用できます。

- p ユーザーはディレクトリ *dirname* および空になる親ディレクトリを削除できます。パスの全体、または一部が削除されていなければ、メッセージが標準出力に出されます。
- s -p が有効なときに標準エラーに出力されるメッセージを表示しません。

オペランド

次のオペランドがサポートされています。

file 削除されるディレクトリエントリのパス名を指定します。

dirname 削除される空ディレクトリのパス名を指定します。

使用法

ファイルが 2G バイト (2³¹ バイト) 以上ある場合の rm と rmdir の動作については、[largefile\(5\)](#) を参照してください。

使用例

以下の例に、有効なコマンドの使用法を示します。

/usr/bin/rm、/usr/xpg4/bin/rm ディレクトリの削除

次のコマンドは、ディレクトリエントリ *a.out* および *core* を削除します。

```
example% rm a.out core
```

例2 確認を求めない、ディレクトリの削除

次のコマンドは、確認を求めずにディレクトリ `junk` とそのすべての内容を削除します。

```
example% rm -rf junk
```

/usr/bin/rmdir

例3 空のディレクトリの削除

現在のディレクトリに `a` というディレクトリがあり、そのディレクトリには `b` というディレクトリだけがあるとします。また `b` には、`c` というディレクトリだけがあるものとします。次のコマンドは、3つのディレクトリすべてを削除します。

```
example% rmdir -p a/b/c
```

環境変数

`rm` と `rmdir` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、および NLSPATH。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの `yesexpr` キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、`yesexpr` に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、`yesexpr` に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#) を参照してください。

終了ステータス

次の終了ステータスが返されます。

- 0 -f オプション省略時は、指定されたすべてのディレクトリエントリが削除されたことを示す。f オプション指定時は、指定されたすべての既存のディレクトリエントリが削除されたことを示す。
- >0 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/rm、/usr/bin/rmdir	属性タイプ	属性値
	使用条件	system/core-os
	CSI	有効

/usr/xpg4/bin/rm	属性タイプ	属性値
	使用条件	system/xopen/xcu4
	CSI	有効

属性タイプ	属性値
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [rmdir\(2\)](#), [unlink\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

診断
うっかりして次のようなことを行わないために、ファイル"."および“..”を削除することは禁じられています。

```
example% rm -r .*
```

ファイル「/」を削除することは禁止されています。これは、間違って以下のような状況になってしまったときに、惨劇を起こさないようにするためです。

```
example% rm -rf $x/$y
```

または

```
example% rm -rf /$y
```

そして、\$xと\$yが空の文字列に展開される場合です。

注意事項
-を使用することにより、ユーザーは任意のコマンド行オプションの最後を明白に記すことができ、rmは-で始まるファイル名の引数を認識できます。BSD 移行の補助として、rmは--を-の同義語として受け入れます。この導入補助は、将来のリリースでは提供されない可能性があります。--と-の両方が同一コマンド行にある場合は、2つめがファイルとして解釈されます。

名前 `rmformat` – 再書き込み可能な取り外し可能メディアのフォーマット

形式 `rmformat [-DeHUV] [-b label] [-c blockno] [-Fquick | long | force ] [-s filename] [devname]`

`rmformat -V read | write devname`

`rmformat -l [devname]`

機能説明 `rmformat` ユーティリティーは、再書き込み可能な取り外し可能媒体の、フォーマット、ラベル付け、パーティション分割、およびその他のさまざまな処理を行います。この再書き込み可能な取り外し可能媒体には、PCMCIA メモリーカードと ata カードがあります。`rmformat` ユーティリティーは、USB ハードドライブなど、あらゆる USB 大容量ストレージデバイスでも使用することをお勧めします。`rmformat` ユーティリティーは、検証、表面解析、不良セクターの修復(ドライブまたはドライブが不良ブロック管理をサポートしている場合)を行うためにも使用できます。

フォーマット後、`rmformat` はラベルを書き込みます。これにより、メディアの全容量が PCMCIA メモリーカード上で1つのスライスとして扱われます。パーティション情報は、`rmformat` のオプションを使用して変更することができます。

オプション サポートしているオプションは、次のとおりです。

-b label

メディアに SUNOS ラベルを作成します。SUNOS ポリウムラベル名は8文字以内にする必要があります。メディアのサイズが1T バイトを超える場合は、EFI ラベルが作成されます。媒体のサイズが1 TB を超える場合は、EFI ラベルが作成されます。DOS ポリウムラベルを書き込むには、`mkfs_pcfs(1M)` を使用してください。

-c blockno

指定されたブロックを修正して修復します。このオプションは、`rmformat` がサポートするすべてのデバイスに適用できるわけではありません。不良ブロック管理機能を持つドライブが装備されているデバイスと、このオプションがドライブによって実装されているデバイスとがあるためです。不良ブロック管理がサポートされているのがドライブまたはドライブのどちらであっても、不良ブロックは可能なかぎり修正されるように機能します。不良ブロックを修正できなかった場合は、修復に失敗したことを示すメッセージが表示されます。メッセージには、該当するブロック番号が10進数、8進数、16進数で示されます。

通常の PCMCIA メモリーおよび ata カードは、不良ブロック管理をサポートしていません。

-e

処理が終了した時にメディアを取り出します。ドライブが自動で取り出しを行う機能をサポートしていない場合、このオプションは利用できません。

-F quick | long | force

メディアをフォーマットします。

quick オプションは、検証をしないでフォーマットを開始するか、またはメディアの特定のトラックだけを検証してフォーマットを開始します。

long オプションは、完全フォーマットを開始します。ドライブ自体によってメディア全体の検証を行うデバイスもあります。

force オプションは、フォーマットを開始する前にユーザーによる確認を行わずに **long** オプションによるフォーマットを行います。

PCMCIA メモリーカードでは、前述のどのオプションを指定しても **long** オプションによるフォーマットを行います。

-l

すべてのリムーバブルデバイスのリストを表示します。デフォルト(オプションを指定しない場合)でも、**rmformat** はすべてのリムーバブルデバイスのリストを表示します。**dev_name** を指定した場合、**rmformat** は **dev_name** に関連するデバイスのリストを表示します。リストには、デバイスのパス名、ベンダー情報、およびデバイスタイプが含まれます。

-s filename

ユーザーが、SUNOS ラベル中のパーティション情報を配置することができるようにします。

ユーザーは、各スライスの情報(バイトオフセット、必要なサイズ、タグ、フラグ)を記述したファイルを入力として指定する必要があります。各スライスの情報を記述する書式は次のとおりです。

slices: *n* = *offset*, *size* [, *flags*, *tags*]

n はスライス番号です。*offset* はスライス *n* の開始位置で、バイトオフセットです。*size* はスライス *n* のサイズです。*offset* および *size* に指定する値は、512 バイトの倍数にする必要があります。これらの数値は、10進数、16進数、または8進数で指定することができます。浮動小数点数は使用できません。スライスの最大数についての詳細は、『[Oracle Solaris の管理: 一般的なタスク](#)』を参照してください。

size または *offset* を K バイト、M バイト、G バイト単位で指定するには、それぞれ数値のあとに KB、MB、GB を付けてください。これらの単位を表す接尾辞なしの数値は、バイトオフセットとみなされます。指定できるフラグは、次のとおりです。

```
wm = read-write, mountable
wu = read-write, unmountable
ru = read-only, unmountable
```

指定できるタグは、次のとおりです。

unassigned、boot、root、swap、usr、backup、stand、var、home、および alternates。

値を特定する必要がない場合は、タグとフラグの指定を省略することができます。ただし、省略する場合はタグとフラグの両方を省略し、指定する場合はタグとフラグの両方を指定する必要があります。特定のスライスに対するタグとフラグの値が指定されない場合、それぞれのデフォルト値が使用されます。フラグのデフォルト値はwm、タグのデフォルト値はunassignedです。

また、タグの名前は完全名または短縮名で指定できます。短縮名は完全名から最初の2文字以上を引用したものです。rmformatは、タグおよびフラグの短縮名を扱う時には大文字・小文字を区別しません。

スライスの指定は、次のように区切って記述します。

例:

```
slices: 0 = 0, 30MB, "wm", "home" :
        1 = 30MB, 51MB :
        2 = 0, 100MB, "wm", "backup" :
        6 = 81MB, 19MB
```

rmformatは、重複パーティションまたはメディアの容量を超える不正な要求を検出するために、必要性検査を行います。1つのスライス n について提供するスライス情報は1項目だけにする必要があります。同じスライス n について複数のスライス情報がある場合は、エラーメッセージが表示されます。スライス2は、ディスク全体を含むバックアップです。#は、入力ファイル中にコメント行を記述するために使用します。rmformatは、行頭にハッシュ記号(#)がある行は、その行の末尾までのすべての文字を無視します。

容量がごく小さいメディアにいくつかのパーティションを作成することはできませんが、そのようなデバイスに対してこのオプションを使用する場合には注意が必要です。

-U

すべてのファイルシステム上でumountを実行し、そのあとでフォーマットします。mount(1M)を参照してください。つまり、マウントされていたスライスをすべてマウント解除し、指定したデバイスに対してlongオプションによるフォーマットを実行します。

-V read | write

フォーマット後にメディアの各ブロックを検証します。書き込みの検証は、破壊的なメカニズム(書き込んだら消去するというメカニズム)で行われます。検証を開始する前に、ユーザーは検証を始めることを確定するように求められます。このオプションを使用すると、不良ブロックとして検出されたブロックの番号のリストが出力されます。

読み取りの検証は、ブロックを検証して、エラーが発生する傾向にあるブロックのリストが出力されます。

出力されるブロック番号のリストは、修復を行う時に `-c` オプションを使って利用することができます。

オペランド 次のオペランドを指定できます。

devname

devname には、絶対パス名または現在のディレクトリからの相対パス名でデバイスのパス名を指定するか、`cdrom` や `rmdisk` などのニックネームを指定します。

ボリューム管理が動作していないシステムでは、`/dev/rdisk/c?t?d?s?` のような絶対パス名で、または現在のディレクトリからの相対パス名で、デバイスを指定することができます。

使用例 例1 PCFS ファイルシステム用にリムーバブルメディアをフォーマットする

以下は、代替の `fdisk` パーティションを作成する例です。

```
example$ rmformat -F quick /dev/rdisk/c0t4d0s2:c
Formatting will erase all the data on disk.
Do you want to continue? (y/n)y
example$ su
# fdisk /dev/rdisk/c0t4d0s2:c
# mkfs -F pcfs /dev/rdisk/c0t4d0s2:c
Construct a new FAT file system on /dev/rdisk/c0t4d0s2:c: (y/n)? y
#
```

ファイル

`/dev/aliases`

別名を使用するボリューム管理によって制御されている異なるメディアに対して、文字型デバイスへのシンボリックリンクを提供するディレクトリ。

`/dev/dsk`

PCMCIA メモリーおよび `ata` カードやリムーバブルメディアのデバイスに対するブロック型デバイスアクセスを提供するディレクトリ。

`/dev/rdisk`

PCMCIA メモリーおよび `ata` カードやリムーバブルメディアのデバイスに対する文字型デバイスアクセスを提供するディレクトリ。

`/dev/aliases/pcmemS`

ソケット `S` 中の PCMCIA メモリーカードに対する文字型デバイスへのシンボリックリンク。`S` は、PCMCIA ソケット番号を表しています。

`/dev/aliases/rmdisk0`

CD-ROM、DVD-ROM、PCMCIA メモリーカード以外の一般的なリムーバブルメディアのデバイスへのシンボリックリンク。

`/dev/rdisk`
PCMCIA メモリーおよび ata カードやその他のリムーバブルデバイスに対する文字型デバイスアクセスを提供するディレクトリ。

`/dev/dsk`
PCMCIA メモリーおよび ata カードやその他のリムーバブルメディアのデバイスに対するブロック型デバイスアクセスを提供するディレクトリ。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/storage/media-volume-manageR

関連項目 [cpio\(1\)](#), [eject\(1\)](#), [tar\(1\)](#), [volcheck\(1\)](#), [volrmount\(1\)](#), [format\(1M\)](#), [mkfs_pcfs\(1M\)](#), [mount\(1M\)](#), [newfs\(1M\)](#), [prvtoc\(1M\)](#), [rmmount\(1M\)](#), [rpc.smserverd\(1M\)](#), [attributes\(5\)](#), [scsa2usb\(7D\)](#), [sd\(7D\)](#), [pcfs\(7FS\)](#), [udfs\(7FS\)](#)

『Oracle Solaris の管理: 一般的なタスク』

注意事項 SPARC ベースのシステム上で ([newfs\(1M\)](#) を使用して) 作成された `ufs` ファイルシステムが含まれている、再書き込み可能なメディア、PCMCIA メモリーカード、または PCMCIA ata カードと、x86 ベースのシステム上で作成された `ufs` ファイルシステムが含まれている再書き込み可能なメディアまたは PCMCIA メモリーカードとは、同じではありません。SPARC ベースのシステムと x86 ベースのシステムとの間で、`ufs` ファイルシステムを含む再書き込み可能な媒体を交換しないでください。SPARC ベースのシステムと x86 ベースのシステムとの間で、メモリーカード中のファイルを転送するには、[cpio\(1\)](#) または [tar\(1\)](#) を使用してください。相互に交換することが可能なファイルシステムについては、[pcfs\(7FS\)](#) および [udfs\(7FS\)](#) を参照してください。

仮想化環境では、`rmformat` がリムーバブルデバイスの一部を表示しないことがあります。

使用上の留意点 現在のところ、PCMCIA メモリーカード上の不良セクターの場所を検出する機能は、サポートされていません。このため、`rmformat` を実行してエラー (`bad sector`) が発生した場合、メモリーカードを使用できません。

名前	rmmount, rmumount – リムーバブルメディアのマウントおよびマウント解除
形式	<pre>rmmount [-u] [-o options] [nickname device] [mount_point] rmmount [-d] [-l] rmumount [nickname mount_point device] rmumount [-d] [-l]</pre>
機能説明	rmmount および rmumount ユーティリティーは、リムーバブルメディアやホットプラグによる取り付けが可能なボリュームをマウントおよびマウント解除します。任意指定の引数を使用すると、ボリュームラベル、マウントポイント、またはブロックデバイスのパスによってボリュームを識別できます。 デフォルトのマウントオプションをオーバーライドするための十分な特権がユーザーにある場合、rmmount に追加のマウントオプションを指定することもできます。 リムーバブルメディアのマウントを解除してもイジェクトはされません。必要に応じてメディアのマウントを解除してイジェクトするには、eject(1)を使用します。
オプション	rmmount と rmumount では、次のオプションがサポートされます。 -d デフォルトのデバイスのデバイスパスを表示します。引数が指定されていない場合は、このデバイスが使用されます。 -l マウント可能なデバイスのパスおよびニックネームを表示します。 次のオプションは、rmmount でのみサポートされます。 -o options マウントオプションを表示します。このオプションは、システムのデフォルトオプションをオーバーライドする特権を持つユーザーのみが使用できます。 -u マウントとは対照的に、ボリュームのマウントを解除します。
オペランド	次のオペランドがサポートされています。 <i>device</i> マウントまたはマウント解除の対象となるデバイスを /dev ディレクトリ中のデバイス名で指定します。 <i>mount_point</i> マウントまたはマウント解除の対象となるデバイスを /dev ディレクトリ中のデバイス名で指定します。 <i>nickname</i> マウントまたはマウント解除の対象となるデバイスを、このコマンドが認識できるニックネームで指定します。
使用例	例1 USB ディスクのマウント 次の例では、USB ディスクを PHOTOS というボリュームラベルでマウントします。 <pre>example% rmmount PHOTOS</pre>

例2 pcfs ボリュームのマウント解除
次の例では、pcfs ボリュームをデバイスパスによってマウント解除しています。
example% rmumount /dev/dsk/c4t0d0p0:1

終了ステータス 次の終了ステータスが返されます。
0 正常終了。
>0 エラーが発生した。
ファイル /media マウントのデフォルトのルート。
属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/storage/media-volume-manager
インタフェースの安定性	不確実

関連項目 [eject\(1\)](#), [attributes\(5\)](#)

名前	roffbib – format and print a bibliographic database
形式	roffbib [-e] [-h] [-m <i>filename</i>] [-np] [-olist] [-Q] [-ra <i>N</i>] [-s <i>N</i>] [-T <i>term</i>] [-V] [-x] [<i>filename</i>] ...
機能説明	roffbib prints out all records in a bibliographic database, in bibliography format rather than as footnotes or endnotes. Generally it is used in conjunction with sortbib(1) : example% sortbib database roffbib
オプション	roffbib accepts all options understood by nroff(1) except -i and -q. -e Produce equally-spaced words in adjusted lines using full terminal resolution. -h Use output tabs during horizontal spacing to speed output and reduce output character count. TAB settings are assumed to be every 8 nominal character widths. -m <i>filename</i> Prepend the macro file /usr/share/lib/tmac/tmac.name to the input files. There should be a space between the -m and the macro filename. This set of macros will replace the ones defined in /usr/share/lib/tmac/tmac.bib. -np Number first generated page <i>p</i>. -olist Print only page numbers that appear in the comma-separated <i>list</i> of numbers and ranges. A range <i>N–M</i> means pages <i>N</i> through <i>M</i>; an initial -<i>N</i> means from the beginning to page <i>N</i>; a final <i>N–</i> means from page <i>N</i> to end. -Q Queue output for the phototypesetter. Page offset is set to 1 inch. -ra<i>N</i> Set register <i>a</i> (one-character) to <i>N</i>. The command-line argument -r<i>N</i>1 will number the references starting at 1. Four command-line registers control formatting style of the bibliography, much like the number registers of ms(5). The flag -rv2 will double space the bibliography, while -rv1 will double space references but single space annotation paragraphs. The line length can be changed from the default 6.5 inches to 6 inches with the -rL6i argument, and the page offset can be set from the default of 0 to one inch by specifying -r01i (capital O, not zero). -s<i>N</i> Halt prior to every <i>N</i> pages for paper loading or changing (default <i>N</i> = 1). To resume, enter NEWLINE or RETURN. -T<i>term</i> Specify <i>term</i> as the terminal type. -V Send output to the Versatec. Page offset is set to 1 inch. -x If abstracts or comments are entered following the %X field key, roffbib will format them into paragraphs for an annotated bibliography. Several %X fields may be given if several annotation paragraphs are desired.

ファイル /usr/share/lib/tmac/tmac.bib file of macros used by nroff/troff

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

関連項目 [addbib\(1\)](#), [indxbib\(1\)](#), [lookbib\(1\)](#), [nroff\(1\)](#) [refer\(1\)](#), [sortbib\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#)

使用上の留意点 Users have to rewrite macros to create customized formats.

名前	roles - ユーザーに許可された役割を表示する
形式	roles [user]...
機能説明	roles コマンドは、自分または任意指定のユーザーに許可されている役割を標準出力に書き込みます。役割は、実際のユーザー (通常ユーザーと呼ばれる) ではなく、機能的な責任に対応する特殊なアカウントです。 各ユーザーはゼロ以上の役割を持つことができます。役割には通常ユーザーのほとんどの属性が含まれ、passwd(4) および shadow(4) では通常のユーザーと同様に識別されます。各役割には、user_attr(4) ファイルのエントリが必要です。このエントリによって役割として識別されます。役割には、独自の承認とプロファイルを指定できます。auths(1) および profiles(1) を参照してください。 役割には、プライマリ的なユーザーとしてシステムにログインすることが許可されません。ユーザーが自分自身としてログインしてから、役割を引き受ける必要があります。役割のアクションは通常ユーザーに起因します。監査が有効の場合、監査される役割のイベントには、役割を引き受けた元のユーザーの監査 ID が含まれます。 役割が役割自身またはほかの役割を引き受けることはできません。役割は階層型ではありません。ただし、権利プロファイル (prof_attr(4) を参照) は階層型で、階層型の役割と同じ効果を得るために使用できます。 役割の引き受けは、su(1M)、ssh(1)、または PAM_AUSER 変数をサポートするその他のサービスを使用して実行できます。引き受けが成功するには、役割の認証とメンバシップの両方が必要です。役割の認証では、その役割の user_attr(4) エントリにおける <code>roleauth</code> プロパティの設定に応じて、ユーザーのパスワードと役割のパスワードのどちらかを要求できます。デフォルトでは、役割のパスワードが要求されます。通常、役割にはプロファイルシェルが割り当てられます。慣例により、プロファイルシェルは、たとえば <code>pfbash</code> のように、標準のシェルの名前の前に <code>pf</code> を付けて指定されます。役割の割り当ては、user_attr(4) で指定されます。
使用例	例1 出力例 roles コマンドの出力の形式は次のとおりです。 <pre>example% roles tester01 tester02 tester01 : admin tester02 : secadmin, root example%</pre>
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 `auths(1)`, `pfexec(1)`, `profiles(1)`, `rlogin(1)`, `ssh(1)`, `su(1M)`, `auth_attr(4)`, `passwd(4)`, `prof_attr(4)`, `shadow(4)`, `user_attr(4)`, `attributes(5)`

名前
形式

```
rpcgen – an RPC protocol compiler

rpcgen infile

rpcgen [-a] [-A] [-b] [-C] [-D name [= value]] [-i size]
      [-I [-K seconds]] [-L] [-M] [-N] [-T] [-v]
      [-Y pathname] infile

rpcgen [-c | -h | -l | -m | -t | -Sc | -Ss | -Sm]
      [-o outfile] [infile]

rpcgen [-s nettype] [-o outfile] [infile]

rpcgen [-n netid] [-o outfile] [infile]
```

機能説明

The `rpcgen` utility is a tool that generates C code to implement an RPC protocol. The input to `rpcgen` is a language similar to C known as RPC Language (Remote Procedure Call Language).

The `rpcgen` utility is normally used as in the first synopsis where it takes an input file and generates three output files. If the *infile* is named `proto.x`, then `rpcgen` generates a header in `proto.h`, XDR routines in `proto_xdr.c`, server-side stubs in `proto_svc.c`, and client-side stubs in `proto_clnt.c`. With the `-T` option, it also generates the RPC dispatch table in `proto_tbl.i`.

`rpcgen` can also generate sample client and server files that can be customized to suit a particular application. The `-Sc`, `-Ss`, and `-Sm` options generate sample client, server and makefile, respectively. The `-a` option generates all files, including sample files. If the *infile* is `proto.x`, then the client side sample file is written to `proto_client.c`, the server side sample file to `proto_server.c` and the sample makefile to `makefile.proto`.

The server created can be started both by the port monitors (for example, `inetd`) or by itself. When it is started by a port monitor, it creates servers only for the transport for which the file descriptor 0 was passed. The name of the transport must be specified by setting up the environment variable `PM_TRANSPORT`. When the server generated by `rpcgen` is executed, it creates server handles for all the transports specified in the `NETPATH` environment variable, or if it is unset, it creates server handles for all the visible transports from the `/etc/netconfig` file. Note: the transports are chosen at run time and not at compile time. When the server is self-started, it backgrounds itself by default. A special define symbol `RPC_SVC_FG` can be used to run the server process in foreground.

The second synopsis provides special features which allow for the creation of more sophisticated RPC servers. These features include support for user-provided `#defines` and RPC dispatch tables. The entries in the RPC dispatch table contain:

- pointers to the service routine corresponding to that procedure
- a pointer to the input and output arguments
- the size of these routines

A server can use the dispatch table to check authorization and then to execute the service routine. A client library can use the dispatch table to deal with the details of storage management and XDR data conversion.

The other three synopses shown above are used when one does not want to generate all the output files, but only a particular one. See the EXAMPLES section below for examples of rpcgen usage. When rpcgen is executed with the `-s` option, it creates servers for that particular class of transports. When executed with the `-n` option, it creates a server for the transport specified by *netid*. If *infile* is not specified, rpcgen accepts the standard input.

All the options mentioned in the second synopsis can be used with the other three synopses, but the changes are made only to the specified output file.

The C preprocessor `cc -E` is run on the input file before it is actually interpreted by rpcgen. For each type of output file, rpcgen defines a special preprocessor symbol for use by the rpcgen programmer:

<code>RPC_HDR</code>	defined when compiling into headers
<code>RPC_XDR</code>	defined when compiling into XDR routines
<code>RPC_SVC</code>	defined when compiling into server-side stubs
<code>RPC_CLNT</code>	defined when compiling into client-side stubs
<code>RPC_TBL</code>	defined when compiling into RPC dispatch tables

Any line beginning with “`%`” is passed directly into the output file, uninterpreted by rpcgen, except that the leading “`%`” is stripped off. To specify the path name of the C preprocessor, use the `-Y` flag.

For every data type referred to in *infile*, rpcgen assumes that there exists a routine with the string `xdr_` prepended to the name of the data type. If this routine does not exist in the RPC/XDR library, it must be provided. Providing an undefined data type allows customization of XDR routines.

Server Error Reporting By default, errors detected by `proto_svc.c` is reported to standard error and/or the system log.

This behavior can be overridden by compiling the file with a definition of `RPC_MSGOUT`, for example, `-DRPC_MSGOUT=mymsgfunc`. The function specified is called to report errors. It must conform to the following `printf`-like signature:

```
extern void RPC_MSGOUT(const char *fmt, ...);
```

オプション

The following options are supported:

`-a` Generates all files, including sample files.

- A Enables the Automatic MT mode in the server main program. In this mode, the RPC library automatically creates threads to service client requests. This option generates multithread-safe stubs by implicitly turning on the -M option. Server multithreading modes and parameters can be set using the [rpc_control\(3NSL\)](#) call. rpcgen generated code does not change the default values for the Automatic MT mode.
- b Backward compatibility mode. Generates transport-specific RPC code for older versions of the operating system.
- c Compiles into XDR routines.
- C Generates header and stub files which can be used with ANSI C compilers. Headers generated with this flag can also be used with C++ programs.
- Dname[=value] Defines a symbol *name*. Equivalent to the `#define` directive in the source. If no *value* is given, *value* is defined as 1. This option can be specified more than once.
- h Compiles into C data-definitions (a header). The -T option can be used in conjunction to produce a header which supports RPC dispatch tables.
- i size Size at which to start generating inline code. This option is useful for optimization. The default *size* is 5.
- I Compiles support for [inetd\(1M\)](#) in the server side stubs. Such servers can be self-started or can be started by `inetd`. When the server is self-started, it backgrounds itself by default. A special define symbol `RPC_SVC_FG` can be used to run the server process in foreground, or the user can simply compile without the -I option.

If there are no pending client requests, the `inetd` servers exit after 120 seconds (default). The default can be changed with the -K option. All of the error messages for `inetd` servers are always logged with [syslog\(3C\)](#).

Note: This option is supported for backward compatibility only. It should always be used in conjunction with the -b option which generates backward compatibility code. By default (that is, when -b is not specified), rpcgen generates servers that can be invoked through portmonitors.

- K seconds By default, services created using rpcgen and invoked through port monitors wait 120 seconds after servicing a request before exiting. That interval can be changed using the -K flag. To create a server that exits immediately upon servicing a request, use -K 0. To create a server that never exits, the appropriate argument is -K -1.

When monitoring for a server, some portmonitors, *always* spawn a new process in response to a service request. If it is known that a server are used with such a monitor, the server should exit immediately on completion. For such servers, rpcgen should be used with `-K 0`.

- `-l` Compiles into client-side stubs.
- `-L` When the servers are started in foreground, uses [syslog\(3C\)](#) to log the server errors instead of printing them on the standard error.
- `-m` Compiles into server-side stubs, but do not generate a “main” routine. This option is useful for doing callback-routines and for users who need to write their own “main” routine to do initialization.
- `-M` Generates multithread-safe stubs for passing arguments and results between rpcgen-generated code and user written code. This option is useful for users who want to use threads in their code.
- `-N` This option allows procedures to have multiple arguments. It also uses the style of parameter passing that closely resembles C. So, when passing an argument to a remote procedure, you do not have to pass a pointer to the argument, but can pass the argument itself. This behavior is different from the old style of rpcgen-generated code. To maintain backward compatibility, this option is not the default.
- `-n netid` Compiles into server-side stubs for the transport specified by *netid*. There should be an entry for *netid* in the netconfig database. This option can be specified more than once, so as to compile a server that serves multiple transports.
- `-o outfile` Specifies the name of the output file. If none is specified, standard output is used (`-c`, `-h`, `-l`, `-m`, `-n`, `-s`, `-Sc`, `-Sm`, `-Ss`, and `-t` modes only).
- `-s nettype` Compiles into server-side stubs for all the transports belonging to the class *nettype*. The supported classes are *netpath*, *visible*, *circuit_n*, *circuit_v*, *datagram_n*, *datagram_v*, *tcp*, and *udp* (see [rpc\(3NSL\)](#) for the meanings associated with these classes). This option can be specified more than once. *Note:* The transports are chosen at run time and not at compile time.
- `-Sc` Generates sample client code that uses remote procedure calls.
- `-Sm` Generates a sample Makefile which can be used for compiling the application.
- `-Ss` Generates sample server code that uses remote procedure calls.
- `-t` Compiles into RPC dispatch table.
- `-T` Generates the code to support RPC dispatch tables.

The options -c, -h, -l, -m, -s, -Sc, -Sm, -Ss, and -t are used exclusively to generate a particular type of file, while the options -D and -T are global and can be used with the other options.

- v Displays the version number.
- Y *pathname* Gives the name of the directory where rpcgen starts looking for the C preprocessor.

オペランド The following operand is supported:

infile input file

- 使用例
- 例 1 Generating the output files and dispatch table
- The following entry
- ```
example% rpcgen -T prot.x
```
- generates all the five files: prot.h, prot\_clnt.c, prot\_svc.c, prot\_xdr.c, and prot\_tbl.i.
- 例 2 Sending headers to standard output
- The following example sends the C data-definitions (header) to the standard output:
- ```
example% rpcgen -h prot.x
```
- 例 3 Sending a test version
- To send the test version of the -DTEST, server side stubs for all the transport belonging to the class datagram_n to standard output, use:
- ```
example% rpcgen -s datagram_n -DTEST prot.x
```
- 例 4 Creating server side stubs
- To create the server side stubs for the transport indicated by *netid* tcp, use:
- ```
example% rpcgen -n tcp -o prot_svc.c prot.x
```

終了ステータス

0 Successful operation.

>0 An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities

関連項目

[inetd\(1M\)](#), [rpc\(3NSL\)](#), [rpc_control\(3NSL\)](#), [rpc_svc_calls\(3NSL\)](#), [syslog\(3C\)](#),
[netconfig\(4\)](#), [attributes\(5\)](#)

The rpcgen chapter in the 『[ONC+ Developer's Guide](#)』 manual.

名前 rpm2cpio – cpio アーカイブへの Red Hat Package (RPM) の変換

形式 rpm2cpio [*file.rpm*]

機能説明 rpm2cpio ユーティリティーは、単一の引数として指定された .rpm ファイルを cpio アーカイブに変換して、標準出力にコピーします。(「注意事項」を参照してください。)引数が指定されていない場合、rpm ストリームは標準入力から読み込まれます。いずれの場合も、標準出力が端末の場合、rpm2cpio は失敗し、使用法に関するメッセージが出力されます。そのため、通常、出力はファイルにリダイレクトするか、cpio(1) ユーティリティーを使用してパイプで連結します。

使用例

例1 rpm ファイルの変換

```
example% rpm2cpio Device3Dfx-1.1-2.src.rpm | cpio -itv
CPIO archive found!
-rw-r--r--  1 root  root  2635 Sep 13 16:39 1998, 3dfx.gif
-rw-r--r--  1 root  root 11339 Sep 27 16:03 1998, Dev3Dfx.tar.gz
-rw-r--r--  1 root  root  1387 Sep 27 16:04 1998, Device3Dfx-1.1-2.spec
31 blocks
```

例2 標準入力からの変換

```
example% rpm2cpio < Device3Dfx-1.1-2.src.rpm | cpio -itv
CPIO archive found!
-rw-r--r--  1 root  root  2635 Sep 13 16:39 1998, 3dfx.gif
-rw-r--r--  1 root  root 11339 Sep 27 16:03 1998, Dev3Dfx.tar.gz
-rw-r--r--  1 root  root  1387 Sep 27 16:04 1998, Device3Dfx-1.1-2.spec
31 blocks
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	package/rpm

関連項目 [cpio\(1\)](#), [attributes\(5\)](#)

注意事項 rpm2cpio は、バージョン 3 および 4 の RPM に対応しています。

名前 rsh, remsh, remote_shell – リモートシェル

形式 rsh [-n] [-a] [-K] [-PN | -PO] [-x] [-f | -F] [-l username]
[-k realm] hostname command

rsh hostname [-n] [-a] [-K] [-PN | -PO] [-x] [-f | -F]
[-l username] [-k realm] command

remsh [-n] [-a] [-K] [-PN | -PO] [-x] [-f | -F] [-l username]
[-k realm] hostname command

remsh hostname [-n] [-a] [-K] [-PN | -PO] [-x] [-f | -F]
[-l username] [-k realm] command

hostname [-n] [-a] [-PN | -PO] [-x] [-f | -F]
[-l username] [-k realm] command

機能説明

rsh は、*hostname* が示すホストに接続し、*command* が示すコマンドを実行します。rsh は自身の標準入力データをリモートコマンドにコピーし、リモートコマンドの標準出力を自身の標準出力にコピーし、さらに、リモートコマンドの標準エラー出力を自身の標準エラー出力にコピーします。割り込み、停止、および終了シグナルは、リモートコマンドに伝えられます。通常 rsh は、リモートコマンドが終了したときに終了します。

ユーザーは、Kerberos V5 認証を使用して、rsh セッションを安全にするかどうかを選択できます。また、転送されるデータも暗号化できます。rsh セッションで Kerberos 認証を使用するには、任意の Kerberos 固有オプションを使用します (-a、-PN または -PO、-x、-f または -F、および -k realm)。これらのオプションのうち、-a、-x、-PN または -PO、および -f または -F は、[krb5.conf\(4\)](#) の [appdefaults] セクションにも指定できます。これらのオプションの使用法と期待される動作については、次の「オプション」セクションを参照してください。Kerberos 承認を使用する場合、アカウントの承認は、[krb5_auth_rules\(5\)](#) の規則によって制御されます。Kerberos 承認が失敗した場合、-PO オプションをコマンド行に明示的に指定するか、[krb5.conf\(4\)](#) 経由で指定している場合にのみ、rhosts を使用する通常の rsh へのフォールバックが発生します。-PN または -PO、-x、-f または -F、および -k realm オプションは、-a オプションの上位集合であることに注意してください。

command を省略すると、rsh は単一のコマンドを実行する代わりに、[rlogin\(1\)](#) を使ってそのユーザーをリモートホストにログインします。

rsh は *command* の終了ステータスを返しません。

シェルのメタキャラクタのうち、引用符で囲まれていないものはローカルマシン上で解釈されます。引用符で囲まれているものは、リモートマシン上で解釈されます。「使用例」を参照してください。

特定のユーザーのログインシェルの初期化ファイル (.cshrc など) でロケールが設定されていない場合、rsh はコマンドを実行するときに、常に C ロケールを使用します。リモートマシンのデフォルトのロケールは使用しません。

このコマンド自身は暗号化されずにリモートシステムに送信されます。それ以降の転送はすべて暗号化されます。-x を参照してください。

オプション

サポートしているオプションは、次のとおりです。

- a このオプションは、Kerberos 認証を明示的に有効にし、アクセス制御において `.k5login` ファイルを信用します。サーバー側での `in.rshd(1M)` による承認チェックが成功し、`.k5login` ファイルがアクセス権を許可する場合、ユーザーは `rsh` コマンドを実行することが許可されます。
- f ローカルの証明書 (Kerberos チケット許可チケット) のコピーをリモートシステムに転送します。この証明書は再転送可能ではありません。このチケット許可チケットを転送する必要があるのは、リモートホスト上で Kerberos 認証を使用しているほかのネットワークサービスで自分自身を認証する必要がある場合です。たとえば、リモートホスト上にある自分のホームディレクトリが Kerberos V5 経由で NFS マウントされている場合などです。この場合、ローカルの証明書を転送しなければ、自分のホームディレクトリにアクセスできません。このオプションは、-F オプションと互いに排他的にあります。
- F ローカルの証明書 (Kerberos チケット許可チケット) のコピーをリモートシステムに転送します。この証明書は再転送可能です。-F オプションは、-f オプションが提供する機能の上位集合を提供します。たとえば、-f オプションの場合、リモートホストに接続した後に、`/usr/bin/ftp`、`/usr/bin/telnet`、`/usr/bin/rlogin`、または `/usr/bin/rsh` を -f オプションまたは -F オプションで呼び出そうとしても失敗します。つまり、単一のネットワーク署名をほかのシステムにプッシュできません。このオプションは、-f オプションとは相互排他的です。
- k *realm* `rsh` は、`krb5.conf(4)` で定義されているリモートホストの領域ではなく、指定した *realm* 内のリモートホスト用のチケットを取得します。
- K このオプションは、Kerberos 認証を明示的に無効にします。このオプションを使用すると、`krb5.conf(4)` の `autologin` 変数をオーバーライドできます。
- l *username* リモートユーザー名として、ユーザー自身のローカルユーザー名の代わりに *username* を使用します。このオプションを省略すると、リモートユーザー名はローカルユーザー名と同じになります。
- n `rsh` の入力先を `/dev/null` に変更します。このオプションは、`rsh` とそれを呼び出したシェルとの間での、予期できない干渉を防ぐ上で便利です。たとえば、`rsh` を実行していて、さらにバックグラウン

ドで `rsh` を呼び出した場合、その入力先を端末以外に変更しないと、リモートコマンドの読み取りがない場合でも処理が妨げられてしまいます。`-n` オプションを指定すれば、このような事態は避けられます。

`-PO`

`-PN`

Kerberos の「rcmd」プロトコルの新しいバージョン (`-PN`) または古いバージョン (`-PO`) を明示的に要求します。新しいプロトコルは、古いプロトコルで蔓延している多くのセキュリティの問題を回避し、より安全であると考えられますが、古い (MIT/SEAM) サービスとは相互運用できません。`PO` オプションで古いプロトコルを明示的に指定するか、[krb5.conf\(4\)](#) 経由で指定していないかぎり、デフォルトでは、新しいプロトコルが使用されます。古い「rcmd」プロトコルを使用しているときに Kerberos 承認が失敗した場合、Kerberos 承認を使用しない通常の `rsh` へのフォールバックが発生します。これは、新しい、より安全な「rcmd」プロトコルを使用している場合には当てはまりません。

`-x`

ネットワークセッションのトラフィックを暗号化します。「機能説明」を参照してください。

リモートシェルの種類 (`sh` や `rsh` など) は、リモートシステム上の `/etc/passwd` ファイル中のユーザーのエントリにより決められます。

オペランド

次のオペランドを指定できます。

command 指定された *hostname* で実行するコマンド

使用法

ファイルが 2G バイト (2^{31} バイト) 以上ある場合の `rsh` と `remsh` の動作については、[largefile\(5\)](#) を参照してください。

`rsh` および `remsh` コマンドは、IPv6 に対応しています。[ip6\(7P\)](#) を参照してください。現在のところ、IPv6 は Kerberos V5 認証ではサポートされません。

ホスト名は *hosts* データベース (`/etc/hosts` ファイルに含めることができる)、インターネットドメイン名データベース、あるいはその両方に書かれています。各ホストには 1 つの正式名 (データベースエントリの最初の名前) があり、さらにいくつかのニックネームが存在することもあります。*hostname* 引数には、正式なホスト名とニックネームのいずれかを指定します。

`rsh` を実行するファイル名が `rsh` でなければ、`rsh` はそのファイル名を *hostname* 引数として使用します。これによりユーザーは、ホスト名で `rsh` に対するシンボリックリンクを生成でき、実行時にそのホスト上のリモートシェルを呼び出せます。ディレクトリを作成し、それを共通に使われるホスト名でシンボリックリンクにつなぎ、さらにそのディレクトリをユーザー自身のシェルの検索パスに含めることにより、シェルに *hostname* を入力するだけで `rsh` を実行することができます。

rsh を remsh で呼び出した場合、rsh は `/usr/bin/remsh` の存在を確認します。このファイルが存在する場合、rsh は remsh を rsh の別名として処理します。`/usr/bin/remsh` が存在しない場合、rsh は remsh をホスト名として処理します。

Kerberos 承認を使用する rsh セッションの場合、各ユーザーは自分のホームディレクトリの `.k5login` ファイルに、専用の承認リストを持つことができます。このファイルの各行には、形式 `principal/instance@realm` の Kerberos 主体名が入っている必要があります。`~/.k5login` ファイルが存在する場合、起点ユーザーが `~/.k5login` ファイルに指定された主体の 1 人であると認証された場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。それ以外の場合、`authenticated-principal-name` → `local-user-name` マッピング規則を使用して、起点ユーザーの認証された主体名をローカルアカウント名にマッピングできる場合にのみ、起点ユーザーのアカウントにアクセス権が付与されます。`.k5login` ファイル (アクセス制御用) が処理されるのは、Kerberos 認証が行われるときだけです。

セキュリティ保護されていない rsh セッションの場合、各リモートマシンで `/etc/hosts.equiv` という名のファイルを使用できます。このファイルには、そのマシンとユーザー名を共有する、信頼できるホスト名のリストが含まれています。ローカルマシン上とリモートマシン上でのユーザー名が同一のユーザーは、リモートマシンの `/etc/hosts.equiv` ファイルにリストされているマシンから rsh を実行することができます。個々のユーザーは、このような同等名リストを個人用の `.rhosts` ファイルとして、自身のホームディレクトリに作成することができます。このファイル中の各行には 2 つの名前、`hostname` と `username` が含まれ、両者はスペースで区切られます。`username` で示すユーザーが `hostname` で示すホストにログインしていれば、そのユーザーは rsh を使って、リモートユーザーとしてリモートマシンにアクセスできます。ローカルホスト名がリモートマシン上の `/etc/hosts.equiv` ファイル中に見つからず、ローカルのユーザー名とホスト名がリモートユーザーの `.rhosts` ファイル中に見つからない場合、アクセスは拒否されます。`/etc/hosts.equiv` または `.rhosts` ファイルに記録されているホスト名は、hosts データベースに登録されている正式なホスト名である必要があります。つまりこの両ファイル中には、ニックネームは指定できません。

ユーザーアカウントがロックされていると、rsh を使用して、ホスト名からのユーザーとしてログインすることはできません。

リモートマシン上でアクセスが拒否されたときは、`command` 引数が省略されていない限り、rsh はパスワードの入力を要求するプロンプトを出力しません。

使用例

例 1 rsh でファイルを追加する

次のコマンドは、リモートのマシン `lizard` 上にあるファイル `lizard.file` を、ローカルのマシン `example` 上にあるファイル `example.file` に追加します。

```
example% rsh lizard cat lizard.file >> example.file
```

例1 rshでファイルを追加する (続き)

次のコマンドは、リモートのマシン `lizard` 上にあるファイル `lizard.file` を、同じマシン `lizard` 上にあるファイル `lizard.file2` に追加します。

```
example% rsh lizard cat lizard.file ">>" lizard.file2
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。

ファイル	<code>/etc/hosts</code>	インターネットホストテーブル
	<code>/etc/hosts.equiv</code>	信頼性のあるリモートホストとユーザー
	<code>/etc/passwd</code>	システムパスワードファイル
	<code>\$HOME/.k5login</code>	アクセスを許可する Kerberos 主体を含むファイル
	<code>/etc/krb5/krb5.conf</code>	Kerberos 構成ファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	<code>service/network/network-clients</code>
CSI	有効

関連項目 [on\(1\)](#), [rlogin\(1\)](#), [ssh\(1\)](#), [telnet\(1\)](#), [vi\(1\)](#), [in.rshd\(1M\)](#), [hosts\(4\)](#), [hosts.equiv\(4\)](#), [krb5.conf\(4\)](#), [attributes\(5\)](#), [krb5_auth_rules\(5\)](#), [largefile\(5\)](#), [ip6\(7P\)](#)

注意事項 `hosts.equiv` に登録されているシステムのセキュリティは、少なくともローカルシステムのセキュリティと同レベルである必要があります。セキュリティレベルの低いシステムが `hosts.equiv` 中に1つでも存在していると、システム全体のセキュリティが損なわれる可能性があります。

[vi\(1\)](#) のような対話型コマンドは実行できません。対話型コマンドは `rlogin` を使って実行してください。

停止シグナルは、ローカルの `rsh` プロセスだけを停止させます。これはバグだという議論があるかもしれませんが、現在のところ修正が大変困難です。その理由は複雑なので、ここでは説明を省きます。

現在のローカル環境は、リモートシェルに引き渡されません。

`-n` オプションを使用すると、原因を特定できないような問題が発生することがあります。次に例を示します。

```
example% rsh somehost dd if=/dev/nrmt0 bs=20b | tar xvpBf -
```

このコマンドを実行すると、シェルの状態がおかしくなります。つまり `rsh` の前に `tar` が終了してしまい、次に `rsh` コマンドが「壊れたパイプ」に書き込みを試みると、正常終了せずに、標準入力をめぐってシェルと競合します。`-n` オプションを指定して `rsh` を実行すれば、このような事態を防ぐことができます。

このバグは `rsh` がパイプラインの先頭で、標準入力を読み込まないときにだけ発生します。`rsh` が標準入力を読み込む必要があるときは、`-n` オプションを使用しないでください。例:

```
example% tar cf - . | rsh sundial dd of=/dev/rmt0 obs=20b
```

この場合、前述のような現象は起こりません。この場合に `-n` オプションを指定すると、`rsh` はパイプから読み込む代わりに、誤って `/dev/null` からの読み込みを試みます。

ほとんどの場合、[ssh\(1\)](#) は `rsh` よりも推奨されています。

名前	runat – 拡張属性名前空間でのコマンドの実行
形式	<code>/usr/bin/runat file [command]</code>
機能説明	runat ユーティリティーは、ファイルの非表示の属性ディレクトリにあるシェルコマンドを実行するために使用します。実際にはこのユーティリティーは、file 引数に関連付けられている非表示の属性ディレクトリに現在の作業ディレクトリを変更し、指定されたコマンドを Bourne シェル (/bin/sh) で実行します。コマンド引数が指定されていない場合は、対話型シェルが起動されます。環境変数 \$SHELL は、起動されるシェルの定義します。この変数が未定義の場合は、デフォルトのシェル /bin/sh が使用されます。 file 引数には任意のファイルを指定できます。これには、拡張属性をサポートするディレクトリが含まれます。runat コマンドを呼び出す前に、このファイルに属性を設定したり、何らかの方法で準備したりする必要はありません。
オペランド	次のオペランドがサポートされています。 <i>file</i> 任意のファイル。これには、拡張属性をサポートするディレクトリが含まれます。 <i>command</i> 属性ディレクトリで実行するコマンド。
エラー	runat が <i>file</i> 引数にアクセスできない場合、または <i>file</i> 引数が拡張属性をサポートしない場合は、ゼロ以外の終了ステータスが返されます。
使用法	拡張ファイル属性の詳細については、fsattr(5) を参照してください。 runat コマンドによって作成されたプロセスのコンテキストでは、ファイルの拡張属性を含む非表示のディレクトリが、現在の作業ディレクトリとして設定されます。このディレクトリの親(「..」エントリ)は、コマンド行で指定されたファイルを常に指しています。そのため、これはディレクトリではない可能性があります。したがって、親エントリが正しい形式であることに依存する(つまり、ディレクトリを参照する)コマンド(pwd など)は失敗する可能性があります。 <i>command</i> 引数が指定されていない場合、runat は、指定されたファイルの非表示の属性ディレクトリを現在の作業ディレクトリに設定して、新しい対話型シェルの起動します。前述のようにディレクトリの親がディレクトリではない場合は、いくつかのシェル(zsh、tcsh など)が正常に動作しない場合があります。これらのシェルは、runat とともに使用しないでください。
使用例	例1 runat を使用したファイルの拡張属性の一覧表示 <pre>example% runat file.1 ls -l example% runat file.1 ls</pre>

例2 拡張属性の作成

```
example% runat file.2 cp /tmp/attrdata attr.1
example% runat file.2 cat /tmp/attrdata > attr.1
```

例3 あるファイルから別のファイルへの属性のコピー

```
example% runat file.2 cat attr.1 | runat file.1 "cat > attr.1"
```

例4 runatを使用した対話型シェルの起動

```
example% runat file.3 /bin/sh
```

これは、file.3の属性ディレクトリで新しいシェルを起動します。このシェルは現在のディレクトリを判別できません。属性ディレクトリから出るには、起動されたシェルを終了するか、絶対パスを使用してディレクトリを変更します(cd)。

属性の基本的な操作を行う際に推奨される方法は、次のとおりです。

display	runat file ls [<i>options</i>]
read	runat file cat <i>attribute</i>
作成/変更	runat file cp <i>absolute-file-path attribute</i>
delete	runat file rm <i>attribute</i>
アクセス権の変更	runat file chmod <i>mode attribute</i> runat file chgrp <i>group attribute</i> runat file chown <i>owner attribute</i>

対話型シェル

```
runat file /bin/sh
または $SHELL に /bin/sh を設定して、
runat file
```

上記のリストには、runat とともに動作することがわかっているコマンドが含まれています。その他の多くのコマンドが動作する可能性はありますが、このリスト以外のコマンドの動作は保証されていません。現在の作業ディレクトリを判別できることに依存するコマンドは、失敗する可能性があります。そのようなコマンドの例を次に示します。

例5 属性ディレクトリでの man コマンドの使用

```
example% runat file.1 man runat
>getcwd: Not a directory
```


例6 属性ディレクトリでの tcsh シェルの起動

```
example% runat file.3 /usr/bin/tcsh
tcsh: Not a directory
tcsh: Trying to start from "/home/user"
```

ユーザーのホームディレクトリを現在の作業ディレクトリに設定して、新しい tcsh シェルが起動されました。

例7 属性ディレクトリでの zsh シェルの起動

```
example% runat file.3 /usr/bin/zsh
example%
```

コマンドは実行されたように見えますが、実際には zsh は現在の作業ディレクトリを「/」に変更しただけです。これは、/bin/pwd を使用して表示できます。

```
example% /bin/pwd
/
```

環境変数 SHELL runat によって呼び出されるコマンドシェルを指定します。

終了ステータス 次の終了ステータスが返されます。

- 125 file 引数によって参照されるファイルの属性ディレクトリにアクセスできない。
- 126 指定された command 引数の実行が失敗しました。

それ以外の場合は、指定されたコマンドを実行するために呼び出されたシェルの終了ステータスが返されます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

関連項目 [open\(2\)](#), [attributes\(5\)](#), [fsattr\(5\)](#)

注意事項 現在の作業ディレクトリを判別できない場合は、runat でコマンドが失敗する原因を特定できないことがあります。発生したエラーは、区別するのが困難であり、さまざまな解釈ができます (tcsh と zsh の例を参照)。

名前	rup – show host status of remote machines (RPC version)				
形式	rup [-hlt] rup [<i>host</i>]...				
機能説明	rup gives a status similar to <code>uptime</code> for remote machines. It broadcasts on the local network, and displays the responses it receives. Normally, the listing is in the order that responses are received, but this order can be changed by specifying one of the options listed below. When <i>host</i> arguments are given, rather than broadcasting rup will only query the list of specified hosts. A remote host will only respond if it is running the <code>rstatd</code> daemon, which is normally started up from inetd(1M). In the absence of a name service, such as LDAP or NIS, rup displays host names as numeric IP addresses.				
オプション	-h Sort the display alphabetically by host name. -l Sort the display by load average. -t Sort the display by up time.				
属性	See attributes(5) for descriptions of the following attributes:				
	<table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>service/network/network-clients</td></tr> </tbody> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	service/network/network-clients
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	service/network/network-clients				
関連項目	ruptime(1) , inetd(1M) , attributes(5) 『Oracle Solaris 11.1 システムのインストール』				
使用上の留意点	Broadcasting does not work through gateways.				

名前 `rup` – show host status of remote machines (RPC version)

```
形式      rup [-hlt]
          rup [host]...
```

機能説明	rup gives a status similar to uptime for remote machines. It broadcasts on the local network, and displays the responses it receives.

Normally, the listing is in the order that responses are received, but this order can be changed by specifying one of the options listed below.

When *host* arguments are given, rather than broadcasting *rup* only queries the list of specified hosts.

A remote host will only respond if it is running the `rsstatd` daemon, which is normally started up from `inetd(1M)`.

オプション	-h	Sort the display alphabetically by host name.
	-l	Sort the display by load average.
	-t	Sort the display by up time.

属性 See `attributes(5)` for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/extended-system-utilities

関連項目 [ruptime\(1\)](#), [inetd\(1M\)](#), [attributes\(5\)](#)

使用上の留意点 Broadcasting does not work through gateways.

名前	runtime – show host status of local machines
形式	runtime [-ar] [-l -t -u]
機能説明	The runtime utility gives a status line like uptime (see uptime(1)) for each machine on the local network; these are formed from packets broadcast by each host on the network approximately every three minutes. Machines for which no status report has been received for 11 minutes are shown as being down. Normally, the listing is sorted by host name, but this order can be changed by specifying one of the options listed below.
オプション	The following options are supported: -a Counts even those users who have been idle for an hour or more. -r Reverses the sorting order. -l -t -u These options are mutually exclusive. The use of one overrides the previous one(s). -l Sorts the display by load average. -t Sorts the display by up time. -u Sorts the display by number of users.
ファイル	/var/spool/rwho/whod.* data files
属性	See attributes(5) for descriptions of the following attributes:
関連項目	uptime(1) , rwho(1) , in.rwhod(1M) , attributes(5)

名前 rusage – print resource usage for a command

形式 /usr/ucb/rusage *command*

機能説明 The rusage command is similar to [time\(1\)](#). It runs the given *command*, which must be specified; that is, *command* is not optional as it is in the C shell's timing facility. When the command is complete, rusage displays the real (wall clock), the system CPU, and the user CPU times which elapsed during execution of the command, plus other fields in the rusage structure, all on one long line. Times are reported in seconds and hundredths of a second.

使用例 例1 The format of rusage output

The example below shows the format of rusage output.

```
example% rusage wc /usr/share/man/man1/csh (1)
3045 13423 78071 /usr/share/man/man1/csh (1)
2.26 real 0.80 user 0.36 sys 11 pf 38 pr 0 sw 11 rb 0 wb 16 vcx 37
      icx 24 mx 0 ix 1230 id 9 is
example%
```

Each of the fields identified corresponds to an element of the rusage structure, as described in [getrusage\(3C\)](#), as follows:

real		elapsed real time
user	ru_utime	user time used
sys	ru_stime	system time used
pf	ru_majflt	page faults requiring physical I/O
pr	ru_minflt	page faults not requiring physical I/O
sw	ru_nswap	swaps
rb	ru_inblock	block input operations
wb	ru_oublock	block output operations
vcx	ru_nvcsw	voluntary context switches
icx	ru_nivcsw	involuntary context switches
mx	ru_maxrss	maximum resident set size
ix	ru_ixrss	currently 0
id	ru_idrss	integral resident set size
is	ru_isrss	currently 0

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目

[csh\(1\)](#), [time\(1\)](#), [getrusage\(3C\)](#), [attributes\(5\)](#)

使用上の留意点

When the command being timed is interrupted, the timing values displayed may be inaccurate.

名前

rusers - リモートマシン上にログインしているユーザーの表示

形式

rusers [-ahilu] host...

機能説明

rusers コマンドは、[who\(1\)](#) と似た形式の出力をリモートマシン用に作成します。通常この出力では、応答を得た順序でユーザー名がリストされますが、後述するオプションを使って順序を変更することができます。

デフォルトではログインしているユーザーの名前だけが出力されます。`-l` オプションを指定すると、それに加えて以下の情報が各ユーザーに対して出力されます。

```
userid hostname:terminal login_date login_time idle_time login_host
```

`hostname` (ホスト名) と `login host` (ログインホスト) が同じ値の場合には、`login_host` フィールドは表示されません。同様に `hostname` で示すホストがアイドル状態になれば、`idle_time` (アイドル時間) は表示されません。

各リモートホストは、`rusersd` デーモンを実行している場合にのみ本コマンドに対して応答します。このデーモンは通常 [inetd\(1M\)](#) により起動されます。

LDAP または NIS などのネームサービスがない場合、`rusers` は数値 IP アドレスとしてホスト名を表示します。

オプション

- `-a` ログインしているユーザーがいないマシンに関しても報告します。
- `-h` ホスト名のアルファベット順にソートします。
- `-i` アイドル時間の順にソートします。
- `-l` [who\(1\)](#) の出力のような、長い形式のリストを表示します。
- `-u` ユーザー数の順にソートします。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/network/network-clients

関連項目

[who\(1\)](#), [inetd\(1M\)](#), [attributes\(5\)](#)

名前	rwho – who is logged in on local machines
形式	rwho [-a]
機能説明	The rwho command produces output similar to who(1), but for all machines on your network. If no report has been received from a machine for 5 minutes, rwho assumes the machine is down, and does not report users last known to be logged into that machine. If a user has not typed to the system for a minute or more, rwho reports this idle time. If a user has not typed to the system for an hour or more, the user is omitted from the output of rwho unless the -a flag is given.
オプション	-a Report all users whether or not they have typed to the system in the past hour.
ファイル	/var/spool/rwho/whod.* information about other machines
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/network-servers

関連項目 [finger\(1\)](#), [ruptime\(1\)](#), [who\(1\)](#), [in.rwhod\(1M\)](#), [attributes\(5\)](#)

注意事項 rwho does not work through gateways.

The directory /var/spool/rwho must exist on the host from which rwho is run.

This service takes up progressively more network bandwidth as the number of hosts on the local net increases. For large networks, the cost becomes prohibitive.

The rwho service daemon, [in.rwhod\(1M\)](#), must be enabled for this command to return useful results.

名前 sar – システムアクティビティレポート

形式 sar [-aAbcdgkmpqruvw] [-o *filename*] *t* [*n*]
 sar [-aAbcdgkmpqruvw] [-e *time*] [-f *filename*] [-i *sec*]
 [-s *time*]

機能説明

最初のインスタンスでは、sar ユーティリティーは、オペレーティングシステムの累積アクティビティカウンタを *t* 秒の間隔で *n* 回サンプリングします。*t* は 5 以上にしてください。*t* を複数のオプションで指定した場合は、すべてのヘッダーが一緒に出力され、出力の判読が困難になることがあります。(サンプリング間隔が 5 より短い場合は、sar そのもののアクティビティがサンプルに影響する可能性があります。) -o オプションを指定した場合、サンプルは *filename* にバイナリ形式で保存されます。*n* のデフォルト値は 1 です。

2 番目のインスタンスでは、サンプリング間隔は指定されません。sar は、前に記録された *filename* からデータを抽出します。このファイルは、-f オプションで指定したファイルか、または、デフォルトでは、現在の日付 *dd* のシステムアクティビティの標準日常データファイル /var/adm/sa/sa *dd* です。レポートの開始時間と終了時間の範囲は、-e および -s 引数を使用して設定できます。*time* は hh[:mm[:ss]] の形式で指定します。-i オプションは、レコードを *sec* 秒間隔で選択します。その他の場合には、データファイル内のすべての間隔が報告されます。

オプション

次のオプションにより、sar で報告される情報のサブセットが変更されます。

- a ファイルアクセスシステムのルーチン iget/s、namei/s、dirblk/s の使用状況を報告します。
- A すべてのデータを報告します。-abcdgkmpqruvw と同等の意味です。
- b バッファアクティビティを報告します。
 - bread/s、bwrit/s システムバッファとディスクまたはその他のブロックデバイスの間の 1 秒当たりのデータ転送数。
 - lread/s、lwrit/s システムバッファのアクセス数。
 - %rcache、%wcache キャッシュヒット比、つまり、(1-bread/lread) の割合。
 - pread/s、pwrit/s raw (物理) デバイスメカニズムを使用した転送数。

非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティを反映します。
- c システムコールを報告します。

	scall/s	すべての種類のシステムコール数。
	sread/s、swrit/s、fork/s、exec/s	特定のシステムコール数。
	rchar/s、wchar/s	読み取りおよび書き込みのシステムコールで転送される文字数。受信または送信の exec(2) および fork(2) 呼び出しは報告されません。
	非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。	
-d	各ブロックデバイス (ディスクドライブやテープドライブなど) のアクティビティーを報告します。ただし、XDC のディスクおよびテープドライブは除きます。データが表示されると、ディスクドライブを表すために、通常はデバイス指定 <i>dsk-</i> が使用されます。テープドライブを表すために使用されるデバイス指定は、マシンに依存します。報告されるアクティビティーデータを次に示します。	
	%busy、avque	タイムデバイスの一部が転送要求に対応するためにビジー状態であった、そのときに顕著だった要求の平均数。
	read/s、write/s、blks/s	デバイスとの間の読み取り/書き込み転送の数、512 バイト単位で転送されたバイト数。
	avwait	ミリ秒単位で表した平均待ち時間。
	avserv	ミリ秒単位で表した平均サービス時間。
	一般的なシステム統計の場合は、iostat(1M)、sar(1M)、または vmstat(1M) を使用してください。	
	ディスクの命名規約については、『System Administration Guide: Advanced Administration』を参照してください。	
-e time	最大で <i>time</i> までデータを選択します。デフォルトは 18:00 です。	
-f filename	<i>filename</i> を <i>sar</i> のデータソースとして使用します。デフォルトは、現在の日常データファイル <i>/var/adm/sa/sadd</i> です。	
-g	ページングアクティビティーを報告します。	
	pgout/s	1 秒当たりのページアウト要求数。

ppgout/s	1 秒当たりのページアウトされたページ数。
pgfree/s	ページスチールデーモンによって空きリストに配置された 1 秒当たりのページ数。
pgscan/s	ページスチールデーモンによってスキャンされた 1 秒当たりのページ数。
%ufs_ipf	UFS の i ノードがそれに関連付けられた再使用可能ページを持つ iget によって、空きリストから取り出された割合。これらのページはフラッシュされ、プロセスが回収できなくなります。したがって、これはページフラッシュを伴う igets の割合です。

非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。

-i <i>sec</i>	できるだけ <i>sec</i> 秒に近い間隔でデータを選択します。
-k	カーネルメモリー割り当て (KMA) のアクティビティーを報告します。
sml_mem、alloc、fail	小さい要求用の領域を予約して割り当てるメモリープールに関する情報。KMA が小さいプール用に保持するメモリー容量 (バイト単位)、小さいメモリー容量用の要求を満たすために割り当てられたバイト数、および小さいメモリー容量用の要求のうち満たされなかった (失敗した) 要求の数。
lg_mem、alloc、fail	大きいメモリープールの情報 (小さいメモリープールの情報と同様)。
ovsz_alloc、fail	特大の要求用に割り当てられたメモリー容量と、満たされなかった特大の要求の数 (特大のメモリーは動的に割り当てられるため、プールがない)。
-m	メッセージとセマフォのアクティビティーを報告します。
msg/s、sema/s	1 秒当たりのプリミティブ数。
	非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。
-o <i>filename</i>	サンプルをファイル <i>filename</i> にバイナリ形式で保存します。
-p	ページングアクティビティーを報告します。

	atch/s	現在メモリーに入っているページを回収して満たされる 1 秒当たりのページフォルト数 (1 秒当たりの付加数)。
	pgin/s	1 秒当たりのページイン要求数。
	ppgin/s	ページインされた 1 秒当たりのページ数。
	pflt/s	1 秒当たりの保護エラーのページフォルト数 (ページへの不正なアクセス) または「書き込み時のコピー」。
	vflt/s	1 秒当たりのアドレス変換のページフォルト数 (メモリーに存在しない有効なページ)。
	slock/s	物理的な入出力が必要なソフトウェアロック要求によって発生した 1 秒当たりの障害発生数。
	非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。	
-q	占有時の待ち行列の平均長と占有時間の割合を報告します。	
	runq-sz、%runocc	メモリー内の実行可能なカーネルスレッドの実行待ち行列
	swpq-sz、%swpocc	プロセスのスワップ待ち行列
-r	未使用のメモリーページとディスクブロックを報告します。	
	freemem	ユーザープロセスが利用可能な平均ページ数。
	freeswap	ページスワップに利用可能なディスクブロック数。
-s <i>time</i>	<i>hh[:mm]</i> の形式で <i>time</i> よりあとのデータを選択します。デフォルトは 08:00 です。	
-u	CPU の使用率を報告します (デフォルト)。	
	%usr、%sys、%wio、%idle	ユーザーモードで動作している時間、システムモードで動作している時間、ブロック入出力を待つプロセスでのアイドル時間、およびその他のアイドル時間の一部。
	非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。	
-v	プロセス、i ノード、ファイルのテーブルのステータスを報告します。	

	proc-sz、inod-sz、file-sz、lock-sz	サンプルポイントで1回評価される、各テーブルのエントリ数またはサイズ。
	ov	各テーブルのサンプルポイント間で発生しているオーバーフロー。
-w		システムのスワップと切り替えのアクティビティーを報告します。
	swpin/s、swpot/s、bswin/s、bswot/s	スワップインとスワップアウトのために転送された転送数と512バイト単位の数(一部のプログラムの初期読み込みを含む)。
	pswch/s	プロセスの切り替え数。
		非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。
-y		TTY デバイスのアクティビティーを報告します。
	rawch/s、canch/s、outch/s	入力文字率、カノンで処理された入力文字率、出力文字率。
	rcvin/s、xmtin/s、mdmin/s	受信、送信、およびモデムの割り込み率。
		非大域ゾーンでの実行とプール機能がアクティブな場合、これらの値は、ゾーンがバインドされているプールのプロセッサセットのプロセッサでのアクティビティーを反映します。

使用例

- 例1 システムアクティビティーを表示する
- 次の例では、今日のここまでの CPU アクティビティーを表示します。
- ```
example% sar
```
- 例2 システムアクティビティーの進展を監視する
- CPU アクティビティーの進展を10分間監視してデータを保存するには、次のようにします。
- ```
example% sar -o temp 60 10
```
- 例3 ディスクとテープのアクティビティーを調べる
- あとで該当する期間のディスクとテープのアクティビティーを調べるには、次のようにします。

例3 ディスクとテープのアクティビティーを調べる (続き)

example% sar -d -f temp

ファイル

/var/adm/sa/sadd

属性

日常データファイル。ddは月の日付を表す数字
属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	system/accounting/legacy-accounting

関連項目

[iostat\(1M\)](#), [sar\(1M\)](#), [vmstat\(1M\)](#), [exec\(2\)](#), [fork\(2\)](#), [attributes\(5\)](#)

『Oracle Solaris の管理: 一般的なタスク』

注意事項

CPU 使用率の合計は、100 からわずかにずれることがあります。これは、割合を示す数字を生成するときの四捨五入エラーのためです。

名前	sccs – front end for the Source Code Control System (SCCS)
形式	<pre>/usr/bin/sccs [-r] [-drootprefix] [-psubdir] subcommand [option]... [file]... /usr/xpg4/bin/sccs [-r] [-d rootprefix] [-p subdir] subcommand [option]... [file]...</pre>
機能説明	The <code>sccs</code> command is a comprehensive, straightforward front end to the various utility programs of the Source Code Control System (SCCS). <code>sccs</code> applies the indicated <i>subcommand</i> to the history file associated with each of the indicated files. The name of an SCCS history file is derived by prepending the 's.' prefix to the filename of a working copy. The <code>sccs</code> command normally expects these 's.' files to reside in an SCCS subdirectory. Thus, when you supply <code>sccs</code> with a <i>file</i> argument, it normally applies the subcommand to a file named <code>s.file</code> in the SCCS subdirectory. If <i>file</i> is a path name, <code>sccs</code> looks for the history file in the SCCS subdirectory of that file's parent directory. If <i>file</i> is a directory, however, <code>sccs</code> applies the subcommand to every <code>s.file</code> file it contains. Thus, the command: <pre>example% sccs get program.c</pre> would apply the <code>get</code> subcommand to a history file named <code>SCCS/s.program.c</code>, while the command: <pre>example% sccs get SCCS</pre> would apply it to every <code>s.file</code> in the SCCS subdirectory. Options for the <code>sccs</code> command itself must appear before the <i>subcommand</i> argument. Options for a given subcommand must appear after the <i>subcommand</i> argument. These options are specific to each subcommand, and are described along with the subcommands themselves (see Subcommands below).
Running Setuid	The <code>sccs</code> command also includes the capability to run “setuid” to provide additional protection. However, this does not apply to subcommands such as sccs-admin(1) , since this would allow anyone to change the authorizations of the history file. Commands that would do so always run as the real user.
オプション	The following options are supported: <pre>/usr/bin/sccs -drootprefix /usr/xpg4/bin/sccs -d rootprefix</pre> Defines the root portion of the path name for SCCS history files. The default root portion is the current directory. <i>rootprefix</i> is prepended to the entire <i>file</i> argument, even if <i>file</i> is an absolute path name. <code>-d</code> overrides any directory specified by the <code>PROJECTDIR</code> environment variable (see <code>ENVIRONMENT VARIABLES</code> below). <pre>/usr/bin/sccs -psubdir</pre>

<code>/usr/xpg4/bin/sccs</code>	<code>-psubdir</code> Defines the (sub)directory within which a history file is expected to reside. SCCS is the default. (See EXAMPLES below). <code>-r</code> Runs <code>sccs</code> with the real user ID, rather than set to the effective user ID.
オペランド	The following operands are supported: <i>file</i> a file passed to <i>subcommand</i> <i>option</i> an option or option-argument passed to <i>subcommand</i> <i>subcommand</i> one of the subcommands listed in Usage
使用法	The usage for <code>sccs</code> is described below.
Subcommands	Many of the following <code>sccs</code> subcommands invoke programs that reside in <code>/usr/bin</code>. Many of these subcommands accept additional arguments that are documented in the reference page for the utility program the subcommand invokes. <code>admin</code> Modify the flags or checksum of an SCCS history file. Refer to sccs-admin(1) for more information about the <code>admin</code> utility. While <code>admin</code> can be used to initialize a history file, you might find that the <code>create</code> subcommand is simpler to use for this purpose.
<code>/usr/bin/sccs</code>	<code>cdc -rsid [-y[comment]]</code>
<code>/usr/xpg4/bin/sccs</code>	<code>cdc -rsid -rsid [-y[comment]]</code> Annotate (change) the delta commentary. Refer to sccs-cdc(1). The <code>fix</code> subcommand can be used to replace the delta, rather than merely annotating the existing commentary. <code>-r sid -rsid</code> Specify the SCCS delta ID (SID) to which the change notation is to be added. The SID for a given delta is a number, in Dewey decimal format, composed of two or four fields: the <i>release</i> and <i>level</i> fields, and for branch deltas, the <i>branch</i> and <i>sequence</i> fields. For instance, the SID for the initial delta is normally <code>1.1</code>. <code>-y“[comment]”</code> Specify the comment with which to annotate the delta commentary. If <code>-y</code> is omitted, <code>sccs</code> prompts for a comment. A null <i>comment</i> results in an empty annotation.
<code>/usr/bin/sccs</code>	<code>check [-b] [-u[username]]</code>

/usr/xpg4/bin/sccs

check [-b] [-u *username*] [-U]

Check for files currently being edited. Like **info** and **tell**, but returns an exit code, rather than producing a listing of files. **check** returns a non-zero exit status if anything is being edited.

-b

Ignore branches.

-u[*username*] | **-u** [*username*] | **-U**

Check only files being edited by you. When *username* is specified, check only files being edited by that user. For /usr/xpg4/bin/sccs, the **-U** option is equivalent to **-u** *<current_user>*.

clean [-b]

Remove everything in the current directory that can be retrieved from an SCCS history. Does not remove files that are being edited.

-b

Do not check branches to see if they are being edited. '**clean -b**' is dangerous when branch versions are kept in the same directory.

comb

Generate scripts to combine deltas. Refer to [sccs-comb\(1\)](#).

create

Create (initialize) history files. **create** performs the following steps:

- Renames the original source file to *,program.c* in the current directory.
- Create the history file called *s.program.c* in the SCCS subdirectory.
- Performs an '**sccs get**' on *program.c* to retrieve a read-only copy of the initial version.

deledit [-s] [-y[*comment*]]

Equivalent to an '**sccs delta**' and then an '**sccs edit**'. **deledit** checks in a delta, and checks the file back out again, but leaves the current working copy of the file intact.

-s

Silent. Do not report delta numbers or statistics.

-y[*comment*]

Supply a comment for the delta commentary. If **-y** is omitted, **delta** prompts for a comment. A null *comment* results in an empty comment field for the delta.

delget [-s] [-y[*comment*]]

Perform an '**sccs delta**' and then an '**sccs get**' to check in a delta and retrieve read-only copies of the resulting new version. See the **deledit** subcommand for a description of **-s** and **-y**. **sccs** performs a **delta** on all the files specified in the argument list, and then a **get** on all the files. If an error occurs during the **delta**, the **get** is not performed.

delta [-s] [-y[*comment*]]

Check in pending changes. Records the line-by-line changes introduced while the file was checked out. The effective user ID must be the same as the ID of the person who has the file checked out. Refer to [sccs-delta\(1\)](#). See the **deledit** subcommand for a description of **-s**

and -y.

```
/usr/bin/sccs      diff [-C] [-I] [-c date-time] [-rsid] diff-options
```

```
/usr/xpg4/bin/sccs diff [-C] [-I] [-c date-time | -cdate-time ]  
[-r sid | -rsid] diff-options
```

Compare (in [diff\(1\)](#) format) the working copy of a file that is checked out for editing, with a version from the SCCS history. Use the most recent checked-in version by default. The `diffs` subcommand accepts the same options as `diff`.

Any -r, -c, -i, -x, and -t options are passed to subcommand `get`. A -C option is passed to `diff` as -c. An -I option is passed to `diff` as -i.

-c *date-time* | -cdate-time

Use the most recent version checked in before the indicated date and time for comparison. *date-time* takes the form: `yy[mm[dd[ hh[mm[ss] ] ] ] ]`. Omitted units default to their maximum possible values; that is -c7502 is equivalent to -c750228235959.

-r *sid* | -rsid

Use the version corresponding to the indicated delta for comparison.

edit

Retrieve a version of the file for editing. ‘`sccs edit`’ extracts a version of the file that is writable by you, and creates a `p`.file in the SCCS subdirectory as lock on the history, so that no one else can check that version in or out. ID keywords are retrieved in unexpanded form. `edit` accepts the same options as `get`, below. Refer to [sccs-get\(1\)](#) for a list of ID keywords and their definitions.

enter

Similar to `create`, but omits the final ‘`sccs get`’. This can be used if an ‘`sccs edit`’ is to be performed immediately after the history file is initialized.

```
/usr/bin/sccs      fix -rsid
```

```
/usr/xpg4/bin/sccs fix -r sid | -rsid
```

Revise a (leaf) delta. Remove the indicated delta from the SCCS history, but leave a working copy of the current version in the directory. This is useful for incorporating trivial updates for which no audit record is needed, or for revising the delta commentary. `fix` must be followed by a -r option, to specify the SID of the delta to remove. The indicated delta must be the most recent (leaf) delta in its branch. Use `fix` with caution since it does not leave an audit trail of differences (although the previous commentary is retained within the history file).

```
/usr/bin/sccs      get [-ekmps] [-Gnewname] [-c date-time] [-r[sid] ]
```

/usr/xpg4/bin/sccs

`get [-ekmps] [-G newname | -Gnewname]
[-c date-time | -cdate-time] [-r sid | -rsid]`

Retrieve a version from the SCCS history. By default, this is a read-only working copy of the most recent version. ID keywords are in expanded form. Refer to [sccs-get\(1\)](#), which includes a list of ID keywords and their definitions.

`-c date-time | -cdate-time`

Retrieve the latest version checked in prior to the date and time indicated by the *date-time* argument. *date-time* takes the form: *yy[mm[dd[hh[mm[ss]]]]]*.

`-e`

Retrieve a version for editing. Same as `sccs edit`.

`-G newname | -Gnewname`

Use *newname* as the name of the retrieved version.

`-k`

Retrieve a writable copy but do not check out the file. ID keywords are unexpanded.

`-m`

Precede each line with the SID of the delta in which it was added.

`-p`

Produce the retrieved version on the standard output. Reports that would normally go to the standard output (delta IDs and statistics) are directed to the standard error.

`-r sid | -rsid`

Retrieve the version corresponding to the indicated SID. For `/usr/bin/sccs`, if no *sid* is specified, the latest *sid* for the specified file is retrieved.

`-s`

Silent. Do not report version numbers or statistics.

`sccs-help message-code|sccs-command`

`sccs-help stuck`

Supply more information about SCCS diagnostics. `sccs-help` displays a brief explanation of the error when you supply the code displayed by an SCCS diagnostic message. If you supply the name of an SCCS command, it prints a usage line. `sccs-help` also recognizes the keyword `stuck`. Refer to [sccs-help\(1\)](#).

/usr/bin/sccs

`info [-b] [-u[username ]]`

/usr/xpg4/bin/sccs

`info [-b] [-u [ username ] | -U]`

Display a list of files being edited, including the version number checked out, the version to be checked in, the name of the user who holds the lock, and the date and time the file was checked out.

`-b`

Ignore branches.

`-u[username] | -u [username] | -U`

List only files checked out by you. When *username* is specified, list only files checked out by that user. For `/usr/xpg4/bin/sccs`, the `-U` option is equivalent to `-u <current_user>`.

`print`

Print the entire history of each named file. Equivalent to an `'sccs prs -e'` followed by an `'sccs get -p -m'`.

`/usr/bin/sccs`

`prs [-el] [-cdate-time] [-rsid]`

`/usr/xpg4/bin/sccs`

`prs [-el] [-c date-time | -cdate-time] [-r sid | -rsid]`

Peruse (display) the delta table, or other portion of an `s` file. Refer to [sccs-prs\(1\)](#).

`-c date-time | -cdate-time`

Specify the latest delta checked in before the indicated date and time. The *date-time* argument takes the form: `yy[mm[dd[ hh[mm[ss] ] ] ]]`.

`-e`

Display delta table information for all deltas earlier than the one specified with `-r` (or all deltas if none is specified).

`-l`

Display information for all deltas later than, and including, that specified by `-c` or `-r`.

`-r sid | -rsid`

Specify a given delta by SID.

`prt [-y]` Display the delta table, but omit the MR field (see [sccsfile\(4\)](#) for more information on this field). Refer to [sccs-prt\(1\)](#).

`-y` Display the most recent delta table entry. The format is a single output line for each file argument, which is convenient for use in a pipeline with [awk\(1\)](#) or [sed\(1\)](#).

`/usr/bin/sccs`

`rm del -rsid`

`/usr/xpg4/bin/sccs`

`rm del -r sid`

Remove the indicated delta from the history file. That delta must be the most recent (leaf) delta in its branch. Refer to [sccs-rmdel\(1\)](#).

`sact`

Show editing activity status of an SCCS file. Refer to [sccs-sact\(1\)](#).

`sccsdiff -r old-sid -r new-sid diff-options`

Compare two versions corresponding to the indicated SIDs (deltas) using `diff`. Refer to [sccs-sccsdiff\(1\)](#).

`/usr/bin/sccs`

`tell [-b] [-u[username] ]`

/usr/xpg4/bin/sccs

tell [-b] [-u *username*] [-U]

Display the list of files that are currently checked out, one file per line.

-b

Ignore branches.

-u*username* | **-u** *username* | **-U**

List only files checked out to you. When *username* is specified, list only files checked out to that user. For /usr/xpg4/bin/sccs, the -U option is equivalent to -u *<current_user>*.

unedit

“Undo” the last edit or ‘get -e’, and return the working copy to its previous condition.

unedit backs out all pending changes made since the file was checked out.

unget

Same as unedit. Refer to [sccs-unget\(1\)](#).

val

Validate the history file. Refer to [sccs-val\(1\)](#).

what

Display any expanded ID keyword strings contained in a binary (object) or text file. Refer to [what\(1\)](#) for more information.

使用例

例1 Checking out, editing, and checking in a file

To check out a copy of program.c for editing, edit it, and then check it back in:

```
example% sccs edit program.c
```

```
1.1
```

```
new delta 1.2
```

```
14 lines
```

```
example% vi program.c
```

```
your editing session
```

```
example% sccs delget program.c
```

```
comments? clarified cryptic diagnostic
```

```
1.2
```

```
3 inserted
```

```
2 deleted
```

```
12 unchanged
```

```
1.2
```

```
15 lines
```

例2 Defining the root portion of the command pathname

sccs converts the command:

```
example% sccs -d/usr/src/include get stdio.h
```

to:

例 2 Defining the root portion of the command pathname (続き)

```
/usr/bin/get /usr/src/include/SCCS/s.stdio.h
```

例 3 Defining the resident subdirectory

The command:

```
example% sccs -pprivate get include/stdio.h
```

becomes:

```
/usr/bin/get include/private/s.stdio.h
```

例 4 Initializing a history file

To initialize the history file for a source file named `program.c`, make the SCCS subdirectory, and then use 'sccs create':

```
example% mkdir SCCS
```

```
example% sccs create program.c
```

```
program.c:
```

```
1.1
```

```
14 lines
```

After verifying the working copy, you can remove the backup file that starts with a comma:

```
example% diff program.c ,program.c
```

```
example% rm ,program.c
```

例 5 Retrieving a file from another directory

To retrieve a file from another directory into the current directory:

```
example% sccs get /usr/src/sccs/cc.c
```

or:

```
example% sccs -p/usr/src/sccs/ get cc.c
```

例 6 Checking out all files

To check out all files under SCCS in the current directory:

```
example% sccs edit SCCS
```

例 7 Checking in all files

To check in all files currently checked out to you:

```
example% sccs delta 'sccs tell -u'
```

例8 Entering multiple lines of comments

If using `-y` to enter a comment, for most shells, enclose the comment in single or double quotes. In the following example, `Myfile` is checked in with a two-line comment:

```
example% sccs deledit Myfile -y"Entering a
multi-line comment"
No id keywords (cm7)
1.2
2 inserted
0 deleted
14 unchanged
1.2
new delta 1.3
```

Displaying the SCCS history of `Myfile`:

```
example% sccs prt Myfile
```

SCCS/s.Myfile:

```
D 1.2  01/04/20  16:37:07  me 2 1    00002/00000/00014
Entering a
multi-line comment

D 1.1  01/04/15  13:23:32  me 1 0    00014/00000/00000
date and time created 01/04/15 13:23:32 by me
```

If `-y` is not used and `sccs` prompts for a comment, the newlines must be escaped using the backslash character (`\`):

```
example% sccs deledit Myfile
comments? Entering a \
multi-line comment
No id keywords (cm7)
1.2
0 inserted
0 deleted
14 unchanged
1.2
new delta 1.3
```

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `sccs`: `LANG`, `LC_ALL`, `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

PROJECTDIR If contains an absolute path name (beginning with a slash), `sccs` searches for SCCS history files in the directory given by that variable.

If `PROJECTDIR` does not begin with a slash, it is taken as the name of a user, and `sccs` searches the `src` or source subdirectory of that user's home directory for history files. If such a directory is found, it is used. Otherwise,

the value is used as a relative path name.

終了ステータス The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred.

ファイル

SCCS	SCCS subdirectory
SCCS/d. <i>file</i>	temporary file of differences
SCCS/p. <i>file</i>	lock (permissions) file for checked-out versions
SCCS/q. <i>file</i>	temporary file
SCCS/s. <i>file</i>	SCCS history file
SCCS/x. <i>file</i>	temporary copy of the s. <i>file</i>
SCCS/z. <i>file</i>	temporary lock file
/usr/bin/*	SCCS utility programs

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/sccs	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	developer/build/make

/usr/xpg4/bin/sccs	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	developer/xopen/xcu4
	Interface Stability	Committed
	Standard	See standards(5) .

関連項目 [awk\(1\)](#), [diff\(1\)](#), [sccs-admin\(1\)](#), [sccs-cdc\(1\)](#), [sccs-comb\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-prs\(1\)](#), [sccs-rmdel\(1\)](#), [sccs-sact\(1\)](#), [sccs-sccsdiff\(1\)](#), [sccs-unget\(1\)](#), [sccs-val\(1\)](#), [sed\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	sccs-admin, admin – create and administer SCCS history files										
形式	<pre>/usr/bin/admin [-bhnz] [-a username groupid]... [-d flag] ... [-e username groupid]... [-f flag [value]] ... [-i [filename]] [-m mr-list] [-rrelease] [-t [description-file]] [-y [comment]] s.filename...</pre>										
機能説明	The admin command creates or modifies the flags and other parameters of SCCS history files. Filenames of SCCS history files begin with the <code>s.</code> prefix, and are referred to as <code>s.</code> files, or history files. The named <code>s.</code> file is created if it does not exist already. Its parameters are initialized or modified according to the options you specify. Parameters not specified are given default values when the file is initialized, otherwise they remain unchanged. If a directory name is used in place of the <code>s.filename</code> argument, the admin command applies to all <code>s.</code> files in that directory. Unreadable <code>s.</code> files produce an error. The use of <code>'-'</code> as the <code>s.filename</code> argument indicates that the names of files are to be read from the standard input, one <code>s.</code> file per line.										
オプション	The following options are supported: <table><tr><td>-a <i>username</i> <i>groupid</i></td><td>Adds a user name, or a numerical group ID, to the list of users who may check deltas in or out. If the list is empty, any user is allowed to do so.</td></tr><tr><td>-b</td><td>Forces encoding of binary data. Files that contain ASCII NUL or other control characters, or that do not end with a NEWLINE, are recognized as binary data files. The contents of such files are stored in the history file in encoded form. See uencode(1C) for details about the encoding. This option is normally used in conjunction with <code>-i</code> to force admin to encode initial versions not recognized as containing binary data.</td></tr><tr><td>-d <i>flag</i></td><td>Deletes the indicated <i>flag</i> from the SCCS file. The <code>-d</code> option may be specified only for existing <code>s.</code> files. See <code>-f</code> for the list of recognized flags.</td></tr><tr><td>-e <i>username</i> <i>groupid</i></td><td>Erases a user name or group ID from the list of users allowed to make deltas.</td></tr><tr><td>-f <i>flag</i> [<i>value</i>]</td><td>Sets the indicated <i>flag</i> to the (optional) <i>value</i> specified. The following flags are recognized: b Enables branch deltas. When <code>b</code> is set, branches can be created using the <code>-b</code> option of the SCCS get command (see sccs-get(1)).</td></tr></table>	-a <i>username</i> <i>groupid</i>	Adds a user name, or a numerical group ID, to the list of users who may check deltas in or out. If the list is empty, any user is allowed to do so.	-b	Forces encoding of binary data. Files that contain ASCII NUL or other control characters, or that do not end with a NEWLINE, are recognized as binary data files. The contents of such files are stored in the history file in encoded form. See uencode(1C) for details about the encoding. This option is normally used in conjunction with <code>-i</code> to force admin to encode initial versions not recognized as containing binary data.	-d <i>flag</i>	Deletes the indicated <i>flag</i> from the SCCS file. The <code>-d</code> option may be specified only for existing <code>s.</code> files. See <code>-f</code> for the list of recognized flags.	-e <i>username</i> <i>groupid</i>	Erases a user name or group ID from the list of users allowed to make deltas.	-f <i>flag</i> [<i>value</i>]	Sets the indicated <i>flag</i> to the (optional) <i>value</i> specified. The following flags are recognized: b Enables branch deltas. When <code>b</code> is set, branches can be created using the <code>-b</code> option of the SCCS get command (see sccs-get(1)).
-a <i>username</i> <i>groupid</i>	Adds a user name, or a numerical group ID, to the list of users who may check deltas in or out. If the list is empty, any user is allowed to do so.										
-b	Forces encoding of binary data. Files that contain ASCII NUL or other control characters, or that do not end with a NEWLINE, are recognized as binary data files. The contents of such files are stored in the history file in encoded form. See uencode(1C) for details about the encoding. This option is normally used in conjunction with <code>-i</code> to force admin to encode initial versions not recognized as containing binary data.										
-d <i>flag</i>	Deletes the indicated <i>flag</i> from the SCCS file. The <code>-d</code> option may be specified only for existing <code>s.</code> files. See <code>-f</code> for the list of recognized flags.										
-e <i>username</i> <i>groupid</i>	Erases a user name or group ID from the list of users allowed to make deltas.										
-f <i>flag</i> [<i>value</i>]	Sets the indicated <i>flag</i> to the (optional) <i>value</i> specified. The following flags are recognized: b Enables branch deltas. When <code>b</code> is set, branches can be created using the <code>-b</code> option of the SCCS get command (see sccs-get(1)).										

cceil

Sets a ceiling on the releases that can be checked out. *ceil* is a number less than or equal to 9999. If *c* is not set, the ceiling is 9999.

dsid

Specifies the default delta number, or SID, to be used by an SCCS get command.

ffloor

Sets a floor on the releases that can be checked out. The floor is a number greater than 0 but less than 9999. If *f* is not set, the floor is 1.

i

Treats the 'No id keywords (ge6)' message issued by an SCCS get or delta command as an error rather than a warning.

j

Allows concurrent updates.

*la**l release[, release...]*

Locks the indicated list of releases against deltas. If *a* is used, this flag locks out deltas to all releases. An SCCS 'get -e' command fails when applied against a locked release.

mmodule

Supplies a value for the module name to which the `sccs-admin.1` keyword is to expand. If the *m* flag is not specified, the value assigned is the name of the SCCS file with the leading `s.` removed.

n

Creates empty releases when releases are skipped. These null (empty) deltas serve as anchor points for branch deltas.

qvalue

Supplies a *value* to which the keyword is to expand when a read-only version is retrieved with the SCCS get command.

snumber

Specifies how many lines of code are scanned for the SCCS keyword.

ttype

Supplies a value for the module type to which the keyword is to expand.

- v*[*program*]
 Specifies a validation *program* for the MR numbers associated with a new delta. The optional *program* specifies the name of an MR number validity checking *program*. If this flag is set when creating an SCCS file, the *-m* option must also be used, in which case the list of MRs may be empty.
- y*[*value*, [*value*]]
 Specifies the SCCS keywords to be expanded. If no *value* is specified, no keywords will be expanded.
- h*
 Checks the structure of an existing *s*.file (see [sccsfile\(4\)](#)), and compares a newly computed check-sum with one stored in the first line of that file. *-h* inhibits writing on the file and so nullifies the effect of any other options.
- i*[*filename*]
 Initializes the history file with text from the indicated file. This text constitutes the initial delta, or set of checked-in changes. If *filename* is omitted, the initial text is obtained from the standard input. Omitting the *-i* option altogether creates an empty *s*.file. You can only initialize one *s*.file with text using *-i*. This option implies the *-n* option.
- m mr-list*
 Inserts the indicated Modification Request (MR) numbers into the commentary for the initial version. When specifying more than one MR number on the command line, *mr-list* takes the form of a quoted, space-separated list. A warning results if the *v* flag is not set or the MR validation fails.
- n*
 Creates a new SCCS history file.
- r release*
 Specifies the release for the initial delta. *-r* may be used only in conjunction with *-i*. The initial delta is inserted into release 1 if this option is omitted. The level of the initial delta is always 1. Initial deltas are named 1.1 by default.
- t*[*description-file*]
 Inserts descriptive text from the file *description-file*. When *-t* is used in conjunction with *-n*, or *-i* to initialize a new *s*.file, the *description-file* must be supplied. When modifying the description for an existing file: a *-t* option without a *description-file* removes the descriptive text, if any; a *-t* option with a *description-file* replaces the existing text.
- y*[*comment*]
 Inserts the indicated *comment* in the "Comments:" field for the initial delta. Valid only in conjunction with *-i* or *-n*. If *-y* option is omitted, a default comment line is inserted that notes the date and time the history file was created.

-z Recomputes the file check-sum and stores it in the first line of the s . file. *Caution:* It is important to verify the contents of the history file (see [sccs-val\(1\)](#), and the print subcommand in [sccs\(1\)](#)), since using -z on a truly corrupted file may prevent detection of the error.

使用例 例1 Preventing SCCS keyword expansion

In the following example, 10 lines of file will be scanned and only the W, Y, X keywords will be interpreted:

```
example% sccs admin -fs10 file
example% sccs admin -fyW,Y,X file
example% get file
```

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of alias and unalias: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

終了ステータス The following exit values are returned:

- 0 Successful completion.
- 1 An error occurred.

ファイル s.* history file

 SCCS/s.* history file in SCCS subdirectory

 z.* temporary lock file

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [sccs\(1\)](#), [sccs-cdc\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-rmdel\(1\)](#), [sccs-val\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 Use the sccs -help command for explanations of SCCS commands. See [sccs-help\(1\)](#).

警告 The last component of all SCCS filenames must have the 's .' prefix. New SCCS files are given mode 444 (see [chmod\(1\)](#)). All writing done by admin is to a temporary file with an x . prefix, created with mode 444 for a new SCCS file, or with the same mode as an existing SCCS file. After successful execution of admin, the existing s . file is removed and replaced with the x . file. This ensures that changes are made to the SCCS file only when no errors have occurred.

It is recommended that directories containing SCCS files have permission mode 755, and that the `s.` files themselves have mode 444. The mode for directories allows only the owner to modify the SCCS files contained in the directories, while the mode of the `s.` files prevents all modifications except those performed using SCCS commands.

If it should be necessary to patch an SCCS file for any reason, the mode may be changed to 644 by the owner to allow use of a text editor. However, extreme care must be taken when doing this. The edited file should *always* be processed by an `'admin -h'` command to check for corruption, followed by an `'admin -z'` command to generate a proper check-sum. Another `'admin -h'` command is recommended to ensure that the resulting `s.` file is valid.

`admin` also uses a temporary lock `s.` file, starting with the `'z.'` prefix, to prevent simultaneous updates to the `s.` file. See [sccs-get\(1\)](#) for further information about the `'z.` file'.

名前	sccs-cdc, cdc – change the delta commentary of an SCCS delta						
形式	<code>cdc -rsid [-mmr-list] [-y [comment]] s.filename...</code>						
機能説明	cdc annotates the delta commentary for the SCCS delta ID (SID) specified by the -r option in each named s . file. If the v flag is set in the s . file, you can also use cdc to update the Modification Request (MR) list. If you checked in the delta, or, if you own the file and directory and have write permission, you can use cdc to annotate the commentary. Rather than replacing the existing commentary, cdc inserts the new comment you supply, followed by a line of the form: <pre>*** CHANGED *** yy/mm/dd hh/mm/ss username</pre> above the existing commentary. If a directory is named as the s.filename argument, the cdc command applies to all s . files in that directory. Unreadable s . files produce an error; processing continues with the next file (if any). If '-' is given as the s.filename argument, each line of the standard input is taken as the name of an SCCS history file to be processed, and the -m and -y options must be used.						
オプション	<table> <tr> <td>-rsid</td><td>Specify the SID of the delta to change.</td></tr> <tr> <td>-mmr-list</td><td>Specify one or more MR numbers to add or delete. When specifying more than one MR on the command line, <i>mr-list</i> takes the form of a quoted, space-separated list. To delete an MR number, precede it with a ! character (an empty MR list has no effect). A list of deleted MRs is placed in the comment section of the delta commentary. If -m is not used and the standard input is a terminal, cdc prompts with MRs? for the list (before issuing the comments? prompt). -m is only useful when the v flag is set in the s . file. If that flag has a value, it is taken to be the name of a program to validate the MR numbers. If that validation program returns a non-zero exit status, cdc terminates and the delta commentary remains unchanged.</td></tr> <tr> <td>-y[comment]</td><td>Use <i>comment</i> as the annotation in the delta commentary. The previous comments are retained; the <i>comment</i> is added along with a notation that the commentary was changed. A null <i>comment</i> leaves the commentary unaffected. If -y is not specified and the standard input is a terminal, cdc prompts with comments? for the text of the notation to be added. An unescaped NEWLINE character terminates the annotation text.</td></tr> </table>	-rsid	Specify the SID of the delta to change.	-mmr-list	Specify one or more MR numbers to add or delete. When specifying more than one MR on the command line, <i>mr-list</i> takes the form of a quoted, space-separated list. To delete an MR number, precede it with a ! character (an empty MR list has no effect). A list of deleted MRs is placed in the comment section of the delta commentary. If -m is not used and the standard input is a terminal, cdc prompts with MRs? for the list (before issuing the comments? prompt). -m is only useful when the v flag is set in the s . file. If that flag has a value, it is taken to be the name of a program to validate the MR numbers. If that validation program returns a non-zero exit status, cdc terminates and the delta commentary remains unchanged.	-y[comment]	Use <i>comment</i> as the annotation in the delta commentary. The previous comments are retained; the <i>comment</i> is added along with a notation that the commentary was changed. A null <i>comment</i> leaves the commentary unaffected. If -y is not specified and the standard input is a terminal, cdc prompts with comments? for the text of the notation to be added. An unescaped NEWLINE character terminates the annotation text.
-rsid	Specify the SID of the delta to change.						
-mmr-list	Specify one or more MR numbers to add or delete. When specifying more than one MR on the command line, <i>mr-list</i> takes the form of a quoted, space-separated list. To delete an MR number, precede it with a ! character (an empty MR list has no effect). A list of deleted MRs is placed in the comment section of the delta commentary. If -m is not used and the standard input is a terminal, cdc prompts with MRs? for the list (before issuing the comments? prompt). -m is only useful when the v flag is set in the s . file. If that flag has a value, it is taken to be the name of a program to validate the MR numbers. If that validation program returns a non-zero exit status, cdc terminates and the delta commentary remains unchanged.						
-y[comment]	Use <i>comment</i> as the annotation in the delta commentary. The previous comments are retained; the <i>comment</i> is added along with a notation that the commentary was changed. A null <i>comment</i> leaves the commentary unaffected. If -y is not specified and the standard input is a terminal, cdc prompts with comments? for the text of the notation to be added. An unescaped NEWLINE character terminates the annotation text.						

使用例

例 1 Changing the annotated commentary
The following command:

example% **cdc -r1.6 -y"corrected commentary" s.program.c**

produces the following annotated commentary for delta 1.6 in s.program.c:

D 1.6 88/07/05 23:21:07 username 9 0 00001/00000/00000
MRs:
COMMENTS:
corrected commentary
*** CHANGED *** 88/07/07 14:09:41 username
performance enhancements in main()

ファイル

z.file temporary lock file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

関連項目

[sccs\(1\)](#), [sccs-admin\(1\)](#), [sccs-comb\(1\)](#), [sccs-delta\(1\)](#), [sccs-help\(1\)](#), [sccs-prs\(1\)](#), [sccs-prt\(1\)](#), [sccs-rmdel\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#)

診断

Use the `sccs-help` command for explanations of SCCS commands. See [sccs-help\(1\)](#).

名前	sccs-comb, comb – combine SCCS deltas
形式	comb [-os] [-csid-list] [-psid] <i>s.filename</i> ...
機能説明	comb generates a shell script (see sh(1)) that you can use to reconstruct the indicated <i>s</i>. files. This script is written to the standard output. If a directory name is used in place of the <i>s.filename</i> argument, the comb command applies to all <i>s</i>. files in that directory. Unreadable <i>s</i>. files produce an error; processing continues with the next file (if any). The use of – as the <i>s.filename</i> argument indicates that the names of files are to be read from the standard input, one <i>s</i>. file per line. If no options are specified, comb preserves only the most recent (leaf) delta in a branch, and the minimal number of ancestors needed to preserve the history.
オプション	The following options are supported: -o For each get -e generated, access the reconstructed file at the release of the delta to be created. Otherwise, the reconstructed file is accessed at the most recent ancestor. The use of -o can decrease the size of the reconstructed <i>s</i>. file. It can also alter the shape of the delta tree of the original file. -s Generate scripts to gather statistics, rather than combining deltas. When run, the shell scripts report: the file name, size (in blocks) after combining, original size (also in blocks), and the percentage size change, computed by the formula: $100 * (original - combined) / original$ This option can be used to calculate the space that is saved, before actually doing the combining. -csid-list Include the indicated list of deltas. All other deltas are omitted. <i>sid-list</i> is a comma-separated list of SCCS delta IDs (SIDs). To specify a range of deltas, use a – separator instead of a comma, between two SIDs in the list. -pSID The SID of the oldest delta to be preserved.
ファイル	<i>s</i>. COMB reconstructed SCCS file comb????? temporary file
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

関連項目	sccs(1) , sccs-admin(1) , sccs-cdc(1) , sccs-delta(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , sccs-rmdel(1) , sccs-sccsdiff(1) , what(1) , sccsfile(4) , attributes(5)

診断	Use the <code>sccs-help</code> command for explanations of SCCS commands. See sccs-help(1) .
使用上の留意点	<code>comb</code> might rearrange the shape of the tree of deltas. It might not save any space; in fact, it is possible for the reconstructed file to actually be larger than the original.

名前	sccs-delta, delta – make a delta to an SCCS file
形式	<pre> /usr/bin/delta [-dnps] [-g sid-list -gsid-list] [-m mr-list -mmr-list] [-r sid -rsid] [-y [comment]] s.filename... /usr/xpg4/bin/delta [-dnps] [-g sid-list -gsid-list] [-m mr-list -mmr-list] [-r sid -rsid] [-y [comment]] s.filename... </pre>
機能説明	The <code>delta</code> utility checks in a record of the line-by-line differences made to a checked-out version of a file under SCCS control. These changes are taken from the writable working copy that was retrieved using the SCCS <code>get</code> command (see sccs-get(1)). This working copy does not have the ‘s.’ prefix, and is also referred to as a g-file. If a directory name is used in place of the <i>s.filename</i> argument, the <code>delta</code> command applies to all <i>s.</i> files in that directory. Unreadable <i>s.</i> files produce an error; processing continues with the next file (if any). The use of ‘-’ as the <i>s.filename</i> argument indicates that the names of files are to be read from the standard input, one <i>s.</i> file per line (requires <code>-y</code>, and in some cases, <code>-m</code>). <code>delta</code> can issue prompts on the standard output depending upon the options specified and the flags that are set in the <i>s.</i> file (see sccs-admin(1), and the <code>-m</code> and <code>-y</code> options below, for details).
/usr/xpg4/bin/delta	The SID of the delta is not echoed to <code>stdout</code> .
オプション	The following options are supported: <code>-d</code> Use command diff(1) instead of bdiff(1). Returns exit status 2 if <i>s.filename</i> argument is not specified. <code>-n</code> Retain the edited g-file, which is normally removed at the completion of processing. <code>-p</code> Display line-by-line differences (in diff(1) format) on the standard output. <code>-s</code> Silent. Do not display warning or confirmation messages. Do not suppress error messages (which are written to standard error). <code>-g sid-list -gsid-list</code> Specify a list of deltas to omit when the file is accessed at the SCCS version ID (SID) created by this delta. <i>sid-list</i> is a comma-separated list of SIDs. To specify a range of deltas, use a ‘-’ separator instead of a comma, between two SIDs in the list. <code>-m mr-list -mmr-list</code> If the SCCS file has the <code>v</code> flag set (see sccs-admin(1)), you must supply one or more Modification Request (MR) numbers for the new delta. When specifying more than one MR number on the command line, <i>mr-list</i> takes the form of a quoted, space-separated list. If <code>-m</code> is not used and the standard input is a terminal, <code>delta</code> prompts with

MRs? for the list (before issuing the comments? prompt). If the v flag in the s.file has a value, it is taken to be the name of a program to validate the MR numbers. If that validation program returns a non-zero exit status, delta terminates without checking in the changes.

- r sid | -rsid

When two or more versions are checked out, specify the version to check in. This SID value can be either the SID specified on the get command line, or the SID of the new version to be checked in as reported by get. A diagnostic results if the specified SID is ambiguous, or if one is required but not supplied.
- y[comment]

Supply a comment for the delta table (version log). A null comment is accepted, and produces an empty commentary in the log. If -y is not specified and the standard input is a terminal, delta prompts with 'comments?'. An unescaped NEWLINE terminates the comment.

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of delta: LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_MESSAGES, and NLSPATH.

- 終了ステータス
- The following exit values are returned:
- 0

Successful completion.
- 1

An error occurred and the -d option had not been specified.
- 2

An error occurred, the -d option had been specified, and the s.filename argument was not specified.

- ファイル
- d.file

temporary file of differences
- p.file

lock file for a checked-out version
- q.file

temporary file
- s.file

SCCS history file
- x.file

temporary copy of the s.file
- z.file

temporary file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/delta

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

/usr/xpg4/bin/delta

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/xopen/xcu4
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [bdiff\(1\)](#), [diff\(1\)](#), [sccs-admin\(1\)](#), [sccs-cdc\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-prs\(1\)](#), [sccs-prt\(1\)](#), [sccs-rmdel\(1\)](#), [sccs-sccsdiff\(1\)](#), [sccs-unget\(1\)](#), [sccs\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 Use the SCCS `help` command for explanations (see [sccs-help\(1\)](#)).

警告 Lines beginning with an ASCII SOH character (binary 001) cannot be placed in the SCCS file unless the SOH is escaped. This character has special meaning to SCCS (see [sccsfile\(4\)](#)) and produces an error.

名前 sccs-get, get – retrieve a version of an SCCS file

形式

```
/usr/bin/get [-begkmpst] [-l [p]] [-asequence]
             [-c date-time | -cdate-time] [-Gg-file]
             [-i sid-list | -isid-list] [-r [sid]]
             [-x sid-list | -xsid-list] s.filename...

/usr/xpg4/bin/get [-begkmpst] [-l [p]] [-asequence]
                  [-c date-time | -cdate-time] [-Gg-file]
                  [-i sid-list | -isid-list] [-r sid | -rsid]
                  [-x sid-list | -xsid-list] s.filename...
```

機能説明 The get utility retrieves a working copy from the SCCS history file, according to the specified options.

For each *s.filename* argument, get displays the SCCS delta ID (SID) and number of lines retrieved.

If a directory name is used in place of the *s.filename* argument, the get command applies to all *s*. files in that directory. Unreadable *s*. files produce an error; processing continues with the next file (if any). The use of '-' as the *s.filename* argument indicates that the names of files are to be read from the standard input, one *s*. file per line.

The retrieved file normally has the same filename base as the *s*. file, less the prefix, and is referred to as the *g*-file.

For each file processed, get responds (on the standard output) with the SID being accessed, and with the number of lines retrieved from the *s*. file.

オプション The following options are supported:

- asequence Retrieves the version corresponding to the indicated delta sequence number. This option is used primarily by the SCCS comb command (see [sccs-comb\(1\)](#)). For users, -r is an easier way to specify a version. The -a option supersedes the -r option when both are used.
- b Creates a new branch. Used with the -e option to indicate that the new delta should have a SID in a new branch. Instead of incrementing the level for version to be checked in, get indicates in the *p*. file that the delta to be checked in should either initialize a new branch and sequence (if there is no existing branch at the current level), or increment the branch component of the SID. If the *b* flag is not set in the *s*. file, this option is ignored.
- c *date-time* | -cdate-time Retrieves the latest version checked in prior to the date and time indicated by the *date-time* argument. *date-time* takes the form:

`yy[mm[dd[ hh[mm[ss] ] ] ] ]`

Units omitted from the indicated date and time default to their maximum possible values; that is -c7502 is equivalent to -c750228235959. Values of *yy* in the range 69–99 refer to the twentieth century. Values in the range 00–68 refer to the twenty-first century. Any number of non-numeric characters can separate the various 2 digit components. If white-space characters occur, the *date-time* specification must be quoted.

- e Retrieves a version for editing. With this option, `get` places a lock on the `s` . file, so that no one else can check in changes to the version you have checked out. If the `j` flag is set in the `s` . file, the lock is advisory: `get` issues a warning message. Concurrent use of ‘`get -e`’ for different SIDs is allowed. However, `get` does not check out a version of the file if a writable version is present in the directory. All SCCS file protections stored in the `s` . file, including the release ceiling, floor, and authorized user list, are honored by ‘`get -e`’.
- g Gets the SCCS version ID, without retrieving the version itself. Used to verify the existence of a particular SID.
- G*newname* Uses *newname* as the name of the retrieved version.
- i *sid-list* | -i*sid-list* Specifies a list of deltas to include in the retrieved version. The included deltas are noted in the standard output message. *sid-list* is a comma-separated list of SIDs. To specify a range of deltas, use a ‘-’ separator instead of a comma, between two SIDs in the list.
- k Suppresses expansion of ID keywords. -k is implied by the -e.
- l [p] Retrieves a summary of the delta table (version log) and write it to a listing file, with the ‘l.’ prefix (called ‘l. file’). When -lp is used, write the summary onto the standard output.
- m Precedes each retrieved line with the SID of the delta in which it was added to the file. The SID is separated from the line with a TAB.
- n Precedes each line with the %M% ID keyword and a TAB. When both the -m and -n options are used, the ID keyword precedes the SID, and the line of text.
- p Writes the text of the retrieved version to the standard output. All messages that normally go to the standard output are written to the standard error instead.

	- s	Suppresses all output normally written on the standard output. However, fatal error messages (which always go to the standard error) remain unaffected.
	- t	Retrieves the most recently created (top) delta in a given release (for example: - r1).
/usr/bin/get	- r[<i>sid</i>]	Retrieves the version corresponding to the indicated SID (delta). The SID for a given delta is a number, in Dewey decimal format, composed of two or four fields: the <i>release</i> and <i>level</i> fields, and for branch deltas, the <i>branch</i> and <i>sequence</i> fields. For instance, if 1.2 is the SID, 1 is the release, and 2 is the level number. If 1.2.3.4 is the SID, 3 is the branch and 4 is the sequence number. You need not specify the entire SID to retrieve a version with get. When you omit - r altogether, or when you omit both release and level, get normally retrieves the highest release and level. If the d flag is set to an SID in the s . file and you omit the SID, get retrieves the default version indicated by that flag. When you specify a release but omit the level, get retrieves the highest level in that release. If that release does not exist, get retrieves highest level from the next-highest existing release. Similarly with branches, if you specify a release, level and branch, get retrieves the highest sequence in that branch.
/usr/xpg4/bin/get	- r <i>sid</i> - r <i>sid</i>	Same as for /usr/bin/get except that SID is mandatory.
	- x <i>sid-list</i> - x <i>sid-list</i>	Excludes the indicated deltas from the retrieved version. The excluded deltas are noted in the standard output message. <i>sid-list</i> is a comma-separated list of SIDs. To specify a range of deltas, use a ‘–’ separator instead of a comma, between two SIDs in the list.

Output

/usr/bin/get	The output format for /usr/bin/get is as follows: "%s\n%d lines\n", <SID>, <number of lines>
/usr/xpg4/bin/get	The output format for /usr/xpg4/bin/get is as follows: "%s\n%d\n", <SID>, <number of lines>

使用法 Usage guidelines are as follows:

ID Keywords In the absence of - e or - k, get expands the following ID keywords by replacing them with the indicated values in the text of the retrieved source.

<i>Keyword</i>	<i>Value</i>
%A%	Shorthand notation for an ID line with data for what(1) : %Z%Y% %M% %I%Z%
%B%	SID branch component
%C%	Current line number. Intended for identifying messages output by the program such as ‘ <i>this shouldn’t have happened</i> ’ type errors. It is <i>not</i> intended to be used on every line to provide sequence numbers.
%D%	Current date: <i>yy/mm/dd</i>
%E%	Date newest applied delta was created: <i>yy/mm/dd</i>
%F%	SCCS s . file name
%G%	Date newest applied delta was created: <i>mm/dd/yy</i>
%H%	Current date: <i>mm/dd/yy</i>
%I%	SID of the retrieved version: %R%. %L%. %B%. %S%
%%	SID level component
%M%	Module name: either the value of the m flag in the s . file (see sccs-admin(1)), or the name of the s . file less the prefix
%P%	Fully qualified s . file name
%Q%	Value of the q flag in the s . file
%R%	SID Release component
%S%	SID Sequence component
%T%	Current time: <i>hh:mm:ss</i>
%U%	Time the newest applied delta was created: <i>hh:mm:ss</i>
%W%	Shorthand notation for an ID line with data for what: %Z%%&;%I%
%Y%	Module type: value of the t flag in the s . file
%Z%	4-character string: ‘@(#)’ , recognized by what

ID String

The table below explains how the SCCS identification string is determined for retrieving and creating deltas.

Determination of SCCS Identification String				
SID (1) Specified	-b Option Used (2)	Other Conditions	SID Retrieved	SID of Delta to be Created
none (3)	no	R defaults to mR	mR.mL	mR.(mL+1)
none (3)	yes	R defaults to mR	mR.mL	mR.mL.(mB+1).1

Determination of SCCS Identification String				
SID (1) Specified	-b Option Used (2)	Other Conditions	SID Retrieved	SID of Delta to be Created
R	no	$R > mR$	mR.mL	R.1 (4)
R	no	$R = mR$	mR.mL	mR.(mL+1)
R	yes	$R > mR$	mR.mL	mR.mL.(mB+1).1
R	yes	$R = mR$	mR.mL	mR.mL.(mB+1).1
R	—	$R < mR$ and R does <i>not</i> exist	hR.mL (5)	hR.mL.(mB+1).1
R	—	Trunk succ. (6) in release $> R$ and R exists	R.mL	R.mL.(mB+1).1
R.L	no	No trunk succ.	R.L	R.(L+1)
R.L	yes	No trunk succ.	R.L	R.L.(mB+1).1
R.L	—	Trunk succ. in release $\geq R$	R.L	R.L.(mB+1).1
R.L.B	no	No branch succ.	R.L.B.mS	R.L.B.(mS+1)
R.L.B	yes	No branch succ.	R.L.B.mS	R.L.(mB+1).1
R.L.B.S	no	No branch succ.	R.L.B.S	R.L.B.(S+1)
R.L.B.S	yes	No branch succ.	R.L.B.S	R.L.(mB+1).1
R.L.B.S	—	Branch succ.	R.L.B.S	R.L.(mB+1).1

- (1) 'R', 'L', 'B', and 'S' are the 'release', 'level', 'branch', and 'sequence' components of the SID, respectively; 'm' means 'maximum'. Thus, for example, 'R.mL' means 'the maximum level number within release R'; 'R.L.(mB+1).1' means 'the first sequence number on the *new* branch (that is, maximum branch number plus one) of level L within release R'. *Note:* If the SID specified is of the form 'R.L', 'R.L.B', or 'R.L.B.S', each of the specified components *must* exist.
- (2) The -b option is effective only if the b flag is present in the file. An entry of '—' means 'irrelevant'.
- (3) This case applies if the d (default SID) flag is *not* present in the file. If the d flag is present in the file, the SID obtained from the d flag is interpreted as if it had been specified on the command line. Thus, one of the other cases in this table applies.
- (4) Forces creation of the *first* delta in a *new* release.
- (5) 'hR' is the highest *existing* release that is lower than the specified, *nonexistent*, release R.
- (6) Successor.

環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of get: LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_MESSAGES, and NLSPATH.	
ファイル	"g-file"	version retrieved by get
	l.file	file containing extracted delta table info
	p.file	permissions (lock) file
	z.file	temporary copy of s.file
属性	See attributes(5) for descriptions of the following attributes:	
/usr/bin/get	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	developer/build/make
/usr/xpg4/bin/get	ATTRIBUTE TYPE	ATTRIBUTE VALUE
	Availability	developer/xopen/xcu4
	Interface Stability	Committed
	Standard	See standards(5) .
関連項目	sccs(1) , sccs-admin(1) , sccs-delta(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , sccs-sact(1) , sccs-unget(1) , what(1) , sccsfile(4) , attributes(5) , environ(5) , standards(5)	
診断	Use the <code>sccs -help</code> command for explanations of SCCS commands. See sccs-help(1) .	
使用上の留意点	If the effective user has write permission (either explicitly or implicitly) in the directory containing the SCCS files, but the real user does not, only one file can be named when using <code>-e</code> .	

- 名前

sccs-help, sccshelp – ask for help regarding SCCS error or warning messages
- 形式

/usr/bin/sccshelp [argument]...
- 機能説明

The `sccs-help` utility retrieves information to further explain errors messages and warnings from SCCS commands. It also provides some information about SCCS command usage. If no arguments are given, `sccs-help` prompts for one.

An *argument* may be a message number (which normally appears in parentheses following each SCCS error or warning message), or an SCCS command name. `sccs-help` responds with an explanation of the message or a usage line for the command.

When all else fails, try `/usr/bin/sccshelp stuck`.
- ファイル

/usr/lib/help directory containing files of message text
- 属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

- 関連項目

[sccs\(1\)](#), [sccs-admin\(1\)](#), [sccs-cdc\(1\)](#), [sccs-comb\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-prs\(1\)](#), [sccs-prt\(1\)](#), [sccs-rmdel\(1\)](#), [sccs-sact\(1\)](#), [sccs-sccsdiff\(1\)](#), [sccs-unget\(1\)](#), [sccs-val\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#)

名前	sccs-prs, prs – display selected portions of an SCCS history
形式	prs [-ael] [-cdate-time] [-ddataspec] [-rsid] s.filename...
機能説明	The prs utility displays part or all of the SCCS file (see sccsfile(4)) in a user supplied format. If a directory name is used in place of the <i>s.filename</i> argument, the prs command applies to all <i>s.</i> files in that directory. Unreadable <i>s.</i> files produce an error; processing continues with the next file (if any). The use of '-' as the <i>s.filename</i> argument indicates that the names of files are to be read from the standard input, one <i>s.</i> file per line.
オプション	In the absence of options, prs displays the delta table (version log). In the absence of -d, or -l, prs displays the entry for each delta indicated by the other options. -a Includes all deltas, including those marked as removed (see sccs-rmdel(1)). -e Requests information for all deltas created <i>earlier</i> than, and including, the delta indicated with -r or -c. -l Requests information for all deltas created <i>later</i> than, and including, the delta indicated with -r or -c. -cdate-time Either options -e or -l must be used with this option. -cdate-time displays information on the deltas checked in either prior to and including the date and time indicated by the <i>date-time</i> argument (option -e); or later than and including the date and time indicated (option -l). <i>date-time</i> takes the form: <pre>yy[mm[dd[hh[mm[ss]]]]]</pre> Units omitted from the indicated date and time default to their maximum possible values; that is -c7502 is equivalent to -c750228235959. Any number of non-numeric characters may separate the various 2 digit components. If white-space characters occur, the <i>date-time</i> specification must be quoted. Values of <i>yy</i> in the range 69–99 refer to the twentieth century. Values in the range of 00–68 refer to the twenty-first century. -ddataspec Produce a report according to the indicated data specification. <i>dataspec</i> consists of a (quoted) text string that includes embedded data keywords of the form: ':key:' (see Data Keywords, below). prs expands these keywords in the output it produces. To specify a TAB character in the output, use \t; to specify a NEWLINE in the output, use \n. -rsid Specifies the SCCS delta ID (SID) of the delta for which information is desired. If no SID is specified, the most recently created delta is used.
使用法	Usage of prs is described below.

Data Keywords

Data keywords specify which parts of an SCCS file are to be retrieved. All parts of an SCCS file (see [sccsfile\(4\)](#)) have an associated data keyword. A data keyword may appear any number of times in a data specification argument to `-d`. These data keywords are listed in the table below:

<i>Keyword</i>	<i>Data Item</i>	<i>File Section*</i>	<i>Value</i>	<i>Format**</i>
:A:	a format for the what string:	N/A	:Z::Y: :M: :I::Z:	S
:B:	branch number	D	<i>nnnn</i>	S
:BD:	body	B	<i>text</i>	M
:BF:	branch flag	F	yes or no	S
:CB:	ceiling boundary	F	:R:	S
:C:	comments for delta	D	<i>text</i>	M
:D:	date delta created	D	:Dy:/:Dm:/:Dd:	S
:Dd:	day delta created	D	<i>nn</i>	S
:Dg:	deltas ignored (seq #)	D	:DS: :DS: ...	S
:DI:	seq-no. of deltas included, excluded, ignored	D	:Dn:/:Dx:/:Dg:	S
:DL:	delta line statistics	D	:Li:/:Ld:/:Lu:	S
:Dm:	month delta created	D	<i>nn</i>	S
:Dn:	deltas included (seq #)	D	:DS: :DS: ...	S
:DP:	predecessor delta seq-no.	D	<i>nnnn</i>	S
:Ds:	default SID	F	:I:	S
:DS:	delta sequence number	D	<i>nnnn</i>	S
:Dt:	delta information	D	:DT: :I: :D: :T: :P: :DS: :DP:	S
:DT:	delta type	D	D or R	S
:Dx:	deltas excluded (seq #)	D	:DS: ...	S
:Dy:	year delta created	D	<i>nn</i>	S
:F:	s. file name	N/A	<i>text</i>	S
:FB:	floor boundary	F	:R:	S
:FD:	file descriptive text	C	<i>text</i>	M

<i>Keyword</i>	<i>Data Item</i>	<i>File Section*</i>	<i>Value</i>	<i>Format**</i>
:FL:	flag list	F	text	M
:GB:	gotten body	B	text	M
:I:	SCCS delta ID (SID)	D	:R: . . L: . . B: . . S:	S
:J:	joint edit flag	F	yes or no	S
:KF:	keyword error/warning flag	F	yes or no	S
:L:	level number	D	nnnn	S
:Ld:	lines deleted by delta	D	nnnnn	S
:Li:	lines inserted by delta	D	nnnnn	S
:LK:	locked releases	F	:R: . . .	S
:Lu:	lines unchanged by delta	D	nnnnn	S
:M:	module name	F	text	S
:MF:	MR validation flag	F	yes or no	S
:MP:	MR validation program	F	text	S
:MR:	MR numbers for delta	D	text	M
:ND:	null delta flag	F	yes or no	S
:Q:	user defined keyword	F	text	S
:P:	user who created delta	D	username	S
:PN:	s . file's pathname	N/A	text	S
:R:	release number	D	nnnn	S
:S:	sequence number	D	nnnn	S
:T:	time delta created	D	:Th: . . :Tm: . . :Ts:	S
:Th:	hour delta created	D	nn	S
:Tm:	minutes delta created	D	nn	S
:Ts:	seconds delta created	D	nn	S
:UN:	user names	U	text	M
:W:	a form of what string	N/A	:Z: :M: \t: I:	S
:Y:	module type flag	F	text	S
:Z:	what string delimiter	N/A	@(#)	S

*B = body, D = delta table, F = flags, U = user names

**S = simple format, M = multi-line format

使用例

例 1 Displaying delta entries

The following command displays delta entries:

example% **prs -e -d":I:t:P:" program.c**

produces:

1.6 username
1.5 username...

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of prs: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

ファイル

/tmp/pr????? temporary file

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[sccs\(1\)](#), [sccs-cdc\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-prt\(1\)](#), [sccs-sact\(1\)](#), [sccs-sccsdiff\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断

Use the `sccs-help` command for explanations of SCCS commands. See [sccs-help\(1\)](#).

名前	sccs-prt, prt – display delta table information from an SCCS file																								
形式	prt [-abdefistu] [-cdate-time] [-rdate-time] [-ysid] s.filename...																								
機能説明	prt prints selected portions of an SCCS file. By default, it prints the delta table (version log). If a directory name is used in place of the <i>s.filename</i> argument, the prt command applies to all <i>s.</i> files in that directory. Unreadable <i>s.</i> files produce an error; processing continues with the next file (if any). The use of ‘-’ as the <i>s.filename</i> argument indicates that the names of files are to be read from the standard input, one <i>s.</i> file per line.																								
オプション	If any option other than -y, -c, or -r is supplied, the name of each file being processed (preceded by one NEWLINE and followed by two NEWLINE characters) appears above its contents. If none of the -u, -f, -t, or -b options are used, -d is assumed. -s, -i are mutually exclusive, as are -c and -r. <table> <tr> <td>-a</td><td>Display log entries for all deltas, including those marked as removed.</td></tr> <tr> <td>-b</td><td>Print the body of the <i>s.</i> file.</td></tr> <tr> <td>-d</td><td>Print delta table entries. This is the default.</td></tr> <tr> <td>-e</td><td>Everything. This option implies -d, -i, -u, -f, and -t.</td></tr> <tr> <td>-f</td><td>Print the flags of each named <i>s.</i> file.</td></tr> <tr> <td>-i</td><td>Print the serial numbers of included, excluded, and ignored deltas.</td></tr> <tr> <td>-s</td><td>Print only the first line of the delta table entries; that is, only up to the statistics.</td></tr> <tr> <td>-t</td><td>Print the descriptive text contained in the <i>s.</i> file.</td></tr> <tr> <td>-u</td><td>Print the user-names and/or numerical group IDs of users allowed to make deltas.</td></tr> <tr> <td>-cdate-time</td><td>Exclude delta table entries that are specified cutoff date and time. Each entry is printed as a single line, preceded by the name of the SCCS file. This format (also produced by -r, and -y) makes it easy to sort multiple delta tables in chronological order. When both -y and -c, or -y and -r are supplied, prt stops printing when the first of the two conditions is met.</td></tr> <tr> <td>-rdate-time</td><td>Exclude delta table entries that are newer than the specified cutoff date and time.</td></tr> <tr> <td>-ysid</td><td>Exclude delta table entries made prior to the SID specified. If no delta in the table has the specified SID, the entire table is printed. If no SID is specified, the most recent delta is printed.</td></tr> </table>	-a	Display log entries for all deltas, including those marked as removed.	-b	Print the body of the <i>s.</i> file.	-d	Print delta table entries. This is the default.	-e	Everything. This option implies -d, -i, -u, -f, and -t.	-f	Print the flags of each named <i>s.</i> file.	-i	Print the serial numbers of included, excluded, and ignored deltas.	-s	Print only the first line of the delta table entries; that is, only up to the statistics.	-t	Print the descriptive text contained in the <i>s.</i> file.	-u	Print the user-names and/or numerical group IDs of users allowed to make deltas.	-cdate-time	Exclude delta table entries that are specified cutoff date and time. Each entry is printed as a single line, preceded by the name of the SCCS file. This format (also produced by -r, and -y) makes it easy to sort multiple delta tables in chronological order. When both -y and -c, or -y and -r are supplied, prt stops printing when the first of the two conditions is met.	-rdate-time	Exclude delta table entries that are newer than the specified cutoff date and time.	-ysid	Exclude delta table entries made prior to the SID specified. If no delta in the table has the specified SID, the entire table is printed. If no SID is specified, the most recent delta is printed.
-a	Display log entries for all deltas, including those marked as removed.																								
-b	Print the body of the <i>s.</i> file.																								
-d	Print delta table entries. This is the default.																								
-e	Everything. This option implies -d, -i, -u, -f, and -t.																								
-f	Print the flags of each named <i>s.</i> file.																								
-i	Print the serial numbers of included, excluded, and ignored deltas.																								
-s	Print only the first line of the delta table entries; that is, only up to the statistics.																								
-t	Print the descriptive text contained in the <i>s.</i> file.																								
-u	Print the user-names and/or numerical group IDs of users allowed to make deltas.																								
-cdate-time	Exclude delta table entries that are specified cutoff date and time. Each entry is printed as a single line, preceded by the name of the SCCS file. This format (also produced by -r, and -y) makes it easy to sort multiple delta tables in chronological order. When both -y and -c, or -y and -r are supplied, prt stops printing when the first of the two conditions is met.																								
-rdate-time	Exclude delta table entries that are newer than the specified cutoff date and time.																								
-ysid	Exclude delta table entries made prior to the SID specified. If no delta in the table has the specified SID, the entire table is printed. If no SID is specified, the most recent delta is printed.																								

使用法

Output Format

The following format is used to print those portions of the `s` file that are specified by the various options.

- NEWLINE
- Type of delta (D or R)
- SPACE
- SCCS delta ID (SID)
- TAB
- Date and time of creation in the form: *yy/mm/dd hh/mm/ss*
- SPACE
- Username the delta's creator
- TAB
- Serial number of the delta
- SPACE
- Predecessor delta's serial number
- TAB
- Line-by-line change statistics in the form: *inserted/deleted/unchanged*
- NEWLINE
- List of included deltas, followed by a NEWLINE (only if there were any such deltas and the `-i` options was used)
- List of excluded deltas, followed by a NEWLINE (only if there were any such deltas and the `-i` options was used)
- List of ignored deltas, followed by a NEWLINE (only if there were any such deltas and the `-i` options was used)
- List of modification requests (MRs), followed by a NEWLINE (only if any MR numbers were supplied).
- Lines of the delta commentary (if any), followed by a NEWLINE.

使用例

例 1 Producing a Display of the Delta Table

The following command produces a one-line display of the delta table entry for the most recent version:

```
example% prt -y program.c
s.program.c: D 1.6   88/07/06 21:39:39 username   5 4 00159/00080/00636
```


属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

関連項目 [sccs\(1\)](#), [sccs-cdc\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-prs\(1\)](#),
[sccs-sact\(1\)](#), [sccs-sccsdiff\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#)

診断 Use the `sccs -help` command for explanations of SCCS commands. See [sccs-help\(1\)](#).

名前	sccs-rmdel, rmdel – remove a delta from an SCCS file								
形式	rmdel -rsid s.filename...								
機能説明	The rmdel utility removes the delta specified by the SCCS delta ID (SID) supplied with -r. The delta to be removed must be the most recent (leaf) delta in its branch. In addition, the SID must <i>not</i> be that of a version checked out for editing: it must not appear in any entry of the version lock file (p.file). If you created the delta, or, if you own the file and directory and have write permission, you can remove it with rmdel. If a directory name is used in place of the s.filename argument, the rmdel command applies to all s.files in that directory. Unreadable s.files produce an error; processing continues with the next file (if any). The use of '-' as the s.filename argument indicates that the names of files are to be read from the standard input, one s.file per line.								
オプション	The following option is supported: -rsid Remove the version corresponding to the indicated SID (delta).								
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of rmdel: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.								
ファイル	p.file permissions file s.file history file z.file temporary copy of the s.file								
属性	See attributes(5) for descriptions of the following attributes:								
	<table><tr><th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr><tr><td>Availability</td><td>developer/build/make</td></tr><tr><td>Interface Stability</td><td>Committed</td></tr><tr><td>Standard</td><td>See standards(5).</td></tr></table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	developer/build/make	Interface Stability	Committed	Standard	See standards(5) .
ATTRIBUTE TYPE	ATTRIBUTE VALUE								
Availability	developer/build/make								
Interface Stability	Committed								
Standard	See standards(5) .								
関連項目	sccs(1) , sccs-admin(1) , sccs-cdc(1) , sccs-comb(1) , sccs-delta(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , sccs-sccsdiff(1) , sccs-unget(1) , what(1) , sccsfile(4) , attributes(5) , environ(5) , standards(5)								
診断	Use the SCCS help command for explanations (see sccs-help(1)).								

名前 sccs-sact, sact – show editing activity status of an SCCS file

形式 sact *s.filename*...

機能説明 The sact utility informs the user of any SCCS files that are checked out for editing.

The output for each named file consists of five fields separated by SPACE characters.

- SID of a delta that currently exists in the SCCS file, to which changes are made to make the new delta
- SID for the new delta to be created
- Username of the person who has the file checked out for editing.
- Date that the version was checked out.
- Time that the version was checked out.

If a directory name is used in place of the *s.filename* argument, the sact command applies to all *s.* files in that directory. Unreadable *s.* files produce an error; processing continues with the next file (if any). The use of '-' as the *s.filename* argument indicates that the names of files are to be read from the standard input, one *s.* file per line.

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of sact: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [sccs\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [sccs-prs\(1\)](#), [sccs-prt\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 Use the sccs-help command for explanations of SCCS commands. See [sccs-help\(1\)](#).

名前	sccs-sccsdiff, sccsdiff – compare two versions of an SCCS file				
形式	sccsdiff [-p] -rsid -rsid [diff-options] s.filename				
機能説明	sccsdiff compares two versions of an SCCS file and displays the differences between the two versions. Any number of SCCS files can be specified. The options specified apply to all named s . files.				
オプション	The following options are supported: -p Pipe output for each file through pr(1). -rsid Specify a version corresponding to the indicated SCCS delta ID (SID) for comparison. Versions are passed to diff(1) in the order given. diff-options Pass options to diff(1), including: -b, -c, -e, -f, -h, -u, -C number, -U number, and -D string.				
ファイル	/tmp/get????? temporary files				
属性	See attributes(5) for descriptions of the following attributes:				
	<table><tr><th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr><tr><td>Availability</td><td>developer/build/make</td></tr></table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	developer/build/make
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	developer/build/make				
関連項目	diff(1) , sccs(1) , sccs-delta(1) , sccs-get(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , what(1) , sccsfile(4) , attributes(5)				
診断	filename: No differences If the two versions are the same. Use the sccs-help command for explanations of SCCS commands. See sccs-help(1).				

名前	sccs-unget, unget – undo a previous get of an SCCS file								
形式	unget [-ns] [-rsid] <i>s.filename</i> ...								
機能説明	The unget utility undoes the effect of a get -e command executed before the creation of the pending delta. If a directory name is used in place of the <i>s.filename</i> argument, the unget command applies to all <i>s.</i> files in that directory. Unreadable <i>s.</i> files produce an error; processing continues with the next file (if any). The use of '-' as the <i>s.filename</i> argument indicates that the names of files are to be read from the standard input, one <i>s.</i> file per line.								
オプション	The following options are supported: -n Retains the retrieved version, which is otherwise removed. -s Suppress display of the SCCS delta ID (SID). -rsid When multiple versions are checked out, this option specifies which pending delta to abort. A diagnostic results if the specified SID is ambiguous, or if it is necessary but omitted from the command line.								
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of unget: LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_MESSAGES, and NLSPATH.								
属性	See attributes(5) for descriptions of the following attributes:								
	<table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>developer/build/make</td></tr> <tr> <td>Interface Stability</td><td>Committed</td></tr> <tr> <td>Standard</td><td>See standards(5).</td></tr> </tbody> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	developer/build/make	Interface Stability	Committed	Standard	See standards(5) .
ATTRIBUTE TYPE	ATTRIBUTE VALUE								
Availability	developer/build/make								
Interface Stability	Committed								
Standard	See standards(5) .								
関連項目	sccs(1) , sccs-delta(1) , sccs-get(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , sccs-rmdel(1) , sccs-sact(1) , sccs-sccsdiff(1) , what(1) , sccsfile(4) , attributes(5) , environ(5) , standards(5)								
診断	Use the sccs-help command for explanations of SCCS commands. See sccs-help(1) .								

名前	sccs-val, val – validate an SCCS file
形式	val [-s] [-m <i>name</i>] [-rsid] [-y <i>type</i>] <i>s.filename</i> ...
機能説明	The val utility determines if the specified <i>s</i> .files meet the characteristics specified by the indicated arguments. val can process up to 50 files on a single command line. val has a special argument, ‘-’, which reads the standard input until the end-of-file condition is detected. Each line read is independently processed as if it were a command line argument list. val generates diagnostic messages on the standard output for each command line and file processed and also returns a single 8–bit code upon exit as described below. The 8-bit code returned by val is a disjunction of the possible errors, that is, it can be interpreted as a bit string where (moving from left to right) the bits set are interpreted as follows: - bit 0 = missing file argument - bit 1 = unknown or duplicate option - bit 2 = corrupted <i>s</i>.file - bit 3 = can not open file or file not in <i>s</i>.file format - bit 4 = the SCCS delta ID (SID) is invalid or ambiguous - bit 5 = the SID does not exist - bit 6 = mismatch between <i>Y</i>% and -y argument - bit 7 = mismatch between sccs-val.1 and -m argument val can process two or more files on a given command line, and in turn can process multiple command lines (when reading the standard input). In these cases, an aggregate code is returned which is the logical OR of the codes generated for each command line and file processed.
オプション	The following options are supported: -s Silent. Suppresses the normal error or warning messages. -m <i>name</i> Compares <i>name</i> with the %M% ID keyword in the <i>s</i> . file. -rsid Checks to see if the indicated SID is ambiguous, invalid, or absent from the <i>s</i> . file. -y <i>type</i> Compares <i>type</i> with the %Y% ID keyword.
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of val: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
Standard	See standards(5) .

- 関連項目
- [sccs\(1\)](#), [sccs-admin\(1\)](#), [sccs-delta\(1\)](#), [sccs-get\(1\)](#), [sccs-help\(1\)](#), [what\(1\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)
- 診断
- Use the `sccs-help` command for explanations of SCCS commands. See [sccs-help\(1\)](#).

名前	scp - セキュリティー保護されたコピー (リモートファイルのコピープログラム)
形式	<pre> scp [-pqrvc46] [-F ssh_config] [-S program] [-P port] [-c cipher] [-i identity_file] [-o ssh_option] [[user@]host1:]file1 [...] [[user@]host2:]file2 </pre>
機能説明	scp ユーティリティーは、ネットワーク上のホスト間でファイルをコピーします。このユーティリティーは、ssh(1) をデータ転送に使用し、ssh(1) と同じ認証を使用して同じセキュリティを提供します。rcp(1) とは異なり、scp は、認証に必要な場合はパスワードまたはパスフレーズを要求します。 ファイル名にホスト指定とユーザー指定を含めて、そのファイルを該当のホストにコピーしたりホストからコピーしたりする必要があることを示すことができます。2つのリモートホスト間のコピーが許可されます。
オプション	サポートしているオプションは、次のとおりです。 -4 強制的に scp が IPv4 アドレスのみを使用するようにします。 -6 強制的に scp が IPv6 アドレスのみを使用するようにします。 -B バッチモードを選択します。(パスワードまたはパスフレーズを要求しないようにします。) -c cipher データ転送の暗号化に使用する暗号化方式を選択します。このオプションは、直接 ssh(1) に渡されます。 -C 圧縮を有効にします。-C フラグを ssh(1) に渡して、圧縮を有効にします。 -F ssh_config ssh(1.) のユーザーごとの代替構成ファイルを指定します。 -i identity_file RSA 認証用の識別情報 (秘密鍵) を読み取るファイルを選択します。このオプションは、ssh(1) に直接渡されます。 -o ssh_option 指定したオプションは、直接 ssh(1) に渡されます。 -p 元のファイルの変更時間、アクセス時間、およびモードを保存します。 -P port リモートホストの接続先ポートを指定します。このオプションは、大文字の「P」で記述されます。これは、-p がすでに rcp(1) でファイルの時間とモードを保存するために予約されているためです。 -q 進行状況メーターを無効にします。 -r ディレクトリ全体を再帰的にコピーします。 -S program 暗号化した接続で使用するプログラムの名前を指定します。このプログラムは、ssh(1) のオプションを認識する必要があります。

-v 冗長モード。`scp` および `ssh(1)` が進行状況に関するデバッグメッセージを出力するようにします。これは、接続、認証、および構成の問題をデバッグするときに役立ちます。

オペランド 次のオペランドがサポートされています。

host1、*host2*、... ファイルのコピー元またはコピー先のホストの名前。

file1、*file2*、... コピーするファイル。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目 [rcp\(1\)](#), [ssh\(1\)](#), [ssh-add\(1\)](#), [ssh-agent\(1\)](#), [ssh-keygen\(1\)](#), [sshd\(1M\)](#), [ssh_config\(4\)](#), [attributes\(5\)](#)

注意事項 一般に、`scp` でパスワードまたはキーボード対話型認証方式と2つのリモートホストを使用しても機能しません。`scp` で使用できるのは、`pubkey`、`hostbased`、または `gssapi-keyex` 認証方式のいずれかです。`pubkey` 認証方式の場合は、パスフレーズで保護されていない秘密鍵か、明示的な `ssh` エージェント転送のいずれかを使用する必要があります。`gssapi-keyex` 認証方式が `kerberos_v5` GSS-API メカニズムで機能するのは、`GSSAPIDelegateCredentials` オプションが有効になっている場合だけです。

名前	script – 端末セッションの記録の生成
形式	script [-a] [filename]
機能説明	script は、そのユーザーの画面上で映し出される全出力の 記録を生成します。記録は、<i>filename</i> 引数で指定されたファイルに書き込まれます。ファイル名を省略すると、記録は <i>typescript</i> という名のファイルに保存されます。「警告」を参照してください。 script コマンドは、\$SHELL の値に従ってサブシェルをフォークしたり作成したり、このセッションからのテキストを記録します。script が終了するのは、フォークされたシェルが終了したとき、または CTRL-D が入力されたときです。
オプション	次のオプションがサポートされています。 -a セッションの記録を、<i>filename</i> の最後尾に追加します。上書きはしません。
注意事項	script は、端末の画面上に現れるすべての出力を記録します。これにはプロンプトも含まれます。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目	attributes(5)
警告	ほかのユーザーが書き込み可能なディレクトリ (たとえば、/tmp) 内で script を使用すると、セキュリティリスクをもたらす可能性があります。特に、特権付きユーザー (root) が実行する場合は危険です。script を実行する前に、typescript がリンクでないことを確認してください。

名前	sdiff - 2つのファイル間の違いを並べて出力
形式	sdiff [-l] [-s] [-o output] [-w n] filename1 filename2
機能説明	sdiff は、diff コマンドの出力を使用して、2つのファイルの並列リストを作成して、異なっている行を示します。2つのファイルの行が同じ場合、これらの行は間に空白を入れて出力されます。また、この行がfilename1 にしかない場合には< が、この行がfilename2 にしかない場合には> が、またこれらの行が異なる場合には が1 つ間に入ります(「使用例」を参照)。
オプション	-l 同一の任意の行の左側だけを出力します。 -s 同一行を出力しません。 -o output 引数 output を filename1 と filename2 のユーザー制御のマージとして作成された第3 ファイルの名前として使用します。filename1 と filename2 の同一行が output にコピーされます。diff が処理した結果を差異の種類によりグループ分けし、出力します。このとき、この結果は共通のガーチャラクタを共有して表示されます。それぞれの異なるセットを出力した後、sdiff はユーザーに%のプロンプトを出し、ユーザーが次のいずれかのコマンドをタイプするのを待ちます。 l 出力ファイルに左側のカラムを追加 r 出力ファイルに右側のカラムを追加 s サイレントモードにし、同一行を出力しない v サイレントモードを終了 e l 左カラムでのエディタの呼び出し e r 右カラムでのエディタの呼び出し e b 左右カラムの連結でのエディタの呼び出し e 長さ 0 のファイルでのエディタの呼び出し q プログラムを終了 エディタの終了時に、結果のファイルを output ファイルの最後に連結します。 -w n 引数 n を出力行の幅として使用。デフォルトの行長は、130 文字です。
使用法	ファイルが2G バイト (2 ³¹ バイト) 以上ある場合の sdiff の動作については、 largefile(5) を参照してください。
使用例	例1 sdiff コマンドの例 sdiff の出力例は、次のとおりです。

例1 sdiff コマンドの例 (続き)

```
x | y
a   a
b <
c <
d   d
   > c
```

環境変数

LC_* 変数 (LC_CTYPE、LC_MESSAGES、LC_TIME、LC_COLLATE、LC_NUMERIC、および LC_MONETARY) ([environ\(5\)](#)) を参照) のいずれも環境に設定されていなければ、それぞれ対応するロケールのカテゴリにおける passwd の動作は、環境変数 LANG によって決定されます。LC_ALL が設定されていれば、その内容が LANG 変数やその他の LC_* 変数より優先されます。前述の変数が環境にまったく設定されていなければ、C ロケールが sdiff の動作を決定します。

LC_CTYPE sdiff の文字の処理方法を決定します。LC_CTYPE に有効な値が設定されていると、sdiff は、そのロケールにあった文字を含むテキストやファイル名を表示および処理できます。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目

[diff\(1\)](#), [ed\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#)

名前	sed - ストリームエディタ
形式	<pre> /usr/bin/sed [-n] script [file]... /usr/bin/sed [-n] [-e script]... [-f script_file]... [file]... /usr/xpg4/bin/sed [-n] script [file]... /usr/xpg4/bin/sed [-n] [-e script]... [-f script_file]... [file]...</pre>
機能説明	sed ユーティリティーは、テキストファイルを読み込み、編集コマンドのスク립トに従って内容を編集し、結果を標準出力に書き出すストリームエディタです。スク립トは、<i>script</i> オペランド文字列か、<i>-e script</i> と <i>-f script_file</i> のオプション引数の組み合わせで指定します。 sed ユーティリティーはテキストエディタですが、バイナリファイルや、ASCII NUL (\0) 文字を含んだファイル、長い行を含んだファイルは編集できません。
オプション	サポートしているオプションは、次のとおりです。 <i>-e script</i> <i>script</i> は、sed の編集コマンドです。<i>script</i> のフォーマットの詳細については、「使用法」を参照してください。<i>-e</i> オプションを1つだけ指定し、<i>-f</i> オプションを指定しない場合、フラグ <i>-e</i> は省略できます。 <i>-f script_file</i> <i>script_file</i> に指定したファイルからスク립トを読み取ります。このファイルの各行には、編集コマンドが1つずつ記述されています。 <i>-n</i> デフォルトの出力を抑止します。 <i>-e</i> と <i>-f</i> の両オプションは、それぞれ複数個指定できます。コマンドはすべて、どのオプションで指定されたかに関係なく、指定の順序でスク립トに追加されます。
オペランド	次のオペランドがサポートされています。 <i>file</i> 内容を読み込んで編集するファイルのパス名。<i>file</i> オペランドを複数個指定すると、指定した順序で読み込まれ、連結されたファイルを編集することになります。<i>file</i> を指定しないと、標準入力を読み込まれます。 <i>script</i> 編集コマンドのスク립トとして使用する文字列。アプリケーション側で、テキストファイルの規約に反するような <i>script</i> を指定することはできませんが、文字列の最後の文字が復帰改行文字でなくてもかまいません。
使用法	スク립トは、複数の編集コマンドで構成されます。スク립ト内の各行に、編集コマンドを1つずつ記述します。次のような書式を使用します。 <pre>[address [, address]] command [arguments]</pre>

最初のアドレス (*address*) の前と、コマンド (*command*) の前には、任意の数の空白文字を入力できます。また、最初のアドレスの前には、任意の数のセミコロンも入力できます。

通常、**sed** は、次の動作を繰り返します。まず、**d** コマンドの実行後に何も残っていないことを確認し、入力行の 1 行を「パターンスペース」にコピーします。このとき、行末の復帰改行文字は含まれません。さらに、このパターンスペースを選択するアドレスを持つコマンドがあれば、そのコマンドをすべてパターンスペースに適用します。次に、その結果を標準出力にコピーし、パターンスペースを削除します。ただし、**-n** が指定されているときは、この処理は行いません。このようにしてパターンスペースを標準出力または指定ファイルに書き出すたびに、**sed** はその後復帰改行文字を付け加えます。

コマンドのいくつかは、以前に取り出したパターンスペースを取り出しやすくするために、その一部または全部を「ホールドスペース」に保存します。パターンスペースとホールドスペースには、それぞれ最大 8192 バイトのデータを記録できます。

sed のアドレス

address には、10 進数 (すべてのファイルの入力行を累積的にカウントする)、**\$** (入力の最終行をアドレス指定する)、コンテキストアドレスのほか、空 (区切りのみ) を指定できます。*address* は、[regexp\(5\)](#) のマニュアルページで解説しているように、スラッシュで囲んだ正規表現 (*/regular expression/*) で構成されます。

アドレスのないコマンド行は、すべてのパターンスペースを選択します。

アドレスを 1 つ持つコマンド行は、そのアドレスと一致するパターンスペースを選択します。

アドレスを 2 つ持つコマンド行は、1 つめのアドレスと一致する最初のパターンスペースから 2 つめのアドレスと一致する次のパターンスペースまでの範囲を選択します。その後プロセスが繰り返され、ふたたび 1 つ目のアドレスを探します。(2 つめのアドレスが、1 つめのアドレスで選択した行番号以下の場合、1 つめのアドレスに対応する行だけが選択されます)。

通常、アドレスとアドレスの間はコンマ (,) で区切りますが、セミコロン (;) を使うこともできます。

sed の正規表現

sed は、[regexp\(5\)](#) のマニュアルページで説明されている正規表現のほか、次の指定をサポートしています。

\cREc コンテキストアドレスでは、**\cREc** は **/RE/** と同じ意味になります (**c** はバックスラッシュと復帰改行文字を除く任意の文字。**RE** は正規表現)。**c** に指定した文字がバックスラッシュの直後に現れた場合、これは正規表現の終了文字ではなく、リテラル文字と見なされます。たとえば、**\xabc\xdefx** というコンテキストアドレスでは、2 つめの **x** はリテラル文字 **x** です。この場合、正規表現は **abcxdef** になります。

`\n` エスケープシーケンス `\n` は、パターンスペースに埋め込まれた復帰改行文字と一致します。コンテキストアドレスの正規表現内や置換コマンド内では、復帰改行文字をリテラル文字として処理できません。

否定コマンド `!` を使用すると、編集コマンドは、選択されていないパターンスペースに適用されます (下記参照)。

sed の編集コマンド 以下の表に、各機能に指定できるアドレスの最大数を一覧します。

`r` コマンドと `w` コマンドには、それぞれ *rfile* と *wfile* という任意指定のパラメータがあります。コマンド文字とパラメータの間は、1 つ以上の空白文字で区切ります。

複数のコマンドをセミコロン (`;`) で区切って、1 つのコマンド行に記述できます。

引数 *text* は、行で構成されます。この引数が複数の行で構成されている場合、最終行以外のすべての行は `\` で終了し、復帰改行をエスケープします。復帰改行文字をテキストに埋め込む場合は、その前にバックスラッシュを付加してください。テキストに埋め込まれたその他のバックスラッシュは削除され、その直後の文字がリテラル文字として処理されます。つまり、テキストに埋め込まれたバックスラッシュは、`s` コマンドの置換文字列中のバックスラッシュと同様に処理されることとなります。また、バックスラッシュを使用して、スクリプト行の行頭の空白文字やタブ文字が削除されるのを防ぐこともできます。*rfile* 引数や *wfile* 引数は、その前に空白文字を 1 つ入力して、コマンド行の末尾に置いてください。*wfile* パラメータを指定すると、そのファイルが作成されます。すでにファイルが存在していれば、その内容が書き換えられます。最大 10 個の *wfile* 引数を指定できます。

正規表現のマッチングは、行単位ではなく、文字列全体で行われます。ただし、sed の正規表現では、復帰改行文字は `\n` と一致します。このため、正規表現内では復帰改行文字は使用できません。また、入力行末尾の復帰改行文字のマッチングに、`\n` は使用できません。パターンスペースには、復帰改行文字は編集コマンド `N` の結果として現れます。

コマンドのうち 2 つは *command-list* の形式をとります。これは復帰改行で区切られたいくつものコマンドの集まりです。以下にその形式を示します。

```
{ command
command
}
```

{ の前後にはスペース文字を置くことができます。また、*command* の前にもスペースを置くことができます。終端の } の前には `NEWLINE` 文字を置く必要があります、`<blank>` を前後に置くことができます。さらに、中括弧の前後にも空白文字を置くことができます。コマンドの前にも空白文字を置くことができますが、後には置けません。

次の表に、機能と、許容されるアドレスの最大数を一覧表示します。

最大アドレス数	コマンド	説明
1	<code>a \text</code>	N コマンド実行または新サイクル開始による追加。次の入力行を読み取る前に <i>text</i> を出力します。
2	<code>b label</code>	<i>label</i> を持つ ; コマンドに分岐します。 <i>label</i> が空の場合は、スクリプトの最後に分岐します。ラベルは最大 8 文字が一意であると認識されます。
2	<code>c \text</code>	変更。パターンスペースを削除します。 <i>text</i> を出力します。次のサイクルを起動します
2	<code>d</code>	パターンスペースを削除します。次のサイクルを起動します
2	<code>D</code>	パターンスペースの先頭セグメントから最初の復帰改行までを削除します。次のサイクルを起動します (下記の N コマンドを参照)。
2	<code>g</code>	パターンスペースの内容をホールドスペースの内容で置換します。
2	<code>G</code>	ホールドスペースの内容をパターンスペースに追加します。
2	<code>h</code>	ホールドスペースの内容をパターンスペースの内容で置換します。
2	<code>H</code>	パターンスペースの内容をホールドスペースに追加します。
1	<code>i \text</code>	挿入。 <i>text</i> を標準出力に出力します。
2	<code>l</code>	<code>/usr/bin/sed</code> : 明白な形式でパターンスペースを標準出力に一覧表示します。非印字文字は 8 進数表記法で表示されます。長い行は複数行に分けて表示されます。
		<code>/usr/xpg4/bin/sed</code>: 明白な形式でパターンスペースを標準出力に一覧表示します。非印字文字は 8 進数表記法で表示されます。長い行は複数行に分けて表示されます。文字 <code>\a</code>、<code>\b</code>、<code>\f</code>、<code>\r</code>、<code>\t</code>、<code>\v</code> が対応するエスケープシーケンスとして出力されます。テーブル中にある非印字文字に関しては、各バイトごと (最上位ビットから) に 3 桁の 8 進数で、前にバックスラッシュが付加された形式で出力されます。1 バイトが 10 ビット以上のシステムでは、非印字文字のフォーマットは導入先により異なります。 長い行は折り返され、折り返しの発生地点はバックスラッシュと復帰改行文字によって示されます。折り返し地点の長さは不特定ですが、出力装置に適した値になっています。各行の終わりは <code>\$</code> でマークされます。

最大アドレス数	コマンド	説明
2	<code>n</code>	デフォルト出力が抑止されていなければ、パターンスペースを標準出力にコピーします。パターンスペースを入力のある行で置換します。
2	<code>N</code>	入力のある行をパターンスペースに追加し、復帰改行文字を埋め込みます。(現在の行番号が変わります)。入力に次の行がなければ、 <code>N</code> コマンド動詞はスクリプトの終わりに分岐し、新たなサイクルを開始せずに、パターンスペースを書き出さずに終了します。
2	<code>p</code>	出力。パターンスペースを標準出力にコピーします。
2	<code>P</code>	パターンスペースの先頭セグメントから最初の復帰改行までを標準出力にコピーします。
1	<code>q</code>	終了。スクリプトの最後に分岐します。新しいサイクルを起動しません。
2	<code>r <i>rfile</i></code>	<i>rfile</i> の内容を読み取ります。次の入力行を読み取る前にこの内容を出力します。 <i>rfile</i> が存在しないまたは読み取れないときは、エラーとはならず、あたかも空のファイルであるかのように扱われます。
2	<code>t <i>label</i></code>	テスト。入力行の最新の読み取りまたは <code>t</code> の実行以降に代入が行われている場合は、 <i>label</i> を持つ <code>:</code> コマンドに分岐します。 <i>label</i> が空の場合は、スクリプトの最後に分岐します。
2	<code>w <i>wfile</i></code>	書き込み。 <i>wfile</i> にパターンスペースを追加します。最初の <code>w</code> が発生すると、 <i>wfile</i> がクリアされます。後続の <code>w</code> が呼び出されると、追加されます。 <code>sed</code> コマンドが使用されるごとに、 <i>wfile</i> が上書きされます。
2	<code>x</code>	パターンスペースおよびホールドスペースの内容を交換します。
2	<code>! <i>command</i></code>	否定。アドレスによって選択されなかった行だけに、 <i>command</i> (または、 <i>command</i> が <code>{</code> の場合はグループ) を適用します。
0	<code>: <i>label</i></code>	このコマンドは何もしません。これは分岐するための <code>b</code> および <code>t</code> コマンド用の <i>label</i> を持っています。
1	<code>=</code>	現在の行番号を 1 行として標準出力します。
2	<code>{<i>command-list</i>}</code>	パターンスペースが選択された場合のみ、 <i>command-list</i> を実行します。
0		空のコマンドは無視されます。

最大アドレス数	コマンド	説明
0	#	スクリプトファイルの行の先頭文字が#の場合、その行全体が注釈行として扱われます。ただし、先頭行に#があり、#の後の文字がnの場合は唯一の例外で、デフォルトの出力が抑止されます。#nの後の残りの行も無視されます。スクリプトファイルには1つ以上の非注釈行が必要です。

最大アドレス数	コマンド (strings を使用) と説明
2	<i>s/regular expression/replacement/flags</i> パターンスペース内の <i>regular expression</i> に一致する文字列を <i>replacement</i> に置換します。正規表現 (<i>regular expression</i>) と置換文字列 (<i>replacement</i>) を区切るのに、スラッシュの代わりに、バックスラッシュと復帰改行以外の任意の文字を使用できます。正規表現および置換文字列内で、区切り文字を文字列の一部として使用したければ、前にバックスラッシュを付加してください。 置換文字列中のアンバサンド (&) は、正規表現に一致する文字列に置き換えられます。この&が持つ特殊な意味は、前にバックスラッシュを付加すれば無効にできます。<i>n</i> の前にバックスラッシュを付加した \n は、対応する後方参照表現と一致するテキストに置き換えられます。<i>replacement</i> を先頭から終端まで走査する際にバックスラッシュ (\) が検出されると、後続の文字は特殊な意味を持っていたとしてもその意味を失います。&、\、数字以外の文字にそれぞれどのような特殊な意味が与えられるかは、不特定です。 行は、その中に復帰改行文字を使用することによって分割できます。アプリケーションは、<i>replacement</i> 内の復帰改行文字の前にバックスラッシュを置くことによって、これをエスケープする必要があります。置換文字列が、置換後の文字列と同一である場合も、置換が行われたことになります。 <i>flags</i> には以下のものを、0 個以上指定できます。 <i>n n</i>=1 - 512。 <i>regular expression</i> の <i>n</i> 番目の発生を置換します。 <i>g</i> グローバル。 <i>regular expression</i> の最初に一致したものだけでなく、重ならないすべての例を置換します。<i>g</i> と <i>n</i> の両方を指定した場合、結果は不確定です。
	<i>p</i> 置換が行われた場合、パターンスペースを出力します。 <i>P</i> パターンスペースの先頭セグメントから最初の復帰改行までを、標準出力にコピーします。 <i>w wfile</i> 書き込み。置換が行われた場合は、<i>wfile</i> にパターンスペースを追加します。最初の <i>w</i> が発生すると、<i>wfile</i> がクリアされます。後続の <i>w</i> が呼び出されると、追加されます。<i>sed</i> コマンドが使用されるごとに、<i>wfile</i> が上書きされます。
2	<i>y/string1 / string2/</i>

最大アドレス数	コマンド (strings を使用) と説明
	変形。string1 にあるすべての文字を string2 の対応する文字と交換します。string1 と string2 の文字は同じ数でなければなりません。string1 中に同じ文字が複数回現れた場合の結果は定義されていません。2つの文字列を区切るのに、スラッシュの代わりに、バックスラッシュと復帰改行以外の任意の文字を使用できます。string1 とstring2内で、区切り文字を文字列の一部として使用したければ、前にバックスラッシュを付加してください。たとえば、y/abc/ABC/は、aをAで、bをBで、およびcをCで置き換えます。

ファイルが2G バイト (2³¹ バイト) 以上ある場合の sed の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 sed スクリプトの例

以下に示す sed スクリプトは、BSD cat -s コマンドをシミュレートし、標準入力から余分な空白行を圧縮します。

```
sed -n '
# Write non-empty lines.
./ {
    p
    d
}
# Write a single empty line, then look for more empty lines.
/^$/ p
# Get next line, discard the held <newline> (empty line),
# and look for more empty lines.
:Empty
/^$/ {
    N
    s/./ /
    b Empty
}
# Write the non-empty line before going back to search
# for the first in a set of empty lines.
p
,
```

環境変数

sed の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス

次の終了ステータスが返されます。

- 0 正常終了。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/sed

属性タイプ	属性値
使用条件	system/core-os
CSI	未対応

/usr/xpg4/bin/sed

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目 [awk\(1\)](#), [ed\(1\)](#), [grep\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [standards\(5\)](#)

名前	sed – stream editor										
形式	sed [-n] [-e <i>script</i>] [-f <i>sfilename</i>] [<i>filename</i>]. . .										
機能説明	The sed utility copies the <i>filenames</i> (standard input default) to the standard output, edited according to a script of commands.										
オプション	The following options are supported: -n Suppresses the default output. -e <i>script</i> <i>script</i> is an edit command for sed. If there is just one -e option and no -f options, the -e flag may be omitted. -f <i>sfilename</i> Takes the script from <i>sfilename</i>.										
使用法											
sed Scripts	sed scripts consist of editing commands, one per line, of the following form: <pre>[address [, address]] function [arguments]</pre> In normal operation, sed cyclically copies a line of input into a <i>pattern space</i> (unless there is something left after a D command), sequentially applies all commands with <i>addresses</i> matching that pattern space until reaching the end of the script, copies the pattern space to the standard output (except under -n), and finally, deletes the pattern space. Some commands use a <i>hold space</i> to save all or part of the pattern space for subsequent retrieval. An <i>address</i> is either: ■ a decimal number linecount, which is cumulative across input files; ■ a \$, which addresses the last input line; ■ or a context address, which is a <i>/regular expression/</i> as described on the regexp(5) manual page, with the following exceptions: <table> <tr> <td>\?RE?</td><td>In a context address, the construction \ <i>?regular expression?</i>, where <i>?</i> is any character, is identical to <i>/regular expression/</i>. Note: in the context address \xabc\xdefx, the second x stands for itself, so that the regular expression is abcxdef.</td></tr> <tr> <td>\n</td><td>Matches a NEWLINE embedded in the pattern space.</td></tr> <tr> <td>.</td><td>Matches any character except the NEWLINE ending the pattern space.</td></tr> <tr> <td><i>null</i></td><td>A command line with no address selects every pattern space.</td></tr> <tr> <td><i>address</i></td><td>Selects each pattern space that matches.</td></tr> </table>	\?RE?	In a context address, the construction \ <i>?regular expression?</i> , where <i>?</i> is any character, is identical to <i>/regular expression/</i> . Note: in the context address \xabc\xdefx, the second x stands for itself, so that the regular expression is abcxdef.	\n	Matches a NEWLINE embedded in the pattern space.	.	Matches any character except the NEWLINE ending the pattern space.	<i>null</i>	A command line with no address selects every pattern space.	<i>address</i>	Selects each pattern space that matches.
\?RE?	In a context address, the construction \ <i>?regular expression?</i> , where <i>?</i> is any character, is identical to <i>/regular expression/</i> . Note: in the context address \xabc\xdefx, the second x stands for itself, so that the regular expression is abcxdef.										
\n	Matches a NEWLINE embedded in the pattern space.										
.	Matches any character except the NEWLINE ending the pattern space.										
<i>null</i>	A command line with no address selects every pattern space.										
<i>address</i>	Selects each pattern space that matches.										

	<i>address1</i> , <i>address2</i>	Selects the inclusive range from the first pattern space matching <i>address1</i> to the first pattern space matching <i>address2</i> . Selects only one line if <i>address1</i> is greater than or equal to <i>address2</i> .																										
Comments	If the first nonwhite character in a line is a '#' (pound sign), sed treats that line as a comment, and ignores it. If, however, the first such line is of the form: #n sed runs as if the -n flag were specified.																											
Functions	The maximum number of permissible addresses for each function is indicated in parentheses in the list below. An argument denoted <i>text</i> consists of one or more lines, all but the last of which end with \ to hide the NEWLINE. Backslashes in text are treated like backslashes in the replacement string of an s command, and may be used to protect initial SPACE and TAB characters against the stripping that is done on every script line. An argument denoted <i>rfilename</i> or <i>wfilename</i> must terminate the command line and must be preceded by exactly one SPACE. Each <i>wfilename</i> is created before processing begins. There can be at most 10 distinct <i>wfilename</i> arguments. <table><tr><td>(1)a\</td><td></td></tr><tr><td><i>text</i></td><td>Append: place <i>text</i> on the output before reading the next input line.</td></tr><tr><td>(2)b <i>label</i></td><td>Branch to the ':' command bearing the <i>label</i>. Branch to the end of the script if <i>label</i> is empty.</td></tr><tr><td>(2)c\</td><td></td></tr><tr><td><i>text</i></td><td>Change: delete the pattern space. With 0 or 1 address or at the end of a 2 address range, place <i>text</i> on the output. Start the next cycle.</td></tr><tr><td>(2)d</td><td>Delete the pattern space. Start the next cycle.</td></tr><tr><td>(2)D</td><td>Delete the initial segment of the pattern space through the first NEWLINE. Start the next cycle.</td></tr><tr><td>(2)g</td><td>Replace the contents of the pattern space by the contents of the hold space.</td></tr><tr><td>(2)G</td><td>Append the contents of the hold space to the pattern space.</td></tr><tr><td>(2)h</td><td>Replace the contents of the hold space by the contents of the pattern space.</td></tr><tr><td>(2)H</td><td>Append the contents of the pattern space to the hold space.</td></tr><tr><td>(1)i\</td><td></td></tr><tr><td><i>text</i></td><td>Insert: place <i>text</i> on the standard output.</td></tr></table>		(1)a\		<i>text</i>	Append: place <i>text</i> on the output before reading the next input line.	(2)b <i>label</i>	Branch to the ':' command bearing the <i>label</i> . Branch to the end of the script if <i>label</i> is empty.	(2)c\		<i>text</i>	Change: delete the pattern space. With 0 or 1 address or at the end of a 2 address range, place <i>text</i> on the output. Start the next cycle.	(2)d	Delete the pattern space. Start the next cycle.	(2)D	Delete the initial segment of the pattern space through the first NEWLINE. Start the next cycle.	(2)g	Replace the contents of the pattern space by the contents of the hold space.	(2)G	Append the contents of the hold space to the pattern space.	(2)h	Replace the contents of the hold space by the contents of the pattern space.	(2)H	Append the contents of the pattern space to the hold space.	(1)i\		<i>text</i>	Insert: place <i>text</i> on the standard output.
(1)a\																												
<i>text</i>	Append: place <i>text</i> on the output before reading the next input line.																											
(2)b <i>label</i>	Branch to the ':' command bearing the <i>label</i> . Branch to the end of the script if <i>label</i> is empty.																											
(2)c\																												
<i>text</i>	Change: delete the pattern space. With 0 or 1 address or at the end of a 2 address range, place <i>text</i> on the output. Start the next cycle.																											
(2)d	Delete the pattern space. Start the next cycle.																											
(2)D	Delete the initial segment of the pattern space through the first NEWLINE. Start the next cycle.																											
(2)g	Replace the contents of the pattern space by the contents of the hold space.																											
(2)G	Append the contents of the hold space to the pattern space.																											
(2)h	Replace the contents of the hold space by the contents of the pattern space.																											
(2)H	Append the contents of the pattern space to the hold space.																											
(1)i\																												
<i>text</i>	Insert: place <i>text</i> on the standard output.																											

(2)l	List the pattern space on the standard output in an unambiguous form. Non-printing characters are spelled in two digit ASCII and long lines are folded.								
(2)n	Copy the pattern space to the standard output. Replace the pattern space with the next line of input.								
(2)N	Append the next line of input to the pattern space with an embedded newline. (The current line number changes.)								
(2)p	Print: copy the pattern space to the standard output.								
(2)P	Copy the initial segment of the pattern space through the first NEWLINE to the standard output.								
(1)q	Quit: branch to the end of the script. Do not start a new cycle.								
(2)r <i>rfilename</i>	Read the contents of <i>rfilename</i> . Place them on the output before reading the next input line.								
(2)s/ <i>regular expression</i> / <i>replacement</i> / <i>flags</i>	Substitute the <i>replacement</i> string for instances of the <i>regular expression</i> in the pattern space. Any character may be used instead of '/'. For a fuller description see regexp(5) . <i>flags</i> is zero or more of: <table> <tr> <td><i>n</i></td><td><i>n</i> = 1 – 512. Substitute for just the <i>n</i>th occurrence of the <i>regular expression</i>.</td></tr> <tr> <td><i>g</i></td><td>Global: substitute for all nonoverlapping instances of the <i>regular expression</i> rather than just the first one.</td></tr> <tr> <td><i>p</i></td><td>Print the pattern space if a replacement was made.</td></tr> <tr> <td><i>w wfilename</i></td><td>Write: append the pattern space to <i>wfilename</i> if a replacement was made.</td></tr> </table>	<i>n</i>	<i>n</i> = 1 – 512. Substitute for just the <i>n</i> th occurrence of the <i>regular expression</i> .	<i>g</i>	Global: substitute for all nonoverlapping instances of the <i>regular expression</i> rather than just the first one.	<i>p</i>	Print the pattern space if a replacement was made.	<i>w wfilename</i>	Write: append the pattern space to <i>wfilename</i> if a replacement was made.
<i>n</i>	<i>n</i> = 1 – 512. Substitute for just the <i>n</i> th occurrence of the <i>regular expression</i> .								
<i>g</i>	Global: substitute for all nonoverlapping instances of the <i>regular expression</i> rather than just the first one.								
<i>p</i>	Print the pattern space if a replacement was made.								
<i>w wfilename</i>	Write: append the pattern space to <i>wfilename</i> if a replacement was made.								
(2)t <i>label</i>	Test: branch to the ':' command bearing the <i>label</i> if any substitutions have been made since the most recent reading of an input line or execution of a t. If <i>label</i> is empty, branch to the end of the script.								
(2)w <i>wfilename</i>	Write: append the pattern space to <i>wfilename</i> .								
(2)x	Exchange the contents of the pattern and hold spaces.								

(2)y/string1/string2/	Transform: replace all occurrences of characters in <i>string1</i> with the corresponding character in <i>string2</i> . The lengths of <i>string1</i> and <i>string2</i> must be equal.
(2)! function	Do not: apply the function (or group, if function is '{') only to lines <i>not</i> selected by the address(es).
(0): label	This command does nothing. It bears a <i>label</i> for b and t commands to branch to. <i>Note:</i> The maximum length of <i>label</i> is seven characters.
(1)=	Place the current line number on the standard output as a line.
(2){	Execute the following commands through a matching '}' only when the pattern space is selected. Commands are separated by ';'.
(0)	An empty command is ignored.

Large Files See [largefile\(5\)](#) for the description of the behavior of sed when encountering files greater than or equal to 2 Gbyte (2³¹ bytes).

診断	Too many commands	The command list contained more than 200 commands.
	Too much command text	The command list was too big for sed to handle. Text in the a, c, and i commands, text read in by r commands, addresses, regular expressions and replacement strings in s commands, and translation tables in y commands all require sed to store data internally.
	Command line too long	A command line was longer than 4000 characters.
	Too many line numbers	More than 256 decimal number linecounts were specified as addresses in the command list.
	Too many files in w commands	More than 10 different files were specified in w commands or w options for s commands in the command list.
	Too many labels	More than 50 labels were specified in the command list.
	Unrecognized command	A command was not one of the ones recognized by sed.
	Extra text at end of command	A command had extra text after the end.

Illegal line number	An address was neither a decimal number linecount, a \$, nor a context address.
Space missing before filename	There was no space between an r or w command, or the w option for a s command, and the filename specified for that command.
Too many { 's	There were more { than } in the list of commands to be executed.
Too many } 's	There were more } than { in the list of commands to be executed.
No addresses allowed	A command that takes no addresses had an address specified.
Only one address allowed	A command that takes one address had two addresses specified.
"\digit" out of range	The number in a \n item in a regular expression or a replacement string in an s command was greater than 9.
Bad number	One of the endpoints in a range item in a regular expression (that is, an item of the form {n} or {n,m}) was not a number.
Range endpoint too large	One of the endpoints in a range item in a regular expression was greater than 255.
More than 2 numbers given in \{ \}	More than two endpoints were given in a range expression.
} expected after \	A \ appeared in a range expression and was not followed by a }.
First number exceeds second in \{ \}	The first endpoint in a range expression was greater than the second.
Illegal or missing delimiter	The delimiter at the end of a regular expression was absent.
\(\) imbalance	There were more \ (than \), or more \) than \ (, in a regular expression.
[] imbalance	There were more [than], or more] than [, in a regular expression.

First RE may not be null	The first regular expression in an address or in a s command was null (empty).
Ending delimiter missing on substitution	The ending delimiter in a s command was absent.
Ending delimiter missing on string	The ending delimiter in a y command was absent.
Transform strings not the same size	The two strings in a y command were not the same size.
Suffix too large - 512 max	The suffix in a s command, specifying which occurrence of the regular expression should be replaced, was greater than 512.
Label too long	A label in a command was longer than 8 characters.
Duplicate labels	The same label was specified by more than one : command.
File name too long	The filename specified in a r or w command, or in the w option for a s command, was longer than 1024 characters.
Output line too long	An output line was longer than 4000 characters long.
Too many appends or reads after line <i>n</i>	More than 20 a or r commands were to be executed for line <i>n</i> .
Hold space overflowed.	More than 4000 characters were to be stored in the <i>hold space</i> .

ファイル

usr/ucb/sed BSD sed

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目

[awk\(1\)](#), [grep\(1\)](#), [lex\(1\)](#), [attributes\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#)

使用上の留意点

There is a combined limit of 200 -e and -f arguments. In addition, there are various internal size limits which, in rare cases, may overflow. To overcome these limitations, either combine or break out scripts, or use a pipeline of sed commands.

名前	set, unset, setenv, unsetenv, export – 現在のシェルおよびそこから起動されたプロセスでの環境変数の特性を決定するシェル組み込み関数
形式	
sh	<pre>set [--aefhkntuvx <i>argument</i>]]... unset [<i>name</i>]... export [<i>name</i>]...</pre>
csh	<pre>set [<i>var</i> [= <i>value</i>]] set <i>var</i> [<i>n</i>] = <i>word</i> unset <i>pattern</i> setenv [VAR [<i>word</i>]] unsetenv <i>variable</i></pre>
ksh88	<pre>set [$\pm$abCefghkmnopstuvx] [$\pm$o <i>option</i>]]... [$\pm$A <i>name</i>] [<i>arg</i>]]... unset [-f] <i>name</i>... **export [<i>name</i> [=value]]]... **export [-p]</pre>
ksh	<pre>+set [$\pm$abCefGhkmnoprstuvx] [$\pm$o <i>option</i>]]... [$\pm$A <i>vname</i>] [<i>arg</i>]]... +unset [-fnv] <i>vname</i>... ++export [-p] [<i>name</i>[=value]]]...</pre>

機能説明

sh	set 組み込みコマンドには次のようなオプションがあります。 -- どのフラグも変更しません。\$1 に - を設定する際に便利です。 -a 修正または作成された変数にエクスポートのマークを付けます。 -e コマンドが 0 以外の終了ステータスで終了した場合、直ちに終了します。 -f ファイル名を生成しないようにします。 -h 関数の定義時に、関数コマンドを検索しその位置を記憶します (通常、関数コマンドは関数実行時に検索されます)。 -k コマンド名の前にあるキーワード引数だけでなく、すべてのキーワード引数がコマンド用の環境に置かれます。 -n コマンドを読み取るが、実行しません。 -t 1つのコマンドを読み取り、実行したあと、終了します。

- u 未設定の変数を置換時にエラーとして扱います。
- v シェル入力行の読み取り時に、その内容を表示します。
- x コマンドの実行時に、コマンドと引数の内容を表示します。

-の代わりに+を使用すると、これらのフラグは無効になります。これらのフラグはシェル起動時にも使用できます。現在セットされているフラグは、\$- で見つけられます。残りの *argument* は定位置パラメータとなり、\$1、\$2、... の順番に割り当てられます。*argument* が指定されない場合、すべての名前の値が出力されます。

unset は *name* ごとに、対応する変数または関数値を削除します。変数 PATH、PS1、PS2、MAILCHECK および IF は設定解除できません。

export 組み込みコマンドでは、指定された *name* に対し、ひきつづき実行されるコマンドの環境へ自動的にエクスポートされるようにマークを付けます。引数を省略すると、現在のシェル実行中にエクスポートのマークが付けられた変数名を一覧表示します。関数名はエクスポートされません。

csch

引数を設定しないと set はすべてのシェル変数の値を表示します。複数ワードからなる値は括弧でくくられて表示されます。引数 *var* のみを設定すると、set は空の (NULL) 値を *var* が示す変数に割り当てます。引数を *var=value* の形式で設定すると、set は *var* が示す変数に *value* が示す値を割り当てます。*value* は次のいずれかです。

word 単一ワード (もしくは引用符付きの文字列)

(*wordlist*) スペースで区切られた、括弧付きワードの並び

値は、割り当てられる前に、コマンドおよびファイル名展開されます。set *var*[*n*] = *word* 形式は、複数ワードからなる値の *n* 番目のワードを *word* に置き換えます。

unset は *pattern* が示すファイル名置換パターンに一致する名の変数を削除します。'unset *' と指定すると、すべての変数が削除されます。

引数を設定しないと setenv はすべての環境変数を表示します。引数 *VAR* を指定すると、setenv は環境変数 *VAR* に空の値 (NULL) を設定します (慣例上、環境変数名は大文字で指定されるのが通常)。VAR と *word* の両引数を指定すると、setenv は、VAR に単一ワードまたは引用符付き文字列である値 *word* を設定します。環境変数 PATH は、コロンで区切られた複数の *word* 引数を指定できます (後述の「使用例」を参照)。最もよく使用される環境変数 USER、TERM、PATH は、自動的に csh 変数 user、term、path から (へ) インポート (エクスポート) されます。これらの変数を変更する場合には setenv を使用してください。さらにシェルは、csh 変数 cwd が変更されるたびに、その値を環境変数 PWD へ設定します。

環境変数 LC_CTYPE、LC_MESSAGES、LC_TIME、LC_COLLATE、LC_NUMERIC、LC_MONETARY は、C シェル内で変更されると新しい値が即座に有効になります。これらの環境変数の詳細については [environ\(5\)](#) を参照してください。

`unsetenv` は環境から *variable* が示す変数を削除します。`unset` のようなパターンマッチングは行いません。

ksh88

`set` コマンドのフラグの意味は以下のとおりです。

- A 配列の代入。*name* で示される変数の設定を解除し、*arg* リストから順々に値を割り当てます。`+A` は、最初の変数設定を解除しません。
- a 定義される後続の変数すべてを自動的にエクスポートします。
- b バックグラウンドジョブの完了を非同期的にユーザーに通知します。
- C シェルのリダイレクト演算子「>」によって既存のファイルが上書きされるのを防ぎます。リダイレクト演算子「>|」は、個々のファイルに対して、このオプションよりも優先されます。
- e コマンドの終了ステータスが 0 でない場合、ERR トラップ (設定されていれば) を実行し、終了します。このモードは、プロファイル読み取り時は無効です。
- f ファイル名を生成しないようにします。
- h 各コマンドは、最初に検出された時点で、検索済み別名になります。
- k コマンド名に先行するものだけでなく、すべての変数代入引数をコマンドの環境に格納します。
- m バックグラウンドジョブを個別のプロセスグループで実行し、完了時にメッセージを 1 行表示します。バックグラウンドジョブの終了ステータスは完了メッセージで報告されます。ジョブ制御を備えたシステムでは、このフラグは対話型シェルに対して自動的に有効になります。
- n コマンドを読み取り、構文エラーがないかチェックします。実行は行いません。対話型シェルに対しては無視されます。
- +o 現在のオプション設定を標準出力に書き出します。コマンドとしてシェルに再入力できる形式で書き込まれるため、同じオプション設定を再現するときに便利です。
- o *option* このフラグの後に指定する引数は、次のオプション名のいずれかです。
 - `allexport` `-a` と同じです。
 - `errexit` `-e` と同じです。

bgnice	バックグラウンドジョブをすべて低い優先度で実行します。これはデフォルトモードです。emacs コマンド入力用に、emacs 形式のインラインエディタを起動します。
gmacs	コマンド入力用に、gmacs 形式のインラインエディタを起動します。
ignoreeof	ファイルの終わりを検出してもシェルは終了しません。終了させるには exit コマンドを使用する必要があります。
keyword	-k と同じです。
markdirs	ファイル名生成によって生成されるディレクトリ名には、すべて最後に / を付加します。
monitor	-m と同じです。
noclobber	シェルのリダイレクト演算子「>」によって既存のファイルが上書きされるのを防ぎます。リダイレクト演算子「> 」は、個々のファイルに対して、このオプションよりも優先されます。-c と同じです。
noexec	-n と同じです。
noglob	-f と同じです。
nolog	履歴ファイルに関数定義を保存しません。
notify	-b と同じです。
nounset	-u と同じです。
privileged	-p と同じです。
verbose	-v と同じです。
trackall	-h と同じです。
vi	vi 形式のインラインエディタの挿入モードになります。033 というエスケープ文字を押すと、これにより制御モードになります。Return で行を送信します。
viraw	各文字を vi モードで入力されたときと同様に処理します。
xtrace	-x と同じです。

オプション名を指定しない場合、o は現在のオプション設定を表示します。

-p \$HOME/.profile ファイルを処理しないようにし、ENV ファイルの代わりに /etc/suid_profile ファイルを使用します。このモードは、実効ユーザー ID が実ユーザー ID と等しくないとき、また実効グループ ID が実グループ ID と

等しくないときには必ず有効になります。このモードを無効にすると、実効ユーザー ID が実ユーザー ID に、実効グループ ID が実グループ ID にそれぞれ設定されます。

- s 定位置パラメータを辞書編集方式の順にソートします。
- t 1つのコマンドを読み取り、実行したあと、終了します。
- u 置換を行う際に、設定されていないパラメータをエラーとして扱います。
- v シェル入力行の読み取り時に、その内容を表示します。
- x コマンドの実行時に、コマンドと引数の内容を表示します。
- -x フラグと -v フラグを無効にし、フラグに対する引数の検査を停止します。
- どのフラグも変更しません。このフラグは、\$1 を - で始まる値に設定する際に便利です。このフラグの後に引数がない場合、定位置パラメータが設定解除されます。

- の代わりに + を使用すると、これらのフラグは無効になります。これらのフラグはシェル起動時にも使用できます。現在セットされているフラグは、\$- で見つけられます。-A を指定しないかぎり、残りの引数は定位置パラメータとなり、次の順番に割り当てられます。\$1 \$2 ... と同じになります。引数を 1 つも指定しない場合には、すべての変数の名前と値を標準出力上に表示します。

name が示す変数の設定を解除します。つまり、それらの変数の値と属性を消去します。読み取り専用の変数は設定を解除できません。-f フラグが設定されていると、*name* 引数は関数名を表します。ERRNO、LINENO、MAILCHECK、OPTARG、OPTIND、RANDOM、SECONDS、TMOUT、および _ の設定を解除すると、これらの変数の特殊な意味が削除されます。後でこれらの変数に値を代入しても、特殊な意味はないままです。

unset を使用すると *name* が示す変数の設定を解除します。つまり、それらの変数の値と属性を消去します。読み取り専用の変数は設定を解除できません。-f フラグが設定されていると、*name* 引数は関数名を表します。ERRNO、LINENO、MAILCHECK、OPTARG、OPTIND、RANDOM、SECONDS、TMOUT、および _ の設定を解除すると、これらの変数の特殊な意味が削除されます。後でこれらの変数に値を代入しても、特殊な意味はないままです。

export 組み込みコマンドでは、指定された *name* に対し、ひきつづき実行されるコマンドの環境へ自動的にエクスポートされるようにマークを付けます。

-p を指定したとき、*export* は、すべてのエクスポートされる変数の名前と値を、次の形式で標準出力に書き出します。

```
"export %s=%s\n", name, value
```

これは、`name` が設定されている場合です。

```
"export %s\n", name
```

これは、`name` が設定されていない場合です。

コマンドとしてシェルに再入力できる形式で書き込まれる (引用も適切に使用される) ため、エクスポートされる変数の名前と値をシェル実行環境で再現するときに便利です。ただし、この場合、次のような制限があります。

1. 出力時に値が設定されていた読み取り専用の変数は、元の状態には戻りません。
2. 出力時に値が設定されていなかった変数に、状態を保存したときから保存した出力をシェルに再入力するまでの時間が値として割り当てられた場合、その変数は未設定の状態には戻りません (つまり、その値は保持されます)。

このマニュアルページでは、1 つまたは 2 つのアスタリスク (*) が先頭に付加されている [ksh88\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は = 符号のあとに実行され、ワード分割とファイル名生成は実行されません。

ksh

`set` はオプションおよび定位置パラメータを設定または設定解除します。- を付けてオプションを指定すると、オプションが設定されます。+ を付けてオプションを指定すると、オプションの設定が解除されます。

オプションや引数を付けずに `set` を使用すると、すべてのシェル変数の名前と値が現在のロケールの照合シーケンスの順序で表示されます。シェルへの再入力に適するように、値は引用符で囲まれます。

引数が指定されていない (オプション引数の終わり -- さえ指定されていない) 場合、定位置パラメータは変更されません。それ以外の場合は、-A オプションを指定しないかぎり、定位置パラメータは引数のリストで置換されます。定位置パラメータを設定するときは、最初の引数 -- が無視されます。

下位互換性のために、オプションを指定せずに `set` コマンドを使用すると (最初の引数は -)、-v および -x オプションが無効になります。追加の引数を指定すると、定位置パラメータが置換されます。

ksh では `set` に次のオプションがあります。

- a 名前に「`.`」が含まれない各変数にエクスポート属性を設定して、現在のシェル環境で値を割り当てます。
- A *name* 定位置パラメータではなく、添字 0 で始まる *name* で指定された配列に、引数を順番に割り当てます。
- b 次のプロンプトまで待機せずに、バックグラウンドジョブの完了が検出されるとすぐにメッセージを標準エラーに書き込みます。
- B {...} グループ展開を有効にします。デフォルトではオンになっています。
- C リダイレクト演算子「`>`」を使用して、既存の標準ファイルが上書きされるのを防ぎます。「`>|`」リダイレクトは、この `noclobber` オプションよりも優先されます。
- e 終了ステータスがゼロ以外の単純コマンドを使用すると、シェルが終了します。ただし、単純コマンドが次の場合は除きます。
 - `&&` または `||` リストに含まれる
 - `if`、`while`、または `until` 直後のコマンド
 - `!` 直後のパイプラインに含まれる
- f パス名展開が無効になります。
- G ** 単独でも、パス名展開時にすべてのサブディレクトリに一致します。
- h 廃止。別名の構文を持つ各コマンドが、最初に検出されたときに検索済みの別名になります。
- H `csh` と同様に `!` 形式の履歴展開を有効にします。
- k 廃止されました。形式 *name=value* の引数はすべて削除され、コマンドの変数代入リストに配置されます。通常、変数代入はコマンド引数の前に指定する必要があります。
- m これを有効にすると、バックグラウンドジョブを個別のプロセスグループで実行し、完了時にメッセージを 1 行表示します。ジョブ制御をサポートするシステム上の対話型シェルでは、このモードはデフォルトで有効になっています。
- n コマンドを読み取り、構文エラーがないかチェックします。ただし、コマンドは実行しません。通常、コマンド呼び出し時に指定されます。
- o [*option*] オプションを指定しない場合、オプションと現在の設定のリストが標準出力に書き込まれます。`+` を付けて呼び出された場合、シェルに再入力して設定を復元できる形式で、オプションが書き込まれます。このオプションを繰り返して、複数のオプションを有効または無効にすることができます。

option の値は、次のいずれかにする必要があります。

allexport	-a と同じです。
bgnice	バックグラウンドジョブをすべて低い優先度で実行します。
braceexpand	-B と同じです。
emacs	emacs 編集モードを有効または無効にします。
errexit	-e と同じです。
globstar	-G と同等です。
gmacs	gmacs を有効または無効にします。gmacs 編集モードは、CTRL-T の処理を除いて emacs 編集モードと同じです。
histexpand	-H と同じです。
ignoreeof	ファイルの終わりを検出しても対話型シェルは終了しません。
keyword	-k と同じです。
markdirs	ファイル名生成によって生成されるディレクトリ名には、すべて最後に / を付加します。
monitor	-m と同じです。
multiline	ウィンドウ幅よりも長い行を編集するときに、複数の行を使用します。
noclobber	-c と同じです。
noexec	-n と同じです。
noglob	-f と同じです。
nolog	このオプションは無効です。下位互換性を確保するために用意されています。
notify	-b と同じです。
nounset	-u と同じです。
pipefail	パイプラインのすべてのコンポーネントが完了するまで、パイプラインは完了しません。パイプラインの終了ステータスは、ゼロ以外の終了ステータスで最後に終了したコマンドの値になり、すべてのコマンドがゼロの終了ステータスを返す場合はゼロになります。

<code>privileged</code>	<code>-p</code> と同じです。
<code>showme</code>	前に「;」が付く単純コマンドは、 <code>-x</code> を有効にした場合と同様に追跡されますが、実行はされません。
<code>trackall</code>	<code>-h</code> と同じです。
<code>verbose</code>	<code>-v</code> と同じです。
<code>vi</code>	<code>vi</code> 編集モードを有効または無効にします。
<code>viraw</code>	<code>vi</code> 編集モードを使用するときは、標準入力モードを使用しないでください。
<code>xtrace</code>	<code>-x</code> と同じです。
<code>-p</code>	特権モード。 <code>-p</code> を無効にすると、実効ユーザー ID が実ユーザー ID に設定され、実効グループ ID が実グループ ID に設定されます。 <code>-p</code> を有効にすると、実効ユーザー ID および実効グループ ID がシェル起動時の値に復元されます。実ユーザー ID と実効ユーザー ID が同じでないとき、または実グループ ID と実効グループ ID が同じでないときは、常に <code>-p</code> オプションは有効になります。 <code>-p</code> が有効になっているときは、ユーザープロファイルは処理されません。
<code>-r</code>	制限されます。制限付きシェルを有効にします。このオプションは、いったん有効にすると設定を解除できません。
<code>-s</code>	定位置パラメータをソートします。
<code>-t</code>	廃止。1つのコマンドを読み取ってから終了します。
<code>-u</code>	有効にすると、未設定の変数を展開しようとするエラーメッセージが表示されます。
<code>-v</code>	冗長。読み取り時に入力を標準エラーに出力します。
<code>-x</code>	実行トレース。すべての展開のあと、および <code>PS4</code> パラメータの展開値が付く実行の前に、各コマンドを表示します。

`ksh` の `set` では、次の終了値が返されます。

- 0 正常終了。
- >0 エラーが発生した。

指定された名前ごとに、`unset` は現在のシェル実行環境から変数または関数 (`-f` が指定されている場合) の設定を解除します。読み取り専用の変数は設定を解除できません。

`ksh` の `unset` には次のオプションがあります。

- `-f` *name* が関数名を指している場合は、関数定義の設定を解除します。

- n *name* が参照の変数を指している場合は、参照される変数ではなく、変数 *name* の設定が解除されます。それ以外の場合、このオプションは -v オプションと同等です。
- v *name* が変数名を指している場合は、設定を解除して、環境から削除します。これはデフォルトの動作です。

ksh の unset では、次の終了値が返されます。

- 0 正常終了。すべての名前は正常に設定解除されました。
- >0 エラーが発生したか、または1つ以上の *name* オペランドの設定を解除できませんでした。

export は、名前で指定された各変数にエクスポート属性を設定します。これにより、あとで実行されるコマンドの環境に変数が配置されます。=value を指定した場合は、変数 *name* が *value* に設定されます。

name を指定しない場合は、すべてのエクスポート済み変数の名前および値が標準出力に書き込まれます。

引数でフィールド分割やパス名展開が実行されないように、export は宣言コマンドとしてシェルに組み込まれています。チルド展開は値に対して行われます。

ksh の export には次のオプションがあります。

- p 出力をシェルへの入力に使用可能な export コマンドの形式にして、現在のエクスポートを再作成できるようにします。

ksh の export では、次の終了値が返されます。

- 0 正常終了。
- >0 エラーが発生した。

このマニュアルページでは、1つまたは2つの+(プラス符号)が先頭に付加されている **ksh(1)** コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

使用例

csh 次の例では、`/bin`、`/usr/bin`、`/usr/sbin`、`/usr/ucb/bin` ディレクトリにあるファイルをその順番で検索するために、環境変数 `PATH` を設定しています。

```
setenv PATH "/bin:/usr/bin:/usr/sbin:usr/ucb/bin"
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [read\(1\)](#), [sh\(1\)](#), [typeset\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#)

名前	setfacl - 1つまたは複数のファイルのアクセス制御リスト (ACL) の変更
形式	<pre>setfacl [-r] -s <i>acl_entries</i> <i>file</i> setfacl [-r] -md <i>acl_entries</i> <i>file</i> setfacl [-r] -f <i>acl_file</i> <i>file</i></pre>
機能説明	指定された各ファイルについて、setfacl は ACL 全体 (ディレクトリ上のデフォルトの ACL を含む) を置き換えるか、あるいは 1 つ以上の ACL エントリ (ディレクトリ上のデフォルトのエントリを含む) を追加、変更、または削除するかのどちらかを行ないます。 setfacl コマンドが使用されると、ファイルのアクセス権ビットが変更される場合があります。ファイル所有者のユーザー ACL エントリが変更された場合は、ファイル所有者クラスのアクセス権ビットが変更されます。ファイルグループクラスのグループ ACL エントリが変更された場合は、ファイルグループクラスのアクセス権ビットが変更されます。その他の ACL エントリが変更された場合は、ファイルのほかのクラスのアクセス権ビットが変更されます。 chmod(1) コマンドを使用して ACL エントリを含むファイルに対するファイルグループ所有者のアクセス権を変更した場合は、ファイルグループ所有者のアクセス権と ACL マスクの両方が新しいアクセス権に変更されます。新しい ACL マスクのアクセス権は、そのファイル上に ACL エントリを持つ追加ユーザーおよびグループの実効アクセス権を変更する場合があるので注意が必要です。 ディレクトリには、デフォルトの ACL エントリを含めることができます。デフォルトの ACL エントリを含むディレクトリ内にファイルまたはディレクトリが作成された場合、新しく作成されたファイルには、デフォルトの ACL エントリと、作成時に要求されたアクセス権の共通部分に従って生成されたアクセス権が与えられます。ディレクトリにデフォルトの ACL エントリが含まれている場合、umask(1) は適用されません。特定のユーザー (または複数のユーザー) に対してデフォルトの ACL が指定されている場合、ファイルには、作成された通常の ACL が与えられます。それ以外の場合は、上で説明した共通部分に従ってモードビットのみが初期化されます。デフォルトの ACL を、付与できる最大の随意アクセス権と考えてください。 POSIX ドラフト ACL (または <code>aclent_t</code> スタイルの ACL) をサポートする UFS ファイルシステム内のファイルに ACL を設定するには、setfacl コマンドを使用します。NFSv4 スタイルの ACL (または <code>ace_t</code> スタイルの ACL) をサポートする ZFS ファイルシステム内のファイルに ACL を設定するには、chmod コマンドを使用します。
<i>acl_entries</i> の構文	-m オプションと -s オプションの場合、<i>acl_entries</i> は、コンマで区切られた 1 つ以上の ACL エントリです。 ACL エントリは、コロンで区切られた次のフィールドで構成されます。

<i>entry_type</i>	ファイルアクセス権を設定する ACL エントリのタイプ。たとえば、 <i>entry_type</i> には、 <i>user</i> (ファイルの所有者) または <i>mask</i> (ACL マスク) を指定できます。
<i>uid</i> または <i>gid</i>	ユーザー名またはユーザー ID 番号。または、グループ名またはグループ ID 番号。
<i>perms</i>	<i>entry_type</i> に設定されているアクセス権を表します。 <i>perms</i> は、シンボル文字 <i>rwX</i> または番号 (<i>chmod</i> コマンドで使用されるのと同じアクセス権の番号) で示すことができます。

次の表は、有効な ACL エントリを示しています (デフォルトのエントリは、ディレクトリに対してのみ指定できます)。

ACL Entry	説明
<i>u[ser]::perms</i>	ファイル所有者のアクセス権。
<i>g[roup]::perms</i>	ファイルのグループ所有者のアクセス権。
<i>o[ther]:perms</i>	ファイル所有者またはファイルグループ所有者のメンバー以外のユーザーのアクセス権。
<i>m[ask]:perms</i>	ACL マスク。マスクエントリは、ユーザー (所有者以外) とグループに許可される最大のアクセス権を示します。マスクは、すべてのユーザーおよびグループに対するアクセス権をすばやく変更するための方法です。
<i>u[ser]:uid:perms</i>	特定のユーザーのアクセス権。 <i>uid</i> には、ユーザー名または数値 UID のどちらかを指定できます。
<i>g[roup]:gid:perms</i>	特定のグループのアクセス権。 <i>gid</i> には、グループ名または数値 GID のどちらかを指定できます。
<i>d[efault]:u[ser]::perms</i>	所有者のデフォルトアクセス権。
<i>d[efault]:g[roup]::perms</i>	ファイルのグループ所有者のデフォルトアクセス権。
<i>d[efault]:o[ther]:perms</i>	ファイル所有者またはファイルグループ所有者のメンバー以外のユーザーのデフォルトのアクセス権。
<i>d[efault]:m[ask]:perms</i>	デフォルトの ACL マスク。
<i>d[efault]:u[ser]:uid:perms</i>	特定のユーザーのデフォルトアクセス権。 <i>uid</i> には、ユーザー名または数値 UID のどちらかを指定できます。
<i>d[efault]:g[roup]:gid:perms</i>	特定のグループのデフォルトアクセス権。 <i>gid</i> には、グループ名または数値 GID のどちらかを指定できます。

-d オプションの場合、*acl_entries* は、アクセス権のない、コンマで区切られた 1 つ以上の ACL エントリです。ファイル所有者、ファイルグループ所有者、ACL マスク、およびその他のユーザーのエントリは削除できないことに注意してください。

オプション

各オプションの意味は次のとおりです。

-d *acl_entries* ファイルから 1 つ以上のエントリを削除します。ACL からファイル所有者、ファイルグループ所有者、およびその他のユーザーのエントリを削除することはできません。エントリの削除が、必ずしもそのエントリからすべてのアクセス権を削除するのと同じ効果があるとは限りません。

-f *acl_file* ファイルの ACL に、*acl_file* という名前のファイルに含まれている ACL エントリを設定します。指定されるエントリには、-s オプションと同じ制約が適用されます。これらのエントリがファイル内に特定の順序で存在する必要はありません。また、*acl_file* にダッシュ (-) を指定した場合は、ファイルの ACL を設定するために標準入力を使用されます。

acl_file 内に文字 # を使用すると、コメントを示すことができます。# で始まる文字は、その行の最後まですべて無視されます。*acl_file* が [getfacl\(1\)](#) コマンドの出力として作成されている場合は、# に続く実効アクセス権がすべて無視されることに注意してください。

-m *acl_entries* 1 つ以上の新しい ACL エントリをファイルに追加するか、またはファイル上の 1 つ以上の既存の ACL エントリを変更するか、あるいはその両方を行ないます。指定された *uid* または *gid* のエントリがすでに存在する場合は、指定されたアクセス権によって現在のアクセス権が置き換えられます。指定された *uid* または *gid* のエントリが存在しない場合は、エントリが作成されます。**-m** オプションを使用してデフォルトの ACL を変更している場合は、完全なデフォルトの ACL (*user*、*group*、*other*、*mask*、および任意の追加のエントリ) を最初に指定する必要があります。

-r ACL マスクエントリのアクセス権を再計算します。ACL マスクエントリで指定されたアクセス権は無視され、ACL 内のすべての追加のユーザー、ファイルグループ所有者、および追加のグループエントリへのアクセスを許可するために必要な最大のアクセス権で置き換えられます。追加のユーザー、ファイルグループ所有者、および追加のグループエントリ内のアクセス権は変更されません。

-s *acl_entries* ファイルの ACL を設定します。すべての古い ACL エントリが削除され、新しく指定された ACL に置き換えられます。これらのエントリが特定の順序で存在する必要はありません。ファイルに適用される前に、このコマンドによってエントリがソートされます。

必要なエントリは次のとおりです。

- ファイル所有者に対して指定された正確に 1 つの `user` エントリ。
- ファイルグループ所有者に対する正確に 1 つの `group` エントリ。
- 指定された正確に 1 つの `other` エントリ。

追加のユーザーおよびグループエントリが存在する場合は次のとおりです。

- ユーザー (所有者以外) とグループに許可される最大のアクセス権を示す、ACL マスクに対して指定された正確に 1 つの `mask` エントリ。
- 同じ `uid` を持つ重複した `user` エントリが存在してはいけません。
- 同じ `gid` を持つ重複した `group` エントリが存在してはいけません。

`file` がディレクトリである場合は、次のデフォルトの ACL エントリを指定できます。

- ファイル所有者に対する正確に 1 つの `default user` エントリ。
- ファイルグループ所有者に対する正確に 1 つの `default group` エントリ。
- ACL マスクに対する正確に 1 つの `default mask` エントリ。
- 正確に 1 つの `default other` エントリ。

追加の `default user` エントリや追加の `default group` エントリを指定できますが、同じ `uid` を持つ重複した追加の `default user` エントリ、または同じ `gid` を持つ重複した `default group` エントリが存在してはいけません。

使用例

例1 読み取り権のみの追加

次の例では、ファイル `abc` に、ユーザー `shea` に読み取り権のみを許可する 1 つの ACL エントリを追加します。

```
setfacl -m user:shea:r— abc
```

例2 ファイルの ACL 全体の置き換え

次の例では、ファイル `abc` の ACL 全体を置き換えます。これにより、`shea` に読み取りアクセス権、ファイル所有者にすべてのアクセス権、ファイルグループ所有者に読み取りアクセス権のみ、ACL マスクに読み取りアクセス権のみがそれぞれ許可され、その他のユーザーにはアクセス権が許可されません。

```
setfacl -s user:shea:rxw,user::rxw,group::rw-,mask:r--,other:--- abc
```

例2 ファイルのACL全体の置き換え (続き)

このコマンドのあと、ファイルのアクセス権ビットは `rwxr-----` になります。ファイルグループ所有者には読み取り権と書き込み権が設定されたにもかかわらず、ACL マスクエントリによって読み取り権のみに制限されています。このマスクエントリはまた、すべての追加のユーザーおよびグループ ACL エントリに使用可能な最大のアクセス権も指定しています。同様に、ユーザー `shea` にはすべてのアクセス権が設定されたにもかかわらず、このマスクによって読み取り権のみに制限されています。ACL マスクエントリは、ACL 内のすべてのユーザーおよびグループエントリへのアクセスをすばやく制限したり、開いたりするための方法です。たとえば、このマスクエントリを読み取りと書き込みに変更することによって、ファイルグループ所有者とユーザー `shea` の両方に読み取りアクセス権と書き込みアクセス権が許可されます。

例3 2つのファイルでの同じACLの設定

次の例では、ファイル `abc` にファイル `xyz` と同じ ACL を設定します。

`getfacl xyz | setfacl -f - abc`

ファイル `/etc/passwd` パスワードファイル
`/etc/group` グループファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [chmod\(1\)](#), [getfacl\(1\)](#), [umask\(1\)](#), [aclcheck\(3SEC\)](#), [aclsort\(3SEC\)](#), [group\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

名前	setlabel – ファイルの有効な機密ラベルの変更
形式	<code>/usr/bin/setlabel newlabel filename...</code>
機能説明	ほとんどの場合、<code>setlabel</code> は、ラベルが <i>newlabel</i> に対応しているゾーンにファイルを移動します。古いファイルパス名は、新規ゾーンのルートパス名との相対位置になるように調整されます。ファイルの親ディレクトリの古いパス名が、新規ゾーン内にディレクトリとして存在しない場合、ファイルは移動されません。移動後は、現在のゾーン内でファイルにアクセスすることはできなくなります。 ただし、<code>setlabel</code> は、マルチレベル ZFS ファイルシステム上に存在するファイルやディレクトリに対しては異なった方法で動作します。zfs(1M) を参照してください。その場合、マルチレベルファイルシステムではファイルごとのラベルがサポートされているため、ファイルは移動されませんが、そのラベルが適切に付け直されます。 <i>newlabel</i> および <i>filename</i> が指定されない場合、ラベルは設定されません。 ラベルの定義は、サイトのセキュリティー管理者により行われます。ラベルは、システム内で常に大文字で表示されます。ユーザーは、大文字と小文字を任意に組み合わせてラベルを入力できます。ラベルの内容の追加や削除も可能です。 このコマンドを実行するのに必要な条件および必要な特権についての説明は、setlabel(3TSOL) を参照してください。
終了ステータス	<code>setlabel</code> は、終了時に次のいずれかの値を返します。 0 正常終了。 1 使用方法のエラー。 2 ラベルの取得、設定、または変換中にエラー発生。
使用法	コマンド行では、1つの単語だけから成るラベルを指定する場合を除き、ラベル全体を二重引用符で囲みます。引用符がないと、スペースの次の単語または文字は、新たな引数として解釈されます。 <pre>% setlabel SECRET somefile % setlabel "TOP SECRET" somefile</pre> ラベルの内容は大文字でも、小文字でも、両者を混在させてもかまいません。ラベル内に複数の項目を記述する場合には、空白、タブ、コンマ、スラッシュ (/) のいずれかで区切ります。それ以外の文字や記号は区切り文字とは見なされません。 <pre>% setlabel "ts a b" somefile % setlabel "ts,a,b" somefile % setlabel "ts/a b" somefile % setlabel " TOP SECRET A B " somefile</pre>

使用例

例1 ラベルを設定します。
somefile のラベルを SECRET A に設定する

```
example% setlabel "Secret a" somefile
```

例2 コンパートメントを有効にします。
プラス記号とマイナス記号を使って、既存のラベルを変更できます。プラス記号を使用すると、*somefile* のラベル用の指定したコンパートメントが有効になります。

```
example% setlabel +b somefile
```

例3 コンパートメントを無効にします。
マイナス記号を使用すると、格付け部に関連付けられたコンパートメントが無効になります。次のようにして、*somefile* のラベル内のコンパートメント A を無効にします。

```
example% setlabel -A somefile
```

既存ラベルの内容を追加または削除する場合で、ラベルの先頭文字がハイフン (-) のときは、その前に二重ハイフン (--) を指定する必要があります。

次のようにして、*somefile* のラベル内のコンパートメント -A を無効にします。

```
example% setlabel -- -A somefile
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/trusted
インタフェースの安定性	確実

関連項目

[zfs\(1M\)](#), [setlabel\(3TSOL\)](#), [label_encodings\(4\)](#), [attributes\(5\)](#)

注意事項

このマニュアルページに記載されている機能を利用できるのは、システムが Trusted Extensions を使用して構成されている場合のみです。

ラベル設定に関するこの実装は、米国国防情報局 (DIA) の必須アクセス制御 (MAC) ポリシーにとって重要です。詳細は、[label_encodings\(4\)](#) を参照してください。

名前	setpggrp - プロセスグループ ID の設定
形式	setpggrp <i>command</i> [<i>arg</i>]...
機能説明	現在のプロセスがまだセッションリーダーではない場合、setpggrp ユーティリティーはプロセスグループ ID とセッション ID に現在のプロセス ID を設定し、 <i>command</i> とその引数 (存在する場合) の <code>exec()</code> を行います。
オペランド	次のオペランドがサポートされています。 <i>command</i> 呼び出すコマンドの名前。 <i>arg</i> <i>command</i> のオプションまたは引数。
終了ステータス	次の終了ステータスが返されます。 1 setpggrp ユーティリティーの実行中、または <i>command</i> の <code>exec()</code> の際にエラーが発生しました。 それ以外の場合、終了ステータスは <i>command</i> の終了ステータスとなります。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [exec\(2\)](#), [setpggrp\(2\)](#), [attributes\(5\)](#)

名前 sftp - セキュリティー保護されたファイル転送プログラム

形式 `sftp [-1Cv] [-B buffer_size] [-b batchfile] [-F ssh_config]
 [-o ssh_option] [-P sftp_server_path] [-R num_requests]
 [-S program] [-s subsystem | sftp_server] host`
`sftp [[user@]host[:file [file]]]`
`sftp [[user@]host[:dir[/]]]`
`sftp -b batchfile [user@]host`

機能説明 sftp ユーティリティーは、[ftp\(1\)](#) に似たユーザーインタフェースを持つ対話型のファイル転送プログラムであり、[ssh\(1\)](#) コマンドを使用してサーバーへのセキュリティー保護された接続を確立します。

sftp は、IETF draft-ietf-secsh-filexfer に定義されている SSH ファイル転送プロトコルを実装します。sftp によって使用されるプロトコルと、[ftp\(1\)](#) が提供する FTP プロトコル (RFC 959) は関係はありません。

最初の書式では、sftp は指定されたホストに接続し、対話型モードに入ります。ユーザー名を指定した場合、sftp は指定されたユーザーとしてログインしようとし、ディレクトリを指定した場合、sftp はサーバーの現在のディレクトリを指定されたディレクトリに変更してから、対話型モードに入ろうとします。

2 番目の書式では、指定されたファイルをサーバーから取り出し、それをクライアント上の指定されたターゲットファイルまたはディレクトリにコピーします。ユーザー名を指定した場合、sftp は指定されたユーザーとしてログインしようとし、

オプション サポートしているオプションは、次のとおりです。

- b *batchfile* バッチモードでは、一連のコマンドを標準入力ではなく入力 *batchfile* から読み取ります。このモードにはユーザーによる介入がないため、対話型でない認証と組み合わせて使用するようにしてください。- のバッチファイルを使用すると、標準入力を指定できます。get、put、rm、rename、ln、rm、mkdir、chdir、ls、lchdir、chmod、および lmkdir コマンドのいずれかでエラーが発生すると、sftp は異常終了します。コマンドに接頭辞として - 文字を付けると (例: -rm /tmp/blah*)、エラー時の終了をコマンドごとに抑制できます。
- B *buffer_size* sftp がファイルの転送時に使用するバッファのサイズを指定します。バッファのサイズを大きくすると、必要な往復回数が少なくなりますが、メモリーが余分に消費されます。デフォルトは 32768 バイトです。
- C 圧縮機能を有効にします ([ssh\(1\)](#) の -c フラグを使用)。

<code>-F ssh_config</code>	ssh のユーザーごとの設定ファイルに別のファイルを指定します。このオプションは、ssh(1) に直接渡されず。
<code>-o ssh_option</code>	ssh(1) に直接渡されるオプションを指定します。
<code>-P sftp_server path</code>	指定されたパスを <i>sftp-server</i> として実行し、ssh 接続ではなくパイプを使ってそのサーバーと通信します。このオプションは、sftp クライアントおよびサーバーのデバッグに役立つことがあります。-P が指定された場合、-S オプションは無視されます。
<code>-R num_requests</code>	未処理の要求を同時にいくつまで受け付けられるかを指定します。この値を大きくすると、ファイル転送速度がわずかに向上することがありますが、メモリー使用量が増加します。デフォルトは、未処理の要求 64 個です。
<code>-s subsystem sftp_server</code>	SSH2 サブシステム、またはリモートホスト上の sftp サーバーのパスを指定します。パスは、プロトコルバージョン 1 を介して sftp を使用する場合や、リモートの sshd に sftp サブシステムが設定されていない場合に便利です。
<code>-S ssh_program path</code>	ssh(1) ではなく、指定されたプログラムを使用して、sftp サーバーに接続します。-P オプションが指定された場合、-s オプションは無視されます。このプログラムは、ssh(1) のオプションを認識する必要があります。
<code>-v</code>	ログレベルを上げます。このオプションは ssh(1) にも渡されます。
<code>-1</code>	プロトコルバージョン 1 を使用することを指定します。

オペランド 次のオペランドがサポートされています。

hostname | *user@hostname* sftp が接続して、ログインするホストの名前。

対話型コマンド 対話型モードに入ると、sftp は ftp(1) に似た一連のコマンドを解釈できます。これらのコマンドは大文字と小文字の区別がなく、スペースが含まれているパス名は引用符で囲むことができます。

<code>bye</code>	sftp を終了します。
<code>cd path</code>	リモート側のディレクトリを <i>path</i> に変更します。
<code>chgrp grp path</code>	ファイル <i>path</i> のグループを <i>grp</i> に変更します。 <i>grp</i> には数値の GID を指定する必要があります。

<code>chmod <i>mode path</i></code>	ファイル <i>path</i> のアクセス権を <i>mode</i> に変更します。
<code>chown <i>own path</i></code>	ファイル <i>path</i> の所有者を <i>own</i> に変更します。 <i>own</i> には数値の UID を指定する必要があります。
<code>exit</code>	sftp を終了します。
<code>get [<i>flags</i>] <i>remote-path</i> [<i>local-path</i>]</code>	<i>remote-path</i> を取り出し、ローカルマシン上に格納します。ローカルパス名を指定しない場合は、リモートマシン上にあるものと同じ名前になります。 <i>-P</i> フラグを指定した場合は、ファイルのアクセス権およびアクセス時間もすべてコピーされます。
<code>help</code>	ヘルプテキストを表示します。 ? コマンドと同じです。
<code>lcd <i>path</i></code>	ローカルディレクトリを <i>path</i> に変更します。
<code>lls [<i>ls-options</i>] [<i>path</i>]</code>	<i>path</i> のローカルディレクトリの一覧を表示します。 <i>path</i> を指定しない場合は、現在のディレクトリの一覧を表示します。
<code>lmkdir <i>path</i></code>	<i>path</i> によって指定されたローカルディレクトリを作成します。
<code>ln <i>oldpath newpath</i></code>	<i>oldpath</i> から <i>newpath</i> へのリンクを作成します。
<code>lpwd</code>	ローカル側の作業用ディレクトリを表示します。
<code>ls [-laflnrSt] [<i>path</i>]</code>	<i>path</i> のリモートディレクトリの一覧を表示します。 <i>path</i> を指定しない場合は、現在のディレクトリの一覧を表示します。 <i>path</i> には、ワイルドカードを使用できます。 ls コマンドは、次のオプションをサポートします。 -a ドット (.) で始まるファイルを一覧表示します。 -f リストをソートしません。デフォルトのソート順序は辞書編集方式の順です。 -l アクセス権や所有権の情報を含む、追加の詳細情報を表示します。

	-n ユーザーおよびグループの情報を数値で示した長い形式のリストを生成します。
	-r リストのソート順序を逆にします。
	-S ファイルサイズでリストをソートします。
	-t 最終変更時間でリストをソートします。
	-l 1 カラムの出力を生成します。
lumask <i>umask</i>	ローカル側の <i>umask</i> を <i>umask</i> に設定します。
mkdir <i>path</i>	<i>path</i> によって指定されたりモートディレクトリを作成します。
put [<i>flags</i>] <i>local-path</i> [<i>local-path</i>]	<i>local-path</i> をアップロードし、リモートマシン上に格納します。リモートパス名を指定しない場合は、ローカルマシン上にあるものと同じ名前になります。-P フラグを指定した場合は、ファイルのアクセス権およびアクセス時間もすべてコピーされます。
pwd	リモート側の作業用ディレクトリを表示します。
quit	sftp を終了します。
rename <i>oldpath</i> <i>newpath</i>	リモートファイルの名前を <i>oldpath</i> から <i>newpath</i> に変更します。
rm <i>path</i>	<i>path</i> によって指定されたりモートファイルを削除します。
rmdir <i>path</i>	<i>path</i> によって指定されたりモートディレクトリを削除します。
symlink <i>oldpath</i> <i>newpath</i>	<i>oldpath</i> から <i>newpath</i> へのシンボリックリンクを作成します。
version	sftp のプロトコルバージョンを表示します。
# [<i>comment</i>]	コメントを組み込みます。これはバッチファイルに役立ちます。
! [<i>command</i>]	<i>command</i> を指定しない場合は、ローカル側のシェルにエスケープします。 <i>command</i> を指定した場合は、ローカル側のシェルで <i>command</i> を実行します。

? ヘルプテキストを表示します。
help コマンドと同じです。

終了ステータス 次の終了ステータスが返されます。
0 正常終了。
>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目 [ftp\(1\)](#), [scp\(1\)](#), [ssh\(1\)](#), [ssh-add\(1\)](#), [ssh-keygen\(1\)](#), [sshd\(1M\)](#), [attributes\(5\)](#)

名前	sh, jsh – 標準シェルとジョブ制御シェルおよびコマンドインタプリタ
形式	<pre> /usr/sunos/bin/sh [-acefhiknprstuvx] [argument]... /usr/xpg4/bin/sh [± abCefhikmnoprstuvx] [± o option]... [-c string] [arg]... /usr/sunos/bin/jsh [-acefhiknprstuvx] [argument]...</pre>
機能説明	<code>/usr/sunos/bin/sh</code> ユーティリティーは、端末またはファイルから読み取られたコマンドを実行するコマンドプログラミング言語です。 <code>/usr/xpg4/bin/sh</code> ユーティリティーは標準に準拠したシェルです。このユーティリティーは ksh88(1) のすべての機能を提供します。ただし、ksh(1) で説明しているように、動作が異なる場合を除きます。 <code>jsh</code> ユーティリティーは、<code>sh</code> のすべての機能を備えた、かつジョブ制御を可能にするシェルへのインタフェースです(下記の「ジョブ制御」を参照)。 注 <code>/usr/bin/sh</code> および <code>/usr/bin/jsh</code> は、ksh93 にリンクされています。ksh93(1) を参照してください。 シェルへの引数については、後述の「呼び出し」にリストされています。
定義	ブランク (<i>blank</i>) はタブ (tab) またはスペース (space) です。名前 (<i>name</i>) は ASCII 英文字、数字、または下線の並びで、先頭文字は英文字または下線です。パラメータ (<i>parameter</i>) は、名前、複数、または <code>*</code>、<code>@</code>、<code>#</code>、<code>?</code>、<code>-</code>、<code>\$</code>、<code>!</code> などの文字のいずれかです。
使用法	
コマンド	単純コマンド (<i>simple-command</i>) は、ブランクで区切られた、ブランクでないワードの並びです。先頭のワードは、実行すべきコマンドの名前を指定します。残りのワードは、次に述べる場合を除き、呼び出されたコマンドに引数として渡されます。コマンド名は引数 0 として渡されます (exec(2) を参照)。単純コマンドの値 (<i>value</i>) は、正常終了した場合は終了ステータス、異常終了した場合は <code>200+status</code> (8 進数) です。ステータス値の一覧表については、signal.h(3HEAD) を参照してください。 パイプライン (<i>pipeline</i>) は、パイプ (<code> </code>) で区切られた 1 つ以上のコマンドの並びです。最後のコマンドを除き、各コマンドの標準出力は pipe(2) によってその次のコマンドの標準入力と結合されます。各コマンドは、別々のプロセスとして実行されます。シェルは最後のコマンドが終了するのを待ちます。最後のコマンドの終了ステータスがパイプライン全体の終了ステータスとなります。 リスト (<i>list</i>) は、<code>;</code>、<code>&</code>、<code>&&</code>、または <code> </code> で区切られた 1 つ以上のパイプラインの並びです。その並びの終わりに <code>;</code> または <code>&</code> を記述することもできます。これら 4 つの記号の中で、<code>;</code> と <code>&</code> の優先度は同じで、<code>&&</code> と <code> </code> の優先度より低くなります。<code>&&</code> と <code> </code> の優先度は同じです。セミコロン (<code>;</code>) によって、直前のパイプラインが順次実行さ

れます。つまりシェルはパイプラインが終了するのを待ってから、セミコロン後のコマンドを実行します。アンパサンド記号(&)によって、直前のパイプラインが非同期的に実行されます。つまりシェルはパイプラインが終了するのを待ちません。&&という記号は、直前のパイプラインの終了ステータスが0の場合にだけ、後続のリストを実行するものです。反対に(| |)は、終了ステータスが0以外の場合にだけ、後続のリストを実行します。コマンドの区切りとして、セミコロンの代わりに任意の数の復帰改行を *list* に指定できます。

コマンド (*command*) は、単純コマンドまたは次のいずれかです。特に断わりのないかぎり、コマンドが返す値は、そのコマンド中で最後に実行された単純コマンドの値です。

for *name* [*in word ...*] **do** *list* **done** *for* コマンドが実行されるたびに、*name* は *in word* リストから次に得られる *word* に設定されます。If *in word ...* を省略すると、*for* コマンドは、設定された各定位値パラメータに対して、**do** *list* を1回実行します(後述の「パラメータ置換」を参照)。リストの *word* がなくなると、実行は終了します。

case *word* **in** [*pattern* [*| pattern*]) *list* ; ;]...**esac**
case コマンドは、*word* に一致する最初の *pattern* に対応した *list* を実行します。パターンの形式は、ファイル名生成に使用される形式と同じです(「ファイル名の生成」のセクションを参照)。ただしスラッシュ、先行するドット、およびスラッシュ直後のドットは、明示的に一致しなくてもかまいません。

if *list* ; **then** *list* **elif** *list* ; **then** *list* ;]...[**else** *list* ;] **fi**

if の後の *list* を実行後、*list* が0の終了ステータスを返すと、最初の **then** の後の *list* を実行します。それ以外の場合、**elif** の後の *list* を実行します。この値が0の場合、次の **then** の後の *list* を実行します。これが失敗すると、**else** *list* を実行します。**else** *list* も **then** *list* も実行しない場合、*if* コマンドは0の終了ステータスを返します。

while *list* **do** *list* **done** *while* コマンドは、*while list* を繰り返し実行し、*list* 中の最後のコマンドの終了ステータスが0の場合、**do** *list* を実行します。それ以外の場合、ループは終了します。**do** *list* 中のコマンドを実行しない場合、*while* コマンドは0の終了ステータスを返します。ループ終了条件の判定を逆にするには、*while* の代わりに *until* を使用します。

(*list*) サブシェル内の *list* の実行

{ *list* ; } 現在の(つまり、親)シェル内での *list* の実行。記号 { の後にはスペースが必要です。

name () { *list* ; } *name* が参照する関数を定義します。{ と } の間のコマンド群 (*list*) が関数の本体になります。記号 { の後にはスペースが必

要です。関数の実行については「実行」のセクションで後述します。関数の本体が、前述の「コマンド」の節で定義したようなコマンドの場合、{ および } は必要ではありません。

下記のワードは、コマンドの最初に現れたとき、およびクォートされずに記述されたときに認識されます。

```
if then else elif fi case esac for while until do done { }
```

注釈行

でワードを始めると、そのワードおよび以降の 復帰改行までの文字がすべて無視されます。

コマンド置換

シェルは、2つの逆引用符 ("") で囲まれた文字列からコマンドを読み取ります。これらのコマンドからの標準出力は、ワードの一部または全体として使用できます。標準出力上で最後につく復帰改行は削除されます。

文字列は、読み取られる前にはいっさい解釈されません。ただし例外として、文字のエスケープに使用されるバックスラッシュ (\) の削除だけは行われます。バックスラッシュは、逆引用符 (') または別のバックスラッシュ (\) をエスケープするためにも使用され、コマンド文字列の読み取り前に削除されます。逆引用符をエスケープすることにより、コマンド置換のネストが可能になります。コマンド置換が、一対の二重引用符に囲まれている場合 ("... '...' ")", 二重引用符 (\") をエスケープするために使用されたバックスラッシュが削除されます。その他の場合は、変更がないままです。

復帰改行文字のエスケープにバックスラッシュを用いた場合は (\newline)、バックスラッシュと復帰改行の両方が削除されます(後述の「クォート」のセクションの後半を参照)。さらに、ドル記号 (\$) をエスケープしているバックスラッシュも削除されます。コマンド文字列に対するパラメータの置換は読み取り前には行われないので、バックスラッシュでドル記号をエスケープしようとしても無意味です。\\、\'、", 復帰改行 (newline)、および \$ 以外の文字の前に付くバックスラッシュは、コマンド文字列の読み取り時にもそのまま残ります。

パラメータ置換

文字 \$ は、置換可能なパラメータ (*parameters*) を示します。パラメータには、定位置パラメータとキーワードパラメータの2種類があります。パラメータが数字の場合、これは定位置パラメータです。定位置パラメータには、set によって値を割り当てることができます。キーワードパラメータ(変数とも呼ばれる)には、次の記述により値を代入することもできます。

```
name=value [ name=value ] ...
```

value に対しては、パターンマッチングは行われません。同じ *name* を持つ関数と変数が存在することはできません。

```
${parameter}
```

パラメータの値(もしあれば)に置き換えられます。中括弧が必要となるのは、パラメータの後に、その名前の一部と

して解釈すべきでない文字、数字、または下線を指定するときだけです。 *parameter* が * または @ の場合、\$1 で始まる定位置パラメータはすべて (スペースで区切られて) 置換されます。パラメータ \$0 は、シェルが呼び出されたときに、引数 0 から設定されます。

- `${parameter:-word}` デフォルト値の使用。 *parameter* が未設定または NULL の場合は、 *word* の展開結果に置換されます。それ以外の場合は、 *parameter* の値に置換されます。
- `${parameter:=word}` デフォルト値の割り当て。 *parameter* が未設定または NULL の場合は、 *word* の展開結果が *parameter* に割り当てられます。どのような場合でも、 *parameter* の最終的な値に置換されます。この割り当て方法は変数だけに使用可能で、定位置パラメータや特殊パラメータには使用できません。
- `${parameter:?word}` *parameter* が NULL 以外の値に設定されている場合、その値に置き換えられます。その他の場合、 *word* を出力しシェルを終了します。 *word* を省略すると、メッセージ “parameter null or not set” が出力されます。
- `${parameter:+word}` *parameter* が NULL 以外の値に設定されている場合、 *word* に置き換えられます。その他の場合は置換を行いません。

前述にあるように、 *word* は、代入用文字列として使用する場合にだけ評価されます。たとえば次の例では、 `pwd` が実行されるのは、 `d` が設定されていないかあるいは NULL に設定されている場合だけです。

```
echo ${d:-'pwd'}
```

前述の式からコロン (:) を省略すると、シェルは *parameter* が設定されているかどうかだけをチェックします。

次のパラメータは、シェルが自動的に設定します。

- # 定位置パラメータ数 (10 進数)
- 呼び出し時に、または `set` コマンドによってシェルに与えられたフラグ。
- ? 最後に同期実行されたコマンドが返した 10 進数
- \$ このシェルのプロセス番号。
- ! 最後に呼び出されたバックグラウンドコマンドのプロセス番号。

次のパラメータはシェルによって使用されます。このセクションのパラメータは、環境変数とも呼ばれます。

HOME	cd コマンドのデフォルト引数 (ホームディレクトリ)。 login(1) によって、パスワードファイルからユーザーのログインディレクトリに設定されます (passwd(4) を参照)。
PATH	コマンド用検索パス (後述の「実行」を参照)。
CDPATH	cd コマンドの検索パスを指定します。
MAIL	このパラメータにメールファイルの名前がセットされていて、かつ MAILPATH パラメータが設定されていない場合、シェルは指定されたファイルにメールが到着するとユーザーに通知します。
MAILCHECK	このパラメータは、MAILPATH または MAIL パラメータで指定されたファイルへメールが到着したか否かを、シェルが何秒ごとにチェックするかを指定します。デフォルト値は 600 秒 (10 分) です。0 が設定された場合、シェルは各プロンプトを出す前にチェックを行います。
MAILPATH	コロン (;) で区切ったファイル名のリスト。このパラメータが設定されると、指定されたいずれかのファイルにメールが到着するたびに、シェルはユーザーに通知します。各ファイル名の後には、% および更新時間の変更時に出力されるメッセージを指定できます。デフォルトのメッセージは you have mail です。
PS1	1 次プロンプト文字列。デフォルトは "\$ " です。
PS2	2 次プロンプト文字列。デフォルトは "> " です。
IFS	内部フィールドセパレータ。通常はスペース、タブ、および復帰改行です (「ブランクの解釈」を参照)。
SHACCT	このパラメータにユーザーが書き込み可能なファイル名が設定された場合、シェルは、実行された各シェルプロシージャールゴとのアカウントレコードをこのファイルに書き込みます。
SHELL	シェルは、呼び出されると、このパラメータが示す名前が環境中に存在するかを確かめます (「環境」のセクションを参照)。

sh の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_CTYPE および LC_MESSAGES。

シェルは、PATH、PS1、PS2、MAILCHECK、および IFS にデフォルト値を割り当てます。HOME および MAIL のデフォルト値は [login\(1\)](#) で設定します。

ブランクの解釈

パラメータとコマンドの置換後、置換の結果内でフィールドセパレータ文字 (IFS で発見されるもの) を検索し、その文字が現れた位置で分割します。分割された各々が引数となります。明示的な NULL 引数 (" " または '') は保持されます。暗示的な NULL 引数 (値を持たないパラメータにより生ずるもの) は削除されます。

入出力のリダイレクション

シェルが解釈する特殊表記法によって、入出力をリダイレクションできます。以下は、単純コマンド内の任意の位置およびコマンドの前後に指定することができ、起動されたコマンドには引数として引き渡されません。注: *word* または *digit* を使用する前にコマンドとパラメータの置換が発生します。

<word *word* というファイルを標準入力 (ファイル記述子 0) として使用します。

>word *word* というファイルを標準出力 (ファイル記述子 1) として使用します。ファイルが存在しない場合は作成します。ファイルが存在していれば、ファイルの長さを 0 にします。

>>word *word* というファイルを標準出力として使用します。ファイルが存在する場合、(EOF までシークしたあと) そのファイルに出力を追加します。そうでない場合は、ファイルを作成します。

< >word *word* というファイルを標準入力として読み書き用に開きます。

<<[-]word *word* に対するパラメータおよびコマンド置換が行われた後、その結果得られた *word* と文字どおり一致する最初の行が現れるまで、または EOF に達するまでシェルへの入力を読み取られます。ただし << に - が付加されて指定された場合は、以下のようになります。

1. まず、シェルへの入力の読み取り前に (ただしパラメータとコマンドの置換後)、先行するタブを *word* から取り除きます。
2. 読み取ったシェルへの入力の各行を *word* と比較する前に、その行から先行するタブを取り除きます。そして、
3. その結果得られた *word* と文字どおり一致する最初の行が現れるまで、または EOF に達するまでシェルへの入力を読み取られます。

word 中のいずれかの文字がクォートされている場合 (後述の「クォート」を参照)、シェルへの入力に対して追加処理は行われません。*word* 中のどの文字もクォートされていない場合は、以下のようになります。

1. まずパラメータとコマンドの置換を実施します。
2. エスケープされた復帰改行 (`\newline`) を削除します。
3. 文字 `\`、`$`、および `'` に対しては、`\` を使ってクォートしなければなりません。

その結果得られるドキュメントが標準入力となります。

<&digit ファイル記述子 *digit* に対応するファイルを標準入力として使用します。同様に、標準出力の使用には **>&digit** を指定します。

<&- 標準入力を閉じます。同様に、標準出力については **>&-** を使用します。

前述のいずれかの前に数字が付く場合、その値が(デフォルトの0または1の代わりに)該当ファイルに対応したファイル記述子となります。例:

```
... 2>&1
```

この例では、現在ファイル記述子1に関連しているファイルに、ファイル記述子2を関連付けます。

リダイレクションを指定する場合、記述する順序が重要になります。シェルは、リダイレクション記述を左から右へ評価します。例:

```
... 1>xxx 2>&1
```

前述の例では、まず `xxx` というファイルにファイル記述子1を関連付けます。次に、ファイル記述子1に関連するファイル(つまり `xxx`)に、ファイル記述子2を関連付けます。リダイレクションの向きが逆であれば、まずファイル記述子2を端末に関連付け(ファイルを記述子1が既に端末に関連付けられているとみなし)、次にファイル記述子1をファイル `xxx` に関連付けます。

最初のページの「コマンド」の節で述べた用語を使って説明すると、次のようになります。コマンドが複数の単純コマンドで構成される場合、リダイレクションは、個々の単純コマンドに対して行う前に、コマンド全体に対して評価されます。すなわち、シェルはまずリスト全体に対してリダイレクションを評価し、次にリスト内の各パイプラインに対して評価し、次にパイプライン内の各コマンドに対して評価し、最後にコマンド内の各単純コマンドに対して評価します。

コマンドの後に `&` を指定すると、コマンドにおけるデフォルトの標準入力はいずれも `/dev/null` という空ファイルになります。それ以外の場合は、コマンドを実行するための環境に起動側シェルのファイル記述子(入出力指定で変更可能)が含まれます。

ファイル名の生成

コマンド実行に先立ち、各コマンドワードは、`*`、`?`、および `[` を含んでいないかチェックされます。これらの文字のいずれかがあると、そのワードはパターンとみなされます。このワードは、パターンと一致する、辞書編集方式の順にソートされたファイル名に置換されます。パターンと一致するファイル名が見つからない場合、ワードは変更されません。ファイル名の先頭のピリオド(`.`)またはスラッシュ(`/`)直後のピリオドは、明示的に一致しなければなりません(後者の場合はスラッシュ自体も含む)。

`*` NULL 文字列を含め、任意の文字列と一致します。

`?` 任意の単一文字と一致します。

`[...]` 括弧内のいずれか1文字を表します。2つの文字を `-` で区切ると、その間にある任意の文字(その2つの文字も含む)に一致します。先頭の `[` の次の文字が `!` である場合、`[]` で囲まれていない任意の文字と一致します。

クォートされているすべての文字(下記「クォート」を参照)は、ファイル名において明示的に一致しなければなりませんので、ご注意ください。

クォート

次の文字はシェルに対しては特別な意味を持ち、クォートしない(後述の説明を参照)限りワードの終わりを表します。

; & () | ^ < > 復帰改行 スペース タブ

これらの文字をクォートする、つまり文字自身を表すには、文字の前にバックスラッシュ(\)を付けるか、または一對の引用符(' 'または" ")で囲みます。処理中、シェルは、特定の文字をクォートして、それらが特別な意味を持たないようにすることができます。単一の文字をクォートするのに用いるバックスラッシュは、コマンド実行前にワードから取り除かれます。\'と復帰改行との組み合わせは、コマンドとパラメータの置換前にワードから取り除かれます。

一對の単一引用符(' ')で囲まれたすべての文字(ただし単一引用符は除く)は、シェルによってクォートされます。バックスラッシュは、一對の単一引用符で囲まれていれば特殊な意味を持ちません。単一引用符は、一對の二重引用符で囲めばクォートされますが(例" ' "), 一對の単一引用符で囲んでもクォートされません。

一對の二重引用符(" ")の中では、パラメータとコマンドの置換が実施され、シェルは、その結果をクォートして、ブランクの解釈とファイル名の生成が行われないうにします。\$*が二重引用符で囲まれている場合は、定位置パラメータは置換され、クォートされ、クォートされたスペースで分けられます("\$1 \$2 ..."). しかし\$@が二重引用符で囲まれている場合は、定位置パラメータは置換され、クォートされ、クォートされていないスペースで分けられます("\$1" "\$2" ...). \は\、\'、\"、,(コンマ)、\$といった文字をクォートします。\'と復帰改行との組み合わせは、コマンドとパラメータの置換前にワードから取り除かれます。バックスラッシュが\、\'、\"、,(コンマ)、\$、および復帰改行以外の文字の前に付く場合は、バックスラッシュ自体がシェルによってクォートされます。

プロンプト

シェルは、対話的に使用すると、コマンドを読み取る前にPS1の値によるプロンプトを発行します。復帰改行を入力したあとで、コマンドを完了するためにさらに入力が必要な場合は、2次プロンプト(PS2の値)が出力されます。

環境

環境(*environment*)は、通常の引数リストが実行されるプログラムに引き渡される場合と同様の方法で引き渡される、名前と値の対の集まりです([environ\(5\)](#)を参照)。シェルが環境と対話する方法はいくつかあります。シェルは、呼び出されると、環境を走査して、見つけた名前ごとに変数を作成し、対応する値を設定します。ユーザーがこれらのパラメータの値を変更したり新しいパラメータを作成したときには、`export` コマンドを用いてシェルのパラメータを環境に関連付けなければ、これらのパラメータは環境に何の影響も与えません(`set -a`の説明を参照)。環境からパラメータを削除するには、`unset` コマンドを使用します。したがって、実行されるコマンドが参照する環境は、シェルが最初に引き継いだ「名前 = 値」の対

のうち変更されていないものから、`unset`によって削除された対を引き、変更または追加した対をくわえたものです。これらはいずれも `export` コマンドで指定する必要があります。

単純コマンドの環境は、いくつかのパラメータ代入指定を先頭に付加すれば拡張できます。したがって、

```
TERM=450  command
```

および

```
(export TERM; TERM=450;  command
```

は、`command` が特殊コマンドでなければ、`command` の実行に関するかぎり同じことを意味します。`command` が特殊コマンドの場合、

```
TERM=450  command
```

現在のシェル内の `TERM` 変数を修正します。

`-k` フラグを設定すると、すべてのキーワード引数は環境に格納されます。これらの引数がコマンド名の後に指定された場合も同様です。以下の例では、まず `a=b c` を、次に `c` を表示します。

```
echo a=b  c
```

```
a=b  c
```

```
set  -k
```

```
echo a=b  c
```

```
c
```

シグナル

起動されたコマンドに対する `INTERRUPT` シグナルと `QUIT` シグナルは、コマンドの後に `&` が付く場合には無視されます。その他の場合、シグナルは、シェルが親から引き継いだ値を持ちます。ただし、シグナル `11` だけは例外です (後述の `trap` コマンドの説明を参照)。

実行

コマンド実行のたびに、前述のコマンドの置換、パラメータの置換、ブランクの解釈、入出力のリダイレクション、およびファイル名の生成が行われます。コマンド名が定義済みの関数名と一致する場合、その関数がシェルスクリプトファイルの実行の間違いに注意)。定義済み関数名とは一致しないが、後述の特殊コマンドのいずれかと一致するコマンド名の場合、そのコマンドがシェルスクリプトファイルの実行の間違いに注意)。定義済み関数名とは一致しないが、後述の特殊コマンドのいずれかと一致するコマンド名の場合、そのコマンドがシェルスクリプトファイルの実行の間違いに注意)。

定位置パラメータの \$1、\$2、... は関数の引数に設定されます。コマンド名が特殊コマンドとも定義済み関数の名前とも一致しない場合、新しいプロセスが作成され、`exec(2)` を用いてそのコマンドの実行が試みられます。

`PATH` というシェルパラメータは、コマンドを含んでいるディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:)で区切ります。`sh` のパスは `/usr/sunos/bin` です。現在のディレクトリは `NULL` パス名によって指定されます。これは等号の直後、パスリスト内の任意の場所にある2つの区切り文字のコロンの間、またはパスリストの最後に記述できます。コマンド名に / が含まれている場合は、検索パスは使用されません。そうでない場合は、パスにおける各ディレクトリに実行可能ファイルがあるか検索します。ファイルが実行権を有するが、それが `a.out` ファイルでない場合、シェルコマンドの入ったファイルとみなされます。そのファイルを読み取るときは、サブシェルが生成されます。括弧で囲まれたコマンドもサブシェル内で実行されます。

シェルは、(あとで不必要な `exec` を行わなくてもいいように) 検索パス内のコマンドの位置を記憶します。コマンドが相対ディレクトリにあった場合、その位置を現在のディレクトリの変更のたびに再決定しなければなりません。シェルは、`PATH` 変数が増え変わるか `hash -r` コマンドが実行されるたびに、記憶していたすべての位置を忘れてしまいます(下記参照)。

特殊コマンド

以下に示す特殊コマンドに対しては、入出力のリダイレクションが可能です。ファイル記述子 1 は、デフォルトの出力位置です。ジョブ制御が可能なのは、さらにいくつかの特殊コマンドがシェル環境に追加されます(「ジョブ制御」のセクションを参照)。

:

コマンドは何もせず、実行による影響は何もありません。終了ステータス 0 が返されます。

`. filename`

`filename` からコマンドを読み取り実行し、戻ります。`PATH` によって指定された検索パスを用いて、`filename` を含むディレクトリを探します。

`bg [%jobid...]`

ジョブ制御が可能なのは、`bg` コマンドはジョブの操作にユーザー環境に追加されます。停止状態のジョブをバックグラウンドで再び実行します。`%jobid` を省略すると、現在のジョブとみなされます。(詳細については後述の「ジョブ制御」のセクションを参照)。

`break [n]`

`for` または `while` ループがあれば抜け出します。`n` を指定すると、`n` レベル分ブレイクします。

`cd [argument]`

現在のディレクトリを `argument` に変更します。シェルパラメータ `HOME` は、`argument` のデフォルト値です。シェルパラメータ `CDPATH` は、`argument` を含

むディレクトリの検索パスを定義します。2つのディレクトリ名は、コロン(:)で区切ります。デフォルトのパスは空の文字列です(現在のディレクトリの指定)。なお現在のディレクトリは空のパス名で指定します。このパス名は、等号の直後か、パスリスト内の区切り文字のコロンの間に指定します。*argument*の先頭文字が/の場合、検索パスは使用しません。それ以外の場合は、パス中の各ディレクトリで*argument*を検索します。

chdir [*dir*]

chdir はシェルの作業用ディレクトリを *dir* が示すディレクトリに変更します。引数を指定しないと、そのユーザーのホームディレクトリに変更します。*dir* が現在のディレクトリからは見つからない相対パス名の場合、変数 **CDPATH** 環境内のディレクトリリストを検索します。*dir* が / で始まる値を持つシェル変数の名前である場合、その変数の値が示すディレクトリに変更します。

continue [*n*]

for または **while** ループの次の繰り返しを実行します。*n* を指定すると、*n* 番目のループから実行します。

echo [*arguments*...]

arguments の文字列がスペース文字に区切られて、シェルの標準出力に書かれます。使用法と説明については **echo(1)** を参照してください。

eval [*argument*...]

引数をシェルへの入力として読み取り、生成されるコマンドを実行します。

exec [*argument*...]

このシェルの代わりに、引数で指定されたコマンドを(新規プロセスは生成せず)実行します。入出力引数が指定可能で、その他の引数が指定されない場合は、これによってシェルの入出力が変更されます。

exit [*n*]

呼び出し元のシェルまたはシェルスクリプトを *n* で指定した終了ステータスで終了させます。*n* を省略すると、最後に実行されたコマンドの終了ステータスがシェルの終了ステータスになります(EOF によっても、シェルは終了します)。

export [*name*...]

指定された *name* 群に対し、後で実行されるコマンドの環境へ自動的にエクスポートされるようにマークを付けます。引数を省略すると、現在のシェル実行中にエクスポートのマークが付けられた変数名を一覧表示します。親シェルからエクスポートされた変数名は、現在のシェル実行中に再びエクスポートされた場合にだけ一覧表示されます。関数名はエクスポートされません。

fg [%*jobid*...]

ジョブ制御が可能なとき、**fg** はジョブの操作用にユーザー環境に追加されます。このコマンドは、フォアグラウンド状態の停止ジョブを再び実行します。また、実行中のバックグラウンドジョブをフォアグラウンドへ移します。%*jobid* を省略すると、現在のジョブとみなされます。(詳細については後述の「ジョブ制御」のセクションを参照。)

getopts

コマンドシンタクス標準のサポート用に、シェルスクリプト内で使用されるコマンドです([Intro\(1\)](#)を参照)。このコマンドは、定位置パラメータを構文解析し、オプションの正当性をチェックします。使用法と説明については、[getoptcvt\(1\)](#)を参照してください。

hash [-r] [name...]

シェルは、各 *name* ごとに、それが示すコマンドの検索パス内の位置を決定し、記憶します。*-r* オプションを指定すると、シェルは記憶したすべての位置を忘れます。引数をまったく指定しないと、記憶されたコマンドに関する情報が表示されます。*hits* はシェルプロセスによってコマンドが呼び出された回数を表示します。*cost* は、検索パスのコマンドを見つけるのに必要な作業です。コマンドが検索パスの「相対」ディレクトリにある場合、そのディレクトリの変更後にそのコマンドが格納された位置が再計算されます。これが行われるコマンドに対しては、*hits* 情報の隣にアスタリスク (*) が示されます。*Cost* の値は、再計算が行われるたびに増加されます。

jobs [-p|-l] [%jobid...]**jobs -x command [arguments]**

停止中またはバックグラウンドで実行中のすべてのジョブを報告します。*%jobid* を省略すると、停止中またはバックグラウンドで稼働中のすべてのジョブが報告されます。(詳細については後述の「ジョブ制御」のセクションを参照。)

kill [-sig] %job...**kill -l**

TERM(終了)シグナルまたは指定されたシグナルのいずれかを、指定されたジョブまたはプロセスに送信します。シグナルは、番号または名前([signal.h\(3HEAD\)](#)に示されるもので、ただし「SIG」という接頭辞を取り除いたもの(CHLDと名付けられるSIGCHD以外))で指定します。送信するシグナルがTERM(終了)またはHUP(ハングアップ)の場合、停止中のジョブまたはプロセスにはCONT(継続)シグナルを送信します。*job* という引数は、活動中のジョブではないプロセスのプロセスIDを指定することもできます。後述の「ジョブ制御」のセクションを参照してください。第2の形式のkill -l は、シグナル番号とシグナル名をリスト表示します。(kill(1)参照)。

login [argument...]

'exec login argument...' と同等です。使用法と説明については、[login\(1\)](#)を参照してください。

newgrp [argument]

exec newgrp argument と同機能です。使用法と説明については、[newgrp\(1\)](#)を参照してください。

pwd

現在の作業用ディレクトリを表示します。使用法と説明については、[pwd\(1\)](#)を参照してください。

`read name...`

標準入力から1行を読み取り、内部フィールドセパレータのIFS (通常はスペースまたはタブ) を用いてワード境界を区切り、最初のワードを最初の *name* に割り当て、2番目のワードを2番目の *name* に割り当て、続くワードも順次割り当てます。残ったワードは最後の *name* に割り当てます。\`\` に続いて復帰改行を入力すれば、行を継続できます。復帰改行以外の文字の前にバックスラッシュを付加すれば、その文字をクォートできます。このバックスラッシュは、ワードが *name* に割り当てられる前に削除され、バックスラッシュの後に位置する文字は解釈されません。EOFに到達した場合を除き、リターンコードは0となります。

`readonly [ name... ]`

指定された各 *name* に「読み取り専用」のマークを付け、これらの名前が後続の割り当てでは変更できないようにします。引数を省略すると、読み取り専用と指定された名前がすべて一覧表示されます。

`return [ n ]`

関数を、*n* が示すリターンステータスで終了させます。*n* を省略すると、リターンステータスは最後に実行されたコマンドのリターンステータスになります。

`set [ -aefhkntuvx [ argument... ] ]`

- a エクスポート用に修正または作成された変数にマークを付けます。
- e コマンドが0以外の終了ステータスで終了した場合、直ちに終了します。
- f ファイル名を生成しないようにします。
- h 関数の定義時に、関数コマンドを検索しその位置を記憶します (通常、関数コマンドは関数実行時に検索されます)。
- k コマンド名の前にあるキーワード引数だけでなく、すべてのキーワード引数がコマンド用の環境に置かれます。
- n コマンドを読み取るが、実行しません。
- t 1つのコマンドを読み取り、実行し、終了します。
- u 未設定の変数を置換時にエラーとして扱います。
- v シェルへの入力行の読み取り時に、その内容を表示出力します。
- x コマンドおよび引数の実行時に、その内容を表示出力します。
- どのフラグも変更しません。`$1` に `-` を設定する際に便利です。

`-` の代わりに `+` を使用すると、これらのフラグは無効になります。これらのフラグはシェル起動時にも使用できます。現在セットされているフラグは、`$-` で見つけられます。残りの引数は定位置パラメータで、次の順に割り当てられます。`$1`、`$2`、... 引数が指定されない場合、すべての名前の値が出力されます。

shift [*n*]

$\$n+1\dots$ から始まる一連の定位置パラメータを、次の名前に変更します。 $\$1\dots$ と同じになります。*n*を省略すると、1と見なされます。

stop *pid*...

pid(プロセス ID 番号)の実行を停止します(**ps(1)**を参照)。

suspend

現在のシェルがログインシェルでない場合、その実行を中断します。

test

条件式を評価します。使用法と説明については、**test(1)**を参照してください。

times

シェルから実行されたプロセスの、ユーザー時間およびシステム時間の累計値を表示します。

trap [*argument n* [*n2*...]]

argument が示すコマンドを、シェルが数値または記号シグナル(*n*)を受信した時に読み取り実行します。(注:*argument* は、トラップ設定時に一度、トラップ取り出し時に一度検索されます。)トラップコマンドは、シグナル番号または対応するシンボリック名の順序で実行されます。現在のシェルへの入力時に無視されたシグナルにトラップを設定しようとしても無効となります。シグナル 11(メモリフォールト)にトラップを指定しようとする、エラーになります。*argument* を省略すると、すべてのトラップ *n* は元の値に再設定されます。*argument* が NULL 文字列の場合、シェルおよびシェルが呼び出したコマンドは、このシグナルを無視します。*n* が 0 の場合、*argument* が示すコマンドはシェル終了時に実行されます。引数なしの **trap** コマンドは、コマンドの一覧を各々が対応しているシグナル番号とともに表示します。

type [*name*...]

各 *name* ごとに、コマンド名として使用される場合にどのように解釈されるかを指示します。

ulimit [[-HS] [-a | -cdfnstv]]

ulimit [[-HS] [-c | -d | -f | -n | -s | -t | -v]] *limit*

ulimit は、リソースの強い制限値または弱い制限値を表示または設定します。これらの制限値については **getrlimit(2)** の説明を参照してください。

limit を省略すると、**ulimit** は指定されている制限値を表示します。制限値は一度にいくつでも表示できます。**-a** オプションは制限値すべてを表示します。

limit 引数を指定すると、**ulimit** は指定されたフラグに対応する制限値をその引数の値に設定します。文字列 **unlimited** は、現在の制限値(ある場合)の削除を要求します。ユーザーはだれでも、強い制限値以下の値に弱い制限値を設定できます。ユーザーはだれでも、強い制限値を下げるができます。適切な特権を持ったユーザーだけが、強い制限値を上げたり、削除したりできます。**getrlimit(2)** を参照してください。

-H オプションは強い制限値を表し、-S オプションは弱い制限値を表します。どちらのオプションも指定しない場合、ulimit は両方の制限値を設定し、弱い制限値を表示します。

次のオプションは、制限値を表示または設定するリソースを指定します。オプションを一切指定しないと、ファイルサイズ限界値を表示または設定します。

- c 最大 core ファイルサイズ (512 バイトブロック単位)
- d データセグメントまたはヒープの最大サイズ (K バイト単位)
- f 最大ファイルサイズ (512 バイトブロック単位)
- n 最大ファイル記述子プラス 1
- s スタックセグメントの最大サイズ (K バイト単位)
- t 最大 CPU 時間 (秒単位)
- v 仮想メモリーの最大サイズ (K バイト単位)

システムで利用可能な最大制限値を調べるには `sysdef(1M)` コマンドを実行してください。表示される値は 16 進数ですが、`bc(1)` コマンドを使って 10 進数に変換できます。`swap(1M)` を参照してください。)

たとえば、0M バイトにコアファイルダンプのサイズを制限するには次のように入力します。

```
ulimit -c 0
```

```
umask [ nnn ]
```

ユーザーファイル作成マスクを *nnn* が示す値に設定します (`umask(1)` を参照)。*nnn* を省略すると、マスクの現在の値を出力します。

```
unset [ name... ]
```

name ごとに、対応する変数または関数値を削除します。変数 PATH、PS1、PS2、MAILCHECK、および IFS は設定解除できません。

```
wait [ n ]
```

当該ユーザーのバックグラウンドプロセスのうち ID が *n* のプロセスを待ち、その終了ステータスを報告します。*n* が省略された場合、当該ユーザーの現在活動中のすべてのバックグラウンドプロセスを待ち、リターンコードは 0 になります。

呼び出し

`exec(2)` を介してシェルが呼び出される場合で、引数 0 の先頭文字が - のとき、コマンドは、まず `/etc/profile` から読み込まれ、次に `$HOME/.profile` から読み込まれます (これらのファイルがある場合)。その後、コマンドは後述のように読み込まれます (シェルが `/usr/sunos/bin/sh` として呼び出される場合も同様)。以下に述べるフラグは、呼び出し時にのみ、シェルによって解釈されます。なお -c または -s フ

ラグが指定されないかぎり、先頭引数はコマンドを含むファイルの名前であるとみなされ、残りの引数は定位置パラメータとしてそのコマンドファイルに引き渡されます。

- c *string* このフラグが指定されると、*string* からコマンドを読み取ります。-
- i このフラグが指定された場合あるいはシェル入出力が端末に接続されている場合、このシェルは対話型となります。-この場合、kill 0 が対話型シェルを終了しないように TERM を無視し、wait が割り込み可能になるように INTERRUPT を捕え、無視します。いずれの場合も、シェルは QUIT を無視します。
- p -p フラグが指定されると、シェルは実効ユーザーおよびグループ ID に、実ユーザーおよびグループ ID を設定しません。
- r -r フラグを指定すると、シェルは制限付きシェルになります ([rsh\(1M\)](#) を参照)。
- s -s フラグが指定された場合、または引数が残っていない場合は、標準入力からコマンドを読み取ります。引数が残っていれば、それらは定位置パラメータを指定します。前述の特殊コマンドの出力を除くシェル出力は、ファイル記述子 2 に書き込まれます。

他のフラグと引数については、前述の set コマンドの箇所ですべて説明されています。

ジョブ制御 (jsh)

シェルの jsh として呼び出すと、sh の説明で述べたすべての機能に加えて、ジョブ制御が可能になります。通常、ジョブ制御は対話型シェルに対してだけ可能です。非対話型シェルは、ジョブ制御の機能が追加されても、その恩恵を受けないのが通常です。

ジョブ制御が可能な場合、ユーザーが端末から入力したコマンドまたはパイプラインは、すべてジョブ (*job*) と呼ばれます。どのジョブも、必ず次のいずれかの状態にあります。フォアグラウンド、バックグラウンド、または停止です。これらの用語の定義を次に示します。

1. フォアグラウンド状態にあるジョブは、制御している端末への読み取り および書き込み権を持っています。
2. バックグラウンド状態にあるジョブは、制御している端末への読み取り権を拒否されていますが、条件付き書き込み権を持っています ([stty\(1\)](#) を参照)。
3. 停止ジョブは保留状態に置かれたジョブであり、通常は SIGTSTP シグナルにより、この状態になります ([signal.h\(3HEAD\)](#) を参照)。

シェルが起動するすべてのジョブには、ジョブ番号 (*job number*) と呼ばれる正の整数が割り当てられます。シェルはこの番号を把握し、特定のジョブを示す識別子として使用します。また、シェルは現在 (*current*) および直前 (*previous*) のジョブも把握しています。現在のジョブとは、最後に起動または再起動されたジョブです。前回のジョブとは、その直前のジョブです。

ジョブ識別子の正しい構文は次のような形式です。

%jobid

このうち *jobid* は、次のいずれかの形式で指定できます。

% または + 現在のジョブ

− 前回のジョブ

?<*string*> コマンド行が *string* を含んでいるジョブ

n ジョブ番号が *n* のジョブ

pref コマンド名の先頭が *pref* のコマンド。たとえば `ls -l name` がバックグラウンドで実行中だった場合、`%ls` と指定すればこのコマンドを示すことができます。*pref* は、クォートしない限り、空白を含めることができません。

ジョブ制御が可能なおとき、ジョブの操作用に次のコマンドがユーザー環境に追加されます。

`bg [%jobid...]` 停止状態のジョブをバックグラウンドで再び実行します。*%jobid* を省略すると、現在のジョブとみなされます。

`fg [%jobid...]` 停止状態のジョブをフォアグラウンドで再び実行します。また、実行中のバックグラウンドジョブをフォアグラウンドへ移します。*%jobid* を省略すると、現在のジョブとみなされます。

`jobs [-p|-l] [%jobid...]`

`jobs -x command`
`[arguments]`

停止中またはバックグラウンドで実行中のすべてのジョブを報告します。*%jobid* を省略すると、停止中またはバックグラウンドで稼働中のすべてのジョブが報告されます。次のオプションは、`jobs` の出力を変更または拡張するために使用します。

-l ジョブのプロセスグループ ID および作業ディレクトリを報告します。

-p ジョブのプロセスグループ ID のみを報告します。

-x コマンドまたは引数中に見つかった *jobid* を、対応するプロセスグループ ID に置き換え、コマンドに引数を渡して実行します。

`kill [-signal] %jobid`

`kill` コマンドの組み込みバージョン。 *jobid* で示すプロセスに対して `kill` コマンドの機能を提供します。

`stop %jobid...`

バックグラウンドジョブの実行を停止します。

suspend

現在のシェルがログインシェルでない場合、その実行を中断します。

wait [%jobid...]

wait コマンドの組み込みバージョンで、ジョブ識別子の指定を受け入れます。%jobid が省略された場合、wait は、前述の「特殊コマンド」で説明したように動作します。

大規模ファイルの動作

ファイルが2G バイト (2³¹ バイト) 以上ある場合の sh と jsh の動作については、[largefile\(5\)](#) を参照してください。

終了ステータス

構文エラーなどのエラーを検出すると、シェルは0以外の終了ステータスを返します。シェルを非対話型で使用している場合、シェルファイルの実行は中止されます。そうでない場合は、シェルは最後に実行されたコマンドの終了ステータスを返します (前述の exit コマンドの説明を参照) 。

jshのみ

シェルが jsh として呼び出された場合、停止ジョブがあるのにシェルを終了させようとすると、シェルは次のような警告を出します。

There are stopped jobs.

これが唯一のメッセージです。もう一度終了が試みられ、停止ジョブがまだ存在している場合、これらのジョブにカーネルから SIGHUP シグナルが送られ、シェルは終了します。

ファイル

\$HOME/.profile

/dev/null

/etc/profile

/tmp/sh*

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/sunos/bin/sh,
/usr/sunos/bin/jsh

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

/usr/xpg4/bin/sh

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目	Intro(1) , bc(1) , echo(1) , getoptcv(1) , kill(1) , ksh88(1) , ksh93(1) , login(1) , newgrp(1) , pfsh(1) , pfexec(1) , ps(1) , pwd(1) , set(1) , shell_builtins(1) , stty(1) , test(1) , umask(1) , wait(1) , rsh(1M) , su(1M) , swap(1M) , sysdef(1M) , dup(2) , exec(2) , fork(2) , getrlimit(2) , pipe(2) , ulimit(2) , setlocale(3C) , signal.h(3HEAD) , passwd(4) , profile(4) , attributes(5) , environ(5) , largefile(5) , XPG4(5)
警告	シェルスクリプトを <code>setuid</code> して使用することは避けてください。
注意事項	入出力のリダイレクションでファイル名に使用されたワードは、ファイル名生成では解釈されません (前述の「ファイル名の生成」のセクションを参照)。たとえば <code>cat file1 >a*</code> という指定は、<code>a*</code> という名前のファイルを生成します。 パイプラインにあるコマンド群はそれぞれ別個のプロセスとして稼動するので、パイプラインに設定された変数は親シェルには何の影響も与えません。 <code>while</code> ループまたは <code>until</code> ループの入力または出力をリダイレクトする場合、ループ内のコマンドはサブシェルで実行されます。サブシェル内で設定または変更された変数は、親プロセスでは効力を失います。 <pre> lastline= while read line do lastline=\$line done < /etc/passwd echo "lastline=\$lastline" # lastline is empty! </pre> このような場合、入力または出力をリダイレクトするには、次の例のように <code>exec</code> を使用します。 <pre> # Save standard input (file descriptor 0) as file # descriptor 3, and redirect standard input from the file /etc/passwd: exec 3<&0 # save standard input as fd 3 exec </etc/passwd # redirect input from file lastline= while read line do lastline=\$line done exec 0<&3 # restore standard input exec 3<&- # close file descriptor 3 echo "\$lastline" # lastline </pre> <code>cannot fork, too many processes</code> というエラーメッセージを受け取った場合には、wait(1) コマンドを用いてユーザーのバックグラウンドプロセスをクリーンアップしてください。これを行なっても問題が解決しない場合には、システムのプ

プロセステーブルが一杯になっていること、または実行中のフォアグラウンドプロセスの数が多すぎることが考えられます。ユーザーログインに結合するプロセス ID の数、およびシステムが把握できる数には限度があります。

パイプラインの最後のプロセスだけが、待つ対象になり得ます。

あるコマンドを実行し、そのあとで同一名のコマンドが、検索パスにおいて元のコマンドがあるディレクトリの前に位置するディレクトリにインストールされた場合、シェルは元のコマンドの方を実行し続けます。新しい方のコマンドを実行させたいければ、`hash` コマンドを使用してください。

Bourne シェルにはプロセスの実効ユーザー ID に対して制限があります。このユーザー ID が 100 よりも小さい (さらにプロセスの実ユーザー ID と同等ではない) 場合には、ユーザー ID はプロセスの実ユーザー ID にリセットされます。

同じプロセスグループでフォアグラウンドジョブとバックグラウンドジョブの両方をシェルが実行しているため、ジョブは同じシグナルを受け取り、予期しない結果を招くことがあります。したがって、特に対話型のシェルを動作している場合は、他のジョブ制御のシェルを使用することをおすすめします。

存在しないコマンドのインタプリタを実行しようとするシェルスクリプトを、シェルが処理した場合、シェルはシェルスクリプトが存在しないという間違った診断メッセージを返します。

名前	shcomp - ksh シェルスクリプトのコンパイル
形式	shcomp [-nv] [<i>infile</i> [<i>outfile</i>]] shcomp -D [<i>infile</i> [<i>outfile</i>]]
機能説明	-D オプションが指定されていない場合、shcomp はシェルスクリプト <i>infile</i> を使用して、ksh が元のスクリプトと同様に読み取りおよび実行を行うバイナリ形式ファイル <i>outfile</i> を作成します。 スクリプトが読み込まれると別名が処理されます。変数の展開が必要な値を持つ別名定義は正常に動作しません。
オプション	サポートしているオプションは、次のとおりです。 -D --dictionary 国際化のためにメッセージカタログに格納する必要がある文字列のリストが生成されます。 このオプションを指定すると、\$ が前に付いた二重引用符で囲まれているすべての文字列が、1 行に 1 つのリテラルという形式で出力されます。リテラル "\$foo" は、出力に "foo" と書き込まれます。これらは、国際化のためにロケール固有のバージョンに変換する必要があるメッセージです。 -n --noexec 廃止された構成または準拠していない構成に対する警告メッセージを表示します。 -v --verbose <i>infile</i> から読み込んだ入力内容を標準エラー出力に書き込みます。
オペランド	次のオペランドがサポートされています。 <i>infile</i> 入力として使用される、シェルスクリプトを含むファイルの名前を指定します。 <i>infile</i> を省略した場合、シェルスクリプトは標準入力から読み込まれます。 <i>outfile</i> 出力ファイルの名前を指定します。 <i>outfile</i> を省略した場合、いずれのモードでも結果は標準出力に書き込まれます。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	下記を参照。

コマンド行インタフェースとシステム変数は「確実」です。コンパイルされたシェルコードの形式は「非公開」です。-d オプションの出力は「流動的」です。

関連項目 [ksh\(1\)](#), [attributes\(5\)](#)

名前 shell_builtins, case, for, foreach, function, if, repeat, select, switch, until, while – シェル組み込み関数

機能説明 シェルコマンドインタプリタである [csh\(1\)](#)、[ksh\(1\)](#)、[ksh88\(1\)](#)、および [sh\(1\)](#) には、特殊な組み込みコマンドが用意されています。シェルは、case、for、foreach、function、if、repeat、select、switch、until、および while をコマンドとして解釈します。これらのコマンドの詳細は、各シェルのマニュアルページの「コマンド」のセクションに記載されています。[ksh\(1\)](#)では、fc、hash、stop、suspend、times、およびtypeはデフォルトでは別名です。

次に示すコマンドは、効率の改善とコマンド間でのデータ共有を可能にするため、シェルに組み込まれています。詳細については、各コマンドのマニュアルページを参照してください。

コマンド	シェル
alarm	ksh
+++alias	csh、ksh88、ksh
bg	csh、ksh88、ksh、sh
+*break	csh、ksh88、ksh、sh
builtin	ksh
case	csh、ksh88、ksh、sh
cd	csh、ksh88、ksh、sh
chdir	csh、sh
command	ksh
+*continue	csh、ksh88、ksh、sh
dirs	csh
disown	ksh
echo	csh、ksh88、ksh、sh
enum	ksh
+*eval	csh、ksh88、ksh、sh
+*exec	csh、ksh88、ksh、sh
+*exit	csh、ksh88、ksh、sh
+++export	ksh88、ksh、sh
false	ksh88、ksh

コマンド	シェル
fc	ksh88、ksh
fg	csch、ksh88、ksh、sh
for	ksh88、ksh、sh
foreach	csch
function	ksh88、ksh
getopts	ksh88、ksh、sh
glob	csch
goto	csch
hash	ksh88、ksh、sh
hashstat	csch
hist	ksh
history	csch
if	csch、ksh88、ksh、sh
jobs	csch、ksh88、ksh、sh
kill	csch、ksh88、ksh、sh
let	ksh88、ksh、
limit	csch
login	csch、ksh88、ksh、sh
logout	csch
nice	csch
+*newgrp	ksh88、ksh、sh
nohup	csch
notify	csch
onintr	csch
popd	csch
print	ksh88、ksh
printf	ksh
pushd	csch

コマンド	シェル
pwd	ksh88, ksh, sh
read	ksh88, ksh, sh
++**readonly	ksh88, ksh, sh
rehash	cs
repeat	cs
+*return	ksh88, ksh, sh
select	ksh88, ksh
+set	cs, ksh88, ksh, sh
setenv	cs
*shift	cs, ksh88, ksh, sh
source	cs
stop	cs, ksh88, ksh, sh
suspend	cs, ksh88, sh
switch	cs
test	ksh88, ksh, sh
time	cs
*times	ksh88, ksh, sh
*+trap	ksh88, ksh, sh
true	ksh88, ksh
type	ksh88, ksh, sh
++**typeset	ksh88, ksh
ulimit	ksh88, ksh, sh
umask	cs, ksh88, ksh, sh
+unalias	cs, ksh88, ksh
unhash	cs
unlimit	cs
+unset	cs, ksh88, ksh, sh
unsetenv	cs

コマンド	シェル
until	ksh88、ksh、sh
vmap	ksh
vpath	ksh
*wait	csh、ksh88、ksh、sh
whence	ksh88、ksh
while	csh、ksh88、ksh、sh

Bourne シェル (sh) の特殊コマンド

以下に示す特殊コマンドに対しては、入出力のリダイレクションが可能です。ファイル記述子 1 は、デフォルトの出力位置です。ジョブ制御が可能なときには、さらにいくつかの特殊コマンドがシェル環境に追加されます。

組み込みの予約コマンド以外に、sh では次のコマンドが使用されます。

- : コマンドは何もせず、実行による影響は何もありません。終了ステータス 0 が返されます。
- .filename filename からコマンドを読み取り実行し、戻ります。PATH によって指定された検索パスを用いて、filename を含むディレクトリを探します。

C シェル (csh)

組み込みコマンドは C シェル内で実行されます。組み込みコマンドが、パイプラインのいずれかのコンポーネント (最後のコンポーネントを除く) として現われると、それはサブシェル内で実行されます。組み込みの予約コマンド以外に、csh では次のコマンドが使用されます。

- : NULL コマンド。このコマンドは解釈されますが、実行はされません。

Korn シェル (ksh88) の特殊コマンド

入出力のリダイレクトが可能です。特に断りのないかぎり、出力はファイル記述子 1 上に書き込まれ、構文エラーがなければ終了ステータスは 0 です。

1 つまたは 2 つのアスタリスク (*) が先頭に付加されているコマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は = 符号のあとに実行され、ワード分割とファイル名生成は実行されません。

組み込みの予約コマンド以外に、ksh88 では次のコマンドが使用されます。

- *:[arg...] パラメータの展開だけを行います。
- *.file[arg...] *file* 全体を読み取ってからコマンドを実行します。コマンドは現在のシェル環境において実行されます。PATHによって指定された検索パスを用いて、*file* を含むディレクトリを探します。引数の *arg* が指定されていれば、それが定位置パラメータになります。そうでない場合、定位置パラメータは変更されません。終了ステータスは、最後に実行されたコマンドの終了ステータスです。

Korn シェル (ksh) の
特殊コマンド

入出力のリダイレクトが可能です。特に断りのないかぎり、出力はファイル記述子 1 上に書き込まれ、構文エラーがなければ終了ステータスは 0 です。

:、true、false、echo、newgrp、および login を除くすべての組み込みコマンドで、オプションの終わりを示す -- を受け入れます。また、オプション --man を、マニュアルページを標準エラーに出力するように求める要求と解釈し、-? を、使用方法に関するメッセージを標準エラーに出力するように求めるヘルプ要求と解釈します。

1 つまたは 2 つの + が先頭に付加されているコマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++ を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は = 符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

組み込みの予約コマンド以外に、ksh では次のコマンドが使用されます。

- : [arg...] パラメータの展開だけを行います。
- .name [arg...] *name* が関数 *name* の予約語構文で定義された関数である場合、この関数は現在の環境で実行されます (*name()* 構文で定義されている場合と同様)。それ以外の場合で *name* がファイルを示しているときは、ファイルの全体が読み取られ、コマンドは現在のシェル環境で実行されます。PATHによって指定された検索パスを用いて、ファイルを含むディレクトリを探します。引数 *arg* が指定されている場合、. コマンドの処理中にこの引数が定位置パラメータになり、終了すると元の定位置パラメータが復元されます。そうでない場合、定位置パラメータは変更されません。終了ステータスは、最後に実行されたコマンドの終了ス

テータスです。

関連項目

[Intro\(1\)](#), [alias\(1\)](#), [break\(1\)](#), [builtin\(1\)](#), [cd\(1\)](#), [chmod\(1\)](#), [csh\(1\)](#), [disown\(1\)](#), [echo\(1\)](#), [exec\(1\)](#), [exit\(1\)](#), [find\(1\)](#), [getoptcv\(1\)](#), [getopts\(1\)](#), [glob\(1\)](#), [hash\(1\)](#), [history\(1\)](#), [jobs\(1\)](#), [kill\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [let\(1\)](#), [limit\(1\)](#), [login\(1\)](#), [logout\(1\)](#), [newgrp\(1\)](#), [nice\(1\)](#), [nohup\(1\)](#), [print\(1\)](#), [printf\(1\)](#), [pwd\(1\)](#), [read\(1\)](#), [readonly\(1\)](#), [set\(1\)](#), [sh\(1\)](#), [shift\(1\)](#), [sleep\(1\)](#), [suspend\(1\)](#), [test\(1\)](#)[test\(1\)](#), [test\(1B\)](#), [time\(1\)](#), [times\(1\)](#), [trap\(1\)](#), [typeset\(1\)](#), [umask\(1\)](#), [wait\(1\)](#), [chdir\(2\)](#), [chmod\(2\)](#), [creat\(2\)](#), [umask\(2\)](#), [getopt\(3C\)](#), [profile\(4\)](#), [environ\(5\)](#)

名前	shift – シェルの引数のリスト、またはフィールドで区切られた単語をずらすためのシェル組み込み関数
形式	
sh	shift [<i>n</i>]
csk	shift [<i>variable</i>]
ksh88	* shift [<i>n</i>]
ksh88	+shift [<i>n</i>]
機能説明	
sh	$\$n+1\dots$ から始まる一連の定位置パラメータを、次の名前に変更します。 $\$1\dots$ と同じになります。 <i>n</i> を指定しない場合、1と見なされます。
csk	<i>argv</i> のコンポーネント (または <i>variable</i> が指定されればその変数のコンポーネント) を、左へずらして最初のコンポーネントを切り捨てます。未設定の変数および NULL 値に対してはエラーとなります。
ksh88	$\\$n+1\ \\$n+1\dots$ の定位置パラメータを $\\$1\dots$ という名前に変更します。<i>n</i> のデフォルト値は1です。パラメータ <i>n</i> に指定できる値は、評価結果が $\\$#$ 以下の負でない数になる算術式です。 このマニュアルページでは、1つまたは2つのアスタリスク (*) が先頭に付加されている ksh88(1) コマンドは、次の方法で特殊な処理を受けます。 1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。 2. 入出力のリダイレクトは変数代入後に行われます。 3. エラーが発生すると、それを含むスクリプトは中止されます。 4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。
ksh	shift は、定位置パラメータを <i>n</i> または 1 (<i>n</i> を省略した場合) で定義された桁数ずつ左にシフトする特別な組み込みシェルです。残りの定位置パラメータの数は、シフトされる桁数だけ削減されます。 <i>n</i> を指定した場合は、シフトする桁数を決定する算術式として評価されます。定位置パラメータの数よりも大きくシフトしたり、負の桁数をシフトしたりすると、エラーが発生します。 ksh の shift では、次の終了値が返されます。 0 正常終了。定位置パラメータは正常にシフトされました。

>0 エラーが発生した。

このマニュアルページでは、1つまたは2つの+(プラス符号)が先頭に付加されている [ksh\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、フィールド分割とファイル名生成は実行されません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

名前	shutdown – close down the system at a given time				
形式	/usr/ucb/shutdown [-fhknr] time [warning-message]...				
機能説明	shutdown provides an automated procedure to notify users when the system is to be shut down. time specifies when shutdown will bring the system down; it may be the word now (indicating an immediate shutdown), or it may specify a future time in one of two formats: +number and hour:min. The first form brings the system down in number minutes, and the second brings the system down at the time of day indicated in 24-hour notation. At intervals that get closer as the apocalypse approaches, warning messages are displayed at terminals of all logged-in users, and of users who have remote mounts on that machine. At shutdown time a message is written to the system log daemon, syslogd(1M), containing the time of shutdown, the instigator of the shutdown, and the reason. Then a terminate signal is sent to init, which brings the system down to single-user mode.				
オプション	As an alternative to the above procedure, these options can be specified: -f Arrange, in the manner of fastboot(1B), that when the system is rebooted, the file systems will not be checked. -h Execute halt(1M). -k Simulate shutdown of the system. Do not actually shut down the system. -n Prevent the normal sync(2) before stopping. -r Execute reboot(1M).				
ファイル	/etc/rmtab remote mounted file system table				
属性	See attributes(5) for descriptions of the following attributes:				
	<table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>compatibility/ucb</td></tr> </tbody> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	compatibility/ucb
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	compatibility/ucb				
関連項目	fastboot(1B) , login(1) , halt(1M) , reboot(1M) , syslogd(1M) , sync(2) , rmtab(4) , attributes(5)				
注意事項	Only allows you to bring the system down between now and 23:59 if you use the absolute time for shutdown.				

名前 size – print section sizes in bytes of object files

形式 size [-f] [-F] [-n] [-o] [-V] [-x] *filename...*

機能説明 The `size` command produces segment or section size information in bytes for each loaded section in ELF object files. `size` prints out the size of the text, data, and bss (uninitialized data) segments (or sections) and their total.

`size` processes ELF object files entered on the command line. If an archive file is input to the `size` command, the information for each object file in the archive is displayed.

When calculating segment information, the `size` command prints out the total file size of the non-writable segments, the total file size of the writable segments, and the total memory size of the writable segments minus the total file size of the writable segments.

If it cannot calculate segment information, `size` calculates section information. When calculating section information, it prints out the total size of sections that are allocatable, non-writable, and not NOBITS, the total size of the sections that are allocatable, writable, and not NOBITS, and the total size of the writable sections of type NOBITS. NOBITS sections do not actually take up space in the *filename*.

If `size` cannot calculate either segment or section information, it prints an error message and stops processing the file.

オプション The following options are supported:

- f Prints out the size of each allocatable section, the name of the section, and the total of the section sizes. If there is no section data, `size` prints out an error message and stops processing the file.
- F Prints out the size of each loadable segment, the permission flags of the segment, then the total of the loadable segment sizes. If there is no segment data, `size` prints an error message and stops processing the file.
- n Prints out non-loadable segment or non-allocatable section sizes. If segment data exists, `size` prints out the memory size of each loadable segment or file size of each non-loadable segment, the permission flags, and the total size of the segments. If there is no segment data, `size` prints out, for each allocatable and non-allocatable section, the memory size, the section name, and the total size of the sections. If there is no segment or section data, `size` prints an error message and stops processing.
- o Prints numbers in octal, not decimal.
- V Prints the version information for the `size` command on the standard error output.
- x Prints numbers in hexadecimal, not decimal.

使用例 The examples below are typical size output.

例1 Producing size information

```
example% size filename
2724 + 88 + 0 = 2812
```

例2 Producing allocatable section size information

```
example% size -f filename
26(.text) + 5(.init) + 5(.fini) = 36
```

例3 Producing loadable segment size information

```
example% size -F filename
2724(r-x) + 88(rwx) + 0(rwx) = 2812 ... (If statically linked)
```

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities

関連項目 [as\(1\)](#), [ld\(1\)](#), [ar.h\(3HEAD\)](#), [a.out\(4\)](#), [attributes\(5\)](#)

注意事項 Since the size of bss sections is not known until link-edit time, the size command will not give the true total size of pre-linked objects.

名前	sleep – 実行の一定期間保留
形式	<code>/usr/bin/sleep interval[d h m s]...</code>
機能説明	<code>sleep</code> は、少なくとも <i>seconds</i> で指定された秒数の間、または SIGALRM シグナルを受信するまで実行を中断します。 <i>seconds</i> オペランドは浮動小数点数として指定できますが、通常、実際の精度はその基盤となるシステムによって異なります。
オペランド	<i>interval</i> 実行を中断する時間を指定する浮動小数点数。浮動小数点数は、C99/XPG6 で必要なすべての形式 (Inf や infinite などの定数を含む) で指定できます。4 つの接尾辞のうち 1 つをオプションで指定できます。これは、指定した数が、日 (d)、時 (h)、分 (m)、または秒 (s) であることを示します。接尾辞を指定しない場合、間隔は秒とみなされます。複数の間隔が指定された場合、それらは合計されます。個々の間隔は負にできますが、合計はゼロ以上である必要があります。
使用例	例1 コマンドの実行の中断 次の例では、特定の時間が経過したあとにコマンドを実行します。 <pre>example% (sleep 105; command)&</pre> 例2 時間ごとのコマンドの実行 次の例では、ある時間ごとにコマンドを実行します。 <pre>example% while true do command sleep 37 done</pre> 例3 コマンドの実行を永久に中断 次の例では、永久に、またはSIGALRM シグナルが受信されるまで、コマンドの実行を中断します。 <pre>example% sleep Inf</pre> 例4 コマンドの実行を、0.5 秒間中断する 代わりに浮動小数点値 "0.5" を使用して、コマンドの実行を 0.5 秒間中断する <pre>example% printf "%a\n" 0.5 0x1.00000000000000000000000000000000p-01 example% sleep 0x1.00000000000000000000000000000000p-01</pre>

例5 実行を23時間中断する

次の例では、文字接尾辞を使用して、実行を23時間中断します。

```
example% sleep 1d -1h
```

環境変数 `sleep` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 *time* で示す秒数以上のあいだ実行を中断することに成功した、または SIGALRM シグナルを受信した(「注意事項」を参照)。

>0 エラーが発生しました。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [wait\(1\)](#), [alarm\(2\)](#), [sleep\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 SIGALRM シグナルを受信すると、`sleep` ユーティリティーは次の動作のうちどちらかを行います。

- 終了ステータス0で正常終了します。
- 事実上、シグナルを無視します。

その他のシグナルを受信した場合には、`sleep` ユーティリティーは標準の動作を行います。

名前 soelim – resolve and eliminate .so requests from nroff or troff input

形式 soelim [*filename*]....

機能説明 soelim reads the specified files or the standard input and performs the textual inclusion implied by the [nroff\(1\)](#) directives of the form:

.so somefile

when they appear at the beginning of input lines.

This is useful as programs such as [tbl\(1\)](#) do not normally do this. It allows the placement of individual tables in separate files to be run as a part of a large document.

An argument consisting of – is taken to be a file name corresponding to the standard input.

Inclusion can be suppressed by using a single quote (') instead of a dot (.) that is,

' so /usr/share/lib/tmac/tmac.s

使用例 例1 Using the soelim Command

The following is an example of the soelim command:

example% soelim exum?.n | tbl | nroff -ms | col | lpr

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

関連項目 [more\(1\)](#), [nroff\(1\)](#), [tbl\(1\)](#), [attributes\(5\)](#)

名前	sort – テキストファイルのソート、マージ、順序の確認
形式	<pre> /usr/bin/sort [-bcdfimMnru] [-k keydef] [-o output] [-S kmem] [-t char] [-T directory] [+pos1 [-pos2]] [file]... /usr/xpg4/bin/sort [-bcdfimMnru] [-k keydef] [-o output] [-S kmem] [-t char] [-T directory] [-y [kmem]] [-z recsz] [+pos1 [-pos2]] [file]...</pre>
機能説明	sort コマンドは、指定されたファイルのすべての行をソートし、その結果を標準出力に書き込みます。 比較は、各入力行から抽出された1つまたは複数のソートキーに基づいて行われます。デフォルトでは、1つのソートキー、つまり入力行全体が使用されます。また、ソートの順は、現在のロケールの照合順序に従います。
オプション	次にあげるオプションは、デフォルトの動作を変更します。
/usr/bin/sort	-c 引数で指定された入力ファイルが、現在のロケールの照合順序に従ってソートされていることを確認します。終了ステータスが設定されますが、正しくソートされていれば何も出力されません。
/usr/xpg4/bin/sort	-c どんな場合でも何も出力されない、という点を除き /usr/bin/sort と同じです。 -y kmem (旧式オプション) sort が最初に使用する主記憶量を指定するために使用されていました。このオプションは、仮想メモリーシステムには適していません。sort が使用するメモリー容量は、-s オプションで指定するようになりました。 -z recsz (旧式オプション) システム固有のデフォルトバッファサイズを超えるような行が存在した場合でも、異常終了を起こさないようにするためのオプションでした。現在では、最長の行が書き込める大きさのバッファを sort が自動的に割り当てるので、このオプションは意味がなくなりました。
/usr/bin/sort and /usr/xpg4/bin/sort	-m マージのみを行います。入力ファイルはソート済みであるものとします。 -o output 標準出力の代わりに使用される出力ファイル名を指定します。このファイルは入力ファイルの1つと同じでも構いません。 -S kmem ソートを行うために使用するスワップメモリー容量の最大値を指定します(デフォルトの単位はキロバイト)。kmem には、バイト(b)、キロバイト(k)、メガバイト(m)、ギガバイト(g)、テラバイト(t)、の数値を直接指定したり、インストールされている物理メモリーに対する割合パーセント(%)で指定することができます。 -T directory directory 引数は、一時ファイルを入れるディレクトリ名です。

- u 同一行を削除します。同じキーを持つ行の組から1行のみを取り出し、他を抑制します。-c オプションと同時に指定すると、入力ファイルがソート済みであることと、同じキーを持つ複数の行が存在しないことの両方を確認します。

並べ換えオプション

デフォルトのソートの並び換え順は、LC_COLLATE の値に依存します。LC_COLLATE が c に設定された場合、ASCII の並べ換え順でソートされます。LC_COLLATE が en_US に設定された場合、ある2つの文字列のうち、一方が他方よりも先に大文字を使用していてその他の点では一様である場合は大文字/小文字を区別しますが、それ以外は大文字/小文字を区別しません。その他のロケールでは、別の並べ換え順でソートされます。

次にあげるオプションは、デフォルトの並べ換え規則を変更します。ソートキーフィールド指定とは無関係に並べ換えオプションがある場合、要求された並べ換え規則は、すべてのソートキーに適用されます。指定された並べ換えオプションが特定のソートキーに付いている場合(「ソートキーオプション」の説明を参照)、このオプションはそのキーに対するすべてのグローバルな並べ換えオプションを無効にします。旧式のキーオプションでは、+pos1 オプションのあとに以下のオプションを1つ以上指定すると、その直前のオプションで指定されたキーフィールドに対してだけ有効になります。

- d 辞書編集順にします。比較では、英字、数字およびスペース(スペースおよびタブ)だけが有効です。
- f 小文字と大文字の区別をしません。
- i 非印字文字を無視します。
- M 月名とみなして比較します。フィールドの先頭3文字(空白文字以外)は大文字に変換し、比較します。たとえば、英語ではソートの並び換え順は JAN < FEB < . . . < DEC です。月名でないフィールドは、"JAN" よりも低く見なされます。-M オプションは -b オプションの動作を含みます(下記を参照)。
- n ソートキーを、省略可能な空白文字、省略可能なマイナス符号、省略可能な小数点および桁区切り文字(現在のロケールに定義されているもの)を持つ0個以上の数値からなる最初の数値列に限定します。キーの値は算術的な数値としてソートされます。空の数値列は、ゼロとみなされます。先行するゼロおよびゼロに付けられた符号は、並べ換えに影響を与えません。
- r 比較の意味を反転します。

フィールド区切り文字オプション

フィールド区切り文字オプションの扱い方は、以下のオプションを使って変更できます。

- b 限定ソートキーの開始および終了位置の決定時に、先行する空白文字を無視します。最初のソートキーオプションの前に -b オプションが指定されている場合、このオプションはすべてのソートキーオプションに適用

されます。それ以外の場合は、オプション引数の `-k field_start, field_end, +pos1, -pos2` の各々に `-b` オプションを個別に指定します(下記を参照)。

`-t char` *char* をフィールド区切り文字として使用します。*char* は、(ソートキーに含まれることがあっても)フィールドの一部とは見なされません。*char* は、繰り返し出てきてもそれぞれが有効です(たとえば、`<char><char>` は空フィールドを区切ります)。`-t` を指定しない場合、デフォルトのフィールド区切り文字として空白文字が用いられます。空白でない文字に続く、最大長の空白文字列が、フィールド区切り文字となります。

ソートキーオプション

ソートキーは以下のオプションを使って指定できます。

`-k keydef` *keydef* 引数は、限定されたソートキーフィールドの定義です。この定義の形式は次のとおりです。

`-k field_start [type] [,field_end [type]]`

各表記の意味は次のとおりです。

field_start と *field_end*

行の一部分に限定したキーフィールドを定義します。

type

文字群 `bdfiMnr` から得られる修飾子です。`b` 修飾子は `-b` オプションと同じ働きですが、対象とした *field_start* または *field_end* だけに適用されます。フィールド内の文字は、フィールド内の最初の空白でない文字から数えられます。なお、これは *first_character* と *last_character* に個別に適用されます。その他の修飾子も、同じ文字で示すオプションと同じ働きですが、対象としたキーフィールドにのみ適用されます。このように動作するのは、*field_start* または *field_end* のどちらか一方、あるいは両方が指定されているときです。*field_start* と *field_end* のいずれかに修飾子が付けられているとき、それらはオプションの対象とはなりません。

複数のキーフィールドがある場合、後にあるキーは、それ以前のキーが等しいと比較された場合にだけ比較されます。`-u` オプションが指定された場合を除き、等しいと比較された行は、`-d`、`-f`、`-i`、`-n`、`-k` のいずれもが省略された場合と同じように並べ換えられます(`-r` オプションは、指定されていれば有効)。また行の中の全バイトを用いて比較されます。

次の表記を見てください。

`-k field_start[type][,field_end[type]]`

これは *field_start* で始まり *field_end* で終わるキーフィールドを定義するものです。ただし *field_start* が行の限界を超えていた

り、*field_end* のあとに位置している場合には、キーフィールドは空となります。*field_end* 指定を省略すると、行の終わりで見なされます。

フィールドは、区切り文字でない最長の文字列で、*-t* オプションが指定されていなければ先行するフィールド区切り文字も含みます。

keydef オプション引数の *field_start* 部分の形式は次のとおりです。

field_number[*.first_character*]

フィールド、およびフィールド内の文字には、1 から始まる番号が付けられます。*field_number* と *first_character* 指定は、いずれも正の 10 進整数と解釈され、ソートキーの部分として用いられる先頭の文字を指定します。*.first_character* 指定を省略すると、フィールドの先頭文字が使用されます。

keydef オプション引数の *field_end* 部分の形式は次のとおりです。

field_number[*.last_character*]

このうち *field_number* の意味は前述の *field_start* のものと同じです。*last_character* 指定は、負でない 10 進整数と解釈され、ソートキーの部分として用いられる最後の文字を指定します。*last_character* の値の評価結果がゼロのとき、または *.last_character* 指定が省略されたときは、*field_number* が示すフィールドの最終文字が使用されます。

-b オプションまたは *b* 修飾子が有効なとき、フィールド中の文字はそのフィールドの最初の空白でない文字から数えられます。なお、これは *first_character* と *last_character* に個別に適用されます。

[*+pos1* [*-pos2*]] (旧式オプション) *-kkeydef* オプションと同等な機能を提供します。

pos1 および *pos2* はそれぞれ *m.n* という形式をとり、オプションでフラグ *bdfiMnr* の中から 1 つまたは複数のフラグを後に付けることができます。*+m.n* によって指定される開始位置は、*m+1* 番目のフィールドにある *n+1* 番目の文字であると解釈されます。*.n* がない場合は、*m+1* 番目のフィールドの先頭文字を指す *.0* を意味します。*b* フラグが有効な場合、*m+1* 番目のフィールドの最初の空白文字以外の文字から *n* が数えられます。*+m.0b* は *m+1* 番目のフィールド内の最初の空白文字以外の文字を指します。

-m.n によって指定される最終位置は、*m* 番目のフィールドの最後の文字から後へ *n* 番目の文字 (区切り文字も含む) を意味するように解釈されます。*.n* がない場合、*m* 番目のフィールドの最後の

文字を指す .0 を意味します。b フラグが有効な場合、 $m+1$ 番目のフィールドの前についている最後の空白文字から n が数えられます。したがって、 $-m.1b$ は、 $m+1$ 番目のフィールド内の最初の空白文字以外の文字を指します。

次の記述は、 $+pos1-pos2$ の形式をタイプ修飾子 T と U とともに指定したものです。

```
+w.xT -y.zU
```

次と同等です。

```
undefined (z==0 & U contains b & -t is present)
-k w+1.x+1T,y.0U      (z==0 otherwise)
-k w+1.x+1T,y+1.zU    (z > 0)
```

ソートキー (-k オプションおよび旧式の $+pos1$ と $-pos2$) は少なくとも 9 個記述できます。複数個ある場合、コマンド行での順序が意味を持ちます。ソートキーを 1 つも指定しないと、行全体のデフォルトのソートキーが使用されます。

オペランド

次のオペランドを指定できます。

file ソート、マージ、または確認するファイルのパス名。*file* オペランドを 1 つも指定しない場合、または *file* オペランドに $-$ を指定した場合には、標準入力 が用いられます。

使用法

ファイルが 2G バイト (2^{31} バイト) 以上ある場合の `sort` の動作については、[largefile\(5\)](#) を参照してください。

使用例

以下の例では、`sort` キーを指定する方法のうち旧式のものと旧式でないものの両方を示します。これにより両形式の関係が理解しやすくなるはずです。

例1 第2フィールドをソートキーとしてソートする

第2フィールドをソートキーとして用いて、`infile` の内容をソートします。

```
example% sort -k 2,2 infile
example% sort +1 -2 infile
```

例2 逆の順序でソートする

第2フィールドの2番目の文字をソートキーとして、`infile1` および `infile2` の内容を逆の順序でソートし、`outfile` に出力します。なお第2フィールドの先頭文字はフィールド区切り文字であると想定しています。

```
example% sort -r -o outfile -k 2.2,2.2 infile1 infile2
example% sort -r -o outfile +1.1 -1.2 infile1 infile2
```

例3 ファイル中の特定の1文字をキーとしてソートする

第2フィールドの2番目の空白でない文字をソートキーとして、infile1とinfile2の内容をソートします。

```
example% sort -k 2.2b,2.2b infile1 infile2
```

```
example% sort +1.1b -1.2b infile1 infile2
```

例4 ユーザーIDでソートする

ユーザーID(コロンで区切った3番目のフィールド)でソートされたパスワードファイル [passwd\(4\)](#) を出力します。

```
example% sort -t : -k 3,3n /etc/passwd
```

```
example% sort -t : +2 -3n /etc/passwd
```

例5 フィールドが重複する行を除外してソートされた行を出力する

ソート済みファイルinfileの行の出力において、同じ第3フィールドを持つ行のうち、最初に現われる行だけを出力します。

```
example% sort -um -k 3.1,3.0 infile
```

```
example% sort -um +2.0 -3.0 infile
```

例6 ホストIPアドレスでソートする

次のコマンドは、どちらも数値形式のIPアドレス(先頭から4番目までの数値フィールド)でソートされた [hosts\(4\)](#) ファイル(IPv4ホストデータベース)を出力します。

```
example$ sort -t . -k 1,1n -k 2,2n -k 3,3n -k 4,4n /etc/hosts
```

```
example$ sort -t . +0 -1n +1 -2n +2 -3n +3 -4n /etc/hosts
```

'.'は、フィールドの区切り文字であると同時に、多くのロケールで10進数の区切り文字として使用されています。そのため、フィールドの末尾の指定に失敗すると、2つめのフィールドが1つめのフィールドの小数部として認識されてしまうといった問題が発生します。

環境変数

sortの実行に影響を与える次の環境変数については、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_COLLATE、LC_MESSAGES、およびNLSPATH。

LC_CTYPE テキストデータのバイト列を文字(たとえば、引数や入力ファイル中のシングルバイト文字や複数バイト文字)として解釈するロケールや、-b、-d、-f、-i、-nの各オプション用に、文字分類の方法を指定するロケールを定義します。

LC_NUMERIC -nオプション用に、小数点や桁区切り文字を指定するロケールを定義します。

- 終了ステータス 次の終了ステータスが返されます。
- 0 入力ファイルはすべて正常に出力された、または `-c` が指定され入力ファイルは正しくソートされていた。
 - 1 `-c` オプションが指定され入力ファイルは指定どおりにソートされていなかった、または `-c` と `-u` の両オプションが指定され2つの入力行のキーが等しかった。
 - >1 エラーが発生した。
- ファイル `/var/tmp/stm???` 一時ファイル
- 属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [comm\(1\)](#), [join\(1\)](#), [uniq\(1\)](#), [nl_langinfo\(3C\)](#), [strftime\(3C\)](#), [hosts\(4\)](#), [passwd\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

診断 さまざまなトラブル(たとえば、入力行が長すぎる)が発生した場合や `-c` オプションにおいてソート済みでないことを発見した場合、コメントを出力し、0以外の終了ステータスで終了します。

注意事項 入力ファイルの最終行に復帰改行文字 (NEWLINE) がない場合、`sort` はこれを追加し、警告メッセージを出力し、処理を継続します。

`sort` で、等しいキーを持つ行の相対的な順序が保存されるという保証はありません。

`-s` オプションを使用して、特定の用途に `sort` の動作を調整することができます。ただし、`sort` は仮想メモリーシステムよりも、ソート時により適切に膨大な量のメモリーを利用できるように注意してください。このため、`-s` を使用して、膨大なメモリーを利用するようにしてソートを実行すると、パフォーマンスがかなり低くなることもあります。

前述のように、フィールド修飾子の一部 (-M や -d など) は、入力データをロケール固有の設定と照合して解釈します。ロケール固有の設定を考慮に入れていないと、解釈の結果が予期しないものになることがあります。たとえば月の名前をキーにする場合、一般に通用する省略名を使用しても、`sort` コマンドはそれを正しい名前に読み換えることはしません。`sort` コマンドが認識する省略名は、`nl_langinfo(3C)` または `strftime(3C)` で定義されているものだけです。表示可能順や辞書編集順でソートする場合、このような定義がロケールで適切に設定されていないと、空のソートキーが返されることがあります。結果として、次のキーが適切な並べ替えを決定する有効なキーになります。

名前 sortbib – sort a bibliographic database

形式 sortbib [-s *KEYS*] *database...*

機能説明 sortbib sorts files of records containing refer key-letters by user-specified keys. Records may be separated by blank lines, or by ‘. [’ and ‘.]’ delimiters, but the two styles may not be mixed together. This program reads through each *database* and pulls out key fields, which are sorted separately. The sorted key fields contain the file pointer, byte offset, and length of corresponding records. These records are delivered using disk seeks and reads, so sortbib may not be used in a pipeline to read standard input.

The most common key-letters and their meanings are given below.

%A	Author's name
%B	Book containing article referenced
%C	City (place of publication)
%D	Date of publication
%E	Editor of book containing article referenced
%F	Footnote number or label (supplied by refer)
%G	Government order number
%H	Header commentary, printed before reference
%I	Issuer (publisher)
%J	Journal containing article
%K	Keywords to use in locating reference
%L	Label field used by -k option of refer
%M	Bell Labs Memorandum (undefined)
%N	Number within volume
%O	Other commentary, printed at end of reference
%P	Page number(s)
%Q	Corporate or Foreign Author (unreversed)
%R	Report, paper, or thesis (unpublished)
%S	Series title
%T	Title of article or book
%V	Volume number
%X	Abstract — used by roffbib, not by refer

%Y,Z Ignored by refer

By default, sortbib alphabetizes by the first %A and the %D fields, which contain the senior author and date.

sortbib sorts on the last word on the %A line, which is assumed to be the author's last name. A word in the final position, such as 'jr.' or 'ed.', will be ignored if the name beforehand ends with a comma. Authors with two-word last names or unusual constructions can be sorted correctly by using the nroff convention '\0' in place of a blank. A %Q field is considered to be the same as %A, except sorting begins with the first, not the last, word. sortbib sorts on the last word of the %D line, usually the year. It also ignores leading articles (like 'A' or 'The') when sorting by titles in the %T or %J fields; it will ignore articles of any modern European language. If a sort-significant field is absent from a record, sortbib places that record before other records containing that field.

No more than 16 databases may be sorted together at one time. Records longer than 4096 characters will be truncated.

オプション -sKEYS Specify new KEYS. For instance, -sATD will sort by author, title, and date, while -sA+D will sort by all authors, and date. Sort keys past the fourth are not meaningful.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/doctools

関連項目 [addbib\(1\)](#), [indxbib\(1\)](#), [lookbib\(1\)](#), [refer\(1\)](#), [roffbib\(1\)](#), [attributes\(5\)](#)

使用上の留意点 Records with missing author fields should probably be sorted by title.

名前	sotruss – 共有ライブラリプロシージャ呼び出しの追跡
形式	<code>/usr/bin/sotruss [-f] [-F bindfromlist] [-T bindtolist] [-o outputfile] executable [executable arguments...]</code>
機能説明	sotruss は、指定されたコマンドを実行し、実行したライブラリ呼び出しのトレースを生成します。トレース出力の各行では、各プロシージャ呼び出しが実行されたときに動的オブジェクト間で発生する結合が報告されます。sotruss は、プロシージャリンケージテーブルを介して動的オブジェクト間で発生するすべてのプロシージャ呼び出しを追跡します。そのため、そのプロシージャリンケージテーブルを介して結合されているプロシージャ呼び出しだけが追跡されます。『 リンカーとライブラリ 』を参照してください。
オプション	-F bindfromlist 追跡するライブラリのコロンで区切られたリスト。これらのライブラリからの呼び出しだけが追跡されます。デフォルトでは、メインの実行可能オブジェクトからの呼び出しだけが追跡されます。 -T bindtolist 追跡するライブラリのコロンで区切られたリスト。これらのライブラリへの呼び出しだけが追跡されます。デフォルトでは、すべての呼び出しが追跡されます。 -o outputfile sotruss の出力を <i>outputfile</i> に送ります。このオプションを -f オプションと組み合わせると、実行しているプログラムの <i>pid</i> がファイル名の後ろに付けられます。デフォルトでは、sotruss の出力は <code>stderr</code> に書き込まれます。 -f <code>fork()</code> によって作成されたすべての子プロセスを追跡し、各子プロセスの <code>truss</code> 出力を書き込みます。このオプションを指定すると、各 <code>truss</code> の出力行に <i>pid</i> が出力されます。

使用例

例1 sotruss の例。

次の簡単な例では、単純な `ls` コマンドの追跡を示します。

```
% sotruss ls | more
ls      ->  libc.so.1:*atexit(0xef7d7d1c, 0x23c00, 0x0)
ls      ->  libc.so.1:*atexit(0x1392c, 0xef7d7d1c, 0xef621bb0)
ls      ->  libc.so.1:*setlocale(0x6, 0x1396c, 0xef621ba8)
ls      ->  libc.so.1:*textdomain(0x13970, 0x1396c, 0xef621ba8)
ls      ->  libc.so.1:*time(0x0, 0xef61f6fc, 0xef621ba8)
ls      ->  libc.so.1:*isatty(0x1, 0xef61f6fc, 0x0)
ls      ->  libc.so.1:*getopt(0x1, 0xfffff8fc, 0x13980)
ls      ->  libc.so.1:*malloc(0x100, 0x0, 0x0)
ls      ->  libc.so.1:*malloc(0x9000, 0x0, 0x0)
ls      ->  libc.so.1:*lstat64(0x23ee8, 0xfffff7a0, 0x0)
...
ls      ->  libc.so.1:*printf(0x13a64, 0x26208, 0x23ef0)
ls      ->  libc.so.1:*printf(0x13a64, 0x26448, 0x23ef0)
```

例1 sotruss の例。 (続き)

```
ls      ->    libc.so.1:*exit(0x0, 0x24220, 0x2421c)
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities

関連項目 [ld.so.1\(1\)](#), [truss\(1\)](#), [whocalls\(1\)](#), [fork\(2\)](#), [attributes\(5\)](#)
『リンカーとライブラリ』

名前 spell, hashmake, spellin, hashcheck – report spelling errors

形式 spell [-bilvx] [+ *local_file*] [*file*] ...

/usr/lib/spell/hashmake

/usr/lib/spell/spellin *n*

/usr/lib/spell/hashcheck *spelling_list*

機能説明

The `spell` command collects words from the named files and looks them up in a spelling list. Words that neither occur among nor are derivable (by applying certain inflections, prefixes, or suffixes) from words in the spelling list are written to the standard output.

If there are no file arguments, words to check are collected from the standard input. `spell` ignores most `troff(1)`, `tbl(1)`, and `eqn(1)` constructs. Copies of all output words are accumulated in the history file (`spellhist`), and a stop list filters out misspellings (for example, `their=thy-y+ier`) that would otherwise pass.

By default, `spell` (like `deroff(1)`) follows chains of included files (`.so` and `.nx troff(1)` requests), unless the names of such included files begin with `/usr/lib`.

The standard spelling list is based on many sources, and while more haphazard than an ordinary dictionary, is also more effective in respect to proper names and popular technical words. Coverage of the specialized vocabularies of biology, medicine and chemistry is light.

Three programs help maintain and check the hash lists used by `spell`:

<code>hashmake</code>	Reads a list of words from the standard input and writes the corresponding nine-digit hash code on the standard output.
<code>spellin</code>	Reads <i>n</i> hash codes from the standard input and writes a compressed spelling list on the standard output.
<code>hashcheck</code>	Reads a compressed <i>spelling_list</i> and recreates the nine-digit hash codes for all the words in it. It writes these codes on the standard output.

オプション

The following options are supported:

<code>-b</code>	Check British spelling. Besides preferring <i>centre</i> , <i>colour</i> , <i>programme</i> , <i>speciality</i> , <i>travelled</i> , and so forth, this option insists upon <i>-ise</i> in words like <i>standardise</i> .
<code>-i</code>	Cause <code>deroff(1)</code> to ignore <code>.so</code> and <code>.nx</code> commands. If <code>deroff(1)</code> is not present on the system, then this option is ignored.
<code>-l</code>	Follow the chains of <i>all</i> included files.
<code>-v</code>	Print all words not literally in the spelling list, as well as plausible derivations from the words in the spelling list.
<code>-x</code>	Print every plausible stem, one per line, with <code>=</code> preceding each word.

+local_file Specify a set of words that are correct spellings (in addition to `spell`'s own spelling list) for each job. *local_file* is the name of a user-provided file that contains a sorted list of words, one per line. Words found in *local_file* are removed from `spell`'s output. Use [sort\(1\)](#) to order *local_file* in ASCII collating sequence. If this ordering is not followed, some entries in *local_file* might be ignored.

オペランド The following operands are supported:

`file` A path name of a text file to check for spelling errors. If no files are named, words are collected from the standard input.

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of `spell`: `LC_CTYPE`, `LC_MESSAGES`, and `NLSPATH`.

終了ステータス The following exit values are returned:

`0` Successful completion.

`>0` An error occurred.

ファイル

<code>D_SPELL=/usr/lib/spell/hlist[ab]</code>	hashed spelling lists, American & British
<code>S_SPELL=/usr/lib/spell/hstop</code>	hashed stop list
<code>H_SPELL=\$HOME/.spellhist</code>	history file
<code>/usr/share/lib/dict/words</code>	master dictionary

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/spelling-utilities

関連項目 [deroff\(1\)](#), [eqn\(1\)](#), [sort\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 `spell` works only on English words defined in the U.S. ASCII codeset.

使用上の留意点 The spelling list's coverage is uneven. New installations might wish to monitor the output for several months to gather local additions.

British spelling was done by an American.

Misspelled words can be monitored by default. To do so, set the `H_SPELL` environment variable to the name of a file which is writable to the `spell` process. If `H_SPELL` is not set, `$HOME/.spellhist` is used as the history file. If no monitoring is desired, one can create the appropriate `spell` history file with write permission disabled.

名前	split – ファイルを複数に分割
形式	<pre>split [-linecount -l linecount] [-a suffixlength] [file [name]] split [-b n nk nm] [-a suffixlength] [file [name]]</pre>
機能説明	split ユーティリティーは、<i>file</i> で示すファイルを読み取り、それを <i>linecount</i> で示す数の行に分けて出力ファイル群に書き込みます。最初の出力ファイル名は、<i>name</i> に <i>aa</i> を追加したもので、その後辞書編集順どおりに <i>zz</i> までを付けた名前となります (最大 676 ファイル)。<i>name</i> の最大長は、ファイルシステムによって認められた最大ファイル名の長さ未満の 2 文字です。statvfs(2) を参照してください。出力名が指定されていない場合、<i>x</i> をデフォルトとして使用します (出力ファイルは <i>xaa</i> または <i>xab</i> などとなります)。
オプション	サポートしているオプションは、次のとおりです。 -linecount -l linecount 各断片の行数。デフォルトは 1000 行。 -a suffixlength 各出力ファイルの接尾辞部分の長さを <i>suffixlength</i> 文字にします。この -a オプションを指定しない場合、接尾辞のデフォルト長は 2 文字となります。<i>name</i> オペランドで指定したファイル名の長さと接尾辞の長さの合計が NAME_MAX バイトを超えてしまう場合には、エラーとなります。このとき split は診断メッセージを発行して処理を終了します。ファイルは生成されません。 -b n 各出力ファイルのサイズを <i>n</i> バイトとします。 -b nk 各出力ファイルのサイズを <i>n</i>*1024 バイトとします。 -b nm 各出力ファイルのサイズを <i>n</i>*1 048 576 バイトとします。
オペランド	次のオペランドがサポートされています。 file 入力となる通常ファイルのパス名。このオペランドを省略するか - を指定すると、標準入力とみなされます。 name split の実行により生成される各出力ファイルの接頭辞を指定します。<i>name</i> オペランドを省略すると、<i>x</i> が接頭辞として使用されます。この接頭辞のベース名の長さと <i>suffixlength</i> 値の合計は、NAME_MAX バイトを超えることはできません。詳しくは「オプション」の項を参照してください。
使用法	ファイルが 2G バイト (2³¹ バイト) 以上ある場合の split の動作については、largefile(5) を参照してください。
環境変数	split の実行に影響を与える次の環境変数についての詳細は、environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

- 終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[csplit\(1\)](#), [statvfs\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	srchtxt – メッセージデータベースの内容表示、またはそこからのテキスト文字列の検索
形式	srchtxt [-s] [-l <i>locale</i>] [-m <i>msgfile</i> ,...] [<i>text</i>]
機能説明	srchtxt ユーティリティーは、メッセージデータベース内のすべてのテキスト文字列を表示したり、あるいはメッセージデータベース内のテキスト文字列を検索したりします(mkmsgs(1)を参照)。これらのデータベースは、-m オプションで設定したファイル名に/が含まれていないかぎり、ディレクトリ /usr/lib/locale/locale/LC_MESSAGES のファイルにあります(setlocale(3C)を参照)。ディレクトリ <i>locale</i> をそのテキスト文字列に使用している言語名とみなすことができます。-l オプションが指定されていない場合、アクセスされるファイルは、環境変数 LC_MESSAGES の値によって決まります。LC_MESSAGES が設定されていない場合、アクセスされるファイルは、環境変数 LANG の値によって決まります。LANG が設定されていない場合、アクセスされるファイルは、デフォルト文字列を含むディレクトリ /usr/lib/locale/C/LC_MESSAGES にあるものになります。 <i>text</i> 引数がない場合、アクセスされたファイル内のすべてのテキスト文字列が表示されます。 -s オプションが指定されていない場合、表示されたテキストの前にメッセージシーケンス番号が付きます。メッセージシーケンス番号は、次のように山括弧で囲まれます。<<i>msgfile</i>:<i>msgnum</i>>。 <i>msgfile</i> 表示されたテキストのあるファイル名 <i>msgnum</i> 表示されたテキストのある <i>msgfile</i> 内のシーケンス番号 これは、gettext(1) および gettext(3C) で使用されたフォーマットで表示されます。
オプション	-s 表示中のメッセージのメッセージシーケンス番号を表示しません。 -l <i>locale</i> ディレクトリ /usr/lib/locale/locale/LC_MESSAGES 内のファイルへアクセスします。-m <i>msgfile</i> も提供された場合、/を含む <i>msgfiles</i> では <i>locale</i> が無視されます。 -m <i>msgfile</i> 1 つないしは複数の <i>msgfile</i> によって指定されたファイルへアクセスします。<i>msgfile</i> に / 文字が含まれている場合、<i>msgfile</i> はパス名として解釈され、それ以外の場合、前述のように決定されたディレクトリ内にあると見なされます。複数の <i>msgfile</i> を指定するには、コンマを用いてファイル名を区切ります。 <i>text</i> <i>text</i> によって指定されたテキスト文字列を検索し、一致する個々の文字列を表示します。<i>text</i> は一般の表現形式をとることができます。regex(5) を参照。

使用例

例1 srchtxtの使用例

`mkmsgs(1)` を用いて、`french` という名のロケールにメッセージファイルがインストールされた場合、ユーザーは次のように入力すると、`french` のロケール (`/usr/lib/locale/french/LC_MESSAGES/*`) のテキスト文字列セット全体を表示できます。

```
example% srchtxt -l french
```

例2 srchtxtの使用例

オペレーティングシステムに関連するエラーメッセージのセットが `french` のロケール用の `UX` というファイル (`/usr/lib/locale/french/LC_MESSAGES/UX`) にインストールされている場合、`LANG` 環境変数の値を用いて、検索するロケールを決定し、次のように入力すると、そのロケール内の該当ファイルから、ファイルに関するすべてのエラーメッセージを検索できます。

```
example% setenv LANG=french; export LANG
```

```
example% srchtxt -m UX "[Ff]ichier"
```

`/usr/lib/locale/french/LC_MESSAGES/UX` に次の文字列が含まれていた場合、

```
Erreur E/S\n
Liste d'arguments trop longue\n
Fichier inexistant\n
Argument invalide\n
Trop de fichiers ouverts\n
Fichier trop long\n
Trop de liens\n
Argument hors du domaine\n
Identificateur supprim\n
Etreinte fatale\n
.\n
.\n
.
```

次の文字列が表示されます。

```
<UX:3>Fichier inexistant\n
<UX:5>Trop de fichiers ouverts\n
<UX:6>Fichier trop long\n
```

例3 srchtxtの使用例

オペレーティングシステムに関連するエラーメッセージ・セットがファイル `UX` にインストールされており、また `INGRESS` データベース製品に関連するエラーメッセージ・セットがファイル `ingress` にインストールされていた場合 (両者とも `german` ロケール内にあるとして)、次のように入力することによって、`german` ロケール内の両ファイル、`UX` と `ingress` でパターン `[Dd]atei` を検索できます。

例3 srchtxtの使用例 (続き)

```
example% srchtxt -l german -m UX,ingress "[Dd]atei"
```

- 環境変数
- srchtxtの実行に影響を与える環境変数 LC_CTYPE についての詳細は、[environ\(5\)](#)を参照してください。
- ファイル
- /usr/lib/locale/C/LC_MESSAGES/*

[mkmsgs\(1\)](#)によって作成されたデフォルトのファイル

/usr/lib/locale/locale/LC_MESSAGES/*

[mkmsgs\(1\)](#)によって作成されたメッセージファイル
- 属性
- 属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

属性タイプ	属性値
使用条件	text/locale

- 関連項目
- [exstr\(1\)](#), [gettext\(1\)](#), [locale\(1\)](#), [mkmsgs\(1\)](#), [gettext\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [locale\(5\)](#), [regex\(5\)](#)
- 診断
- srchtxtによって出力されたエラーメッセージは明白で詳しく解説しなくても理解できるでしょう。このメッセージは、 コマンド行内のエラーまたは特定のロケールまたはメッセージファイル(あるいは両方)を検索中に発生したエラーを指します。

名前 ssh - SSH クライアント (リモートログインプログラム)

形式 ssh [-l login_name] hostname | user@hostname [command]
ssh [-afgknqstvxACNTX1246] [-b bind_address] [-m mac_spec]
[-c cipher_spec] [-e escape_char] [-i identity_file]
[-i PKCS#11-URI]
[-l login_name] [-F configfile] [-o option] [-p port]
[-L [bind_address:]port:host:hostport]
[-R [bind_address:]port:host:hostport]
[-D [bind_address:]port] hostname | user@hostname [command]

機能説明

ssh (Secure Shell) は、リモートマシンにログインしたり、リモートマシン上でコマンドを実行したりするためのプログラムです。これは、rlogin や rsh に代わるものとして開発され、安全でないネットワーク上にある 2 台の信頼できないホスト間において、暗号化された安全な通信を提供します。X11 接続と任意の TCP/IP ポートも、セキュリティ保護されたチャネルを介して転送されます。

この ssh の実装は、SSH プロトコルバージョン 1 と 2 を同時にサポートします。v1 プロトコルのセキュリティ上の弱点のため、可能な場合は v2 のみ実行するようにしてください。

v1 用のサポートは、ssh v1 サーバーが存在するサイトが v2 に移行することをサポートするために提供されています。v1 は今後のリリースでサポートされなくなる可能性があります。

ssh は指定されたホスト名に接続してログインします。ユーザーは、リモートマシンに対して本人であることを証明する必要があります。これには、使用するプロトコルのバージョンに応じて、いくつかの方法のうちの 1 つを使用します。

SSH プロトコルバージョン 1

まず、ユーザーがリモートマシン上の /etc/hosts.equiv または /etc/shosts.equiv に記述されているマシンからログインし、かつそのユーザーの名前が両方の側で同じである場合、そのユーザーはすぐにログインが許可されます。次に、リモートマシン上のそのユーザーのホームディレクトリに .rhosts または .shosts が存在し、かつその中にクライアントマシンの名前とそのマシン上におけるユーザー名を記述した行が含まれている場合、そのユーザーはログインが許可されます。この形の認証は安全ではないため、通常、この認証のみの使用はサーバーによって認められません。

2 番目 (およびプライマリ) の認証方式は、rhosts 方式または hosts.equiv 方式を RSA ベースのホスト認証と組み合わせたものです。これ

は、\$HOME/.rhosts、\$HOME/.shosts、/etc/hosts.equiv、または /etc/shosts.equiv によってログインが許可されていて、さらにサーバーがクライアントのホスト鍵 (「ファイル」セクションの「/etc/ssh_known_hosts」を参照) を検証できる場合にだけ、ログインが許可されるということです。この認証方式では、IP スプーフィング、DNS スプーフィング、およびルーティングスプーフィングによるセキュリティホールをふさぎます。

管理者の方への注意: `/etc/hosts.equiv`、`$HOME/.rhosts`、および一般的な `rlogin/rsh` プロトコルは本質的に安全ではないため、セキュリティが要求される場合は無効にするべきです。

3 番目の認証方式として、`ssh` は RSA ベースの認証をサポートしています。この手法は公開鍵暗号方式に基づいています。暗号方式の中には、暗号化と復号化がそれぞれ異なる鍵を使って行われ、暗号化鍵から復号化鍵を導き出せないものがあります。RSA はそのような暗号方式の 1 つであり、次のような仕組みに基づいて認証を行います。まず、各ユーザーは認証のための公開鍵と秘密鍵のペアを作成します。サーバーは公開鍵を知っていますが、秘密鍵を知っているのはユーザーだけです。`$HOME/.ssh/authorized_keys` ファイルには、ログインを許可された公開鍵が一覧表示されています。ユーザーがログインする際、`ssh` プログラムはどの鍵ペアが認証に使われるのかをサーバーに知らせます。サーバーは、この鍵が許可されているかどうかを調べ、許可されている場合は、ユーザー (実際にはユーザーのために動作している `ssh` プログラム) にチャレンジを送ります。チャレンジとは、ユーザーの公開鍵で暗号化された乱数です。このチャレンジは、適切な秘密鍵を使ってのみ復号化できます。ユーザーのクライアントはこのとき、秘密鍵を使ってチャレンジを復号化して、自分が秘密鍵を知っていることを証明しますが、それをサーバーに開示することはしません。

`ssh` は、RSA 認証プロトコルを自動的に実装します。ユーザーは、`ssh-keygen(1)` を実行することで、自分の RSA 鍵ペアを作成します。これにより、秘密鍵がユーザーのホームディレクトリ内の `$HOME/.ssh/identity` に格納され、公開鍵が `$HOME/.ssh/identity.pub` に格納されます。次に、ユーザーは `identity.pub` をリモートマシン上の自分のホームディレクトリ内の `$HOME/.ssh/authorized_keys` にコピーするべきです (`authorized_keys` ファイルは従来の `$HOME/.rhosts` ファイルに相当し、1 行につき 1 つの鍵を保持しますが、各行は非常に長くなることがあります)。このあと、ユーザーはパスワードなしでログインできるようになります。RSA 認証は、`rhosts` 認証よりもはるかに安全です。

RSA 認証を使用するもっとも便利な方法は、認証エージェントを使うことです。詳細は、`ssh-agent(1)` を参照してください。

他の認証方式が失敗した場合、`ssh` はパスワードを入力するようユーザーに要求します。このパスワードは検査のためにリモートホストに送られます。しかし、すべての通信は暗号化されているため、ネットワークを盗聴している何者かによってパスワードが見られることはありません。

SSH プロトコルバージョン 2

SSH プロトコルバージョン 2 では、SSH プロトコルバージョン 1 で利用できる認証方式に似たものも含めて、複数のユーザー認証方式をサポートしています。これらの認証メカニズムは、クライアントとサーバーによってネゴシエーションが行われ、クライアントは `PreferredAuthentications` クライアント構成オプションで指定された順番で各方式を試します。サーバーは、どの時点でプロトコルの認証フェーズを完了するのに十分な数の認証方式が正常に通過したかを決めます。

ユーザーがプロトコルバージョン2を使用して接続するときも、同様の認証方式を利用できます。PreferredAuthentications のデフォルト値を使用すると、クライアントはまずホストベース方式を使って認証を試みます。この方式が失敗すると、公開鍵認証を試みます。この方式にも失敗すると、最後にキーボード対話型認証およびパスワード認証を試みます。

公開鍵方式は、前のセクションで説明した RSA 認証に似ており、RSA または DSA アルゴリズムを使用できます。クライアントは自分の秘密鍵である `$HOME/.ssh/id_dsa` または `$HOME/.ssh/id_rsa` を使ってセッション識別子に署名し、その結果をサーバーに送ります。サーバーはそれに合致する公開鍵が `$HOME/.ssh/authorized_keys` ファイル中に存在するかどうかを調べ、鍵が見つかり、かつその署名が正しければ、アクセスを許可します。セッション識別子は共有の Diffie-Hellman 値から導き出され、クライアントとサーバーだけがこの値を知っています。

公開鍵認証が失敗するか、それが使用できない場合、リモートホストにそのユーザー本人であることを証明するパスワードを暗号化して送ることができます。あるいは、拡張プロンプト/応答プロトコルを実行できます。

さらに、ssh ではホストベース認証やチャレンジ応答認証もサポートしています。

また、プロトコル2には、機密性(トラフィックは3DES、Blowfish、CAST128、または Arcfour を使って暗号化される)や整合性(hmac-sha2-256、hmac-sha2-256-96、hmac-sha2-512、hmac-sha2-512-96、hmac-sha1、および hmac-md5)を保持するためのメカニズムも備わっています。プロトコル1には、接続の整合性を保証するための強力なメカニズムがありません。

ログイン
セッションとリ
モート実行

そのユーザーが本人であることが確認できると、サーバーは指定されたコマンドを実行するか、ユーザーをそのマシンにログインさせてリモートマシン上で通常のシェル環境を提供します。リモートコマンドまたはシェルとの通信はすべて自動的に暗号化されます。

仮想端末が割り当てられている場合(通常のログインセッション時)、ユーザーは後述するエスケープ文字を使用できます。仮想端末が割り当てられている場合(通常のログインセッション時)、ユーザーは `~.` で接続を切り離したり、`~^Z` で ssh を中断したりできます。転送されたコネクションの一覧は `~#` で表示できます。セッションがブロックされて、転送された X11 コネクションまたは TCP/IP コネクションが終了待ちになっている場合、`~&` で ssh をバックグラウンドに移行させることができます(ユーザーシェルがアクティブになっている間はシェルがハングアップすることがあるため、これは使用しないでください)。使用できるエスケープ文字の一覧は、`~?` で表示できます。

チルダ文字を1つだけ送るには、`~~` を押すか、前述した文字以外の文字をチルダのあとに続けます。エスケープ文字は、必ず改行の直後に入力されなければ特殊な文字とは見なされません。エスケープ文字は、設定ファイルまたはコマンド行で変更できます。

仮想端末 (pseudo tty) が割り当てられていない場合、そのセッションは透過的であるため、バイナリデータを確実に転送できます。ほとんどのシステムでは、端末 (tty) が使用されている場合でも、エスケープ文字を「none」に設定することにより、そのセッションを透過的にすることができます。

セッションは、リモートマシン上のコマンドやシェルが終了し、すべての X11 コネクションおよび TCP/IP コネクションが閉じられると終了します。リモートプログラムの終了ステータスは、ssh の終了ステータスとして返されます。

エスケープ文字

仮想端末が要求されている場合、ssh ではエスケープ文字を使った機能がいくつかサポートされています。

チルダ文字を 1 つだけ入力するには、`~` を押すか、後述する文字以外の文字をチルダのあとに続けます。エスケープ文字は、必ず改行の直後に入力されなければ特殊な文字とは見なされません。エスケープ文字は、構成ファイルの `EscapeChar` 構成指示またはコマンド行の `-e` オプションで変更できます。

サポートされているエスケープ機能 (エスケープ文字はデフォルトの `~` とする) は次のとおりです。

- `~.` 接続を切り離します。
- `~^Z` ssh をバックグラウンドに移行させます。
- `~#` 転送されたコネクションの一覧を表示します。
- `~&` ssh をバックグラウンドに移行させ、転送されたコネクションまたは X11 のセッションが終了するのを待ってログアウトします。
- `~?` エスケープ文字の一覧を表示します。
- `~B` リモートシステムに `break` 信号を送ります。SSH プロトコルバージョン 2 で、相手側もそれをサポートしている場合にのみ有効です。
- `~C` コマンド行を開きます。`-L` オプションや `-R` オプションを使ってポート転送を追加する場合にのみ有効です。
- `~R` そのコネクションの鍵の変更を要求します。SSH プロトコルバージョン 2 で、相手側もそれをサポートしている場合にのみ有効です。

X11 と TCP の転送

`ForwardX11` 変数が「yes」に設定されており (後述の `-x` および `-X` オプションの説明を参照)、ユーザーが X11 を使用している (`DISPLAY` 環境変数が設定されている) 場合、X11 ディスプレイへのコネクションは自動的にリモート側に転送されます。つまり、シェル (またはコマンド) から起動された X11 プログラムはどれも暗号化されたチャネルを経由し、本物の X サーバーへのコネクションはローカルマシンから行われるようになります。ユーザーは `DISPLAY` を手動で設定しないでください。X11 コネクションの転送は、コマンド行または設定ファイルのどちらでも指定できます。

sshによって設定されたDISPLAY値はサーバーマシンを指していますが、ディスプレイ番号はゼロよりも大きくなります。これは正常な動作です。sshは暗号化されたチャネル経由でコネクションを転送します。そのため、サーバーマシン上にX11サーバーの「プロキシ」を生成するのでこうなるのです。

また、sshはサーバーマシン上でXauthorityデータを自動的に設定します。このためにランダムな承認cookieを生成して、サーバー側のXauthorityに格納し、転送されたコネクションがこのcookieを運び、コネクションが開かれるときにこれが本物のcookieと置き換わることを確認します。本物の認証cookieがサーバーマシンに送られることはありません(また、cookieが平文で送られることもありません)。

ForwardAgent変数が「yes」に設定されており(または、後述の-Aおよび-aオプションの説明を参照)、ユーザーが認証エージェントを使用している場合、エージェントへのコネクションは自動的にリモート側に転送されます。

セキュリティー保護されたチャネルを経由した任意のTCP/IPコネクションの転送は、コマンド行または設定ファイルのどちらでも指定できます。TCP/IP転送の応用の1つとして、電子財布へのセキュリティー保護された接続が考えられます。ほかにもファイアウォールをまたいだ接続などが考えられます。

サーバー認証

sshは、これまでに使用されたすべてのホストの識別情報が入っているデータベースを自動的に保守し、検査します。ホスト鍵はユーザーのホームディレクトリ内の\$HOME/.ssh/known_hostsに格納されています。さらに、既知のホストについては/etc/ssh_known_hostsファイルが自動的にチェックされます。不明なホスト鍵に関するsshの動作は、StrictHostKeyCheckingパラメータによって制御されます。あるホストの識別情報がこれまでと変わった場合、sshはこれに関する警告を発し、パスワード認証を無効にすることで、トロイの木馬がユーザーのパスワードを盗むのを防ぎます。このメカニズムのもう一つの目的は、中間者による攻撃で暗号化が巧みにかわされてしまうのを防ぐことです。StrictHostKeyCheckingオプションを使用すれば、ホスト鍵が不明だったり、変更されていたりするマシンへのログインを阻止することができます。

しかし、GSS-APIで保護された鍵交換を使用している場合、サーバーはホスト鍵を通知できます。クライアントはこのホスト鍵を既知ホストファイル\$HOME/.ssh/known_hostsに自動的に追加します。これは、通知されたホスト鍵が既存の既知ホストエントリと衝突しないかぎり、StrictHostKeyCheckingオプションの設定に関係なく行われます。

ユーザーのGSS-API資格の有効期限が切れると、クライアントはサーバーの公開ホスト鍵を使って引き続きセッションの鍵の変更を行うことができ、これにより鍵交換が保護されます。

GSS-APIユーザーおよびサーバー認証

sshでは、GssKeyExまたはGssAuthentication(あるいはその両方)が設定されている場合、可能な限り、ユーザーのGSS-API資格を使ってクライアントをサーバーに認証させます。

GssKeyEx が設定されている場合、GssKeyEx だけが使用されるように、ホストの公開鍵を持たない SSHv2 サーバーを 1 台持つことができます。そのようなサーバーでは、クライアントの資格の有効期限が切れた場合、鍵の変更は失敗します。

GSS-API ユーザー認証には SSH ホスト鍵の必要性を取り除かないというデメリットがありますが、その障害は鍵の変更に影響ありません。ssh では、GSS-API 認証に失敗した場合、ほかの認証方式 (公開鍵、パスワードなど) を試すことができます。

GSS-API 資格の委譲は極めて有効な場合がありますが、危険がないわけではありません。パスワードと同様に、ユーザーは GSS 資格を信頼できないサーバーに委譲しないでください。改ざんされたサーバーはユーザーの委譲された GSS 資格を使ってそのユーザーになりすます可能性があるからです。

GSS-API ユーザー承認については、[gss_auth_rules\(5\)](#) を参照してください。

GssKeyEx が「yes」のときは、鍵の変更を使って資格を再委譲できます。(前述の「エスケープ文字」の「~R」を参照してください。)

次を使用して ssh を構成します:

```
UseFIPS140 yes
```

...OpenSSL を FIPS-140 モードで実行します。SSH プロトコルバージョン 2 のみがサポートされます。SunSSH は、ユーザー/ホスト認証の暗号化操作を Solaris のほかの部分に引き続き委任する場合があります、その場合は FIPS 140 が認定されることもされないこともあります。UseOpenSSLEngine オプションのデフォルト値は no で、UseOpenSSLEngine を yes に設定しても FIPS モードには影響しません。ssh を FIPS-140 モードで実行するためのほかの要件として、クライアントは ssh-keygen -8 コマンドを使用してユーザーの非公開鍵を PKCS#8 形式で生成する必要があります。

FIPS-140 が有効な ssh の場合、FIPS-140 ではない sshd にログインするときに、このシナリオの「暗号化方式」を使用して、[sshd_config\(4\)](#) 内でサポートおよび承認されている FIPS 暗号化方式を明示的に指定する必要があります。

オプション

サポートしているオプションは、次のとおりです。

- 1 ssh がプロトコルバージョン 1 のみを試すように強制します。
- 2 ssh がプロトコルバージョン 2 のみを試すように強制します。
- 4 ssh が IPv4 アドレスのみを使用するように強制します。
- 6 ssh が IPv6 アドレスのみを使用するように強制します。

- a 認証エージェント接続の転送を無効にします。
- A 認証エージェント接続の転送を有効にします。これは設定ファイル内でホストごとに指定することもできます。
- エージェントの転送を有効にするときは注意が必要です。リモートホスト上で(エージェントの UNIX ドメインソケットに対する)ファイルアクセス権を無視できるユーザーは、転送されたコネクションを介してローカルエージェントにアクセスできてしまいます。攻撃者は、エージェントから鍵そのものを盗むことはできません。しかし、鍵を操作して、攻撃者がエージェントに読み込まれたアイデンティティを使って認証できるようにすることは可能です。
- b *bind_address* 複数のインタフェースまたはエイリアス処理されたアドレスを持つマシン上で転送元となるインタフェースを指定します。
- c *cipher_spec* セッションを暗号化するための暗号化仕様を選択します。
- プロトコルバージョン1では、*cipher_spec* は単一の暗号化です。詳細は、[ssh_config\(4\)](#) の「Cipher」オプションを参照してください。
- プロトコルバージョン2では、*cipher_spec* は優先度順に暗号化をコンマで区切ったリストです。詳細は、[ssh_config\(4\)](#) の「Ciphers」オプションを参照してください。
- C すべてのデータ(標準入力、標準出力、標準エラー出力、転送された X11 コネクションや TCP/IP コネクションのデータを含む)を圧縮するように要求します。圧縮アルゴリズムは、gzip(1) で使用されるものと同じです。gzip のマニュアルページは、SUNWsftman パッケージから入手できます。「レベル」は CompressionLevel オプション([ssh_config\(4\)](#) を参照)によって制御できます。圧縮は、モデム回線などの低速な接続には適していますが、高速のネットワークでは速度が低下するだけです。デフォルト値は、設定ファイル内でホスト

`-D [bind_address:]port`

ごとに指定できます。ssh_config(4)の「Compression」オプションを参照してください。

ローカル側でのアプリケーションレベルの「動的な」ポート転送を指定します。これは、(オプションで指定された *bind_address* にバインドされた) ローカル側のポートを待機するソケットを割り当てると機能します。このポートに接続されると、常に接続はセキュリティー保護されたチャネルを介して転送されます。アプリケーションプロトコルはこのとき、リモートマシンからどこに接続するかを決めるために使われます。現在、SOCKS4 および SOCKS5 プロトコルがサポートされており、SOCKS サーバーとして ssh が動作します。十分な特権を持っているユーザーのみが特権ポートにアクセスできます。動的ポート転送は、設定ファイル内で指定することもできます。

IPv6 アドレスは、代替の構文

`[bind_address/]port` を使用するか、アドレスを角括弧で囲むことによって指定できます。デフォルトでは、ローカルポートは「GatewayPorts」設定に従ってバインドされます。ただし、明示的に *bind_address* を使用すると、特定のアドレスに接続をバインドできます。localhost の *bind_address* は、待機ポートがローカルでのみ使用されるようにバインドされることを示します。一方、空のアドレスや * は、ポートがすべてのインタフェースからアクセス可能であることを示します。

`-e ch|^ch|none`

仮想端末 (pty) を使用するセッションのためのエスケープ文字を設定します (デフォルトは「~」)。エスケープ文字は、行の先頭にあるときのみ認識されます。エスケープ文字の後ろにドット (.) を付けると、接続が閉じられます。エスケープ文字のあとに CTRL-z が続く場合、その接続は中断されます。エスケープ文字そのものが続く場合、その文字が1度だけ送られます。エスケープ文字を none に設定すると、すべてのエスケープが無効になり、セッションは完全に透過的になります。

- `-f` ssh がコマンドを実行する直前に、バックグラウンドに移行するように要求します。これは、ssh からパスワードやパスフレーズの入力を求められるが、それをバックグラウンドで実行させたい場合に便利です。これは暗に `-n` オプションを指定したことになります。リモート側で X11 プログラムを起動させる場合は、ssh `-f host xterm` などとするのがよいでしょう。
- `-F configfile` ユーザーごとの設定ファイルに別のファイルを指定します。コマンド行に設定ファイルを指定すると、システム全体の設定ファイル `/etc/ssh_config` が無視されます。デフォルトでは、ユーザーごとの設定ファイルは `$HOME/.ssh/config` です。
- `-g` リモートホストが転送済みのローカルポートに接続することを許可します。
- `-i identity_file` RSA または DSA 認証用のアイデンティティー (秘密鍵) が読み取られるファイルを選択します。デフォルトは、プロトコルバージョン 1 の場合は `$HOME/.ssh/identity`、プロトコルバージョン 2 の場合は `$HOME/.ssh/id_rsa` と `$HOME/.ssh/id_dsa` です。アイデンティティーファイルは、設定ファイル内でホストごとに指定することもできます。複数の `-i` オプションを指定することも可能です (設定ファイルで複数のアイデンティティーを指定することもできます)。
- `-I PKCS#11-URI` 識別ファイルの代わりに、PKCS#11 トークンに格納された証明書および秘密鍵で動作します。詳細は、[sshd\(1M\)](#) のマニュアルページの「Using X.509 Certificates」セクションを参照してください。
- `-l login_name` リモートマシン上でログインする際のユーザー名を指定します。これは設定ファイル内でホストごとに指定することもできます。
- `-L [bind_address:]port:host:hostport` 指定されたローカル (クライアント) ホスト上のポートが、指定されたリモート側のホストおよびポートに転送されることを指定します。これは、(オプションで指定された `bind_address` にバインドされた) ローカル側のポートを待機する

ソケットを割り当てると機能します。次に、このポートへのコネクションが確立されると常に、そのコネクションがセキュリティー保護されたチャネル経由で転送され、リモートマシンから *host* のポート *hostport* へのコネクションが確立されます。ポート転送は、設定ファイル内で指定することもできます。十分な特権を持っているユーザーのみが特権ポートにアクセスできます。IPv6 アドレスは、代替の構文 `[bind_address/]port/host/hostport` を使用するか、アドレスを角括弧で囲むことによって指定できます。

デフォルトでは、ローカルポートは「GatewayPorts」設定に従ってバインドされます。ただし、明示的に *bind_address* を使用すると、特定のアドレスに接続をバインドできます。*localhost* の *bind_address* は、待機ポートがローカルでのみ使用されるようにバインドされることを示します。一方、空のアドレスや *** は、ポートがすべてのインタフェースからアクセス可能であることを示します。

`-m mac_spec`

プロトコルバージョン 2 では、複数の MAC (message authentication code、メッセージ認証コード) アルゴリズムを優先度の高い順にコマンドで区切って指定できます。詳細は、MAC というキーワードで調べてください。

`-n`

標準入力を `/dev/null` からリダイレクトします (実際には標準入力からの読み取りが行えないようにします)。ssh がバックグラウンドで動作する場合は、これを使用する必要があります。よくある手としては、リモートマシン上で X11 プログラムを実行するときこのオプションを使用することです。たとえば、次のように指定します。

```
ssh -n shadows.cs.hut.fi emacs &
```

これにより、emacs が shadows.cs.hut.fi 上で起動し、X11 コネクションが暗号化されたチャネル経由で自動的に転送されます。ssh プログラムはバックグラウンドに移行します。ssh がパスワードやパスフレーズの入力を求める必要が

	ある場合、このオプションは機能しません。-f オプションも参照してください。
-N	リモートコマンドを実行しません。このオプションは、ポート転送のみを行う場合に便利です(プロトコルバージョン2のみ)。
-o <i>option</i>	設定ファイルと同じ形式でオプションを指定するために使われます。これは、個別のコマンド行フラグがない場合にオプションを指定するのに便利です。このオプションの形式は、設定ファイル内の行と同じです。
-p <i>port</i>	リモートホストの接続先ポートを指定します。これは、設定ファイル内でホストごとに指定することができます。
-P	廃止されたオプションです。特権ポートからの SSHv1 コネクションはサポートされていません。
-q	静寂モード。すべての警告メッセージや診断メッセージは抑制されます。致命的なエラーだけが表示されます。
-R [<i>bind_address</i>][: <i>port</i> : <i>host</i> : <i>hostport</i>]	指定されたりリモート(サーバー)ホスト上のポートが、指定されたりリモート側のホストおよびポートに転送されることを指定します。このオプションを指定すると、まずリモート側でポートを待機するソケットが割り当てられます。次に、このポートへのコネクションが確立されると常に、そのコネクションがセキュリティー保護されたチャネル経由で転送され、ローカルマシンから <i>host</i> のポート <i>hostport</i> へのコネクションが確立されます。ポート転送は、設定ファイル内で指定することもできます。十分な特権を持っているユーザーとしてリモートマシンにログインした場合にのみ、特権ポートに転送できます。

IPv6 アドレスは、アドレスを角括弧で囲むか、代替の構文 [*bind_address*/]*host*/*port*/*hostport* を使用して指定できます。

デフォルトでは、サーバーの待機ソケットはループバックインタフェースにのみバインドされます。この設定は、*bind_address* を指定する

と上書きできます。空の *bind_address* またはアドレス*は、すべてのインタフェースでリモートソケットが待機することを示します。サーバーの「GatewayPorts」オプションが有効になっている場合にのみ、リモート *bind_address* の指定が成功します。ssh_config(4)を参照してください。

- s リモートシステムでサブシステムの呼び出しを要求するために使われます。サブシステムはSSH2 プロトコルの機能であり、SSH をほかのアプリケーション (sftp など) のためのセキュリティー保護された転送路として使用しやすくします。サブシステムはリモートコマンドとして指定されます。
 - t 強制的に仮想端末 (pseudo-tty) を割り当てます。これはリモートマシン上で任意の画面ベースのプログラムを実行するために使われ、メニューサービスを実装する場合などに非常に役立ちます。複数の -t オプションを指定すると、ssh がローカル tty を持っていない場合でも強制的に割り当てが行われます。
 - T 仮想端末 (pseudo-tty) の割り当てを無効にします (プロトコルバージョン 2 のみ)。
 - v 冗長モード。ssh がその進捗状況についてデバッグメッセージを表示するようにします。これは、接続、認証、および構成の問題をデバッグするときに役立ちます。-v オプションを複数個指定すると、出力の詳細レベルが高くなります。最大 3 個まで指定できます。
 - x X11 転送を無効にします。
 - X X11 転送を有効にします。これは設定ファイル内でホストごとに指定することもできます。
- X11 の転送を有効にするときは注意が必要です。リモートホスト上で (ユーザーの X 承認データベースに対する) ファイルアクセス権を無視できるユーザーは、転送されたコネクションを介してローカル側の X11 ディスプレイにアクセスできてしまいます。攻撃者はこのとき、キーストロークを盗み見るなどの行為を行える可能性があります。

このため、X11 転送は X11 SECURITY 拡張制限に従う場合があります。詳細は、[ssh_config\(4\)](#) の「ForwardX11Trusted」の指示を参照してください。

X11 転送が有効になっている場合、リモート X11 クライアントがデフォルトで信頼されます。つまり、元の X11 ディスプレイに完全にアクセスできます。

環境変数

通常、ssh は次の環境変数を設定します。

DISPLAY

DISPLAY 変数は、X11 ディスプレイの転送を機能させるために設定する必要があります。

SSH_ASKPASS

ssh がパスフレーズを必要とする場合、ssh が端末から起動されていれば、現在の端末からパスフレーズを読み取ります。ssh に端末が関連付けられていないが、DISPLAY と SSH_ASKPASS が設定されている場合は、SSH_ASKPASS で指定されたプログラムを実行し、X11 ウィンドウを開いてパスフレーズを読み取ります。これは、.Xsession やそれに関連するスクリプトから ssh を呼び出す際に特に役立ちます。マシンによっては、これを機能させるために、入力を /dev/null からリダイレクトする必要がある場合があります。システムは、SSH_ASKPASS のデフォルト値である /usr/lib/ssh/ssh-askpass が設定された状態で出荷されます。

SSH_AUTH_SOCK

エージェントとの通信に使用される UNIX ドメインソケットのパスを示しています。

SSH_LANGS

IETF 言語タグをコンマで区切ったリスト (RFC 3066 を参照)。ユーザーが読み書きできる言語を示しています。サーバー側でのロケールのネゴシエーションに使われます。

LANG、LC_ALL、LC_COLLATE、LC_CTYPE、

LC_MESSAGES、LC_MONETARY、LC_NUMERIC、LC_TIME

これらの環境変数の値は、クライアント側でのロケール設定やサーバー側でのこれらのロケールのサポート状況に従って、リモートセッションで設定することができます。環境変数をサーバー側に引き渡すときには、RFC 4254 の「Environment Variable Passing」を参照してください。

サーバー側の構成に応じてロケール設定をさらに変更する方法については、[sshd\(1M\)](#) のマニュアルページの「ENVIRONMENT VARIABLES」のセクションを参照してください。

関連項目

[rlogin\(1\)](#), [rsh\(1\)](#), [scp\(1\)](#), [ssh-add\(1\)](#), [ssh-agent\(1\)](#), [ssh-keygen\(1\)](#),
[ssh-http-proxy-connect\(1\)](#), [ssh-socks5-proxy-connect\(1\)](#), [telnet\(1\)](#), [sshd\(1M\)](#),
[ssh_config\(4\)](#), [sshd_config\(4\)](#), [attributes\(5\)](#), [gss_auth_rules\(5\)](#), [kerberos\(5\)](#),
[privileges\(5\)](#)

[krb5_auth_rules\(5\)](#) で、`.k5login` ファイルの説明を参照してください。

[RFC 1928](#)

[RFC 4254](#)

名前	ssh-add – 認証エージェントへの RSA または DSA アイデンティティの追加
形式	ssh-add [-lLdDxX] [-t <i>life</i>] [<i>file</i>] ...
機能説明	ssh-add ユーティリティは、認証エージェント ssh-agent(1) に RSA または DSA アイデンティティを追加します。引数なしで実行された場合は、存在するすべてのファイル <code>\$HOME/.ssh/identity</code> (RSA v1)、<code>\$HOME/.ssh/id_rsa</code> (RSA v2)、および <code>\$HOME/.ssh/id_dsa</code> (DSA v2) の追加を試みます。秘密鍵が複数存在する場合は、別のパスフレーズを入力し直すよう求める前に、同じパスフレーズによる各鍵の復号化を試みます。パスフレーズは、ユーザーの <code>tty</code> から、または <code>SSH_ASKPASS</code> (下を参照) で定義されたプログラムを実行することによって読み取られます。 認証エージェントが実行されている必要があります。
オプション	サポートしているオプションは、次のとおりです。 -d アイデンティティを追加するのではなく、このオプションは、エージェントからアイデンティティを削除します。 -D エージェントからすべてのアイデンティティを削除します。 -l 現在エージェントによって表されるすべてのアイデンティティのフィンガープリントを一覧表示します。 -L 現在エージェントによって表されるすべてのアイデンティティの公開鍵パラメータを一覧表示します。 -t <i>life</i> エージェントにアイデンティティを追加するときの最大の有効期間を設定します。この有効期間は秒単位で、または sshd(1M) で指定された時間形式で指定できます。 -x エージェントをパスワードでロックします。 -X エージェントのロックを解除します。
環境変数	DISPLAY SSH_ASKPASS ssh-add は、パスフレーズが必要な場合、自身が端末から実行されていれば現在の端末からパスフレーズを読み取ります。ssh-add には端末が関連付けられていないが、DISPLAY と SSH_ASKPASS が設定されている場合は、SSH_ASKPASS で指定されたプログラムを実行し、X11 ウィンドウを開いてパスフレーズを読み取ります。これは、<code>.Xsession</code> やそれに関連するスクリプトから ssh-add を呼び出す場合に特に役立ちます。システムには、SSH_ASKPASS のデフォルト値である <code>/usr/lib/ssh/ssh-askpass</code> が標準装備されています。 SSH_AUTH_SOCK エージェントと通信するために使用される UNIX ドメインソケットのパスを識別します。

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

1

エラーが発生した。

ファイル

これらのファイルは、このユーザー以外のだれからも読み取り可能であってはいけません。あるファイルがほかのユーザーからアクセス可能な場合、ssh-addはそのファイルを無視します。鍵を生成するときにパスフレーズを指定できます。そのパスフレーズは、このファイルのプライベートな部分を暗号化するために使用されます。

これらのファイルがネットワークファイルシステム上に格納されている場合は、ファイル自体で提供される保護か、またはネットワークファイルシステムのトランスポートレイヤーのどちらかによってサイトポリシーのための十分な保護が提供されていると見なされます。そうでない場合は、鍵ファイルをリムーバブルメディア上に格納するか、または関連するホスト上にローカルに格納することをお勧めします。

DSA および RSA 鍵ファイルに推奨される名前は次のとおりです。

\$HOME/.ssh/identity	プロトコルバージョン 1 の場合のユーザーの RSA 認証アイデンティティーが含まれます。
\$HOME/.ssh/identity.pub	プロトコルバージョン 1 の場合のユーザーの RSA 認証アイデンティティーのパブリックな部分が含まれます。
\$HOME/.ssh/id_dsa	ユーザーのプライベートな DSA 認証アイデンティティーが含まれます。
\$HOME/.ssh/id_dsa.pub	ユーザーの DSA 認証アイデンティティーのパブリックな部分が含まれます。
\$HOME/.ssh/id_rsa	ユーザーのプライベートな RSA 認証アイデンティティーが含まれます。
\$HOME/.ssh/id_rsa.pub	ユーザーの RSA 認証アイデンティティーのパブリックな部分が含まれます。
/usr/lib/ssh/ssh-askpass	SSH_ASKPASS のデフォルト値が含まれます。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

1874

マニュアルページセクション 1: ユーザーコマンド • 最終更新日 2009 年 5 月 20 日

関連項目 [ssh\(1\)](#), [ssh-agent\(1\)](#), [ssh-keygen\(1\)](#), [sshd\(1M\)](#), [attributes\(5\)](#)

名前 ssh-agent – 認証エージェント

形式 ssh-agent [-a *bind_address*] [-c | -s] [-d]
 [-t *life*] [*command* [*args*]...]

ssh-agent [-c | -s] -k

機能説明

ssh-agent は、公開鍵認証 (RSA、DSA) に使用される秘密鍵を保持するためのプログラムです。ssh-agent は多くの場合、ログインセッションの開始時に起動されます。ほかのウィンドウまたはプログラムはすべて、ssh-agent プログラムのクライアントとして起動されます。環境変数を使用すると、ssh(1) を使用してほかのマシンにログインするときに、エージェントを検索して認証に自動的に使用できます。『Oracle Solaris 11.1 の管理: セキュリティーサービス』を参照してください。

コマンド行が指定されている場合、これはエージェントのサブプロセスとして実行されます。このコマンドが終了すると、エージェントも終了します。

エージェントには最初、秘密鍵がありません。鍵は、エージェントにアイデンティティーを送信する ssh-add(1) を使用して追加されます。エージェントには複数のアイデンティティーを格納できます。エージェントは、これらのアイデンティティーのいずれかを自動的に使用できます。エージェントが現在保持しているアイデンティティーを表示するには、ssh-add(1) の -l オプションを使用します。

エージェントは、ユーザーのローカルホストで実行されます。認証データをほかのどのマシンにも格納する必要がなく、認証のパスフレーズはネットワーク経由で移動されません。ただし、エージェントへの接続が SSH リモートログイン経由で転送される場合、ユーザーは、ネットワーク内の任意の場所にあるアイデンティティーで指定された特権をセキュリティー保護された方法で使用できます。

エージェントを設定するための主な方法には2つあります。エージェントが、一部の環境変数がエクスポートされる新しいサブコマンドを起動できるようにするか、またはエージェントが、呼び出し元のシェルで評価できる必要なシェルコマンド (sh(1) または csh(1) のどちらかの構文を生成できます) を表示できるようにします。あとで、ssh(1) を使用してこれらの変数を調べ、それを使用してエージェントへの接続を確立します。

UNIX ドメインソケットが作成され (/tmp/ssh-XXXXXXX/agent.pid)、このソケットの名前が SSH_AUTH_SOCK 環境変数に格納されます。このソケットは、現在のユーザーからのみアクセス可能になります。この方法は root によって、または同じユーザーの別のインスタンスによって容易に悪用されます。

SSH_AGENT_PID 環境変数は、エージェントの PID を保持します。

コマンド行で指定されたコマンドが終了すると、エージェントは自動的に終了します。

オプション	サポートしているオプションは、次のとおりです。
-a <i>bind_address</i>	エージェントを UNIX ドメインソケットの <i>bind_address</i> にバインドします。デフォルトは <code>/tmp/ssh-XXXXXXXX/agent.pid</code> です。
-c	標準出力に C シェルコマンドを生成します。SHELL で <code>csh</code> スタイルのシェルであることが示されている場合は、これがデフォルトです。
-d	デバッグモード。このオプションが指定されている場合、 <code>ssh-agent</code> はフォークしません。
-k	現在のエージェント (<code>SSH_AGENT_PID</code> 環境変数で指定されます) を強制終了します。
-s	標準出力に Bourne シェルコマンドを生成します。SHELL で <code>csh</code> スタイルのシェルであることが示されていない場合は、これがデフォルトです。
-t <i>life</i>	エージェントに追加されるアイデンティティーの最大の有効期間のデフォルト値 (<i>life</i>) を設定します。 <i>life</i> は秒単位で、または sshd_config(4) で指定された時間形式で指定できます。この値は、 ssh-add(1) を使用してアイデンティティーに対して指定された <i>life</i> によってオーバーライドされます。このオプションを指定しない場合、デフォルトの最大値 <i>life</i> は無期限です。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。

ファイル `/tmp/ssh-XXXXXXXX/agent.pid` 認証エージェントへの接続を含めるために使用される UNIX ドメインソケット。これらのソケットは、所有者のみが読み取れるようにする必要があります。これらのソケットは、エージェントが終了すると削除されます。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目 [ssh\(1\)](#), [ssh-add\(1\)](#), [ssh-keygen\(1\)](#), [sshd\(1M\)](#), [sshd_config\(4\)](#), [attributes\(5\)](#)

『Oracle Solaris 11.1 の管理: セキュリティーサービス』

名前	ssh-http-proxy-connect – HTTP 用の Secure Shell プロキシ
形式	<code>/usr/lib/ssh/ssh-http-proxy-connect [-h <i>http_proxy_host</i>] [-p <i>http_proxy_port</i>] <i>connect_host connect_port</i></code>
機能説明	HTTP CONNECT を使用する、 ssh(1) 用のプロキシコマンド。通常は、ネットワークの外部の接続がプロキシ Web サーバー経由でのみ許可される場合に使用されます。
オプション	サポートしているオプションは、次のとおりです。 <code>-h <i>http_proxy_host</i></code> 接続に使用するプロキシ Web サーバーを指定します。HTTPPROXY および <code>http_proxy</code> 環境変数が設定されている場合は、それらの環境変数をオーバーライドします。 <code>-p <i>http_proxy_port</i></code> プロキシ Web サーバーが実行されるポートを指定します。指定されていない場合は、ポート 80 が使用されます。HTTPPROXYPORT および <code>http_proxy</code> 環境変数が設定されている場合は、それらの環境変数をオーバーライドします。
オペランド	次のオペランドがサポートされています。 <code>http_proxy_host</code> プロキシのホスト名または IP アドレス (IPv4 または IPv6)。 <code>http_proxy_port</code> <code>http_proxy_host</code> 上の接続先の数値のポート番号。 <code>connect_host</code> プロキシ Web サーバーで接続する先のリモートホストの名前。 <code>connect_port</code> <code>http_proxy_host</code> 上の、プロキシ Web サーバーで接続する先の数値のポート番号。
使用例	プロキシ接続コマンドを使用するには、ssh_config(4) で ProxyCommand を構成することをお勧めします (例 1 および例 2 を参照)。例 3 は、ssh(1) を実行しているときに、プロキシコマンドをコマンド行でどのように指定できるかを示しています。 例 1 環境からのプロキシの設定 次の例では、環境からプロキシが設定されるときに <code>ssh_config(4)</code> で <code>ssh-http-proxy-connect</code> を使用します。 <pre>Host playtime.foo.com ProxyCommand /usr/lib/ssh/ssh-http-proxy-connect \ playtime.foo.com 22</pre> 例 2 プロキシ環境変数のオーバーライド 次の例では、<code>ssh_config(4)</code> で <code>ssh-http-proxy-connect</code> を使用して、プロキシ環境変数をオーバーライド (設定されていない場合は設定) します。 <pre>Host playtime.foo.com ProxyCommand /usr/lib/ssh/ssh-http-proxy-connect -h webcache \</pre>

例2 プロキシ環境変数のオーバーライド (続き)

```
-p 8080 playtime.foo.com 22
```

例3 コマンド行の使用

次の例では、ssh(1) コマンド行から ssh-http-proxy-connect を使用します。

```
example$ ssh -o ProxyCommand="/usr/lib/ssh/ssh-http-proxy-connect \  
-h webcache -p 8080 playtime.foo.com 22" playtime.foo.com
```

- 環境変数
- HTTPPROXY

デフォルトのプロキシホストを指定するために、*http_proxy_host* オペランドを取得します。両方が設定されている場合は、*http_proxy* をオーバーライドします。
- HTTPPROXYPORT

デフォルトのプロキシポートを指定するために、*http_proxy_port* オペランドを取得します。HTTPPROXY が設定されていない場合は、無視されます。
- http_proxy

プロキシホストおよびポートを指定するための URL 形式。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目 [ssh\(1\)](#), [ssh-socks5-proxy-connect\(1\)](#), [ssh_config\(4\)](#), [attributes\(5\)](#)

名前 ssh-keygen – 認証鍵の生成

形式

```
ssh-keygen [-q] [-b bits] -t type [-N new_passphrase]
           [-C comment] [-f output_keyfile]

ssh-keygen -p [-P old_passphrase] [-N new_passphrase]
           [-f keyfile]

ssh-keygen -i [-f input_keyfile | PKCS#11-URI]

ssh-keygen -e [-f input_keyfile]

ssh-keygen -y [-f input_keyfile]

ssh-keygen -c [-P passphrase] [-C comment] [-f keyfile]

ssh-keygen -l [-f input_keyfile | PKCS#11-URI]

ssh-keygen -B [-f input_keyfile]

ssh-keygen -F hostname [-f known_hosts_file]

ssh-keygen -H [-f known_hosts_file]

ssh-keygen -R hostname [-f known_hosts_file]
```

機能説明

ssh-keygen ユーティリティーは、[ssh\(1\)](#) の認証鍵を生成、管理、および変換します。ssh-keygen は、SSH プロトコルバージョン 1 で使用される RSA 鍵、および SSH プロトコルバージョン 2 で使用される RSA または DSA 鍵を作成できます。生成される鍵のタイプは、-t オプションで指定されます。ssh-keygen はまた、フィンガープリントを生成したり、PKCS#11 URI として指定された X.509v3 証明書の公開鍵を変換したりすることもできます。

通常、RSA または DSA 認証に SSH を使用しようとしている各ユーザーは、これを 1 回実行して \$HOME/.ssh/identity、\$HOME/.ssh/id_dsa、または \$HOME/.ssh/id_rsa 内に認証鍵を作成します。また、システム管理者は、これを使用してホスト鍵も生成できます。

通常、このプログラムは鍵を生成して、秘密鍵を格納するためのファイルの入力を求めます。公開鍵は、同じ名前のあとに「.pub」拡張子が付加されたファイル内に格納されます。このプログラムはまた、パスフレーズの入力も求めます。パスフレーズは、パスフレーズがないことを示すために空にすることも(ホスト鍵のパスフレーズも空である必要があります)、任意の長さの文字列にすることもできます。適切なパスフレーズは、長さが 10 から 30 文字、単純な文章やその他の方法で推測しやすい文字列ではなく、さらに大文字と小文字の英字、数字、および英数字以外の文字が混在して含まれています。(英語の散文には単語あたり 1 から 2 ビットのエントロピシカ含まれておらず、非常に劣ったパスフレーズが提供されます。)パスフレーズが設定されている場合は、少なくとも 4 文字の長さが必要です。

パスフレーズは、-p オプションを使用してあとで変更できます。

失われたパスフレーズを回復する方法はありません。パスフレーズを失ったり忘れたりした場合は、新しい鍵を生成し、対応する公開鍵をほかのマシンにコピーする必要があります。

RSA の場合は、鍵ファイル内に、鍵の識別に役立つユーザーの便宜のためだけのコメントフィールドも存在します。*comment* は、鍵の目的や、その他の役立つあらゆる内容を伝えることができます。このコメントは、鍵が作成されると「*user@host*」に初期化されますが、*-c* オプションを使用して変更できます。

鍵が生成されたあと、その鍵をどこに配置して有効にするかについて次に詳細に示します。

オプション

サポートしているオプションは、次のとおりです。

-b bits

作成する鍵内のビット数を指定します。最小数は512ビットです。一般には、2048ビットで十分と考えられます。これを超えるキーサイズではもうセキュリティは向上せず、処理が遅くなります。デフォルトは2048ビットです。

-B

指定された非公開または公開鍵ファイルの bubblebabble ダイジェストを示します。

-c

非公開および公開鍵ファイル内のコメントの変更を要求します。プログラムは、秘密鍵を含むファイル、パスフレーズ(鍵にパスフレーズがある場合)、および新しいコメントの入力を求めます。

このオプションは、*rsa1* (SSHv1) 鍵にのみ適用されます。

-C comment

新しいコメントを指定します。

-e

このオプションは非公開または公開の OpenSSH 鍵ファイルを読み取り、その鍵を「SECSH」公開鍵ファイル形式で標準出力に表示します。このオプションを使用すると、ほかのいくつかの SSH 実装で使用するために鍵をエクスポートできます。

-f

鍵ファイルのファイル名を指定します。

-F

known_hosts ファイル内の指定された *hostname* を検索し、見つかったものをすべて一覧表示します。このオプションは、ハッシュされたホスト名またはアドレスの検索に有効なほか、見つかった鍵をハッシュされた形式で表示するために *-H* オプションと組み合わせて使用することもできます。

-H

`known_hosts` ファイルをハッシュします。これは、すべてのホスト名とアドレスを、指定されたファイル内のハッシュされた表現に置き換えます。元の内容は、`.old` の接尾辞を持つファイルに移動されます。これらのハッシュは `ssh` や `sshd` で正常に使用できますが、ファイルの内容が公表されても、識別情報は明らかにされません。このオプションは既存のハッシュされたホスト名を変更しないため、ハッシュされた名前とハッシュされていない名前が混在したファイルでも安全に使用できます。

-i

このオプションは、SSH2 と互換性のある形式の暗号化されていない非公開 (または公開) 鍵ファイルを読み取り、OpenSSH と互換性のある非公開 (または公開) 鍵を標準出力に表示します。`ssh-keygen` はまた、「SECSH」公開鍵ファイル形式も読み取ります。このオプションを使用すると、ほかのいくつかの SSH 実装から鍵をインポートできます。

-l

指定された非公開または公開鍵ファイルのフィンガープリントを示します。

-N *new_passphrase*

新しいパズフレーズを指定します。

-p

新しい秘密鍵の作成ではなく、秘密鍵ファイルのパズフレーズの変更を要求します。プログラムは、秘密鍵を含むファイルと古いパズフレーズの入力を求めたあと、新しいパズフレーズを 2 回入力するよう求めます。

-P *passphrase*

(古い) パズフレーズを指定します。

-q

`ssh-keygen` を休止します。

-t *type*

鍵に使用されるアルゴリズムを指定します。ここで、*type* は `rsa`、`dsa`、`rsa1` のいずれかです。タイプ `rsa1` は、SSHv1 プロトコルにのみ使用されます。

-R *hostname*

`known_hosts` ファイルから *hostname* に属するすべての鍵を削除します。このオプションは、ハッシュされたホストを削除する場合に役立ちます。-H を参照してください。

-x

廃止。-e オプションによって置き換えられました。

-X

廃止。-i オプションによって置き換えられました。

-y
このオプションは非公開の OpenSSH 形式ファイルを読み取り、OpenSSH 公開鍵を標準出力に表示します。

-8
ssh-keygen が鍵を PKCS#8 形式で生成することを指定します。生成されるサポート対象の鍵のタイプは rsa または dsa です。

終了ステータス 次の終了ステータスが返されます。

0
正常終了。

1
エラーが発生した。

ファイル

`$HOME/.ssh/identity`

このファイルには、SSHv1 プロトコルの RSA 秘密鍵が含まれています。このファイルは、このユーザー以外のだれからも読み取り可能であってはけません。鍵を生成するときにパスフレーズを指定できます。そのパスフレーズは、128 ビット AES を使用してこのファイルのプライベートな部分を暗号化するために使用されます。このファイルは ssh-keygen から自動的にアクセスされませんが、秘密鍵のためのデフォルトファイルとして提供されます。このファイルは、ログインが試行されたときに `sshd(1M)` によって読み取られます。

`$HOME/.ssh/identity.pub`

このファイルには、SSHv1 プロトコルの RSA 公開鍵が含まれています。このファイルの内容を、RSA 認証を使用してログインするすべてのマシン上の `$HOME/.ssh/authorized_keys` に追加するようにしてください。このファイルのシークレットの内容を保管する必要はありません。

`$HOME/.ssh/id_dsa`

`$HOME/.ssh/id_rsa`

これらのファイルには、それぞれ、SSHv2 プロトコルの DSA 秘密鍵または RSA 秘密鍵が含まれています。これらのファイルは、このユーザー以外のだれからも読み取り可能であってはけません。鍵を生成するときにパスフレーズを指定できます。そのパスフレーズは、3DES を使用してこのファイルのプライベートな部分を暗号化するために使用されます。これらのいずれのファイルも ssh-keygen から自動的にアクセスされませんが、秘密鍵のためのデフォルトファイルとして提供されます。このファイルは、ログインが試行されたときに `sshd(1M)` によって読み取られます。

`$HOME/.ssh/id_dsa.pub`

`$HOME/.ssh/id_rsa.pub`

これらのファイルには、それぞれ、SSHv2 プロトコルの DSA 公開鍵または RSA 公開鍵が含まれています。これらのファイルの内容を、それぞれ、DSA または RSA 認証を使用してログインするすべてのマシン上の `$HOME/.ssh/authorized_keys` に追加するようにしてください。これらのファイル

のシークレットの内容を保管する必要はありません。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh/ssh-key
インタフェースの安定性	确实

関連項目 [ssh\(1\)](#), [ssh-add\(1\)](#), [ssh-agent\(1\)](#), [sshd\(1M\)](#), [attributes\(5\)](#)

名前	ssh-keyscan – 複数のホストの公開 SSH ホスト鍵の収集
形式	ssh-keyscan [-v46] [-p <i>port</i>] [-T <i>timeout</i>] [-t <i>type</i>] [-f <i>file</i>] [-] [<i>host...</i> <i>addrlist namelist</i>] [...]
機能説明	ssh-keyscan は、複数のホストの公開 SSH ホスト鍵を収集するためのユーティリティです。このユーティリティは、ssh_known_hosts ファイルの構築および検証を支援するように設計されました。ssh-keyscan は、シェルや Perl スクリプトでの使用に適した最小のインタフェースを提供します。ssh-keyscan の出力は、標準出力にリダイレクトされます。 ssh-keyscan は、非ブロックソケット入出力を使用してできるだけ多くのホストに並列に接続するため、非常に効率的です。一部のホストが停止しているか、または SSH を実行していない場合でも、1,000 のホストから成るドメインからの鍵を数十秒で収集できます。走査のために、走査されているマシンへのログインアクセスも、走査プロセスでの暗号化も必要ありません。
ファイル形式	入力形式: 1.2.3.4,1.2.4.4 <i>name.my.domain,n.my.domain,n,1.2.3.4,1.2.4.4</i> rsa1 鍵の出力形式: <i>host-or-namelist bits exponent modulus</i> rsa および dsa 鍵の出力形式。ここで、<i>keytype</i> は ssh-rsa または ssh-dsa のどちらかです。 <i>host-or-namelist keytype base64-encoded-key</i>
オプション	サポートしているオプションは、次のとおりです。 -f <i>filename</i> このファイルからホストまたはアドレスリストと名前リストのペアを読み取ります (1 行に 1 つ)。ファイル名の代わりに - を指定した場合、ssh-keyscan は、ホストまたはアドレスリストと名前リストのペアを標準入力から読み取ります。 -p <i>port</i> リモートホスト上の接続先のポート。 -T <i>timeout</i> 接続試行のタイムアウトを設定します。ホストへの接続が開始されてから、またはそのホストから任意のデータが最後に読み取られてから <i>timeout</i> 秒が経過すると、その接続は閉じられ、対象のホストは使用不可であると見なされます。<i>timeout</i> のデフォルトは 5 秒です。 -t <i>type</i> 走査されるホストから取得する鍵のタイプを指定します。<i>type</i> の指定可能な値は、プロトコルバージョン 1 の場合は rsa1、プロトコルバージョン 2 の場合は rsa または dsa です。複数の値はコンマで区切って指定します。デフォルトは rsa1 です。

- v
詳細表示モードを指定します。進捗に関するデバッグメッセージを表示します。
- 4
強制的に IPv4 アドレスのみを使用します。
- 6
強制的に IPv6 アドレスのみを使用します。

セキュリティー ssh_known_hosts ファイルが、鍵を検証せずに ssh-keyscan を使用して構築されている場合、ユーザーは中間者攻撃に対して脆弱です。セキュリティーモデルでこのようなリスクが許可されている場合、ssh-keyscan は、改ざんされた鍵ファイルや、ssh_known_hosts ファイルが作成されたあとに開始された「man-in-the-middle attack」を検出するのに役立ちます。

使用例 例1 rsa1 ホスト鍵の表示
次の例では、マシン hostname の rsa1 ホスト鍵を表示します。
\$ ssh-keyscan hostname

例2 すべてのホストの検索
次のコマンドは、ソートされたファイル ssh_known_hosts 内の鍵とは異なる鍵や新しい鍵を含むファイル ssh_hosts からすべてのホストを検索します。
\$ ssh-keyscan -t rsa,dsa -f ssh_hosts | \
sort -u - ssh_known_hosts | diff ssh_known_hosts -

ファイル /etc/ssh_known_hosts
公開 SSH ホスト鍵のリストが含まれます。

終了ステータス 次の終了ステータスが返されます。
0
使用方法のエラーはありません。ssh-keyscan が、指定されたホストのうちの 1 つ、それ以上、またはすべての走査に成功したか失敗したかは明確ではありません。
1
使用方法のエラー。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目	ssh(1) , sshd(1M) , attributes(5)
著者	David Mazieres が最初のバージョンを記述し、Wayne Davison がプロトコルバージョン 2 のサポートを追加しました。
使用上の留意点	サーバーがバージョン 2.9 よりも古い場合、ssh-keyscan はスキャンするすべてのマシンのコンソールに次のメッセージを生成します: <pre>Connection closed by remote host</pre> これは、ssh-keyscan が ssh ポートへの接続を開いて公開鍵を読み取り、その鍵を取得するとすぐに接続を解除するためです。

名前	ssh-socks5-proxy-connect – SOCKS5 用の Secure Shell プロキシ
形式	<code>/usr/lib/ssh/ssh-socks5-proxy-connect</code> <code>[-h socks5_proxy_host]</code> <code>[-p socks5_proxy_port] connect_host connect_port</code>
機能説明	SOCKS5 (RFC 1928) を使用する、ssh(1) 用のプロキシコマンド。通常は、ネットワークの外部の接続が SOCKS ゲートウェイサーバー経由でのみ許可される場合に使用されます。 このプロキシコマンドは、RFC 1928 で定義されたどの SOCKS5 認証メカニズムも提供しません。匿名接続だけが可能です。
オプション	サポートしているオプションは、次のとおりです。 <code>-h socks5_proxy_host</code> 接続に使用するプロキシ Web サーバーを指定します。SOCKS5_SERVER 環境変数をオーバーライドします。 <code>-p socks5_proxy_port</code> プロキシ Web サーバーが実行されるポートを指定します。指定されていない場合は、ポート 80 が使用されます。SOCKS5_PORT 環境変数をオーバーライドします。
オペランド	次のオペランドがサポートされています。 <code>socks5_proxy_host</code> プロキシのホスト名または IP アドレス (IPv4 または IPv6)。 <code>socks5_proxy_port</code> <code>socks5_proxy_host</code> 上の接続先の数値のポート番号。 <code>connect_host</code> SOCKS ゲートウェイで接続する先のリモートホストの名前。 <code>connect_port</code> <code>connect_host</code> 上の、SOCKS ゲートウェイで接続する先の数値のポート番号。
使用例	プロキシ接続コマンドを使用するには、ssh_config(4) で ProxyCommand を構成することをお勧めします (例 1 および例 2 を参照)。例 3 は、ssh(1) を実行しているときに、プロキシコマンドをコマンド行でどのように指定できるかを示しています。 例1 環境からのプロキシの設定 次の例では、環境からプロキシが設定されるときに <code>ssh_config(4)</code> で <code>ssh-socks5-proxy-connect</code> を使用します。 <pre>Host playtime.foo.com ProxyCommand /usr/lib/ssh/ssh-socks5-proxy-connect \ playtime.foo.com 22</pre> 例2 プロキシ環境変数のオーバーライド 次の例では、<code>ssh_config(4)</code> で <code>ssh-socks5-proxy-connect</code> を使用して、プロキシ環境変数をオーバーライド (設定されていない場合は設定) します。

例2 プロキシ環境変数のオーバーライド (続き)

```
Host playtime.foo.com
    ProxyCommand /usr/lib/ssh/ssh-socks5-proxy-connect -h socks-gw \
        -p 1080 playtime.foo.com 22
```

例3 コマンド行の使用

次の例では、[ssh\(1\)](#) コマンド行から ssh-socks5-proxy-connect を使用します。

```
example$ ssh -o'ProxyCommand=/usr/lib/ssh/ssh-socks5-proxy-connect \
    -h socks-gw -p 1080 playtime.foo.com 22' playtime.foo.com
```

環境変数

- SOCKS5_SERVER デフォルトのプロキシホストを指定するために、`socks5_proxy_host` オペランドを取得します。
- SOCKS5_PORT デフォルトのプロキシポートを指定するために、`socks5_proxy_port` オペランドを取得します。

終了ステータス

次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/ssh
インタフェースの安定性	確実

関連項目

[ssh\(1\)](#), [ssh-http-proxy-connect\(1\)](#), [ssh_config\(4\)](#), [attributes\(5\)](#)

名前	strchg, strconf - ストリーム構成の変更または照会								
形式	<pre>strchg -h <i>module1</i> [, <i>module2...</i>] strchg -p [-a -u <i>module</i>] strchg -f <i>filename</i> strconf [-m -t <i>module</i>]</pre>								
機能説明	これらのコマンドは、ユーザーの標準入力に対応するストリームの構成を変更または照会するために使用されます。strchg コマンドは、ストリーム上へモジュールをプッシュ、もしくはストリームからモジュールをポップします。strconf コマンドは、ストリームの構成を問い合わせます。ストリームの構成を変更できるのは、スーパーユーザーおよび対応する STREAMS デバイスの所有者だけです。 strconf を引数なしで実行した場合、ストリーム上の全モジュール名と最上位のドライバ名のリストが返されます。出力リストは1行に1つの名前の形式で、もしストリーム上に最上位モジュールが存在していれば先頭にその名前が置かれています。最後の項目はドライバ名となります。								
オプション	次に示すオプションは、いずれも strchg コマンド用のものです。なお -h、-f、-p は相互に排他的なオプションです。 <table><tr><td>-a</td><td>最上位ドライバよりも上位に位置するすべてのモジュールをストリーム上からポップする対象とします。このオプションは、必ず -p とともに指定してください。</td></tr><tr><td>-f <i>filename</i></td><td>ストリームの構成の変更内容を示すモジュールのリストを含んでいるファイルを <i>filename</i> で指定します。このファイルは、1行に1モジュール名という形式でなければなりません。最上位のモジュール名を1行目に、ドライバに最も近いモジュール名を最終行に記述してください。strchg は、ストリームの構成が最終的にこのファイルで指定した内容になるよう、モジュールのプッシュとポップを実行します。</td></tr><tr><td>-h <i>module1</i> [, <i>module2...</i>]</td><td>push の二モニック。ストリームへモジュールをプッシュします。1つ以上のプッシュ可能なストリームモジュールの名前を引数として指定します。これらのモジュールは指定された順序でプッシュされます。つまり最初に <i>module1</i>、次に <i>module2</i> というようにプッシュされます。</td></tr><tr><td>-p</td><td>pop の二モニック。ストリームからモジュールをポップします。-p だけを記述して strchg を実行すると、最上位のモジュールがポップされます。</td></tr></table>	-a	最上位ドライバよりも上位に位置するすべてのモジュールをストリーム上からポップする対象とします。このオプションは、必ず -p とともに指定してください。	-f <i>filename</i>	ストリームの構成の変更内容を示すモジュールのリストを含んでいるファイルを <i>filename</i> で指定します。このファイルは、1行に1モジュール名という形式でなければなりません。最上位のモジュール名を1行目に、ドライバに最も近いモジュール名を最終行に記述してください。strchg は、ストリームの構成が最終的にこのファイルで指定した内容になるよう、モジュールのプッシュとポップを実行します。	-h <i>module1</i> [, <i>module2...</i>]	push の二モニック。ストリームへモジュールをプッシュします。1つ以上のプッシュ可能なストリームモジュールの名前を引数として指定します。これらのモジュールは指定された順序でプッシュされます。つまり最初に <i>module1</i> 、次に <i>module2</i> というようにプッシュされます。	-p	pop の二モニック。ストリームからモジュールをポップします。-p だけを記述して strchg を実行すると、最上位のモジュールがポップされます。
-a	最上位ドライバよりも上位に位置するすべてのモジュールをストリーム上からポップする対象とします。このオプションは、必ず -p とともに指定してください。								
-f <i>filename</i>	ストリームの構成の変更内容を示すモジュールのリストを含んでいるファイルを <i>filename</i> で指定します。このファイルは、1行に1モジュール名という形式でなければなりません。最上位のモジュール名を1行目に、ドライバに最も近いモジュール名を最終行に記述してください。strchg は、ストリームの構成が最終的にこのファイルで指定した内容になるよう、モジュールのプッシュとポップを実行します。								
-h <i>module1</i> [, <i>module2...</i>]	push の二モニック。ストリームへモジュールをプッシュします。1つ以上のプッシュ可能なストリームモジュールの名前を引数として指定します。これらのモジュールは指定された順序でプッシュされます。つまり最初に <i>module1</i> 、次に <i>module2</i> というようにプッシュされます。								
-p	pop の二モニック。ストリームからモジュールをポップします。-p だけを記述して strchg を実行すると、最上位のモジュールがポップされます。								

-u module *module* で示したモジュールよりも上位にあるすべてのモジュールをストリーム上からポップする対象とします。*module* 自体は対象とはなりません。このオプションは、必ず **-p** とともに指定してください。

以下に示すオプションは、いずれも **strconf** コマンド用のもので、**-m** と **-t** は相互に排他的です。

-m module *module* で示したモジュールがストリーム上に存在しているかを調べます。存在していれば、**strconf** は **yes** というメッセージを出力してゼロを返します。存在しなければ、**no** というメッセージを出力してゼロ以外の値を返します。この **-m** オプションは、次の **-t** とは排他的です。

-t module 最上位モジュールが存在していればその名前を出力します。この **-m** オプションは、次の **-t** とは排他的です。

使用例

例1 **strchg** コマンドの例

次の例は、ユーザーの標準入力に対応するストリーム上に **ldterm** という名のモジュールをプッシュするものです。

```
example% strchg -h ldterm
```

次の例は、**/dev/term/24** に対応するストリームから最上位モジュールをポップするものです。このコマンドを発行するユーザーは、当該デバイスの所有者もしくはスーパーユーザーでなければなりません。

```
example% strchg -p < /dev/term/24
```

次は、以下のような内容を含む **fileconf** というファイルがあるとした場合の例で、

```
ttcompat
ldterm
ptem
```

次のコマンドを実行すると、

```
example% strchg -f fileconf
```

ユーザーの標準入力ストリームを構成し、結果として **ptem** がドライバの上にプッシュされ、その次に **ldterm** が置かれ、**ttcompat** がストリームの先頭にもっとも近い位置に置かれます。

引数なしの **strconf** コマンドは、ストリーム上の全モジュールと最上位のドライバのリストを出力します。たとえばストリームの現在の内容が、ドライバ **zs** の上にモジュール **ldterm** がプッシュされているだけの状態の場合、引数なしで **strconf** コマンドを実行すると次のような出力が得られます。

例1 strchg コマンドの例 (続き)

```
ldterm
zs
```

この内容のストリームに対して、次のコマンドは、ldterm がストリーム上に存在しているかを問い合わせます。

```
example% strconf -m ldterm
```

このコマンドは以下のメッセージを出力し、終了コード 0 を返して実行が終了します。

```
yes
```

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [attributes\(5\)](#), [streamio\(7I\)](#)

診断 strchg コマンドは、実行に成功すればゼロを返します。エラーが発生すると、エラーメッセージを表示してゼロ以外の値を返します。起こりうるエラーとしては、使用方法の誤り、不当なモジュール名、プッシュ対象のモジュールが多すぎる、ストリーム上の ioctl の失敗、-f オプションで指定した *filename* がオープンできない、などが考えられます。

strconf コマンドも、実行に成功すればゼロを返します（ここで言う成功とは、-m オプション使用時には指定したモジュールが存在していることを、また -t オプション使用時には最上位モジュールが存在していることを意味します）。-m または -t オプションが指定され、該当するモジュールが存在しない場合、ゼロ以外の値を返します。また、使用方法の誤りやストリーム上の ioctl の失敗などのエラーが発生すると、エラーメッセージを表示してゼロ以外の値を返します。

注意事項 スーパーユーザーでもなく、ストリームの所有者でもないユーザーは、strchg コマンドを実行することはできません。また、スーパーユーザーでもなく、ストリームに対する読み取り権を持っていないユーザーは、strconf コマンドを実行することはできません。

誤った順序でモジュールをプッシュしてしまうと、ストリームが期待どおりに機能しなくなる恐れがあります。また、tty のストリームで回線規約モジュールを誤った位置にプッシュしてしまうと、端末がどのコマンドに対しても応答しなくなる恐れがあります。

名前	strings - オブジェクトファイルおよびバイナリファイルからの印字可能な文字列の検索
形式	strings [-a -] [-t <i>format</i> -o] [-n <i>number</i> -number] [-N <i>name</i>] [<i>file</i>]...
機能説明	strings ユーティリティーは、バイナリファイル中の ASCII 文字列を探します。文字列は、4 バイト以上の印字可能文字で、復帰改行 (NEWLINE) または NULL 文字で終わらなければなりません。 strings はランダムオブジェクトファイルなどを識別するのに便利です。 デフォルトでは、文字列はメモリーにロードされるプログラムセクションを検索します。プログラムセクションは、セクションタイプ SHT_PROGBITS で識別されます。メモリーにロードされるセクションは、セクションフラグ SHF_ALLOC で識別されます。ファイルに関する完全なセクション情報を表示するには、elfdump(1) を使用します。 すべてのセクションは -a オプションで検査できます。個々のセクションは -N オプションで検査できます。
オプション	サポートしているオプションは、次のとおりです。 -a - ファイル全体を検索の対象とします。 -n <i>number</i> -number <i>number</i> を最短文字列のバイト数とします。デフォルトは 4 バイトです。無効な数字を指定すると、デフォルトの文字列の長さが使用されます。 -N <i>name</i> ELF セクション名のみを調べます。elfdump(1) を参照してください。複数の -N オプションを指定すると、複数のセクションを検査できます。 -a または - オプションを指定した場合は、すべての -N オプションが無視されます。 -o -t d オプションと同等の機能です。 -t <i>format</i> ファイル内での先頭からのバイトオフセットの値を文字列の前に付加して各文字列を出力します。出力形式は、オプション引数の <i>format</i> により 1 文字で指定します。 d オフセットを 10 進数で出力 o オフセットを 8 進数で出力 x オフセットを 16 進数で出力

- オペランド

次のオペランドを指定できます。

file 入力に用いる通常ファイルのパス名。*file* オペランドを省略すると、strings ユーティリティーは標準入力から読み込みます。
- 環境変数

strings の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
- 終了ステータス

次の終了ステータスが返されます。

0 正常終了。
>0 エラーが発生した。
- 属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	下記を参照。

- strings ユーティリティー (-N を除くすべてのオプションを含む) は標準で指定されています。[standards\(5\)](#) を参照してください。現在、-N オプションは標準で指定されていません。
- 関連項目

[elfdump\(1\)](#), [od\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)
- 注意事項

文字列を認識するアルゴリズムは、きわめて原始的です。

旧バージョンとの互換性において、-a の代わりに - を指定することができます。

名前	<code>strip</code> – strip symbol table, debugging and line number information from an object file
形式	<code>strip [-lVx] file...</code>
機能説明	The <code>strip</code> command removes the symbol table <code>SHT_SYMTAB</code> and its associated string table, debugging information, and line number information from ELF object files. That is, besides the symbol table and associated string table, the following sections are removed: <pre>.line .debug* .stab*</pre> Once this stripping process has been done, limited symbolic debugging access is available for that file. Therefore, this command is normally run only on production modules that have been debugged and tested. If <code>strip</code> is executed on a common archive file (see ar.h(3HEAD)) in addition to processing the members, <code>strip</code> removes the archive symbol table. The archive symbol table must be restored by executing the ar(1) command with the <code>-s</code> option before the archive can be linked by the ld(1) command. <code>strip</code> produces appropriate warning messages when this situation arises. <code>strip</code> is used to reduce the file storage overhead taken by the object file.
オプション	The amount of information stripped from the ELF object file can be controlled by using any of the following options. The following options are supported: -l Strip line number information only. Does not strip the symbol table or debugging information. -V Prints, on standard error, the version number of <code>strip</code>. -x Does not strip the symbol table. Debugging and line number information might be stripped.
オペランド	The following operand is supported: <i>file</i> A path name referring to an executable file.
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of <code>strip</code> : <code>LANG</code> , <code>LC_ALL</code> , <code>LC_CTYPE</code> , <code>LC_MESSAGES</code> , and <code>NLSPATH</code> .
終了ステータス	The following exit values are returned: 0 Successful completion. >0 An error occurred.

属性 See `attributes(5)` for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities
Interface Stability	Committed
Standard	See standards(5) .

注意事項	The symbol table section is not removed if it is contained within a segment or if the file is a relocatable object.

The `strip` command is used to remove a standard predefined set of sections from an ELF object file. To remove a user specified section by name, see [mcs\(1\)](#).

名前	stty – 端末用オプションの設定								
形式	<pre> /usr/bin/stty [-a] [-g] /usr/bin/stty [modes] /usr/xpg4/bin/stty [-a -g] /usr/xpg4/bin/stty [modes] /usr/xpg6/bin/stty [-a -g] /usr/xpg6/bin/stty [modes] </pre>								
機能説明	stty コマンドは、現在標準入力に使用しているデバイスに 特定の端末入出力 (I/O) オプションを設定します。引数を指定しないで実行すると、特定のオプションの設定を報告します。 この報告で、直前にキャレット (^) を伴う文字がある場合、そのオプションの値は対応する制御文字になります。たとえば、<code>^h</code> は CTRL-H を表します (CTRL-H は Backspace キーと同じ)。<code>^@</code> は、オプションに NULL 値があることを示しています。 この「制御モード」の節から「ローカルモード」の節で紹介するモードの詳細については、termio(7I) を参照してください。「ハードウェアフロー制御モード」の節と「クロックモード」の節で紹介するモードの詳細については、termiox(7I) を参照してください。 「組み合わせモード」のセクションに示すオペランドを使用するときは、これより前のセクションで説明するオプションが必要です。あまり多くのオプションを組み合わせても意味がありませんが、組み合わせるオプションの数をチェックする機能は特に用意されていません。また、ハードウェアインタフェースの種類によっては、ハードウェアフロー制御オプションやクロックモードオプションがサポートされない場合があります。								
オプション	サポートしているオプションは、次のとおりです。 -a 使用端末のオプション設定をすべて標準出力に書き出します。 -g 現在の設定を報告します。この報告は、他の stty コマンドの引数として使用できる形式で出力されます。ドライバが <code>termios</code> をサポートしている場合は <code>termios</code> 型、そうでない場合は <code>termio</code> 型の出力になります。								
オペランド	次の <i>mode</i> オペランドがサポートされています。								
制御モード	<table> <tr> <td><code>parenb(-parenb)</code></td><td>パリティの生成と検出を有効または無効にします。</td></tr> <tr> <td><code>parext(-parext)</code></td><td>拡張パリティの生成と、マークパリティおよびスペースパリティの検出を有効または無効にします。</td></tr> <tr> <td><code>parodd(-parodd)</code></td><td>奇数 (偶数) パリティを選択します。<code>parext</code> が有効になっている場合はマーク (スペース) パリティを選択します。</td></tr> <tr> <td><code>cs5 cs6 cs7 cs8</code></td><td>文字サイズを選択します (termio(7I) を参照)。</td></tr> </table>	<code>parenb(-parenb)</code>	パリティの生成と検出を有効または無効にします。	<code>parext(-parext)</code>	拡張パリティの生成と、マークパリティおよびスペースパリティの検出を有効または無効にします。	<code>parodd(-parodd)</code>	奇数 (偶数) パリティを選択します。 <code>parext</code> が有効になっている場合はマーク (スペース) パリティを選択します。	<code>cs5 cs6 cs7 cs8</code>	文字サイズを選択します (termio(7I) を参照)。
<code>parenb(-parenb)</code>	パリティの生成と検出を有効または無効にします。								
<code>parext(-parext)</code>	拡張パリティの生成と、マークパリティおよびスペースパリティの検出を有効または無効にします。								
<code>parodd(-parodd)</code>	奇数 (偶数) パリティを選択します。 <code>parext</code> が有効になっている場合はマーク (スペース) パリティを選択します。								
<code>cs5 cs6 cs7 cs8</code>	文字サイズを選択します (termio(7I) を参照)。								

<code>0</code>	ただちに回線を切断します。
<code>hupcl (-hupcl)</code>	最終的な終了時に接続を切断するかしないかを指定します。
<code>hup (-hup)</code>	<code>hupcl (-hupcl)</code> と同じです。
<code>cstopb (-cstopb)</code>	文字ごとに2つ(1つ)の停止ビットを使用します。
<code>cread (-cread)</code>	受信を有効または無効にします。
<code>crtsets (-crtsets)</code>	出力ハードウェアフロー制御を有効にします。RTS (送信依頼) モデム制御回線を立てます。CTS (送信可能) 回線が立てられるまで、出力を保留します。
<code>crtsexoff (-crtsexoff)</code>	入力ハードウェアフロー制御を有効にします。データ受け取りのため RTS (送信依頼) モデム制御回線を立てます。RTS の信号がオフの場合、入力を保留します。
<code>clocal (-clocal)</code>	回線でモデム制御が行われていない、または行われているものと想定します。
<code>defecuw</code>	マルチバイト文字の幅を、 <code>LC_CTYPE</code> で指定された現在のロケールで定義されている値に設定します。内部的には、文字幅はバイト単位で文字ごとに表現されます。また、スクリーンまたはディスプレイカラムで文字ごとに表現されます。
<code>110 300 600 1200 1800 2400 4800 9600 19200 38400 357600 76800 115200 153600 230400 307200 460800</code>	端末ボーレートに指定された可能な数値を設定します。すべてのハードウェアインタフェースがすべての速度をサポートしているわけではありません。
<code>ispeed 0 110 300 600 1200 1800 2400 4800 9600 19200 38400 57600 76800 115200 153600 230400 307200 460800</code>	端末入力ボーレートに指定された可能な数値を設定します。ただし、すべてのハードウェアが分割ボーレートをサポートしているわけではありません。入力ボーレートを <code>0</code> に設定すると、この値は出力ボーレートの値によって指定されます。
<code>ospeed 0 110 300 600 1200 1800 2400 4800 9600 19200 38400 57600 76800 115200 153600 230400 307200 460800</code>	端末出力ボーレートに指定された可能な数値を設定します。ただし、すべてのハードウェアが分割

ボーレートをサポートしているわけではありません。出力ボーレートを 0 に設定すると、ただちに回線が切断されます。

入力モード	<code>ignbrk (-ignbrk)</code>	入力時のブレークを無視するかどうかを指定します。
	<code>brkint (-brkint)</code>	ブレーク時に INTR シグナルを送るかどうかを指定します。
	<code>ignpar (-ignpar)</code>	パリティエラーを無視するかどうかを指定します。
	<code>parmrk (-parmrk)</code>	パリティエラーをマークするかどうかを指定します (termio(7I) を参照)。
	<code>inpck (-inpck)</code>	入力パリティチェックを有効または無効にします。
	<code>istrip (-istrip)</code>	入力文字を 7 ビットにストリップするか、ストリップしないかを指定します。
	<code>inlcr (-inlcr)</code>	入力時に NL を CR に変換するかどうかを指定します。
	<code>igncr (-igncr)</code>	入力時に CR を無視するかどうかを指定します。
	<code>icrnl (-icrnl)</code>	入力時に CR を NL に変換するかどうかを指定します。
	<code>iuclic (-iuclic)</code>	入力時に大文字のアルファベットを小文字のアルファベットに変換するかどうかを指定します。
	<code>ixon (-ixon)</code>	START/STOP 出力制御を有効または無効にします。STOP 制御文字を送信することによって出力を停止し、START 制御文字を送信することによって出力を開始します。
	<code>ixany (-ixany)</code>	任意の文字 (DC1 のみ) に出力の再開を許可します。
	<code>ixoff (-ixoff)</code>	入力待ち行列がほとんど空であるか、いっぱいになっているときに、システムから START/STOP 文字が送信されるように、またはされないようにします。
	<code>imaxbel (-imaxbel)</code>	入力行が長すぎるときに、BEL をエコーするかしないかを指定します。imaxbel を設定した場合、入力ストリームがオーバーフローすると、ASCII BEL の BEL 文字 (16 進数の 07) がエコーされます。これ以上の入力は格納されませんが、すでに (入力待ち行列内に) 存在する入力には影響ありません。-imaxbel を設定した場合、入力ストリームがオーバーフローしても、BEL 文字はエコーされず、入力待ち行列内に存在する (読み込まれていない) 入力はすべて破棄されます。
出力モード	<code>opost (-opost)</code>	出力の後処理を行うかどうかを指定します。後処理を行わない場合は、他のすべての出力モードを無視します。
	<code>olcuc (-olcuc)</code>	出力時に小文字のアルファベットを大文字のアルファベットに変換するかどうかを指定します。

ローカルモード	<code>onlcr (-onlcr)</code>	出力時に NL を CR-NL に変換するかどうかを指定します。
	<code>ocrnl (-ocrnl)</code>	出力時に CR を NL に変換するかどうかを指定します。
	<code>onocr (-onocr)</code>	カラム 0 に CR を出力するかどうかを指定します。
	<code>onlret (-onlret)</code>	端末で NL が CR 関数を実行するかどうかを指定します。
	<code>ofill (-ofill)</code>	遅延時に fill 文字を使用します。または、タイミングを使用します。
	<code>ofdel (-ofdel)</code>	fill 文字は DEL (NUL) です。
	<code>cr0 cr1 cr2 cr3</code>	キャリッジリターンの遅延スタイルを選択します (termio(7I) を参照)。
	<code>nl0 nl1</code>	ラインフィードの遅延スタイルを選択します (termio(7I) を参照)。
	<code>tab0 tab1 tab2 tab3</code>	水平タブの遅延スタイルを選択します (termio(7I) を参照)。
	<code>bs0 bs1</code>	バックスペースの遅延スタイルを選択します (termio(7I) を参照)。
	<code>ff0 ff1</code>	フォームフィードの遅延スタイルを選択します (termio(7I) を参照)。
	<code>vt0 vt1</code>	垂直タブの遅延スタイルを選択します (termio(7I) を参照)。
	<code>isig (-isig)</code>	特殊制御文字 INTR、QUIT、SWTCH、SUSP に対する文字のチェックを有効または無効にします。SWTCH については、「注意」の節を参照してください。
	<code>icanon (-icanon)</code>	標準的な入力 (ERASE および KILL の処理) を有効または無効にします。MIN や TIME は設定されません。
	<code>xcase (-xcase)</code>	標準的な (未処理の) 大文字/小文字を表示します。
	<code>echo (-echo)</code>	入力されたすべての文字を表示するかどうかを指定します。
	<code>echoe (-echoe)</code>	ERASE 文字を「バックスペース-スペース文字-バックスペース」の文字列として表示するかどうかを指定します。このモードを使用すると、多くの CRT 端末では、ERASE された文字が消去されます。ただし、このときカラム位置は記録されないため、エスケープした文字、タブ、バックスペースの区別が付きにくくなります。
	<code>echok (-echok)</code>	KILL 文字の後の NL を表示するかどうかを指定します。
	<code>lfkc (-lfkc)</code>	<code>echok (-echok)</code> と同じです。この指定は廃止されました。
	<code>echonl (-echonl)</code>	NL をエコーするかどうかを指定します。

	<code>noflsh (-noflsh)</code>	INTR、QUIT、SUSP の後のフラッシュを無効または有効にします。
	<code>stwrap (-stwrap)</code>	同期回線で 79 文字より長い行の切り捨てを無効または有効にします。
	<code>tostop (-tostop)</code>	バックグラウンドプロセスが端末に書き込むときに、SIGTTOU を送信するかどうかを指定します。
	<code>echoctl (-echoctl)</code>	制御文字を <i>^char</i> としてエコーし、 <i>^?</i> として削除します。または、エコーしません。
	<code>echoprt (-echoprt)</code>	消去文字を、消去されたものとしてエコーするかどうかを指定します。
	<code>echoke (-echoke)</code>	行の削除の際、BS-SP-BS で行全体を削除するかどうかを指定します。
	<code>flusho (-flusho)</code>	出力をフラッシュするかどうかを指定します。
	<code>pendin (-pendin)</code>	次の読み取りまたは入力文字で、保留中の入力を再入力するかどうかを指定します。
	<code>ixten (-ixten)</code>	現在、 <code>icanon</code> 、 <code>isig</code> 、 <code>ixon</code> 、 <code>ixoff</code> のモード指定によって制御できない特殊な制御文字を有効または無効にします。このような特殊文字には、VEOL、VSWTCH、VREPRINT、VDISCARD、VDSUSP、VWERASE、VLNEXT があります。
	<code>stflush (-stflush)</code>	write(2) を実行するたびに、その後で同期回線でのフラッシュを有効または無効にします。
	<code>stappl (-stappl)</code>	同期回線でアプリケーションモードまたは回線モードを使用します。
ハードウェアフロー制御モード	<code>rtsxoff (-rtsxoff)</code>	入力時に RTS ハードウェアフロー制御を有効または無効にします。
	<code>ctsxon (-ctsxon)</code>	出力時の CTS ハードウェアフロー制御を有効または無効にします。
	<code>dtrxoff (-dtrxoff)</code>	入力時の DTR ハードウェアフロー制御を有効または無効にします。
	<code>cdxon (-cdxon)</code>	出力時の CD ハードウェアフロー制御を有効または無効にします。
	<code>isxoff (-isxoff)</code>	入力時の等時的なハードウェアフロー制御を有効または無効にします。

クロックモード	xcibrg	内部ボーレートジェネレータから送信クロックを取得します。
	xctset	送信シグナル要素のリードタイミング (DCE ソース)、CCITT V.24 回線 114、EIA-232-D ピン 15 から送信クロックを取得します。
	xcrset	受信シグナル要素のリードタイミング (DCE ソース)、CCITT V.24 回線 115、EIA-232-D ピン 17 から送信クロックを取得します。
	rcibrg	内部ボーレートジェネレータから受信クロックを取得します。
	rctset	送信シグナル要素のリードタイミング (DCE ソース)、CCITT V.24 回線 114、EIA-232-D ピン 15 から受信クロックを取得します。
	rcrset	受信シグナル要素のリードタイミング (DCE ソース)、CCITT V.24 回線 115、EIA-232-D ピン 17 から受信クロックを取得します。
	tsetcoff	送信シグナル要素のタイミングクロックが提供されていません。
	tsetcrbrg	送信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 113、EIA-232-D ピン 24 上の受信ボーレートジェネレータを出力します。
	tsetctbrg	受信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 113、EIA-232-D ピン 24 上の送信ボーレートジェネレータを出力します。
	tsetctset	送信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 113、EIA-232-D ピン 24 上の送信シグナル要素のタイミング (DCE ソース) を出力します。
	tsetcrset	送信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 113、EIA-232-D ピン 24 上の受信シグナル要素のタイミング (DCE ソース) を出力します。
	rsetcoff	受信シグナル要素のタイミングクロックが提供されていません。
	rsetcrbrg	受信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 128、非 EIA-232-D ピン上の受信ボーレートジェネレータを出力します。
	rsetctbrg	受信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 128、非 EIA-232-D ピン上の送信ボーレートジェネレータを出力します。
	rsetctset	受信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 128、非 EIA-232-D ピン上の送信シグナル要素のタイミング (DCE ソース) を出力します。
	rsetcrset	受信シグナル要素のリードタイミング (DTE ソース)、CCITT V.24 回線 128、非 EIA-232-D ピン上の受信シグナル要素タイミング (DCE ソース) を出力します。

制御の割り当て *control-character c*

control-character を *c* に設定します。control-character、*c* には、それぞれ次の値を使用できます。

control-character ctab, discard, dsusp, eof, eol, eol2, erase, intr, kill, lnext, (ctab は -stappl とともに使用。termio(7I) を参照)。swtch については、「注意」の節を参照してください。

c *c* が単一の文字である場合、制御文字はその文字に設定されます。

POSIX ロケールでは、*c* の前にシェルからのエスケープを表すキャレット (^) が付いている場合、その値は制御文字に対応しています。この対応関係については、次の表を参照してください。たとえば、^d は CTRL-D を表し、^? は DEL と解釈されます。また、^_ は未定義です。

^c 値		^c 値		^c 値	
a, A	<SOH>	l, L	<FF>	w, W	<ETB>
b, B	<STX>	m, M	<CR>	x, X	<CAN>
c, C	<ETX>	n, N	<SO>	y, Y	
d, D	<EOT>	o, O	<SI>	z, Z	<SUB>
e, E	<ENQ>	p, P	<DLE>	[	<ESC>
f, F	<ACK>	q, Q	<DC1>	\	<FS>
g, G	<BEL>	r, R	<DC2>	]	<GS>
h, H	<BS>	s, S	<DC3>	^	<RS>
i, I	<HT>	t, T	<DC4>	_	<US>
j, J	<LF>	u, U	<NAK>	?	<DEL>
k, K	<VT>	v, V	<SYN>		

min number

time number *number* に min または time の値を設定します。MIN と TIME は非標準モードの入力処理 (-icanon) で使用されます。

line i 回線制御手順を *i* に設定します (0<*i*<127)。

組み合わせモード *saved settings* 現在の端末特性を -g オプションで生成され、保存された設定に変更します。

	evenp または parity	parenb と cs7 を有効にするか、parodd を無効にします。
	oddp	parenb、cs7、parodd を有効にします。
	spacep	parenb、cs7、parext を有効にします。
	markp	parenb、cs7、parodd、parext を有効にします。
	-parity または -evenp	parenb を無効にし、cs8 を設定します。
	-oddp	parenb と parodd を無効にし、cs8 を設定します。
	-spacep	parenb と parext を無効にし、cs8 を設定します。
	-markp	parenb、parodd、parext を無効にし、cs8 を設定します。
	raw (-raw または cooked)	raw 入力と raw 出力を有効または無効にします。raw モードは次のように設定した場合と同等です。 stty cs8 -icanon min 1 time 0 -isig -xcase \ -inpck -opost
/usr/bin/stty, /usr/xpg6/bin/stty	nl (-nl)	icrnl と onlcr の設定を解除、または設定します。-nl は、inlcr、igncr、ocrnl、onlret の設定も解除します。
/usr/xpg4/bin/stty	nl (-nl)	icrnl を設定、または設定解除します。-nl は、inlcr、igncr、ocrnl、onlret の設定も解除します。つまり、-nl は onlcr を設定し、nl は onlcr の設定を解除します。
	lcase (-lcase)	xcase、iuclc、olcuc を設定、または設定解除します。
	LCASE (-LCASE)	lcase (-lcase) と同じです。
	tabs (-tabs または tab3)	出力時にタブを保存 (スペースを拡張) します。
	ek	ERASE 文字、KILL 文字を、通常の DEL および CTRL-u にそれぞれリセットします。
	sane	すべてのモードを適切な値にリセットします。
	term	すべてのモードを端末タイプ <i>term</i> に合うように設定します。ここで、 <i>term</i> は、tty33、tty37、vt05、tn300、ti700、tek のいずれかです。
	async	通常の非同期通信 (クロック設定が xcibrg、rcibrg、tsetcoff、rsetcoff) を設定します。

ウィンドウサイズ `rows n` ウィンドウサイズを *n* 行に設定します。

`columns n` ウィンドウサイズを *n* カラムに設定します。

`cols n` ウィンドウサイズを *n* カラムに設定します。 `cols` は、`columns` の省略形です。

`ypixels n` 垂直ウィンドウサイズを *n* ピクセルに設定します。

`xpixels n` 水平ウィンドウサイズを *n* ピクセルに設定します。

使用法 `-g` フラグは、シェルレベルから端末状態の保存や復元を行うために 指定します。次に、プログラムの例を示します。

```
saveterm="$(stty -g)"        # save terminal state
stty (new settings)        # set new state
...                        # ...
stty $saveterm             # restore terminal state
```

`-a` を使用すると、表記が冗長になります。このため、端末設定の保存および復元を行うスクリプトには、`-g` オプションを使用してください。

環境変数 `stty` の実行に影響を与える次の環境変数の詳細については、[environ\(5\)](#) を参照してください。 `LANG`、`LC_ALL`、`LC_CTYPE`、`LC_MESSAGES`、および `NLSPATH`。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

`/usr/bin/stty`

属性タイプ	属性値
使用条件	system/core-os

`/usr/xpg4/bin/stty`

属性タイプ	属性値
使用条件	system/xopen/xcu4
インタフェースの安定性	確実
標準	standards(5) を参照してください。

`/usr/xpg6/bin/stty`

属性タイプ	属性値
使用条件	system/xopen/xcu6

属性タイプ	属性値
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目
- [tabs\(1\)](#), [ioctl\(2\)](#), [write\(2\)](#), [getwidth\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#), [ldterm\(7M\)](#), [termio\(7I\)](#), [termiox\(7I\)](#)
- 注意事項
- Solaris は System V リリース 4 の sxt ドライバが使用していた `swtch` のアクションをサポートしません。Solaris は、`swtch` 値を設定または出力できますが、`swtch` のアクションは実行できません。

Solaris 上のジョブ切り替え機能は、実際には、ジョブ制御により処理されます。これに対する正しい設定は `susp` です。

名前	stty – set the options for a terminal	
形式	/usr/ucb/stty [-a] [-g] [-h] [<i>modes</i>]	
機能説明	stty sets certain terminal I/O options for the device that is the current standard output. Without arguments, stty reports the settings of certain options.	
オプション	In this report, if a character is preceded by a caret (^), then the value of that option is the corresponding CTRL character (for example, ^h is CTRL-h. In this case, recall that CTRL-h is the same as the BACKSPACE key.) The sequence ^@ means that an option has a null value. -a Reports all of the option settings. -g Reports current settings in a form that can be used as an argument to another stty command. -h Reports all the option settings with the control characters in an easy to read column format. Options in the last group are implemented using options in the previous groups. Many combinations of options make no sense, but no sanity checking is performed. Hardware flow control and clock modes options might not be supported by all hardware interfaces. The options are selected from the following:	
Special Requests	all	Reports the same option settings as stty without arguments, but with the control characters in column format.
	everything	Everything stty knows about is printed. Same as -h option.
	speed	The terminal speed alone is reported on the standard output.
	size	The terminal (window) sizes are printed on the standard output, first rows and then columns. This option is only appropriate if currently running a window system.
		size and speed always report on the settings of /dev/tty, and always report the settings to the standard output.
Control Modes	parenb (-parenb)	Enable (disable) parity generation and detection.
	parext (-parext)	Enable (disable) extended parity generation and detection for mark and space parity.
	parodd (-parodd)	Select odd (even) parity, or mark (space) parity if parext is enabled.
	cs5 cs6 cs7 cs8	Select character size (see termio(7I)).
	0	Hang up line immediately.

110 300 600 1200 1800 2400 4800 9600 19200 exta 38400 extb

Set terminal baud rate to the number given, if possible. (All speeds are not supported by all hardware interfaces.)

ispeed 0 110 300 600 1200 1800 2400 4800 9600 19200 exta 38400 extb

Set terminal input baud rate to the number given, if possible. (Not all hardware supports split baud rates.) If the input baud rate is set to zero, the input baud rate is specified by the value of the output baud rate.

ospeed 0 110 300 600 1200 1800 2400 4800 9600 19200 exta 38400 extb

Set terminal output baud rate to the number given, if possible. (Not all hardware supports split baud rates.) If the baud rate is set to zero, the line is hung up immediately.

hupcl (-hupcl)

Hang up (do not hang up) connection on last close.

hup (-hup)

Same as hupcl (-hupcl).

cstopb (-cstopb)

Use two (one) stop bits per character.

cread (-cread)

Enable (disable) the receiver.

clocal (-clocal)

Assume a line without (with) modem control.

crtsets (-crtsets)

Enable hardware flow control. Raise the RTS (Request to Send) modem control line. Suspends output until the CTS (Clear to Send) line is raised.

loblk (-loblk)

Block (do not block) output from a non-current layer.

Input Modes

ignbrk (-ignbrk) Ignore (do not ignore) break on input.

brkint (-brkint) Signal (do not signal) INTR on break.

ignpar (-ignpar) Ignore (do not ignore) parity errors.

parmrk (-parmrk) Mark (do not mark) parity errors (see [termio\(7I\)](#)).

inpck (-inpck) Enable (disable) input parity checking.

istrip (-istrip) Strip (do not strip) input characters to seven bits.

inlcr (-inlcr) Map (do not map) NL to CR on input.

igncr (-igncr) Ignore (do not ignore) CR on input.

icrnl (-icrnl) Map (do not map) CR to NL on input.

iuclc (-iuclc) Map (do not map) upper-case alphabets to lower case on input.

	<code>ixon (-ixon)</code>	Enable (disable) START/STOP output control. Output is stopped by sending an STOP and started by sending an START.
	<code>ixany (-ixany)</code>	Allow any character (only START) to restart output.
	<code>decctlq (-decctlq)</code>	Same as <code>-ixany</code> .
	<code>ixoff (-ixoff)</code>	Request that the system send (not send) START/STOP characters when the input queue is nearly empty/full.
	<code>tandem (-tandem)</code>	Same as <code>ixoff</code> .
	<code>imaxbel (-imaxbel)</code>	Echo (do not echo) BEL when the input line is too long.
	<code>iexten (-iexten)</code>	Enable (disable) extended (implementation-defined) functions for input data.
Output Modes	<code>opost (-opost)</code>	Post-process output (do not post-process output; ignore all other output modes).
	<code>olcuc (-olcuc)</code>	Map (do not map) lower-case alphabets to upper case on output.
	<code>onlcr (-onlcr)</code>	Map (do not map) NL to CR-NL on output.
	<code>ocrnl (-ocrnl)</code>	Map (do not map) CR to NL on output.
	<code>onocr (-onocr)</code>	Do not (do) output CRs at column zero.
	<code>onlret (-onlret)</code>	On the terminal NL performs (does not perform) the CR function.
	<code>ofill (-ofill)</code>	Use fill characters (use timing) for delays.
	<code>ofdel (-ofdel)</code>	Fill characters are DELs (NULs).
	<code>cr0 cr1 cr2 cr3</code>	Select style of delay for carriage returns (see termio(7I)).
	<code>nl0 nl1</code>	Select style of delay for line-feeds (see termio(7I)).
	<code>tab0 tab1 tab2 tab3</code>	Select style of delay for horizontal tabs (see termio(7I)).
	<code>bs0 bs1</code>	Select style of delay for backspaces (see termio(7I)).
	<code>ff0 ff1</code>	Select style of delay for form-feeds (see termio(7I)).
	<code>vt0 vt1</code>	Select style of delay for vertical tabs (see termio(7I)).
Local Modes	<code>isig (-isig)</code>	Enable (disable) the checking of characters against the special control characters INTR, QUIT, and SWTCH. For information on SWTCH, see NOTES.
	<code>icanon (-icanon)</code>	Enable (disable) canonical input (ERASE and KILL processing). Does not set MIN or TIME.
	<code>cbreak (-cbreak)</code>	Equivalent to <code>-icanon min 1 time 0</code> .

<code>xcase (-xcase)</code>	Canonical (unprocessed) upper/lower-case presentation.
<code>echo (-echo)</code>	Echo back (do not echo back) every character typed.
<code>echoe (-echoe)</code>	Echo (do not echo) ERASE character as a backspace-space-backspace string. Note: This mode erases the ERASEed character on many CRT terminals; however, it does <i>not</i> keep track of column position and, as a result, can be confusing on escaped characters, tabs, and backspaces.
<code>crterase (-crterase)</code>	Same as <code>echoe</code> .
<code>echok (-echok)</code>	Echo (do not echo) NL after KILL character.
<code>lfkc (-lfkc)</code>	The same as <code>echok (-echok)</code> ; obsolete.
<code>echonl (-echonl)</code>	Echo (do not echo) NL.
<code>noflsh (-noflsh)</code>	Disable (enable) flush after INTR, QUIT, or SWTCH. For information on SWTCH, see NOTES.
<code>stwrap (-stwrap)</code>	Disable (enable) truncation of lines longer than 79 characters on a synchronous line. (Does not apply to the 3B2.)
<code>tostop (-tostop)</code>	Send (do not send) SIGTTOU for background processes.
<code>echoctl (-echoctl)</code>	Echo (do not echo) control characters as <i>^char</i> , delete as <i>^?</i>
<code>ctlecho (-ctlecho)</code>	Same as <code>echoctl</code> .
<code>echoprt (-echoprt)</code>	Echo (do not echo) erase character as character is “erased”.
<code>prterase (-prterase)</code>	Same as <code>echoprt</code> .
<code>echoke (-echoke)</code>	BS-SP-BS erase (do not BS-SP-BS erase) entire line on line kill.
<code>crtkill (-crtkill)</code>	Same as <code>echoke</code> .
<code>flusho (-flusho)</code>	Output is (is not) being flushed.
<code>pendin (-pendin)</code>	Retype (do not retype) pending input at next read or input character.
<code>stflush (-stflush)</code>	Enable (disable) flush on a synchronous line after every <code>write(2)</code> . (Does not apply to the 3B2.)
<code>stappl (-stappl)</code>	Use application mode (use line mode) on a synchronous line. (Does not apply to the 3B2.)
Hardware Flow Control Modes	<code>rtsxoff (-rtsxoff)</code> Enable (disable) RTS hardware flow control on input.
	<code>ctsxon (-ctsxon)</code> Enable (disable) CTS hardware flow control on output.
	<code>dterxoff (-dterxoff)</code> Enable (disable) DTER hardware flow control on input.

	<code>rlsdxon (-rlsdxon)</code>	Enable (disable) RLSD hardware flow control on output.
	<code>isxoff (-isxoff)</code>	Enable (disable) isochronous hardware flow control on input.
Clock Modes	<code>xcibrg</code>	Get transmit clock from internal baud rate generator.
	<code>xctset</code>	Get the transmit clock from transmitter signal element timing (DCE source) lead, CCITT V.24 circuit 114, EIA-232-D pin 15.
	<code>xcrset</code>	Get transmit clock from receiver signal element timing (DCE source) lead, CCITT V.24 circuit 115, EIA-232-D pin 17.
	<code>rcibrg</code>	Get receive clock from internal baud rate generator.
	<code>rctset</code>	Get receive clock from transmitter signal element timing (DCE source) lead, CCITT V.24 circuit 114, EIA-232-D pin 15.
	<code>rcrset</code>	Get receive clock from receiver signal element timing (DCE source) lead, CCITT V.24 circuit 115, EIA-232-D pin 17.
	<code>tsetcoff</code>	Transmitter signal element timing clock not provided.
	<code>tsetcrc</code>	Output receive clock on transmitter signal element timing (DTE source) lead, CCITT V.24 circuit 113, EIA-232-D pin 24, clock source.
	<code>tsetcxc</code>	Output transmit clock on transmitter signal element timing (DTE source) lead, CCITT V.24 circuit 113, EIA-232-D pin 24, clock source.
	<code>rsetcoff</code>	Receiver signal element timing clock not provided.
	<code>rsetcrc</code>	Output receive clock on receiver signal element timing (DTE source) lead, CCITT V.24 circuit 128, no EIA-232-D pin, clock source.
	<code>rsetcxc</code>	Output transmit clock on receiver signal element timing (DTE source) lead, CCITT V.24 circuit 128, no EIA-232-D pin, clock source.
Control Assignments	<code>control-character c</code>	Set <i>control-character</i> to <i>c</i> , where <i>control-character</i> is intr, quit, erase, kill, eof, eol, eol2, swtch, start, stop, susp, dsusp, rprnt, flush, werase, lnext min, ctab, time, or brk) (ctab is used with -stappl; min and time are used with -icanon; see termio(7I)). If <i>c</i> is preceded by an (escaped from the shell) caret (^), then the value used is the corresponding CTRL character (for example, “^d” is a Control-d). “^?” is interpreted as DEL and “^–” is interpreted as undefined. For information on SWTCH, see NOTES.
	<code>line i</code>	Set line discipline to <i>i</i> ($0 < i < 127$).
Combination Modes	<code>evenp or parity</code>	Enable parenb and cs7.
	<code>-evenp, or -parity</code>	Disable parenb, and set cs8.
	<code>even (-even)</code>	Same as evenp (-evenp).

oddp	Enable parenb, cs7, and parodd.
-oddp	Disable parenb and parodd, and set cs8.
odd (-odd)	Same as oddp (-oddp).
spacep	Enable parenb, cs7, and parext.
-spacep	Disable parenb and parext, and set cs8.
markp	Enable parenb, cs7, parodd, and parext.
-markp	Disable parenb, parodd, and parext, and set cs8.
raw (-raw or cooked)	Enable (disable) raw input and output (no ERASE, KILL, INTR, QUIT, SWTCH, EOT, or output post processing). For information on SWTCH, see NOTES.
nl (-nl)	Unset (set) icrnl, onlcr. In addition -nl unsets inlcr, igncr, ocrnl, and onlret.
lcase (-lcase)	Set (unset) xcase, iuclc, and olcuc.
LCASE (-LCASE)	Same as lcase (-lcase).
tabs (-tabs or tab3)	Preserve (expand to spaces) tabs when printing.
ek	Reset ERASE and KILL characters back to normal DEL and CTRL-u, respectively.
sane	Reset all modes to some reasonable values.
term	Set all modes suitable for the terminal type <i>term</i> , where <i>term</i> is one of tty33, tty37, vt05, tn300, ti700, or tek.
async	Set normal asynchronous communications where clock settings are xcibrg, rcibrg, tsetcoff and rsetcoff.
litout (-litout)	Disable (enable) parenb, istrip, and opost, and set cs8 (cs7).
pass8 (-pass8)	Disable (enable) parenb and istrip, and set cs8 (cs7).
crt	Set options for a CRT (echoe, echoctl, and, if >= 1200 baud, echoke.)
dec	Set all modes suitable for Digital Equipment Corp. operating systems users ERASE, KILL, and INTR characters to ^?, ^U, and ^C, decctlq, and crt.)

Window Size

rows <i>n</i>	Set window size to <i>n</i> rows.
columns <i>n</i>	Set window size to <i>n</i> columns.
cols <i>n</i>	An alias for columns <i>n</i> .

`ypixels`*n* Set vertical window size to *n pixels*.
`xpixels`*n* Set horizontal window size to *n pixels*.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [tabs\(1\)](#), [ioctl\(2\)](#), [attributes\(5\)](#), [termio\(7I\)](#), [termiox\(7I\)](#)

注意事項 Solaris does not support any of the actions implied by `swtch`, which was used by the `sxt` driver on System V release 4. Solaris allows the `swtch` value to be set, and prints it out if set, but it does not perform the `swtch` action.

The job switch functionality on Solaris is actually handled by job control. `susp` is the correct setting for this.

名前	sum – ファイルのチェックサムおよびブロックカウントの出力
形式	/usr/bin/sum [-r] [file...]
機能説明	sum は、各ファイル引数のチェックサムを一覧表示します。ファイル引数が指定されていない場合は、標準入力から読み取られます。
オプション	次のオプションがサポートされています。 -r チェックサムの計算には、代替(マシン依存型)アルゴリズムを使用します。
オペランド	次のオペランドがサポートされています。 file ファイルのパス名。ファイルの指定がない場合は、標準入力を使用します。
使用法	ファイルが2G バイト (2 ³¹ バイト) 以上ある場合の sum の動作については、 largefile(5) を参照してください。
環境変数	sum の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LC_CTYPE、LC_MESSAGES、およびNLSPATH。
終了ステータス	次の終了値が返されます。 0 正常終了。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

関連項目	cksum(1) , getconf(1) , sum(1B) , wc(1) , libmd(3LIB) , attributes(5) , environ(5) , largefile(5)
診断	大部分のデバイスでは、読み取りエラーとファイルの終わりとを区別できません。ブロック数をチェックしてください。
注意事項	移植性の必要なアプリケーションには cksum(1) を使用してください。このコマンドのデフォルトアルゴリズムは POSIX 標準に定義されていて、プロットフォーム間で同一です。 sum と usr/ucb/sum (sum(1B) 参照) の返すチェックサムは異なります。

名前	sum – calculate a checksum for a file
形式	<code>/usr/ucb/sum file...</code>
機能説明	sum calculates and displays a 16-bit checksum for the named file and displays the size of the file in kilobytes. It is typically used to look for bad spots, or to validate a file communicated over some transmission line. The checksum is calculated by an algorithm which may yield different results on machines with 16-bit ints and machines with 32-bit ints, so it cannot always be used to validate that a file has been transferred between machines with different-sized ints.
使用法	See largefile(5) for the description of the behavior of sum when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目	sum(1) , wc(1) , attributes(5) , largefile(5)
診断	Read error is indistinguishable from EOF on most devices; check the block count.
注意事項	sum and <code>/usr/bin/sum</code> (see sum(1)) return different checksums. This utility is obsolete.

名前 suspend – 現在のシェルを停止させるためのシェル組み込み関数

形式

sh suspend
csh suspend
ksh suspend

機能説明

sh 現在のシェルがログインシェルでない場合、その実行を中断します。

csh ^Z を使用して停止シグナルが送信されたときと同様に、トラック内でシェルを停止します。このコマンドは、su コマンドによって開始されたシェルを停止するときによく使用します。

ksh 現在のシェルがログインシェルでない場合、その実行を中断します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [csh\(1\)](#), [kill\(1\)](#), [ksh\(1\)](#), [sh\(1\)](#), [su\(1M\)](#), [attributes\(5\)](#)

名前	svccprop - サービス設定プロパティの値の取得
形式	<pre>svccprop [-afqtv] [-C -c -s snapshot] [-p [name/]name]... [-g pgtype]... -l layer[,...] {FMRI pattern}... svccprop -w [-fqtv] [-p [name/]name] {FMRI pattern}</pre>
機能説明	svccprop ユーティリティーは、サービス設定リポジトリにあるプロパティの値を出力します。プロパティの選択は、<code>-p</code> オプションとオペランドで行います。<code>-p</code> を使用してプロパティまたはプロパティグループを明確に選択しない場合、svccprop は SMF テンプレート定義プロパティを出力しません。将来のリリースでは、より多くの SMF インフラストラクチャープロパティがデフォルト出力で非表示になる可能性があります。<code>-a</code> を使用すると、SMF テンプレート定義プロパティを含むすべてのプロパティが表示されます。 <code>-C</code>、<code>-c</code>、または <code>-s</code> オプションを使用しない場合は、svccprop は有効なプロパティにアクセスします。サービスの有効なプロパティは、サービスに直接付属するプロパティです。サービスインスタンスの有効なプロパティは、「実行中の」スナップショットの合成されたビューのプロパティと、インスタンスに直接付属するプロパティの合成されたビューにある非永続的なプロパティグループ内のプロパティとの集合です。プロパティの合成についての説明は、smf(5) を参照してください。「実行中の」スナップショットが存在しない場合、代わりにインスタンスに直接付属するプロパティが使用されます。
出力形式	デフォルトでは、1つのプロパティを選択した場合には、それぞれの値が別の行に出力されます。空の ASCII 文字列は、二重引用符 ("") で表現されます。ASCII 文字列の値に Bourne シェルのメタキャラクタ (<code>;</code>、<code>&</code>、<code>(</code>、<code>)</code>、<code> </code>、<code>^</code>、<code><</code>、<code>></code>、改行、スペース、タブ、バックスラッシュ、<code>"</code>、単一引用符、<code>'</code>) が含まれる場合は、そのキャラクタの前にバックスラッシュ (<code>\</code>) が付きます。 複数のプロパティを選択した場合は、各プロパティが1行で出力されます。各行はプロパティ指示子、プロパティタイプ、および値で構成され、それぞれの間はスペース文字で区切られます。値は、前述の説明のとおりに表示されます。デフォルトでは、FMRI オペランドを1つだけ指定した場合のプロパティ指示子は、プロパティグループ名、スラッシュ (<code>/</code>)、プロパティ名という構成になります。複数の FMRI オペランドを指定した場合のプロパティ指示子は、そのプロパティの正規の FMRI になります。 アクセス制御によってプロパティ値の読み取りが禁止されていて、プロパティまたはプロパティグループが <code>-p</code> オプションによって明示的に指定されていない場合、プロパティは値がない場合のように表示されます。プロパティまたはプロパティグループの名前が <code>-p</code> オプションによって1つ以上指定されていて、アクセス制御のためにプロパティ値が読み取れない場合は、エラーが発生します。 エラーメッセージが標準エラーストリームに出力されます。

オプション

サポートしているオプションは、次のとおりです。

-a

SMF テンプレート定義プロパティグループのプロパティを含む、すべてのプロパティを表示します。

-C

直接付属するプロパティを合成しないで使用します。

-c

サービスインスタンスの場合、直接付属するプロパティの合成されたビューを使用します。

-f

完全な FMRI を指示子として、複数プロパティ出力形式を選択します。

-g *pgtype*

pgtype タイプのプロパティグループに属する、選択されたプロパティのみを表示します。複数の -g オプションが指定されると、複数のプロパティグループタイプのプロパティを表示します。このオプションは、複数プロパティ出力形式 (-t) を暗黙的に意味します。

-l *layer*[,...]

選択されたレイヤーで定義されたプロパティのみを表示します。使用可能なレイヤーは、manifest、system-profile、site-profile、および admin です。すべてのレイヤーの選択に別名 all を使用できます。このオプションを使用する場合、非持続性プロパティグループに属するプロパティは表示されません。

-p *name*

オペランドで指定されたそれぞれのサービスまたはサービスインスタンスについて、*name* プロパティグループ内のすべてのプロパティを選択します。オペランドで指定されたプロパティグループについては、*name* プロパティを選択します。-g と同時に使用する場合、一致するすべてのプロパティグループタイプで *name* プロパティを選択します。

-p *pg/prop*

オペランドによって指定されたそれぞれのサービスまたはサービスインスタンスについて、プロパティグループ *pg* 内のプロパティ *prop* を選択します。

-q

非出力。出力を生成しません。

-s *name*

サービスインスタンスについて、*name* スナップショットの合成されたビューを使用します。

-t

複数のプロパティを出力する形式を選択します。

-v

冗長。オプション -q が使用されている場合でも、存在しないプロパティに関するエラーメッセージを出力します。

-w

指定されたプロパティグループ、または指定されたプロパティを含むプロパティグループが変更されるまで待機してから、出力します。

このオプションは、指定されているエンティティが1つの場合にのみ有効になります。複数のオペランドが指定されている場合、または1つのオペランドが複数のインスタンスに一致する場合は、エラーメッセージが出力され、処理は行われません。-C オプションの意味を含んでいます。

オペランド

次のオペランドがサポートされています。

FMRI

サービスの FMRI、サービスインスタンス、プロパティグループ、またはプロパティ。

インスタンスおよびサービスは、インスタンス名、またはサービス名の最後の部分を指定する方法で、省略して入力することができます。プロパティおよびプロパティグループの場合は、完全な FMRI を指定する必要があります。たとえば、次の FMRI があるとして。

```
svc:/network/smtp:sendmail
```

次のような省略が有効です。

```
sendmail
:sendmail
smtp
smtp:sendmail
network/smtp
```

次のような省略は無効です。

```
mailnetwork
network/smt
```

FMRI の省略形は「不確実」なので、スクリプトやその他の長期に渡って使用するツールには使用しないでください。省略形が複数のインスタンスとマッチする場合、svccprop は各インスタンス上で動作します。

pattern

リポジトリ内のサービスおよびサービスインスタンスの FMRI に対して一致した glob パターン。fnmatch(5) を参照してください。パターンが複数のサービスまたはインスタンスとマッチする場合、svccprop は各サービスまたはインスタンス上で動作します。

使用例

例1 1つのプロパティーの値を表示する

次の例は、サービス `system/cron` のインスタンス `default` のリスタートプロパティーグループ内にある、状態プロパティーの値を表示します。

```
example% svccprop -p restarter/state system/cron:default
online
```

例2 サービスが有効かどうかの情報を取得する

サービスが有効かどうかは、その `-general/enabled` プロパティーによって決定されます。このプロパティーはすぐに影響するため、`-c` オプションを使用する必要があります。

```
example% svccprop -c -p general/enabled system/cron:default
true
```

例3 特定のプロパティーグループに含まれるすべてのプロパティーを表示する

Solaris のデフォルトのインストールでは、次の例は `network/ntp` サービスのそれぞれのインスタンスの `general` プロパティーグループ内にある、すべてのプロパティーを表示します。

```
example% svccprop -p general ntp
general/package astring SUNWntpr
general/enabled boolean true
general/entity_stability astring Uncommitted
general/single_instance boolean true
```

例4 プロパティーが存在することをテストする

次の例では、`identity` サービスのすべてのインスタンスの `general/enabled` プロパティーが存在することをテストします。

```
example% svccprop -q -p general/enabled identity:
example% echo $?
0
```

例5 プロパティーが変更されるのを待機する

次の例では、`sendmail` インスタンスの状態が変更されるのを待機します。

```
example% svccprop -w -p restarter/state sendmail
```

例6 スクリプトの Boolean プロパティーの値を取得する

次の例は、スクリプトの `boolean` プロパティーの値を取得します。

```
set -- 'svccprop -c -t -p general/enabled service'
code=$?
if [ $code -ne 0 ]; then
    echo "svccprop failed with exit code $code"
    return 1
fi
```


例6 スクリプトの Boolean プロパティの値を取得する (続き)

```
fi
if [ $2 != boolean ]; then
    echo "general/enabled has unexpected type $2"
    return 2
fi
if [ $# -ne 3 ]; then
    echo "general/enabled has wrong number of values"
    return 3
fi
value=$3
...
```

例7 svccprop をスクリプトで使用する

次の例では、サービスプロパティの値を取得し、その値をスクリプト (/usr/bin/Xserver) 内で使用します。

```
fmri=$1
prop=$2
if svccprop -q -p ${prop} ${fmri} ; then
    propval="$(svccprop -p ${prop} "${fmri}")"
    if [ "${propval}" == "\"\"" ] ; then
        propval=""
    fi
fi
```

例8 プロパティグループタイプによる出力のフィルタ処理

次の例では、svc:/network/ssh:default のメソッドを取得します。

```
example% svccprop -p exec -g method svc:/network/ssh:default
start/exec astring /lib/svc/method/sshd\ start
stop/exec astring :kill
refresh/exec astring /lib/svc/method/sshd\ restart
unconfigure/exec astring /lib/svc/method/sshd\ -u
```

例9 管理用にカスタマイズされたプロパティの表示

次のコマンドは、SMF レイヤーを使用して管理用にカスタマイズされたプロパティを表示します。

```
example% svccprop -p config -l admin svc:/network/dns/client
config/domain      astring      admin      my.domain.com
config/nameserver  net_address  admin      10.22.33.44  10.44.33.11
```

終了ステータス 次の終了ステータスが返されます。

```
0
  正常終了。
```

- 1 エラーが発生した。
- 2 無効なコマンド行オプションが指定された。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [svcs\(1\)](#), [inetd\(1M\)](#), [svcadm\(1M\)](#), [svccfg\(1M\)](#), [svc.startd\(1M\)](#), [service_bundle\(4\)](#), [attributes\(5\)](#), [fnmatch\(5\)](#), [smf\(5\)](#), [smf_method\(5\)](#), [smf_security\(5\)](#)

名前	svcs – サービスのステータスを報告
形式	<pre>svcs [-aHpv?] [-o col[,col]]... [-R FMRI-instance]... [-sS col]... [FMRI pattern]... svcs {-d -D} [-Hpv?] [-o col[,col]]... [-sS col]... [FMRI pattern] ... svcs -n [FMRI] ... svcs -l [-v] [FMRI pattern]... svcs -x [-v] [FMRI]...</pre>
機能説明	svcs コマンドは、サービス設定リポジトリに記録されているサービスインスタンスに関する情報を表示します。 このコマンドの1番目の形式では、引数に指定したサービスインスタンスのステータスが、1行のリストとして出力されます。各インスタンスは1回だけ出力されます。引数を指定しない場合は、有効になっているすべてのサービスインスタンス(一時的に無効になっているものも含む)が、あとに示すように列のリストとして出力されます。 2番目の形式では、引数に指定したサービスインスタンスの依存関係(依存しているサービス)のステータスが、1行のリストとして出力されます。 3番目の形式では、特定のサービスおよびインスタンスに関する詳細情報が出力されます。 4番目の形式では、サービスインスタンスの状態の説明が出力されます。指定した引数ごとに、サービスの状態およびその状態になった理由の説明が、人間が読み取れる形式のテキストで表示されます。引数を指定しない場合は、問題の発生したサービスの説明が表示されます。 エラーメッセージが標準エラー 스트リームに出力されます。 このコマンドの出力は、svcadm(1M) コマンドへの適切な入力として使用できます。 サポートしているオプションは、次のとおりです。 -? 列指定子を含めて、使用状況のメッセージを詳細に表示します。 -a 無効なサービスや不完全なサービスを含め、すべてのサービスを表示します。不完全なサービスについては、<code>svcs -x<service></code> を使用してさらに詳細を表示できます。 サービスが選択されている場合は、このオプションの効果はありません。

- d
指定されたサービスインスタンスが依存するサービスまたはサービスインスタンスの一覧を表示します。
- D
指定されたサービスまたはサービスインスタンスに依存するサービスインスタンスの一覧を表示します。
- H
列ヘッダーを省略します。
- l
(小文字のエル)選択されたサービスおよびサービスインスタンスについて、表示可能な情報をすべて表示します。各行に1つのサービス属性が表示されます。異なるインスタンスの情報は、空白行で区切られます。

次の属性について個別に説明します。

dependency

依存関係に関する情報。グループ化および `restart_on` プロパティーが最初に表示されます。プロパティーの間はスラッシュ (/) で区切られます。次に、各エンティティーとその状態が表示されます。状態の詳細については、[smf\(5\)](#) を参照してください。標準の状態のほかに、各サービスの依存関係に次の状態説明が表示される場合があります。

absent

このサービスはシステムに定義されていません。

invalid

障害管理リソース識別子 (FMRI) は無効です ([smf\(5\)](#) を参照)。

multiple

このエンティティーは、複数のインスタンスを持つサービスです。

ファイルの依存関係には、次のいずれかの状態説明のみ表示されます。

absent

このファイルはシステムに存在しません。

online

このファイルは存在します。

最後に `svc.startd` がサービスの依存関係を評価したときにファイルが存在しなかった場合には、依存関係が満たされていないものとみなされることがあります。`svcadm refresh` は、強制的に依存関係を再評価します。

unknown

`ENOENT` 以外の理由で、[stat\(2\)](#) が失敗しました。

依存関係、グループ化、および `restart_on` の値の詳細については、[smf\(5\)](#) を参照してください。

enabled

このサービスが有効であるかどうか、および一時的に (次回のシステムリブートまで) 有効であるか無効であるかの情報。サービスが有効かどうかは、true または false によって示されます。一時的に有効かどうかは、(temporary) が存在するかどうかで示されます。

管理者が `svcadm disable -t` を実行した場合、`svcadm milestone` を使用した場合、または特定のタイミングにシステムをブートした場合に、サービスが一時的に無効になることがあります。詳細については、[svcadm\(1M\)](#) を参照してください。

-n

通知パラメータを出力します。[smf\(5\)](#) を参照してください。これは、選択された FMRI またはパターンに関係なく、FMA イベント通知パラメータとシステム全体の SMF 状態遷移通知パラメータを常に出力します。

-o *col[,col]...*

指定された列を出力します。各 *col* は列名でなければなりません。指定できる列については、後述の「列」を参照してください。

-p

各サービスインスタンスに関連付けられているプロセスの一覧を表示します。サービスインスタンスによっては、プロセスが関連付けられていないことがあります。プロセス ID、開始時間、およびコマンド名 ([ps\(1\)](#) の PID、STIME、および CMD フィールド) がプロセスごとに表示されます。

-R *FMRI-instance*

指定されたサービスインスタンスをリスタータとして使用するサービスインスタンスを選択します。

-s *col*

出力を列でソートします。*col* は列名でなければなりません。指定できる列については、後述の「列」を参照してください。複数の -s オプションが指定された場合は、指定された順番にソートします。

-S *col*

オプション -s と反対の順序に、*col* でソートします。

-v

-x または -l が指定されていない場合は、次の冗長列を表示します：
STATE、NSTATE、STIME、CTID、および FMRI。

-x が指定されている場合は、各説明に関する追加情報を表示します。

-l が指定されている場合は、タイプ *application* のプロパティグループのユーザーが表示可能なプロパティとその説明を表示します。

-x

サービスの状態の説明を表示します。

引数が指定されていない場合の `-x` オプションは、次のようなサービスの状態説明を表示します。

- 有効になっているが、実行されていない。
- 別の有効なサービスが動作しているので、実行できない。

オペランド

次のオペランドがサポートされています。

FMRI

1 つまたは複数のインスタンスを指定する障害管理リソース識別子 (FMRI) ([smf\(5\)](#) を参照)。FMRI は、インスタンス名、またはサービス名の最後の部分を指定する方法で、省略して入力することができます。たとえば、次の FMRI があります。

```
svc:/network/smtp:sendmail
```

次のような省略が有効です。

```
sendmail
:sendmail
smtp
smtp:sendmail
network/smtp
```

次のような省略は無効です。

```
mail
network
network/smt
```

FMRI にサービスを指定した場合は、`-D` オプションと一緒に使用している場合を除いて、そのサービスのすべてのインスタンスにコマンドが適用されます。

FMRI の省略形は不安定なので、スクリプトやその他の長期に渡って使用するツールには使用しないでください。

pattern

[fnmatch\(5\)](#) で説明されている展開規則に従ってサービスインスタンスの FMRI と照合されるパターン。このパターンが「`svc:`」で始まっていない場合は、「`svc:/`」が前に付加されます。glob パターンの一般的な例を次に示します。

```
qexample% svcs \*keyserver\*
STATE      STIME      FMRI
disabled   Aug_02     svc:/network/rpc/keyserver:default
```

FMRI-instance

あるインスタンスを指定する FMRI。

列

列名の大文字と小文字は区別されません。デフォルトの出力形式は「-o state,time,fmri」です。デフォルトのソート列はSTATE、STIME、FMRIです。

CTID

サービスインスタンスのプライマリ契約ID。すべてのインスタンスに有効なプライマリ契約IDが割り当てられているわけではありません。

DESC

テンプレート要素から渡された、サービスの簡単な説明。説明が渡されていないサービスの場合は、値が空であることを示すために、ハイフン(-)が使用されます。

FMRI

サービスインスタンスのFMRI。

INST

サービスインスタンスのインスタンス名。

NSTA

サービスインスタンスの次の状態(省略名)。STA列の説明を参照してください。ハイフンは、インスタンスが移行していないことを示します。それ以外はSTAと同じです。

NSTATE

サービスの次の状態。ハイフンは、インスタンスが移行していないことを示します。それ以外はSTATEと同じです。

SCOPE

サービスインスタンスの有効範囲名。

SVC

サービスインスタンスのサービス名。

STA

サービスインスタンスの状態(省略名)([smf\(5\)](#)を参照)。

DGD

機能低下

DIS

無効

LRC

レガシー rc*.d スクリプト主導インスタンス

MNT

保守

OFF

オフライン

ON
online

UN
未初期化

状態が存在しないか認識されない場合は、疑問符(?)が表示されます。移行中のインスタンスにはアスタリスク(*)が付加されます。ただし、NSTA 列または NSTATE 列と一緒に表示されている場合には付加されません。

サービスの状態の説明については、smf(5)を参照してください。

STATE
サービスインスタンスの状態。移行中のインスタンスには、アスタリスクが付加されます。ただし、NSTA 列または NSTATE 列と一緒に表示されている場合には付加されません。

サービスの状態の説明については、smf(5)を参照してください。

STIME
直前の 24 時間の間にサービスインスタンスが現在の状態になった場合は、この列にその時間が表示されます。それ以外の場合は、その日付が表示されます。空白の場所は下線(_)に変換されます。

使用例

例1 デフォルトの出力を表示する
この例では、デフォルトの出力が表示されます。

```
example% svcs
STATE      STIME      FMRI
...
legacy_run 13:25:04 lrc:/etc/rc3_d/S42myscript
...
online     13:21:50 svc:/system/svc/restarter:default
...
online     13:25:03 svc:/milestone/multi-user:default
...
online     13:25:07 svc:/milestone/multi-user-server:default
...
```

例2 すべてのローカルインスタンスの一覧を表示する
この例では、service1 サービスのすべてのローカルインスタンスの一覧が表示されます。

```
example% svcs -o state,nstate,fmri service1
STATE      NSTATE      FMRI
online     -           svc:/service1:instance1
disabled   -           svc:/service1:instance2
```


例3 冗長情報を表示する

この例では、冗長情報が表示されます。

```
example% svcs -v network/rpc/rstat:udp
STATE      NSTATE      STIME      CTID      FMRI
online     -           Aug_09     -         svc:/network/rpc/rstat:udp
```

例4 詳細情報を表示する

この例では、system/service3 のすべてのインスタンスに関する詳細情報が表示されます。管理しているリスタータに応じて、追加フィールドが表示されます。

```
example% svcs -l network/rpc/rstat:udp

fmri          svc:/network/rpc/rstat:udp
enabled       true
state         online
next_state    none
restarter     svc:/network/inetd:default
contract_id
dependency    require_all/error svc:/network/rpc/bind (online)
```

例5 プロセスを表示する

```
example% svcs -p sendmail
STATE      STIME      FMRI
online     13:25:13  svc:/network/smtp:sendmail
           13:25:15  100939  sendmail
13:25:15   100940  sendmail
```

例6 svcs -x を使用したサービス状態の説明

(a) この例では、svcs -x を実行することで、無効になっている print/server サービスが、ある2つのサービスが有効なのにオンラインになっていない根本原因であることが明らかになりました。svcs -xv を実行すると、それらのサービスが print/rfc1179 と print/ipp-listener であることがわかります。この状況を解決するには、print/server を有効にするか、あるいは rfc1179 と ipp-listener を無効にします。

```
example% svcs -x
svc:/application/print/server:default (LP print server)
State: disabled since Mon Feb 13 17:56:21 2006
Reason: Disabled by an administrator.
See: http://support.oracle.com/msg/SMF-8000-05
See: lpsched(1M)
Impact: 2 dependent services are not running. (Use -v for list.)
```

(b) この例では、NFS が動作していません。

```
example$ svcs nfs/client
STATE      STIME      FMRI
```

例 6 svcs -x を使用したサービス状態の説明 (続き)

```
offline          16:03:23 svc:/network/nfs/client:default
```

(c) 次の例から、問題は `nfs/status` にあることがわかります。 `nfs/client` が待機中になっている理由は、 `nfs/status` に依存する `nfs/nlockmgr` に、自身が依存しているからです。

```
example$ svcs -xv nfs/client
svc:/network/nfs/client:default (NFS client)
  State: offline since Mon Feb 27 16:03:23 2006
Reason: Service svc:/network/nfs/status:default
        is not running because a method failed repeatedly.
  See: http://support.oracle.com/msg/SMF-8000-GE
  Path: svc:/network/nfs/client:default
        svc:/network/nfs/nlockmgr:default
        svc:/network/nfs/status:default
  See: man -M /usr/share/man -s 1M mount_nfs
  See: /var/svc/log/network-nfs-client:default.log
Impact: This service is not running.
```

終了ステータス 次の終了ステータスが返されます。

- 0 コマンド呼び出しに成功しました。
- 1 致命的エラーが発生したことを表します。
- 2 無効なコマンド行オプションが指定された。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	下記を参照。

画面出力は「不確実」です。呼び出しは「確実」です。

関連項目 [ps\(1\)](#), [svcprop\(1\)](#), [svcadm\(1M\)](#), [svccfg\(1M\)](#), [svc.startd\(1M\)](#), [stat\(2\)](#), [libscf\(3LIB\)](#), [attributes\(5\)](#), [fnmatch\(5\)](#), [smf\(5\)](#)

名前 symorder – シンボルリストの並べ替え

形式 symorder [-s] *objectfile symbolfile*

機能説明 symorder は、SunOS 4.x で特に vmunix からシンボルを取得するオーバーヘッドを削減するために使用されていました。カーネルシンボルのエントリは /dev/ksyms から動的に取得されるため、これが適用されなくなりました。

 このスクリプトは、さまざまなオペレーティングシステム間で移植可能なスクリプトを保守する必要があるソフトウェア開発者に役立つように提供されています。

終了ステータス symorder の終了ステータスは 0 です。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities

関連項目 [nlist\(3ELF\)](#), [attributes\(5\)](#), [ksyms\(7D\)](#).

名前	sys-suspend – システムの保存停止またはシャットダウンと電源の切断
形式	<code>/usr/bin/sys-suspend [-fnxh][<i>-d displayname</i>]</code>
機能説明	<code>sys-suspend</code> には、システム全体を保存停止またはシャットダウンするためのオプションが用意されています。 システムを保存停止すると、電源を節約したり、システムに転送の準備をしたりできます。ハードウェアの再構成または交換を行う際には、保存停止を使用してはいけません。 保存停止では、メモリーへの電力供給を維持するか (Suspend to RAM)、または状態を不揮発性記憶領域に保存する (Suspend to Disk) ことによって、電源投入または復帰イベントによって復元再開操作が実行されるまで、現在のシステム状態が保持されます。 ウィンドウ環境で復元再開を行うと、承認されたユーザーのみがシステムにアクセスできるようにロック画面が表示されます。ウィンドウ以外の環境では、ユーザーはパスワードを求められます。 デバイスまたはプロセスが重要な操作やタイミングの影響を受ける操作 (リアルタイムの操作など) を実行しているときは、システムの保存停止に失敗する可能性があります。これが発生すると、システムは現在の実行状態のままになります。コンソールまたはシステムログに、失敗を報告するメッセージが表示されます。いったんシステムが正常に保存停止されると、ハードウェアの再構成など外部からの影響がないかぎり、復元再開操作は必ず成功します。 シャットダウンの場合、システムは <code>poweroff(1M)</code> を実行したときと同様に応答します。 このコマンドによって、<code>solaris.system.power.suspend</code> 承認が強制されます。デフォルトのインストールでは、これらの承認はコンソールユーザーに関連付けられています。その他のユーザーは、これらの承認を追加するか、保存停止プロファイルを追加する必要があります。
オプション	次のオペランドがサポートされています。 <code>-d displayname</code> <code>displayname</code> で指定された X サーバーに接続します。 <code>-f</code> 強制的に保存停止します。保存停止に失敗した場合、<code>poweroff(1M)</code> が発生します。システム状態は保存されず、通常のブートに進みます。 <code>-h</code> デフォルトを保存停止からシャットダウンに変更します。 <code>-n</code> メッセージを表示しないか、またはユーザーの介入を要求しません。 <code>-x</code> 「ロック画面」を無効にします。このフラグにより、復元再開時の「ロック画面」の実行が無効になります。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/kernel/power
インタフェースの安定性	確実

関連項目 [halt\(1M\)](#), [poweroff\(1M\)](#), [shutdown\(1M\)](#), [attributes\(5\)](#), [cpr\(7\)](#)

名前	sysV-make – maintain, update, and regenerate groups of programs
形式	<code>/usr/lib/svr4.make [-f <i>makefile</i>] [-eiknpqrst] [<i>names</i>]</code>
機能説明	This is the <i>vanilla</i> System V version of make. If the environment variable <code>USE_SVR4_MAKE</code> is set, then the command <code>make</code> will invoke this version of make. (See also the <code>ENVIRONMENT</code> section.) <code>make</code> allows the programmer to maintain, update, and regenerate groups of computer programs. <code>make</code> executes commands in <i>makefile</i> to update one or more target <i>names</i> (<i>names</i> are typically programs). If the <code>-f</code> option is not present, then <code>makefile</code>, <code>Makefile</code>, and the Source Code Control System (SCCS) files <code>s.makefile</code> and <code>s.Makefile</code> are tried in order. If <i>makefile</i> is <code>'-'</code> the standard input is taken. More than one <code>-f <i>makefile</i></code> argument pair may appear. <code>make</code> updates a target only if its dependents are newer than the target. All prerequisite files of a target are added recursively to the list of targets. Missing files are deemed to be outdated. The following list of four directives can be included in <i>makefile</i> to extend the options provided by <code>make</code>. They are used in <i>makefile</i> as if they were targets: <code>.DEFAULT:</code> If a file must be made but there are no explicit commands or relevant built-in rules, the commands associated with the name <code>.DEFAULT</code> are used if it exists. <code>.IGNORE:</code> Same effect as the <code>-i</code> option. <code>.PRECIOUS:</code> Dependents of the <code>.PRECIOUS</code> entry will not be removed when quit or interrupt are hit. <code>.SILENT:</code> Same effect as the <code>-s</code> option.
Options	The options for <code>make</code> are listed below: <code>-e</code> Environment variables override assignments within makefiles. <code>-f <i>makefile</i></code> Description filename (<i>makefile</i> is assumed to be the name of a description file). <code>-i</code> Ignore error codes returned by invoked commands. <code>-k</code> Abandon work on the current entry if it fails, but continue on other branches that do not depend on that entry. <code>-n</code> No execute mode. Print commands, but do not execute them. Even command lines beginning with an <code>'@'</code> are printed. <code>-p</code> Print out the complete set of macro definitions and target descriptions. <code>-q</code> Question. <code>make</code> returns a zero or non-zero status code depending on whether or not the target file has been updated. <code>-r</code> Do not use the built-in rules. <code>-s</code> Silent mode. Do not print command lines before executing.

-t Touch the target files (causing them to be updated) rather than issue the usual commands.

Creating the makefile The makefile invoked with the -f option is a carefully structured file of explicit instructions for updating and regenerating programs, and contains a sequence of entries that specify dependencies. The first line of an entry is a blank-separated, non-null list of targets, then a ':', then a (possibly null) list of prerequisite files or dependencies. Text following a ';' and all following lines that begin with a tab are shell commands to be executed to update the target. The first non-empty line that does not begin with a tab or '#' begins a new dependency or macro definition. Shell commands may be continued across lines with a backslash-new-line (\-NEWLINE) sequence. Everything printed by make (except the initial TAB) is passed directly to the shell as is. Thus,

```
echo a\  
b
```

will produce

```
ab
```

exactly the same as the shell would.

Number-sign (#) and NEWLINE surround comments including contained '\-NEWLINE' sequences.

The following makefile says that `pgm` depends on two files `a.o` and `b.o`, and that they in turn depend on their corresponding source files (`a.c` and `b.c`) and a common file `incl.h`:

```
pgm: a.o b.o  
    cc a.o b.o -o pgm  
a.o: incl.h a.c  
    cc -c a.c  
b.o: incl.h b.c  
    cc -c b.c
```

Command lines are executed one at a time, each by its own shell. The `SHELL` environment variable can be used to specify which shell make should use to execute commands. The default is `/usr/bin/sh`. The first one or two characters in a command can be the following: '@', '-', '@-', or '-@'. If '@' is present, printing of the command is suppressed. If '-' is present, make ignores an error. A line is printed when it is executed unless the -s option is present, or the entry `.SILENT` is included in *makefile*, or unless the initial character sequence contains a @. The -n option specifies printing without execution; however, if the command line has the string \$(MAKE) in it, the line is always executed (see the discussion of the MAKEFLAGS macro in the make Environment sub-section below). The -t (touch) option updates the modified date of a file without executing any commands.

Commands returning non-zero status normally terminate make. If the `-i` option is present, if the entry `.IGNORE:` is included in *makefile*, or if the initial character sequence of the command contains `'-'`, the error is ignored. If the `-k` option is present, work is abandoned on the current entry, but continues on other branches that do not depend on that entry.

Interrupt and quit cause the target to be deleted unless the target is a dependent of the directive `.PRECIOUS`.

make Environment

The environment is read by make. All variables are assumed to be macro definitions and are processed as such. The environment variables are processed before any makefile and after the internal rules; thus, macro assignments in a makefile override environment variables. The `-e` option causes the environment to override the macro assignments in a makefile. Suffixes and their associated rules in the makefile will override any identical suffixes in the built-in rules.

The `MAKEFLAGS` environment variable is processed by make as containing any legal input option (except `-f` and `-p`) defined for the command line. Further, upon invocation, make “invents” the variable if it is not in the environment, puts the current options into it, and passes it on to invocations of commands. Thus, `MAKEFLAGS` always contains the current input options. This feature proves very useful for “super-makes”. In fact, as noted above, when the `-n` option is used, the command `$(MAKE)` is executed anyway; hence, one can perform a `make -n` recursively on a whole software system to see what would have been executed. This result is possible because the `-n` is put in `MAKEFLAGS` and passed to further invocations of `$(MAKE)`. This usage is one way of debugging all of the makefiles for a software project without actually doing anything.

Include Files

If the string *include* appears as the first seven letters of a line in a *makefile*, and is followed by a blank or a tab, the rest of the line is assumed to be a filename and will be read by the current invocation, after substituting for any macros.

Macros

Entries of the form *string1* = *string2* are macro definitions. *string2* is defined as all characters up to a comment character or an unescaped NEWLINE. Subsequent appearances of `$(string1[:subst1=[subst2]])` are replaced by *string2*. The parentheses are optional if a single-character macro name is used and there is no substitute sequence. The optional `:subst1=subst2` is a substitute sequence. If it is specified, all non-overlapping occurrences of *subst1* in the named macro are replaced by *subst2*. Strings (for the purposes of this type of substitution) are delimited by BLANKs, TABs, NEWLINE characters, and beginnings of lines. An example of the use of the substitute sequence is shown in the `Libraries` sub-section below.

Internal Macros

There are five internally maintained macros that are useful for writing rules for building targets.

`$(*)` The macro `$(*)` stands for the filename part of the current dependent with the suffix deleted. It is evaluated only for inference rules.

- \$@ The \$@ macro stands for the full target name of the current target. It is evaluated only for explicitly named dependencies.
- \$< The \$< macro is only evaluated for inference rules or the .DEFAULT rule. It is the module that is outdated with respect to the target (the “manufactured” dependent file name). Thus, in the .c.o rule, the \$< macro would evaluate to the .c file. An example for making optimized .o files from .c files is:

```
.c.o:
    cc c O $*.c
```

or:

```
.c.o:
    cc c O $<
```
- \$? The \$? macro is evaluated when explicit rules from the makefile are evaluated. It is the list of prerequisites that are outdated with respect to the target, and essentially those modules that must be rebuilt.
- % The % macro is only evaluated when the target is an archive library member of the form lib(file.o). In this case, \$@ evaluates to lib and % evaluates to the library member, file.o.

Four of the five macros can have alternative forms. When an upper case D or F is appended to any of the four macros, the meaning is changed to “directory part” for D and “file part” for F. Thus, \$(@D) refers to the directory part of the string \$@. If there is no directory part, ./ is generated. The only macro excluded from this alternative form is \$?.

Suffixes

Certain names (for instance, those ending with .o) have inferable prerequisites such as .c, .s, etc. If no update commands for such a file appear in *makefile*, and if an inferable prerequisite exists, that prerequisite is compiled to make the target. In this case, make has inference rules that allow building files from other files by examining the suffixes and determining an appropriate inference rule to use. The current default inference rules are:

.c	.c~	.f	.f~	.s	.s~	.sh	.sh~	.C	.C~
.c.a	.c.o	.c~.a	.c~.c	.c~.o	.f.a	.f.o	.f~.a	.f~.f	.f~.o
.h~.h	.l.c	.l.o	.l~.c	.l~.l	.l~.o	.s.a	.s.o	.s~.a	.s~.o
.s~.s	.sh~.sh	.y.c	.y.o	.y~.c	.y~.o	.y~.y	.C.a	.C.o	.C~.a
.C~.C	.C~.o	.L.C	.L.o	.L~.C	.L~.L	.L~.o	.Y.C	.Y.o	.Y~.C
.Y~.o	.Y~.Y								

The internal rules for make are contained in the source file `make.rules` for the make program. These rules can be locally modified. To print out the rules compiled into the make on any machine in a form suitable for re-compilation, the following command is used:

```
make -pf -2>/dev/null < /dev/null
```

A tilde in the above rules refers to an SCCS file (see [sccsfile\(4\)](#)). Thus, the rule `.c~.o` would transform an SCCS C source file into an object file (`.o`). Because the `s.` of the SCCS files is a prefix, it is incompatible with the make suffix point of view. Hence, the tilde is a way of changing any file reference into an SCCS file reference.

A rule with only one suffix (for example, `.c:`) is the definition of how to build `x` from `x.c`. In effect, the other suffix is null. This feature is useful for building targets from only one source file, for example, shell procedures and simple C programs.

Additional suffixes are given as the dependency list for `.SUFFIXES`. Order is significant: the first possible name for which both a file and a rule exist is inferred as a prerequisite. The default list is:

```
.SUFFIXES: .o .c .c~ .y .y~ .l .l~ .s .s~ .sh .sh~ .h .h~ .f .f~ .C .C~ .Y .Y~ .L .L~
```

Here again, the above command for printing the internal rules will display the list of suffixes implemented on the current machine. Multiple suffix lists accumulate; `.SUFFIXES:` with no dependencies clears the list of suffixes.

Inference Rules

The first example can be done more briefly.

```
pgm: a.o b.o
    cc a.o b.o o pgm
a.o b.o: incl.h
```

This abbreviation is possible because make has a set of internal rules for building files. The user may add rules to this list by simply putting them in the *makefile*.

Certain macros are used by the default inference rules to permit the inclusion of optional matter in any resulting commands. Again, the previous method for examining the current rules is recommended.

The inference of prerequisites can be controlled. The rule to create a file with suffix `.o` from a file with suffix `.c` is specified as an entry with `.c.o:` as the target and no dependents. Shell commands associated with the target define the rule for making a `.o` file from a `.c` file. Any target that has no slashes in it and starts with a dot is identified as a rule and not a true target.

Libraries

If a target or dependency name contains parentheses, it is assumed to be an archive library, the string within parentheses referring to a member within the library. Thus, `lib(file.o)` and `$(LIB)(file.o)` both refer to an archive library that contains `file.o`. (This example assumes the `LIB` macro has been previously defined.) The expression `$(LIB)(file1.o file2.o)` is not

legal. Rules pertaining to archive libraries have the form `.XX.a` where the `XX` is the suffix from which the archive member is to be made. An unfortunate by-product of the current implementation requires the `XX` to be different from the suffix of the archive member. Thus, one cannot have `lib(file.o)` depend upon `file.o` explicitly. The most common use of the archive interface follows. Here, we assume the source files are all C type source:

```
lib:    lib(file1.o) lib(file2.o) lib(file3.o)
        @echo lib is now up-to-date
.c.a:
        $(CC) -c $(CFLAGS) $<
        $(AR) $(ARFLAGS) $@ $*.o
        rm -f $*.o
```

In fact, the `.c.a` rule listed above is built into `make` and is unnecessary in this example. A more interesting, but more limited example of an archive library maintenance construction follows:

```
lib:    lib(file1.o) lib(file2.o) lib(file3.o)
        $(CC) -c $(CFLAGS) $(?:.o=.c)
        $(AR) $(ARFLAGS) lib $?
        rm $?
        @echo lib is now up-to-date
.c.a:;
```

Here the substitution mode of the macro expansions is used. The `$?` list is defined to be the set of object filenames (inside `lib`) whose C source files are outdated. The substitution mode translates the `.o` to `.c`. (Unfortunately, one cannot as yet transform to `.c~`; however, this transformation may become possible in the future.) Also note the disabling of the `.c.a:` rule, which would have created each object file, one by one. This particular construct speeds up archive library maintenance considerably. This type of construct becomes very cumbersome if the archive library contains a mix of assembly programs and C programs.

環境変数

USE_SVR4_MAKE If this environment variable is set, then the `make` command will invoke this System V version of `make`. If this variable is not set, then the default version of [make\(1S\)](#) is invoked.

`USE_SVR4_MAKE` can be set as follows (Bourne shell):

```
$ USE_SVR4_MAKE=""; export USE_SVR4_MAKE
```

or (C shell):

```
% setenv USE_SVR4_MAKE
```

ファイル

<code>[Mm]akefile</code>	
<code>s.[Mm]akefile</code>	default makefiles
<code>/usr/bin/sh</code>	default shell for make
<code>/usr/share/lib/make/make.rules</code>	default rules for make

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

関連項目 [cd\(1\)](#), [make\(1S\)](#), [sh\(1\)](#), [printf\(3C\)](#), [sccsfile\(4\)](#), [attributes\(5\)](#)

注意事項 Some commands return non-zero status inappropriately; use `-i` or the `'-'` command line prefix to overcome the difficulty.

Filenames containing the characters `=`, `:`, and `@` do not work. Commands that are directly executed by the shell, notably [cd\(1\)](#), are ineffectual across NEWLINEs in make. The syntax `lib(file1.o file2.o file3.o)` is illegal. You cannot build `lib(file.o)` from `file.o`.

名前	tabs – 端末へのタブ設定
形式	<pre> tabs [-n —file [[-code] -a -a2 -c -c2 -c3 -f -p -s -u]] q!! [+m [n]] [-T tabs [-T type] [+ m [n]] n1 [, n2 ,...]</pre>
機能説明	tabs は、以前のタブ設定をクリアした後、タブ仕様に従って、ユーザー端末にタブストップを設定します。ユーザー端末には、リモートで設定可能なハードウェアタブがなければなりません。
オプション	以下のオプションがサポートされています。同じフラグを複数回記述した場合には、最後に書かれたものが有効です。 -T type tabs は、タブとマージンを設定するために端末のタイプを知らなければなりません。type は term(5) で記述されている名前です。-T フラグが指定されない場合、tabs は環境変数 TERM の値を使用します。TERM の値が NULL のとき、または TERM が環境中に定義されていないときに (environ(5) を参照)、tabs は端末タイプとして <code>ansi+tabs</code> を使用し、多くの端末で作動するシーケンスを提供します。 +m[n] いくつかの端末に対しては、マージン引数を使用することもあります。これを用いると、カラム <i>n</i>+1 を左マージンにすることによって、すべてのタブが <i>n</i> カラム移動します。<i>n</i> の値なしで +m が指定された場合、値は 10 と見なされます。TermiNet では、タブリストの最初の値は 1 でなければなりません。さもなければ、マージンはさらに右に移動します。ほとんどの端末上にある通常 (左端) のマージンは、+m0 によって得られます。ほとんどの端末のマージンは、+m フラグが明示的に指定されたときにだけ、リセットされます。
タブ仕様	4 種類のタブ仕様が認められています。それらは次のとおりです。内蔵 (Canned -code)、反復 (-n)、任意 (n1,n2,...)、およびファイル (—file) の 4 つです。 タブ仕様が指定されない場合、デフォルト値は -8、すなわち UNIX システムの「標準」タブです。カラム番号の最小値は 1 です。注意: tabs の場合、カラム 1 は常に端末上のいちばん左のカラムを指します。これは、DASI 300、DASI 300s、および DASI 450 などのようにカラムマーカが 0 で始まる場合でも同様です。
内蔵-	内蔵されたタブの設定を選択するには下記のコードの 1 つを使用してください。複数個指定すると、最後に指定したものだけが有効となります。指定可能なコードとその意味を次に示します。 <pre> -a 1,10,16,36,72 アセンブラ IBM S/370 の第 1 フォーマット -a2 1,10,16,40,72 アセンブラ IBM S/370 の第 2 フォーマット -c 1,8,12,16,20,55</pre>

COBOL 標準フォーマット

-c2 1,6,10,14,49

COBOL 圧縮フォーマット (カラム 1-6 を省略)。このコードを使用すると、最初に入力された文字はカードカラム 7 に対応し、スペースを 1 個入れるとカラム 8 となり、タブ文字を入力するとカラム 12 となります。このタブ設定を使用するファイルには、次のようなフォーマット仕様があります ([fspec\(4\)](#) を参照)。

```
<:t-c2 m6 s66 d:>
```

-c3 1,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62,67

-c2 より多いタブを伴う COBOL 圧縮フォーマット (カラム 1-6 を省略)。これは COBOL で推奨されるフォーマットです。適切なフォーマット仕様は次のとおりです ([fspec\(4\)](#) を参照)。

```
<:t-c3 m6 s66 d:>
```

-f 1,7,11,15,19,23

FORTRAN

-p 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61

PL/I

-s 1,10,55

SNOBOL

-u 1,12,20,44

UNIVAC 1100 アセンブラ

反復 -n 反復仕様は、カラム $1+n$ 、 $1+2*n$ などにタブを要求します。 n は 1 桁の 10 進数です。特に重要なのは、値 8 です。これは UNIX システムの「標準」タブ設定を表し、端末でもっとも多くみられるタブ設定です。値 -0 を指定すると、タブはクリアされ新たな設定は行われません。

任意 「オペランド」の節を参照のこと。

File -file *file* *file* の名前が指定されると、tabs はフォーマット仕様を検索しながら、そのファイルの先頭行を読み取ります ([fspec\(4\)](#) を参照)。先頭行にフォーマット仕様があった場合、その仕様に従ってタブストップを設定します。そうでない場合、これはタブを -8 に設定します。この種の仕様は、タブ設定されたファイルが正しいタブ設定で出力されているかを確認する際に用いられます。また pr コマンドとともに用いることもできます。

```
example% tabs -file; pr file
```

タブおよびマージンの設定は、標準出力を介して行われます。

オペランド

次のオペランドを指定できます。

`n1[,n2,...]` 任意フォーマットは、コンマまたはスペースで区切られた一連のタブストップ位置指定で構成されます。タブストップ位置は、10進整数を昇順で指定しなければなりません。最大40個まで指定可能です。数値(最初のを除く)の前に正の符号が付いている場合、これは直前の値からの増分と見なされます。したがって、フォーマット `1,10,20,30` と `1,10,+10,+10` は同じ意味となります。

使用例

例1 tabs コマンドの使用例

次の例では、`-code` (内蔵仕様) を用いて IBM アセンブラが要求する設定、すなわちカラム 1, 10, 16, 36, 72 にタブを設定します。

```
example% tabs -a
```

次の例では、`-n` (反復仕様) (n は 8) を用いる使用例で、これにより、タブは 8 カラムごとに設定されます。1+(1*8), 1+(2*8), ... すなわち、カラム 9, 17, ...:

```
example% tabs -8
```

次では、`n1,n2,...` (任意仕様) を用いた使用例で、カラム 1、8、および 36 にタブを設定します。

```
example% tabs 1,8,36
```

次の例では `-file` (ファイル仕様) を用いた使用例で、タブは `$HOME/fspec.list/att4425` の先頭行に従って設定されることを示しています ([fspec\(4\)](#) を参照)。

```
example% tabs -$HOME/fspec.list/att4425
```

環境変数

`tabs` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

TERM 端末のタイプを決定します。この変数が設定されていないかあるいは NULL に設定されていて、`-T` オプションが省略された場合、端末タイプとして `ansi+tabs` が用いられます。

終了ステータス

次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目
- [expand\(1\)](#), [newform\(1\)](#), [pr\(1\)](#), [stty\(1\)](#), [tput\(1\)](#), [fspec\(4\)](#), [terminfo\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [term\(5\)](#), [standards\(5\)](#)
- 注意事項
- タブの解除および左マージンの設定方法は端末によってそれぞれ異なります。

`tabs` は (長いシーケンスを要求する端末では) 20 タブしか解除しませんが、64 タブまで設定できます。

`tabs` コマンドとともに使用される `tabspec` は、`newform` コマンドとともに使用されるものとは異なります。たとえば `tabs` においては、`tabs -8` がタブを 8 カラムごとに設定します。一方 `newform` においては、`newform -i-8` がタブを 8 カラムごとに設定することを示します。

名前	tail – ファイルの最後の部分の出力
形式	<pre> /usr/bin/tail [$\pm$s number [lbcrl]] [file] /usr/bin/tail [-lbcrl] [file] /usr/bin/tail [$\pm$ number [lbcfl]] [file] /usr/bin/tail [-lbcfl] [file] /usr/xpg4/bin/tail [-f -r] [-c number -n number] [file] /usr/xpg4/bin/tail [$\pm$ number [l b c] [f]] [file] /usr/xpg4/bin/tail [$\pm$ number [l]] [f r]] [file] </pre>
機能説明	tail ユーティリティーは、指定されたファイルの内容のうち、指定地点から終わりまでを標準出力にコピーします。ファイルの指定がない場合は、標準入力を使用します。 コピーの開始地点は、<code>-cnumber</code>、<code>-nnumber</code>、<code>$\pm$number</code> の各オプションで指定されます。<code>+number</code> は先頭からの距離、<code>-number</code> は末尾からの距離を表します。<code>number</code> が NULL のときは 10 と見なされます。<code>number</code> の単位は <code>-c</code> または <code>-n</code> オプションの指定に従い、行またはバイトとなり、さらに <code>l</code>、<code>b</code>、<code>c</code> のいずれかのオプションが指定された場合には、それに従って、行、ブロック、バイトのいずれかとなります。単位指定がすべて省略された場合には、行と見なされます。
オプション	次のオプションは、<code>/usr/bin/tail</code> と <code>/usr/xpg4/bin/tail</code> で指定できます。<code>-r</code> と <code>-f</code> オプションを同時に指定することはできません。両方がコマンド行に指定された場合、<code>-f</code> オプションは無効になります。 <code>-b</code> ブロック単位。 <code>-c</code> バイト単位。 <code>-f</code> 継続。入力ファイルがパイプでない場合、プログラムは入力ファイルの行がコピーされた後に終了せず、無限ループに入り、その中で 1 秒間休止し、入力ファイルからさらにレコードを読み取ったり、コピーしようとします。したがって、このオプションは、ほかのプロセスによって作成中のファイルの成長を監視するために使用できます。 <code>-l</code> 行単位。 <code>-r</code> 逆順。ファイルに指定された開始点から行を逆の順番にコピーします。<code>r</code> のデフォルトは、ファイル全体を逆順に出力します。 <code>/usr/xpg4/bin/tail</code> 次のオプションは、<code>/usr/xpg4/bin/tail</code> でのみ指定できます。 <code>-c number</code> <code>number</code> オプションの引数は、ファイル内のコピーの開始位置をバイト単位で表す 10 進整数です。符号もファイル内の位置に影響しません。

- + ファイルの先頭からの相対位置を指定してコピーを開始します。
- ファイルの末尾からの相対位置を指定してコピーを開始します。
- none ファイルの末尾からの相対位置を指定してコピーを開始します。

カウントは1から始まります。つまり `-c+1` はファイルの第1バイト目を表し、`-c-1` は最後のバイトを表します。

`-n number` ファイル内の開始位置がバイト単位でなく行単位で計測される点を除き、`-cnumber` と同等の意味です。カウントは1から始まります。つまり `-n+1` はファイルの第1行目を表し、`-n-1` は最後の行を表します。

オペランド 次のオペランドを指定できます。

file 入力ファイルのパス名。*file* オペランドを指定しないと、標準入力が使われます。

使用法 ファイルが2Gバイト(2³¹バイト)以上ある場合の `tail` の動作については、[largefile\(5\)](#) を参照してください。

使用例 例1 `tail` コマンドの使用

次のコマンドは、ファイル `fred` の最後の10行を出力し、その後には `tail` を開始してから終了するまでの間に `fred` に追加されたすべての行が続きます。

```
example% tail -f fred
```

次のコマンドは、ファイル `fred` の最後の15バイトを出力し、その後には `tail` を開始してから終了するまでの間に `fred` に追加されたすべての行が続きます。

```
example% tail -15cf fred
```

環境変数 `tail` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/tail

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

/usr/xpg4/bin/tail

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目

[cat\(1\)](#), [head\(1\)](#), [more\(1\)](#), [pg\(1\)](#), [dd\(1M\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項

ファイルの末尾で相対的にパイプされた `tail` はバッファに格納されるので、長さに制限があります。文字型の特殊なファイルでは、さまざまな種類の変則的な動作が起こる可能性があります。

名前 talk – talk to another user

形式 talk *address* [*terminal*]

機能説明 The talk utility is a two-way, screen-oriented communication program.

When first invoked, talk sends a message similar to:

```
Message from TalkDaemon@ her_machine at time...  
talk: connection requested by your_address  
talk: respond with: talk your_address
```

to the specified *address*. At this point, the recipient of the message can reply by typing:

```
talk your_address
```

Once communication is established, the two parties can type simultaneously, with their output displayed in separate regions of the screen. Characters are processed as follows:

- Typing the alert character will alert the recipient's terminal.
- Typing Control-L will cause the sender's screen regions to be refreshed.
- Typing the erase and kill characters will affect the sender's terminal in the manner described by the [termios\(3C\)](#) interface.
- Typing the interrupt or end-of-file (EOF) characters will terminate the local talk utility. Once the talk session has been terminated on one side, the other side of the talk session will be notified that the talk session has been terminated and will be able to do nothing except exit.
- Typing characters from LC_CTYPE classifications print or space will cause those characters to be sent to the recipient's terminal.
- When and only when the stty iexten local mode is enabled, additional special control characters and multi-byte or single-byte characters are processed as printable characters if their wide character equivalents are printable.
- Typing other non-printable characters will cause them to be written to the recipient's terminal as follows: control characters will appear as a caret (^) followed by the appropriate ASCII character, and characters with the high-order bit set will appear in “meta” notation. For example, ‘\003’ is displayed as ‘^C’ and ‘\372’ as ‘M-z’.

Permission to be a recipient of a talk message can be denied or granted by use of the [mesg\(1\)](#) utility. However, a user's privilege may further constrain the domain of accessibility of other users' terminals. Certain commands, such as [pr\(1\)](#), disallow messages in order to prevent interference with their output. talk will fail when the user lacks the appropriate privileges to perform the requested action.

Certain block-mode terminals do not have all the capabilities necessary to support the simultaneous exchange of messages required for talk. When this type of exchange cannot be

supported on such terminals, the implementation may support an exchange with reduced levels of simultaneous interaction or it may report an error describing the terminal-related deficiency.

オペランド

The following operands are supported:

address The recipient of the talk session. One form of *address* is the *username*, as returned by the [who\(1\)](#) utility. If you wish to talk to someone on your own machine, then *username* is just the person's login name. If you wish to talk to a user on another host, then *username* is one of the following forms:

host!user
host.user
host:user
user@host

although *user@host* is perhaps preferred.

terminal If the recipient is logged in more than once, *terminal* can be used to indicate the appropriate terminal name. If *terminal* is not specified, the talk message will be displayed on one or more accessible terminals in use by the recipient. The format of *terminal* will be the same as that returned by [who](#).

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of talk: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

TERM Determine the name of the invoker's terminal type. If this variable is unset or null, an unspecified terminal type will be used.

終了ステータス

The following exit values are returned:

0 Successful completion.

>0 An error occurred, or talk was invoked on a terminal incapable of supporting it.

ファイル

/etc/hosts host name database

/var/adm/utmpx user and accounting information for talk

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/network-servers
Interface Stability	Committed
Standard	See standards(5) .

関連項目	mail(1) , mesg(1) , pr(1) , stty(1) , who(1) , write(1) , talkd(1M) , termios(3C) , attributes(5) , environ(5) , standards(5)
注意事項	Typing Control-L redraws the screen, while the erase, kill, and word kill characters will work in talk as normal. To exit, type an interrupt character. talk then moves the cursor to the bottom of the screen and restores the terminal to its previous state.

名前	tar - テープアーカイブの作成およびファイルの追加または抽出
形式	<pre> tar c[BDFHijlnPTvwzZ@/[0-7]][bf][X...] [blocksize] [tarfile] [size] [exclude-file]... {file -I include-file -C directory file}... tar r[BDFHijlnPTvwzZ@/[0-7]][bf] [blocksize] [tarfile] [size] {file -I include-file -C directory file}... tar t[BFHijlnTvzZ[0-7]][f][X...] [tarfile] [size] [exclude-file]... {file -I include-file}... tar u[BDFHijlnPTvwzZ@/[0-7]][bf] [blocksize] [tarfile] [size] file... tar x[BFHilmnjopTvwzZ@/[0-7]][f][X...] [tarfile] [size] [exclude-file]... [file]...</pre>
機能説明	tar コマンドは、複数のファイルを <i>tarfile</i> と呼ばれる 1 つのファイルに保管したり、抽出したりします。通常、<i>tarfile</i> は磁気テープですが、その他のファイルであってもかまいません。tar の動作は、キー (<i>key</i>) 引数によって制御されます。<i>key</i> は、1 個の機能文字 (c、r、t、u、または x) と、この機能文字に応じた 0 個以上の機能修飾子 (文字または数字) からなる文字列です。<i>key</i> 文字列はスペース文字を含みません。機能修飾子に対する引数は <i>key</i> 文字列で使用した順序でコマンド行に指定します。 -I <i>include-file</i>、-C <i>directory file</i>、<i>file</i> 引数は、どのファイルまたはディレクトリを保管または抽出するかを指定します。いずれの場合も、ディレクトリ名を使用すると、そのディレクトリのすべてのファイルやサブディレクトリを (再帰的に) 参照します。中括弧 ({ }) で囲まれた引数は、そのうちのどれか 1 つの引数を指定することを示します。
オペランド	次のオペランドがサポートされています。 -C <i>directory file</i> <i>directory</i> への <code>chdir</code> (cd(1) を参照) 操作を行なった上、c (作成) 操作または r (置換) 操作を <i>file</i> に対して行います。<i>file</i> には、短い相対パス名を使用します。<i>file</i> が「.」の場合、<i>directory</i> 内のすべてのファイルを保存します。このオペランドを使用すると、共通の親を持たない複数のディレクトリ内にあるファイルを保管できます。 -I <i>include-file</i> 1 行につき 1 つのファイル名からなるファイルの一覧を含んだ <i>include-file</i> をオープンし、各ファイルがコマンド行で指定されたのと同様に処理します。<i>include-file</i> の行末に不要なスペースを置かないように注意してください。また、各行頭にも不要なスペースを置かないように注意してください。組み込むファイルの最初の文字列に対する一致には、改行で分かれている行全体が使用されます。除外されたファイルが存在する場合には (x 機能修飾子を参照)、除外さ

れたファイルは組み込まれたファイルよりも優先されます。あるファイルが *exclude-file* と *include-file* の両方のファイル(またはコマンド行上)で指定されていれば、そのファイルは除外されます。

file

保存(c、r、またはu機能を指定した場合)、抽出(x)、または表示(t)される通常ファイルまたはディレクトリのパス名。*file* がディレクトリのパス名である場合、動作はそのディレクトリ以下のすべてのファイルと(再帰的に)サブディレクトリに適用します。

ファイルがアーカイブで、Eフラグ(「機能修飾子」を参照)が指定されていない場合、ファイル名は256文字を超えることはできません。さらに、親ディレクトリ名との間を分割して考えると、ディレクトリ名部分は155文字、ファイル名部分は100文字を超えることはできません。Eフラグが指定されている場合、PATH_MAXで指定されている文字までファイル名に指定できます。

たとえば、ファイル名部分が100文字を超えるファイルは、Eフラグを指定しないと保存されない可能性があります。ディレクトリ名部分が200文字で、ファイル名部分が50文字であるファイルは、ディレクトリ名の151文字目から156文字目のどこかにスラッシュが1つ入っていれば、Eフラグを指定しなくても保存されます。

機能文字

キーの機能部分は、次の文字のいずれか1つで指定します。

c

作成。tarfileの最初から書き込みます。最後からではありません。

r

置換。指定した*file*をtarfileの最後に関き込みます。拡張ヘッダーを付けて作成したファイルは、拡張ヘッダーを付けて更新する必要があります(「機能修飾子」のEフラグの説明を参照)。拡張ヘッダーなしで作成したファイルは、拡張ヘッダーを付けて修正することはできません。

t

内容の一覧表示。指定されたファイルがtarfileに検出されるごとに、ファイル名が表示されます。*file* 引数を指定していない場合は、tarfileのファイル名全部と関連する拡張属性が表示されます。v機能修飾子とともに指定すると、指定したファイルの追加情報が表示されます。

u

更新。指定された*file*がtarfileにまだ入っていない場合、または最後にtarfileへ書き込まれて以来変更があった場合は、*file*はtarfileの最後に追加されます。更新には時間のかかる場合があります。SunOS 5.xシステム上で作成されたtarfileはSunOS 4.xシステム上では更新できません。拡張ヘッダーを付けて作成したファイルは、拡張ヘッダーを付けて更新する必要があります(「機能修飾子」のEフラグの説明を参照)。拡張ヘッダーなしで作成したファイルは、拡張ヘッダーを付けて修正することはできません。

x

抽出または復元。指定された *file* が *tarfile* から抽出され、(現在のディレクトリからの相対パスで) *tarfile* で指定されたディレクトリに書き込まれます。抽出するファイルとディレクトリには、相対パス名を使用してください。

tar アーカイブに含まれる絶対パス名を復元するときには、絶対パス名を使用します。つまり、先頭のスラッシュ (/) は取り除かれません。

デフォルトでは、絶対パス名 (/ 文字で始まる) の先頭のスラッシュは削除されるので、このようなファイルやディレクトリは現在のディレクトリと相対的に抽出されます。

指定されたファイルが、ディレクトリで内容が *tarfile* に書き込まれている場合は、このディレクトリは再帰的に抽出されます。可能な場合、所有者、変更時間、およびモードが復元されます。それ以外の場合、所有者を復元するにはスーパーユーザーになる必要があります。文字型特殊デバイスとブロック型特殊デバイス (**mknod(1M)** で作成) は、スーパーユーザーによってのみ抽出できます。*file* 引数が指定されない場合は、*tarfile* の全体の内容を抽出します。*tarfile* に同じ名前のファイルが複数ある場合、各ファイルがパス名どおりのディレクトリに書き込まれ、それ以前のファイルを上書きします。アーカイブからファイルを抽出する場合は、ファイル名にワイルドカードを使用することはできません。この場合には、次の形式のコマンドを使用してください。

```
tar xvf ... /dev/rmt/0` tar tf ... /dev/rmt/0 | \
grep 'pattern'`
```

r または **u** 機能を使って作成した *tarfile* から抽出した場合、ディレクトリ修正時間が正しく設定されないことがあります。また、これらの機能は、バックスペースや追加などの機能がいないというテープドライブの制限のために、多くのテープドライブでは使用できません。

r、**u**、または **x** 機能、あるいは **x** 機能修飾子を使用する場合には、*tarfile* 中の対応するファイルとパス名が一致している必要があります。たとえば、*./thisfile* を抽出するためには、*./thisfile* を指定する必要があります。*thisfile* ではありません。**t** 機能はどのように各ファイルが保存されているかを表示します。

機能修飾子

次の文字は、使用する機能文字に付けて使用します。

b *blocksize*

ブロック化因数。生の磁気テープアーカイブに読み取り、または書き込みをする場合に使用します (下記の **f** を参照)。*blocksize* 引数では *tarfile* 上で実行した読み取りや書き込みの各操作で含まれる 512 バイトのテープブロックの数を指定します。最小は 1、デフォルトは 20 です。最大値は使用可能なメモリーの総量と使用するテープデバイス固有のブロック化条件によって決まります (詳細は **mtio(7L)** を参照)。最大値は `INT_MAX/512 (4194303)` を超えることはできません。

テープアーカイブを読み取る場合は、実際のブロック化因数が自動的に検出され、名目上のブロック化因数 (**b** 修飾子を指定していない場合は、*blocksize* 引数の

値かデフォルト値)よりも少ないか、あるいは等しい値が割り当てられます。実際のブロック化因数が名目上のブロック化因数よりも大きい場合、結果は読み取りエラーになります。「使用例」の「例 5」を参照してください。

B

ブロック。tar は、(必要であれば) 複数の読み取り操作を実行し、ブロックを埋めるのに十分なバイト数だけを読み取ります。パイプやソケットは、それ以降の入力がある場合でもブロックを部分的に返すので、この機能修飾子は tar がイーサネットを介して動作することを可能にしています。標準入力 (-) から読み取る場合、tar がブロックを埋めるのに十分なバイト数を読み取れるようにこの機能修飾子がデフォルトで選択されます。

D

データ変更の警告。c、r、または u 機能文字と一緒に使用します。t または x 機能文字と一緒に使用した場合は、無視されます。ファイルのアーカイブ中にファイルのサイズが変わった場合には、この状態はエラーではなく警告と見なしてください。メッセージは書き込まれますが、終了ステータスには影響しません。

E

拡張ヘッダーを付けて tarfile を書き込みます (c、r、u オプションで使用可能、t または x 機能文字と一緒に使用した場合は、無視されます)。tarfile が拡張ヘッダー付きで書き込まれた場合、その修正時間は秒単位ではなくマイクロ秒単位で続けられます。さらにファイル名の文字数が PATH_MAX 文字 (保存には E フラグが必要) 以下で、ファイルサイズが 8G バイトを超えるファイルの場合もサポートします。E フラグは、サイズの大きいファイルや名前の長いファイル、またはその両方の場合、あるいはユーザー ID またはグループ ID が 2097151 を超えるときに保存したい場合、マイクロ秒単位の時間を取りたい場合に有用です。

f

ファイル。tarfile の名前として tarfile 引数を使用します。f を指定すると、/etc/default/tar を検索しません。f を省略した場合には、tar は TAPE 環境変数 (設定されている場合) が示すデバイスを使用します。そうでない場合、tar は、/etc/default/tar に定義されているデフォルト値を使用します。archiveN 文字列に一致する数字が出力デバイスとして、ファイル内のブロック化とサイズの指定とともに使用されます。たとえば、次のように指定します。

```
tar -c 2/tmp/*
```

このコマンドは、/etc/default/tar において archive2 に指定されているデバイスに出力を書き込みます。

tarfile の名前が - である場合は、tar は標準出力への書き込みあるいは標準入力からの読み取りのいずれか適当な操作を行います。tar はパイプラインの先頭もしくは末尾として使用することができます。また、次のコマンドを使えば、tar は、ディレクトリ階層を移動するためにも使用することができます。

```
example% cd fromdir; tar cf - . | (cd todir; tar xfBp -)
```

F

F 引数を 1 つ指定すると、tar は tarfile から SCCS および RCS の名前のついたすべてのディレクトリを除外します。FF のように引数を 2 つ指定すると、tar は SCCS および RCS の名前のついたすべてのディレクトリ、接尾辞として .o を持つすべてのファイル、および errs、core、a.out という名前のファイルをすべて除外します。

h

シンボリックリンクをたどり、通常ファイルあるいはディレクトリとして扱います。通常、tar はシンボリックリンクをたどりません。

i

ディレクトリ・チェックサム・エラーを無視します。

j

c モードのみ。生成されたアーカイブを bzip2 で圧縮します。抽出またはリストモードでは、このオプションは無視されます。この実装では、アーカイブの読み取り時に bzip2 圧縮タイプが自動的に認識されます。アップグレード/置換は、まず圧縮解除してから、同じメカニズムを適用して自動的にアーカイブを圧縮します。

l

リンク。保管されるファイルのリンクで、tar が解決できないものがある場合、エラーメッセージを出力します。l が指定されない場合は、エラーメッセージは表示されません。

m

修正。ファイルの変更時間を抽出した際の時間に設定します。この機能修飾子は、x 機能とともに用いた場合のみ有効です。

n

テープデバイスにないファイルを読み込みます。tar は、アーカイブ内部を読み取り専用で探ることができるため、アーカイブの読み込みは速くなります。

o

オーナーシップ。抽出されたファイルに、tarfile 上のユーザー識別子やユーザーのグループ識別子を使用する代わりに、プログラムを実行するユーザーのユーザー識別子やグループ識別子を割り当てます。これは、ルートではないユーザーに対してデフォルトで行われます。o 機能修飾子が設定されておらず、さらにユーザーがルートである場合、抽出されたファイルは、tarfile 上のファイルのグループ識別子とユーザー識別子を使用します (詳細は、chown(1) を参照)。o 機能修飾子は、x 機能とともに用いた場合のみ有効です。

p

現在の umask(1) を無視して、指定されたファイルを元のモード、および ACL が有効であれば、ACL を戻します。スーパーユーザーとして x 機能文字を指定して起動した場合は、これがデフォルトの動作になります。スーパーユーザーである場合は、SETUID およびスティッキ情報も抽出し、ファイルは

スーパーユーザーの所有ではなく、元の所有者とアクセス権に復元されます。この機能修飾子を、c 機能とともに使用した場合、ACL はほかの情報とともに tarfile に作成されます。ACL の入った tarfile が tar の前のバージョンによって抽出された場合、エラーが起こることがあります。

P

アーカイブ作成の場合、アーカイブのディレクトリエントリに最後の (/) を追加しません。

アーカイブ抽出の場合、パス名を保持します。デフォルトでは、アーカイブ抽出時に絶対パス名 (/ 文字で始まる) の先頭のスラッシュが削除されます。また、tar はパス名にドットドット (..) が含まれるアーカイブエントリの抽出を拒否します。

このオプションは、これらの動作を無効にします。

T

この修飾子を使用できるのは、システムが Trusted Extensions で構成されている場合だけです。

この修飾子を機能文字 c、r、または u と併用して tarfile の作成、置換、または更新を行う場合、保存対象の各ファイルやディレクトリに関連付けられた機密ラベルが、tarfile 内に格納されます。

T を指定すると、機能修飾子 p も指定されたものとみなされます。

機能文字 x と併用して tarfile の抽出を行う場合、tar プログラムは、アーカイブ内に指定されたファイルの機密ラベルが抽出先ディレクトリの機密ラベルに等しいかどうかを確認します。等しくない場合、ファイルは復元されません。この操作は大域ゾーンから起動してください。アーカイブ内のファイルが相対パス名を持つ場合、そのファイルは、同じラベルを持つ対応するディレクトリが使用できるときには、そのディレクトリに復元されます。これは、そのファイルに等しいラベルを持つゾーンのルートパス名を、現在の抽出先ディレクトリの前に付加することで実現されます。そのようなゾーンが存在しない場合、ファイルは復元されません。

Trusted Solaris 8 のラベル付きアーカイブの抽出については、限定されたサポートが提供されます。機密ラベルとマルチレベルディレクトリ指定のみが解釈されます。特権指定と監査属性フラグは自動的に無視されます。シングルレベルディレクトリへのシンボリックリンクを含むマルチレベルディレクトリ指定は、同じラベルを持つゾーンが使用可能な場合には、そのゾーンの相対パス名へとマップされます。このサポートの目的は、ホームディレクトリの移行を行いやすくすることです。アーキテクチャーが異なるため、Trusted Solaris 8 の任意のラベル付きファイルを Trusted Extensions の同一パス名に抽出することはできません。ファイルを抽出できるのは、そのアーカイブ上のラベルが抽出先のラベルに一致する場合だけです。

v

冗長。機能文字に続けて、各ファイル名を出力します。t 機能とともに使用すると、v は、tarfile エントリに関する詳しい情報も提供します。表示は **ls(1)** コマンドの -l オプションによる出力形式に似ています。

w

処理の指示。とるべき処理とファイル名を出力し、ユーザーの確認を待ちます。応答が肯定である場合、この動作が実行されます。そうでない場合には処理は行われません。この機能修飾子は、t 機能と同時に使えません。

x

除外。機能 c、x、または、t を使用した場合に、tarfile から除外されるファイル(あるいはディレクトリ)の相対パス名のリストを収めたファイルとして、*exclude-file* 引数を使用します。*include-file* の行末に不要なスペースを置かないように注意してください。また、各行頭にも不要なスペースを置かないように注意してください。除外するファイルの最初の文字列に対する一致には、改行で分かれている行全体が使用されます。*exclude-file* 内の行は厳密に照合されます。したがって、tar が相対パス名をバックアップしている場合、「/var」のようなエントリを指定しても、/var ディレクトリは除外されません。このような状況では、エントリを「./var」と指定する必要があります。tar コマンドは、*exclude-file* 内にあるシェルのメタ文字を展開しません。したがって、「*.o」のようなエントリを指定しても、名前に接尾辞「.o」が付いたすべてのファイルを除外するような効果はありません。複雑なファイルのリストを除外する場合は、**find(1)** コマンドを適切な条件で使用するなどして、*exclude-file* を生成する必要があります。

引数 1 つにつき 1 つの *exclude-file* で、複数の x 引数を使用できます。この場合、組み込まれたファイルが存在する場合には (-I *include-file* オペランドを参照)、除外されたファイルは組み込まれたファイルよりも優先されます。あるファイルが *exclude-file* と *include-file* の両方のファイル(またはコマンド行上)で指定されていれば、そのファイルは除外されます。

z

c モードのみ。生成されたアーカイブを gzip で圧縮します。抽出またはリストモードでは、このオプションは無視されます。この実装では、アーカイブの読み取り時に gzip 圧縮タイプが自動的に認識されます。アップグレード/置換は、まず圧縮解除してから、同じメカニズムを適用して自動的にアーカイブを圧縮します。

Z

c モードのみ。生成されたアーカイブを compress で圧縮します。**compress(1)** を参照してください。抽出またはリストモードでは、このオプションは無視されます。この実装では、アーカイブの読み取り時に compress 圧縮タイプが自動的に認識されます。アップグレード/置換は、まず圧縮解除してから、同じメカニズムを適用して自動的にアーカイブを圧縮します。

@

拡張属性をアーカイブに取り込みます。デフォルトでは、tar は拡張属性をアーカイブに取り込みません。このフラグを使用すると、tar はファイルが拡張属性を持っているかどうかを調べて、持っていれば、拡張属性をアーカイブに取り込みます。アーカイブにおいて、拡張属性は特殊な種類のラベルを持つ特殊なファイルとして格納されます。この修飾子を x 機能と一緒に使用すると、拡張属性は通常のファイルのデータと一緒にテープから抽出されます。拡張属性ファイルは、通常のファイルのデータの一部としてのみ抽出できます。拡張属性ファイルだけを明示的に抽出しようとするとう無視されます。

/

拡張システム属性をアーカイブに取り込みます。デフォルトでは、tar は拡張システム属性をアーカイブに取り込みません。このフラグを使用すると、tar はファイルが拡張システム属性を持っているかどうかを調べて、持っていれば、拡張属性をアーカイブに取り込みます。アーカイブにおいて、拡張システム属性は特殊な種類のラベルを持つ特殊なファイルとして格納されます。この修飾子を x 機能と一緒に使用すると、拡張システム属性は通常のファイルのデータと一緒にテープから抽出されます。拡張システム属性ファイルは、通常のファイルのデータの一部としてのみ抽出できます。拡張属性ファイルだけを明示的に抽出しようとするとう無視されます。

[0-7]

テープをマウントする代替ドライブを選択します。デフォルトは、/etc/default/tar に指定されています。数字または f 機能修飾子が指定されなければ、0 を持つ /etc/default/tar 中のエントリがデフォルトになります。

使用法

検出するファイルが 2G バイト (2^{31} バイト) 以上ある場合の tar の動作については、[largefile\(5\)](#) を参照してください。

実際のブロック化因数の自動決定は、パイプやソケットからの読み取り時には正しく行われないことがあります (B 機能修飾子を参照)。

1/4 インチのストリームテープは 512 バイト単位のブロック化因数を持ちます。すべてのブロック化因数を使用して、読み取りまたは書き込みが可能です。

この機能修飾子は、ディスクファイルやブロック型特殊デバイス上のアーカイブに対して動作しますが、主としてテープデバイス用を目的としています。

tar のヘッダー形式の情報については、[archives.h\(3HEAD\)](#) を参照してください。

使用例

例1 ユーザーのホームディレクトリのアーカイブを作成する

tar を使用して、ドライブ /dev/rmt/0 にマウントされたテープにユーザーのホームディレクトリのアーカイブを作成する例を示します。

```
example% cd
example% tar cvf /dev/rmt/0 .
messages from tar
```

例1 ユーザーのホームディレクトリのアーカイブを作成する (続き)

c 機能文字は、アーカイブの作成を意味します。v 機能修飾子は、tar の動作状況を説明するメッセージを出力します。f 機能修飾子は、tarfile を指定している (この例では /dev/rmt/0) ことを示します。コマンド行の最後のドット (.) は現在のディレクトリを示し、f 機能修飾子の引数になります。

次のコマンドで、tarfile の内容の一覧を表示します。

```
example% tar tvf /dev/rmt/0
```

POSIX ロケールでは、次のように出力されます。

```
rw-r--r-- 1677/40 2123 Nov 7 18:15 1985 ./test.c
...
example%
```

各カラムは次の意味を持ちます。

- カラム 1 は、./test.c へのアクセス権
- カラム 2 は、./test.c へのアクセス権
- カラム 3 は、./test.c のバイトサイズ
- カラム 4 は、./test.c の修正時刻。LC_TIME カテゴリが POSIX ロケールに設定されていない場合、形式や日付順フィールドが異なって使用されることがあります。
- カラム 5 は、./test.c へのアクセス権

アーカイブから、ファイルを抽出するためには、次を使用します。

```
example% tar xvf /dev/rmt/0
messages from tar
example%
```

テープ上に複数のアーカイブファイルがある場合は、それぞれのファイルは、EOF マーカによって次のファイルと区切られます。tar を使用して複数のアーカイブファイルが入っているテープから 1 番目と 2 番目のアーカイブファイルを読ませる場合、以下のように f 機能修飾子として渡すテープデバイス名は non-rewind 指定のある方を使用してください。

```
example% tar xvfp /dev/rmt/0n read first archive from tape
messages from tar
example% tar xvfp /dev/rmt/0n read second archive from tape
messages from tar
example%
```

以前のリリースでは、前述の処理が正しく動作しなかったり、mt(1) と tar の実行との間に調整が必要になったりする場合がありますので注意してください。以前のリリースでの動作をエミュレートするには、BSD 用の動作を行う b 文字を含んだ

例1 ユーザーのホームディレクトリのアーカイブを作成する (続き)

`non-rewind` のデバイス名を使用してください。詳細については [mtio\(7I\)](#) マニュアルページの「Close Operations」の項を参照してください。

例2 デフォルトのテープドライブ 0 への `/usr/include` および `/etc` からのファイルのアーカイブ
デフォルトのテープドライブ 0 に `/usr/include` のファイルおよび `/etc` のファイルを保存するためには、次を使用します。

```
example% tar c -C /usr include -C /etc .
```

このコマンドによって保存された `tarfile` からの内容の一覧を表示すると、たとえば次のようになります。

```
include/
include/a.out.h
and all the other files in /usr/include ...
./chown and all the other files in /etc
```

`include` の下にあるすべてのファイルを抽出するためには、次を使用します。

```
example% tar xv include
x include/, 0 bytes, 0 tape blocks \
and all files under include ...
```

例3 ネットワークを介してのファイルの転送

`tar` を使用して、ネットワークを介してファイルを転送する例を示します。最初に、ローカルマシン (`example`) からリモートシステム (`host`) 上のテープヘファイルを保管する方法を示します。

```
example% tar cvfb - 20 files | \
ssh host dd of=/dev/rmt/0 obs=20b
messages from tar
example%
```

前述の例では、`c` キー文字を使用して `tarfile` を作成し、`v` 機能修飾子により `tar` からの冗長出力を要求し、`f` 機能修飾子を用いて出力 `tarfile` の名前を指定し (`-` で、標準出力を指定)、`b` 機能修飾子によりブロックサイズ (`20`) を指定しています。ユーザーがブロックサイズを変更したい場合は、ユーザーは、`tar` コマンドおよび `dd` コマンド両方のブロックサイズ引数を変更する必要があります。

例4 リモートシステム上のテープからローカルシステムへファイルを戻す

次に、`tar` を使用して、リモートシステム上のテープからローカルシステムへファイルを戻す例を示します。

```
example% ssh -n host dd if=/dev/rmt/0 bs=20b | \
tar xvbfb - 20 files
```


例4 リモートシステム上のテープからローカルシステムへファイルを戻す (続き)

```
messages from tar
example%
```

前述の例では、**x**キー文字を使用して *tarfile* からファイルの抽出を行い、**v** 機能修飾子により **tar** からの冗長出力を要求し、**B** 機能修飾子によりパイプから読み取りを行うように **tar** に指示し、**f** 機能修飾子を用いて入力 *tarfile* の名前を指定し(−で、標準入力を指定)、**b** 機能修飾子によりブロックサイズ (20) を指定しています。

例5 ホームディレクトリのアーカイブの作成

次に実際のブロック化因数を 19 にして /dev/rmt/0 上にホームディレクトリのアーカイブを作成する例を示します。

```
example% tar cvfb /dev/rmt/0 19 $HOME
```

b 機能修飾子を使用しないでアーカイブの実際のブロック化因数を認識するためには、次を使用します。

```
example% tar tvf /dev/rmt/0
tar: blocksize = 19
...
```

実際のブロック化因数よりも大きい名目上のブロック化因数を使用して、アーカイブの実際のブロック化因数を認識するためには、次を使用します。

```
example% tar tvf /dev/rmt/0 30
tar: blocksize = 19
...
```

実際のブロック化因数に対して小さすぎる名目上のブロック化因数を使用して、アーカイブの実際のブロック化因数を認識しようとした場合は次のようになります。

```
example% tar tvf /dev/rmt/0 10
tar: tape read error
```

例6 圧縮アーカイブの作成

次の例では、**bzip** を使用して圧縮アーカイブを作成します。

```
example% tar cjf tarfile /tmp/*
```

圧縮ファイルの名前は *tarfile.bz2* です。

この場合、代わりに次のコマンドシーケンスを使用しても、同じ圧縮アーカイブが作成されます。

```
example% tar cf tarfile /tmp/*
example% bzip2 tarfile
```

例6 圧縮アーカイブの作成 (続き)

ただし、中間ファイルの作成および削除は行われません。機能修飾子 `z` および `Z` も同様に動作しますが、それぞれ `gzip` および `compress` を使用します。

次の例では、`compress` を使用して圧縮アーカイブを作成します。

```
example% tar czf tarfile /tmp/*
```

圧縮ファイルの名前は `tarfile.Z` です。

次の例では、`gzip` を使用して圧縮アーカイブを作成します。

```
example% tar czf tarfile /tmp/*
```

圧縮ファイルの名前は `tarfile.gz` です。

例7 圧縮アーカイブからのファイルの抽出

次の例では、圧縮アーカイブからファイルを抽出します。`bzip2` 圧縮モードを使用してアーカイブを圧縮した場合

```
example% tar xvf tarfile.bz2
example% tar xvfj tarfile.bz2
example% bzcat tarfile.bz2 | tar xvf -
```

`compress` 圧縮モードを使用してアーカイブを圧縮した場合

```
example% tar xvf tarfile.Z
example% tar xvfZ tarfile.Z
example% zcat tarfile.Z | tar xvf -
```

`gzip` 圧縮モードを使用してアーカイブを圧縮した場合

```
example% tar xvf tarfile.gz
example% tar xvfz tarfile.gz
example% gzcat tarfile.gz | tar xvf -
```

環境変数

TMPDIR

デフォルトでは、一時ファイルは `/tmp` に作成されます。それ以外の場合、`tar` は `TMPDIR` で指定したディレクトリを使用します。

`tar` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LC_COLLATE、LC_CTYPE、LC_MESSAGES、LC_TIME、TZ、およびNLSPATH。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの `yesexpr` キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、`yesexpr` に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、`yesexpr` に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#) を参照してください。

終了ステータス 次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。

ファイル

- /dev/rmt/[0-7][b][n]
- /dev/rmt/[0-7]l[b][n]
- /dev/rmt/[0-7]m[b][n]
- /dev/rmt/[0-7]h[b][n]
- /dev/rmt/[0-7]u[b][n]
- /dev/rmt/[0-7]c[b][n]
- /etc/default/tar

/etc/default/tar の設定は次のようになります:

```
archive0=/dev/rmt/0
archive1=/dev/rmt/0n
archive2=/dev/rmt/1
archive3=/dev/rmt/1n
archive4=/dev/rmt/0
archive5=/dev/rmt/0n
archive6=/dev/rmt/1
archive7=/dev/rmt/1n
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

関連項目

[ar\(1\)](#), [basename\(1\)](#), [cd\(1\)](#), [chown\(1\)](#), [compress\(1\)](#), [cpio\(1\)](#), [csh\(1\)](#), [dirname\(1\)](#), [find\(1\)](#),
[ls\(1\)](#), [mt\(1\)](#), [pax\(1\)](#), [setfacl\(1\)](#), [umask\(1\)](#), [mknod\(1M\)](#), [archives.h\(3HEAD\)](#),
[attributes\(5\)](#), [environ\(5\)](#), [fsattr\(5\)](#), [largefile\(5\)](#), [mtio\(7I\)](#)

診断

キー文字の誤りとテープ読み取り/書き込みエラー、およびリンクテーブルを保持する十分なメモリがないという診断メッセージが出力されます。

注意事項

ファイルの *n* 番目に現れるものをアクセスする方法がありません。

テープエラーの処理は不十分です。

tar アーカイブ形式ではアーカイブヘッダーにユーザー ID およびグループ ID を 2097151 まで格納することができます。この値よりも大きい UID および GID を持つファイルは 60001 の UID および GID で格納されます。

アーカイブを作成するときに、複数のロケールで実行している処理によってファイル名を作成したファイルを含んでいる場合、アーカイブの作成とアーカイブからのファイルの抽出のどちらも、フル 8 ビットコードセットを使用するロケール (たとえば、en_US ロケール) 上で使用すべきです。

1/4 インチのアーカイブテープ用のテープドライブはバックスペースできないので、このようなテープでは **r** オプションと **u** オプションは使用できません。

tar にはオプションがないため、標準の「—」引数は必要ありません。ほかのユーティリティーでは、この引数は通常、オプションの認識を停止するために使用されます。**tar** に指定した場合、この引数は最初の引数として認識されるだけで、無視されます。

-C directory file と **-I include-file** は複数の引数オペランドをとります。したがって、「**-C**」または「**-I**」という名前のファイルを保管または抽出するには、次のいずれかの方法を使用します。

1. これらのファイルを、スラッシュ (/) 文字を含む file オペランドとしてコマンド行に指定します。たとえば、「/home/joe/**-C**」や「./**-I**」です。
2. これらのファイルを含む include-file を作成して、**-I include-file** オプションで指定します。
3. これらのファイルが存在するディレクトリを指定します。

-C directory -C

または

-C directory -I

4. これらのファイルが存在するディレクトリ全体を指定します。

-C directory .

名前	tbl - nroff または troff 用のテーブルの書式化
形式	tbl [-me] [-mm] [-ms] [filename]...
機能説明	tbl は、nroff(1) または troff(1) 用のテーブルを書式化するプリプロセッサです。入力ファイル <i>filename</i> は、標準出力にコピーされます。この時、コマンド行 .TS と .TE の間にある行だけが、テーブルを記述するものとみなされ、書式化し直されます。 引数を省略すると、tbl は標準入力を読み取ります。したがって tbl はフィルタとして使用することができます。tbl を eqn(1) または neqn とともに使用する場合は、tbl コマンドを最初に指定し、パイプを通して渡すデータを最小限に抑えるようにします。
オプション	-me -me マクロパッケージを出力ファイルの前にコピーします。 -mm -mm マクロパッケージを出力ファイルの前にコピーします。 -ms -ms マクロパッケージを出力ファイルの前にコピーします。
使用例	例1 tbl の使用 例として、次のような入力ファイルがあるとしします (実際には '@' はタブ文字を入力する)。 <pre>.TS c s s c c s c c c l n n. Household Population Town@Households @Number@Size Bedminster@789@3.26 Bernards Twp.@3087@3.74 Bernardsville@2018@3.30 Bound Brook@3425@3.04 Branchburg@1644@3.49 .TE</pre> この出力は次のようになります。

Household Population		
Town	Households	Size
	数値	
Bedminster	789	3.26

例1 tblの使用 (続き)

Bernards Twp.	3087	3.74
Bernardsville	2018	3.30
Bound Brook	3425	3.04
Branchburg	1644	3.49

ファイル

/usr/share/lib/tmac/e	-me マクロ
/usr/share/lib/tmac/m	-mm マクロ
/usr/share/lib/tmac/s	-ms マクロ

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目

[eqn\(1\)](#), [nroff\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#)

名前	tcopy – copy a magnetic tape				
形式	tcopy <i>source</i> [<i>destination</i>]				
機能説明	The tcopy utility copies the magnetic tape mounted on the tape drive specified by the <i>source</i> argument. The only assumption made about the contents of a tape is that there are two tape marks at the end. When only a source drive is specified, tcopy scans the tape, and displays information about the sizes of records and tape files. If a destination is specified, tcopy makes a copies the source tape onto the <i>destination</i> tape, with blocking preserved. As it copies, tcopy produces the same output as it does when only scanning a tape. The tcopy utility requires the use of Berkeley-compatible device names. For example, <pre>example% tcopy /dev/rmt/1b /dev/rmt/2b</pre>				
属性	See attributes(5) for descriptions of the following attributes:				
	<table><tr><th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr><tr><td>Availability</td><td>system/core-os</td></tr></table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	system/core-os
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	system/core-os				
関連項目	mt(1) , ioctl(2) , attributes(5)				
注意事項	tcopy will only run on systems supporting an associated set of ioctl(2) requests.				

名前	tee – 標準出力の複写
形式	/usr/bin/tee [-ai] [file...]
機能説明	tee は標準入力を標準出力およびゼロ以上のファイルにコピーします。オプションは、指定したファイルを上書きするか、または追加するかを決定します。tee ユーティリティーは出力をバッファに格納しません。ファイルへの書き込みに失敗すると、tee はゼロ以外の終了ステータスで終了しますが、その他のファイルへの書き込みは継続します。 指定可能な file オペランドの数は、基盤となるオペレーティングシステムで制限されます。
オプション	サポートしているオプションは、次のとおりです。 -a 出力はファイルを上書きするのではなく、ファイルに追加されます。 -i 割り込みを無視します。
オペランド	次のオペランドがサポートされています。 file 出力ファイルのパス名。
使用法	ファイルが 2G バイト (2 ³¹ バイト) 以上ある場合の tee の動作については、 largefile(5) を参照してください。
環境変数	tee の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。標準入力は、正常にすべての出力ファイルにコピーされました。 >0 エラーが発生した。オープンできない、またはステータスが得られないファイルの数。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[cat\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	telnet – TELNET プロトコルを使用したりリモートシステムへのユーザーインタフェース
形式	<pre>telnet [-8EFKLacdfrx] [-X atype] [-e escape_char] [-k realm] [-l user] [-n file] [[[!] @hop1 [@hop2...] @] host [port]]</pre>
機能説明	telnet ユーティリティーは、TELNET プロトコルを使って他のホストと通信を行います。telnet が引数なしで起動された場合、コマンドモードに入り、telnet> プロンプトが表示されます。このモードでは、関連付けられたコマンドを受け入れて実行します。「使用法」を参照してください。telnet が引数を付けて起動された場合、それらの引数を使って open コマンドが実行されます。 たとえば、host が @hop1@hop2@host と指定された場合、host への接続はルーズソースルーティングを使って、ホスト hop1 と hop2 を経由して行われます。先頭に ! が付いていると、ストリクトソースルーティングに従って接続が行われます。telnet が IPv6 を使用する場合は、ルーズソースルーティングのみが可能で、! の指定は無視されることに注意してください。 接続が開始されると、telnet は入力モードに入ります。このモードでは、入力されたテキストがリモートホストに送信されます。入力モードは、リモートシステムでのサポート状況に応じて、「行モード」、「文字単位モード」、または「旧行単位モード」のいずれかになります。 「行モード」では、文字処理がリモートシステムの制御下に置かれたローカルシステムで行われます。入力の編集や文字のエコーがオフに設定された場合、リモートシステムはその情報を中継します。リモートシステムは、そこで発生する特殊文字の変更も中継するので、その変更をローカルシステムに反映させることができます。 「文字単位」モードでは、入力されたテキストの大部分は、ただちにリモートホストに送信され、処理されます。 「旧行単位」モードでは、テキストはすべてローカルでエコーされ、(通常は)完了した行だけがリモートホストに送信されます。「ローカルエコー文字」(初期値は ^E) は、ローカルエコーのオンとオフを切り替えるために使われます。(これは主にパスワードをエコーしないで入力する場合に使われます。) 「行モード」オプションが有効になっている場合、または localchars トグルが TRUE (これは「旧行単位」モードではデフォルト) の場合、ユーザーの quit、intr、および flush 文字はローカルでトラップされ、リモート側には TELNET プロトコルシーケンスとして送信されます。「行モード」が一度でも有効になっていると、ユーザーの susp と eof も TELNET プロトコルシーケンスとして送信されます。このとき、quit は BREAK ではなく、TELNET ABORT として送信されます。toggle autoflush および toggle autosynch オプションを指定すると、この処理によって、(リモートホストが TELNET シーケンスを確認するまで) 後続の端末への出力がフラッシュされたり、quit や intr の場合はそれまでの端末への入力がフラッシュされます。

リモートホストとの接続中に、ユーザーは `telnet` エスケープ文字 (初期値は `^I`) を入力することで、`telnet` コマンドモードに入ることができます。コマンドモードでは、通常の端末編集規則が利用できます。`telnet` コマンドプロンプトで `RETURN` を押すと、コマンドモードが終了します。

オプション

サポートしているオプションは、次のとおりです。

-8

8 ビットデータパスを指定します。入力と出力の両方に対して、`TELNET BINARY` オプションを用いるようにネゴシエーションが試みられます。

-a

自動ログインを試みます。これにより、`ENVIRON` オプションの `USER` 変数によってユーザー名が送信されます (リモートシステムでサポートされている場合)。使用される名前は、現在のユーザー ID と適合する場合は、`getlogin(3C)` によって返される現在のユーザーの名前になります。それ以外の場合は、ユーザー ID に関連付けられた名前になります。

-c

ユーザーの `telnetrc` ファイルの読み取りを無効にします。(このリファレンスページの `toggle skiprc` コマンドを参照してください。)

-d

`debug` トグルの初期値を `TRUE` に設定します。

-e *escape_char*

エスケープ文字の初期値を *escape_char* に設定します。*escape_char* は、`^` (Control キー) とそれに続く 1 文字から成る 2 文字のシーケンスにすることも可能です。2 番目の文字が `?` の場合、`DEL` 文字が選択されます。それ以外の場合、2 番目の文字は制御文字に変換され、エスケープ文字として使われます。*escape_char* が `NULL` 文字列 (つまり、`-e ''`) として定義された場合、これは `-e '^@'` (Control-@) と同等です。エスケープ文字にできる文字がないことを指定するには、`-E` オプションを使用します。

-E

すべての文字が、エスケープ文字として認識されなくなります。

-f

ローカルの資格情報のコピーをリモートシステムに転送します。

-F

転送可能なローカルの資格情報のコピーをリモートシステムに転送します。

-k *realm*

Kerberos 認証を使用している場合、`krb5.conf(4)` で定められたリモートホストのデフォルトのレルムではなく、*realm* で指定されたリモートホスト用のチケットを `telnet` が取得するようにリクエストします。

-K

リモートシステムへの自動ログインを行わないことを指定します。

-l user

リモートシステムが **ENVIRON** オプションを解釈できる場合、リモートシステムへの接続時に、**ENVIRON** オプションの **USER** 変数の値として *user* がリモートシステムに送信されます。

-L

出力に対して 8 ビットデータパスを指定します。これにより、出力に対して **BINARY** オプションを用いるようにネゴシエーションが行われます。

-n tracefile

トレース情報を記録するために *tracefile* を開きます。後述の **set tracefile** コマンドを参照してください。

-r

rlogin に似たユーザーインタフェースを指定します。このモードでは、エスケープ文字はチルダ (~) 文字に設定されます。ただし、**-e** オプションで変更された場合を除きます。**rlogin** のエスケープ文字は、キャリッジリターンの直後に入力された場合にのみ認識されます。このモードでも、**telnet** コマンドの前に **telnet** のエスケープ文字 (通常は「^」) を入力する必要があります。また、**rlogin** のエスケープ文字に続けて「.\r」を入力すると接続を閉じることができ、「^z」を入力すると接続を中断できます (**rlogin(1)** と同様)。このオプションは不確実なインタフェースであり、将来変更されることがあります。

-x

データストリームの暗号化を有効にします。このオプションが有効になっていると、認証のネゴシエーションが行えない場合または暗号化を有効にできない場合に **telnet** はエラーを表示して終了します。

-X atype

atype タイプの認証を無効にします。

使用法

telnet コマンド

このセクションでは、**telnet** で使用できるコマンドについて説明します。各コマンドは、そのコマンドであることを識別するのに十分なだけの文字数を入力すれば済みます。(これは、**mode**、**set**、**toggle**、**unset**、**environ**、および **display** の各コマンドの引数についても同様です。)

auth argument ...

auth コマンドは、**TELNET AUTHENTICATE** オプションを通じて送信される情報を操作します。**auth** コマンドの有効な引数は次のとおりです。

disable type

指定されたタイプの認証を無効にします。指定できるタイプの一覧を表示するには、**auth disable ?** コマンドを使用します。

enable type

指定されたタイプの認証を有効にします。指定できるタイプの一覧を表示するには、**auth enable ?** コマンドを使用します。

status

さまざまな認証タイプの現在のステータスを一覧表示します。

open [-l *user*] [[!] @*hop1* [@*hop2* ...]@*host* [*port*]

指定されたホストに接続します。ポート番号を指定しない場合、telnet はデフォルトポートの TELNET サーバーに接続を試みます。ホストの指定は、ホスト名 ([hosts\(4\)](#) を参照) でも、「ドット表記」で指定したインターネットアドレス ([inet\(7P\)](#) または [inet6\(7P\)](#) を参照) でも構いません。*host* が @*hop1*@*hop2*@*host* と指定された場合、*host* への接続はルーズソースルーティングを使って、ホスト *hop1* と *hop2* を経由して行われます。@ は、指定するホスト間の区切り記号として必須です。IPv4 の使用時に先頭に ! を付けると、ストリクトソースルーティングに従って接続が行われます。

-l オプションは、ENVIRON オプションの USER 変数の値として *user* をリモートシステムに送信します。

close

開いている TELNET セッションがある場合は閉じます。EOF (コマンドモードで入力) も同様にセッションを閉じて終了します。

encrypt

encrypt コマンドは、TELNET ENCRYPT オプションを通じて送信される情報を操作します。

encrypt コマンドの有効な引数は次のとおりです。

disable type [input|output]

指定された暗号化タイプを無効にします。input と output を省略した場合は、入力と出力の両方で無効になります。指定できるタイプの一覧を表示するには、encrypt disable ? コマンドを使用します。

enable type [input|output]

指定された暗号化タイプを有効にします。input と output を省略した場合は、入力と出力の両方で有効になります。指定できるタイプの一覧を表示するには、encrypt enable ? コマンドを使用します。

input

これは encrypt start input コマンドと同じです。

-input

これは encrypt stop input コマンドと同じです。

output

これは encrypt start output コマンドと同じです。

-output

これは encrypt stop output コマンドと同じです。

start [input|output]

暗号化の開始を試みます。input と output を省略した場合は、入力と出力の両方で有効になります。指定できるタイプの一覧を表示するには、**encrypt enable ?** コマンドを使用します。

status

現在の暗号化のステータスを一覧表示します。

stop [input|output]

暗号化を停止します。input と output を省略した場合は、入力と出力の両方で暗号化が停止されます。

type *type*

あとの **encrypt start** または **encrypt stop** コマンドで使用されるデフォルトの暗号化タイプを設定します。

quit

close と同じです。

z

telnet を中断します。このコマンドは、ジョブ制御をサポートするシェル (**sh(1)** など) をユーザーが使用している場合にのみ機能します。

mode *type*

要求されたモードに移行できるかどうかをリモートホストに問い合わせます。リモートホストがそのモードに移行できる場合は、要求されたモードに移行します。引数 *type* には次のいずれかを指定します。

character

TELNET LINEMODE オプションを無効にします。リモート側が LINEMODE オプションを解釈できない場合は、「文字単位」モードに移行します。

line

TELNET LINEMODE オプションを有効にします。リモート側が LINEMODE オプションを解釈できない場合は、「旧行単位」モードへの移行を試みます。

isig (-isig)

LINEMODE オプションの TRAPSIG モードを有効(無効)にしようとします。このためには、LINEMODE オプションが有効になっている必要があります。

edit (-edit)

LINEMODE オプションの EDIT モードを有効(無効)にしようとします。このためには、LINEMODE オプションが有効になっている必要があります。

softtabs (-softtabs)

LINEMODE オプションの SOFT_TAB モードを有効(無効)にしようとします。このためには、LINEMODE オプションが有効になっている必要があります。

litecho (-litecho)

LINEMODE オプションの LIT_ECHO モードを有効(無効)にしようとします。このためには、LINEMODE オプションが有効になっている必要があります。

?

mode コマンドのヘルプ情報を表示します。

status

telnet の現在のステータスを表示します。これには、現在のモードだけでなく、接続されている相手側も含まれます。

display

[*argument*...] set および toggle 値のすべてまたは一部を表示します (toggle *argument*... を参照)。

?

[*command*] ヘルプを表示します。引数を付けない場合、telnet はヘルプのサマリーを表示します。コマンドを指定した場合、telnet はそのコマンドのヘルプ情報だけを表示します。

send *argument* . . .

1 つ以上の特殊文字シーケンスをリモートホストに送信します。指定できる引数は次のとおりです (一度に複数の引数を指定することもできます)。

escape

現在の telnet エスケープ文字 (初期値は ^j) を送信します。

synch

TELNET SYNCH シーケンスを送信します。このシーケンスにより、リモートシステムでそれまでに打ち込まれた入力 (ただし、まだ読み取られていないもの) がすべて破棄されます。このシーケンスは、TCP 緊急データとして送信され、リモートシステムが 4.2 BSD システムの場合は機能しないことがあります。機能しない場合は、端末に小文字の「r」がエコーされます。

brk または break

TELNET BRK (Break) シーケンスを送信します。これは、リモートシステムにとって重要な意味を持つ場合があります。

ip

TELNET IP (Interrupt Process) シーケンスを送信します。これにより、リモートシステムで現在実行されているプロセスが中止されます。

abort

TELNET ABORT (Abort Process) シーケンスを送信します。

ao

TELNET AO (Abort Output) シーケンスを送信します。これにより、リモートシステムからユーザーの端末へのすべての出力がフラッシュされます。

ayt

TELNET AYT (Are You There) シーケンスを送信します。リモートシステムは、このシーケンスに応答する場合も応答しない場合もあります。

ec

TELNET EC (Erase Character) シーケンスを送信します。これにより、最後に入力された文字が消去されます。

el

TELNET EL (Erase Line) シーケンスを送信します。これにより、リモートシステムは現在入力中の行を消去します。

eof

TELNET EOF (End Of File) シーケンスを送信します。

eor

TELNET EOR (End Of Record) シーケンスを送信します。

ga

TELNET GA (Go Ahead) シーケンスを送信します。これは、おそらくリモートシステムにとって意味がありません。

getstatus

リモート側が TELNET STATUS コマンドをサポートしている場合、`getstatus` はサブネゴシエーションを送信して、現在のオプションのステータスを送るようにサーバーに要求します。

nop

TELNET NOP (No Operation) シーケンスを送信します。

susp

TELNET SUSP (Suspend Process) シーケンスを送信します。

do *option*

dont *option*

will *option*

wont *option*

指定された TELNET プロトコルオプションのネゴシエーションを送信します。*option* には、プロトコルオプションのテキスト名を指定することも、オプションに対応する番号を指定することもできます。指定されたオプションのネゴシエーションが現在の状態で有効でない場合、このコマンドはそのまま無視されます。*option* が `help` または `?` と指定された場合は、既知のオプション名の一覧が表示されます。このコマンドは主に、通常あまり見られないデバッグの状況で役立ちます。

?

`send` コマンドのヘルプ情報を表示します。

set *argument* [*value*]

unset *argument*

`telnet` 変数の 1 つを特定の値に設定します。特殊な値 `off` は、変数に関連付けられている機能をオフにします。変数の値は、`display` コマンドを使って調べるこ

とができます。 *value* を省略した場合、値は真 (つまり「on」) と見なされます。 *unset* 形式を使用した場合、値は偽 (つまり off) と見なされます。指定できる変数は次のとおりです。

echo

これは、「行単位」モードのとき、入力された文字を通常の処理としてローカルでエコーするか、入力された文字のエコーを抑制するか (パスワードが入力された場合など) を切り替える値です (初期値は ^E)。

escape

これは、リモートシステムに接続されているときに、telnet コマンドモードに入るための telnet エスケープ文字です (初期値は ^J)。

interrupt

telnet が localchars モード (toggle、localchars を参照) にある場合に、interrupt 文字が入力されると、TELNET IP シーケンス (send、ip を参照) がリモートホストに送信されます。interrupt 文字の初期値は、端末の intr 文字になります。

quit

telnet が localchars モードにある場合に、quit 文字が入力されると、TELNET BRK シーケンス (send、brk を参照) がリモートホストに送信されます。quit 文字の初期値は、端末の quit 文字になります。

flushoutput

telnet が localchars モードにある場合に、flushoutput 文字が入力されると、TELNET AO シーケンス (send、ao を参照) がリモートホストに送信されます。flushoutput 文字の初期値は、端末の flush 文字になります。

erase

telnet が localchars モードにあり、かつ「文字単位」モードで動作している場合に、erase 文字が入力されると、TELNET EC シーケンス (send、ec を参照) がリモートシステムに送信されます。erase 文字の初期値は、端末の erase 文字になります。

kill

telnet が localchars モードにあり、かつ「文字単位」モードで動作している場合に、kill 文字が入力されると、TELNET EL シーケンス (send、el を参照) がリモートシステムに送信されます。kill 文字の初期値は、端末の kill 文字になります。

eof

telnet が「行単位」モードで動作している場合に、eof 文字を行の先頭文字として入力すると、この文字がリモートシステムに送信されます。eof の初期値は、端末の eof 文字になります。

ayt

telnet が localchars モードにあるか、LINEMODE が有効になっている場合に、ステータス文字が入力されると、TELNET AYT (Are You There) シーケンスが

リモートホストに送信されます。(前述の `send`、`ayt` を参照してください。) `ayt` の初期値は、端末のステータス文字になります。

`forw1`

`forw2`

`telnet` が `LINEMODE` で動作している場合に、`forw1` 文字または `forw2` 文字が入力されると、行の一部がリモートシステムに転送されます。転送文字の初期値は、端末の `eol` 文字と `eol2` 文字になります。

`lnext`

`telnet` が `LINEMODE` または「旧行単位」モードで動作している場合、`lnext` 文字は端末の `lnext` 文字と見なされます。`lnext` 文字の初期値は、端末の `lnext` 文字になります。

`reprint`

`telnet` が `LINEMODE` または「旧行単位」モードで動作している場合、`reprint` 文字は端末の `reprint` 文字と見なされます。`reprint` の初期値は、端末の `reprint` 文字になります。

`rlogin`

これは、`rlogin` のエスケープ文字です。これを設定すると、行頭でこの文字の直後に置かれている場合を除いて、`telnet` の通常のエスケープ文字は無視されます。行頭で `rlogin` 文字のあとに「`.`」を続けると、接続が閉じられます。「`^Z`」を続けると、`rlogin` コマンドは `telnet` コマンドを中断します。初期状態では、`rlogin` のエスケープ文字は無効になっています。

`start`

`TELNET TOGGLE-FLOW-CONTROL` オプションが有効になっている場合、`start` 文字は端末の `start` 文字になります。`start` 文字の初期値は、端末の `start` 文字になります。

`stop`

`TELNET TOGGLE-FLOW-CONTROL` オプションが有効になっている場合、`stop` 文字は端末の `stop` 文字になります。`stop` 文字の初期値は、端末の `stop` 文字になります。

`susp`

`telnet` が `localchars` モードにあるか、`LINEMODE` が有効になっている場合に、`suspend` 文字が入力されると、`TELNET SUSP` シーケンス (前述の `send`、`susp` を参照) がリモートホストに送信されます。`suspend` 文字の初期値は、端末の `suspend` 文字になります。

`tracefile`

これは、`netdata` または `debug` オプションが `TRUE` の場合に出力が書き込まれるファイルです。`tracefile` が「`-`」に設定されている場合は、トレース情報が標準出力に書き込まれます (デフォルト)。

worderase

telnet が LINEMODE または「旧行単位」モードで動作している場合、この文字は端末の worderase 文字と見なされます。worderase 文字の初期値は、端末の worderase 文字になります。

?

有効な set および unset コマンドを表示します。

slc state

slc (Set Local Characters) コマンドは、TELNET LINEMODE オプションが有効な場合に、特殊文字の状態を設定または変更するために使われます。特殊文字とは、TELNET コマンドシーケンスにマップされる文字 (ip や quit など) または行編集文字 (erase や kill など) のことです。デフォルトでは、ローカルの特殊文字はエクスポートされます。state の有効な値は次のとおりです。

check

現在の特殊文字の設定を検証します。リモート側に、現在の特殊文字の設定をすべて送信するように要求します。ローカル側との間に食い違いがある場合は、ローカル側の設定をリモート側の値に切り替えます。

export

特殊文字をローカル側のデフォルト文字に切り替えます。ローカル側のデフォルト文字とは、telnet が起動したときのローカル端末の特殊文字です。

import

特殊文字をリモート側のデフォルト文字に切り替えます。リモート側のデフォルト文字とは、TELNET 接続が確立されたときのリモートシステムの特特殊文字です。

?

slc コマンドのヘルプ情報を表示します。

toggle argument...

telnet がイベントにどのように応答するかを制御する各種フラグを TRUE と FALSE の間で切り替えます。複数の引数を指定できます。これらのフラグの状態は、display コマンドを使って調べることができます。有効な引数は次のとおりです。

authdebug	認証コードのデバッグ情報を有効にします。
autodecrypt	TELNET ENCRYPT オプションのネゴシエーションが行われるときに、デフォルトではデータストリームの実際の暗号化 (復号化) は自動的に開始されません。autoencrypt (autodecrypt) コマンドは、出力 (入力) ストリームの暗号化ができるだけ早く有効になるようにします。
autologin	リモート側で TELNET AUTHENTICATION オプションがサポートされている場合、telnet はそのオプションを使って自動認証を行おうとします。AUTHENTICATION オプションがサポートされ

ていない場合は、TELNET ENVIRON オプションを通じてユーザーのログイン名が渡されます。このコマンドは、open コマンドで -a オプションを指定するのと同じです。

autoflush	autoflush と localchars がどちらも TRUE の場合に、ao、intr、または quit 文字が認識され、TELNET シーケンスに変換されると (詳細は set を参照)、それらの TELNET シーケンスが処理されたことが (TELNET Timing Mark オプションを使って) リモートシステムで確認されるまで、telnet はユーザーの端末にデータを何も表示しません。端末ユーザーが「stty noflsh」を実行していない場合、このトグルの初期値は TRUE になります。それ以外の場合は、FALSE になります (stty(1) を参照)。
autosynch	autosynch と localchars の両方が TRUE の場合に、interrupt 文字または quit 文字が入力されると (interrupt と quit については、set を参照)、結果として送信される TELNET シーケンスのあとに TELNET SYNCH シーケンスが続きます。この手順により、リモートシステムは、この両方の TELNET シーケンスが読み取られて処理が完了するまで、それまでに打ち込まれたすべての入力を破棄するはずですが。このトグルの初期値は FALSE です。
binary	入力と出力の両方に対して、TELNET BINARY オプションを有効または無効にします。
inbinary	入力に対して、TELNET BINARY オプションを有効または無効にします。
outbinary	出力に対して、TELNET BINARY オプションを有効または無効にします。
crlf	キャリッジリターンがどのように送信されるかを決めます。この値が TRUE の場合、キャリッジリターンは <CR><LF> として送信されます。この値が FALSE の場合、キャリッジリターンは <CR><NUL> として送信されます。このトグルの初期値は FALSE です。
crmod	RETURN モードを切り替えます。このモードが有効の場合、リモートホストから受信したほとんどの RETURN 文字は RETURN LINEFEED にマップされます。このモードは、ユーザーの入力した文字には影響せず、リモートホストから受信した文字だけに影響します。このモードは、リモートホストが LINEFEED を送信せずに RETURN だけを送信する場合にのみ役立ちます。このトグルの初期値は FALSE です。

debug	ソケットレベルのデバッグを切り替えます (スーパーユーザーのみ利用可能)。このトグルの初期値は FALSE です。
encdebug	暗号化コードのデバッグ情報を有効にします。
localchars	このトグルが TRUE の場合、flush、interrupt、quit、erase、および kill 文字 (set を参照) はローカルで認識され、対応する TELNET 制御シーケンス (それぞれ ao、ip、brk、ec、および el、send を参照) に変換されます。このトグルの初期値は、「行単位」モードでは TRUE、「文字単位」モードでは FALSE です。LINEMODE オプションが有効の場合、localchars の値は無視され、常に TRUE と見なされます。LINEMODE が一度でも有効になっていると、quit は abort として送信され、eof と suspend は eof と susp として送信されます (前述の send を参照)。
netdata	すべてのネットワークデータ (16 進形式) の表示を切り替えます。このトグルの初期値は FALSE です。
options	一部の内部 TELNET プロトコル処理 (telnet オプションに関係がある) の表示を切り替えます。このトグルの初期値は FALSE です。
prettydump	netdata トグルが有効の場合に、prettydump を有効にすると、netdata コマンドの出力がユーザーにとってより読みやすい形式になります。出力内のそれぞれの文字の間にスペースが挿入されます。TELNET エスケープシーケンスの先頭にアスタリスク (*) を付けて、それらを見つけやすくします。
skiprc	skiprc トグルが TRUE の場合、TELNET は、接続が開かれるときに、ユーザーのホームディレクトリにある .telnetrc ファイルの読み取りをスキップします。このトグルの初期値は FALSE です。
termdata	すべての端末データ (16 進形式) の表示を切り替えます。このトグルの初期値は FALSE です。
verbose_encrypt	verbose_encrypt フラグが TRUE の場合、TELNET は暗号化が有効または無効になるたびにメッセージを表示します。このトグルの初期値は FALSE です。
?	有効な toggle コマンドを表示します。
<i>environ argument...</i>	
environ コマンドは、TELNET ENVIRON オプションを通じて送信される変数を操作するために使われます。変数の初期値のセットは、ユーザーの環境から取得されます。DISPLAY 変数と PRINTER 変数だけはデフォルトでエクスポートされます。environ コマンドの有効な引数は次のとおりです。	

define variable value
変数 *variable* が値 *value* を持つように定義します。このコマンドで定義された変数はすべて自動的にエクスポートされます。*value* は単一引用符または二重引用符で囲んでも構わないため、タブやスペースを含めることができます。

undefine variable
変数 *variable* を環境変数のリストから削除します。

export variable
変数 *variable* がリモート側にエクスポートされるようにマークします。

unexport variable
変数 *variable* が、リモート側から明示的に要求されないかぎり、リモート側にエクスポートされないようにマークします。

list
現在の環境変数のセットを一覧表示します。アスタリスク (*) でマークされた変数は自動的に送信されます。その他の変数は、明示的に要求された場合にのみ送信されます。

?
environ コマンドのヘルプ情報を表示します。

logout
telnet logout オプションをリモート側に送信します。このコマンドは、**close** コマンドに似ています。ただし、リモート側が **logout** オプションをサポートしていない場合は何も行われません。リモート側が **logout** オプションをサポートしている場合は、このコマンドにより、リモート側は **TELNET** 接続を閉じます。リモート側が、あとで再接続するためにユーザーのセッションを中断するという概念もサポートしている場合、**logout** 引数は、ただちにセッションを終了するようにリモート側に指示します。

ファイル **\$HOME/.telnetrc** **telnet** セッションを開始する前に実行されるコマンドを記述するファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	network/telnet

関連項目 [rlogin\(1\)](#), [sh\(1\)](#), [stty\(1\)](#), [getlogin\(3C\)](#), [hosts\(4\)](#), [krb5.conf\(4\)](#), [nologin\(4\)](#), [telnetrc\(4\)](#), [attributes\(5\)](#), [inet\(7P\)](#), [inet6\(7P\)](#)

診断 **NO LOGINS: System going down in N minutes**
マシンがシャットダウンの処理中であり、ログインが無効になっています。

注意事項

リモートシステムによっては、「行単位」モードのときに手動でエコーをオフにする必要があります。

「旧行単位」モードまたは `LINEMODE` では、端末の EOF 文字は、それが行頭にある場合にのみ認識 (およびリモートシステムに送信) されます。

telnet プロトコルは、セッションの保護にシングル DES だけを使用します。クライアントはシングル DES セッション鍵を使ってサービスチケットを要求します。KDC は、telnet サービスを提供するホストサービスの主体がシングル DES をサポートしていることを認識している必要があります、実際にこれは、そのような主体がシングル DES 鍵を KDC データベース内に持っている必要があることを意味します。

名前	test – 条件の評価
形式	<code>/usr/bin/test [condition]</code> <code>[[condition]]</code>
sh	<code>test [condition]</code> <code>[[condition]]</code>
csch	<code>test [condition]</code> <code>[[condition]]</code>
ksh88	<code>test [condition]</code> <code>[[condition]]</code>
ksh	<code>test [condition]</code> <code>[[condition]]</code>

機能説明

test ユーティリティーは、*condition* が示す条件を評価し、その結果を終了ステータスで表します。評価結果が真のときは終了ステータスが 0 に、偽のときは 1 になります。

test ユーティリティーの第 1 の形式は次のとおりです。

`test [condition]`

角括弧は *condition* が省略可能なオペランドであり、必ずしもコマンド行で指定する必要がないことを示します。

test ユーティリティーの第 2 の形式は次のとおりです。

`[ [ condition ] ]`

最初の開く角括弧 ([) は必須のユーティリティー名です。内側の角括弧は *condition* がオプションであることを示しています。最後の閉じる角括弧 (]) は必須のオペランドです。

2G バイト (2³¹ バイト) 以上のファイルに対する test の動作については、[largefile\(5\)](#) を参照してください。

test と [ユーティリティーは条件 *condition* を評価して、その値が真である場合は終了ステータスを 0 に設定します。そうでない場合はゼロ以外の値 (偽) に設定します。引数がない場合は、test と [ユーティリティーは終了ステータスをゼロ以外の値に設定します。アクセス権をテストするときは、プロセスの実効ユーザー ID が使用されます。

test と [ユーティリティーに指定する引数、つまり、オペレータ、フラグ、および (最後の行を示す) 括弧はすべて、区切り文字で区切る必要があります。通常、これらの引数はスペースで区切られます。

オペランド

2つの要素を持つ、次の形式のプライマリ:

-primary_operator primary_operand

これを、「単項プライマリ」と呼びます。3つの要素を持つ、次のいずれかの形式のプライマリ:

primary_operand -primary_operator primary_operand

primary_operand primary_operator primary_operand

これを、「2項プライマリ」と呼びます。

ファイルオペランド (*-h* と *-L* プライマリを除く) がシンボリックリンクを示している場合、シンボリックリンクは展開され、*test* は展開後のファイルに実行されます。

自分のファイルをテストした場合でも (*-r*、*-w*、または *-x* のテスト)、テストしたファイルのアクセス権に所有者ビットが設定されていない場合は、ファイルのアクセス権に「グループ」ビットまたは「その他」のビットが設定されていても、ゼロ以外 (偽) の終了ステータスが返されます。

= と *!=* プライマリは単項プライマリよりも優先されます。*=* と *!=* プライマリは常に引数をとります。したがって、*=* と *!=* は単項プライマリの引数として使用できません。

condition を記述するために以下の基本式が使用できます。

- a file* ファイル *file* が存在すれば真。(sh では使用不可)
- b file* ファイル *file* が存在し、ブロック型特殊ファイルであれば真。
- c file* ファイル *file* が存在し、文字型特殊ファイルであれば真。
- d file* ファイル *file* が存在し、ディレクトリであれば真。
- e file* ファイル *file* が存在すれば真。(sh では使用不可)
- f file* ファイル *file* が存在し、そのファイルが通常ファイルであれば真。あるいは、*/usr/bin/sh* ユーザーが自分の *PATH* 環境変数内で */usr/bin* よりも前に */usr/ucb* を指定しており、*file* が存在し、そのファイルがディレクトリでなければ、*test* は真を返します。*csh* の *test* と [ビルトインは常に後者の動作を行います。
- g file* ファイル *file* が存在し、セットグループ ID フラグが設定されていれば真。
- G file* ファイル *file* が存在し、ファイルのグループがプロセスの実効グループ ID と一致していれば真。(sh では使用不可)
- h file* ファイル *file* が存在し、シンボリックリンクであれば真。

-k <i>file</i>	ファイル <i>file</i> が存在し、スティッキビットが設定されていれば真。
-L <i>file</i>	ファイル <i>file</i> が存在し、シンボリックリンクであれば真。
-n <i>string</i>	<i>string</i> の長さがゼロでなければ真。
-o <i>option</i>	オプション <i>option</i> がついていれば真。csh または sh では、このオプションは使用できません。
-O <i>file</i>	ファイル <i>file</i> が存在し、プロセスの実効ユーザー ID がそのファイルを所有していれば真。sh では、このオプションは使用できません。
-p <i>file</i>	ファイル <i>file</i> が名前付きパイプ (FIFO) であれば真。
-r <i>file</i>	ファイル <i>file</i> が存在し、読み取り可能であれば真。
-s <i>file</i>	ファイル <i>file</i> が存在し、サイズがゼロより大きければ真。
-S <i>file</i>	ファイル <i>file</i> が存在し、そのファイルがソケットであれば真。sh では、このオプションは使用できません。
-t [<i>file_descriptor</i>]	ファイル記述子番号が <i>file_descriptor</i> であるファイルがオープンされていて、端末装置に対応していれば真。 <i>file_descriptor</i> が指定されていない場合、デフォルト値として 1 が使用されます。
-u <i>file</i>	ファイル <i>file</i> が存在し、セットユーザー ID フラグが設定されていれば真。
-w <i>file</i>	ファイル <i>file</i> が存在し、書き込み可能であれば真。真であっても、書き込みフラグが設定されていることを表すだけです。ファイルシステムが読み取り専用であれば、この条件が真であっても <i>file</i> は書き込み不可能です。
-x <i>file</i>	ファイル <i>file</i> が存在し、実行可能であれば真。真であっても、実行フラグが設定されていることを表すだけです。 <i>file</i> がディレクトリの場合には、真は <i>file</i> が検索可能なことを表します。
-z <i>string</i>	文字列 <i>string</i> の長さがゼロであれば真。
<i>file1</i> -nt <i>file2</i>	ファイル <i>file1</i> が存在し、 <i>file2</i> よりも新しい場合は真。sh では、このオプションは使用できません。
<i>file1</i> -ot <i>file2</i>	ファイル <i>file1</i> が存在し、 <i>file2</i> よりも古い場合は真。sh では、このオプションは使用できません。

<i>file1</i> -ef <i>file2</i>	ファイル <i>file1</i> と <i>file2</i> が存在し、同じファイルを参照している場合は真。sh では、このオプションは使用できません。
<i>string</i>	文字列 <i>string</i> が NULL 文字列でなければ、真です。
<i>string1</i> = <i>string2</i>	文字列 <i>string1</i> と <i>string2</i> が等しければ真。
<i>string1</i> != <i>string2</i>	文字列 <i>string1</i> と <i>string2</i> が等しくなければ真。
<i>n1</i> -eq <i>n2</i>	数値 <i>n1</i> と <i>n2</i> が代数的に等しければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>n1</i> -ne <i>n2</i>	数値 <i>n1</i> と <i>n2</i> が代数的に等しくなければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>n1</i> -gt <i>n2</i>	数値 <i>n1</i> が代数的に数値 <i>n2</i> より大きければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>n1</i> -ge <i>n2</i>	数値 <i>n1</i> が代数的に数値 <i>n2</i> より大きいかに等しければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>n1</i> -lt <i>n2</i>	数値 <i>n1</i> が代数的に数値 <i>n2</i> より小さければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>n1</i> -le <i>n2</i>	数値 <i>n1</i> が代数的に数値 <i>n2</i> より小さいかに等しければ真。数値は C99/XPG6/SUS で指定された任意の形式の整数、浮動小数点、または浮動小数点定数 ([+/-]Inf, [+/-]NaN など) です。
<i>condition1</i> -a <i>condition2</i>	<i>condition1</i> および <i>condition2</i> の両方が真であれば真。-a 2 項プライマリは左結合であり、-o 2 項プライマリよりも優先される。
<i>condition1</i> -o <i>condition2</i>	<i>condition1</i> または <i>condition2</i> のどちらかが真であれば真。-o バイナリプライマリは左結合である。

これらの基本式は、以下の演算子と組み合わせて指定できます。

! *condition* 条件 *condition* が偽であれば真。

(*condition*) *condition* が真であれば真。丸括弧 () を使用すると、通常の優先順位や結合規則を変更できる。丸括弧はシェルにとっても意味を持つので、引用する必要があります。

演算子の優先順位を決めるアルゴリズム、および生成される戻り値は、*test* に渡す引数の数により異なります。ただし [...] 形式を使う場合、最終引数の右角括弧は、このアルゴリズムでは引数には含まれません。

以下のリストにおいて、\$1、\$2、\$3、\$4 は、*condition*、*condition1*、または *condition2* として *test* に渡す引数を表します。

引数 0 個 偽 (1) で終了。

引数 1 個 \$1 が空でなければ真 (0) で終了。NULL なら偽。

引数 2 個

- \$1 が ! のとき、\$2 が空なら真、\$2 が空でなければ偽。
- \$1 が単項基本式のとき、単項テストの結果が真なら真、偽なら偽。
- その他の場合、結果は不確定です。

引数 3 個

- \$2 が 2 項基本式のとき、\$1 と \$3 を対象に 2 項テストを実行。
- \$1 が ! のとき、\$2 と \$3 の 2 つの引数に対するテストを否定形に。
- その他の場合、結果は不確定です。

引数 4 個

- \$1 が ! のとき、\$2、\$3、\$4 の 3 つの引数に対するテストを否定形に。
- その他の場合、結果は不確定です。

使用法

基本式と演算子が混在しているような、ユーザーが用意した入力データを処理する際、スクリプト側では十分な注意が必要です。スクリプトに対して入力データが生成されるケースをアプリケーション作成者がすべて理解していない限り、*test "\$1" -a "\$2"* というような記述は、*test "\$1" && test "\$2"* と変えた方がいいでしょう。ユーザーが \$1 を ! に設定し、\$2 を空の文字列に設定したような場合の問題を防ぐためです。つまり移植性の高さが問題の場合には、*test expr1 -a expr2* というような記述は *test expr1 && test expr2* に変更し、また、*test expr1 -o expr2* の記述は *test expr1 || test expr2* に変更してください。ただし、シェルでは && と -| | が同等の優先順位を持つのに対し、*test* では -a の方が o よりも優先順位が高いことに注意してください。

シェルコマンド言語では、グループ分け用に小括弧と中括弧が使用できます。

sh を使うときは、小括弧はエスケープ付きで記述しなければなりません。例:

```
test \( expr1 -a expr2 \) -o expr3
```

このコマンドは、XSI 準拠のシステム以外では、必ずしも移植可能ではありません。この形式の代わりに、以下のように記述できます。

```
( test expr1 && test expr2 ) || test expr3
```

次に、以下の2つのコマンドを見てください。

```
test "$1"
test ! "$1"
```

古いシステムの一部では、このコマンドの動作は信頼性を欠くことがあります。つまりこのような *string* 条件を使い \$1 を !、(、または単項基本式に展開すると、その結果は保証できません。したがって、それぞれ以下のように記述する方がいいでしょう。

```
test -n "$1"
test -z "$1"
```

同様に、古いシステムでは以下のようなコマンドの信頼性は保証できません。

```
test "$response" = "expected string"
```

これは、次のどちらかの形式に変えるようにしてください。

```
test "X$response" = "Xexpected string"
test "expected string" = "$response"
```

2 番目の形式は、expected string が単項基本式とまぎらわしくなることはない、というのが前提です。つまり expected string の先頭文字が -, (, !, = のいずれかのときは、1 番目の形式を使用してください。注記した事項を除き、前述の規則を使用すれば、前述の3つの比較形式はどのような入力データに対しても信頼性の高いものです。なお、どの形式でも文字列は引用符で囲んで記述する点に注意してください。

引数の数が4個を超えている場合、文字列比較の2項基本式 = と != はどの単項基本式よりも優先順位が高いため、引数が正しく指定されていないと処理結果は保証できません。たとえば次の例を見てください。

```
test -d $1 -o -d $2
```

ここで \$1 の評価結果がディレクトリ名 = となった場合、先頭の3つの引数は文字列比較を表すものと見なされます。したがって2つ目の -d を検出したときに構文エラーとなります。これを避けるには、次に示すどちらかの形式、できれば2番目の方を使用してください。

```
test \( -d "$1" \) -o \( -d "$2" \)
test -d "$1" || test -d "$2"
```

また引数が4個を超えているときに次のように指定したとします。

```
test "$1" = "bat" -a "$2" = "ball"
```

このとき \$1 の評価結果が (または !)だと構文エラーとなります。これを避けるには、次に示すいずれかの形式、できれば3番目を使用してください。

```
test "X$1" = "Xbat" -a "X$2" = "Xball"
test "$1" = "bat" && test "$2" = "ball"
test "X$1" = "Xbat" && test "X$2" = "Xball"
```

使用例

if の例では次の3つの状態が調べられ、3つのすべてが真、または正常終了とみなされた場合、その結果が画面に出力されます。調べられる3つの内容は次のとおりです。

- 1 に設定されている変数の値が0より大きいか
- 2 に設定されている変数の値が2に等しいか
- root という語がテキストファイル /etc/passwd に含まれているか

/usr/bin/test

例1 /usr/bin/test の使用

ディレクトリが存在しなければmkdirを実行します。

```
test ! -d tempdir && mkdir tempdir
```

読み取り不可になるまで少し待つ。

```
while test -r thefile
do
    sleep 30
done
echo "thefile" is no longer readable'
```

引数が3つの文字列(2つは変形)のうちの1つであり、test コマンドの開く角括弧バージョン ([]) を使用していればコマンドを実行します。

```
if [ "$1" = "pear" ] || [ "$1" = "grape" ] || [ "$1" = "apple" ]
then
    command
fi
case "$1" in
    pear|grape|apple) command;;
esac
```

例2 /usr/bin/test での -e オプションの使用

実際に sh で -e オプションを使用する場合、次のように /usr/bin/test を使用します。

```
if [ ! -h $PKG_INSTALL_ROOT$rLink ] && /usr/bin/test -e
$PKG_INSTALL_ROOT/usr/bin/$rFile ; then
    ln -s $rFile $PKG_INSTALL_ROOT$rLink
fi
```

test 組み込みコマンド 以下の例では、Bourne シェルの if に続いて test 組み込みコマンドの 2 つの形式を記します。

例3 sh 組み込みコマンドの使用

```
ZERO=0 ONE=1 TWO=2 ROOT=root
```

```
if [ $ONE -gt $ZERO ]
```

```
[ $TWO -eq 2 ]
```

```
grep $ROOT /etc/passwd >&1 > /dev/null # discard output
```

```
then
```

```
    echo "$ONE is greater than 0, $TWO equals 2, and $ROOT is" \
        "a user-name in the password file"
```

```
else
```

```
    echo "At least one of the three test conditions is false"
fi
```

例4 test 組み込みコマンドの使用

test 組み込みコマンドの例

```
test `grep $ROOT /etc/passwd >&1 /dev/nul`l # discard output
```

```
echo $? # test for success
```

```
[ `grep nosuchname /etc/passwd >&1 /dev/nul`l ]
```

```
echo $? # test for failure
```

csh

例5 csh 組み込みコマンドの使用

```
@ ZERO = 0; @ ONE = 1; @ TWO = 2; set ROOT = root
```

```
grep $ROOT /etc/passwd >&1 /dev/null # discard output
```

```
    # $status must be tested for immediately following grep
```

```
if ( "$status" == "0" && $ONE > $ZERO && $TWO == 2 ) then
```

```
    echo "$ONE is greater than 0, $TWO equals 2, and $ROOT is" \
        "a user-name in the password file"
```

```
endif
```

ksh88

例6 ksh88/ksh 組み込みコマンドの使用

```
ZERO=0 ONE=1 TWO=$((ONE+ONE)) ROOT=root
```

```
if ((ONE > ZERO)) # arithmetical comparison
```

```
[[ $TWO = 2 ]] # string comparison
```

例6 ksh88/ksh 組み込みコマンドの使用 (続き)

```
[` grep $ROOT /etc/passwd >&1 /dev/nul`l ] # discard output
then
    echo "$ONE is greater than 0, $TWO equals 2, and $ROOT is" \
        "a user-name in the password file"

else
    echo "At least one of the three test conditions is false"
fi
```

環境変数 test の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 condition の評価結果は真だった。
- 1 condition の評価結果は偽だった、または condition が指定されていなかった。
- >1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/test、csh、ksh88、sh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

ksh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	不確実

関連項目 [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [test\(1B\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 -f オプションの代替である not-a-directory は BSD アプリケーションの移行を補助するものであり、将来のリリースではサポートされなくなる可能性があります。

XPG4 sh, ksh, ksh 次のような算術式を使用します。

```
$(( x > 3.1 )) #
```

次の代わりに


```
$ /usr/bin/test "$x" -gt 3.1 # )
```

2つの浮動小数点変数、または定数と浮動小数点変数を比較して、(base16からbase10への変換による)丸め誤差が結果に影響を与えることを避ける場合です。さらに、XPG4 sh、ksh88、およびkshでの組み込み演算のサポートは、比較するたびに文字列を明示的に変換する必要がないため、大幅に高速になります。

名前	test – condition evaluation command
形式	<i>/usr/ucb/test expression</i> <i>expression</i>
機能説明	test evaluates the expression <i>expression</i> and, if its value is true, sets 0 (true) exit status; otherwise, a non-zero (false) exit status is set. test also sets a non-zero exit status if there are no arguments. When permissions are tested, the effective user ID of the process is used. All operators, flags, and brackets (brackets used as shown in the second SYNOPSIS line) must be separate arguments to the test command; normally these items are separated by spaces.
使用法	
Primitives	The following primitives are used to construct <i>expression</i>: -r <i>filename</i> True if <i>filename</i> exists and is readable. -w <i>filename</i> True if <i>filename</i> exists and is writable. -x <i>filename</i> True if <i>filename</i> exists and is executable. -f <i>filename</i> True if <i>filename</i> exists and is a regular file. Alternatively, if /usr/bin/sh users specify /usr/ucb before /usr/bin in their PATH environment variable, then test will return true if <i>filename</i> exists and is (not-a-directory). This is also the default for /usr/bin/csh users. -d <i>filename</i> True if <i>filename</i> exists and is a directory. -c <i>filename</i> True if <i>filename</i> exists and is a character special file. -b <i>filename</i> True if <i>filename</i> exists and is a block special file. -p <i>filename</i> True if <i>filename</i> exists and is a named pipe (fifo). -u <i>filename</i> True if <i>filename</i> exists and its set-user- ID bit is set. -g <i>filename</i> True if <i>filename</i> exists and its set-group- ID bit is set. -k <i>filename</i> True if <i>filename</i> exists and its sticky bit is set. -s <i>filename</i> True if <i>filename</i> exists and has a size greater than zero. -t[<i>fildev</i>] True if the open file whose file descriptor number is <i>fildev</i> (1 by default) is associated with a terminal device. -z <i>s1</i> True if the length of string <i>s1</i> is zero. -n <i>s1</i> True if the length of the string <i>s1</i> is non-zero. <i>s1</i> = <i>s2</i> True if strings <i>s1</i> and <i>s2</i> are identical. <i>s1</i> != <i>s2</i> True if strings <i>s1</i> and <i>s2</i> are <i>not</i> identical.

- s1* True if *s1* is *not* the null string.
- n1* *-eq* *n2* True if the integers *n1* and *n2* are algebraically equal. Any of the comparisons *-ne*, *-gt*, *-ge*, *-lt*, and *-le* may be used in place of *-eq*.

Operators These primaries may be combined with the following operators:

- !* Unary negation operator.
- a* Binary *and* operator.
- o* Binary *or* operator (*-a* has higher precedence than *-o*).
- (expression)* Parentheses for grouping. Notice also that parentheses are meaningful to the shell and, therefore, must be quoted.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [find\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

注意事項 The `not-a-directory` alternative to the `-f` option is a transition aid for BSD applications and may not be supported in future releases.

If you test a file you own (the `-r`, `-w`, or `-x` tests), but the permission tested does not have the *owner* bit set, a non-zero (false) exit status will be returned even though the file may have the *group* or *other* bit set for that permission. The correct exit status will be set if you are super-user.

The `=` and `!=` operators have a higher precedence than the `-r` through `-n` operators, and `=` and `!=` always expect arguments; therefore, `=` and `!=` cannot be used with the `-r` through `-n` operators.

If more than one argument follows the `-r` through `-n` operators, only the first argument is examined; the others are ignored, unless a `-a` or `-o` is the second argument.

名前	tftp – 簡易ファイル転送プログラム
形式	tftp [<i>host</i> [<i>port</i>]]
機能説明	tftp は、ユーザーがリモートマシン間でファイルを転送できるインターネット TFTP (Trivial File Transfer Protocol) へのユーザーインタフェースです。リモートの <i>host</i> およびオプションの <i>port</i> は、コマンド行で指定できます。この場合、tftp では今後の転送で、<i>host</i> がデフォルトホストとして使用され、指定されていれば <i>port</i> がデフォルトポートとして使用されます。下記の connect コマンドを参照してください。
使用法	tftp が実行されると、プロンプト tftp> が表示され、次のコマンドが認識されます。
コマンド	connect <i>host-name</i> [<i>port</i>] 転送用の <i>host</i> および <i>port</i> (任意) を設定します。TFTP プロトコルは FTP プロトコルとは異なり、転送間で接続は保持されません。したがって、connect コマンドは実際に接続を作成せずに、転送用に使われるホストを記憶するだけです。connect コマンドを使用する必要はありません。リモートホストは get または put コマンドの一部として指定できます。 mode <i>transfer-mode</i> 転送モードを設定します。<i>transfer-mode</i> には、ascii または binary のいずれかを指定できます。デフォルト値は ascii です。 put <i>filename</i> put <i>localfile remotefile</i> put <i>filename1 filename2...filenameN remote-directory</i> 指定されたリモートファイルまたはディレクトリにファイル(またはファイルセット)を転送します。転送先は次の 2 つの形式で指定できます。ホストがすでに指定されている場合はリモートホスト上のファイル名、または <i>host:filename</i> <i>host</i> と <i>filename</i> の両方を同時に指定する場合は文字列の形式で指定します。後者の形式を使用する場合は、指定されたホスト名が今後の転送のデフォルトになります。remote-directory 形式を使用する場合は、リモートホストで UNIX システムが実行されていると見なされます。 <i>host</i> には、ホスト名 (hosts(4) を参照) か、IPv4 または IPv6 アドレス文字列 (inet(7P) または inet6(7P) を参照) を指定できます。IPv6 のアドレスには「:」が含まれているので、IPv6 アドレスを使用するときには <i>host</i> を角括弧で囲む必要があります。角括弧で囲まないと、最初のコロンが <i>host</i> と <i>filename</i> 間の区切り文字と見なされます。たとえば、次のように指定します。 [1080::8:800:200c:417A]:myfile ファイルがすでに存在し、公的に書き込み可能な場合にのみ、ファイルを書き込むことができます。in.tftpd(1M) を参照してください。

get filename

get remotename localname

get filename1 filename2 filename3...filenameN

指定されたりモートソースからファイルまたはファイルセット (3 つ以上) を取得します。source は次の 2 つの形式で指定できます。ホストがすでに指定されている場合はリモートホスト上のファイル名、または

host:filename

host と filename の両方を同時に指定する場合は文字列の形式で指定します。後者の形式を使用する場合は、最後に指定されたホスト名が今後の転送のデフォルトになります。host の指定については、put コマンドを参照してください。

quit

tftp を終了します。EOF も終了します。

verbose

詳細モードを切り替えます。

trace

パケットトレース機能を切り替えます。

status

現在のステータスを表示します。

rexmtretransmission-timeout

パケットごとの転送タイムアウトを秒単位で設定します。

timeouttotal-transmission-timeout

合計転送タイムアウトを秒単位で設定します。

ascii

mode ascii の省略形。

binary

mode binary の省略形。

blksize transfer-blocksize

サーバーとネゴシエーションを行うための転送ブロックサイズオプションの値。値 0 を指定すると、このオプションのネゴシエーションは無効になります。

srexmt server-retransmission-timeout

サーバーが使用するように要求するための転送タイムアウトオプションの値。値 0 を指定すると、このオプションのネゴシエーションは無効になります。

tsize

転送サイズオプションをサーバーに送信するように切り替えます。デフォルトでは、このオプションは送信されません。transfer-mode が ascii の場合は、転送サイズオプションは write 要求と同時に送信されません。

? [*command-name...*]

ヘルプ情報を出力します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	service/network/tftp

関連項目 [in.tftpd\(1M\)](#), [hosts\(4\)](#), [attributes\(5\)](#), [inet\(7P\)](#), [inet6\(7P\)](#)

Malkin, G., Harkin, A. 著、『RFC 2347, TFTP Option Extension』、The Internet Society 発行。1998 年 3 月

Malkin, G., Harkin, A. 著、『RFC 2348, TFTP Blocksize Option』、The Internet Society 発行。1998 年 3 月

Malkin, G., Harkin, A. 著、『RFC 2349, TFTP Timeout Interval and Transfer Size Options』、The Internet Society 発行。1998 年 3 月

『RFC 1350, The TFTP Protocol (Revision 2)』、Sollins, K.R. 著、Network Working Group 発行、1992 年 7 月

注意事項 デフォルトの *transfer-mode* は *ascii* です。これは SunOS 4.0 以前および 4.3BSD 以前のシステムとは異なるため、実行可能コマンドなどの ASCII 以外のバイナリファイルを転送するときは、明示的な操作を行う必要があります。

TFTP プロトコルにはユーザーログインや検証がないため、多くのリモートサイトがさまざまな方法でファイルアクセスを制限しています。承認されているファイルアクセス方法は各サイト固有のものであるため、このドキュメントでは説明できません。

get コマンドを使用してリモートホストから複数のファイルを転送する場合は、3 つ以上のファイルを指定する必要があります。2 つのファイルが指定されている場合は、2 番目のファイルがローカルファイルとして使用されます。

デフォルトのブロックサイズ 512 オクテットおよび 16 ビットブロック数を使用すると、一部の TFTP 実装において、サイズが 33,553,919 オクテット (32MB に 513 オクテット不足) を超えるファイルで問題が発生する可能性があります。Solaris 実装では、最大で 4GB のサイズのファイルを転送できます。

デフォルトでは、Solaris TFTP クライアントは *blocksize* または転送サイズオプションを有効化できません。*blocksize* オプションを高い値に設定すると、上限が 32MB のピアを処理する際の回避方法として役立つことがあります。

名前	time – 単純コマンドの時間測定
形式	time [-p] <i>utility</i> [<i>argument</i>]...
機能説明	time ユーティリティーは <i>utility</i> オペランドで指定されたユーティリティーを <i>argument</i> とともに呼び出し、<i>utility</i> の時間統計情報をまとめて標準エラー出力に書き出します。この情報には以下のものが含まれます。 ■ <i>utility</i> を呼び出してからその実行が終了するまでの経過 (実) 時間 ■ ユーザー CPU 時間。これは <i>utility</i> を実行したプロセスに対して <code>times(2)</code> 関数が返す <code>tms_utime</code> と <code>tms_cutime</code> の値の合計と同等の意味です。 ■ システム CPU 時間。これは <i>utility</i> を実行したプロセスに対して <code>times()</code> 関数が返す <code>tms_stime</code> と <code>tms_cstime</code> の値の合計と同等の意味です。 time をパイプラインの一部として使用した場合、それがパイプライン内のグループ分けコマンド中の 唯一のコマンドである場合を除き、報告される時間情報の内容は不確定です。以下の例を見てください。左側のコマンドの結果は不確定ですが、右側のコマンドは、上の例ではユーティリティー a について、下の例ではユーティリティー c について報告されます。 <pre>time a b c { time a } b c a b time c a b (time c)</pre>
オプション	次のオプションがサポートされています。 -p 時間情報を以下の形式で標準エラー出力に書き出します。 <pre>real %f\nuser %f\nsys %f\n < real seconds>, <user seconds>, <system seconds></pre>
オペランド	次のオペランドがサポートされています。 <i>utility</i> 呼び出すユーティリティーの名前 <i>argument</i> <i>utility</i> を呼び出す際に引数として与えるべき文字列
使用法	time ユーティリティーは、エラーが発生すると終了ステータス 127 を返します。これによりアプリケーションは、「ユーティリティーが見つからなかった」エラーと「呼び出したユーティリティーがエラーで終了した」ステータスとを見分けることができます。127 という値を選んだのは、通常この値はほかの意味で使われることがないためです。ほとんどのユーティリティーは、小さい値を使って「一般的なエラー状態」を表し、128 より大きな値は、シグナル受信による終了の場合と区別が困難になりかねないためです。同じような観点から、ユーティリティーは見つかったが実行できなかった、という場合に使う値として 126 が選ばれています。

使用例

例1 time コマンドの使用

time は、パイプラインまたは一連のコマンドに対して使用すると 便利ながことがよくあります。まずパイプラインやコマンド群を1つのファイルに入れ、それをユーティリティーとして呼び出します。それにより time をファイル中のすべてに適用させることができます。

他の方法として、以下のような使い方でも time を複雑なコマンドに適用できます。

example% **time sh -c 'complex-command-line'**

例2 Cシェルに組み込まれている time の使用

次の2つの例では、cshに組み込まれている time と /usr/bin/time との違いを示しています。現在使用中のシェルが csh であるとしします。

example% **time find / -name csh.1 -print**
/usr/share/man/man1/csh.1
95.0u 692.0s 1:17:52 16% 0+0k 0+0io 0pf+0w

time の出力形式に関する詳細は、[csh\(1\)](#) を参照してください。

example% **/usr/bin/time find / -name csh.1 -print**
/usr/share/man/man1/csh.1
real 1:23:31.5
user 1:33.2
sys 11:28.2

環境変数

time の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、LC_NUMERIC、NLSPATH、および PATH。

終了ステータス

utility で指定したユーティリティーが呼び出された場合、そのユーティリティーの終了ステータスが time の終了ステータスとなります。呼び出されなかった場合には、time は以下のいずれかの値を返して終了します。

1-125 time ユーティリティーの中でエラーが発生しました。

126 *utility* は見つかったが呼び出すことができませんでした。

127 *utility* は見つかりませんでした。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

属性タイプ	属性値
標準	standards(5) を参照してください。

- 関連項目

[csh\(1\)](#), [shell_builtins\(1\)](#), [timex\(1\)](#), [times\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)
- 注意事項

`time` コマンドをマルチプロセッサマシン上で実行した場合、出力された `user` と `sys` の値の合計が `real` 値を超える場合があります。これは、マルチプロセッサ・マシンでは、処理を複数のプロセッサに分けて実行することが可能なためです。

時間測定対象のコマンド処理中に割り込みが発生すると、出力される時間の値に誤差が生じることがあります。
- 使用上の留意点

経過時間の精度は秒単位までですが、CPU 時間は 100 分の 1 秒まで計測されます。そのため CPU 時間の合計値は経過時間を最大 1 秒上回る可能性があります。

名前	times – 現在のシェルの時間使用を報告するためのシェル組み込み関数
形式	
sh	times
ksh	times
機能説明	
sh	シェルから実行されたプロセスの、ユーザー時間およびシステム時間の累計値を表示します。
ksh	シェルおよびシェルから実行されたプロセスの、ユーザー時間およびシステム時間の累計値を表示します。 1つまたは2つのアスタリスク (*) が先頭に付加されている ksh(1) コマンドは、以下のような特殊な処理を受けます。 1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。 2. 入出力のリダイレクトは変数代入後に行われます。 3. エラーが発生すると、それを含むスクリプトは中止されます。 4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [ksh\(1\)](#), [sh\(1\)](#), [time\(1\)](#), [attributes\(5\)](#)

名前	timex - コマンドの時間測定、プロセスデータおよびシステムアクティビティの報告
形式	timex [-o] [-p [-fhkmrt]] [-s] <i>command</i>
機能説明	指定されたコマンドが実行されます。実行時に費やされた経過時間、ユーザー時間、およびシステム時間は秒単位で報告されます。オプションで、コマンドおよびそのすべての子に関するプロセスアカウンティングデータを一覧表示したり集約したりでき、実行間隔中の合計システムアクティビティを報告できます。 timex の出力は標準エラーに書き込まれます。
オプション	サポートしているオプションは、次のとおりです。 -o 読み取られたブロックまたは書き込まれたブロックの合計数、およびコマンドとそのすべての子によって転送された合計文字数を報告します。このオプションは、プロセスアカウンティングソフトウェアがインストールされている場合にのみ機能します。 -p コマンドとそのすべての子に関するプロセスアカウンティングレコードを一覧表示します。このオプションは、プロセスアカウンティングソフトウェアがインストールされている場合にのみ機能します。サブオプション f、h、k、m、r、および t は報告されるデータ項目を変更します。オプションは、次のとおりです。 -f 出力に <code>fork(2)</code>/<code>exec(2)</code> フラグとシステム終了ステータス情報を書き出します。 -h 平均メモリーサイズの代わりに、プロセスが実行中に使用した合計 CPU 時間の割合を表示します。この値は「ホッグ係数」と呼ばれ、(合計 CPU 時間)/(経過時間) で計算されます。 -k メモリサイズの代わりに、実行時に使われたキロバイト/分ごとのコアサイズの合計を表示します。 -m 平均コアサイズを表示します(デフォルト)。 -r CPU 係数を出力します。これは(ユーザー時間/(システム時間 + ユーザー時間)) で計算されます。 -t システム CPU 時間とユーザー CPU 時間を区別して出力します。読み取られたブロックまたは書き込まれたブロックの数、および転送された文字の数は常に報告されます。 -s コマンドの実行間隔中に発生した合計システムアクティビティを(コマンドによるアクティビティ以外も)報告します。<code>sar(1)</code>に表示されたすべてのデータ項目が報告されます。

使用例

例1 timex の例。

簡単な例は次のとおりです。

```
example% timex -ops sleep 60
```

サブシェルの時間を測定すると、任意に複雑な端末セッションを測定できます。

```
example% timex -opskmt sh
      session commands
EOT
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/accounting/legacy-accounting

関連項目

[sar\(1\)](#), [time\(1\)](#), [exec\(2\)](#), [fork\(2\)](#), [times\(2\)](#), [attributes\(5\)](#)

注意事項

プロセスシステムが使用できないため、コマンドに関連付けられたプロセスレコードは、インタフェースによってアカウンティングファイル `/var/adm/pacct` から選択されます。同じユーザー ID、端末 ID、および実行時間ウィンドウを含むバックグラウンドプロセスが誤って追加されます。

名前	tip - リモートシステムへ接続する
形式	<code>tip [-v] [-speed-entry] {hostname phone-number device}</code>
機能説明	tip ユーティリティはリモートホストに対し、全二重端末接続を確立します。いったん接続が確立されると、tip を使用したリモートセッションはローカル端末での対話型セッションのように振舞います。 remote ファイルは tip が使用するリモートシステムと回線速度のことを記述したエントリを含みます。 各ホストは接続に対してデフォルトのボーレートを持っています。特定のスピードを設定する場合は、<code>-speed-entry</code> コマンド行の引数として指定します。 <code>phone-number</code> が指定されると、tip は remote ファイルに次の形式のエントリを探します: <pre>tip -speed-entry</pre> tip がこのようなエントリを見つけた場合は、それに従って接続速度を設定します。そのようなエントリを見つからない場合は tip は、<code>-speed-entry</code> をシステム名のように解釈し、エラーメッセージを出します。 <code>-speed-entry</code> を省略した場合、tip は <code>tip0</code> エントリを使用して接続速度を設定します。 <code>device</code> が指定されると、tip はそのデバイスを開こうとします。しかしその際、tip の通常アクセス特権 (<code>setuid uucp</code>) ではなく、ユーザー特権でアクセスします。そのユーザーはそのデバイスへの読み取り/書き込み権限がなければなりません。tip ユーティリティはどんなキャラクタ文字列でもスラッシュ文字 (<code>/</code>) で始まるものをデバイス名として解釈します。 接続を確立すると、tip は接続メッセージをリモートシステムへ送信します。メッセージのデフォルト値は remote ファイルにあります。 リモートシステムに接続を試みる際、tip は、<code>exclusive-open ioctl(2)</code> 呼び出しで関連デバイスをオープンします。このように、一度に一ユーザーのみがデバイスにアクセスできます。これは端末回線をマルチプロセスがサンプリングすることを回避するためです。さらに、tip は <code>uucp(1C)</code> の使用するロッキングプロトコルを尊重します。 tip が起動すると、それはファイル <code>.tiprc</code> からユーザーのホームディレクトリへコマンドを読み込みます。
オプション	<code>-v</code> 実行に従い、 <code>.tiprc</code> ファイルからコマンドを表示します。

使用法	入力した文字列は通常エコーされることなく直接リモートマシンへ転送されます。 tip がプロンプトを表示する場合 (たとえば、ファイル転送の設定中に)、入力した行は標準の削除および kill 文字で編集できます。プロンプトに対して空行を入力したり、中断を実行したりすると、対話が終了してリモートマシンへ戻ります。
コマンド	行の最初にあるチルダ(~)はエスケープ信号で、tip に何か特別の動作をさせます。tip は次のエスケープシーケンスを理解します: ~^D ~. 接続を破棄し終了する (ユーザーはまだリモートマシンにログイン中)。注意: rlogin した上でリモートホスト上で tip を実行中の場合、tip セッションを終了するために ~. (チルダチルダドット) を入力しなければなりません。~. (チルダドット) を入力すると、rlogin を終了してしまいます。 ~c [name] ディレクトリを name に変えます。何の引数も与えなければ自分のホームディレクトリへ移ります。 ~! ローカルマシンで対話型シェルにエスケープします。シェルから抜けると tip へ戻ります。 ~> ローカルからリモートへファイルをコピーします。 ~< リモートからローカルへファイルをコピーします。 ~p from [to] UNIX システムを実行しているリモートホストへファイルを送ります。put コマンドを使用すると、リモートシステムはコマンド文字列を実行します。 cat > to tip がそれを from ファイルへ送ります。to ファイルを指定していなければ、from ファイル名が使用されます。このコマンドは実際 UNIX システム固有バージョンの '~>' コマンドです。 ~t from [to] UNIX システムを実行しているリモートホストからファイルを取得します。put コマンドにおける to ファイルのようにファイル名を指定しなければ from ファイル名が標準になります。リモートホストは以下のコマンド行を実行して cat from ; echo ^A ファイルを tip へ送ります。 ~ リモートコマンドの出力をローカルプロセスへパイプします。ローカルシステムへ送られたコマンド文字列はシェルで処理されます。

- ~C プログラムをリモートマシンへ接続します。プログラムへ送られた
 コマンド文字列はシェルで処理されます。プログラムはファイル記
 述子 0 をリモート入力行、1 をリモート出力行、2 を tty 標準エ
 ラーとして継承します。
- ~\$ ローカルプロセスの出力をリモートホストへパイプします。ローカ
 ルシステムへ送られたコマンド文字列はシェルで処理されます。
- ~# BREAK をリモートシステムへ送ります。
- ~s 変数を設定します(下記参照)。
- ~^Z tip を終了します。C シェルのようにジョブ制御をサポートする
 シェルの元で実行した場合にのみ使用可能です。
- ~^Y tip の "local side" のみを停止します。C シェルのようにジョブ制御
 をサポートするシェルの元で実行した場合にのみ使用可能で
 す。tip の "remote side"、すなわち、リモートホストからの出力を表
 示する側面は実行し続けるといことです。
- ~? チルダエスケープのサマリーを取得します。

ファイルのコピーにはリモートホストの協力をいくらか必要とします。ファイルを送るために ~> または ~< エスケープを使用する場合、tip は (送信または受信する) ファイル名を求めます。そしてリモートホストからファイルを受信する場合、リモートホストへコマンドが送信されます。tip がファイルを転送中、転送行数はスクリーンに表示され続けます。ファイル転送は割り込みによって中断されます。

Auto-call ユニット

tip は、多くの auto-call ユニット (ACU) を使用したリモートホストへのダイアルアップに使用することができます。リモートシステムの説明が du 機能を含んでいれば、tip は提供された call ユニット (cu)、ACU タイプ (at)、電話番号 (pn) を使用します。通常、tip ダイアル中に詳細なメッセージを表示します。

接続の確立に使用した auto-dialer の種類によっては、接続時に、リモートホストに文字化けした内容が送信されているかもしれません。別ホストに向けて最初に入力した文字が必ず最初に届くとは限りません。推奨されるやり方は接続確立したところでただちに kill 文字を入力することです。(ほとんどの UNIX システムは @ か Control-U を初期 kill 文字としてサポートしています)。

tip は今のところ Ventel MD-212+ モデムと DC Hayes 互換モデムをサポートしています。

tip がダイアルするために Hayes 互換モデムを初期化すると、それはモデムを auto-answer に設定します。通常、通信が終了すると、tip は DTR を落としモデムを "hang up" させます。

ほとんどのモデムはDTRが落ちたとき、自分でプログラム以前の状態に再初期化するように、構成できます。必要な場合は、この方法でモデムをリセットし auto-answer を無効にできます。

さらに、電話番号をヘイズの s コマンドで開始して、モデムをダイヤル前に構成することができます。たとえば、auto-answer を無効にするには、pn=S0=0DT5551212 のようにして全電話番号を /etc/remote の中に設定できます。S0=0 は auto-answer を無効にします。

リモートホストの説明

リモートホストの説明は通常 system-wide ファイル /etc/remote の中にあります。しかし、ユーザーは個人的な説明のファイル(と電話番号を) シェル変数 REMOTE に定義してエクスポートすることもできます。remote は tip によって読み取り可能でなければなりません。しかし電話番号を記述したセカンダリファイルはユーザーによってのみ読み取り可能でも構いません。このセカンダリ電話番号ファイルはシェル変数 PHONES が定義されエクスポートされない限り /etc/phones です。電話番号ファイルは以下の書式に従った行を含みます:

system-name phone-number

システムのための各電話番号は接続を確立するかファイルの終わりに到達するまで試されます。電話番号は '0123456789-=*' で構成されます。'=' と '*' は第二発信音を (ACU に応じて) 待つべきであるということを示すために使用されます。

tip の内部変数

tip は通常の操作に使用する変数のセットを持っています。これらの変数のいくつかは一般ユーザーには読み取り専用です。(root ユーザーは何でも変更することができます)。変数は ~s エスケープで表示、設定できます。変数の構文は vi(1) と mail(1) にならって作られています。all を ~s エスケープの引数として与えることでユーザーが読むことの出来るすべての値が表示されます。その代わりに、? ユーザーは最後に ? をつけることで指定の値を表示するよう要求できます。たとえば、'-s escape?' は現在のエスケープ文字を表示します。

値は数値 (num)、文字列 (str)、文字型 (char) またはブール (bool) 値です。ブール値は単に名前を特定するためにのみ設定されます。それらは名前の先頭に ! を追加することでリセットされるかもしれません。他の変数は = と値を後ろに付けることで設定します。全割り当てに空白があってははいけません。1 セットのコマンドで値の数値を調べたり設定したりできます。

変数は実行時に設定コマンドを (~s を前につけることなしに) ユーザーのホームディレクトリにある .tiprc ファイルに記述することで初期化できます。-v オプションは tip が設定されたとおりに表示します。# の印をつけたコメントを .tiprc ファイルの中に記述できます。

最後に、変数名は完全に指定するか省略形も指定できます。次に示すのは tip の理解できる変数です。

beautify	(bool) セッションが用意されると出力不可能な文字列を破棄します。省略形 be。nb 能力がある場合は、beautify はまず off に設定されます。そうでない場合は、beautify はまず on に設定されます。
baudrate	(num) 接続確立時のボーレート。省略形は ba。ボーレートがコマンド行で指定された場合は、baudrate はまず指定した値に設定されます。または、br 能力がある場合は、baudrate はまずその能力に合わせて設定されます。そうでない場合は、baudrate は 300 ボーに設定されます。いったん tip が起動すると、baudrate はスーパーユーザーのみが変更できます。
dialtimeout	(num) 電話番号をダイヤルしているとき、接続が確立するまでに待つ時間(秒)。省略形は dial。dialtimeout はまず 60 秒に設定され、スーパーユーザーのみが変更できます。
disconnect	(str) 切断するためにリモートホストへ送る文字列。省略形は di。di 能力がある場合は、disconnect はまずその能力に合わせて設定されます。そうでない場合は、disconnect は NULL 文字列("")に設定されます。
echocheck	(bool) 転送された最終文字のエコーを待つことでファイル転送中にリモートホストと同期します。省略形は ec。ec 能力がある場合は、echocheck はまず on に設定されます。そうでない場合は、echocheck はまず off に設定されます。
eofread	(str) ファイル転送コマンド ~< の最中に転送終了を示す文字列。省略形は eofr。ie 能力がある場合は、eofread はまず値をその能力に設定します。そうでない場合は、eofread は NULL 文字列("")に設定されます。
eofwrite	(str) 文字列はファイル転送コマンド ~> の最中に転送終了を示す文字列。省略形は eofw。oe 能力がある場合は、eofread はまず値をその能力に設定します。そうでない場合は、eofread は NULL 文字列("")に設定されます。
eol	(str) 行の終わりを示す文字列。tip は end-of-line の後でのみエスケープ文字を認識します。el 能力がある場合は、eol はまずその能力に合わせて設定されます。そうでない場合は、eol は NULL 文字列("")に設定されます。
escape	(char) (escape) 文字列を頭につけるコマンド。省略形は es。es 能力がある場合は、escape はまずその能力に合わせて設定されます。そうでない場合、escape は '~' に設定されます。
etimeout	(num) echocheck が設定されているとき、tip がエコーチェックの応答を待たなければならない時間(秒単位)。省略形は et。et 能力

	がある場合は、 <code>etimetype</code> はまずその能力に合わせて設定されます。そうでない場合は、 <code>etimetype</code> は 10 秒に設定されます。
<code>exceptions</code>	(str) <code>beautify</code> スイッチによって破棄してはいけない文字列。省略形は <code>ex</code> 。 <code>ex</code> 能力がある場合は、 <code>exceptions</code> はまずその能力に合わせて値を設定します。そうでない場合は、 <code>exceptions</code> は <code>'\t\n\f\b'</code> に設定されます。
<code>force</code>	(char) リテラルデータ転送を強制するための文字列。省略形は <code>fo</code> 。 <code>fo</code> 能力がある場合は、 <code>force</code> はまず値をその能力に合わせて設定します。そうでない場合は、 <code>force</code> は <code>\377</code> (使用不能) に設定されます。
<code>framesize</code>	(num) ファイル受信時にファイルシステム書込み間でバッファーするデータ量 (バイト単位)。省略形は <code>fr</code> 。 <code>fs</code> 能力がある場合は、 <code>framesize</code> はまず値をその能力に合わせて設定します。そうでない場合は、 <code>framesize</code> は 1024 に設定されます。
<code>halfduplex</code>	(bool) ホストが半二重なのでローカルエコーを行います。省略形は <code>hdx</code> 。 <code>hd</code> 能力がある場合は、 <code>halfduplex</code> はまず <code>on</code> に設定されます。そうでない場合は、 <code>halfduplex</code> はまず <code>off</code> に設定されます。
<code>hardwareflow</code>	(bool) ハードウェアフロー制御を行います。省略形は <code>hf</code> 。 <code>hf</code> 能力がある場合は、 <code>hardwareflow</code> はまず <code>on</code> に設定されます。そうでない場合は、 <code>hardwareflowcontrol</code> はまず <code>off</code> に設定されます。
<code>host</code>	(str) 接続したホストの名前。省略形は <code>ho</code> 。 <code>host</code> はコマンド行上で与えられるか <code>HOST</code> 環境変数で設定した名前に恒久的に設定されません。
<code>localecho</code>	(bool) <code>halfduplex</code> と同義。省略形は <code>le</code> 。
<code>log</code>	(str) 外向けの電話かけに関するログ情報ファイルの名前。 <code>log</code> はまず <code>/var/adm/aculog</code> に設定されます。それからスーパーユーザーのみが調査または変更できます。
<code>parity</code>	(str) リモートホストに更新するときパリティが生成されチェックされます。 省略形 <code>par</code> 。 指定できる値は次のとおりです。 <code>none</code> > <code>zero</code> 入力ではパリティはチェックされません。そして出力ではパリティはゼロに設定されます。 <code>one</code> 入力ではパリティはチェックされません。そして出力ではパリティビットが設定されます。 <code>even</code> 入力では偶数パリティがチェックされ、出力では生成されます。

<code>odd</code>	入力では奇数パリティがチェックされ、出力では生成されます。
<code>pa</code>	能力がある場合は、 <code>parity</code> はまず値をその能力に設定します。そうでない場合は、 <code>parity</code> は <code>none</code> に設定されます。
<code>phones</code>	隠し電話番号を入れておくファイル。もし環境変数 <code>PHONES</code> が設定されていれば、 <code>phones</code> は <code>PHONES</code> の値に設定されます。そうでない場合は、 <code>phones</code> は <code>/etc/phones</code> に設定されます。 <code>phones</code> の値は <code>tip</code> から変更することはできません。
<code>prompt</code>	(char) リモートホストで「行の終端」を示す文字列。省略形 <code>pr</code> 。この値はデータ転送中、同期を行うために使用されます。ファイル転送コマンド中の行数はこの文字列の受領によって確定します。 <code>pr</code> 能力がある場合は、 <code>prompt</code> はまず値をその能力に合わせます。そうでない場合は、 <code>prompt</code> は <code>\n</code> に設定されます。
<code>raise</code>	(bool) 大文字マッピングモード。省略形は <code>ra</code> 。このモードが有効化されていると、リモートマシンとの転送においてすべての文字は <code>tip</code> によって大文字に割り当てられます。 <code>ra</code> 能力がある場合は、 <code>raise</code> はまず <code>on</code> に設定されます。そうでない場合は、 <code>raise</code> はまず <code>off</code> に設定されます。
<code>raisechar</code>	(char) 大文字マッピングモードを切り替える入力文字。省略形は <code>rc</code> 。 <code>rc</code> 能力がある場合は、 <code>raisechar</code> はまずその能力に合わせて設定されます。そうでない場合は、 <code>raisechar</code> は <code>\377</code> (使用不能) に設定されます。
<code>rawftp</code>	(bool) ファイル転送中に全文字列を送信します。印刷不可能文字をフィルタせず、 <code>\n</code> を <code>\r</code> にのような変換を行わない。省略形は <code>raw</code> 。 <code>rw</code> 能力がある場合は、 <code>rawftp</code> はまず <code>on</code> に設定されます。そうでない場合は、 <code>rawftp</code> はまず <code>off</code> に設定されます。
<code>record</code>	(str) セッションスクリプトが記録されるファイル名。省略形は <code>rec</code> 。 <code>re</code> 能力がある場合は、 <code>record</code> はまずその能力に合わせて設定されます。そうでない場合は、 <code>record</code> は <code>tip.record</code> に設定されます。
<code>remote</code>	リモートシステム説明を記載したファイル。環境変数 <code>REMOTE</code> が設定されている場合、 <code>remote</code> は <code>REMOTE</code> の値に設定されます。そうでない場合は、 <code>remote</code> は <code>/etc/remote</code> に設定されます。 <code>remote</code> の値は <code>tip</code> で変更することはできません。
<code>script</code>	(bool) セッションスクリプトモード。省略形は <code>sc</code> 。 <code>script</code> が <code>on</code> の時、 <code>tip</code> はリモートホストによって転送されたすべてを <code>record</code> の中に指定されたスクリプト記録ファイルに記録します。 <code>beautify</code> スイッチが <code>on</code> の場合、ASCII 文字のみがスクリプトファイルに含まれます。(これらの文字は 040 と 0177 の間です)。値 <code>exceptions</code>

は通常の `beautify` ルールにはずれる文字を示すために使用します。 `sc` 能力がある場合は、 `script` はまず `on` に設定されます。そうでない場合は、 `script` はまず `off` に設定されます。

<code>tabexpand</code>	(bool) ファイル転送中 TAB 文字を SPACE 文字に展開します。省略形 <code>tab</code> 。 <code>tabexpand</code> が <code>on</code> のとき、各 <code>tab</code> は 8 つの SPACE 文字に展開されます。 <code>tb</code> 能力があれば <code>tabexpand</code> はまず <code>on</code> に設定されます。そうでない場合は、 <code>tabexpand</code> はまず <code>off</code> に設定されます。
<code>tandem</code>	(bool) リモートホストから送られるデータのレートを制限するために <code>XON/XOFF</code> フロー制御を使用します。省略形は <code>ta</code> 。 <code>nt</code> 能力がある場合は、 <code>tandem</code> はまず <code>off</code> に設定されます。そうでない場合は、 <code>tandem</code> はまず <code>on</code> に設定されます。
<code>verbose</code>	(bool) 冗長モード。省略形は <code>verb</code> 。冗長モードが有効な場合、 <code>tip</code> はダイアル中にメッセージを、ファイル転送操作中に現在の電話番号を、さらに多くを表示します。 <code>nv</code> 能力がある場合は、 <code>verbose</code> はまず <code>off</code> に設定されます。そうでない場合は、 <code>verbose</code> はまず <code>on</code> に設定されます。
<code>SHELL</code>	(str) <code>~!</code> コマンドに使用されるシェル。標準値は <code>/bin/sh</code> です、そうでない場合は環境から取得されます。
<code>HOME</code>	(str) <code>~c</code> コマンドの使用するホームディレクトリ。標準値は環境から取得されます。

使用例

例1 `tip` コマンドを使う

ファイル転送に使用するダイアログの例は以下の通りです。

```
arpa% tip monet
[connected]
...(assume we are talking to a UNIX system)...
ucbmonet login: sam
Password:
monet% cat sylvester.c
-> Filename: sylvester.c
32 lines transferred in 1 minute 3 seconds
monet%
monet% ~< Filename: reply.c
List command for remote host: cat reply.c
65 lines transferred in 2 minutes
monet%
...(or, equivalently)...
monet% ~p sylvester.c
...(actually echoes as ~[put] sylvester.c)...
32 lines transferred in 1 minute 3 seconds
monet%
monet% ~t reply.c
```

例1 tip コマンドを使う (続き)

```
... (actually echoes as ~[take] reply.c)...
65 lines transferred in 2 minutes
monet%
... (to print a file locally)...
monet% ~|Local command: pr h sylvester.c | lpr
List command for remote host: cat sylvester.c
monet% ~^D
[EOT]
... (back on the local system)...
```

環境変数

次の環境変数は tip によって読み込まれます。

REMOTE remote ファイルの場所。
 PHONES 個人電話番号が記載されているファイルの場所。
 HOST デフォルト接続先ホスト。
 HOME ユーザーのログインディレクトリ(chdirs のために)。
 SHELL ‘~!’ エスケープでフォークするシェル。

ファイル

/etc/phones
 /etc/remote
 /var/spool/locks/LCK.* UUCP との競合を避けるためのロックファイル
 /var/adm/aculog 外向けの発信がログされるファイル
 ~/.tiprc 初期化ファイル

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[cu\(1C\)](#), [mail\(1\)](#), [uucp\(1C\)](#), [vi\(1\)](#), [ioctl\(2\)](#), [attributes\(5\)](#)

使用上の留意点

2つの追加変数 chardelay と linedelay があるが現在は実装していない。

名前	touch, settime – ファイルのアクセス日時および更新日時の変更
形式	<pre>touch [-acm] [-r ref_file -t time -d date_time] file... touch [-acm] [time_spec] file... settime [-f ref_file] [time_spec] file...</pre>
機能説明	touch ユーティリティーは、各ファイルのアクセス日時または更新日時を設定します。指定した <i>file</i> が存在しない場合には、<i>file</i> が作成されます。 新たに設定する日時は、<i>-t time</i>、<i>-d date_time</i>、<i>-r ref_file</i> に指定したファイルの時間フィールド、または <i>time_spec</i> オペランドによって指定できます。これらのオプションを1つも指定しない場合、touch は現在の日時を使用します。 <i>-a</i> オプションと <i>-m</i> オプションの両方が省略された場合、touch は更新日時とアクセス日時の両方を更新します。 ファイルへの書き込み権を持つが、そのファイルの所有者ではなく、スーパーユーザーでもないユーザーは、ファイルの更新日時とアクセス日時を現在の日時にだけ変更できます。touch で特定の日時に設定しようとするエラーになります。 settime ユーティリティーは、touch <i>-c [time_spec] file</i> と同等です。
オプション	touch および settime には、次のオプションがサポートされています。
touch	touch には、次のオプションがサポートされています。 <i>-a</i> <i>file</i> のアクセス日時だけを変更します。<i>-m</i> オプションも同時に指定された場合を除き、更新日時は変更しません。 <i>-c</i> <i>file</i> が示す名前のファイルが存在しない場合、そのファイルを作成しません。また、この状態についての診断メッセージを出力しません。 <i>-d date_time</i> 現在の日時の代わりに <i>date_time</i> を使用します。オプション引数には次の形式の文字列を指定してください。 <pre>YYYY-MM-DDThh:mm:ss[.frac][tz]</pre>または <pre>YYYY-MM-DDThh:mm:ss[,frac][tz]</pre>各情報の意味は次のとおりです。 - YYYY には、西暦年を表す4桁の10進数を指定する必要があります。 - MM、DD、hh、mm、およびssは、<i>-t time</i> と同様です - T には文字Tまたは1つのスペース文字を指定します。

- `[frac]` および `[,frac]` には、空あるいはそれぞれピリオド (.) またはコンマ (,) のあとに 1 桁以上の 10 進数を続けて、小数点以下の秒を指定します
 - `[tz]` は、ローカルの日時を示す空にするか、UTC を示す文字 `z` にします。`[tz]` を空にすると、結果として得られる日時の値は環境変数 `TZ` の値の影響を受けます。
- m**
file の更新日時だけを変更します。**-a** オプションも同時に指定しなければ、アクセス日時は更新しません。
- r ref_file**
 現在の日時の代わりに、*ref_file* に指定されたファイルの日時を使用します。
- t time**
 設定する日時として、現在の日時の代わりに *time* が示す値を使用します。*time* は、次の形式の 10 進数です。
- [[CC]YY]MMDDhhmm[.SS]
- 各 2 桁の値は次のような意味を持ちます。
- MM*
 月 [01-12]
- DD*
 日 [01-31]
- hh*
 時 [00-23]
- mm*
 分 [00-59]
- CC*
 西暦年の上 2 桁
- YY*
 西暦年の下 2 桁
- SS*
 秒 [00-61]

`CC` と `YY` はともに省略可能です。両方とも省略すると、現在の年と見なされます。`YY` を指定して `CC` を省略すると、`CC` は次に示す値と見なされます。

YY の値	CC のデフォルト
69-99	19
00-38	20

YYの値	CCのデフォルト
39-68	エラー

結果として得られる日時の値は、環境変数 TZ の値の影響を受けます。許される日時の値の範囲は、1970 年 1 月 1 日 0 時 0 分 0 秒 (グリニッジ標準時) 以後から 2038 年 1 月 18 日までです。

SS の範囲は、[00-59] ではなく [00-61] です。これはうるう秒を考慮しているためです。SS が 60 または 61、TZ により加工されたあとの時間の値がうるう秒を表していない場合、結果の時間値は SS を 59 とした時間の 1 秒後または 2 秒後となります。SS を省略すると 0 と見なされます。

settime ユーティリティーには、次のオプションがサポートされています。

-f *ref_file*
現在の日時の代わりに、*ref_file* に指定されたファイルの日時を使用します。

オペランド touch および settime には、次のオペランドがサポートされています。

file
日時を変更するファイルのパス名

time_spec
設定する日時として、現在の日時の代わりに *time_spec* が示す値を使用します。このオペランドは、次の形式の 10 進数です。

MMDDhhmm[YY]
各 2 桁の値は次のような意味を持ちます。

MM
月 [01-12]

DD
日 [01-31]

hh
時 [00-23]

mm
分 [00-59]

YY
西暦年の下 2 桁

YY は省略可能です。これを省略すると、現在の年と見なされます。YY を指定すると、年 は次に示す値と見なされます。

YY	対応する年
69-99	1969-1999
00-38	2000-2038
39-68	エラー

-d、-r、または -t オプションを省略し、少なくとも 2 つのオペランドを指定し、そのうちの先頭のオペランドの値が 8 または 10 桁の 10 進数である場合、その先頭オペランドは *time_spec* であると見なされます。それ以外の場合には、先頭オペランドは *file* であると見なされます。

使用法 ファイルが 2G バイト (2³¹ バイト) 以上ある場合の touch の動作については、[largefile\(5\)](#) を参照してください。

環境変数 touch の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

TZ
time または date_time オプション引数や time_spec オペランドに適用するタイムゾーンを指定します。

終了ステータス 次の終了ステータスが返されます。

- 0
touch の実行が正常終了し、要求されたすべての変更が行われた。
- >0
エラーが発生した。touch は、日時の変更を実施できなかったファイルの数を返す。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [futimens\(2\)](#), [stat\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項

BSD 環境に精通しているユーザーであれば、`-f` オプションが `touch` に受け入れられたのに `f` オプションが無視されることに気づきます。ファイル上のアクセス権にかかわらず、`touch` はユーザーが所有するすべてのファイルに対して成功するので、`-f` オプションは不要です。

名前	touch – change file access and modification times
形式	/usr/ucb/touch [-acfm] file...
機能説明	touch sets the access and modification times of each file to the current time. file is created if it does not already exist.
オプション	-a Change the access time of file. Do not change the modification time unless -m is also specified. -c Do not create file if it does not exist. -f Attempt to force the touch in spite of read and write permissions on file. -m Change the modification time of file. Do not change the access time unless -a is also specified.
使用法	See largefile(5) for the description of the behavior of touch when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
終了ステータス	The following exit values are returned: 0 touch executed successfully and all requested changes were made. >0 An error occurred. touch returns the number of files for which the times could not be successfully modified.
属性	See attributes(5) for descriptions of the following attributes:
関連項目	touch(1) , attributes(5) , largefile(5)

名前 tplot, t300, t300s, t4014, t450, tek, ver – さまざまなプロッタ用のグラフィックスフィルタ

形式 /usr/bin/tplot [-T *terminal*]

機能説明 tplot は、標準入力からプロット命令を読み取り、標準出力で特定の端末に適したプロット命令を生成します。

端末が指定されていない場合、環境変数 **TERM** が使用されます。デフォルトの端末は **tek** です。

ファイル /usr/lib/t300
/usr/lib/t300s
/usr/lib/t4014
/usr/lib/t450
/usr/lib/tek
/usr/lib/vplot

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [vi\(1\)](#), [attributes\(5\)](#)

名前	tput – 端末の初期化または terminfo データベースの照会
形式	<pre>tput [-T type] capname [parm]... tput -S <<</pre>
機能説明	tput は、terminfo データベースを使って、端末に依存した機能やシェルに渡される情報 (sh(1) を参照) の値を決定するか、端末をクリア、初期化、またはリセットするか、または要求された端末タイプの長い名前 (正式な名前) を返します。capname 引数で指定された機能属性が文字列型であれば、tput は文字列を出力します。同様に、属性が整数型であれば整数を出力します。属性がブール型であれば、tput は単に終了ステータスを設定するだけで何も出力しません。この場合の終了ステータスの値は、端末がその機能を持っていれば真を示す 0 であり、持っていない場合は偽を示す 1 となります。標準出力に返された値を使用する前に、必ず終了ステータス (\$?, sh(1) を参照) の値が 0 であることを確認してください。詳細については後述の「終了ステータス」の項を参照してください。
オプション	サポートしているオプションは、次のとおりです。 -Ttype 端末の種類を type 引数で指定します。通常は、デフォルトの種類が TERM 環境変数から得られるので、このオプションは必要ありません。-T オプションを指定すると、シェル変数の LINES と COLUMNS、およびレイヤーサイズは参照されません。 -S 1 回の tput の呼び出しで複数の機能指定を可能にするオプションです。この場合、対象となる機能はコマンド行からではなく標準入力から tput に渡すこととなります。「使用例」の項にある例を参照してください。1 行には 1 つの capname しか指定できません。なお、-s オプションを指定した場合、ブール型および文字列型の終了ステータス 0 と 1 の意味が変わります。詳しくは「終了ステータス」の項を参照してください。
オペランド	次のオペランドがサポートされています。 capname terminfo データベースから得られる、機能属性を表します。機能とそれに対応する capname の一覧については、terminfo(4) を参照してください。 C ロケールを使ったシステムでは、以下の文字列がオペランドとしてサポートされています。 clear クリアスクリーンのシーケンスを表示します。 init terminfo データベースが存在し、ユーザーの端末用のエントリが存在していれば (前述の -Ttype の説明を参照)、以下のすべての処理が実行されます。 1. 端末の初期化文字列が存在すれば、それが出力される (is1, is2, is3, if, iprog)

2. エントリ中の遅延指定 (たとえば復帰改行) が、tty ドライバに設定される
3. エントリ中の指定に従い、タブの展開指定を有効または無効にする
4. タブを展開しない場合、標準タブ (8 文字ごと) が設定される。これら 4 つの処理を行うための情報がエントリ中に存在しなければ、その該当処理はスキップされますが、それを表すメッセージは出力されません。

reset 初期化文字列を出力する代わりに、端末のリセット文字列が存在すればそれを出力します (rs1、rs2、rs3、rf)。リセット文字列が存在せず、初期化文字列が存在している場合には、その初期化文字列が出力されます。それ以外の場合には、reset の動作は init と同一です。

longname terminfo データベースが存在し、ユーザーの端末用のエントリが存在していれば (前述の `-Ttype` の説明を参照)、端末の長い名前が出力されます。ここでいう長い名前とは、terminfo データベース中の当該端末の記述における、先頭行の最後にならされている名前です ([term\(5\)](#) を参照)。

parm 前述の *capname* で指定した属性がパラメータを伴う文字列の場合、この **parm** で指定する値が文字列に代入されます。数字で表された引数は、すべて数値として属性に渡されます。

使用例

例1 TERM に従った端末の初期化

最初の例は、環境変数 `TERM` が示す端末タイプに従って端末を初期化する処理です。なお、[profile\(4\)](#) マニュアルページで説明されているように、このコマンドは (環境変数 `TERM` をエクスポートしたあと) どのユーザーの `.profile` にも入れておくべきコマンドです。

```
example% tput init
```

例2 端末の初期化

次の例は、AT&T 5620 端末を初期化して、環境変数 `TERM` が示す端末タイプを置き換えるコマンドです。

```
example% tput -T5620 reset
```

例3 カーソルの移動

次の例は、画面上の行番号 0 列番号 0、つまり左上角 (通常ホームポジションと呼ばれる位置) にカーソルを移動するためのシーケンスを送るコマンドです。

例3 カーソルの移動 (続き)

```
example% tput cup 0 0
```

次の例は、カーソルを 23 行 4 列に移動させるシーケンスを送信するコマンドです。

```
example% tput cup 23 4
```

例4 画面をクリアするシーケンスのエコー

次の例は、使用している端末の画面をクリアするシーケンスをエコーさせるコマンドです。

```
example% tput clear
```

例5 列の数の表示

次の例は、使用している端末の列の数を表示するコマンドです。

```
example% tput cols
```

次の例は、450 型端末の列の数を表示するコマンドです。

```
example% tput -T450 cols
```

例6 シェル変数の設定

次の例は、使用している端末を、シェル変数 **bold** を強調モード・シーケンスを開始する値に設定し、**offbold** を強調モード・シーケンスを終了する値に設定するコマンドです。このコマンドを実行すると、以下のプロンプトが出力されることがあります。

```
echo "${bold}Please type in your name: ${offbold}\c"
```

```
example% bold='tput smso'
```

```
example% offbold='tput rmso'
```

例7 終了ステータスの設定

次の例は、使用している端末がハードコピー端末であるかどうかを示す終了ステータスを設定するコマンドです。

```
example% tput hc
```

例8 terminfo からの長い名前の表示

次の例は、環境変数 **TERM** が示す端末タイプに対応した長い名前を terminfo データベースから検索して表示するコマンドです。

```
example% tput longname
```

例9 1回の呼び出しでの複数機能の処理

最後の例は、1回の呼び出しで複数の機能処理する `tput` コマンドです。この例では、画面をクリアし、カーソルを10行10列の位置に移動させ、ボールド(超高輝度)モードをオンに設定するという一連の処理を実行しています。処理の記述の終わりは、感嘆符(!)だけの行で表します。

```
example% tput -S <<!  
> clear  
> cup 10 10  
> bold  
> !
```

環境変数

`tput` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

TERM 端末のタイプを決定します。この変数が設定されていないかあるいはNULLに設定されていて、`-T` オプションが省略された場合、デフォルトの端末タイプ(予測できない)が用いられます。

終了ステータス 次の終了ステータスが返されます。

0

- *capname* がブール型で `-s` オプションが省略された場合、真を表す。
- *capname* が文字列型で `-s` オプションが省略された場合、*capname* がこの端末タイプに対して定義されていることを表す。
- *capname* がブール型または文字列型で `-s` が指定された場合、すべての行が正常に処理されたことを表す。
- *capname* が整数型であることを表す。
- 要求された文字列が正しく書き込まれたことを表す。

1

- *capname* がブール型で `-s` オプションが省略された場合、偽を表す。
- *capname* が文字列型で `-s` が省略された場合、*capname* がこの端末タイプに対して定義されていないことを表す。

2 使用方法のエラー。

3 指定された端末タイプに関して、情報が何も存在しません。

4 不正なオペランドが指定されました。

>4 エラーが発生した。

-1 *capname* は terminfo データベースに指定された数値ではなかった。たとえば `tput -T450 lines` や `tput -T2621 xmc` などが指定されました。

ファイル

/usr/include/curses.h

/usr/include/term.h

/usr/lib/tabset/*

/usr/share/lib/terminfo/?/*

curses(3CURSES) ヘッダー

terminfo ヘッダー

ある端末用のタブ設定を、その端末に出力する上で適切な形式(マージンとタブを設定するエスケープシーケンス)で指定。詳細については[terminfo\(4\)](#)の説明中の「タブと初期化」の項を参照

コンパイルされた端末記述データベース

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[clear\(1\)](#), [sh\(1\)](#), [stty\(1\)](#), [tabs\(1\)](#), [curses\(3CURSES\)](#), [profile\(4\)](#), [terminfo\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#), [term\(5\)](#)

名前	tr – 文字の変換
形式	<pre>/usr/bin/tr [-cds] [string1 [string2]] /usr/xpg4/bin/tr [-cs] string1 string2 /usr/xpg4/bin/tr -s -d [-c] string1 /usr/xpg4/bin/tr -ds [-c] string1 string2 /usr/xpg6/bin/tr [-c -C] [-s] string1 string2 /usr/xpg6/bin/tr -s [-c -C] string1 /usr/xpg6/bin/tr -d [-c -C] string1 /usr/xpg6/bin/tr -ds [-c -C] string1 string2</pre>
機能説明	tr ユーティリティーは、選択した文字を置き換えるか削除して、標準入力を標準出力へコピーします。指定されたオプションと <i>string1</i> と <i>string2</i> の両オペランドにより、文字や単一文字比較要素のコピー中に発生する変換を制御します。
オプション	サポートしているオプションは、次のとおりです。 -c <i>string1</i> で指定された値のセットを補完します。 -C <i>string1</i> で指定された文字のセットを補完します。 -d <i>string1</i> で指定された文字を入力中からすべて削除します。 -s 繰り返して現れた文字を 1 個の文字に置き換えます。 -d オプションを指定しない場合、以下の処理が行われます。 ■ <i>string1</i> で指定した配列中の文字が入力中で見つかった場合、<i>string2</i> で指定した配列中の同じ位置にある文字に置き換えます。<i>string2</i> が示す配列が <i>string1</i> が示す配列より短い場合、処理の結果は不確定です。 ■ -c オプションを指定した場合、<i>string1</i> で指定した値の補完は、配列内に、バイナリ値による昇順で配置されます。 ■ -c オプションを指定すると、<i>string1</i> で指定された文字の補完集合 (LC_CTYPE の現在の設定値で定義されている現文字セット内のすべての文字から、<i>string1</i> オペランドで実際に指定された文字を除いたもの) が、LC_COLLATE の現在の設定値で定義されている照合順序に従って、昇順で配列に置かれます。 ■ なお文字クラス表現や同等クラス表現により指定される文字の順序は定義されていないため、そのような表現を使用するのは、複数の文字を同一の文字にマップする場合に限るべきです。ただし前述のように、大文字と小文字との間の変換は例外です。 -d オプションを指定すると、以下の処理が行われます。 ■ <i>string1</i> で指定した配列中にある入力文字は削除されます。

- `-c` オプションも一緒に指定された場合、*string1* で指定した文字を除くすべての文字が削除されます。`--s` オプションが指定されていなければ、*string2* の内容は無視されます。
- `-c` オプションを指定した場合、*string1* で指定した値の補完は、配列内に、バイナリ値による昇順で配置されます。
- `-d` と `-s` の2つのオプションに、同じ文字列を指定することはできません。2つのオプションをともに指定する場合、削除用の *string1* と圧縮用の *string2* を両方とも指定しなければなりません。

`-s` オプションを指定すると、何らかの削除または変換発生後に同じ文字が連続して出現し、その文字が最終オペランドで指定した配列中に存在していれば、一連の文字はその文字1個に置き換えられます。最終オペランドとして、次の例に示すような文字クラスが指定されたとします。

```
tr -s '[:space:]'
```

この場合、最終オペランドの配列には、その文字クラスのすべての文字が含まれることになります。ただし、前述のように、大文字と小文字の変換が次のように指定された場合は異なります。

```
tr -s '[:upper:]' '[:lower:]'
```

この場合、最終オペランドの配列には、`toupper` と `tolower` の対のうち2番目の文字として定義されている方の文字だけが含まれます (`toupper(3C)` と `tolower(3C)` を参照)。

string1 または *string2* に空の文字列を指定した場合、処理の結果は予測できません。

オペランド

次のオペランドがサポートされています。

string1

string2 変換を制御する文字列です。各文字列は、変換に使用される文字配列に置き換えられる、一群の文字を表します。

オペランド *string1* と *string2* により2個の文字配列を定義します。以下に示す要素を使って、文字や単一文字照合要素を指定できます。これらの要素を使った結果、複数文字照合要素が得られた場合には、`tr` は配列からその複数文字要素を除きます。このとき、診断メッセージは発行しません。

character 以下の説明文中に現れない文字は、その文字自体を表します。

`\octal`

8進数のシーケンスを使って、具体的なコード値を示す文字を表すことができます。8進数シーケンスは、バックスラッシュの後に最大1、2、または3桁の8進数 (01234567) を付加したものです。ここで指定したシーケンスにより、1、2、または3桁の8進整数で表される文字が、配列中に配置されます。複数バイト文字は、この種のエス

ケープシーケンスがいくつか連続したものを必要とします。そのとき、各バイトの先頭に `\` が必要です。

`\ character` バックスラッシュ付きのエスケープシーケンスとして、`\a`、`\b`、`\f`、`\n`、`\r`、`\t`、`\v` がサポートされています。バックスラッシュの後に、これ以外の文字で 8 進数字でもない文字を指定した場合、結果は不確定です。

/usr/xpg4/bin/tr

`c-c`

/usr/bin/tr

`[c-c]`

POSIX ロケールでは、照合シーケンスで定義されたように、この構文は範囲端点間にある照合要素の範囲を表します (どちらの端点も `\octal` 形式の 8 進数シーケンスでない場合に限る)。この範囲に含まれる文字または照合要素は、昇順で配列中に置かれます。照合シーケンスで 2 番目の端点が開始端点の前に来る場合は、照合要素の範囲が空であるのか、またはこの構文が無効として処理されるかどうかは特定されません。POSIX 以外のロケールでは、この構文の動作は特定されません。

範囲端点のどちらかまたは両方が `\octal` 形式の 8 進数シーケンスである場合、2 つの範囲端点間にあるコーディングされた特定のバイナリの範囲を表します。

`[:class:]` 現在の LC_CTYPE ロケールカテゴリの設定値に従い、指定された文字クラスに属するすべての文字を表します。以下の文字クラス名が `string1` として指定できます。

```
alnum blank digit lower punct upper
alpha cntrl graph print space xdigit
```

さらに、`[:name:]` 形式の文字クラス式も指定できます。ただし、`name` が LC_CTYPE カテゴリ内で `charclass` 定義を与えられているようなロケールにおいてのみです。

`-d` と `-s` の両オプションが指定されていれば、どんな文字クラス名でも `string2` に指定できます。そうでなければ、`lower` または `upper` だけが `string2` として指定できます。ただし、これに対応する文字クラス `upper` または `lower` が `string1` 中の同等の位置に指定された場合だけです。このような指定は、大文字と小文字間の変換要求と見なされます。`[:lower:]` が `string1` 中に現れ、`[:upper:]` が `string2` 中に現れた場合、現在のロケールの LC_CTYPE カテゴリ中の `toupper` マッピングから得られる文字が、配列に含まれることになります。`[:upper:]` が `string1` 中に現れ、`[:lower:]` が `string2` 中に現れた場合、現在のロケールの LC_CTYPE カテゴリ中の `tolower` マッピングから得られる文字が、配列に含まれることになります。各マッピングの対の最初の文字が `string1` の配列に、2 番目の文字が `string2` の配列に置かれます。配列内での位置は同じになります。

大文字と小文字間の変換の場合を除き、文字クラス式で指定された文字を配列中に置く順序は決まっています。

class に指定した文字が現在のロケール用の正しい文字クラスを表していない場合、処理の結果は予測できません。

[*=equiv*=] 現在の LC_COLLATE ロケールカテゴリの設定による定義に従い、指定された文字クラスに属するすべての文字を表します。*equiv* と同じ同等クラスに属するすべての文字または照合要素を表します。同等クラス式は *strint1* に指定できます。また *-d* と *-s* の両オプションを指定した場合には、*string2* にも指定できます。この同等クラスに属する文字を配列に置く順序は、決まっています。

[*x***n*] *x* が示す文字の *n* 回の連続発生を表します。この式は複数の文字を1個の文字にマップするために使用するので、*string2* にだけ指定できます。*n* の先頭の数字が 0 のとき、この数値は 8 進数と見なされます。それ以外の場合、10 進数と見なされます。

n を省略した場合、または 0 の場合は、*/usr/bin/tr* はこれを非常に大きいと解釈します。*/usr/xpg4/bin/tr* および */usr/xpg6/bin/tr* は、これを *string2* ベースのシーケンスを *string1* ベースのシーケンスの長さまで拡張できるほど十分な大きさがあると解釈します。

使用法

ファイルが 2G バイト (2³¹ バイト) 以上ある場合の *tr* の動作については、[largefile\(5\)](#) を参照してください。

使用例

例1 文字リストを作成する

以下の例は、*file1* 中にあるすべての単語を1行に1個の形式で *file2* に出力します。ここで言う単語とは、最大文字列を表します。

```
tr -cs "[:alpha:]" "[\n*]" <file1 >file2
```

例2 文字を変換する

次の例は、*file1* 中のすべての小文字を大文字に変換し、その結果を標準出力に書き出します。

```
tr "[:lower:]" "[:upper:]" <file1
```

なお、対応する XPG3 での例に示されている警告は、この場合有効ではありません。この大文字・小文字変換は、*tolower* と *toupper* を使った特殊なケースで、ロケールが正しく定義されているとき、マッピングが正しく行われることを確認するものです。

例3 同じ文字を識別する

次の例は、同等クラスを使って *file1* 中でアクセント記号付きのベース文字 *e* を認識し、その記号を取り除いて *file2* に書き出します。

例3 同じ文字を識別する (続き)

```
tr "[=e]" e <file1 >file2
```

環境変数 trの実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 入力データはすべて正常に処理されました。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの[attributes\(5\)](#)を参照してください。

/usr/bin/tr

属性タイプ	属性値
使用条件	system/core-os
CSI	有効

/usr/xpg4/bin/tr

属性タイプ	属性値
使用条件	system/xopen/xcu4
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

/usr/xpg6/bin/tr

属性タイプ	属性値
使用条件	system/xopen/xcu6
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [ed\(1\)](#), [sed\(1\)](#), [sh\(1\)](#), [tolower\(3C\)](#), [toupper\(3C\)](#), [ascii\(5\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [regex\(5\)](#), [standards\(5\)](#)

注意事項 以前のバージョンとは違って、/usr/xpg4/bin/trは入力ストリーム中のNUL文字を正しく処理します。NUL文字はtr -d '\000'によって取り除くことができます。

名前 tr – translate characters

形式 /usr/ucb/tr [-cds] [*string1* [*string2*]]

機能説明 The `tr` utility copies the standard input to the standard output with substitution or deletion of selected characters. The arguments *string1* and *string2* are considered sets of characters. Any input character found in *string1* is mapped into the character in the corresponding position within *string2*. When *string2* is short, it is padded to the length of *string1* by duplicating its last character.

In either string the notation:

a–b

denotes a range of characters from *a* to *b* in increasing ASCII order. The character `\`, followed by 1, 2 or 3 octal digits stands for the character whose ASCII code is given by those digits. As with the shell, the escape character `\`, followed by any other character, escapes any special meaning for that character.

オプション Any combination of the options `-c`, `-d`, or `-s` may be used:

- c Complement the set of characters in *string1* with respect to the universe of characters whose ASCII codes are 01 through 0377 octal.
- d Delete all input characters in *string1*.
- s Squeeze all strings of repeated output characters that are in *string2* to single characters.

使用例 例 1 Creating a list of all the words in a filename

The following example creates a list of all the words in *filename1*, one per line, in *filename2*, where a word is taken to be a maximal string of alphabetic. The second string is quoted to protect `\` from the shell. 012 is the ASCII code for NEWLINE.

example% `tr -cs A-Za-z '\012' < filename1 > filename2`

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [ed\(1\)](#), [ascii\(5\)](#), [attributes\(5\)](#)

注意事項 Will not handle ASCII NUL in *string1* or *string2*. `tr` always deletes NUL from input.

名前 trap, onintr - (ハードウェア) シグナルに応答するためのシェル組み込み関数

形式

sh trap [*argument n* [*n2*]...]

csH onintr [-| *label*]

ksh88 *trap [*arg sig* [*sig2*...]]

ksh +trap [-p] [*action condition*...]

機能説明

sh *argument* が示す trap コマンドを、シェルが数値形式または 名前形式で指定されているシグナル (*n*) を受信した時に読み取り、実行します。(注: *argument* は、トラップ設定時とトラップ取り出し時に1度ずつ検索されます。)トラップコマンドは、シグナル番号または対応するシンボリック名の順序で実行されます。現在のシェルへの入力時に無視されたシグナルにトラップを設定しようとしても無効となります。シグナル 11 (メモリフォールト) にトラップを指定しようすると、エラーになります。*argument* を省略すると、*n* のトラップはすべてその元の値に再設定されます。*argument* が NULL 文字列の場合、シェルおよびシェルが呼び出すコマンドは、このシグナルを無視します。*n* が 0 の場合、*argument* が示すコマンドはシェル終了時に実行されます。引数なしの trap コマンドは、コマンドの一覧を各々が対応しているシグナル番号とともに表示します。

csH onintr は割り込み時のシェルの動作を制御します。引数を指定しないと、onintr はデフォルトの動作を復元します(すなわち、シェルはシェルスクリプトを終了して、端末のコマンド入力レベルに戻ります)。- 引数を指定すると、シェルはすべての割り込みを無視します。*label* 引数を指定すると、割り込みを受信するか割り込みのために子プロセスが終了したときに、シェルは goto *label* を実行します。

ksh88 trap は *arg* を *sig* が示すシグナルをシェルが受信したときに読み取られ、実行されるコマンドとして使用します。*arg* は、トラップ設定時とトラップ取り出し時に1度ずつ検索されます。各 *sig* は、数値またはシグナルの名前です。trap コマンドは、シグナル番号の順序で実行されます。現在のシェルへの入力時に無視されたシグナルにトラップを設定しようとしても無効となります。*arg* を省略するか、または - と指定する場合、各 *sig* 用のトラップはすべてその元の値に再設定されます。*arg* が NULL 文字列 (" " などの空の文字列) の場合、シェルおよびシェルが呼び出すコマンドは、このシグナルを無視します。*sig* が ERR の場合は、コマンドが 0 以外の終了ステータスで終わると必ず *arg* が実行されます。*sig* が DEBUG の場合は、各コマンドのあとで *arg* が実行されます。*sig* が 0 または EXIT で、トラップが関数の外側で設定されている場合、シェルの終了時に *arg* が示すコマンドが実行されます。引数なしの trap コマンドは、コマンドの一覧を各々が対応しているシグナル番号とともに表示します。

このマニュアルページでは、1 つまたは 2 つのアスタリスク (*) が先頭に付加されている ksh88(1) コマンドは、次の方法で特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** から始まるコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

ksh

trap は、シグナル受信などの状況が発生したときに行われるアクションを定義する特別な組み込みコマンドです。**trap** を使用すると、現在のトラップ設定を標準出力に表示することもできます。

action が - の場合、**trap** は各状況をデフォルト値にリセットします。*action* が空の文字列の場合は、各状況が発生してもシェルは無視します。それ以外の場合、引数 *action* は該当する状況のいずれかが発生したときに **eval** によって処理された場合と同様に、シェルによって読み取られ実行されます。トラップのアクションは、指定された各状況に関連付けられた以前のアクションよりも優先されます。値 \$? はトラップの実行によって変更されません。

condition には、シグナルの名前または番号、あるいは次のいずれかを指定できます。

EXIT シェルの終了時に、このトラップを実行します。予約語 **function** を含む関数内に定義した場合は、関数が復帰するときに呼び出し元の環境でトラップを実行します。トラップのアクションは、関数の呼び出し時の値に復元されます。

0 **EXIT** と同じです。

DEBUG 各単純コマンドが実行される前 (ただし、引数が展開されたあと) に実行します。

ERR **set -e** によりシェルが終了するたびに実行します。

KEYBD 端末装置からキー入力されたときに実行します。

シグナル名は大文字と小文字の区別がなく、**sig** 接頭辞はオプションです。非対話型シェルの開始時に無視されたシグナルは、トラップもリセットもできません。ただし、トラップやりセットを行なってもエラーは報告されません。1、2、3、6、9、14、15 以外のシグナル番号の使用は移植できません。

trap は特殊な組み込みコマンドですが、シェルが認識しない状況を指定すると、トラップがゼロ以外の終了ステータスで終了します。ただし、呼び出し側のシェルは終了しません。

アクションまたは状況を指定しない場合は、現在のトラップ設定がすべて標準出力に書き込まれます。

ksh の trap 組み込みコマンドでは、次のオプションがサポートされています。

-p 現在のトラップを再作成できるように、現在のトラップがシェルへの入力として処理可能な形式で出力されます。

ksh の trap の組み込みコマンドは、次の値のいずれかで終了します。

0 正常終了。

>0 エラーが発生した。

このマニュアルページでは、1つまたは2つの+ (プラス符号) が先頭に付加されている **ksh(1)** コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. これらは有効な関数名ではありません。
5. 変数代入形式で、++を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は=符号のあとに実行され、ワード分割とファイル名生成は実行されません。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

```
csh(1),eval(1),exit(1),ksh(1),ksh88(1),sh(1),attributes(5)
```

名前	troff - ドキュメントのタイプセットと清書
形式	troff [-a] [-f] [-Fdir] [-i] [-mname] [-nN] [-olist] [-raN] [-sN] [-uN] [-z] [filename]...
機能説明	troff は、 <i>filename</i> 引数で指定されたファイル群中のテキストを、タイプセット用またはレーザープリンタでの印刷用に清書します。troff への入力ファイルの中身は、清書用の要求とマクロを含んだテキストです。 <i>filename</i> 引数が省略された場合、troff は標準入力から読み込みます。複数のファイル名も指定可能で、そのうちの1つとして標準入力を指定したければ、該当する位置にハイフン (-) を記述します。
オプション	以下のオプションがサポートされています。任意の順序で指定できますが、すべてのオプションは先頭の <i>filename</i> より前に指定してください。 -a ASCII の近似形式で清書した出力を標準出力に書き出します。(注: 最近はほとんど使用されない <code>/usr/bin/ta</code> コマンドを使うと、ASCII の未完版も通常端末上に出力されます。) -f 最終ページを出力したあと、トレーラページを印刷しないか、またはポストプロセッサが装置の制御権を放棄しないようにします。 -Fdir フォント幅や端末テーブルを検索するディレクトリとして、システムのデフォルトディレクトリの代わりに <i>dir</i> を使用します。 -i 指定した入力ファイルをすべて処理したあと、標準入力を読み込みます。 -mname 入力の <i>filename</i> の前に、マクロファイル <code>/usr/share/lib/tmac/name</code> を付加します。なお、マクロパッケージへの参照の多くは、通常、名前の先頭に <i>m</i> を含みます。たとえば <code>man(5)</code> マクロは、<code>/usr/share/lib/tmac/an</code> 中に存在しています。マクロディレクトリを変更したければ、環境変数 <code>TROFFMACS</code> の値として希望するパスを設定してください。なお、パス名の最後に必ずスラッシュ (/) を付加するよう注意してください。 -nN 出力される先頭ページのページ番号を <i>N</i> に設定します。 -olist 特定のページだけを印刷出力したい場合、<i>list</i> 引数を使ってそのページ番号のリストを指定します。リスト中では項目をコンマで区切って記述し、各項目として単一のページ番号または番号の範囲を記述できます。範囲指定には3つの形式があり、<i>N-M</i> はページ番号 <i>N</i> から <i>M</i> までを表し、先頭項目として指定する <i>-N</i> は先頭ページからページ <i>N</i> までを表し、最終項目として指定する <i>N-</i> はページ <i>N</i> から最終ページまでを表します。 -q nroff では静寂モードを示すオプションです。troff では無視されます。 -raN レジスタ <i>a</i> の値として <i>N</i> を設定します。<i>a</i> は常に1文字の名前です。

- sN N ページを出力するごとに 写真植字機を停止させます。機器の種類によっては、 troff がトレーラページを出力するのでカセットを 交換することができます。機器のスタートボタンを押せば印刷が再開できます。
- uN ポジション 3 にマウントされているフォント用の ボールド（太文字）化ファクタ値を N に設定します。N 引数を省略すると、ボールド化ファクタ値は 0 に設定されます。
- z 清書した出力結果を印刷しません。つまり診断メッセージや .tm 要求を使って出力されたメッセージだけが印刷されます。

オペランド 次オペランドを指定できます。

filename troff によって処理されるテキストを含むファイル

- ファイル /tmp/trtmp 一時ファイル
- /usr/share/lib/tmac/* 標準マクロファイル
- /usr/lib/font/* マウントされている代替 troff フォント用のフォント幅テーブル
- /usr/share/lib/nterm/* nroff 用の端末駆動テーブル

属性 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目 checknr(1), col(1), eqn(1), man(1), nroff(1), tbl(1), attributes(5), man(5), me(5), ms(5)

注意事項 troff は 7 ビット ASCII に基づいて設計されているので、8 ビットクリーンではありません。

以前のドキュメントでは、数値レジスタ yr を現在の年の末尾の 2 桁であると説明していましたが、これは正しくありません。yr は、実際には 1900 年以來の年数を示します。2099 年までの、現在の年号の末尾の 2 桁を正しく獲得するには、次の文字列レジスタ yy の定義をドキュメントに含めると、2 桁の年号を表示するために使用できます。yr の代わりに、別の 1 つまたは 2 つの文字をレジスタ名として使用できます。

```
.\" definition of new string register yy--last two digits of year
.\" use yr (# of years since 1900) if it is < 100
.ie \n(yr<100 .ds yy \n(yr
.el \{
    .\" else, subtract 100 from yr, store in ny
    .nr ny \n(yr-100
    .ie \n(ny>9 \{
        .\" use ny if it is two digits
        .ds yy \n(ny
    .\" remove temporary number register ny
```

```
.rr ny \}  
.el \{.ds yy 0  
.\" if ny is one digit, append it to 0  
.as yy \n(ny  
.rr ny \} \}
```

名前	true, false – provide truth values
形式	true false
機能説明	The <code>true</code> utility does nothing, successfully. The <code>false</code> utility does nothing, unsuccessfully. They are typically used in a shell script <code>sh</code> as: <pre>while true do <i>command</i> done</pre> which executes <i>command</i> forever.
終了ステータス	<code>true</code> has exit status <code>0</code>. <code>false</code> always will exit with a non-zero value.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [sh\(1\)](#), [attributes\(5\)](#), [standards\(5\)](#)

名前	truss – システムコールとシグナルの追跡
形式	truss [-fcaeildDE] [- [tTvX] [!]] syscall ,... [- [sS] [!]] signal ,... [- [mM] [!]] fault ,... [
機能説明	truss ユーティリティーは指定されたコマンドを実行し、それ自体が実行するシステムコール、受け取ったシグナル、および検出したマシン障害の追跡情報を生成します。追跡出力の各行には、障害またはシグナルの名前、あるいはシステムコール名とその引数および戻り値が表示されます。システムコールの引数は、可能なかぎの、関連するシステムヘッダー内の定義に従って記号で表示されます(パス名へのポインタ引数の場合、ポイント先の文字列が表示される)。エラーが発生した場合の戻り値は、Intro(3)に説明されているエラーコード名を使用して報告されます。エラーが発生して、カーネルが失われた特権を報告する場合、特権名(privileges(5)を参照)は、エラーコード名の後に、角括弧([]) で囲まれて報告されます。エラーレポートの詳細については、「注意事項」を参照してください。 truss は、-u オプションにより、追跡するプロセスによって実行されたユーザーレベルの関数呼び出しの開始および終了時の追跡情報も生成します。この場合、入れ子レベルを示すためにインデントが行われます。
オプション	リスト引数を受け付けるオプションでは、リスト内のすべての可能なメンバーを指定する短縮形として名前 all を使用できます。リストが ! で始まるオプションは否定を意味します(たとえば、追跡せずに除外するなど)。同じオプションを複数回指定できます。リスト内の同じ名前に対しては、あとから指定されたオプションが先に指定されているオプション(リスト内で左側にあるもの)を無効にします。 サポートしているオプションは、次のとおりです。 -a 各 exec() システムコールに渡される引数文字列を表示します。 -c 追跡情報を 1 行ずつ表示するのではなく、追跡されたシステムコール、障害、およびシグナルをカウントします。追跡されたコマンドが終了するか、あるいは truss が割り込まれたときに、サマリーレポートが生成されます。-f も指定すると、子プロセスについて追跡されたシステムコール、障害、およびシグナルもすべてカウントに含まれます。 -d 追跡出力の各行にタイムスタンプを含めます。タイムスタンプは、seconds.fraction という形式で行頭に示されます。これは、追跡の開始時からの経過秒数を示したものです。追跡出力の最初の行には、個々のタイムスタンプ測定の基点となるベースタイムが、epoch からの経過秒数(time(2)を参照)と日付文字列(ctime(3C)およびdate(1)を参照)の両方の形式で表示されま す。報告される時間は、当該イベントが発生した時間

です。どのシステムコールについても、イベントはシステムコールの終了時であり、システムコールの開始時ではありません。

- D 追跡出力の各行にデルタタイムを含めます。この値は、*seconds.fraction* を含むフィールドとして表示され、同じ LWP が原因であった最後に報告されたイベント以降のイベントの原因となった LWP の経過時間を表します。システムコールの場合、これはシステムコール内で経過した時間ではありません。
- e 各 `exec()` システムコールに渡される環境文字列を表示します。
- E 追跡出力の各行にデルタタイムを含めます。*seconds.fraction* という形式で示されるこの値は、システムコールが始まってから終わるまでの経過時間を示します。

-D オプションとは異なり、この時間は、システムコール内で費やされた総時間です。
- f `fork()` または `vfork()` によって作成されたすべての子プロセスを追跡し、それらのシグナル、障害、およびシステムコールを追跡出力に含めます。通常は、最初のコマンドまたはプロセスだけが追跡されます。`-f` を指定すると、追跡出力の各行にプロセス ID が表示され、どのプロセスがシステムコールを実行したか、あるいはどのプロセスがシグナルを受信したかが示されます。
- i 割り込み可能な休眠状態のシステムコールを表示しません。端末デバイスまたはパイプ上での `open()` や `read()` など、特定のシステムコールは不確定時間に休眠でき、割り込み可能です。一般に、システムコールが 1 秒を超えてそのような休眠状態になっている場合、truss はそれらを報告します。システムコールは完了時に再度、報告されます。`-i` オプションは、そのようなシステムコールを、完了時のみの 1 度だけ報告します。
- l 追跡出力の各行に対応する軽量プロセス (LWP) の ID を含めます。`-f` も指定すると、プロセス ID と LWP ID の両方を含めます。
- m [!]*fault*, ... 追跡または除外するマシン障害を指定します。コマンドで区切ったリストに指定した障害が追跡されます。障

- 害は、名前または番号で指定できます (<sys/fault.h> を参照)。リストが ! で始まる場合、指定された障害は追跡出力から除外されます。デフォルトは `-mall -m !fltpage` です。
- `-M[!]fault,...` プロセスを停止するマシン障害を指定します。指定した障害は、`-m` で指定されるセットに追加されます。指定された障害の 1 つが見つかり、truss はプロセスを停止したままにし、終了します (`-T` オプションを参照)。デフォルトは `-M!all` です。
- `-o outfile` 追跡出力に使用されるファイル。デフォルトでは、出力は標準エラー出力に送られます。
- `-p` truss に指定される *command* 引数を、実行されるコマンドとしてではなく、既存のプロセス (`ps(1)` のマニュアルページを参照) のプロセス ID のリストとして解釈します。プロセスのユーザー ID とグループ ID が実行するユーザーの ID と一致するか、あるいはユーザーが特権ユーザーである場合、truss は各プロセスを制御し、それらの追跡を開始します。ユーザーは、選択した (つまり、`/thread-id` をプロセス ID に追加した) スレッドだけを追跡できます。複数のスレッドを選択するには、区切り文字「`-`」と「`,`」を使用します。たとえば、「`/1,2,7-9`」は、スレッド 1、2、7、8、および 9 を追跡します。プロセスの指定は、`/proc` ディレクトリ内の名前を指定することによっても行えます (例: `/proc/12345`)。
- `-r[!]fd,...` 指定したファイル記述子の `read()` ごとに、入出力バッファの全内容を表示します。出力は行当たり 32 バイトに整形され、各バイトは ASCII 文字 (先頭に 1 個の空白が入る)、または、水平タブ (`\t`) や復帰改行 (`\n`) などの制御文字のために 2 文字の C 言語エスケープシーケンスとして表示されます。ただし、ASCII 解釈が不可能な場合は、2 文字の 16 進表現となります (`-r` が指定されない場合でも追跡された各 `print>read()` の入出力バッファの最初の 12 バイトは表示される)。デフォルトは `-r!all` です。
- `-s[!]signal,...` 追跡または除外するシグナルを指定します。コンマで区切ったリストに指定したシグナルを追跡します。シグナルが無視される (ブロックされてない) 場合でも、追跡出力には、指定された各シグナルの受信が表示されます (ブロックされているシグナルはブロックが

	解放されるまで受信されない)。シグナルは、名前または番号で指定できます(<sys/signal.h>を参照)。リストが!で始まる場合、指定されたシグナルは追跡出力から除外されます。デフォルトは -sall です。
-S [!]signal, ...	プロセスを停止するシグナルを指定します。指定されたシグナルは、-s で指定されるセットに追加されます。指定されたシグナルのどれかが受信された場合、truss はプロセスを停止したままにし、終了します(-T オプションを参照)。デフォルトは -s!all です。
-t [!]syscall, ...	追跡または除外するシステムコールを指定します。コンマで区切ったリストに指定されたシステムコールの追跡が行われます。リストが!で始まる場合、指定したシステムコールが追跡出力から除外されます。デフォルトは -tall です。
-T [!]syscall, ...	プロセスを停止するシステムコールを指定します。指定されたシステムコールが、-t によって指定されるセットに追加されます。指定されたシステムコールの1つが見つかり、truss はプロセスを停止したままにし、終了します。つまり、truss はプロセスを解放して実行を終了しますが、当該システムコールの完了時にそのプロセスを停止状態のままにします。これにより、停止したプロセスにデバッガなどのプロセス検査ツール(proc(1))のマニュアルページを参照)を適用できるようになります。追跡を継続するには、同じオプションまたは異なるオプションを指定して、停止されたプロセスにtrussを適用し直します。デフォルトは -T!all です。
	この方法で停止されたままになったプロセスは、アプリケーション kill -CONT によって再開することはできません。これは、停止シグナル(signal.h(3HEAD) を参照)のデフォルトアクションによってではなく、/proc を介したイベント上でこのプロセスが停止されているためです。停止中のプロセスを再実行するように設定するには、 proc(1) のマニュアルページで説明されている prun(1) コマンドを使用できます。
-u [!]lib,...:[!] <i>func</i> , ...	ユーザーレベルの関数呼び出しを追跡します。 <i>lib</i> ,... は、動的なライブラリ名(.so.n 接尾辞を除く)をコンマで区切ったリストです。 <i>func</i> ,... は、関数名をコンマで区切ったリストです。どちらの場合でも、名前の表現にメタ文字 *, ?, [] を使用できま

す。これらのメタ文字の指定は **sh(1)** における指定と同じ意味を持ちますが、ファイルに対してではなくライブラリ名または関数名に対して使用されることになります。ライブラリまたは関数のリストを空にすると、デフォルトで*が使用され、ライブラリ内のすべてのライブラリまたは関数が追跡されます。リストの先頭に!を付けると、追跡から除外されるライブラリまたは関数の名前を指定したことになります。1つのライブラリを除外すると、そのライブラリ内のすべての関数が除外されます。つまり、ライブラリ除外リストのあとに続く関数リストは無視されます。

関数リストとライブラリリストを分離する1つの：は、ライブラリの外部から、それらのライブラリに対する呼び出しは追跡しますが、ライブラリ内部の他の関数からの呼び出しは除外することを意味します。2つの::は、呼び出し元に関係なくすべての呼び出しを追跡することを意味します。

ライブラリのパターンには、正確な一致がないかぎり、実行可能ファイルと動的リンカーのいずれとも対応付けはしません(l*はld.so.1に対応付けられない)。これらのオブジェクトのどちらかに含まれる関数を追跡するには、次のように名前を明確に指定する必要があります。

```
truss -u a.out -u ld ...
```

a.out はこの目的で使用されるリテラル名であり、実行可能ファイルの名前を意味するわけではありません。**a.out** 関数呼び出しを追跡すると、すべての呼び出しが暗黙に追跡されます(デフォルトは::)。

-u オプションは複数回指定することが可能で、この場合左から順に受け付けられます。軽量プロセスのIDと関数呼び出しを実行したスレッドのIDがそれぞれ、スラッシュ文字(/)とアットマーク記号(@)で区切られて、呼び出しのトレース出力に含まれます。

```
-U [!]lib, ... : [!]func, ...
```

プロセスを停止するユーザーレベルの関数呼び出しを指定します。指定される関数は、**-u** で指定されるセットに追加されます。指定された関数の1つが呼び出されると、**truss** はプロセスを停止したままにし、終了します(**-T** オプションを参照)。

- `-v [!]syscall,...` 冗長。指定されたシステムコールに対してアドレスで渡された任意の構造体の内容を表示します (`-t` による追跡が行われた場合)。入力した値とオペレーティングシステムによって返される値が示されます。入力と出力の両方に使用されるフィールドについては、出力値だけが示されます。デフォルトは `-v!all` です。
- `-w [!]fd,...` 指定したファイル記述子の `write()` ごとに入出力バッファの内容を表示します (`-r` オプションを参照)。デフォルトは `-w!all` です。
- `-x [!]syscall,...` 指定されたシステムコールの引数を raw 形式で表示します (`-t` による追跡が行われた場合)。これは、通常、記号表示ではなく 16 進表示であり、raw ビットのままの方が良いと考えるハッカーのためのものです。デフォルトは `-x!all` です。

`-t`、`-T`、`-v`、および `-x` オプションが受け入れるシステムコール名については、『[man pages section 2: System Calls](#)』を参照してください。システムコール番号も指定できます。

指定したコマンドを開始および追跡するのに `truss` を使用する場合、`-o` オプションを使用するか、あるいは標準エラー出力を端末以外のファイルにリダイレクトすると、`truss` はハングアップ、割り込みシグナル、および終了シグナルを無視して動作します。これにより、端末からの割り込みシグナルと終了シグナルを受け取る対話型プログラムの追跡が容易になります。

追跡出力を端末に転送したままにした場合、あるいは、既存のプロセスを追跡する (`-p` オプション) 場合、`truss` は追跡したすべてのプロセスを解放して処理を終了するので、ハングアップ、割り込み、終了の各シグナルに応答します。これにより、ユーザーは過度の追跡出力を抑制でき、既存のプロセスを解放できるようになります。解放されたプロセスは、なんの影響も受けず、それまでどおりの通常の処理を継続します。

既存のプロセスを追跡すると、`truss` はプロセスを解放して、`truss` 終了時に実行中に設定します。これには、`SIGINT`、`SIGHUP`、`SIGQUIT` などのシグナルによる終了も含まれます。これにより、ユーザーは過度の追跡出力を抑制でき、既存のプロセスを解放できるようになります。解放されたプロセスは、なんの影響も受けず、それまでどおりの通常の処理を継続します。

使用例

例1 コマンドを追跡する

この例は、端末上の `find(1)` コマンドの追跡情報を生成します。

```
example$ truss find . -print >find.out
```

例2 一般的なシステムコールを追跡する

オープン、クローズ、読み取り、書き込みの各システムコールの追跡情報だけを表示するには、次のように指定します。

```
example$ truss -t open,close,read,write find . -print >find.out
```

例3 シェルスクリプトを追跡する

この例は、ファイル `truss.out` 上の `spell(1)` コマンドの追跡情報を生成します。

```
example$ truss -f -o truss.out spell document
```

`spell` はシェルスクリプトであるため、シェルだけでなくシェルによって生成されたプロセスも追跡するためには `-f` フラグが必要です (`spell` スクリプトは8つのプロセスのパイプラインを実行する)。

例4 出力を簡潔にする

出力を簡潔にする例を示します。

```
example$ truss nroff -mm document >nroff.out
```

この例は、出力の97%が `lseek()`、`read()`、および `write()` システムコールの追跡情報であり、冗長です。出力を簡潔にまとめるには次のように指定します。

```
example$ truss -t !lseek,read,write nroff -mm document >nroff.out
```

例5 Cライブラリの外部からのライブラリ呼び出しを追跡する

この例は、Cライブラリの外部からCライブラリ内の任意の関数に対して行われるユーザーレベルの呼び出しをすべて追跡します。

```
example$ truss -u libc ...
```

例6 Cライブラリ内からのライブラリ呼び出しを追跡する

この例では、CライブラリからそのCライブラリ自体の関数に対して行われる呼び出しが含まれます。

```
example$ truss -u libc:: ...
```

例7 Cライブラリ以外のライブラリ呼び出しを追跡する

この例は、Cライブラリ以外のすべてのライブラリに対して行われるユーザーレベルの呼び出しをすべて追跡します。

```
example$ truss -u '*' -u !libc ...
```

例8 `printf` および `scanf` 関数呼び出しを追跡する

この例は、Cライブラリに含まれる `printf` および `scanf` ファミリ内の関数に対するユーザーレベルの呼び出しをすべて追跡します。

例8 printfおよびscanf関数呼び出しを追跡する (続き)

```
example$ truss -u 'libc:*printf,*scanf' ...
```

例9 ユーザーレベルの関数呼び出しをすべて追跡する

この例は、任意の場所から任意の場所に対して行われるユーザーレベルの関数呼び出しをすべて追跡します。

```
example$ truss -u a.out -u ld:: -u :: ...
```

例10 システムコールの追跡結果を詳細に表示する

この例は、プロセス #1 の **init(1M)** システムコールのアクティビティーを追跡し、詳細な追跡情報を表示します (特権ユーザーがこのコマンドを実行できます)。

```
example# truss -p -v all 1
```

truss に割り込みを行うと、init は通常の動作に戻ります。

ファイル /proc/* プロセスファイル

属性 属性についての詳細は、マニュアルページの **attributes(5)** を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 **date(1), find(1), proc(1), ps(1), sh(1), spell(1), init(1M), Intro(3), exec(2), fork(2), lseek(2), open(2), read(2), time(2), vfork(2), write(2), ctime(3C), signal.h(3HEAD), proc(4), attributes(5), mwait(5), privileges(5), threads(5)**

『man pages section 2: System Calls』

注意事項 『『man pages section 2: System Calls』』で説明されているシステムコールの中には、実際のオペレーティングシステムインタフェースとは異なるものがあります。追跡情報が、この説明と若干異なる場合もあります。

マシン障害 (ページフォルトは除く) が発生するたびに、障害の原因となった LWP にシグナルが送られます。シグナルがブロックされない場合、各マシン障害 (ページフォルトを除く) が通知された直後に、受信シグナルが通知されます。

オペレーティングシステムは、プロセスの追跡に一定のセキュリティ制限を課します。具体的には、ユーザーは、自身が読み取りできないオブジェクトファイル (a.out) を持つコマンドを追跡することはできず、特権ユーザー以外は **setuid** と **setgid** をもつコマンドを追跡できません。特権ユーザーが実行する場合を除き、truss は **set-id** または読み取り不可能なオブジェクトファイルの **exec()** を実行するプロセスを制御できません。このようなプロセスは、truss からは独立して、**exec()** ポイントで通常どおり処理を継続します。

ほかの制御プロセスとの衝突を防ぐため、truss は、/proc インタフェースを介してほかのプロセスによって制御されているプロセスを追跡しません。このため、truss を [proc\(4\)](#) ベースのデバッガだけでなく、それ自身の別のインスタンスにも適用できます。

8 カラムごとに標準のタブストップが設定されていると仮定して、追跡出力にはタブ文字が含まれます。

複数のプロセスまたはマルチスレッドプロセス (複数の LWP) を含むプロセス) の追跡出力は、厳密に時間順には生成されません。たとえば、パイプにおける `read()` は、対応する `write()` よりも前に報告される場合があります。しかし、個々の LWP (従来型のプロセスには 1 つしか含まれない) について出力は、厳密に時間順に行なわれます。

複数のプロセスを追跡する場合、truss は追跡対象のプロセスごとに 1 つの制御プロセスとして動作します。前述の `spell` コマンドの例の場合、`spell` 自身が 9 つのプロセススロット (シェル用に 1 つ、8 メンバーを持つパイプライン用に 8 つ) を使用し、truss がさらに 9 つのプロセスを追加するため、プロセススロットは合計で 18 になります。

-v オプションでは、すべてのシステムコールで渡すことができるあらゆる構造体を表示できるわけではありません。

truss は、失われた特権が原因で発生したシステムコールによって返されたエラーをレポートする場合、エラーコードと単純な特権名を続けて表示するか、または特権の詳細な説明を表示します。[privileges\(5\)](#) を参照してください。詳細な説明には、以下の内容を含めることができます。

- [ALL] このプロセスでは、要求された操作のための特権がすべて必要になります。
- [MULTIPLE] このプロセスには、複数の特権はありません。
- [ZONE] このプロセスでは、ゾーン (ALL のゾーンローカルバリエント) で使用できる特権の 1 つがありません。
- [GLOBAL] 要求された操作では、プロセスがグローバルゾーンで実行されている必要があります。
- [MWAC] 要求された操作は、プロセス用に定められている [mwac\(5\)](#) ポリシーに違反しています。

名前 tset, reset – establish or restore terminal characteristics

形式 tset [-InQrs] [-ec] [-kc]
 [-m [*port-ID* [*baudrate*] : *type*]...] [*type*]
 reset [-] [-ec] [-I] [-kc]
 [-n] [-Q] [-r] [-s]
 [-m [*indent*] [*test baudrate*] : *type*]... [*type*]

機能説明

The tset utility sets up your terminal, typically when you first log in. It does terminal dependent processing such as setting erase and kill characters, setting or resetting delays, sending any sequences needed to properly initialize the terminal, and the like. tset first determines the type of terminal involved, and then does necessary initializations and mode settings. If a port is not wired permanently to a specific terminal (not hardwired) it is given an appropriate generic identifier such as dialup.

reset clears the terminal settings by turning off CBREAK and RAW modes, output delays and parity checking, turns on NEWLINE translation, echo and TAB expansion, and restores undefined special characters to their default state. It then sets the modes as usual, based on the terminal type (which will probably override some of the above). See [stty\(1\)](#) for more information. All arguments to tset may be used with reset. reset also uses rs= and rf= to reset the initialization string and file. This is useful after a program dies and leaves the terminal in a funny state. Often in this situation, characters will not echo as you type them. You may have to type LINEFEED reset LINEFEED since RETURN may not work.

When no arguments are specified, tset reads the terminal type from the TERM environment variable and re-initializes the terminal, and performs initialization of mode, environment and other options at login time to determine the terminal type and set up terminal modes.

When used in a startup script (.profile for [sh\(1\)](#) users or .login for [csh\(1\)](#) users) it is desirable to give information about the type of terminal you will usually use on ports that are not hardwired. Any of the alternate generic names given in the file /etc/termcap are possible identifiers. Refer to the -m option below for more information. If no mapping applies and a final type option, not preceded by a -m, is given on the command line then that type is used.

It is usually desirable to return the terminal type, as finally determined by tset, and information about the terminal's capabilities, to a shell's environment. This can be done using the -, -s, or -S options.

For the Bourne shell, put this command in your .profile file:

```
eval 'tset -s options...'
```

or using the C shell, put these commands in your .login file:

```
set noglob
eval 'tset -s options...'unset noglob
```


With the C shell, it is also convenient to make an alias in your `.cshrc` file:

```
alias ts 'eval `tset -s \!*\`'
```

This also allows the command:

```
ts 2621
```

to be invoked at any time to set the terminal and environment. It is not possible to get this aliasing effect with a Bourne shell script, because shell scripts cannot set the environment of their parent. If a process could set its parent's environment, none of this nonsense would be necessary in the first place.

Once the terminal type is known, `tset` sets the terminal driver mode. This normally involves sending an initialization sequence to the terminal, setting the single character erase (and optionally the line-kill (full line erase)) characters, and setting special character delays. TAB and NEWLINE expansion are turned off during transmission of the terminal initialization sequence.

On terminals that can backspace but not overstrike (such as a CRT), and when the erase character is `#`, the erase character is changed as if `-e` had been used.

オプション

- The name of the terminal finally decided upon is output on the standard output. This is intended to be captured by the shell and placed in the `TERM` environment variable.
- ec Set the erase character to be the named character *c* on all terminals. Default is the BACKSPACE key on the keyboard, usually `^H` (CTRL-H). The character *c* can either be typed directly, or entered using the circumflex-character notation used here.
- ic Set the interrupt character to be the named character *c* on all terminals. Default is `^C` (CTRL-C). The character *c* can either be typed directly, or entered using the circumflex-character notation used here.
- I Suppress transmitting terminal-initialization strings.
- kc Set the line kill character to be the named character *c* on all terminals. Default is `^U` (CTRL-U). The kill character is left alone if `-k` is not specified. Control characters can be specified by prefixing the alphabetical character with a circumflex (as in CTRL-U) instead of entering the actual control key itself. This allows you to specify control keys that are currently assigned.
- n Specify that the new tty driver modes should be initialized for this terminal. Probably useless since `stty new` is the default.
- Q Suppress printing the 'Erase set to' and 'Kill set to' messages.
- r In addition to other actions, reports the terminal type.
- s Output commands to set and export `TERM`. This can be used with

```
set noglob
eval 'tset -s ...'
unset noglob
```

to bring the terminal information into the environment. Doing so makes programs such as [vi\(1\)](#) start up faster. If the SHELL environment variable ends with csh, C shell commands are output, otherwise Bourne shell commands are output.

`-m [ port-ID [ baudrate ] : type ] ...` Specify (map) a terminal type when connected to a generic port (such as *dialup* or *plugboard*) identified by *port-ID*. The *baudrate* argument can be used to check the baudrate of the port and set the terminal type accordingly. The target rate is prefixed by any combination of the following operators to specify the conditions under which the mapping is made:

- > Greater than
- @ Equals or “at”
- < Less than
- ! It is not the case that (negates the above operators)
- ? Prompt for the terminal type. If no response is given, then type is selected by default.

In the following example, the terminal type is set to `adm3a` if the port is a `dialup` with a speed of greater than 300 or to `dw2` if the port is a `dialup` at 300 baud or less. In the third case, the question mark preceding the terminal type indicates that the user is to verify the type desired. A NULL response indicates that the named type is correct. Otherwise, the user's response is taken to be the type desired.

```
tset -m 'dialup>300:adm3a' -m 'dialup:dw2' -m 'plugboard:?adm3a'
```

To prevent interpretation as metacharacters, the entire argument to `-m` should be enclosed in single quotes. When using the C shell, exclamation points should be preceded by a backslash (`\`).

使用例

These examples all use the `-'` option. A typical use of `tset` in a `.profile` or `.login` will also use the `-e` and `-k` options, and often the `-n` or `-Q` options as well. These options have been omitted here to keep the examples short.

例 1 Selecting a terminal

To select a 2621, you might put the following sequence of commands in your `.login` file (or `.profile` for Bourne shell users).

```
set noglob
eval 'tset -s 2621'
unset noglob
```

If you want to make the selection based only on the baud rate, you might use the following:

```
set noglob
eval 'tset -s -m '>1200:wy' 2621'
unset noglob
```

例 2 Selecting terminals according to speed or baud rate

If you have a switch which connects to various ports (making it impractical to identify which port you may be connected to), and use various terminals from time to time, you can select from among those terminals according to the *speed* or baud rate. In the example below, `tset` will prompt you for a terminal type if the baud rate is greater than 1200 (say, 9600 for a terminal connected by an RS-232 line), and use a Wyse® 50 by default. If the baud rate is less than or equal to 1200, it will select a 2621. Note the placement of the question mark, and the quotes to protect the `>` and `?` from interpretation by the shell.

```
set noglob
eval 'tset -s -m 'switch>1200:?wy' -m 'switch<=1200:2621''
unset noglob
```

例 3 Selecting the terminal used most often

The following entry is appropriate if you always dial up, always at the same baud rate, on many different kinds of terminals, and the terminal you use most often is an `adm3a`.

```
set noglob
eval 'tset -s ?adm3a'
unset noglob
```

例 4 Selecting a terminal with specific settings

The following example quietly sets the erase character to BACKSPACE, and kill to CTRL-U. If the port is switched, it selects a Concept™ 100 for speeds less than or equal to 1200, and asks for the terminal type otherwise (the default in this case is a Wyse 50). If the port is a direct dialup, it selects Concept 100 as the terminal type. If logging in over the ARPANET, the terminal type selected is a Datamedia® 2500 terminal or emulator. Note the backslash escaping the NEWLINE at the end of the first line in the example.

```
set noglob
eval 'tset -e -k^U -Q -s -m 'switch<=1200:concept100' -m\
'switch:?wy' -m dialup:concept100 -m arpanet:dm2500'
```

例 4 Selecting a terminal with specific settings (続き)

ファイル

```
unset noglob
.login
.profile
/etc/termcap
```

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目

[csh\(1\)](#), [sh\(1\)](#), [stty\(1\)](#), [vi\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項

The `tset` command is one of the first commands a user must master when getting started on a UNIX system. Unfortunately, it is one of the most complex, largely because of the extra effort the user must go through to get the environment of the login shell set. Something needs to be done to make all this simpler, either the `login` program should do this stuff, or a default shell alias should be made, or a way to set the environment of the parent should exist.

This program cannot intuit personal choices for erase, interrupt and line kill characters, so it leaves these set to the local system standards.

It could well be argued that the shell should be responsible for ensuring that the terminal remains in a sane state; this would eliminate the need for the `reset` program.

名前	tsort – topological sort
形式	tsort [<i>file</i>]
機能説明	The <code>tsort</code> command produces on the standard output a totally ordered list of items consistent with a partial ordering of items mentioned in the input <i>file</i>. The input consists of pairs of items (non-empty strings) separated by blanks. Pairs of different items indicate ordering. Pairs of identical items indicate presence, but not ordering.
オペランド	The following operand is supported: <i>file</i> A path name of a text file to order. If no <i>file</i> operand is given, the standard input is used.
使用例	例 1 An example of the <code>tsort</code> command The command: <pre>example% tsort <<EOF a b c c d e g g f g e f EOF</pre> produces the output: <pre>a b c d e f g</pre>
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of <code>tsort</code> : <code>LANG</code> , <code>LC_ALL</code> , <code>LC_CTYPE</code> , <code>LC_MESSAGES</code> , and <code>NLSPATH</code> .
終了ステータス	The following exit values are returned: 0 Successful completion. >0 An error occurred.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [lorder\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 Odd data: there are an odd number of fields in the input file.

名前	tty - ユーザーの端末名の応答
形式	/usr/bin/tty [-l] [-s]
機能説明	tty ユーティリティーは、標準入力としてオープンしている端末名を標準出力に書き込みます。使用される名前は ttyname(3C) 関数によって返される文字列と同じです。
オプション	サポートしているオプションは、次のとおりです。 -l ユーザー端末がアクティブな状態の同期回線につながっている場合は、ユーザー端末に接続されている同期回線番号を出力します。 -s 端末パス名は出力しませんが、終了ステータスをテストできるようにします。
環境変数	tty の実行に影響を与える次の環境変数の詳細については、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
終了ステータス	次の終了ステータスが返されます。 0 標準入力は端末である。 1 標準入力は端末ではない。 >1 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目	isatty(3C) , ttyname(3C) , attributes(5) , environ(5) , standards(5)
診断	not on an active synchronous line 標準入力は同期端末ではなく、-l が指定されています。 not a tty 標準入力は端末ではなく、-s は指定されていません。
注意事項	-s オプションは終了ステータスが必要な場合のみ有効です。有効なパス名を表示することはできません。移植性の必要なアプリケーションには <code>test -t</code> を使用してください。

名前	type - コマンドのタイプの検査
形式	type <i>name</i> ...
機能説明	type ユーティリティーは、オペランドに指定された各 <i>name</i> が、コマンドとして使用された場合どのように解釈されるかをチェックします。具体的には、おのこの <i>name</i> オペランドがシェルの組み込みコマンド、関数、別名、ハッシュドコマンド、キーワードのいずれであるかを表示し、さらに (該当するものについては) オペランドのパス名を表示します。 type というシェルの組み込みコマンドもあります。機能は type ユーティリティーと同様です。
オペランド	次のオペランドを指定できます。 <i>name</i> 解釈される名前
環境変数	type の実行に影響を与える次の環境変数についての詳細は、environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。 PATH <i>name</i> の位置を表す
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [typeset\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	typeset, whence – シェル変数と関数の属性と値を設定または取得するためのシェル組み込み関数
形式	typeset [-CDHLRZfirltux [<i>n</i>] [<i>name</i> [= <i>value</i>]]]... whence [-pv] <i>name</i> ...
機能説明	typeset はシェル変数と関数の属性と値を設定します。関数内で typeset を実行すると、<i>name</i> が示す変数の新しいインスタンスが生成されます。関数が完了すると、その変数の値と型が復元されます。サポートされている属性は次のとおりです。 -C 複合変数。各名前は複合変数です。<i>value</i> によって複合変数の名前が指定されると、その値が <i>name</i> にコピーされます。それ以外の場合で変数がすでに存在するとき、その変数は最初に未設定になります。 -D これらのフィールドは、将来的な使用のために予約されています。 -H UNIX 以外のマシン上で、UNIX とホスト名ファイルとのマッピング情報を提供します。 -L 左詰めを行い、先行する空白文字を <i>value</i> から取り除きます。<i>n</i> は、ゼロ以外であればフィールドの幅を定義します。そうでなければ、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ右側にブランクが詰められ、長ければ切り捨てられます。<i>-z</i> フラグも設定されていれば、先行する 0 を削除します。<i>-R</i> フラグは無効になります。 -R 右詰めを行い、先行する空白文字を挿入します。<i>n</i> は、0 以外であればフィールドの幅を定義します。0 の場合、フィールドの幅は最初に代入される値の幅で決定されます。変数に値を代入したとき、フィールド幅より短ければ左側にブランクが詰められ、長ければ終端が切り捨てられます。<i>-L</i> フラグは無効になります。 -Z 最初の、空白文字でない文字が数字で、かつ <i>-L</i> フラグが設定されていない場合、右詰めを行い先頭に 0 を詰めます。<i>n</i> は、ゼロ以外であればフィールドの幅を定義します。そうでなければ、フィールドの幅は最初に代入される値の幅で決定されます。 -f 大文字をすべて小文字に変換します。大文字フラグ <i>-u</i> は無効になります。 関数が参照されると、関数定義を見つけるために <i>FPATH</i> 変数が検索されます。<i>-x</i> フラグを指定すると、名前で呼び出されるシェル手続き全体で関数定義が有効になります。 -i パラメータを整数とします。これにより算術演算が高速化されます。<i>n</i> は、ゼロ以外であればその値を底として定義します。それ以外の場合は、最初の代入によってその値の底が判断されます。 -l 大文字をすべて小文字に変換します。大文字フラグ <i>-u</i> は無効になります。

- m 移動。value は、値が name に移動される変数の名前です。元の変数は未設定になります。このオプションは、その他のオプションと同時に使用できません。
- r 指定された名前は読み取り専用としてマークされ、これらの名前は、以降の代入で変更できなくなります。
- t 変数にタグを付けます。タグはユーザーが定義可能で、シェルに対して特別の意味を持ちません。
- u 小文字をすべて大文字に変換します。小文字フラグ -l は無効になります。
- x 指定された名前に対し、あとで実行されるコマンドの環境へ自動的にエクスポートされるようにマークを付けます。

i 属性は、--R、-L、-Z、または -f と同時に指定することはできません。

- の代わりに + を使用すると、これらのフラグは無効になります。name 引数をまったく指定せずにフラグを指定すると、これらのフラグが設定されている変数の名前（および選択により値も）が一覧表示されます。- の代わりに + を使用すると、値は出力されません。name とフラグを 1 つも指定しないと、すべての変数の名前と属性が出力されます。

whence コマンドは、name ごとに、コマンド名として使用される場合にどのように解釈されるかを指示します。

-v フラグをつけると、より冗長に表示されます。

-p フラグは、name が別名、関数名、または予約語の場合でも、name に対するパス検索を行います。

このマニュアルページでは、1 つまたは 2 つの * (アスタリスク) が先頭に付加されている [ksh\(1\)](#) コマンドは、次のような特殊な処理を受けます。

1. コマンドが完了しても、コマンドの直前の変数代入リストは依然として有効です。
2. 入出力のリダイレクトは変数代入後に行われます。
3. エラーが発生すると、それを含むスクリプトは中止されます。
4. 変数代入形式で、** を先頭に持つコマンドに続くワードは、変数代入と同一の規則で展開されます。つまり、チルド置換は = 符号のあとに実行され、ワード分割とファイル名生成は実行されません。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目

[ksh\(1\)](#), [ksh88\(1\)](#), [set\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#)

名前	ul – do underlining				
形式	ul [-i] [-t <i>terminal</i>] [<i>filename</i>]...				
機能説明	ul reads the named <i>filenames</i> (or the standard input if none are given) and translates occurrences of underscores to the sequence which indicates underlining for the terminal in use, as specified by the environment variable TERM. ul uses the /usr/share/lib/terminfo entry to determine the appropriate sequences for underlining. If the terminal is incapable of underlining, but is capable of a standout mode then that is used instead. If the terminal can overstrike, or handles underlining automatically, ul degenerates to cat(1) . If the terminal cannot underline, underlining is ignored.				
オプション	-t <i>terminal</i> Override the terminal kind specified in the environment. If the terminal cannot underline, underlining is ignored. If the terminal name is not found, no underlining is attempted. -i Indicate underlining by a separate line containing appropriate dashes ‘–’; this is useful when you want to look at the underlining which is present in an nroff(1) output stream on a CRT-terminal.				
戻り値	ul returns exit code 1 if the file specified is not found.				
ファイル	/usr/share/lib/terminfo/*				
属性	See attributes(5) for descriptions of the following attributes:				
	<table><tr><th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr><tr><td>Availability</td><td>text/doctools</td></tr></table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	text/doctools
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	text/doctools				
関連項目	cat(1) , man(1) , nroff(1) , attributes(5)				
使用上の留意点	nroff usually generates a series of backspaces and underlines intermixed with the text to indicate underlining. ul makes attempt to optimize the backward motion.				

名前	umask – ファイルモード生成マスクの獲得と設定
形式	/usr/bin/umask [-S] [<i>mask</i>]
sh	umask [<i>ooo</i>]
csch	umask [<i>ooo</i>]
ksh88	umask [-S] [<i>mask</i>]
ksh	umask [-S] [<i>mask</i>]
機能説明	umask ユーティリティーは、現在のシェル実行環境のファイルモード生成マスクを <i>mask</i> オペランドが示す値に設定します。このマスクは、以降生成されるファイルのファイル許可ビットの初期値に影響します。次のように、umask がサブシェルや別のユーティリティー実行環境で呼び出された場合は、 <pre>(umask 002) nohup umask ... find . -exec umask ...</pre> 呼び出し側の環境のファイルモード生成マスクには影響しません。このため /usr/bin/umask ユーティリティーは、進行中のセッションにある umask の変更には使用できません。呼び出し側の umask を確認するためだけに使用できます。進行中のセッションにある umask を変更するには、シェルの組み込みコマンドを使用する必要があります。 <i>mask</i> オペランドが省略されると、umask ユーティリティーは呼び出し側プロセスのファイルモード生成マスクの値を標準出力に書き出します。 sh ユーザーのファイル生成モードマスクを <i>ooo</i> で示す値に設定します。<i>ooo</i> は3つの8進数で、左から順番に所有者、グループ、その他のユーザー用の、読み取り/書き込み/実行権を表します。詳しくは chmod(1)、chmod(2)、umask(2) の説明を参照してください。umask で指定したそれぞれの値は、ファイル生成時にシステムが指定する値から減算されます (creat(2) を参照)。たとえば umask 022 と指定すると、グループとその他のユーザーに対する書き込み権が取り除かれます。モード 777 で標準的に生成されたファイル(およびディレクトリ)はモード 755 になります。モード 666 で生成されたファイル(およびディレクトリ)はモード 644 になります。 ■ <i>ooo</i> 引数を省略すると、マスクとして用いられている現在の値が出力されます。 ■ umask はシェルにより認識され実行されます。 ■ ユーザーの .profile に umask を登録しておくと (profile(4) を参照)、ログイン時に実行され、生成されたファイルやディレクトリに対するユーザーのアクセス権モードが自動的に設定されます。 csch 前述の、sh の umask 組み込みコマンドについての説明を参照してください。

ksh88 ユーザーファイルの作成時のマスクを *mask* 引数が示す値に設定します。*mask* には、[chmod\(1\)](#) で説明する記号値または8進数を指定できます。記号値を指定すると、新しい *umask* 値は、*mask* を直前の *umask* 値の補数に適用した結果の補数になります。*mask* 引数を省略すると、マスクの現在の値を表示します。

ksh *umask* は、現在のシェル実行環境のファイル生成マスクを *mask* オペランドが示す値に設定します。このマスクは、以降生成されるファイルのファイル許可ビットに影響します。*mask* には、[chmod\(1\)](#) で説明する記号値または8進数を指定できます。記号値を指定すると、新しいファイル生成マスクは、*mask* を現在のファイル生成マスクの補数に適用した結果の補数になります。*mask* が省略されると、*umask* は現在のプロセスのファイル生成マスクの値を標準出力に書き出します。

オプション

ksh88 次のオプションは、ksh88 の `/usr/bin/umask` と *umask* で使用できます。

-s 記号出力を生成します。

デフォルトの出力形式は不定ですが、同じシステム上の以降に *umask* が呼び出されたときには、以前のファイルモード生成マスクを復元する *mask* オペランドとして認識されます。

ksh 次のオプションは、ksh で指定できます。

-s ファイル生成マスクが、8進数ではなく記号値として設定または使用されません。

オペランド 次のオペランドを指定できます。

マスク 新しいファイルモード生成マスクを指定する文字列。この文字列は [chmod\(1\)](#) マニュアルページで説明されている *mode* オペランドと同じ方法で扱われます。

記号値の場合、ファイルモード生成マスクの新しい値は、記号文字列が指定しているファイルモードのファイル許可ビット部分の論理補数です。

記号値の許可 *op* 文字 + と - は、現在のファイルモード生成マスクとの相対的な関係で解釈されます。+ は、示された許可をマスクからクリアします。- は、示された許可ビットをマスクに設定します。

ファイル許可ビット以外のファイルモードビットを指定する *mode* 値の解釈は不定です。

ファイルモード生成マスクは、結果の数値に設定されます。

同じシステム上で以前にオペランドを指定せずに *umask* を実行した場合のデフォルト出力も、*mask* オペランドとして認識されます。この方法

で得たオペランドは、たとえ 8 進数であっても、無効ではありません。

出力

mask オペランドが省略されると、*umask* ユーティリティーは、あとで *umask* の *mask* オペランドとして使用可能なメッセージを標準出力に書き出します。

-s が指定されると、メッセージは次の形式になります。

```
"u=%s,g=%s,o=%s\n", owner permissions, group permissions, \
other permissions
```

3 つの値は、{r, w, x} セットの文字の組み合わせです。文字がある場合は、ファイルモード生成マスク内の対応するビットが設定されていないことを示しています。

mask オペランドが指定されると、標準出力には何も出力されません。

使用例

例 1 *umask* コマンドの使用

このセクションの例では、*/usr/bin/umask* ユーティリティーと *ksh88* の組み込み *umask* について説明します。

次のコマンドを見てください。

```
umask a=rx,ug+w
umask 002
```

これらのコマンドは、どちらも以降生成されるファイルの *S_IWOTH* ビットがクリアされるように、モードマスクを生成します。

前述のコマンドはどちらもモードマスクを設定したあと、*umask* コマンドを使用してモードマスクの現在の値を書き出すことができます。

```
example$ umask
0002
```

出力形式が指定されていませんが、これまでの実装では従来の 8 進整数モードの書式が使用されます。

```
example$ umask -S
u=rwx,g=rwx,o=rx
```

この出力はどちらも、以降の *umask* 呼び出しで *mask* オペランドとして使用できません。

前述のようにモードマスクが設定されていると想定した場合、

```
umask g-w
```

このコマンドは、以降生成されるファイルの *S_IWGRP* ビットと *S_IWOTH* ビットがクリアされるように、モードマスクを設定します。

例1 umask コマンドの使用 (続き)

次のコマンド

umask --w

このコマンドは、以降生成されるファイルの書き込みビットがクリアされるように、モードマスクを設定します。なお *mask* オペランドの *r*、*w*、*x*、またはハイフン (-) で始まるものは、オプションと間違われないようにその前に - を付ける必要があります。

環境変数 umask の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_COLLATELC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 ファイルモード生成マスクが正常に変更された、または *mask* がなかった。
- >0 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

/usr/bin/umask、csh、ksh88、sh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	確実
	標準	standards(5) を参照してください。

ksh	属性タイプ	属性値
	使用条件	system/core-os
	インタフェースの安定性	流動的

関連項目 [chmod\(1\)](#), [csh\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [sh\(1\)](#), [chmod\(2\)](#), [creat\(2\)](#), [umask\(2\)](#), [profile\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	uname – 現在のシステム名の表示
形式	uname [-aimnprsvX] uname [-S <i>system_name</i>]
機能説明	uname ユーティリティーは、現在のシステムに関する情報を標準出力に出力します。オプションを指定した場合、1 つ以上のシステム特性を表すシンボルが標準出力に書き込まれます。オプションを指定しない場合、uname は、現在のオペレーティングシステムのシステム名を出力します。オプションを指定すると、uname(2) と sysinfo(2) (またはいずれか一方) が返した選択情報を出力します。
オプション	サポートしているオプションは、次のとおりです。 -a システムから現在、入手できる基本情報を出力します。 -i プラットフォーム名を出力します。sun4v アーキテクチャーのマシンの場合、-i オプションで sun4v が返されます。sun4v マシンのプラットフォーム名を取得するには、prtconf(1M) を -b オプションを付けて実行します。 -m コンピュータのハードウェア名 (クラス) を出力します。このオプションを使用するのは望ましくありません。代わりに uname -p を指定してください。(「注意事項」のセクションを参照) -n ノード名を出力します (ノード名とは、システムが通信ネットワークに認識されている名前です)。 -p 現在のホストの ISA またはプロセッサ型を出力します。 -r オペレーティングシステムのリリースレベルを出力します。 -s オペレーティングシステムのシステム名を出力します。これはデフォルトです。 -S <i>system_name</i> システム名の引数を指定することによって、ノード名を変更できます。システム名の引数は、SYS_NMLN の文字数に限られています。SYS_NMLN は、<sys/utsname.h> に定義された実装特有の値です。スーパーユーザーだけがこの機能を使用できます。このオプションを使用して変更した内容は、システムのリブート後には無効になります。 -v オペレーティングシステムのバージョンを出力します。

- X
- システムの拡張情報を出力します。SCO UNIX による情報と同じように、1 行に 1 つの情報が出力されます。表示される情報には次のものが含まれます。
- システム名、ノード、リリース、バージョン、マシン、CPU 番号
 - バス形式、シリアル、およびユーザー (Solaris では unknown に設定)
 - OEM 番号 (0 に設定) とオリジナルの番号 (1 に設定)

使用例

例1 オペレーティングシステム名とリリースレベルの表示

次のコマンドを実行すると、オペレーティングシステム名とリリースレベルが1文字のスペース文字で区切られて出力されます。

```
example% uname -sr
```

環境変数

uname の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

次の終了ステータスが返されます。

0

正常終了。

>0

エラーが発生した。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[arch\(1\)](#), [isalist\(1\)](#), [prtconf\(1M\)](#), [sysinfo\(2\)](#), [uname\(2\)](#), [nodename\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

独立ソフトウェアベンダー (ISV) やその他のベンダーで、それぞれの提供するソフトウェアがインストールされているまたは実行されているプラットフォームの詳しい特性を知りたい場合は、uname コマンドを使用してください。

オペレーティングシステム名とオペレーティングシステムのリリースレベルを表示するには、uname -sr を使用してください。オペレーティングシステムのリリースレベルだけを表示するには、uname -r を使用してください。オペレーティングシステムのリリースレベルは x.y 形式 (たとえば 5.3、5.4、5.5 など) に限らず、x.y.z 形式 (たとえば 5.3.1、5.3.2、5.4.1 など) でも表示される場合があります。

SunOS 4.x リリースにおいて、`uname` コマンドと同様の情報を得るために `arch(1)` コマンドがよく使用されていました。この `arch(1)` コマンドの `sun4` という出力が SunOS SPARC システムを示すものとして誤って解釈されることがありました。ハードウェアプラットフォームの情報を得るには、`uname -sp` を使用してください。

`arch -k` コマンドと `uname -m` コマンドは同じ値を返します。ただし、一般的に `arch` コマンドの使用が望ましくないのと同様に、この 2 つのコマンドをサードパーティのプログラム上で使用することは望ましくありません。マシンの Instruction Set Architecture (ISA またはプロセッサ型) を確認するには、`uname -p` を使用してください。

名前	unifdef – resolve and remove ifdefed lines from C program source
形式	<pre>unifdef [-clt] [-Dname] [-Uname] [-iDname] [-iUname] ... [filename]</pre>
機能説明	unifdef removes ifdefed lines from a file while otherwise leaving the file alone. It is smart enough to deal with the nested ifdefs, comments, single and double quotes of C syntax, but it does not do any including or interpretation of macros. Neither does it strip out comments, though it recognizes and ignores them. You specify which symbols you want defined with -D options, and which you want undefined with -U options. Lines within those ifdefs will be copied to the output, or removed, as appropriate. Any ifdef, ifndef, else, and endif lines associated with <i>filename</i> will also be removed. ifdefs involving symbols you do not specify are untouched and copied out along with their associated ifdef, else, and endif lines. If an ifdefX occurs nested inside another ifdefX, then the inside ifdef is treated as if it were an unrecognized symbol. If the same symbol appears in more than one argument, only the first occurrence is significant. unifdef copies its output to the standard output and will take its input from the standard input if no <i>filename</i> argument is given.
オプション	The following options are supported: -c Complement the normal operation. Lines that would have been removed or blanked are retained, and vice versa. -l Replace “lines removed” lines with blank lines. -t Plain text option. unifdef refrains from attempting to recognize comments and single and double quotes. -Dname Lines associated with the defined symbol <i>name</i>. -Uname Lines associated with the undefined symbol <i>name</i>. -iDname Ignore, but print out, lines associated with the defined symbol <i>name</i>. If you use ifdefs to delimit non-C lines, such as comments or code which is under construction, then you must tell unifdef which symbols are used for that purpose so that it will not try to parse for quotes and comments within them. -iUname Ignore, but print out, lines associated with the undefined symbol <i>name</i>.
終了ステータス	The following exit values are returned: 0 Successful operation. 1 Operation failed.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/base-developer-utilities

関連項目

[diff\(1\)](#), [attributes\(5\)](#)

診断

Premature EOF Inappropriate else or endif.

名前	uniq - ファイルの中の重複行の報告またはフィルタへの出力
形式	<pre>/usr/bin/uniq [-c -d -u] [-f <i>fields</i>] [-s <i>char</i>] [<i>input_file</i> [<i>output_file</i>]] /usr/bin/uniq [-c -d -u] [-n] [+m] [<i>input_file</i> [<i>output_file</i>]]</pre>
機能説明	uniq は、入力を読み取り、隣接する行を比較して、各入力行のコピーを 1 行ずつ出力に書き込みます。隣接する重複行の 2 番目以降の行は書き込まれません。 出力ファイル <i>output_file</i> が指定されていない場合、uniq は標準出力に書き込みます。<i>input_file</i> が指定されていない場合、または <i>input_file</i> が - の場合、uniq は、現在のオフセットをファイルの開始位置として定義し、標準入力から読み取ります。
オプション	サポートしているオプションは、次のとおりです。 -c 各行の先頭に、その行が入力中に連続して現れた回数を出力します。 -d 入力中で重複していなかった行は出力しません。 -f 比較を行う際に、各入力行の最初の <i>fields</i> <i>fields</i> を無視します。<i>fields</i> は正の整数です。<i>field</i> は、基本的な正規表現で一致する最大の文字列です。 <pre>[[[:blank:]]*[^[:blank:]]*</pre>入力行にあるより多くの <i>fields</i> を <i>fields</i> に指定した場合、比較には NULL 文字列が使われます。 -s 比較を行う際に、最初の <i>chars</i> 文字を無視します。<i>chars</i> は正の整数です。-f オプションと一緒に指定した場合には、先頭の <i>fields</i> <i>fields</i> のあとの <i>chars</i> 文字が無視されます。入力行に残っているより多くの文字を <i>chars</i> に指定した場合、比較には NULL 文字列が使われます。 -u 入力中で重複していた行は出力しません。 -n <i>fields</i> の値が <i>n</i> である -f <i>fields</i> 指定と同じ意味です。 +m <i>chars</i> の値が <i>m</i> である -s <i>chars</i> 指定と同じ意味です。
オペランド	次のオペランドがサポートされています。 <i>input_file</i> 入力ファイルのパス名。<i>input_file</i> が指定されない場合、または <i>input_file</i> が - の場合は、標準入力を使用されます。 <i>output_file</i> 出力ファイルのパス名。<i>output_file</i> が指定されない場合、標準出力が使用されます。<i>input_file</i> と同じファイルを <i>output_file</i> で指定した場合、処理の結果は予測できません。

使用例

例1 uniq コマンドの使用

次の例では `uniq.test` ファイルの内容をリストで示し、重複行を出力しています。

```
example% cat uniq.test
This is a test.
This is a test.
TEST.
Computer.
TEST.
TEST.
Software.
```

```
example% uniq -d uniq.test
This is a test.
TEST.
example%
```

次の例では `uniq.test` ファイルで重複していない行だけを出力しています。

```
example% uniq -u uniq.test
TEST.
Computer.
Software.
example%
```

最後の例では、ファイルの中でそれぞれの行が現われた回数を 各行の先頭に付けて出力しています。

```
example% uniq -c uniq.test
  2 This is a test.
  1 TEST.
  1 Computer.
  2 TEST.
  1 Software.
example%
```

環境変数

`uniq` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

次の終了ステータスが返されます。

```
0      正常終了。
>0     エラーが発生した。
```

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目

[comm\(1\)](#), [pack\(1\)](#), [pcat\(1\)](#), [sort\(1\)](#), [uncompress\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	units – converts quantities expressed in standard scales to other scales
形式	units
機能説明	units converts quantities expressed in various standard scales to their equivalents in other scales. It works interactively in this fashion: <pre>You have:~~inch You want:~~cm * 2.540000e+00 / 3.937008e-01</pre> A quantity is specified as a multiplicative combination of units optionally preceded by a numeric multiplier. Powers are indicated by suffixed positive integers, division by the usual sign: <pre>You have:~~15 lbs force/in2 You want:~~atm * 1.020689e+00 / 9.797299e-01</pre> units only does multiplicative scale changes; thus it can convert Kelvin to Rankine, but not Celsius to Fahrenheit. Most familiar units, abbreviations, and metric prefixes are recognized, together with a generous leavening of exotica and a few constants of nature including: pi ratio of circumference to diameter, c speed of light, e charge on an electron, g acceleration of gravity, force same as g, mole Avogadro's number, water pressure head per unit height of water, au astronomical unit. Pound is not recognized as a unit of mass; lb is. Compound names are run together, (for example, lightyear). British units that differ from their U.S. counterparts are prefixed thus: brgallon. For a complete list of units, type: <pre>cat /usr/share/lib/unittab</pre>
ファイル	/usr/share/lib/unittab
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [attributes\(5\)](#)

名前	unix2dos – convert text file from ISO format to DOS format
形式	<pre> unix2dos [-ascii] [-iso] [-7] [-437 -850 -860 -863 -865] <i>originalfile</i> <i>convertedfile</i> </pre>
機能説明	The <code>unix2dos</code> utility converts ISO standard characters to the corresponding characters in the DOS extended character set. This command may be invoked from either DOS or SunOS. However, the filenames must conform to the conventions of the environment in which the command is invoked. If the original file and the converted file are the same, <code>unix2dos</code> will rewrite the original file after converting it.
オプション	The following options are supported: <code>-ascii</code> Adds carriage returns and converts end of file characters in SunOS format text files to conform to DOS requirements. <code>-iso</code> This is the default. Converts ISO standard characters to the corresponding character in the DOS extended character set. <code>-7</code> Converts 8 bit SunOS characters to 7 bit DOS characters. On non-i386 systems, <code>unix2dos</code> will attempt to obtain the keyboard type to determine which code page to use. Otherwise, the default is US. The user may override the code page with one of the following options: <code>-437</code> Use US code page <code>-850</code> Use multilingual code page <code>-860</code> Use Portuguese code page <code>-863</code> Use French Canadian code page <code>-865</code> Use Danish code page
オペランド	The following operands are required: <i>originalfile</i> The original file in ISO format that is being converted to DOS format. <i>convertedfile</i> The new file in DOS format that has been converted from the original ISO file format.
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目

[dos2unix\(1\)](#), [ls\(1\)](#), [attributes\(5\)](#)

診断

File *filename* not found, or no read permission

The input file you specified does not exist, or you do not have read permission. Check with the SunOS command, `ls -l` (see [ls\(1\)](#)).

Bad output filename *filename*, or no write permission

The output file you specified is either invalid, or you do not have write permission for that file or the directory that contains it. Check also that the drive is not write-protected.

Error while writing to temporary file

An error occurred while converting your file, possibly because there is not enough space on the current drive. Check the amount of space on the current drive using the DIR command. Also be certain that the default drive is write-enabled (not write-protected). When this error occurs, the original file remains intact.

Translated tmpfile name = *filename*.

Could not rename tmpfile to *filename*.

The program could not perform the final step in converting your file. Your converted file is stored under the name indicated on the second line of this message.

名前	updatehome – 現在のラベル用のホームディレクトリのコピーファイルとリンクファイルの更新
形式	/usr/bin/updatehome [-cirs]
機能説明	updatehome は、ユーザーの最下位ラベルのコピーおよびリンク制御ファイル (.copy_files と .link_files) を読み込みます。これらのファイルには、コピーするファイルの一覧、およびユーザーの最下位ラベルのホームディレクトリから現在のラベルのホームディレクトリへシンボリックリンクするファイルの一覧が含まれます。 デフォルトでは、最下位ユーザーラベルは <code>label_encodings(4)</code> に指定され、<code>user_attr(4)</code> に明示的に指定できます。<code>txzonemgr(1M)</code> を使用して作成した場合、<code>public</code> ゾーンはデフォルトの最下位ラベルに割り当てられ、マルチレベル NFS サーバーとして構成されます。<code>public</code> ゾーンの承認された管理者は <code>share(1M)</code> コマンドを使用すると、より高レベルのゾーンによって読み取り専用モードでマウントできるようにホームディレクトリをエクスポートできます。<code>txzonemgr(1M)</code> を使用して作成された追加のゾーンは、<code>/zone/public/home</code> にある <code>public</code> ゾーンのホームディレクトリをマウントする <code>automount(1M)</code> エントリを使用して構成されます。 この方法でユーザーの最下位ホームディレクトリが共有されている場合、ユーザーは <code>updatehome</code> コマンドを手動または起動ファイルで実行すると、より高レベルのゾーンで実行できます。たとえば、たいていの場合、ユーザーは <code>.profile</code>、<code>.login</code>、<code>.cshrc</code>、<code>.exrc</code>、<code>.mailrc</code>、および <code>~/bin</code> などのファイルへのシンボリックリンクを必要とします。<code>updatehome</code> は、このシンボリックリンクを実現する便利なメカニズムを提供します。ユーザーは、コピーするファイルのリスト (.copy_files) およびシンボリックリンクを設定するファイルのリスト (.link_files) にファイルを追加できます。
オプション	-c 現在のラベルを持つ既存のホームディレクトリのコピーを置換します。デフォルトでは、既存のコピーの置換は実行しません。 -i エラーが発生しても無視します。デフォルトでは、エラー発生時は異常終了します。 -r 現在のラベルを持つ既存のホームディレクトリのコピーまたはシンボリックリンクを置換します。このオプションは、<code>-c</code> および <code>-s</code> オプションを合わせた意味を持ちます。デフォルトでは、既存のコピーやシンボリックリンクの置換は実行しません。 -s 現在のラベルを持つ既存のホームディレクトリのシンボリックリンクを置換します。デフォルトでは、既存のシンボリックリンクの置換は実行しません。
終了ステータス	正常終了時に、updatehome は 0 を返します。エラー終了時には、updatehome は 1 を返して、診断メッセージを標準エラー出力に書き込みます。

使用例

例1 .copy_files ファイルの例

.copy_files に記述されているファイルは、すべてのユーザーラベルで変更できます。

```
.cshrc
.mailrc
.mozilla/bookmarks.html
```

例2 .link_files ファイルの例

.link_files に記述されているファイルは、最下位のラベルで変更できます。変更は、ユーザーが利用可能なほかのラベルに伝達されます。

```
~/bin
.mozilla/preferences
.xrc
.rhosts
```

例3 リンクファイルとコピーファイルを更新する

.copy_files および .link_files は、ユーザーにより最下位のラベルで更新されました。ユーザーは、より高位のラベルでコピーおよびリンクをリフレッシュします。このコマンドを実行するのに特権は不要です。

```
% updatehome -r
```

ファイル

```
$HOME/.copy_files
```

 コピーするファイルのリスト

```
$HOME/.link_files
```

 シンボリックリンクを作成するファイルのリスト

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/trusted
インタフェースの安定性	確実

関連項目

[automount\(1M\)](#), [share\(1M\)](#), [txzonemgr\(1M\)](#), [label_encodings\(4\)](#), [user_attr\(4\)](#), [attributes\(5\)](#)

『[Trusted Extensions 構成と管理](#)』の「[.copy_files ファイルと .link_files ファイル](#)」

注意事項

このマニュアルページに記載されている機能を利用できるのは、システムが Trusted Extensions を使用して構成されている場合のみです。

名前 uptime – show how long the system has been up

形式 uptime

機能説明 The uptime command prints the current time, the length of time the system has been up, and the average number of jobs in the run queue over the last 1, 5 and 15 minutes. It is, essentially, the first line of a [w\(1\)](#) command.

使用例 Below is an example of the output uptime provides:

```
example% uptime
10:47am up 27 day(s), 50 mins, 1 user, load average: 0.18, 0.26, 0.20
```

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [w\(1\)](#), [who\(1\)](#), [whodo\(1M\)](#), [attributes\(5\)](#)

注意事項 who -b gives the time the system was last booted.

名前	userattr – ユーザーまたは役割に付与された属性値の出力
形式	userattr [-v] attribute_name [user]
機能説明	userattr コマンドは、属性 attribute_name で最初に見つかった値を標準出力に出力します。ユーザーが指定されていない場合、ユーザーはプロセスの実際のユーザー ID から取得されます。属性名は、user_attr(4) および prof_attr(4) で定義されている中で見つかるものになります。ユーザーにプロファイルが割り当てられている場合は、profiles(1) コマンドを使用します。ユーザーに承認が割り当てられている場合は、auths(1) コマンドを使用します。検索の順序は、ユーザーの user_attr エントリのあとにユーザーのプロファイルが続きます。 属性 attribute_name がユーザーに割り当てられていない場合にエラーが発生すると、userattr は 0 以外の終了コードを返します。それ以外の場合、userattr は 0 の終了コードを返します。 -v オプションを使用すると、属性が見つかった場所が追加で出力されます。
使用例	例1 userattr の使用 <pre>example% userattr lock_after_retries root no</pre>
ファイル	/etc/user_attr /etc/security/policy.conf /etc/security/prof_attr
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 1 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
インタフェースの安定性	下記を参照。

終了コードは「確実」です。-v オプションの出力は「非インタフェース」です。

関連項目 [auths\(1\)](#), [profiles\(1\)](#), [policy.conf\(4\)](#), [prof_attr\(4\)](#), [user_attr\(4\)](#), [attributes\(5\)](#)

名前 users – display a compact list of users logged in

形式 /usr/ucb/users [*filename*]

機能説明 The users utility lists the login names of the users currently on the system in a compact, one-line format.

Specifying *filename* tells users where to find its information; by default it checks /var/adm/utmpx.

Typing users is equivalent to typing who -q.

使用例 例 1 Listing current users

```
example% users
paul george ringoexample%
```

ファイル /var/adm/utmpx

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [who\(1\)](#), [attributes\(5\)](#)

名前 uucp, uulog, uuname – UNIX-to-UNIX system copy

形式 uucp [-c | -C] [-d | -f] [-ggrade] [-jmr] [-nuser] [-sfile]
 [-xdebug_level] source-file destination-file

uulog [-ssys] [-fssystem] [-x] [-number] system

uuname [-c | -l]

機能説明

uucp The uucp utility copies files named by the *source-file* arguments to the *destination-file* argument.

uulog The uulog utility queries a log file of uucp or uuxqt transactions in file /var/uucp/.Log/uucico/system or /var/uucp/.Log/uuxqt/system.

uuname The uuname utility lists the names of systems known to uucp.

オプション

uucp The following options are supported by uucp:

- c Does not copy local file to the spool directory for transfer to the remote machine (default).
- C Forces the copy of local files to the spool directory for transfer.
- d Makes all necessary directories for the file copy (default).
- f Does not make intermediate directories for the file copy.
- g *grade* *grade* can be either a single letter, number, or a string of alphanumeric characters defining a service grade. The `uuglist` command can determine whether it is appropriate to use the single letter, number, or a string of alphanumeric characters as a service grade. The output from the `uuglist` command is a list of service grades that are available, or a message that says to use a single letter or number as a grade of service.
- j Prints the uucp job identification string on standard output. This job identification can be used by `uustat` to obtain the status of a uucp job or to terminate a uucp job. The uucp job is valid as long as the job remains queued on the local system.
- m Sends mail to the requester when the copy is complete.
- n *user* Notifies *user* on the remote system that a file was sent.

When multiple -n options are passed in, uucp only retains the value specified for the last -n option. This is the only user notified.
- r Does not start the file transfer, just queue the job.

- s *file* Reports status of the transfer to *file*. This option is accepted for compatibility, but it is ignored because it is insecure.
- x *debug_level* Produce debugging output on standard output. *debug_level* is a number between 0 and 9. As *debug_level* increases to 9, more detailed debugging information is given. This option may not be available on all systems.

uulog

The following options cause uu`log` to print logging information:

- s *sys* Prints information about file transfer work involving system *sys*.
- f *system* Executes a `tail -f` command of the file transfer log for *system*. You must press BREAK to exit this function.

Other options used in conjunction with the above options are:

- x Looks in the uuxqt log file for the given system.
- number* Executes a `tail` command of *number* lines.

uuname

The following options are supported by uuname:

- c Displays the names of systems known to cu. The two lists are the same, unless your machine is using different `Systems` files for cu and uucp. See the `Sysfiles` file.
- l Displays the local system name.

オペランド

The source file name may be a path name on your machine, or may have the form:

system-name!pathname

where *system-name* is taken from a list of system names that uucp knows about. *source_file* is restricted to no more than one *system-name*. The destination *system-name* may also include a list of system names such as

system-name!system-name!...!system-name!pathname

In this case, an attempt is made to send the file, using the specified route, to the destination. Care should be taken to ensure that intermediate nodes in the route are willing to forward information. See NOTES for restrictions.

For C-Shell users, the exclamation point (!) character must be surrounded by single quotes ('), or preceded by a backslash (\).

The shell metacharacters ?, * and [...] appearing in *pathname* are expanded on the appropriate system.

Pathnames may be one of the following:

1. An absolute pathname.

2. A pathname preceded by `~user` where *user* is a login name on the specified system and is replaced by that user's login directory.
3. A pathname preceded by `~/destination` where *destination* is appended to `/var/spool/uucppublic`. This destination is treated as a filename unless more than one file is being transferred by this request or the destination is already a directory. To ensure that the destination is a directory, follow it with a forward slash (/). For example, `~/dan/` as the destination creates the directory `/var/spool/uucppublic/dan` if it does not exist and put the requested file(s) in that directory.

Anything else is prefixed by the current directory.

If the result is an erroneous path name for the remote system, the copy fails. If the *destination-file* is a directory, the last part of the *source-file* name is used.

Invoking `uucp` with shell wildcard characters as the remote *source-file* invokes the `uux(1C)` command to execute the `uucp` command on the remote machine. The remote `uucp` command spools the files on the remote machine. After the first session terminates, if the remote machine is configured to transfer the spooled files to the local machine, the remote machine initiates a call and send the files; otherwise, the user must "call" the remote machine to transfer the files from the spool directory to the local machine. This call can be done manually using `Uutry(1M)`, or as a side effect of another `uux(1C)` or `uucp` call.

Notice that the local machine must have permission to execute the `uucp` command on the remote machine in order for the remote machine to send the spooled files.

`uucp` removes execute permissions across the transmission and gives `0666` read and write permissions (see `chmod(2)`).

環境変数

See `environ(5)` for descriptions of the following environment variables that affect the execution of `uucp`: `LANG`, `LC_ALL`, `LC_COLLATE`, `LC_CTYPE`, `LC_MESSAGES`, `LC_TIME`, `NLSPATH`, and `TZ`.

終了ステータス

The following exit values are returned:

- `0` Successful completion.
- `>0` An error occurred.

ファイル

<code>/etc/uucp/*</code>	other data files
<code>/var/spool/uucp</code>	spool directories
<code>/usr/lib/uucp/*</code>	other program files
<code>/var/spool/uucppublic/*</code>	public directory for receiving and sending

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp
Interface Stability	Committed
Standard	See standards(5) .

関連項目

[mail\(1\)](#), [uuglist\(1C\)](#), [uustat\(1C\)](#), [uux\(1C\)](#), [Uutry\(1M\)](#), [uuxqt\(1M\)](#), [chmod\(2\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項

For security reasons, the domain of remotely accessible files may be severely restricted. You probably are not able to access files by path name. Ask a responsible person on the remote system to send them to you. For the same reasons you are probably not able to send files to arbitrary path names. As distributed, the remotely accessible files are those whose names begin `/var/spool/uucppublic` (equivalent to `~/`).

All files received by uucp are owned by uucp.

The `-m` option only works when sending files or receiving a single file. Receiving multiple files specified by special shell characters `?`, `&`, and `[ . . . ]` does not activate the `-m` option.

The forwarding of files through other systems may not be compatible with the previous version of uucp. If forwarding is used, all systems in the route must have compatible versions of uucp.

Protected files and files that are in protected directories that are owned by the requester can be sent by uucp. However, if the requester is root, and the directory is not searchable by "other" or the file is not readable by "other", the request fails.

Strings that are passed to remote systems may not be evaluated in the same locale as the one in use by the process that invoked uucp on the local system.

Configuration files must be treated as C (or POSIX) locale text files.

名前	uuencode, uudecode – encode a binary file, or decode its encoded representation
形式	<pre>uuencode [source-file] decode_pathname uuencode [-m] [source-file] decode_pathname uudecode [-p] [encoded-file] uudecode [-o outfile] [encoded-file]</pre>
機能説明	These commands encode and decode files as follows: The uuencode utility converts a binary file into an encoded representation that can be sent using mail(1). It encodes the contents of <i>source-file</i>, or the standard input if no <i>source-file</i> argument is given. The <i>decode_pathname</i> argument is required. The <i>decode_pathname</i> is included in the encoded file's header as the name of the file into which uudecode is to place the binary (decoded) data. uuencode also includes the permission modes of <i>source-file</i> (except <code>setuid</code>, <code>setgid</code>, and sticky-bits), so that <i>decode_pathname</i> is recreated with those same permission modes. The uudecode utility reads an <i>encoded-file</i>, strips off any leading and trailing lines added by mailer programs, and recreates the original binary data with the filename and the mode specified in the header. The encoded file is an ordinary portable character set text file; it can be edited by any text editor. It is best only to change the mode or <i>decode_pathname</i> in the header to avoid corrupting the decoded binary.
オプション	The following options are supported: uuencode <code>-m</code> Encodes <i>source-file</i> using Base64 encoding and sends it to standard output. uudecode <code>-o outfile</code> Specifies a file pathname that should be used instead of any pathname contained in the input data. Specifying an <i>outfile</i> option-argument of <code>/dev/stdout</code> indicates standard output. This allows uudecode to be used in a pipeline. <code>-p</code> Decodes <i>encoded-file</i> and sends it to standard output. This allows uudecode to be used in a pipeline.
オペランド	The following operands are supported by uuencode and uudecode: uuencode <i>decode_pathname</i> The pathname of the file into which the uudecode utility will place the decoded file. If there are characters in <i>decode_pathname</i> that are not in the portable filename character set, the results are unspecified. <i>source-file</i> A pathname of the file to be encoded. uudecode <i>encoded-file</i> The pathname of a file containing the output of uuencode.

使用法	See largefile(5) for the description of the behavior of uuencode and uudecode when encountering files greater than or equal to 2 Gbyte (2 ³¹ bytes).
環境変数	See environ(5) for descriptions of the following environment variables that affect the execution of uuencode and uudecode: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.
Output	stdout
uuencode Base64 Algorithm	The standard output is a text file, encoded in the character set of the current locale, that begins with the line: <pre>begin-base64 %s %s\n, mode, decode_pathname</pre> and ends with the line: <pre>====\n</pre>

In both cases, the lines have no preceding or trailing blank characters.

The encoding process represents 24-bit groups of input bits as output strings of four encoded characters. Proceeding from left to right, a 24-bit input group is formed by concatenating three 8-bit input groups. Each 24-bit input group is then treated as four concatenated 6-bit groups, each of which is translated into a single digit in the Base64 alphabet. When encoding a bit stream by means of the Base64 encoding, the bit stream is presumed to be ordered with the most-significant bit first. That is, the first bit in the stream is the high-order bit in the first byte, and the eighth bit is the low-order bit in the first byte, and so on. Each 6-bit group is used as an index into an array of 64 printable characters, as shown in the following table.

Value	Encoding	Value	Encoding	Value	Encoding	Value	Encoding
0	A	17	R	34	i	51	z
1	B	18	S	35	j	52	0
2	C	19	T	36	k	53	1
3	D	20	U	37	l	54	2
4	E	21	V	38	m	55	3
5	F	22	W	39	n	56	4
6	G	23	X	40	o	57	5
7	H	24	Y	41	p	58	6
8	I	25	Z	42	q	59	7
9	J	26	a	43	r	60	8
10	K	27	b	44	s	61	9
11	L	28	c	45	t	62	+
12	M	29	d	46	u	63	/
13	N	30	e	47	v		
14	O	31	f	48	w	(pad)	=
15	P	32	g	49	x		
16	Q	33	h	50	y		

The character referenced by the index is placed in the output string.

The output stream (encoded bytes) is represented in lines of no more than 76 characters each. All line breaks or other characters not found in the table are ignored by decoding software (see `uudecode`).

Special processing is performed if fewer than 24 bits are available at the end of a message or encapsulated part of a message. A full encoding quantum is always completed at the end of a message. When fewer than 24 input bits are available in an input group, zero bits are added on the right to form an integral number of 6-bit groups. Output character positions that are not required to represent actual input data are set to the equals (=) character. Since all Base64 input is an integral number of octets, only the following cases can arise:

1. The final quantum of encoding input is an integral multiple of 24 bits. Here, the final unit of encoded output is an integral multiple of four characters with no '=' padding.
2. The final quantum of encoding input is exactly 16 bits. Here, the final unit of encoded output is three characters followed by one '=' padding character.
3. The final quantum of encoding input is exactly 8 bits. Here, the final unit of encoded output is two characters followed by two '=' padding characters.

A terminating "====" evaluates to nothing and denotes the end of the encoded data.

uuencode Historical Algorithm

The standard output is a text file (encoded in the character set of the current locale) that begins with the line:

```
begin %s %s\n, mode, decode_pathname
```

and ends with the line:

```
end\n
```

In both cases, the lines have no preceding or trailing blank characters.

The algorithm that is used for lines between `begin` and `end` takes three octets as input and writes four characters of output by splitting the input at six-bit intervals into four octets, containing data in the lower six bits only. These octets are converted to characters by adding a value of `0x20` to each octet, so that each octet is in the range `0x20–0x5f`, and each octet is assumed to represent a printable character. Each octet is then translated into the corresponding character codes for the codeset in use in the current locale. For example, the octet `0x41`, representing 'A', would be translated to 'A' in the current codeset, such as `0xc1` if the codeset were EBCDIC.

Where the bits of two octets are combined, the least significant bits of the first octet are shifted left and combined with the most significant bits of the second octet shifted right. Thus, the three octets A, B, C are converted into the four octets:

```
0x20 + (( A >> 2 ) & 0x3F)
0x20 + (((A << 4) ((B >> 4) & 0xF)) & 0x3F)
0x20 + (((B << 2) ((C >> 6) & 0x3)) & 0x3F)
```


$0x20 + ((C \text{ }) \& 0x3F)$

These octets are then translated into the local character set.

Each encoded line contains a length character, equal to the number of characters to be decoded plus 0x20 translated to the local character set as described above, followed by the encoded characters. The maximum number of octets to be encoded on each line is 45.

終了ステータス The following exit values are returned:

0 Successful completion.

>0 An error occurred.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [mail\(1\)](#), [mailx\(1\)](#), [uucp\(1C\)](#), [uux\(1C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

注意事項 The size of the encoded file is expanded by 35% (3 bytes become 4, plus control information), causing it to take longer to transmit than the equivalent binary.

The user on the remote system who is invoking uudecode (typically uucp) must have write permission on the file specified in the *decode_pathname*.

If you invoke uuencode and then execute uudecode on a file in the same directory, you will overwrite the original file.

名前	uuglist – print the list of service grades that are available on this UNIX system
形式	uuglist [-u]
機能説明	uuglist prints the list of service grades that are available on the system to use with the -g option of uucp(1C) and uux(1C) .
オプション	-u List the names of the service grades that the user is allowed to use with the -g option of the uucp and uux commands.
ファイル	/etc/uucp/Grades contains the list of service grades
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp

関連項目 [uucp\(1C\)](#), [uux\(1C\)](#), [attributes\(5\)](#)

名前	uustat – uucp status inquiry and job control
形式	<pre>uustat [[-m] [-p] [-q] [-k <i>jobid</i> [-n]] [-r <i>jobid</i> [-n]]] uustat [-a] [-s <i>system</i> [-j]] [-u <i>user</i>] [-S <i>qric</i>] uustat -t <i>system</i> [-c] [-d <i>number</i>]</pre>
機能説明	The uustat utility functions in the following three areas: 1. Displays the general status of, or cancels, previously specified uucp commands. 2. Provides remote system performance information, in terms of average transfer rates or average queue times. 3. Provides general remote system-specific and user-specific status of uucp connections to other systems.
オプション	The following options are supported:
General Status	These options obtain general status of, or cancel, previously specified uucp commands: -a Lists all jobs in queue. -j Lists the total number of jobs displayed. The -j option can be used in conjunction with the -a or the -s option. -k<i>jobid</i> Kills the uucp request whose job identification is <i>jobid</i>. The killed uucp request must belong to the user issuing the uustat command unless the user is the super-user or uucp administrator. If the job is killed by the super-user or uucp administrator, electronic mail is sent to the user. -m Reports the status of accessibility of all machines. -n Suppresses all standard output, but not standard error. The -n option is used in conjunction with the -k and -r options. -p Executes the command <code>ps -flp</code> for all the process-ids that are in the lock files. -q Lists the jobs queued for each machine. If a status file exists for the machine, its date, time and status information are reported. In addition, if a number appears in parentheses next to the number of C or X files, it is the age in days of the oldest C./X. file for that system. The Retry field represents the number of hours until the next possible call. The Count is the number of failure attempts. <i>Note:</i> For systems with a moderate number of outstanding jobs, this could take 30 seconds or more of real-time to execute. An example of the output produced by the -q option is: <pre>eagle 3C 04/07-11:07 NO DEVICES AVAILABLE mh3bs3 2C 07/07-10:42 SUCCESSFUL</pre> This indicates the number of command files that are waiting for each system. Each command file may have zero or more files to be sent (zero means to call the system

and see if work is to be done). The date and time refer to the previous interaction with the system followed by the status of the interaction.

-r*jobid* Rejuvenates *jobid*. The files associated with *jobid* are touched so that their modification time is set to the current time. This prevents the cleanup daemon from deleting the job until the jobs' modification time reaches the limit imposed by the daemon.

Remote System Status These options provide remote system performance information, in terms of average transfer rates or average queue times. The -c and -d options can only be used in conjunction with the -t option:

-t*system* Reports the average transfer rate or average queue time for the past 60 minutes for the remote *system*. The following parameters can only be used with this option:

-c Average queue time is calculated when the -c parameter is specified and average transfer rate when -c is not specified. For example, the command:

```
example% uustat -teagle -d50 -c
```

produces output in the following format:

```
average queue time to eagle for last 50 minutes:
      5 seconds
```

The same command without the -c parameter produces output in the following format:

```
average transfer rate with eagle for last 50 minutes:
      2000.88 bytes/sec
```

-d*number* *number* is specified in minutes. Used to override the 60 minute default used for calculations. These calculations are based on information contained in the optional performance log and therefore may not be available. Calculations can only be made from the time that the performance log was last cleaned up.

User- or System-Specific Status These options provide general remote system-specific and user-specific status of uucp connections to other systems. Either or both of the following options can be specified with uustat. The -j option can be used in conjunction with the -s option to list the total number of jobs displayed:

-ssystem Reports the status of all uucp requests for remote system *system*.

-uuser Reports the status of all uucp requests issued by *user*.

Output for both the -s and -u options has the following format:

```
eagleN1bd7 4/07-11:07 S eagle dan 522 /home/dan/A
eagleC1bd8 4/07-11:07 S eagle dan 59 D.3b2a12ce4924
           4/07-11:07 S eagle dan rmail mike
```

With the above two options, the first field is the *jobid* of the job. This is followed by the date/time. The next field is an S if the job is sending a file or an R if the job is requesting a file. The next field is the machine where the file is to be transferred. This is followed by the user-id of the user who queued the job. The next field contains the size of the file, or in the case of a remote execution (*rmail* is the command used for remote mail), the name of the command. When the size appears in this field, the file name is also given. This can either be the name given by the user or an internal name (for example, *D.3b2a1ce4924*) that is created for data files associated with remote executions (*rmail* in this example).

-Sqric Reports the job state:

- q for queued jobs
- r for running jobs
- i for interrupted jobs
- c for completed jobs

A job is queued if the transfer has not started. A job is running when the transfer has begun. A job is interrupted if the transfer began but was terminated before the file was completely transferred. A completed job is a job that successfully transferred. The completed state information is maintained in the accounting log, which is optional and therefore may be unavailable. The parameters can be used in any combination, but at least one parameter must be specified. The **-S** option can also be used with **-s** and **-u** options. The output for this option is exactly like the output for **-s** and **-u** except that the job states are appended as the last output word. Output for a completed job has the following format:

```
eagleC1bd3 completed
```

When no options are given, *uustat* writes to standard output the status of all *uucp* requests issued by the current user.

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of *uustat*: *LANG*, *LC_ALL*, *LC_COLLATE*, *LC_CTYPE*, *LC_MESSAGES*, *LC_TIME*, *NLSPATH*, and *TZ*.

終了ステータス

The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred.

ファイル

<i>/var/spool/uucp/*</i>	spool directories
<i>/var/uucp/.Admin/account</i>	accounting log
<i>/var/uucp/.Admin/perflog</i>	performance log

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [uucp\(1C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

診断 The -t option produces no message when the data needed for the calculations is not being recorded.

注意事項 After the user has issued the uucp request, if the file to be transferred is moved, deleted or was not copied to the spool directory (-C option) when the uucp request was made, uustat reports a file size of -99999. This job will eventually fail because the file(s) to be transferred can not be found.

名前 uuto, uupick – public UNIX-to-UNIX system file copy

形式 uuto [-mp] *source-file*... *destination*

uupick [-s *system*]

機能説明

uuto sends *source-file* to *destination*. **uuto** uses the [uucp\(1C\)](#) facility to send files, while it allows the local system to control the file access. A source-file name is a path name on your machine. Destination has the form:

system[!*system*] ... !*user*

where *system* is taken from a list of system names that **uucp** knows about. *User* is the login name of someone on the specified system.

The files (or sub-trees if directories are specified) are sent to PUBDIR on *system*, where PUBDIR is a public directory defined in the **uucp** source. By default, this directory is `/var/spool/uucppublic`. Specifically the files are sent to

`PUBDIR/receive/user/mysystem/files`.

The recipient is notified by [mail\(1\)](#) of the arrival of files.

uupick accepts or rejects the files transmitted to the user. Specifically, **uupick** searches PUBDIR for files destined for the user. For each entry (file or directory) found, the following message is printed on standard output:

from system *sysname*: [file *file-name*] [dir *dirname*] ?

uupick then reads a line from standard input to determine the disposition of the file:

<new-line> Go to next entry.

d Delete the entry.

m [*dir*] Move the entry to named directory *dir*. If *dir* is not specified as a complete path name (in which \$HOME is legitimate), a destination relative to the current directory is assumed. If no destination is given, the default is the current directory.

a [*dir*] Same as m above, except it moves all the files sent from *system*.

p Print the content of the file.

q Stop.

EOT (control-d) Same as q.

!command Escape to the shell to do command.

* Print a command summary.

オプション

uuto The following options are supported by uuto:

- m Send mail to the sender when the copy is complete.
- p Copy the source file into the spool directory before transmission.

uupick The following option is supported by uupick:

- s *system* Search only the PUBDIR for files sent from *system*.

オペランド The following operands are supported for uuto:

destination A string of the form:

system-name ! user

where *system-name* is taken from a list of system names that uucp knows about; see uuname. The argument *user* is the login name of someone on the specified system. The destination *system-name* can also be a list of names such as

system-name ! system-name ! . . . ! system-name ! user

in which case, an attempt is made to send the file via the specified route to the destination. Care should be taken to ensure that intermediate nodes in the route are willing to forward information.

source-file A pathname of a file on the local system to be copied to *destination*.

環境変数 See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of uuto and uupick: LC_TYPE, LC_MESSAGES, and NLSPATH.

終了ステータス The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred.

ファイル PUBDIR /var/spool/uucppublic public directory

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp

関連項目

[mail\(1\)](#), [uucp\(1C\)](#), [uustat\(1C\)](#), [uux\(1C\)](#), [uucleanup\(1M\)](#), [attributes\(5\)](#)

注意事項

In order to send files that begin with a dot (for instance, `.profile`), the files must be qualified with a dot. For example, the following files are correct:

```
.profile .prof* .profil?
```

The following files are incorrect:

```
*prof* ?profile
```

名前	uux – UNIX-to-UNIX system command execution
形式	<pre>uux [-] [-bcCjnprz] [-a name] [-g grade] [-s filename] [-x debug_level] command-string</pre>
機能説明	The uux utility will gather zero or more files from various systems, execute a command on a specified system and then send standard output to a file on a specified system. <i>Note:</i> For security reasons, most installations limit the list of commands executable on behalf of an incoming request from uux, permitting only the receipt of mail (see mail(1)). (Remote execution permissions are defined in <code>/etc/uucp/Permissions</code>.) The <i>command-string</i> is made up of one or more arguments that look like a shell command line, except that the command and file names may be prefixed by <i>system-name</i>!. A null <i>system-name</i> is interpreted as the local system. File names may be one of the following: ■ An absolute path name. ■ A path name preceded by <code>~xxx</code>, where <code>xxx</code> is a login name on the specified system and is replaced by that user's login directory. Anything else is prefixed by the current directory. As an example, the command: <pre>example% uux "!!diff sys1!/home/dan/filename1 \ sys2!/a4/dan/filename2 > !~/dan/filename.diff"</pre> will get the <code>filename1</code> and <code>filename2</code> files from the <code>sys1</code> and <code>sys2</code> machines, execute a diff(1) command and put the results in <code>filename.diff</code> in the local <code>PUBDIR/dan/</code> directory. <code>PUBDIR</code> is a public directory defined in the uucp source. By default, this directory is <code>/var/spool/uucpublic</code>. Any special shell characters (such as <code><</code>, <code>></code>, <code> </code>) should be quoted either by quoting the entire <i>command-string</i>, or quoting the special characters as individual arguments. The redirection operators <code>>></code>, <code><<</code>, <code>></code>, <code> </code>, and <code>>&</code> cannot be used. uux will attempt to get all appropriate files to the specified system where they will be processed. For files that are output files, the file name must be escaped using parentheses. For example, the command: <pre>example% uux "a!cut -f1 b!/usr/filename > c!/usr/filename"</pre> gets <code>/usr/filename</code> from system <code>b</code> and sends it to system <code>a</code>, performs a <code>cut</code> command on that file and sends the result of the <code>cut</code> command to system <code>c</code>. uux will notify you if the requested command on the remote system was disallowed. This notification can be turned off by the <code>-n</code> option. The response comes by remote mail from the remote machine.

オプション

The following options are supported:

- The standard input to uux is made the standard input to the *command-string*.
- a *name* Uses *name* as the user job identification replacing the initiator user-id. (Notification will be returned to user-id *name*.)
- b Returns whatever standard input was provided to the uux command if the exit status is non-zero.
- c Does not copy local file to the spool directory for transfer to the remote machine (default).
- C Forces the copy of local files to the spool directory for transfer.
- g *grade* *grade* can be either a single letter, number, or a string of alphanumeric characters defining a service grade. The [uuglist\(1C\)](#) command determines whether it is appropriate to use the single letter, number, or a string of alphanumeric characters as a service grade. The output from the *uuglist* command will be a list of service grades that are available or a message that says to use a single letter or number as a grade of service.
- j Outputs the jobid string on the standard output which is the job identification. This job identification can be used by [uustat\(1C\)](#) to obtain the status or terminate a job.
- n Does not notify the user if the command fails.
- p Same as —. The standard input to uux is made the standard input to the *command-string*.
- r Does not start the file transfer, but just queues the job.
- s *filename* Reports status of the transfer in *filename*. This option is accepted for compatibility, but it is ignored because it is insecure.
- x *debug_level* Produces debugging output on the standard output. *debug_level* is a number between 0 and 9. As *debug_level* increases to 9, more detailed debugging information is given.
- z Sends success notification to the user.

環境変数

See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of uux: LANG, LC_ALL, LC_CTYPE, LC_MESSAGES, and NLSPATH.

終了ステータス

The following exit values are returned:

- 0 Successful completion.
- >0 An error occurred.

ファイル	/etc/uucp/*	other data and programs
	/etc/uucp/Permissions	remote execution permissions
	/usr/lib/uucp/*	other programs
	/var/spool/uucp	spool directories

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/uucp
Interface Stability	Committed
Standard	See standards(5) .

関連項目 [cut\(1\)](#), [mail\(1\)](#), [uucp\(1C\)](#), [uuglist\(1C\)](#), [uustat\(1C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 The execution of commands on remote systems takes place in an execution directory known to the uucp system.

All files required for the execution will be put into this directory unless they already reside on that machine. Therefore, the simple file name (without path or machine reference) must be unique within the uux request. The following command will NOT work:

```
example% uux "a!diff b!/home/dan/xyz c!/home/dan/xyz > !xyz.diff"
```

But the command:

```
example% uux "a!diff a!/home/dan/xyz c!/home/dan/xyz > !xyz.diff"
```

will work (if diff is a permitted command.)

Protected files and files that are in protected directories that are owned by the requester can be sent in commands using uux. However, if the requester is root, and the directory is not searchable by "other", the request will fail.

The following restrictions apply to the shell pipeline processed by uux:

- In gathering files from different systems, pathname expansion is not performed by uux. Thus, a request such as

```
uux "c89 remsys!~/*.c"
```

would attempt to copy the file named literally *.c to the local system.
- Only the first command of a shell pipeline may have a *system-name*!. All other commands are executed on the system of the first command.
- The use of the shell metacharacter * will probably not do what you want it to do.

- The shell tokens << and >> are not implemented.
- The redirection operators >>, <<, >|, and >& cannot be used.
- The reserved word ! cannot be used at the head of the pipeline to modify the exit status.
- Alias substitution is not performed.

名前 vacation – reply to mail automatically

形式 vacation [-I]

```
vacation [-a alias] [-e filter_file] [-f database_file]
        [-j] [-m message_file] [-s sender] [-tN] username
vacation [-f database_file] -l
```

機能説明 The vacation utility automatically replies to incoming mail.

Installation The installation consists of an interactive program which sets up vacation's basic configuration.

To install vacation, type it with no arguments on the command line. The program creates a .vacation.msg file, which contains the message that is automatically sent to all senders when vacation is enabled, and starts an editor for you to modify the message. (See USAGE section.) Which editor is invoked is determined by the VISUAL or EDITOR environment variable, or vi(1) if neither of those environment variables are set.

A .forward file is also created if one does not exist in your home directory. Once created, the .forward file will contain a line of the form:

One copy of an incoming message is sent to the *username* and another copy is piped into vacation:

```
\username, "|/usr/bin/vacation username"
```

If a .forward file is present in your home directory, it will ask whether you want to remove it, which disables vacation and ends the installation.

The program automatically creates .vacation.pag and .vacation.dir, which contain a list of senders when vacation is enabled.

Activation and Deactivation The presence of the .forward file determines whether or not vacation is disabled or enabled. To disable vacation, remove the .forward file, or move it to a new name.

Initialization The -I option clears the vacation log files, .vacation.pag and .vacation.dir, erasing the list of senders from a previous vacation session. (See OPTIONS section.)

Additional Configuration vacation provides configuration options that are not part of the installation, these being -a, -e, -f, -j, -m, -s, and -t. (See OPTIONS section.)

Reporting vacation provides a reporting option, -l. See OPTIONS.

オプション The following options are supported:

-I Initializes the .vacation.pag and .vacation.dir files and enables vacation. If the -I flag is not specified, and a *user* argument is given, vacation reads the first line from the standard input (for a From: line, no colon). If absent, it produces an error message.

Options `-a`, `-e`, `-f`, `-j`, `-m`, `-s`, and `-t` are configuration options to be used in conjunction with `vacation` in the `.forward` file, not on the command line. For example,

```
\username, "|/usr/bin/vacation -t1m username"
```

repeats replies to the sender every minute.

- `-a alias` Indicates that *alias* is one of the valid aliases for the user running `vacation`, so that mail addressed to that alias generates a reply.
- `-e filter_file` Uses *filter_file* instead of `.vacation.filter` as the source of the domain and email address filters.
- `-f database_file` Uses *database_file* instead of `.vacation` as the base name for the database file.
- `-j` Does not check whether the recipient appears in the `To:` or the `Cc:` line. Warning: use of this option can result in `vacation` replies being sent to mailing lists and other inappropriate places; its use is therefore strongly discouraged.
- `-m message_file` Uses `~/message_file` as the message to send for the reply instead of `~/vacation.msg`. *message_file* is a relative path to the desired vacation message file. To prevent directory/file “not found” errors, *message_file* should be on the same disk partition as `~/forward`.
- `-s sender` Replies to *sender* instead of the value read from the UNIX `From` line of the incoming message.
- `-tN` Changes the interval between repeat replies to the same sender. The default is 1 week. A trailing `s`, `m`, `h`, `d`, or `w` scales *N* to seconds, minutes, hours, days, or weeks, respectively.

The `-l` option is neither for initialization nor configuration., but for reporting. The `-f` option can also be used in conjunction with the `-l`.

- `-l` Lists the addresses to which a reply has been sent since the last invocation of `vacation` `-I`, along with a date and time stamp.

使用法

`.vacation.msg` should include a header with at least a `Subject:` line (it should not include a `To:` line). For example:

```
Subject: I am on vacation
I am on vacation until July 22. If you have something urgent,
please contact Joe Jones (jones@fb0).
--John
```

If the string `$SUBJECT` appears in the `.vacation.msg` file, it is replaced with the subject of the original message when the reply is sent. Thus, a `.vacation.msg` file such as

Subject: I am on vacation
I am on vacation until July 22.
Your mail regarding "\$SUBJECT" will be read when I return.
If you have something urgent, please contact
Joe Jones (jones@fB0).
--John

will include the subject of the message in the reply.

No message is sent if the To: or the Cc: line does not list the user to whom the original message was sent or one of a number of aliases for them, if the initial From line includes the string –REQUEST@, or if a Precedence: bulk or Precedence: junk line is included in the header.

vacation will also not respond to mail from either postmaster or Mailer-Daemon.

In addition to the above criteria, if a .vacation.filter file exists, it is used to constrain further the set of addresses to which a reply is sent. Each line in that file should be either a domain name, an email address, a negated domain name or a negated email address. A negated line starts with the single character !.

Each line is compared in the order listed to the sender address. A line containing an email address matches if the sender address is exactly the same except for case, which is ignored. A line containing a domain name matches if the sender address is *something@domain-name* or *something@something.domain-name*. A reply is sent if the first match is an entry that is not negated. If the first match is a negated entry, or if no lines match, then no reply is sent.

A sample filter file might look like the following:

```
!host.subdomain.sun.com
sun.com
!wife@mydomain.com
mydomain.com
onefriend@hisisp.com
anotherfriend@herisp.com
```

Blank lines and lines starting with “#” are ignored.

ファイル

```
~/ .forward
~/ .vacation.filter
~/ .vacation.msg
```

A list of senders is kept in the dbm format files .vacation.pag and .vacation.dir in your home directory. These files are dbm files and cannot be viewed directly with text editors.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	service/network/smtp/sendmail

関連項目

[vi\(1\)](#), [sendmail\(1M\)](#), [getusershell\(3C\)](#), [aliases\(4\)](#), [shells\(4\)](#), [attributes\(5\)](#)

名前 vc – version control

形式 vc [-a] [-t] [-c *char*] [-s]
[*keyword=value*... *keyword=value*]

機能説明 This command is obsolete and will be removed in the next release.

The vc command copies lines from the standard input to the standard output under control of its arguments and of “control statements” encountered in the standard input. In the process of performing the copy operation, user-declared *keywords* may be replaced by their string *value* when they appear in plain text and/or control statements.

The copying of lines from the standard input to the standard output is conditional, based on tests (in control statements) of keyword values specified in control statements or as vc command arguments.

A control statement is a single line beginning with a control character, except as modified by the -t keyletter (see below). The default control character is colon (:), except as modified by the -c keyletter (see below). Input lines beginning with a backslash (\) followed by a control character are not control lines and are copied to the standard output with the backslash removed. Lines beginning with a backslash followed by a non-control character are copied in their entirety.

A keyword is composed of 9 or less alphanumerics; the first must be alphabetic. A value is any ASCII string that can be created with ed; a numeric value is an unsigned string of digits. Keyword values may not contain blanks or tabs.

Replacement of keywords by values is done whenever a keyword surrounded by control characters is encountered on a version control statement. The -a keyletter (see below) forces replacement of keywords in all lines of text. An uninterpreted control character may be included in a value by preceding it with \. If a literal \ is desired, then it too must be preceded by \.

オプション The following options are supported:

- a Forces replacement of keywords surrounded by control characters with their assigned value in all text lines and not just in vc statements.
- t All characters from the beginning of a line up to and including the first tab character are ignored for the purpose of detecting a control statement. If a control statement is found, all characters up to and including the tab are discarded.
- c*char* Specifies a control character to be used in place of the “:” default.
- s Silences warning messages (not error) that are normally printed on the diagnostic output.

vc recognizes the following version control statements:

:dc1 *keyword*[, ..., *keyword*] Declare keywords. All keywords must be declared.

:*asg keyword=value*

Assign values to keywords. An *asg* statement overrides the assignment for the corresponding keyword on the *vc* command line and all previous *asg* statements for that keyword. Keywords that are declared but are not assigned values have null values.

:*if condition*

...

:*end*

Skip lines of the standard input. If the condition is true, all lines between the *if* statement and the matching *end* statement are copied to the standard output. If the condition is false, all intervening lines are discarded, including control statements. Note: Intervening *if* statements and matching *end* statements are recognized solely for the purpose of maintaining the proper *if*-*end* matching.

The syntax of a condition is:

<cond> ::= ["not"] <or>

<or> ::= <and> | <and> " | " <or>

<and> ::= <exp> | <exp> "&" <and>

<exp> ::= "(" <or> ")" | <value> <op> <value>

<op> ::= "=" | "!=" | "<" | ">"

<value> ::= <arbitrary ASCII string> | <numeric string>

The available operators and their meanings are:

= equal

!= not equal

& and

| or

> greater than

< less than

() used for logical groupings

not may only occur immediately after the *if*, and when present, inverts the value of the entire condition

The > and < operate only on unsigned integer values (for example, : 012 > 12 is false). All other operators take strings as arguments (for example, : 012 != 12 is true).

The precedence of the operators (from highest to lowest) is:

= != > < all of equal precedence
&
|

Parentheses may be used to alter the order of precedence.

Values must be separated from operators or parentheses by at least one blank or tab.

: :*text*

Replace keywords on lines that are copied to the standard output. The two leading control characters are removed, and keywords surrounded by control characters in text are replaced by their value before the line is copied to the output file. This action is independent of the -a keyletter.

:on

Turn on or off keyword replacement on all lines.

:off

:ctl *char*

Change the control character to *char*.

:msg *message*

Print *message* on the diagnostic output.

:err *message*

Print *message* followed by:

ERROR: err statement on line ... (915)

on the diagnostic output. vc halts execution, and returns an exit code of 1.

属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	developer/build/make

関連項目

[ed\(1\)](#), [attributes\(5\)](#)

名前	vgrind – プログラムリストの整形
形式	<code>vgrind [-2fntwx] [-d <i>defs-file</i>] [-h <i>header</i>] [-l <i>language</i>] [-s <i>n</i>] [-o <i>pagelist</i>] [-P <i>printer</i>] [-T <i>output-device</i>] <i>filename</i>...</code>
機能説明	vgrind ユーティリティーは、troff(1) を使用して、<i>filename</i> 引数で指定されたプログラムソースを整形された形式でフォーマットします。コメントは斜体、キーワードは太字で配置され、各関数が出現すると、その名前がページの余白に表示されます。 vgrind は、2つの基本モード (フィルタモードまたは通常モード) で実行されます。フィルタモードでは、vgrind は tbl(1) と同様のフィルタとして機能します。troff のようなマクロで囲まれた行を除いて、標準入力 は 標準出力 に直接渡されます。 .vS 処理の開始 .vE 処理の終了 これらの行は上記のとおりフォーマットされます。このフィルタからの出力は、出力用に troff に渡すことができます。eqn(1) または tbl(1) では、特定の順序は必要ありません。 通常モードでは、vgrind は入力 <i>filename</i> を受け入れ、それを処理して出力用に troff に渡します。標準入力を指定するには、ハイフン (-) を使用します。それ以外の場合は、vgrind は標準入力からの読み取りを試みることなく終了します。ファイル名は、その他のすべてのオプション引数のあとに指定する必要があります。 通常モードでは、-t または -P オプションが指定されている場合の出力は次のとおりです。 ■ -t オプションが指定されている場合は、標準出力に (troff 形式で) 出力されます。 ■ -P オプションが指定されている場合は、指定されたプリンタに (PostScript として) 出力されます。 それ以外の場合の出力は次のとおりです。 ■ システムのデフォルトプリンタが 1 台定義され、コマンドの標準出力が tty の場合は、そのプリンタに (PostScript として) 出力されます。 ■ 標準出力が tty 以外の場合 (つまり、標準出力がパイプまたはファイルへのリダイレクトである場合) は、標準出力に (PostScript として) 出力されます。 どちらのモードでも、vgrind は小数点で始まる行を変換せずに渡します。

オプション

サポートしているオプションは、次のとおりです。

- 2 2つのカラムの出力を生成します。このオプションを指定すると、デフォルトのポイントサイズが8に変更されます(-s8 オプションが指定された場合と同様)。また、横モードで表示されるように出力が配置されます。
- f 強制的にフィルタモードになります。
- n キーワードを太字にしません。
- w TAB 文字が通常の 8 カラムではなく 4 カラムの間隔を空けるものと見なされます。
- x インデックスファイルを「整形された」形式で出力します。vgrind が現在のディレクトリに存在する index というファイルで実行されるたびに、インデックスファイル自体が生成されます。vgrind に -x オプションおよびファイル index を引数として指定すると、関数定義のインデックスを出力できます。
- d defs-file 代替の言語定義ファイルを指定します(デフォルトは /usr/lib/vgrindefs)。
- h header すべての出力ページの中央にヘッダーが表示されるように指定します。スペースが埋め込まれたヘッダーを指定するには、引用符を使用します。
- l language 使用する言語を指定します。現在認識されている言語は、Bourne シェル(-lsh)、C(-lc、デフォルト)、C++(-lc++)、C シェル(-lcsh)、emacs MLisp(-lml)、FORTRAN(-lff)、Icon(-lI)、ISP(-li)、LDL(-lLDL)、Model(-lm)、Pascal(-lp)、およびRATFOR(-lrr)です。
- P printer 指定されたプリンタに出力を送信します。
- sn 出力で使用するポイントサイズを指定します(troff.ps ポイントサイズ要求の引数とまったく同じ)。

vgrind は、TROFF 環境変数で指定されたフォーマッタに次のオプションを渡します。「環境変数」の節を参照してください。

- t troff の同オプションと同様です。つまり、フォーマット済みのテキストが標準出力に渡されます。
- opagelist pagelist で示したページだけを出力します。このリストには、複数のページ番号またはページ番号の範囲(またはその両方)をコンマで区切って記述します。範囲として *N-M* と記述すればページ番号 *N* から *M* までが出力され、リストの先頭に -*N* と記述

すれば先頭ページからページ番号 N までが出力され、最後に N と記述すればページ番号 N から最終ページまでが出力されます。

-T *output-device* 指定された *output-device* の出力をフォーマットします。

オペランド 次のオペランドを指定できます。

filename vgrind で処理されるプログラムソースの名前。標準入力を指定するには、`-` を使用します。

環境変数 通常モードでは、`vgrind` は中間出力を `TROFF` 環境変数の値で指定されたテキストフォーマットに出力します。この変数が環境に定義されていない場合は、`/usr/bin/troff` に出力します。このメカニズムでは、`troff` の名前で地域のバリエーションが許可されます。

ファイル	index	インデックスのソースが作成されるファイル
	/usr/lib/vgrindefs	言語の記述
	/usr/lib/vfontedpr	プリプロセッサ
	/usr/share/lib/tmac/tmac.vgrind	マクロパッケージ

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	text/doctools

関連項目 [csh\(1\)](#), [ctags\(1\)](#), [eqn\(1\)](#), [tbl\(1\)](#), [troff\(1\)](#), [attributes\(5\)](#), [vgrindefs\(5\)](#)

使用上の留意点 `vgrind` では、特定のプログラミング形式に従うことを前提にしています。

C 関数名は1行でSPACE、TAB、またはアスタリスク(*)のあとにのみ指定できます。括弧で囲まれた引数も同じ行に指定する必要があります。

FORTRAN 関数名は、キーワード `function` または `subroutine` と同じ行に指定する必要があります。

MLisp 関数名は、直前の `defun` と同じ行に指定してはいけません。

Model	関数名は、キーワード <code>is beginproc</code> と同じ行に指定する必要があります。

Pascal	関数名は、キーワード <code>function</code> または <code>procedure</code> と同じ行に指定する必要があります。

これらの規則に従っていない場合は、インデックスおよび限界関数名のコメントメカニズムが失敗します。

さらに一般的には、通常はプログラムの任意のフォーマット形式の体裁が悪くなります。vgrind ではさまざまな幅のフォントが使用されるため、プログラムに vgrind 出力の準備をするには、SPACE 文字ではなく TAB を使用してソースコードを適切にそろえます。

ここでは、関数の認識時に [ctags\(1\)](#) のメカニズムを使用する必要があります。

-w オプションは不便ですが、目的の効果を実現する方法はほかにありません。

tmac.vgrind で定義されたマクロは、その他のマクロパッケージのマクロとの共存が許容されないため、フィルタモードを有効に使用することが困難になります。

vgrind は、[csh\(1\)](#) スクリプトで特定の特殊文字を処理しません。

tmac.vgrind フォーマットマクロによって2カラムモードで使用されるページの高さと幅で固定されるため、標準的な米国のサイズである 8.5 インチ X 11 インチ以外の用紙サイズでは2カラム出力が事実上役に立たなくなります。その他の用紙サイズを使用するには、tmac.vgrind で指定されたサイズ値を編集する必要があります。より適切なソリューションとして、横方向出力専用の troff 出力デバイス仕様を作成して、そこにサイズ情報を記録することが考えられます。

名前 vi, view, vedit – screen-oriented (visual) display editor based on ex

形式

```
/usr/bin/vi [-] [-s] [-l] [-L] [-R] [-r [filename]] [-S]
    [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/bin/view [-] [-s] [-l] [-L] [-R] [-r [filename]] [-S]
    [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/bin/vedit [-] [-s] [-l] [-L] [-R] [-r [filename]] [-S]
    [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg4/bin/vi [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg4/bin/view [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg4/bin/vedit [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg6/bin/vi [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg6/bin/view [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...

/usr/xpg6/bin/vedit [-] [-s] [-l] [-L] [-R] [-r [filename]]
    [-S] [-t tag] [-v] [-V] [-wn]
    [+command | -c command] filename...
```

機能説明

The vi (visual) utility is a display-oriented text editor based on an underlying line editor ex. It is possible to use the command mode of ex from within vi and to use the command mode of vi from within ex. The visual commands are described on this manual page; how to set options (like automatically numbering lines and automatically starting a new output line when you type carriage return) and all ex line editor commands are described on the [ex\(1\)](#) manual page.

When using vi, changes you make to the file are reflected in what you see on your terminal screen. The position of the cursor on the screen indicates the position within the file.

The view invocation is the same as vi except that the readonly flag is set.

The `vedit` invocation is intended for beginners. It is the same as `vi` except that the `report` flag is set to 1, the `showmode` and `novice` flags are set, and `magic` is turned off. These defaults make it easier to learn how to use `vi`.

オプション

The following options are supported:

Invocation Options

The following invocation options are interpreted by `vi` (previously documented options are discussed under NOTES):

- `-s` Suppresses all interactive user feedback. This is useful when processing editor scripts.
- `-l` Sets up for editing LISP programs.
- `-L` Lists the name of all files saved as the result of an editor or system crash.
- `-r filename` Edits *filename* after an editor or system crash. (Recovers the version of *filename* that was in the buffer when the crash occurred.)
- `-R` Readonly mode. The `readonly` flag is set, preventing accidental overwriting of the file.
- `-S` This option is used in conjunction with the `-t tag` option to tell `vi` that the tags file can not be sorted and that, if the binary search (which relies on a sorted tags file) for *tag* fails to find it, the much slower linear search should also be done. Since the linear search is slow, users of large tags files should ensure that the tags files are sorted rather than use this flag. Creation of tags files normally produces sorted tags files. See [ctags\(1\)](#) for more information on tags files.
- `-t tag` Edits the file containing *tag* and position the editor at its definition. It is an error to specify more than one `-t` option.
- `-v` Starts up in display editing state, using `vi`. You can achieve the same effect by typing the `vi` command itself.
- `-V` Verbose. When `ex` commands are read by means of standard input, the input is echoed to standard error. This can be useful when processing `ex` commands within shell scripts.
- `-wn` Sets the default window size to *n*. This is useful when using the editor over a slow speed line.
- `-command` | `-c command` Begins editing by executing the specified editor *command* (usually a search or positioning command).

/usr/xpg4/bin/vi and
/usr/xpg6/bin/vi

If both the `-t tag` and the `-c command` options are given, the `-t tag` option is processed first. That is, the file containing *tag* is selected by `-t` and then the command is executed.

オペランド

The following operands are supported:

filename A file to be edited.

Command Summary vi Modes

The vi command modes are summarized in this section.

Command Normal and initial mode. Other modes return to command mode upon completion. ESC (escape) is used to cancel a partial command.

Input Entered by setting any of the following options:

a A i I o O c C s S R

Arbitrary text can then be entered. Input mode is normally terminated with the ESC character, or, abnormally, with an interrupt.

Last line Reading input for `:` `/` `?` or `!`. Terminate by typing a carriage return. An interrupt cancels termination.

Sample Commands

In the descriptions, CR stands for carriage return and ESC stands for the escape key.

←, →

down-arrow

up-arrow arrow keys move the cursor

h j k l same as arrow keys

i text ESC insert *text*

cw new ESC change word to *new*

ea s ESC pluralize word (end of word; append s; escape from input state)

x delete a character

dw delete a word

dd delete a line

3dd delete 3 lines

u undo previous change

ZZ exit vi, saving changes

:q! CR quit, discarding changes

/text CR search for *text*

^U ^D scroll up or down

:cmd CR any ex or ed command

Counts Before vi Commands Numbers can be typed as a prefix to some commands. They are interpreted in one of these ways:

line/column number `z G |`
scroll amount `^D ^U`
repeat effect most of the rest

Interrupting, Canceling `ESC` end insert or incomplete command
 `DEL` (delete or rubout) interrupts

File Manipulation `ZZ` if file modified, write and exit; otherwise, exit

`:wCR` write back changes
`:w!CR` forced write, if permission originally not valid
`:qCR` quit
`:q!CR` quit, discard changes
`:e nameCR` edit file *name*
`:e!CR` reedit, discard changes
`:e + nameCR` edit, starting at end
`:e +nCR` edit, starting at line *n*
`:e #CR` edit alternate file
`:e! #CR` edit alternate file, discard changes
`:w nameCR` write file *name*
`:w! nameCR` overwrite file *name*
`:shCR` run shell, then return
`:!cmdCR` run *cmd*, then return
`:nCR` edit next file in arglist
`:n argsCR` specify new arglist
`^G` show current file and line
`:ta tagCR` position cursor to *tag*

In general, any ex or ed command (such as *substitute* or *global*) can be typed, preceded by a colon and followed by a carriage return.

Positioning Within a File	F	forward screen
	^B	backward screen
	^D	scroll down half screen
	^U	scroll up half screen
	<i>n</i> G	go to the beginning of the specified line (end default), where <i>n</i> is a line number
	<i>/pat</i>	next line matching <i>pat</i>
	? <i>pat</i>	previous line matching <i>pat</i>
	n	repeat last / or ? command
	N	reverse last / or ? command
	<i>/pat/+n</i>	<i>n</i> th line after <i>pat</i>
	? <i>pat?−n</i>	<i>n</i> th line before <i>pat</i>
	]]	next section/function
	[[	previous section/function
	(	beginning of sentence
	)	end of sentence
	{	beginning of paragraph
	}	end of paragraph
	%	find matching () or { }
Adjusting the Screen	^L	clear and redraw window
	^R	clear and redraw window if ^L is → key
	z-CR	redraw screen with current line at top of window
	z-CR	redraw screen with current line at bottom of window
	z.CR	redraw screen with current line at center of window
	<i>/pat/z-CR</i>	move <i>pat</i> line to bottom of window
	<i>zn.CR</i>	use <i>n</i> -line window
	^E	scroll window down one line
	^Y	scroll window up one line
Marking and Returning	``	move cursor to previous context
	``	move cursor to first non-white space in line

	mx	mark current position with the ASCII lower-case letter <i>x</i>
	`x	move cursor to mark <i>x</i>
	´x	move cursor to first non-white space in line marked by <i>x</i>
Line Positioning	H	top line on screen
	L	last line on screen
	M	middle line on screen
	+	next line, at first non-white space character
	—	previous line, at first non-white space character
	CR	return, same as +
	down-arrow or j	next line, same column
	up-arrow or k	previous line, same column
Character Positioning	^	first non-white space character
	0	beginning of line
	\$	end of line
	l or →	forward
	h or ←	backward
	^H	same as ← (backspace)
	space	same as → (space bar)
	fx	find next <i>x</i>
	Fx	find previous <i>x</i>
	tx	move to character following the next <i>x</i>
	Tx	move to character following the previous <i>x</i>
	;	repeat last f, F, t, or T
	,	repeat inverse of last f, F, t, or T
	<i>n</i>	move to column <i>n</i>
	%	find matching () or { }
Words, Sentences, Paragraphs	w	forward a word
	b	back a word

	e	end of word
	)	to next sentence
	}	to next paragraph
	(	back a sentence
	{	back a paragraph
	W	forward a blank-delimited word
	B	back a blank-delimited word
	E	end of a blank-delimited word
Corrections During Insert	^H	erase last character (backspace)
	^W	erase last word
	erase	your erase character, same as ^H (backspace)
	kill	your kill character, erase this line of input
	\	quotes your erase and kill characters
	ESC	ends insertion, back to command mode
	Control-C	interrupt, suspends insert mode
	^D	backtab one character; reset left margin of <i>autoindent</i>
	^^D	caret (^) followed by control-d (^D); backtab to beginning of line; do not reset left margin of <i>autoindent</i>
	0^D	backtab to beginning of line; reset left margin of <i>autoindent</i>
Insert and Replace	^V	quote non-printable character
	a	append after cursor
	A	append at end of line
	i	insert before cursor
	I	insert before first non-blank
	o	open line below
	O	open line above
	rx	replace single character with <i>x</i>
	RtextESC	replace characters

Operators	Operators are followed by a cursor motion and affect all text that would have been moved over. For example, since <code>w</code> moves over a word, <code>dw</code> deletes the word that would be moved over. Double the operator, for example <code>dd</code>, to affect whole lines. <code>d</code> delete <code>c</code> change <code>y</code> yank lines to buffer <code><</code> left shift <code>></code> right shift <code>!</code> filter through command
Miscellaneous Operations	<code>C</code> change rest of line (<code>c\$</code>) <code>D</code> delete rest of line (<code>d\$</code>) <code>s</code> substitute characters (<code>c\l</code>) <code>S</code> substitute lines (<code>cc</code>) <code>J</code> join lines <code>x</code> delete characters (<code>d\l</code>) <code>X</code> delete characters before cursor (<code>dh</code>) <code>Y</code> yank lines (<code>yy</code>)
Yank and Put	Put inserts the text most recently deleted or yanked; however, if a buffer is named (using the ASCII lower-case letters <code>a - z</code>), the text in that buffer is put instead. <code>3yy</code> yank 3 lines <code>3yl</code> yank 3 characters <code>p</code> put back text after cursor <code>P</code> put back text before cursor <code>"xp</code> put from buffer <code>x</code> <code>"xy</code> yank to buffer <code>x</code> <code>"xd</code> delete into buffer <code>x</code>
Undo, Redo, Retrieve	<code>u</code> undo last change <code>U</code> restore current line <code>.</code> repeat last change <code>"dp</code> retrieve <code>d</code>'th last delete

- 使用法** See [largefile\(5\)](#) for the description of the behavior of vi and view when encountering files greater than or equal to 2 Gbyte (2^{31} bytes).
- 環境変数** See [environ\(5\)](#) for descriptions of the following environment variables that affect the execution of vi: LANG, LC_ALL, LC_COLLATE, LC_CTYPE, LC_TIME, LC_MESSAGES, NLSPATH, PATH, SHELL, and TERM.
- COLUMNS** Override the system-selected horizontal screen size.
- EXINIT** Determine a list of ex commands that are executed on editor start-up, before reading the first file. The list can contain multiple commands by separating them using a vertical-line (|) character.
- LINES** Override the system-selected vertical screen size, used as the number of lines in a screenful and the vertical screen size in visual mode.
- ファイル**
- `/var/tmp` default directory where temporary work files are placed; it can be changed using the `directory` option (see the [ex\(1\)](#) command)
 - `/usr/share/lib/terminfo/??/*` compiled terminal description database
 - `/usr/lib/.COREterm/??/*` subset of compiled terminal description database
- 属性** See [attributes\(5\)](#) for descriptions of the following attributes:

/usr/bin/vi,
/usr/bin/view,
/usr/bin/vedit

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os
CSI	Not enabled

/usr/xpg4/bin/vi,
/usr/xpg4/bin/view,
/usr/xpg4/bin/vedit

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/xopen/xcu4
CSI	Enabled
Interface Stability	Committed
Standard	See standards(5) .

/usr/xpg6/bin/vi,
/usr/xpg6/bin/view,
/usr/xpg6/bin/vedit

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/xopen/xcu6
CSI	Enabled
Interface Stability	Standard

関連項目	Intro(1) , ctags(1) , ed(1) , edit(1) , ex(1) , attributes(5) , environ(5) , largefile(5) , standards(5) 『Solaris Advanced User's Guide』
Author	vi and ex were developed by The University of California, Berkeley California, Computer Science Division, Department of Electrical Engineering and Computer Science.
注意事項	Two options, although they continue to be supported, have been replaced in the documentation by options that follow the Command Syntax Standard (see Intro(1)). An <code>-r</code> option that is not followed with an option-argument has been replaced by <code>-L</code> and <code>+command</code> has been replaced by <code>-c</code> command. The message file too large to recover with <code>-r</code> option, which is seen when a file is loaded, indicates that the file can be edited and saved successfully, but if the editing session is lost, recovery of the file with the <code>-r</code> option is not possible. The editing environment defaults to certain configuration options. When an editing session is initiated, vi attempts to read the EXINIT environment variable. If it exists, the editor uses the values defined in EXINIT; otherwise the values set in <code>\$HOME/.exrc</code> are used. If <code>\$HOME/.exrc</code> does not exist, the default values are used. To use a copy of <code>.exrc</code> located in the current directory other than <code>\$HOME</code>, set the <code>exrc</code> option in EXINIT or <code>\$HOME/.exrc</code>. Options set in EXINIT can be turned off in a local <code>.exrc</code> only if <code>exrc</code> is set in EXINIT or <code>\$HOME/.exrc</code>. In order to be used, <code>.exrc</code> in <code>\$HOME</code> or the current directory must fulfill these conditions: ■ It must exist. ■ It must be owned by the same userid as the real userid of the process, or the process has appropriate privileges. ■ It is not writable by anyone other than the owner. Tampering with entries in <code>/usr/share/lib/terminfo/??/*</code> or <code>/usr/share/lib/terminfo/??/*</code> (for example, changing or removing an entry) can affect programs such as vi that expect the entry to be present and correct. In particular, removing the “dumb” terminal can cause unexpected problems. Software tabs using <code>^T</code> work only immediately after the <i>autoindent</i>. Left and right shifts on intelligent terminals do not make use of insert and delete character operations in the terminal. Loading an alternate <code>malloc()</code> library using the environment variable <code>LD_PRELOAD</code> can cause problems for <code>/usr/bin/vi</code>. The vi utility currently has the following limitations: 1. Lines, including the trailing NEWLINE character, can contain no more than 4096 bytes.

If a longer line is found, Line too long is displayed in the status line.

2. The editor's temporary work file can be no larger than 128Mb.

If a larger temporary file is needed, Tmp file too large is displayed in the status line.

名前 vipw – edit the password file

形式 /usr/ucb/vipw

機能説明 vipw edits the password file while setting the appropriate locks, and does any necessary processing after the password file is unlocked. If the password file is already being edited, then you will be told to try again later. The [vi\(1\)](#) editor will be used unless the environment variable VISUAL or EDITOR indicates an alternate editor.

vipw performs a number of consistency checks on the password entry for root, and will not allow a password file with a “mangled” root entry to be installed. It also checks the /etc/shells file to verify the login shell for root.

ファイル /etc/ptmp
/etc/shells

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [passwd\(1\)](#), [vi\(1\)](#), [passwd\(4\)](#), [attributes\(5\)](#)

名前	volcheck – ドライブ内にメディアがあるかどうかの確認
形式	volcheck [-v] [-i secs] [-t secs] <i>pathname</i>
機能説明	volcheck ユーティリティーは、各 <i>dev/pathname</i> を順に調べ、ドライブに新しいメディアが挿入されていないかを検査するようにボリューム管理に指示します。 デフォルトでは、ボリューム管理によって管理されている、検査が可能なすべてのメディアが volcheck の対象となります。
オプション	サポートしているオプションは、次のとおりです。 -i secs デバイス検査の頻度を <i>secs</i> 秒に設定します。デフォルトは2秒です。最小の頻度は1秒です。 -t secs <i>secs</i> に指定される秒数の間、指定されたデバイスを検査します。指定できる最大の秒数は28800 (8時間) です。検査の頻度は -i で指定します。この合計時間にデフォルト値はありません。 -v 冗長。
オペランド	次のオペランドがサポートされています。 <i>pathname</i> メディアデバイスのパス名
ファイル	/dev/volctl ボリューム管理の制御ポート
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/storage/media-volume-manager

関連項目 [eject\(1\)](#), [rmmount\(1M\)](#), [attributes\(5\)](#)

名前 volrmount - メディアをマウントまたはマウント解除するための rmmount の呼び出し

形式 volrmount [-i | -e] [*name* | *nickname*]
volrmount [-d]

機能説明 volrmount ユーティリティーは **rmmount(1M)** を呼び出して、実際には挿入 (-i) または取り出し (-e) をシミュレートします。多くの場合、挿入をシミュレートすることは rmmount がメディアをマウントすることを意味します。反対に、取り出しをシミュレートすることは rmmount がメディアをマウント解除することを意味します。ただし、これらの操作は rmmount の構成およびメディアタイプによって異なることがあります。

たとえば、デフォルト設定を使用した場合に、音楽 CD を挿入してもマウントされないことがあります。ただし、音楽 CD を挿入するたびに workman が呼び出されるように rmmount を構成できます。

volrmount ユーティリティーを使用すると、ボリューム管理によるメディアの通常の動作を無効にできます。

オプション サポートしているオプションは、次のとおりです。

- i rmmount を呼び出して、指定されたメディアの挿入をシミュレートします。
- e rmmount を呼び出して、指定されたメディアの取り出しをシミュレートします。
- d volrmount で処理されるデフォルトデバイス名を表示します。このデバイスは、*name* または *nickname* が指定されていない場合に使用されます。

オペランド 次のオペランドがサポートされています。

- name* ボリューム管理がデバイス名として認識する名前。
- nickname* デバイス名の短縮形。認識されるニックネームの一覧は次のとおりです。

ニックネーム	パス
cdrom0	/dev/rdisk/cXtYdZ/ <i>label</i>
zip0	/dev/rdisk/cXtYdZ/ <i>label</i>
jaz0	/dev/rdisk/cXtYdZ/ <i>label</i>
rmdisk0	/dev/rdisk/cXtYdZ/ <i>label</i>

ファイル `/dev/volctl` ボリューム管理の制御ポート

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/storage/media-volume-manager

関連項目 [cpio\(1\)](#), [eject\(1\)](#), [tar\(1\)](#), [rmmount\(1M\)](#), [attributes\(5\)](#)

名前	w - 現在ログインしているユーザーに関する情報の表示
形式	w [-hlsuw] [user]
機能説明	w コマンドは、システム内の現在のアクティビティーのサマリーを表示します。これには、各ユーザーが行なっている処理が含まれます。ヘッダー行には、現在の時間、システムが起動されてからの時間、システムにログインしているユーザー数、および(直近の 1、5、および 15 分間に) 実行待ち行列に入っていた平均ジョブ数が表示されます。 表示されるフィールドは、ユーザーのログイン名、ユーザーが使用している tty の名前、ユーザーがログオンした時間(時:分の形式)、アイドル時間(ユーザーが最後に何かを入力してから経過した時間、時:分の形式)、その端末のすべてのプロセスおよびその子プロセスによって使用された CPU 時間(分:秒の形式)、現在アクティブなプロセスによって使用された CPU 時間(分:秒の形式)、および現在のプロセスの名前と引数です。
オプション	サポートしているオプションは、次のとおりです。 -h ヘッダー行を抑制します。 -l 長形式の出力を生成します。これはデフォルトです。 -s 短形式の出力を生成します。短形式では、tty は省略形になり、ログイン時間と CPU 時間は出力されず、コマンドの引数も出力されません。 -u ヘッダー行を生成します。ヘッダー行には、現在の時間、システムが起動されてからの時間、システムにログインしているユーザー数、および(直近の 1、5、および 15 分間に) 実行待ち行列に入っていた平均ジョブ数が表示されます。 -w 長形式の出力を生成します。これもデフォルトと同じです。
オペランド	user ログイン情報を表示する、特定のユーザーの名前。指定すると、出力がそのユーザーに限定されます。
使用例	例1 w コマンドの出力例 <pre>example% w</pre> <pre>10:54am up 27 day(s), 57 mins, 1 user, load average: 0.28, 0.26, 0.22 User tty login@ idle JCPU PCPU what ralph console 7:10am 1 10:05 4:31 w</pre>
環境変数	w の実行に影響を与える次の環境変数についての詳細は、 environ(5) を参照してください。LC_CTYPE、LC_MESSAGES、および LC_TIME。

ファイル /var/adm/utmpx ユーザーとアカウント情報の情報

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os

関連項目 [ps\(1\)](#), [who\(1\)](#), [whodo\(1M\)](#), [utmpx\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#)

注意事項 「現在のプロセス」の概念は明確ではありません。現在のアルゴリズムでは、「割り込みを無視しない端末上のもっとも大きい番号のプロセス、またはそれが存在しない場合は、端末上のもっとも大きい番号のプロセス」です。これには欠陥があり、たとえば、シェルやエディタなどのプログラムの重要なセクションにおいて、またはバックグラウンドで実行される不正なプログラムが生成され、割り込みを無視することに失敗した場合などに誤ったプロセスが選択されます。プロセスを見つけることができない場合、wは-を出力します。

CPU 時間は見積りでしかありません。特に、あるユーザーがバックグラウンドプロセスを実行したままログアウトした場合、現在その端末を使用しているユーザーに時間で「課せられ」ます。

バックグラウンドプロセスは、システムに大きな負荷をかけますが出力には表示されません。

プロセス (特にバックグラウンドプロセスなど) は、NULL または不要になった引数として出力されることがあります。これらの場合、コマンド名が括弧で囲まれて出力されます。

w では、バックグラウンドジョブを検出する規則が認識されません。このため、通常のジョブではなくバックグラウンドジョブを検出することがあります。

名前 wait – 他のプロセスの終了を待つ

形式

/bin/sh wait [*pid*]...

/bin/jsh /bin/ksh88 wait [*pid*]...

/usr/xpg4/bin/sh wait [% *jobid*...]

/bin/csh wait

ksh wait [*job*...]

機能説明

シェルは、新しいプロセスを作成せずに、シェル自身で `wait` を実行します。エラーメッセージ `cannot fork, too many processes` が表示された場合、`wait` コマンドを使用して、バックグラウンドプロセスをすべてクリアしてください。これを行なっても問題が解決しない場合には、システムのプロセステーブルが一杯になっていること、または実行中のフォアグラウンドプロセスの数が多すぎるのが考えられます。ユーザーのログインに対するプロセス ID の数、およびシステムが把握できるプロセスの数には制限があります。

段階以上のパイプラインのプロセスは、必ずしもすべてがシェルの子プロセスであるわけではなく、そのようなプロセスは待つことができません。

/bin/sh, /bin/jsh
プロセス ID が *pid* であるバックグラウンドプロセスを待ち、そのプロセスの終了ステータスを報告します。*pid* を省略すると、ユーザーのシェルで現在実行中のバックグラウンドプロセスをすべて待ち、戻り値は 0 になります。ジョブ制御が有効 (`jsh`) なとき、`wait` はジョブ識別子の指定を受け入れます。引数の *jobid* は、先頭にパーセント記号 (%) を付加して指定します。

pid が実行中のプロセス ID でない場合には、`wait` ユーティリティーはすぐに戻り、戻り値は 0 になります。

csh
バックグラウンドプロセスを待ちます。

ksh88
シェルが非同期リストを開始したとき、非同期リストの各要素内の最後のコマンドのプロセス ID が、現在のシェル実行環境に知らされます。

`wait` ユーティリティーがオペランドなしで呼び出されると、呼び出し側シェルが認識しているすべてのプロセス ID が終了するまで待ち、終了ステータス 0 で終了します。

認識しているプロセス ID (またはジョブ ID) を示すいくつかの *pid* または *jobid* オペランドが指定された場合、`wait` はすべてのプロセス ID が終了するまで待ちます。認識していないプロセス ID (またはジョブ ID) を示すいくつかの *pid* または *jobid* オペランドが指定された場合、`wait` ユーティリティーはプロセス ID (またはジョブ ID) が終了ステータス 127 で終了した場合と同様に扱います。`wait` ユーティリティーが戻した終了ステータスは、最後の *pid* または *jobid* オペランドで要求したプロセスの終了ステータスです。

認識されたプロセス ID は、現在のシェル実行環境内の `wait` 呼び出しに対してだけ適用可能です。

ksh

オペランドなしの `wait` は、呼び出し側シェルが認識しているすべてのジョブが終了するまで待ちます。job オペランドが 1 つ以上指定された場合、`wait` はそれらすべてが終了するまで待ちます。各 job は、次のいずれかの形式で指定します。

`number` `number` はプロセス ID を意味します。

`-number` `number` はプロセスグループ ID を意味します。

`%number` `number` はジョブ番号を意味します。

`%string` 名前が `string` で始まるジョブを意味します。

`%?string` 名前に `string` が含まれているジョブを意味します。

`%+`

`%%` 現在のジョブを意味します。

`%-` 前回のジョブを意味します。

1 つ以上の job オペランドに現在のシェル環境で認識されないプロセス ID またはプロセスグループ ID が指定されている場合、`wait` はそれらを終了ステータス 127 で終了したプロセスと同様に扱います。

オペランド

次のオペランドがサポートされています。

`pid` ユーティリティーが停止を待つ対象であるコマンドの符号なし 10 進整数プロセス ID

`jobid` 待つ対象のバックグラウンドプロセスグループを識別するジョブ制御ジョブ ID。ジョブ制御ジョブ ID 表記は、現在のシェル実行環境内の `wait` 呼び出しに対してだけ適用可能です。なお `pid` のジョブ制御ジョブ ID の形式は、ジョブ制御オプションをサポートしているシステム上でだけ使用可能です。

使用法

ほとんどのシステムでは、`wait` はシェルに組み込まれています。サブシェルや別のユーティリティー実行環境で、次のように `wait` が呼び出されたとします。

```
(wait)
nohup wait ...
find . -exec wait ... \;
```

これらの環境では待つ対象の認識されたプロセス ID がないため、ただちに戻ります。

使用例

例1 終端シグナルを識別するスクリプトを使用する

プロセスがシグナルによって停止されたときに使用する正確な値は不定ですが、シグナルがプロセスを停止したことがわかっていれば、以下のようにすると、どのシグナルが kill を使用しているかをかなり正確に特定できます (/bin/ksh88 および /usr/xpg4/bin/sh)。

```
sleep 1000&
pid=$!
kill -kill $pid
wait $pid
echo $pid was terminated by a SIG$(kill -l $(( $?-128 ))) signal.
```

例2 プロセスの終了ステータスを戻す

以下のコマンドが 31 秒未満で実行されたとします (/bin/ksh88 および /usr/xpg4/bin/sh)。

```
sleep 257 | sleep 31 &
```

```
jobs -l %%
```

その場合、次のコマンドのどちらかが 2 番目の sleep の終了ステータスを戻します。

```
wait <pid of sleep 31>
wait %%
```

環境変数

wait の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス

ksh

ksh に組み込まれた wait により、次の終了値が返されます。

0 wait がオペランドなしで呼び出されました。呼び出し側プロセスが認識しているすべてのプロセスが終了しました。

127 job は、現在のシェル環境が認識できないプロセス ID またはプロセスグループ ID である。

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [csh\(1\)](#), [jobs\(1\)](#), [ksh\(1\)](#), [ksh88\(1\)](#), [pwait\(1\)](#), [sh\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	wc – ファイルの中の行数、単語数および文字数の表示
形式	<code>/usr/bin/wc [-c -m -C] [-lw] [file...]</code>
機能説明	wc は、1 つ以上の入力ファイルを読み取り、デフォルトでは各ファイルごとに標準出力に 1 行を書き込みます。各行には、各ファイルに含まれている復帰改行の数、単語数、およびバイト数がこの順番で書き込まれ、その後ろにファイル名が続きます。単語は、isspace(3C) 文字で区切られた、長さがゼロではない文字列として定義されます。 複数のファイルが指定された場合、wc は、ファイル名の代わりに total と表示して、指定されたすべてのファイルの合計を書き込みます。 デフォルトでは、wc は 3 種類のカウンントをすべて書き込みます。オプションを指定すると、特定のカウンントのみを書き込むようにできます。-c オプションと -m オプションは相互に排他的です。 file が指定されていないか、file に - が指定された場合、wc は標準入力から読み取り、標準出力にファイル名を書き込みません。現在のオフセットがファイルの開始位置として定義されます。
オプション	次のオプションは、<code>usr/bin/wc</code> と <code>ksh</code> の両方でサポートされます。長形式のオプションは、<code>ksh</code> でのみ使用できます。 -c バイト数を数えます。 -C 文字数を数えます。-m と同じです。 -l 行数を数えます。 -m 文字数を数えます。-c と同じです。 -w 空白文字または復帰改行で区切られた単語数を数えます。区切り文字は、isspace(3C) によって定義された任意のコードセット中の拡張 Unix コード (EUC) 文字です。 オプションが指定されていない場合、デフォルトは -lwc です (行数、単語数、バイト数の表示)。
オペランド	次のオペランドを指定できます。 <i>file</i> 入力ファイルのパス名。<i>file</i> オペランドを指定しないと、標準入力を使用されます。
使用法	ファイルが 2G バイト (2³¹ バイト) 以上ある場合の wc の動作については、largefile(5) を参照してください。

- 環境変数** `wc` の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。
- 終了ステータス** `0` 正常終了。
 `>0` エラーが発生した。
- 属性** 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

- 関連項目** [cksum\(1\)](#), [isspace\(3C\)](#), [iswalph\(3C\)](#), [iswspace\(3C\)](#), [setlocale\(3C\)](#), [attributes\(5\)](#), [environ\(5\)](#), [largefile\(5\)](#), [standards\(5\)](#)

名前	what – ファイルからの SCCS バージョン情報の抽出
形式	what [-s] filename...
機能説明	what ユーティリティーは、SCCS get コマンド (sccs-get(1) を参照) が @(#) ID キーワードに置き換えるパターン @(#) の出現を各 <i>filename</i> で検索し、"、>、復帰改行文字、\、または NULL 文字に続く部分を出力します。
オプション	次のオプションがサポートされています。 -s パターンが最初に出現したあとに停止します。
使用例	例1 SCCS バージョン情報の抽出 ファイル <code>program.c</code> の C プログラムに次の行が含まれていて、 <pre>char sccsid[] = " @(#)identification information ";</pre> <code>program.c</code> が <code>program.o</code> と <code>a.out</code> を生成するようにコンパイルされている場合に、次のコマンドを実行します。 <pre>example% what program.c program.o a.out</pre> これを実行すると次の出力が得られます。 <pre>program.c: identification information program.o: identification information a.out: identification information</pre>
終了ステータス	次の終了ステータスが返されます。 0 一致するものが見つかった。 1 一致するものが1つも見つからなかった。
環境変数	what の実行に影響を与える次の環境変数については、 environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	developer/build/make
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目	sccs(1) , sccs-admin(1) , sccs-cdc(1) , sccs-comb(1) , sccs-delta(1) , sccs-get(1) , sccs-help(1) , sccs-prs(1) , sccs-prt(1) , sccs-rmdel(1) , sccs-sact(1) , sccs-sccsdiff(1) , sccs-unget(1) , sccs-val(1) , sccsfile(4) , attributes(5) , environ(5) , standards(5)
診断	SCCS コマンドの説明を表示するには、 <code>sccs-help</code> コマンドを使用します。 sccs-help(1) を参照してください。
使用上の留意点	<code>what</code> が <code>@(#)</code> パターンの誤った出現を見つける可能性がわずかにあります。

名前	whatis - キーワードに関するサマリーの表示								
形式	whatis <i>command</i> ...								
機能説明	whatis は <i>command</i> 引数で指定されたマニュアルページを参照し、そのヘッダー行を表示します。ヘッダーを見て、そのコマンドについての詳細を知りたいければ、man(1) を実行することによりマニュアルページ全体を得ることができます。行が <i>name(section)</i> で始まる場合です。..<i>man -ssection name</i> を実行して該当するドキュメントを得ることができます。試しに <i>whatis ed</i> を実行し、その後 <i>man -s 1 ed</i> を実行してみてください。ed(1) のマニュアルページが得られます。 whatis は、機能的には -f オプション付きの man(1) コマンドと同じです。 whatis は、<i>/usr/share/man/man_index/*</i> インデックスファイルを使用します。インデックスファイルは、man(1) と man(5) で説明されているように SMF サービスによって自動的に生成されるか、または catman(1M) で -w オプションを指定して手動で生成します。インデックスファイルがない場合、whatis はマニュアルページファイルに対して直接検索を実行するため、実行速度が遅くなります。								
ファイル	<i>/usr/share/man/man_index/*</i> 目次とキーワードのデータベース。 次のファイルが生成されます。 ■ <i>/usr/share/man/man_index/man.idx</i> ■ <i>/usr/share/man/man_index/man.dic</i> ■ <i>/usr/share/man/man_index/man.frq</i> ■ <i>/usr/share/man/man_index/man.pos</i>								
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。								
	<table><tr><th>属性タイプ</th><th>属性値</th></tr><tr><td>使用条件</td><td>text/doctools</td></tr><tr><td>CSI</td><td>有効</td></tr><tr><td>インタフェースの安定性</td><td>確実</td></tr></table>	属性タイプ	属性値	使用条件	text/doctools	CSI	有効	インタフェースの安定性	確実
属性タイプ	属性値								
使用条件	text/doctools								
CSI	有効								
インタフェースの安定性	確実								
関連項目	apropos(1) , man(1) , catman(1M) , attributes(5) , man(5)								

名前	whereis – locate the binary, source, and manual page files for a command
形式	<code>/usr/ucb/whereis [-bmsu] [-BMS <i>directory...</i> -f] <i>filename...</i></code>
機能説明	The whereis utility locates source/binary and manuals sections for specified files. The supplied names are first stripped of leading pathname components and any (single) trailing extension of the form <i>.ext</i>, for example, <i>.c</i>. Prefixes of <i>s</i> resulting from use of source code control are also dealt with. whereis then attempts to locate the desired program in a list of standard places: <pre> etc /sbin /usr/bin /usr/lang /usr/lbin /usr/lib /usr/sbin /usr/ucb /usr/ucblib /usr/ucbinclude /usr/games /usr/local /usr/local/bin /usr/new /usr/old /usr/hosts /usr/include </pre>
オプション	The following options are supported: -b Searches only for binaries. -B Changes or otherwise limits the places where whereis searches for binaries. -f Terminates the last directory list and signals the start of file names, and <i>must</i> be used when any of the -B, -M, or -S options are used. -m Searches only for manual sections. -M Changes or otherwise limits the places where whereis searches for manual sections. -s Searches only for sources. -S Changes or otherwise limit the places where whereis searches for sources. -u Searches for unusual entries. A file is said to be unusual if it does not have one entry of each requested type. Thus 'whereis -m -u *' asks for those files in the current directory which have no documentation.

使用例 例1 Finding files

Find all files in /usr/bin which are not documented in /usr/share/man/man1 with source in /usr/src/cmd:

```
example% cd /usr/ucb
example% whereis -u -M /usr/share/man/man1 -S /usr/src/cmd -f *
```

ファイル /usr/src/*

 /usr/{doc,man}/*

 /etc, /usr/{lib,bin,ucb,old,new,local}

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [chdir\(2\)](#), [attributes\(5\)](#)

使用上の留意点 Since whereis uses [chdir\(2\)](#) to run faster, pathnames given with the -M, -S, or -B must be full; that is, they must begin with a /.

名前	which – locate a command and display its pathname or alias				
形式	which [<i>name</i>]...				
機能説明	which takes a list of names and determines which alias or utility would be executed had these names been given as commands. For each <i>name</i> operand, if it names an alias the alias is expanded. Otherwise the user's path is searched for a utility name matching <i>name</i>. Aliases are taken from the user's .cshrc file. <i>path</i> is taken from the current shell execution environment.				
オペランド	The following operand is supported: <i>name</i> The name of a command to be located.				
終了ステータス	The following exit values are returned: 0 Successful completion. >0 One or more <i>name</i> operands were not located or an error occurred.				
ファイル	~/ .cshrc source of aliases and path values				
属性	See attributes(5) for descriptions of the following attributes:				
	<table border="1"> <thead> <tr> <th>ATTRIBUTE TYPE</th><th>ATTRIBUTE VALUE</th></tr> </thead> <tbody> <tr> <td>Availability</td><td>system/core-os</td></tr> </tbody> </table>	ATTRIBUTE TYPE	ATTRIBUTE VALUE	Availability	system/core-os
ATTRIBUTE TYPE	ATTRIBUTE VALUE				
Availability	system/core-os				
関連項目	csh(1) , attributes(5)				
診断	A diagnostic is given for names which are aliased to more than a single word, or if an executable file with the argument name was not found in the path.				
注意事項	The which utility is not a shell built-in command.				
使用上の留意点	To compensate for ~/ .cshrc files in which aliases depend upon the prompt variable being set, which sets this variable to NULL. If the ~/ .cshrc produces output or prompts for input when prompt is set, which can produce some strange results.				

名前 who - システムにログインしているユーザーの表示

形式

```
/usr/bin/who [-abdHlmpqrstTu] [file]
/usr/bin/who -q [-n x] [file]
/usr/bin/who am i
/usr/bin/who am I
/usr/xpg4/bin/who [-abdHlmpqrstTu] [file]
/usr/xpg4/bin/who -q [-n x] [file]
/usr/xpg4/bin/who -s [-bdHlmpqrtu] [file]
/usr/xpg4/bin/who am i
/usr/xpg4/bin/who am I
```

機能説明

who ユーティリティーは、現在の UNIX システムを使用しているそれぞれのユーザーについて、ユーザー名、端末回線名、ログイン時間、回線を稼働し始めてからの経過時間、コマンドインタプリタ (シェル) のプロセス ID を表示することができます。who は、`/var/adm/utmpx` ファイルを検索して、その情報を入手します。`file` を指定した場合、そのファイルを検索します (ファイルは、`utmpx(4)` フォーマットでなければなりません)。通常、`file` は `/var/adm/wtmpx` で、これはファイルが最後に作成されてからのすべてのログイン記録を格納しています。

出力のための一般形式は次のようになります。

```
name [state] line time [idle] [pid] [comment] [exit]
```

各表記の意味は次のとおりです。

<i>name</i>	ユーザーのログイン名
<i>state</i>	端末への書き込み権
<i>line</i>	<code>/dev</code> に存在する回線名
<i>time</i>	ユーザーがログインしてからの時間
<i>idle</i>	ユーザーが最後に実行してからの経過時間
<i>pid</i>	ユーザーのプロセス ID
<i>comment</i>	<code>inittab(4)</code> の注釈欄
<i>exit</i>	活動していないプロセスの終了ステータス

オプション サポートしているオプションは、次のとおりです。

-a `/var/adm/utmpx` または指定された `file` を、-b、-d、-l、-p、-r、-t、-T、-u オプションをつけた場合と同様に処理します。

-b 直前のリポートの時間と日付を示します。

- d 時間切れになっていて、initによって再生成されていないすべてのプロセスを表示します。exitフィールドは、活動していないプロセスについて出力され、最終プロセスの最終値および終了値 (wait(3C) が戻す値) を表示します。これは、プロセス終了の原因を究明するのに役立ちます。
- H 標準出力の上のカラムヘッダーを出力します。
- l 誰かがログインするのをシステムが待っている 回線のみを表示します。そのような場合、name フィールドは、LOGIN です。state フィールドが存在しないということを除けば、その他のフィールドは、ユーザーエン트리と同じです。
- m 使用している端末についての情報だけを出力します。
- n x ユーザーの数を行ごとにディスプレイへ指定する数値引数、xが付きます。xは、1以上でなければなりません。-n オプションは、-q と共に使用してください。
- p 現在アクティブで、initによって以前に生成された他のプロセスを表示します。name フィールドは、/usr/sbin/inittab 内の initによって実行されたプログラム名です。state、line、および idle フィールドは、意味がありません。comment フィールドは、このプロセスを生成した /usr/sbin/inittab からの回線の id フィールドを示します。inittab(4) を参照してください。
- q (who の簡略版) 現在ログオンしているユーザーの名前とユーザー番号しか表示しません。このオプションを使用するときには、その他のオプションはすべて無視されます。
- r このオプションは、init プロセスの現在の実行レベル (run-level) を示します。
- s (デフォルト) name、line、および time のフィールドのみを表示します。
- /usr/bin/who -T -s オプションと同じです。ただし T では、state、idle、pid、および comment フィールドが表示されます。state は次に示す文字のどれか 1 つです。
- + 端末に他のユーザーの書き込みが可能である
 - 端末に他のユーザーの書き込みが可能でない
 - ? 端末の書き込みアクセス状態が判定できない
- /usr/xpg4/bin/who -T -s オプションと同じです。ただし T では、state フィールドが表示されます。state はこのオプションの /usr/bin/who バージョンに表示されている文字のどれか 1 つです。-u オプションが -T と一緒に使用されている場合、アイドル時間は前のフォーマットの最後に追加されます。
- t root によるシステムクロックの最後の変更 (date ユーティリティーを使用) を示します。su(1M) および date(1) を参照してください。

-u 現在ログイン中のユーザーのみを表示します。name は、ユーザーのログイン名です。line は回線名です。/dev ディレクトリにあります。time は、ユーザーがログインした時間です。idle カラムは、ある特定の回線を最後に稼働し始めてからの経過時間を意味します。ドット(.)は、端末がごく最近起動したことを認識しており、それが現在の入力であることを示します。24 時間以上が経過したり、ブート時間から回線が使用されていない場合は、エントリに old とマークされます。人が端末で操作しているかどうかを判別しようとするとき、このフィールドが役に立ちます。pid は、ユーザーのシェルのプロセス ID です。comment は、この回線に関連する注釈欄です。/usr/sbin/inittab (inittab(4) を参照) にあります。注釈欄には、どこに端末があるか、データセットの電話番号、直結の場合は端末の型などについての情報を盛り込むことができます。

オペランド 次のオペランドがサポートされています。

am i
am I C ロケールでは、起動したユーザーの出力に限られます。これは -m オプションと同じです。am と、i または I 引数は空白で区切らなければなりません。

file who がデフォルトで使用する、ログインしたユーザーのデータベースの代わりに使用するファイルのパス名を指定します。

環境変数 who の実行に影響を与える次の環境変数についての詳細は、environ(5) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、LC_TIME、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

>0 エラーが発生した。

ファイル /usr/sbin/inittab init のスクリプト

/var/adm/utmpx 現在のユーザーとアカウントिंगの情報

/var/adm/wtmp 記録されているユーザーとアカウントिंगの情報

属性 属性についての詳細は、マニュアルページの attributes(5) を参照してください。

/usr/bin/who	属性タイプ	属性値
	使用条件	system/core-os

/usr/xpg4/bin/who	属性タイプ	属性値
	使用条件	system/xopen/xcu4

属性タイプ	属性値
インタフェースの安定性	确实
標準	standards(5) を参照してください。

関連項目 [date\(1\)](#), [login\(1\)](#), [mesg\(1\)](#), [init\(1M\)](#), [su\(1M\)](#), [wait\(3C\)](#), [inittab\(4\)](#), [utmpx\(4\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

注意事項 スーパーユーザーへの注意: シャットダウン後のシングルユーザー状態において、`who` は、プロンプトを返します。理由は、`/var/adm/utmpx` がログイン時に更新され、シングルユーザー状態のログインがないと、`who` がこの状態を正確に報告できないからです。しかし、`who am i` は、正しい情報を返します。

名前	whoami – display the effective current username
形式	/usr/ucb/whoami
機能説明	whoami displays the login name corresponding to the current effective user ID. If you have used su to temporarily adopt another user, whoami will report the login name associated with that user ID. whoami gets its information from the geteuid and getpwuid library routines (see getuid and getpwnam(3C) , respectively).
ファイル	/etc/passwd username data base
属性	See attributes(5) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	compatibility/ucb

関連項目 [su\(1M\)](#), [who\(1\)](#), [getuid\(2\)](#), [getpwnam\(3C\)](#), [attributes\(5\)](#)

名前	whocalls – 特定のプロシージャの呼び出しに関する報告
形式	whocalls [-l <i>wholib</i>] [-s] <i>funcname executable</i> [<i>arguments</i>]...
機能説明	whocalls は、特定の関数呼び出しの追跡を許可する ld.so.1(1) の <i>Link-Auditing</i> 機能に基づいたユーティリティーの簡単な例です。 <i>Link-Auditing</i> メカニズムについての詳細は、『 リンカーとライブラリ 』を参照してください。 <i>executable</i> は、関連する引数を使用して通常どおりに実行されます。プロシージャ <i>funcname</i> が呼び出されるたびに、そのプロシージャへの引数とスタックトレースの両方が標準出力に表示されます。
オプション	サポートしているオプションは、次のとおりです。 -l <i>wholib</i> 使用する代替の <i>who.so Link-Auditing</i> ライブラリを指定します。 -s 使用可能な場合、ローカルシンボル用の <i>.symtab</i> シンボルテーブルを検証して使用します。これは、 <i>.dynsym</i> シンボルテーブルを使用するよりも少し負荷がかかりますが、より詳細なスタックトレース情報を生成できます。
使用例	例1 関数呼び出しの追跡 次の例では、単純な <i>helloworld</i> プログラムで作成された <i>printf()</i> の呼び出しを追跡します。 <pre>example% whocalls printf helloworld printf(0x106e4, 0xef625310, 0xef621ba8) helloworld:main+0x10 helloworld:_start+0x5c Hello World</pre>
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities

関連項目 [ld.so.1\(1\)](#), [sotruss\(1\)](#), [attributes\(5\)](#)
『[リンカーとライブラリ](#)』

名前	whois - インターネットのユーザー名ディレクトリサービス
形式	whois [-h host] identifier
機能説明	whois は、名前 (Smith など) とハンドル (SRI-NIC など) のどちらかの識別子 <i>identifier</i> について、インターネット・ディレクトリ・エントリを検索します。名前だけの検索をする場合には、名前の前にピリオドを付けてください。ハンドルだけの検索をする場合には、ハンドルの前に感嘆符を付けてください。 グループまたは組織のエントリを検索する場合には、引数の前にアスタリスク (*) を付けてください。これにより、グループの全メンバーのリストが、レコードとともに表示されます。 感嘆符とアスタリスク、または、ピリオドとアスタリスクを同時に使用することもできます。
使用例	例1 whois コマンドの使用 次のコマンド <pre>example% whois Smith</pre> SMITH という名前またはハンドルを検索します。 次のコマンド <pre>example% whois !SRI-NIC</pre> SRI-NIC というハンドルだけを検索します。 次のコマンド <pre>example% whois .Smith, John</pre> JOHN SMITH という名前だけを検索します。 名前またはハンドルの引数に . . . を付けると、その名前またはハンドルの引数で始まるものがすべて一致します。つまり、zu . . . は、ZUL、ZUM などと一致します。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	service/network/network-clients

関連項目 [attributes\(5\)](#)

名前	write – 他のユーザーへのメッセージの書き込み
形式	<code>write user [terminal]</code>
機能説明	<code>write</code> ユーティリティーは、ユーザーの標準入力から行を読み込み、他のユーザーの端末へ書き出します。最初に呼び出されたときに、 <pre>Message from sender-login-id (sending-terminal) [date]...</pre> というメッセージを <code>user</code> で指定されたユーザーの端末へ書き出します。接続が正常に行われると、送り側ユーザーの端末にベルを2回送って、入力中の内容が宛先端末に送信されつつあることを示します。 受け取り側は、返答を送りたい場合、最初のメッセージの受信時に <pre>write sender-login-id [sending-terminal]</pre> と入力します。標準入力モードでは、NL、EOF、または EOL 特殊文字で区切られた 1 行分の入力データが累積されるたびに、その 1 行分が宛先ユーザーに送信されます。文字は次のように処理されます。 ■ 警告(アラート)文字を入力すると、警告文字が宛先に送られます。 ■ 行消去文字または文字消去文字を入力すると、termios(3C) インタフェースの規定に従って、送り側の端末が影響を受けます。 ■ 割り込みまたは EOF 文字を入力すると、<code>write</code> は対応するメッセージ (C ロケールでは EOT\004) を宛先端末に送って処理を終了します。 ■ LC_CTYPE における <code>print</code> または <code>space</code> に属する文字を入力すると、それらの文字が宛先端末に送られます。 ■ <code>stty ixexten</code> ローカルモードが有効な場合に限り、特殊制御文字および複数バイト文字やシングルバイト文字は、対応するワイド文字が印刷可能であれば、印刷可能として処理されます。 ■ その他の印刷不可能な文字を入力した場合は、次に述べるような方法で宛先端末に送られます。制御文字は '^' 文字のあとに適切な ASCII 文字が表示されます。最上位ビットに 1 がたっている文字は「メタ」表記法で表示されます。たとえば、'\003' は '^c' と表示され、'\372' は '^M-z' と表示されます。 複数ログインしているユーザーへメッセージを送りたいときには、接続する端末を指定するために <code>terminal</code> 引数を使用できます。それ以外の場合には、<code>/usr/adm/utmpx</code> 中で見つかった、対象ユーザーの最初の出力可能な端末が宛先端末となります。このとき、どの端末が選ばれたかを示すために以下の通知メッセージが送り側の標準出力に書き込まれます。 <pre>user is logged on more than one place. You are connected to terminal. Other locations are:terminal</pre>

write メッセージの受け取り側になると、mesg ユーティリティーを使って拒否したり承諾したりすることが可能となります。ただしユーザーがどのような特権を持っているかにより、他のどのユーザーの端末にアクセスできるかが制限されます。つまり要求した動作を行う上で必要な特権を持っていないと、write ユーティリティーの実行は成功しません。

! 文字が行の先頭にあると、write は、シェルを呼び出して行の残りの部分をコマンドとして実行します。

write は、ほかのユーザー端末への書き込み権を得るために、グループ ID tty への setgid() ([setuid\(2\)](#)を参照) を実行します。

write を使用するために次の手順をお勧めします。ユーザーが相手のユーザーに最初の write を実行したら、メッセージを送り始める前に write を返してくるのを待ちます。相手がいつ返事をすればよいかがわかるように、それぞれのユーザーは終了を示すシグナル (o) は over (応答どうぞ) を表す) を入れてメッセージを終了します。会話を終了するときは、シグナル (oo) (over and out (通信終わり) を表す) をお勧めします。

オペランド 次のオペランドがサポートされています。

- user* メッセージの送信先となる人のユーザー (ログイン) 名。指定形式は、[who\(1\)](#) ユーティリティーが返す形式と同じでなければなりません。
- terminal* 端末の ID で、who ユーティリティーが返す形式と同一形式。

環境変数 write の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、および NLSPATH。

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- >0 指定されたユーザーはログインしていなかった、または指定されたユーザーが書き込みを拒否しました。

- ファイル /var/adm/utmpx write 用のユーザーおよびアカウント情報
- /usr/bin/sh Bourne シェルの実行可能ファイル

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実

属性タイプ	属性値
標準	standards(5) を参照してください。

関連項目	mail(1) , mesg(1) , pr(1) , sh(1) , talk(1) , who(1) , setuid(2) , termios(3C) , attributes(5) , environ(5) , standards(5)	
診断	user is not logged on	ユーザーがwriteを試みている相手はログインしていません。
	Permission denied	ユーザーがwriteを試みている相手は、書き込みをmesgを使って拒否しています。
	Warning: cannot respond, set mesg -y	ユーザーの端末はmesg nに設定され、相手ユーザーが応答できません。
	Can no longer write to user	ユーザーが書き込みを始めた後で、相手ユーザーが、書き込みを拒否 (mesg n) しました。

名前 xargs - 引数リストを構築してコマンドを実行

形式 xargs [-t] [-p] [-e[*eofstr*]] [-E *eofstr*]
 [-I *replstr*] [-i[*replstr*]] [-L *number*] [-l[*number*]]
 [-n *number* [-x]] [-s *size*] [*utility* [*argument*...]]

機能説明 xargs ユーティリティーは、オペランドで指定された *utility* と *argument* からなり、その後いくつかの引数を伴ったコマンド行を生成します。付加する引数は、標準入力から読み込んだものがその順序どおりに与えられます。ただしその長さや数は、オプションで指定した値を超えることはありません。コマンド行を生成したら、xargs ユーティリティーはそのコマンドを実行して、完了を待ちます。この一連の処理は、標準入力上でファイルの終わりが検出されるまで、あるいは実行したコマンドが終了ステータス 255 で終了するまで繰り返されます。

標準入力から読み込まれる引数は、引用符で囲まれていない空白文字、エスケープされていない空白文字、もしくは復帰改行文字で区切られていなければなりません。0 個以上の、二重引用符 (") および復帰改行以外の文字からなる文字列は、二重引用符で囲むことができます。0 個以上の、アポストロフィ (') および復帰改行以外の文字からなる文字列は、アポストロフィで囲むことができます。また引用符でかこまれていない文字は、いずれもバックスラッシュ (\) を先頭に付加することによりエスケープできます。*utility* で指定したユーティリティーは、ファイルの終わりが検出されるまで、繰り返して実行されます。そのユーティリティーが標準入力からデータを読み込もうとした場合、処理の結果は不確定です。

生成したコマンド行の長さは、ユーティリティー名のバイト長と文字列として認識された各引数のバイト長の合計となります。文字列の終端用の NULL 文字も含まれます。xargs ユーティリティーは、生成したコマンドを呼び出すときに、引数と環境リストをまとめた長さが {ARG_MAX}-2048 バイトを超えないよう、コマンド行の長さを制限します。この制限範囲の中で、-n と -s の両オプションがともに省略された場合、デフォルトのコマンド行の長さの最小値は {LINE_MAX} となります。

オプション サポートしているオプションは、次のとおりです。

- 0 入力項目は、スペースまたは復帰改行ではなく NULL 文字で終了します。引用符とバックスラッシュは特殊文字ではなく、すべての文字は文字どおりに扱われます。ファイルの終わりを表す文字列も無効にされ、その他の引数と同じように扱われます。これは、入力項目にスペース、引用符、またはバックスラッシュが含まれている可能性がある場合に便利です。find の -print0 オプションでは、このモードに適した入力生成されます。
- e[*eofstr*] 論理的なファイルの終わり (EOF) を表す文字列を *eofstr* で指定します。この -e オプションを省略し、さらに -E も省略すると、デフォルトとして下線 (_) が論理的 EOF 文字列となります。また *e* だけを記述して *eofstr* を省略すると、論理的 EOF 文字列機能は無効となり、下線

はそのまま引数の値の一部とみなされます。xargs の標準入力からの読み込みは、ファイルの終わりに達するか、eofstr が示す文字列が現れた時点で終了します。

- E eofstr 論理的 EOF 文字列として、デフォルトの下線の代わりに eofstr で示す文字列を使用します。xargs の標準入力からの読み込みは、ファイルの終わりに達するか、eofstr が示す文字列が現れた時点で終了します。eofstr がヌル文字列の場合、論理的なファイルの終わり (EOF) を表す文字列の機能が無効になり、下線文字はそのまま引数の値の一部と見なされます。
- I replstr 挿入モードを表します。標準入力から読み込んだ各行に対して utility で指定したユーティリティが実行されます。このとき、読み込んだ行全体を 1 つの引数と見なし、replstr が示す文字列が argument s 中に含まれていればその位置に読み込んだ行の内容を挿入します。argument で指定する引数のうち、最大 5 つの引数に replstr を 1 つまたは複数個含めることができます。標準入力からの行の先頭が空白文字の場合、それらは無視されます。挿入によって生成される引数の合計長は、255 バイトを超えることはできません。この -I オプションを指定した場合は、x オプションが強制的に有効となります。-I と -i は相互に排他的なオプションで、両方が指定された場合には最後に現れた方が有効です。
- i [replstr] このオプションは -I replstr と同等の意味を持ちます。引数の replstr を省略すると、デフォルトの文字列として { } が用いられます。
- L number number で示す数の引数行 (空の行を除く) を標準入力から読み込むたびに、utility を実行します。最後の呼び出し時、標準入力に残っている行数が number より少なくとも、その引数で utility を実行します。各行は、その行で最初に現れた復帰改行をもって終わりと見なされます。ただし行の最後の文字が空白文字の場合、次の空ではない行に継続することになります。-L、-l、-n の 3 つのオプションは相互に排他的で、2 つ以上が指定された場合には最後に現れたものが有効です。
- l [number] (小文字のエル) このオプションは、-L number と同等の意味を持ちます。number を省略すると、1 とみなされます。この -I オプションを指定した場合は、x オプションが強制的に有効となります。
- n number できるだけ多くの、ただし最大 number 個の引数を標準入力から読み込んで utility を呼び出します。number は正の 10 進整数です。次の場合は、より少ない引数が使用されます。
 - コマンド行の累積長が、-s オプションで指定された長さ (-s オプション省略時には LINE_MAX が示す長さ) を超えた場合。
 - 繰り返し実行の最終回に、残っている引数の数が number 未満でゼロではない場合

- p** プロンプトモードを表します。*utility* を呼び出すたびに、実行してもよいかどうかユーザーに問い合わせます。トレースモード (-t) が自動的に有効となってその回のコマンド内容を表示し、その後にプロンプトメッセージが表示されます。この表示は標準エラー出力に現れます。*/dev/tty* から肯定の応答 (ユーザーのロケールに固有) を読み取るとコマンドを実行し、肯定の応答でない場合はその回の *utility* の実行は行われません。
- s size** コマンド行の長さが *size* で示すバイト数を超えない範囲で、できるだけ多くの引数を標準入力から読み込んで *utility* を呼び出します。*size* は正の 10 進整数です。次の場合は、より少ない引数を使用されます。
- 引数の合計個数が *-n* オプションで指定した数を超えた場合
 - 行数が *-L* オプションで指定した値を超えた場合
 - 行の長さが *size* バイトに到達する前にファイルの終わりを検出した場合
- 「機能説明」節で記述されている制限が満たされている場合は、*size* の値としては、少なくとも {*LINE_MAX*} バイトまでの値がサポートされます。その制限値を超えたり、システムがサポートしている最大値を超えたりした場合でも、エラーとは見なされません。そのような場合 *xargs* は、制限の範囲内で、サポートできる最大値を使用します。
- t** トレースモードを有効にします。生成されたコマンド行は、実行する直前に標準エラー出力に書き込まれます。
- x** 引数の数を表す *number* 引数 (*-n* オプションを参照) あるいは行数を表す *number* 引数 (*-L* オプションを参照) を含むコマンド行が、明に指定された長さ (*-s* オプションを参照) あるいは暗に指定された長さを超えた場合、実行を中止します。

オペランド 次のオペランドがサポートされています。

utility 実行するユーティリティの名前。環境変数 *PATH* を使った検索パスにより得られます (*environ(5)* を参照)。*utility* を省略すると、デフォルトとして *echo(1)* ユーティリティが実行されます。*shell_builtins(1)* に定義されている特殊な組み込みユーティリティの名前を *utility* で指定した場合の処理結果は定義されていません。

argument utility *utility* を実行するときの最初のオプションまたはオペランド。

使用法 *xargs* から実行されるユーティリティは、再度呼び出されても現データストリームを使うかぎり実行は失敗する、ということがわかっている場合、終了ス

ステータス 255 を使用して xargs に対し実行を終了するよう通知できます。したがって *utility* は、偶然 255 を返すことのないよう、適切な終了コード値を明に指定して実行を終了することが必要です。

入力データは行の集まりとして解析されます。引数は空白文字により区切られます。xargs を使って `find dir -print` や `ls` などのコマンドの出力を、実行対象コマンドの入力とする場合、ファイル名に空白文字や復帰改行文字が含まれていると、処理の結果は予測できません。これを防ぐには、見つかった各ファイル名を引用符付きの文字列に変換するスクリプトを `find` を使って呼び出し、そのスクリプトを xargs にパイプでつなげるようにしてください。なお xargs が使う引用符の規則は、シェルの規則とは異なります。同じ規則を採用しないのは、既存のアプリケーションが現状の規則に依存しているのに対し、シェルの構文規則はそれと互換性を持たないためです。文字列を xargs が正しく解釈できる形式に変換する簡単な方法は、各文字の前にバックスラッシュ (\) を付加することです。

{ARG_MAX} 値が大きいシステムにおいては、xargs は {LINE_MAX} より長いコマンド行を生成することがあります。これはユーティリティーを呼び出すうえでは問題ではありません。xargs を使ってテキストファイルを作成する場合、ユーザーは -s オプションを使ってコマンド行の長さを明確に設定しなければなりません。

xargs ユーティリティーは、エラーが発生すると終了ステータス 127 を返します。これによりアプリケーションは、「ユーティリティーが見つからなかった」エラーと「呼び出したユーティリティーがエラーで終了した」ステータスとを見分けることができます。127 という値を選んだのは、通常この値はほかの意味で使われることがないためです。ほとんどのユーティリティーは、小さい値を使って「一般的なエラーステータス」を表し、128 より大きな値は、シグナル受信による終了の場合と区別が困難になりかねないためです。同じような観点から、ユーティリティーは見つかったが実行できなかった、という場合に使う値として 126 が選ばれています。

使用例

例1 xargs コマンドの使用

次の例では、ディレクトリ \$1 中の全ファイルをディレクトリ \$2 へ `mv` コマンドで移動することと、各実行の直前にそのコマンドの内容を表示することを指定しています。

```
example% ls $1 | xargs -I {} -t mv $1/{} $2/{} 
```

次では、カッコで囲まれたコマンドの実行結果を 1 行にまとめ、それを `log` というファイルの終わりに追加出力することを指定しています。

```
example% (logname; date; printf "%s\n" "$0 $*") | xargs >>log 
```

次のコマンドは、元来コマンド行引数として入力されたいくつかの対の引数を使って `diff` を実行します。なお元の引数リストの各要素には空白文字が埋め込まれていない、と仮定しています。

例1 xargs コマンドの使用 (続き)

```
example% printf "%s\n" "$*" | xargs -n 2 -x diff
```

ユーザーは、現在のディレクトリに含まれるファイルのうち、どのファイルのアーカイブを作成するかを確認されます。ファイルのアーカイブが作成されて arch ; に渡されます (a: 1 度に 1 ファイルずつ、b: 1 度に複数のファイル)。

```
example% ls | xargs -p -L 1 ar -r arch
ls | xargs -p -L 1 | xargs ar -r arch
```

次のコマンドは、元来コマンド行引数として入力されたいくつかの対の引数を使って diff コマンドを実行します。

```
example% echo $* | xargs -n 2 diff
```

環境変数

xargs の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#)を参照してください。LANG、LC_ALL、LC_COLLATE、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

PATH utility の位置を指定します。

肯定応答は、ユーザーのロケールの LC_MESSAGES カテゴリの yesexpr キーワードで定義されている拡張正規表現を使用して処理されます。LC_COLLATE カテゴリに指定されているロケールは、yesexpr に定義されている式に使用される範囲、同等クラス、および複数文字照合要素の動作を定義します。LC_CTYPE に指定されているロケールは、テキストデータのバイト列を文字として解釈するロケールや、yesexpr に定義されている式に使用される文字クラスの動作を定義します。[locale\(5\)](#)を参照してください。

終了ステータス 次の終了ステータスが返されます。

- 0 utility で指定したユーティリティのすべての実行の終了ステータスは 0 だった。
- 1–125 指定された要求に合うコマンド行が生成できなかった、または utility で指定されたユーティリティのうちいくつかの実行で 0 でない終了ステータスが返されました。またはほかの何らかのエラーが発生しました。
- 126 utility で指定されたユーティリティが見つかったが実行できなかった。
- 127 utility で指定されたユーティリティが見つからなかった。

指定された要求に合うコマンド行が生成できない、またはユーティリティが実行できない、またはユーティリティの実行がシグナルにより中断した、またはユーティリティの実行が終了ステータス 255 で終了した場合、xargs ユーティリティは診断メッセージを出力し、残りの入力データを処理することなく終了します。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
CSI	有効
インタフェースの安定性	確実
標準	standards(5) を参照してください。

関連項目 [echo\(1\)](#), [shell_builtins\(1\)](#), [attributes\(5\)](#), [environ\(5\)](#), [standards\(5\)](#)

名前	xgettext – extract gettext call strings from C programs												
形式	<pre>xgettext [-ns] [-a [-x <i>exclude-file</i>]] [-c <i>comment-tag</i>] [-d <i>default-domain</i>] [-j] [-m <i>prefix</i>] [-M <i>suffix</i>] [-p <i>pathname</i>] -i <i>filename...</i></pre> xgettext -h												
機能説明	The xgettext utility is used to automate the creation of portable message files (.po). A .po file contains copies of “C” strings that are found in ANSI C source code in <i>filename</i> or the standard input if ‘-’ is specified on the command line. The .po file can be used as input to the msgfmt(1) utility, which produces a binary form of the message file that can be used by application during run-time. xgettext writes <i>msgid</i> strings from gettext(3C) calls in <i>filename</i> to the default output file <i>messages.po</i>. The default output file name can be changed by -d option. <i>msgid</i> strings in <i>dgettext()</i> calls are written to the output file <i>domainname.po</i> where <i>domainname</i> is the first parameter to the <i>dgettext()</i> call. By default, xgettext creates a .po file in the current working directory, and each entry is in the same order that the strings are extracted from <i>filenames</i>. When the -p option is specified, the .po file is created in the <i>pathname</i> directory. An existing .po file is overwritten. Duplicate <i>msgids</i> are written to the .po file as comment lines. When the -s option is specified, the .po is sorted by the <i>msgid</i> string, and all duplicated <i>msgids</i> are removed. All <i>msgstr</i> directives in the .po file are empty unless the -m option is used.												
オプション	The following options are supported: <table><tr><td>-n</td><td>Add comment lines to the output file indicating file name and line number in the source file where each extracted string is encountered. These lines appear before each <i>msgid</i> in the following format: # # File: <i>filename</i>, line: <i>line-number</i></td></tr><tr><td>-s</td><td>Generate output sorted by <i>msgids</i> with all duplicate <i>msgids</i> removed.</td></tr><tr><td>-a</td><td>Extract all strings, not just those found in gettext(3C), and <i>dgettext()</i> calls. Only one .po file is created.</td></tr><tr><td>-c <i>comment-tag</i></td><td>The comment block beginning with <i>comment-tag</i> as the first token of the comment block is added to the output .po file as # delimited comments. For multiple domains, xgettext directs comments and messages to the prevailing text domain.</td></tr><tr><td>-d <i>default-domain</i></td><td>Rename default output file from <i>messages.po</i> to <i>default-domain.po</i>.</td></tr><tr><td>-j</td><td>Join messages with existing message files. If a .po file does not exist, it is created. If a .po file does exist, new messages are appended. Any duplicate <i>msgids</i> are commented out in the resulting .po file. Domain</td></tr></table>	-n	Add comment lines to the output file indicating file name and line number in the source file where each extracted string is encountered. These lines appear before each <i>msgid</i> in the following format: # # File: <i>filename</i> , line: <i>line-number</i>	-s	Generate output sorted by <i>msgids</i> with all duplicate <i>msgids</i> removed.	-a	Extract all strings, not just those found in gettext(3C) , and <i>dgettext()</i> calls. Only one .po file is created.	-c <i>comment-tag</i>	The comment block beginning with <i>comment-tag</i> as the first token of the comment block is added to the output .po file as # delimited comments. For multiple domains, xgettext directs comments and messages to the prevailing text domain.	-d <i>default-domain</i>	Rename default output file from <i>messages.po</i> to <i>default-domain.po</i> .	-j	Join messages with existing message files. If a .po file does not exist, it is created. If a .po file does exist, new messages are appended. Any duplicate <i>msgids</i> are commented out in the resulting .po file. Domain
-n	Add comment lines to the output file indicating file name and line number in the source file where each extracted string is encountered. These lines appear before each <i>msgid</i> in the following format: # # File: <i>filename</i> , line: <i>line-number</i>												
-s	Generate output sorted by <i>msgids</i> with all duplicate <i>msgids</i> removed.												
-a	Extract all strings, not just those found in gettext(3C) , and <i>dgettext()</i> calls. Only one .po file is created.												
-c <i>comment-tag</i>	The comment block beginning with <i>comment-tag</i> as the first token of the comment block is added to the output .po file as # delimited comments. For multiple domains, xgettext directs comments and messages to the prevailing text domain.												
-d <i>default-domain</i>	Rename default output file from <i>messages.po</i> to <i>default-domain.po</i> .												
-j	Join messages with existing message files. If a .po file does not exist, it is created. If a .po file does exist, new messages are appended. Any duplicate <i>msgids</i> are commented out in the resulting .po file. Domain												

- directives in the existing .po file are ignored. Results not guaranteed if the existing message file has been edited.
- m *prefix* Fill in the *msgstr* with *prefix*. This is useful for debugging purposes. To make *msgstr* identical to *msgid*, use an empty string ("") for *prefix*.
 - M *suffix* Fill in the *msgstr* with *suffix*. This is useful for debugging purposes.
 - p *pathname* Specify the directory where the output files will be placed. This option overrides the current working directory.
 - x *exclude-file* Specify a .po file that contains a list of *msgids* that are not to be extracted from the input files. The format of *exclude-file* is identical to the .po file. However, only the *msgid* directive line in *exclude-file* is used. All other lines are simply ignored. The -x option can only be used with the -a option.
 - h Print a help message on the standard output.

属性 See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	text/locale

関連項目 [msgfmt\(1\)](#), [gettext\(3C\)](#), [attributes\(5\)](#)

注意事項 `xgettext` is not able to extract cast strings, for example ANSI C casts of literal strings to `(const char *)`. This is unnecessary anyway, since the prototypes in `<libintl.h>` already specify this type.

In messages and translation notes, lines greater than 2048 characters are truncated to 2048 characters and a warning message is printed to `stderr`.

名前	xstr – extract strings from C programs to implement shared strings
形式	<pre>xstr -c filename [-v] [-l array] xstr [-l array] xstr filename [-v] [-l array]</pre>
機能説明	xstr maintains a file called <code>strings</code> into which strings in component parts of a large program are hashed. These strings are replaced with references to this common area. This serves to implement shared constant strings, which are most useful if they are also read-only. The command: <pre>example% xstr -c filename</pre> extracts the strings from the C source in <code>name</code>, replacing string references by expressions of the form <code>&xstr[number]</code> for some number. An appropriate declaration of <code>xstr</code> is prepended to the file. The resulting C text is placed in the file <code>x.c</code>, to then be compiled. The strings from this file are placed in the <code>strings</code> data base if they are not there already. Repeated strings and strings which are suffixes of existing strings do not cause changes to the data base. After all components of a large program have been compiled, a file declaring the common <code>xstr</code> space called <code>xs.c</code> can be created by a command of the form: <pre>example% xstr</pre> This <code>xs.c</code> file should then be compiled and loaded with the rest of the program. If possible, the array can be made read-only (shared) saving space and swap overhead. <code>xstr</code> can also be used on a single file. A command: <pre>example% xstr filename</pre> creates files <code>x.c</code> and <code>xs.c</code> as before, without using or affecting any <code>strings</code> file in the same directory. It may be useful to run <code>xstr</code> after the C preprocessor if any macro definitions yield strings or if there is conditional code which contains strings which may not, in fact, be needed. <code>xstr</code> reads from the standard input when the argument <code>'-'</code> is given. An appropriate command sequence for running <code>xstr</code> after the C preprocessor is: <pre>example% cc -E name.c xstr -c - example% cc -c x.c example% mv x.o name.o</pre> <code>xstr</code> does not touch the file <code>strings</code> unless new items are added; thus <code>make(1S)</code> can avoid remaking <code>xs.o</code> unless truly necessary.

オプション	<code>-c filename</code>	Take C source text from <i>filename</i> .
	<code>-v</code>	Verbose: display a progress report indicating where new or duplicate strings were found.
	<code>-l array</code>	Specify the named <i>array</i> in program references to abstracted strings. The default array name is <code>xstr</code> .
ファイル	<code>strings</code>	data base of strings
	<code>x.c</code>	massaged C source
	<code>xs.c</code>	C source for definition of array <code>"xstr*(rq</code>
	<code>/tmp/xs*</code>	temp file when <i>xstr filename</i> doesn't touch <code>strings</code>
属性	See attributes(5) for descriptions of the following attributes:	

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

関連項目 [make\(1S\)](#), [attributes\(5\)](#)

使用上の留意点 If a string is a suffix of another string in the data base, but the shorter string is seen first by `xstr` both strings will be placed in the data base, when just placing the longer one there would do.

注意事項 Be aware that `xstr` indiscriminately replaces all strings with expressions of the form `&xstr[number]` regardless of the way the original C code might have used the string. For example, you will encounter a problem with code that uses `sizeof()` to determine the length of a literal string because `xstr` will replace the literal string with a pointer that most likely will have a different size than the string's.

To circumvent this problem:

- use `strlen()` instead of `sizeof()`; note that `sizeof()` returns the size of the array (including the null byte at the end), whereas `strlen()` doesn't count the null byte. The equivalent of `sizeof("xxx")` really is `(strlen("xxx")+1)`.
- use `#define` for operands of `sizeof()` and use the define'd version. `xstr` ignores `#define` statements. Make sure you run `xstr` on *filename* before you run it on the preprocessor.

You will also encounter a problem when declaring an initialized character array of the form

```
char x[] = "xxx";
```

`xstr` will replace `xxx` with an expression of the form `&xstr[number]` which will not compile. To circumvent this problem, use `static char *x = "xxx"` instead of `static char x[] = "xxx"`.

名前	yacc – 構文解析プログラムの生成
形式	yacc [-dltVv] [-b <i>file_prefix</i>] [-Q [y n]] [-P <i>parser</i>] [-p <i>sym_prefix</i>] <i>file</i>
機能説明	yacc コマンドは、文脈自由文法を、LALR(1) 構文解析アルゴリズムを実行する簡易オートマトン用のテーブル群に変換します。文法はあいまいであっても構いません。指定された優先順位規則に従ってあいまいな記述を処理します。 出力ファイル <i>y.tab.c</i> は、<i>yyparse()</i> 関数を生成するために C コンパイラを使ってコンパイルする必要があります。この関数は、字句解析プログラム <i>yylex()</i>、<i>main()</i>、およびエラー処理ルーチン <i>yyerror()</i> とともにロードすることが必要です。これらのルーチンはユーザーが用意しなければなりません。yacc が使用できる字句解析プログラムを生成するには、lex(1) コマンドを使うと便利です。
オプション	サポートしているオプションは、次のとおりです。 -b <i>file_prefix</i> すべての出力ファイルに対して、接頭辞として <i>y</i> の代わりに <i>file_prefix</i> を使用します。コードファイルの <i>y.tab.c</i>、ヘッダーファイルの <i>y.tab.h</i> (<i>-d</i> が指定されたとき)、記述ファイルの <i>y.output</i> (<i>-v</i> が指定されたとき) は、それぞれ <i>file_prefix.tab.c</i>、<i>file_prefix.tab.h</i>、および <i>file_prefix.output</i> に変更されます。 -d yacc またはユーザーが割り当てたトークン番号を、ユーザーが宣言したトークン名に対応させる <i>#define</i> 文を含んだ <i>y.tab.h</i> ファイルを生成します。この対応付けにより、<i>y.tab.c</i> 以外のソースファイルからトークン番号を参照することが可能となります。 -l <i>y.tab.c</i> 中に生成するコードには <i>#line</i> 構造を含めないことを指定します。このオプションは、プログラムの文法および動作が完全にデバッグされるまでは使用すべきではありません。 -p <i>sym_prefix</i> yacc が生成するすべての外部名の接頭辞として、<i>yy</i> の代わりに <i>sym_prefix</i> を使用します。影響を受ける名前としては、関数 <i>yyparse()</i>、<i>yylex()</i>、<i>yyerror()</i>、および変数 <i>yylval</i>、<i>yychar</i>、<i>yydebug</i> が挙げられます。ここでは、これらの6つのシンボルを便宜上デフォルト名を使って表しています。局所的な名前も <i>-p</i> オプションの影響を受けます。しかし、<i>-p</i> オプションは yacc が生成する <i>#define</i> シンボルには影響を及ぼしません。 -P <i>parser</i> /usr/share/lib/ccs/yaccpar の代わりに使用する構文解析プログラムを指定できます。たとえば、以下のように指定できます。 example% yacc -P ~/myparser parser.y -Q[y n] <i>-Qy</i> オプションは、バージョン情報を <i>y.tab.c</i> 中に書き出すことを表します。この情報により、どのバージョンの yacc によって

ファイルが生成されたかが分かります。-Qn オプションは、このバージョン情報を書き出さないことを表すもので、これがデフォルトとなります。

- t 実行時デバッグコードをコンパイルすることをデフォルトとします。このコードは条件付きコンパイル制御行とともに、常に `y.tab.c` ファイル中に置かれます。デフォルトでは、この部分のコードを含まないよう `y.tab.c` はコンパイルされます。-t オプションを指定したか否かに関わらず、実行時デバッグコードは、プリプロセッサシンボルである `YYDEBUG` により制御されます。`YYDEBUG` の値がゼロ以外るとき、実行時デバッグコードが含まれるようにコンパイルされます。値が `0` のときは含まれません。このコードなしで生成されたプログラムは、サイズが小さくなり、実行の速度も少し速くなります。
- v 構文解析用テーブル群、および文法上のあいまいさにより発生した矛盾点に関するレポートを記述した `y.output` ファイルを生成します。
- V yacc 用のバージョン情報を標準エラー出力に書き出します。

オペランド

次のオペランドを指定できます。

`file` 構文解析プログラム生成の対象とする命令を含んでいるファイルのパス名

使用例

例1 yacc コマンドの使用

`cc` コマンドのライブラリ検索オペランドに指定することにより、`yacc` ライブラリを使用することが可能です。`main` という `yacc` ライブラリを使用するには、以下のコマンドを実行します。

```
example% cc y.tab.c -ly
```

`lex` と `yacc` の両方のライブラリに `main` が含まれています。`yacc` ライブラリの `main` を使うには以下のコマンドを実行します。

```
example% cc y.tab.c lex.yy.c -ly -ll
```

これにより、最初に `yacc` ライブラリが検索されるので、`yacc` ライブラリ中の `main` が用いられます。

`yacc` ライブラリには、通常はアプリケーションプログラマが作成する 2 つの簡単な関数が定義されています。これらの関数は以下のようなコードです。

```
#include <locale.h>
int main(void)
{
    extern int yyparse();
```

例1 yacc コマンドの使用 (続き)

```

    setlocale(LC_ALL, "");

    /* If the following parser is one created by lex, the
       application must be careful to ensure that LC_CTYPE
       and LC_COLLATE are set to the POSIX locale. */
    (void) yyparse();
    return (0);
}

#include <stdio.h>

int yyerror(const char *msg)
{
    (void) fprintf(stderr, "%s\n", msg);
    return (0);
}

```

環境変数

yacc の実行に影響を与える次の環境変数についての詳細は、[environ\(5\)](#) を参照してください。LANG、LC_ALL、LC_CTYPE、LC_MESSAGES、およびNLSPATH。

yacc は EUC の主および補助コードセット中の文字を 1 文字トークン記号として扱うことができます。この場合、EUC コードは引用符によって囲まれた 1 文字の終端記号でなければなりません。yacc は、`yylex()` がこれらの 1 文字トークン記号に対して 1 つのワイド文字 (`wchar_t`) を返すものと想定しています。

終了ステータス

次の終了ステータスが返されます。

0 正常終了。
>0 エラーが発生した。

ファイル

y.output	生成した構文解析プログラムの状態遷移
y.tab.c	生成した構文解析プログラムのソースコード
y.tab.h	生成した構文解析プログラムのヘッダファイル
yacc.acts	一時ファイル
yacc.debug	一時ファイル
yacc.tmp	一時ファイル
yaccpar	C プログラム用の構文解析プログラムのプロトタイプ

属性

属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	developer/base-developer-utilities
インタフェースの安定性	确实
標準	standards(5) を参照してください。

- 関連項目

[lex\(1\)](#),
[attributes\(5\)](#),
[environ\(5\)](#),
[standards\(5\)](#)
- 診断

還元 - 還元衝突 および シフト - 還元衝突の矛盾の数が標準エラー出力に報告されます。詳細な情報は `y.output` ファイルに出力されます。また開始記号からたどることのできない規則があれば、これについても報告されます。
- 注意事項

出力するファイル名が固定されているので、1つのディレクトリ内で同時に複数の `yacc` プロセスを動作させることはできません。

識別名にはドル記号 (\$) を使用しないようにしてください。

- 名前

yes – generate repetitive affirmative output
- 形式

yes [*term*] . . .
- 機能説明

The yes utility repeatedly outputs y, or if *term* is specified, *term* is output repeatedly. In the output, either y or *term* is followed by a NEWLINE. Multiple arguments are output separated by spaces and followed by a NEWLINE. To terminate yes, issue an interrupt character.

yes can be used to respond programatically to programs that require an interactive response.
- 属性

See [attributes\(5\)](#) for descriptions of the following attributes:

ATTRIBUTE TYPE	ATTRIBUTE VALUE
Availability	system/core-os

- 関連項目

[attributes\(5\)](#)

名前	ypcat – NIS データベース内の値の出力
形式	ypcat [-kx] [-d <i>ypdomain</i>] <i>mname</i>
機能説明	ypcat コマンドは、<i>mname</i> で指定された NIS ネームサービスマップ内の値を出力します。<i>mname</i> は、マップ名かマップニックネームのいずれかです。ypcat は NIS ネームサービスを使用するため、NIS サーバーは指定されません。 NIS ネットワークサービスの概要については、ypfiles(4) を参照してください。
オプション	-k 値がヌルであったり、キーが値の一部ではないマップについて、キーを表示します。<i>/etc</i> 内に ASCII バージョンがあるファイルから取り出されたマップは、このクラスに当てはまりません。 -d <i>ypdomain</i> デフォルトドメイン以外のドメインを指定します。 -x マップのニックネームを表示します。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/network/nis

関連項目 [ypmatch\(1\)](#), [ypfiles\(4\)](#), [attributes\(5\)](#)

名前 ypmatch – NIS マップ内の 1 つまたは複数のキーの値の出力

形式 ypmatch [-k] [-t] [-d *domain*] *key* [*key*]... *mname*
ypmatch -x

機能説明 ypmatch は、*mname* で指定された NIS のネームサービスマップ内の 1 つまたは複数のキーに対応付けられた値を出力します。mname は、マップ名またはマップのニックネームです。

複数のキーを指定でき、すべてのキーが同じマップ内で検索されます。これらのキーは、大文字と小文字を区別し同じ長さにしなければなりません。またパターン照合は使用できません。キーが一致しない場合、診断メッセージが生成されます。

オプション サポートしているオプションは、次のとおりです。

-k キーの値を表示する前に、キー自体とコロン(:)を付けて出力します。

-t マップのニックネームの使用を禁止します。

-d *domain* デフォルトドメイン以外のドメインを指定します。

-x マップのニックネームテーブルを表示します。このテーブルには、コマンドが認識しているニックネーム、および各ニックネームに対応付けられているマップネームが示されます。

オペランド 次のオペランドを指定できます。

mname NIS's のネームサービスマップ

終了ステータス 次の終了ステータスが返されます。

0 正常終了。

1 エラーが発生した。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/network/nis

関連項目 [ypcat\(1\)](#), [ypfiles\(4\)](#), [attributes\(5\)](#)

注意事項 十分な数のファイル記述子が使用できない場合、ypmatch は yp 操作に失敗し、RPC エラーメッセージを出力します。このような場合は、ファイル記述子の数を増やす必要があります。

名前	yppasswd - NIS データベース内のネットワークパスワードの変更				
形式	yppasswd [username]				
機能説明	yppasswd ユーティリティーは、NIS (Network Information Service) データベース内のユーザー <i>username</i> に対応付けられたネットワークパスワードを変更します。ユーザーが、keylogin(1) を実行しており、NIS の <code>publickey.byname</code> マップ内にそのユーザー用の公開鍵と秘密鍵のペアが存在する場合には、yppasswd は新しいパスワードを使用して秘密鍵を再度、暗号化します。NIS パスワードは、ユーザーのマシンで設定されているローカルなパスワードとは異なる場合もあります。 yppasswd は、古い NIS パスワードの入力を求めたあとで、新しいパスワードの入力を求めます。変更を有効にするためには、古いパスワードを正しく入力する必要があります。新しいパスワードは、誤った入力を防ぐために 2 回入力する必要があります。 新しいパスワードの長さは、大文字や小文字、記号を組み合わせる場合は 4 文字以上、大文字または小文字だけのアルファベットの場合には 6 文字以上でなければなりません。しかし、ユーザーが執拗に要求する場合には、これらの規則は緩和されます。パスワードを変更できるのは、名前の所有者とスーパーユーザーだけです。ルートマスター上のスーパーユーザーは、古いパスワードの入力を求められず、パスワード作成の要件に従う必要もありません。 新しいパスワードを有効にするためには、NIS パスワードデーモン <code>rpc.yppasswdd</code> が NIS サーバー上で実行していなければなりません。				
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。				
	<table border="1"> <thead> <tr> <th>属性タイプ</th><th>属性値</th></tr> </thead> <tbody> <tr> <td>使用条件</td><td>system/network/nis</td></tr> </tbody> </table>	属性タイプ	属性値	使用条件	system/network/nis
属性タイプ	属性値				
使用条件	system/network/nis				
関連項目	keylogin(1) , login(1) , passwd(1) , getpwnam(3C) , getspnam(3C) , secure_rpc(3NSL) , nsswitch.conf(4) , attributes(5)				
警告	このコマンドを使用してユーザーがパスワードを正常に変更したあとでも、新しいパスワードによる login(1) が行なえるのは、ユーザーのパスワード情報とシャドー情報が NIS から取得される場合だけです。 getpwnam(3C) 、 getspnam(3C) 、および nsswitch.conf(4) を参照してください。				
注意事項	現在、yppasswd は passwd(1) コマンドのラッパーでしかないため、yppasswd の使用はお勧めできません。代わりに passwd(1) を使用してください。 <code>-r nis</code> オプションを指定して passwd(1) を使用すると、同じ結果が得られます。この方法は、現在利用できる種々のネームサービスすべてに渡って一貫性のあるものです。				

使用上の留意点 更新プロトコルは、1つのRPC呼び出しで、すべての情報を確認することなくサーバーに渡します。このため、古いパスワードを間違っても、新しいパスワードの入力を終えるまで通知されることはありません。

名前 ypwhich - NIS サーバーまたはマップマスターの名前の表示

形式 ypwhich [-d *domain*] [[-t] -m [*mname*] | [-Vn] *hostname*]

ypwhich -x

機能説明 ypwhich は、NIS クライアントに NIS ネームサービスを提供している NIS サーバーの名前、またはマップのマスターである NIS サーバーの名前を返します。引数なしで呼び出した場合、このコマンドはローカルマシンの NIS サーバーを表示します。hostname を指定すると、そのマシンの照会が行われ、マシンで使用されている NIS マスターが表示されます。

NIS ネームサービスの概要については、[ypfiles\(4\)](#) を参照してください。

オプション

- d *domain* デフォルトドメインの代わりに *domain* を使用します。
- t マップのニックネームの使用を禁じます。
- m *mname* マップのマスター NIS サーバーを見つけます。-m を使用してホスト名を指定することはできません。*mname* には、マップ名、またはマップのニックネームを指定できます。*mname* を省略すると、使用可能なマップの一覧が生成されます。
- x マップのニックネームの対応テーブルを表示します。
- Vn ypbind のバージョン。デフォルトは V3 です。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/network/nis

関連項目 [ypfiles\(4\)](#), [attributes\(5\)](#)

名前 zlogin - ゾーンにログインする

形式 zlogin [-dCE] [-e c] [-l username] zonename
 zlogin [-ES] [-e c] [-l username] zonename utility
 [argument]...

機能説明 zlogin ユーティリティーは、オペレーティングシステムゾーンにログインするために使用します。このユーティリティーは、大域システムゾーンで操作しているユーザーのみが使用できますが、すべての特権で実行する必要があります。さらに、ユーザーは「オプション」セクションで説明されている、特定のオプションを使用することが承認されている必要があります。

zlogin は、前にスラッシュ文字が付き、接尾辞として指定された zonename に任意で含まれる承認文字列をチェックします。省略すると、承認はすべてのゾーンと一致します。

zlogin は、次の3つのモードのいずれかで動作します。

対話型モード ユーティリティーの引数を指定せず、zlogin プロセスの標準入力ファイル記述子が tty デバイスの場合、zlogin は対話型モードで動作します。このモードの zlogin は、ログインセッションで使用するための新しい仮想端末を作成します。tty デバイスを必要とするプログラム ([vi\(1\)](#)) などは、このモードで正しく動作します。このモードの zlogin は、[login\(1\)](#) を呼び出すことで適切なログインセッションを生成します。

非対話型モード ユーティリティーを指定した場合、zlogin は非対話型モードで動作します。このモードで使用した場合でも、標準入力、標準出力、および標準エラー出力を使用することができ、終了時に utility の終了ステータスが返されるので、スクリプトを作成するときにも利用できます。このモードの zlogin は、ユーザーの環境を設定するために [su\(1M\)](#) を呼び出して、ログイン環境を生成します。

指定されたコマンドは文字列として渡され、非大域ゾーンで実行されているシェルによって解釈されます。[rsh\(1\)](#) を参照してください。

コンソールモード -c オプションを指定した場合、ゾーンコンソールデバイスに接続され、zlogin はコンソールモードで動作します。ゾーンコンソールは、そのゾーンがインストール済みの状態になると使用可能になります。コンソールへの接続は、ゾーンをリブートしても切断されることはありません。

オプション	サポートしているオプションは、次のとおりです。
-c	ゾーンコンソールに接続します。ゾーンコンソールにアクセスするには、承認 <code>zone.manage/zonename</code> が必要です。
-d	ゾーンが停止している場合は、コンソールから切断します。このオプションは、-c とともにのみ指定できます。
-e <i>c</i>	拡張機能にアクセスしたり、ログインから切断したりするためのキーシーケンスを使用する場合は、別のエスケープ文字 <i>c</i> を指定します。デフォルトのエスケープ文字はチルド (~) です。
-E	エスケープシーケンス文字を使用して拡張機能にアクセスしたり、ログインから切断したりする機能を無効にします。
-l <i>username</i>	ゾーンログイン時のユーザー名として別の名前 (<i>username</i>) を指定します。このオプションを使用しない場合、使用されるゾーンのユーザー名は <code>root</code> です。このオプションは、-c オプションを指定した場合には無効になります。 ユーザー名は、ゾーン内で有効である必要があります。対話型ログインでは、<code>solaris.zone.login/zonename</code> の承認が必要であり、パスワード承認がゾーンで実行されます。非対話型ログインの場合、またはパスワード承認を省略する場合は、<code>solaris.zone.manage/zonename</code> の承認が必要です。
-s	「安全な」ログインモード。zlogin は、<code>login(1)</code> や <code>su(1M)</code> を呼び出さずに、最小限の処理を行います。ゾーンのユーザー名は <code>root</code> に設定されます。-l オプションを使用してユーザー名を指定した場合は、-s オプションを使用できません。このモードは、コンソールログインでは使用できません。このモードは、ほかの形式のログインを使用できなくなったときに、破損したゾーンを回復するためだけに使用してください。 このオプションを使用するには、<code>solaris.zone.manage/zonename</code> の承認が必要です。
エスケープシーケンス	端末から入力した行がチルド (~) で始まっている場合、その行はエスケープシーケンスと見なされます。-e オプションを使って、エスケープ文字をチルド以外に変更することもできます。 ~, ゾーンから切断します。この操作はログアウトとは異なります。ローカルホストが接続を切断しても、ゾーンが終了するという警告は通知されません。
セキュリティー	大域ゾーン以外のゾーンに実装されたプロセスは、そのゾーンを変更することはできず、すべての子プロセスもゾーンを変更できません。

- オペランド

次のオペランドがサポートされています。

zonename

ログインするゾーンの名前。

utility

指定したゾーンで実行するユーティリティ。

argument...

ユーティリティに渡す引数。
- 終了ステータス

対話型モードおよび非対話型モードでは、非大域ゾーンでコマンドまたはシェルが終了すると、zlogin ユーティリティは終了します。非対話型モードでは、リモートプログラムの終了ステータスはzloginの終了ステータスとして返されます。対話型モードとコンソールログインモードでは、終了ステータスは返されません。zloginは、接続関連のエラーが発生した場合を除いて、終了ステータスとして0を返します。

すべてのモードについて、ゾーンへの接続を確立できなかった場合には、予期しないステータスで接続に失敗するか、または要求した操作を実行するために必要な特権がユーザーにないために、zloginはステータス1で終了します。

要約すると、次の終了値が返されます。

0

ログインに成功しました。

1

アクセス権が拒否されたか、またはゾーンへのログインに失敗しました。

Any

ユーティリティ、またはsu(1M) (非対話型モードで操作している場合) からコードが返される。
- 属性

属性についての詳細は、マニュアルページのattributes(5)を参照してください。

属性タイプ	属性値
使用条件	system/zones
インタフェースの安定性	確実

- 関連項目

login(1), rsh(1), vi(1), su(1M), zoneadm(1M), zonecfg(1M), attributes(5), zones(5)
- 注意事項

zloginは、このユーティリティが開いているファイルまたはユーティリティのアドレス空間の一部がNFSファイルに対応している場合、失敗します。この問題は、実行可能ファイル自体または共有ライブラリも関係します。

名前	zonename – 現在のゾーン名の出力
形式	zonename
機能説明	zonename ユーティリティーは、現在のゾーンの名前を出力します。
終了ステータス	次の終了ステータスが返されます。 0 正常終了。 >0 エラーが発生した。
属性	属性についての詳細は、マニュアルページの attributes(5) を参照してください。

属性タイプ	属性値
使用条件	system/core-os
インタフェースの安定性	確実

関連項目 [zlogin\(1\)](#), [zoneadm\(1M\)](#), [zonecfg\(1M\)](#), [attributes\(5\)](#), [zones\(5\)](#)

名前	zonestat – アクティブなゾーン統計の報告
形式	<pre>zonestat [-z zonelist] [-r reslist] [-n namelist] [-T u d i] [-R reports] [-q] [-x] [-p [-P lines]] [-S cols] interval [duration [report]]</pre>
機能説明	zonestat ユーティリティーは、現在実行中のゾーンの cpu、メモリー、ネットワーク、およびリソース制御使用率について報告します。各ゾーンの使用率は、システムのリソースとゾーンの構成済み制限の両方の割合 (パーセント) として報告されます。 zonestat ユーティリティーは、指定された間隔で一連の間隔レポートを出力します。また、指定された間隔で 1 つ以上のサマリーレポートも任意で出力します。 デフォルトの出力は、cpu、物理メモリー、仮想メモリー、ネットワークの使用率のサマリーです。-r オプションを使用すると、特定のリソースに関する詳細な出力を選択できます。
セキュリティ	非大域ゾーン (NGZ) から実行すると、NGZ から参照可能なプロセッサセットのみが報告されます。NGZ の出力には、メモリーや制限などのその他のすべてのシステムリソースが含まれます。 報告されるすべてのリソースについて、NGZ の使用量が出力されます。システム、大域ゾーン、およびその他のすべてのゾーンによる各リソースの使用量は、[system] で使用されたものとして報告されます。 ネットワークリソースでは、NGZ の使用量のみが出力されます。NGZ は、ほかのゾーンのネットワークリソースと統計からは見えません。 zonestat ユーティリティーを使用するには、proc_info 特権が必要です。この特権は、基本特権セットのメンバーです。
オプション	サポートしているオプションは、次のとおりです。 -n <i>name</i> [,<i>name</i>] 報告するリストリソース名を指定します。pset リソースでは、これはプロセッサセットの名前です。physical-memory、locked-memory、および virtual-memory リソースでは、名前は mem_default および vm_default のみです。ネットワークリソースでは、これはデータリンクの名前です。 dedicated-cpu プロセッサセットは、pset 名 (SUNWtmp_zonename) または単に zonename で指定できます。 psrset で作成されたプロセッサセットは、プール pset 名 (SUNWlegacy_pset id) または単に pset id で指定できます。 リソースセットについて報告するときは、コンマ区切りリストに加えて複数の -n オプションを指定できます。

-p

解析可能な出力。

安定した機械解析可能な形式を使用して出力します。各フィールドはコロン(:)で区切ります。行形式は次のとおりです。

```
report type:resource:field[:field]*
```

-Tが指定されている場合は、各行に接頭辞としてタイムスタンプが付きます。

報告タイプは、report-total、report-average、report-high、およびintervalです。

リソースタイプ

は、header、footer、summary、physical-memory、virtual-memory、locked-memory、process、networkです。

header リソースは、間隔またはサマリーレポートの開始を示すために使用される特別なリソースです。header リソース間のすべての出力行は、同じ報告に属します。各 header (ヘッダー) には、一致する footer (フッター) があります。

残りのフィールドはリソースタイプに固有のものです。詳細は、zonestat ユーティリティの出力を参照してください。

既存のすべての出力フィールドは安定しています。将来のバージョンでは、新しい報告タイプおよびリソースタイプが導入される可能性があります。さらに、将来のバージョンでは、既存の出力 n 行の末尾に新しいフィールドが追加される可能性もあります。

-P line[,line]

解析可能な出力について、解析可能な出力内の出力する行を指定します。次の行のタイプから 1 つ以上を選択できます。

header, footer	各間隔で、サマリーレポートに間隔情報やカウント情報などの詳細を出力する「header (ヘッダー)」が含まれます。各報告のあとには、「footer (フッター)」も出力されます。
resource	各リソースを記述する行。
system	システムによる各リソースの使用率。これには、カーネル、および特定のゾーンに関与することのない、すべてのリソース消費が含まれます。zonestat が非大域ゾーン内から実行される場合、この値はシステムおよびその他のすべてのゾーンで消費される総リソースとなります。ネットワークリソースタイプの system 使用率はサポートされていません。
total	各リソースの合計使用率。
zones	各リソースのゾーンごとの使用率を詳細に示す行。

- r resource[,resource]

summary cpu、physical-memory、virtual memory、network 使用率のサマリー。

psets processor-set

```
limits      processes, lwps, lofi.
```

```
sysv      shm-memory、shm-ids、sem-ids、msg-ids。
```

all すべてのリソースタイプ

デフォルトでは、summary リソースが出力されます。

リソースタイプセットについて指定するときは、コンマ区切りリストに加えて複数の **-r** オプションを指定できます。

システムの `cpu` は、プロセッサセット (`psets`) にパーティション化できます。デフォルトでは、すべての `cpu` が `pset_default` という名前の単一の `pset` にあります。

メモリーはセットへのパーティション化には対応していません。これらのリソースに対する zonestat ユーティリティの出力では、mem_default および vm default という名前で表示されます。

all リソースは、すべてのリソースタイプが報告されるように指定します。

```
-R report[,report]
```

サマリーレポートを出力します。サポートされる報告タイプについては、下記を参照してください。サマリーレポートセットには、コンマ区切りリストに加えて複数の-Rオプションを指定できます。

total リソースごとに、次の詳細を示すサマリーレポートを出力します。

psets コマンド呼び出しの開始以降に使用された
合計 cpu。各ゾーンで使用される割合に
は、ゾーンが実行されていなかった時間も

含まれます。たとえば、ゾーンが実行中に100%のcpuを使用しても、間隔の半分は停止していた場合、サマリーレポートにはゾーンが50%のcpu時間を使用したことが示されます。

memory, limits, sysv

コマンド呼び出し以降に報告されたすべての間隔で使用された平均リソース。この平均の計算には、ゾーンが実行されなかった間隔も含まれます。たとえば、ゾーンが実行中に平均で100Mの物理メモリーを使用しても、間隔の半分でしか実行していなかった場合、サマリーレポートにはゾーンが平均で50Mの物理メモリーを使用したことが示されます。

network

物理帯域幅を使用しているすべてのデータリンクによって送受信された全バイト数の合計合計はコマンドの呼び出しが開始されてから計算され、秒あたりのバイト数に標準化されます。使用された割合は、使用可能な合計帯域幅に基づいて計算されます。

average

totalと同様ですが、ゾーンが実行されている間隔のみが計算に含まれます。たとえば、ゾーンが単一の間隔のみで実行されていて、その間隔中にゾーンが200Mの仮想メモリーを使用した場合、サマリーレポート前に報告された間隔の数に関係なく平均仮想メモリーは200Mになります。

high

zonestat ユーティリティー呼び出しの間隔中に、各リソースおよびゾーンの最大使用量を詳細に示すサマリーレポートを出力します。

-S col[,col]

各リソースを利用するゾーンをソートします。

次のソート列を指定できます。

name

ゾーン名で英数字順にソートします。

used

使用される資源の数量でソートします。ネットワークリソースでは、これはバイトでソートした場合と同じです。

これはデフォルトです。

cap

構成されている上限でソートします。

pcap

使用されている上限の割合(パーセント)でソートします。

shr

割り当てられている共有でソートします。

pshru	使用されている共有の割合 (パーセント) でソートします。
bytes	送受信された合計バイト数でネットワークをソートします。
prbyte	有線で受信されたバイト数の割合でネットワークをソートします。
pobyte	有線で送信されたバイト数の割合でネットワークをソートします。
maxbw	使用された帯域幅の割合でネットワークをソートします。
cpu	サマリーの CPU 使用量でソートします。これはデフォルトです。
physical-memory	サマリーの物理メモリー使用量でソートします。
virtual-memory	サマリーの仮想メモリー使用量でソートします。
network	サマリーのネットワーク使用量でソートします。
network	サマリーのネットワーク使用量でソートします。

- T u | d | i
各レポートのタイムスタンプを含みます。次の形式がサポートされています。
- d 標準日付形式。date(1) を参照してください。--p オプションと同時に使用できません。
- i ISO 8601 に準拠する形式としてフォーマットされた時間。
YYYYMMDDThhmmssZ
- u 時間の内部表現の出力表現。time(2) を参照してください。Unix 時間とも呼ばれます。

- x
拡張表示で詳細情報を表示します。たとえば、ネットワークリソースで使用情况、拡張表示には各仮想データリンクの詳細が一覧表示されます。

- z zonename[,zonename]
報告するゾーンの一覧を指定します。デフォルトでは、すべてのゾーンが報告されます。
- ゾーンセットについて報告するときは、コンマ区切りリストに加えて複数の -z オプションを指定できます。出力には、指定されたゾーンで使用されたリソースが含まれます。

オペランド 次オペランドがサポートされています。

interval

各間隔レポート間で一時停止する長さを秒数で指定します。「デフォルト」の間隔では、ゾーン監視サービスの構成済み間隔が使用されます。[zonestatsd\(1M\)](#)を参照してください。

間隔が必要です。ゼロの間隔は許可されていません。間隔は、`10s` や `1m` のように、`[nh][nm][n s]` の形式で指定できます。

duration

報告する間隔の数を指定します。指定しない場合、デフォルトでは `infinity` に設定されます。コマンド期間は $(interval * duration)$ です。ゼロの期間は無効です。値 `inf` を指定して、明示的に `infinity` を選択することもできます。

期間は、`[nh][nm][ns]` の形式でも指定できます。この場合、期間は実行時間の期間であると解釈されます。実際の期間は、間隔のもっとも近い倍数に切り上げられます。

report

サマリーレポート期間を指定します。たとえば、4 のレポートでは4間隔ごとに報告が生成されます。コマンド期間が報告の倍数ではない場合、最後の報告は残りの間隔に属します。

報告は、`[nh][nm][ns]` の形式でも指定できます。この場合、レポートは指定された期間に、もっとも近い間隔に切り上げられて出力されます。コマンド期間が報告の倍数ではない場合、最後の報告は残りの間隔に属します。

`-R` が必要です。`-R` が指定されていても、報告は指定されていない場合、報告期間はコマンド期間全体となり、指定された報告が実行の最後に生成されます。

出力

次の一覧には、コマンド出力の列ヘッダーが定義されています。

SYSTEM-MEMORY

物理ホストで使用可能なメモリーの合計量。

SYSTEM-LIMIT

物理ホストで使用可能なリソースの最大量。

CPUS

プロセッサセットに割り当てられた `cpu` の数

ONLINE

プロセッサセットに割り当てられた `cpu` のうち、プロセスを実行できる `cpu` の数。

MIN/MAX

システムで設定されたプロセッサに割り当てることができる `cpu` の最大数および最小数。

ZONE

リソースを使用しているゾーン。この列には、ゾーン名に加えて次のものを含めることができます。

[total] システム全体で使用されるリソースの合計数量。

[system] カーネルで使用されるリソース、または特定のゾーンに関連付けない方法で使用されるリソースの数量。

zonestat が非大域ゾーン内で使用される場合、[system] はシステムおよびその他のすべてのゾーンで使用される総リソースを指定します。

ネットワークリソースの場合、ネットワークのシステム使用量は使用できません。

USED

使用されているリソースの量。

%USED

合計リソースの割合 (パーセント) として使用されているリソースの量。

PCT

合計リソースの割合 (パーセント) として使用されているリソースの量。

%PART

ゾーンがバインドされたプロセッサセットで合計 cpu の割合 (パーセント) として使用される cpu の量。大域ゾーンの場合、または [psrset\(1M\)](#) psets が使用されている場合、ゾーンには複数のプロセッサセットにバインドされたプロセスのみを含めることができます。ゾーンに複数のバインディングが見つかった場合、%PART はバインドされたすべての psets が使用される割合になります。[total] および [system] では、%PART はシステム上のすべての cpu が使用される割合 (パーセント) になります。

CAP

特定のリソースに上限を設定するようにゾーンが構成されている場合、この列にその上限が表示されます。

%CAP

ゾーンの構成済み上限の割合として使用されるリソースの量。

SHRS

ゾーンに割り当てられている共有の数。[total] 行では、リソースを共有するすべてのゾーンに割り当てられた共有の合計数です。共有を使用するようにゾーンが構成されていなくても、共有を使用するように構成されているその他のゾーンとリソースを共有している場合、この列にはゾーンの no-fss が含まれます。

%SHRS

ゾーンに割り当てられている合計共有の割合。たとえば、2つのゾーン(それぞれ10共有)が1つのプロセッサセットを共有する場合、各ゾーンの%SHRは50%になります。

%SHRU

ゾーンに割り当てられている共有のうち、リソース100%の割合。共有はリソースの競合がある場合にのみ強制されるため、ゾーンの%SHRUは100%を超えることが可能です。

TOBYTES

データリンクまたは仮想リンクによって送受信されたバイト数。

PRBYTE

物理帯域幅を消費する受信バイト数。

POBYTE

物理帯域幅を消費する送信バイト数。

%PRBYTE

PRBYTEの受信に使用できる物理帯域幅の割合。

%POBYTE

POBYTEの送信に使用できる物理帯域幅の割合。

%PUSE

使用可能な合計物理帯域幅の割合で表示したPRBYTEとPOBYTEの合計。

LINK

データリンクの名前。

MAXBW

データリンク上で構成された最大帯域幅

%MAXBW

構成済みの最大帯域幅の割合で表示したすべての送受信バイト数の合計

使用例

例1 zonestatを使用したcpuおよびメモリ使用率のサマリーの表示

次のコマンドは、cpuおよびメモリ使用率のサマリーを5秒ごとに表示します。

```
# zonestat 5 1
SUMMARY  Cpus/Online: 4/4  Physical: 8063M  Virtual: 11.8G
---CPU---  --PHYSMEM--  ---VMEM---  ---NET---
      ZONE  USED %PART  USED %USED  USED %USED  PBYTE %PUSE
[total]  0.23 5.76% 3211M 39.8% 4191M 34.6% 350M 18.7%  -
[system] 0.03 0.83% 2791M 34.6% 3890M 32.1%  -    -
  global 0.19 4.86% 324M 4.01% 228M 1.89% 200M 10.7%
    zoneA 0.00 0.03% 47.9M 0.59% 36.3M 0.30% 100M 5.3%
    zoneB 0.00 0.02% 48.1M 0.59% 36.4M 0.30% 50M 2.7%
```

例2 zonestatを使用した解析可能な出力の生成
次のコマンドは、解析可能な出力を生成します。各 pset リソースを使用するゾーンごとに1行を5秒間隔で出力します。

```
# zonestat -p -P zones -r psets 5 1
```

例3 zonestatを使用したデフォルトのpsetについての報告
次のコマンドは、デフォルトの pset について1秒に1回、1分間を対象として報告します。

```
# zonestat -r default-pset 1 1m
```

例4 zonestatを使用した合計および最大使用効率の報告
次のコマンドは、10秒に1回、24時間を対象としてメッセージを表示することなく監視し、1時間ごとに合計および最大の報告を生成します。

```
# zonestat -q -R total,high 10s 24h 1h
```

例5 zonestatを使用したデータリンク使用率の報告
次のコマンドは、e1000g0 という名前のデータリンクについて5秒間隔で5回報告します。

```
# zonestat -r network -n e1000g0 5 5
```

終了ステータス 次の終了ステータスが返されます。

- 0 正常終了。
- 1 エラーが発生した。
- 2 無効な使用法。
- 3 svc:system/zones_monitoring: デフォルトで実行なし、または応答なし。

属性 属性についての詳細は、マニュアルページの [attributes\(5\)](#) を参照してください。

属性タイプ	属性値
使用条件	system/zones
インタフェースの安定性	下記を参照。

コマンド呼び出しおよび解析可能な出力は「確実」です。人間が読める形式の出力(デフォルト出力)は「不確実」です。

関連項目	<code>date(1)</code> , <code>prctl(1)</code> , <code>pooladm(1M)</code> , <code>poolcfg(1M)</code> , <code>psrset(1M)</code> , <code>rcapadm(1M)</code> , <code>zoneadm(1M)</code> , <code>zonecfg(1M)</code> , <code>zonestatd(1M)</code> , <code>time(2)</code> , <code>timezone(4)</code> , <code>attributes(5)</code> , <code>privileges(5)</code> , <code>resource_controls(5)</code>
注意事項	zonestat ユーティリティーは、ゾーン監視サービス <code>svc/system/zonestat:default</code> に依存します。zonestat ユーティリティーの実行中に zonestat サービスが停止した場合、zonestat コマンド呼び出しは追加報告を出力せずに終了します。次の報告期間に達する前に zonestat が (CTRL/c、SIGINT によって) 中断された場合、報告 (-R) が出力されます。

