Contents

Preface 5

1 Starting the Korean Solaris Software 9
  Installing Korean Solaris Software 10
  Applications Defaults Files 10
  Hangul-Hanja Conversion 11

2 System Environment 13
  Changing the Default Locale 13
    Locale and Category Terminology 14
    Keybinding for the htt Input Server 14
    Customizing Keybinding Control Keys 15
  Interfacing with the Korean Solaris Localization Facility 15

3 Setting Up Korean Solaris Printing Facilities 17
  Line Printer Support 17
    Korean Solaris Code Filters 18
  Laser Printer Support 19
    Using xetops and xutops Utilities 19

4 TTY Environment and Support 21
  TTY Streams 21
    Traditional STREAMS 21
    Korean Solaris STREAMS 22
TTY Utilities 22
  EUC ioctl Features 23
  termcap 23
  terminfo 24
TTY Commands 24
  setterm Command 24
  /bin/stty Command 25
TTY Setup Examples 25
  Configuring STREAMS for Korean Solaris Software 26
  Configuring STREAMS for the ko.UTF-8 Locale 27
Terminal Support 28
Installing a Terminal 28
  Serial Port Interface for Adding Terminals 28
  Command Line Interface for Adding Terminals 29
  Setting a User’s TTY 30
  Using Packed Code and Johap TTY in the ko Locale 30
  Using Combination Code and Johap TTY in the ko Locale 31
  Using EUC TTY in the ko.UTF-8 Locale 31
  Using Johap TTY in the ko.UTF-8 Locale 31

A OpenWindows Information 33
Starting Up OpenWindows 33
  Setting .cshrc for the Required Environment 33
  htt Input Server and openwin-init Files 34
  Setting the .OWdefaults File 34
  Applications Defaults Files 35
Customizing Mail Transmission and Storage 36
  Mail Transmission Formats 36
  Mail Reception and Storage Formats 37
  Making .mailrc Changes Take Effect 37
  Report of Incoming Mail 38

Index 39
Preface

*Korean Solaris System Administrator’s Guide* provides system administration information specific to Korean Solaris™ operation in the Common Desktop Environment (CDE) and the OpenWindows™ environment. This guide also includes some additional information that advanced users and developers can use to access and control the features of the Korean Solaris operating environment.

Who Should Use This Book

You should read this guide if:

- You need specific instructions on how to set up features for users.
- You are a system administrator who has not used the Korean Solaris operating environment, CDE, or the OpenWindows 3.x environment before.
- You are a developer who needs information on accessing and controlling the Korean features of the Korean Solaris operating environment.
- You are an advanced user who wants to use or customize the Korean Solaris operating environment.
- You want information on a variety of details internal to the operation of the Korean Solaris operating environment.

You should already be familiar with Sun’s standard product documentation and the documentation of the window system that you are using, either CDE or OpenWindows. This guide adds only Korean features.
Before You Read This Book

Before you read this book, please review the product overview and any last-minute changes that arrived too late to be included in this document:

- Korean Solaris Release Overview

Make sure to install your system properly as described in the document appropriate to your hardware platform:

- Solaris Advanced Installation Guide
- Information Library for Solaris 8 (SPARC Platform Edition)
- Information Library for Solaris 8 (Intel Platform Edition)

Each chapter of this manual addresses a different aspect of administration of the Korean Solaris operating environment. Some chapters give step-by-step instructions for using or customizing product features.

Chapter 1, “Starting the Korean Solaris Software,” introduces the Korean Solaris operating environment, including CDE and the locales included in the product.


Chapter 4, “TTY Environment and Support,” covers setting terminals to use the proper protocols for the input and display of Korean characters.

Appendix A, “OpenWindows Information,” describes administration tasks relating to modifications made to the OpenWindows 3.x environment to localize it for Korean and types of mail transmission and storage available.

Related Books

The following books are related to the topic of this book and may prove helpful for further reading.

For information on how to use the window system and associated applications:
What Typographic Changes Mean

The following table describes the typographic changes used in this book.

<table>
<thead>
<tr>
<th>Typeface or Symbol</th>
<th>Meaning</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>AaBbCc123</td>
<td>The names of commands, files, and directories; on-screen computer output</td>
<td>Edit your .login file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Use ls -a to list all files.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>machine_name% You have mail.</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>What you type, contrasted with on-screen computer output</td>
<td>machine_name% su</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Password:</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>Command-line placeholder: replace with a real name or value</td>
<td>To delete a file, type rm filename.</td>
</tr>
<tr>
<td>AaBbCc123</td>
<td>Book titles, new words or terms, or words to be emphasized</td>
<td>Read Chapter 6 in User’s Guide. These are called class options.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You must be root to do this.</td>
</tr>
</tbody>
</table>
Shell Prompts in Command Examples

The following table shows the default system prompt and superuser prompt for the C shell, Bourne shell, and Korn shell.

<table>
<thead>
<tr>
<th>Shell</th>
<th>Prompt</th>
</tr>
</thead>
<tbody>
<tr>
<td>C shell prompt</td>
<td>machine_name%</td>
</tr>
<tr>
<td>C shell superuser prompt</td>
<td>machine_name#</td>
</tr>
<tr>
<td>Bourne shell and Korn shell prompt</td>
<td>$</td>
</tr>
<tr>
<td>Bourne shell and Korn shell superuser prompt</td>
<td>#</td>
</tr>
</tbody>
</table>
Starting the Korean Solaris Software

The Korean Solaris operating environment must be specially set up for using Korean text facilities. This chapter describes the steps required to set up the Korean environment and to start Korean Solaris operation.

The Korean Solaris operating environment provides two window environments, the Common Desktop Environment (CDE) and OpenWindows. CDE is a fully internationalized environment; it does not require most of the administration tasks that OpenWindows requires to handle Korean. For information on starting up OpenWindows, see Appendix A, “OpenWindows Information.” The Korean Solaris product includes three locales:

- **C** – ASCII English environment
- **ko** – Korean extended UNIX code (EUC). This locale supports the Unicode 3.0 standard.
- **ko.UTF-8** – Korean Universal Multiple Octet Coded Character Set (UCS) Transmission Format. This locale supports the Unicode 3.0 standard.
- **korean** – Symbolic link to ko locale.
- **ko_KR.EUC** – Symbolic link to ko locale.
- **ko_KR.UTF-8** – Symbolic link to ko.UTF-8 locale.
Installing Korean Solaris Software

Make sure the Korean Solaris operating environment is installed as directed in the documents appropriate to your hardware platform:

- Solaris Advanced Installation Guide
- Solaris 8 (SPARC Platform Edition) Information Library
- Solaris 8 (Intel Platform Edition) Information Library

Applications Defaults Files

The Korean CDE includes three directories for applications defaults. One is for system-wide defaults, and two are specific to locale features:

- The /usr/dt/app-defaults/C directory stores system wide application defaults. These values are for the C locale.
- The /usr/dt/app-defaults/ko directory stores application default values specific to the ko locale.
- The /usr/dt/app-defaults/ko.UTF-8 directory stores application default values specific to the ko.UTF-8 locale.

For the ko locale:

- The $OPENWINHOME/lib/locale/ko/app-defaults/Htt file has all htt resource default values.
- The /usr/dt/app-defaults/ko/Sdthanja file has all sdtanja resource default values.

For the ko.UTF-8 locale:

- The $OPENWINHOME/lib/locale/ko.UTF-8/app-defaults/Htt file has all htt resource default values.
The /usr/dt/app-defaults/ko.UTF-8/Sdthanja file has all sdthanja resource default values.

Hangul-Hanja Conversion

Hangul-Hanja conversion is maintained by Hanja Tool, which manages two system files:

- In the ko locale, the /usr/lib/mle/ko/syshjd file is system-wide and read-only.
- In the ko.UTF-8 locale, the /usr/lib/mle/ko.UTF-8/syshjd file is system-wide and read-only.
- In the ko and ko.UTF-8 locales, the $HOME/.usrhjd file can be manipulated by the user.
Users can change their locale settings with shell environment variables. Each category names an existing locale. The `setlocale()` function directly sets or queries the setting of these categories. Internationalized functions use these settings to access the appropriate tables for the desired locale.

Environment variables can indirectly set the categories: when `setlocale()` sets the categories to the default setting for that site, it uses the setting of each environment variable to set the associated categories. The `setlocale()` function does not change the settings of environment variables, it only reads their settings.

### Changing the Default Locale

You can change the default locale system-wide with the following procedure. For OpenWindows users, a default setting of `ko.UTF-8` will revert to the `C` locale.

1. **Edit the `/etc/default/init` file by adding or changing the line.**
   Substitute `C`, `ko`, or `ko.UTF-8` for `locale`.
   
   ```
   LANG=locale
   ```

2. **Have all users exit CDE.**

3. **Type the following commands:**
   ```
   % su
   # /usr/dt/bin/dtconfig -kill
   ```

4. **Type the following commands:**
   ```
   % su
   # reboot
   ```
Locale and Category Terminology

The terms locale and category relate to each other as follows:

- A locale includes specification of a language, territory, code set, and other features. The Korean Solaris operating environment includes three locales:
  - C—For the ASCII English environment, locale must be set to C.
  - ko—For the Korean environment in EUC, locale must be set to ko.
  - ko.UTF-8—For the Korean environment in UTF-8, locale must be set to ko.UTF-8.

- A category is a set of features that comprise a locale, such as character displays or time/date representations, whose behavior depends on the locale. Korean Solaris categories include the following:
  - LC_CTYPE sets the character-type for classification and conversion.
  - LC_TIME sets the locale for representation of date and time.
  - LC_NUMERIC sets the number representation locale (used also for input and output).
  - LC_MONETARY sets the currency representation locale.
  - LC_MESSAGES sets the language locale for messages to users.
  - LC_COLLATE sets the locale-dependent collation of strings.

The environmental variable LC_ALL explicitly sets the same locale for all categories; it has the highest priority. If categories or LC_ALL aren’t set, the LANG environmental variable will determine the category setting.

Keybinding for the Ṣṭṭ Input Server

In reference to the Korean window system input server, the keybinding process links certain keys on the keyboard with certain actions by an application. You can keybind a complex action by an application, such as closing an application’s open windows, to a single key or sequence of keys like Control-H or Esc w c.

All input conversion mode Control-key commands associated with non-ASCII input conversion are set in /usr/lib/mle/ko/keybind.dat and /usr/lib/mle/ko.UTF-8/keybind.dat, depending upon the locale setting. The default commands are listed in the table at the end of Chapter 4, “Entering Korean Input,” in Korean Solaris User’s Guide.
Customizing Keybinding Control Keys

Keybindings can be changed by changing the names of the keys in keybind.dat and restarting the htt input server. You can create and use your own customized keybind.dat file as follows:

1. Create a customized copy named keybind.dat in another directory.

2. Set the environment variable MLE_PATH to the path name of the directory containing this customized file.

3. Start the input server.
   The directory set in MLE_PATH is searched for an instance of the keybind.dat file and the commands in that file are set for the user.
   If MLE_PATH is not set or does not contain a usable keybind.dat file, /usr/lib/mle/ko/keybind.dat or /usr/lib/mle/ko.UTF-8/keybind.dat is used, according to the locale.
   To change the keybindings, edit the keybinding file to replace default keys with new keys. Key combinations and ON/OFF toggling can also be used.

Interfacing with the Korean Solaris Localization Facility

At the C-shell level, each environment variable can be set to locale (ko or ko.UTF-8 for Korean or C for ASCII) by a shell command as follows:

- **C-shell users can enter a shell command as follows:**
  
  system% setenv LC_TIME locale

- **Bourne shell (sh) users can use set or export:**
  
  $ set -a LC_TIME
  $ LC_TIME=locale

  or

  $ LC_TIME=locale
  $ export LC_TIME

Setting the locale to ko or ko.UTF-8 allows the user’s environment to display time in Korean format and text. A user can define a mix of locales for the working environment. For example, characters can be typed and converted in Korean, time can be displayed in French format and messages can appear in English.
Many users work in a single cultural environment. The LC_ALL and LANG environment variables set the system default for all categories. For example, these C-shell commands set the system default for all categories to locale.

```
system$ setenv LC_ALL locale
system$ setenv LANG locale
```

System administrators or users can set the default, and the setenv syntax can be used in programming.

This setting is put into effect the next time a setlocale() function call in an application program line sets a category to the default setting: `setlocale(LC_XXX "r")`
Setting Up Korean Solaris Printing Facilities

The Korean Solaris operating environment supports printing Korean output through the following types of printing facilities:

- Line printer containing built-in Korean fonts
- PostScript-based printer containing built-in scalable fonts
- Any PostScript-based printer for bitmap printing

The system administrator installs printer(s) as described in the printer product documentation. Users can then print Korean text using procedures described in this chapter.

Follow the printer documentation supplied by the manufacturer for physically connecting the printer. Then use the following instructions.

Line Printer Support

For the Korean Solaris operating environment to run a line printer, the printer must recognize at least one of the appropriate code sets:

- Completion code, also called Wansung (Korean EUC, based on KS C 5601)
- Combination code, also called Johap (either KS C 5601-1987-3 or KS C 5601-1992-3)
- N-byte code
Korean Solaris Code Filters

Printing an EUC (Wansung) File on a Printer that Does Not Support EUC

A printer that does not support EUC needs filters that convert EUC files for printing. For example, the following command sequence tells LP, the print service, that printer lp1 accepts only Packed (the KS C 5601-1987 version of Combination code) format files.

```
# lpadmin -p lp1 -v /dev/ttya -I PACK
# accept lp1
# enable lp1
```

The following command sequence tells LP that printer lp1 accepts only Johap (the KS C 5601-1992 version of Combination code) format files.

```
# lpadmin -p lp1 -v /dev/ttya -I JOHAP
# accept lp1
# enable lp1
```

The above command lines also install printer lp1 on port ttya. See the `lpadmin(1)` man page for more information.

An `lpfilter` command line such as the following can be used to print files whose formats are not supported by the printer:

```
# lpfilter -f filter-name -F pathname
```

The above command tells LP that a converter called `filter-name` (for example `comptopack`) is available through the filter description file named `pathname`. The content of `pathname` can be as follows:

- **Input types:** simple
- **Output types:** PACK
- **Command:** comptopack

The above filter takes default type file input and converts it to Packed format by using comptopack.

- **Input types:** simple
- **Output types:** JOHAP
- **Command:** wansungtojohap

The above filter takes the default type file input and converts it to Johap format using wansungtojohap.

To print an EUC file, use the following command:

```
system% lp euc-filename
```
To print a Packed format file, use the following command:

```
system% lp -T PACK PACK-filename
```

To print a Johap format file, use the following command:

```
system% lp -T JOHAP JOHAP-filename
```

### Printing a ko.UTF-8 File to Printers that Do Not Support It

The first line converts the file to an EUC file. The output will be missing any characters that are not defined in EUC.

```
system% iconv -f ko_KR-UTF-8 -t ko_KR-euc ko.UTF-8_filename >euc-filename
system% lp euc-filename
```

The first line converts the file to a Johap file.

```
system% iconv -f ko_KR-UTF-8 -t ko_KR-johap92 ko.UTF-8_filename >johap92-filename
system% lp johap92-filename
```

### Laser Printer Support

To print Korean characters using a PostScript-based printer, a Korean Solaris software application must have the Korean Solaris `xetops` utility to print EUC files or `xutops` to print files in the `ko.UTF-8` locale.

### Using `xetops` and `xutops` Utilities

The `xetops` and `xutops` utilities produce bitmapped graphics as printed images. Korean Solaris software includes the `xetops` and `xutops` utilities so any system can print Korean text on a PostScript printer.

- `xetops` handles files in the `ko` locale
- `xutops` handles files in the `ko.UTF-8` locale

Using `xetops` and `xutops` is described in *Korean Solaris User’s Guide*, in the chapter "Korean Printing Facilities," and in the `xetops(1)` and `xutops(1)` man pages.

A typical command line for printing a file named `filename` containing Korean characters with `xetops`, would be as follows:

```
```
The syntax for xutops is similar:

```
system$ pr filename | xutops | lp
```

Make *filename* the name of the file to print. This file can contain ASCII/English characters as well as Korean.
TTY Environment and Support

This chapter assumes you are familiar with how:
- The Solaris operating environment communicates with external devices using STREAMS and ioctl
- Different terminal types are supported by termcap and terminfo

Refer to the termio(7) man pages for background information on STREAMS and TTY drivers.

TTY Streams

The data path between a user’s shell and the terminal is called a stream. The data on a stream contain characters and control information that affect data handling, such as the control sequences that precede a change in code set or communication protocols. Data entering the stream from the terminal are raw or unprocessed. Data are sequentially processed by STREAMS modules for appropriate use by the shell or an application.

STREAMS provides a way to modularize the processing on a line, allowing processing instructions to be grouped in functional modules. These modules can be added or removed from the line so that different environments can be provided to a terminal according to the user’s needs.

Traditional STREAMS

The traditional STREAMS TTY environment contains a raw device driver, a line discipline module, and a stream head. The raw device driver provides an I/O
interface between the kernel and the hardware. Because it is closest to the physical hardware, it provides basic communication protocols, baud rate switching, and other low level services. The line discipline module is a set of instructions or disciplines that transforms the raw data to processed data. This includes handling the delete character, line kill character, and others. The stream head provides an interface between the user’s process and the stream.

Korean Solaris STREAMS

The Korean Solaris operating environment uses the modular nature of STREAMS to support Korean. In addition to the traditional TTY modules, this product implements code conversion in STREAMS. Hangul-Hanja conversion is typically supplied by many existing Korean TTYs and is not available in the Korean Solaris TTY environment.

The Korean Solaris operating environment enhances the traditional modules. Its line discipline handles proper cursor movement for wide characters as well as normal protocols. The Korean Solaris software code conversion modules convert between two different character code formats such as between Packed (also called Combination code of KS C 5601-1987) and EUC (called Completion code).

Code conversion depends on the appropriate flags or parameters being set. For example, if a Packed code terminal is being used, the input from the terminal is converted to EUC and the output to the terminal is converted to Packed code.

The major modules that can be pushed onto the stream are ldterm, kpack and kjohap:

- ldterm(7) is a generic EUC line discipline module. It processes all normal line discipline functions and also handles proper cursor movement and backspacing for wide characters (EUC).
- kpack controls code conversion between Combination code of KS C 5601-1987 and EUC.
- kjohap controls code conversion between Combination code of KS C 5601-1992 code and EUC.

TTY Utilities

ioctl (input/output control) calls are low-level routines for handling device input and output.
The *termcap* and *terminfo* databases are used by applications to configure their terminal display appropriately.

**EUC ioctl Features**

The Korean Solaris operating environment uses `ioctl(2)` STREAMS commands for general EUC handling. The following is a summary of these `ioctl` calls and their effects:

**TABLE 4-1 ioctl Requests and Descriptions**

<table>
<thead>
<tr>
<th>ioctl Request</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EUC_WGET</td>
<td>Get cswidth values from TTY stream</td>
</tr>
<tr>
<td>EUC_WSET</td>
<td>Set cswidth values for TTY stream</td>
</tr>
<tr>
<td>EUC_OXLOFF</td>
<td>Set code conversion to OFF</td>
</tr>
<tr>
<td>EUC_OXLON</td>
<td>Set code conversion to ON</td>
</tr>
</tbody>
</table>

Character code conversion to and from the terminal is controlled by `EUC_OXLON` and `EUC_OXLOFF`.

**termcap**

*termcap* and *terminfo* are the databases used to tailor the terminal characteristics for an application. The following are extensions to the *termcap* database:

**TABLE 4-2 termcap Variables and Descriptions**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dv</td>
<td>Device type: language and codeset</td>
</tr>
<tr>
<td>ci</td>
<td>Init sequence for multiple codesets</td>
</tr>
<tr>
<td>s0</td>
<td>Shift into codeset 0</td>
</tr>
<tr>
<td>s1</td>
<td>Shift into codeset 1</td>
</tr>
<tr>
<td>s2</td>
<td>Shift into codeset 2</td>
</tr>
<tr>
<td>s3</td>
<td>Shift into codeset 3</td>
</tr>
</tbody>
</table>
terminfo

The following are extensions to terminfo. The s0-s3 string values are used as data announcement mechanisms for the respective code sets during terminal I/O.

TABLE 4-3 terminfo Variables and Definitions

<table>
<thead>
<tr>
<th>Variable</th>
<th>Capname</th>
<th>Tc</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>device_type</td>
<td>devt</td>
<td>dv</td>
<td>Device type: language and codeset</td>
</tr>
<tr>
<td>code_set_init</td>
<td>csin</td>
<td>ci</td>
<td>Init sequence for multiple codesets</td>
</tr>
<tr>
<td>set0_des_set</td>
<td>s0ds</td>
<td>s0</td>
<td>Shift into codeset 0</td>
</tr>
<tr>
<td>set1_des_set</td>
<td>s1ds</td>
<td>s1</td>
<td>Shift into codeset 1</td>
</tr>
<tr>
<td>set2_des_set</td>
<td>s2ds</td>
<td>s2</td>
<td>Shift into codeset 2</td>
</tr>
<tr>
<td>set3_des_set</td>
<td>s3ds</td>
<td>s3</td>
<td>Shift into codeset 3</td>
</tr>
</tbody>
</table>

TTY Commands

The two commands for configuring and using the TTY environment are setterm and /bin/stty. setterm is used primarily to build the TTY stream for a particular terminal type, pushing the necessary modules onto the stream. stty changes the behavior of the modules in the stream.

setterm Command

setterm is used to configure the TTY STREAMS environment. It can inquire about and manipulate STREAMS modules for a particular TTY port. setterm allows users to tailor their TTY STREAMS environment using system-provided or user-provided STREAMS modules.

setterm uses a terminal device name that reflects the devt (device type) field in the terminfo database for configuring STREAMS modules for a TTY port. This device name is matched with an entry of the same name in the setterm configuration file, /usr/share/lib/setterm/ko/conf.file or /usr/share/lib/setterm/ko.UTF-8/conf.file. This entry contains detailed instructions on which modules to pop and push in order to properly configure the STREAMS environment.
setterm can also take the device type as a direct argument. This device type is similarly matched with an entry in /usr/share/lib/setterm/ko/conf.file or /usr/share/lib/setterm/ko.UTF-8/conf.file.

The setterm configuration file uses a special language for instructions on what actions to take. This language allows users to determine the names of modules on the STREAMS stack, to push or pop modules on the stack, and to do other operations. setterm manipulates the STREAMS stack by making ioctl calls.

For more information, see the setterm(1) man page.

/bin/stty Command

The defeucw option to the /bin/stty command is for modifying STREAMS modules to reflect changes in the user’s environment. It does not work with the /usr/ucb version of stty, which has not been internationalized.

The following command queries the user’s environment for information on EUC code-set width and sets that information in the line discipline:

system% /bin/stty defeucw

For example, if the user has the environment variable LC_CTYPE set to locale, this option gets information on the number of bytes per character and the screen width per character for the codesets in the ko and ko.UTF-8 environments and then sends this information to relevant modules in the stream.

TTY Setup Examples

The system administrator can add setterm in the startup script in /etc/rcn.d directory (where n is the run level), to run at the system boot time. Also, users can run the setterm command at login to configure the stream for their terminal, including the appropriate modules for Korean input code conversion. The following examples using setterm work as commands typed at a system prompt or included in system files such as .cshrc, .login, and the startup script. Such commands can either explicitly set the device type or use the terminfo database.
Configuring STREAMS for Korean Solaris Software

To explicitly configure the STREAMS module for EUC (Completion code) terminal type:

```bash
system% setterm -x EUC
```

For some more examples, consider using a Packed (Combination) code terminal, which uses either the Packed or Johap module.

To explicitly initialize the Packed STREAMS module, which supports a Combination code of KS C 5601-1987:

```bash
system% setterm -x PACK
```

To explicitly initialize the Johap STREAMS module, which supports a Combination code of KS C 5601-1992:

```bash
system% setterm -x JOHAP
```

This usage is independent of terminfo.

Further consider using a FAST-15 Packed code terminal on a system with an entry like the following (which is appropriate for such a terminal) in the terminfo database:

```bash
fast-15 | fast-pack | korean terminal packed mode, 
devt=PACK, 
use=vt100-w,
```

A configuring command that references this entry would be:

```bash
system% setterm -t fast-15
```

For `setterm` to work properly in this application, `/usr/share/lib/setterm/ko/conf.file` must contain an entry that corresponds to the device type. This entry gives `setterm` instructions for placing appropriate conversion modules in the TTY stream; for example:

```bash
# KoreanPACK|PACK throw \n  popto zs|mcp|mti|ptem \n  push kpack \n  push ldterm \n  push ttcompat \n  run {stty defeucw} \n  catch

# KoreanJOHAP|JOHAP throw \n  popto zs|mcp|mti|ptem \n  push kjohap \n  push ldterm \n  push ttcompat \n  run {stty defeucw} \n  catch
```
For more information, refer to the `setterm(1)` man page.

Configuring STREAMS for the `ko.UTF-8` Locale

To explicitly configure the STREAMS module for the `ko.UTF-8` locale:

```
system% setterm -x u8
```

To explicitly initialize the STREAMS module for an EUC terminal to use the `ko.UTF-8` locale:

```
system% setterm -x ku8euc
```

To explicitly initialize the Johap STREAMS module, which supports a Combination code of KS C 5601-1992 terminal, to use the `ko.UTF-8` locale:

```
system% setterm -x ku8johap
system% stty defeucw cs8 -istrip
```

For `setterm` to work properly in this application, `/usr/share/lib/setterm/ko.UTF-8/conf.file` must contain an entry that corresponds to the device type. This entry gives `setterm` instructions for placing appropriate conversion modules in the TTY stream; for example

```
# Korean specific entries:
#
# Completion/Wansung/EUC code terminal support (KS C 5601-1992)
KoreanU8EUC|KUEUCthrow \npopto zs|mcp|mti|ptem \npush ku8euc \npush eucu8 \npush ldterm \npush ttcompat \npush u8euc \nrun {stty defeucw} \ncatch
```
Terminal Support

The Korean Solaris operating environment supports Completion-code terminals and Packed-code (also called Combination-code, both KS C 5601-1987 and KS C 5601-1992) terminals. The terminals should have built-in automata and Hangul/Hanja fonts and input methods.

Installing a Terminal

If you have not added a terminal to your system before, first try installing a terminal in ASCII mode only. For more information, see System Administration Guide.

Serial Port Interface for Adding Terminals

Serial Ports is available from the Admintool menu to configure serial ports for terminals. Serial Ports provides the easiest method of installing a terminal. Serial Ports is invoked by admintool. For more information on admintool, see System Administration Guide.
Accessing Serial Ports

Using Serial Ports Menus
A Korean terminal that supports KS C 5601 is installed as you would install an ASCII terminal.

Command Line Interface for Adding Terminals

The following procedure is required to set up a terminal on ttya port via the command line:

1. **Determine the port monitor version number.**
The port monitor version number will display.

   ```
   # ttysadm -V
   ```

2. **Enter the following commands, substituting the port monitor version number for `ver`**.
   (For more information on sacadm (1M) and pmadm (1M) see their man pages.)

   ```
   # pmadm -r -p zsmon -s ttya
   # sacadm -a -p zsmon -t ttymon -c /usr/lib/saf/ttymon -v ver
   ```

3. **Use the pmadm command that matches your terminal type to add a login service:**
   For EUC terminals, use the following command:

   ```
   # pmadm -a -p zsmon -s ttya -i root -fu -v ver -m " 'ttysadm -S y \ -T terminal_type -d /dev/ttya -l 9600 -m ldterm,ttcompat -s \ /usr/bin/login'"
   ```
   For Korean Packed code terminals, which use the KS C 5601-1987 version of Combination code, type the STREAMS module `kpack` in the ttysadm command:

   ```
   # pmadm -a -p zsmon -s ttya -i root -fu -v ver -m "'ttysadm -S y \ -T terminal_type -d /dev/ttya -l 9600 -m kpack,ldterm,ttcompat -s \ /usr/bin/login'"
   ```
   For Korean Combination code terminals, which use the KS C 5601-1992 version of Combination code, type the STREAMS module `kjohap` in the ttysadm command:

   ```
   # pmadm -a -p zsmon -s ttya -i root -fu -v ver -m "'ttysadm -S y \ -T terminal_type -d /dev/ttya -l 9600 -m kjohap,ldterm,ttcompat -s \ /usr/bin/login'"
   ```

4. **Turn on the terminal.**
   Follow the documentation that accompanies the terminal.
5. Log in the terminal.

6. Check the correctness of the installation:

```
# setenv LANG ko
# /bin/stty cs8 -istrip defeucw
```

**Note** – These values show that the operating system is set to communicate with the terminal in “8-bit no-parity” mode. Make sure the terminal is set up in “8-bit no-parity” mode. Refer to the terminal’s setup manual for the proper way to set terminal options.

---

**Setting a User’s TTY**

To verify that your TTY is properly set up:

1. **Type the** `/bin/stty` **command with the** `-a` **option:**

   `system% /bin/stty -a`

   2. **If the values from above** `(cs8, -istrip)` **are not listed, then use the following command to set them:**

   `system% /bin/stty cs8 -istrip defeucw`

   This is the last step in setting up a terminal. The default setting of a Korean terminal is the Completion-code mode. For installing a Packed-code TTY, continue to the next section.

---

**Using Packed Code and Johap TTY in the ko Locale**

If you are using a Packed code type terminal, you must load the STREAMS module into the kernel.

1. **If you are using a Packed code terminal** *(KS C 5601-1987)*, you should load `kpack` by typing the following command as superuser:

   `system% su
   Password: (Type superuser password here if required.)
   # modload /kernel/strmod/kpack`

2. **Type the following command:**

   `system% setterm -x PACK`
Using Combination Code and Johap TTY in the ko Locale

1. If you are using a Combination code terminal (KS C 5601-1992), load kjohap by typing the following command as superuser:

   ```
   system% su
   Password: (Type superuser password here if required.)
   # modload /usr/kernel/strmod/kjohap
   ```

2. Type the following command:

   ```
   system% setterm -x JOHAP
   ```

Using EUC TTY in the ko. UTF-8 Locale

1. If you are using an EUC terminal (KS C 5601-1987-0), load the ku8euc module by typing the following command as superuser:

   ```
   system% su
   Password: (Type superuser password here if required.)
   # modload /kernel/strmod/ku8euc
   ```

2. To enable the stream module:

   ```
   system% setterm -x KU8EUC
   ```

3. To enable 8-bit I/O:

   ```
   system% stty defeucw cs8 -istrip
   ```

4. If you want to use dtterm, type the following command:

   ```
   system% setterm -x u8
   ```

Using Johap TTY in the ko. UTF-8 Locale

1. If you are using a Johap code terminal (KS C 5601-1992-3), load the ku8johap module by typing the following command as superuser:

   ```
   system% su
   Password: (Type superuser password here if required.)
   # modload /kernel/strmod/ku8johap
   ```

2. To enable the stream module:

   ```
   system% setterm -x KU8JOHAP
   ```
3. To enable 8-bit I/O:

   `system$ stty defucw cs8 -istrip`

4. If you want to use `dtterm`, type the following command:

   `system$ setterm -x u8`
OpenWindows Information

This appendix describes how to administer the Korean OpenWindows environment.

Starting Up OpenWindows

This section describes the steps required to set up the Korean environment and to start Korean Solaris operation.

Setting .cshrc for the Required Environment

Each user’s environment variables and ~/.cshrc (in other words, $HOME/.cshrc) file command lines must be set as described in this section to use Korean text. You must make sure the following three .cshrc file features (and consequently the users’ C shells) are set correctly before any user starts up the Korean OpenWindows environment.

The following three conditions are prerequisites to using the Korean OpenWindows environment:

- OPENWINHOME shell variable set to the path to /usr/openwin
- LANG shell-environment language-locale variable set for Korean
- TTY mode set appropriately for Korean character codes

To set these features, make sure each user’s .cshrc file includes the following lines:

```
setenv LANG ko
setenv OPENWINHOME /usr/openwin

set path=( /usr/SUNWale/bin $OPENWINHOME/bin $path )
```
... 

```bash
if ($?USER != 0 & & $?prompt != 0) then
  /bin/stty cs8 -istrip defeucw
endif 
```

Only `/bin/stty` can set the required features. Do not use `/usr/ucb/stty`, as it does not set all required features.

Also, make sure each `.cshrc` file puts `$OPENWINHOME/bin` in the user’s path before any other OpenWindows file. One way to ensure this is to put the following line in after other path assignments:

```bash
set path=/usr/SUNWale/bin $OPENWINHOME/bin $path
```

### htt Input Server and openwin-init Files

The `htt` input server must be running before any application that uses Korean input can run. It is started at OpenWindows startup from each user’s home directory `.openwin-init` file. This file must contain the line:

```bash
toolwait $OPENWINHOME/bin/htt
```

This line must be ahead of the lines that start Korean Solaris applications because they depend on the `htt` input server for Korean operation. If `.openwin-init` is missing from the home directory, `htt` is started from the `$OPENWINHOME/lib/locale/ko/openwin-init` file distributed with the Korean Solaris operating environment. The `htt(1)` man page explains more about the operation of `htt`.

### Setting the `.OWdefaults` File

The `.OWdefaults` file in the user’s home directory specifies the language used for several Solaris features: display language, numbers, time/date, messages, and other basic Korean OpenWindows properties. Some other entries in `.OWdefaults` affect the behavior and appearance of the user’s OpenWindows user interface.

Each user’s `.OWdefaults` file should contain the following five lines before running the Korean OpenWindows environment. So add these five lines at the end of users’ existing `.OWdefaults` files. (Refer to the “Using Localization on the Workspace Properties Worksheet” section in *Korean Solaris User’s Guide*.)

```bash
OpenWindows.BasicLocale: ko
OpenWindows.DisplayLang: ko
OpenWindows.InputLang: ko
OpenWindows.TimeFormat: ko
```
These fields can be set to ko, for Korean, or C, for English/ASCII operation. These five Korean OpenWindows variables have the following properties:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BasicLocale</td>
<td>Specifies the country (locale) of the user interface. With the basic locale set, a user can set other specific settings, such as input language.</td>
</tr>
<tr>
<td>DisplayLang</td>
<td>Specifies the language for labels, messages, menu items, help text, and other displays.</td>
</tr>
<tr>
<td>InputLang</td>
<td>Specifies the language used for keyboard input.</td>
</tr>
<tr>
<td>TimeFormat</td>
<td>Specifies the representation format of the time and date.</td>
</tr>
<tr>
<td>NumericFormat</td>
<td>Specifies the character system for number input/display.</td>
</tr>
</tbody>
</table>

These five fields can be added to an .OWdefaults file by using the localization category (Locale) in the Workspace Properties worksheet as described in the “Using Localization on the Workspace Properties Worksheet” section in Korean Solaris User’s Guide.

.xinitrc File

If you want to maintain your own .xinitrc, update it according to $OPENWINHOME/lib/Xinitrc.

Applications Defaults Files

Two directories for applications defaults are part of the Korean OpenWindows environment. One is for system-wide defaults, and one is specific to locale features:

The $OPENWINHOME/lib/app-defaults/C directory stores system-wide application defaults. These values are for the C locale.

The $OPENWINHOME/lib/locale/ko/app-defaults directory stores application defaults that are specific to the ko locale.

The $OPENWINHOME/lib/locale/ko/app-defaults/Http file has all htt resource default values that depend on the ko locale.

The $OPENWINHOME/lib/locale/ko/app-defaults/Olwm file lists all olwm window manager resources default values that depend on the ko locale. Such resource
file names have the locale name suffixed to the resource name, for example ButtonFont.ko. When a resource named with the locale suffix is lacking, the resource named without the suffix is used.

Customizing Mail Transmission and Storage

As you compose a mail message on the keyboard, the Korean Solaris Mail Tool sends the characters in EUC (8-bit) format to the workstation or TTY screen for display. But for transmitting the message across the network, receiving, and storing received messages, the tool has several standard transmission formats available.

Mail Transmission Formats

The initial default setting for mail transmission is the commonly used (7-bit) ISO 2022 encoding standard. This is the same as having the following command line in a user’s .mailrc file:

```
set encoding=ko_KR.iso2022-7
```

However, Mail Tool can instead transmit via N-byte, Johap (Packed or Combination), or EUC format as directed by one of the following command lines in a user’s ~/.mailrc file:

```
set encoding=ko_KR.nbyte
```

or:

```
set encoding=ko_KR.johap
```

or:

```
set encoding=ko_KR.euc
```

To set Mail Tool to use one of these transmission formats, perform the following steps:

1. **Put the encoding=ko_KR.format command line in your .mailrc file.**

2. **Follow the directions in the following section “Making .mailrc Changes Take Effect.”**
Mail Reception and Storage Formats

The Korean Solaris Mail Tool stores incoming mail in the /var/mail directory in whatever format the mail arrives.

Then, as Mail Tool reads the messages from the spool file and sends them to the screen display, it converts the text from the original ISO 2022, N-byte or Johap format, to EUC format, for display on the screen.

When you then direct Mail Tool to save a message, its initial default setting is to save the message in EUC format, regardless of the format originally received. This initial default setting is the same as having the following command line in the user’s .mailrc file:

```
set folderconv
```

To have mail stored in its originally received format (ISO2022-7, N-byte, Johap, or other formats) and not converted, use a command line, such as the following, in the user’s .mailrc file:

```
set nofolderconv
```

The following section explains how to make such command lines take effect.

Making .mailrc Changes Take Effect

Whenever Mail Tool is started up from the Workspace Programs menu, or a mailx session is started at a system prompt, the mail utility uses the settings in ~/.mailrc. After a Mail Tool or session is running, it ignores changes in the .mailrc file. Therefore, changes to .mailrc affect only Mail Tool or mail sessions that are started after the changes are saved.

However, changes made to .mailrc after a mailx session has been started can be put into effect in that existing mailx session using the source .mailrc command issued inside the session, as follows:

```
system% mailx
(Ongoing mailx session during which .mailrc is changed, for example from another Shell Tool window)
```

```
& source .mailrc
(Continuing mailx session during which new .mailrc settings are in effect.)
```

```
& q
system%
```

In the above example, the user types q to “quit” the mailx program.
Report of Incoming Mail

comsat is the server process that listens for reports of incoming mail and notifies users, who have requested notification, when mail arrives. To be able to this notification in Korean, the following steps should be taken:

File /etc/inet/inetd.conf contains the line:

comsat dgram udp wait root /usr/sbin/incomsat in.comsat

This line should be manually edited by superuser (root) or changed by running install_comsat.

comsat dgram udp wait root /usr/SUNWale/sbin/in.comsat in.comsat -l ko
Index

A
admintool, 28,
applications default files, 10, 35
/bin/stty command, 25, 30

C
category definition, 14
commands
/bin/stty, 30
ioctl, 23
pmadm, 29
setterm, 24
TTY, 24
configuring a terminal port, 28, 29
.cshrc, 33

E
EUC files, printing, 18
EUC ioctl command, 23

H
htt, 34

I
installing
a terminal, 28, 29
Korean Solaris, 10
ioctl command, 23

J
Johap TTY, 30

K
keybinding, customizing control keys, 15
ko locale, 10
ko.UTF-8 files, printing
ko_UTF-8 locale, 10

L
laser printer support, 19
locale
Korean, 15
overview, 10
terminology, 14
M
mail
  incoming reports, 38
  transmission formats, 36
Mail Tool, 36
.mailrc, 37

O
OpenWindows
  default environment, 15
openwin-init file, 34

P
Packed TTY, 30
pmaxm command, 29
printer support, 19
printing
  EUC files, 18
  ko.UTF-8 files,

S
Serial Ports, 28,
setlocale, 13
setterm command, 24
setting
  a user’s TTY, 30
storage formats, mail, 37
STREAMS
  Korean Solaris, 22
  traditional, 21

T
termcap database, 23
terminal port, configuring, 28, 29
terminal support, 28
termino database, 24
TTY
  commands, 24
  setting, 30

TTY (continued)
  setup examples, 25
  STREAMS, 21
  utilities, 22

X
X start-up file, 35
.Xdefaults, 34
xetops print filter, 19
.xinitrc, 35
xutops print filter, 19