



Sun Java™ System

Identity Manager 7.0

ワークフロー、フォーム、およびビュー

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054
U.S.A.

Part No: 820-1582

Copyright © 2006 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun Microsystems, Inc. は、この製品に含まれるテクノロジーに関する知的所有権を保持しています。特に限定されることなく、これらの知的所有権は <http://www.sun.com/patents> に記載されている 1 つ以上の米国特許および米国およびその他の国における 1 つ以上の追加特許または特許出願中のものが含まれている場合があります。

この製品は SUN MICROSYSTEMS, INC. の機密情報と企業秘密を含んでいます。SUN MICROSYSTEMS, INC. の書面による許諾を受けることなく、この製品を使用、開示、複製することは禁じられています。

U.S. Government Rights - Commercial software. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

本製品には、サードパーティーが開発した技術が含まれている場合があります。

本製品の一部は Berkeley BSD システムより派生したもので、カリフォルニア大学よりライセンスを受けています。UNIX は、X/Open Company, Ltd. が独占的にライセンスしている米国ならびにほかの国における登録商標です。

Sun、Sun Microsystems、Sun ロゴ、Java、Solaris、JDK、Java Naming and Directory Interface、JavaMail、JavaHelp、J2SE、iPlanet、Duke ロゴ、Java Coffee Cup ロゴ、Solaris ロゴ、SunTone Certified ロゴおよび Sun ONE のロゴマークは、米国およびその他の国における米国 Sun Microsystems, Inc. (以下、米国 Sun Microsystems 社とします) の商標もしくは登録商標です。

すべての SPARC 商標は、米国 SPARC International, Inc. のライセンスを受けて使用されている、米国およびその他の国における同社の商標または登録商標です。SPARC 商標が付いた製品は、米国 Sun Microsystems 社が開発したアーキテクチャに基づくものです。

Legato および Legato のロゴマークは Legato Systems, Inc. の商標であり、Legato NetWorker は同社の商標または登録商標です。Netscape Communications Corp のロゴマークは Netscape Communications Corporation の商標または登録商標です。

OPEN LOOK および Sun™ Graphical User Interface は、米国 Sun Microsystems 社が自社のユーザーおよびライセンス実施権者向けに開発しました。米国 Sun Microsystems 社は、コンピュータ産業用のビジュアルまたはグラフィカルユーザーインターフェースの概念の研究開発における米国 Xerox 社の先駆者としての成果を認めるものです。米国 Sun Microsystems 社は米国 Xerox 社から Xerox Graphical User Interface の非独占的ライセンスを取得しており、このライセンスは、OPEN LOOK GUI を実装するか、または米国 Sun Microsystems 社の書面によるライセンス契約に従う米国 Sun Microsystems 社のライセンス実施権者にも適用されます。

この製品は、米国の輸出規制に関する法規の適用および管理下であり、また、米国以外の国の輸出および輸入規制に関する法規の制限を受ける場合があります。核、ミサイル、生物化学兵器もしくは原子力船に関連した使用またはかかる使用者への提供は、直接的にも間接的にも、禁止されています。このソフトウェアを、米国の輸出禁止国へ輸出または再輸出すること、および米国輸出制限対象リスト (輸出が禁止されている個人リスト、特別に指定された国籍者リストを含む) に指定された、法人、または団体に輸出または再輸出することは一切禁止されています。

本書は、「現状のまま」をベースとして提供され、商品性、特定目的への適合性または第三者の権利の非侵害の黙示の保証を含みそれに限定されない、明示的であるか黙示的であるかを問わない、なんらの保証も行われぬものとします。

目次

はじめに	11
対象読者	11
内容の紹介	11
表記上の規則	12
表記上の規則	12
記号	13
シェルプロンプト	14
関連ドキュメントとヘルプ	14
オンライン上の Sun リソースへのアクセス	15
Sun テクニカルサポートへのお問い合わせ	15
関連するサードパーティー Web サイト	16
ご意見、ご要望の送付先	16
第 1 章 ワークフロー	17
この章の内容	17
関連する章	17
ワークフローについて	18
ワークフローとは	18
ワークフローコンポーネント	23
デフォルトのワークフロープロセス	24
遷移の作成	27
Identity Manager が使用するプロセスの更新	28
本稼動環境のワークフローの編集	28
標準ワークフロー	29
プロセスのカスタマイズ	29
ワークフローのデフォルトアクティビティ	30
ワークフロータスクのアクティビティ	34
ワークフローの進行状況の追跡	35
ワークフローの監査の有効化	37
概要	37

記録される情報と記録される場所	37
アプリケーションの追加	38
ワークフロープロパティの設定	39
Synchronize User Password ワークフロー	40
変数	41
アクティビティ	42
第2章 ワークフローサービス	45
関連する章	45
ワークフローの組み込み変数	46
一般的なセッションワークフローサービスの呼び出しの構造	47
サポートされるセッションワークフローサービス	48
addDeferredTask セッションワークフローサービス	49
audit セッションワークフローサービス	51
authorize セッションワークフローサービス	53
checkStringQualityPolicy セッションワークフローサービス	55
checkinObject セッションワークフローサービス	56
checkinView セッションワークフローサービス	56
checkoutObject セッションワークフローサービス	57
checkoutView セッションワークフローサービス	58
createView セッションワークフローサービス	60
disableUser セッションワークフローサービス	61
enableUser セッションワークフローサービス	62
getObject セッションワークフローサービス	64
getProperty セッションワークフローサービス	65
getResourceObject セッションワークフローサービス	65
getView セッションワークフローサービス	66
getViewForm セッションワークフローサービス	67
listResourceObjects セッションワークフローサービス	68
queryObjectNames セッションワークフローサービス	70
queryObjects セッションワークフローサービス	71
queryReferencingRoles セッションワークフローサービス	72
refreshView セッションワークフローサービス	72
removeDeferredTask セッションワークフローサービス	73
removeProperty セッションワークフローサービス	74
setProperty セッションワークフローサービス	75
unlockObject セッションワークフローサービス	75
unlockView セッションワークフローサービス	76
プロビジョンワークフローサービス	76
一般的なプロビジョンワークフローサービスの呼び出しの構造	77
サポートされるプロビジョンワークフローサービス	78
approveProvision プロビジョンワークフローサービス	79
auditNativeChangeToAccountAttributes プロビジョンワークフローサービス	80

bulkReprovision プロビジョンワークフローサービス	81
authenticateUserCredentials プロビジョンワークフローサービス	82
changeResourceAccountPassword プロビジョンワークフローサービス	82
cleanupResult プロビジョンワークフローサービス	83
checkDeProvision プロビジョンワークフローサービス	83
createResourceObject プロビジョンワークフローサービス	84
deleteResourceAccount プロビジョンワークフローサービス	85
deleteResourceObject プロビジョンワークフローサービス	85
deProvision プロビジョンワークフローサービス	86
deleteUser プロビジョンワークフローサービス	87
disable プロビジョンワークフローサービス	88
enable プロビジョンワークフローサービス	89
getApprovals プロビジョンワークフローサービス	90
lockOrUnlock プロビジョンワークフローサービス	90
notify プロビジョンワークフローサービス	91
provision プロビジョンワークフローサービス	91
questionLock プロビジョンワークフローサービス	92
reject プロビジョンワークフローサービス	92
reProvision プロビジョンワークフローサービス	94
runResourceAction プロビジョンワークフローサービス	94
updateResourceObject プロビジョンワークフローサービス	96
タイプ名	97
権限名	99
アクション名	100
第3章 Identity Manager フォーム	101
この章の内容	101
関連する章	101
フォームについて	102
フォームとは	102
フォームのカスタマイズ	114
カスタマイズの概要	115
カスタマイズに関連するその他のトピック	125
フォームの編集	162
Display 要素の操作	162
その他の Display 要素の操作	186
値の計算	207
フォームへのガイダンスヘルプの追加	210
フォームに関連するその他のタスク	216
デフォルトのユーザー作成フォームとユーザー編集フォームの代替フォーム	220
使用できるスケーラブルフォーム	222
Tabbed User Form のカスタマイズ: 属性エリアへのパスワードフィールドの移動	227
カスタマイズしたフォームの検証	230

フォームとフォームフィールドのサンプル	231
ユーザーフォームライブラリ	232
第 4 章 FormUtil メソッド	237
関連する章	237
メソッドの呼び出し	238
メソッド	238
callResourceMethod メソッド	238
buildDn メソッド	239
buildDns メソッド	240
戻り値	241
例	241
checkStringQualityPolicy メソッド	243
controlsAtLeastOneOrganization メソッド	244
getObject メソッド	245
getObjectNames メソッド	245
getOrganizationsDisplayNames メソッド	247
getResources メソッド	249
getResourceObjects メソッド	252
getRoles メソッド	255
getUnassignedApplications メソッド	256
getSubordinates メソッド	258
getUnassignedResources メソッド	260
getUsers メソッド	261
listResourceObjects メソッド	262
testObject メソッド	266
testUser メソッド	267
hasCapability メソッド	267
hasCapabilities メソッド	268
追加オプション	269
第 5 章 Identity Manager のビュー	283
この章の内容	283
関連する章	283
Identity Manager ビューについて	284
ビューとは	284
ビューハンドラとは	285
ビューとフォーム	285
ビューとワークフロー	286
代表的なビュー	286
ユーザービューについて	288
ユーザービューとフォームの統合	288

ユーザービューとワークフローの統合	289
汎用オブジェクトクラス	289
パス式	290
アカウントタイプとユーザー用ビュー	291
ユーザービューの属性	292
accounts 属性	304
deferred 属性	319
アカウント関連ビュー	322
correlation	322
管理者ロールビュー	325
ユーザーの秘密質問の回答変更ビュー	328
questions	328
loginInterface	329
ユーザー機能変更ビュー	330
adminRoles	330
capabilities	330
controlledOrganizations	330
委任承認者ビュー	331
name	331
delegateApproversTo	331
delegateApproversSelected	331
delegateApproversStartDate	332
delegateApproversEndDate	332
プロビジョニング解除ビュー	333
resourceAccounts	333
無効化ビュー	337
resourceAccounts	337
有効化ビュー	339
resourceAccounts	339
オブジェクト検索ビュー	341
objectType	341
allowedAttrs	342
attrsToGet	343
attrConditions	343
maxResults	345
results	345
sortColumn	345
selectEnable	346
組織ビュー	347
代表的な属性	347
ディレクトリジャンクションと仮想組織の属性	350
動的組織の属性	351
システム生成 ID に代わる組織パス名の使用	352

パスワードビュー	353
resourceAccounts	353
プロセスビュー	358
ビューオプション	360
checkinView メソッドの結果	360
調整ビュー	361
調整ポリシービュー	362
調整ポリシーと調整ポリシービュー	362
ビュー属性	363
調整状態ビュー	368
ユーザー名変更ビュー	370
再プロビジョニングビュー	373
resourceAccounts	373
ユーザーパスワードのリセットビュー	376
resourceAccounts	376
リソースビュー	379
リソースオブジェクトビュー	385
ロールビュー	387
タスクスケジュールビュー	390
scheduler	390
task	393
ロック解除ビュー	394
作業項目ビュー	397
すべてのアクティブ作業項目に関する情報の取得	397
作業項目リストビュー	404
ビューオプション	409
フォームでのビューオプションの設定	411
deferred 属性	411
deferred 属性を使用する状況	412
deferred 属性の使用	412
ビューの拡張	413
属性の登録	413
第 6 章 XPRESS 言語	417
この章の内容	417
XPRESS 言語について	417
プレフィックス表記	418
XML 構文と例	418
Identity Manager との統合	419
式を使用する理由	419
式の操作	420
フィールドの可視性の制御	420
フィールドのデフォルト値の計算	421

フィールド値の取得	423
フィールド値の生成	424
ワークフローの遷移条件	425
ワークフローアクション	426
ワークフローアクションからの Java メソッドの呼び出し	426
式の検証	427
関数	430
値コンストラクタ式	430
演算式	432
論理式	434
文字列操作式	442
リスト操作式	448
条件、繰り返し、およびブロックの式	457
変数と関数の定義式	462
オブジェクト操作式	466
Java および JavaScript 式	470
デバッグと検証の式	472
データ型	473
 第 7 章 XML オブジェクト言語	475
この章の内容	475
関連する章	475
XML オブジェクト言語について	476
例	476
XML オブジェクト言語とそれに対応する XPRESS	477
XPRESS での XML オブジェクトの使用	478
XPRESS の代わりに使用される XML オブジェクト言語	478
XML オブジェクト言語と XPRESS でのリストの表現	479
 第 8 章 HTML 表示コンポーネント	483
この章の内容	483
HTML 表示コンポーネント	483
HTML コンポーネントとは	484
表示コンポーネントの指定	484
HTML コンポーネントに関するページプロセッサの要件	484
コンポーネントクラス	485
基本コンポーネントクラス	485
コンテナクラス	486
コンポーネントサブクラス	492
表記上の規則	492
データ型	493
基本コンポーネントクラス	494

基本コンポーネント	501
フォームのマッピング	521
プロセスのマッピング	524
索引	527

はじめに

本書『Sun Java™ System Identity Manager ワークフロー、フォーム、およびビュー』では、Sun Java™ System Identity Manager を環境に合わせてカスタマイズするときに使用する参照情報と手順の概要について説明します。

対象読者

『Sun Java™ System Identity Manager ワークフロー、フォーム、およびビュー』は、製品配備のさまざまな段階で Identity Manager を顧客インストール用にカスタマイズするのに必要なワークフロー、画面、規則、システム設定、およびその他の設定ファイルを作成および更新するデブロイヤーおよび管理者に向けて作成されました。

デブロイヤーは、プログラミングに関する予備知識があり、XML、Java、Emacs や IDE (Eclipse または NetBeans など) に精通していることが望まれます。

内容の紹介

『Identity Manager ワークフロー、フォーム、およびビュー』は、次の章で構成されています。

- 第 1 章、「Identity Manager のワークフロー」 - Sun Java™ System Identity Manager (Identity Manager) のワークフローについて説明します。
- 第 2 章、「ワークフローサービス」 - カスタムワークフローを修正または作成する実装者が利用できる、Sun Java™ System Identity Manager のワークフローサービスについて説明します。
- 第 3 章、「Identity Manager のフォーム」 - Identity Manager の管理者向けおよびユーザー向けインタフェースの各種ページを定義するフォームをカスタマイズすることで、選択したページの外観や動作をカスタマイズする方法について説明します。

- 第 4 章、「FormUtil メソッド」 - もっともよく利用される FormUtil メソッドについて説明します。
- 第 5 章、「XPRESS 言語」 - Identity Manager 全体で使用される、式とスクリプトを記述するための XML ベースの言語、XPRESS の基本機能について紹介します。
- 第 6 章、「XML オブジェクト言語」 - string、list、map などの一般的な Java オブジェクトを表現できる XML 要素の集合、XML オブジェクト言語の基本機能について紹介します。
- 第 7 章、「Identity Manager のビュー」 - Identity Manager で使用されるデータ構造、Identity Manager ビューについて説明します。
- 第 8 章、「HTML 表示コンポーネント」 - Identity Manager の HTML 表示コンポーネントライブラリについて説明します。HTML 表示コンポーネントは、フォームのカスタマイズで使用されます。
- 付録 A、「フォームとプロセスのマッピング」 - Identity Manager で使用されるフォームおよびワークフロープロセスと、それに対応するシステム名の一覧を示します。

表記上の規則

この項の表は、本書で使用する次の表記規則について説明しています。

- 表記上の規則
- 記号
- シェルプロンプト

表記上の規則

次の表は、本書で使用する表記上の規則について説明しています。

表 1 表記上の規則

字体または記号	意味	例
AaBbCc123 (モノスペース)	API および言語の要素、HTML タグ、Web サイトの URL、コマンド名、ファイル名、ディレクトリパス名、画面出力の表示、サンプルコードを示します。	.login ファイルを編集します。 ls -a を使用してすべてのファイルを表示します。 % You have mail.

表 1 表記上の規則 (続き)

字体または記号	意味	例
AaBbCc123 (太字のモノスペース)	ユーザーが入力する文字を、画面上のコンピュータ出力とは区別して示します。	% su Password:
<i>AaBbCc123</i> (イタリック)	実際の名前または値によって置き換えられるコマンドまたはパス名の可変部分。	これらを、 <i>class</i> オプションと呼びます。 このファイルは、 <i>install-dir/bin</i> ディレクトリにあります。

記号

次の表は、本書で使用する記号の表記規則を示しています。

表 2 記号の表記規則

記号	説明	例	意味
[]	省略可能なコマンドオプションが入ります。	ls [-l]	-l オプションは省略可能です。
-	同時に押すキーを連結します。	Control-A	Ctrl キーと A キーを同時に押します。
+	連続して押すキーを連結します。	Ctrl+A+N	Ctrl キーを押し、離してから、以後のキーを続けて押します。
>	グラフィカルユーザーインタフェースで選択するメニュー項目を示します。	「ファイル」>「新規」>「テンプレート」	「ファイル」メニューから「新規」を選択します。「新規」サブメニューから、「テンプレート」を選択します。

シェルプロンプト

次の表は、本書で使用するシェルプロンプトを示しています。

表 3 シェルプロンプト

シェル	プロンプト
UNIX または Linux の C シェル	<i>machine-name%</i>
UNIX または Linux の C シェルのスーパーユーザー	<i>machine-name#</i>
UNIX または Linux の Bourne シェルおよび Korn シェル	\$
UNIX または Linux の Bourne シェルおよび Korn シェルのスーパーユーザー	#
Windows のコマンド行	C:¥

関連ドキュメントとヘルプ

Sun Microsystems は、Identity Manager をインストール、使用、および設定する際に役立つ次のような追加のドキュメントと情報を提供しています。

- 『Identity Manager インストール』: Identity Manager と関連ソフトウェアをインストールおよび設定する手順と参照情報が記載されています。
- 『Identity Manager Upgrade』: Identity Manager と関連ソフトウェアをアップグレードおよび設定する手順と参照情報が記載されています。
- 『Identity Manager 管理ガイド』: Identity Manager を使用して企業情報システムへのセキュリティ保護されたユーザーアクセスを実現するために、手順、チュートリアル、実例を説明します。
- 『Identity Manager 配備ツール』: Identity Manager のさまざまな配備ツールの使用方法を示す参照情報と手順が記載されています。これらの情報は、Identity Manager サーバーによって提供される規則と規則ライブラリ、共通のタスクとプロセス、辞書サポート、および SOAP ベースの Web サービスインタフェースを対象としています。
- 『Identity Manager の配備に関する技術概要』 – Identity Manager 製品の概念に関する概要 (オブジェクトアーキテクチャーを含む) および基本的な製品コンポーネントの紹介が記載されています。
- 『Identity Manager リソースリファレンス』: アカウント情報をリソースから Sun Java™ System Identity Manager に読み込んで同期する方法を示す参照情報と手順が記載されています。

- 『Identity Manager Tuning, Troubleshooting, and Error Messages』 : Sun Java™ System Identity Manager のチューニングに関するガイドンス、問題の追跡とトラブルシューティングの手順、およびこの製品を操作したときに発生する可能性があるエラーメッセージと例外についての説明を提供する参照情報と手順が記載されています。
- 『Identity Manager Service Provider Edition Deployment』 : Sun Java™ System Identity Manager Service Provider Edition の計画と実装の方法を示す参照情報と手順が記載されています。
- Identity Manager ヘルプ : Identity Manager の完全な手順、参照情報、用語の説明を記載したオンラインガイドンス、オンライン情報です。ヘルプにアクセスするには、メニューバーの「ヘルプ」リンクをクリックします。主要なフィールドには、ガイドンス (フィールド固有の情報) があります。

オンライン上の Sun リソースへのアクセス

製品のダウンロード、プロフェショナルサービス、パッチとサポート、および開発者向け追加情報については、次の Web サイトにアクセスしてください。

- ダウンロードセンター
<http://www.sun.com/software/download/>
- プロフェショナルサービス
<http://www.sun.com/service/sunps/sunone/index.html>
- Sun Enterprise サービス、Solaris パッチ、およびサポート
<http://sunsolve.sun.com/>
- 開発者向け情報
<http://developers.sun.com/prodtech/index.html>

Sun テクニカルサポートへのお問い合わせ

製品のドキュメントで解決できない、本製品に関する技術的な質問については、次のいずれかの方法でカスタマサポートにお問い合わせください。

- オンラインサポート Web サイト <http://www.sun.com/service/online/us>
- 保守契約に基づいて提供されるサポート電話番号

関連するサードパーティー Web サイト

Sun は、本書に記載されているサードパーティー Web サイトの利用について責任を負いません。Sun は、このようなサイトまたはリソースで得られるあらゆる内容、広告、製品、およびその他素材を保証するものではなく、責任または義務を負いません。Sun は、このようなサイトまたはリソースで得られるあらゆるコンテンツ、製品、またはサービスによって生じる、または生じたと主張される、または使用に関連して生じる、または信頼することによって生じる、いかなる損害または損失についても責任または義務を負いません。

ご意見、ご要望の送付先

Sun ではマニュアルの品質向上のため、お客様のご意見、ご要望をお受けしております。

コメントをお送りになる場合は、<http://docs.sun.com> にアクセスして「コメントの送信」をクリックしてください。オンラインフォームで、ドキュメントのタイトルと Part No. を入力します。Part No. は、マニュアルのタイトルページまたは最上部に記載されている 7 桁または 9 桁の番号です。

たとえば、本書のタイトルは『Sun Java™ System Identity Manager ワークフロー、フォーム、およびビュー』であり、Part No. は 820-1582 です。

ワークフロー

この章では、Sun Java™ System Identity Manager のワークフローについて説明します。

この章の内容

- [ワークフローについて](#)
- [デフォルトのワークフロープロセス](#)

関連する章

- 「[ワークフローサービス](#)」 - Identity Manager をカスタマイズする実装者が使用できるワークフローサービスのメソッドについて説明します。
- 「[XPRESS 言語](#)」 - ワークフローやフォームにロジックを含めるために XPRESS 言語で記述する式の一覧を示し、それぞれについて説明します。
- 『Identity Manager 配備ツール』に記載されている「Identity Manager 統合開発環境の概要」 - Identity Manager 統合開発環境 (IDE) と、ツールの起動、エディタオプションの設定、フォームの保存とデバッグの方法について説明します。

注	一般的なワークフローについてより深く理解するには、Identity Manager IDE を使用して、実際にサンプルワークフローの表示、実行、およびデバッグを行ってください。

ワークフローについて

Identity Manager ワークフローは、定義されている規則に従って一貫して実行される、一連のアクションとタスクを定義します。Identity Manager 統合開発環境 (IDE) のグラフィカルユーザーインターフェースを使用して、Identity Manager によって起動される各ワークフローをカスタマイズできます。

ワークフローを操作する前に、次の項目について理解してください。

- ワークフローの一般的な概念
- Identity Manager でワークフローがどのように使用されるか

ワークフローとは

一般的に、ワークフローは、論理的で反復可能なプロセスであり、ある関与者から別の関与者にドキュメント、情報、またはタスクが渡されると、一連の手順規則に従ってアクションが実行されます。この関与者は、人物またはマシンを表し、その両方を表す場合もあります。

Identity Manager では、この概念は、特に Identity Manager のワークフロー機能として実装されています。この機能は、ユーザーアカウントの作成、更新、有効化、無効化、および削除を管理する、複数のプロセス (ワークフロー) から構成されます。

実行するほとんどの Identity Manager タスクは、ワークフロープロセスのセットとして定義できます。Identity Manager にユーザーを作成するときは、たとえば、対応するワークフロープロセスが、次の処理を定義、実行します。

- パスワードポリシーを確認する
- 承認者に電子メールを送信する
- 各承認の結果を評価する
- ユーザーアカウントを作成する
- 監査レコードを作成する

これらのアクティビティは、Identity Manager 統合開発環境 を使用してカスタマイズできます。

通常、ワークフローはビューのチェックインに付随して呼び出されます。ビューのチェックは、フォームとビューを実装するページで「保存」をクリックしたときに行われます。

ワークフロープロセス

製品インタフェースのどの位置から呼び出すかに応じて、ワークフローはワークフロー、タスク、プロセス、またはタスク定義として参照されます。

タスク定義

Identity Manager の管理者インタフェースの「デバッグ」ページから、TaskDefinition のリストを検索し、ワークフロープロセスにアクセスします。「List Objects」ボタンの横にある「タイプ」メニューから「TaskDefinition」を選択し、「List Objects」をクリックします。アクセスできる TaskDefinition の一覧が表示されます。ワークフローのカスタマイズでは、そのワークフローの TaskDefinition オブジェクトを編集します。

TaskDefinition の起動インスタンスは、TaskInstance オブジェクトとして表されます。このオブジェクトも「デバッグ」ページから表示できます。

タスク定義パラメータ

次の表は、標準の設定パラメータを示しています。

表 1-1 ワークフローの標準設定パラメータ

パラメータ	説明
name	ユーザーが設定する、ワークフローの名前。Identity Manager のインタフェースには、この名前が表示されます。このタイプの他のオブジェクトと重複する名前は付けられませんが、タイプが異なれば、同じ名前を付けることができます。
taskType	フィルタリング専用に使われます
executor	タスクを実装するクラスの名前を識別します。デフォルトでは、ワークフローのこのパラメータのクラスは com.waveset.workflow.WorkflowExecutor です。
suspendedable	(ブール型) タスクを保留し、再開できることを表します。デフォルトは true です。
syncControlAllowed	(ブール型) 同期または非同期実行の要求をユーザーに許可するかどうかを表します。デフォルトは true です。
execMode	デフォルトで使用する実行タイプを指定します。デフォルトは sync です。 この値が NULL の場合、または ExecMode.DEFAULT に設定されている場合は、ExecMode.ASYNC として扱われます。

表 1-1 ワークフローの標準設定パラメータ (続き)

パラメータ	説明
executionLimit	タスクを実行できる制限時間を、秒単位で指定します。タスクには、実行の制限時間を指定できます。この制限時間が経過すると、スケジューラはそのタスクを終了できます。制限時間をゼロに設定すると、無制限として解釈されます。 デフォルトは 0 です。
resultLimit	タスクの完了後、インスタンスが存続できる制限時間を秒単位で指定します。デフォルトは 0 です。 タスクが完了または終了すると、タスクの結果を含む TaskInstance は、通常は指定した時間だけリポジトリに維持され、その時間が経過すると自動的に削除されます。 0 - TaskInstance は、タスクの完了後ただちに削除されます。 -1 - TaskInstance は自動的に削除されません。ただし、ユーザーが手動で削除することは可能です。 後から分析するレポートを生成するタスクでは、通常は、このパラメータに数日間に相当する値を設定します。副次的な結果を生じる目的のみで実行され、それ自体は意味のある結果を生じないタスクでは、ゼロに設定します。
resultOption	(String 型) 繰り返しタスクの以前の実行結果をどのように扱うかを示すオプションを指定します。このオブジェクトはそのデータを定義し、データの処理方法を確認します。デフォルトは delete です。 wait - 古い結果が手動で削除される、または有効期限が切れるまで、タスクが実行されないようにします。これがスケジュールされていないタスクである場合は、起動時にエラーとなります。スケジュールされているタスクであれば、スケジューラはこれを無視します。 delete - タスクを実行する前に、古い結果を自動的に削除します。古いタスクは完了状態でなければなりません。 rename - 古い結果の名前を変更してからタスクを実行します。古いタスクは完了状態でなければなりません。 terminate - 現在実行中のタスクを終了し、削除します。これは delete オプションに似ていますが、タスクが実行中の場合は、それを自動的に終了します。

表 1-1 ワークフローの標準設定パラメータ (続き)

パラメータ	説明
ayncExec	このパラメータを true に設定すると、アクションの完了後も、次の手動アクションまでワークフローの実行を継続し、ユーザーには次の作業項目を表示します。この設定は、ウィザード形式のワークフローをサポートします。 false に設定した場合は、ワークフローはバックグラウンドで実行を継続し、ユーザーがワークフローの次のステップを実行するときは、別のページ (通常は承認ページ) への移動が必要となります。
visibility	(String 型) このタスク定義の表示設定を宣言します。デフォルトは run schedule です。これ以外に、invisible、run task、および schedule task のオプションがあります。
progressInterval	Identity Manager が進行状況の変化を確認する間隔を、ミリ秒単位で指定します。 タスクには、タスクが進行状況を更新する間隔を指定できます。デフォルトは 5000 ミリ秒 (5 秒) です。間隔を短く設定するほど、より最新の状態を確認できますが、サーバーの負荷は増大します。

手動アクションについて

手動アクションは、手動によるやり取りを定義した、ワークフロープロセス定義の一部です。ワークフローエンジンが手動アクションを処理するときは、リポジトリに **WorkItem** オブジェクトが作成されます。作業項目に「完了」のマークが付いてからでないと、ワークフローは次に進むことができません。ほとんどの手動アクションは、承認の要求に使用されるため、作業項目の表は「承認」タブにあります。

ワークフローに属する手動アクションはどれも、リポジトリ内の **WorkItem** オブジェクトで表されます。

TaskInstance

ワークフロータスクが呼び出されると、ワークフローエンジンはリポジトリに **TaskInstance** を作成します。**TaskInstance** はリポジトリ内のオブジェクトの 1 つで、実行するワークフロープロセスの実行時状態を保持します。また、作成元となった **TaskDefinition** のコンテキスト変数と即時遷移情報を格納します。

TaskInstance は、TaskDefinition の生成 ID を使用して、説明的な TaskDefinition オブジェクトを参照します。TaskDefinition を編集すると、すでに実行中の TaskInstance は変更前の TaskDefinition オブジェクトを継続して使用しますが、新たに実行する TaskInstance は、新たに生成された ID を使用して修正後の TaskDefinition を使用します。

タスクインスタンスの削除のタイミング

TaskInstance の存続期間は、resultLimit パラメータによって決定されます。結果の有効期間の値にゼロが設定されている場合は、タスクは完了後ただちに削除されます。正の値が設定されている場合は、TaskInstance はその時間 (単位は分) だけ維持されます。

保留になっているワークフローの TaskInstance を削除するには、次の手順に従います。

1. Identity Manager の管理者インタフェースで、「サーバータスク」タブをクリックします。
2. 「すべてのタスク」を選択します。
3. 保留中の TaskInstance を選択し、「終了」をクリックします。

リポジトリ内のワークフロー

Identity Manager のリポジトリには、通常は、タイプが WFProcess の設定オブジェクトとしてワークフローが存在します (Create User ワークフローは、このオブジェクト定義の唯一の例外で、ProvisioningTask オブジェクトとして定義される)。taskType は常に Workflow です。

注	Identity Manager は、ワークフローの実行中のリポジトリをロックしません。これは、ワークフローが数日にわたって実行される可能性があり、その間中リポジトリをロック状態で維持することはできないためです。ただし、同じユーザーによる、別の更新ワークフローの呼び出しは、Identity Manager によって禁止されます。

ワークフローエンジン

ワークフローエンジンは、実行時のワークフロープロセスの実行を可能にするソフトウェアサービスです。ワークフロープロセスをサポートするワークフローエンジンの主な機能は次のとおりです。

- プロセス定義を解釈する
- プロセスインスタンスを作成し、その実行を管理する

- アクティビティを切り替え、そのアクティビティの処理のために作業項目を作成する

ワークフローコンポーネント

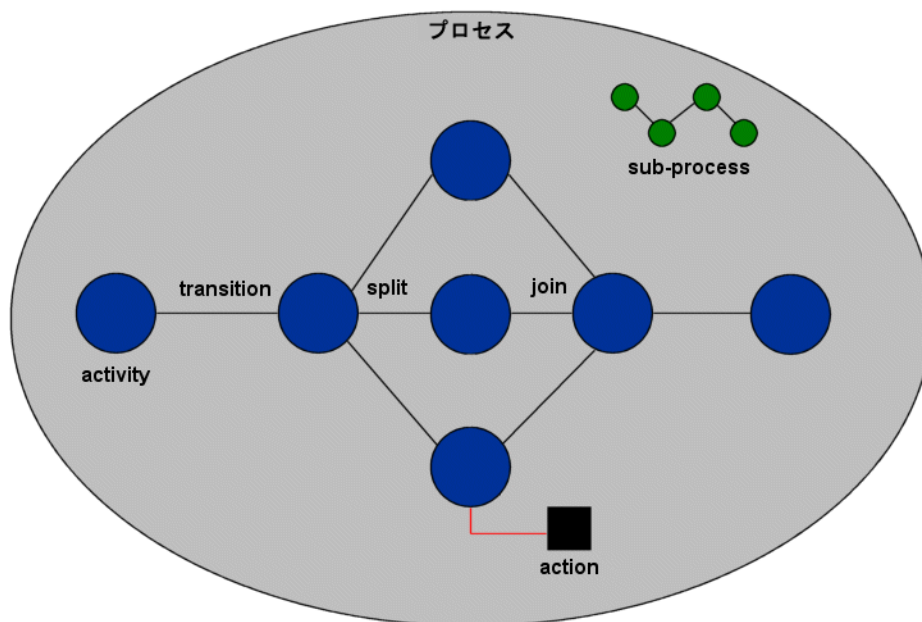
ワークフロープロセスは、目的を達成するまでに必要なステップを表します。プロセスが定義する内容は次のとおりです。

- ワークフローアクティビティ
- アクティビティ間の関係
- プロセスの開始、進行、および完了に必要な条件

各ワークフロープロセスは、次のコンポーネントのいずれか、またはその組み合わせによって定義されます。

- **Activity** - プロセスに含まれる 1 つの論理手順を表します。
- **Action** - アクティビティの実行方法を定義します。アクションは、簡単な式評価の場合もあれば、複雑な Java クラスの呼び出しの場合もあります。
- **Transition** - アクティビティから別のアクティビティへの移動を定義します。
- **Split** - 1 つのアクティビティから複数のアクティビティへの移動を定義します。Split は、さらに次のように定義されます。
 - **OR Split** - 各遷移パスをテストします。TRUE という値を持つ最初のパスが採用されます。
 - **AND Split** - 各遷移パスを採用します。
- **Join** - 複数のアクティビティから単一アクティビティへの移動を定義します。Join コンポーネントは、さらに次のように定義されます。
 - **OR Join** - 最初のパスが完了すると、次のアクティビティが開始されるように指定します。
 - **AND Join** - すべてのパスが完了してから次のアクティビティを開始するように指定します。
- **Subprocess** - プロセスに含まれる別のアクティビティから呼び出されるアクティビティ、アクション、および遷移のセットを定義します。

図 1-1 ワークフローの一般的なプロセスとコンポーネント



デフォルトのワークフロープロセス

Identity Manager IDE を使用することで、デフォルトの Identity Manager プロセスを編集し、一連のステップをカスタマイズできます。Identity Manager のワークフロー機能には、次のようなデフォルトプロセスのライブラリが含まれます。

- **ユーザーワークフロー** - ユーザーの作成、削除、更新、有効化、無効化、名前の変更など、Identity Manager ユーザーに関連するタスクのステップを定義します。
- **Identity Manager オブジェクトワークフロー** - リソース、リソースグループ、組織、組織単位など、Identity Manager のオブジェクトに関連するすべてのタスクのステップを定義します。たとえば、ロール管理やワークフロー管理などのワークフローは、単にビューの変更をリポジトリに適用するだけですが、承認や、その他のカスタマイズのきっかけを提供することもできます。
- **その他のワークフロー** - 調整、Remedy テンプレート、その他のカスタムタスクなど、Identity Manager の各種機能やオブジェクトのステップを定義します。

例

次のユーザー作成ワークフローは、タイムアウト時間に達した時点で、エスカレーションアクティビティを呼び出すように変更されています。タイムアウト時間に達するまでは、APPROVED 変数の結果が評価されます。評価の結果に基づいて、承認と拒否に対応したアクティビティのどちらに遷移するかが決定されます。

```
<Activity name='Wait'>
  <ManualAction name='approve' timeout='180'>
    <Owner name='${APPROVER}' />
    <Variable name='APPROVAL' value='false' />
    <Return from='APPROVAL' to='APPROVED' />
    <FormRef>
      <ObjectRef type='UserForm' id='#ID#UserForm:ApprovalForm' />
    </FormRef>
    <ReportTitle>
      <concat>
        <s>Awaiting approval from %n</s>
        <ref>APPROVER</ref>
      </concat>
    </ReportTitle>
  </ManualAction>
  <Transition to='Escalate'>
    <eq>
      <ref>WF_ACTION_TIMEOUT</ref>
      <s>true</s>
    </eq>
  </Transition>
  <Transition to='Approved'>
    <eq>
      <ref>APPROVED</ref>
      <s>true</s>
    </eq>
  </Transition>
```

```
<Transition to='Rejected' />
</Activity>
```

WorkItem のタイプ

手動アクションには、ワークフローエンジンによって手動アクションが実行された場合に生成される作業項目に、タイプを割り当てる機能があります。カスタマイズ内容に作業項目のタイプを割り当てることで、表示する作業項目や操作対象となる作業項目のセットをフィルタリングできます。

システムは、次のタイプの作業項目を認識します。

表 1-2 作業項目のタイプ

作業項目のタイプ	説明
承認	承認の作業項目であることを示します。
ウィザード	ユーザーによる任意操作の作業項目であることを表します。
保留	一時的な作業項目であることを表します。ワークフローにバックグラウンドでの実行を強制するときは、このタイプを使用します。

これ以外に、カスタマイズした作業項目のタイプを割り当てることもできます。たとえば、リソースの承認を表す作業項目タイプを resource に設定したり、ロールの承認を表す作業項目タイプを role に設定したりできます。

認証のタイプ

手動アクションには、作成する作業項目の認証タイプも指定できます。認証のタイプは作業項目のタイプとは異なり、現在の管理者が権限を持たない項目を排除できるように、クエリーに返される作業項目をシステムが自動的にフィルタリングするタイプです。通常は、承認者としての権限を持つどの管理者にも、担当する組織のすべての作業項目を表示する権限が割り当てられます。

手動アクションに作業項目の認証タイプを指定するには、次のように authType 属性を使用します。

```
<ManualAction authType='RoleApproval'>
```

作業項目タイプの割り当て

ManualAction 定義に項目タイプを指定するには、itemType 属性を設定します。次に例を示します。

```
<ManualAction itemType='approval'>
```

WorkItem の管理表示機能の制限

通常は、承認者としての権限を持つどの管理者にも、担当する組織のすべての作業項目を表示する権限が割り当てられます。管理者が組織の作業項目のサブセットのみを表示できるようにするときは、次の手順に従います。

1. WorkItem タイプを拡張する、新しい認証タイプを定義します。たとえば、RoleApproval タイプを定義します。
2. 作業項目自体ではなく、新しい認証タイプに対して権限を持つ、新しい機能を定義します。たとえば、RoleApproval タイプに対する権限を持つ、承認者ロールの機能を定義します。
3. 一般的な承認者の機能ではなく、管理者に対して、承認者ロールの機能を割り当てます。
4. ワークフローで使用される各手動アクションに、適切な認証タイプを設定します。

遷移の作成

遷移は、アクティビティが 1 つまたは複数の別アクティビティに移動するための規則を定義します。遷移には条件を設定できます。つまり、特定の条件が満たされる場合にのみ、遷移が行われるように設定できます。単純なアクティビティには、アクティビティを構成するアクションが完了するとただちに実行される、1 つの無条件遷移のみを含めることができます。

Identity Manager が使用するプロセスの更新

プロセスをカスタマイズするときは、ワークフロープロセスが想定どおりに正しく完了するように、変更内容を検証してから保存してください。保存したら、Identity Manager が使用できるように、変更したワークフローをインポートします。Identity Manager IDE デバッガを使用することもできます。Identity Manager IDE によるワークフロープロセスの編集については、「Identity Manager IDE の使用」を参照してください。

本稼働環境のワークフローの編集

本稼働環境のワークフロープロセスはカスタマイズしないでください。

元のワークフローのインスタンスの実行中にワークフローのアクティビティやアクションを編集すると、問題が生じることがあります。具体的には、TaskInstance には TaskDefinition ワークフローへの参照が含まれ、ID によって特定される現在のアクティビティまたはアクションが格納されています。これらの ID を変更すると、実行を再開したときに、想定されているとおりにタスクが再開されない可能性があります。

本稼働環境のワークフローの編集を回避できない場合は、次の手順に従います。これにより、古い定義を使用するタスクインスタンスによって保留されている作業項目の喪失を回避できます。

1. TaskDefinition の現在の名前を、タイムスタンプを付けて変更します。たとえば、Create User ワークフローを変更する場合は、TaskDefinition の名前を、Create User から Create User 20030701 に変更します。ワークフローの TaskDefinition の名前は、Identity Manager IDE を使用して変更できます。
2. 編集したワークフローを保存し、インポートします。

この手順に従うと、Identity Manager 内で保留状態にある既存の Create User タスクに問題が生じるのを回避できます。この場合、保留中のタスクで参照される、TaskDefinition の一意の ID は維持されます。

標準ワークフロー

Identity Manager には、使用されるプロセスにマップされた、標準のワークフローが最初から用意されています。これらのデフォルトワークフローの簡単な説明については、「[ワークフローのデフォルトアクティビティ](#)」を参照してください。デフォルトのワークフローを表示、編集するには、次の手順に従います。

1. Identity Manager IDE を起動します。Identity Manager IDE の使用方法については、『Identity Manager 配備ツール』に記載されている「Identity Manager IDE の概要」を参照してください。
2. 「ファイル」>「リポジトリオブジェクトを開く」>「ワークフロープロセス」を選択します。標準ワークフロープロセスと、配備に含まれるカスタムワークフローを示す「ワークフロープロセス」リストが表示されます。
3. 表示または編集するワークフローの名前をダブルクリックします。

Identity Manager の管理者インタフェースから「設定」>「フォームおよびプロセスマッピング」を選択すると、プロセスマッピングを表示できます。

プロセスのカスタマイズ

1 つまたは複数の Identity Manager のプロセスを変更することで、ステップを削除して新しいステップを設定したり、既存のステップをカスタマイズしたりできます。プロセスの各ステップは、アクティビティによって表されます。

ワークフローの編集時または作成時に利用できる定義済みアクティビティを提供するワークフローツールボックスは、ワークフローの変更に役立ちます。

このツールボックスを開くには、ダイアグラムビューを右クリックし、ツールボックスオプションを選択します。

ワークフローのデフォルトアクティビティ

次に、用意されているデフォルトアクティビティをカテゴリ別に示します。

表 1-3 ワークフローのデフォルトアクティビティ

アクティビティ	説明
Add Deferred Task	延期タスクのスキナ情報をオブジェクトに追加します。
Audit Object	監査ログレコードを作成します。
Authenticate User Credentials	
Authorize Object	リポジトリ内のオブジェクトに対して、対象となる認証をテストします。
Checkin Object	オブジェクトに変更を適用します。
Checkin View	更新されたビューを適用します。
Checkout Object	リポジトリのオブジェクトをロックし、編集のために取得します。 延期タスクのスキナ情報をオブジェクトに追加します。
Checkout View	更新可能なビューを取得します。
Create Resource Object	リソースオブジェクトを作成します。
Create View	新規ビューを初期化します。
Delete Resource Object	リソースオブジェクトを削除します。
Deprovision Primitive	リソースアカウントをプロビジョニング解除します。
Disable Primitive	リソースアカウントを無効にします。
Disable User	Identity Manager のユーザーアカウント、リソースアカウント、またはその両方を無効にします。
Email Notification	アクションを通知する電子メールを送信します。
Enable Primitive	リソースアカウントを有効にします。
Enable User	Identity Manager のユーザーアカウント、リソースアカウント、またはその両方を有効にします。
Get Object	リポジトリオブジェクトを取得します。
Get Property	プロパティを取得します。
Get View	読み取り専用ビューを取得します。

表 1-3 ワークフローのデフォルトアクティビティ (続き)

アクティビティ	説明
List Resource Objects	
Query Object Names	一致する属性を持つオブジェクトを検索します。
Query Objects	一致する属性を持つオブジェクトを検索します。
Query Reference	
Refresh View	以前にチェックアウトされたビューを更新します。
Remove Deferred Task	延期タスクのスキナ情報をオブジェクトから削除します。
Remove Property	オブジェクトの拡張プロパティを削除します。
Reprovision Primitive	リソースアカウントを再プロビジョニングします。
Run Resource Actions	
Set Property	拡張プロパティをオブジェクトに追加します。
Unlock Object	以前にチェックアウトされたオブジェクトのロックを解除します。
Unlock View	以前にチェックアウトされたビューのロックを解除します。
Update Resource Object	リソースによって管理されるオブジェクトを修正します。

表 1-4 Approval ワークフローのデフォルトアクティビティ

アクティビティ	説明
Approval	基本的な単一の承認者プロセスを実行します。
Approval Evaluator	複雑な承認プロセスを実装するために、承認定義オブジェクトの評価を繰り返します。 使用するフォームとテンプレートの受け渡しは許可されますが、別の設定が指定されている場合は、そちらが優先されます。
Lighthouse Approval	割り当てられている組織、ルール、およびリソースの、デフォルトの Identity Manager 承認プロセスを実行します。Approval Evaluator プロセスを使用します。
Multi Approval	複数の承認者に承認を配布します。各承認者の Approval プロセスを使用します。

表 1-4 Approval ワークフローのデフォルトアクティビティー (続き)

アクティビティー	説明
Notification Evaluator	複雑な通知プロセスを実装するために、承認定義オブジェクトの評価を繰り返します。通常は、Approval Evaluator に定義されている構造と同じ構造です。標準ワークフローでは、承認定義と通知定義は同じ構造にあります。カスタマイズされたワークフローでは、これは要件となりません。
Provisioning Notification	プロビジョニング処理が完了したあとに管理者に通知するための標準プロセス。

表 1-5 User ワークフローのデフォルトアクティビティー

アクティビティー	説明
DeProvision	既存の Identity Manager ユーザーをプロビジョニング解除するための標準ステップを実行します。リソースアカウントの削除、Identity Manager ユーザーの削除、リンク解除、および割り当て解除を詳細に制御できます。個々のリソース操作は、成功するまで再試行されます。
Provision	新しい Identity Manager ユーザーを作成し、リソースアカウントをプロビジョニングする標準ステップを実行します。個々のリソース操作は、成功するまで再試行されます。
Set Password	Identity Manager アカウントとリソースアカウントのパスワードを変更します。
Update User Object	WSUser オブジェクトをチェックアウトし、変更内容を適用してからチェックインします。
Update User View	ユーザービューをチェックアウトし、提供される一連の更新を適用してからチェックインします。
Update View	任意のビューに一連の変更を適用します。

表 1-6 End User ワークフローのデフォルトアクティビティ

アクティビティ	説明
End User Update Groups	マネージャーのいずれかのレポートに割り当てられるリソースのグループ割り当てを更新します (グループをサポートするリソースが対象)。
End User Update My Groups	ログインしているアカウントに割り当てられるリソースのグループ割り当てを更新します (グループをサポートするリソースが対象)。
End User Update Roles	マネージャーのいずれかのレポートのロール割り当てを更新します。
End User Update My Roles	ログインしているアカウントに割り当てられるロール割り当てを更新します。
End User Update Resources	マネージャーのいずれかのレポートの、リソース割り当てと、関連付けられている属性を更新します。
End User Update My Resources	ログインしているアカウントの、リソース割り当てと、関連付けられている属性を更新します。

表 1-7 Auditor ワークフローのデフォルトアクティビティ

アクティビティ	説明
Scan Notification	各アクセススキンの最後に、保留中のアテステーション作業項目があることをアテスターに通知します。保留中の作業項目の数に関係なく、各アテスターに 1 つの通知を送信します。また、スキャンに所有者が存在する場合は、スキャンの開始と完了をその所有者にも通知します。このワークフローは、次の入力をとります。 • <i>scanName</i> -- アクセススキンの名前 • <i>scanOwner</i> -- アクセススキンの所有者の名前 • <i>recipients</i> -- 通知先となる IDM ユーザーの名前リスト • <i>notificationType</i> -- 通知タイプ (begin、end、attest などのタイプが有効) • <i>userCount</i> -- スキャン対象ユーザーの数 (begin のみ)
Standard Attestation	指定された各アテスターのアテステーションサブプロセスを作成します。

表 1-7 Auditor ワークフローのデフォルトアクティビティー (続き)

アクティビティー	説明
Attestation	各アテスターの作業項目を作成し、すべての作業項目が承認済みの状態で完了した場合は、ユーザーのエンタイトルメントレコードに APPROVED のマークを付け、いずれかの作業項目が拒否された場合は、その時点でただちに REJECTED のマークを付けます。1 つの作業項目が拒否されると、その他すべての作業項目はキャンセルされます。
Launch Access Scan	Access Review タスクから得られる設定に基づいて、 Access Scan タスクを呼び出すか、またはスケジューリングします。このアクティビティーは、 Access Review ワークフロー / タスクから直接呼び出されます。

ワークフロータスクのアクティビティー

表 1-8

アクティビティー	説明
Add Result	名前を付けたデータ項目をタスク結果に追加します。
Add Result Error	タスク結果にエラーメッセージを追加します。
Add Result Message	タスク結果に情報メッセージを追加します。
Background Task	Identity Manager の管理者インタフェースから呼び出された場合に、ワークフローを強制的にバックグラウンドで実行します。
Get Resource Result	最後のプロビジョニング操作でリソースアダプタから返された結果オブジェクトを取得します。
Get Resource Result Item	最後のプロビジョニング操作でリソースアダプタから返された結果オブジェクトから、1 つの結果項目を取得します。
Rename Task	リポジトリ内のタスクインスタンスの名前を変更します。
Set Result	タスクに入力される結果にエントリを追加します。これは、ワークフローの概要レポートに記載されます。

表 1-8 (続き)

アクティビティー	説明
Set Result Limit	終了したタスクインスタンスをリポジトリ内で維持する時間を、秒単位で指定します。正の値は、タスクの完了後に、その秒数の間だけタスクインスタンスを維持することを表します。 負の値は、タスクインスタンスが自動的に削除されないことを表します。ただし、タスクを手動で削除することは可能です。

デフォルトの名前変更タスクの使用

カスタマイズを行わずにデフォルトの名前変更タスクを使用する場合は、ワークフローに次のアクションを指定します。

```
<Action process='Rename Task'>
    <Argument name='name' value=' 新しいタスク名 ' />
</Action>
```

ワークフローの進行状況の追跡

タスクに指定されている所有者は、ワークフロータスクの状態を常に確認できます。多くの場合、所有者はタスクを開始した人物ですが、所有者情報を定義し直すことができます。タスクはリポジトリ内のオブジェクトであるため、適切な権限を持つユーザーであれば、誰もがそれを表示できます。

通常、ワークフローの状態は **executing**、**pending**、**creating**、および **suspended** という文字列で「状態」列に表示されます。ワークフローの状態を要約した、より説明的な文字列をこの列の表示に追加できます。

この機能を実装するには、追加可能な 2 つの式のいずれかを WFProcess ファイルに追加します。

```
<WFProcess name='queryRoleTask' maxSteps='0'>
```

```
  <Status>
```

```
    <s>Customized Status</s>
```

```
  </Status>
```

```
  <Activity id='0' name='start'>
```

```
    <Transition to='GetReferencingRoles' />
```

```
  </Activity>
```

```
  <Activity name='GetReferencingRoles'>
```

```
    <Action id='0'>
```

```
      <expression>
```

<Status> には、文字列となる任意の XPRESS ステートメントを設定できます。次に例を示します。

```
<Status>
```

```
  <s> カスタム文字列 </s>
```

```
</Status>
```

または

```
<Status>
```

```
  <block>
```

```
    <s>not appearing</s>
```

```
    <s> カスタム文字列 </s>
```

```
  </block>
```

```
</Status>
```

```
....
```

この例の式の結果は、条件が当てはまる結果が保留中の場合に、「状態」列に表示されます (たとえば、**pending (カスタム文字列)**)。

ワークフローの監査の有効化

ワークフローの監査は、通常の監査と類似していますが、ワークフローの監査では、時刻を計算するための追加情報も記録されます。通常の監査では、イベントの発生をレポートしますが、イベントがいつ開始され、終了したのかは示されません。Identity Manager の監査の詳細については、『Identity Manager Audit Logging』を参照してください。

ワークフローの監査を有効にするには、ワークフロー、あるいは、1 つまたは複数のアクティビティに audit 属性を追加します。この属性を追加し、管理者インタフェースの適切なタスクテンプレートで「ワークフロー全体の監査」ボックスにチェックマークを付けることで、ワークフローの監査が有効になります。タスクテンプレートで監査を有効にする手順については、『Identity Manager 管理ガイド』の「タスクテンプレート」の章を参照してください。

概要

ワークフロー監査の処理では、事前に定義されている属性の名前とその値が、監査対象イベントごとに記録されます。ワークフロー内の監査を有効にするには、WFProcess 要素、あるいは、1 つまたは複数の Activity 要素に audit 属性 (値は true に設定) を追加します。WFProcess レベルで属性を設定すると、ワークフロー全体が監査対象となり、個々の Activity 要素に属性を追加した場合は、特定のアクティビティのみが監査対象となります。監査属性を設定しない場合、監査は無効になります。また、ワークフローを呼び出すタスクテンプレート内でも監査を有効にする必要があります。

次の例は、Notify アクションの監査を有効にします。

```
<Activity name='Notify' audit='true'>
```

記録される情報と記録される場所

デフォルトでは、ワークフローの監査は、通常の監査イベントで記録されるほとんどの情報が収集されます。これには、次の属性が含まれます。

- **WORKFLOW:** 実行対象ワークフローの名前
- **PROCESS:** 現在の実行対象プロセスの名前
- **INSTANCEID:** 実行対象ワークフローの一意のインスタンス ID
- **ACTIVITY:** イベントがログに記録されるアクティビティ
- **MATCH:** ワークフローインスタンス内の一意の識別子

- **ORGANIZATION:** ユーザーの組織の名前

これらの属性は、logattr テーブルに格納され、auditableAttributesList から取得されます。

Identity Manager は、workflowAuditAttrConds 属性が定義されているかどうかについても調べます。

プロセスまたはワークフローの 1 つのインスタンスの中で、特定のアクティビティを何度も呼び出すことができます。特定のアクティビティインスタンスの監査イベントを一致させるために、**Identity Manager** は、ワークフローインスタンス内の一意の識別子を logattr テーブルに記録します。

ワークフローの logattr テーブルに追加属性を記録するには、workflowAuditAttrConds リスト (GenericObjects のリストと見なされる) を定義する必要があります。workflowAuditAttrConds リストに attrName 属性を定義すると、**Identity Manager** は、コード内のオブジェクトから attrName を抽出します。まず、attrName をキーとして使用し、次に、attrName の値を記録します。すべてのキーと値は、大文字の値として記録されます。

アプリケーションの追加

Identity Manager IDE からアクセスできるように、独自に用意した Java メソッドを登録できます。次の手順で実行します。

1. idm/config/workflowregistry.xml ファイルを編集します。
2. 次の例のような形式で、アプリケーションの定義を追加します。

```
<WorkflowApplication name='Increment Counter'
  class='com.waveset.util.RandomGen' op='nextInt'>
  <ArgumentDefinition name='start' value='10'>
    <Comments>Get the next counter</Comments>
  </ArgumentDefinition>
</WorkflowApplication>
```

3. **Identity Manager IDE** を再起動します。

アプリケーションメニューに新しいアプリケーションが追加されます。

ワークフロープロパティの設定

ワークフローの設定プロパティは、System Configuration オブジェクトによって制御されます。次の表は、頻繁に使用される設定プロパティを示しています。

Identity Manager 統合開発環境を使用して System Configuration オブジェクトを表示または編集する方法については、『Identity Manager 配備ツール』の「Identity Manager 統合開発環境の概要」を参照してください。

表 1-9 システム設定オブジェクトのワークフロープロパティ

属性名	データ型	デフォルト値
consoleTrace	String	false
fileTrace	NULL	
maxSteps	String	5000
resultTrace	String	false
retainHistory	String	false
traceAllObjects	String	false
traceLevel	String	1
validationLevel	String	CRITICAL

consoleTrace

コンソールにトレースメッセージを送信するかどうかを指定します。**true** に設定した場合は、トレースメッセージが送信されます。デフォルトは **false** です。

fileTrace

名前が付けられたファイルにトレースメッセージを送信するかどうかを指定します。特定のファイルにトレースメッセージを送信するには、そのファイルの名前を入力します。

maxSteps

ワークフロープロセスまたはサブプロセスで許容される、最大ステップ数を指定します。このレベルを超過すると、その時点で Identity Manager はワークフローを終了します。この設定は、無限ループによるワークフローのスタックを検出するための安全装置として使用されます。デフォルトは 5000 ステップです。

resultTrace

タスクの結果オブジェクトにトレースメッセージを維持するかどうかを指定します。**true** に設定した場合は、タスクの結果オブジェクトにトレースメッセージが累積されます。

retainHistory

タスクの完了時に、履歴全体を返すかどうかを指定します。**true** に設定した場合は、Identity Manager は、ケース履歴全体を返します。これは、履歴を分析してプロセスの問題を診断する場合には便利ですが、結果全体のサイズが大きくなります。

traceAllObjects

汎用オブジェクトをワークフローのトレース処理の対象とするかどうかを指定します。

traceLevel

ワークフローのトレースに含める詳細度を指定します。値を指定しないか、または **0** を指定した場合は、もっとも詳細な情報がトレースされます。デフォルトは **1** です。

validationLevel

実行前にワークフローの検証に適用される厳密度を指定します。設定したレベル以上のエラーが検出された場合は、ワークフローの実行は開始されません。有効な値は、**CRITICAL**、**ERROR**、**WARNING**、または **NONE** です。**NONE** を指定した場合は、検証を完全に無効にします。

デフォルトは **CRITICAL** です。

Synchronize User Password ワークフロー

ユーザーパスワードを同期させるために、PasswordSync アプリケーションによって使用されます。JMS リスナーアダプタによって呼び出されるこのワークフローは、アダプタが送信する個々の要求を受け取り、「ユーザーパスワードの変更」ビューをチェックアウトして後ほどチェックインします。チェックインを完了すると、このワークフローは、デフォルトで、チェックインの成功または失敗を示す確認の電子メールをユーザーに送信します。

変数

Synchronize User Password ワークフローには、次の変数があります。

IDMAccountId

メッセージマッパーによって解決される Identity Manager アカウント ID を指定します。NULL は、ユーザーが Identity Manager に見つからなかったことを示します。

password

すべてのリソースと同期させる必要があるパスワードを識別します。

sourceResource

Identity Manager にリソースアカウントが見つかったリソースの名前を識別します。

userEmail

ユーザーの電子メールアドレスを指定します。この値は、Active Directory から取得されます。NULL の場合は、Identity Manager に記録されている電子メールアドレスです。

PasswordSyncThreshold

同期を開始するまでの秒数を指定します。この設定により、Identity Manager のユーザーインタフェースで更新されたパスワードがソースリソースに送信されても、ユーザーパスワードの同期は繰り返されません。

lastSourcePasswordUpdate

Identity Manager がソースリソースのパスワードを最後に更新してから経過した時間を、ミリ秒単位で示します。この設定により、Identity Manager のユーザーインタフェースでリセットしたパスワードがリソースに送信されても、ユーザーパスワードの同期が再度繰り返されることはありません。

lastUpdateSecs

パスワードが更新されてから経過した時間を、秒単位で示します。

PasswordSyncResourceExcludeList

常に同期対象から除外されるべきリソースの名を一覧表示します。この属性を設定するには、System Configuration オブジェクトの PasswordSyncResourceExcludeList 属性を編集します。

resourceAccountId

ソースリソースから受け取るネイティブ `accountId` を指定します。

アクティビティ

checkForIDMAccountID

IDMAccountId が `NULL` であるかどうかを調べます。 `NULL` である場合は、 `noIDMAccountForResourceAccount` という空のアクティビティに遷移します。 `NULL` 以外のアカウントが存在する場合は、ワークフローは `checkLastUpdateThreshold` に遷移します。

noIDMAccountForResourceAccount

デフォルトでは何も行いませんが、受け取ったリソースアカウント ID や GUID をメッセージマッパーが **Identity Manager** のユーザー名に解決しなかった場合に呼び出されます。

checkLastUpdateThreshold

`lastUpdateSecs` 変数の値が `PasswordSyncThreshold` の値未満であることを確認します。このアクティビティは、**Identity Manager** 製品でパスワードが変更された場合に、パスワードのリセットが繰り返されることを防ぎます。たとえば、ユーザーがリソース (**Active Directory** を含む) のパスワードを変更します。この後、パスワード通知ツールは、パスワードが変更されたことを **Identity Manager** に伝えます。この確認を行わない場合、**Identity Manager** はパスワードの変更を余分に実行することになります。 `passwordThreshold` の値を超過した場合は、`checkoutView` アクティビティは継続されます。

CheckoutView

指定された **Identity Manager** アカウントの `ChangeUserPassword` をチェックアウトします。チェックアウトに成功したときは、ワークフローは `SetPasswordView` に遷移します。失敗した場合は、`NotifyFailure` に遷移します。

SetPasswordView

リソースアカウントに対して繰り返され、ソースリソースでもなく、排除リストにも含まれないすべてのリソースを選択します。それ以外のすべてのリソースは選択され、直後のパスワード変更が不要であることを示すために、`expiredPassword` フラグを `false` に設定します。このアクティビティは常に `CheckTargets` に遷移します。

CheckTargets

同期の対象が、ほかに少なくとも 1 つ残されていることを確認します。**Identity Manager** がその他のリソースアカウントを検出しない場合は、ワークフローは **UnlockView** に遷移します。

UnlockView

現在チェックアウトされたビューのロックを解除し、終了に遷移します。

CheckinView

NotifyFailure への失敗通知遷移がある場合に、ビューをチェックインします。それ以外の場合は、**Notify** に遷移します。

Notify

PasswordSyncNotice 電子メールテンプレートを使用して、同期に成功したことをエンドユーザーに通知します。このアクティビティは常に終了に遷移します。

NotifyFailure

PasswordSyncNotice 電子メールテンプレートを使用して、同期に失敗したことをエンドユーザーに通知します。このアクティビティは常に終了に遷移します。

ワークフローサービス

この章では、カスタムワークフローを修正または作成している Identity Manager の実装者が利用できる、Identity Manager のワークフローサービスについて説明します。

Identity Manager には、ユーザーアカウントのプロビジョニングと操作を目的とするプロセスを定義するために、デフォルトのワークフローが用意されています。実装時にこれらのワークフローを変更し、独自のビジネスルールを反映させることができます。ワークフローを使用することで、アカウントプロビジョニングに関する独自のビジネスルールを Identity Manager に実装できます。

関連する章

- 「[ワークフロー](#)」 - Identity Manager のワークフローの概要を説明します。
- 「[Identity Manager のビュー](#)」 - ユーザービューと呼ばれる Identity Manager の内部データ構造と連携する Identity Manager フォームについて説明します。フォームをカスタマイズするときは、ビューの属性を呼び出します。

ワークフローの組み込み変数

ワークフローエンジンは、いくつかの組み込み変数を使用します。これらの変数のほとんどは、ワークフロー内で宣言する必要はありません。組み込み変数は、ワークフローの実行状態の確認に使用できます。また、多くの変数は、ワークフローサービスの影響を受けて設定されます。

表 2-1 ワークフローの組み込み変数

名前	説明
WF_ACTION_ERROR	直前に実行されたアクションが、エラーを含む結果を返したり、例外をスローしたりした場合は、組み込み変数は true に設定されます。
WF_ACTION_RESULT	この組み込み変数には、直前のアクションから返される WavesetResult オブジェクトが設定されます。この変数は、アクションの WavesetResult を取り込み、それをグローバル TaskInstance の結果に追加せずに処理したい場合に使用します。この変数はリソースのリトライをサポートするために追加されたのが始まりです。リトライのたびにタスク結果にリソースのエラーメッセージを追加することは必ずしも望ましくはないためです。あまり頻繁に使用される変数ではありませんが、アクションの結果をタスク結果に追加する前に調整する場合には便利です。
WF_ACTION_SUPPRESSED	この組み込み変数は、<Condition> 式の評価が false であるためにアクションが実行されない場合に true に設定されます。
WF_ACTION_TIMEOUT	この組み込み変数は、直前に実行されたアクションがタイムアウトになった場合に true に設定されます。
WF_CASE_OWNER	この組み込み変数には、ワークフロータスクを呼び出した管理者の名前が設定されます。
WF_CASE_RESULT	この組み込み変数には、 TaskInstance の WavesetResult が設定されます。これは XPRESS または JavaScript で実装されるアクションにおいて結果を取得するために使用されます。 WorkflowApplication クラスとして実装されるアクションは WorkflowContext を経由して結果を取り込むことも可能です。 WorkflowContext の全体は WF_CONTEXT 変数を使って参照できるため、この変数は絶対に必要であるとは言えませんが、便利ではありません。

表 2-1 ワークフローの組み込み変数 (続き)

名前	説明
WF_CONTEXT	この組み込み変数には、WorkflowContext オブジェクトが設定されます。これは XPRESS または JavaScript で実装されるアクションにおいて結果を取得するために使用されます。WorkflowApplication クラスとして実装されるアクションでは、コンテキストは排除されます。

一般的なセッションワークフローサービスの呼び出しの構造

ワークフローサービスは、ワークフローアクションから呼び出されます。次のコード例は、セッションワークフローサービスアクションの一般的な形式を示しています。

コード例 2-1

```
<Action class='com.waveset.session.WorkflowServices'>
  <Condition/>
  <Argument name='op' value=workflowServiceOp/>
  <Argument name=argname1>
    <expression>value1expression</expression>
  </Argument>
  <Argument name=argname2>
    <expression>value2expression</expression>
  </Argument>
  ...
  <Argument name=argnameN>
    <expression>valueNexpression</expression>
  </Argument>
</Action>
```

サポートされる各ワークフローサービスには、それぞれ異なる数の、必須および省略可能な引数があります。

サポートされるセッションワークフローサービス

現在、Identity Manager でサポートされるセッションワークフローサービスは次のとおりです。セッションワークフローサービスの呼び出しに指定する *op* 引数は、次のいずれかの値である必要があります。

- addDeferredTask
- audit
- authenticateUserCredentials
- authorize
- checkinObject
- checkinView
- checkoutObject
- checkoutView
- checkStringQualityPolicy
- createView
- disableUser
- enableUser
- findUser
- getObject
- getProperty
- getResource
- getView
- getViewForm
- listResourceObjects
- queryObjectNames
- queryObjects
- queryReferencingRoles
- refreshView
- removeDeferredTask
- removeProperty
- setProperty

- `unlockObject`
- `unlockView`

`op` 引数に上記以外の値を指定した場合、ワークフローサービスは次の結果を返します。

```
'Unknown WorkflowServices op'
```

また、ワークフローコンテキスト変数 `WF_ACTION_ERROR` は、`NULL` 以外の値に設定されます。

addDeferredTask セッションワークフローサービス

Deferred Task Scanner タスクによって認識されるプロパティを設定するために使用されます。通常、**Deferred Task Scanner** は、特定のプロパティを見つけるために、**WSUser** オブジェクトごとに繰り返し処理を行います。

この方法でユーザーに延期タスクを設定するときは、`taskDefinition` 引数を使用して任意の値を設定できます。この引数は、任意の値を持つオブジェクトである必要があります。これらの値は、延期タスクが開始されるときに実行されるワークフローに渡されます。

同一タスク定義の複数バージョンの追加

このメソッドによって、同じタスク定義の複数のバージョンをユーザーに追加することができます。

`taskDefinition` 属性の値として追加引数を渡すことができるのは、`taskDefinition` が汎用オブジェクトとして定義されている場合に限られます。`taskDefinition` を `map` または `List` として定義することはできません。

注	<code>date</code> や <code>description</code> などの組み込み引数は、呼び出されたワークフローですでに使用できる状態になっています。

表 2-2 AddDeferredTask メソッドの引数

名前	必要性	有効値	説明
<i>type</i>	なし	タイプのリスト	延期タスクが追加されるオブジェクトのタイプを指定します。指定しない場合は、デフォルトのタイプである <i>user</i> と見なされます。
<i>name</i>	あり		延期タスクが追加されるオブジェクトの名前を指定します。
<i>task</i>	なし		実行する <i>TaskDefinition</i> の名前を指定します。 <i>taskDefinition</i> 引数が文字列である場合は、この引数は、プロパティリストに配置される汎用オブジェクトの名前となります。
<i>taskDefinition</i>	なし		タスク定義を指定します。文字列または汎用オブジェクトのいずれかを指定できます。 文字列 -- この引数が文字列である場合は、タスク定義またはタスクテンプレートのいずれかの名前を定義しています。 <i>task</i> 引数は、プロパティリストに配置される汎用オブジェクトの名前です。 汎用オブジェクト -- <i>taskDefinition</i> 引数が汎用オブジェクトである場合は、この引数は、事前に構築されているプロパティ、 <i>listObject</i> です。
<i>instanceName</i>	なし		作成する <i>TaskInstance</i> の名前を指定します。
<i>date</i>	なし	java.util.Date オブジェクト、または通常形式の文字列のいずれかです。	延期タスクを実行する日付を指定します。
<i>organization</i>	なし		<i>TaskInstance</i> を割り当てる組織を指定します。

表 2-2 AddDeferredTask メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>owner</i>	なし		<i>TaskInstance</i> の有効な所有者の名前を指定します (後の管理で使用される)。
<i>description</i>	なし		タスク管理ページに表示するために <i>TaskInstance</i> に割り当てるオプションの説明文を指定します。
<i>executeOnce</i>	なし		true に設定した場合、延期タスクの実行が 1 回でも成功すれば次の実行は行われません。 タスクの実行時にエラーが発生しても、タスクが 1 回成功する、または <i>removeDeferredTask</i> メソッドの呼び出しによって手動で削除されるまで、タスクは延期タスクスキャナによって実行され続けます。

audit セッションワークフローサービス

Identity Manager の監査サービスを使用して監査イベントを記録するように、Identity Manager に要求します。監査は、ワークフローを呼び出したワークフロー管理者 (WF_CASE_OWNER) の下に記録されます。

表 2-3 audit メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	audit	
<i>type</i>	あり	「タイプ名」を参照	監査対象オブジェクトの Identity Manager タイプ。

表 2-3 audit メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>logResultErrors</i>	なし	true/false	false の場合、このメソッドは、別のパラメータによって示される 1 つのエラーを記録します。true の場合は、タスク結果に含まれる各エラーメッセージのログレコードを作成します。タスクの結果は、ワークフローコンテキストから取得されます。これは、ワークフローの実行中に発生するすべてのエラーを記録する場合に便利です。
<i>name</i>	なし		監査対象のオブジェクトの名前を指定します。
<i>status</i>	なし	success failure	監査の状態を指定します。状態が伝達されない場合は、失敗と見なされます。
<i>reason</i>	なし		現時点では、 <code>DATABASE_ACCESS_FAILED</code> にハードコードされています (指定したすべての値は無視される)。
<i>action</i>	あり	「アクション名」を参照	監査対象となる Identity Manager アクションを指定します。
<i>resource</i>	なし		監査対象のリソースを指定します。
<i>accountId</i>	なし		監査対象の <i>accountId</i> を指定します。
<i>error</i>	なし		監査対象のエラーを指定します。
<i>parameters</i>	なし		名前と値のペアを持つ <code>java.util.Map</code> オブジェクトである必要があります。このペアは、各イベントに固有です。たとえば、 <i>RoleName</i> というパラメータは、通常はユーザーに関連するログレコードで設定されます。

表 2-3 audit メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>attributes</i>	なし		名前と値のペアを持つ <code>java.util.Map</code> オブジェクトである必要があります。これは、監査システムによって、 <code>name1=value1;; name2=value2;;</code> という形式でログレコードに記録されます。 通常は、特定のアカウント属性に対する変更の記録に使用されます。
<i>organizations</i>	なし		監査対象オブジェクトに関連付けられている組織のリスト (<code>ObjectGroups</code>) を指定します。

戻り値と副作用

指定された情報を持つ監査ログが追加されます。

authorize セッションワークフローサービス

ワークフローの対象が、指定したオブジェクトに対して指定したアクションを実行する、指定した権限を持つかどうかを調べるように、**Identity Manager** に要求します。ワークフローの対象は、ワークフローを実行している管理者です。これは、組み込み変数 `WF_CASE_OWNER` の値です。

表 2-4 authorize メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	authorize	
<i>type</i>	あり	「タイプ名」を参照	認可対象のオブジェクトのタイプを指定します。
<i>name</i>	あり		認可対象のオブジェクトの名前を指定します。
<i>right</i>	あり	「権限名」を参照	認可する権限を指定します。

表 2-4 authorize メソッドの引数

名前	必要性	有効値	説明
<i>action</i>	あり	「アクション名」を参照	認可するアクションを指定します。

戻り値と副作用

指定したオブジェクトに対して、指定した権限またはアクションが `WF_CASE_OWNER` に認められない場合、`WF_ACTION_ERROR` は `true` となります。
`WavesetResult` にエラーが格納されます。

checkStringQualityPolicy セッションワークフローサービス

指定したポリシーに対して、文字列の品質を確認します。

表 2-5 checkStringQualityPolicy メソッドの引数

名前	必要性	有効値	説明
<i>policy</i>	あり		ポリシーを指定します (String)。
<i>map</i>	なし		文字列に含まれてはならないデータのマップを提供します (Map)。 <i>returnNull</i> -- (省略可能) true に設定すると、このメソッドは成功時に NULL オブジェクトを返します。
<i>value</i>	あり		調査する文字列の値を指定します (Object)。
<i>pwdhistory</i>	なし		ユーザーの過去のパスワードを、大文字の、暗号化された形式で一覧表示します。
<i>owner</i>	あり		調査対象となる文字列値のユーザーを指定します。

戻り値と副作用

このメソッドは、checkPolicyResult オブジェクトを返します。true という値は、文字列がポリシーのテストに合格したことを示します。文字列がポリシーテストに合格しなかった場合、このメソッドはエラーメッセージを返します。map パラメータの *returnNull* オプションを **true** に設定した場合は、成功時に NULL オブジェクトが返されます。

checkinObject セッションワークフローサービス

指定したオブジェクトが、現時点でメモリ内でどのように表されているかを保持します。Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトをチェックインする権限を持っているかどうかを確認します。

注	オブジェクトを使用するよりも、ビューを使用するほうが便利です。オブジェクトの修正には、ビューを使用してみてください。ビューについては、「 Identity Manager のビュー 」の章を参照してください。

表 2-6 checkinObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	checkinObject	
<i>object</i>	なし		チェックイン対象のオブジェクトを指定します。オブジェクトを指定しない場合、ワークフローサービスは機能しません。
<i>lockedBy</i>	なし		オブジェクトのロックに使用された代替ロック名を指定します。

戻り値と副作用

指定したオブジェクトがチェックインされます。エラーが発生した場合は、WF_ACTION_ERROR は true となり、Wave setResult にエラーが格納されます。

checkinView セッションワークフローサービス

ビューをチェックインします。

表 2-7 checkinView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	checkinView	
<i>view</i>	なし		チェックインするビューを指定します。ビューを指定しない場合、checkinView ワークフローサービスは機能しません。

戻り値と副作用

指定したビューは、Identity Manager のリポジトリにチェックインされます。エラーが発生した場合は、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

checkoutObject セッションワークフローサービス

持続的な Identity Manager オブジェクトを取得し、それをロックします。Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトをチェックアウトする権限を持っているかどうかを確認します。

注 オブジェクトを使用するよりも、ビューを使用するほうが便利です。オブジェクトの修正には、ビューを使用してみてください。ビューについては、「[Identity Manager のビュー](#)」を参照してください。

表 2-8 checkoutObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	checkoutObject	
<i>type</i>	あり		チェックアウト対象のオブジェクトのタイプを指定します。
<i>name</i>	あり		チェックアウト対象のオブジェクトの名前を指定します。
<i>lockedBy</i>	なし		ロックの記録に使用する代替名を指定します。
<i>lockWait</i>	なし		ロックに競合がある場合の待機時間を、秒単位で指定します。

戻り値と副作用

チェックアウトに成功すると、チェックアウトされたオブジェクトは *object* 変数に格納されます。ロックがタイムアウトした場合は、チェックアウトアクションのために WF_ACTION_TIMEOUT が true に設定されます。エラーが発生した場合は (管理者にオブジェクトをチェックアウトする権限がない場合を含む)、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

checkoutView セッションワークフローサービス

Identity Manager ビューをフェッチし、それをロックします。

表 2-9 checkoutView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	checkoutView	
<i>viewid</i>	なし		チェックアウトするビューの ID を指定します。 この引数を指定しない場合は、タイプと ID の両方を指定してください。ワークフローでは、タイプと ID を使用するほうが容易な場合があります。
<i>name</i>	あり		チェックアウト対象のオブジェクトの名前を指定します。
<i>type</i>	なし		ビューのタイプを指定します。
<i>id</i>	なし		ビューで操作されるオブジェクトの ID。たとえば、アカウントオブジェクトを操作するビューを作成するときは、これは accountId となります。
<i>raiseViewErrors</i>	なし		display.errors を WF_ACTION_ERROR に昇格させることを指定します。true に設定すると、ビューの display.errors 属性に含まれるエラーが自動的に抽出され、ワークフローエンジンにエラーとして返されます。 WF_ACTION_ERROR ワークフローの変数が設定されます。
<i>subject</i>	なし		ビューのチェックアウトを行う Identity Manager 管理者。これを指定しない場合は、ワークフローを呼び出した管理者 (WF_CASE_OWNER) が使用されます。

表 2-9 checkoutView メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>options</i>	なし		ビューに固有のオプション。渡すことができる値は、使用するビューによって異なります。もっとも一般的な値は、ユーザービューです。オプションは、 <code>session.UserViewConstants</code> にあります。単純なビューでは、 <code>*Viewer.java</code> ファイルでオプションの定数を宣言してください。ワークフローで使用される、その次に一般的なビューは <code>ProcessViewer</code> で、以後、 <code>PasswordViewer</code> 、 <code>DisableViewer</code> 、 <code>EnableViewer</code> 、および <code>RenameViewer</code> が続きます。これらのビューのオプションは、比較的少数です。

戻り値と副作用

チェックアウトに成功すると、チェックアウトされたビューは *view* 変数に格納されます。ロックがタイムアウトした場合は、チェックアウトアクションのために `WF_ACTION_TIMEOUT` が `true` に設定されます。エラーが発生した場合は (管理者にビューをチェックアウトする権限がない場合を含む)、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

createView セッションワークフローサービス

適切なフォームを使用して、指定したタイプの空のオブジェクトを作成します。

表 2-10 createView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	createView	
<i>viewid</i>	なし		作成するビューの ID を指定します。この引数を指定しない場合は、タイプと ID の両方を指定してください。ワークフローでは、タイプと ID を使用するほうが容易な場合があります。
<i>type</i>	なし		ビューのタイプを指定します。
<i>id</i>	なし		ビューで操作されるオブジェクトの ID を指定します。たとえば、アカウントオブジェクトを操作するビューを作成するときは、これは <i>accountId</i> となります。
<i>options</i>	なし		ビューに固有のオプションを指定します。渡すことができる値は、使用するビューによって異なります。もっとも一般的な値は、ユーザービューです。オプションは、 <code>session.UserViewConstants</code> にあります。単純なビューでは、 <code>Viewer.java</code> ファイルでオプションの定数を宣言してください。ワークフローで使用される、その次に一般的なビューは <code>ProcessViewer</code> で、以後、 <code>PasswordViewer</code> 、 <code>DisableViewer</code> 、 <code>EnableViewer</code> 、および <code>RenameViewer</code> が続きます。これらのビューのオプションは、比較的少数です。

戻り値と副作用

正しく作成されたビューは *view* 変数に格納されます。エラーが発生した場合は、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

disableUser セッションワークフローサービス

1 つまたは複数の、ユーザーのリソースアカウントを無効にします。

アカウントの無効化には、3 つの方法があります。

- `com.waveset.provision.WorkflowServices op=disable` を設定します。この章の「[disableUser セッションワークフローサービス](#)」を参照してください。これは、無効化の最低レベルの方法です。これは、プロビジョニングエンジンによって直接実装されます。ワークフロー、認証、または監査は関係しません。
- `com.waveset.session.WorkflowServices op=disableUser` を設定します。
- 次の引数を指定して、`com.waveset.session.WorkflowServices op=checkoutView` を設定します。

```
type=DisableViewer
id=<user account id>
```

2 番目と 3 番目の方法は、基本的には同じです。2 番目の方法では、チェックアウトと、Disable ビューへのチェックインを行います。これは、Identity Manager インタフェースでユーザーを無効にするのと同じです。副次的な影響として、この場合は Disable User ワークフローが呼び出されます。

カスタマイズした Disable User プロセスを実装する場合は、`provision.WorkflowServices` メソッドを使用します。その他のプロセスを実装する場合に、標準の Disable User プロセスを使用して無効化を行うときは、別の方法のいずれかを使用します。

注	デフォルトでは、Identity Manager はアカウントを無効にします。アカウントを無効化しないようにするには、disableUser ワークフローに次の引数を追加します。 <Argument name='doWaveset' value='false' />

表 2-11 disableUser メソッドの引数

名前	必要性	有効値	説明
op	あり	disableUser	作成するビューの ID を指定します。この引数を指定しない場合は、タイプと ID の両方を指定してください。ワークフローでは、タイプと ID を使用するほうが容易な場合があります。

表 2-11 disableUser メソッドの引数 (続き)

名前	必要性	有効値	説明
doWaveset	なし	true/false	無効にするリソースのリストを指定します。アカウントを無効化しないようにするには、この引数を false に設定します。 <Argument name='doWaveset' value='false' />
services	なし		作成するビューの ID を指定します。この引数を指定しない場合は、タイプと ID の両方を指定してください。ワークフローでは、タイプと ID を使用するほうが容易な場合があります。

戻り値と副作用

指定したリソースアカウントは無効化されます。エラーが発生した場合は、**WF_ACTION_ERROR** は **true** となり、**WavesetResult** にエラーが格納されます。

enableUser セッションワークフローサービス

1 つまたは複数の、ユーザーのリソースアカウントを有効にします。

表 2-12 enableUser メソッドの引数

名前	必要性	有効値	説明
op	あり	enableUser	
accountId	あり		アカウントを有効にする Identity Manager ユーザーを指定します。
doWaveset	なし	true/false	true の場合は、Identity Manager アカウントは、このユーザーで有効化されます。値を指定しない場合は、デフォルトとして true が適用され、アカウントは有効化されます。
services	なし		有効にするリソースのリストを指定します。この引数を指定しない場合は、ユーザーのすべてのリソースアカウントが有効化されます。

戻り値と副作用

指定したリソースアカウントは有効化されます。エラーが発生した場合は、`WF_ACTION_ERROR`は`true`となり、`WavesetResult`にエラーが格納されます。

findUser セッションワークフローサービス

Identity Manager リポジトリからユーザーを検索します。この機能は、より柔軟な `query` メソッドワークフローサービスより古いものです。 `query` メソッドワークフローサービスの使用を検討してください。

表 2-13 findUser メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	findUser	
attribute Name	あり		ユーザーの検索に使用する、Identity Manager の検索可能な属性の名前を指定します。
attribute Value	あり	true/false	ユーザーの検索に使用する、検索可能な属性の値を指定します。

戻り値と副作用

指定した属性の指定した値を持つ、最初に見つかったユーザーの `accountId` は、`accountId` 変数に格納されます。エラーが発生した場合は、`WF_ACTION_ERROR`は`true`となり、`WavesetResult`にエラーが格納されます。

getObject セッションワークフローサービス

指定した名前とタイプを持つ Identity Manager オブジェクトを返します。ワークフローを呼び出した管理者 (WF_CASE_OWNER) は、オブジェクトを取得する権限を持つかどうかを確認されます。

注 オブジェクトを使用するよりも、ビューを使用するほうが便利です。オブジェクトの表示には、ビューを使用してみてください。詳細については、「[Identity Manager のビュー](#)」を参照してください。

表 2-14 getObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	getObject	
<i>type</i>	あり	「タイプ名」を参照	Identity Manager からフェッチされるオブジェクトのタイプを指定します。
<i>name</i>	あり		Identity Manager からフェッチされるオブジェクトの名前を指定します。

戻り値と副作用

指定したタイプと名前を持つ Identity Manager オブジェクトは、object 変数に格納されます。エラーが発生した場合は、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

getProperty セッションワークフローサービス

Identity Manager オブジェクトからプロパティ値を取得します。Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトを表示する権限を持っているかどうかを確認します。

表 2-15 getProperty メソッドの引数

名前	必要性	有効値	説明
<i>name</i>	あり	getObject	プロパティを取得するオブジェクトの名前を指定します。
<i>propertyName</i>	なし	「タイプ名」を参照	オブジェクトから取得するプロパティの名前を指定します。 <i>propertyName</i> を指定しない場合、サービスに効果または戻り値はありません。

戻り値と副作用

オブジェクトに *propertyName* が見つかったら、プロパティの値が *propertyValue* 変数に格納されます。エラーが発生した場合は (管理者にオブジェクトを表示する権限がない場合を含む)、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

getResourceObject セッションワークフローサービス

指定したリソースから、タイプと ID によって指定されるリソースオブジェクトを取得します。

表 2-16 getResourceObject メソッドの引数

名前	必要性	有効値	説明
<i>resourceId</i>	あり		有効なリソースオブジェクトを指定します。
<i>objectType</i>	なし	「タイプ名」を参照	リソースの <ObjectType> セクションに定義される、有効なオブジェクトタイプを指定します。
<i>objectId</i>			このリソースの、有効な完全修飾オブジェクト識別子 (たとえば、dn) を指定します。

表 2-16 getResourceObject メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>options</i>			要求の動作を制御します。 有効なキーには、取得するオブジェクトの <i>objectType</i> に固有の属性名のリストを示す <i>searchAttrsToGet</i> などがあります。

戻り値と副作用

指定したリソースから、*objectType* と *objectId* によって指定されるオブジェクトを返します。

getView セッションワークフローサービス

Identity Manager ビューをフェッチします。ワークフローを呼び出した管理者 (WF_CASE_OWNER) は、オブジェクトを表示する権限を持つかどうかを確認されます。

表 2-17 getView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	getView	
<i>viewId</i>	なし		フェッチするビューの ID を指定します。この引数を指定しない場合は、タイプと ID の両方を指定してください。ワークフローでは、タイプと ID を使用するほうが容易な場合があります。
<i>type</i>	なし		ビューのタイプを識別します。
<i>id</i>	なし		ビューでフェッチされるオブジェクトの ID を指定します。たとえば、アカウントオブジェクトを表示するビューをフェッチするときは、これは <i>accountId</i> となります。
<i>raiseViewErrors</i>	なし	true または false	<i>display.errors</i> を WF_ACTION_ERROR に昇格させることを指定します。true に設定すると、ビューの <i>display.errors</i> 属性に含まれるエラーが自動的に抽出され、ワークフローエンジンにエラーとして返されます。WF_ACTION_ERROR ワークフローの変数が設定されます。

表 2-17 getView メソッドの引数 (続き)			
名前	必要性	有効値	説明
<i>options</i>	なし	<code>getView</code>	ビューに固有のオプションを指定します。渡すことができる値は、使用するビューによって異なります。もっとも一般的な値は、ユーザービューです。これらのオプションは、 <code>session.UserViewConstants</code> にあります。単純なビューでは、 <code>*Viewer.java</code> ファイルでオプションの定数を宣言してください。ワークフローで 사용되는、その次に一般的なビューは <code>ProcessViewer</code> で、以後、 <code>PasswordViewer</code> 、 <code>DisableViewer</code> 、 <code>EnableViewer</code> 、および <code>RenameViewer</code> が続きます。これらのビューのオプションは、比較的少数です。

戻り値と副作用

正しくフェッチされたビューは *view* 変数に格納されます。エラーが発生した場合は、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

getViewForm セッションワークフローサービス

指定したビューと関連付けられているフォームを返します。

表 2-18 getViewForm メソッドの引数			
名前	必要性	有効値	説明
<i>form</i>	なし	<code>getViewForm</code>	<i>formId</i> に置き換えられた引数を指定します。
<i>formId</i>	なし		取得するフォームの名前または ID を指定します。
<i>op</i>	あり		
<i>view</i>	なし		フォームからフェッチするビューを指定します。この引数を指定しない場合、 <code>getViewForm</code> ワークフローサービスは機能しません。

戻り値と副作用

正しくフェッチされたフォームは *form* 変数に格納されます。エラーが発生した場合は、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

listResourceObjects セッションワークフローサービス

指定した *resourceIds* のリストから、指定したリソースオブジェクトタイプのリソースオブジェクト名のリストを返します。

表 2-19 listResourceObjects メソッドの引数

名前	必要性	有効値	説明
<i>objectType</i>	あり		指定されているこの <i>resourceId</i> の、有効なオブジェクトクラスの名前を指定します。NULL の場合は、次の条件で定義されるすべてのオブジェクトタイプのオブジェクトが返されます。 指定した検索コンテナに含まれる、指定した <i>resId</i> 検索範囲と、指定した検索フィルタ
<i>resourceId</i>	なし		1 つの <i>resourceId</i> を指定します。この値は、名前が付けられたリソースから取得されます。NULL の場合、メソッドは <i>resourceIdsList</i> 引数を探します。
<i>resourceIdList</i>	あり		<i>resourceIds</i> を一覧表示します。これらの ID は、名前が付けられたリソースから取得されます。これは、 <i>resourceId</i> 引数が NULL (未定義) の場合にのみ使用されます。
<i>options</i>	なし		検索動作を制御する、オプションのマップを指定します。

オプションについては、次の表で説明します。

表 2-20 listResourceObjects メソッドのオプションの値

オプション	説明
<i>searchContext</i>	検索の実行対象となるコンテキストを指定します。指定しない場合、メソッドは RA_BASE_CONTEXT から値を取得しようとします。RA_BASE_CONTEXT にも値が指定されていない場合は、論理上の上位から検索が行われます。

表 2-20 listResourceObjects メソッドのオプションの値 (続き)

オプション	説明
searchFilter	(省略可能) RFC 1558 に規定されている LDAP 検索フィルタフォーマットでは、1 つまたは複数のオブジェクト <attr name> <condition> <value> が、and または or で連結されます。規定されていない場合は、指定した objectType を使用してフィルタが作成されます。
searchScope	検索対象を、指定した searchContext のコンテキストのみに含まれる現在のオブジェクト、または指定した searchContext に含まれるすべてのサブコンテキストに設定します。 有効な値には、object、oneLevel、または subTree があります。subTree は、指定した searchContext に含まれるすべてのサブコンテキストを対象に検索を実行することを表します。
searchTimeLimit	検索の制限時間を、ミリ秒単位で指定します。検索は、この制限時間内に行われます。
searchAttrsToGet	オブジェクトごとに取得する、objectType に固有の属性名のリストを指定します。
runAsUser	この要求を実行するユーザーの名前を指定します。指定しない場合は、デフォルト値として、リソースプロキシ管理ユーザーが適用されます。
runAsPassword	runAsUser のパスワードを指定します。指定したユーザーとしてリスト要求を実行する権限があることを、リソースに対して認証するために必要とされます。
cache	返されたオブジェクトリストをサーバーがキャッシュするかどうかを指定します。true に設定した場合は、サーバーは返されたオブジェクトリストをキャッシュします。デフォルトは false です。
cacheTimeout	キャッシュがタイムアウトになるまでの制限時間を、ミリ秒単位で指定します (cacheList が true の場合のみ有効)。キャッシュがタイムアウトになると、オブジェクトが、次の要求時に自動的にリソースから取得されます。
clearCache	キャッシュをクリアし、次の要求時に、オブジェクトをリソースから再フェッチするように強制します。

queryObjectNames セッションワークフローサービス

指定したクエリー属性と一致する名前のリストを返します。

表 2-21 queryObjectNames メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	queryObjectNames	
<i>type</i>	あり		検索対象のオブジェクトタイプを指定します。
<i>attributes</i>	なし		オブジェクトのクエリーに使用する、クエリー可能な属性の名前と値のペアのリストまたはマップを指定します。指定しない場合は、指定したタイプのすべてのオブジェクトが返されます。
<i>single</i>	なし	true false	true に設定した場合は、指定したクエリーと最初に一致したオブジェクトの名前が返されます。false に設定した場合、または値を指定しない場合は、指定したクエリーと一致するすべてのオブジェクトの名前が返されます。

戻り値と副作用

single が false または未指定の場合は、指定したクエリー属性と一致するオブジェクトの名前リストが *queryResult* 変数に格納されます。*single* が true の場合は、指定したクエリー属性と最初に一致するオブジェクトの名前が *queryResult* 変数に格納されます。エラーが発生した場合は、*WF_ACTION_ERROR* は true となり、*WavesetResult* にエラーが格納されます。

queryObjects セッションワークフローサービス

指定したクエリー属性と一致するオブジェクトのリストを返します。

このメソッドは、次のいずれかを受け付けます。

表 2-22 queryObjects メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	queryObjects	
<i>type</i>	あり		検索対象のオブジェクトタイプを指定します。
<i>attributes</i>	なし		次のいずれかを指定します。 WSAttribute タイプの要素を含むマップ AttributeCondition タイプの要素を含むリスト 指定しない場合は、指定したタイプのすべてのオブジェクトが返されます。
<i>single</i>	なし	true false	true に設定した場合は、指定したクエリーと最初に一致するオブジェクトが返されます。false に設定した場合は、指定したクエリーと一致するすべてのオブジェクトが返されます。

戻り値と副作用

single が false または未指定の場合は、指定したクエリー属性と一致するオブジェクトのリストが **queryResult** 変数に格納されます。**single** が true の場合は、指定したクエリー属性と最初に一致するオブジェクトが **queryResult** 変数に格納されます。エラーが発生した場合は、**WF_ACTION_ERROR** は true となり、**WavesetResult** にエラーが格納されます。

queryReferencingRoles セッションワークフローサービス

指定したロールを直接的または間接的に参照するすべてのロールのリストを返します。関連付けられているロールを特定する、対象のロールを指定するために、ロール引数が必要です。

戻り値と副作用

指定したロールが *role* として渡されると、このメソッドは、結果を *queryResult* 変数で返します。これは、指定したロールを直接的または間接的に参照する、Identity Manager リポジトリ内のすべてのロールオブジェクトのリストです。

refreshView セッションワークフローサービス

Identity Manager ビューを更新します。

表 2-23 refreshView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	unlockObject	
<i>view</i>	なし		更新するビューを指定します。この引数を指定しない場合、refreshView ワークフローサービスは機能しません。
<i>raiseViewErrors</i>	なし	true または false	display.errors を WF_ACTION_ERROR に昇格させることを指定します。true に設定すると、ビューの display.errors 属性に含まれるエラーが自動的に抽出され、ワークフローエンジンにエラーとして返されます。WF_ACTION_ERROR ワークフローの変数が設定されます。

戻り値と副作用

正しく更新されたビューは *view* 変数に格納されます。エラーが発生した場合は、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

removeDeferredTask セッションワークフローサービス

Identity Manager オブジェクトから延期タスクを削除するときに使用されます。
Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトを表示する権限を持っているかどうかを確認します。

表 2-24 removeDeferredTask メソッドの引数

名前	必要性	有効値	説明
<i>type</i>	なし	有効な値は、タイプのリストです。	延期タスクが削除されるオブジェクトのタイプを指定します。指定しない場合は、デフォルトのタイプである <i>user</i> と見なされます。
<i>name</i>	あり		延期タスクが削除されるオブジェクトの名前を指定します。
<i>task</i>	なし		実行する <i>TaskDefinition</i> の名前を指定します。
<i>taskDefinition</i>	なし		完全なタスク定義オブジェクトを指定します。この引数は、次に説明するその他のすべてのタスク引数の代わりに使用できます。完全なタスクオブジェクトと、その他の引数の両方を指定した場合は、独立して指定した引数が、タスク定義オブジェクトの引数の値に優先して適用されます。
<i>instanceName</i>	なし		作成する <i>TaskInstance</i> の名前を指定します。
<i>date</i>	なし	<i>java.util.Date</i> オブジェクト、または通常形式の文字列のいずれかです。	延期タスクを実行する日付を指定します。
<i>organization</i>	なし		<i>TaskInstance</i> を割り当てる組織を指定します。
<i>owner</i>	なし		<i>TaskInstance</i> の有効な所有者の名前を指定します (後の管理で使用される)。

表 2-24 removeDeferredTask メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>description</i>	なし		タスク管理ページに表示するために TaskInstance に割り当てるオプションの説明文を指定します。

removeProperty セッションワークフローサービス

Identity Manager オブジェクトのプロパティを削除します。Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトを表示する権限を持っているかどうかを確認します。

表 2-25 removeProperty メソッドの引数

名前	必要性	有効値	説明
<i>name</i>	あり		プロパティを設定するオブジェクトの名前を指定します。
<i>propertyName</i>	なし		変更するオブジェクトのプロパティの名前を指定します。 propertyName を指定しない場合、サービスは機能しません。

戻り値と副作用

Identity Manager オブジェクトのプロパティは削除されます。エラーが発生した場合は (管理者にオブジェクトを表示する権限がない場合を含む)、**WF_ACTION_ERROR** は **true** となり、**WavesetResult** にエラーが格納されます。

setProperty セッションワークフローサービス

Identity Manager オブジェクトのプロパティの値を設定します。Identity Manager は、ワークフローを呼び出した管理者が、オブジェクトを表示する権限を持っているかどうかを確認します。

表 2-26 setProperty メソッドの引数

名前	必要性	有効値	説明
<i>name</i>	あり	setObject	プロパティを設定するオブジェクトの名前を指定します。
<i>propertyName</i>	なし	「タイプ名」を参照	オブジェクトから取得するプロパティの名前を指定します。propertyName を指定しない場合、サービスは機能しません。
<i>propertyValue</i>	なし	プロパティに設定する値	propertyValue を指定しない場合、プロパティは NULL に設定されます。

戻り値と副作用

Identity Manager オブジェクトのプロパティに、指定した値を設定します。エラーが発生した場合は (管理者にオブジェクトを表示する権限がない場合を含む)、WF_ACTION_ERROR は true となり、Wave setResult にエラーが格納されます。

unlockObject セッションワークフローサービス

Identity Manager オブジェクトのロックを解除します。ワークフローを呼び出した管理者 (WF_CASE_OWNER) は、オブジェクトを修正する権限を持つかどうかを確認されます。オブジェクトを使用するよりも、ビューのほうが便利です。オブジェクトの修正には、ビューを使用してみてください。

表 2-27 unlockObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	unlockObject	
<i>type</i>	あり		ロック解除の対象となるオブジェクトのタイプを指定します。
<i>name</i>	あり		ロック解除の対象となるオブジェクトの名前を指定します。

表 2-27 unlockObject メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>lockedBy</i>	なし		オブジェクトのロックに使用された代替ロック名を指定します。

戻り値と副作用

成功した場合は、指定したオブジェクトのロックが解除されます。エラーが発生した場合は、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

unlockView セッションワークフローサービス

Identity Manager ビューのロックを解除します。

表 2-28 unlockView メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	unlockview	
<i>view</i>	なし		ロックを解除するビューを指定します。この引数を指定しない場合、unlockView ワークフローサービスは機能しません。

戻り値と副作用

成功した場合は、指定したビューのロックが解除されます。エラーが発生した場合は、`WF_ACTION_ERROR` は `true` となり、`WavesetResult` にエラーが格納されます。

プロビジョンワークフローサービス

`com.waveset.provision.WorkflowServices` にもサービスのセットがありますが、これらのサービスは、`com.waveset.session.WorkflowServices` のメソッドほど使用頻度が高くありません。これらのサービスは、アカウントを管理するための、低レベルの権限です。これをストックワークフローと呼びます。カスタムワークフローでは、これらのサービスを使用するか、`checkoutView/checkinView` でより高いレベルのビューを使用して、ストックワークフローを呼び出すことができます。

一般的なプロビジョンワークフローサービスの呼び出しの構造

ワークフローサービスは、ワークフローアクションから呼び出されます。プロビジョンワークフローサービスのアクションの一般的な形式は次のとおりです。

```
<Action class='com.waveset.provision.WorkflowServices'>
  <Condition/>
  <Argument name='op' value=workflowServiceOp/>
  <Argument name=argname1>
    <expression>value1expression</expression>
  </Argument>
  <Argument name=argname2>
    <expression>value2expression</expression>
  </Argument>
  ...
  <Argument name=argnameN>
    <expression>valueNexpression</expression>
  </Argument>
</Action>
```

サポートされる各ワークフローサービスには、それぞれ異なる数の、必須および省略可能な引数があります。

サポートされるプロビジョンワークフローサービス

次のリストは、Identity Manager が現在サポートしているプロビジョンワークフローサービスを示しています。ワークフローサービスの呼び出しに指定する *op* 引数は、次のいずれかの値である必要があります。

- approve
- auditNativeChangeToAccountAttributes
- bulkReprovision
- changeResourceAccountPassword
- checkDeProvision
- cleanupResult
- createResourceObject
- deleteResourceAccount
- deProvision
- disable
- enable
- getApprovals
- getApprovers
- lockOrUnlock
- notify
- provision
- questionLock
- reject
- reProvision
- runResourceAction
- unlinkResourceAccountsFromUser
- updateResourceObject
- validate

op 引数に上記以外の値を指定した場合、ワークフローサービスは次の結果を返します。

'Unknown WorkflowServices op'

また、ワークフローコンテキスト変数 `WF_ACTION_ERROR` は、`NULL` 以外の値に設定されます。

approveProvision プロビジョンワークフローサービス

ユーザーアカウントの承認を記録します。

表 2-29 approveProvision メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	approve	
<i>user.waveset.accountId</i> または <i>accountId</i>	あり		承認する Identity Manager ユーザーの名前を指定します。
<i>options</i>	なし		プロビジョニングの特性を指定するオプションを指定します。Javadoc で、 <code>com.waveset.provision</code> を参照してください。 ProvisioningOptions。

戻り値と副作用

承認の結果を記録した `WavesetResult` オブジェクト。

auditNativeChangeToAccountAttributes プロビ ジョンワークフローサービス

リソースアカウントの1つまたは複数の監査可能属性に加えられたネイティブ変更のレポートを生成します。

表 2-30 auditNativeChangeToAccountAttributes メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	auditNativeChangeToAccountAttributes	
<i>user.waveset.accountId</i> または <i>accountId</i>	あり		監査する Identity Manager ユーザーの名前を指定します。
<i>resource</i>	あり		監査対象のネイティブ変更が行われたリソースを指定します。
<i>prevAttributes</i>	なし	Map	変更前の属性の名前と値のペアのマップを指定します。
<i>newAttributes</i>	なし	Map	変更後の属性の名前と値のペアのマップを指定します。
<i>formattedChanges</i>	なし	Map	ログに記録する、フォーマットされた変更を指定します。

戻り値と副作用

監査の結果を記録した WavesetResult オブジェクト。

bulkReprovision プロビジョンワークフローサービス

このメソッドは、クエリーのセットを実行し、指定した条件と一致するすべてのユーザーを検索します。その後、このリストを繰り返し、ユーザーを一度に再プロビジョニングします。

このメソッドを使用してすべてのロール更新の自動再プロビジョニングをサポートするには、**Role** フォームに次のフィールドを追加します。

```
<Field name='processInputs.reprovision'>
  <Expansion><s>true</s></Expansion>
</Field>
```

表 2-31 bulkReprovision メソッドの引数

パラメータ	説明
<i>role</i>	ロール名を指定します
<i>roles</i>	ロール名のリスト
<i>organization</i>	組織名を指定します
<i>organizations</i>	組織名のリスト
<i>conditions</i>	LighthouseContext.list オブジェクトに適したクエリーオプションのマップを指定します
<i>options</i>	プロビジョニングオプションを指定します (たとえば、ターゲット、フェッチなど)
<i>maxErrors</i>	処理を停止するまでに許容されるエラーの最大数を指定します

戻り値

successes - 正しく再プロビジョニングされたユーザーを示します

failures - 完全に再プロビジョニングできなかったユーザーを示します

authenticateUserCredentials プロビジョンワークフローサービス

パスワードを使用して、リソースに対しユーザーを認証します。

表 2-32 authenticateUserCredentials メソッドの引数

名前	必要性	有効値	説明
<i>resourceName</i>	あり		認証の対象となるリソースの名前を指定します。
<i>accountId</i>	あり		名前が付けられたリソースに表示される、このアカウントの ID を指定します。
<i>accountPassword</i>	あり		このアカウントのパスワードを指定します (暗号化されていない文字列)。

戻り値

認証への呼び出しの結果を返します。成功した場合は、**WFContext** の変数 **passwordIsValid** を **true** に設定し、失敗した場合は何もしません。

changeResourceAccountPassword プロビジョンワークフローサービス

1 つまたは複数のリソースアカウントのパスワードを変更します。

表 2-33 changeResourceAccountPassword メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり		
<i>resources</i>	あり		パスワードを変更するリソースのリストを指定します。
<i>options</i>	なし		プロビジョニングの特性を指定するオプションを指定します。Javadoc で、 <code>com.waveset.provision.ProvisioningOptions</code> を参照してください。

戻り値と副作用

パスワード変更の結果を記録した `WavesetResult` オブジェクト。

cleanupResult プロビジョンワークフローサービス

タスク結果から `NULL` の `ResultError` を削除します。このメソッドは、引数として `op` をとります。有効な値は、`cleanupResult` です。

戻り値と副作用

クリーンアップの結果を記録した `WavesetResult` オブジェクト。

checkDeProvision プロビジョンワークフローサービス

削除前に、アカウントがプロビジョニング解除を必要とするかどうかを調べます。

表 2-34 checkDeProvision メソッドの引数

名前	必要性	有効値	説明
<code>op</code>	あり	<code>checkDeProvision</code>	プロビジョニング解除の必要性を調べる <code>Identity Manager</code> ユーザーの名前を指定します。
<code>user.waveset.accountId</code> または <code>accountId</code>	あり		プロビジョニング解除の必要性を調べるユーザーの名前を指定します。

戻り値と副作用

クリーンアップの結果を記録した `WavesetResult` オブジェクト。

createResourceObject プロビジョンワークフローサービス

リソースオブジェクト (たとえば、グループ) を作成します。

表 2-35 createResourceObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	createResourceObject	
<i>object</i>	なし	GenericObject	指定しない場合は、サービスは、 <i>ObjectType</i> の後に名前が指定された引数を探します。たとえば、リソースが <i>ObjectType</i> というグループの管理をサポートする場合に、 <i>object</i> 引数が指定されていないときは、サービスは <i>group</i> という引数にリソースオブジェクトが指定されているものと見なします。できる限り、 <i>object</i> 引数を使用してください。
<i>objectType</i>	あり		Identity Manager リソースは、管理するオブジェクトのタイプを指定できます。この指定には、リソースのプロトタイプ XML に定義される <i>ObjectType</i> 要素リストが使用されます。この引数は、これらの <i>ObjectType</i> のいずれかの名前です。
<i>resourceId</i>	あり		オブジェクトの作成先リソースのオブジェクト ID を指定します。

戻り値と副作用

オブジェクト作成の結果を記録した WavesetResult オブジェクト。

deleteResourceAccount プロビジョンワークフローサービス

リソースアカウントを削除します。

表 2-36 deleteResourceAccount メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	deleteResourceAccount	
<i>user.waveset.accountId</i> または <i>accountId</i>	あり		削除する Identity Manager ユーザーの名前を指定します。
<i>resource</i>	あり		アカウントの削除が必要なリソースの名前を指定します。

戻り値と副作用

アカウント削除の結果を記録した WavesetResult オブジェクト。

deleteResourceObject プロビジョンワークフローサービス

リソースオブジェクト (たとえば、グループ) を削除します。

表 2-37 deleteResourceObject メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	deleteResourceObject	
<i>objectType</i>	あり		Identity Manager リソースは、管理するオブジェクトのタイプを指定できます。この指定には、リソースのプロトタイプ XML に定義される ObjectType 要素リストが使用されます。この引数は、これらの ObjectTypes のいずれかの名前です。

表 2-37 deleteResourceObject メソッドの引数 (続き)

名前	必要性	有効値	説明
<i>object</i>	なし	GenericObject	指定しない場合は、サービスは、ObjectType の後に名前が指定された引数を探します。たとえば、リソースが ObjectType というグループの管理をサポートする場合に、object 引数が指定されていないときは、サービスは group という引数にリソースオブジェクトが指定されているものと見なします。できる限り、object 引数を使用してください。
<i>resourceId</i>	あり		オブジェクトを削除するリソースのオブジェクト ID を指定します。

戻り値と副作用

オブジェクト作成の結果を記録した WavesetResult オブジェクト。

deProvision プロビジョンワークフローサービス

Identity Manager アカウント、リソースアカウント、またはその両方を削除します。

表 2-38 deProvision メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	deProvision	
<i>accountId</i>	あり		プロビジョニング解除する Identity Manager ユーザーの名前を指定します。
<i>subject</i>	なし		呼び出しの有効な主体を指定します。指定しない場合は、タスクの主体が適用されます。この引数の値として none を指定した場合は、認証は行われません。
<i>options</i>	なし		指定しない場合は、次に指定された引数が適用されます。指定した場合は、次に指定された引数が、このオプションマップに含まれる同じ引数に優先して適用されます。

戻り値と副作用

プロビジョニング解除の結果を記録した `WavesetResult` オブジェクト。

deleteUser プロビジョンワークフローサービス

Identity Manager ユーザーを削除します。

表 2-39 deleteUser メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	<code>deleteUser</code>	
<i>subject</i>	なし		呼び出しの有効な主体を指定します。指定しない場合は、タスクの主体が適用されます。この引数の値として <code>none</code> を指定した場合は、認証は行われません。
<code>user.waveset.accountId</code> または <code>accountID</code>	あり		承認する Identity Manager ユーザーの名前を指定します。
<i>options</i>	なし	Map	オプション名とオプション値のペアのマップの値。指定しない場合は、次に指定された引数が適用されます。指定した場合は、次に指定された引数が、このオプションマップに含まれる同じ引数に優先して適用されます。
<code>user.waveset.accountId</code> または <code>accountID</code>	あり		削除する Identity Manager ユーザーの名前を指定します。
<i>adminName</i>	なし		削除を実行する管理者の名前を指定します。
<i>loginAppName</i>	なし		ログインアプリケーションの名前を指定します。
<i>force</i>	なし	<code>true/false</code>	<code>true</code> を指定した場合は、プロビジョニング解除が必要なリソースアカウントの有無に関係なく、ユーザーが削除されます。

戻り値と副作用

ユーザー削除の結果を記録した `WavesetResult` オブジェクト。

disable プロビジョンワークフローサービス

Identity Manager アカウント、リソースアカウント、またはその両方を無効にします。

表 2-40 disable メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	disable	
<i>subject</i>	なし		呼び出しの有効な主体。指定しない場合は、タスクの主体が適用されます。この引数の値として none を指定した場合は、認証は行われません。
<i>options</i>	なし		オプション名とオプション値のペアのマップの値。指定しない場合は、次に指定された引数が適用されます。指定した場合は、次に指定された引数が、このオプションマップに含まれる同じ引数に優先して適用されます。
<i>accountId</i>	なし		
<i>adminName</i>	なし		処理を実行する管理者の名前を指定します。
<i>loginAppName</i>	なし		ログインアプリケーションの名前を指定します。
<i>doLighthouse</i>	なし	true/false	Identity Manager アカウントを無効にするかどうかを指定します。
<i>doResources</i>	なし	true/false	ユーザーのアカウントを無効にするかどうかを指定します。
<i>doAuthenticators</i>	なし	true/false	true に設定した場合は、すべてのパススルー認証リソースが無効化されます。

戻り値と副作用

無効化の結果を記録した WavesetResult オブジェクト。

enable プロビジョンワークフローサービス

Identity Manager アカウント、リソースアカウント、または両方を有効にします。

表 2-41 enable メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	enable	
<i>subject</i>	なし		呼び出しの有効な主体を指定します。指定しない場合は、タスクの主体が適用されます。この引数の値として none を指定した場合は、認証は行われません。
<i>options</i>	なし	Map	オプション名とオプション値のペアのマップの値。指定しない場合は、次に指定された引数が適用されます。指定した場合は、次に指定された引数が、このオプションマップに含まれる同じ引数に優先して適用されます。
<i>accountId</i>	なし		有効化する Identity Manager ユーザーの名前を指定します。
<i>adminName</i>	なし		処理を実行する管理者の名前を指定します。
<i>loginAppName</i>	なし		ログインアプリケーションの名前を指定します。
<i>doLighthouse</i>	なし	true/false	Identity Manager アカウントを有効にするかどうかを指定します。
<i>doResources</i>	なし	true/false	ユーザーのアカウントを有効にするかどうかを指定します。
<i>doAuthenticators</i>	なし	true/false	true に設定した場合は、すべてのパススルー認証リソースが有効化されます。

戻り値と副作用

有効化の結果を記録した WavesetResult オブジェクト。

getApprovals プロビジョンワークフローサービス

既存のアカウントに割り当てられたルール、組織、およびリソースの承認のリストを決定します。

表 2-42 getApprovals メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	getApprovers	
<i>user</i>	あり	GenericObject	ユーザーを表示するビューを指定します。

戻り値と副作用

getApprovals ワークフローが成功すると、ユーザーに必要な承認を表す GenericObject が *approvals* 変数に返されます。エラーが発生した場合は、WF_ACTION_ERROR は true となり、WavesetResult にエラーが格納されます。

lockOrUnlock プロビジョンワークフローサービス

指定されたユーザーに関連付けられている Lighthouse アカウントポリシーがロックの有効期限の時刻を指定している場合に、そのユーザーをロックまたはロック解除します。

次のいずれかの条件と一致する場合に、ユーザーのロックは解除されます。

- ユーザーのロック解除機能が割り当てられているユーザーが、ユーザーの Lighthouse アカウントのロックを解除する
- ユーザーのロック有効期限の日時が設定されている場合に、現在の日時がその日時を過ぎている

ロックされているユーザーは、ログインアプリケーションインタフェース (たとえば、ユーザーまたは管理者インタフェース) からログインできません。デフォルトでは、パスワードログインの失敗試行回数が、ユーザーの Lighthouse アカウントポリシーで許可されている回数を超えた場合に、Password Login プロセスの Failure アクティビティによって lock サービスが呼び出されます。

引数

lock -- (ブール型) ユーザーがロックされているかどうかを示します (ロックされている場合は true)。

戻り値と副作用

ロック要求の場合は、このメソッドは NULL を返します。ロック解除要求の場合は、ロック解除の結果を含む `WavesetResult` を返します。

notify プロビジョンワークフローサービス

通知を送信します。ほとんどの場合は、電子メールが使用されます。

表 2-43 notify メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	notify	
<i>type</i>	あり		通知のタイプを指定します。指定しない場合は、電子メールタイプが適用されます。
その他の各種引数	なし。ただし、できるだけ指定してください。		通知ツールに渡される引数を指定します。名前と値のペアのマップ。

戻り値と副作用

通知の結果を記録した `WavesetResult` オブジェクト。

provision プロビジョンワークフローサービス

新しい Identity Manager アカウントと、(オプションとして) リソースアカウントを作成します。

表 2-44 provision メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	Provision	
<i>user</i>	あり	GenericObject	プロビジョニングされるビューを含む GenericObject を指定します。
各種プロビジョニングオプション	なし		プロビジョニングの特性を指定するオプションを指定します。Javadoc で、 <code>com.waveset.provision.ProvisioningOptions</code> を参照してください。

戻り値と副作用

プロビジョニングの結果を記録した `WavesetResult` オブジェクト。

questionLock プロビジョンワークフローサービス

ユーザーをロックしますが、ロックの有効期限の日時は設定しません。

このサービスによってロックされたユーザーは、ユーザーのロック解除機能持つ管理者が明示的にロック解除できます。管理者がこのユーザーのロックを解除しない場合、次にユーザーのパスワードが変更またはリセットされると、そのユーザーのアカウントは自動的にロック解除されます。このサービスによってロックされたユーザーも、「パスワードをお忘れですか」 ページ以外の **Identity Manager** アプリケーションインタフェースからはログイン可能です。

デフォルトでは、質問ログインの失敗試行回数が、ユーザーの **Lighthouse** アカウントポリシーで許可されている回数を超えた場合に、**Question Login** プロセスの **Failure** アクティビティによって `questionLock` サービスが呼び出されます。

戻り値と副作用

`NULL` が返されます。

reject プロビジョンワークフローサービス

リソースアカウントの却下を記録します。

表 2-45 reject メソッドの引数

名前	必要性	有効値	説明
<code>op</code>	あり	<code>reject</code>	
<code>user.waveset.accountId</code> または <code>accountId</code>	あり		却下する Identity Manager ユーザーの名前を指定します。
<code>options</code>	なし	<code>Map</code>	プロビジョニングの特性を指定するオプションを指定します。Javadoc で、 <code>com.waveset.provision.ProvisioningOptions</code> を参照してください。

戻り値と副作用

却下の結果を記録した `WavesetResult` オブジェクト。

reProvision プロビジョンワークフローサービス

既存の Identity Manager アカウントを更新します。

表 2-46 reProvision メソッドの引数

名前	必要性	有効値	説明
<i>op</i>	あり	reProvision	
<i>user</i>	あり	GenericObject	再プロビジョニングされる ビューを含む GenericObject を 指定します。
各種プロビジョ ニングオブショ ン	なし		プロビジョニングの特性を指定 するオプションを指定します。 Javadoc で、 com.waveset.provision.Provi sioningOptions を参照してく ださい。

戻り値と副作用

再プロビジョニングの結果を記録した WavesetResult オブジェクト。

runResourceAction プロビジョンワークフローサービス

リソースの指定されたリソースアダプタに対して、リソースアクションを実行します。

リソースアクションは、直接 (*resourceAction* を使用)、または ID (*resourceId* を使用) で指定できます。*resourceAction* がリソースアクションを正しく参照する場合は、その値は、*resourceId* に指定されている ID に優先して適用されます。

このメソッドには、認証チェックは含まれません。リソースアクションに適用されるすべての規則が適用されます。

表 2-47 runResourceAction メソッドの引数

名前	必要性	有効値	説明
<i>resourceId</i>			使用するリソースを指定します。
<i>resourceAction</i>			(省略可能) 実行するリソースアクションを指定します。この値は、 <i>resourceId</i> に指定されている ID に優先して適用されます。
<i>resourceActionId</i>			アダプタが実行するリソースアクションを ID で指定します。
<i>resourceActionArgs</i>			このメソッドが使用する引数のマップを提供します。これらの引数は、環境値としてアクションのスクリプトに渡されます。変数の名前は、RESACTIONARG_<key> で、値はそのキーに対応する値です。キーと値は、文字列で指定する必要があります。
注	前述した変数へのアクセスには、例外があります。ACTIONS をサポートする HostAccessResourceAdapter サブクラスは、アクションの実行に JavaScript を使用し、引数は additionalArgs というグローバル変数に渡されます。		

戻り値と副作用

実行したアクションの結果を記録した WavesetResult オブジェクト。

updateResourceObject プロビジョンワークフローサービス

リソースオブジェクト (たとえば、グループ) を更新します。

表 2-48 updateResourceObject メソッドの引数

名前	必要性	有効値	説明
op	あり	updateResourceObject	
objectType	あり		Identity Manager リソースは、管理するオブジェクトのタイプを指定できます。この指定には、リソースのプロトタイプ XML に定義される ObjectType 要素リストが使用されます。この引数は、これらの ObjectTypes のいずれかの名前です。
group	あり	GenericObject	リソースで更新する GenericObject を指定します。指定しない場合は、サービスは、ObjectType の後に名前が指定された引数を探します。たとえば、リソースが、group というオブジェクトタイプの管理をサポートしているとします。object 引数を指定しない場合、サービスは、group という引数にリソースオブジェクトが指定されているものと見なします。できる限り、object 引数を使用してください。
resourceId	あり		オブジェクトを更新するリソースのオブジェクト ID を指定します。

戻り値と副作用

オブジェクト更新の結果を記録した WavesetResult オブジェクト。

タイプ名

次に、Identity Manager で使用される有効なタイプ名を示します。

- AdminGroup
- Administrator
- Application
- AttributeDefinition
- AuditConfig
- AuditLog
- AuditLogPrunerTask
- AuditQuery
- Configuration
- DataStore
- DataType
- Discovery
- Encryptionkey
- EmailTemplate
- Event
- Extract
- LoadConfig
- Log
- LoginConfig
- NotifyConfig
- ObjectGroup
- Policy
- RemedyConfig
- ReportCounter
- ReportLog
- Resource
- ResourceAccount

- ResourceObject
- Role
- RoleAttribute
- Rule
- TaskDefinition
- TaskInstance
- TaskResult
- TaskResultPage
- TaskSchedule
- TaskTemplate
- TestItem
- UsageReport
- User
- UserExternalAttribute
- WorkItem

権限名

次に、Identity Manager で使用される有効な権限名を示します。

- Approve
- Bypass Verify
- Change Password
- Configure
- Connect
- Create
- Delete
- Deprovision
- Disable
- Disconnect
- Enable
- Execute
- Export
- Import
- List
- Modify
- Provision
- Reject
- Rename
- Reset Password
- Unlock
- View

アクション名

次に、Identity Manager で使用される有効なアクション名を示します。

- Approve
- CancelReconcile
- ChangePassword
- Configure
- Connect
- Create
- CredentialsExpired
- Delete
- Deprovision
- Disable
- Disconnect
- Enable
- Load
- Login
- Logout
- Modify
- NativeChange
- Provision
- Reject
- RequestReconcile
- Terminate
- View

Identity Manager フォーム

この章では、Sun Java™ System Identity Manager の管理者向けおよびユーザー向けインタフェースの各種ページを定義するフォームを編集することで、選択したページの外観や動作をカスタマイズする方法について説明します。

この章の内容

この章は、次の各節で構成されています。

- [フォームについて](#) - フォームの基本概念を紹介し、Identity Manager にフォームを統合する方法について説明します。
- [フォームのカスタマイズ](#) - フォームを操作するときに使用する、プログラミングの構文とロジックに関するガイドラインについて説明し、各種フォーム要素の例を示します。
- [カスタマイズしたフォームの検証](#) - フォームの構文を検証したり、カスタムフォームのフィールドロジックを追跡したりする方法について説明します。

関連する章

- 「[Identity Manager のビュー](#)」 - ユーザービューと呼ばれる Identity Manager の内部データ構造と連携する Identity Manager フォームについて説明します。フォームをカスタマイズするときは、ビューの属性を呼び出します。
- 「[HTML 表示コンポーネント](#)」 - フォームを編集するときは、フィールド定義の作成に HTML コンポーネント言語を使用します。
- 「[XPRESS 言語](#)」 - 式を使用して、フォームにロジックを設定します。
- 「Identity Manager 配備ツール」に記載されている「Identity Manager 統合開発環境の概要」 - Identity Manager 統合開発環境 (IDE) と、ツールの起動、エディタオプションの設定、フォームの保存とデバッグの方法について説明します。

フォームについて

Identity Manager の Web ベースのユーザーインタフェースの外観と機能をカスタマイズするには、編集する Web ページに関連するフォームを修正します。

「フォーム」という用語は、ユーザーが情報を入力する Web ページと、ビューへのデータの表示方法に関する規則を含むオブジェクトの両方を意味します。このガイドでは、「フォーム」という用語を、主に、ビューへのデータの表示方法に関する規則を含むオブジェクトという意味で使用します。

この節で説明する内容は次のとおりです。

- フォームとは
- フォームの編集が必要な理由
- フォームが使用される Identity Manager ページ
- 編集されたフォーム
- フォームが機能するしくみ

フォームとは

フォームは、ページに関連付けられたオブジェクトであり、そのページでユーザービュー属性をブラウザ上にどのように表示するかについての規則が含まれています。フォームにはビジネスロジックを組み込むことができ、通常はユーザーに表示する前に、表示データを処理するためにフォームが使用されます。

たとえば、新しいユーザーアカウントを作成するときは、新規ユーザーに関する情報を入力するための「ユーザーの作成」ページを使用します。このページは、**Tabbed User Form** という Identity Manager リポジトリに含まれるオブジェクト (フォーム) から生成されます。このフォームは、「ユーザーの作成」ページにどのフィールドを表示するか、および各フィールドの表示にどの HTML フォームエレメント (たとえば、テキストボックス、チェックボックス、選択ボックスなど) を使用するかを指定します。また、このフォームは、フィールドの無効化、空のフィールドへのデフォルト値の挿入、別フィールドの値からのフィールド値の計算を行う追加ロジックも指定します。

フォームが制御する内容

フォームによって制御されるオブジェクトやアクティビティは次のとおりです。

- ページのレイアウトや表示特性

フォームはフィールドから構成されます。表示できるフィールドのタイプには、単純なテキストボックス、ラジオボタン、複数の値を持つ選択ボックスなどがあります。フィールドには、別のフィールドから算出される値を割り当てることができ、さらに、それを読み取り専用にしたり、非表示にしたりすることができます。

- ページで使用するデータ

データは、リソースから動的に取得したり、別のフィールドから算出したりすることができます。Identity Manager の式言語である XPRESS を使用することで、フィールドデータを計算、結合、および論理的に評価することができます。

- システムに渡されるデータ

フォームは、Web ページ用のインタフェースにもなりますが、ActiveSync リソースのように対話的でないシステムのインタフェースとなることもできます。この場合、フォームは表示フィールドを持ちませんが、デフォルト値やその他のフィールド値を設定する規則を提供できます。

たとえば、「フルネーム」フィールドは、ページを使用する管理者には表示されないかもしれませんが、ユーザーが「名」、「ミドルネーム」、「姓」の各フィールドに入力した値に基づいて設定させることができます。別のフィールドからフィールドに値を取り込むことで、ユーザーや管理者が行うデータ入力が減り、そのためデータ入力エラーが生じる可能性も低くなります。同様に、テキスト入力フィールドにオプションメニューを用意することで、管理者は部署名を入力する代わりに、リストから選択することができます。デフォルトの Identity Manager フォームを定義する各種 HTML コンポーネントについては、「[HTML 表示コンポーネント](#)」を参照してください。

- Identity Manager のバックグラウンド処理

Identity Manager では、バックグラウンド処理にもフォームが使用されます。たとえば、フォームとリソースアダプタを一緒に使用して、外部リソースから得た情報を処理し、Identity Manager リポジトリにこれを保存できます。

ユーザーに表示されないフォームでは、外観は意味を持たないため、バックグラウンドでデータを操作するフォームを作成するときは、主にエンコーディングのロジックに注意します。非表示のコンポーネントの詳細については、「[非表示コンポーネントの使用](#)」の節を参照してください。

フォームの例

次のXMLコード例は、ユーザーがアカウントID、姓、名、およびフルネームを入力するときに使用するフォームフィールドを定義しています。この例は、「姓」フィールドと「名」フィールドに入力された情報から、ユーザーのフルネームがどのように構築されるのかを示しています。

```
<Field name='waveset.accountId' />
  <Display class='text'>
    <Property name='title' value='AccountID' />
  </Display>
</Field>
<Field name='global.firstname'>
  <Display class='Text'>
    <Property name='title' value='First Name' />
    <Property name='size' value='32' />
    <Property name='maxLength' value='128' />
  </Display>
</Field>
<Field name='global.lastname'>
  <Display class='Text'>
    <Property name='title' value='Last Name' />
    <Property name='noNewRow' value='true' />
    <Property name='size' value='32' />
    <Property name='maxLength' value='128' />
  </Display>
</Field>
<Field name='global.fullname'>
  <Display class='Text'>
    <Property name='title' value='FullName' />
    <Property name='size' value='32' />
    <Property name='maxLength' value='32' />
  </Display>
  <Expansion>
```



```
<concat>

  <ref>global.firstname</ref>

  <s> </s>

  <ref>global.lastname</ref>

</concat>

</Expansion>

</Field>
```

フォームの編集が必要な理由

デフォルトの Identity Manager ページには、製品でのアクションの実行に必要なすべてのフィールドがすでに用意されていますが、なぜこれをカスタマイズするのでしょうか。これは、デフォルトのフォームをカスタマイズすることで、企業のポリシーとプロセスをより確実に適用できるからです。

- **画面に表示されるユーザーのアカウント情報を制限することで、プライバシーを保護します。** プライバシーに対する懸念や、不要な情報が招く混乱を回避するために、ユーザーに応じて、特定の情報をユーザーアカウントに表示しないようにすることが必要な場合があります。
- 各フィールドの使い方に適したヘルプを表示します。これにより、混乱を少なくし、ヘルプデスクへの問い合わせも減らすことができます。
- **特定のタスクを実行するユーザーに不要な情報による混乱を減らします。** 通常、情報をもっとも効果的に表示する方法は、現在のタスクの実行に必要なフィールドのみを表示することです。

Identity Manager フォームのデフォルトフィールドをカスタマイズすることで、実際の環境に合わせてアプリケーションを拡張し、カスタマイズすることができます。具体的には、デフォルトフォームを次のようにカスタマイズできます。

- 組織内のユーザーの固有のニーズに対応します。これは、複数の異なる種類の管理者が、同じビューデータの異なる部分にアクセスする必要があり、記録されているすべてのデータ属性を表示できない場合に特に重要です。たとえば、人事担当の管理者は、ヘルプデスクの管理者とは異なるサブセットのユーザーアカウント属性にアクセスする必要があります。
- ユーザーアカウント属性の表示と内容 (特に、「ユーザーの作成」ページと「ユーザーの編集」ページに表示される属性) を制御します。これらのページには、制御が必要なほとんどの属性が含まれます。
- ユーザービュー属性のデフォルト値 (および、関連するその他の属性) を定義します。たとえば、管理者がホームディレクトリのパスを入力する代わりに、ユーザーのデフォルトのホームディレクトリを定義できます。

- 画面に表示する前にユーザービュー属性を処理できます。たとえば、頭字語や数値 ID としてリソースに記録されている部署コードを、ユーザーにわかりやすいフルネーム表記で表示することができます。
- 入力後にユーザービュー属性データを処理できます。たとえば、場所のフィールドの値に基づいて、メールアカウントを自動的に作成できます。
- 複数のフィールドを 1 つの行に配置することで、画面の構成を制御できます。Identity Manager フォームのフィールドの配置をカスタマイズすることで、印刷時の形式や、以前からの Web フォームにより近づけることができます。
- 非表示の属性を計算するための規則を定義します。たとえば、ユーザーの名前、ピリオド、ユーザーの姓、次にメールアドレス (joe.user@sun.com) となるように、ユーザーの電子メールアドレスを計算できます。

シナリオの例

さまざまなニーズや目的を持つ人々が同じデータにアクセスする環境では、フォームは特に便利です。

たとえば、企業で採用を担当するマネージャー用のフォームを作成し、新規従業員アカウントの作成に使用できます。デフォルトの **Tabbed User Form** には、採用責任者に必要な情報より多くの情報が表示されます。99 すべてのフィールドが表示される、冗長で込み入ったフォームでは、ユーザータスクが複雑になる可能性があります。その代わりに、採用責任者に必要な 10 の属性フィールドだけを表示し、それ以外の 89 の属性は、管理者が定義した規則に従って設定されるようなフォームを作成できます。

フォームが使用される Identity Manager ページ

通常は、Identity Manager フォームは次のいずれかのカテゴリに分類されます。

- **グラフィカルユーザーインターフェースを駆動するフォーム。**これらのフォームは、Identity Manager の管理者インターフェースやユーザーインターフェースの一部として使用ことができ、次のような場合にユーザーが使用するページを含みます。
 - パスワードを変更する
 - セルフサービスを実行する
 - アカウント作成、システム設定、およびワークフロータスクに関連する管理タスクを実行する

Identity Manager に付属するデフォルトのフォームは、独自のカスタムフォームを作成するためのたたき台として使用できます。これらのフォームのサブセットのみをコピーし、直接編集する必要がある場合は(「**編集されたフォーム**」の節を参照)、たとえば、特定の属性や動作をエンコードする方法について、別のフォームを参考にできます。

- 外部リソースから Identity Manager にインポートされる情報のバックグラウンド処理を行うフォーム。たとえば、PeopleSoft データベースから Identity Manager への情報の読み込み処理の一部として、受信レコードに含まれる、従業員の状態を調べるフォームなどがあります。従業員の状態がアクティブでない場合 (A)、そのユーザーの Identity Manager アカウントを無効にするフィールドがフォームによって定義されます。

次の表は、最初のタイプのフォームを使用する Identity Manager ページの一部を示しています。編集するページの表示特性を制御するフォームを特定するときは、この表を使用してください。

表 3-1 ページ、および関連 JSP とフォーム

編集するページ	関連する JSP	関連するフォーム
ユーザーの作成 / ユーザーの編集	account/modify.jsp	Tabbed User Form
ユーザーアカウント属性の変更	user/changeAll.jsp	エンドユーザーフォーム
ようこそ	user/anonmmain.jsp	Anonymous User Menu
作業項目の編集	approval/itemEdit.jsp	承認フォーム

編集されたフォーム

Identity Manager に付属するデフォルトフォームの中で、次の 5 つのフォームのいずれかの編集が必要となることが考えられます。

- エンドユーザーメニューフォーム
- Anonymous User Menu Form
- Tabbed User Form
- エンドユーザーフォーム
- 承認フォーム

これらの編集されたフォームは、ユーザーの作成と修正を制御し、ユーザーに表示されるメインメニューの表示を制御します。各フォームについては、次の節でさらに詳しく説明します。

エンドユーザーメニューフォーム

エンドユーザーメニューフォームは、Identity Manager のユーザーインタフェースのメインメニューの表示を制御します。通常、このフォームには、ユーザーパスワードの変更、アカウント属性の編集、秘密質問の回答の変更などを行うためのリンクが含まれます。

エンドユーザーメニューフォームをカスタマイズして、ユーザーがアクセスできる特別なワークフロープロセス（たとえば、システムへのアクセスを要求するプロセス）を呼び出すリンクを追加できます。

たとえば、エンドユーザーページでクリックするリンクとしてエンドユーザーテストプロセスを表示するときは、次のコード例のようなエントリを追加します。

コード例 3-1 エンドユーザーメニューフォームへのエンドユーザーテストプロセスリンクの追加

```
<Configuration id='#ID#Configuration:EndUserTasks' name='End User
Tasks'>
  <Extension>
    <List>
      <List>
        <String>End-User Test Process</String>
        <String>An example end-user workflow</String>
      </List>
    </List>
  </List>
</Configuration>
```

Identity Manager のユーザーインタフェースには、選択可能なセルフサービスプロセスリストが表示されます。これは、リストのサブリストとなります。サブリストの最初の要素はプロセス名を示し、2 番目の要素はプロセスの機能を示します。

注	ページの表示を更新するたびに、Identity Manager はフォームの <Default> 式を再評価します。エンドユーザーメニューフォームに doNotRegenerateEndUserMenu プロパティ (true に設定) を追加することで、このフォームの再生成を無効にできます。

ページの表示を更新するたびに、Identity Manager はフォームの <Default> 式を再評価します。エンドユーザーメニューフォームに doNotRegenerateEndUserMenu プロパティ (true に設定) を次のように追加することで、このフォームの再生成を無効にできます。

```
<Properties>

  <Property name='doNotRegenerateEndUserMenu'>

    <Boolean>true</Boolean>

  </Property>

</Properties>
```

Anonymous User Menu Form

Anonymous User Menu Form は、未知のユーザーがログインしたときに、Identity Manager のユーザーインタフェースのメインメニューの表示を制御します。

ユーザーセルフプロビジョニングのプロセス経由でシステムに定義されていないユーザーに対しては、Identity Manager は匿名エンドユーザーページを使用します。たとえば、Identity Manager 管理者は、Active Directory リソースのパススルー認証を設定することができます。それにより、Active Directory アカウントを持つ人であれば、Identity Manager のユーザーインタフェースにログインできるようになります。ユーザーが Identity Manager アカウントログインを持たない場合に、Identity Manager のユーザーオブジェクトが作成され、Active Directory リソースが追加されるように、これらのページをカスタマイズできます。それに続いて一連の質問を設けることで、システムはユーザーのロール、組織、およびその他のリソースを設定できます。

Identity Manager ユーザーが作成される前に、サービスを要求するワークフロープロセスを呼び出すよう Anonymous User Menu Form をカスタマイズすることができます。

Tabbed User Form

Tabbed User Form は、Identity Manager の管理者インタフェースでユーザーを作成、修正するときに使用されるデフォルトフォームです。独自にデザインしたフォームとして、このフォームの拡張版を作成するためには、このフォームのコピーをカスタマイズします。

ヒント

Tabbed User Form を直接編集しないでください。代わりに、これらのフォームのコピーを作成して一意の名前を付け、このファイル名を変更したコピーフォームを編集することをお勧めします。こうすることにより、カスタマイズしたコピーがサービスパックの更新やアップグレードの際に上書きされるのを防止できます。

Tabbed User Form のコピーは、次の処理を行うようにカスタマイズできます。

- 「ユーザーの編集」ページに表示される属性の数を制限します。デフォルトでは、このページにリソースのスキーママップに定義されているすべての属性が表示されるため、採用責任者に入力や設定が求められる属性のリストが多くなり過ぎてしまいます。
- デフォルトのフィールドタイプを、より便利な選択ボックス、チェックボックス、および複数の値を示すフィールドに設定します。デフォルトでは、ユーザーに割り当てられるリソースに定義されるすべての属性は、「ユーザーの作成」ページと「ユーザーの編集」ページにテキストボックスとして（または、Boolean 型の場合はチェックボックスとして）表示されます。
- 複数のページで共通のフォームを使用できるように、追加のフォームを含めます。

Tabbed User Form には、次のフィールドが含まれます。

- accountId
- firstname
- lastname
- role
- organization
- password
- confirm password
- email
- resource list
- application list
- MissingFields

注	「MissingFields」フィールドは、実際にはフィールドではなく、フォームジェネレータに通知されたエレメントです。Tabbed User Form に明示的に宣言されず、割り当てリソースに適用されるすべての属性のグローバルネームスペースに、テキストフィールドとして自動的に生成されます。フォームロケータがこれらのフィールドを明示的に生成しないようにする場合は、このフィールドを削除してください。

デフォルトでは、ユーザーに割り当てられるリソースに定義されるすべての属性は、「ユーザーの作成」ページと「ユーザーの編集」ページにテキストボックスとして（または、ブール型の値ではチェックボックスとして）表示されます。

エンドユーザーフォーム

エンドユーザーフォームは、ユーザーが Identity Manager のユーザーインタフェースで /user/main.jsp から **アカウント属性** を選択した場合に、システムによって自動的に表示されるページを制御します。ユーザーはこのページで、自分自身のパスワード、秘密の質問、および電子メールアドレスを変更できます。

エンドユーザーフォームをカスタマイズして、電話番号、住所、事務所の地理的な場所などを扱うフィールドをユーザーが制御できるように権限を与えることができます。たとえば、ユーザーが、Identity Manager ページを通じて追加の NT グループへのアクセスを要求するためにフィールドを追加できます。

承認フォーム

承認フォームは、ユーザー要求の承認者を指定するときに、リソース、ロール、または組織の所有者に表示される情報を制御します。デフォルトでは、このページには、プロセスを開始した管理者の名前を示す、一連の読み取り専用フィールドが表示されます。また、アカウント ID、ロール、組織、電子メールアドレスなど、ユーザーに関する情報も表示されます。

このフォームは、ユーザーの作成前に、ユーザーの値を変更する最後の機会をリソースの所有者に提供できるようにします。デフォルトでは、ユーザーを承認すると、読み取り専用フィールドにすべてのユーザー属性が表示されます。

承認フォームは、次の処理を行うようにカスタマイズできます。

- ユーザーに関する情報の追加、削除。
- ユーザーの初期フォームに入力された情報を変更するため、この情報を編集する権限の承認者への割り当て。
- 異なる目的に合わせて独自の承認フォームを作成。たとえば、管理者やリソース所有者がアカウント作成またはユーザー削除を開始するときに使用される専用の承認フォームを作成できます。

フォームが機能するしくみ

ブラウザによるフォームの表示には、さまざまな要因が影響します。ただし、ブラウザでのフォームの動作は、主に次の要素によって決定されます。

- **フォームに関連付けられているビュー**。すべてのフォームは、ビューとともに使用されます。フォームとともに使用されるもっとも一般的なビューは、ユーザービューです。ビューは、フォームが評価されるときに使用できるデータを定義します。
- **未定義の属性**。Tabbed User Form は、フィールドが明示的に定義されていないリソースアカウント属性を編集するためのテキストフィールドを自動的に生成するメカニズムを提供します。この機能は、フォームで無効にすることができます。

- **フォームとその他の Identity Manager コンポーネントとの連携。**これには、Identity Manager がフォームまたは フォーム評価 を評価するためのプロセスが含まれます。フォームによって駆動されるすべてのページは、同様に処理されます。Identity Manager がフォームを評価する方法については、この章の「[フォームの評価](#)」を参照してください。
- **フォームで使用される表示コンポーネント。**フォームフィールドには、ブラウザにフィールドをどのように表示するかを決定する、表示コンポーネントを関連付けることができます。

ユーザービューとフォーム

ユーザービューは、Identity Manager ユーザーの利用可能な全情報が含まれるデータ構造です。このビューには、次の項目が含まれます。

- Identity Manager のリポジトリに格納される属性
- リソースアカウントからフェッチされる属性
- リソース、ロール、組織など、ほかのソースから取得される情報

ビューには多くの属性があり、ビュー属性はビュー内にある名前を付けられた値（たとえば、`waveset.accountId` は、Identity Manager のアカウント名が値となるユーザービュー中の属性）です。

ほとんどのフォームフィールドの名前は、ビュー属性と関連付けられています。フォームフィールドの名前として、ビュー属性の名前を指定することで、フィールドとビュー属性を関連付けます。詳細については、「[フィールド名の定義](#)」を参照してください。

ユーザービューの全属性の参照情報など、ユーザービューの詳細については、「[Identity Manager のビュー](#)」を参照してください。

未定義の属性

管理者インタフェースでユーザーにリソースやロールを割り当てると、表示が更新されます。新しいリソースアカウントの属性は、ユーザービューで定義されます。

Tabbed User Form の `<FormRef name = 'Missing Fields' />` は、対応するフィールドがフォームに明示的に定義されていないリソースアカウント属性について、テキストフィールドを生成するようにフォーム生成ツールに伝えます。この機能を無効にするときは、Tabbed User Form から `<FormRef name = 'Missing Fields' />` を削除します。

フォームの評価

システムがフォームをどのように処理するかを理解することは、ブラウザでのフォームの動作を決定するときに役立ちます。フォームによって駆動されるすべてのページは、次のように同様に処理されます。

1. ページは、Identity Manager のユーザーインタフェースまたは管理者インタフェースから要求されます。
2. インタフェースは、サーバーからビューを要求します。ビューは、編集が可能な、名前が付けられた値の集合です。それぞれのビューは、ビューの値をユーザーにどのように表示するかを定義するフォームと関連付けられます。
3. サーバーは、リポジトリ内の 1 つまたは複数のオブジェクトからデータを読み込むことで、ビューを構築します。ユーザービューの場合は、リソースアダプタを通じてリソースからアカウント属性も取得されます。
4. Derivation 式が評価されます。これらの式は、リソースからの暗号化され、エンコードされた値を、ユーザーにとって意味のある値に変換するときに使用されます。この Derivation 式は、フォームが最初に読み込まれるとき、あるいは、1 つまたは複数のリソースからデータがフェッチされるときに評価されます。
5. Default 式が評価されます。フィールドが NULL の場合は、フィールドにデフォルト値が設定されます。
6. HTML コードが生成されます。システムはビューデータとフォームを処理し、HTML ページを生成します。この処理では、式に含まれる allowedValues プロパティが評価され、Select または MultiSelect HTML コンポーネントが構築されます。
7. ブラウザにページが表示されます。ユーザーは、表示される値を編集できます。編集時にユーザーがフィールドを修正すると、多くの場合は、それによってページの表示が更新されたり、再計算されたりします。これにより、ページが再生成されます。ただしこの時点では、システムは編集後のデータをリポジトリにまだ格納しません。
8. 変更した値が、ビューに同化されます。表示更新イベントが発生した場合は、インタフェースはブラウザで編集されたすべてのフォームフィールドの値を受信します。
9. Expansion 式が評価されます。これにより、追加の値がビューに配置される場合があります。Expansion 規則は、ページを再計算するとき、またはフォームを保存するときに実行されます。
10. ビューの表示が更新されます。インタフェースは、ビューの表示を更新し、編集された値の現在の設定を提供するようにサーバーに要求します。サーバーは、リポジトリまたはリソースからデータを読み込んで、より多くの値をビューに挿入する場合があります。
11. Derivation 式が評価されます。通常は、ビューの表示を更新するときには、Derivation 式は評価されません。一部の複雑な事例では、表示の更新後にシステムが導出を要求する場合があります。

12. システムは、表示が更新されたビューとフォームを処理し、**ブラウザに返される別の HTML ページを構築します**。ユーザーは、表示更新の結果を確認し、編集を継続します。ユーザーは、変更を保存またはキャンセルするまで、任意の回数だけビューの表示を更新できます (毎回、手順 7 ~ 12 を繰り返す)。
13. A. 編集をキャンセルすると、ビューに取り込まれたすべてのデータは破棄され、サーバーにそれが通知されます。これにより、サーバーは任意のリポジトリロックを解除し、別のページへのパスを制御できます。
14. B. 編集を保存すると、インタフェースは変更された値を受信し、それをビューに同化させます (手順 8 を参照)。
15. **Validation 式**が評価されます。フィールドの値が必要な仕様を満たしていない場合は、エラーが示され、フィールドの値を修正できます。変更を加えると、プロセスは手順 13 に戻ります。
16. 最後にもう一度、**Expansion 式**が評価されます (手順 9 を参照)。
17. サーバーがビューを保存すると、通常は、**リポジトリ内の 1 つまたは複数のオブジェクトが修正されます**。ユーザービューでは、リソースアカウントも更新される場合があります。

これらの手順の一部は、フォームのすべてのフィールドに対して繰り返す必要があります。これには、Derivation 式の評価、Default および Validation 式の評価、HTML の生成、および Expansion 式の評価が含まれます。すべてのフィールドに対する繰り返しでは、このフィールドの処理が必要であるかどうかを判断するために、Disable 式が評価されます。Disable 式の評価結果が **true** であれば、そのフィールド (および、入れ子になっているすべてのフィールド) は無視されます。これら特別な種類の式の詳細については、この章の「**フィールド名の定義**」を参照してください。

フォームのカスタマイズ

Identity Manager 製品のデフォルトの動作を理解すると、カスタマイズが必要なページを特定できます。

1. 編集可能なページと、それに対応するフォームについては、「**編集されたフォーム**」の節を参照してください。
2. フォームを編集するときは、**Identity Manager IDE** を起動し、「リポジトリオブジェクトを開く」を選択します。表示されるポップアップダイアログから、編集するフォームを選択します。

ここで説明する内容は次のとおりです。

- **カスタマイズの概要**
- **カスタマイズに関連するその他のトピック**

- フォームの構造
- フォームフィールドとは
- フィールドの表示を定義するフォームの作成に関するガイドライン
- フォームフィールド内の式の最適化
- 新しいリソースとユーザーの自動リンク設定の無効化
- 結果ページでのクリアテキストによる属性の表示の回避
- フォームへのガイダンスヘルプの追加

カスタマイズの概要

フォームをカスタマイズして、よりわかりやすくし、表示特性を変更したり、フィールドデータを処理するロジックを含めたりすることができます。

基本的な手順

Identity Manager システムでフォームをカスタマイズする基本的な手順は次のとおりです。

- **カスタマイズするフォームを選択します。** カスタマイズするフォームを特定する方法を指定します。
- **フォームを編集し、保存します。** 製品に付属する、エンドユーザーと管理者のデフォルトのフォームの変更について、基本的な情報を示します。
- **変更をテストします。** 運用環境に読み込む前に変更をテストし、エラーのログ記録を有効にするためのガイドラインを示します。

一般的なタスク

フォームを編集するときは、通常、次のタスクを実行します。

- **フォームのフィールドを追加、削除します。** 一般的なタスクには、一部のデフォルトフィールドの削除と、環境に合わせてカスタマイズされたフィールドの追加が含まれます。
- **フィールドをフォームにどのように表示するかを定義します。** この操作には、Identity Manager に付属する HTML コンポーネントのライブラリを使用する必要があります。フィールドの表示特性の編集については、「[フィールドの表示プロパティ](#)」の節を参照してください。
- **フィールドの値を定義する論理式を設定します。** この操作では、XPRESS 言語を使用して論理式を作成する必要があります。XPRESS の使用については、「[XPRESS 言語](#)」を参照してください。

表 3-2 フォームの要素

プロパティ	説明
Title	フォームフィールドの横に表示されるテキストを指定します。
Class	要素が属す HTML 表示クラスを指定します。
Required	フォームの処理にその要素が必須であるかどうかを指定します。送信の時点では、このフィールドに NULL 以外の値が入力されている必要があります。必須に設定すると、フィールドの右に赤いアスタリスクが表示されます。フォームの下部にはメッセージテキストが表示され、フィールドの横の赤いアスタリスクが、送信時にフィールドへの値の入力が必須であることを示していることを通知します。
Action	これを設定すると、変更によって、すべての Select コントロールまたは MultiSelect コントロールの表示が更新されます。 Identity Manager の管理者インタフェースでは、この設定によって、基本となるビューの表示が更新されます。ロールの選択は、この動作のよい例です。Tabbed User Form で新しいロールを選択すると、編集セッション中にそのロールを通じて割り当てられるリソースを反映して、ビューの表示が更新されます。ビューの表示が更新されると、新たに割り当てられたリソースのリソースアカウント属性を明示的に設定できるようになります。
No New Row	フォームのレイアウトに限定して使用されます。true に設定すると、そのフィールドが直前のフィールドの右に強制的に表示されます。たとえば、Name フィールドでは、この設定が便利です。ユーザーは、姓、名、およびミドルネームのイニシャルを上から下へではなく、右から左に入力できるようになります。
Hidden	そのフィールドをユーザーに表示しないことを指定します。このようなフィールドは、姓と名を連結してフルネームを構築する場合のように、通常は別のフィールドから計算される属性値の設定に使用されます。
Title	コントロール (テキストボックス) の文字の幅を指定します。
Class	コントロールバッファ (テキストボックス) の文字の幅を指定します。ユーザーが、size プロパティに指定した値より長い文字列を入力した場合は、文字がスクロールされます。
Required	このフォームフィールドの名前を指定します。通常は、このフォームで使用されるビューのパス式が指定されます。

次の特性は、メインタブビューから設定します。

表 3-3 メインタブビューから設定する特性

フィールド	説明
Name	このフィールドの名前を入力します。多くの場合、フィールド名は、このフォームで使用されるビューのパス式です。テキストボックス、チェックボックス、選択フィールドなどの編集コンポーネントとして表示されるすべてのフィールドには、ビューのパスを示す名前が必要です。SectionHead や Javascript のように編集コンポーネントとして表示されないフィールドは、名前を必要としません。ただし、フィールド参照を通じて別のフォームから参照する必要がある場合は、編集コンポーネント以外のフィールドにも名前を付けることができます。
Title	そのフィールドのタイトルを指定します。このタイトルは、フォーム上のフィールドの横に表示されます。このフィールドの横にあるドロップダウンメニューから、この要素のデータ型を選択してください。このフィールドに表示されるテキストを編集するには、横に表示される「 編集 」ボタンをクリックします。
Sub Title	(省略可能) フォームタイトルの下に表示されるテキストを指定します。このフィールドの横にあるドロップダウンメニューから、この要素のデータ型を選択してください。このフィールドに表示されるテキストを編集するには、横に表示される「 編集 」ボタンをクリックします。
Help Catalog	フィールドにガイダンスヘルプを関連付けるヘルプキーを指定します。この値は、フォームによって指定される、関連するヘルプカタログに含まれるエントリの名前です。ヘルプキーを指定すると、フィールドの左にアイコンが表示されます。このアイコンにカーソルを合わせると、ヘルプカタログから参照されるテキストが表示されます。
Base Context	(標準のユーザーフォームでは、通常、使用されない) すべてのフィールドに完全パスを指定する必要性をなくすためのベースコンテキストを指定します。ベースコンテキストは、基本となるマップを特定します。具体的には <code>com.waveset.object.Genericobject</code> で、通常は <code>user</code> または <code>userview</code> という名前が付けられます。Identity Manager の管理者インタフェースでは、編集コンテキストはユーザーであり、ベースコンテキストの参照は空白のまま残されます。承認のように、手動アクションによって呼び出されるフォームでは、フォームのコンテキストはワークフローコンテキストとなります。

表 3-3 メインタブビューから設定する特性 (続き)

フィールド	説明
Options	フィールドの、1 つまたは複数の表示オプションを選択します。 Required - フォームの処理にその要素が必須であるかどうかを指定します。送信の時点では、このフィールドに NULL 以外の値が入力されている必要があります。必須に設定すると、フィールドの右に赤いアスタリスクが表示されます。フォームの下部にはメッセージテキストが表示され、フィールドの横の赤いアスタリスクが、送信時にフィールドへの値の入力が必須であることを示していることを通知します。 Button - フォーム下部の 1 つの水平行にフィールドを表示します。このオプションを選択しない場合、フィールドはフォームの次の行に表示されます。ほとんどの場合、このオプションは Button 表示クラスを使用するフィールドに設定されます。 Action - これを設定すると、変更によって、すべての Select コントロールまたは MultiSelect コントロールの表示が更新されます。 Identity Manager の管理者インタフェースでは、この設定によって、基本となるビューの表示が更新されます。ロールの選択は、この動作のよい例です。Tabbed User Form で新しいロールを選択すると、編集セッション中にそのロールを通じて割り当てられるリソースを反映して、ビューの表示が更新されます。ビューの表示が更新されると、新たに割り当てられたリソースのリソースアカウント属性を明示的に設定できるようになります。 Library - 宣言したときではなく、参照した場合にのみフィールドを表示することを指定します。このオプションは、フォームで評価されるフィールドの順序が、ユーザーに表示されるフィールドの順序と異なる場合に便利です。
Default	フィールドのデフォルト値を計算するための式を指定します。このフィールドの現在の値が NULL の場合は、フォームの表示前に Default 式が呼び出されます。
Derivation	表示前にフィールドの値を計算するための式を指定します。これは Default 式に似ていますが、現在のフィールド値が NULL 以外でも、この式は評価されます。Derivation 式は、フォームが最初に表示される前に評価され、その後、フォームの表示が更新されるたびに評価されます。
Validation	フォームに入力した値が有効であるかどうかを判断するためのロジックを指定します。Validation 式は、成功を示す場合は NULL を返し、失敗を示す場合は判読可能なエラーメッセージを含む文字列を返します。

表 3-3 メインタブビューから設定する特性 (続き)

フィールド	説明
Expansion	フォームの送信後にフィールドの値を計算するための式を指定します。多くの場合、Expansion 式は非表示のマークが付けられたフィールドで使用されます。非表示フィールドは、ユーザーが直接編集することができないため、値は Expansion 式を使用して計算できます。「 フィールドの非表示 」を参照してください。
Disable	true と評価された場合に、フィールドと、その入れ子フィールドを無効にする式を指定します。無効化されたフィールドは、フォームに表示されません。このオプションは、ユーザーが特定タイプのリソースを持っているかどうかを判断するときに使用されます。ユーザーがこのようなリソースを持っている場合は、フォームにはそのリソースに適したフィールドが表示されます。
Display Class	このフォームコンポーネントをブラウザに表示するときに使用される HTML コンポーネントクラスを指定します。デフォルトでは、EditForm 表示クラスが選択されます。フォームがエンドユーザーメニューのようなリンクフォームの場合は、Display Class オプションから LinkForm を選択します。 「 HTML 表示コンポーネント 」の HTML 表示クラスの表を参照してください。
size	コントロール (テキストボックス) の文字の幅を指定します。
maxLength	この要素の最大文字数を指定します。

ヒント	多くの場合、フィールド名はこのフォームで使用するビューのパス式で、通常はリソースの特定の属性と関連付けられます。リソースとその属性のリストを表示するには、「 リソースの表示 」をクリックします。展開可能なリソースタイプのツリーを示す「リソースの表示」ダイアログが表示されます。リソースインスタンスと、その属性名のリストを表示するときは、リソースタイプの名前をクリックします。新しいフォームフィールドの名前として、リソースの属性名を使用するときは、リソースの属性名をクリックし、「OK」をクリックします。これにより、Name フィールドに属性名が挿入されます。

表 3-4 表示クラスのオプション

HTML コンポーネント	目的
Apple	ページにアプレット参照を挿入します。
BackLink	前のページに戻るリンクを表示します。
BorderedPanel	コンポーネントを、東西南北と中央の 5 つの領域に配置するコンテナ。
Button	ボタンを表示します。
ButtonRow	それぞれの間に隙間を設けて、コンポーネントを水平の行に配置するコンテナ。通常は、 Button コンポーネントの行の表示に使用されます。
CheckBox	それぞれの間に隙間を設けて、コンポーネントを水平の行に配置します。通常は、 Button コンポーネントの行の表示に使用されます (Container)。
DatePicker	ページにカレンダーのアイコンを表示します。ユーザーは、このアイコンをクリックして日付を選択し、ページフィールドに値を適用できます。
EditForm	フォームのデフォルトコンテナ。1 つの列にコンポーネントのタイトルを表示し、もう 1 つの列にコンポーネントを表示します。各行の背景は、交互に灰色と白になります。
FileUpload	アップロードするファイルの名前を指定するための、 Text コンポーネントの形式。
Hidden	表示されないデータを HTML ページに含めるときに使用されるコンポーネント。
Html	事前にフォーマットされている HTML をページに挿入します。
Javascript	JavaScript 関数を定義します。
Label	読み取り専用のテキストを表示します。
Link	ページにリンクを配置します。
LinkForm	縦の箇条書きリストに、タイトルなしでコンポーネントを配置します。通常は、 Link コンポーネントのリストを含むページで使用されます。 EditForm コンテナの代わりに使用されます (Container)。

表 3-4 表示クラスのオプション (続き)

HTML コンポーネント	目的
MultiSelect	複数の選択項目を持つボックスを表示します。これは、1 つのボックスに含まれる定義済みの値セットを選択ボックスに移動できる、2 つの部分から構成されるオブジェクトです。
NameValueTable	ベージュ色の背景を持つ簡易テーブルに、名前と値のペアのリストを表示します。
Panel	コンポーネントを水平の行、または垂直の列に配置します (Container)。
Radio	1 つまたは複数のラジオボタンの水平リストを表示します。ユーザーが一度に選択できるラジオボタンは 1 つだけです。コンポーネントの値が NULL の場合、または許可される値のいずれとも一致しない場合は、ボタンは選択されません。
SectionHead	セクションの見出しを表示します。これは EditForm コンテナで認識され、タイトルとコンポーネントの列を太字で表示します。
Select	1 つの項目を選択できるリストボックスを表示します。
SimpleTable	列のタイトル行を持つ、簡単なグリッドにコンポーネントを配置します。
SubTitle	フォームタイトルの下に表示されるテキストを指定します。
Text	読み取り専用のテキストを表示します。
TextArea	ページにリンクを配置します。
Title	フォーム上部に表示されるテキストを指定します。

表 3-5 フォームの要素

フォーム要素	説明
Name	このフィールドの名前を入力します。多くの場合、フィールド名は、このフォームで使用するビューのパス式です。テキストボックス、チェックボックス、選択フィールドなどの編集コンポーネントとして表示されるすべてのフィールドには、ビューのパスを示す名前が必要です。SectionHead や Javascript のように編集コンポーネントとして表示されないフィールドは、名前を必要としません。ただし、フィールド参照を通じて別のフォームから参照する必要がある場合は、編集コンポーネント以外のフィールドにも名前を付けることができます。
Title	そのフィールドのタイトルを指定します。このタイトルは、フォーム上のフィールドの横に表示されます。このフィールドの横にあるドロップダウンメニューから、この要素のデータ型を選択してください。このフィールドに表示されるテキストを編集するには、横に表示される「編集」ボタンをクリックします。
Help Key	フィールドにガイダンスヘルプを関連付けるヘルプキーを指定します。この値は、フォームによって指定される、関連するヘルプカタログに含まれるエントリの名前です。ヘルプキーを指定すると、フィールドの左にアイコンが表示されます。このアイコンにカーソルを合わせると、ヘルプカタログから参照されるテキストが表示されます。

表 3-5 フォームの要素 (続き)

フォーム要素	説明
Options	フィールドの、1 つまたは複数の表示オプションを選択します。 Required - このフィールドへの入力または選択が、フォームを処理する上で必須であるかどうかを指定します。 Button - フォーム下部の 1 つの水平行にフィールドを表示します。このオプションを選択しない場合、フィールドはフォームの次の行に表示されます。ほとんどの場合、このオプションは Button 表示クラスを使用するフィールドに設定されます。 Action - これを設定すると、変更によって、すべての Select コントロールまたは MultiSelect コントロールの表示が更新されます。Identity Manager の管理者インタフェースでは、この設定によって、基本となるビューの表示が更新されます。ロールの選択は、この動作のよい例です。Tabbed User Form で新しいロールを選択すると、編集セッション中にそのロールを通じて割り当てられるリソースを反映して、ビューの表示が更新されます。ビューの表示が更新されると、新たに割り当てられたリソースのリソースアカウント属性を明示的に設定できるようになります。 Library - 宣言したときではなく、参照した場合にのみフィールドを表示することを指定します。このオプションは、フォームで評価されるフィールドの順序が、ユーザーに表示されるフィールドの順序と異なる場合に便利です。
Default	フィールドのデフォルト値を計算するための式を指定します。このフィールドの現在の値が NULL の場合は、フォームの表示前に Default 式が呼び出されます。
Derivation	表示前にフィールドの値を計算するための式を指定します。これは Default 式に似ていますが、現在のフィールド値が NULL 以外でも、この式は評価されます。Derivation 式は、フォームが最初に表示される前に評価され、その後、フォームの表示が更新されるたびに評価されます。
Validation	フォームに入力した値が有効であるかどうかを判断するためのロジックを指定します。Validation 式は、成功を示す場合は NULL を返し、失敗を示す場合は判読可能なエラーメッセージを含む文字列を返します。Validation 規則は、フォームの送信時にのみ評価され、表示を更新したり、再計算したりした場合には評価されません。

表 3-5 フォームの要素 (続き)

フォーム要素	説明
Expansion	フォームの送信後にフィールドの値を計算するための式を指定します。多くの場合、 Expansion 式は非表示のマークが付けられたフィールドで使用されます。非表示フィールドは、ユーザーが直接編集することができないため、値は Expansion 式を使用して計算できます。
Disable	true と評価された場合に、フィールドと、その入れ子フィールドを無効にする式を指定します。無効化されたフィールドは、フォームに表示されません。このオプションは、ユーザーが特定タイプのリソースを持っているかどうかを判断するときに使用されます。ユーザーがこのようなリソースを持っている場合は、フォームにはそのリソースに適したフィールドが表示されます。
Display Class	このフォームコンポーネントをブラウザに表示するときに使用される HTML コンポーネントクラスを指定します。デフォルトでは、 EditForm 表示クラスが選択されます。フォームがエンドユーザーメニューのようなリンクフォームの場合は、 Display Class オプションから LinkForm を選択します。 「 HTML 表示コンポーネント 」の HTML 表示クラスの表を参照してください。
value	プロパティの属性を指定します。通常は文字列です。
maxLength	この要素の最大文字数を指定します。

表 3-6 フォームツールボックスのデフォルトサービス

サービス	説明
Text	通常のテキスト入力ボックスを表示します。
Secret Text	テキストをアスタリスク (*) として表示します。通常は、パスワードのように暗号化されるデータで使用されます。
Select	1 つの項目を選択できるリストボックスを表示します。このリストボックスの値は、 allowedValues プロパティで指定します。

表 3-6 フォームツールボックスのデフォルトサービス (続き)

サービス	説明
MultiSelect	複数の選択項目を持つテキストボックスを表示します。これは、1つのボックスに含まれる定義済みの値セットを選択ボックスに移動できる、2つの部分から構成されるオブジェクトです。左のボックスの値は <code>allowedValues</code> プロパティーで指定します。多くの場合、この値は <code>FormUtil.getResources</code> などの Java メソッドを呼び出すことで動的に取得されます。複数選択ボックスの右側部分に表示される値には、フィールド名によって特定される関連ビュー属性の現在値が適用されます。
Checkbox	チェックボックスを表示します。チェックマークが付けられると、そのボックスの値は <code>true</code> となります。選択されていないボックスの値は <code>false</code> となります。
Label	複数行のテキスト入力ボックスを表示します。
TextArea	1つまたは複数のラジオボタンの水平リストを表示します。ユーザーが一度に選択できるラジオボタンは1つだけです。コンポーネントの値が <code>NULL</code> の場合、または許可される値のいずれとも一致しない場合は、ボタンは選択されません。
Radio	ページにリンクを配置します。
Link	ボタンを表示します。
Button	このフォームで使用されるビューによって定義される変数を参照します。
accountId	複数行のテキスト入力ボックスを表示します。

カスタマイズに関連するその他のトピック

ここでは、次のトピックについて説明します。

- フォームの構造
- フォームのコンポーネント
- フィールドの定義
- フォームの作成に関するガイドライン

フォームの構造

フォームは、XML オブジェクトとして **Identity Manager** リポジトリに格納されます。各フォームは、次の構造を持つ独自のオブジェクトとして格納されます。

注 フォームの XML 構造を理解している必要はありません。Identity Manager IDE を使用することで、フォームの構造を簡単に操作できます。次に示す情報は、参照のみを目的としています。

次のスタブフォームは、フォームの一般的な構造を示しています。

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE Configuration PUBLIC 'waveset.dtd' 'waveset.dtd'>
<!-- id="#ID#UserForm:EndUserMenu" name="End User Menu"-->
  <Configuration id="#ID#UserForm:EndUserMenu" name='End User Menu'
createDate='1012185191296' lastModifier='Configurator'
lastModDate='1013190499093' lastMod='44' counter='0'
wstype='UserForm'>
  <Extension>

  <Form name='End User Menu'>
    <Display class='LinkForm'>
      <Property name='title' value='User Self Service'/>
      <Property name='subtitle' value='Select one of the
following          options' />
    </Display>

    ... フィールドコンテンツ

  </Form>
</Extension>
<MemberObjectGroups>
  <ObjectRef type='ObjectGroup' id='#ID#Top' name='Top' />
</MemberObjectGroups>
</Configuration>
```

注	Identity Manager のユーザーインタフェースページは、ナビゲーションバーを含む 2 番目の XPRESS フォームを実装しています。これは、描画されるページに 2 つの <FORM> タグがあり、各タグで name 属性の設定が次のように異なることを意味します。 <form name="endUserNavigation"> および <form name="mainform">

フォームのコンポーネント

次の表は、フォームに表示される順序でフォームコンポーネントを示しています。各フォームコンポーネントの詳細については後述します。

表 3-7 フォームのコンポーネント

フォームコンポーネント	目的
ヘッダー	フォームのオブジェクト定義に関する情報を示します。 <Form>、<Extension>、および <Configuration> 要素の開始タグが含まれ、フォームのプロパティ（フォームの呼び出し時に表示される title、subtitle、titleWidth など）を定義します。
フォーム本文	フィールド定義、フォーム関数、およびフォーム変数が含まれます。これは、フォームの中で編集する部分です。
フッター	<Form>、<Extension>、および <Configuration> 要素の終了タグが含まれます。

ヘッダー

フォームヘッダーに含まれる内容は次のとおりです。

- この XML ファイルと関連する DTD を含む、XML 宣言やドキュメント宣言など、XML ファイルに含める標準の導入情報。前述の例には、次のような導入情報が含まれます。

```
<?xml version='1.0' encoding='UTF-8'?>  
  
<!DOCTYPE Configuration PUBLIC 'waveset.dtd' 'waveset.dtd'>
```

このシステムは、ファイルにこの情報を追加します。編集しないでください。

- <Extension> および <Configuration> 要素の開始タグ。フォームの外観と動作を表現する HTML コンポーネントは、これらの要素で囲まれます。
Configuration 要素には、フォームオブジェクトのプロパティを表現する属性が含まれます。

ヘッダーには、作成日、ファイルを最後に修正したユーザーのログイン、フォームタイプなどの内部識別情報はじめとする、フォームに関する情報が含まれます。この情報は、通常、ページプロセッサによって生成されます。

注 システムが生成する次の情報は、内部使用限定です。これらの属性は編集しないでください。

表 3-8 フォームヘッダーのコンポーネント

要素	定義	構文 / 例
<Extension>	<Form> 要素を囲むために使用されます。	<Element>...</Element>
<Configuration>	最後の修正日や、このフォームを最後に修正したユーザーのログインなど、フォームオブジェクトを処理するときにシステムが内部的に使用する情報が含まれます。これらの情報のほとんどは、通常、Identity Manager リポジトリに格納されているいずれかの持続オブジェクトと関連付けられています。多くの場合、この情報を編集する必要はありません。	Configuration id='#ID#UserForm:EndUserMenu' name='End User Menu' createDate='1012185191296' lastModifier='Configurator' lastModDate='1013190499093' lastMod='44' counter='0' wstype='UserForm'>

フォーム本文

フォーム本文は、次の内容から構成されます。

- title、subtitle、width などのフォームプロパティ。これらのプロパティは、Form Properties というテーブルに定義されます。
- Field 要素。これは、製品インタフェースでユーザーに表示するフィールドの外観と機能の設定に使用されます。フィールドには、情報を計算するための XPRESS ロジックも含まれます。XPRESS 言語の使用の詳細については、「XPRESS 言語」を参照してください。

次の表は、フォームヘッダーのプロパティを示しています。

表 3-9 フォームヘッダーのプロパティ

プロパティ	目的
title	フォーム上部に表示されるテキストを指定します。多くの場合、このタイトルは、画面のその他のフォントより大きな太字で表示されます。フォームの title は、Identity Manager ページの下に表示されます。タイトルの表示特性は編集できません。 「フォームのコンポーネント」の節で示した例では、title の値は User Self Service です。
subtitle	このフォームによって定義されるページのフォームタイトルの下に表示されるテキストを指定します。タイトルの表示特性は編集できません。 前述の例では、subtitle の値は Select one of the following options です。
titleWidth	ブラウザウィンドウのタイトルの幅を、ピクセル単位で指定します。 例 <pre><Display> <Property name='titleWidth'> <Integer>120</Integer> </Property> </Display></pre>

次の表は、フォーム本文で使用するすべての要素を示しています。

表 3-10 フォーム本文で使用する要素

コンポーネント	定義	例
defun	XPRESS 関数を定義します。この要素は、フォーム内のどのフィールド要素からも呼び出すことができます。	<pre><defun name='add100'> <def arg name='x' /> <add><i>x</i><i>100</i> </add> </defun></pre>

表 3-10 フォーム本文で使用される要素 (続き)

コンポーネント	定義	例
defvar	計算結果を保持するための、XPRESS 変数を定義します。	<pre><defvar name='nameLength' <length> <ref>fullname</ref> </length> </defvar></pre>
Display	フィールドの外観を定義する表示コンポーネントを指定します。詳細については、「 Display 要素 」の節を参照してください。	<pre><Display class='LinkForm'> <Property name='title' value='User Self Service' /> <Property name='subtitle' value='Select one of the following options' /> </Display></pre>
Field	フォーム本文で使用される主要要素。詳細については、「 Field 要素 」の節を参照してください。	<pre><Field name='fullname' /></pre>
FieldRef	含まれるフォームに定義されているフィールドへの参照を設定します。	<pre><FieldRef name='fieldName' /></pre>
Include	別のフォームオブジェクトへの参照を設定します。現在のフォームにこのコンポーネントを含めると、そのフォームに定義されるフィールドを参照し、表示できるようになります。	<pre><Include> <ObjectRef type='UserForm' id='#ID#UserForm:UserFormLibrary' /> </Include></pre>
FormRef	別のフォームオブジェクトへの参照を設定します。	<pre><FormRef name='formName' /></pre>
Namespace	ビューへのショートカットを定義するときに使用されます。フィールド名と参照に、長い名前の代わりに短縮名を使用できます。代替名を使用するときは、名前の後にコロン(:) を続けます。	<pre><Namespace name='w' value='waveset' /></pre>

Form 要素

<Form> 要素は、すべての Field 要素を囲み、フォームの一意の名前を含みます。前のページに示される要素は、Form の開始タグと終了タグの間に記述されます。

```
<Form name='Create User Form'
  <Field name='waveset.accountId'>
    additional fields
  </Field>
</Form>
```

別の例：

```
<Form name='Task Launch Form'>
  <Display class='EditForm'>
    <Property name='title' value='Task Launch' />
    <Property name='subTitle' value='Enter task launch
parameters' />
  </Display>
  ...
</Form>
```

Display 要素

Form 要素内の Display 要素は、フォームの表示に使用されるコンポーネントを表現します。デフォルトでは、この Display 要素は EditForm コンポーネントで使用されます。Form コンポーネントクラスの変更が必要となることはあまりありませんが、コンポーネントのプロパティは設定可能です。もっとも多く指定される 2 つのプロパティは title と subTitle です。

EditForm は、隣接するフィールドのタイトルの幅を設定する adjacentTitleWidth プロパティもサポートします。このプロパティが定義されていない場合は、デフォルト値のゼロが適用されます。

adjacentTitleWidth の値をゼロに設定すると、列のタイトルの大きさは自動的に変更されます。ゼロ以外の値を設定した場合は、隣接する列 (たとえば、2 番目と 3 番目の列) のタイトルの幅は adjacentTitleWidth の値に設定されます。

```
<Form name='Default User Form' help='account/modify-help.xml'>
  <Display class='EditForm'>
    <Property name='titleWidth' value='120'>
    <Property name='adjacentTitleWidth' value='60'>
  </Display>
```

Field 要素

Field 要素は、フォーム本文で使用される主要要素です。フィールドは、ユーザーの各属性の定義に使用されます。Field 要素を使用して、フォームフィールドに XPRESS ロジックを含めることができます。フォームの Field 要素の操作の詳細については、「フィールドの定義」の節を参照してください。

次の例は、Email Address というラベルを持つ編集フィールドを作成します。

```
<Field name='waveset.email'>
  <Display class='Text'>
    <Property title='Email Address' />
    <Property size='60' />
    <Property maxLength='128' />
  </Display>
  ...
</Field>
```

編集フィールドの名前は、多くの場合、フォームで使用されるビュー内のパス式です。この例で waveset.email は、Identity Manager リポジトリ内のユーザーオブジェクトに関連付けられている電子メールアドレスを参照します。

フッター

フッターには、フォームが関連付けられている Identity Manager オブジェクトグループまたは組織に関する情報が含まれます。また、</Form>、</Extension>、および </Configuration> 要素の終了タグ、またはヘッダー内で開始したその他の要素の終了タグも含まれます。前述の例には、次のようなフッターが含まれます。

```
</Form>
</Extension>
  <MemberObjectGroups>
    <ObjectRef type='ObjectGroup' id='#ID#Top' name='Top' />
  </MemberObjectGroups>
</Configuration>
```

<MemberObjectGroups> は、システムがオブジェクトを格納するオブジェクトグループまたは組織を特定します。オブジェクトグループを指定しない場合は、システムはデフォルトの組織である Top にオブジェクトを割り当てます。多くの場合、フォームを含む Configuration オブジェクトは、次の構文で All グループに含まれます。

```
<MemberObjectGroups>
```

```
<ObjectRef type='ObjectGroup' name='All' />
</MemberObjectGroups>
```

フォームフィールドとは

フォーム本文には、Web ページの各要素の外観と動作を定義する Field 要素が含まれます。各 Field 要素には、それぞれが独自の表示コンポーネントを持つその他のフィールドを含めることができます。

フォームフィールドは、<Field> タグのセットに囲まれた、いくつかの部分から構成されます。

- **値の式**：フィールドには、フィールドの値を計算したり、許可される値のセットを定義したりする XPRESS 式を数多く含めることができます。たとえば、フィールドのデフォルト値を定義するときは <Default> を使用し、フォームを最初に読み込んだときに計算されるフィールド値を定義するときは <Derivation> を使用します。すべてのフィールド要素が式を含むわけではありません。「[フィールド名の定義](#)」の節を参照してください。
- **HTML 表示コンポーネント**：表示コンポーネントは、可視要素をどのように表示するかを決定します。Identity Manager のフォームフィールドでは、フォームフィールドの動作と外観の設定に表示コンポーネントが使用されます (フォームには <Display> 要素で定義)。それぞれのフィールドに設定できる表示コンポーネントは 1 つだけです。これらの表示コンポーネントの詳細については、[第 8 章「HTML 表示コンポーネント」](#)を参照してください。
- **Disable 式**：Disable 式を使用することで、条件に応じて、フィールドをフォームに含めることができます。Disable 式が **true** と評価される場合は、フィールドは無視されます。

変数の作成

定数または静的データの長いリストを含む変数を作成するときは、次の構文を使用します。この構文は、静的なリストを 1 回作成し、それぞれの参照で再利用します。

```
<defvar name='states'>
    <List>
        <String>Alabama</String>
        ...
    </List>
</defvar>
```

参照のたびに新しいリストを作成するときは、<list><s>Alabama</s>...</list> という構文を使用します。

フィールドの定義

ここでは、フォームをカスタマイズするときに実行する手順について説明します。次の手順が含まれます。

- フィールド名の定義
- フィールド要素の定義
- 可視フィールドの追加
- フィールドの非表示化。フィールドを非表示に設定すると、フィールドと、その入れ子フィールドはページに表示されなくなります。ただし、フォームの処理によって、フィールドの値は設定されます。
- フィールドの無効化。フィールドを無効にすると、フィールドと、その入れ子フィールドはページに表示されなくなります。また、値の式も評価されません。無効化されたフィールドの値がすでにビューに含まれる場合、その値は変更されません。
- フィールド値の設定
- 関数の呼び出し

次に、設定するフィールド特性について、より詳しく説明します。

フィールド名の定義

リソースに定義されている属性と、Web ページに表示されるテキスト入力フィールドを一致させるには、フィールド名を使用します。リソースを定義すると、システムはリソースのアカウント属性と **Identity Manager** の属性をマップするスキーママップを設定します。たとえば、**Active Directory** リソースには、`firstname`、`lastname`、`Office Phone` などの属性が含まれる可能性があります。フォームでこれらの属性を参照するときは、**Identity Manager** スキーマに記録されている属性名と、ビューから属性へのパスがわかっている必要があります。

Field 要素の `name` 属性を定義するには、2 つの方法があります。

- `name` 属性は、多くの場合に、ユーザービュー内の属性へのパスを含みます。
- `name` 属性は、フォーム内の別のフィールド、または `FieldRef` 要素がフィールドを参照できるように、フィールドの特定に使用されます。これは、フィールドが別のフィールドのコンテナを表すように定義され、ビューのその他の属性に対応しない場合に行われます。

Field 名がビューのパス式を表すか、単なる参照名を表すかは、`Display` 要素で選択した `class` 属性の値によって決まります。編集するコンポーネントクラスの名前が **Display** クラスである場合は、その名前はビューのパス式であると考えられます。コンポーネントクラスの詳細については、「[HTML 表示コンポーネント](#)」の節を参照してください。

ビュー属性へのパス式の作成

Field 名を定義するときは、通常、ユーザービュー内の属性へのパス (パス式) を指定します。これらの属性のリストについては、「[Identity Manager のビュー](#)」を参照してください。

次のフィールド定義は、Identity Manager の電子メールアドレスを編集できるようにテキストフィールドを表示します。

```
<Field name='waveset.email'>
  <Display class='Text'>
    <Property name='size' value='60' />
  </Display>
</Field>
```

waveset.email という文字列は、Identity Manager のリポジトリに格納される電子メールアドレスをターゲットとした、ユーザービューのパス式です。

例：

このフィールド例は、特定のリソースアカウント用に定義されている電子メールアドレスを編集します。フィールド名は、アカウント内のリソースを参照します。

```
<Field name='accounts[Microsoft Exchange].email'>
  <Display class='Text'>
    <Property name='size' value='60' />
  </Display>
</Field>
```

accounts[Microsoft Exchange].email という文字列は、指定されたリソースのアカウント属性に関する情報を保持するユーザービュー内の別の場所へのパス式です。この例では、リソースの名前は Microsoft Exchange です。

例：

このフィールド例は、スキーママップの左側部分に email という属性を持つ、Identity Manager を含むすべてのリソースの電子メールアドレスを定義します。

```
<Field name='global.email'>
  <Display class='Text'>
    <Property name='size' value='60' />
  </Display>
</Field>
```

参照するフィールドの指定

フィールドに名前を付けることで、別のフィールドからそのフィールドの値を参照できます。別のフィールドからフィールド値を参照するときは、<ref></ref> のタグセットを使用します。次の例は、firstname フィールドと lastname フィールドの値を取得し、lastname, firstname のように、フィールドの文字列、カンマ、および空白文字を連結して、fullname フィールドの値を導出します。文字列の指定には、<s> タグを使用します。

```
<Field name='global.firstname'>
  <Display class='Text' />
</Field>
<Field name='global.lastname'>
  <Display class='Text' />
</Field>
<Field name='global.fullname'>
  <Expansion>
    <concat>
      <ref>global.lastname</ref><s>, </s>
      <ref>global.firstname</ref>
    </concat>
  </Expansion>
</Field>
```

すべての Field 名がビューのパス式を表すわけではありません。一部のフィールドは、別のフィールドのコンテナを表すように定義され、ビューのその他の属性に対応しません。このような場合は、FieldRef 要素によって参照できるように、Field 名を使用してフィールドを特定します。フィールドを参照する必要がある場合は、名前を指定する必要はありません。

たとえば、フォームボタンはアクションを実行しますが、値を持たず、別のフォームから参照する必要がありません。このため、フィールド名の指定も必要ありません。

```
<Field>
  <Display class='Button'>
    <Property name='label' value='Recalculate' />
```



```
<Property name='command' value='Recalculate' />
</Display>
</Field>
```

ユーザービューの詳細については、「[ユーザービューとフォーム](#)」の節を参照してください。

フィールドの表示プロパティ

Display 要素は、すべての可視フォームフィールドに共通です。Display 要素には、ブラウザに表示されるフィールドの特性を定義する Property 要素が含まれます。フィールドに **true** と評価される Disable 要素が含まれる場合を除き、フォームの Display 要素を定義すると、その要素を画面に表示できます。フィールドを再計算するフォームが画面に表示されないように設定できる場合に、別のフィールドまたは値が設定されるまでフォームを表示する、という条件を指定することができます。「[フィールドの無効化](#)」の節を参照してください。

Display

可視フィールドのクラスとプロパティを表現します。この要素は、インスタンス化するコンポーネントクラスと、そのインスタンスに割り当てるプロパティ値のセットを指定します。

```
<Display class='Text'>
  <Property name='size' value='20' />
  <Property name='maxLength' value='100' />
</Display>
```

Display 要素の class 属性は、Component クラスの名前である必要があります。デフォルトでは、これらのクラスは **Applet**、**Button**、**DatePicker** などを含み、com.waveset.ui.util.html パッケージに格納されています。デフォルトのすべてのクラスとその説明については、「[HTML 表示コンポーネント](#)」の「基本コンポーネントクラス」の節を参照してください。このパッケージに含まれないクラスを参照するときは、class 属性の完全修飾クラス名を指定します。このマニュアルで説明するすべてのクラスはデフォルトパッケージに含まれるので、修飾名を指定する必要はありません。

Property

Display 要素内に指定されます。Property の値は、コンポーネントに割り当てられるプロパティの名前と、その値を定義します。プロパティ名は、常に name 属性で指定されます。

Display 要素の Property 値の指定

Display 要素の Property の値は、次の方法で指定できます。

- value 属性
- XML オブジェクト言語
- 値を指定する式

ほとんどのプロパティの値は、value 属性を使用して、値に適切なタイプを割り当てるようにシステムに強制することで設定できます。

value 属性の使用

プロパティの値を設定するもっとも一般的な方法は、value 属性の使用です。value 属性の値は文字列として認識されますが、システムは必要に応じてコンポーネントに適したデータ型を強制します。前述の例では、size には整数値 20 が設定され、maxLength には整数値 100 が設定されています。

次の例では、SimpleTable を使用して、いくつかのサブフィールドを構成するフィールドを作成します。XML フォームでもっとも一般的に使用される Container コンポーネントは、SimpleTable と ButtonRow です。

```
<Field name='SelectionTable'>
  <Display class='SimpleTable'>
    <Property name='columns'>
      <List>
        <String>Account</String>
        <String>Description</String>
      </List>
    </Property>
  </Display>

  <Field name='accounts[LDAP].selected'>
    <Display class='Checkbox'>
      <Property name='label' value='LDAP' />
    </Display>
  </Field>

  <Field>
    <Display class='Label'>
      <Property name='text' value='Primary Corporate LDAP Server' />
    </Display>
  </Field>
</Field>
```

```

        </Display>
    </Field>

    <Field name='accounts[W2K].selected'>
        <Display class='Checkbox'>
            <Property name='label' value='Windows 2000' />
        </Display>
    </Field>

    <Field>
        <Display class='Label'>
            <Property name='text' value='Primary Windows 2000
                Server' />
        </Display>
    </Field>

    </Field>

```

Display 要素には、ゼロ個以上の Property 要素が含まれます。これらの要素は、そのコンポーネントに割り当てられるプロパティの名前と値を定義します。Property 要素の名前は、常に name 属性で指定されます。プロパティ値の設定には、多くの場合、value 属性が使用されます。value 属性の値は文字列として認識されますが、必要に応じて、コンポーネントに適したデータ型が強制されます。

XML オブジェクト言語の使用

XML オブジェクト言語を使用してプロパティ値を指定することもできます。これは、主にリストの値を指定するときに便利です。この言語の構文を使用することで、いくつかの標準 Java オブジェクトと、Identity Manager によって定義されるその他オブジェクトを表現できます。

もっとも一般的な Java XML オブジェクトは次のとおりです。

- List
- Map
- MapEntry
- String

- Integer
- Boolean
- Object

XML オブジェクト構文を使用してプロパティ値を指定するときは、要素が Property 要素内に配置されます。XML オブジェクト言語の詳細については、「[XML オブジェクト言語](#)」を参照してください。

```
<Property name='size'>
  <Integer>10</Integer>
</Property>

<Property name='title'>
  <String>New Password</String>
</Property>

<Property name='leftLabel'>
  <Boolean>true</Boolean>
</Property>

<Property name='allowedValues'>
  <List>
    <String>Texas</String>
    <String>Iowa</String>
    <String>Berkshire</String>
  </List>
</Property>
```

リストの値を認識するすべてのプロパティは、List 要素を認識します。また、ほとんどの属性は、リストを指定するカンマ区切りのリスト構文を認識します。

値を計算する式の使用

Property の値を式で指定することもできます。これにより、固定のリテラル値と、ページプロセッサによって定義される変数の値を組み合わせるなどの方法で、実行時に値を計算することができます。次に例を示します。

```
<Property name='title'>

    <concat>

        <s>Welcome </s>

        <ref>waveset.accountId</ref>

        <s>, select one of the following options.</s>

    </concat>

</Property>
```

この例では、waveset.accountId が変数を参照しています。システムがこのコンポーネントの HTML を生成するときに、ページプロセッサシステムが waveset.accountId 変数の値を供給します。参照できる変数の名前は、ページプロセッサによって定義されます。ほとんどの場合、これらの変数の定義には、XML フォームで使用される *view* が使用されます。フォームは、そのフォームを使用するビューと、そのビューによって定義される参照属性のみに注意するだけで設計できます。

Disable 要素

ブール型の値を計算します。計算結果が **true** となる場合は、現在のフォームの処理で、そのフィールドと、すべての入れ子フィールドが無視されます。

例

この例は、<Disable> 要素内の式を使用してフィールドの可視性を制御するフィールド定義を示しています。accountInfo.typeNames は、そのユーザーが割り当てられているすべてのリソースのタイプの検出に使用されます。これらのタイプは、ユーザーのすべてのリソースタイプのリストとして返されます (たとえば、Windows NT)。返されるタイプ名のリストに Solaris が含まれるときは、そのフィールドは画面に表示されます。含まれない場合は、そのフィールドは無効化されます。

```
<Field name='HomeDirectory' prompt='Home Directory'>
  <Display class='Text' />
  <Disable>
    <not>
      <contains>
        <ref>accountInfo.typeNames</ref>
        <s>Solaris</s>
      </contains>
    </not>
  </Disable>
</Field>
```

Default 要素

NULL 以外の値がフィールドに設定されていない場合にのみ、このフィールドの値として使用される値を計算します。Default は基本的には Derivation と同じですが、現在値が NULL 以外の場合にのみ値が適用されます。Default 式は、次の場合に計算されます。

- フォームを初めて読み込む
- 1 つまたは複数のリソースからデータを取得する
- フィールドの値が NULL 以外の状態で、フォームを再計算または保存する

例

この例は、文字列操作式を使用して、ユーザーの名の頭文字と姓から構成されるデフォルトのアカウント ID を返すフィールド定義を示しています。

```
<Field name='waveset.accountId'>
  <Display class='Text'>
    <Property name='title' value='AccountID' />
  </Display>
  <Default>
    <concat>
      <substr>
        <ref>accounts[Exchange].firstname</ref>
        <i>0</i>
        <i>1</i>
        <ref>accounts[Exchange].lastname</ref>
      </substr>
    </concat>
  </Default>
</Field>
```

Derivation 要素

フィールドの値を無条件で計算します。**Derivation** 式が評価されると、フィールドの現在値は常に置き換えられます。

Derivation 式は、フォームが最初に読み込まれるとき、あるいは、1 つまたは複数のリソースからデータが返されるときに計算されます。

次の例は、条件ロジックを使用して、値のセットを別のセットにマップするフィールド定義を示しています。このフィールドを処理すると、<Derivation> 要素内の式が評価され、リソースから返される場所のコードに基づいて、このフィールドに表示される説明的な値が決定されます。

```
<Field name='location'>
  <Display class='Text'>
    <Property name='title' value='Location' />
  </Display>
  <Derivation>
    <switch>
      <ref>accounts[Oracle].locCode</ref>
      <case>
        <s>AUS</s>
        <s>Austin</s>
      </case>
      <case>
        <s>HOU</s>
        <s>Houston</s>
      </case>
      <case>
        <s>DAL</s>
        <s>Dallas</s>
      </case>
      <case default='true'>
        <s>unknown</s>
      </case>
    </switch>
  </Derivation>
</Field>
```

Expansion 要素

フィールドの値を無条件で計算します。この要素は、式の評価のタイミングが Derivation 要素とは異なります。

Expansion ステートメントは、次の場合に計算されます。

- ページが再計算される
- フォームが保存される

次の例は、条件ロジックを使用して、前述の例の `location` フィールドから得られる値を、**Oracle** リソースに格納される 3 文字の略号に変換するフィールド定義を示しています。フィールド名の違いに注意してください。**location** フィールドの値は、どのリソースにも格納されません。これは、別のフィールドの計算に使用されます。

```
<Field name='accounts[Oracle].locCode'>
  <Expansion>
    <switch>
      <ref>location</ref>
      <case>
        <s>Austin</s>
        <s>AUS</s>
      </case>
      <case>
        <s>Houston</s>
        <s>HOU</s>
      </case>
      <case>
        <s>Dallas</s>
        <s>DAL</s>
      </case>
    </switch>
  </Expansion>
</Field>
```

Validation 要素

フォームに入力された値が有効であるかどうかを検証します。**Validation** 規則は、フォームを送信するたびに評価されます。

この例では、ユーザーの郵便番号が 5 桁であるかどうかを **Validation** 規則によって確認されます。

```
<Validation>
  <cond>
    <and>
      <eq><length><ref>global.zipcode</ref></length>
        <i>5</i>
      </eq>
      <gt><ref>global.zipcode</ref><i>99999</i></gt>
    </and>
    <null/>
    <s>zip codes must be five digits long</s>
  </cond>
</Validation>
```

編集フィールドとコンテナフィールド

Field 要素に指定された Display 要素は、フィールドの表示に使用されるコンポーネントを表現します。これには、次の 2 種類のフィールドが使用されます。

- **編集フィールド**：編集する特定の値に関連します。
- **コンテナフィールド**：1 つまたは複数のフィールドを囲みます。

編集フィールドは必ず名前を持ち、常に Text や Checkbox など、いずれかの編集コンポーネントとともに使用されます。

編集フィールドの例

```
<Field name='waveset.email'>
  <Display class='Text'>
    <Property title='Email Address' />
    <Property size='60' />
    <Property maxLength='128' />
  </Display>
```

```
...  
</Field>
```

編集フィールドの名前は、多くの場合、フォームで使用されるビュー内のパス式です。前述の例で `waveset.email` は、**Identity Manager** リポジトリ内のユーザーオブジェクトに関連付けられている電子メールアドレスを参照します。

コンテナフィールドは名前を持たない場合があります、常に `ButtonRow`、`SimpleTable`、`EditForm` などのいずれかのコンテナコンポーネントとともに使用されます。

よく使用されるコンテナのタイプには、1つの列にタイトル、別の列にコンポーネントを持つ **HTML** テーブルを作成する `EditForm` コンテナがあります。これらのタイトルは `title` プロパティに定義され、そのフォームに関連付けられている **Identity Manager** ページに表示されます。

フィールドの無効化

フィールドを無効にすると、フィールドと、その入れ子フィールドはページに表示されなくなります。その値の式は評価されないか、またはどの `global.*` 属性にも組み込まれません。無効化されたフィールドの値がすでにビューに含まれる場合、その値は変更されません。

```
<Disable></Disable>
```

例

```
<Field name='waveset id'>  
  <Display class='Text'>  
    <Property title='accountId'>  
  </Display>  
  <Disable>  
    <eq><ref>userExists</ref><s>true</s></eq>  
  </Disable>  
</Field>
```

注	Disable 式は、その他の式よりも頻繁に評価されます。このため、 Disable 式は、比較的単純に記述してください。データベースのルックアップのように、負荷の大きい処理を実行する Java クラスは呼び出さないようにしてください。

Disable 式の規則でフィールドを参照するときは注意が必要です。誤って設定すると、コンテナ内のフィールドが無効化される場合があります。

フィールドの非表示

フィールドを非表示に設定すると、フィールドと、その入れ子フィールドはページに表示されなくなります。ただし、フォームの処理によって、フィールドの値は設定されます。

フィールドに **Display** クラスを割り当てないだけで、フィールドは簡単に非表示にできます。

```
<Field name='field A' />
```

フィールド値の計算

フィールドの値は、別のフィールドの値や、任意の論理式の値から計算できます。たとえば、ユーザーの姓、ミドルネームのイニシャル、および名からフルネームを導出できます。

```
<Field name='global.fullname'>
  <Expansion>
    <concat>
      <ref>global.firstname</ref>
      <s> </s>
      <ref>global.middle</ref>
      <s> </s>
      <ref>global.lastname</ref>
    </concat>
  </Expansion>
</Field>
```

デフォルト値の設定

ユーザーの名の頭文字と、姓の最初の7文字を使用して、電子メールアドレスを設定することができます。この例では、それらを連結する前に値が設定されていることを確認するため、システムが追加チェックを行います。この追加チェックの目的は次のとおりです。

- アカウントの最初の作成時にのみ、電子メールアドレスを設定できるようにする
- 姓、名の各フィールドに値が設定されていることを確認する

```
<Field name='global.email'>
  <Default>
    <and>
      <notnull><ref>global.firstname</ref></notnull>
      <notnull><ref>global.lastname</ref></notnull>
    </and>
    <concat>
      <downcase>
        <substr>
          <ref>global.firstname</ref>
          <i>0</i>
          <i>1</i>
        </substr>
      </downcase>

      <downcase>
        <substr>
          <ref>global.lastname</ref>
          <i>0</i>
          <i>6</i>
        </substr>
      </downcase>
      <s>@waveset.com</s>
    </concat>
  </Default>
</Field>
```

フィールド値の導出

一部のフィールドは、別のフィールドを計算するときにだけ、フォーム上で使用されます。これらのフィールドは、ユーザーが属すどのリソースにも格納できません。ユーザーレコードを編集すると、それぞれのリソースが参照され、属性のフィールド値が取り込まれます。計算に使用されるフィールドの値を取り込むときは、**Derivation** 規則を記述します。

例

電話番号のフィールドは、1つのテキストボックスとしてフォームに表示されます。しかし、より高度なフォームには、リソースに保存する電話番号の計算に使用される、市外局番と電話番号の3つのフィールドが含まれる場合があります。

電話番号を表す単純な例では、次のようなフォームフィールドを使用できます。

```
<Field name='P1'>
  <Display class='Text'>
    <Property name='title' value='Office Phone Number' />
    <Property name='size' value='3' />
    <Property name='maxLength' value='3' />
  </Display>
</Field>
<Field name='P2'>
  <Display class='Text'>
    <Property name='title' value='- ' />
    <Property name='size' value='3' />
    <Property name='maxLength' value='3' />
  </Display>
</Field>
<Field name='P3'>
  <Display class='Text'>
    <Property name='title' value='- ' />
    <Property name='size' value='4' />
    <Property name='maxLength' value='4' />
  </Display>
</Field>
```

```

<Field name='global.OfficePhone'>
  <Expansion>
    <concat>
      <ref>P1</ref><s>-</s>
      <ref>P2</ref><s>-</s>
      <ref>P3</ref>
    </concat>
  </Expansion>
</Field>

```

例

次の例は、前述の例で定義されている P1 フィールドのフィールド定義を拡張します。これは、電話番号の属性をフォームに読み込む方法を定義し、それを 3 つのフィールドの表示に展開します。

```

<Field name='P1'>
  <Display class='Text'>
    <Property name='title' value='Office Number' />
    <Property name='size' value='3' />
    <Property name='maxlength' value='3' />
  </Display>
</Field>

```

ユーザーが **Identity Manager** にデータを入力すると、データが適切に入力されかどうかフォームによって確認されます。しかし、リソースに直接入力された場合は、リソースが同じ要件を満たしているかどうかについて **Identity Manager** が検証することはできません。たとえば、長年にわたって、管理者が電話番号を 123-4567 (8 文字) と入力したり、123-123-4567 (12 文字) または (123) 123-4567 (14 文字) のように入力したりしている可能性もあります。

例

OfficePhone フィールドの定義は、これまでの例と同様ですが、**Derivation** 規則を使用するには、3つのフィールド(P1、P2、および P3)のそれぞれを更新する必要があります。この例は、P1 フィールドの更新を表しています。

```
<defvar name='lenOfficePhone'>
  <length><ref>Office Phone</ref></length>
</defvar>

<Field name='P1'>
  <Display class='Text'>
    <Property name='title' value='Office Phone Number' />
    <Property name='size' value='3' />
    <Property name='maxLength' value='3'>
  </Display>

  <Derivation>
    <or>
      <cond><eq>
        <ref>lenOfficePhone</ref>
        <s>8</s></eq>
        <s> </s></eq>
      </cond>
      <cond><eq>
        <ref>lenOfficePhone</ref>
        <s>12</s></eq>
        <substr>
          <ref>Office Phone</ref>
          <i>0</i>
          <i>1</i>
        </substr>
      </cond>
    </or>
  </Derivation>
</Field>
```



```

        <cond><eq>
            <ref>lenOfficePhone</ref>
            <s>14</s></eq>
            <substr>
                <ref>Office Phone</ref>
                <i>0</i>
                <i>1</i>
            </substr>
        </cond>
    </or>
</Derivation>
</Field>

```

フィールドを計算するときは、データの現在の形式と、リソースの品質に注意してください。新規ユーザーを作成するときは、正しいフィールド値の確認は比較的容易です。リソースから読み込む場合に、既存のデータをフィールドに適合させることのほうがずっと困難です。どのフィールドにも **Derivation** 規則を適用し、読み込むときに属性の形式を確認することができます。

フィールドの再計算

ユーザーがフォームを操作するときに、システムは何度もフィールドを計算します。フィールドの計算は、フィールドを最初に表示したときにデフォルト値を設定するために行われ、フォームの計算は、ユーザーが「**保存**」をクリックしたときに行われます。これ以外に、「ユーザーの編集」ページの「**再計算**」をクリックすると、フォームが評価されます。また、**action** フィールドを使用してフォームを評価することもできます。

例

```

<Field>
    <Display class='Button'>
        <Property name='label' value='Recalculate' />
        <Property name='command' value='Recalculate' />
    </Display>
</Field>

```

システムがフィールドの値を再計算するように設定するときは、次のように、Display クラス要素の action を **true** に設定します。

```
<Display class='Select' action='true'>
```

この値は、ユーザーが選択またはクリックするフィールドのみに追加します。テストエリアやテキストエリアのフィールドに追加しないでください。フィールドに action=true が設定されると、ブラウザでフィールドを変更するたびにフォームが再計算されます。

例

```
<Field name='Region'>
  <Display class='Select' action='true'>
    <Property name='title' value='Geographic Region' />
    <Property name='allowedvalues' value='North, South,
Central, Midwest' ./>
  <Property name='nullValue' value='Select a region' />
  </Display>
</Field>
```

フォームの作成に関するガイドライン

新しいフォームの構造を作成したり、既存のフォームを編集したりするときは、次のガイドラインに注意してください。

- ページに表示する順序で、フィールド要素を一覧にします。ブラウザに表示される要素の順序は、フォーム内のフィールド要素の順序によって決定されます。
- 参照先のフィールドは、参照元のフィールドの前に配置します。フィールドが別のフィールドの値を参照する式を持つ場合は、参照元のフィールドを参照先のフィールドの後に配置します。
- 論理的に true と評価される場合、無効化されたフィールドは無視されます。Disable 式を持つすべてのフィールドは評価されます。Disable 式の結果が論理的に true となる場合、フォームの評価時にそのフィールドは無視されます。

フォームフィールド内の式の最適化

フォームで実行される一部のアクティビティは、Identity Manager の外部にあるリソースを呼び出すことができます。これらのリソースへのアクセスは、Identity Manager のパフォーマンスに影響します。特に、グループや電子メール配信のリストのコンパイルのように、結果が値の長いリストになる場合は大きく影響します。このような呼び出し実行中のパフォーマンスを改善する方法については、「[フィールドデータを取得する Java クラスの使用](#)」の節のガイドラインを参照してください。

シナリオの例

次に、式を最適化する例を示します。

Identity Manager に格納されていない情報や、リソースアカウント属性としてアクセスできない情報をデータベースに照会する場合の一般的な手順は次のとおりです。

1. データベースアクセスを実行する Java クラスを記述します。
2. Java クラスを呼び出す Default 式を使用するフォームフィールドを定義します。
3. 非表示の値を参照します。

フィールドデータを取得する Java クラスの使用

情報の取得時に呼び出されるメソッドを持つ Java クラスを記述します。次の節「非表示フォームフィールドの定義」で紹介する例は、カスタムメソッドである

getJobGrade メソッドを使用します。このカスタムクラスは、`idm¥WEB-INF¥classes¥com¥waveset¥custom` ディレクトリ構造に配置してください。システムにこれらのディレクトリが存在しない場合は、作成する必要があります。

このクラスを記述するときは、次のガイドラインに注意してください。

- データベース要求のように負荷の大きい操作を実行するメソッドは、非表示フォームフィールドの Default 式から呼び出すようにします。こうすることで、フォームを最初に読み込むときに値をビューに格納できます。格納された値は、データベースのオーバーヘッドなしで何度も参照できます。
- 呼び出されるメソッドが静的に宣言されていない場合は、次の例のように、新しい要素を使用して最初にクラスをインスタンス化します。

非表示フォームフィールドの定義

まず、Java クラスを呼び出す Default 式を使用する非表示フォームフィールドを、フィールド定義に Display クラスを含めずに定義します。

```
<Field name='jobGrade'>
  <Default>
    <invoke name='getJobGrade'
      class='com.waveset.custom.DatabaseAccessor'>
      <ref>waveset.accountId</ref>
    </invoke>
  </Default>
</Field>
```

Default 式は、jobGrade 属性の値がビューに含まれない場合にのみ評価されます。一度 Default 式を実行すると、結果が jobGrade に格納され、再実行されません。

要素の「フォーム要素」ダイアログで次の操作を行います。

1. 「Display Class」メニューから「非表示」を選択します。
2. 「OK」をクリックします。

非表示属性の参照

非表示属性を定義すると、次のように別の式からその属性を参照できるようになります。

```
<Field name='secureKey'>
    <Disable><lt><ref>jobGrade</ref><i>10</i></lt></Disable>
    ...
</Field>
```

XPRESS の defvar 変数を使用して計算結果を保持することもできますが、通常は非表示フォームフィールドを使用する場合ほど効率的ではありません。

繰り返し時の変数の最適化に関する注意点

XPRESS 変数の値は、通常、フォームフィールドに対する 1 回の実行の間だけ持続します。このため、Expansion 式に変数を使用することはできませんが、それに続く Derivation 式では使用できません。計算結果の値を複数回の実行でも有効にする必要があるときは、代わりに非表示フォームフィールドを使用します。非表示フィールドの値はビューに格納され、変数セッションをキャンセルまたは保存するまで持続します。

新しいリソースとユーザーの自動リンク設定の無効化

Identity Manager には、ユーザーに新しいリソースを割り当てるときに、既存のアカウントとのリンクを制御する方法が用意されています。

ユーザーに新しいリソースを割り当てるときに、ID を割り当てられたアカウントがすでにリソースに存在する場合は、Identity Manager はデフォルトで、そのアカウントを自動的に Identity Manager ユーザーにリンクしてからプロビジョニングを継続します。反対に、この自動リンク設定を無効にし、ユーザーの新規アカウントを作成するときに別のアカウント ID を入力することもできます。

ユーザーに新規アカウントをどのようにリンクさせるかは、次の 2 つの方法で制御できます。

- ユーザーフォームで、この情報の手動リンク設定を有効にする
- プロビジョニング時の自動リンク設定を回避する

ユーザーフォームでの手動リンク設定の有効化

手動リンク設定を有効にする方法は次のとおりです。

1. 各ユーザーフォームにプロパティ定義を設定します
2. 標準フォームライブラリに含まれるフィールドを参照します

手順 1: プロパティ定義の設定

フォームの先頭に次のようなプロパティ定義を追加します。

```
<Form>

  <Properties>

    <Property name='InteractiveLinking' value='true' />

  </Properties>

  ...

</Form>
```

手順 2: 標準フォームライブラリに含まれるフィールドの参照

フォーム内の任意の場所にフィールド参照を追加します。次に例を示します。

```
<FieldRef name='DiscoveredAccountFields' />
```

このフィールドを参照するには、ユーザーフォームに次の **Include** ステートメントを設定してください。この **Include** 式は、通常、すべてのユーザーフォームにすでに存在します。

```
<Include>

  <ObjectRef type='UserForm' name='User Library' />

</Include>
```

これらのフォーム変更を完了すると、**Identity Manager** はフォームの表示を更新するたびに、また、フォームを保存する前に、既存のアカウントを調べます。**Identity Manager** が既存のアカウントを検出すると、フォームの最上部に警告メッセージが表示され、検出されたアカウントごとに新しいフィールドが挿入されます。これらの新規フィールドには、リンクを設定するアカウントを手動で指定するためのチェックボックスがあります。

さらに、**Identity Manager** は、各属性のフィールドをリソースのアイデンティティテンプレート内に生成します。このフィールドを使用して、アカウントに別のアイデンティティを指定することができます。**Identity Manager** は既存アカウントの属性をフェッチし、それをビューに含めます。

これらの属性は、**MissingFields** 参照または独自のカスタムフィールドを使用して表示できます。存在しないアカウントには別のアイデンティティを指定するか、フォームを保存する前に既存アカウントとのリンクを設定できるようにチェックボックスを選択してください。

プロビジョニング時の自動リンク設定の回避

対話的でない方法でワークフローからプロビジョニングを行うときは、Identity Manager にアカウントのリンク設定を自動的に行わせるかどうか制御できます。checkinView の呼び出しに NoLinking ビューオプションを渡すことで、リンクの自動設定を回避できます。このオプションは、いくつかの方法で指定できます。

- 次のように、このオプションを引数として WorkflowServices メソッドに渡します。

```
<Action application='com.waveset.provision.WorkflowServices'>
    <Argument name='op' value='checkinView' />
    <Argument name='view' value='${user}' />
    <Argument name='NoLinking' value='true' />
</Action>
```

- このオプションをビューの属性として設定します。この場合のビュー属性の名前は、viewOptions.NoLinking となります。この属性をワークフローに設定するときは、次のように XPRES ロジックを使用します。

```
<set name='user.viewOptions.NoLinking'>
    <s>true</s>
</set>
```

結果ページでのクリアテキストによる属性の表示の回避

Identity Manager では、編集フォームにアスタリスク付きで表示されるように属性を設定した場合でも、結果ページの属性の値は平文形式で表示されます。

属性が結果ページにクリアテキストとして表示されないようにするには、その属性をシークレット属性として登録します。シークレット属性を登録するには、次のようにしてシークレット属性を System Configuration オブジェクトに追加します。

```
<Attribute name='secretAttributes'>
    <List>
        <String>email</String>
        <String>myAttribute</String>
    </List>
</Attribute>
```

フォームからのリソースメソッドの呼び出し

invoke メソッドを使用することで、フォームからリソースのメソッドを呼び出すことができます。

invoke メソッドを呼び出すには、クラス名とメソッド名を指定します。invoke タグ内部のメソッドには、次の例のように引数を渡すこともできます。

```
<Default>
  <block>
    <defvar name='vmsResName'>
      <index i='0'>
        <ref>accountInfo.accounts[type=vms].name</ref>
      </index>
    <defvar>
      <invoke name='callResourceMethod'
        class='com.waveset.ui.FormUtil'>
        <ref>display.session</ref>
        <ref>vmsResName</ref>
      <null/>
    </invoke>
  </Default>
```

フィールドの「フォーム要素」ダイアログで次の操作を行います。

1. 「Display Class」メニューから「Javascript」を選択します。
2. 「OK」をクリックします。

別のフォームからのフォームの参照

<FormRef> 要素を使用することで、完全なフォームではなく、別のフォーム内の特定のフィールドを参照できます。

外部フォームからの別のフォームを含めるには、<FormRef> 要素を使用します。次の例は、MissingFields というフォームを呼び出します。

```
<FormRef name='MissingFields' />
```

```

<FieldRef name='AuthenticationAnswers' />
<FieldRef name='AccountInformation' />
<Field name='waveset.backgroundSave'>
    <Display class='Hidden' />
</Field>

```

別のフォームからのフィールドの参照

<FieldRef> 要素を使用することで、完全なフォームではなく、別のフォーム内の特定のフィールドを参照できます。

外部フォームからの特定のフィールドを含めるには、<FieldRef> 要素を使用します。このとき、次の項目を指定します。

- フィールドが存在するフォームの名前。このフォーム名は、<ObjectRef> 要素を使用して、フォームヘッダーの **Include** セクションのリストに含めてください。フォーム名 (UserForm) と一意の設定 ID は、プロパティタイプで指定します。name プロパティは、後で参照するフィールドの名前を識別します。
- ページ上に表示する位置に対応したフォームセクションに挿入されるフィールド名。

```

<Include>
    <ObjectRef type='UserForm'
        id='#ID#04F5F14E01889DFE:2E5C94:F131DD723D:-7FE4'
        name='Password Library' />
    <ObjectRef type='UserForm'
        id='#ID#04F5F14E01889DFE:2E5C94:F131DD723D:-7FE3'
        name='Account Summary Library' />
    <ObjectRef type='UserForm'
        id='#ID#UserForm:UserFormLibrary' />
    <ObjectRef type='UserForm' name='Global Attributes' />
</Include>

```

次の例では、ページ上に表示する位置に対応したフォームセクションにフィールド名が挿入されています。


```

<Field name='global.fullname' hidden='true'>
  <Expansion>
    <cond>
      <and>
        <ref>global.firstname</ref>
        <ref>global.lastname</ref>
      </and>
      <concat>
        <ref>global.firstname</ref>
        <s> </s>
        <ref>global.lastname</ref>
      </concat>
    </cond>
  </Expansion>
</Field>

```

次の例の <FieldRef> 要素は、参照する属性の名前を識別します。

```

<Field>
  <Disable>
    <isnull>
      <ref>waveset.id</ref>
    </isnull>
  </Disable>
  <FieldRef name='DynamicChangePasswordFields' />
</Field>

```

フォームの編集

フォームを編集することで表示特性を変更したり、論理処理を追加してフィールドやコンポーネントを選択したりできます。ここでは、次の2つのカテゴリに分けて、フォーム関連編集タスクについて説明します。

- **Display 要素の操作**：ここでは、Identity Manager のフォームを編集するときに、特にユーザーに表示される基本ページコンポーネントの表示特性の変更について説明します。このようなコンポーネントには、ボタン、ラジオボタン、チェックボックスなどがあります。
- **非表示コンポーネントの操作**：このセクションでは、バックグラウンド処理や、表示されるフォームへの論理処理の追加などの目的で Identity Manager のフォームに追加される、HTML 要素コンポーネントについて説明します。このような要素には、<Disable> コンポーネントや <Expansion> コンポーネント、FormUtil メソッドなどがあります。

タスク関連のセクションで説明される HTML コンポーネントは、「[HTML 表示コンポーネント](#)」でアルファベット順に説明されています。

Display 要素の操作

Identity Manager フォームでの変更や追加がもっとも多い Display 要素は、ボタン、フィールド、およびテキスト入力ボックスです。Display 要素には、これ以外にテーブルやセクションのヘッダーがあります。

クラスが指定されていない Display 要素は非表示となります。

ボタン

一般的な押しボタンを作成するときは、<Button> コンポーネントを使用します。

複数のボタンを水平に並べるときは、<ButtonRow> コンポーネントを使用します。

<Field>

```
<Display class='Button'>
  <Property name='location' value='true'/>
  <Property name='label' value=' キャンセル '/>
  <Property name='command' value='Cancel' />
</Display>
```

</Field>

ボタンをボタン行に配置するときは、ボタンの定義に `<Property name='location ' value=' button ' />` というコードを追加します。この Property フィールドを設定しない場合、ボタンはフォームに記述されている順序で縦に表示されます。

ボタンラベルの割り当てと変更

ボタンを定義するとき、ラベルの値は、次のように label プロパティの value の設定によって指定されます。

```
<Display class='Button'>  
    <Property name='label' value=' キャンセル ' />
```

ブラウザは、このコードに指定されている「**キャンセル**」をボタンのラベルとして表示します。

ボタンのデフォルト名の無効化

Identity Manager フォームの下部には、通常は2つのボタンが表示されます。しかし、これらのボタンには「**保存**」と「**キャンセル**」というデフォルトラベルが付けられています。これらのボタン名を変更するときは、フォームを次のように変更します。

1. フォーム名を定義する行 (ヘッダー内) で、name フィールドの設定を変更します。

```
<Form name='Anonymous User Menu'>
```

上記の行を次のように変更します。

```
<Form name='Anonymous User Menu' noDefaultButtons=true>
```

フォームの下部に次のような「**保存**」ボタンと「**キャンセル**」ボタンのフィールドを追加し、ラベル名を適切に変更します。

```

<Field>
  <Display class='Button'>
    <Property name='label' value='Submit' />
    <Property name='name' value='submitButton' />
    <Property name='value' value='true' />
    <Property name='command' value=' 保存 ' />
  </Display>
</Field>

<Field>
  <Display class='Button'>
    <Property name='label' value=' キャンセル ' />
    <Property name='command' value='Cancel' />
    <Property name='location' value='true' />
  </Display>
</Field>

```

command パラメータの値とボタン

注 ここで説明する内容は、Button オブジェクトを作成する場合にのみ重要です。XML フォームでコンポーネントを作成するときは、次の表の値が認識されることが前提になると判断できます。

Identity Manager のインタフェースのすべてのページでは、クリックされたフォーム送信ボタンを伝達するメカニズムとして、*command* というデータ送信パラメータが使用されてきました。この条件は、コンポーネントを使用するページプロセッサシステムには適用されませんが、特に Button などの一部のコンポーネントには、*command* パラメータ用の特別サポートが含まれます。

XML フォームを処理するシステムをはじめとする一部のページプロセッサシステムは、*command* パラメータの使用を前提としています。さらに、特定のアクションを示すために、いくつかの *command* パラメータの値が使用されています。次の表は、このような値について説明しています。

表 3-11 command パラメータに指定できる値

パラメータ	説明
Save	フォームコンテンツを保存すべきであることを示します。
Cancel	フォームコンテンツを破棄すべきであることを示します。
Recalculate	入力データに基づいてフォーム表示を更新すべきであることを示します。

command パラメータでは、どのような値も使用できますが、認識されない値がページの再表示につながることを覚えておいてください。

<ButtonRow> 要素によるボタンの整列

複数のボタンを 1 行に整列させるときは、ButtonRow 要素を使用します。

```
<Field name='OrganizeButtons'>
  <Display class='ButtonRow'>
    <Property name='title' value='Choose a Button' />
  </Display>
<Field name='ChangePassword'>
  <Display class='Button'>
    <Property name='label' value='Change Password' />
    <Property name='value' value='Recalculate' />
  </Display>
</Field>
<Field name='ResetPassword'>
  <Display class='Button'>
    <Property name='label' value='Reset Password' />
    <Property name='value' value='Recalculate' />
  </Display>
</Field>
```

テキストフィールド

フォームには、単一行と複数行の両方のテキスト入力ボックスを含めることができます。単一行のテキスト入力フィールドを作成するときは、<Text> 要素を使用します。複数行のテキスト入力フィールドを作成するときは、<TextArea> 要素を使用します。

```
<Display class='Text'>

  <Property name='title' value='Zip Code' />

  <Property name='size' value='10' />

  <Property name='maxLength' value='10' />

  <Property name='required' value='true' />

</Display>
```

フィールドラベルの割り当てと変更

テキストフィールドまたはエリアを定義するとき、ラベルの値は、次のように label プロパティの value プロパティで設定されます。

```
<Display class='Text'>

  <Property name='label' value=' 入力 ' />
```

ブラウザは、このコードに指定されている「入力」をテキスト入力フィールドのラベルとして表示します。

コンテナ

一部の Display 要素は、コンテナコンポーネントというコンポーネント内に配置されます。コンテナコンポーネントには、次のような用途があります。

- 複数のコンポーネントをまとめ、特定の方法で表示します。単純なコンポーネントは、コンポーネントを縦または横に並べることができます。より複雑なコンテナは、より複雑にコンポーネントを配置でき、コンポーネントの周囲に装飾を加えることもできます。
- フォームで非表示または無効にするコンポーネントをグループ化します。

コンテナクラスを作成すると、通常は HTML table タグが生成されます。

次の表は、代表的なコンテナコンポーネントについて説明しています。

表 3-12 代表的なコンテナコンポーネント

コンポーネント	説明
<SimpleTable>	コンポーネントをグリッドに配置します。オプションとして、上部に列のタイトル行も設定できます。

表 3-12 代表的なコンテナコンポーネント (続き)

コンポーネント	説明
<ButtonRow>	ボタンを横方向で 1 行に配置します。このコンポーネントは、基本的には、水平レイアウトが事前に設定されたパネルです。
<BorderedPanel>	コンポーネントを、東西南北と中央の 5 つの領域に配置します。
<SortingTable>	ソートに対応した列を持つ、行の背景色が交互に青とベージュで示される表を表示します。

簡単な表の作成

Identity Manager フォームのコンテナコンポーネントでは、<SimpleTable> というコンポーネントが頻繁に使用されます。これは、コンポーネントをグリッドに配置します。オプションとして、上部に列のタイトル行も設定できます。この **Display** コンポーネントの唯一のプロパティは、列にタイトルを割り当て、プロパティ文字列のリストで表の幅を定義する `columns` です。

次の例は、複数のサブフィールドの配置に SimpleTable を使用するフィールドを示しています。

```
<Field name='SelectionTable'>

    <Display class='SimpleTable'>
        <Property name='columns'>
            <List>
                <String>Account</String>
                <String>Description</String>
            </List>
        </Property>
    </Display>

    <Field name='accounts[LDAP].selected'>
        <Display class='Checkbox'>
            <Property name='label' value='LDAP' />
        </Display>
```

```

</Field>

<Field>
  <Display class='Label'>
    <Property name='text' value='Primary Corporate LDAP
Server' />
  </Display>
</Field>

<Field name='accounts[W2K].selected'>
  <Display class='Checkbox'>
    <Property name='label' value='Windows 2000' />
  </Display>
</Field>

<Field>
  <Display class='Label'>
    <Property name='text' value='Primary Windows 2000 Server' />
  </Display>
</Field>

</Field>

```

コンポーネントのグループ化

フォームの複数のコンポーネントをグループ化して非表示または無効にするときは、`<SimpleTable>` コンテナを次の例のように使用します。


```
<Field>
  <Disable>
    <not>
      <contains>
        <ref>accountInfo.typeNames</ref>
        <s>Windows Active Directory</s>
      </contains>
    </not>
  </Disable>
  <Field name='accounts[AD].HomeDirectory'>
    <Display class='Text'>
      <Property name='title' value='Home Directory'>
    </Display>
  </Field>
</Field>
```

リストの操作

リストの作成に使用するコンポーネントは、リストの長さと、ユーザーが複数のオプションを同時に選択できるかどうかによって異なります。

多くのテキストボックスには、選択可能な複数のオプションを示すリストを備えています。これらのリストのデータを取り込むときは、`allowedValues` というプロパティの内部でリストを指定して選択するか、リソースのメソッド呼び出し (`FormUtil` クラスメソッド) を利用して動的に取得します。リストのテキストエリアへの情報の取り込みについては、この章の「[リストデータの取り込み](#)」の節を参照してください。

次の表は、代表的なリストタイプと、その作成に使用される HTML Display コンポーネントについて説明しています。

表 3-13 代表的なリストタイプと関連する Display コンポーネント

リストタイプ	HTML コンポーネント
true と false のように、互いに排他的な値を持つオプションリスト	<CheckBox> 「 チェックボックスの作成 」の節を参照してください。
ユーザーが 1 つのオプションのみを選択できる複数オプションのリスト	<RadioButton> 「 ラジオボタンの作成 」の節を参照してください。
ユーザーが 1 つのオプションのみを選択できる複数オプションのリスト (オプション数が多いリスト)	<Select> 「 単一選択リストの作成 」の節を参照してください。
複数のオプションを同時に選択できる複数オプションのリスト	<MultiSelect> 「 複数選択リストの作成 」の節を参照してください。

チェックボックスの作成

チェックボックスの表示には、<Checkbox> コンポーネントを使用します。チェックボックスを選択すると、そのボックスの値は true となります。選択していないボックスの値は false です。label プロパティの値を編集することで、チェックボックス名を変更できます。

例 1

```
<Field name='accounts[LDAP].selected'>  
  <Display class='Checkbox'>  
    <Property name='label' value='LDAP' />  
  </Display>  
</Field>
```

例 2

```
<Field name='global.Password.Expired'>  
  <Display class='CheckBox'>  
    <Property name='title' value='User must change password at  
    next login' />  
  </Display>  
</Field>
```

```
        <Property name='alignment' value='left' />
    </Display>
</Field>
```

ラジオボタンの作成

1 つまたは複数のラジオボタンの水平リストを表示するには、<Radio> コンポーネントを使用します。ユーザーが一度に選択できるラジオボタンは 1 つだけです。コンポーネントの値が NULL の場合、または許可される値のいずれとも一致しない場合は、ボタンは選択されません。

```
<Field name='global.EmployeeType'>
    <Display class='Radio'>
        <Property name='title' value='EmployeeType' />
        <Property name='labels' value='Employee, Contractor,
Temporary, Part Time' />
        <Property name='required' value='true' />
    </Display>
</Field>
```

単一選択リストの作成

<MultiSelect> コンポーネント以外に、<Select> コンポーネントも、選択できる項目のリストを作成できます。選択できる値のリストが長くなる場合、ラジオボタンではフォームを占める面積が大きくなります。一方、select リストでは、選択できる値の長いリストからユーザーが値を選択できます。リストが整っている場合は、このリストは先行入力をサポートします。ユーザーが選択できる項目を指定するには、allowedValues プロパティを使用します。

```
<Field name='global.title'>
  <Display class='Select'>
    <Property name='title' value='Title' />
    <Property name='allowedValues'>
      <List>
        <String>Staff</String>
        <String>Manager</String>
        <String>Director</String>
        <String>VP</String>
      </List>
    </Property>
  </Display>
</Field>
```

複数選択リストの作成

複数のオプションを選択できるリストボックスの表示には、`<MultiSelect>` コンポーネントを使用します。このテキストボックスは、1つのボックスに含まれる定義済みの値セットを選択ボックスに移動できる、2つの部分から構成されるオブジェクトを表示します。リストボックスの値の指定には、`allowedValues` 要素を使用するか、`getResources` などのメソッド呼び出しを利用して動的に値を取得します。

`<Select>` コンポーネント以外に、`<MultiSelect>` コンポーネントも、選択できる項目リストを動的に作成できます。これらのリストのデータを取り込むときは、`allowedValues` というプロパティの内部でリストを指定して選択するか、リソースのメソッド呼び出しを利用して動的に取得します。複数選択入力ボックスに表示するリストの取り込みについては、「[リストデータの取り込み](#)」の節を参照してください。

```
<Field name='waveset.roles'>
  <Display class='MultiSelect' action='true'>
    <Property name='title' value='Roles' />
    <Property name='availableTitle' value='Available Roles' />
    <Property name='selectedTitle' value='Current Roles' />
    <Property name='allowedValues'>
      <invoke name='getObjectNames'
        class='com.waveset.ui.FormUtil'>
        <ref>display.session</ref>
        <s>Role</s>
        <ref>waveset.original.roles</ref>
      </invoke>
    </Property>
  </Display>
</Field>
```

選択リストの別の表示値セット

フィールドに実際に割り当てられている値セットとは異なる値セットを表示する選択リストを作成できます。このリストは、暗号化された複数の値の判読可能な名前を表示したり、フォームを国際化するために別の表示言語に対応したりする場合によく使用されます。このようなリストを作成するには、valueMap プロパティを使用して、実際の値と表示値を関連付けます。次の例を参照してください。

コード例 3-3 valueMap プロパティによる選択リストの値の変更

```
<Field name='waveset.organization'>
  <Display class='Select'>
    <Property name='title' value='Add Account' />
    <Property name='nullLabel' value='Select...' />
    <Property name='valueMap'>
      <list>
        <s>Top</s>
        <s>Top Level</s>
        <s>Top:OrgB</s>
        <s>Ted's Organization</s>
        <s>Top:OrgC</s>
        <s>Super Secret Org</s>
      </list>
    </Property>
  </Display>
</Field>
```

前述の例では、値のマッピングが文字列ペアのリストとして指定されます。リスト項目の奇数行の文字列は、このフィールドに割り当てられる実際の値です。偶数行の文字列は、選択リストに表示される値です。たとえば、選択リストから Ted's Organization を選択すると、このフィールドの値は Top:OrgB となります。

リストデータの取り込み

リストには、ユーザーオブジェクトや外部リソースに格納されている情報から動的に計算されたオプションが取り込まれることがよくあります。このようなリストを作成するには、まず、リストデータを取り込む前に、**HTML List** コンポーネントを作成します。**HTML テキストボックスコンポーネント**の使用についての詳細は、「[単一選択リストの作成](#)」と「[複数選択リストの作成](#)」の節を参照してください。

ここで説明するメソッドを含め、これらのリストデータの取り込みには2つの方法があります。

- `allowedValues` プロパティによるリストデータの取り込み

- FormUtil メソッドによる、単一選択リストまたは複数選択リストへの、外部リソースから動的に取得した情報の取り込み

特定のタスクの実行に、XPRESS ではなく XML オブジェクト言語を使用する利点については、「[XML オブジェクト言語と XPRESS でのリストの表現](#)」の節を参照してください。

リストへの選択可能値セットの取り込み

フォームで使用されるリストにデータを取り込むためのもっとも一般的な方法は、allowedValues プロパティを使用することです。このプロパティを使用することで、<Select> 要素と <MultiSelect> 要素で選択できる値のオプションリストを指定できます。このコンポーネントの値は常にリストであり、その値は通常、文字列です。

```
<Field name='department'>
  <Display class='Select' action='true'>
    <Property name='title' value='Department' />
    <Property name='allowedValues'>
      <List>
        <String>Accounting</String>
        <String>Human Resources</String>
        <String>Sales</String>
        <String>Engineering</String>
      </List>
    </Property>
  </Display>
</Field>
```

グループの複数選択リストへの動的なデータ取り込み

複数選択リストは、通常、2つの部分から構成されます。

- リストの左側には、選択できる項目が表示されます。値の定義には allowedValues プロパティを使用します。このプロパティには、文字列のリスト、XML オブジェクト文字列のリスト、または Java メソッドの呼び出しによって取得された文字列のリストを指定します。

- リストの右側には、現在選択されている項目が表示されます。これらの値を設定するには、左側の `allowedValues` リストから1つまたは複数の項目を選択し、それを右側の選択リストに移動します。フォームの読み込み時に現在の設定を取得し、それをリストの右側に取り込むこともできます。

グループの複数選択リストの追加

リソースから動的に取得されたデータを、グループの複数選択リストに追加する方法は次のとおりです。

- スキーママップの右側にグループを追加します。複数選択リストを表示するテキストエリアの右側部分に表示される値には、フィールド名によって特定される関連ビュー属性の現在値が取り込まれます。
- 次のテキストを任意のフォームに追加し、必要に応じて `Field name`、`prompt`、`availableTitle`、`selectedTitle`、およびリソースの `name` のみを変更してください。

注 次の例の `display.session` の直前の `:(コロン)` は、フォームのベースコンテキストを無視し、ワークフローコンテキストのルートからオブジェクトを参照できることを表します。

次の例の `display.session` の直前の `:(コロン)` は、フォームのベースコンテキストを無視し、ワークフローコンテキストのルートからオブジェクトを参照できることを表します。

```
<Field name='global.AD Groups'>
  <Display class='MultiSelect' action='true'>
    <Property name='title' value='AD Group Membership' />
    <Property name='availableTitle' value='Available AD Groups' />
    <Property name='selectedTitle' value='Selected AD Groups' />
    <Property name='allowedValues'>
      <invoke class='com.waveset.ui.FormUtil' name='listResourceObjects'>
        <!-- メソッドがユーザー権限の検証に使用するセッション情報を送信します -->
        <ref>:display.session</ref>
        <!-- リソースオブジェクトタイプ - これはリソースごとに異なりますが、アカウント、グループ、および " 配信リスト " のタイプは共通です -->
        <s>Group</s>
        <!-- 呼び出されるリソースの名前 -->
        <s>AD Resource Name</s>
```


<!-- オプションマップ - 一部のリソースにはコンテキストのようなオプションがあり、グループのリストが含まれます。たとえば、Active Directory の場合は、複数のコンテナが含まれます。デフォルトでは、リソースに指定されているコンテナが使用されます。ここに指定する値は、デフォルト値に優先して適用されます。リソースがオプションをサポートしない場合は、値を <null/> にしてください -->

<Map>

```
<MapEntry key='context'
value='ou=Austin,ou=Texas,dc=Sun,dc=com' />
```

</Map>

<!-- cacheList - このリストをリソースオブジェクトのリストキャッシュに取り込むかどうかに応じて、true または false に設定します -->

```
<s>true</s>
```

</invoke>

</Property>

</Display>

</Field>

注

リソースがオプションをサポートしない場合は、options map の値を null にしてください。一部のリソースにはコンテキストのようなオプションがあり、グループのリストが含まれます。たとえば、Active Directory の場合は、複数のコンテナが含まれます。デフォルトでは、リソースに指定されているコンテナが使用されます。ここに指定する値は、デフォルト値に優先して適用されます。

cacheList の値は、このリストをリソースオブジェクトのリストキャッシュに格納するかどうかに応じて、true または false に設定します。これにより、メソッドは 1 回実行され、結果はサーバーに格納されます。

選択リストのテキスト入力フィールドの作成

項目を選択するだけでなく、ユーザーが値を入力できる選択リストが必要となる場合もあります。このようなリストを作成するには、次の例のように 3 つのフィールドを実装します。

- この例は、Other というテキスト文字列と、それに隣接するテキストボックスを持つ選択ボックスを作成します。ユーザーが選択ボックスから「Other」オプションを選択すると、ユーザーがカスタム情報を入力するためのフィールドがページに表示されます。

- ユーザーが所属業務を選択するための、業務リストを定義する変数を作成するには、defvar 要素を実装します。

注	フォームで複数回参照される変数を、規則に取り込むようにしてください。次の例では、選択項目のリストは変数 (この例では titleList) に格納されます。これにより、Derivation 規則はこの変数を検索できるようになります。

次の例には、説明的なテキストが組み込まれています。

```
<defvar name='titleList'>

  <list>

    <s>Manager</s>

    <s>Accountant</s>

    <s>Programmer</s>

    <s>Assistant</s>

    <s>Travel Agent</s>

    <s>Other</s>

  </list>

</defvar>
```

この例の次の部分には、title と otherTitle という 2 つの可視フィールドが含まれます。otherTitle フィールドは、ユーザーが選択リストから「Other」オプションを選択した場合にのみ表示されます。3 番目の非表示フィールドは global.Title です。これは、Title と otherTitle のいずれかから設定されます。

ユーザーが選択するためのメインフィールドは Title フィールドです。リストに必要な項目を見つけられなかった場合、ユーザーは「Other」オプションを選択できます。これは一時的なフィールドであり、ユーザーが「保存」をクリックしたときに、格納されたり、ワークフロープロセスに渡されることはありません。リソースからの値の送信と、値がリストに含まれるかどうかの判断には、Derivation 規則が使用されます。

注	次の例では、フォームフィールドに値が自動的に設定されるように、action が true に設定されています。

```

<Field name='Title'>
  <Display class='Select' action='true'>
    <Property name='title' value='Title' />
    <Property name='allowedValues'>
      <Property name='nullLabel' value='Select Ö' />
      <expression>
        <ref>titleList</ref>
      </expression>
    </Property>
  </Display>
  <Derivation>
    <cond>
      <isnull><ref>global.Title</ref></isnull>
      <null/>
    <cond>
      <eq>
        <contains>
          <ref>titleList</ref>
          <ref>global.Title</ref>
        </contains>
        <i>1</i>
      </eq>
      <ref>global.Title</ref>
      <s>Other</s>
    </cond>
  </cond>
</Derivation>
</Field>

```

Other フィールドは、ユーザーが **Title** フィールドから「**Other**」を選択した場合にのみフォームに表示されます。Other フィールドの値は、フォームの読み込み時に設定されます。この値は、Title フィールドと `global.title` フィールドの値に基づきます。

```
<Field name='otherTitle'>
  <Display class='Text'>
    <Property name='title' value='Other Title' />
    <Property name='rowHold' value='true' />
    <Property name='noWrap' value='true' />
    <Property name='size' value='15' />
    <Property name='maxLength' value='25' />
  </Display>
  <Disable>
    <neq>
      <ref>Title</ref>
      <s>Other</s>
    </neq>
  </Disable>
  <Derivation>
    <cond>
      <eq>
        <ref>Title</ref>
        <s>Other</s>
      </eq>
      <ref>global.Title</ref>
    </cond>
  </Derivation>
</Field>
```

Field の値は、Title フィールドの値に基づきます。このフィールドの値が Other に設定されると、フィールドの値は otherTitle フィールドの値となります。それ以外の値に設定された場合は、Title フィールドの値となります。

```
<Field name='Title'>
  <Expansion>
    <cond>
      <eq>
        <ref>global.fieldTitle</ref>
        <s>Other</s>
      </eq>
      <ref>otherTitle</ref>
      <ref>Title</ref>
    </cond>
  </Expansion>
</Field>
```

フォームに表示する前のリソースアカウントリストのフィルタリング処理

フォームに表示する前に、リソースアカウントのリストをフィルタリングできます。ユーザーに表示されるリストから無効化されたアカウントを除去するデフォルトのフィルタリング動作が維持されるユーザーインタフェースの「Change Password Form」を除き、デフォルトでは、フィルタは適用されません。

この除去フィルタは、Form プロパティとして定義されます。このフィルタは、1 つまたは複数の属性条件のリストです。これは、指定されたリソースアカウントを表示リストから除外するかどうかを決定するときに評価されます。

この機能をサポートするフォーム

次のフォームは、Form プロパティによる除去フィルタの指定をサポートします。

Change Password Form (ユーザーインタフェース)

管理者インタフェースのフォーム：

- ユーザーパスワード変更フォーム
- プロビジョニング解除フォーム

- 無効化フォーム
- Enable Form
- Rename Form
- 再プロビジョニングフォーム
- ユーザーパスワードリセットフォーム
- ロック解除フォーム

排除プロパティの形式

フォームの排除 (Exclude Form) プロパティの形式は次のとおりです。

```
<Configuration wstype='UserForm' ...
  <Extension>
    <Form noDefaultButtons='true'>
      ...
    <Properties>
```

表示されるアカウントのリストに、無効化されたリソースアカウントを含めるときは、リストから無効化属性条件を削除します。

```
<Property name='Exclude'>
  <list>
    <new class='com.waveset.object.AttributeCondition'>
      <s>disabled</s>
      <s>equals</s>
      <s>yes</s>
    </new>
  </list>
</Property>
</Properties>
...
</Form>
</Extension>
</Configuration>
```

有効なビュー属性

有効な属性名は、currentResourceAccounts オブジェクトの各インスタンスの、前述の各フォームと関連付けられているビューによって参照される名前です。次の属性は、有効な属性です。

- accountDisplayName (文字列)
- accountId (文字列)
- directlyAssigned (true/false)
- disabled (yes/no)
- exists (yes/no)
- id (文字列)
- lastPasswordUpdate (文字列)
- resource (文字列)
- selected (true/false)
- type (文字列)
- userPwdRequired (yes/no)

例：リソースアカウントリストからの LDAP リソースタイプの排除

指定されたフォームリストから、直接割り当てられない、LDAP タイプのすべてのリソースアカウントを排除するには、排除プロパティを次のように設定します。

```
<Property name='Exclude'>
  <list>
    <new class='com.waveset.object.AttributeCondition'>
      <s>type</s>
      <s>equals</s>
      <s>LDAP</s>
    </new>
    <new class='com.waveset.object.AttributeCondition'>
      <s>directlyAssigned</s>
      <s>equals</s>
      <s>>false</s>
    </new>
  </list>
</Property>
```

allowedValues プロパティ内からの FormUtil メソッドの呼び出し

データベースのような Identity Manager の外部にあるリソースから情報を動的に取得し処理するための FormUtil メソッドは、allowedValues プロパティ内からも呼び出すことができます。

この例は、FormUtil メソッドを呼び出して、選択リストにデータを取り込む方法を示しています。次の例では、allowedValues プロパティの内部からメソッドが呼び出されます。getOrganizationsWithPrefixes メソッド (または任意の FormUtil メソッド) は、式の内部から呼び出されます。


```

<Field name='waveset.organization'>
  <Display class='Select'>
    <Property name='title' value='Organization' />
    <Property name='autoSelect' value='true' />
    <Property name='allowedValues'>
      <expression>
        <invoke class='com.waveset.ui.FormUtil'
          name='getOrganizationsWithPrefixes'>
          <ref>:display.session</ref>
        </invoke>
      </expression>
    </Property>
  </Display>
</Field>

```

XPRESS は、リソースまたは ActiveSync アダプタ内から Java メソッドを呼び出す機能もサポートしています。呼び出しの結果として得られるデータは、複数選択リストまたは単一選択リストに取り込むことができます。式の内部からのメソッドの呼び出しについては、「[XPRESS 言語](#)」を参照してください。

ラベルフィールドの作成

ラベルは、読み取り専用フィールドの値を表示するときに便利なコンポーネントです。<Label> コンポーネントのプロパティを使用することで、色、値 (文字列)、フォントスタイルなどの表示特性を定義できます。

```

<Field>
  <Display class='Label'>
    <Property name='text' value='Primary Corporate LDAP
      Server' />
  </Display>
</Field>

```

value 属性の値は常に文字列です。

その他の Display 要素の操作

これまでに説明した要素以外に、次の Display 要素をフォームに組み込むことができます。

- セクションヘッダー
- カレンダーアイコン
- バックリンク

フォームへのセクション見出しの追加

長いフォームを目立つラベルで分割するときは、セクション見出しが便利です。

<SectionHead> 要素は、title (prompt) プロパティの値によって定義される、新しいセクション見出しを表示します。これは、font プロパティに大きな太字のスタイルを設定する Label クラスの実行に相当します。また、pad プロパティはゼロに設定され、デフォルトの 2 文字分の隙間は取り除かれます。

```
<Field>

  <Display class='SectionHead'>

    <Property name='title' value ='Calculated Fields' />

  </Display>

</Field>
```

フォームへのカレンダーアイコンの追加

DatePicker 要素を使用することで、ページにカレンダーアイコンを追加できます。ユーザーは、このアイコンをクリックして日付を選択し、ページフィールドに値を適用できます。たとえば、Identity Manager の「監査レポートの作成」ページは、開始日と終了日の選択に、このコンポーネントを使用しています。

DatePicker 要素は、日付オブジェクトを返します。DatePicker 要素を使用して設定するほとんどのリソース属性は、文字列形式の日付を必要とします。追加のテキストフィールドは、新しい日付オブジェクトから文字列への変換、または現在の設定の表示に使用されます。

次の表に示される、さまざまな形式の文字列を渡して dateToString メソッドを呼び出すことで、形式が異なるいずれかの日付を取得できます。

表 3-14 有効期限日の形式

有効期限日のフィールド	形式
AIX	MMddHHmmyy

表 3-14 有効期限日の形式 (続き)

有効期限日のフィールド	形式
HPUX	MM/dd/yy
Solaris	MM/dd/yyyy

```

<Field name='aix_account_expire'>
  <Display class='DatePicker'>
    <Property name='title' value='Set Password Expiration Date' />
  </Display>
</Field>

```

次のフィールドは、/etc/security/user ファイルに指定されている、パスワードの有効期限日を表示します。このフィールドは、新しい日付を選択したあとに表示の更新や再計算を行った場合、aix_account_expire フィールドによって選択される新しい日付も表示します。**Identity Manager** は、aix_account_expire 日付フィールドに **DatePicker** フィールドから NULL 以外の値が設定されたかどうかを確認します。

この日付フィールドが設定されている場合、**Identity Manager** は invoke メソッドを呼び出して、指定された形式 (MMddHHmmyy) に日付オブジェクトを変換します。

設定されていない場合は、AIX OS に設定されている現在の日付 (accounts[AIX].aix_expires) が表示されます。

```

<Field name='accounts[AIX].aix_expires'>
  <Display class='Text'>
    <Property name='title' value='Current Password
      Expiration Date' />
    <Property name='noNewRow' value='true' />
    <Property name='readOnly' value='true' />
    <Property name='size' value='10' />
  </Display>
  <Expansion>
    <cond>
      <notnull>
        <ref>aix_account_expire</ref>
      </notnull>
      <invoke name='dateToString'
        class='com.waveset.util.Util'>
        <!-- dateToString メソッドの最初の引数は日付オブジェクトです -->

        <ref>aix_account_expire</ref>
        <!-- 2 番目の引数は、変換後の日付 / 文字列の形式です -->

        <s>MMddHHmmyy</s>
      </invoke>
      <ref>accounts[AIX].aix_expires</ref>
    </cond>
  </Expansion>
</Field>

```

バックリンクの追加

ブラウザの「戻る」ボタンと同じように動作するコンポーネントを追加できます。このコンポーネントを使用することで、フォーム上の任意の場所にバックリンクを追加できます。

```
<Field name='back'>
    <Display class='BackLink'>
        <Property name='title' value='Back' />
        <Property name='value' value='previous page' />
    </Display>
</Field>
```

フォーム上のコンポーネントの位置

フォーム上でのコンポーネントの位置は、次の要因によって決定されます。

- このフォームに関連付けられている **Java Service Page (JSP)**: フォームのタイトルとサブタイトルは、ここに設定できます。
- **フォームにコンポーネントを表示する順序**: ブラウザは、フォームに設定されている順序でフォームフィールドを表示します。
- **コンテナフォームの使用**: たとえば、縦に並べたボタンを作成するには、`<ButtonRow>` コンテナコンポーネントを使用します。

非表示コンポーネントの使用

多くのフォームはユーザーに表示されず、リソースアダプタを通じて外部リソースから取得したデータを、**Identity Manager** に渡す前に処理するために使用されます。表示されるフォームでも、一部のコンポーネントは非表示に設定できます。非表示コンポーネントは、この入力データの処理と、判読可能な形式へのデータ変換にも使用されます。

フォーム内の一部の非表示処理は、**FormUtil Java** クラス内のメソッドによって実行されます。これは、フォーム内のリストの情報を、外部リソースから動的に取得する場合に頻繁に使用されます。

ここでは、データを処理し、フォームで行われるその処理をオプションとして非表示に設定するタスクについて説明します。代表的なタスクには、次のものがあります。

- **Derivation 要素と Expansion 要素**を使用して、XPRESS ロジックを含める
- メソッドを呼び出して、リストにデータを取り込む
 - DN 文字列を作成する
 - セッション所有者がアクセスできるオブジェクトタイプのリストを取得する

- 組織のリストを取得する
- 割り当てられていないリソースのリストを取得する
- リソースオブジェクト名のリストを取得する
- コンポーネントを無効化する
- コンポーネントを非表示にする

Derivation 要素と Expansion 要素による XPRESS ロジックの追加

一般に、フィールドには Derivation 規則または Expansion 規則のいずれかが適用されます。フィールドに両方の規則が含まれる場合は、これらのフィールドが相互に競合しないことを確認してください。

XPRESS を使用してフォームフィールドの値を計算するときは、<Expansion> コンポーネントと <Derivation> コンポーネントを実装します。これらの式は、式が評価されるタイミングが異なるだけで、基本的には同じです。多くの場合、Derivation 規則は、フォームの読み込み時にフィールドの値を設定する場合に使用されます。Expansion 規則は、ページを再計算するとき、またはフォームを保存するときにフィールドの値を設定する場合に使用されます。

表 3-15 Derivation 式と Expansion 式

コンポーネント	説明	評価
<Derivation>	このフィールドの値として使用される任意の値を無条件で計算します。Derivation 式が評価されると、フィールドの現在値は常に置き換えられます。	Derivation 規則は、フォームが最初に読み込まれるとき、あるいは、1 つまたは複数のリソースからデータがフェッチされるときに実行されます。
<Expansion>	フィールドの値を無条件で計算します。	Expansion 規則は、ページを再計算するとき、またはフォームを保存するときに実行されます。 ユーザービュー以外のすべてのフォームでは、Expansion 規則は、ページを再計算するとき、またはフォームを保存するときに実行されます。ユーザービューでは、ユーザーフォームを最初に読み込むときにも <Expansion> タグが実行されます。
<Validation>	フォームに入力された値が有効であるかどうかを検証します。	Validation 規則は、フォームを送信するたびに評価されます。

<Derivation> ステートメントの例

次の 2 つの例は、Derivation 要素の使用例を示しています。

- 例 1: グローバルフィールドの信頼できるソースの指定
- 例 2: 値セットと別の値セットのマッピング

例 1:

最初の値が定義されている場合、次の例はその値を使用します。最初の値が定義されていない場合は、2 番目の値が使用されます。

```
<Derivation>
  <or>
    <ref>accounts[AD].fullname</ref>
    <ref>accounts[LDAP].fullname</ref>
  </or>
</Derivation>
```

例 2:

次の <Derivation> 要素の使用例は、条件ロジックを使用して、値のセットを別のセットにマップするフィールド定義を示します。

この例では、リソースアカウント属性 accounts[Oracle].locCode が、まず、case 要素 AUS に対して評価されます。結果が true となる場合は、返される値が 2 番目の値となり、location フィールドに表示されます。いずれの case 要素とも一致しない場合は、case 式のデフォルト値が返されます。case 式に含まれる最初の式で一致する略号が見つかった場合は、case 式に含まれる 2 番目の式の値が、switch 式の結果として返されます。

```
<Field name='location' prompt='Location'>
  <Display class='Text' />
  <Derivation>
    <switch>
      <ref>accounts[Oracle].locCode</ref>
      <case>
        <s>AUS</s>
        <s>Austin</s>
      </case>
      <case>
        <s>HOU</s>
        <s>Houston</s>
      </case>
      <case>
        <s>DAL</s>
        <s>Dallas</s>
      </case>
      <case default='true'>
        <s>unknown</s>
      </case>
    </switch>
  </Derivation>
</Field>
```

<Expansion> ステートメントの例

次の2つの例は、Expansion 要素の使用例を示しています。

- 例 1: フィールドに入力されるテキストの大文字と小文字を標準化する規則の実装
- 例 2: Expansion ロジックを非表示にする

例 1:

Expansion 規則は、フィールドに入力された情報を、リソース側で前提とされる形式、または規則によって指定される形式と一致する値に変換します。たとえば、ユーザーが名前を入力する自由形式のテキストボックスに Expansion 規則を含め、頭文字を大文字、残りの文字を小文字にそろえることができます。

フィールドでグローバル変数を使用すると、フォームの保存時に、この値を持つすべてのリソースが設定されます。このフォームを読み込むと、**Identity Manager** は各リソースから値を取り込みます (フィールドが無効化されている場合を除く)。最後のリソースの読み込みによって、フォームの値が設定されます。ユーザーがローカルに変更を加えた場合は、その変更は表示されない可能性があります。このため、フィールドの信頼できるソースとして 1 つまたは複数のリソースを指定する Derivation 規則を使用することで、正しい値を確実に属性に適用することができます。

```
<Field name='global.lastname'>
  <Display class='Text'>
    <Property name='title' value='Last Name' />
    <Property name='size' value='32' />
    <Property name='maxLength' value='128' />
    <Property name='noNewRow' value='true' />
    <Property name='required'>
      <Boolean>false</Boolean>
    </Property>
  </Display>
  <Expansion>
    <block>
      <defvar name='lname'>
        <downcase>
          <ref>global.lastname</ref>
        </downcase>
      </defvar>
      <defvar name='nlength'>
        <sub>
          <length>
            <ref>global.lastname</ref>
          </sub>
        </defvar>
      </block>
    </Expansion>
  </Field>
</pre>
```

```

        </length>
        <s>1</s>
    </sub>
</defvar>
<concat>
    <substr>
        <upcase>
            <ref>global.lastname</ref>
        </upcase>
        <s>0</s>
        <s>1</s>
    </substr>
    <substr>
        <ref>lname</ref>
        <s>1</s>
        <ref>nlength</ref>
    </substr>
</concat>
</block>
</Expansion>
</Field>

```

この XPRESS ロジックが複数のフィールドに実装される場合は、それを規則に含めることを検討してください。

例 2:

次の例には Display クラス定義が含まれないため、このフィールドは非表示にも設定されます。Display クラス定義を取り除くことで、フィールドはフォームに表示されなくなります。ただし、フィールドは依然としてフォームのアクティブ部分と見なされ、<Expansion> 式を利用してリソース属性の値を生成します。

```
<Field name='accounts[Oracle].locCode'>
  <Expansion>
    <switch>
      <ref>location</ref>
      <case>
        <s>Austin</s>
        <s>AUS</s>
      </case>
      <case>
        <s>Houston</s>
        <s>HOU</s>
      </case>
      <case>
        <s>Dallas</s>
        <s>DAL</s>
      </case>
    </switch>
  </Expansion>
</Field>
```

この例では、location フィールドによるマッピングとは逆のマッピングが行われます。

<Validation> ステートメントの例

Validation 式を使用することで、フォームに入力した値が有効であるかどうかを判断するためのロジックを指定できます。

Validation 式は、成功を示す場合は NULL を返し、失敗を示す場合は判読可能なエラーメッセージを含む文字列を返します。**Validation** 式のエラーメッセージは、フォーム最上部に赤のテキストで表示されます。

次の例には、ユーザーがフィールドに入力した年齢の値が、0 より大きいことを確認するロジックが含まれます。入力された年齢が 0 より大きいか、または 0 である場合、この式は NULL を返します。

```
<Field name='age'>
  <Validation>
    <cond>
      <lt>
        <ref>age</ref>
        <i>0</i>
      </lt>
      <s>Age may not be less than zero.</s>
    </cond>
  </Validation>
</Field>
```

メソッドの呼び出しによるリストへのデータの取り込み

単一選択リストと複数選択リストのテキストボックスには、外部リソースから取得した情報が取り込まれることがよくあります。Sun でサポートされるいずれかの **FormUtil** メソッドを呼び出すことで、この情報を動的にリストに取り込むことができます。これらの一般的なメソッドは、次のタスクを実行できます。

- リソースオブジェクト名のリストを取得する
- マップオプションを持たないリソースオブジェクトのリストを取得する
- DN 文字列を作成する
- アクセスできるオブジェクトタイプのリストを取得する
- セッション所有者がアクセスできるオブジェクトタイプのリストを取得する
- プレフィックスを持つ組織のリストを取得する
- プレフィックスを持たない組織のリストを取得する
- プレフィックスを持つ組織の表示名のリストを取得する
- ユーザーに割り当てられていないアプリケーションのリストを取得する

<Select> コンポーネントと <MultiSelect> コンポーネント、および allowedValues プロパティについては、「[リストデータの取り込み](#)」の節を参照してください。

リソースオブジェクトの名前について

リソース上の情報を検索または要求し、Identity Manager にインポートするには、Identity Manager がサポートするオブジェクト定義を使用する必要があります。

次の表は、Identity Manager でサポートされるオブジェクトタイプを示しています。

表 3-16

サポートされるオブジェクトタイプ	説明
account	ユーザーアカウント ID のリスト
Administrator_Groups	ユーザーが所属できる管理者グループの名前
Applications	アプリケーションのリスト
Distribution Lists	電子メール配信エイリアスのリスト
Entitlements	PKI 権利書のリスト
group	セキュリティオブジェクトと配信リストグループオブジェクトのリスト
Group	セキュリティグループ
groupofNames	Exchange 5.5 用の電子メール配信エイリアスのリスト
Nodes	SP2 ノードのリスト
PostOffices	GroupWise ポストオフィスのリスト
profile	トップシークレットプロファイルのリスト
PROFILE	DBA_PROFILES テーブルからの Oracle プロファイルのリスト
ROLE	DBA_ROLES テーブルからの Oracle ロールのリスト
shell	使用できる UNIX シェルのリスト
Template	NDS テンプレートのリスト
USERS	DBA_USERS テーブルからの Oracle プロファイルのリスト
UnassignedTokens	使用できる未割り当てトークンのリスト
User_Properties	ユーザープロパティ定義 (ユーザーに設定できるプロパティ) のリスト

リソースオブジェクト名のリストの取得

使用している特定のリソースに定義されたオブジェクト名のリストを取得するときは、`listResourceObjects` メソッドを使用します。マップオプションを持つ、または持たないリストを取得できます。マップオプションは、完全なリストを返す代わりに、返される値をフィルタリングして1つのコンテナに集約できるディレクトリ構造を持つリソースのみで使用されます。

サーバーのキャッシュからではなく、必ずリソースからリソースオブジェクトリストを取得するには、まず、`clearResourceObjectListCache()` メソッドを呼び出すか、`cacheList` 引数を `false` に設定します。ただし、大規模なリストでは、キャッシュを使用するほうがパフォーマンスが向上します。リソースへのアクセスは1回のみで、結果はキャッシュに格納されます。このため、キャッシュを利用することをお勧めします。

また、オブジェクトリストの要求対象となるリソースを指定する、キーと文字列値の1つまたは複数のペアのセットを指定することもできます。

次の表は、各リソースでサポートされるオブジェクトタイプを示しています。

表 3-17 サポートされるオブジェクトタイプ	
リソース	サポートされるオブジェクトタイプ
AIX	account、Group
ACF2	account
ClearTrust	account、Group、group、Administrator_Groups、Applications、Entitlements、User_Properties
Entrust	Group、Role
Exchange5.5	groupofNames
GroupWise	account、Distribution Lists、PostOffices
HP-UX	account、Group、shell
LDAP	account、Group
Oracle	USERS、ROLE、PROFILE
Natural	account、Group
NDS	account、Group
PeopleSoft	account
RACF	account、Group
SAP	account、table、profiles、activitygroups

表 3-17 サポートされるオブジェクトタイプ (続き)

リソース	サポートされるオブジェクトタイプ
SecurID	UnassignedTokens
SP2	Nodes
Solaris	account、Group、shell
TopSecret	account
VMS	account
Windows Active Directory	account、Group Active Directory のどの有効オブジェクトクラス名も、オブジェクトタイプとして指定できます (オブジェクトクラス名のリストは Active Directory スキーマドキュメントに記載されている)。返されるリストには、オブジェクトの識別名が含まれます。デフォルトでは、メソッドは、Container リソース属性によって指定されるコンテナを検索します。ただし、listResourceObjects の呼び出し時のオプションとして、コンテナを指定することができます。その名前は、コンテナの識別名としてください。この場合、そのコンテナに含まれるオブジェクトの名前のみがリストに取得されます。
Windows NT	account、Group

マップオプションを持たないリソースオブジェクトのリストの取得

マップオプションを持たないリソースオブジェクトのリストを取得するには、リソースオブジェクトのタイプとリソース名を指定します。一部のリソースは、リストのサブセットでの機能をサポートすることに注意してください。これは、開始ディレクトリを指定することで実行できます。

次の例の詳細は次のとおりです。

- <UnassignedTokens> という文字列は、取得するリソースのオブジェクトタイプを表します。一般的なリソースオブジェクトタイプには、これ以外にも groups、distribution lists、および accounts があります。
- <SecurID> という文字列は、オブジェクトタイプの取得元リソースを表します。
- null という値は、マップオプションを含めないことを表します。
- true という値は、結果をキャッシュに格納することをサーバーに伝えます。

```
<invoke name='listResourceObjects'
class='com.waveset.ui.FormUtil'>
```

```
<ref>:display.session</ref>  
  
<s>UnassignedTokens</s>  
  
<s>SecurID</s>  
  
<null/>  
  
<s>>false</s>  
  
</invoke>
```

マップオプションを持つリソースオブジェクトのリストの取得

マップオプションを持つリソースオブジェクトのリストを取得するには、リソースオブジェクトのタイプ、リソース名、および検索を開始するディレクトリを表すマップオプションを指定します。リソースは、ディレクトリベースで指定してください。

たとえば、**Software Access** ディレクトリ内のすべての **Active Directory** グループのリストを取得するには、指定したディレクトリパス (ou=Software Access, dc=mydomain, dc=com) で検索を行うマップオプションを作成します。

例：

次の例の詳細は次のとおりです。

- **Group** という文字列は、取得するリソースのオブジェクトタイプを表します。リソースオブジェクトのタイプを表す文字列は、**Available Resource Object Types** というテーブルに指定されます。
- **AD** という文字列は、オブジェクトタイプの取得元リソース名を表します。マップオプションは、リストの取得元ディレクトリを指定します。
- **true** という値は、結果をキャッシュに格納することをサーバーに伝えます。
- **false** という値は、結果をキャッシュに格納しないことをサーバーに伝えます。


```
<invoke name='listResourceObjects'  
  class='com.waveset.ui.FormUtil'>  
  <ref>:display.session</ref>  
  <s>Group</s>  
  <s>AD</s>  
  <Map>  
    // この設定により、指定したコンテナ / 組織の単位、およびその下部に  
    含まれるグループのみのリストを取得できます  
    <MapEntry key='container'  
value='LDAP://hostX.domainX.com/cn=Users,dc=domainX,dc=com' />  
  </Map>  
  <s>>false</s>  
</invoke>
```

DN 文字列の作成

指定したユーザー ID とベースコンテキストから、識別名のリスト、または単一の識別名を動的に作成できます。このメソッドでは、リストは返されません。通常は、Expansion 規則で使用されます。

DN 文字列のリストの動的な作成

ユーザー ID とベースコンテキストを指定して、DN 文字列のリストを動的に作成することができます。

次の例は、ユーザー ID とベースコンテキストを使用して DN 文字列のリストを動的に作成する方法を示しています。

次のコードでは、まず、ユーザーに追加するベースコンテキストが定義されます。

```
<Field name='baseMemberContextContractor'>
  <Default>
    <s>ou=Contractors,dc=example,dc=com</s>
  </Default>
</Field>

<Field name='baseMemberContextEmployee'>
  <Default>
    <s>ou=Employees,dc=example,dc=com</s>
  </Default>
</Field>
```

このフォームのユーザーは、次のフィールドにデータを入力します。多くの場合、これは、動的に生成したユーザー ID のリストの格納場所を表します。

```
<Field name='userIds'>
  <Display class='TextArea'>
    <Property name='title' value='UserIds' />
  </Display>
</Field>
```

次の非表示フィールドには、値を計算するロジックが含まれます。

```

<Field name='Members'>
  <Expansion>
    <switch>
      // ユーザーに割り当てられているロールを参照します
      <ref>waveset.role</ref>
      <case>
        // ユーザーに "Contractor ロール" が割り当てられている場合に、次の
        // ような DN を生成します :
        // ex: CN=jsmith,ou=Contractors,dc=example,dc=com
        <s>Contractor Role</s>
        <invoke name='buildDns' class='com.waveset.ui.FormUtil'>
          <ref>userId</ref>
          <ref>baseMemberContextContractor</ref>
        </invoke>
      </case>
      <case>
        // また、ユーザーに "Employee ロール" が割り当てられている場合は、次の
        // ような DN を生成します :
        // ex: CN=jdoe,ou=Employees,dc=example,dc=com
        <s>Employee Role</s>
        <invoke name='buildDns' class='com.waveset.ui.FormUtil'>
          <ref>userId</ref>
          <ref>baseMemberContextEmployee</ref>
        </invoke>
      </case>
    </switch>
  </Expansion>
</Field>

```

単一 DN 文字列の作成

1 つの DN を指定して、リストまたはテキストエリアにデータを取り込む buildDn メソッドを呼び出すことができます。次に例を示します。

```
<invoke name='buildDn' class='com.waveset.ui.FormUtil'>
  <s>jdoe</s>
  <s>dc=example,dc=com</s>
</invoke>
```

この例は、CN=jdoe,dc=example,dc=com という文字列を返します。

未割り当てリソースのリストの取得

指定したユーザー ID には表示する権限はあるが、現時点では割り当てられていないすべてのリソースのリストを取得するときは、getUnassignedResources メソッドを使用します。

<ref> ステートメントは、指定したユーザーに関する情報を持つビュー属性を表します。次に例を示します。

```
<invoke name='getUnassignedResources'
class='com.waveset.ui.FormUtil'>
  <ref>:display.session</ref>
  <ref>waveset.role</ref>
  <ref>waveset.original.resources</ref>
</invoke>
```

アクセス可能なオブジェクトタイプのリストの取得

セッション所有者が現時点でアクセスできるオブジェクトタイプのリストを取得するときは、getObjectNames メソッドを使用します。

検索できるオブジェクトタイプは次のとおりです。

- Account
- Administrator
- Configuration
- EmailTemplate
- Resource
- Role
- System

- TaskInstance
- User
- UserForm

すべてのオブジェクトタイプのリストは、「デバッグ」ページの「List Objects」オプションで表示できます。次に例を示します。

```
<invoke name='getObjectNames' class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
    <s>UserForm</s>
</invoke>
```

セッション所有者がアクセスできるオブジェクトタイプのリストの取得

セッション所有者がアクセスできるオブジェクト名のリストを取得するときは、getObjectNames メソッドを使用します。次に例を示します。

```
<invoke name='getObjectNames' class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
</invoke>
```

プレフィックスを持つ組織のリストの取得

プレフィックス (たとえば、TOP, TOP:IT, TOP:HR) を持つ組織のリストを取得するときは、getOrganizationsWithPrefixes メソッドを使用します。次に例を示します。

```
<invoke name='getOrganizationsWithPrefixes'
class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
</invoke>
```

プレフィックスを持たない組織のリストの取得

プレフィックス (たとえば、TOP, TOP, TOP) を持たない組織のリストを取得するときは、getOrganizations メソッドを使用します。次に例を示します。

```
<invoke name='getOrganizations'
class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
</invoke>
```

プレフィックスを持つ組織表示名のリストの取得

プレフィックスを持つ組織表示名のリストを取得するときは、`getOrganizationsDisplayNamesWithPrefixes` メソッドを使用します。

```
<invoke name='getOrganizationsDisplayNamesWithPrefixes'  
class='com.waveset.ui.FormUtil'>  
    <ref>:display.session</ref>  
</invoke>
```

ユーザーに割り当てられていないアプリケーションのリストの取得

現時点でユーザーに割り当てられていないアプリケーションのリストを取得するときは、`getUnassignedApplications` メソッドを使用します。次に例を示します。

```
<invoke name='getUnassignedApplications'  
class='com.waveset.ui.FormUtil'>  
    <ref>:display.session</ref>  
    <ref>waveset.roles</ref>  
    <ref>waveset.original.applications</ref>  
</invoke>
```

ハッシュマップの作成

`listResourceObjects` メソッドと `callResourceMethods` メソッドは、ハッシュマップを受け付けます。ハッシュマップは、`<Map>` 要素を使用して作成できます。

次の例の `<Map>` 要素は、変更されない制的なマップを作成します。

```
<Map>  
    <MapEntry name='key1' value='value1' />  
    <MapEntry name='key2' value='value2' />  
</Map>
```

`<map>` 要素を使用して、XPRESS 式でマップを作成することもできます。`<map>` 要素は、コンテンツが別の式によって定義されるマップを動的に作成することができます。

XPRESS 言語によるマップの作成については、「[XPRESS 言語](#)」を参照してください。

フィールドの無効化

フィールドを無効にすると、フィールドと、その入れ子フィールドはページに表示されなくなります。また、値の式も評価されません。無効化されたフィールドの値がすでにビューに含まれる場合、その値は変更されません。

```
<Disable></Disable>
```

注 global.* 属性は、有効なフィールドのみから取得されます。フォームがフィールドを非表示にするのではなく、動的に無効にする場合は、global.* 属性を使用して、そのフィールドの値を取得することはできません。

例：

```
<Disable>  
    <eq><ref>userExists</ref><s>true</s></eq>  
</Disable>
```

注 Disable 式は、その他の式よりも頻繁に評価されます。このため、Disable 式は、比較的単純に記述してください。データベースのロックアップのように、負荷の大きい処理を実行する Java クラスは呼び出さないようにしてください。

フィールドの非表示

フィールドを非表示に設定すると、フィールドと、その入れ子フィールドはページに表示されなくなります。ただし、フォームの処理によって、フィールドの値は設定されます。

フィールドを非表示にするには、表示しない特定のフィールドの Display プロパティを定義しません (これは条件に依存しない)。

```
<Field name='field A' />
```

値の計算

フォーム内での動的な値の計算には、次の方法があります。

- フィールド値の生成
- フォーム内での規則の使用
- フォーム内での XPRESS ステートメントの使用

フィールド値の生成

一部のフォームでは、最初は抽象的な導出フィールドのセットをユーザーに表示し、その後のフォームの送信時に、リソースアカウントの実際の属性値のさまざまなセットを生成することが必要となる場合があります。これは、フォームの展開と呼ばれます。展開されたフィールドは、よく導出フィールドと組み合わせて使用されます。

フォーム内での規則の使用

フォームでは、通常、allowedValues 表示プロパティを計算する規則を呼び出し、<Disable> 式の中でフィールドの可視性を制御したりします。フォーム内で次の項目を格納および再利用するもっとも効率的なメカニズムは、規則の使用であると考えられます。

- 企業の部署名のリスト
- デフォルト値
- オフィスビルのリスト

規則の詳細については、『Identity Manager 配備ツール』の「規則」の章を参照してください。

XPRESS ステートメントの使用

XPRESS 言語は、式とスクリプトを記述するための XML ベースの言語です。この言語で記述されたステートメントは式と呼ばれ、フォームにデータ変換機能を追加したり、ワークフローやフォームなどの Identity Manager オブジェクトに状態遷移ロジックを組み込んだりするために、Identity Manager 全体で使用されます。

XPRESS は、XML に基づいた構文を使用する関数言語です。この言語で記述されるすべてのステートメントは、0 個以上の引数をとる関数を呼び出し、値を返します。製品には関数が組み込まれていますが、新しい関数を定義することもできます。XPRESS は、任意の Java クラスでのメソッドの呼び出しと、式に含まれる Javascript の評価もサポートします。

XPRESS の機能の詳細については、「[XPRESS 言語](#)」を参照してください。

XPRESS を使用する理由

式は、主に次の Identity Manager タスクに使用されます。

- **エンドユーザーと管理者のフォームのカスタマイズ**: フォームは、フィールドの可視性の制御と、表示および保存できる形式へのデータの変換に XPRESS を使用します。
- **ワークフローでのコントロールフローの定義**: ワークフローは、ワークフロープロセスの実行手順の順序を決定する遷移条件の定義に XPRESS を使用します。

- **ワークフローアクションの実装**: ワークフローアクションは、XPRESS を使用して実装できます。アクションの式は、単純な計算を実行するだけでなく、Java クラスまたは JavaScript を呼び出して、複雑な処理を実行することもできます。

これらの要素に含まれる式は、Identity Manager 全体で使用できます。

式の例

次の例の `<add>` 要素は、`add` という XPRESS 関数の呼び出しを表します。

```
<add> <ref>counter</ref> <i>10</i> </add>
```

この関数には、2 つの引数が渡されます。

- **最初の引数**- `ref` という関数の呼び出しによって定義される値。`ref` 関数の引数は、変数名のリテラル文字列です。`ref` 関数から返される値は、`counter` という変数の現在値です。
- **2 番目の引数**- `i` という関数の呼び出しによって定義される値。`i` 関数の引数は、整数のリテラル文字列です。`i` 関数から返される値は、`10` という整数です。

`add` 関数から返される値は、`counter` 変数の現在値に整数 `10` を加えた結果となります。すべての関数呼び出しは、次の処理で使用される値を返します。たとえば、`ref` の呼び出しによってカウンタの値が得られると、`<i>` 呼び出しは整数 `10` を返し、`<add>` 呼び出しは 2 つの呼び出しの結果を合計します。

フォームに組み込まれた式の例

次の例は、Identity Manager フォームに組み込まれた XPRESS ロジックの使用例を示しています。XPRESS は、ブラウザに表示するロール関連の選択項目を生成するいずれかの `FormUtil Java` メソッドの呼び出しに使用されています。式全体が `<expression>` というタグに囲まれていることに注目してください。

```
<Field name='waveset.role'>
  <Display class='Select' action='true'>
    <Property name='title' value='Role' />
    <Property name='nullLabel' value='None' />
    <Property name='allowedValues'>
      <expression>
        <invoke class='com.waveset.ui.FormUtil'
name='getRoles'>
          <ref>:display.session</ref>
          <ref>waveset.original.role</ref>
        </invoke>
      </expression>
    </Property>
  </Display>
</Field>
```

フォームへのガイダンスヘルプの追加

Identity Manager には、次の 2 種類のオンラインヘルプが用意されています。

- タスク関連の情報を示すヘルプ。Identity Manager のメニューバーからアクセスできます。このヘルプを編集することはできません。
- ガイダンス (ポップアップヘルプ)。フィールドレベルのヘルプです。フィールドまたはエリアの左側に表示されるガイダンスアイコン (i) からアクセスできます。

コンポーネントにガイダンスヘルプを指定する方法

ガイダンスヘルプのテキストは、どのコンポーネントにも関連付けることができます。ただし、現時点では、表示に EditForm コンテナが必要です。ガイダンスヘルプのテキストは、次の 3 つの方法で指定できます。

- help プロパティを使用して、コンポーネントに明示的に割り当てます。「[help プロパティの使用](#)」の節を参照してください。

- ヘルプカタログ内のエントリを参照する `helpKey` プロパティを使用して、コンポーネントに間接的に割り当てます。「[helpKey プロパティの使用](#)」の節を参照してください。
- コンポーネントの `title` プロパティとヘルプカタログ内のエントリを一致させることで、コンポーネントに間接的に割り当てます。「[コンポーネントの title プロパティとヘルプエントリの一致](#)」の節を参照してください。

help プロパティの使用

ガイダンステキストを指定するもっとも簡単な方法は、`help` プロパティの使用です。XML フォームの使用時は、これは、大容量である可能性もある XML にヘルプテキストを組み込むことを意味し、この場合、ヘルプテキストの表示と編集は比較的困難になります。これに代わる方法として、ヘルプカタログにヘルプテキストを保存し、コンポーネントからカタログ内のエントリを参照する方法があります。

helpKey プロパティの使用

ヘルプカタログ内の各エントリは、一意の名前またはキーを持ちます。これを利用し、`helpKey` プロパティを設定して、コンポーネントからカタログエントリを参照することができます。この設定を行うと、システムはこの値が現在のヘルプカタログ内のキーの値であるものと見なし、コンポーネントを表示するときに関連ヘルプテキストを読み込みます。

コンポーネントの title プロパティとヘルプエントリの一致

コンポーネントのタイトルと一致する、ヘルプカタログ内のキーの値を使用して、カタログエントリをコンポーネントに自動的に関連付けることもできます。XML フォームの使用時は、`Property` 要素を使用して、コンポーネントのタイトルを明示的に指定できます。このプロパティが設定されていない場合は、`Field` 要素に含まれる `prompt` 属性の値が適用されます。

Identity Manager によるヘルプリソースの評価

コンポーネントを表示するとき、システムはまず、`help` プロパティが設定されているかどうかを調べます。設定されている場合は、そのテキストが使用されます。次に、`helpKey` が設定されているかどうかを調べ、キーを利用してエントリのカタログを検索します。`helpKey` プロパティが設定されていない、または、一致するキーがカタログに見つからない場合は、システムは、コンポーネントのタイトルをヘルプキーとして検索を再試行します。

ヘルプカタログの使用方法

ヘルプカタログは、XML テキストが記録されたファイルです。特定のコンポーネントで 사용되는ヘルプカタログは、次のように決定されます。

- ページプロセッサシステムは、HTML を生成する前に、`HtmlPage` オブジェクトの `helpCatalog` プロパティにカタログ名を割り当てることで、そのカタログを指定できます。

- helpCatalog プロパティの値が設定されていない場合は、HtmlPage は要求 URL に基づく命名規則を利用してカタログの特定を試みます。HtmlPage は、URL (通常は .jsp ページの URL) を調べ、ベースコンテキストの先頭の trailing.jsp を削除します。次に、残りの URL の先頭には Identity Manager のインストールディレクトリ、末尾には -help.xml を追加します。

XML フォームの設計者は、フォームプロセッサシステムを利用して、適切なカタログを自動的に特定できます。しかし、使用するカタログファイルを、より明示的に制御することが必要となる場合もあります。フォームで使用されるヘルプカタログは、Form 要素の help 属性で指定できます。たとえば、次のようにします。

```
<Form name='Task Launch Form' help='task/common-help.xml'>
```

```
...
```

```
</Form>
```

これは、それぞれが異なる要求 URL で参照される複数のフォーム間で 1 つのヘルプカタログを共有する場合に便利です。

ヘルプカタログの構文

ヘルプカタログファイルの構文は、比較的簡単です。これには、1 つまたは複数の Page 要素を持つドキュメント要素、WebHelp を使用します。これらの Page 要素には、1 つまたは複数の Item 要素が含まれます。それぞれの Item 要素は、ヘルプテキストのカタログキーを表す key 属性を持ちます。Item 要素の内容は、ヘルプテキストです。

ヘルプカタログの構文を利用することで複数の Page 要素を指定できますが、現時点では、複数の要素をサポートするページプロセッサシステムは存在しません。

簡単なヘルプカタログファイルの例

次に、簡単なヘルプカタログファイルの例を示します。

```

<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE WebHelp>

<WebHelp>
  <Page name='user/changeAll.jsp'>

    <Item key='Account Information'>
      Miscellaneous information about your accounts.
    </Item>

    <Item key='Passwords'>
      <![CDATA[
        <b>Passwords</b><br>Enter and confirm a new password.
      ]]>
    </Item>

  </Page>
</WebHelp>

```

Item 要素へのヘルプテキストの指定

Item 要素には、次の 2 つの方法でヘルプテキストを指定できます。

- **単純なテキストを指定します。**単純なテキストを指定するときは、前述の例の Account Information のように、Item タグの間にテキストを入力します。ブラウザは、ヘルプテキストの先頭および末尾に含まれる改行文字を削除します。
- **HTML マークアップをヘルプテキストに組み込むことで、フォントを指定したり、改行を挿入したりします。**この方法では、XML が呼び出す CDATA というマークが付けられたセクションのテキスト (前述の例の Passwords を参照) をより柔軟に囲むことができます。「CDATA」というテキストは、ヘルプテキストの一部ではありません。これは、内容を解析対象外文字データとして処理するようにパーサーに指示する、特別な XML 宣言です。この場合、マークが付けられたセクションに含まれる などの要素は、XML 要素としてではなく、リテラルテキストとして解釈されます。

CDATA のマークが付けられたセクションを使用しない場合は、次の例のように、ヘルプテキスト中の右開き不等号を代替文字列 `<` で表記しなければなりません。

```
&lt;b>Passwords&lt;/b>&lt;br>Enter and confirm a new password.
```

ガイドンスヘルプの優先指定

カスタムメッセージカタログを使用して、ポップアップウィンドウに表示されるガイドンステキストを優先指定できます。カスタムメッセージカタログに `defaultCustomCatalog` という名前を付けると、Identity Manager はこれを認識し、このカタログを自動的に使用します。また、別のカタログ名を付け、`customMessageCatalog` という属性名の後に、System Configuration オブジェクトの値としてそのカタログ名を指定することもできます。

たとえば、次のようにします。

```
<Attribute name='customMessageCatalog' value= 'sampleCustomCatalog' />
```

次に示すカスタムカタログの例は、「ヘルプ」タブの吹き出しテキストとして「Waveset Lighthouse 4.1 SP2」と表示します。

```
<Waveset>
  <Configuration name="sampleCustomCatalog">
    <Extension>
      <CustomCatalog id="defaultCustomCatalog" enabled="true">
        <MessageSet language="en" country="US">
          <Msg id="UI_END_USER_VERSION">Waveset Lighthouse 4.1
            SP2</Msg>
        </MessageSet>
        <MessageSet language="es" country="ES" variant=
          "Traditional">
          <Msg id="UI_END_USER_VERSION">Sun Lighthouse 4.1 SP2</Msg>
        </MessageSet>
        <MessageSet language="fr">
          <Msg id="UI_END_USER_VERSION">Waveset Lighthouse 4.1
            SP2</Msg>
        </MessageSet>
      </CustomCatalog>
    </Extension>
  </Configuration>
</Waveset>
```

注 吹き出しテキストにバージョン番号を表示しないようにするには、
UI_END_USER_VERSION メッセージの文字列を削除します。

フォームに関連するその他のタスク

フォーム関連では、ほかに次のようなタスクがあります。

- FormUtil メソッドの呼び出し
- フォームへの Javascript の挿入
- ユーザーまたはオブジェクトの存在確認

FormUtil メソッドの呼び出し

FormUtil クラスは、フォームオブジェクトを使用して XPRESS 式から呼び出せるユーティリティメソッドの集合です。これは、許可される値のリストへの取り込みと、入力の検証に使用できます。一般に、FormUtil メソッドは、リストまたはフィールドの許可される値の定義を支援するために呼び出されます。

これらのメソッドの一覧と説明については、「[FormUtil メソッド](#)」を参照してください。

```
<invoke class = 'com.waveset.ui.FormUtil'  
    name = 'listResourceObjects'>  
  
</invoke>
```

この name フィールドは、メソッドの名前を表します。

フォーム内でのこれらのメソッドの使用例については、「[非表示コンポーネントの使用](#)」の節を参照してください。

フォームへの JavaScript の挿入

事前にフォーマットされた Javascript をフォームに挿入するとき、特に、コンポーネントの `onClick` または `onChange` プロパティを使用し、カスタム Javascript 機能を使用するときは、`<JavaScript>` コンポーネントを使用します。

コンポーネントには、JavaScript テキストを格納できる `script` という拡張プロパティがあります。

これとは別に、ソースプロパティを設定して JavaScript を含めることもできます。これは、ベースコンテキストに関連する URL フラグメントを含んだ文字列です。指定されたファイルに含まれる JavaScript が、ブラウザに読み込まれます。

オブジェクトまたはユーザーの存在確認

アクションを実行する前に、オブジェクトの存在を確認できると便利かもしれません。たとえば、新規ユーザーを作成したり、フィールドに入力したマネージャー名が有効であることを確認したりする前に、ユーザー名が Identity Manager に存在するかどうかを確認できます。

オブジェクトの存在を確認するときは、testObject メソッドを使用します。このメソッドの使用時にオブジェクトタイプを指定するときは、[204 ページの「アクセス可能なオブジェクトタイプのリストの取得」](#)の節に記載されているオブジェクトタイプを使用してください。次の例では、`<s>User</s>` というユーザータイプが指定されています。2 番目の文字列は、オブジェクトタイプの値 (この例では jdoe) を指定します。

例:

```
<invoke name='testObject' class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
    <s>User:</s>
    <s>jdoe</s>
</invoke>
```

オブジェクトが検出されると、testObject メソッドは **true** を返します。検出に失敗した場合は、NULL が返されます。

ユーザーの存在を確認するときは、testUser メソッドを使用します。検索するユーザーオブジェクトの名前は、`<s>` 要素で指定されます。次に例を示します。

```
<invoke name='testUser' class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
    <s>jdoe</s>
</invoke>
```

検出に成功した場合は、**true** が返されます。検出に失敗した場合は、NULL が返されます。

ウィザードフォームとタブ付きフォーム

ウィザードフォームとタブ付きフォームはどちらも、扱いにくい単一ページのフォームを、管理がより容易な、複数ペインのフォームに変えるメカニズムです。どちらのフォームも、論理セクションまたはページの間にセパレータを持ちます。これらのページセパレータは、Tabbed User Form のようにフォームの上部に配置されるタブである場合と、前後のページに移動するためのナビゲーションボタンを利用してページを切り替えながらユーザーに操作を促すウィザードである場合があります。

デフォルトの Tabbed User Form の XML バージョンについては、この章の「[Tabbed User Form](#)」を参照してください。

ウィザードフォームとは

次のような場合は、タスクから複数のフォームを呼び出すよりも、ウィザードフォームを使用するほうが便利です。

- ページ間の遷移ロジックが単純である
- ページ間で特権システム呼び出しが必要とされる

ウィザードフォームには、次に説明する 2 つのボタン行があります。

表 3-18 最初のボタン行

ボタン行	説明
上の行	フォームペインを切り替えるための「次へ」および「戻る」ボタン
次の行	次の表に示される、標準のユーザーフォームボタン。この行は、noDefaultButtons オプションを true に設定して、独自のボタンを実装することで制御できます。

2 番目のボタン行は、次のように異なります。

表 3-19 2 番目のボタン行

ウィザードのページ	デフォルトボタン
最初のページ	「次へ」、「キャンセル」
中間のページ	「戻る」、「次へ」、「キャンセル」
最後のページ	「戻る」、「OK」、「キャンセル」

ウィザードフォームの実装

ウィザードフォームの構文は、Tabbed User Form の構造によく似ています。ウィザードフォームを作成する方法は次のとおりです。

1. 最上位レベルのコンテナに、TabbedPanel の代わりに WizardPanel 表示クラスを割り当てます。
2. noCancel プロパティを **true** に設定します。
3. ウィザードのページを格納した、1 つまたは複数の EditForm フィールドを定義します。

次の例には、説明のためのコメントが含まれます。

```

<Form>

  <Display class="HtmlPage"/> ----- これを設定しないと、インデントと色に問題
  が生じます

  ...

  <Field name='MainTabs'> -- タブページを格納した最上位コンテナの名前

    <Display class='TabPanel' /> -- 最上位コンテナの表示クラス、TabPanel ま
    たは WizardPanel のいずれか

    <Field name='Identity'> -- タブのラベル

      <Display class='EditForm'> -- 各 " ページ " は編集フォームである必要があ
      ります

        <Property name='helpKey' value='Identity and Password Fields' />

      </Display>

    <Field name='waveset.accountId'>

      <Display class='Text'>

        <Property name='title' value='_FM_ACCOUNT_ID' />

      </Display>

      <Disable> <ref>waveset.id</ref> </Disable>

    </Field>

    ...

  </Field>

  ...

</Field>

```

ヒントおよび対策

- 検証エラーは、属性が表示されるページではなく、ユーザーが最後に表示したページに表示されます。これに対応するには、適切なページに戻るよう促すように、検証メッセージに説明を追加します。
- 複雑なウィザードでは、プロセス全体のどの段階にいるかを視覚的にユーザーに表示してください。各ページの上部に表示されるラベルまたはセクション見出しを使用して、「ページ 1」のようなテキストを表示します。

- ウィザードフォームでは、条件付きナビゲーションを使用しないでください。この機能の実装が必要なときは、WizardPanel の直下にある各子要素の **Disable** 式を使用します。たとえば、次のようにします。

```
<Field name='Page2'>
    <Display class='EditForm' />
    <Disable><neq><ref>showPage2</ref><s>true</s></neq></Disable>
    ...
</Field>
```

- フィールドやボタンは、ゲート変数を設定する、直前のページに配置します。無効化されたページは、遷移ロジックから自動的に削除されます。

デフォルトのユーザー作成フォームとユーザー編集フォームの代替フォーム

管理者がデフォルトのユーザーフォームを使用してユーザーを編集する場合、管理者がユーザーアカウントの編集を開始すると同時に、そのユーザーが所有するすべてのリソースがフェッチされます。ユーザーが多くのリソースにアカウントを持つ場合、この処理に時間がかかる可能性があり、パフォーマンスを低下させかねません。このような環境に **Identity Manager** を配備したときは、デフォルトのユーザー作成インタフェースやユーザー編集インタフェースの代わりに、スケーラブルフォームを使用してみてください。

スケーラブルフォームの概要

スケーラブルフォームは、ユーザー数とリソース数が多い環境で、**Identity Manager** のユーザー作成インタフェースとユーザー編集インタフェースのパフォーマンスを改善するようにカスタマイズされたフォームです。パフォーマンスの改善は、次のような機能によるものです。

- リソースの差分フェッチ
- ユーザーリソースの選択的な参照
- 複数リソースの編集

Identity Manager には、デフォルトのユーザー編集フォームとユーザー作成フォームのスケーラブルバージョンが用意されています。

リソースの差分フェッチ

リソースの差分フェッチは、ネットワーク接続などの方法で、Identity Manager サーバーがリソースから直接、情報を取得する手段の 1 つです。管理者がデフォルトのユーザーフォームを使用してユーザーを編集する場合、通常は、管理者がユーザーアカウントの編集を開始するのと同時に、そのユーザーが所有するすべてのリソースがフェッチされます。これに対し、スケーラブルフォームの設計に込められた目的は、管理者が表示または変更するリソースのみをフェッチすることで、フェッチに制限をかけることです。

選択的な参照

選択的な参照は、スケーラブルフォームに組み込まれたもう 1 つの機能です。管理者は、所有ロール、リソースタイプ、またはリソースリストに基づいて、リソースを差分表示できます。

複数リソースの編集

複数リソースの編集により、管理者はリソースのサブセットを選択してリソース属性を編集できます。管理者は、ロール、リソースタイプ、またはリソースリストに基づいてサブセットを選択できます。

スケーラブルフォームの使用

次のようなケースでは、スケーラブルフォームの使用を検討してください。

- **管理者が、多数のリソースアカウントを持つユーザーを手動で編集する場合。**これらの状況でスケーラブルフォームを実装することで、管理者は、ユーザーのすべてのリソースアカウントのデータをフェッチするオーバーヘッドを回避し、特定のリソースアカウントを選択的に編集できます。このメカニズムは、ユーザーに関連付けられている特定のリソースタイプが、その他のリソースタイプと比較して応答が極端に遅い場合に特に便利です。
- **ActiveSync などのカスタムプロビジョニングプロセスが、特定のリソースの更新のみを目的としている場合**

注	別のリソースを参照する属性がフォームロジックに含まれる場合は、スケーラブルフォームを使用しないでください。この設定では、対象リソースをまとめてフェッチしない限り、このような相互参照属性に最新データが取り込まれません。

別のリソースを参照する属性がフォームロジックに含まれる場合は、スケーラブルフォームを使用しないでください。この設定では、対象リソースをまとめてフェッチしない限り、このような相互参照属性に最新データが取り込まれません。

新規ユーザーは、最初はリソースを持たないため、ユーザー作成フォームのスケラブルバージョンには、限定的ではありますが、標準のデフォルトバージョンに対してメリットがあります。

使用できるスケラブルフォーム

Identity Manager には、次の 2 つのスケラブルユーザーフォームが用意されています。各フォームについては後述します。

- **Dynamic Tabbed User Form**。デフォルトの **Tabbed User Form** の代替フォームです。
- **リソースステابلユーザーフォーム**。デフォルトの **Tabbed User Form** とユーザー編集フォームの代替フォームです。

Dynamic Tabbed User Form

管理者が編集を開始すると同時にすべてのリソースをフェッチする、デフォルトの **Tabbed User Form** の代替フォームです。**Dynamic Tabbed User Form** には、差分フェッチ機能と、リソースタイプに基づいて複数のリソースを編集する機能が用意されています。

注	実装の詳細については、WSHOME/samples/form_name.xml の各ユーザーフォームに関するコメントを参照してください。

フォームのインポートとマッピング

Dynamic Tabbed User Form によるデフォルトの **Tabbed User Form** の代用には、次の 3 つのフォームが関連します。

表 3-20 Dynamic Tabbed User Form と関連するフォーム

フォーム	説明
Dynamic Tabbed User Form	デフォルトの Tabbed User Form の機能を持ちますが、リソースタイプごとに 1 つのタブを動的に作成します。
動的ユーザーフォーム	ユーザーフォームのリソースタイプタブを作成するためのフィールドを持ちます。
動的フォームの規則ライブラリ	指定されたユーザーフォームを持たないリソースの属性を動的に出力するための規則ライブラリを持ちます。

表 3-20 Dynamic Tabbed User Form と関連するフォーム (続き)

フォーム	説明
動的リソースフォーム	現時点で Dynamic Tabbed User Form との互換性を持つすべてのフォームを持ちます。ユーザーは、このリストをカスタマイズできます。

Dynamic Tabbed User Form のインストールは、フォームのインポートと、フォームマッピングの変更という 2 段階で行われます。

手順 1: フォームのインポート

1. Identity Manager のメニューバーから、「設定」>「変換ファイルのインポート」を選択します。
2. ファイル名 (dynamicformsinit.xml) を入力するか、「参照」をクリックして、./sample ディレクトリ内の dynamicformsinit.xml ファイルを探します。
3. 「インポート」をクリックします。インポートに成功したことを示すメッセージが表示されます。

手順 2: フォームマッピングの変更

ユーザーフォームをエンドユーザーに割り当てるには、2 つの方法があります。これらのフォームマッピングを編集する方法は、実際の環境で管理者がこれらのフォームをどのように使用するかによって異なります。次の方法があります。

- すべての管理者のデフォルトユーザーフォームとしてスケーラブルユーザーフォームを割り当てる。この方法で割り当てるときは、「[デフォルトユーザーフォームとしてのスケーラブルユーザーフォームの割り当て](#)」を参照してください。Identity Manager の管理者は、すべての管理者が使用する 1 つのフォームを割り当てることができます。
- 特定の管理者に、個別にスケーラブルユーザーフォームを割り当てる。この方法で割り当てるときは、「[管理者ごとのスケーラブルユーザーフォームの割り当て](#)」を参照してください。

デフォルトユーザーフォームとしてのスケーラブルユーザーフォームの割り当て

1. メニューバーから、「設定」>「フォームおよびプロセスマッピングの設定」を選択します。
2. 「フォームマッピング」セクションで、「フォームタイプ」列から userForm を探します。
3. 「マップされるフォーム名」列の下の入力ボックスで、Dynamic Tabbed User Form を指定します。

管理者ごとのスケーラブルユーザーフォームの割り当て

1. メニューバーで、「アカウント」>「ユーザーの編集」を選択します。
2. ユーザーを、次のいずれかの方法で選択します。
 - ユーザー名をクリックしてから「編集」をクリックします。
または
 - ユーザー名を右クリックしてポップアップメニューを表示させ、「編集」メニューオプションを選択します。
3. デフォルトのユーザー編集フォームが表示されたら、「セキュリティー」タブをクリックします。
4. 「ユーザーフォーム」フィールドから「Dynamic Tabbed User Form」を選択します。
5. 「保存」をクリックして、設定を保存します。

リソーステーブルユーザーフォーム

リソーステーブルユーザーフォームには、ユーザー編集フォームのスケーラブルバージョンのほとんどの駆動ロジックが含まれます。このフォームは、差分フェッチ機能と、リソースタイプに基づいて複数リソースを編集する機能を実装します。

実装の詳細については、`WSHOME/samples/resourcetableformsinit.xml` に記載されているコメントを参照してください。

フォームのインポートとマッピング

リソーステーブルユーザーフォームによるデフォルトの Tabbed User Form の代用には、次の 5 つフォームが関連します。

表 3-21 リソーステーブルユーザーフォームと関連するフォーム

フォーム	説明
リソーステーブルユーザーフォーム	ナビゲーション、差分フェッチ、およびフォームのレイアウトに使用される、グローバルに利用できるすべてのフィールドを持ちます。リソース関連のその他すべてのスケーラブルフォームは、このメインフォームによって駆動されます。
リソーステーブルユーザーフォームライブ러리	リソーステーブルユーザーフォームのプライマリフィールドを持ちます。ブレッドクラムフィールドとナビゲーションフィールドが含まれます。
リソーステーブルアカウント情報フォーム	リソーステーブルフォームのアカウント情報セクションのフィールドを持ちます。

表 3-21 リソーステーブルユーザーフォームと関連するフォーム (続き)

フォーム	説明
リソーステーブル規則ライブラリ	ユーザーのリソースの取得、カウント、分析に適用される規則ライブラリを持ちます。これは、主にユーザーフォームライブラリで使用され、ロールとリソースに基づいてテーブルデータを生成します。
リソーステーブルユーザーティリティーライブラリ	たとえば、ロールごと、またはタイプごとにリソースを取得する規則のように、リソーステーブルフォームでの選択プロセスで使用される規則を持ちます。

リソーステーブルユーザーフォームのインストールは、フォームのインポートと、フォームマッピングの変更という 2 段階で行われます。

手順 1: フォームのインポート

1. Identity Manager のメニューバーから、「設定」>「交換ファイルのインポート」を選択します。
2. ファイル名を入力するか、「参照」をクリックして、
WSHOME/sample/resourcetableforms.xml を探します。このファイルをインポートすると、次の内容もインポートされます。

手順 2: フォームマッピングの変更

1. メニューバーから、「設定」>「フォームおよびプロセスマッピングの設定」を選択します。
2. 「フォームマッピング」セクションで、「フォームタイプ」列から userForm を探します。
3. 「マップされるフォーム名」列の下の入力ボックスで、リソーステーブルユーザーフォームを指定します。

スケーラブルフォームのカスタマイズ

スケーラブルユーザーフォームのインポートとマッピングが完了したら、それをカスタマイズしてください。差分フェッチを有効にするには、次の情報を指定しなければなりません。

- **最初にフェッチするリソースアカウント。** フェッチの対象とするリソース名を指定するには、TargetResources フォームプロパティを使用します。
- **最終的な保存の段階で更新される操作。**

動的なユーザーフォームとリソーステーブルユーザーフォームはどちらも、ユーザーのリソースに固有の属性を表示するのに、リソース固有のフォームを使用します。次のユーザーフォームは、スケーラブルフォームで使用するために WSHOME/sample/forms ディレクトリに追加されました。

- ./ACF2UserForm.xml
- ./ActivCardUserForm.xml
- ./ADUserForm.xml
- ./AIXUserForm.xml
- ./BlackberryUserForm.xml
- ./ClearTrustUserForm.xml
- ./Exchange55UserForm.xml
- ./HP-UXUserForm.xml
- ./NDSUserForm.xml
- ./OS400UserForm.xml
- ./PeopleSoftCompIntfcUserForm.xml
- ./RACFUserForm.xml
- ./SAPPortalUserForm.xml
- ./SolarisUserForm.xml
- ./SunISUserForm.xml
- ./TopSecretUserForm.xml

これらのフォームは、Dynamic Tabbed User Form とリソーステーブルユーザーフォームの両方とともに自動的にインポートされます。

配備したシステムが上記以外のリソースタイプを使用している場合は、スケーラブルフォームには、スキーママッピングに指定されているすべての属性名と値を示すデフォルトのユーザーフォームが表示されます。上記以外の既存のカスタムリソースユーザーフォームを使用するときは、スケーラブルフォームとの互換性を維持するために、特定の修正を加えてください。次に、互換性の維持に必要な手順の一部について説明します。

注	この修正の例として、上記リストのいずれかのフォームを参照してください。

スケーラブルユーザーフォームとの互換性を維持するためのリソースフォームのカスタマイズ

独自にカスタマイズしたリソースフォームを追加して、Dynamic Tabbed User Form またはリソーステーブルユーザーフォームと併用する場合は、次の一般的な手順に従ってください。

手順 1: 動的リソースフォームの修正

独自のリソースフォームを追加する方法については、dynamicformsinit.xml ファイルを参照してください。このファイルで動的リソースフォームを検索し、フォームに記載されている手順を実行します。

注	フォームには手順の説明がコメントとして記載されていますが、フォームのインポート後は表示されません。

手順 2: リソースフォームの修正

前述のリストに含まれないフォームを使用している場合は、互換性を維持できるようにリソースフォームに修正を加えます。前述のリストに含まれるいずれかのファイルを例として参照してください。方法は、各リソースフォームの冒頭に記載されています。

Tabbed User Form のカスタマイズ: 属性エリアへのパスワードフィールドの移動

同時に異なるパスワードを持つ 2 つのリソースを更新するには、割り当てられているリソースごとにパスワードフィールドを作成します。たとえば、「アカウント」ページの Exchange リソースの属性エリアに「Exchange」パスワードフィールドを設けることによって、その他のリソースとは別に設定できるパスワードポリシーを適用できます。

デフォルトパスワードポリシーの表示

デフォルトでは、パスワードポリシーの情報は、「アカウント」>「ID」タブに次のように表示されます。

パスワードフィールドを、デフォルトの位置である ID エリアから属性エリアに移動するには、次の 3 段階の手順を実行して、Identity Manager のデフォルトのパスワード同期メカニズムを無効にしなければなりません。

1. manualPasswordSynchronization チェックアウトプロパティを設定します。

2. **Tabbed User Form** に **Field** コンポーネントと **FieldLoop** コンポーネントを追加します。

3. **Tabbed User Form** に、リソースに固有のパスワードフィールドを追加します。

次に、これらの手順について詳しく説明します。

手順 1: *manualPasswordSynchronization* チェックアウトプロパティの設定

フォームに次のプロパティを追加して、`manualPasswordSynchronization` ビューチェックアウトオプションを設定します。

```
<Form>

  <Properties>

    <Property name='manualPasswordSynchronization' value='true' />

    ...

  </Properties>

  ...

</Form>
```

`manualPasswordSynchronization` を **true** に設定すると、**Identity Manager** は、パスワードシンクロナイザを使用する代わりに、リソースごとのパスワードフィールドを表示します。

手順 2: パスワードの同期の無効化

「パスワード」ビューの下に **selectAll** フラグをオフにすることで、パスワードの同期を無効にできます。このためには、デフォルトフォームに次のフィールドを追加します。

```

<Field name='password.selectAll'>
  <Comments>
    同期を行わないように、selectAll フラグを強制的にオフにします。
    ビューハンドラによって true に設定されることがあるため、この無効化が必要です。
  </Comments>
  <Expansion><s>false</s></Expansion>
</Field>
<FieldLoop for='res'>
  <expression>
    <remove>
      <ref>password.targets</ref>
      <s>Lighthouse</s>
    </remove>
  </expression>
  <Comments>
    また、新規アカウントではビューハンドラによってデフォルトで true に設定される
    ため、個々の選択フラグも強制的に false に設定し、リソースごとにパスワードプロ
    ンプトを表示する必要があります。
  </Comments>
  <Field name='password.accounts[${res}].selected'>
    <Expansion><s>false</s></Expansion>
  </Field>
</FieldLoop>

```

手順 3: 「属性」 ページへのリソース固有のパスワードフィールドの追加
 各リソースに固有のパスワードフィールドを次のように記述します。

```

<Field name='accounts[resname].password'>

```

カスタマイズしたフォームの検証

実行時環境に実装する前に、編集したフォームに関する情報を、次の方法で収集できます。

- エラーロギングを使用して、フォームフィールド内の式ステートメントのエラーを確認します。
- フォームエディタを使用して、個々の式ステートメントの構文を検証します。フォームエディタは、パーサーから返された構文エラーメッセージをポップアップウィンドウに表示します。フォームエディタの使用方法については、フォームエディタに関連するオンラインヘルプを参照してください。

エラーロギングの有効化と無効化

Identity Manager のエラーロギングユーティリティーは、フォームに含まれる式の構文に関する問題を標準出力に出力します。XPRESS のトレースを有効にすると、`<block>` タグを使用して、フォームのサブセットの XPRESS ステートメントにログメッセージを限定できます。XPRESS ステートメントの処理の詳細情報を取得するには、`waveset.properties` ファイルの設定オプション、`xpress.trace` を `true` に設定します。このオプションを `true` に設定すると、XPRESS ステートメントのすべての評価が、トレースメッセージとしてコンソールに出力されます。これは、XPRESS API によるトレースを有効にするためにコードを変更できない実行中アプリケーションで評価される、ステートメントのデバッグに使用できます。

すべての XPRESS フィールドの XPRESS トレースは、コマンド行または Identity Manager の管理者インタフェースから有効にできます。この方法で有効にしたトレースは、すべてのフィールドに影響します。ログメッセージをフォームのサブセットに限定するときは、`<block>` タグセットを使用して、エラートレースの対象を `<block></block>` タグ内のコードのみに制限します。

Identity Manager のすべての式の評価に関するエラーロギングをコマンド行から有効にする方法は次のとおりです。

1. 編集する `config/waveset.properties` ファイルを開きます。
2. `xpress.trace=false` という行を探します。
3. `false` という値を `true` に変更します。
4. ファイルを保存します。
5. アプリケーションサーバーを再起動します。

これとは別に、Identity Manager の管理者インタフェースからエラーロギングの有効化と無効化を切り替えることもできます。

1. Configurator として Identity Manager にログインします。

2. 「デバッグ」を選択して「Debug」ページを開きます。

3. 「デバッグ」ページで、「**Reload Properties**」を選択します。

XPRESS のトレースを無効にするときは、`xpress.trace` の値を `false` に変更し、`waveset.properties` ファイルを読み込みなおします。

フォームとフォームフィールドのサンプル

ここでは、製品に付属するデフォルトフォームのサンプルを示します。また、環境にサンプルフォームを組み込む方法についても説明します。

注 Identity Manager の特定バージョンに付属するフォームのバージョンによっては、これらのサンプルに若干の相違がある場合があります。

- Tabbed User Form
- エンドユーザーメニューフォーム
- Anonymous User Menu Form

ユーザーフォームライブラリ

フォームは、フォーム自体としてではなく、フィールドの集合を格納したコンテナとして使用できます。Identity Manager は、ユーザーフォームライブラリというオブジェクトを利用して、フォームのこの用途をサポートしています。このライブラリには、パスワードの変更などに使用される、リソースの詳細な選択に関連する複雑なフィールドが含まれます。

次のリストは、ユーザーライブラリに割り当てられている各ライブラリの概要を示しています。

ユーザーライブラリ	ユーザーフォームのプライマリライブラリ。このライブラリは、このリストに示されるその他のライブラリを含み、秘密質問の回答を表示、編集するための AuthenticationAnswers フィールドを定義します。
パスワードライブラリ	パスワードの仕様と同期に関するフィールドを持ちます。
アカウント概要ライブラリ	ユーザーに関連付けられているアカウントに関する、読み取り専用の概要情報を表示するフィールドを持ちます。
アカウントリンクライブラリ	アカウントのリンクと、リソースごとの複数のアカウントに関するフィールドを持ちます。

ユーザーセキュリティ
ライブラリ

機能、フォームの割り当て、承認の転送など、ユーザー
セキュリティに関するフィールドを持ちます。

ユーザーフォームライブラリ

このライブラリには、次のような、リソースアカウントビューに関連するフィールドのみが含まれます。

- ChangeUserPassword
- Deprovision
- Disable
- Enable
- Password
- RenameUser
- ResetPassword
- ResetUserPassword
- ResourceAccounts

このライブラリは主に、Identity Manager ユーザーに関連付けられているリソースアカウントの情報を表示したテーブルから構成され、各種操作で使用されるこれらの情報を選択できます。

サンプルフォーム

次の表は、Identity Manager に付属するサンプルフォームと、その格納場所を示しています。どのフォームも sample\$forms に格納されています。

表 3-22 サンプルフォーム

ユーザーフォーム名	ファイル名
Active Directory、Active Sync	ActiveDirectoryActiveSyncForm.xml
AD ユーザーフォーム	ADUserForm.xml
Domino ユーザーフォーム	DominoUserForm.xml
GroupWise ユーザーフォーム	GroupWiseUserForm.xml
HP-UX ユーザーフォーム	HP-UXUserForm.xml
LDAP Active Sync	SkeletonDatabaseActiveSyncForm.xml

表 3-22 サンプルフォーム (続き)

ユーザーフォーム名	ファイル名
Linux ユーザーフォーム	LinuxUserForm.xml
Netegrity Siteminder Admin	SiteMinderAdminUserForm.xml
Netegrity Siteminder LDAP	SiteMinderLDAPUserForm.xml
Netegrity Siteminder ExampleTable	SiteMinderExampleTableUserForm.xml
NDS ユーザーフォーム	NDSUserForm.xml
NT ユーザーフォーム	NTform.xml
Open Networks	ONTUserForm.xml
Oracle ERP	OracleERPUserForm.xml
OS400 ユーザーフォーム	OS400UserForm.xml
RACF ユーザーフォーム	RACFUserForm.xml
SAP	SAPUserForm.xml
RSA ClearTrust	ClearTrustUserForm.xml
SecurID ユーザーフォーム	SecurIDUserForm.xml
Solaris ユーザーフォーム	SolarisUserForm.xml
AIX ユーザーフォーム	AIXUserForm.xml
Exchange55 ユーザーフォーム	Exchange55UserForm.xml
Active Directory、Active Sync	vitalStatform.xml
AD ユーザーフォーム	AccessManagerUserForm.xml
Domino ユーザーフォーム	TopSecretUserForm.xml
GroupWise ユーザーフォーム	ACF2UserForm.xml

サンプルフォームライブラリの使用

<FormRef> 要素を使用することで、Identity Manager に付属するサンプルフォームは、カスタマイズしているどのフォームにも含めることができます。

環境にサンプルフォームを追加する一般的な手順は次のとおりです。

手順 1: 規則のインポート

手順 2: フォームのインポート

手順 3: デフォルトフォームからの新規フォームの作成 (Include 参照の追加と、フォーム参照の追加)

手順 1: 規則のインポート

Identity Manager の管理者インタフェースを使用して、サンプル規則を読み込みます。次の手順で実行します。

1. Identity Manager のメニューバーから、「設定」>「交換ファイルのインポート」を選択します。
2. サンプルファイル名を入力するか、「参照」をクリックして、`idm¥sample¥rules` ディレクトリからファイルを探します。

サンプル規則の共通ファイルの名前は次のとおりです。

- `sample¥rulesListGroups.xml`
- `sample¥rules¥NamingRules.xml`
- `sample¥rules¥RegionalConstants.xml`

サンプル規則のリソースファイルの名前は次のとおりです。

- `sample¥rules¥ADRules.xml`
- `sample¥rules¥NDSRules.xml`
- `sample¥rules¥NTRules.xml`
- `sample¥rules¥OS400UserFormRules.xml`
- `sample¥rules¥RACFUserFormRules.xml`
- `sample¥rules¥TopSecretUserFormRules.xml`

3. 「インポート」をクリックします。インポートに成功したことを示すメッセージが表示されます。

手順 2: フォームのインポート

Identity Manager の管理者インタフェースを使用して、サンプルフォームを読み込みます。次の手順で実行します。

1. Identity Manager のメニューバーから、「設定」>「交換ファイルのインポート」を選択します。
2. サンプルファイル名を入力するか、「参照」をクリックして、`idm¥sample¥forms` ディレクトリからファイルを探します。サンプルフォームのファイル名は次のとおりです。
 - `sample¥forms¥ACF2UserForm.xml`
 - `sample¥forms¥AIXUserForm.xml`
 - `sample¥forms¥Exchange55UserForm.xml`

- sample¥forms¥HP-UXUserForm.xml
 - sample¥forms¥NDSUserForm.xml
 - sample¥forms¥NTform.xml
 - sample¥forms¥OS400UserForm.xml
 - sample¥forms¥SecurIDUserForm.xml
 - sample¥forms¥SolarisUserForm.xml
 - sample¥forms¥TopSecretUserForm.xml
 - sample¥forms¥vitalStatform.xml
3. 「インポート」をクリックします。インポートに成功したことを示すメッセージが表示されます。

手順 3: Tabbed User Form の更新 (Include 参照の追加)

Tabbed User Form または作成するメインフォームから、サンプルフォームに Include 参照を追加します。次の手順で実行します。

1. Tabbed User Form をコピーし、名前を変更します (たとえば、`<CompanyName>tabbedUserForm`)。
2. Web ブラウザのアドレス行にこの URL を入力し、**Enter** キーを押します。
`http://ApplicationServerHost:Port/idm/debug`
3. 認証が完了すると、「システム設定」ページが表示されます。
4. 「タイプ」リストから **UserForm** オプションを選択し、「List Objects」をクリックします。
5. `<CompanyName>tabbedUserForm` (または、作成するメインフォーム) の横の「編集」をクリックします。
6. フォームの **Include** エリアを変更し、次の例に太字で表示されている各サンプルフォームを追加します。

```
<Include>
  <ObjectRef type='UserForm' id='#ID#UserForm:UserformLibrary'
name='UserForm Library' />
  <ObjectRef type='UserForm' name='UserFormName' />
</Include>
```

`UserFormName` に設定できる値は次のとおりです。

- ACF2 User Form
- AIX User Form
- Exchange55 User Form
- HP-UX User Form

- LDAP Active Sync User Form
- Netegrity Siteminder Admin Form
- Netegrity Siteminder LDAP User Form
- Netegrity Siteminder ExampleTable User Form
- NDS User Form
- NT User Form
- Open Networks User Form
- OS400 User Form
- Oracle ERP User Form
- RACF User Form
- RSA ClearTrust User Form
- SecurID User Form
- Skeleton Database Active Sync User Form
- Solaris User Form
- Tivoli Access Manager
- Top Secret User Form
- グローバル属性 (vitalStatform.xml)

フォームを保存する前に、継続して次の手順を実行します。

手順 4: Tabbed User Form の更新 (フォームの追加)

各サンプルフォームの FormRef を追加するには、それをメインフォームに追加します。

1. メインフォーム内の適切な場所に、サンプルフォームごとに次の行を追加します。
`<FormRef name='UserFormName' />`
2. 次の行を削除します。
`<FormRef name='MissingFields' />`
3. 「保存」をクリックして、フォームの変更を保存します。

FormUtil メソッド

この章では、もっともよく使用される **FormUtil** メソッドについて説明します。

FormUtil クラスは、フォームオブジェクト内の **XPRESS** 式から呼び出せるユーティリティーメソッドの集合です。**FormUtil** メソッドは、通常、**Select** フィールドと **MultiSelect** フィールドの **valueMap** 内で、使用できる値のリストを制限するために使用されます。また、日付やディレクトリ **DN** などの文字列値をフォーマットするための追加メソッドも用意されています。

関連する章

- 「[Identity Manager のビュー](#)」 - ユーザービューと呼ばれる **Identity Manager** の内部データ構造と連携する **Identity Manager** フォームについて説明しています。フォームをカスタマイズするときは、ビューの属性を呼び出します。
- 「[Identity Manager フォーム](#)」 - フォーム内での **FormUtil** メソッドの使用例については、「[非表示コンポーネントの使用](#)」と「[リストデータの取り込み](#)」の節を参照してください。

メソッドの呼び出し

フォーム内から `FormUtil` メソッドを呼び出すときは、次の構文を使用します。

```
<invoke class = 'com.waveset.ui.FormUtil'
  name = 'method_name'>
  <ref>:display.session</ref>
  <s>arg2</s>
</invoke>
```

`name` フィールドには、メソッドの名前を指定します。

ほとんどの `FormUtil` メソッドでは、ビュー属性 `display.session` を参照し、最初の引数として `LighthouseContext` または `Session` オブジェクトを渡す必要があります。フォームは、ベースコンテキストプレフィックスとともに使用されることが多いので、ベースコンテキストプレフィックスを削除できるように、`display.session` を参照する際には先頭に常にコロンを付けて参照することをお勧めします。

メソッド

ここでは、もっともよく使用される `FormUtil` メソッドを紹介します。

callResourceMethod メソッド

```
callResourceMethod(LighthouseContext s, String resourceName, String
methodName, Map args) throws WavesetException {
```

説明

引数を指定して、リソース上の指定メソッドを呼び出します。

表 4-1 `callResourceMethod` のパラメータ

パラメータ	説明
<code>s</code>	有効な <code>Identity Manager</code> コンテキストを指定します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます
<code>resourceName</code>	呼び出すメソッドのリソース名を指定します
<code>methodName</code>	呼び出すリソースメソッドの名前を指定します

表 4-1 callResourceMethod のパラメータ (続き)

パラメータ	説明
<i>args</i>	リソースメソッドが受け付けるマップを <i>args</i> として指定します

buildDn メソッド

```
buildDn(String name,
String baseContext)
```

説明

名前を作成するために名前とベースコンテキストを受け取ります。このメソッドは、完全修飾識別 (DN) 名を返します。たとえば、group1 と dc=example,dc=com を渡すと、cn=group1, dc=example, dc=com という文字列が返されます。

表 4-2 buildDn のパラメータ

パラメータ	説明
<i>name</i>	ディレクトリオブジェクトの名前を指定します (一般的には、共通名 cn の値を指定します)。
<i>baseContext</i>	ベースコンテキストまたはコンテナを指定します。これは、ディレクトリオブジェクトの完全識別名を形成するために共通名と連結される、ベース識別名の値です。

戻り値

CN=<name>,<baseContext> という形式で、1 つの値を返します。

例

```
<invoke name='buildDn' class='com.waveset.ui.FormUtil'>
  <s>North America</s>
  <s>ou=marketing,dc=acme,dc=com</s>
</invoke>
```

この例の結果として返される内容は次のとおりです。<s>CN=North America,ou=marketing,dc=acme,dc=com</s>

XPRESS では、同じ機能を次のように表現できます。

```
<concat>
  <s>CN=</s>
  <s>North America</s>
  <s>,</s>
  <s>ou=marketing,dc=acme,dc=com</s>
</concat>
```

buildDns メソッド

```
buildDns(List names,
String baseContext)

または

buildDns(String names,
String delimiter,
String baseContext)
```

説明

リスト内の各トークンに追加する、名前のリストとベースコンテキストを受け取ります。どちらのメソッドでも、完全修飾 DN 名を返します。

たとえば、group1, group2 と dc=example, dc=com を渡した場合は、cn=group1, dc=example, dc=com と cn=group2, dc=example, dc=com とのリストが返されます。

パラメータ

1 つめのメソッドは、次の表で説明する引数を受け取ります。

表 4-3 buildDns のパラメータ (1 つめのメソッド)

パラメータ	説明
<i>names</i>	オブジェクト名のリストを指定します
<i>baseContext</i>	ベースコンテキストを指定します

2 つめのメソッドは、次の表で説明する 3 つの引数を受け取ります。

表 4-4 buildDns のパラメータ (2 つめのメソッド)

パラメータ	説明
<i>names</i>	コンマやセミコロンなどの区切り文字によって区切られた、複数の名前を含む文字列を指定します。
<i>delimiter</i>	複数の名前を含む文字列内で使用される区切り文字を指定します。区切り文字は、通常、コンマ (,) またはピリオド (.) です。
<i>baseContext</i>	ベースコンテキストを指定します。

戻り値

それぞれの値が CN=<name>,<baseContext> の形式の、値または文字列のリストを返します。

例

例 1: buildDns が返すリスト

```
<invoke name='buildDns' class='com.waveset.ui.FormUtil'>
  <list>
    <s>North America</s>
    <s>Europe</s>
  </list>
  <s>ou=marketing,dc=acme,dc=com</s>
</invoke>
```

この例の結果として返される内容は次のとおりです。

```
<list>
  <s>CN=North America,ou=marketing,dc=acme,dc=com</s>
  <s>CN=Europe,ou=marketing,dc=acme,dc=com</s>
</list>
```

例 2: *buildDns (String names、String delimiter、String baseContext) の返すリスト*

```
<invoke name='buildDns' class='com.waveset.ui.FormUtil'>
  <s>North America,Europe,China</s>
  <s>,</s>
  <s>ou=marketing,dc=acme,dc=com</s>
</invoke>
```

この例の結果として返される内容は次のとおりです。

```
<list>
  <s>CN=North America,ou=marketing,dc=acme,dc=com</s>
  <s>CN=Europe,ou=marketing,dc=acme,dc=com</s>
  <s>CN=China,ou=marketing,dc=acme,dc=com</s>
</list>
```

この機能を、XPRESS を記述することで実現しようとはしないでください。

checkStringQualityPolicy メソッド

checkStringQualityPolicy(LighthouseContext s, String policy, Object value, Map map, List pwdhistory, String owner)

説明

文字列ポリシーに対して、指定された文字列の値を検証します。

表 4-5 checkStringQualityPolicy のパラメータ

パラメータ	説明
<i>S</i>	ユーザーの現在の Lighthouse コンテキストを指定します。
<i>policy</i>	(必須) 文字列の検証に適用されるポリシーの名前を指定します。
<i>value</i>	(必須) 検証する文字列の値を指定します。
<i>map</i>	(省略可能) 文字列に含めることができないデータのマップを指定します。 returnNull -- (省略可能) true に設定すると、このメソッドは成功時に NULL オブジェクトを返します。
<i>pwdhistory</i>	(省略可能) ユーザーの過去のパスワードを、大文字の、暗号化された形式で一覧表示します。
<i>owner</i>	(必須) 調査対象となる文字列値のユーザーを指定します。

戻り値

文字列がポリシーのテストに合格した場合には、このメソッドは **true** を返します。文字列がポリシーテストに合格しなかった場合、このメソッドはエラーメッセージを返します。map パラメータの returnNull オプションを **true** に設定した場合は、成功時に NULL オブジェクトが返されます。

controlsAtLeastOneOrganization メソッド

controlsAtLeastOneOrganization(LighthouseContext s, List organizations)

throws WaveSetException {

説明

現在認証されているユーザーが、1 つまたは複数の組織 (ObjectGroup) 名のリストに含まれるいずれかの組織を制御できるかどうかを返します。サポートされる組織リストには、ObjectGroup タイプのすべてのオブジェクトのリスト表示で返される組織が含まれます。

表 4-6 controlsAtLeastOneOrganization のパラメータ

パラメータ	説明
s	ユーザーの現在の Lighthouse コンテキスト (セッション) を指定します。
organizations	1 つまたは複数の組織名のリストを指定します。サポートされる組織リストには、ObjectGroup タイプのすべてのオブジェクトのリスト表示で返される組織が含まれます。

戻り値

true - 現在認証されている Identity Manager ユーザーが、リストに含まれるいずれかの組織を制御できることを表します。

false - 現在認証されている Identity Manager ユーザーが、リストに含まれるどの組織も制御できないことを表します。

getObject メソッド

```
getObject(LighthouseContext s,  
String typeName,  
String id)  
throws WavesetException
```

説明

リポジトリからオブジェクトを取得します (認証されていることが前提)。

表 4-7 getObject のパラメータ

パラメータ	説明
<i>s</i>	有効な Identity Manager セッションを指定します。通常は、 <ref>:display.session</ref> という形式で参照されます。
<i>typeName</i>	オブジェクトタイプを指定します。一般的なオブジェクトタイプ は、User、Object Group、Resource です。すべてのタイプのリ ストを知るためには、「デバッグ」ページの「タイプ」オブショ ンリストを参照してください。
<i>id</i>	検索するオブジェクトの ID または名前を指定します。User オブ ジェクトでは、ここに accountId を指定します。

getObjectNames メソッド

```
getObjectNames(LighthouseContext s,  
String typeName)  
throws WavesetException  
  
または  
getObjectNames(LighthouseContext s,  
String typeName,  
Map options)  
throws WavesetException
```

説明

指定されたタイプのオブジェクトでセッションの所有者 (または現在のログインユーザー) がアクセスできるオブジェクトの名前のリストを返します。オプションマップに追加パラメータを指定し、返される名前のリストを制御できます。

オブジェクト名のリストを返すときは、`session.getObjects()` を実行するよりも、このメソッドを使用することをお勧めします。メソッドは、まず、`ObjectCache` を検索し、必要に応じて次にリポジトリを検索します。

表 4-8 getObjectNames のパラメータ

パラメータ	説明
<code>s</code>	有効な Identity Manager セッションを指定します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<code>typeName</code>	オブジェクトタイプを指定します。一般的なオブジェクトタイプは、 User 、 Object Group 、 Resource です。すべてのタイプのリストを知るためには、「デバッグ」ページの「タイプ」オプションリストを参照してください。
<code>options</code>	次の表を参照してください

表 4-9 オプションの値

オプション	値
<code>conditions</code>	「追加オプション」を参照してください。
<code>current</code>	「追加オプション」を参照してください。
<code>scopingOrg</code>	「追加オプション」を参照してください。

戻り値

このメソッドは、指定されたタイプのオブジェクトでセッションの所有者がアクセスできるオブジェクトの名前のリストを返します。

getOrganizationsDisplayNames メソッド

getOrganizationsDisplayNames(LighthouseContext s)

throws WaveSetException

または

getOrganizationsDisplayNames(LighthouseContext s, Map options)

throws WaveSetException

説明

現在の管理者がアクセスできる、組織ハンドルのリストを返します。組織の選択リストおよび複数選択リストを使用するフォームでは、このメソッドを使用する必要があります。

注

このメソッドは、getOrganizationsDisplayNames() の呼び出しに pathPrefix オプションが指定されていない場合にのみ、useOrganizationDisplayNames にシステム設定のデフォルト値を適用します。

パラメータ

オプションは、キーと値のペアの引数のマップから構成されます。

表 4-10 getOrganizationsDisplayNames のパラメータ

パラメータ	説明
s	有効な Identity Manager セッションを識別します。通常は、<ref>:display.session</ref> という形式で参照されます。
options	pathprefixes、excluded、current、filterVirtual、conditions、scopingOrg。次の表を参照してください。

表 4-11 オプションの値

オプション	値
pathPrefixes	true - 組織のパスを含む、返されるハンドルのリスト。 false - 表示名を含む、返されるハンドルのリスト true 以外 (未指定) - ハンドルタイプに、システムの useOrganizationsDisplaynames の設定のデフォルト値を適用します。 Identity Manager のインストールシステムが重複する組織名を持つ場合は、パスを使用します。
excluded	返される valueMap に含めない組織を指定します。
current	この引数に組織名のリストを指定すると、これらの追加組織を含む list が返されます。パスを要求しているときは、これはパスのリストとなります。表示名を要求しているときは、これは表示名のリストとなります。
conditions	「追加オプション」を参照してください。
scopingOrg	「追加オプション」を参照してください。
filterVirtual	オプションマップでこの引数を true に設定した場合は、返される valueMap に仮想組織が含まれません。

バージョンに固有の動作

このメソッドの動作は、インストールされている Identity Manager のバージョンが v3.1x 以降であるかどうかによって異なります。バージョン 3.1.x では、このメソッドは同一名の複数の組織をサポートしません。この製品のバージョン 4.x 以降では、同一名の複数の組織がサポートされます。

このメソッドの v3.1.x バージョンを使用する理由

カスタマイズしたインストールシステムが次のような特徴を持つときは、複数の組織をサポートしないように、このメソッドをリセットしてください。

- 重複する組織名を使用しない、また、使用する予定がなく、組織を短い表示名で表示するように設定されている
- 以前から組織の短い名前を返していたメソッドを使用するようにカスタマイズされている
- 使用できる組織のリスト表示、または式の評価に getOrganizations メソッドを使用している

- ビュー属性に追加するリソース属性を設定する式に、以前から組織の短い表示名を返していた `waveset.organization` ビュー属性への参照を使用している

v3.1.x の動作に合わせるためのメソッドの再設定

v3.1.x より後のバージョンの Identity Manager を使用していて、このメソッドを前述のように動作させる必要がある場合は、次のシステム設定オブジェクト属性を、次のように変更します。

```
<Attribute name='useOrganizationDisplay Names'>
  <Boolean>false</Boolean>
</Attribute>
```

getResources メソッド

```
getResources(LighthouseContext s)
```

```
  throws WavesetException
```

または

```
getResources(LighthouseContext s,
```

```
List current)
```

```
  throws WavesetException
```

または

```
getResources(LighthouseContext s,
```

```
String matchType,
```

```
String value)
```

```
  throws WavesetException
```

または

```
getResources(LighthouseContext s,
```

```
String matchType,
```

```
String value
```

```
List current)
```

または

```
getResources(LighthouseContext s,
```

```
Map Options)
```

説明

特定のリソース属性値 (type=LDAP など) と一致するリソースの名前リストを作成します。現在のリストを渡すと、リストはマージされます。

- このメソッドの最初の変数にとる値は session のみで、管理者が表示できるすべてのリソースが返されます。
- このメソッドの 2 番目の変数は、すべてのリソースを返し、それを現在のリストにマージします。
- このメソッドの 3 番目の変数は、特定の属性値と一致するすべてのリソースを返します。
- このメソッドの 4 番目の変数は、特定の属性値と一致するすべてのリソースを返し、それを現在のリストにマージします。

表 4-12 getResources のパラメータ

パラメータ	説明
s	有効な Identity Manager セッションを識別します。通常は、<ref>:display.session</ref> という形式で参照されます。
matchtype	リソース名のリストに含めるリソースを特定するときに、このメソッドが検索するリソース属性を指定します。リソースオブジェクトタイプの、クエリー可能な属性を指定してください。クエリー可能な属性には、type、supportsScanning、supportsContainerObjectTypes、および startupType があります。これらの属性については後述します。
value	リソースを特定するときに、このメソッドが検索する属性の値を指定します。
options	current、conditions、scopingOrg。次の表を参照してください。

サポートされるリソースタイプ名については、「ビュー」を参照してください。

表 4-13 オプションの値

オプション	値
conditions	「追加オプション」を参照してください。
current	「追加オプション」を参照してください。

表 4-13 オプションの値

オプション	値
scopingOrg	「追加オプション」を参照してください。

matchType パラメータに指定する、クエリー可能な属性の有効なタイプは次のとおりです。

表 4-14 *matchType* パラメータに指定できる値

<i>matchType</i> パラメータ	説明
type	リソースオブジェクトのタイプを指定します。有効な値については、リソースの XML プロトタイプを参照してください。たとえば、Active Directory を表すオブジェクトタイプは、Windows Active Directory です。Groupwise では、GroupWise となります。
supportsScanning	true に設定すると、リソースがスキャンをサポートすることを表します。
supportsContainerObjectTypes	true に設定すると、リソースがコンテナオブジェクトをサポートすることを表します。
startupType	ActiveSync の起動タイプを指定します。有効な値には、Automatic、Automatic with failover、Manual、および Disabled があります。

getResourceObjects メソッド

```
getResourceObjects(LighthouseContext session, String objectType,
String resourceId, Map options, String cacheList, String
cacheTimeout, String clearCacheIfExists)
```

または

```
getResourceObjects(LighthouseContext session, String objectType,
String resourceId, Map options)
```

または

```
getResourceObjects(String subjectString, String objectType,
String resourceId, Map options)
```

または

```
getResourceObjects(String subjectString, String objectType,
String resourceId, Map options, String cacheList, String
cacheTimeout, String clearCacheIfExists)
```

説明

各オブジェクトが、タイプ、名前、ID (DN または完全修飾名)、および要求される searchAttrsToGet の値の属性セットを持つオブジェクトのリストを返します。返される値は、GenericObjects のリストです。それぞれの GenericObject には、マップへのアクセスと同様の方法でアクセスできます。各オブジェクトで get メソッドを呼び出します。このとき、属性名が渡され、属性値が返されます。

パラメータ

表 4-15 getResourceObjects のパラメータ

パラメータ	説明
session	有効な Identity Manager セッションを識別します。通常は、<ref>:display.session</ref> という形式で参照されます。
subjectString	ユーザーオブジェクトの、シリアル化された XML 表現を指定します。この表現は、現行セッションを持たないワークフローで便利です。たとえば、主体を取得する場合は、次のように指定します。 <pre><invoke name='getSubject'> <ref>WF_CONTEXT</ref> </invoke></pre>

表 4-15 getResourcesObjects のパラメータ (続き)

パラメータ	説明
<i>objectType</i>	指定されているこの resId の、有効なオブジェクトクラスの名前を指定します。NULL に設定した場合は、指定検索コンテナおよび検索範囲内の、指定した resId によって定義されるすべてのオブジェクトタイプのオブジェクトが返されます。
<i>resourceId</i>	オブジェクトの取得元となるリソースを指定します。
<i>cacheList</i>	true に設定すると、結果がサーバーにキャッシュされます (これは、メインフレーム上のプロファイルのリストのように、取得に時間がかかるクエリーで不可欠)。
<i>cacheTimeout</i>	true に設定するときは、キャッシュがタイムアウトになるまでの時間をミリ秒単位で指定します。キャッシュがタイムアウトになると、検索結果に含まれるオブジェクトが、次の要求時に自動的にリソースから取得されます (たとえば、6000 = 1 分間)。
<i>clearCacheIfExists</i>	キャッシュをクリアし、次の要求時に、検索対象オブジェクトをリソースから再フェッチするように強制します。
<i>options</i>	検索の制限に使用される、リソースに固有の名前と値のペアを指定します。次の表を参照してください。

表 4-16 Option パラメータに指定できる値

オプション	値
<i>searchContext</i>	検索対象のコンテキスト (ResourceAdapter.RA_SEARCH_CONTEXT) を指定します。指定しない場合、メソッドは RA_BASE_CONTEXT から値を取得しようとします。値が指定されていない場合は、このメソッドは論理上の上位から検索を行います。
<i>searchFilter</i>	(省略可能) RFC 1558 に規定されている LDAP 検索フィルタフォーマットでは、1 つまたは複数のオブジェクトが and または or で連結されます。規定されていない場合、このメソッドは、指定されている <i>objectType</i> (ResourceAdapter.SEARCH_FILTER) を使用してフィルタを作成します。

表 4-16 Option パラメータに指定できる値 (続き)

オプション	値
searchScope	メソッドの検索対象を、指定した searchContext のコンテキストに含まれる現在のオブジェクト、または指定した searchContext (ResourceAdapter.RA_SEARCH_SCOPE) に含まれるすべてのサブコンテキストに設定します。有効な値には、object、oneLevel、または subTree があります。オブジェクトを指定した場合は、searchContext は単一オブジェクトの完全 DN と見なされるため、1 つのオブジェクトのみが返されます。oneLevel は、特定の baseContext (searchContext) 以下の、1 つまたは複数のフェッチを意味します。subTree は、oneLevel 検索の実行を繰り返します。searchContext は、指定した searchContext に含まれるすべてのサブコンテキストを対象に検索を実行することを表します。
searchTimeLimit	検索の制限時間を、ミリ秒単位で指定します (ResourceAdapter.RA_SEARCH_TIME_LIMIT)。検索は、この制限時間内に行われます。
searchAttrsToGet	オブジェクトごとに取得する、objectType に固有の属性名のリストを指定します。
runAsUser	この要求を実行するユーザーの名前を指定します。指定しない場合は、デフォルト値として、リソースプロキシ管理ユーザーが適用されます。
runAsPassword	runAsUser のパスワードを指定します。指定したユーザーとしてリスト要求を実行するために、その権限があることをリソースに対して認証するのに必要とされます。
current	「追加オプション」を参照してください。
conditions	「追加オプション」を参照してください。
scopingOrg	「追加オプション」を参照してください。

このメソッドは、Session の代わりに subjectString を指定して実行することもできます。

getRoles メソッド

`getRoles(LighthouseContext s)`

または

`getRoles(LighthouseContext s, String current)`

または

`getRoles(LighthouseContext s, List current)`

または

`getRoles(LighthouseContext s, Map options)`

説明

現在の管理者がアクセスできる、ロール名のリストを返します。現在の値、または現在のリストを指定すると、返されるロール名にリスト内のロール名が追加されます。

表 4-17 getRoles のパラメータ

パラメータ	説明
<i>s</i>	有効な Identity Manager セッションを識別します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<i>options</i>	<i>current</i> 、 <i>conditions</i> 、および <i>scopingOrg</i> 。次の表を参照してください。

表 4-18 オプションの値

オプション	値
<i>current</i>	「追加オプション」 を参照してください。
<i>conditions</i>	「追加オプション」 を参照してください。
<i>scopingOrg</i>	「追加オプション」 を参照してください。

戻り値

現在の管理者がアクセスできる、ロール名のリストを返します。現在の値、または現在のリストを指定すると、返されるロール名にリスト内のロール名が追加されます。

getUnassignedApplications メソッド

```
getUnassignedApplications(LighthouseContext s, Map options)
throws WavesetException {
```

説明

ユーザーのプライベートアプリケーションに適したアプリケーション名のリストを作成します (プライベートアプリケーションは、ユーザーに直接割り当てられているアプリケーション)。これは、ルールを通じてすでにユーザーに割り当てられているアプリケーションの名前を除く、すべての利用可能なアプリケーションのリストです。作成されるリストは、フォームでのプライベートアプリケーションの割り当てに便利です。

パラメータ

getUnassignedApplications は、次のオプションをとります。

表 4-19 getUnassignedApplications のパラメータ

パラメータ	説明
<i>context</i>	Identity Manager コンテキストオブジェクト
<i>options</i>	current、currentRoles、conditions。次の表を参照してください。

表 4-20 オプションの値

オプション	値
current	現在のアプリケーション名のリスト。返されるリストには、その他のオプションに関係なく、これらのアプリケーションのそれぞれが常に含まれます。
currentRoles	現在のルール名のリスト。返されるリストには、現在のリストに名前が指定されているアプリケーションを除き、指定されるいずれかのルールによって割り当てられるアプリケーションは含まれません。
conditions	次のように、マップまたはリストのいずれかを指定できます。

表 4-21 conditions パラメータに指定できる値

パラメータ	説明
<map>	最初のマップエントリ - このオブジェクトタイプがサポートする、クエリー可能な属性の名前 2 番目のマップエントリ - 関連付けられているクエリー可能な属性を取得するために必要な、このタイプのオブジェクトの値 (演算子は equals と見なされる) 属性名と値の複数のペアを指定した場合は、それらは論理 and で連結されます。
<list>	AttributeCondition オブジェクトのリスト。 次に例を示します。 <pre><list> <newclass='com.waveset.object.AttributeCondition'> <s>MemberObjectGroups</s> <s>equals</s> <ref>waveset.organization</ref> </new> </list></pre> 複数の AttributeCondition を指定した場合は、それらは論理 and で連結されます。

getSubordinates メソッド

```
getSubordinates(LighthouseContext s, manager, Map options)
throws WavesetException {
```

説明

ユーザーの、指定された管理対象部下のリストを取得します。

表 4-22 getSubordinates のパラメータ

パラメータ	説明
s	Identity Manager コンテキストオブジェクト
manager	要求する部下のマネージャーを指定します。
options	firstlevel、lastLevel、conditions。次の表を参照してください。

表 4-23 オプションの値

オプション	値
firstLevel	結果に含める最上位組織レベルを指定します。 0 は、直接レポートを表します。
lastLevel	結果に含める最下位組織レベルを指定します。 -1 は、すべてのレベルを表します。
conditions	(省略可能) 返されるユーザー名のリストをさらにフィルタリングするための、ユーザー属性条件のマップ。「追加オプション」を参照してください。

戻り値

このメソッドは、部下のソート済みリストを返すか、または WavesetException をスローします。

getSubordinates を使用した規則の例

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE Rule PUBLIC 'waveset.dtd' 'waveset.dtd'>
<Rule name='testGetSubordinates'>
  <invoke name='getSubordinates' class='com.waveset.ui.FormUtil'>
    <ref>:display.session</ref>
    <ref>waveset.accountId</ref>
    <map>
      <s>firstLevel</s>
      <i>0</i>
      <s>lastLevel</s>
      <i>0</i>
      <s>conditions</s>
      <map>
      </map>
    </map>
  </invoke>
</Rule>
```

getUnassignedResources メソッド

```
static public List getUnassignedResources(LighthouseContext s, Map
options) throws WavesetException
```

説明

ユーザーのプライベートリソースに適したリソース名のリストを作成します (プライベートリソースは、ユーザーに直接割り当てられているリソース)。これは、ロールを通じてすでにユーザーに割り当てられているリソースの名前を除く、すべての利用可能なリソースのリストです。

作成されるリストは、フォームでのプライベートリソースの割り当てに便利です。

表 4-24 getUnassignedResources のパラメータ

パラメータ	説明
<i>context</i>	Identity Manager コンテキストオブジェクト
<i>options</i>	availableToOrgScope、current、currentRoles、currentResourceGroups、conditions、scopingOrg。 次の表を参照してください。

表 4-25 オプションの値

オプション	値
availableToOrgScope	(リスト) 組織名 (または、一意の場合は <i>paths</i> または <i>displayNames</i>) を指定します。このリストに含まれる組織が利用できる、または組織階層でその組織より下位レベルにある組織が利用できるリソースが返されます。現在のリストに含まれるリソースも常に返されます。
current	「追加オプション」を参照してください。
currentRoles	(リスト) 現在のロール名を指定します。返されるリストには、現在のリストに名前が指定されているリソースを除き、指定されるいずれかのロールによって割り当てられるリソースは含まれません。

表 4-25 オプションの値 (続き)

オプション	値
<code>currentResourceGroups</code>	(リスト) 現在のリソースグループの名前を指定します。返されるリストには、現在のリストに名前が指定されているリソースを除き、指定されるいずれかのリソースグループによって割り当てられるリソースは含まれません。
<code>conditions</code>	「追加オプション」を参照してください。
<code>scopingOrg</code>	「追加オプション」を参照してください。

戻り値

このメソッドは、ユーザーのプライベートリソースに適したリソース名のリストを返します。

getUsers メソッド

`getUsers(LighthouseContext s)`

または

`getusers(LighthouseContext s, Map options)`

説明

このメソッドの最初の変数は、すべてのユーザーを返します。2 番目の変数は、デフォルトではすべてのユーザーを返しますが、リストをさらにフィルタリングするためのオプションのマップを指定できます。

表 4-26 getUsers のパラメータ

パラメータ	説明
<code>s</code>	有効な Identity Manager コンテキストを識別します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<code>options</code>	<code>current</code> 、 <code>conditions</code> 、 <code>exclude</code> 、 <code>scopingOrg</code> 。次の表を参照してください。

表 4-27 オプションの値

オプション	値
current	「追加オプション」を参照してください。
conditions	「追加オプション」を参照してください。サポートされるユーザーのクエリー可能属性名のほかに、擬似属性 <code>userType</code> 、 <code>similarTo</code> も使用できます。詳細については、 <code>FomUtil</code> の <code>Javadoc</code> を参照してください。
exclude	返されるリストから除外するユーザーのタイプを指定します。値は、<code>administrators</code> と <code>endUsers</code> のいずれかです。 <code>administrators</code> -- 返されるユーザーのリストには、直接的または間接的に 1 つまたは複数の機能が割り当てられ (<code>AdminRoles</code> の設定)、1 つまたは複数の組織を制御するユーザーは含まれません。 <code>endUsers</code> -- 返されるリストには、1 つまたは複数の機能と、制御対象組織が直接的または間接的に割り当てられているユーザーのみが含まれます。
scopingOrg	「追加オプション」を参照してください。

listResourceObjects メソッド

```
listResourceObjects(LighthouseContext s,  
String objectType,  
List resourceList,  
Map options,  
String cacheList)  
または  
listResourceObjects(LighthouseContext s,  
String objectType,  
List resourceList,  
Map options,  
String cacheList,  
String clearCacheIfExists}
```

2 番目のメソッドがキャッシュをクリアする点を除き、最初の 2 つの変数は同じです。

```
listResourceObjects(String subjectString,  
String resourceObjectType,  
List resourceList,  
Map options,  
String cacheList)
```

または

```
listResourceObjects(String subjectString,  
String objectType,  
String resourceId,  
Map options,  
String cacheList)  
String clearCacheIfExists)
```

または

```
listResourceObjects(String subjectString,  
String objectType,  
String resourceID,  
Map options,  
String cacheList)
```

または

```
listResourceObjects(String subjectString,  
String objectType,  
String resourceID,  
Map options,  
String cacheList)  
String cacheTimeout  
String clearCacheIfExists)
```

2 番目のメソッドがキャッシュをクリアする点を除き、最初の 2 つの変数は同じです。

```
listResourceObjects(LighthouseContext session,  
String objectType,
```

```

String resourceId,
Map options,
String cacheList,
String clearCacheIfExists)
    throws WavesetException {
または
listResourceObjects(LighthouseContext session,
String objectType,
String resourceId,
Map options,
String cacheList)
    throws WavesetException {
または
listResourceObjects(LighthouseContext session,
String objectType,
String resourceId,
Map options,
String cacheList)
String cacheTimeout
String clearCacheIfExists
    throws WavesetException
または
listResourceObjects (String subjectString, String objectType,
List resourceList, Map options, String cacheList,
String clearCacheIfExists)

```

説明

指定したタイプ (たとえば、group) のリソースオブジェクトのリストを取得します。このメソッドの最初の試行では、サーバーの resourceObjectListCache からリストが取得されます。見つかった場合は、リストが返されます。

リストが見つからなかった場合は、メソッドは各リソースで `listResourceObjects` メソッドを呼び出し、結果リストのマージ、ソート、および重複削除を行います。最後に、同じリソースからの同じリソースオブジェクトタイプを求める次回以後の要求に備えて、この新しいリストをサーバーの `resourceObjectListCache` にキャッシュします。

このメソッドは、現在認証されている管理者 (たとえば、`subject`) として実行されます。変数は、単一のリソース ID、または `subject` 文字列と既存のセッションを値としてとります。

このメソッドは、次の条件に応じて、異なる複数の変数をとります。

- メソッドが、単一のリソースを返すか、リソースリストを返すか。
- キャッシュをクリアする必要があるかどうか。
- メソッドが、セッション ID (ユーザーがすでに認証されている場合に実装される) を送信するか、主体を表す文字列 (`subjectString`) を送信するか。通常は、`Session` を使用します。

表 4-28 `listResourceObjects` のパラメータ

パラメータ	説明
<code>s</code>	有効な Identity Manager セッションを識別します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<code>resourceObjectType</code>	この指定のリソースオブジェクトクラスのタイプ (たとえば、 <code>group</code>) を指定します。NULL の場合は、すべてのオブジェクトタイプのオブジェクトが返されます。
<code>resourceList</code>	オブジェクトの取得元となるリソースのリスト。
<code>options</code>	検索の制限に使用される、リソースに固有の名前と値のペアを指定します。
<code>cacheList</code>	このメソッドが各リソースで <code>listResourceObjects</code> メソッドを呼び出すときに作成される、サーバーの <code>resourceObjectListCache</code> に格納される新しいリストを指定します。キャッシュへの格納後、結果リストのマージ、ソート、および重複削除が行われます。
<code>clearCacheIfExists</code>	メソッドがリソースオブジェクトのリストを取得したあとで、サーバーの <code>resourceObjectListCache</code> 内のキャッシュをクリアするかどうかを指定します。

戻り値

このメソッドは、リソースのリスト (ID または名前の `resourceList`) から、指定されたリソースオブジェクトタイプのリソースオブジェクト名のリストを返します。

testObject メソッド

```
testObject (LighthouseContext s,  
String typeName,  
String id)
```

説明

主体がオブジェクトの表示を認証されていない場合でも、指定したオブジェクトが存在するかどうかを確認します。ツリー全体を表示できない管理者がユーザーの新規作成プロセスを開始するときは、このメソッドを使用して、重複オブジェクトの作成を防止してください。

表 4-29 testObject のパラメータ

パラメータ	説明
<code>s</code>	有効な Identity Manager セッションを識別します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<code>typeName</code>	メソッドによって検証されるオブジェクトのタイプを指定します。
<code>id</code>	名前または ID を入力して、オブジェクトを指定します。ID を指定した場合は、 <code>typeName</code> の値は無視されます。このパラメータを NULL にすることはできません。

戻り値

このメソッドが返す値は次のとおりです。

`true` - オブジェクトが存在する

`null` - オブジェクトは存在しない

testUser メソッド

```
testUser(LighthouseContext s,  
String id)
```

説明

主体がオブジェクトの表示を認証されていない場合でも、指定したユーザーが存在するかどうかを確認します。ツリー全体を表示できない管理者がユーザーの新規作成プロセスを開始するときは、このメソッドを使用して、重複オブジェクトの作成を防止してください。

表 4-30 testUser のパラメータ

パラメータ	説明
<i>s</i>	有効な Identity Manager セッションを識別します。通常は、 <code><ref>:display.session</ref></code> という形式で参照されます。
<i>id</i>	名前または ID を入力して、ユーザーを指定します。このパラメータを NULL にすることはできません。

戻り値

このメソッドが返す値は次のとおりです。

true - ユーザーが存在する

null - ユーザーは存在しない

hasCapability メソッド

```
hasCapability(LighthouseContext s, String capability)  
  
throws WavesetException {
```

説明

ユーザーが指定された機能 (文字列) を持つかどうかを確認します。このメソッドは、直接割り当てられている機能、または AdminGroups、AdminRoles、またはその両方によって間接的に割り当てられている機能を調べます。session の値が必要です。

表 4-31 hasCapability のパラメータ

パラメータ	説明
s	有効な Identity Manager コンテキストを識別します。通常は、 <ref>:display.session</ref> という形式で参照されます。
capabilities	クエリー対象となる機能を指定します。

戻り値

true - 現在認証されている Identity Manager ユーザーが、指定された機能を持つことを意味します。

false - 現在のユーザーが、指定された機能を持たないことを意味します。

hasCapabilities メソッド

hasCapabilities(LighthouseContext s, List capabilities)

throws WaveSetException {

説明

ユーザーが、指定された機能 (文字列) のリストを持つかどうかを確認します。これらの機能は、直接割り当てるか、AdminGroups、AdminRoles、またはその両方を使用して間接的に割り当てることができます。session の値が必要です。

表 4-32 hasCapabilities のパラメータ

パラメータ	説明
s	有効な Identity Manager コンテキストを識別します。通常は、 <ref>:display.session</ref> という形式で参照されます。
capabilities	クエリー対象となる機能のリストを指定します。

戻り値

true - 現在認証されている Identity Manager ユーザーが、指定されたすべての機能を持つことを意味します。

false - ユーザーが、指定されているすべての機能を持たないことを意味します。

追加オプション

FormUtil メソッドのサブセットでは、次のオプションが使用されます。

- *scopingOrg*
- *conditions*
- *current*

scopingOrg

ユーザーに複数の **AdminRoles** が割り当てられている場合に使用されます。値には、組織名を指定します。この値は、**scopingOrg** 組織を制御する、また、ログイン中のユーザーに割り当てられている **AdminRole** によって制御される組織が使用できる名前のみを返すように指定します。

このオプションは、通常、ユーザーが別のユーザーを作成または編集するときに割り当てる名前 (たとえば、**Resourcenames**) が、必ず編集対象ユーザーが属す組織に適した名前となるようにするために使用されます。

scopingOrg パラメータの使用

次のような状況では、この属性を設定します。

- 指定されるユーザーに、複数の **AdminRole** が割り当てられている
- 管理者がユーザーを作成または編集するときに割り当てるタイプのオブジェクトの名前が、必ず作成 / 編集対象ユーザーが属す組織に適した名前となるようにする

たとえば、管理者に **Engineering AdminRole** と **Marketing AdminRole** が割り当てられ、その管理者が **Engineering** 組織に属すユーザーを編集する場合、そのユーザーに割り当てることができるリソースは、**Engineering AdminRole** によって制御される組織で使用できるリソースに制限されます。

scopingOrg 属性の実装

前述の動作を実装するには、ユーザーフォームの `waveset.resources` フィールドに `scopingOrg` 属性を追加します。

現在の組織で使用できる値を参照する方法は次のとおりです。

```

<Field name='waveset.resources'>
  <Display class='MultiSelect'>
    <Property name='title' value='_FM_PRIVATE_RESOURCES' />
    <Property name='availableTitle'
      value='_FM_AVAILABLE_RESOURCES' />
    <Property name='selectedTitle' value='_FM_SELECTED_RESOURCES' />
    <Property name='allowedValues'>
      <invoke class='com.waveset.ui.FormUtil'
        name='getUnassignedResources'>
        <ref>:display.session</ref>
        <map>
          <s>currentRoles</s>
          <ref>waveset.roles</ref>
          <s>currentResourceGroups</s>
          <ref>waveset.applications</ref>
          <s>current</s>
          <ref>waveset.original.resources</ref>
          <s>scopingOrg</s>
          <ref>waveset.organization</ref>
        </map>
      </invoke>
    </Property>
  </Display>
</Field>

```

current

返される値とマージされる名前のリストを指定します。たとえば、選択されたすべての名前が必ず複数選択リストから選択できるようにするときは、通常、**MultiSelect** フィールドで選択される名前のリストを指定します。

conditions

この値は、3つの方法で指定できます。

表 4-33 conditions 属性の値

値の形式	説明
Map	<MapEntry> キーは、このタイプのオブジェクトがクエリーできる属性の名前で、2番目は、関連付けられているクエリー可能属性を取得するために必要な、このタイプのオブジェクトの値です (演算子は「equals」と見なされる)。属性名と値の複数のペアを指定した場合は、それらは論理 and で連結されます。 例 <pre><Map> <MapEntry key='memberObjectGroups' value='Top' /> </Map></pre>
map	最初のエントリは、このオブジェクトタイプがサポートする、クエリー可能な属性の名前です。2番目のエントリは、関連付けられているクエリー可能属性を取得するために必要な、このタイプのオブジェクトの値です (演算子は「equals」と見なされる)。 属性名と値の複数のペアを指定した場合は、それらは論理 and で連結されます。 例 <pre><map> <s>memberObjectGroups</s> <ref>waveset.organizations</ref> </map></pre>

表 4-33 conditions 属性の値 (続き)

値の形式	説明
list	AttributeCondition オブジェクトを一覧表示します。複数の AttributeCondition を指定した場合は、それらは論理 and で連結されます。 例 <pre><list> <newclass= 'com.waveset.object.AttributeCondition'> <s>MemberObjectGroups</s> <s>equals</s> <ref>waveset.organization</ref> </new> </list></pre>

conditions 属性の使用

オブジェクトタイプに固有の、1 つまたは複数のクエリー属性条件のリストを指定して、特定の FormUtil メソッドから返される名前のリストをフィルタリングできます。これらのメソッドには、引数として options マップを受け付けるメソッドが含まれます。これらのクエリー属性条件を、conditions をキーとしたクエリーオプションとして指定することができます。オプションの値にはマップ、または AttributeConditions のリストを指定できます。

例: condition 属性による名前のフィルタリング

次の例は、conditions 属性を使用して、オプションマップを引数として受け付ける FormUtil メソッドから返される名前のリストに、追加のフィルタを適用する方法を示しています。

例 1

```
<Field name='waveset.resources'>
  <Display class='MultiSelect' action='true'>
    ...
  <Property name='allowedValues'>
    <invoke class='com.waveset.ui.FormUtil'
      name='getUnassignedResources'>
      <ref>:display.session</ref>
      <map>
        <s>currentRoles</s>
        <ref>waveset.roles</ref>
        <s>currentResourceGroups</s>
        <ref>waveset.applications</ref>
        <s>current</s>
        <ref>waveset.original.resources</ref>
        <s>conditions</s>
        <map>
          <s>supportsContainerObjectTypes</s>
          <s>true</s>
          <s>type</s>
          <s>LDAP</s>
        </map>
      </map>
    </invoke>
  </Property>
</Display>
</Field>
```

例 2

```
<Field name='orgResource'>
  <Display class='Select' action='true'>
    ...
    <Property name='allowedValues'>
      <invoke class='com.waveset.ui.FormUtil'
        name='getResourcesSupportingContainerObjectTypes'>
        <ref>:display.session</ref>
        <map>
          <s>conditions</s>
          <list>
            <new class='com.waveset.object.AttributeCondition'>
              <s>name</s>
              <s>starts with</s>
              <s>ldap</s>
            </new>
          </list>
        </map>
      </invoke>
    </Property>
  </Display>
</Field>
```

例 3

```
<Field name='accounts[Lighthouse].capabilities'>
  <Display class='MultiSelect'>
    ...
    <Property name='allowedValues'>
      <invoke class='com.waveset.ui.FormUtil'
        name='getUnassignedCapabilities'>
        <ref>:display.session</ref>
        <ref>waveset.original.capabilities</ref>
        <map>
          <s>conditions</s>
          <list>
            <new class='com.waveset.object.AttributeCondition'>
              <s>name</s>
              <s>starts with</s>
              <s>bulk</s>
            </new>
          </list>
        </map>
      </invoke>
    </Property>
  </Display>
</Field>
```

サポートされるクエリー可能属性の名前

サポートされるクエリー可能属性の名前は、オブジェクトタイプごとに次のように分類されます。

- すべてのオブジェクトタイプ
- Account

- AdminGroups
- AdminRole
- Configuration
- Event
- LoginApp
- LoginModGroup
- ObjectGroup
- Policy
- ResourceRole
- WorkItem
- User

上記以外のクエリー可能属性の名前は `UIConfig.xml` に定義されます (たとえば、`firstname` と `lastname`)。

表 4-34 すべてのオブジェクトタイプ

クエリー可能な属性	説明
<code>authType</code>	該当する場合に、認証タイプを指定します (たとえば、 Configuration や Rule などのタイプ)
<code>id</code>	このオブジェクトのリポジトリ ID を指定します
<code>name</code>	このオブジェクトの名前を指定します
<code>memberObjectGroups</code>	このオブジェクトにアクセスできる、またはこのオブジェクトが属す ObjectGroups を指定します

表 4-35 Account

クエリー可能な属性	説明
<code>accountId</code>	リソースの <code>accountId</code> を指定します
<code>accountExists</code>	アカウントが存在するかどうかを指定します (<code>true</code> <code>false</code>)
<code>disabled</code>	アカウントが無効化されているかどうかを指定します (<code>true</code> <code>false</code>)
<code>discoveredSituation</code>	調整中に検出された初期状態を指定します。この状態は、 confirmed と deleted が含まれます。

表 4-35 Account (続き)

クエリー可能な属性	説明
owner	このユーザーのリポジトリ ID を指定します (表示名は userid)
nativeGUID	リソースがこの属性をサポートする場合に、アカウントの GUID を指定します
resource	リソース名を指定します
resourceId	このリソースのリポジトリ ID を指定します (表示名は resource)
situation	検出された状況に応答が適用されたあとの、アカウントの状態を指定します。この状態は、confirmed、missing、deleted が含まれます。
situationChanged	状況が最後に変更された日時を指定します。
typeString	リソースタイプを指定します (たとえば、LDAP)
userid	このアカウントが割り当てられている Identity Manager ユーザーを一意に指定します

表 4-36 AdminGroup

クエリー可能な属性	説明
hidden	AdminGroup を非表示にするかどうかを指定します
memberAdminGroups	この AdminGroup に直接割り当てられる AdminGroups を一覧表示します

表 4-37 AdminRole

クエリー可能な属性	説明
controlledObjectGroups	この AdminRole によって制御される ObjectGroups を一覧表示します
memberAdminGroups	この AdminRole に割り当てられている AdminGroups を一覧表示します
adminGroupsRule	機能規則の名前を指定します
controlledObjectGroupsRule	制御される組織規則の名前を指定します

表 4-38 Configuration

クエリー可能な属性	説明
configType	設定を拡張するクラスの Java クラス名を指定します (たとえば、UserUIConfig、UserForm)

表 4-39 Event

クエリー可能な属性	説明
eventType	類似したイベントをグループ化するために、イベントタイプを指定します

表 4-40 LoginApp

クエリー可能な属性	説明
hidden	LoginApp を非表示にするかどうかを指定します

表 4-41 LoginModGroup

クエリー可能な属性	説明
hidden	LoginModGroup を非表示にするかどうかを指定します

表 4-42 ObjectGroup

クエリー可能な属性	説明
directoryJunction	ObjectGroup がディレクトリジャンクションであるかどうかを指定します (true false)
displayName	オブジェクトグループのユーザーの、判読可能な名前を指定します

表 4-43 Policy

クエリー可能な属性	説明
class	このポリシーを実装する Java クラスを指定します (たとえば、StringQualityPolicy)
typeString	ポリシーのタイプを指定します (たとえば、password または accountId)

表 4-44 Resource

クエリー可能な属性	説明
startupType	startupType を指定します。これには、 disabled と automatic があります
supportsContainerObjectTypes	このリソースが、コンテナオブジェクトタイプをサポートするかどうかを指定します (true false)
supportedObjectTypes	サポートされるオブジェクトタイプを指定します (たとえば、 group 、 ou 、 o 、 domain)
supportsScanning	このリソースがスキャンをサポートするかどうかを指定します (true false)
syncSource	このリソースを Active Sync リソースにすることができるかどうかを指定します (true false)
type	リソースタイプを指定します (たとえば、LDAP、AIX、RACF)

表 4-45 Role

クエリー可能な属性	説明
role_resources	ロールに割り当てられているリソースを一覧表示します
role_approvers	ロールに割り当てられている承認者を一覧表示します

表 4-46 WorkItem

クエリー可能な属性	説明
dirty	
itemType	作業項目のタイプを指定します (たとえば、approval または wizard)
owner	この作業項目の所有ユーザーを指定します
taskId	この作業項目を作成したタスクインスタンスのリポジトリ ID を指定します

表 4-47 User

クエリー可能な属性	説明
adminRoles	このユーザーに割り当てられている AdminRoles のリストを指定します
controlledObjectGroups	このユーザーによって制御されるすべての ObjectGroups を一覧表示します (平常化された文字列)
correlationKey	一括読み込み時にユーザーを識別するためのキーを指定します
dis	現在の無効化状態を表す整数を指定します 0 は、アカウントなしを表します 1 は、一部が無効化されていることを表します 2 は、すべてが無効化されていることを表します
lhdiss	Identity Manager ユーザーが無効化されているかどうかを指定します (true false)
memberAdminGroups	このユーザーに割り当てられている AdminGroups を一覧表示します (平常化された文字列)
prov	現在のプロビジョニング状態を表す整数を指定します 1 は、更新が必要であることを表します 2 は、正常 (OK) 状態を表します
resourceAccountGuids	このユーザーに割り当てられているリソースアカウントの GUID を一覧表示します (accountGUID@resourceId)
resourceAccountIds	このユーザーに割り当てられているリソースアカウントの ID を一覧表示します (accountId@resourceId)

表 4-47 User (続き)

クエリー可能な属性	説明
role	このユーザーに割り当てられているロールを一覧表示します
user_resources	このユーザーに割り当てられているリソースを一覧表示します

サポートされる演算子

- equals または is
- notEquals または is not
- greaterThan または greater than
- greaterThanOrEqualTo または not less than
- lessThan または less than
- lessThanOrEqualTo または not greater than
- startsWith または starts with
- endsWith または ends with
- contains または contains'
- isPresent または exists
- notPresent'
- isOneOf または is one of'
- containsAll

Identity Manager のビュー

この章では、Identity Manager で使用されるデータ構造、Sun Java™ System Identity Manager ビューについて説明します。Identity Manager のワークフロー、フォーム、および参照情報にビューを実装する方法など、ビューに関する背景情報を提供します。

この章の内容

この章は、次の各節で構成されています。

- [Identity Manager ビューについて](#)
- [ユーザービューについて](#)
- [代表的なビュー](#)
- [ビューオプション](#)
- [deferred 属性](#)
- [ビューの拡張](#)

関連する章

- 「[Identity Manager フォーム](#)」 - ユーザービューと呼ばれる Identity Manager の内部データ構造と連携する Identity Manager フォームについて説明します。フォームをカスタマイズするときは、ビューの属性を呼び出します。

Identity Manager ビューについて

Identity Manager ビューは、Identity Manager によって管理される 1 つまたは複数のオブジェクトから構成される属性の集合です。ビューは一時的、かつ動的なもので、リポジトリには保存されません。ビューの表示を更新すると、ビューに含まれるデータは、新しいロールやリソースの割り当てに応じて変更されます。

Identity Manager の使用中は、主にフォームやワークフローにビューが表示されます。Identity Manager フォームは、編集時にビュー属性をどのようにブラウザに表示するかを表現するオブジェクトです。また、フォームには、表示対象属性から非表示属性をどのように算出するかを決める規則も設定されています。ワークフロープロセスは、論理的で反復可能な一連のアクティビティーで、ある関与者から別の関与者にドキュメント、情報、またはタスクが渡されると、一連の手順規則に従ってアクションを実行します。

ビューを操作するときは、まず、次の内容を理解してください。

- ビューの一般的な概念
- Identity Manager でのビューの用途
- ビューのカスタマイズ頻度

ビューとは

もっとも重要なビューは、Identity Manager に格納されているユーザー属性と、Identity Manager が管理するアカウントから読み込まれた属性が含まれるユーザービューです。ユーザービューの一部の属性は、Identity Manager のユーザーインタフェースと管理者インタフェースで使用されるフォームに表示されます。それ以外の属性は、非表示であるか、読み取り専用です。通常、非表示の属性は、別の表示対象属性の導出や、フィールド値の計算に使用されます。

たとえば、ユーザーを作成する際 (ユーザービューとして表される)、管理者は、「ユーザーの作成」ページの適切なフォームフィールドに姓、名を入力します。管理者がフォームを保存すると、システムは姓と名を連結し、非表示フィールドにフルネームを導出できます。このフルネームは、Identity Manager を含め、1 つまたは複数のリソースに保存できるようになります。承認を受けると (承認が必須とされる場合)、システムはユーザービューを Identity Manager リポジトリ内の 1 つまたは複数のオブジェクトに戻し、ユーザーに割り当てられているリソースにビューを送信して、ユーザーのリソースアカウントを作成または更新します。

ビュー属性

ビューとは、リポジトリに格納された 1 つ以上のオブジェクトを組み合わせて構成した名前と値のペア、またはリソースから読み取った名前と値のペアの集合です。ビュー属性では、文字列などの個々の値、リストなどの集合、または別のオブジェクトの参照を値として使用できます。

ブール型のすべての属性は、ビューから削除されます。値が削除された属性は、論理的に `false` と見なされます。

ビューハンドラとは

ビューハンドラは、ビューの作成と指定アクションの実行に必要なロジックを含む Java クラスで、ビューの属性設定によって指定されます。ビューハンドラには、対話的なフォームの利便性を向上させるための情報が含まれる場合があります。ビューがチェックインされると、ビューハンドラはビュー属性を読み取り、それをリポジトリオブジェクトに対する操作に変換します。承認やプロビジョニングなど、より複雑なタスクの実行では、ビューハンドラはワークフローを頻繁に呼び出します。ビューハンドラが、あるユーザーを対象に処理を実行する場合、そのユーザーに対してすでに実行中のワークフローが存在するときは、ほとんどのビューハンドラはビューへのチェックインを回避します。

ビューとフォーム

Identity Manager フォームには、ビューのデータを変換する規則と、ブラウザでビュー属性をどのように表示、変更するかを指定する規則が含まれます。Identity Manager のユーザーインタフェースは、ビューとフォームを処理して HTML フォームを生成します。ユーザーが HTML フォームを送信すると、Identity Manager は、送信された値をビューにマージし、ビューの更新をビューハンドラに要求します。対話的な編集セッションでは、ビューの表示が数回更新されることがあり、フォームに設定されているロジックに基づいて、異なる HTML フィールドを生成できます。ユーザーが対話的な操作を完了すると、ビューはチェックインされ、多くの場合、ワークフロープロセスへの入力としてそのビューが渡されます。

ビューとワークフロー

多くの場合、ビューをチェックインすると、ビューに指定された変更を完了するために、新しいワークフロープロセスが呼び出されます。ワークフローは、時間がかかるタスクをバックグラウンドで実行したり、承認プロセスを呼び出したりすることができます。また、リソースをクエリーしたり、必要に応じて適切なタスクを実行したりすることもできます。承認時は、管理者はビューの内容を確認し、必要に応じて変更を加えることができます。承認されたビューの属性は、1つまたは複数のリポジトリオブジェクトの変更に変換されます。ユーザーに関連するビューでは、選択されたリソースアカウントに変更を反映させるためにプロビジョニングが行われる場合があります。

代表的なビュー

カスタマイズされたフォームとワークフローの両方でよく使われるビューは次のとおりです。

User (ユーザー)	Identity Manager ユーザーの操作とリソースアカウントのプロビジョニングに使用されます。
AccountCorrelation (アカウント 相関)	指定されたアカウント (またはアカウント属性) と相互に関連するユーザーの検索に使用されます。
AdminRole (管理者ロール)	ユーザーに管理者ロールを割り当てるときに使用されます。
Enable (有効化)	無効化するリソースアカウントのリストの表示と選択に使用されます。
Deprovision (プロビジョニング 解除)	プロビジョニング解除するリソースリストの表示と選択に使用されます。
Disable (無効化)	有効化するリソースアカウントのリストの表示と選択に使用されます。
ChangeUserAnswers (ユーザー 回答変更)	ユーザーの秘密質問に対する回答の変更に使用されます。
ChangeUserCapabilities (ユー ザー機能変更)	Identity Manager ユーザーの機能を変更するとき使用されます。
List (リスト)	Identity Manager ユーザーインターフェースの作業項目とプロセスのリストの生成に使用されます。

Org (組織)	作成する組織のタイプと、それを処理するオプションのタイプの指定に使用されます。
Password (パスワード)	Identity Manager ユーザーのパスワードの変更に使われます。また、オプションとして、パスワードをリソースアカウントに反映させます。
Process (プロセス)	ワークフローやレポートなどのタスクの呼び出しに使用されます。
Reconcile (調整)	調整処理の要求またはキャンセルに使用されます。
ReconcileStatus (調整状態)	最後に要求された調整処理の状態を取得するときに使用されます。
RenameUser (ユーザー名変更)	Identity Manager ユーザーとリソースアカウントアイデンティティの名前の変更に使用されます。
Reprovision (再プロビジョニング)	再プロビジョニングするリソースのリストの表示と選択に使用されます。
ResetUserPassword (ユーザーパスワードのリセット)	管理者がパスワードをランダム生成のパスワードにリセットするときに使用されます。また、オプションとして、新しいパスワードをリソースアカウントに反映させます。
Resource (リソース)	リソースの操作に使用されます。
ResourceObject (リソースオブジェクト)	たとえば、グループやメーリングリストのように、リソースによってサポートされる任意のオブジェクトの操作に使用されるビューのファミリーです。
Role (ロール)	作成する Identity Manager ロールのタイプの指定に使用されます。
TaskSchedule (タスクスケジュール)	TaskSchedule オブジェクトの作成と変更に使用されます。
Unlock (ロック解除)	リソース側の機能としてアカウントのロックがサポートされている場合、それらのリソースに対するアカウントのロック解除に使用されます。
WorkItem (作業項目)	ワークフロー承認フォームの記述に使用されます。

WorkItemList (作業項目リスト) リポジトリ内の作業項目の集合に関する情報の表示と、複数の作業項目に対する処理の同時実行に使用されます。

ユーザービューについて

ユーザービューは、Identity Manager ユーザーに関する次のような情報を含む属性の集合です。

- Identity Manager のリポジトリに格納される属性
- リソースアカウントから取得される属性
- リソース、ロール、組織など、ほかのソースから取得される情報

フォームでもっとも頻繁に使用されるユーザービューは、ユーザーを作成または編集するページで使用するよう設計されています。これらのページは、変更されたユーザービューを格納したワークフロープロセスを呼び出し、必要時には、変更されたビューの情報を Identity Manager と関連リソースに戻します。ユーザービューがワークフロープロセスに格納されている間、ワークフロープロセスはワークフローアクションによって属性値を操作できます。ワークフローは、手動アクションや承認フォームによるユーザー入力のために、属性値を表示することもできます。

ユーザービューとフォームの統合

多くの場合、ユーザービューはフォームと組み合わせて使用されます。フォームには、HTML フィールドによってデータをどのように表示するかを制御する規則と、フォームを表示する HTML ページの送信後にデータをどのように処理するかを制御する規則が含まれます。フォーム定義とビューは、フォームジェネレータ (フォーム生成ツール) というシステムコンポーネントによって組み合わされ、ブラウザに表示される HTML ページが生成されます。

フォーム内の HTML コンポーネントに割り当てることで、ビュー属性の値を表示できます (ビュー属性の表示方法の詳細については、[第 8 章「HTML 表示コンポーネント」](#)を参照)。

ビューは、GenericObject クラスのインスタンスとして実装されます。このクラスは、名前と値のペアを表示するメカニズムと、パス式によってオブジェクトの複雑な階層を切り替えるユーティリティを提供します。パス式は、オブジェクト階層を切り替えて属性の値を取得または割り当てるために、実行時に解釈される文字列です。

有効なフォームフィールド名を割り当てるには、パス式の記述方法を理解する必要があります。パス式の使用の詳細については、「パス式」の節を参照してください。

ユーザービューとワークフローの統合

ユーザービューが含まれるワークフロープロセスの多くは、そのビューを `user` というワークフロー変数に格納します。ユーザービューのパスに `user` というプレフィックスを付けることで、ワークフローの式でビューを参照できます。たとえば、`user.waveset.accountId` のように指定します。この `waveset` という文字列は、それ自体がユーザービューオブジェクトに属する `waveset` というオブジェクトを指し、そのオブジェクトに属する属性の `accountId` を識別します。

ビュー用に記述される承認フォームは、*WorkItem* (作業項目) ビューと呼ばれます。作業項目ビューでは、すべてのワークフロー変数が、デフォルトで `variables` という属性に格納されます。ユーザービューを含むワークフロー用に記述された承認フォームでユーザービューの属性を参照するには、`variables.user.` というプレフィックスを使用します。たとえば、`variables.user.waveset.roles` のように指定します。詳細については、この章の「作業項目ビュー」を参照してください。

汎用オブジェクトクラス

高次では、オブジェクトは、名前と値のペアから構成される、名前が付けられた単なる属性の集合に過ぎません。属性の値は、文字列などの個々の値、リストなどの集合、または別のオブジェクトの参照として使用できます。ほとんどすべてのオブジェクトは、`Map`、`List`、`String` の各 `Java` クラスを使用して抽象的に表現できます。

`Identity Manager` システムでは、任意のオブジェクトと集合を表現する単純なメモリーモデルとして `GenericObject` クラスが提供されます。これには、オブジェクト階層を簡単に移動して、属性値を表示、変更するための機能も含まれます。

`GenericObject` クラスは、`java.util.Map` インタフェースを実装し、`java.util.HashMap` を内部的に使用して、名前と値のペアの集合を管理します。このマップのエントリは、属性と呼ばれます。属性の値には、XML として直列化できる任意の `Java` オブジェクトを指定できます。`GenericObject` のもっとも一般的な属性値は次のとおりです。

次のクラスのインスタンスは次のとおりです。

- `String`
- `Integer`
- `Boolean`
- `EncryptedData`
- `List`
- `Date`

- GenericObject
- X509cert

属性値として List または GenericObject を割り当てることで、オブジェクトの複雑な階層を構築できます。属性値の割り当てが完了したら、その階層を切り替えて、属性の値にアクセスします。

パス式

パス式は、オブジェクト階層を切り替えて属性の値を取得または割り当てるために、GenericObject クラスによって実行時に解釈される文字列です。**Identity Manager** は、ドットと括弧を組み合わせて、階層内のオブジェクトと属性を表現します。

フォーム (たとえば、<Field name='user.waveset.roles' />) をカスタマイズするときは、フォームフィールドの name 属性の値としてパス式を指定します。

オブジェクトをたどる

次の例は、2 つの属性が指定された簡単な GenericObject を示しています。

- name (String)
- address (GenericObject) address オブジェクトには、street という文字列の属性があります。

address オブジェクトの street 属性を示すパス式を作成するときは、address.street と指定します。

パス式では、あるオブジェクトから別のオブジェクトへの切り替えをドット文字 (.) で表します。これは、Java で使用されるドット、または C で使用される「->」演算子と似ています。次の例のように、パスが長くなることもあります。

```
user.role.approver.department.name
```

リストの切り替え

パス式を使用して、リストの値を切り替えることもできます。java.util.List という値が指定された children という属性を持つオブジェクトを考えてみましょう。リストを構成する各オブジェクトは、それ自体が name 属性と age 属性を持つ GenericObject です。最初の子の名前を示すパスは、次のように記述します。

```
children[#0].name
```

パス式では、角括弧を使用してリストのインデックスを表します。括弧内のトークンは、インデックス式と呼ばれます。単純な用例では、要素の位置によってリストのインデックスを表す正の整数が指定されます。

通常は、リスト内のオブジェクトの位置は任意です。インデックス式では、簡単な検索条件を指定して、リスト内のオブジェクトを特定することもできます。リスト内のオブジェクトは、多くの場合、name 属性を持ちます。これは、リスト内のオブジェクトを一意に識別する属性です。パス式は、インデックス式に指定された、オブジェクトを示す name 属性の暗黙的な参照をサポートします。

次に例を示します。

```
children[hannah].age
```

このパス式は、children 属性に格納されたオブジェクトのリストを取得します。このリストの検索は、hannah という値を持つ name 属性のオブジェクトが見つかるまで継続されます。一致する属性が見つかり、age 属性の値が返されます。前述の例は、次に示すより一般的な形式の短縮形です。

```
children[name=hannah].age
```

リストの計算

オブジェクトに格納されていない List 値を計算するパス式も記述できます。次に例を示します。

```
accounts[*].name
```

インデックス式に指定されたアスタリスクは、リスト内の各要素に対する繰り返しを表します。式の結果は、リスト内の各要素に残りのパス式を適用した結果のリストとなります。前述の例では、結果は String オブジェクトのリストとなります。文字列は、accounts リスト内の各オブジェクトの name 属性から取得されます。

アスタリスク (*) を持つパス式は、フィールドの集合を複製するフォームで、FieldLoop 構成体と組み合わせて使用されます。

アカウントタイプとユーザー用ビュー

アカウントタイプをユーザーに割り当てると、そのアカウントタイプと accountId を使用できるようになります。User、Enable、Disable、Deprovision などのユーザー用のビューを操作するときは、対処方法を示した次のガイドラインに従ってください。

- デフォルトタイプのアカウントを示すときは、NULL 値を使用します。デフォルトタイプのアカウントを参照するときは、accounts[corp-ad] などのリソース名を使用します
- 特定のタイプのアカウントを参照するときは、リソース名の代わりにタイプ修飾した名前を使用します。タイプ修飾したリソース名の形式は次のとおりです。

<リソース名>|<アカウントのタイプ>

リソース **corp-ad** の Admin タイプのアカウントのアカウントデータを参照するには、`accounts[corp-ad|Admin]` と指定します。

ユーザービューの属性

Web ブラウザからユーザーアカウントを作成または変更するときは、常にユーザービューを間接的に操作していることになります。ユーザーのアカウント情報の変更という点では、これは **Identity Manager** システムでもっとも重要なビューであるともいえます。

ワークフロープロセスも、ユーザービューと連携しています。要求をワークフロープロセスに渡すと、属性がビューとしてプロセスに送信されます。ワークフロープロセスの中で手動プロセスを要求すると、ユーザービューの属性を表示して詳細に変更できます。

MetaView 属性

配備がアイデンティティ属性を使用する場合は、ユーザービューに追加のネームスペースが作成されます。この追加ネームスペースは *metaView* と呼ばれ、アイデンティティ属性に関する情報を持ちます。この **MetaView** オブジェクトは、**Identity Manager** リポジトリへのメタビューおよびアイデンティティ属性情報の格納に使用されます。

定義されるアイデンティティ属性ごとに、*metaView* ネームスペース内にこの属性の値を格納し、アイデンティティ属性を保持します。たとえば、`firstname`、`lastname`、および `waveset.roles` というアイデンティティ属性がある場合、ユーザービューはそれぞれに対応する `metaView.firstname`、`metaView.lastname`、および `metaView.waveset.roles` という属性を持ち、各アイデンティティ属性の計算値が保持されます。

「調整」、「リソースから読み込み」、または「ファイルから読み込み」用のユーザービューに値を取り込むときに、**MetaView** は、**Identity Manager** ユーザーに割り当てられたリソースからのリソースアカウントデータの読み込みのみをサポートします。これらのリソースは、読み込み中のリソースをソースとするターゲットでもあるため、変更される可能性があります。**Identity Manager** ユーザーの、アカウントに影響を与えないオペレーショナル属性（たとえば、アカウント ID や組織）は、`firstname` や `lastname` などのユーザー拡張属性と同様に、常にビュー内にあります。

詳細については、『**Identity Manager** の配備に関する技術情報』で属性の操作に関する情報を参照してください。

はじめに

その他のビューと同様に、ユーザービューも属性セットを持つ `GenericObject` として実装されます。ルートオブジェクト内の属性の値は、`GenericObjects` 自体です。属性は、入れ子構造にすることができます。

ユーザービューは、次の表に示される属性を持ちます。各属性については、後の節で詳しく説明します。

表 5-1 ユーザービューの最上位属性

属性	説明
<code>waveset</code>	Identity Manager repository (WSUser object) に格納されている情報を持ちます。このビューは、基本ビューとも呼ばれます。
<code>accounts</code>	リソースからフェッチされたすべてのリソースアカウント属性の値を持ちます。通常、これらの値はフォームで編集される値です。
<code>accountInfo</code>	ユーザーに関連付けられているリソースとアカウントに関する読み取り専用の情報を持ちます。
<code>display</code>	インタフェースの実行時状態に関する読み取り専用の情報を持ちます。これは、ユーザーの対話的な編集のみに使用されます。 <code>display.session</code> は、ログイン情報とアクセス情報を表します。 <code>display.subject</code> は、ユーザーがログインに使用しているアカウントを識別します。 <code>display.eventType</code> は、ユーザービューが作成処理と更新処理のどちらに使用されているかを表します。
<code>global</code>	すべてのリソースアカウントの間で同期される属性を持ちます。
<code>password</code>	ユーザーのパスワード、パスワードの有効期限、およびターゲットシステムに固有の属性値を持ちます。

フォームを設計するとき、通常は、ユーザービューオブジェクトである `waveset`、`global`、および `account` 属性 (たとえば、`global.firstname`) へのパスがフィールド名となります。

適切な変数ネームスペースの選択

アカウント関連の情報を導出するために、ユーザービューにはいくつかのネームスペースがあります。次の表は、変数ネームスペースの概要を示しています。

表 5-2 アカウント関連のユーザービュー属性

アカウント関連のネームスペース	説明
waveset.accounts	チェックイン処理の実行中に差異を検出するために、内部的に使用されます。これには、すべてのアカウント属性の開始値が含まれます。この値を変更しないでください。
accountInfo.accounts	ユーザー、およびその関連リソースへのリンクを持つアカウントに関する、導出された読み取り専用の情報。この属性はフォームで使用できますが、変更は行わないでください。
accounts	アカウント属性の読み取り / 書き込みコピーを保持します。更新可能なフィールドは、このネームスペースを指すようにします。
global	グローバル属性のコピーを保持します。このエリアの値は、フォームがグローバルフィールドを定義する場合、または特別な MissingFields 参照を使用している場合にのみ表示されます (グローバル属性をどのように処理するかは、フォームによって決定される)。 ワークフローにグローバル属性を設定するときは、フォームにグローバルフィールドも定義してください。ビューにグローバルな値を蓄積するだけでは不十分です。

属性の参照

フォームでは、次の 2 つの方法で属性を参照できます。

- Field 要素の名前属性を使用して、属性の完全パス名を指定します。次に例を示します。

```
<Field name='waveset.accountId'>
```

フォームフィールドへの Field 要素の名前属性の設定の詳細については、「*Identity Manager* フォーム」の章を参照してください。

- 別のフィールドから属性を参照します。

```
<Expansion>
```

```
<concat>
```

```
<ref>global.firstname</ref><s> </s>
```

```

        <ref>global.lastname</ref>

    </concat>

</Expansion>

```

ワークフローでは、プロセス変数 (ワークフローエンジンが認識できる変数)、またはアクションと遷移の XPRESS ステートメントとして Field 属性を参照できます。ワークフローでこれらの属性を参照するときは、ビューが格納されているワークフロービューの名前をプレフィックスとしてパスに追加します (たとえば、`user.waveset.accountId`)。

一時的な値を持つ属性

ユーザービューの最上位に、一時的な値を格納するフィールドを定義できます。これらのフィールドの値は、メモリ内でユーザービューが有効な間 (通常はプロセス終了までの間) は存続しますが、**Identity Manager** リポジトリに格納されることも、リソースアカウントに反映されることもありません。

たとえば、電話番号の値は、3 つのフォームフィールドの値を連結した結果です。次の例では、p1 は市外局番、p2 と p3 は電話番号の残りの部分をそれぞれ参照します。これらの値は連結され、`global.workPhone` というフィールドに格納されます。リソースに反映する値は連結された電話番号のみであるため、このフィールドにのみ `global` というプレフィックスが追加されます。

一般に、最上位フィールドの構文は次の場合に使用されます。

- フィールドの値を **Identity Manager** またはその他のリソースに渡さない
- フィールドが、電子メール通知、または別のフィールドの計算のみに使用される

次のレベルに渡されるすべてのフィールドには、前述の「ユーザービューの属性」の表で説明した、いずれかのパスプレフィックスを付ける必要があります。

```

Field name='p1' required='true'>
    <Display class='Text'>
        <Property name='title' value='Work Phone Number' />
        <Property name='size' value='3' />
        <Property name='maxLength' value='3' />
    </Display>
</Field>

<Field name='p2' display='true' required='true'>
    <Display class='Text'>
        <Property name='rowHold' value='true' />
    </Display>
</Field>

```

```

        <Property name='noNewRow' value='true' />
        <Property name='size' value='3' />
        <Property name='maxLength' value='3' />
    </Display>
</Field>
<Field name='p3' display='true' required='true'>
    <Display class='Text'>
        <Property name='rowHold' value='true' />
        <Property name='noNewRow' value='true' />
        <Property name='size' value='4' />
        <Property name='maxLength' value='4' />
    </Display>
</Field>
<Field name='global.workPhone' required='true' hidden='true'>
    <Expansion>
        <concat>
            <ref>p1</ref>
            <s>-</s>
            <ref>p2</ref>
            <s>-</s>
            <ref>p3</ref>
        </concat>
    </Expansion>
</Field>

```

waveset 属性

waveset 属性セットは、Identity Manager リポジトリ内の WSUser オブジェクトに格納されている情報を持ちます。この属性セットの下に入れ子になった一部の属性は、フォーム内での直接的な操作には使用されず、ビュー内の WSUser オブジェクトのすべての情報を Identity Manager が完全に表現できるように指定されます。

頻繁に使用される属性

新規ユーザーを作成するときに、すべての属性が必要となるわけではありません。次のリストは、作成または編集でよく使用される waveset 属性を示しています。一部の属性は読み取り専用ですが、その値は、別の属性の値の計算に使用されます。この表の後の節で、すべての waveset 属性について説明します。

表 5-3 頻繁に使用される waveset の属性 (ユーザービュー)

属性	編集の可能性	データ型
waveset.accountId	読み取り / 書き込み	String
waveset.applications	読み取り / 書き込み	String
waveset.correlationKey	読み取り / 書き込み	String
waveset.creator	読み取り専用	String
waveset.createDate	読み取り専用	String
waveset.disabled	読み取り / 書き込み	String
waveset.email	読み取り / 書き込み	String
waveset.exclusions	読み取り / 書き込み	List
waveset.id	読み取り	String
waveset.lastModDate	読み取り	String
waveset.lastModifier	読み取り	String
waveset.locked	読み取り	String
waveset.lockExpiry	読み取り / 書き込み	String
waveset.organization	読み取り / 書き込み	String
waveset.questions	読み取り / 書き込み	List
waveset.resources	読み取り / 書き込み	List
waveset.resourceAssignments		
waveset.roles	読み取り / 書き込み	String
waveset.serverId	読み取り / 書き込み	String

waveset.accountId

Identity Manager ユーザーオブジェクトの表示名を指定します。この属性は、ユーザー作成時に設定します。ユーザーの作成後にこの属性の設定を変更すると、Identity Manager アカウントの名前変更が開始されます。

ユーザー名の変更については、『Identity Manager 管理ガイド』を参照してください。

waveset.applications

ユーザーに直接割り当てられている各アプリケーションの名前のリスト (Identity Manager のユーザーインタフェースではリソースグループ) を持ちます。ロールを通じてユーザーに割り当てられるアプリケーションは、これに含まれません。

waveset.attributes

Identity Manager リポジトリ内の WSUser オブジェクトに格納されている任意の属性の集合。waveset.attributes 属性の値は NULL か、または別のオブジェクトです。このオブジェクトに格納されている属性の名前は、*Extended User Attributes* というシステム設定オブジェクトによって定義されます。拡張された属性の例としては、firstname、lastname、fullname が一般的です。これらの属性は、次の方法で参照できます。

```
waveset.attributes.fullname
```

または

```
accounts[Lighthouse].fullname
```

通常は、waveset.attributes 属性の値を変更することはありません。その代わりに、accounts[Lighthouse] 属性の値を変更します。属性が格納されると、accounts[Lighthouse] 内の値は格納前に waveset.attributes にコピーされます。waveset.attributes は、属性の元の値の記録に使用されます。システムは、ここに記録されている値と accounts[Lighthouse] に格納されている値を比較し、更新された概要レポートを生成します。ユーザー属性の拡張方法を示す例については、account[Lighthouse] 属性の説明を参照してください。

waveset.correlationKey

調整時およびユーザー検出時にユーザーを特定するために使用される、相互関係の値を持ちます。この属性は、通常は公開されませんが、直接編集できます。

waveset.creator

このユーザーを作成した管理者の名前を持ちます。

この属性は読み取り専用です。

waveset.createDate

このアカウントが作成された日時を持ちます。日時は、MM/dd/yy HH:mm:ss z という形式で表示されます。

例

05/21/02 14:34:30 CST

この属性は 1 度だけ設定され、それ以後は読み取り専用です。

waveset.disabled

Identity Manager ユーザーの無効化状態を持ちます。アカウントが無効化されると、この属性は論理 **true** に設定されます。メモリーモデルでは、ブール型のオブジェクトであるか、true または false の文字列です。フォームからアクセスするときは、文字列と見なされます。

この属性の設定を変更することで、Identity Manager ユーザーを有効または無効にすることができます。ただし、global.disable を使用するほうが一般的です。システムが、変数を認識するすべてのリソース (Identity Manager を含む) に変数の値を必ず適用するようにするときは、変数名に global. というプレフィックスを追加します。

この値が **true** になると、ユーザーは Identity Manager のユーザーインターフェースにログインできなくなります。

waveset.email

Identity Manager リポジトリに格納される、ユーザーの電子メールアドレスを指定します。これは、通常は、リソースアカウントに反映される電子メールアドレスと同じです。

この属性の変更は、Identity Manager リポジトリのみに適用されます。電子メールの各種値をリソース間で同期させるときは、global.email 属性を使用します。

この属性の設定は変更可能です。

waveset.exclusions

ロール、リソースグループ、またはディレクトリを通じてユーザーにリソースが割り当てられる場合でも、プロビジョニングから除外されるリソースの名前を指定します。

waveset.id

Identity Manager ユーザーオブジェクトのリポジトリ ID を特定します。Identity Manager にユーザーを作成すると、この属性は NULL 以外の値になります。この値を調べることで、ユーザーが作成中であるか、編集中心であるかを確認できます。この属性は、フォーム内のロジックによって調べられます。この値を使用して、新規ユーザーの作成中であるか (waveset.id の値が NULL)、既存ユーザーアカウントの編集中心であるか (waveset.id の値が NULL 以外) に応じて表示されるフィールドをカスタマイズできます。

例

次の例は、waveset.id の値が NULL であるかどうかを調べる XPRESS ステートメントを示しています。

```
<isnull><ref>waveset.id</ref></isnull>
```

waveset.lastModDate

最後に加えられた変更の日時を持ちます。この日時は、1970 年 1 月 (GMT) の深夜零時から経過したミリ秒数で表されます。この属性は、ユーザーアカウントを変更するたびに更新されます。

この属性は読み取り専用です。

waveset.lastModifier

このユーザーアカウントを最後に変更した管理者またはユーザーの名前を持ちます。

この属性は読み取り専用です。

waveset.locked

ユーザーがロックされているかどうかを示します。値が true の場合は、ユーザーはロックされています。

waveset.lockExpiry

ユーザーの Lighthouse アカウントポリシーに、ロックされたアカウントの有効期限日としてゼロ以外の値が設定されている場合に、ユーザーロックの有効期限がいつ切れるかを指定します。この属性の値は、判読可能な日時です。

waveset.organization

ユーザーが所属する組織 (または ObjectGroup) の名前を持ちます。新しい組織に関する十分な権限を持っている管理者は、この属性の設定を変更できます。

組織の変更は重大なイベントであるため、組織の元の値も `waveset.original` 属性に格納されます。格納された値は、後で実行する比較に使用できます。

waveset.original

`waveset` 属性のいくつかの重要な属性の変更前の値に関する情報を持ちます。ビューが作成されると、システムがこの値を設定し、その後も変更されません。システムは、概要レポートと監査ログレコードの作成時に、この情報を使用します。

`waveset` 属性の変更前のすべての属性値がここに保存されるわけではありません。変更の追跡が現在定義されている属性は次のとおりです。

- `password`
- `role`
- `organization`

これらの属性を参照するときは、属性名のプリフィックスとして `waveset.original.` を追加します (たとえば、`waveset.original.role`)。

password

Identity Manager ユーザーのパスワードを指定します。ビューが新規に作成された直後は、この属性は暗号化されたユーザーパスワードを持ちません。その代わりに、ランダムに生成された文字列が設定されます。

`password` 属性セットは、次の表に示される属性を持ちます。

表 5-4 password の属性 (ユーザービュー)

属性	説明
<code>password</code>	設定するパスワードを指定します。
<code>confirmPassword</code>	設定するパスワードを確認します。パスワードは、 <code>password.password</code> の値と一致する必要があります。
<code>targets</code>	パスワードを変更できるリソースのリストを指定します。
<code>selectAll</code>	パスワードをすべてのリソースに反映させることを表す、ブール型のフラグを指定します。
<code>accounts[]</code>	各リソースに関する情報を持つオブジェクトのリストを指定します。この属性は、次に説明する 2 つの属性を持ちます。
<code>accounts[<resource>].selected</code>	ブール型。設定した場合は、そのリソースでパスワードの変更が必要であることを示します。

表 5-4 password の属性 (ユーザービュー) (続き)

属性	説明
accounts[<resource>]. expire	ブール型。設定した場合は、パスワードの有効期限が切れることを示します。 ユーザーが自身のパスワードを変更する場合は、この属性は false に設定されます。ただし、管理者が別のユーザーのパスワードを変更する場合は、このフラグは true に設定されます。 管理者またはユーザー以外のプロキシアカウントがアカウントのパスワードを変更する場合に、パスワードの有効期限が切れないようにするには、次のように設定します。 <pre>accounts [<resource>].expire = <s>false</s></pre> この設定は、次のように機能します。 パスワードの有効期限は切れません Identity Manager はパスワードの変更をユーザーに再強制しません

waveset.passwordExpiry

Identity Manager パスワードの有効期限が切れる日付を持ちます。ビューの新規作成時は、メモリ表現は `java.util.Date` オブジェクトとなります。ビューがフォームで処理されると、この属性の値は `mm/dd/yy` という形式の日付のテキスト表現を持つ `Date` オブジェクト、または `String` オブジェクトのいずれかとなります。

waveset.passwordExpiryWarning

ユーザーが **Identity Manager** のユーザーインタフェースにログインするときに、警告メッセージを表示し始める日付を持ちます。通常は、`waveset.passwordExpiry` の日付より前の、同じ形式 (`mm/dd/yy`) の日付を持ちます。

waveset.questions

そのユーザーに割り当てられる秘密の質問と、その回答に関する情報を持ちます。この属性の値は、`waveset.questions` 属性を要素として持つ `List` です。

`waveset.questions` 属性セットは、次の表に示される属性を持ちます。

表 5-5 waveset.questions の属性 (ユーザービュー)

属性	編集の可能性	説明
answer	読み取り / 書き込み	暗号化された、秘密の質問に対する回答
id	読み取り	秘密の質問に割り当てられる、システム生成の ID
name	読み取り	この質問を識別する名前
question	読み取り	秘密の質問の内容 (テキスト)

name 属性は格納されません。システムは、id を変換して名前を生成します。このようなプロセスが必要となるのは、質問 ID は通常は番号であり、パス式で配列のインデックスに使用される番号がオブジェクト名としてではなく、絶対インデックスと見なされるためです。

たとえば、waveset.questions[#1].question という式は、質問リストの 2 番目の要素を示します (リストインデックスの開始番号はゼロ)。ただし、リストに含まれる質問が 1 つのみで、その ID 番号が 1 である場合、ID はリストインデックスには適さない場合があります。リストの要素を確実に指定するために、システムは、q という文字と、それに続く ID という構成の名前 (たとえば、Q1) をそれぞれの質問に付けます。これにより、waveset.questions[Q1].question というパスは、常に正しい質問を示すようになります。

waveset.resources

ユーザーに直接割り当てられる各リソースの名前のリストを持ちます。ロールやアプリケーションを通じてユーザーに割り当てられるリソースは、このリストには含まれません。この属性に追加できるのは、非修飾リソース名のみです。ユーザーに割り当てられているすべてのリソースを確認する方法については、accountInfo 属性の説明を参照してください。

waveset.resourceAssignments

割り当てられているリソースのリストを修飾します。この属性は、既存の waveset.resources 属性と並立します。この属性に含まれるすべてのリソースは、waveset.resources 属性には非修飾の状態で格納されます。ユーザーにデフォルト以外のタイプのアカウントのみが割り当てられている場合でも、リソースは waveset.resources に格納されます。

`waveset.resource` または `waveset.resourceAssignments` に新しい割り当てを追加することができ、ビューの表示を更新すると、リストは自動的に再同期されます。これにより、デフォルトタイプのアカウントの割り当てが追加されます。
`waveset.resourceAssignments` には、修飾と非修飾の両方の形式のリソース名を追加できます。これにより、修飾子に基づいて指定されるタイプのアカウントが追加されます。

waveset.roles

このユーザーに割り当てられるロールの名前を持ちます。新しいロールに関する十分な権限を持っている管理者は、この属性の設定を変更できます。

ロールの変更は重大なイベントであるため、ロール属性の元の値も元のビューに格納されます。格納された値は、後で実行する比較に使用できます。

waveset.serverId

1 つの物理サーバー上の 1 つのリポジトリを指す複数の **Identity Manager** インスタンスが配備に含まれる場合に、一意のサーバー名を設定するために使用されます。詳細については、『**Identity Manager** インストール』を参照してください。

accounts 属性

`accounts` 属性は、**Identity Manager** ユーザーにリンクされた各アカウントのオブジェクトのリストを持ちます。各アカウントオブジェクトは、リソースから取得したアカウント属性の値を持ちます。

各アカウントオブジェクトの名前は、通常は、関連付けられているリソースの名前です。特定のリソースに複数のアカウントが関連付けられている場合は、オブジェクト名に `|n` という形式のサフィックスが付けられます。この `n` は整数を表します。リソースの最初のアカウントには、サフィックスは付けられません。2 番目のアカウントのサフィックスは `|2` となります。リソースの 3 番目のアカウントには `|3` というサフィックスが付けられ、以後、同様に続きます。

たとえば、**Profile** というアカウント属性を定義する、**Exchange Server** というリソース名がある場合、この属性を示すビューパスは次のようになります。

```
accounts[Exchange Server].Profile
```

このビューパスをフォームフィールドで使った場合は、`global.Profile` 属性の値は **Exchange Server** アカウントに反映されなくなります。

注	すべてのリソースへの値の反映を回避するために、グローバル属性の代わりに、アカウントに固有の属性をフォームで使うことが必要となる場合もあります。

リソース属性のオーバーライド

アカウント属性を設定する以外に、各アカウントのリソース属性のオーバーライドも指定できます。リソース属性は、**Identity Manager** のリソース定義用に定義され、したがって、リソースタイプ用に定義される属性です。これらは、個々のアカウントに関連付けられた属性ではありません。リソース属性の例には、サーバーのホスト名や、ディレクトリ内のベースコンテキストなどがあります。

リソースにアカウントを作成するが、1つのリソース属性で別の値を使用する場合を考えてみましょう。これは、リソースを複製し、値を変更することで解決できます。しかし、リソースの過度な複製は混乱を生じる可能性があります。その代わる方法として、ビューでアカウントごとにリソース属性をオーバーライドすることができます。

リソース属性のオーバーライドは、`resourceAttributes` という属性の下属性オブジェクトに格納されます。たとえば、リソースが `host` という属性を定義した場合は、次のパスを記述することでビューに指定できます。

```
accounts[Exchange Server].resourceAttributes.host
```

注

リソース属性のオーバーライドはあまりお勧めできませんが、それを避けられない場合もあります。同じ物理リソースを示すが、1つの属性が異なる複製リソースの作成を回避するために、リソースの上書きを選択する場合もあります。たとえば、複数の **Exchange 5.5** サーバーを使用する環境であれば、新しいリソースを作成するよりも、フォームで **Exchange Server** のリソース属性をオーバーライドするほうが賢明かもしれません。詳細については、**Identity Manager** のご購入先までお問い合わせください。

accounts[Lighthouse]

Identity Manager リポジトリに格納されている属性のみの値を設定します。新規作成したビューは、`waveset.attributes` 属性セットに含まれる属性のコピーを持ちます。ビューを保存するときに、システムは `accounts[Lighthouse]` の内容と `waveset.attributes` の内容を比較し、更新レポートと監査ログエントリを生成します。この属性は **Identity Manager** のリポジトリに格納されますが、この属性の変更はリソースには自動的に反映されません。

拡張ユーザー属性設定オブジェクトは、このビューで利用できる属性を定義します。システムは、この属性セットの中で、設定オブジェクトに登録されていない名前を無視します。

次のコードは、拡張ユーザー属性設定オブジェクトの例を示しています。このオブジェクトは、`waveset.attribute` 属性セットによって管理される属性のリストを保持します。

```

<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE Configuration PUBLIC 'waveset.dtd' 'waveset.dtd'>
<!-- id="#ID#Configuration:UserExtendedAttributes" name="User Extended
Attributes"-->

  <Configuration id='#ID#Configuration:UserExtendedAttributes'
name='User Extended Attributes' creator='Configurator'
createDate='1019603369733' lastMod='2' counter='0'>

    <Extension>
      <List>
        <String>firstname</String>
        <String>lastname</String>
        <String>fullname</String>
<!-- 文字列の値をここに追加 - - >
        <String>SSN</String>
      </List>
    </Extension>

    <MemberObjectGroups>
      <ObjectRef type='ObjectGroup' id='#ID#Top' name='Top' />
    </MemberObjectGroups>
  </Configuration>

```

オブジェクトを修正し、デフォルトの `firstname`、`lastname`、および `fullname` 属性からリストを拡張できます。この例では、`SSN` という属性が追加されています。

accounts[Lighthouse].delegateApproversTo

ユーザーが承認を委任している相手を指定します。有効な値には、`manager`、`selectedUsers`、または `delegateApproversRule` があります。

accounts[Lighthouse].delegateApproversSelected

- `delegateApproversRule` の値が `selectedUsers` である場合は、選択したユーザーの名前リストを指定します。

- `delegateApproversTo` の値が `delegatedApproversRule` である場合は、選択した規則を指定します。
- `delegateApproversTo` の値が `manager` である場合は、この属性は値を持ちません。

accounts[Lighthouse].delegateApproversStartDate

承認の委任を開始する日時を指定します。デフォルトでは、開始日時は指定した日付の午前 12:01 となります。

accounts[Lighthouse].delegateApproversEndDate

承認の委任を終了する日時を指定します。デフォルトでは、終了日時は指定した日付の午後 11:59 となります。

accounts[Lighthouse].properties

この属性の値は、ユーザーが定義したプロパティに対応する属性名を持つオブジェクトです。ユーザープロパティを設定することで、任意のカスタムデータをユーザーとともに Identity Manager リポジトリに格納できます。格納されたプロパティは、フォームやワークフローで使用できます。プロパティは、拡張ユーザー属性と似ていますが、文字列や整数のような基本的なデータ型に限定されません。

Identity Manager は `tasks` システムプロパティを定義します。このプロパティは、将来の特定の日にワークフロータスクを実行するために、延期タスクスキャナで使用されます。`tasks` プロパティの値は、オブジェクトのリストです。次の表は、リストに含まれるオブジェクトの属性を示しています。

表 5-6

属性	説明
<code>name</code>	実行する <code>TaskDefinition</code> オブジェクトの名前を指定します。
<code>date</code>	タスクを実行する日付を指定します。
<code>taskName</code>	作成される <code>TaskInstance</code> を指定します。指定しない場合は、Identity Manager によってランダムな名前が生成されます。
<code>owner</code>	タスクの所有者と見なされる Identity Manager 管理者を指定します。指定しない場合は、デフォルト値の <code>Configurator</code> が適用されます。
<code>organization</code>	<code>TaskInstance</code> の配置先となる Identity Manager 組織を指定します。指定しない場合は、タスク所有者によって制御される組織がランダムに選択されます。

表 5-6 (続き)

属性	説明
description	作成時に TaskInstance に格納される説明文。このテキストは、 Identity Manager の管理者インタフェースのタスク状態ページに表示されます。

用例

`accounts[Lighthouse].properties` の値を使用することで、ユーザーに割り当てられている延期タスクの表を表示できます。このリストは、`sample/formlib.xml` 内の **Default User Library** というフォームライブラリに追加されます。

延期タスクの表を表示するフィールドには、**Deferred Tasks** という名前が付けられています。`waveset.properties` 属性を変更すると、デフォルトの **Tabbed User Form** で延期タスクテーブルを参照できるようになります。延期タスクが存在する場合は、「ID」タブパネルの下部に表が表示されます。

`accounts[Lighthouse].viewUserForm`

表示専用ユーザーフォームの表示に使用されます。この表示専用フォームには、フィールド情報が **Labels** として表示され、管理者はそのユーザー情報の値を変更することはできませんが、一覧、表示、および検索を行うことは可能です。管理者がユーザーの詳細を表示するときは、アカウントリストからユーザーを選択し、「表示」をクリックします。

`accounts[<resource>].properties`

Identity Manager リポジトリへのアカウントプロパティの格納に使用されます。この属性は、アカウントに関する何らかの情報 (たとえば、アカウントの作成日) をネイティブアカウント属性としてリソースに格納できない場合に使用されます。

`accounts[<resource>].waveset.forceUpdate`

この属性は、ユーザーが変更されたあとに、属性値が変更前の状態でリソースアクションに残っている場合に、更新のためにリソースに常に送信されるリソースアカウント属性のリストを指定するときに使用されます。この属性は、ユーザーがリソースから割り当てられていない場合のリソースアクションの実行で必要となります。

ユーザーフォームからの次のフィールド定義は、**Solaris** リソースを使用しています (`<resource>` は具体的なリソース名に置き換えられている)。

```
<Field name='accounts[waterloo].waveset.forceUpdate'>
  <Default>
    <List>
      <String>delete after action</String>
      <String>Home directory</String>
    </List>
  </Default>
</Field>
```

前述のコードにより、**Identity Manager** はプロビジョニングツールとリソースアダプタに、delete after action 属性と Home directory 属性を送信します。

global 属性

ユーザービューの global 属性セットを使用することで、多数のリソースアカウント (**Identity Manager** を含む) に簡単に属性を割り当てることができます。global 属性の値は、global 属性として参照される属性を持つオブジェクトです。ビューを保存すると、システムは、スキーママップに global 属性名を定義するすべてのリソースアカウントに、各 global 属性の値を割り当てます。同じ名前の拡張属性が存在する場合、これらの値は **Identity Manager** のリポジトリにも反映されます。

たとえば、R1 と R2 という 2 つのリソースが、fullname という属性を定義するとします。ビューに global.fullname 属性が格納されると、その値は accounts[R1].fullname 属性と accounts[R2].fullname 属性に自動的にコピーされます。

global 属性を使用して、**Identity Manager** リポジトリに格納される拡張属性を割り当てることもできます。global 属性が **Identity Manager** の拡張属性としても宣言されている場合、これは accounts[Lighthouse] にコピーされます。

注	アカウントの作成時は、global.accountId を使用しないでください。アカウント ID は、リソースの DN テンプレートによって作成されます。global.accountId はこれに優先して適用されるため、問題が生じる可能性があります。

異なる 2 つの fullname 属性の参照

global 属性は、同じ名前を持つ account 属性と組み合わせて使用できます。たとえば、Active Directory リソースでは、fullname は lastname と firstname から構成されます。しかし、それ以外のすべてのリソースの fullname は、firstname lastname を使用します。

次の例は、フォームでこれら 2 つのフィールドを参照する方法を示しています。

```
<Field name='global.fullname'>
  <Expansion>
    <concat>
      <ref>global.firstname</ref><s> </s>
      <ref>global.lastname</ref>
    </concat>
  </Expansion>
</Field>

<Field name='accounts[ActiveDir].fullname'>
  <Expansion>
    <concat>
      <ref>global.lastname</ref><s>, </s>
      <ref>global.firstname</ref>
    </concat>
  </Expansion>
</Field>
```

前述の例では、新規ユーザーの作成は、想定どおりに行われます。ただし、そのユーザーを読み込むときは、Active Directory リソースからの fullname 属性を使用して、global.fullname フィールドに値が取り込まれる可能性があります。

このような事例をより正確に実装するには、1 つのリソースを信頼できる属性のソースとして宣言し、次のような Derivation 規則を作成します。

```

<Field name='global.fullname'>
  <Derivation>
    <または>
      <ref>accounts[LDAP res].fullname</ref>
      <ref>accounts[NT res].fullname</ref>
    </or>
  </Derivation>
  <Expansion>
    <concat>
      <ref>global.firstname</ref><s> </s>
      <ref>global.lastname</ref>
    </concat>
  </Expansion>
</Field>

```

Derivation 規則を定義したことで、fullname フィールドへの値の取り込みには、最初に LDAP リソース内の fullname 属性が使用されます。LDAP に値が存在しない場合は、NT リソースからの値が取り込まれます。

accountInfo 属性

ユーザーに関連付けられているリソースアカウントの、読み取り専用の情報を持ちます。これは、ユーザービュー以外にも、システムビューで使用されます。このビューの一部の情報は、waveset.accounts 属性に格納されている情報の複製です。この複製には、2つの理由があります。

- このビューの情報は、フォームで使いやすいように構成されている
- waveset ビュー全体を含めなくても、このビューは別のビューのコンポーネントとして使用できる

accountsInfo.accounts 属性には、ほとんどのアカウント情報が格納されます。その他の属性は、単にアカウント名のリストを含むに過ぎません。いずれかの名前リスト属性に格納されている名前に対して繰り返しを行い、この名前を使用してアカウントリスト属性にインデックスを指定する場合は、フォームの FieldLoop がよく使用されます。

たとえば、次のフォーム要素は、ロールを通じて間接的に割り当てられた各リソースの名前を持つラベルのリストを生成します。

```
<FieldLoop for='name' in='accountInfo.fromRole'>
  <Field name='accountInfo.accounts[${(name)}].name'>
    <Display class='Label' />
  </Field>
</FieldLoop>
```

次の表は、accountInfo ビュー属性を示しています。これらの属性は、ユーザーの特性を表現しています。

表 5-7 accountInfo の属性 (ユーザービュー)

属性	説明
accountInfo.accounts	ユーザーに関連付けられている各リソースアカウントに関する情報を持つオブジェクトのリスト (たとえば、created、disabled)。
accountInfo.assigned	ユーザーに割り当てられているリソースのリスト。
accountInfo.fromRole	ルールを通じてユーザーに割り当てられているリソースの、フラットリスト形式のリスト。
accountInfo.privates	ユーザーに直接割り当てられているリソースの、フラットリスト形式のリスト。
accountInfo.toCreate	ユーザーには割り当てられているが、そのアカウントが Identity Manager にはまだ存在しないすべてのリソースの名前のリスト。
accountInfo.toDelete	ユーザーには割り当てられなくなったが、存在は認識されているリソースの名前のリスト。
accountInfo.types	ユーザーに現在割り当てられている、または予約グループを通じて割り当てられている、各リソースタイプのリスト。
accountInfo.typeNames	割り当てられているすべてのリソースの、一意のタイプ名のリスト。

accountInfo.accounts

関連付けられている各リソースアカウントに関する情報をそれ自体に持つ、オブジェクトのリストを持ちます。アカウントリストの要素は、名前で参照されます。この名前は、リソース名です。

例

```
accountInfo.accounts[Microsoft Exchange].type
```

accountInfo.accounts リストを構成するオブジェクトは、次の表に示される属性を持ちます。

表 5-8 accountInfo.accounts の属性 (ユーザービュー)

属性	説明
attributes	このリソースによって定義されるすべてのアカウント属性に関する情報。
name	アカウントが存在するか、またはアカウントが作成されるリソースの名前。
id	リソースのリポジトリ ID。
type	リソースのタイプ名。
accountId	このリソースでのユーザーアカウントの名前。
assigned	アカウントが現在割り当てられている場合は、 true となります。割り当てられていないアカウントは、 Identity Manager によって削除される可能性があります。
protected	アカウントが現在保護されている場合は、 true となります。この場合、そのアカウントに対する更新や削除の操作は無視されます。
passwordPolicy	このリソースに適用されるパスワードポリシーに関する情報。

accountInfo.accounts[].attributes[]

このリソースによって定義されるすべてのアカウント属性に関する情報を持ちます。これらの属性は、リソースのスキーママップページに一覧されます。属性の値は、オブジェクトのリスト (List) です。

次の表は、これらのオブジェクトが持つ属性を示しています。

表 5-9 accountInfo.accounts の属性 (ユーザービュー)

属性	説明
name	Identity Manager リソースアカウント属性の名前。この名前は、リソーススキーママップに定義されます。
syntax	属性値の構文。syntax 属性の値は、次のいずれかです。 • int • string • boolean • encrypted • binary • complex リソースで、バイナリ属性または複雑な属性がサポートされるかどうかを調べるときは、『Identity Manager リソースリファレンス』を参照してください。バイナリ属性や複雑な属性をサポートしないリソースに対して、このような属性を送信しようとする、例外がスローされます。 バイナリ属性は、できるだけ小さく維持してください。350K バイトを超えるバイナリ属性を管理しようとする、Identity Manager は例外をスローします。350K バイトを超える属性の管理が必要な場合は、カスタマサポートまでお問い合わせください。
multi	属性が複数の値をサポートする場合は、true となります。

フォームの設計時には、宣言されたリソースアカウントの属性タイプを気にする必要はありません。必要に応じて、ユーザービューのプロセッサシステムが、適切なタイプを強制します。

accountInfo.accounts[].passwordPolicy

リソースには、パスワードポリシーを割り当てることができます。属性にパスワードポリシーが割り当てられている場合、その属性の値には、そのポリシーに関する情報が含まれます。

次の表は、accountInfo.accounts[resname].passwordPolicy の属性を示しています。

表 5-10 `accountInfo.accounts[resname].passwordPolicy` の属性 (ユーザービュー)

属性	説明
<code>name</code>	ポリシーの名前。これは、 Identity Manager リポジトリ内の <code>policy</code> オブジェクトの名前に対応します。
<code>summary</code>	各ポリシー属性に関する情報などの、ポリシーの簡単な説明。
<code>attributes</code>	この属性の値は、各ポリシー属性の名前と値を持つ別のオブジェクトです。

ポリシー情報を表示するアプリケーションは、通常は概要テキストを表示しますが、各ポリシー属性の表示をより詳細に制御する必要がある場合は、属性マップを使用できます。

パスワードの変更と同期のためのインタフェースを持つフォームは、多くの場合、この情報を使用します。

`accountInfo.accounts[Lighthouse]`

この `accountInfo` リストの特別エントリは、**Identity Manager** のデフォルトパスワードポリシーに関する情報の保持に使用されます。リソースアカウントに関する情報以外にも、**Identity Manager** のパスワードとポリシーに関する情報を表示するため、これはパスワードフォームを表示するときに便利です。

この要素は、パススルー認証が使用されていない場合にのみ表示されます。リソースタイプは `Lighthouse` です。

`accountInfo` のリソース名リスト

`accountInfo` ビューには、リソース名のリストを持つ属性が含まれます。各リストは、特定の特性を持つリソースに対して繰り返しを行う場合に、フォームで `FieldLoop` 構成体と組み合わせて使用することを目的としています。

リソース名を持つことができる `accountInfo` 属性は次のとおりです。

- `assigned`
- `created`
- `fromRole`
- `private`
- `toCreate`
- `toDelete`

accountInfo.assigned

ユーザーに割り当てられるリソースを指定します。フォームの設計時には、この属性を呼び出すことで、ロールやアプリケーションを通じて割り当てられているリソースのリストや、ユーザーに直接割り当てられているリソースのリストを表示できます。

accountInfo.typeNames

割り当てられているすべてのリソースの、一意のタイプ名のリスト。このリストは、特定タイプのリソースが選択されていないフィールドを無効にする場合に、フォームの Disable 式で使用されます。

```
<Field name='HomeDirectory' prompt='Home Directory'>
  <Display class='Text' />
  <Disable>
    <not>
      <contains>
        <ref>accountInfo.typeNames</ref>
        <s>Solaris</s>
      </contains>
    </not>
  </Disable>
</Field>
```

これは、accountInfo.types[*].name というパスと同じ情報を返しますが、こちらのほうが効率的です。これは、Disable 式で使用する場合に重要な条件です。このリストには、共通リソースタイプを含めることができます。

Identity Manager の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「**タイプ**」列には、現在定義されているリソースのタイプ名が表示されます。「**新規リソース**」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountInfo.types

この属性は、現在割り当てられている各リソースタイプに関する情報を持ちます。この属性の値は、List (オブジェクト) です。

次の表は、各オブジェクトの属性を示しています。

表 5-11 accountInfo.types の属性 (ユーザービュー)

属性	説明
accounts	ユーザーに割り当てられている、このタイプの各アカウントの accountId のリスト
name	リソースタイプ名

たとえば、accountInfo.types[Unix].accounts というパスによって、UNIX アカウントの ID のリストを指定できます。

display 属性

display 属性は、ビューの処理に適用されるコンテキストに関連する情報を持ちます。ほとんどの属性は、フォームの対話的な処理でのみ有効です。

次の表は、頻繁に使用される display ビュー属性を示しています。

表 5-12 頻繁に使用される display 属性 (ユーザービュー)

属性	説明
eventType	create または update という読み取り専用の値によって、ユーザービューが作成要求または更新要求を処理しているかどうかを示します。

表 5-12 頻繁に使用される **display** 属性 (ユーザービュー) (続き)

属性	説明
session	認証された Identity Manager セッションへのハンドル。この属性は、Identity Manager の管理者インタフェースで対話的な編集セッションを実行している場合にのみ有効です。これは、Identity Manager リポジトリへのアクセスポイントを提供します。この属性の値は、<code>com.waveset.ui.FormUtil</code> クラス内のメソッドに渡すことができます。 <code>display.session</code> 属性は、フォームの処理が行われる可能性があります、次のような場合には無効です。 - 一括読み込みに含まれる場合 - バックグラウンド再プロビジョニングの実行中 - アクションまたは承認の同期がとれていない場合 推奨される方法は、この属性を Property 要素内、または Constraints 要素内のみで使用することです。ほとんどすべての既存のフォームでは、<code>display.session</code> 属性は Constraints 要素内のみで使用されます。
subject	Identity Manager ユーザーまたは管理者のクレデンシャルに関する情報を保持するオブジェクト。この情報は、ほとんどすべての場合に設定されますが、通常は、<code>display.session</code> が無効になっている場合に、バックグラウンドアクティビティで呼び出されるワークフローアプリケーションで使用されます。subject は、新規セッションの取得に使用できます。この場合は、リポジトリへのアクセスの取得に使用されます。
state	<code>_javax.servlet.http.HttpSession_</code> などの HTTP 要求に関連するオブジェクトへのハンドルを持つ <code>_com.waveset.ui.util.RequestState_</code> オブジェクトへのハンドル。

itemType のデフォルト動作

要求者が作業項目の所有者である場合に、ワークフローを作業項目に直接遷移させることができるのは、通常、**wizard** 項目タイプ (**itemType**) のみです。

itemType を次のように設定すると、ワークフローは作業項目に遷移しなくなり、その代わりに「承認」タブに表示されるようになります。

- approval
- custom
- itemType

デフォルト動作のオーバーライド

次のように、フォームのプロパティとして allowedWorkItemTransitions オプションを設定することで、ユーザービューでのデフォルトの動作をオーバーライドすることができます。

```
<Form .....>
  <Properties>
    <Property name='allowedWorkItemTransitions'>
      <list>
        <s>myCustomType</s>
      </list>
    </Property>
  </Properties>
```

deferred 属性

deferred 属性は、別のアカウントの属性値から値を取得する属性です。ビュー (および WSUser モデル) で **deferred** 属性を宣言すると、プロビジョニングエンジンは、アダプタを呼び出す前に、この置き換えをただちに実行します。

deferred 属性が、別のリソースの GUID 属性から値を取得する場合は、ソースアダプタはアクションを実行する必要がありません。しかし、ソース属性が GUID でない場合は、realCreate 操作の二次的な影響として、アダプタは

ResourceInfo.resultsAttributes マップ内の属性を返さなければなりません。アダプタが属性を返さない場合は、プロビジョニングエンジンはアカウントをフェッチして、値を取得します。これは、値を返すようにアダプタを修正するよりも非効率です。

deferred 属性を使用する状況

deferred 属性は、新しいアカウントを作成し、アカウント属性の値を、ソースアカウントが作成されるまで認識されない別のアカウントの属性の値から取得するように指定するときに使用します。一例を挙げれば、生成される一意の識別子の値を属性に設定する場合がこれに該当します。

deferred 属性の使用

deferred 属性は、主に次の 2 つの手順で定義されます。

1. アカウントは、必ず 2 番目のアカウントの作成前に、ソースリソースに作成します。これは、リソースと、ユーザーへのリソースグループの割り当ての両方を含む、順序が付けられたリソースグループを作成することで行われます。
2. 次の例のように、作成するアカウントのユーザービューに特別な属性を設定します。それぞれの deferred 属性は、ソースアカウントを指定するビュー属性と、ソース属性を指定するビュー属性の 2 つを必要とします。これは、次の形式のパスを使用して設定されます。

```
accounts[<resource>].deferredAttributes.<attname>.resource  
accounts[<resource>].deferredAttributes.<attname>.attribute
```

この <resource> は実際のリソース名に置き換えられ、<attname> は実際の属性名に置き換えられます。

たとえば、1) アカウントの作成時に uid 属性を生成する LDAP というリソースと、2) LDAP リソースの uid と同じ値を保持する directoryid という directoryid 属性を持つ HR というリソースの、2 つのリソースを作成する場合を考えてみましょう。

次のフォームフィールドは、必要なビュー属性を設定し、この関係を定義します。

```
<Field  
name='accounts[HR].deferredAttributes.directoryid.resource'  
  <Expansion><s>LDAP</s></Expansion>  
</Field>  
  
<Field name='accounts[HR].deferredAttributes.directoryid  
  <Expansion><s>uid</s></Expansion>  
</Field>
```


ユーザービューのデバッグ

ユーザービューをデバッグするときは、ビューの内容を新しいファイルにダンプすると便利な場合があります。ダンプファイルを作成するには、次の **Derivation** ステートメントをユーザービューに追加します。

```
<Field name='DumpView'>
  <Derivation>
    <invoke name='dumpFile'>
      <ref>form_inputs</ref>
      <s>c:/temp/view.xml</s>
    </invoke>
  </Derivation>
</Field>
```

この **Derivation** 式は、dumpFile メソッドを呼び出すことで、ユーザーフォームが初めて表示されたあとにファイルを生成します。form_inputs 変数は、このフォームで使用されるビューに自動的にバインドされます。

前述の例では、dumpFile メソッドへの **String** 引数は、ファイルシステムのパスです。このパスは、実際には c:/temp/view.xml という有効パスに置き換えられています。

アカウント関連ビュー

指定されたアカウント (またはアカウント属性) と相互に関連するユーザーの検索に使用されます。このビューは、アカウント調整プロセスの一部として使用されます。

このビューには、次の **root** 属性が含まれます。これらの属性の値は **GenericObjects** です。新しい ID は `<account_name>@<resource_name>` となります。

表 5-13 アカウント関連ビューの最上位属性

属性	説明
correlation	相互関係の設定方法に関する情報を持ちます
matches	相互関係の結果を持ちます

関連要求は、ビューの取得処理と表示更新要求の両方で実行されます。表示更新の場合は、ビューに指定されている要求が使用されます (ただし、ビュー ID の値が優先的に適用される `accountId` と `resource` を除く)。取得要求の場合は、ビュー属性と同じ名前のビューオプション (たとえば、`correlator`) を使用して、ビューから提供される要求部分を指定できます。

注	ビューオプションとして指定される場合は、 <code>accountAttributes</code> は <code>WSUser</code> (リソースアダプタメソッドから返される) または <code>GenericObject</code> として指定できます。

correlation

表 5-14 `correlation` の属性 (アカウント関連ビュー)

属性	編集の可能性	データ型	必要性
<code>accountId</code>	読み取り	<code>String</code>	あり
<code>accountGUID</code>	読み取り / 書き込み	<code>String</code>	なし (ただし、 <code>accountId</code> と <code>resource</code> がリソースを明確に識別できない場合を除く)
<code>resource</code>	読み取り	<code>String</code>	あり
<code>accountAttributes</code>	読み取り / 書き込み	<code>String</code>	

表 5-14 correlation の属性 (アカウント相関ビュー) (続き)

属性	編集の可能性	データ型	必要性
correlator	読み取り / 書き込み	String	なし
confirmer	読み取り / 書き込み	String	なし

accountId

相互に関連付けるアカウントの名前を指定します。これは、ビュー ID から自動的に取得されます。

accountGUID

相互に関連付けるアカウントの GUID を指定します。accountId と resource がリソースを明確に識別できない場合にのみ、必須属性となります。

resource

アカウントが存在するリソースの名前を指定します。この値は、ビュー ID から自動的に取得されます。

accountAttributes

アカウントの属性を指定します。指定されている場合、ビューアは、相関規則と確認規則に渡すときに、現在のアカウント属性をフェッチしません。その代わりに、これらの属性が渡されます。

correlator

使用する相関規則を指定します。指定しない場合は、リソースの調整ポリシーによって指定される相関規則が使用されます。NULL 値を指定した場合は、相関規則は使用されません。

confirmer

使用する確認規則を指定します。指定しない場合は、リソースの調整ポリシーによって指定される確認規則が使用されます。NULL 値を指定した場合は、確認規則は使用されません。

これらのリストは、ユーザーの概要属性を持つ GenericObject から構成されます。

表 5-15 confirmer の属性 (アカウント関連ビュー)

属性	編集の可能性	データ型
claimants	読み取り	List
correlated	読み取り	List
unconfirmed	読み取り	List

claimant

要求者 (claimant) をもう一方のリストにも表示させるために、関連アルゴリズムに関係なく計算された要求者のリスト。要求者の検出は、ビューオプションで ignoreClaimants を **true** に設定することで無効にできます。アカウントを明示的に参照する ResourceInfo を持つ場合、ユーザーはアカウントを要求します。

correlated

リソースアカウントと相互に関連付けられているユーザーのリスト。

unconfirmed

関連規則によって選択されたが、確認規則によって拒否されたユーザーのリスト。このリストは、ビューオプションで includeUnconfirmed が true に設定されている場合にのみ存在します。

管理者ロールビュー

ユーザーに管理者ロールを作成または更新するときに使用されます。管理者ロールを使用すると、組織の組み合わせごとに一意の機能セットを定義できます。機能と制御する組織は、直接割り当てだけでなく、ロールを通じて間接的に割り当てることもできます。

1 つまたは複数の管理者ロールを 1 人のユーザーに割り当てたり、1 人または複数のユーザーを同じ管理者ロールに割り当てたりすることができます。

表 5-16 管理者ロールビューの最上位属性

名前	編集の可能性	データ型	必要性
id	読み取り / 書き込み	String	なし
name	読み取り / 書き込み	String	あり
capabilities		List	あり
capabilitiesRule		String	あり
controlledOrganizations		List	あり
controlledOrganizationsRule		String	あり
controlledOrganizationsUserform		String	あり
controlledSubOrganizations		List (オブジェクト)	なし
memberObjectGroup		List	あり

id

Identity Manager 内の AdminRole オブジェクトを一意に識別します。システム生成の値。

name

管理者ロールの名前を指定します。

capabilities

この管理者ロールに割り当てられる機能名のリストを指定します。

capabilitiesRule

ゼロ個以上の割り当て機能名のリストを返す、評価対象となる規則の名前を指定します。

controlledOrganizations

関連付けられている機能が許可される組織名のリスト。

controlledOrganizationsRule

評価対象となる規則の名前を指定します。この規則は、割り当てられるゼロ個以上の制御対象組織名のリストを返します。

controlledOrganizationsUserform

この管理者ロールによる制御の対象となる組織の範囲内で、ユーザーの編集または作成に使用されるユーザーフォームを指定します。この管理者規則が適用されるユーザーに、このユーザーフォームが直接割り当てられていない場合に有効です。

controlledSubOrganizations

使用可能オブジェクトのサブセットが含まれるか、または除外される、制御対象組織のリスト。この属性の値は、controlledSubOrganization オブジェクトのリストから構成されます。各 ControlledOrganization オブジェクトのビュー属性は次のとおりです。

表 5-17 controlledSubOrganizations のビュー属性 (管理者ロールビュー)

属性	データ型	必要性
name	String (制御対象オブジェクトグループの名前)	
types	List (オブジェクト)	

types は、タイプ (たとえば、Resource、Role、および Policy) ごとに含まれるか、または除外されるオブジェクトのリストです。各オブジェクトタイプのビューは次のとおりです。

表 5-18 controlledSubOrganizations ビュー属性のオブジェクトタイプ (管理者ロールビュー)

属性	データ型	必要性
name	String	

表 5-18 **controlledSubOrganizations** ビュー属性のオブジェクトタイプ (管理者ロール
ビュー) (続き)

属性	データ型	必要性
include	List (オブジェクト)	
exclude	List (オブジェクト)	

name

オブジェクトタイプの名前を指定します。

include

含められる関連オブジェクトタイプのオブジェクト名のリスト。

exclude

除外される関連タイプのオブジェクト名のリスト。

memberObjectGroup

この管理者ロールがメンバーとして属す **ObjectGroup** のリスト。これは、この管理者ロールを使用できるオブジェクトグループ (組織) です。

ユーザーの秘密質問の回答変更ビュー

- 1 つまたは複数のログインインタフェースで使用する、既存ユーザーの秘密質問の回答を変更するときに使用されます。
- 2 つの上位属性があります。

表 5-19 ユーザーの秘密質問の回答変更ビューの属性

属性	編集の可能性	データ型	必要性
questions		List	
loginInterface		String	

questions

質問を表現します。含まれる属性は次のとおりです。

表 5-20 questions の属性 (ユーザーの秘密質問の回答変更ビュー)

属性	データ型	必要性
qid	String	
question	String	
answer	String	
answerObfuscated	ブール型	

qid

質問を一意に識別します。これは、その質問を、ポリシーに定義されている質問に関連付けるために使用されます。

question

ポリシーに定義される質問の文字列を指定します。

answer

指定する場合、qid の値と関連付けられる、ユーザーの質問の回答を指定します。

answerObfuscated

回答をそのまま表示するか、暗号化するかを指定します。

loginInterface

この質問が関連付けられるログインインタフェースを指定します。値は、各ログインインタフェースの、一意のメッセージカタログキーです。

含まれる属性は次のとおりです。

表 5-21 loginInterface の属性 (ユーザーの秘密質問の回答変更ビュー)

属性	データ型	必要性
name	String	
questionPolicy	String	
questionCount	String	

name

質問が関連付けられるログインインタフェースの名前を指定します。

次に有効な値を示します。

- UI_LOGIN_CONFIG_DISPLAY_NAME_ALL_INTERFACES
- UI_LOGIN_CONFIG_DISPLAY_NAME_ADMIN_INTERFACE
- UI_LOGIN_CONFIG_DISPLAY_NAME_CLI_INTERFACE
- UI_LOGIN_CONFIG_DISPLAY_NAME_DEFAULT_USER_INTERFACE
- UI_LOGIN_CONFIG_DISPLAY_NAME_IVR_INTERFACE
- UI_LOGIN_CONFIG_DISPLAY_NAME_QUESTION_INTERFACE
- UI_LOGIN_CONFIG_DISPLAY_NAME_USER_INTERFACE

questionPolicy

この質問が関連付けられるポリシーを指定します (たとえば、All、Random、Any、または RoundRobin)。

questionCount

questionPolicy 属性を Any または Random に設定した場合にのみ、設定します。

ユーザー機能変更ビュー

Identity Manager ユーザーの機能を変更するときに使用されます。

表 5-22 ユーザー機能変更ビューの属性

属性	編集の可能性	データ型	必要性
adminRoles		List [String]	
capabilities		List [String]	
controlledOrganizations		List [String]	

adminRoles

ユーザーに割り当てられている管理者ロールのリスト。

capabilities

このユーザーに割り当てられている機能のリスト。

controlledOrganizations

ユーザーが、割り当てられている能力によって制御する組織のリスト。

委任承認者ビュー

既存の承認者に、1人以上の Identity Manager ユーザーを委任承認者として割り当てるときは、このビューを使用します。これにより承認者は、指定された期間に限り、承認者になれないユーザーに対して、自身の承認機能を委任することができます。

次の最上位属性があります。

表 5-23 委任承認者ビューの属性

属性	編集の可能性	データ型	必要性
name		List [String]	あり
delegateApproversTo		List [String]	あり
delegateApproversSelected		List [String]	
delegateApproversStartDate		String	
delegateApproversEndDate		String	

name

承認を委任するユーザーを指定します。

delegateApproversTo

ユーザーが承認を委任している相手を指定します。有効な値には、`manager`、`selectedUsers`、または `delegateApproversRule` があります。

delegateApproversSelected

- `delegateApproversRule` の値が `selectedUsers` である場合は、選択したユーザーの名前リストを指定します。
- `delegateApproversTo` の値が `delegatedApproversRule` である場合は、選択した規則を指定します。
- `delegateApproversTo` の値が `manager` である場合は、この属性は値を持ちません。

delegateApproversStartDate

承認の委任を開始する日時を指定します。デフォルトでは、開始日時は指定した日付の午前 12:01 となります。

delegateApproversEndDate

承認の委任を終了する日時を指定します。デフォルトでは、終了日時は指定した日付の午後 11:59 となります。

プロビジョニング解除ビュー

プロビジョニング解除するリソースリストの表示と選択に使用されます。1つの最上位属性があります。

resourceAccounts

この属性は、次の属性を持ちます。

表 5-24 resourceAccounts の属性 (プロビジョニング解除ビュー)

名前	編集の可能性	データ型	必要性
id	読み取り / 書き込み	String	
selectAll	読み取り / 書き込み	ブール型	
unassignAll	読み取り / 書き込み	ブール型	
unlinkAll	読み取り / 書き込み	ブール型	
currentResourceAccounts	読み取り	List (オブジェクト)	
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

アカウントの一意の識別子を指定します。

selectAll

すべてのリソースを選択するかどうかを制御します。

unassignAll

プライベートリソースのユーザーのリストから、すべてのリソースを削除するかどうかを指定します。

unlinkAll

Identity Manager ユーザーから、すべてのリソースとのリンクを解除するかどうかを指定します。

tobeCreatedResourceAccounts

この Identity Manager ユーザーに割り当てられているが、まだ作成されていないアカウントを表します。まだ作成されていないアカウントでは、パスワードのロックを解除することはできません。

tobeDeletedResourceAccounts

すでに作成されているが、このユーザーには割り当てられなくなったアカウントを表します。削除されるアカウントでは、パスワードを変更することはできません。

3つのアカウントリストはすべて、各リソースのアカウントの状態を表現するオブジェクトを含み、ユーザーはアカウントを個別に選択できます。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。

すべてのアカウントリストには、リソース名によるインデックスが付けられます。

表 5-25 currentResourceAccounts の属性 (プロビジョニング解除ビュー)

名前	編集の可能性	データ型
selected	読み取り / 書き込み	ブール型
unassign	読み取り / 書き込み	ブール型
unlink	読み取り / 書き込み	ブール型
name	読み取り	String
type	読み取り	String
accountId	読み取り	String
exists	読み取り	ブール型
disabled	読み取り	ブール型
authenticator	読み取り	ブール型
directlyAssigned	読み取り	ブール型

selected

true に設定されている場合は、指定されたリソースで、関連付けられているアカウントがプロビジョニング解除されることを示します。選択されているアカウントが **Lighthouse** である場合は、すでに選択されている場合を除き、**Identity Manager** ユーザーとすべての関連リソースの割り当てが削除されます。ただし、関連付けられているリソースアカウントは削除されません。

unassign

true に設定されている場合は、ユーザーのプライベートリソースのリスト (たとえば、`waveset.resources`) から、指定されているリソースが削除されることを示します。

unlink

true に設定されている場合は、**Identity Manager** ユーザーから、指定されているリソースとのリンクが解除されることを示します (たとえば、関連 **ResourceInfo** オブジェクトの削除)。

注

selected または **unassign** が true に設定されている場合は、**unlink** も true に設定されていることになります。しかし、その反対は必ずしもそのように限定されません。**unlink** を true に設定し、**selected** と **unassign** を false に設定することは可能です。

name

リソースの名前を指定します。これは、**Identity Manager** リポジトリ内の **resource** オブジェクトの名前に対応します。

type

リソースのタイプ (たとえば、**Solaris**) を指定します。**Identity Manager** の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「**タイプ**」列には、現在定義されているリソースのタイプ名が表示されます。「**新規リソース**」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountId

リソースアカウントの ID を指定します。

exists

リソースにすでにアカウントが存在するかどうかを示します (`currentResourceAccounts` のみ)。

disabled

アカウントが現在有効であるか、無効であるかを示します (currentResourceAccount のみ)。

authenticator

アカウントが、ユーザーのログインが設定されているアカウントの 1 つであるかどうかを示します。

directlyAssigned

true に設定されている場合は、そのユーザーにアカウントが直接割り当てられていることを示します。値が false の場合は、ロールまたはアプリケーションを通じて、アカウントが間接的に割り当てられていることを示します。

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

無効化ビュー

Identity Manager ユーザーのアカウントを無効にするときに使用されます。このビューは、カスタムワークフローでよく使用されます。

resourceAccounts

このビューで属性にアクセスするときの、最上位属性を表します。

表 5-26 resourceAccounts の属性 (無効化ビュー)

名前	編集の可能性	データ型	必要性
id	読み取り	String	
selectAll	読み取り	ブール型	
currentResourcesAccount	読み取り	String	
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

ユーザーの Identity Manager ID を指定します。

selectAll

設定すると、Identity Manager アカウントを含むすべてのリソースアカウントが無効化されます。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。特定のリソースを有効に指定するときは、selected フィールドを使用します。

表 5-27 resourceAccounts.currentResourceAccounts の属性 (無効化ビュー)

名前	編集の可能性	データ型
name	読み取り	String

表 5-27 resourceAccounts.currentResourceAccounts の属性 (無効化ビュー)

名前	編集の可能性	データ型
type	読み取り	String
accountId	読み取り	String
exists	読み取り	ブール型
disabled	読み取り	ブール型
selected	読み取り / 書き込み	ブール型

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。
詳細については、この章の「フォームでのビューオプションの設定」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。
詳細については、この章の「フォームでのビューオプションの設定」を参照してください。

有効化ビュー

Identity Manager ユーザーのアカウントを有効にするときに使用されます。このビューは、カスタムワークフローでよく使用されます。

resourceAccounts

このビューで属性にアクセスするときの、最上位属性を表します。

表 5-28 resourceAccounts の属性 (有効化ビュー)

名前	編集の可能性	データ型	必要性
id	読み取り	String	
selectAll	読み取り	ブール型	
currentResourcesAccount	読み取り	String	
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

ユーザーの Identity Manager ID を指定します。

selectAll

設定すると、Identity Manager アカウントを含むすべてのリソースアカウントが有効化されます。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。特定のリソースを有効に指定するときは、selected フィールドを使用します。

表 5-29 resourceAccount.currentResourceAccounts の属性 (有効化ビュー)

名前	編集の可能性	データ型
name	読み取り	String

表 5-29 resourceAccount.currentResourceAccounts の属性 (有効化ビュー)

名前	編集の可能性	データ型
type	読み取り	String
accountId	読み取り	String
exists	読み取り	ブール型
disabled	読み取り	ブール型
selected	読み取り / 書き込み	ブール型

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。
詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。
詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

オブジェクト検索ビュー

カスタマイズが可能で、一般的な Identity Manager リポジトリ検索インタフェースを提供します。Identity Manager に定義されているオブジェクトタイプと適切な権限を持ち、非推奨とされていたり、内部使用に限定されていたりしていないオブジェクトを検索できます。オブジェクト検索ビューハンドラには、1 つまたは複数の属性クエリー条件とパラメータを指定したり、検索結果を表示したりするための関連フォームが用意されています。また、ビューオプションを使用して、属性クエリー条件とパラメータを指定できます。

このビューは、次の属性を持ちます。

表 5-30 最上位属性 (オブジェクト検索ビュー)

名前	編集の可能性	データ型	必要性
objectType	読み取り / 書き込み	String	あり
allowedAttrs	読み取り / 書き込み	List	なし
attrsToGet	読み取り / 書き込み	List	なし
attrConditions	読み取り / 書き込み	List	なし
maxResults	読み取り / 書き込み	String	なし
results	読み取り	List	なし
sortColumn	読み取り / 書き込み	String	なし
selectEnable	読み取り / 書き込み	ブール型	なし

objectType

検索する Identity Manager リポジトリオブジェクトのタイプ (たとえば、Role、User、または Resource) を指定します。

allowedAttrs

デフォルトでは `objectType` の `listQueryableAttributeAttrs()` メソッドを呼び出すことで取得される、クエリー可能な属性名のオブジェクトタイプ (`objectType` 属性によって指定) を指定するリスト。このメソッドは、**PersistentObject** を拡張する各クラスによって開示されます。オブジェクトタイプクラスによってオーバーライドされないときは、すべての **PersistentObject** でサポートされる、クエリー可能な属性のデフォルトセットを返す **PersistentObject** 実装を継承します。

デフォルトの設定は、`findObjectsDefaults.xml` 設定ファイルのデフォルトセクションまたは `objectType` に固有のセクションのいずれかに、`allowedAttrs` のセットを指定することでオーバーライドできます。このファイルは、サンプルディレクトリに格納されています。許可されるそれぞれの属性を、次のように `sample/findObjectsDefaults.xml` ファイルに指定します。

name

属性を指定します。

displayName

Identity Manager の管理者インタフェースに表示する属性名を指定します。指定しない場合は、この属性の値はデフォルト値である `name` と同じ値になります。

syntax

属性値のデータ型を指定します。サポートされる値は、`string`、`int`、`boolean` です。指定しない場合は、この値はデフォルト値である `string` になります。

multiValued

属性が複数の値をサポートするかどうかを指定します。`true` という値は、属性が複数の値をサポートすることを表します。指定しない場合は、この値はデフォルト値である `false` になります。この属性は、属性構文が `string` である場合にのみ適用されます。

allowedValuesType

属性の許可される値が **Identity Manager** タイプのインスタンス (たとえば、**Role**、**Resource**) である場合に、**Identity Manager** のタイプ名を指定します。指定しない場合は、この属性にはデフォルト値である `NULL` が適用されます。

`name` 属性が、**Identity Manager** によって定義される属性である場合は、`name` の指定のみが必須となります。`name` 属性が拡張属性である場合は、`name` の指定は必須となりますが、デフォルト値が適切である限り、その他の属性の指定は省略可能です。

許可される属性の設定例については、`sample/findObjectsDefaults.xml` を参照してください。

`allowedAttrs` のリストは、文字列のリスト、オブジェクトのリスト、または両者の組み合わせによって指定できます。

attrsToGet

指定された属性クエリ条件と一致する各オブジェクトとともに返される、指定されたオブジェクトタイプ (`objectType`) の概要属性名のリスト。オブジェクトタイプの、サポートされる概要属性のセットを取得するときは、そのオブジェクトタイプの `listSummaryAttributeAttrs()` メソッドを呼び出します。このメソッドは、`PersistentObject` を拡張する各クラスによって開示されます。`objectType` クラスによってオーバーライドされないときは、すべての `PersistentObject` でサポートされる、概要属性のデフォルトセットを返す `PersistentObject` 実装を継承します。

デフォルトは、`sample/findObjectsDefaults.xml` 設定ファイルのデフォルトセクションまたは `objectType` に固有のセクションのいずれかに、`resultColumnNames` のリストを指定することでオーバーライドできます。

attrConditions

指定された属性条件 (`attrConditions`) と一致する、指定されたオブジェクトタイプ (`objectType`) のオブジェクトの検索に使用される属性条件のリスト。リストを構成する各属性条件は、次のように指定します。

selectedAttr

許可される属性のリスト (`allowedAttrs`) から、いずれかの属性名を指定します。

selectedAttrRequired

(省略可能) この属性条件に合わせて、選択された属性 (`selectedAttr`) を変更できるかどうかを指定します。`true` という値は、選択された属性を、この属性条件に合わせて変更できないことを表し、この場合、属性条件のリストからその属性条件を削除することはできません。

defaultAttr

(省略可能) 許可される属性のリストがインタフェースに表示されるときに、デフォルトで選択される `allowedAttrs` の名前を指定します。

allowedOperators

選択された属性 (selectedAttr) に指定されている構文に基づいて許可される演算子のリスト。デフォルトでは、getAllowedOperators メソッドを呼び出し、選択された属性 (selectedAttr) の syntax 属性と multiValued 属性の値を渡すことで、このリストを取得できます。デフォルトは、**sample/findObjectsDefaults.xml** 設定ファイルのデフォルトセクションまたは objectType に固有のセクションのいずれかに、許可される演算子 (allowedOperators) のセットを指定することでオーバーライドできます。

selectedOperator

allowedOperators に指定されているリストから、いずれかの演算子の名前を指定します。

selectedOperatorRequired

(省略可能) この属性条件に合わせて、選択された演算子 (selectedOperator) を変更できるかどうかを指定します。true という値は、選択された演算子を、この属性条件に合わせて変更できないことを表し、この場合、属性条件のリストからその属性条件を削除することはできません。

defaultOperator

(省略可能) 許可される演算子 (allowedOperators) のリストがフォームに表示されるときに、デフォルトで選択される演算子 (allowedAttrs) の名前を指定します。

value

指定されたオブジェクトタイプ (objectType) のオブジェクトを返す必要があると Identity Manager が判断した場合にテストの対象となる、選択された属性名と演算子の値またはオペランドを指定します。selectedOperator の値が exists または notPresent である場合は、この属性を省略できます。

valueRequired

(省略可能) 属性条件の value を変更できるかどうかを指定します。値が true の場合は、値を変更できます。これは、属性条件のリストからその属性条件を削除できないことも意味します。

removeAttrCond

この属性条件の削除の必要性を指定します (内部)。

FindObjects.ATTR_CONDITIONS 定数または attrCondition 文字列を使用することで、属性条件をビューオプションとして指定できます。attrConditions を指定しない場合は、Identity Manager は指定されたオブジェクトタイプのすべてのオブジェクトを返します。

maxResults

(省略可能) 検索要求によって返される、指定された objectType のオブジェクトの最大数を指定します。指定しない場合のデフォルト値は 100 です。デフォルトは、sample/findObjectsDefaults.xml 設定ファイルのデフォルトセクションまたは objectType に固有のセクションのいずれかに、resultMaxRows 属性の値を指定することでオーバーライドできます。

この属性を使用することで、指定したタイプの Identity Manager リポジトリオブジェクトが多数存在する場合のパフォーマンスを改善できます。

results

attrsToGet の値が NULL の場合、result の値は、指定された属性条件と一致するオブジェクト名のリストとなります。attrsToGet の値が NULL 以外の場合は、results は、指定された attrConditions と一致するオブジェクトのリストとなります。各オブジェクトは、次の内容から構成されます。

- *columns* - 要求された attrsToGet と一致する、表示可能な列名のリスト
- *rows* - 0 から行数 (たとえば、10) までの番号が名前として付けられた row オブジェクトのリスト
- *row* - 0 から列数 (たとえば 6) までの番号と、それに該当する rows 列の値が名前として付けられたオブジェクトのリスト

sortColumn

(省略可能) 結果をソートする列の値を指定します。指定しない場合のデフォルト値は 0 です。デフォルトは、sample/findObjectsDefaults.xml 設定ファイルのデフォルトセクションまたは objectType に固有のセクションのいずれかに、resultSortColumn の値を指定することでオーバーライドできます。

selectEnable

(省略可能) 複数の結果行を同時に選択できるかどうかを指定します。値が `true` の場合は、複数の結果行を選択できます。デフォルトは `false` です。デフォルトは、`sample/findObjectsDefaults.xml` 設定ファイルのデフォルトセクションまたは `objectType` に固有のセクションのいずれかに、`resultSelectEnable` の値を指定することでオーバーライドできます。

組織ビュー

作成する組織のタイプと、それを処理するオプションのタイプの指定に使用されます。

代表的な属性

次の表は、このビューの上位属性を示しています。

表 5-31 組織ビューの属性			
名前	編集の可能性	データ型	必要性
orgName	読み取り	String	システム生成値
orgDisplayName	読み取り / 書き込み	String	あり
orgType	読み取り / 書き込み	String	なし
orgId	読み取り	String	システム生成値
orgAction	書き込み	String	なし
orgNewDisplayName	書き込み	String	なし
orgParentName	読み取り / 書き込み	String	なし
orgChildOrgNames	読み取り	List	システム生成値
orgApprovers	読み取り / 書き込み	List	なし
allowsOrgApprovers	読み取り	List	システム生成値
allowedOrgApproverIds	読み取り	List	システム生成値
orgUserForm	読み取り / 書き込み	String	なし
orgViewUserForm	読み取り / 書き込み	String	なし
orgPolicies	読み取り / 書き込み	List	なし
orgAuditPolicies	読み取り / 書き込み	List	なし
renameCreate	読み取り / 書き込み	String	なし
renameSaveAs	読み取り / 書き込み	String	なし

orgName

組織の UID を指定します。短い組織名は同じであっても、親組織が異なる場合があるため、この値はほとんどのビューオブジェクト名と異なります。

orgDisplayName

短い組織名を指定します。これは、表示のためにのみ使用される値であり、一意である必要はありません。

orgType

組織のタイプを指定します。指定できる値は、`junction` または `virtual` です。`junction` または `virtual` というタイプ以外の組織は、値を持ちません。

orgId

Identity Manager 内で組織を一意に識別するための ID を指定します。

orgAction

ディレクトリジャンクション、仮想組織、および動的組織のみでサポートされます。指定できる値は、`refresh` です。組織がディレクトリジャンクションまたは仮想組織である場合、表示更新の動作は、`orgRefreshAllOrgsUserMembers` の値によって異なります。

orgNewDisplayName

組織の名前を変更する場合の、新しい短い名前を指定します。

orgParentName

親組織の完全パス名を指定します。

orgChildOrgNames

直接および間接的なすべての子組織の Identity Manager インタフェース名のリスト。

orgApprovers

この組織に追加または変更されたユーザーの承認を担当する Identity Manager 管理者のリスト。

allowedOrgApprovers

この組織に追加または変更されたユーザーの承認者になりうる、ユーザーの名前のリスト。

allowedOrgApproverIds

この組織に追加または変更されたユーザーの承認者になりうる、ユーザーの ID のリスト。

orgUserForm

この組織に属すユーザーが、ユーザーの作成時または編集時に使用する userForm を指定します。

orgViewUserForm

この組織に属すユーザーが、ユーザーを表示するときに使用するビューユーザーフォームを指定します。

orgPolicies

この組織に属すすべてのユーザーに適用されるポリシーを指定します。これは、String 型のキーが付けられたオブジェクトのリストです。各ポリシーオブジェクトには、orgPolicies[<type>] というプレフィックスを持つ、次のビュー属性が含まれます。このプレフィックスの <type> は、ポリシーのタイプ (たとえば、Lighthouse アカウント) を表します。

- policyName -- 名前を指定します
- id -- ID を表します
- implementation -- このポリシーを実装するクラスを識別します

orgAuditPolicies

この組織に属すすべてのユーザーに適用される監査ポリシーを指定します。

renameCreate

true に設定すると、この組織を複製し、orgNewDisplayName の値を使用して新しい組織を作成します。

renameSaveAs

true に設定すると、orgNewDisplayName の値を使用して、この組織の名前を変更します。

ディレクトリジャンクションと仮想組織の属性

表 5-32 ディレクトリジャンクションと仮想組織の属性

名前	編集の可能性	データ型	必要性
orgContainerId	読み取り	String	システム生成値
orgContainerTypes	読み取り	List	システム生成値
orgContainers	読み取り	List	システム生成値
orgParentContainerId	読み取り	String	システム生成値
orgResource	読み取り / 書き込み	String	あり (ディレクトリジャンクションまたは仮想組織の場合)
orgResourceType	読み取り	String	システム生成値
orgResourceId	読み取り	String	システム生成値
orgRefreshAllOrgsUserMembers	書き込み	String	なし

orgContainerId

関連付けられている LDAP ディレクトリコンテナの **dn** (たとえば、**cn=foo,ou=bar,o=foobar.com**) を指定します。

orgContainerTypes

別のリソースオブジェクトを持つことができる、許可されるリソースオブジェクトタイプのリスト。

orgContainers

選択リストを表示するときに **Identity Manager** インタフェースが使用する、リソースのベースコンテナのリスト。

orgParentContainerId

関連付けられている親 LDAP ディレクトリコンテナの **dn** (たとえば、**ou=bar,o=foobar.com**) を指定します。

orgResource

ディレクトリジャンクションと仮想組織の同期に使用される Identity Manager リソースの名前 (たとえば、West Directory Server) を指定します。

orgResourceType

ディレクトリジャンクションと仮想組織の同期に使用される Identity Manager リソースのタイプ (たとえば、LDAP) を指定します。

orgResourceId

ディレクトリジャンクションと仮想組織の同期に使用される Identity Manager リソースの ID を指定します。

orgRefreshAllOrgsUserMembers

orgAction の値が refresh の場合に true に設定すると、Identity Manager 組織に属すユーザーと、選択された組織とそのすべての子組織のリソースコンテナに属すユーザーが同期されます。false に設定した場合は、リソースコンテナに属すユーザーは同期されず、リソースコンテナと、選択された組織とそのすべての子組織の Identity Manager 組織のみが同期されます。

動的組織の属性

表 5-33 動的組織の属性

名前	編集の可能性	データ型	必要性
orgUserMembersRule	読み取り / 書き込み	String	なし
orgUserMembersRuleCacheTimeout	読み取り / 書き込み	String	なし

orgUserMembersRule

ユーザーの所属を確認するために実行時に評価される、authType が UserMembersRule の規則を、名前または UID で指定します。

orgUserMembersCacheTimeout

orgUserMembersRule によって返されるユーザーがキャッシュされる場合に、タイムアウトとなるまでの時間をミリ秒単位で指定します。値 0 は、キャッシュなしを表します。

システム生成 ID に代わる組織パス名の使用

ワークフローでこのビューを呼び出すときは、システム生成の ID を使用するか、checkoutView の値として組織パスの式 (たとえば、top:us:central:texas) を指定することができます。

ワークフローの例

```
<Activity id="1" name="Refresh Organization">
  <Variable name="orgView"/>
  <Action name="Get Organization"
    Application="com.waveset.session.WorkflowServices">
    <Argument name="op" value="checkoutView"/>
    <Argument name="subject" value="#ID#Configurator"/>
    <Argument name="viewId" value="OrgViewer:top:us:central:texas"/>
  </Argument>
  <Return from="view" to="orgView"/>
</Action>
```


パスワードビュー

Identity Manager ユーザー、またはそのリソースアカウントのパスワードを変更するときに、管理者によって使用されます。

このビューには、1 つの最上位属性があります。

resourceAccounts

この属性は、次の属性を持ちます。

表 5-34 resourceAccounts の属性 (パスワードビュー)

属性	編集の可能性	データ型	必要性
id	読み取り / 書き込み	String	あり
selectAll	読み取り / 書き込み	ブール型	なし
currentResourceAccounts	読み取り	List (オブジェクト)	なし
tobeCreatedResourceAccounts	読み取り	List (オブジェクト)	なし
tobeDeletedResourceAccounts	読み取り	List (オブジェクト)	なし
password	読み取り / 書き込み	暗号化	あり
confirmPassword	読み取り / 書き込み	暗号化	あり (ビューを対話的に使用する場合)
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

パスワード変更の対象となる Identity Manager ユーザーのアカウント ID を指定します。通常はビューハンドラによって設定され、フォームから変更することはありません。

selectAll

すべてのパスワードを選択するかどうかを制御します。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。

tobeCreatedResourceAccounts

この Identity Manager ユーザーに割り当てられているが、まだ作成されていないアカウントを表します。まだ作成されていないアカウントでは、パスワードを変更することはできません。

tobeDeletedResourceAccounts

このユーザーに割り当てられてはいるが、まだ Identity Manager による管理の対象となっていないリソースのセット (たとえば、resinfo オブジェクトがまだ関連付けられていないリソース) を表します。削除されるアカウントでは、パスワードを変更することはできません。

3 つのアカウントリストはすべて、各リソースのアカウントの状態を表現するオブジェクトを含み、ユーザーはアカウントを個別に選択できます。

どちらのリソースアカウントリストにもリソース名によるインデックスが付けられ、このユーザーがアカウントを持つリソースを表現するオブジェクトを含みます。

表 5-35 tobeDeletedResourceAccounts の属性 (パスワードビュー)

属性	編集の可能性	データ型
selected	読み取り / 書き込み	ブール型
name	読み取り	String
type	読み取り	String
accountId	読み取り	String
exists	読み取り	ブール型 (currentResourceAccounts のみ)

表 5-35 tobeDeletedResourceAccounts の属性 (パスワードビュー) (続き)

属性	編集の可能性	データ型
disabled	読み取り	ブール型 (currentResourceAccounts のみ)
passwordPolicy	読み取り	Object
authenticator	読み取り	ブール型
changePasswordLocation	読み取り	String (currentResourceAccounts のみ)
expirePassword	読み取り / 書き込み	ブール型

password

Identity Manager アカウントまたはリソースアカウントに割り当てる新しいパスワードを指定します。

confirmPassword

password 属性に指定したパスワードを確認します。ビューを対話的に使用する場合は、フォームの「password」フィールドと「confirmPassword」フィールドに同じ値を入力してください。ワークフローでの使用のように、ビューをプログラマ的に使用する場合は、confirmPassword 属性は無視されます。このビューを対話的に使用している場合は、この属性を設定してください。

selected

指定されたリソースが、新しいパスワードを必要とすることを指定します。

name

リソースの名前を指定します。これは、Identity Manager リポジトリ内の resource オブジェクトの名前に対応します。

type

リソースのタイプ (たとえば、Solaris) を指定します。Identity Manager の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「タイプ」列には、現在定義されているリソースのタイプ名が表示されます。「新規リソース」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountId

このリソースにアカウントが作成されている場合は、そのアカウントの ID を指定します。

exists

リソースにすでにアカウントが存在するかどうかを示します。

disabled

アカウントが現在無効化されているかどうかを示します。

passwordPolicy

このリソースにパスワードポリシーが設定されている場合は、その説明を指定します。NULL も指定できます。これには、次の属性があります。

表 5-36 passwordPolicy の属性 (パスワードビュー)

属性	説明
name	String
summary	String

また、この属性には、宣言されている各ポリシー属性のビュー属性も含まれます。ビュー属性の名前 (**name**) は、ポリシーに定義されている名前と同じです。

概要文字列 (**summary**) には、事前に書式が設定された、ポリシー属性の説明が含まれます。

authenticator

true に設定した場合は、このリソースが Identity Manager のパススルー認証リソースとして機能することを示します。

changePasswordLocation

(省略可能) パスワードの変更が行われる場所 (たとえば、Active Directory のドメインコントローラの DNS 名) を指定します。このフィールドの値の形式は、リソースによって異なります。

expirePassword

変更の直後に、パスワードに有効期限切れのマークを付けるかどうかを制御するときは、NULL 以外のブール型の値を設定します。NULL に設定した場合は、パスワードが変更されたユーザーが、パスワードを変更したユーザーと異なる場合に、デフォルトとして、パスワードを有効期限切れにします。

tobeCreatedResourceAccounts

この Identity Manager ユーザーに割り当てられているが、まだ作成されていないアカウントを表します。まだ作成されていないアカウントでは、パスワードを変更することはできません。

tobeDeletedResourceAccounts

すでに作成されているが、このユーザーには割り当てられなくなったアカウントを表します。削除されるアカウントでは、パスワードを変更することはできません。

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

プロセスビュー

ワークフローやレポートなどのタスクの呼び出しに使用されます。呼び出されるタスクは、Identity Manager の TaskDefinition オブジェクトまたは TaskTemplate オブジェクトを使用して定義しておく必要があります。タスクを呼び出すと、TaskInstance オブジェクトが作成されます。

このビューには、task という 1 つの最上位属性があります。それ以外の最上位属性は、入力としてタスクに渡されますが、値の指定は任意です。

task

この最上位属性は、タスクを呼び出す方法を定義します。

表 5-37 プロセスビューの属性

属性	編集の可能性	データ型	必要性
process	読み取り / 書き込み	String	あり
taskName	読み取り / 書き込み	String	あり
organization	読み取り / 書き込み	String	あり
taskDisplay	読み取り / 書き込み	String	なし
description	読み取り / 書き込み	String	なし
execMode	読み取り / 書き込み	String	なし
result	読み取り / 書き込み	WavesetResult	なし
owner	読み取り / 書き込み	String	なし

process

呼び出すプロセスの名前。この属性には、Identity Manager の TaskDefinition オブジェクトまたは TaskTemplate オブジェクトの名前を指定できます。また、System Configuration オブジェクトの process の設定を利用してマップされる抽象プロセス名も指定できます。これは、必須属性です。

taskName

タスクの実行時状態を保持するために作成される、TaskInstance オブジェクトに付ける名前を指定します。この属性に値を指定しない場合は、生成されるランダムな名前が適用されます。

organization

`TaskInstance` の配置先となる組織の名前を指定します。この属性に値を指定しない場合、`TaskInstance` は最上位に配置されます。

taskDisplay

`TaskInstance` の表示名を指定します。

description

`TaskInstance` を説明する文字列を指定します。この文字列は、製品インタフェースの「サーバータスク」テーブルに表示されます。

execMode

実行モードを指定します。通常は、値を指定しません。この場合、実行モードは `TaskDefinition` によって決定されます。この属性に指定した設定は、`TaskDefinition` の値に優先して適用されます。

`execMode` の許可される値は次のとおりです。

表 5-38 `execMode` 属性の値 (プロセスビュー)

値	説明
<code>sync</code>	同期実行またはフォアグラウンド実行を指定します
<code>async</code>	非同期実行またはバックグラウンド実行を指定します
<code>asyncImmediate</code>	ただちにスレッドを開始する非同期実行を指定します

`asyncImmediate` 実行モードは、直列化できない値をビューを通じてタスクに渡さなければならない、特別なシステムタスクのみで使用してください。タスクスレッドはただちに開始されます。デフォルトの動作では、`TaskInstance` は一時的にリポジトリに保存され、後からスケジューラによって再開されます。

result

`TaskInstance` の初期結果を指定します。この設定を利用して、タスクの完了時にタスクの結果を最終的に表示するためのタスクに、情報を渡すことができます。

owner

タスクの所有者と見なされるユーザーの名前を指定します。値を設定しない場合は、ログインユーザーが所有者と見なされます。

ビューオプション

次のオプションは、createView メソッドと checkinView メソッドが認識するオプションです。

endUser

タスクが Identity Manager のユーザーインターフェースから呼び出されることを指定します。これにより、正式な権限を持たないユーザーも、特別に指定されたエンドユーザータスクを呼び出すことができます。

process

呼び出すプロセスの名前。この名前は、createView メソッドによって認識され、ビューの process 属性の値となります。

suppressExecuteMessage

true に設定すると、非同期タスクを呼び出したときに、タスクの結果に追加されるデフォルトのメッセージが抑制されます。デフォルトの英文テキストは、「The task is being executed in the background (タスクはバックグラウンドで実行中)」です。

checkinView メソッドの結果

checkinView メソッドから返される WavesetResult オブジェクトには、次の名前が付けられた結果項目が記録されます。

表 5-39 checkinView メソッドの結果

結果	説明
taskId	TaskInstance のリポジトリ ID を識別します
taskState	TaskInstance の現在の状態を識別します。値は、ready (準備完了)、executing (実行中)、suspended (保留)、または finished (完了) のいずれかです。
extendedResults	true が指定されている場合は、TaskInstance が拡張された結果を持つことを表します。

調整ビュー

リソースに対する調整処理の要求またはキャンセルに使用されます。このビューは、ワークフローの一部として、オンデマンド調整を行うときに使用されます。また、調整のためのカスタムスケジューラの実装にも使用されます。

このビューは書き込み専用です。取得やチェックアウトの操作はサポートされません。

request

実行する操作を指定します。次の有効な操作のいずれかを指定してください。

表 5-40 request 属性で有効な操作 (調整ビュー)

操作	説明
FULL	リソースの完全調整を開始します
INCREMENTAL	リソースの差分調整を開始します
ACCOUNT	アカウントの調整を開始します
CANCEL	現在アクティブなリソース調整プロセスをキャンセルします

accountId

調整するアカウントを指定します。要求が ACCOUNT 以外の場合、この文字列は無視されます。

例

- リソースの単一アカウントの調整を要求する方法は次のとおりです。ここでは、Active Directory リソースを例に取ります。

```
request = "ACCOUNT"
accountId = "cn=maurelius, ou=Austin, DC=Waveset, DC=com"
```
- 保留中、または現在アクティブな、リソースの調整プロセスをキャンセルします。

```
request = "CANCEL"
```

調整ポリシービュー

Identity Manager のシステム設定オブジェクトの一部として格納されている調整ポリシーを表示、変更するときに使用されます。

調整ポリシーと調整ポリシービュー

調整ポリシーの設定は、ツリー構造に格納されます。このツリーの一般的な構造は次のとおりです。

- デフォルトまたはグローバルポリシー (Default)。これは root ポリシーレベルです。
- リソースタイプ (ResType:) ポリシー
- リソースポリシー (Resource:)

設定は、ツリーの任意のポイントに指定できます。レベルがポリシーの値を指定しない場合は、その次に上位のポリシーから継承されます。

ビューは、ポリシーツリー上の指定のポイントで、有効なポリシーを表します。これは、ビュー名で識別されます。

表 5-41 調整ポリシーツリーとビュー名

ビュー名	説明
Default	ポリシーツリーの root を示します
ResType: リソースタイプ	root の下位の指定リソースタイプを示します
Resource: リソース名	リソースのリソースタイプの下位の指定リソースを示します

ポリシー値

ポリシー設定の値は、常にポリシー値となります。ポリシー値には、最大で、次の表に示される 3 つのコンポーネントを含めることができます。

表 5-42 ポリシー値の設定属性 (調整ポリシービュー)

ポリシー値の設定	説明
value	設定の値を指定します。

表 5-42 ポリシー値の設定属性 (調整ポリシービュー)

ポリシー値の設定	説明
scope	この設定の取得元となる範囲を指定します。範囲の値には、Local、ResType、および Default があります。これは、ポリシーを指定しているラベルを示します。たとえば、SCOPE_LOCAL という値は、現在のポリシーレベルに設定される値を表します。 SCOPE_LOCAL -- リソースレベルまたは現在のポリシーレベルにポリシーが設定されます SCOPE_RESTYPE -- restype (リソースタイプ) レベルにポリシーが設定されます SCOPE_GLOBAL. -- グローバルレベルにポリシーが設定されます
inheritance	このレベルで継承されたポリシー設定を識別します。scope が Local でない場合、inheritance は有効な値と一致します。Default レベルのポリシー設定には存在しません。

認証の必要性

ビューを変更するユーザーには、Reconcile Administrator 機能が必要です。

ビューにアクセスするユーザーには、Reconcile Administrator 機能または Reconcile Request Administrator 機能が必要です。

ビュー属性

次の表は、このビューの上位属性を示しています。

表 5-43 調整ポリシービューの属性

属性	説明
scheduling	調整の自動化されたスケジュールに関する情報を含みます。
correlation	リソースアカウントの所有を決定する方法に関する情報を含みます。

表 5-43 調整ポリシービューの属性

属性	説明
workflow	ユーザーが調整プロセスに指定する拡張に関する情報を含みます。
response	検出された状況に対して調整がどのように応答するかに関する情報を含みます。
resource	調整がリソースとどのように連携するかに関する情報を含みます。

scheduling

表 5-44 scheduling の属性 (調整ポリシービュー)

属性	編集の可能性	データ型
reconcileServer	読み取り / 書き込み	String
reconcileModes	読み取り / 書き込み	String
fullSchedule	読み取り / 書き込み	Schedule
incrementalSchedule	読み取り / 書き込み	Schedule
nextFull	読み取り	Date
nextIncremental	読み取り	Date

reconcileServer

スケジューリングされた調整の実行に使用される調整サーバーを指定します。

reconcileModes

有効にする調整モードを指定します。有効な値は、BOTH、FULL、NONE です。

fullSchedule

完全調整が有効な場合のスケジュールを指定します。

incrementalSchedule

差分調整が有効な場合のスケジュールを指定します。

nextFull

差分調整が有効な場合の、次の調整の日時を指定します。

nextIncremental

スケジュールの繰り返し回数を指定します。スケジュールの値は、次の属性を持つ `GenericObject` です。

- `count` -- スケジュールの繰り返し回数を指定します
- `units` -- スケジュールの繰り返し単位を指定します
- `time` -- スケジュールの開始時刻を指定します

correlation

相関規則の名前を指定します。

表 5-45 相関規則 (調整ポリシービュー)

属性	編集の可能性	データ型
<code>correlationRule</code>	読み取り / 書き込み	String
<code>confirmationRule</code>	読み取り / 書き込み	String

correlationRule

アカウントとユーザーを関連付けるときに使用される、相関規則の名前を指定します。

confirmationRule

関連付けられているユーザーをアカウントに対して確認するとき使用される、確認規則の名前を指定します。確認が必要ない場合は、`CONFIRMATION_RULE_NONE` という値を指定します。

workflow

表 5-46 workflow の属性 (調整ポリシービュー)

属性	編集の可能性	データ型
<code>proxyAdministrator</code>	読み取り / 書き込み	String
<code>preReconWorkflow</code>	読み取り / 書き込み	String
<code>perAccountWorkflow</code>	読み取り / 書き込み	String
<code>postReconWorkflow</code>	読み取り / 書き込み	String

proxyAdministrator

管理機能を持つユーザーの名前を指定します。

preReconWorkflow、*perAccountWorkflow*、*postReconWorkflow*

調整プロセスの適切なタイミングで実行するワークフローの名前を指定します。実行するワークフローを指定しないときは、AR_WORKFLOW_NONE という値を指定します。

response

表 5-47 response の属性 (調整ポリシービュー)

属性	編集の可能性	データ型
situations	読み取り / 書き込み	List
explanations	読み取り / 書き込み	ブール型

situations

指定された状況で実行する、自動化された応答を指定します。有効な応答は次のとおりです。

表 5-48 situations 属性のオプション (調整ポリシービュー)

応答	説明
DO_NOTHING	自動化された応答を実行しません
CREATE_NEW_USER	リソースアカウントに基づいて、新規ユーザーを作成します
LINK_ACCOUNT	要求ユーザーにアカウントを割り当てます
CREATE_ACCOUNT	リソースにアカウントを再作成します
DELETE_ACCOUNT	リソースからアカウントを削除します
DISABLE_ACCOUNT	リソースのアカウントを無効にします

explainActions

調整が、アクションの詳細な説明をアカウントインデックスに記録するかどうかを指定します。

resource

表 5-49 resource の属性 (調整ポリシービュー)

属性	編集の可能性	データ型
reconcileNativeChanges	読み取り / 書き込み	ブール型
reconciledAttributes	読み取り / 書き込み	List (String のリスト)
listTimeout	読み取り / 書き込み	Integer
fetchTimeout	読み取り / 書き込み	Integer

reconcileNativeChanges

アカウント属性へのネイティブ変更を調整の対象とするかどうかを指定します。

reconciledAttributes

ネイティブ変更の中で監視の対象となるアカウント属性のリストを指定します。

listTimeout

リソースに存在するアカウントを列挙するときに、調整が応答を待機する最大時間をミリ秒単位で指定します。

fetchTimeout

リソースからアカウントをフェッチするときに、調整プロセスが応答を待機する最大時間をミリ秒単位で指定します。

調整状態ビュー

最後に要求された調整処理の状態を取得するときに使用されます。このビューは読み取り専用です。

status

状態コード要求 (文字列) を示します。有効な状態コードは次のとおりです。

表 5-50 調整状態ビューの属性	
状態コード	説明
UNKNOWN	状態を特定できません。その他の属性の値は指定されません。
PENDING	要求は受信されましたが、まだ処理されていません。
RUNNING	要求は、現在処理中です。
COMPLETE	要求は完了しています。その他の要求の成功または失敗を特定するには、属性を確認します。
CANCELLED	要求は管理者によってキャンセルされました。

reconcileMode

要求の調整モードを示します。有効な値は、FULL または INCREMENTAL です。

reconciler

調整要求を処理する Identity Manager サーバーを指定します。

requestedAt

要求を受信した日付を示します。

startedAt

調整処理が開始された日付を示します。調整処理がまだ開始されていないか、または保留中にキャンセルされた場合、この値は NULL となります。

finishedAt

調整処理が完了した日付を示します。調整処理がまだ完了していない場合は、この値は NULL となります。

errors.fatal

調整処理の中止原因となったエラー (存在する場合) を説明します。エラーは、文字列のリストとして返されます。

errors.warnings

調整処理の実行中に発生した、致命的ではないエラーを説明します。エラーは、文字列のリストとして返されます。

statistics.accounts.discovered

調整処理の実行時にリソースで検出されたアカウントの数を示します。

statistics.situation[<situation>].resulting

応答処理の実行後に (成功、失敗を問わず)、指定された調整状態にあるアカウントの数を示します。

有効な状況は次のとおりです。

- CONFIRMED
- FOUND
- DELETED
- MISSING
- COLLISION
- UNMATCHED
- UNASSIGNED
- DISPUTED

ユーザー名変更ビュー

Identity Manager ユーザーとリソースアカウントアイデンティティの名前の変更に使用されます。このビューは、通常は企業内のユーザーの名前を変更するときに使用されます。また、それ以外にも、ディレクトリ構造での移動の要因となるディレクトリユーザーのアイデンティティを変更するときに、このビューが使用されます。

表 5-51 ユーザー名変更ビューの属性

名前	編集の可能性	データ型	必要性
newAccountId	読み取り / 書き込み	String	
toRename	読み取り	List	
noRename	読み取り	List	
resourceAccounts	読み取り		
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

newAccountId

Identity Manager ユーザーに設定され、リソースアカウントのアイデンティティテンプレートで使用される、新しい accountId を指定します。

toRename

名前変更処理をサポートする currentResourceAccounts リストに含まれるアカウントのリストを指定します。

noRename

名前変更機能をサポートしないアカウントのリストを指定します。

resourceAccounts

リソースアカウントに関する、主に読み取り専用の情報を持ちます。リソースアカウント名の変更に使用される属性は次のとおりです。

表 5-52 resourceAccounts の属性

属性	データ型	説明
selectAll	ブール型	すべてのアカウントの名前を変更するかどうかを制御します。
currentResourceAccounts [<resourcename>].selected	ブール型	このリソースアカウントのアイデンティティ名の変更に、新しい accountId を使用することを指定します。
currentResourceAccounts [Lighthouse].selected	ブール型	Identity Manager アカウントの名前を変更するかどうかを制御します。selectAll=true は、この設定に優先して適用されます。

accounts[<resourcename>].identity

このリソースアカウントの accountId の作成時に、アイデンティティテンプレートの使用をオーバーライドします。

accounts[<resourcename>].<attribute>

新規 accountId の作成時に、アイデンティティテンプレートに属性を渡す accounts[<resourcename>].identity 属性を指定しない場合に使用されます。

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

例

```
renameView.newAccountId="saurelius"
renameView.resourceAccounts.selectAll="false"
renameView.resourceAccounts.currentResourceAccounts[Lighthouse].
selected="true"
renameView.accounts[AD].identity="cn=saurelius,OU=Austin,DC=Wave
set,DC=com"
renameView.resourceAccounts.currentResourceAccounts[AD].selected
="true"
renameView.accounts[LDAP].identity="CN=saurelius,CN=Users,DC=us,
DC=com"
renameView.resourceAccounts.currentResourceAccounts[LDAP].select
ed="true"
renameView.accounts[NT].identity="Marcus Aurelius"
renameView.resourceAccounts.currentResourceAccounts[NT].selected
="true"
```

再プロビジョニングビュー

再プロビジョニングするリソースのリストの表示と選択に使用されます。このビューには、1つの最上位属性 (resourceAccounts) があります。

resourceAccounts

この属性は、次の属性を持ちます。

表 5-53 resourceAccounts の属性 (再プロビジョニングビュー)

名前	編集の可能性	データ型	必要性
id	読み取り	String	
selectAll	読み取り / 書き込み	ブール型	
currentResourceAccounts	読み取り	List (オブジェクト)	
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

アカウントの一意の識別子を指定します。

selectAll

すべてのリソースを選択するかどうかを制御します。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。

すべてのアカウントリストには、リソース名によるインデックスが付けられます。

表 5-54 currentResourceAccounts の属性 (再プロビジョニングビュー)

名前	編集の可能性	データ型
selected	読み取り / 書き込み	ブール型
name	読み取り	String

表 5-54 `currentResourceAccounts` の属性 (再プロビジョニングビュー) (続き)

名前	編集の可能性	データ型
<code>type</code>	読み取り	String
<code>accountId</code>	読み取り	String
<code>exists</code>	読み取り	ブール型
<code>disabled</code>	読み取り	ブール型
<code>authenticator</code>	読み取り	ブール型

selected

`true` に設定されている場合は、指定されたリソースで、関連付けられているアカウントが再プロビジョニングされることを示します。選択されているアカウントが **Lighthouse** である場合は、すでに選択されている場合を除き、**Identity Manager** ユーザーとすべての関連リソースの割り当てが再プロビジョニングされます。ただし、関連付けられているリソースアカウントは再プロビジョニングされません。

name

リソースの名前を指定します。これは、**Identity Manager** リポジトリ内の `resource` オブジェクトの名前に対応します。

type

リソースのタイプ (たとえば、**Solaris**) を指定します。**Identity Manager** の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「**タイプ**」列には、現在定義されているリソースのタイプ名が表示されます。「**新規リソース**」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountId

リソースアカウントの ID を指定します。

exists

リソースにすでにアカウントが存在するかどうかを示します (`currentResourceAccounts` のみ)。

disabled

アカウントが現在有効であるか、無効であるかを示します (`currentResourceAccount` のみ)。

authenticator

アカウントが、ユーザーのログインが設定されているアカウントの 1 つであるかどうかを示します。

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、割り当てられているすべてのリソースが使用されます。

詳細については、この章の「[フォームでのビューオプションの設定](#)」を参照してください。

ユーザーパスワードのリセットビュー

管理者がパスワードをランダム生成のパスワードにリセットするときに使用されます。
また、オプションとして、新しいパスワードをリソースアカウントに反映させます。

resourceAccounts

リソースアカウントの特性を定義します。この属性は、次の属性を持ちます。

表 5-55 resourceAccounts の属性 (ユーザーパスワードのリセットビュー)

属性	編集の可能性	データ型	必要性
id	読み取り	String	
selectAll	読み取り / 書き込み	ブール型	
currentResourceAccounts	読み取り	List (オブジェクト)	
tobeCreatedResourceAccounts	読み取り	List (オブジェクト)	
tobeDeletedResourceAccounts	読み取り	List (オブジェクト)	

id

パスワード変更の対象となる Identity Manager ユーザーのアカウント ID を指定します。

selectAll

すべてのパスワードを選択するかどうかを制御します。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。

tobeCreatedResourceAccounts

この Identity Manager ユーザーに割り当てられているが、まだ作成されていないアカウントを表します。まだ作成されていないアカウントでは、パスワードを変更することはできません。

tobeDeletedResourceAccounts

すでに作成されているが、このユーザーには割り当てられなくなったアカウントを表します。削除がスケジューリングされるアカウントでは、パスワードを変更することはできません。

アカウントリストの属性には、tobeDeletedResourceAccounts、tobeCreatedResourceAccounts、currentResourceAccounts の3つがあります。これらの属性は、次の表に示される属性を持ちます。これらの属性は、各リソースのアカウントの状態を表現し、ユーザーはアカウントを個別に選択できます。

表 5-56 tobeDeletedResourceAccounts の属性 (ユーザーパスワードのリセットビュー)

属性	編集の可能性	データ型	必要性
selected	読み取り / 書き込み	ブール型	
name	読み取り	String	
type	読み取り	String	
accountId	読み取り	String (currentResourceAccounts のみ)	
exists	読み取り	ブール型 (currentResourceAccounts のみ)	
disabled	読み取り	ブール型 (currentResourceAccounts のみ)	
passwordPolicy	読み取り	Object	
authenticator	読み取り	ブール型	
changePasswordLocation	読み取り	String	

selected

このアカウントのパスワードをリセットする場合は、true に設定します。

name

リソースの名前を指定します。これは、Identity Manager リポジトリ内の Resource オブジェクトの名前に対応します。

type

リソースのタイプ (たとえば、Solaris) を指定します。Identity Manager の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「**タイプ**」列には、現在定義されているリソースのタイプ名が表示されます。「**新規リソース**」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountId

このリソースにアカウントが作成されている場合は、そのアカウントの ID を指定します。

exists

リソースにすでにアカウントが存在するかどうかを示します。

disabled

アカウントが現在無効化されているかどうかを示します。

passwordPolicy

このリソースにパスワードポリシーが設定されている場合は、その説明を指定します。NULL も指定できます。これには、次の属性があります。

表 5-57 passwordPolicy の属性 (ユーザーパスワードのリセットビュー)

属性	データ型	編集の可能性	必要性
name	String		
summary	String		

また、この属性には、宣言されている各ポリシー属性のビュー属性も含まれます。ビュー属性の名前 (**name**) は、ポリシーの **WSAttribute** の名前と同じです。

概要文字列 (**summary**) には、事前に書式が設定された、ポリシー属性の説明が含まれます。

authenticator

true に設定した場合は、このリソースが Identity Manager のパススルー認証リソースとして機能することを示します。

changePasswordLocation

パスワードの変更が行われる場所 (たとえば、Active Directory のドメインコントローラの DNS 名) を指定します。このフィールドの値の形式は、リソースによって異なります。

リソースビュー

リソースの変更に使用されます。

表 5-58 リソースビューの属性

属性	編集の可能性	データ型	必要性
name	読み取り / 書き込み	String	あり
adapterClassName	読み取り / 書き込み	String	あり
typeString	読み取り / 書き込み	String	あり
typeDisplayString	読み取り / 書き込み	String	あり
startupType	読み取り / 書き込み	String	なし
organizations	読み取り / 書き込み	List (String)	あり
resourceAttributes	読み取り / 書き込み	List (String)	なし
displayName	読み取り	String	なし
type	読み取り	String	なし
multivalued	読み取り	String	なし
syncSource	読み取り / 書き込み	ブール型	なし
facets	読み取り	String	なし
description	読み取り	String	なし
noTrim	読み取り	String	なし
accountAttributes	読み取り / 書き込み	List (String)	なし
identityTemplate	読み取り / 書き込み	String	なし
approvers	読み取り / 書き込み	List (String)	なし
allowedApprovers	読み取り	List (String)	なし
allowedApproversIds	読み取り	List (String)	なし

表 5-58 リソースビューの属性 (続き)

属性	編集の可能性	データ型	必要性
passwordPolicy	読み取り / 書き込み	String	なし
accountPolicy	読み取り / 書き込み	String	なし
resourcePasswordPolicy	読み取り / 書き込み	String	なし
respolExcludeAccountsRule	読み取り / 書き込み	String	なし
retryMax	読み取り / 書き込み	Integer	なし
retryDelay	読み取り / 書き込み	Integer	なし
retryEmail	読み取り / 書き込み	String	なし
retryEmailThreshold	読み取り / 書き込み	Integer	なし
form	読み取り	String	なし
licensedProducts	読み取り	List (String)	なし
available.MetaViewAttribute	読み取り	List (String)	なし
available.extendedAttributes	読み取り	List (String)	なし
available.formFieldNames	読み取り	List (String)	なし
host			
password			
TCPPort			
user			

リソースビューアは、各種ビューメソッドのリソースパラメータを次のようにインスタンス化します。

- createView メソッドは、リソースタイプに適した **prototypeXML** の特定に使用される、**typeString** オプションを必要とします。**prototypeXML** には、リソースパラメータの初期セットと、その初期値が含まれます。そのため、ビューには、この初期リソースパラメータのリストとそのデフォルト値が取り込まれます。
- getView メソッドと checkoutView メソッドは、リソースオブジェクトに存在するリソースパラメータのみを返します。実際のリソースオブジェクトにいずれかのリソースパラメータが不足している場合は、このリストへのデータの取り込みに **prototypeXML** は使用されません。

- リポジトリに格納されているリソースオブジェクトのリソースパラメータのリストは、checkInView メソッドによって置き換えられます。この場合も、checkInView 処理の実行中に提供されなかった不足リソースパラメータへのデータの取り込みに、prototypeXML は使用されません。

リソースパラメータ

リソースパラメータは、設定されているリソースアダプタの種類によって異なります。各リソースには、リソースビューアがリソースパラメータのセットとそのデフォルト値の特定に使用する prototypeXML 文字列が含まれます。Identity Manager によってリソースオブジェクトが作成されると、リソースビューアは prototypeXML 文字列を使用しなくなり、代わりに実際のオブジェクトからのリソースパラメータを使用します。

次の属性は、リソースオブジェクトを一意に識別します。

name

リソースを外部的に識別します。ユーザーが指定するこの名前は、リソースオブジェクト間で重複しません。

adapterClassName

リソースへのプロビジョニングで使用するリソースアダプタクラスを指定します。

type

リソースのデータタイプを指定します。

typeString

リソースタイプの内部名を指定します。

typeDisplayString

リソースタイプの表示名を指定します。これは、メッセージカタログに含まれるメッセージキーまたは ID です。

syncSource

true に設定すると、リソースが同期化イベントをサポートすることを表します。

facets

description

リソースを説明するテキストを指定します。

startupType

activeSync リソースの起動を自動で行うか、手動で行うかを指定します。

追加属性

追加属性は、設定されているアダプタの種類によって異なります。これらの属性は、少なくともリソースとの接続方法を指定します。代表的なパラメータには、TCP Port、user、password があります。

host

ホストを一意に識別します。

password

接続に適用するユーザー (ホスト管理者) のパスワードを指定します。

TCPPort

ホスト上の接続先ポートを指定します。

user

接続に適用するユーザー (ホスト管理者) を指定します。

accountAttributes

このリソースで管理の対象となるアカウントを指定します。属性は、リソースタイプによって異なります。

一般的な属性は次のとおりです。

accountId

リソースがこのアカウントを識別するための ID を指定します。

roles

アカウントがリソース上に持つロールを指定します。

identityTemplate

このリソースでのユーザーのアイデンティティの生成に使用される、アイデンティティテンプレートを指定します。

Identity Manager パラメータ

Identity Manager パラメータは、Identity Manager がリソースの管理を支援するときに使用されます。

resourceName

Identity Manager がこのリソースオブジェクトを識別するための名前を指定します。

displayName

Identity Manager のユーザー編集ページとパスワードページに表示される、ユーザーを識別するための表示名を指定します。

retryMax

リソース上のオブジェクトの管理時に発生するエラーに対して、再試行を行う最大回数を指定します。

retryDelay

再試行の間隔を秒単位の時間で指定します。

retryEmail

通知の再試行しきい値に達した場合に送信される通知の、送信先電子メールアドレスを指定します。

retryEmailThreshold

電子メールの送信後の再試行回数を指定します。

form

リソース上のアカウントを編集するワークフローで使用されるユーザーフォームを指定します。

passwordPolicy

このリソースのアカウントに適用されるパスワードポリシーを指定します。

resourcePasswordPolicy

このリソースのリソースアカウントに適用されるリソースパスワードポリシーを指定します。

accountPolicy

このリソースのアカウント ID に適用されるポリシーを指定します。

excludedResourceAccountsPolicy

アカウントリストからリソースアカウントを除外するためのポリシーを指定します。

available

表 5-59 リソースビューの属性

available の属性	説明
available.organizations	
available.formFieldNames	
available.metaViewAttributes	
available.extendedAttributes	

approvers

このリソースを承認する管理者のリスト。

allowedApprovers

allowedApproversIds

organizations

リソースで使用できる組織のリスト。

licensedProducts

リソースオブジェクトビュー

リソースオブジェクトの変更に使用されます。

更新する属性レベルの変更の計算に使用される

<resourceobjectType>.oldAttributes を除くすべての属性は、編集可能です。

実際には、<resourceobjectType> を、リソースに固有のオブジェクトタイプの小文字の名前に置き換えます (たとえば、group、organizationalunit、organization、または role)。

表 5-60 リソースオブジェクトビューの属性

属性	編集の可能性	データ型	必要性
resourceType	読み取り / 書き込み	String	
resourceName	読み取り / 書き込み	String	
resourceId	読み取り / 書き込み	String	
objectType	読み取り / 書き込み	String	
objectName	読み取り / 書き込み	String	
objectId	読み取り / 書き込み	String	
requestor	読み取り / 書き込み	String	
attributes	読み取り / 書き込み	Object	
oldAttributes	読み取り	Object	
organization	読み取り / 書き込み	String	
attrstoget	読み取り / 書き込み	List	
searchContext	読み取り / 書き込み	Object	
searchAttributes	読み取り / 書き込み	List	

<resourceobjectType>.resourceType

Identity Manager のリソースタイプ名のリスト (たとえば、LDAP、Active Directory)。

<resourceobjectType>.resourceName

Identity Manager のリソース名のリスト。

<resourceobjectType>.resourceId

Identity Manager のリソース ID または名前のリスト。

<resourceobjectType>.objectType

リソースに固有のオブジェクトタイプ (たとえば、Group) を指定します。

<resourceobjectType>.objectName

リソースオブジェクト名のリスト。

<resourceobjectType>.objectId

リソースオブジェクト (たとえば、dn) の完全修飾名を指定します。

<resourceobjectType>.requestor

ビューを要求しているユーザーの ID を指定します。

<resourceobjectType>.attributes

新しい、または更新されたリソースオブジェクト属性の名前と値のペア (オブジェクト) を指定します。この属性は、次の下位属性を持ちます。

resourceattrname -- 指定されたリソース属性の値を取得または設定するための文字列 (たとえば、<objectType>.attributes.cn、この cn はリソース属性 common name)。

<resourceobjectType>.oldAttributes

フェッチされたリソースオブジェクト属性の名前と値のペア (オブジェクト) を指定します。この値を編集することはできません。ビューはこの属性を使用して、更新の属性レベルの変更を計算します。

<resourceobjectType>.organization

リソースがメンバーとして属す組織のリストを指定します。このリストは、将来の分析やレポート生成のために用意されている関連監査イベントレコードにアクセスできる組織の特定に使用されます。

<resourceobjectType>.attrstoget

checkoutView メソッドまたは getView メソッドを使用してオブジェクトを要求したときに返される、オブジェクトタイプに固有の属性のリスト。

<resourceobjectType>.searchContext

階層的なネームスペースを使用して、リソース内の不完全な修飾名を検索するためのコンテキストを指定します。

<resourceobjectType>.searchAttributes

階層的なネームスペースを使用して、指定された searchContext 内でリソース名を検索するときに使用される、リソースオブジェクトタイプに固有の属性名のリスト。

<resourceobjectType>.searchTimelimit

フォームに入力された名前を検索する最大時間 (リソースがサポートする場合) を指定します。

ロールビュー

Identity Manager のロールオブジェクトの定義に使用されます。

このビューがチェックインされると、管理ロールワークフローが呼び出されます。デフォルトでは、このワークフローは単にビューの変更をリポジトリに適用するだけです。承認や、その他のカスタマイズのきっかけを提供することもできます。

次の表は、このビューの上位属性を示しています。

表 5-61 ロールビューの属性

属性	編集の可能性	データ型	必要性
name	読み取り / 書き込み	String	あり
resources	読み取り / 書き込み	List	なし
applications	読み取り / 書き込み	List	なし
roles	読み取り / 書き込み	List	なし
assignedResources	読み取り / 書き込み	List	なし
notifications	読み取り / 書き込み	List	なし
approvers	読み取り / 書き込み	List	なし
properties	読み取り / 書き込み	List	
organizations	読み取り / 書き込み	List	あり

name

ロールの名前を指定します。これは、Identity Manager リポジトリ内の Role オブジェクトの名前に対応します。

resources

ローカルに割り当てられたリソースの名前を指定します。

applications

ローカルに割り当てられたアプリケーションの名前 (リソースグループ) を指定します。

roles

ローカルに割り当てられたロールの名前を指定します。

assignedResources

リソース、アプリケーション、およびロールによって割り当てられたすべてのリソースの、フラット化されたリスト。

表 5-62 assignedResource の属性 (ロールビュー)

属性	編集の可能性	データ型
resourceName		String
name		String
attributes		Object

resourceName

割り当てられているリソースの名前を指定します。

name

リソースの名前または ID (可能な限り ID) を指定します。

attributes

リソースの特性を指定します。すべての下位属性は文字列であり、編集可能です。

表 5-63 attribute 属性のオプション (ロールビュー)

属性	説明
name	リソース属性の名前
valueType	この属性に設定される値のタイプ。Rule、text、none などの値を指定できます。
requirement	この属性によって設定される値のタイプ。指定できる値には、Rule (規則)、Text (テキスト)、None (なし)、Value (値)、Merge with Value (値とマージ)、Remove from Value (値から削除)、Merge with Value clear existing (値とマージ、既存の値をクリア)、Authoritative set to value (強制的に値を設定)、Authoritative merge with value (強制的に値とマージ)、Authoritative merge with value clear existing (強制的に値とマージ、既存の値をクリア) があります。
rule	値のタイプが Rule である場合に、規則名を指定します。
value	値のタイプが Text である場合に、値を指定します。

notifications

ユーザーへのこのロールの割り当てを承認する管理者の名前のリスト。

approvers

ユーザーへのこのロールの割り当てを承認する承認者の名前を指定します。

properties

このロールに格納される、ユーザー定義のプロパティを指定します。

organizations

このロールがメンバーとして属す組織のリスト。

タスクスケジュールビュー

TaskSchedule オブジェクトの作成と変更に使用されます。

このビューは、次の属性を持ちます。

表 5-64 タスクスケジュールビューの属性			
名前	編集の可能性	データ型	必要性
scheduler	読み取り / 書き込み	String	
task	読み取り / 書き込み	ブール型	

scheduler

スケジューリングされているすべてのタスクに共通する、スケジューラ自体に関連する属性を持ちます。次の属性があります。

表 5-65 scheduler の属性 (タスクスケジュールビュー)			
名前	編集の可能性	データ型	必要性
name	読み取り / 書き込み	String	なし
id	読み取り	String	なし
definition	読み取り / 書き込み	String	なし
template	読み取り / 書き込み	String	なし
taskOrganization	読み取り / 書き込み	String	なし
taskName	読み取り / 書き込み	String	なし
description	読み取り / 書き込み	String	なし
disabled	読み取り / 書き込み	ブール型	なし
skipMissed	読み取り / 書き込み	ブール型	なし
start	読み取り / 書き込み	Date	なし
repeatCount	読み取り / 書き込み	Int	なし
repeatUnit	読み取り / 書き込み	String	なし
resultOption	読み取り / 書き込み	String	なし

表 5-65 scheduler の属性 (タスクスケジュールビュー) (続き)

名前	編集の可能性	データ型	必要性
allowMultiple	読み取り / 書き込み	ブール型	なし

注 通常は、`scheduler.definition` または `scheduler.template` を値として指定します。いずれの値も指定しない場合は、後から編集して定義またはテンプレートを指定できる **TaskSchedule** オブジェクトが作成されます。

name

既存の **TaskSchedule** オブジェクトの名前、または新しい **TaskSchedule** オブジェクトの適切な名前を指定します。これは必須属性ではありませんが、指定しない場合は、システムによってランダムな識別子が生成されます。

id

既存の **TaskSchedule** オブジェクトを一意に識別します。

definition

スケジューリングする **TaskDefinition** オブジェクトの名前を指定します。

template

スケジューリングする **TaskTemplate** オブジェクトの名前を指定します。definition と template の両方を指定した場合は、template が優先されます。

taskOrganization

スケジュールタスクの呼び出し時に **TaskInstance** が配置される組織の名前を指定します。

taskName

スケジュールタスクの呼び出し時に作成される **TaskInstance** の名前を指定します。

description

スケジュールタスクの呼び出し時に作成される **TaskInstance** に保存される、説明テキストを指定します。この説明は、製品インタフェースのタスクテーブルに表示されます。

disabled

タスクスケジューラが **TaskSchedule** オブジェクトを処理するかどうかを制御します。**disable** 属性が **true** に設定された **TaskSchedule** は、スケジューラによって無視されます。この属性を使用することで、**TaskSchedule** オブジェクトを削除して再作成することなく、スケジュールタスクの実行を一時的に停止できます。

start

タスクを呼び出す日時を指定します。

repeatCount

repeatUnit と組み合わせて、タスクの実行頻度を決定します。**repeatCount** をゼロに設定するか、または設定しない場合、スケジューリングされたタスクの実行は1回のみです。**repeatCount** に正の値を指定した場合は、タスクは **repeatUnit** に指定された間隔で複数回実行されます。

repeatUnit

repeatCount に正の値が指定されたタスクの実行間隔を指定します。有効な値は、**second**、**minute**、**hour**、**day**、**week**、**month** です。たとえば、1年間を通じて週に1回実行するようにタスクをスケジューリングする場合は、**repeatUnit** を **week** に、**repeatCount** を **52** に設定し、タスクの実行を開始する日付を **start** に設定します。

resultOption

スケジューリングされたタスクの実行時に、指定された名前の **TaskInstance** がすでに存在する場合のスケジューラの反応を指定します。有効な値は、**wait**、**delete**、**rename**、および **terminate** です。

wait

スケジューラがタスクを再実行するか、次の繰り返しを待機するかを指定します。この属性が有効となるのは、**repeatCount** と **repeatUnit** が設定されている場合のみです。

delete

既存の **TaskInstance** の実行が完了している場合に、スケジューラはそのインスタンスを削除します。

rename

既存の `TaskInstance` の実行が完了している場合に、スケジューラはそのインスタンスの名前を変更します。

skipMissed

スケジューリングされた時刻にタスクを実行できなかった場合に、それを回復するためにタスクをただちに試行するか (`false`)、スケジューリングされている次の時刻まで単に待機するか (`true`) を指定します。

`false` に設定した場合は、**Identity Manager** はスケジューリングされた時刻に実行できなかったタスクの実行をただちに試行します。`true` に設定した場合は、**Identity Manager** はスケジューリングされている次の時刻まで待機します。デフォルトは `false` です。

terminate

`delete` に似ていますが、既存のタスクが現在も実行中の場合は、そのタスクを終了します。

allowMultiple

同じタスク定義またはタスクテンプレートの、複数のインスタンスの実行を許可するかどうかを制御します。`true` (デフォルト) に設定した場合は、スケジューラはタスクの新しいインスタンスを常に作成します。`false` に設定した場合は、すでに実行中のインスタンスが存在するとき、スケジューラは新しいインスタンスを作成しません。

task

タスクに固有の属性を持ちます。各タスクは独自の属性を定義し、タスクのフォームは、`task` ネームスペースに相対的な属性としてそれらを参照します。

ロック解除ビュー

アカウントのネイティブロックをサポートするリソースのアカウントのロックを解除するときに使用されます。このビューは、ロック解除するリソースアカウントのリストの表示と選択に使用されます。

注 リソースがアカウントのネイティブロックをサポートするアカウントでは、無効化ビューの代わりにロック解除ビューを使用します。

次の上位属性を持ちます。

表 5-66 ロック解除ビューの属性

名前	編集の可能性	データ型	必要性
id	読み取り	String	あり
selectAll	読み取り / 書き込み	ブール型	なし
currentResourceAccounts	読み取り	List (オブジェクト)	なし
tobeCreatedResourceAccounts	読み取り	List (オブジェクト)	なし
tobeDeletedResourceAccounts	読み取り	List (オブジェクト)	なし
fetchAccounts	読み取り / 書き込み	ブール型	
fetchAccountResources	読み取り / 書き込み	List	

id

パスワードのロック解除の対象となる Identity Manager ユーザーのアカウント ID を指定します。

selectAll

すべてのパスワードのロックを解除するかどうかを制御します。

currentResourceAccounts

Identity Manager によって現在管理されているアカウントのセットを表します (Identity Manager アカウント自体を含む)。

tobeCreatedResourceAccounts

この Identity Manager ユーザーに割り当てられているが、まだ作成されていないアカウントを表します。まだ作成されていないアカウントでは、パスワードのロックを解除することはできません。

tobeDeletedResourceAccounts

すでに作成されているが、このユーザーには割り当てられなくなったアカウントを表します。削除されるアカウントでは、パスワードを変更することはできません。

3つのアカウントリストはすべて、各リソースのアカウントの状態を表現するオブジェクトを含み、ユーザーはアカウントを個別に選択できます。

どちらのリソースアカウントリストにもリソース名によるインデックスが付けられ、このユーザーがアカウントを持つリソースを表現するオブジェクトを含みます。

表 5-67 tobeDeletedResourceAccounts の属性 (ロック解除ビュー)

名前	編集の可能性	データ型
selected	読み取り / 書き込み	ブール型
name	読み取り / 書き込み	String
type	読み取り / 書き込み	String
accountId	読み取り / 書き込み	String
exists	読み取り / 書き込み	ブール型
locked	読み取り / 書き込み	ブール型
authenticator	読み取り / 書き込み	ブール型

selected

ロック解除の対象として、このリソースが選択されていることを示します。

name

リソースの名前を指定します。これは、Identity Manager リポジトリ内の resource オブジェクトの名前に対応します。

type

リソースのタイプ (たとえば, **Solaris**) を指定します。Identity Manager の管理者インタフェースでこのリソースリストを表示することで、リソースタイプの名前を確認できます。このページの「**タイプ**」列には、現在定義されているリソースのタイプ名が表示されます。「**新規リソース**」の横のオプションリストにも、現在インストールされているリソースアダプタの名前が表示されます。

accountId

このリソースにアカウントが作成されている場合は、そのアカウントの ID を指定します。

exists

リソースにすでにアカウントが存在するかどうかを示します
(currentResourceAccounts のみ)。

locked

アカウントが現在ロックされているか、ロック解除されているかを示します。exists の値は、リソースにすでにアカウントが存在するかどうかを示します
(currentResourceAccounts のみ)。

authenticator

true に設定した場合は、このリソースが Identity Manager のパススルー認証リソースとして機能することを示します。

fetchAccounts

ビューに、ユーザーに割り当てられているリソースのアカウント属性を含めます。

詳細については、この章の「フォームでのビューオプションの設定」を参照してください。

fetchAccountResources

フェッチ先リソースの名前のリスト。指定しない場合は、すべてのリソースが使用されます。

詳細については、この章の「フォームでのビューオプションの設定」を参照してください。

作業項目ビュー

リポジトリ内の **WorkItem** オブジェクトの表示と変更に使われます。

WorkItem オブジェクトは、ワークフロープロセスに定義されている手動アクションが作動するたびに作成されます。作業項目ビューには、**WorkItem** オブジェクト自体を表現するいくつかの属性と、ワークフロータスクからコピーされた、選択されたワークフロー変数の値が含まれます。

Identity Manager は、`workItem.related` 属性の下の作業項目ビューに含まれる、作業項目に関する情報を返します。

すべてのアクティブ作業項目に関する情報の取得

このビューでは、ワークフロータスクで現在アクティブなすべての作業項目に関する情報を取得できます。デフォルトでは、関連するすべての作業項目ではなく、指定された作業項目のみにに関する情報が返されます。ただし、別のオプションを使用して、表示する作業項目をフィルタリングしたり、作業項目に関連する属性をフィルタリングしたりすることができます。

このビューのデフォルトの動作を変更するには、次の3つのフォームプロパティーを使用します。

表 5-68

実行する処理	使用するフォームプロパティー
デフォルトで関連するすべての項目を取得する	<code>includeRelatedItems</code> フォームプロパティー
取得する追加属性を要求する	<code>relatedItemAttributes</code> フォームプロパティー
取得する項目を制限する	<code>relatedItemFilter</code> フォームプロパティー

例 : `includeRelatedItems` フォームプロパティーの使用

デフォルトでは、**Identity Manager** は作業項目の表示に承認フォームを使用します。このフォームを編集し、関連作業項目を含めるための `includeRelatedItems` 要素を追加します。

```
<Properties>

  <Property name='includeRelatedItems' value='true'/>
</Properties>
```

例 : relatedItemAttributes フォームプロパティの使用

relatedItemAttributes オプションを使用して、追加属性を要求することもできます。このオプションには、名前の CSV 文字列、または名前のリストを指定できます。要求できる標準属性は次のとおりです。

- request
- requester
- description
- activityName

このリストに含まれない属性名を要求した場合は、Identity Manager はそれを任意のワークフロー変数と見なします。作業項目にその編集が存在する場合は、その値が返されます。標準ワークフローに共通する変数は次のとおりです。

- accountId
- objectType
- objectName
- diagramLabel

例 : includeRelatedItems フォームプロパティの使用

request 属性と description 属性を含めるには、承認フォームに次のプロパティを追加します。

```
<Properties>
  <Property name='includeRelatedItems' value='true' />
  <Property name='relatedItemAttributes' value='request,description' />
</Properties>
```

例 : relatedItemFilter フォームプロパティの使用

指定できるフィルタ属性は次のとおりです。

表 5-69 relatedItemFilter オプションの値

relatedItemFilter オプションの値	フィルタリングの結果
itemType	itemType が一致する作業項目のみが返されます
activityName	同じアクティビティから作成された作業項目のみが返されます

表 5-69 relatedItemFilter オプションの値 (続き)

relatedItemFilter オプションの値	フィルタリングの結果
request	同じユーザー定義要求文字列を持つ作業項目のみが返されます
locked	現在編集がロックされている作業項目のみが返されます

リストに複数のフィルタ属性が含まれる場合は、それらは論理 AND で連結されます。たとえば、同じ要求文字列を持ち、現在ロックされている作業項目のみを取得するときは、承認フォームに次のプロパティを追加します。

```
<Properties>
  <Property name='includeRelatedItems' value='true' />
  <Property name='relatedItemAttributes' value='request,description' />
  <Property name='relatedItemFilter' value='request,locked' />
</Properties>
```

関連作業項目に関する情報のテーブルを示すフィールド例が承認ライブラリフォームライブラリに追加され、フィールド名が「**Related Approvers**」であるとしします。標準の認証フォームからこのフィールドを参照するには、次のように指定します。

```
<FieldRef name='Related Approvers' />
```

作業項目のリポジトリロックタイムアウトの変更

リポジトリ内の作業項目のロックに適用されるデフォルトの間隔は 5 分間です。
RepositoryConfiguration 設定オブジェクトの RelocatedTypes 要素に次の要素を追加することで、この値を変更できます。

```
<TypeDataStore typeName='WorkItem' lockTimeoutMillis='10000' />
```

最上位属性

次の表は、作業項目ビューの最上位属性を示しています。

表 5-70 作業項目ビューの属性

属性	編集の可能性	データ型	必要性
id	読み取り	String	
name	読み取り	String	
taskId	読み取り	String	

表 5-70 作業項目ビューの属性 (続き)

属性	編集の可能性	データ型	必要性
taskName	読み取り	String	
processName	読み取り	String	
activityName	読み取り	String	
description	読み取り / 書き込み	String	
owner	読み取り / 書き込み	String	
complete	読み取り / 書き込み	ブール型	
variables			
workItem			

id

WorkItem オブジェクトのリポジトリ ID を指定します。この ID は、通常、Identity Manager によって生成され、表示されません。

name

WorkItem オブジェクトのリポジトリ名を指定します。

taskId

ワークフローの TaskInstance のリポジトリ ID を指定します。この属性は、システムが作業項目とワークフロータスクを関連付けるときに使用され、変更できません。

taskName

ワークフローの TaskInstance のリポジトリ名を指定します。この名前は、通常は説明的な値に設定され、表示可能です。この値を変更しないでください。ユーザー更新の一般的なタスク名は、たとえば Updating User jdoe となります。

processName

手動アクションを含むワークフロープロセス定義の名前を指定します。

activityName

手動アクションを含むワークフローアクティビティの名前を指定します。

description

作業項目を説明するテキストを指定します。この内容は、ワークフロープロセス定義によって設定されます。この説明は、通常は作業項目の概要を示すテーブルに表示され、作業項目フォームにも表示されます。

owner

ワークフロープロセスを作成した、現在の **Identity Manager** 管理者またはユーザーの名前を指定します。この属性の値は、通常は **Identity Manager** ユーザーの名前です。この作業項目が匿名ユーザーに割り当てられている場合は、名前に **Temp:** というプレフィックスが付けられます。

complete

手動アクションが完了したときにワークフローを再開する場合は、true に設定します。complete 属性の割り当ては、作業項目フォームで実行してください。

このブール値は編集可能です。

variables

ワークフロータスクからコピーされた変数を含む属性を持つ、別のオブジェクトを指定します。デフォルトでは、手動アクションの作動時に範囲に含まれるすべてのワークフロー変数が、作業項目にコピーされます。これは、プロセス定義の **Exposed Variables** オプションと **Editable Variables** オプションで制御できます。ほとんどの作業項目は、検出された情報を variables 属性の下に表示します。この属性の使用の詳細については、この章で後述する「variables 属性の使用」の節を参照してください。

workItem

作業項目に関する追加情報を指定します。含まれる属性は次のとおりです。

views

views という値を持つワークフロー変数のリストが含まれます。システムはこの属性を使用して、作業項目ビューの表示更新時に、ビューに固有の表示更新処理を行います。

この値を変更しないでください。

related

指定された作業項目を表現する属性のリストを含みます。

表 5-71 workItem.related 属性の下位属性 (作業項目ビュー)

属性	説明
name	作業項目のリポジトリ ID を指定します。
owner	項目の所有者を指定します。
locked	作業項目が編集中であるかどうかを指定します。true という値は、作業項目が現在編集中であることを示します。
complete	作業項目が完了しているかどうかを指定します。true という値は、作業項目が完了していることを示します。
itemType	プロセスによって定義される項目タイプを指定します。デフォルトは approval です。

request

作業項目の目的を簡単に説明します。この説明は、通常、description 属性の値より短く、多くの場合は概要テーブルに表示されます。

requester

承認を開始したユーザーを指定します。

ignoreTimeOut

タイムアウトを無視するかどうかを指定します。システムによって割り当てられる true という値は、これが読み取り専用の作業項目であり、表示中にタイムアウトになる可能性があることを示します。これは、作業項目がすでに存在しない場合に、エラーメッセージを表示する代わりに、無視する作業項目ビューのチェックイン失敗をシステムに伝えます。これは、ユーザーにはメッセージを表示しながらワークフローを継続できるように、ただちにタイムアウトになる状態メッセージのみを目的とした作業項目で便利です。

この値を変更しないでください。

variables 属性の使用

作業項目フォームの記述時にもっともよく参照される属性は、complete と variables です。ワークフローを再開するには、complete 属性の値が true に設定されていなければなりません。多くの場合、これは「承認」や「却下」などのラベルが付けられた押しボタンフィールドへの応答として、非表示フィールドによって設定されます。

variables 属性には、ワークフロータスクからコピーされた変数を値として持つオブジェクトが含まれます。作業項目で頻繁に使用されるワークフロー変数の 1 つに user があります。この変数には、ユーザービューが含まれます。たとえば、作業項目フォームから global.email 属性を参照するには、次のパス式を使用します。

```
variables.user.global.email
```

これは、標準のユーザーフォームで使用される属性パスとは異なります。まず、user というワークフロー変数にビュー全体が格納されます。このため、属性パスに user. というプレフィックスが必要になります。次に、ワークフロー変数が作業項目ビューの variables 属性の下に格納されます。このため、属性パスに variables. という追加プレフィックスが必要になります。

ユーザービュー属性のこの入れ子構造のため、作業項目ビューでは、修正なしで標準のユーザーフォームを使用することはできません。ただし、base context オプションを使用して、ユーザーフォームを参照する作業項目フォームを定義することができます。

例

```
<Form name='WorkItemForm'>
  <Include>
    <ObjectRef Type='UserForm' name='Default User Form' />
  </Include>
  <FormRef name='Default User Form' baseContext='variables.user' />
</Form>
```

注

実際には、作業項目フォームは「承認」や「却下」などのボタンの追加フィールドを必要としますが、作業項目フォームに表示されるデフォルトのユーザーフォームに、一部の項目を表示しないようにすることが必要となる場合もあります。通常は、ユーザーフォームと作業項目フォームの両方で参照できるフォームライブラリに、ユーザーフォームから除外するフィールドを指定できます。

作業項目リストビュー

リポジトリ内の作業項目の集合に関する情報の表示と、複数の作業項目に対する処理の同時実行に使用されます。

このビューのハンドラは、次の情報を収集します。

- 選択されたユーザーに割り当てられているすべての作業項目
- 表示可能な作業項目が割り当てられているユーザー
- 作業項目の転送先にできるユーザー

このビューは、Identity Manager の管理者インタフェースの「承認」ページで使用されます。このビューで使用するデフォルトフォームの名前は、「作業項目リスト」です。

次の表は、作業項目リストビューの最上位属性を示しています。

表 5-72 作業項目リストビューの属性		
属性	編集の可能性	データ型
authType	読み取り / 書き込み	String
userId	読み取り	String
user	読み取り / 書き込み	String
self	読み取り	ブール型
forwardedUser	読み取り	ブール型
itemType	読み取り / 書き込み	String
users	読み取り	List
userIds	読み取り	String
forwardingApproverStyle	読み取り	
forwardingUsers	読み取り	List
forwardingUserIds	読み取り	List
workItems	読み取り / 書き込み	String
selectedWorkItems	読み取り / 書き込み	String
forwardTo	読み取り / 書き込み	ブール型
forwardToNow	読み取り / 書き込み	String
variables	読み取り / 書き込み	String

表 5-72 作業項目リストビューの属性 (続き)

属性	編集の可能性	データ型
action	読み取り / 書き込み	ブール型
confirm	読み取り / 書き込み	ブール型

authType

作業項目へのアクセスをタイプによって指定します。たとえば、**EndUserRule** という認証タイプが組み込まれています。すべてのエンドユーザーは、**EndUserRule** という認証タイプがタグ付けされたすべての規則へのアクセスを暗黙的に取得します。

userId

workItem リストに作業項目が含まれる **Identity Manager** ユーザーの名前を指定します。この属性の初期値は、現在のセッションユーザーの名前です。**NULL** に設定し、認証権限を持つすべての制御対象ユーザーの作業項目を表示することもできます。この属性の値は常に **Identity Manager** のユーザー名であり、表示名ではありません。

フォームでこの値を変更しないでください。ユーザーを変更するには、**user** 属性を設定します。

user

作業項目が一覧表示される **Identity Manager** ユーザーの表示名を指定します。表示名を使用していない場合、この値は **userId** と同じになります。フォームでこの値を変更すると、システムは表示更新時に作業項目リストを再計算します。**NULL** 値は、すべての作業項目の表示を示します。

self

userId が現在のセッションユーザーと同じ場合は、**true** に設定します。

forwardedUser

この属性を設定すると、**userId** によって特定されるユーザーが、別ユーザーへの作業項目の転送を希望したことを示します。その他のユーザーは、表示名で識別されます。

users

現在のユーザーによって制御され、作業項目機能を持つ Identity Manager ユーザーの表示名のリスト。この値は、通常はユーザー選択ボックスの作成に使用されます。カスタムフォームに別の方法でユーザーリストを計算させる場合は、ビューオプションまたはフォームプロパティとして **CustomUserLists** ビューオプションを指定します。

userIds

通常は NULL です。代替表示名を使用するように設定されている場合は、ユーザーリストには表示名が含まれ、このリストには本当のリポジトリ名が含まれます。

forwardingUsers

現在のユーザーが作業項目を転送できる Identity Manager ユーザーの表示名のリスト。この値は、ForwardingApproverStyle 属性の値によって異なります。この属性のデフォルト値は peers です。

itemType

これを設定した場合は、その値と一致する項目タイプを持つ作業項目のみを残すように、リストに含まれる作業項目がフィルタリングされます。これにより、作業項目リストビューは、作業項目のタイプに基づいて項目リストをフィルタリングする機能を持ちます。

forwardingUserIds

通常は NULL です。代替表示名を使用するように設定されている場合は、forwardingUsers リストには表示名が含まれ、このリストには本当のリポジトリ名が含まれます。

workItems

選択されたユーザーの作業項目に関する情報を持つオブジェクトのリスト。オブジェクト名は、作業項目のリポジトリ ID です。

workItems[].owner

所有者の表示名を指定します。user が NULL で、すべての作業項目が表示される場合にのみ設定します。

workItems[].request

要求しているオブジェクトの簡単な説明を指定します。この値は、ワークフロープロセスで実行される手動アクションの WorkItemRequest 式によって計算されます。

workItems[].requester

要求を行ったユーザーの表示名を識別します。

workItems[].description

作業項目のより詳細な説明を指定します。値は、ワークフロープロセスで実行される手動アクションの `WorkItemDescription` 式によって計算されます。この説明は、通常は作業項目の概要を示すテーブルに表示され、作業項目フォームにも表示されます。

workItems[].selected

個々の項目の選択フラグ。selectedWorkItems に代わる属性です。

selectedWorkItems

次のアクションで処理される項目を表す、作業項目 ID のリスト。作業項目オブジェクト内に設定される selected 属性に代わる属性です。SortingTable コンポーネントでは、こちらのほうが便利です。この属性と個別の選択フラグの両方が設定されている場合は、この属性の値が優先されます。

forwardTo

action 属性が Forward に設定されている場合の、選択されているすべての作業項目の転送先となる Identity Manager ユーザーの名前を指定します。

forwardToNow

forwardTo に似ていますが、これは action 属性でもあります。この属性は、forwardTo 属性と action 属性が個別に設定されているかのように、forwardTo の値をコピーし、action=Forward に設定した上で表示更新を行います。フォームコンポーネントでユーザーを選択した直後に、フォームに転送処理を実行させるようにするときは、この属性を使用します。転送をボタンで制御する場合は、フォームコンポーネントに forwardTo 属性を設定し、Forward の action 値を送信するボタンを用意します。

action

(ブール型) NULL 以外の場合は、選択された作業項目に対する処理を開始します。

次に有効な値を示します。

- approve
- reject
- forward
- refresh

NoConfirm オプションが設定されている場合は、アクションはただちに開始されます。それ以外の場合は、**Identity Manager** は `confirm` 属性が `true` に設定されるまで待機します。通常であれば、独自の確認ページの表示がフォームによって定義されます。

confirm

(ブール型) `action` 属性に指定されている処理が実行可能であることを示します。

variables 属性の使用

個々の作業項目を編集するときは、承認または却下に関する追加情報を監査のためにワークフロープロセスに渡せるように、フォームは `comments` などの作業項目変数を設定できます。

作業項目リストビューでアクションを実行するときは、任意の作業項目変数も設定できます。`variables` 属性の値は、承認または却下されたときに属性が作業項目にコピーされるオブジェクトに設定できます。たとえば、`variables` オブジェクトが `comments` という属性を持つ場合、選択されるすべての作業項目とともに同じコメントが保存されます。

```
<Form name='variables.comments'>
  <Default>
    <concat>
      <s>Approval performed on </s>
      <invoke class='com.waveset.util.Util' name='dateToString'>
        <new class='java.util.Date' />
      </invoke>
    </concat>
  </Default>
</Form>
```


注	実際には、作業項目フォームは「承認」や「却下」などのボタンの追加フィールドを必要としますが、作業項目フォームに表示されるデフォルトのユーザーフォームに、一部の項目を表示しないようにすることが必要となる場合もあります。通常は、ユーザーフォームと作業項目フォームの両方で参照できるフォームライブラリに、ユーザーフォームから除外するフィールドを指定できます。

ビューオプション

ビューの作成時または表示更新時に次のオプションを指定して、作業項目リストビューアの動作を制御できます。

userId

作業項目が表示される初期ユーザーの名前を指定します。この名前は、デフォルトでは現在のセッションユーザーとなりますが、このオプションによってデフォルト設定をオーバーライドできます。

CustomUserLists

true に設定すると、users リストと forwardingUsers リストの両方がフォームによってカスタム生成され、ビューハンドラによる生成が抑制されます。システムに多くの承認者が存在する場合は、これらのリストの生成には時間がかかります。フォームがデフォルトの users リストと forwardingUsers リストを必要としない場合は、このオプションを有効にします。

ForwardingApproverStyle

forward To リストに名前を含める管理者のタイプを指定します。この属性のデフォルト値は peers です。設定できる値は次のとおりです。

表 5-73 ForwardingApproverStyle ビューオプションの値

オプションの値	説明
peers	現在のユーザーと同じ組織レベル以上の管理者を指定します
controlled	現在のユーザーが制御している組織の管理者を指定します
all	controlled と peers の両方の管理者を指定します

このオプションとその他のビューオプションは、フォームプロパティとして設定できます。

```
<Form...>

  <Properties>

    <Property name='ForwardingApproverStyle' value='peers' />

  </Properties>

  ...

</Form>
```

NoUserListCache

true に設定した場合は、ビューハンドラは users リストと forwardingUsers リストをキャッシュせずに、フォームの表示を更新するたびにそれらを再計算します。ユーザーリストの計算は負荷が大きいため、通常はリストをキャッシュし、action 属性を Refresh に設定して明示的に指定した場合にのみ表示を更新します。

UserDisplayName

このオプションは、ユーザーリストで、値がリポジトリ名の代わりに使用される拡張ユーザー属性の名前に設定できます。UserUIConfig オブジェクトにも設定できますが、フォームに設定するほうが便利な場合もあります。

NoUserDisplayName

true に設定すると、UserUIConfig オブジェクトに表示名が設定されている場合でも、表示名を使用しません。このオプションをフォームに設定し、UserUIConfig の設定を選択的にオーバーライドすることができます。

NoConfirm

true に設定すると、action 属性によって指定されるアクションが、確認なしでただちに実行されます。

フォームでのビューオプションの設定

一部のフォームでは、ビューオプションを簡単に設定できます。フォームにビューオプションを設定する手順は次のとおりです。次に示す手順では、例として作業項目リストビューを使用します。

1. フォームを Identity Manager IDE または任意の XML エディタにコピーします。
2. フォーム名を変更します。
3. それを `form.workItemList` 属性の下に **System Configuration** オブジェクトに登録します。

カスタムフォームでは、次の例に示されるように、フォームのプロパティとしてビューオプションを指定できます。

例

```
<Form>

  <Properties>

    <Property name='CustomUserLists' value='true' />

  </Properties>

  ...

</Form>
```

deferred 属性

deferred 属性は、別のアカウントの属性値から値を取得する属性です。ビュー (および WSUser モデル) で **deferred** 属性を宣言すると、プロビジョニングエンジンは、アダプタを呼び出す前に、この置き換えをただちに実行します。

deferred 属性が、別のリソースの **GUID** 属性から値を取得する場合は、ソースアダプタはアクションを実行する必要がありません。しかし、ソース属性が **GUID** でない場合は、**realCreate** 操作の二次的な影響として、アダプタは **ResourceInfo.resultsAttributes** マップ内の属性を返さなければなりません。アダプタが属性を返さない場合は、プロビジョニングエンジンはアカウントをフェッチして、値を取得します。これは、値を返すようにアダプタを修正するよりも非効率です。

deferred 属性を使用する状況

deferred 属性は、新しいアカウントを作成し、アカウント属性の値を、ソースアカウントが作成されるまで認識されない別のアカウントの属性の値から取得するように指定するときに使用します。一例を挙げれば、生成される一意の識別子の値を属性に設定する場合がこれに該当します。

deferred 属性の使用

deferred 属性は、主に次の 2 つの手順で定義されます。

1. アカウントは、必ず 2 番目のアカウントの作成前に、ソースリソースに作成します。これは、リソースと、ユーザーへのリソースグループの割り当ての両方を含む、順序が付けられたリソースグループを作成することで行われます。
2. 次の例のように、作成するアカウントのユーザービューに特別な属性を設定します。それぞれの deferred 属性は、ソースアカウントを指定するビュー属性と、ソース属性を指定するビュー属性の 2 つを必要とします。これは、次の形式のパスを使用して設定されます。

```
accounts[<resource>].deferredAttributes.<attname>.resource  
accounts[<resource>].deferredAttributes.<attname>.attribute
```

この <resource> は実際のリソース名に置き換えられ、<attname> は実際の属性名に置き換えられます。

たとえば、1) アカウントの作成時に uid 属性を生成する LDAP というリソースと、2) LDAP リソースの uid と同じ値を保持する directoryid という directoryid 属性を持つ HR というリソースの、2 つのリソースを作成する場合を考えてみましょう。

次のフォームフィールドは、必要なビュー属性を設定し、この関係を定義します。

```
<Field name='accounts[HR].deferredAttributes.directoryid.resource'  
  <Expansion><s>LDAP</s></Expansion>  
</Field>  
<Field name='accounts[HR].deferredAttributes.directoryid  
  <Expansion><s>uid</s></Expansion>  
</Field>
```

ビューの拡張

パスワードや有効化フラグなどの特定のリソースアカウント属性を設定するビューを使用して、追加のアカウント属性を設定できます。ただし、セキュリティを確保するために、これらの拡張属性は登録を必要とします。

属性の登録

属性は、次のいずれかの場所に登録できます。

表 5-74 属性の登録場所

場所	属性をその場所に登録する状況
リソース内の AccountAttributeType 定義	... 更新する属性が、そのタイプのすべてのリソースではなく、特定のリソースのみに適用されます。
システム設定オブジェクト	特定タイプのすべてのリソースに適用されるグローバル登録を行います。このような登録は、XML フォーマットで行います。

異なる属性を異なるビューに登録できます。たとえば、パスワードビューに lock 属性を登録し、名前変更ビューに firstname 属性を登録することができます。

グローバル登録

すべてのリソースに適用されるグローバル登録を行うときは、次のパスを使用して、属性を System Configuration オブジェクトに追加します。

`updatableAttributes.ViewName.ResourceTypeName`

この *ViewName* は Password、Reset、Enable、Disable、Rename、Delete のいずれかで、*ResourceTypeName* はリソースのタイプ名です。all というタイプ名は、すべてのリソースに適用される登録のために予約されています。

この属性の値は、List (String のリスト) です。各文字列は、更新する属性の名前です。

次の例は、すべてのリソースを対象とした delete before action という属性をプロビジョニング解除ビューに登録します。

```

<Attribute name='updatableAttributes'>
  <Object>
    <Attribute name='Delete'>
      <Object>
        <Attribute name='all'>
          <List>
            <String>delete before action</String>
          </List>
        </Attribute>
      </Object>
    </Attribute>
    <Attribute name='Enable'>
      <Object>
        <Attribute name='all'>
          <List>
            <String>enable before action</String>
          </List>
        </Attribute>
      </Object>
    </Attribute>
  </Object>
</Attribute>

```

リソースに固有の登録

リソースに固有の登録を行うには、**Identity Manager** の「デバッグ」ページでリソースオブジェクトを修正し、`AccountAttributeType` 要素に `<Views>` 下位要素を挿入します。`<Views>` には、この属性が更新されるビューの名前を示す文字列のリストを指定します。

```

<AccountAttributeType name='lastname' mapName='sn' mapType='string'>
  <Views>

```

```
        <String>Rename</String>
    </Views>
</AccountAttributeType>
```

ビューでは、変更する属性が、次のオブジェクトの内部に配置されます。

```
resourceAccounts.currentResourceAccounts[ResourceTypeName].attributes
```

```
<Field name=
'resourceAccounts.currentResourceAccounts[OS400ResourceName].attribut
es.delete before action' hidden='true'>
    <Expansion>
        <s>os400BeforeDeleteAction</s>
    </Expansion>
</Field>
```


XPRESS 言語

この章では、式とスクリプトを記述するための XML ベースの言語、XPRESS の基本機能について説明します。この言語で記述されたステートメントは式と呼ばれ、フォームにデータ変換機能を追加したり、ワークフローやフォームなどのオブジェクトに状態遷移ロジックを組み込んだりするために、Identity Manager 全体で使用されます。

この章の内容

この章で説明する主な内容は次のとおりです。

- プレフィックス表記や XML 構文の使用などの XPRESS 言語の基本機能
- Identity Manager で使用される一般的な式の例
- Identity Manager に付属する関数のライブラリ
- 関数から返されるデータの型

XPRESS 言語について

XPRESS は、XML に基づいた構文を使用する関数言語です。この言語で記述されるすべてのステートメントは、0 個以上の引数をとる関数を呼び出し、値を返します。

Identity Manager には多数の関数が組み込まれているだけでなく、新しい関数を定義することもできます。XPRESS は、式の中で任意の Java クラスでのメソッドの呼び出しと、JavaScript の評価もサポートします。

プレフィックス表記

XPRESS 言語は、関数の呼び出しと、C などの言語が式の演算子として参照する内容を区別しません。このため、プレフィックス表記と呼ばれる構文スタイルが採用されています。プレフィックス表記は、式の演算子を記述したあとにオペランドを続けるという点で一般的なインフィックス表記とは異なります。たとえば、インフィックス表記を使用して C 言語で記述された簡単な論理式を考えてみましょう。

```
x == 42
```

このステートメントを、プレフィックス表記を使用した C 言語で記述すると、次のようになります。

```
== x 42
```

C 言語に式の演算子がなく、関数だけを利用できる場合は、このステートメントは次のように表記されます。

```
equals(x, 42)
```

式を記述するのではなく、関数を呼び出すという観点に立つと、プレフィックス表記は簡単に理解できます。

XML 構文と例

XPRESS は、解析と操作が簡単で、Identity Manager で使用されるその他の XML 表記に自然に組み込むことができる XML 構文を使用します。XML 要素の名前は、呼び出す関数の名前です。入れ子構造に含まれる要素は、関数に渡される引数です。また、各要素には、開始と終了のタグがあります (次の例では、<add></add>)。

例

```
<add> <ref>counter</ref> <i>10</i> </add>
```

この例の <add> 要素は、add という関数の呼び出しを表します。この関数には、2 つの引数が渡されます。

- 最初の引数 - ref という関数の呼び出しによって定義される値。ref 関数の引数は、変数名のリテラル文字列です。ref 関数から返される値は、*counter* という変数の現在値です。
- 2 番目の引数 - i という関数の呼び出しによって定義される値。i 関数の引数は、整数のリテラル文字列です。i 関数から返される値は、10 という整数です。

add 関数から返される値は、*counter* 変数の現在値に整数 10 を加えた結果となります。すべての関数呼び出しは、値を返すか、またはいずれかの引数に対する処理を実行します。たとえば、ref の呼び出しによってカウンタの値が得られると、<i> 呼び出しは整数 10 を返し、<add> 呼び出しは 2 つの呼び出しの結果を合計します。

次の例は、古典的な Hello World プログラムです。これは、XPRESS では次のように表記されます。

```
<print><s>Hello World!</s></print>
```

Identity Manager との統合

XPRESS はスタンドアロンインタプリタでも使用できますが、通常は、動作の制御とカスタマイズに XPRESS ステートメントを使用するアプリケーションに組み込まれます。このようなアプリケーションをホストアプリケーションと呼びます。Identity Manager システムで利用される 2 つの重要なホストアプリケーションは、ワークフローとフォームです。

ホストアプリケーションは XPRESS インタプリタへの呼び出しを作成し、このインタプリタにサービスを提供します。ホストアプリケーションが提供する重要なサービスの 1 つに、外部変数参照の解決があります。多くの式は、式に定義されていない変数を参照します。これらの変数の値は、ホストアプリケーションによって提供されます。ワークフローホストアプリケーションでは、式は、ワークフロープロセスに定義されているあらゆる変数を参照できます。フォームホストアプリケーションでは、式は、フォームフィールドのあらゆる値、または式の評価前に設定される defvar の値を参照できます。

式を使用する理由

式は、主に次のタスクに使用されます。

- **ユーザーインタフェースと管理者インタフェースのフォームのカスタマイズ：**フォームは、フィールドの可視性の制御と、表示できる形式へのデータの変換に XPRESS を使用します。
- **ワークフローでの制御の流れの定義：**ワークフローは、ワークフロープロセスの実行手順の順序を決定する遷移条件の定義に XPRESS を使用します。
- **ワークフローアクションの実装：**ワークフローアクションは、XPRESS を使用して実装できます。アクションの式は、単純な計算を実行するだけでなく、Java クラスまたは JavaScript を呼び出して、複雑な処理を実行することもできます。

ワークフロースクリプトとフォームの編集での式の使用については、「[ワークフロー](#)」の章を参照してください。

式の操作

ここでは、Identity Manager での一般的な式の使用、特に次の内容について説明します。

- フィールドの可視性の制御
- フィールドのデフォルト値の計算
- フィールド値の取得
- フィールド値の生成
- ワークフローの遷移条件
- ワークフローアクション
- ワークフローアクションからの Java メソッドの呼び出し

フィールドの可視性の制御

フォームの設計でよく問題となるのは、特定の条件と一致するまで、一部のフィールドの表示を抑制する必要がある場合です。たとえば、特定のリソースに固有のフィールドは、そのリソースがユーザーに割り当てられた場合にのみ意味を持ちます。このようなフィールドは、そのようなリソースが割り当てられた場合にのみ表示されるようにします。それ以外の場合は、これらのフィールドが表示されないようにし、評価の対象からも外します。次の例は、このようなフィールドの可視性を制御するために、<Disable> 要素内の式を使用するフィールド定義を示しています。

```
<Field name='HomeDirectory'>
  <Display class='Text' />
  <Property name='title' value='HomeDirectory' />
</Display>

<Disable>
  <not>
    <contains>
      <ref>accountInfo.typeNames</ref>
      <s>Solaris</s>
    </contains>
  </not>
</Disable>
</Field>
```

<Disable> 要素は、フォーム XML 言語の一部です。<Disable> 要素には、XPRESS 言語で記述された任意の式を含めることができます。この例の式は、accountInfo.typeNames という外部変数に格納されているリストに、Solaris という文字列が含まれているかどうかを確認します。フォームでは、この変数には、ユーザーに現在割り当てられているすべてのリソースタイプのリストが格納されます。

画面表示のためにフォームが処理されると、<Disable> 要素に定義されている式が評価されます。true が返される場合は、このフィールドは表示されません。

NULL および 0 という値は、論理 false を表します。NULL 以外またはゼロ以外の値が設定されたフィールドは、論理 true を表します。つまり、<s>false</s> という式で表される文字列は、NULL 値ではないので、論理 true となります。

フィールド宣言に指定された Derivation、Default、Expansion のいずれかの要素を使った XPRESS によりフィールド値を算出することができます。

フィールドのデフォルト値の計算

フィールドの値は、別のフィールドから計算することも、<Default> 要素を使用して初期値として単純に設定することもできます。<Default> 要素は、編集可能なフィールドの初期化でよく使用され、フィールドにまだ値が割り当てられていない場合にのみ評価されます。また、<Default> 要素は、ユーザーの姓と名に基づいてアカウント ID を算出する場合にもよく使用されます。次の例は、文字列操作式を使用して、ユーザーの名の頭文字と姓を組み合わせたデフォルトのアカウント ID を算出するフィールド定義を示しています。

```
<Field name='waveset.accountId'>
  <Display class='Text' />
  <Property name='title' value='AccountID' />
</Display>
<Default>
  <concat>
    <substr>
      <ref>accounts[Exchange].firstname</ref>
      <i>0</i>
      <i>1</i>
    </substr>
    <ref>accounts[Exchange].lastname</ref>
  </concat>
</Default>
</Field>
```

<Default> 要素は、フォーム XML 言語の一部です。この要素には、XPRESS 式か、または XML オブジェクトという別の言語で表現された要素を含めることができます。XML オブジェクト言語の詳細については、「XML オブジェクト言語」の章を参照してください。

このフィールドを処理すると、システムはまず、waveset.accountId 属性にすでに値が設定されているかどうかを確認します。値が存在しない場合は、<Default> 要素内の式を評価します。この例では、名の頭文字と姓を組み合わせることで値が計算されます。

次の例のように、firstname フィールドと lastname フィールドに値が設定されていることを確認しなければならない場合もあります。

```
<cond>
  <and>
    <notnull><ref>accounts[Exchange].firstname</ref></notnull>
    <notnull><ref>accounts[Exchange].lastname</ref></notnull>
  </and>
  <concat>
    <substr>
      <ref>accounts[Exchange].firstname</ref>
      <i>0</i>
      <i>1</i>
    </substr>
    <ref>accounts[Exchange].lastname</ref>
  </concat>
</cond>
```

このコード例は、その他のプログラミング言語の IF-THEN ステートメントのように構築されています。この cond 式には、次の 2 つの引数があります。

- 条件式
- THEN 式

まず、条件式が評価されます。この式の結果が論理 **true** であれば、cond の値が THEN 式の値となります。条件式の結果が **false** となる場合は、cond の値は NULL になります。

この例の cond ステートメントは、2 つのアカウント属性に値が設定されていることを確認した上で、それぞれの値を使用して accountID を計算します。Default 式は、前提条件が最終的に満たされるまで、またはユーザーがフィールドに値を指定するまで、フォームの表示を更新またはフォームを保存するたびに評価されます。関連するフィールドに NULL 以外の値が設定されている場合は、Default 式は評価されません。

フィールド値の取得

<Derivation> 式は <Default> 式に似ていますが、フィールドに NULL 以外の値が設定されている場合でもフィールドの値を常に計算するという点で異なります。一般的には、別のフィールドの値によって値が置き換えられるフィールドの表示に使用されます。これは、リソースの属性値がエンコードされ、ユーザーに明示されない場合に有効な設計機能です。

次の例は、条件ロジックを使用して、値のセットを別のセットにマップするフィールド定義を示しています。

```
<Field name='location' prompt='Location'>
  <Display class='Text' />
  <Derivation>
    <switch>
      <ref>accounts[Oracle].locCode</ref>
      <case>
        <s>AUS</s>
        <s>Austin</s>
      </case>
      <case>
        <s>HOU</s>
        <s>Houston</s>
      </case>
      <case>
        <s>DAL</s>
        <s>Dallas</s>
      </case>
      <case default='true'>
        <s>unknown</s>
      </case>
    </switch>
  </Derivation>
</Field>
```

<Derivation> 要素は、式を含めることができるフォーム XML 言語の一部です。このフィールドを処理すると、<Derivation> 要素内の式が評価され、このフィールドに表示される値が決定されます。

この例では、リソースアカウント属性 accounts[Oracle].locCode の値と、各 case 式の最初の値が比較されます。一致が見つかると、switch 式の結果は、一致した case 式の 2 番目の値となります。一致が見つからない場合、switch 式の結果は、デフォルトの case 式の値となります。

フィールド値の生成

一部のフォームでは、最初は抽象的に生成されたフィールドのセットをユーザーに表示し、その後のフォームの送信時に、また別の、具体的なリソースアカウントの属性値のセットを生成することが必要となる場合があります。これは、**form expansion** と呼ばれます。<Expansion> 要素は、通常は、フォーム内の編集可能フィールドに依存する非表示フィールドで使用されます。<Expansion> 要素の目的の1つは、エンドユーザーが慣れ親しんだ判読可能な形式のデータを、リソースが認識できる形式のデータに変換することです。たとえば、ユーザーはフォームでマネージャーのフルネームを確認できますが、システムが受け取るのは、そのマネージャーを認識するための一意の ID です。

次の例は、条件ロジックを使用して、前述の例の location フィールドから得られる値を、3 文字の略号に変換するフィールド定義を示しています。

```
<Field name='accounts[Oracle].locCode'>
  <Expansion>
    <switch>
      <ref>location</ref>
      <case>
        <s>Austin</s>
        <s>AUS</s>
      </case>
      <case>
        <s>Houston</s>
        <s>HOU</s>
      </case>
      <case>
        <s>Dallas</s>
        <s>DAL</s>
      </case>
    </switch>
  </Expansion>
</Field>
```

<Expansion> 要素は、式を含めることができるフォーム XML 言語の一部です。このフィールドを処理すると、<Expansion> 要素内の式が評価され、このフィールドの値が決定されます。

この例では、**location** フィールドによるマッピングとは逆のマッピングが行われます。また、**Display** クラスが割り当てられていないため、このフィールドは非表示になります。**Display** クラスを取り除くことで、フィールドはフォームに表示されなくなります。ただし、フィールドは依然としてフォームのアクティブな部分と見なされ、`<Expansion>` 式を利用してリソース属性の値を生成します。

注	ユーザービュー以外のすべてのフォームでは、 Expansion 規則は、ページを再計算するとき、またはフォームを保存するときに実行されます。ユーザービューでは、ユーザーフォームを最初に読み込むときにも <code><Expansion></code> タグが実行されます。

ワークフローの遷移条件

ワークフロープロセスを定義するときは、あるワークフローアクティビティから別のワークフローアクティビティにコントロールを移行するための規則を指定します。2つのアクティビティ間の移行を遷移と呼びます。遷移の使用を制御する規則は、遷移条件と呼ばれます。

たとえば、次のようなアクティビティ定義を考えてみましょう。

```
<Activity name='Check Results'>

    <Transition to='Log Errors'>
        <gt> <ref>ERROR_COUNT</ref> <i>0</i> </gt>
    </Transition>

    <Transition to='end' />

</Activity>
```

このアクティビティは **Log Errors** と **end** という別々のアクティビティへの2つの異なる遷移を定義しています。ワークフローがこのアクティビティを処理するときは、条件が **true** を返す最初の遷移が採用されます。

この例の最初の遷移には、**ERROR_COUNT** 変数の値がゼロより大きいかどうかを調べる条件が設定されています。この遷移は、エラーカウントが正の値である場合にのみ採用されます。2番目の遷移には、条件は設定されていません。このため、最初の遷移条件の結果が **false** となる場合は、常にこの遷移が採用されます。

ワークフローアクション

ワークフローアクティビティは、1つまたは複数のアクションを実行できます。アクションの1つに XPRESS 式の評価があります。次に例を示します。

```
<Activity name='Increment Counter'>
  <Action>
    <expression>
      <set name='counter'>
        <add> <ref>counter</ref> <i>1</i> </add>
      </set>
    </expression>
  </Action>

  <Transition to='Next' />

</Activity>
```

XPRESS によってワークフローアクションが実装されると、XPRESS 式は `expression` 要素に囲まれ、`Action` 要素内に配置されます。この例では、式は *counter* という変数の現在値を参照して、その値に 1 を追加し、加算後の値を同名の変数に割り当てます。

ワークフローアクションからの Java メソッドの呼び出し

複雑なワークフローアクションは、Java によって実装できます。複雑なワークフローアクションの代表的な例には、リレーショナルデータベースへのデータの格納や、ヘルプデスクシステムへのメッセージの送信などがあります。これらの Java クラスは、XPRESS を使用してワークフローに統合できます。

```
<Activity name='Log Status'>

    <Action>
        <expression>
            <invoke name='logStatus'
                class='custom.OracleStatusLog'>
                <ref>accountId</ref>
                <ref>email</ref>
                <ref>status</ref>
            </invoke>
        </expression>
    </Action>
    <Transition to='Next' />

</Activity>
```

この例では、カスタム Java クラス `custom.OracleStatusLog` に定義されている `logStatus` という静的なメソッドが、XPRESS の `invoke` 関数を使用して呼び出されます。このメソッドに渡される 3 つの引数は、ワークフロー変数から取得された値です。

このような場合、主な計算は Java クラスで実行され、XPRESS によってそのクラスがワークフローに統合されます。

式の検証

式の検証は、次の 2 段階で行われます。

1. `lh` コマンドによる XML 構文の検証
2. XPRESS 評価のトレース

lh コマンドによる式構文の検証

ロジックを実際に評価せずに、式の XML 構文を検証する方法は次のとおりです。

1. `PATH` 環境変数に `%WSHOME%\bin` が含まれていることを確認します。**Identity Manager** で使用する環境変数の変更については、『**Identity Manager** インストール』のコマンド行ツールの使用に関する説明を参照してください。

パスに %WSHOME%\bin が設定されていないときは、%WSHOME%\bin を設定するまでツールを実行できません。

2. コマンド行に **lh xmlparse <xpress_file>** と入力します。この **xpress_file** は、検証する XML が記述されたファイルの名前です。このコマンドは XML の正確さを解析し、コンソールにエラーメッセージを表示します。

注 PATH 環境変数に %WSHOME%\bin を含めてください。こうすることで、どのディレクトリも作業ディレクトリとして使用できるようになります。また、Identity Manager の lh コマンドを、どの作業ディレクトリからも実行できるようになります。

XPRESS 評価のトレース

式を正しく記述し、それをリポジトリに格納すると、XPRESS トレースを有効にして、式が正しく機能するかどうかを確認することができます。XPRESS トレースメッセージは、標準出力デバイスに送信されます。多くの場合、XPRESS はアプリケーションサーバー内で評価されるため、トレースメッセージはコンソールウィンドウ、またはアプリケーションサーバーの起動時にアクティブであったログファイルに出力されます。

XPRESS トレースには、次の 2 つの形式があります。

- **グローバルトレース**。グローバルトレースが有効な場合は、すべての XPRESS 式が追跡の対象となります。
- **ブロックレベルトレース**。ブロックレベルのトレースが有効な場合は、指定したブロック内の式のみがトレースの対象となります。ブロックレベルトレースを設定できるのは、フォームのフィールド要素、またはワークフローに含まれる式に限定されます。

トレース出力の量を減らして分析しやすくできるため、通常はブロックレベルトレースが利用されます。

トレースの有効化

グローバルトレースを有効にするには、Waveset.properties ファイルの xpress.trace というエントリの値を **true** に設定します。アプリケーションサーバーの稼動中、Waveset.properties ファイルに変更を加えたときは、アプリケーションサーバーを再起動するか、「デバッグ」ページの「Reload Properties」をクリックしてください。

ブロックレベルトレースを実行するときは、トレースする式を <block> 要素で囲みます。ブロックの開始タグには、trace='true' という属性を設定してください。

```
<block trace='true'>
  <invoke name='getTime' class='java.util.Date' />
</block>
```

または

```
<Default>
  <block trace = 'true'>
    <ref>global.accountId</ref>
  </block>
</Default>
```

無効な設定の例

<block> 要素を、次のように使用しないでください。

```
<block trace='true'>
  <Field name ='field1'>
...
  </Field>
</block>
```

または

```
<Field name='Field2'>
  <block trace='true'>
    <Default>
      <ref>global.accounts</ref>
    </Default>
  </block>
</Field>
```

トレースメッセージには、関数名、各引数の値、戻り値が含まれます。

XPRESS のトレースを無効にするときは、`xpress.trace` の値を `false` に変更し、`Waveset.properties` ファイルを読み込みなおします。

関数

Identity Manager には、式で利用できる XPRESS 関数のライブラリが含まれます。これらの関数は、次のように分類されます。

- 値コンストラクタ式
- 演算式
- 論理式
- 文字列操作式
- リスト操作式
- 条件、繰り返し、およびブロックの式
- 変数と関数の定義式
- オブジェクト操作式
- Java および JavaScript 式
- デバッグと検証の式

値コンストラクタ式

XPRESS では、リテラル値は XML 要素内のテキストとして記述されます。要素名は関数の名前で、リテラルテキストはその関数の引数として扱われます。次に、簡単な原子データ型を構成する関数の例を示します。

i 関数

整数値を構成します。この関数は 1 つの引数を取り、その引数がリテラルテキストとして扱われます。テキストに含めることができるのは数値のみで、オプションとして先頭にプラスまたはマイナス記号を追加できます。

例 1

```
<i>0</i>
```

例 2

```
<i>42</i>
```

例 3

```
<i>-1234</i>
```

list 関数

各引数式を評価し、その戻り値を連結することで、*list* 型の値を構成します。この関数は、複数の引数をとることができます。

例

```
<list>
  <s>apples</s>
  <s>oranges</s>
  <s>wiper blades</s>
</list>
```

null 関数

NULL 値を構成します。

例 1

```
<null/>
```

例 2

```
<null></null>
```

s 関数

文字列値を構成します。この関数は 1 つの引数を取り、その引数がリテラルテキストとして扱われます。文字列の長さは、Java 環境で利用できる隣接メモリーの容量のみによって制限されます。

例

```
<s>Now is the time</s>
```

演算式

式で演算処理を行うには、次の関数を使用します。

add 関数

すべての引数の値に整数を加算します。整数以外の引数は、強制的に整数として扱われます。

例

次の式の結果は、整数 (42) となります。

```
<add> <i>40</i> <i>1</i> <s>1</s> </add>
```

div 関数

すべての引数の値を整数値で割ります。整数以外の引数は、強制的に整数として扱われます。

例

次の式の結果は、整数 (42) となります。

```
<div> <i>84</i> <i>2</i> </div>
```

mod 関数

すべての引数の値を、整数値でモジュロ演算します。引数の値は、強制的に整数として扱われます。NULL が指定された引数は無視されます。

例

次の式の結果は、整数 (42) となります。

```
<mod> <i>142</i> <i>100</i> </mod>
```

mult 関数

すべての引数の値に整数値を掛けます。整数以外の引数は、強制的に整数として扱われます。

例

次の式の結果は、整数 (42) となります。

```
<mult> <i>7</i> <i>3</i> <i>2</i> </mult>
```


sub 関数

すべての引数の値を整数値で減算します。整数以外の引数は、強制的に整数として扱われます。

例

次の式の結果は、整数 (42) となります。

```
<sub> <i>50</i> <i>6</i> <i>2</i> </sub>
```

論理式

式で論理操作を行うには、次の関数を使用します。ほとんどの論理関数は、**true** または **false** を表す 1 および 0 を返します。例外は、cmp、ncmp、and、および or です。

and 関数

すべての引数の値を取得し、いずれかの引数値が論理 **false** である場合にゼロを返します。すべての引数が論理 **true** となる場合は、最後の引数の値が返されます。ゼロ (`<i>0</i>` または `<s>0</s>`) と `<null>` は論理 **false** と見なされます。

例 1

次の式はゼロを返します。

```
<and> <i>42</i> <s>cat</s> <i>null</i> </and>
```

例 2

次の式は cat を返します。

```
<and> <i>42</i> <s>cat</s> </and>
```

cmp 関数

2 つの文字列値を比較します。この関数を使用して、文字列のリストをソートできません。

この関数から返される値は次のとおりです。

- 負の値 - 最初の引数の値が、語彙的に 2 番目の引数の値より小さい
- 正の値 - 最初の引数の値が、語彙的に 2 番目の引数の値より大きい
- 0 (ゼロ) - 2 つの引数の値が等しい

引数の値は、必要に応じて、強制的に文字列として扱われます。

例 1

次の式は -1 を返します。

```
<cmp>  
  <i>20</i>  
  <i>100</i>  
</cmp>
```

例 2

次の式は -16 を返します。この式は、r と b の間にアルファベット順序の違いがあることを示します。b と r の間に 16 文字分の開きがあるため、ray に対して bob を比較した場合は、-16 という値が返されます。反対に、b に対して r を比較した場合は、返される値は 16 となります。

```
<cmp>
  <s>bob</s>
  <s>ray</s>
</cmp>
```

例 3

次の式は 0 (ゼロ) を返します。

```
<cmp>
  <s>daryl</s>
  <s>daryl</s>
</cmp>
```

eq 関数

等価性を検証します。この関数は、通常は 2 つの値をとりますが、複数の値をとることもできます。等価性の検証方法は、最初の引数のデータ型によって決定されます。最初の引数のデータ型に応じて、検証方法が次のように切り替わります。

- string - 以後のすべての引数は強制的に文字列として扱われ、文字列の比較が行われます
- integer - 以後のすべての引数は強制的に整数として扱われ、数値の比較が行われます
- object - 以後のすべての引数は、同じオブジェクトを参照する object 型の値として扱われます

この関数から返される値は次のとおりです。

0 - ステートメントが論理 false となる

1 - ステートメントが論理 true となる

例

```
<eq> <ref>role</ref> <s>engineering</s> </eq>
```

gt 関数

2つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 最初の引数の値が、数値的に 2 番目の引数の値より小さいか、または等しい
- 1 - 最初の引数の値が、数値的に 2 番目の引数の値より大きい

例

```
<gt>  
  <ref>age</ref>  
  <i>42</i>  
</gt>
```

gte 関数

2つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 最初の引数の値が、2 番目の引数の値より小さい
- 1 - 最初の引数の値が、数値的に 2 番目の引数の値より大きい、または等しい

例

次の式は 1 を返します。

```
<gte>  
  <i>10</i>  
  <i>5</i>  
</gte>
```

isFalse 関数

0 と 1 の数値ではなく、true と false の文字列で表現されるブール型の値を参照するときに使用されます。2つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 引数の値が論理 true (ゼロ以外、NULL 以外) で、false という文字列ではない
- 1 - 引数の値が論理 false であるか、または false という文字列を返す

例

次の式は 1 を返します。

```
<isFalse>
  <s>>false</s>
</isFalse>
```

isnull 関数

1 つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - ステートメントが NULL 以外となる
- 1 - ステートメントが NULL になる

例 1

次の式は 1 を返します。

```
<isnull> <null/> </isnull>
```

例 2

次の式は 0 を返します。

```
<isnull> <i>0</i> </isnull>
```

isTrue 関数

0 と 1 の数値ではなく、true と false の文字列で表現されるブール型の値を参照するときに使用されます。2 つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 引数の値が論理 false (ゼロ以外、NULL 以外) となるか、または引数の値が false という文字列である
- 1 - 引数の値が論理 true で、false という文字列ではない

例

次の式は 0 を返します。

```
<isTrue>
  <s>>false</s>
</isTrue>
```

lt 関数

2つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 最初の引数の値が、数値的に2番目の引数の値より大きい、または等しい
- 1 - 最初の引数の値が、数値的に2番目の引数の値より小さい

例 1

次の式は0 (ゼロ) を返します。

```
<lt>
  <i>10</i>
  <i>5</i>
</lt>
```

例 2

次の式は1 を返します。

```
<lt>
  <i>5</i>
  <i>10</i>
</lt>
```

lte 関数

2つの引数をとります。

この関数から返される値は次のとおりです。

- 0 - 最初の引数の値が、数値的に2番目の引数の値より大きい
- 1 - 最初の引数の値が、数値的に2番目の引数の値より小さい、または等しい

例

```
<lte>
  <ref>age</ref>
  <i>42</i>
</lte>
```

ncmp 関数

大文字と小文字を区別せずに、2つの文字列値を比較します。

この関数から返される値は次のとおりです。

- 負の値 - 最初の引数の値が、語彙的に2番目の引数の値より小さい
- 正の値 - 最初の引数の値が、語彙的に2番目の引数の値より大きい
- 0 (ゼロ) - 2つの引数の値が等しい

引数の値は、必要に応じて、強制的に文字列として扱われます。

例

次の式は0を返します。

```
<ncmp>
  <s>Daryl</s>
  <s>daryl</s>
</ncmp>
```

neq 関数

非等価性を検証します。この関数は、eq 関数で行われる等価性検証の否定形のように機能します。

この関数から返される値は次のとおりです。

- 0 - 2つの引数の値が等しい
- 1 - 2つの引数の値が等しくない

例

```
<neq>
  <ref>role</ref>
  <s>management</s>
</neq>
```

not 関数

入れ子構造の式のロジックを反転させます。

この関数から返される値は次のとおりです。

- 1 - 引数の値が論理 **false** となる
- 0 - 引数の値が論理 **true** となる

例

次の例は 0 を返します。

```
<not> <eq> <i>42</i> <i>24</i> </eq> </not>
```

or 関数

複数の引数をとります。

この関数から返される値は次のとおりです。

null - すべての引数の値が論理 **false** となる

式の中で最初に論理 **true** と評価される引数の値

例 1

次の式は論理 **false** となり、NULL を返します。

```
<or> <i>0</i> <i>0</i> </or>
```

例 2

次の式は、論理 **true** となる cat という文字列を返します。

```
<or> <i>0</i> <s>cat</s> </or>
```


notnull 関数

1 つの引数をとります。

この関数から返される値は次のとおりです。

0 - 引数の値が NULL である

1 - 引数の値が NULL 以外である

例 1

次の式は、`firstname` に値が設定されている場合は 1 を返し、`firstname` が NULL の場合は 0 (ゼロ) を返します。

```
<notnull>  
  <ref>firstname</ref>  
</notnull>
```

例 2

次の式は、値が NULL であるため、0 を返します。

```
<notnull><null/></notnull>
```

文字列操作式

式で文字列操作を行うには、次の関数を使用します。

indexOf 関数

別の文字列に含まれる、指定文字列の位置を返します。

例

次の関数は 3 を返します。

```
<indexOf>
  <s>Austin City Limits</s>
  <s>tin</s>
</indexOf>
```

concat 関数

複数の文字列値を連結します。

例

次の式は、<s>Now is the time</s> という文字列を返します。

```
<concat>
  <s>Now </s><s>is </s><s>the </s><s>time</s>
</concat>
```

downcase 関数

強制的に文字列として扱われる、1 つの引数をとります。この関数は、すべての大文字を小文字に変換した状態で、引数の値を返します。

例

次の式は <s>abc</s> を返します。

```
<downcase><s>ABC</s></downcase>
```

ltrim 関数

強制的に文字列として扱われる、1 つの引数をとります。

この関数は、先頭の空白文字を削除した状態で、引数の値を返します。

例

次の式は <s>hello</s> を返します。

```
<ltrim><s>  hello</s></ltrim>
```

match 関数

文字列に含まれる下位文字列を検証します。indexOf 関数の機能に似ていますが、この関数は条件付きの式で、より便利に使用できます。下位文字列が文字列の最初の位置から始まる場合、indexOf は論理 **false** となる 0 を返します。

この関数から返される値は次のとおりです。

0 - 下位文字列が見つからない

1 - 下位文字列が存在する

例

次の式は 1 を返します。

```
<match>
  <s>Austin City Limits</s>
  <s>tin</s>
</match>
```

pad 関数

文字列が指定の長さになるように、文字列に空白文字を挿入します。

最初の引数 - pad 文字が挿入される文字列

2 番目の引数 - 指定する文字列長さ

3 番目の引数 - (省略可能) pad 文字 (デフォルトは空白文字)

例

次の式は <s> email </s> を返します。

```
<pad>
  <s> email</s>
  <i>10</i>
</pad>
```

rtrim 関数

強制的に文字列として扱われる、1つの引数をとります。この関数は、最後に続く空白文字を削除した状態で、引数の値を返します。

例

次の例は 0 (ゼロ) を返します。

```
<cmp>
  <s>hello</s><rtrim><s>hello   </s></rtrim>
</cmp>
```

split 関数

文字列を文字列のリストに分割します。

最初の引数 - 分割する文字列

2番目の引数 - 1つまたは複数の区切り文字のセット。ここに指定された文字が文字列に含まれると、そのたびに文字列が分割されます。

区切り文字の間の各下位文字列から構成されるリストが作成されます。

例 1

```
<split>
  <s>Austin City Limits</s>
  <s> </s>
</split>
```

この式は、次のリストを返します。

```
<list>
  <s>Austin</s>
  <s>City</s>
  <s>Limits</s>
</list>
```

例 2

次の式では、複数の区切り文字が使用されます。

```
<split>
  <s>(512)338-1818</s>
  <s>()-</s>
</split>
```

この式は、次のリストを返します。

```
<list>
  <s>512</s>
  <s>338</s>
  <s>1818</s>
</list>
```

substr 関数

文字列から文字の範囲を抽出します。

この関数には、次の 2 つの形式があります。

- *start* と *length* が引数として指定される (substr 要素の子ノード)。
- *start* と *length* が **substr** ノードの属性として指定され、*s* が *start*、*l* が *length* を表す。

たとえば、次の 2 つは同じ呼び出しを表します。

```
<substr>
  <s>Hello World</s>
  <i>3</i>
  <i>4</i>
</substr>
```

および

```
<substr s='3' l='4'>
  <s>Hello World</s>
</substr>
```

どちらの関数が返す文字列も lo W です。

```
<block>
  <substr s='3' l='4'>
    <s>Hello World</s> --> Hello World
  </substr> --> lo W
</block> --> lo W
```

start パラメータと *length* パラメータの指定は省略可能です。次のように文字列が **substr** ノードの子として文字列のみが指定され、*start* 引数を省略した場合を考えてみましょう。

```
<substr>
  <s>Hello World</s>
</substr>
```

この場合、**substr** ノードの *s* 属性も省略され、開始位置 (**start**) は文字列の先頭と見なされます。つまり、明示しなかった場合は、デフォルト値のゼロが適用されます。

最初の引数 - 文字列

2 番目の引数 - 開始位置

3 番目の引数 - 抽出する文字の数

例

次の式は `<s>Now</s>` を返します。

```
<substr>
  <s>Now is the time</s>
  <i>0</i>
  <i>3</i>
</substr>
```

次の例では、`start` 属性が省略されているため、0 と見なされます。

```
<block>
  <substr l='4'>
    <s>Hello World</s> --> Hello World
  </substr> --> Hell
</block> --> Hell
```

`length` 引数も省略できます。`length` 引数を省略した場合は、この関数は残りの文字列を返します。次のように、`substr` の子ノードとして `string` 引数と `start` 引数のみを指定する場合は、`length` を省略できます。

```
<substr>
  <s>Hello World</s>
  <i>3</i>
</substr>
```

または、`substr` ノードで `l` 属性が省略されている場合にも、省略できます。次の例には `length` 引数が指定されていませんが、指定されている開始位置より後の残りの文字列が返されます。

```
<block>
  <substr s='3'>
    <s>Hello World</s> --> Hello World
  </substr> --> lo World
</block> --> lo World
```

trim 関数

強制的に文字列として扱われる、1つの引数をとります。

この関数は、先頭の空白文字と最後に続く空白文字を削除した状態で、引数の値を返します。

例

次の式は `<s>hello</s>` を返します。

```
<trim><s> hello </s></trim>
```

upcase 関数

強制的に文字列として扱われる、1つの引数をとります。

この関数は、すべての小文字を大文字に変換した状態で、引数の値を返します。

例

次の式は `<s>ABC</s>` を返します。

```
<upcase><s>abc</s></upcase>
```

リスト操作式

ほとんどのリスト操作関数には、関数要素に `name` 属性が含まれているかどうかに応じて2つの形式があります。

- 関数要素に含まれている場合は、その名前はリストを格納した変数に解決されます。この場合、参照された変数は完全に変更されます。次の例は、`someList` 変数に格納されているリストを変更して、2つの要素を追加します。

```
<append name='someList'>

  <s>Hello</s>

  <s>World</s>

</append>
```

関数要素に名前が含まれない場合は、新しいリストが生成されます。次の例では、`someList` 変数に格納されているリストの要素と2つの追加要素を組み合わせ、新しいリストを作成します。`someList` 変数の値自体は変更されません。

```
<append>
  <ref>someList</ref>
  <s>Hello</s>
  <s>World</s>
</append>
```

リスト要素を操作するときは、次の関数を使用します。

append 関数

リストに値を追加します。`name` 属性が指定されているかどうかに応じて、引数のリストには2つの形式があります。`name` が指定されていない場合は、最初の引数がリスト名と見なされ、残りの引数はリストに追加する要素と見なされます。返されるのは、要素が追加されたリストのコピーであり、元のリストは変更されません。`name` 引数が指定されている場合は、すべての引数が追加オブジェクトと見なされ、その名前の変数に格納されているリストに追加されます。この場合は、リストが直接変更されます。

例 1

次の式は、`srcList` 変数に格納されているリストをコピーし、1つの要素を追加します。

```
<append>
  <ref>srcList</ref>
  <s>oranges</s>
</append>
```


例 2

次の式は、値を追加して既存のリストを変更します。

```
<set name= 'somelist'>
  <List>
    <s>We</s>
    <s>say</s>
  </List>
</set>

<append name= 'somelist'>
  <s>Hello</s>
  <s>World</s>
</append>
<ref>someList</ref>
```

appendAll 関数

複数のリストの要素をマージします。name 属性が指定されている場合は、既存のリストが変更されます。指定されていない場合は、新しいリストが作成されます。

例 1

次の式は、*srclist* リストの要素と 3 つの追加要素を組み合わせて、新しいリストを作成します。

```
<appendAll>
  <ref>srclist</ref>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>peaches</s>
  </list>
</appendAll>
```

例 2

次の式は、*srclist* 変数に格納されているリストに 3 つの要素を追加します。

```
<appendAll name='srclist'>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>peaches</s>
  </list>
</appendAll>
```

contains 関数

最初の引数 - リスト

2 番目の引数 - リストから検索する任意のオブジェクト

この関数から返される値は次のとおりです。

1 -- 指定した値がリストに含まれる

例

次の式は 1 を返します。

```
<contains>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
  <s>apples</s>
</contains>
```

containsAll 関数

2 つのリスト引数をとります。

この関数から返される値は次のとおりです。

1 -- 指定したリストに、もう一方のリストのすべての要素が含まれる

0 (ゼロ) -- 指定したリストに、2 番目のリストの一部またはすべての要素が含まれない

例

次の式は 0 を返します。

```
<containsAll>
  <ref>fruitlist</ref>
  <list>
    <s>oranges</s>
    <s>wiper blades</s>
  </list>
</containsAll>
```

containsAny 関数

最初の引数 - 検索される側のリスト

2 番目の引数 - 最初のリストから検索する要素または要素リスト

この関数から返される値は次のとおりです。

1 -- 2 番目のリストの一部またはすべての要素が、最初のリストに含まれる

0 (ゼロ) -- 2 番目のリストのどの要素も、最初のリストに含まれない

例

次の式は 1 を返します。

```
<containsAny>
  <ref>fruitlist</ref>
  <list>
    <s>oranges</s>
    <s>wiper blades</s>
  </list>
</containsAny>
```

filterdup 関数

リストの重複要素をフィルタリングします。リストを指定すると、重複要素が削除された新しいリストが返されます。

例

```
<filterdup>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>apples</s>
  </list>
</filterdup>
```

この式は、次のリストを返します。

```
<list>
  <s>apples</s>
  <s>oranges</s>
</list>
```

filternull 関数

リストの NULL 要素をフィルタリングします。

この関数に 1 つのリストを指定すると、すべての NULL 要素が削除された 1 つのリストが返されます。

例

```
<filternull>
  <list>
    <s>apples</s>
    <null>
    <s>oranges</s>
    <null/>
  </list>
</filternull>
```

この式は、次のリストを返します。

```
<list>
  <s>apples</s>
  <s>oranges</s>
</list>
```

get 関数

リスト内の *n* 番目の要素の値を取得します。リストのインデックスは、ゼロ (0) カウントから始まります。引数はリストと整数です。

例

```
<get>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
  <i>1</i>
</get>
```

この式は `<s>oranges</s>` を返します。

indexOf 関数

最初の引数 - 値が検索されるリスト

2 番目の引数 - 検索する値

この関数は、指定した値がリストに存在する場合はその要素の位置番号を返し、指定した値がリストに存在しない場合は -1 を返します。

例

次の式は 1 を返します。

```
<indexOf>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
  <s>oranges</s>
</indexOf>
```

insert 関数

リストに値を挿入します。新しい要素を追加するスペースを作るために、挿入インデックスより下の要素は下に移動されます。

最初の引数 - 要素が挿入されるリスト

2 番目の引数 - 新しい要素を挿入するリスト内の位置を示す整数

3 番目の引数 - リストに挿入される値

例

```
<insert>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
  <i>1</i>
  <s>wiper blades</s>
</insert>
```

この式は、次のリストを返します。

```
<list>
  <s>apples</s>
  <s>wiper blades</s>
  <s>oranges</s>
</list>
```

length 関数

リスト内の要素の数を返します。この関数を使用して、文字列の長さを取得することもできます。

最初の引数 - リストまたは文字列

例 1

次の式は 2 を返します。

```
<length>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
</length>
```

例 2

```
<length>
  <s>Hello world!</s>
</length>
```

この式は、11 という値を返します。

remove 関数

リストから 1 つまたは複数の要素を削除します。name 属性が指定されているかどうかに応じて、引数のリストには 2 つの形式があります。name が指定されていない場合は、最初の引数がリスト名と見なされ、残りの引数はリストから削除する要素と見なされます。返されるのは、要素が削除されたリストのコピーです。元のリストは変更されません。name 引数が指定されている場合は、すべての引数が削除オブジェクトと見なされ、その名前の変数に格納されているリストから削除されます。この場合は、リストが直接変更されます。

例 1

次の式は、srclist 変数に格納されているリストをコピーして、1 つの要素を削除し、結果リストを返します。

```
<remove>
  <ref>srclist</ref>
  <s>oranges</s>
</remove>
```

例 2

次の式は、値を削除して既存のリストを変更します。

```
<set name= 'somelist'>
  <List>
    <s>We</s>
    <s>say</s>
  </List>
</set>

<remove name= 'somelist'>
  <s>say</s>
  <s>say</s>
</remove>
<ref>someList</ref>
```

removeAll 関数

指定したリストに含まれるすべての要素を、もう一方のリストから削除します。name 属性が指定されている場合は、既存のリストが変更されます。指定されていない場合は、新しいリストが作成されます。

例 1

次の式は、*srclist* リストの要素から 3 つの要素を削除して、新しいリストを作成します。

```
<removeAll>
  <ref>srclist</ref>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>peaches</s>
  </list>
</removeAll>
```

例 2

次の式は、*srclist* 変数に格納されているリストから 3 つの要素を削除します。

```
<removeAll name='srclist'>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>peaches</s>
  </list>
</removeAll>
```

この式によって、リストは次のように変更されます。

```
<list>
  <s>wiper blades</s>
</list>
```

set 関数

リスト内の指定位置に値を割り当て、現在の値を上書きします。インデックスが付けられた要素を含めるために必要となる場合は、リストが拡張されます。リストの拡張時に作成される新しい要素は、NULL になります。

最初の引数 - リスト

2 番目の引数 - 新しい要素を挿入するリスト内の位置を示す整数 (要素の位置番号はゼロから開始)

3 番目の引数 - 要素

例 1

```
<set>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>wiper blades</s>
  </list>
  <i>2</i>
  <s>bassoons</s>
</set>
```

この式によって、リストは次のように変更され、NULL が返されます。

```
<list>
  <s>apples</s>
  <s>oranges</s>
  <s>bassoons</s>
</list>
```

例 2

```
<set>
  <list>
    <s>apples</s>
    <s>oranges</s>
    <s>wiper blades</s>
  </list>
  <i>5</i>
  <s>bassoons</s>
</set>
```

この式によって、リストは次のように変更され、NULL が返されます。


```
<list>
  <s>apples</s>
  <s>oranges</s>
  <s>wiper</>
</null>
</null>
  <s>bassoons</s>
</list>
```

条件、繰り返し、およびブロックの式

式で条件処理とブロック処理を行うには、次の関数を使用します。

block 関数

複数の式を 1 つの式にグループ化します。block 関数の値は、最後の引数の値となります。

注	<set> 関数は値を返しません。block ステートメントの最後の行で set の操作が行われている場合、その block ステートメントは値を返しません。block ステートメントが変数の値を返すようにするには、block ステートメントの最後の行に <ref>variable_name</ref> を追加してください。

例

```
<block>
  <s>Hello there!</s>
  <add> <i>100</i> <i>2</i> </add>
  <i>42</i>
</block>
```

このブロックは、最後の引数の値である 42 を返します。

trace ステートメントでの block の使用例については、「[デバッグと検証の式](#)」を参照してください。

break 関数

式の早期終了を強制します。**break** を使用できる式は、block、dolist、while、and、or です。break 式の値は、それを含む式の値となります。オプションとしてブロック名を指定すると、**break** を使用して、式のいくつかのレベルを終了できます。

例 1

次の式には、ループの単純な **break** 終了が含まれます。

```
<dolist name='el'>
<ref>list</ref>
  <cond><eq><ref>el</ref><s>000</s></eq>
    <break>
    <ref>el</ref>
  </break>
</cond>
<null/>
</dolist>
```

この例では、リストに含まれる要素に対して **dolist** 関数が繰り返され、000 という値が検索されます。**dolist** 関数の値は、各繰り返しの最後の部分式から返された値を連結して構成されたリストとなります。

例 2

次の式は、ブロック名を使用して複数のレベルを終了する例を示しています。

```
<block name='outer block'>
  <dolist name='el'>
    <ref>listOfLists</ref>
    <dolist name='el2'>
      <ref>el</ref>
      <cond><eq><ref>el</ref><s>000</s></eq>
        <break name='outer block'>
        <ref>el</ref>
      </break>
    </cond>
  </dolist>
  <null/>
</dolist>
</block>
```

ループが 2 つ存在する以外は、この式は前の例と同じです。外側のループは、それ自体がリストである要素を持つリストに対して繰り返されます。内側のループは、各要素リストに対して繰り返されます。000 という値が検出されると、両方のループが終了し、**break** 式の **outer block** というブロック名が参照されます。

cond 関数

この関数は、2つの式の値を条件付きで選択する機能を持ちます。これは、C と Java の **ternary** 条件演算子 ($a ? b : c$) と類似した機能です。

例

`cond` 関数は3つの引数をとることができます。最初の引数は条件です。条件の値が論理 **true** であれば、`cond` の値は、2番目の引数の値となります。条件の値が **false** の場合は、`cond` の値は、3番目の引数の値となります。条件の値が **false** で、3番目の引数が指定されていない場合は、`cond` の値は **NULL** となります。

```
<cond>
  <gt;
    <ref>age</ref>
    <i>40</i>
  </gt>
  <s>old</s>
  <s>young</s>
</cond>
```

dolist 関数

リストの要素に対して繰り返されます。`name` 属性の値は、ループ内で参照できる変数の名前となります。

この変数の値は、以後のリスト要素の値となります。

最初の部分式は、ループを適用するリストを返します。残りの部分式は、リスト内の要素ごとに1回繰り返されます。

`dolist` 関数の値は、各繰り返しの最後の部分式から返された値を連結して構成されたリストとなります。

例 1

次の式は、`subset` というリストを作成します。このリストには、`src` に格納されている要素の中で、値が **10** を超える要素のサブセットが含まれます。

```
<set name='subset'>
  <dolist name='el'>
    <cond>
      <gt;
        <ref>el</ref>
        <i>10</i>
      </gt>
    </cond>
  </dolist>
</set>
```

```

        <ref>el</ref>
    </cond>
</dolist>
</set>

```

例 2

次の式は apples を返します。

```

<switch>
  <s>A</s>
  <case default='true'>
    <s>unknown</s>
  </case>
  <case>
    <s>A</s>
    <s>apples</s>
  </case>
  <case>
    <s>B</s>
    <s>oranges</s>
  </case>
</switch>

```

select 関数

リスト内の最初の NULL 以外の値を返します。この関数は、ワークフローなどのコンテキストフォームを正しく取得する必要がある場合に使用されます。

例

```

<select>
  <ref>:display.session</ref>
  <ref>context</ref>
</select>

```

while 関数

条件と一致するまで一連の式を繰り返します。最初の部分式は条件で、ループのたびに評価されます。条件が論理 **false** になると、ループは終了します。while 式の値は、最後の繰り返しでの、ループの最後の式の値となります。

例

次の式は NULL を返します。

```
<while>
  <gt;
    <ref>counter</ref>
    <i>0</i>
  </gt>

  <set name='counter'>
    <sub> <ref>counter</ref>
      <i>1</i>
    </sub>
  </set>
</while>
```

変数と関数の定義式

式で変数と関数を参照、定義するときは、次の関数を使用します。

ref 関数

変数の値を参照します。変数には、ホストアプリケーションがサポートする外部変数と、`<defvar>` を使用して定義される内部変数があります。

例

```
<ref>waveset.role</ref>
```

defvar 関数

新しい変数を定義します。定義した変数は、その変数を定義した式とそれ以降の式で自由に参照できます。変数名の指定には、XML 属性 `name` を使用します。

`defvar` ステートメント自体がその変数を参照することはできません。このような参照を設定すると、ループが生じます。

既存の変数の値を変更する方法については、この節の `set` 関数の説明を参照してください。

注 次のような構造は使用できません。

```
<defvar name='fullname'>
  <ref>fullname</ref>
</defvar>

または

<defvar name='counter' />
  <add><ref>counter</ref>
  <i>0</i>
</add>
</defvar>
```

例 1

次の式は変数を定義し、2つの要素を持つリストとしてその値を初期化します。

```
<defvar name='theList'>
  <list>
    <s>apples</s>
    <s>oranges</s>
  </list>
</defvar>
```

例 2

次の式は変数を定義し、その値を整数ゼロに初期化します。

```
<defvar name='counter'>
  <i>0</i>
</defvar>
```

defarg 関数

<defun> を使用して定義された関数に、引数を定義します。引数は変数に似ていますが、関数に渡す順に定義する必要があります。

例

```
<defarg name='arg1' />
<defarg name='arg2' />
```

defun 関数

新しい関数を定義します。関数の引数を宣言するときは、<defarg> 関数を使用します。関数の実行には、<call> 関数を使用します。関数は、通常はフォーム内で定義されます。

例

```
<defun name='add100'>
  <defarg name='input' />
  <add>
    <ref>input</ref>
    <i>100</i>
  </add>
</defun>
```

call 関数

ユーザー定義の関数を呼び出します。呼び出しの引数には、<defarg> 関数で定義した引数が適用されます。call 関数の引数の順序は、<defarg> の順序と同じである必要があります。以前のリリースでは、call 関数を使用して規則を呼び出すことができませんでした。今後は、**rule** 関数を使用してください。

例

次の式は 142 を返します。

```
<call name='add100'>
  <i>42</i>
</call>
```

rule 関数

規則を呼び出します。規則の引数は、**argument** 要素を使用して、名前で指定して割り当てます。引数の値は、単純な文字列であれば、**value** 属性を使用して設定できます。**value** 属性を省略し、**argument** 要素の本文に式を記述することで、引数の値をその式で計算することもできます。

呼び出す別の規則の名前を動的に計算することで、`<rule>` 要素は別の規則も呼び出すことができます。

フォームやワークフロー内での規則の作成または呼び出しの詳細については、「規則」の章を参照してください。

例

次の式は、指定したユーザーの従業員 ID を返します。

```
<rule name='getEmployeeId'>
  <argument name='accountId' value='maurelius' />
</rule>
```

```
<rule name='getEmployeeId'>
  <argument name='accountId'>
    <ref>username</ref>
  </argument>
</rule>
```

次の式は、返される値を計算する別の規則を呼び出します。

```
<rule>
  <cond>
    <eq><ref>var2</ref><s>specialCase</s></eq>
    <s>Rule2</s>
    <s>Rule1</s>
  </cond>
  <argument name='arg1'>
    <ref>variable</ref>
  </argument>
</rule>
```


set 関数

既存の変数の値を変更します。この関数の動作は、最初の引数のデータ型によって異なります。

set 関数は、次の 4 つの形式のいずれかをとります。

- name 属性が指定されている場合は、関数は変数の値をその名前に変更します。set 関数の引数は、その変数の新しい値となります。
- name 属性が指定されていない場合は、最初の引数のデータ型によって決定される動作が適用されます。最初の引数のデータ型が **string** の場合は、その文字列は変数の名前として扱われ、変数の値は、set 関数の 2 番目の引数の値に変更されます。これは、変更する変数の名前の計算に使用できます。
- 最初の引数のデータ型が *list* の場合は、2 番目の引数は整数のリストインデックスと見なされ、3 番目の引数は、そのリストに配置される値と見なされます。リストの大きさがインデックスよりも大きい場合は、そのインデックスに含まれる現在のリスト要素は置き換えられます。インデックスの大きさがリストと同じか、または大きい場合は、リストは拡張され、新しい値が追加されます。
- 最初の引数のデータ型が **Map** の場合は、2 番目の引数はマップキーと見なされ、3 番目の引数はマップ値と見なされます。エントリは、そのキーと値でマップに配置されます。

例

次の例では、最初の引数に *var_name* 変数が設定されています。

```
<set name='var_name'>  
  <s>new value</s>  
</set>
```

オブジェクト操作式

式に含まれる任意のオブジェクトの値を操作するときは、次の関数を使用します。

get 関数

オブジェクトの属性にアクセスします。または、リストから要素を抽出します。

最初の引数 - オブジェクト

2 番目の引数 - オブジェクトを呼び出すアクセサ関数の名前。たとえば、getName メソッドが存在する場合は、このパラメータは `<s>name</s>`。

最初の引数のデータ型が **list** である場合、この関数の動作は異なります。最初の引数が **list** である場合は、2 番目の引数は整数のリストインデックスと見なされます。そのインデックスに該当する要素が返されます。

例

次の式は、ユーザーに現在割り当てられているロールの名前の文字列を返します。

```
<get>
  <ref>userView</ref>
  <s>role</s>
</get>
```

この式は、Java コードで `userView.getRole()` を呼び出した場合と同じ機能を持ちます。

getObj 関数

Identity Manager リポジトリからオブジェクトを取得します。これは、タイプ名が **User** のユーザーオブジェクトへのアクセスで頻繁に使用されます。

引数は連結され、次のような表記の文字列を構成します。

`<typename>:<objectname>`

この *typename* はリポジトリのタイプ名、*objectname* はそのタイプのオブジェクトの名前です。

例

次の式は、通知ワークフローアプリケーションに渡すための Identity Manager ユーザーの電子メールアドレスを取得します。

```
<Action application='notify'>
  <Argument name='template' value='Provisioning Approval' />
  <Argument name='toAddress'>
    <get>
```

```

        <getobj>
            <s>User:</s>
            <ref>APPROVER</ref>
        </getobj>
        <s>Email</s>
    </get>
</Argument>
</Action>

```

putmap 関数

オブジェクトにマップ要素を割り当てます。

map - マップを指定します。

key - マップキーを指定します。

value - マップキーに割り当てる値を指定します。

例

```

<putmap>
    <ref>userView</ref>
    <s>waveset.role</s>
    <s>engineering</s>
</putmap>

```

set 関数

この関数の動作は、最初の引数のデータ型によって異なります。

list - この関数を使用して、リスト要素を割り当てることができます。

string - 文字列は変数の名前と見なされ、変数の値が割り当てられます。

その他のオブジェクト - 2 番目の引数はそのオブジェクトの属性名、3 番目の引数は割り当てる値と見なされます。

例

次の式は、role 属性の値を engineering に設定します。

```

<set>
    <ref>userView</ref>
    <s>waveset.role</s>
    <s>engineering</s>
</set>

```

setlist 関数

オブジェクトにリスト要素を割り当てます。

list - リストを指定します

index - リスト内の要素の順序を指定します

value - リスト要素に割り当てる値を指定します

例

```
<setlist>
  <ref>myList</ref>
  <i>s</i>
  <s>accounts</s>
</setlist>
```

setvar 関数

変数の値を設定します。この関数は、静的な変数名を受け付けます。

name - 変数の名前を指定します

value - 変数に割り当てる値を指定します

例

```
<setvar>
  <ref>var</ref>
  <s>text</s>
</setvar>
```

instanceOf

オブジェクトが、name パラメータに指定されたタイプのインスタンスであるかどうかを判断します。

name - 調べるオブジェクトタイプを指定します。

この関数は、部分式のオブジェクトが name パラメータに指定されたタイプのインスタンスであるかどうかに応じて、1 または 0 (**true** または **false**) を返します。

例

ArrayList のデータ型は **List** であるため、次の式は 1 を返します。

```
<instanceof name='List'>  
  <new class='java.util.ArrayList' />  
</instanceof>
```

Java および JavaScript 式

式で Java クラスまたは JavaScript の呼び出しと操作を行うには、次の関数を使用します。

invoke 関数

Java オブジェクトまたはクラスでメソッドを呼び出します。Java オブジェクトでメソッドを呼び出す場合は、最初の引数に Java オブジェクトを指定します。残りの引数は、引数としてそのメソッドに渡されます。メソッド名の指定には、XML 属性 `name` を使用します。静的なクラスメソッドを呼び出す場合は、`class` XML 属性で Java クラス名を指定します。すべての引数は、引数としてクラスメソッドに渡されます。

この関数を使用するには、呼び出すクラスとメソッドの名前、それぞれがとる引数、およびメソッドの機能を理解する必要があります。この関数は、次の **Identity Manager** クラスの呼び出しに頻繁に使用されます。

- `FormUtil`
- `LighthouseContext`
- `WorkflowContext`
- `WavesetResult`

詳細については、各クラスのドキュメントを参照してください。

new 関数

Java クラスのインスタンスを作成します。XML 属性 `class` を使用して指定されるクラス名は、クラスのパッケージ名で完全に修飾してください。

この関数を使用して新しいオブジェクトを作成し、メソッドを呼び出さずに、式または規則の値として返すこともできます。

例

```
<new class='java.util.Date' />
```

script 関数

JavaScript のフラグメントをカプセル化します。この式の評価時に、JavaScript インタプリタが呼び出され、スクリプトが処理されます。この式の値は、最後の JavaScript ステートメントの値です。スクリプト内では、env オブジェクトを使用して、ホストアプリケーション内の変数にアクセスできます。

フォームの <Disable> 式のように、パフォーマンスに大きく影響する式で JavaScript を使用することは避けてください。組み込まれているトレース機能を使用すると、短い XPRESS 式でより簡単にデバッグできます。ワークフローアクションの複雑なロジックには JavaScript を使用します。

例

```
<script>
  var arg1 = env.get('arg1');
  arg1 + 100;
</script>
```

```
<script>
  importPackage(Packages.java.util);
  var cal Now = Calendar.getInstance();
  cal Now.getTime()
</script>
```

デバッグと検証の式

トレースを有効にすると、豊富なトレースデータが生成されます。

式の問題の診断に役立つ式のトレース、またはテキストの出力を有効にするには、次の関数を使用します。

注	トレースをグローバルに有効化すると、出力されるトレースデータが膨大になる可能性があります。ブロックレベルでのトレースを有効にしたほうが便利なときは、ブロック要素の <code>trace</code> 属性を <code>true</code> に設定します。

trace 関数

式のトレースを有効または無効にします。引数の評価が `true` になると、トレースが有効になります。

トレースを有効にすると、トレースデータが標準出力に出力されます。

例 1

```
<trace><i>1</i></trace>
```

例 2

```
<trace><i>0</i></trace>
```

print 関数

各部分式の値を標準出力に出力します。

例

```
<print>  
  <s>Ashley World!</s>  
</print>
```


データ型

すべての関数は、次の表に示されるいずれかのデータ型の値を返します。

表 6-1 戻り値のデータ型

データ型	定義
integer	符号が付けられた整数値を表します。値の精度は、32 ビット以上です。
list	順序が付けられた、その他の値のリストを表します。リストに含まれる値は、要素と呼ばれます。NULL もリスト要素になることができます。要素を持たないリストの値は、NULL 値として認識されません。
null	データの欠落を表します。副次的な影響のみを目的として呼び出された場合や、指定された引数から意味のある値を計算できない場合、関数は NULL を返すことがあります。NULL 値の処理方法は、NULL 引数が渡された関数によって異なります。一般的には、NULL 値は論理 false と見なされ、演算式では無視されます。
object	XPRESS 言語以外で定義された任意のオブジェクトへの参照を表します。
string	文字列を表します。XML 構文では、文字列は常に Unicode 文字セットを使用します。string 値に文字を含めないこともできます。このような文字列は空と見なされますが、これは NULL ではありません。

一部の関数は、引数の値を論理 true または false と見なします。XPRESS では、ブール型は使用されません。その代わりに、NULL 値または整数のゼロが false と見なされます。それ以外の値は、true と解釈されます。

ブール型の値を返す eq などの論理関数は、false を表すために整数のゼロを返し、true を表すために整数の 1 を返します。

XML オブジェクト言語

XML オブジェクト言語は、ストリング、リスト、マップなどの一般的な Java オブジェクトを表現できる XML 要素の集合です。

この章の内容

- [XML オブジェクト言語について](#)
- [XML オブジェクト言語とそれに対応する XPRESS](#)

関連する章

- 「[XPRESS 言語](#)」 - 式を使用して、フォームにロジックを設定します。

XML オブジェクト言語について

XML オブジェクトはフォームで頻繁に使用されますが、ワークフローと規則で使用することもできます。もっとも一般的な用途は、フォームの Select フィールドまたは MultiSelect フィールドで許可される値のリストの作成です。次に例を示します。

例

```
<Field name='global.state'>
  <Display class='Select'>
    <Property name='title' value='State' />
    <Property name='allowedValues'>
      <List>
        <String>Alabama</String>
        <String>Alaska</String>
        <String>Arizona</String>
        <String>Arkansas</String>
        <String>California</String>
        <String>Washington</String>
        <String>Washington D.C.</String>
        <String>West Virginia</String>
        <String>Wisconsin</String>
        <String>Wyoming</String>
      </List>
    </Property>
  </Display>
</Field>
```

XML オブジェクト言語の要素は、XPRESS 言語の要素に似ていますが、値が静的な場合は、XML オブジェクト言語を使用するほうが効率的です。

この2つの言語の主な相違点は、XML オブジェクト言語では、オブジェクトの内容を、式を使用して計算できないことです。この制限があるために、システムはオブジェクトをより効率的に作成することができ、オブジェクトが大きい場合の処理が迅速になります。

XML オブジェクト言語を使用してリストを定義すると、リポジトリからフォームが読み込まれるときにリストが作成され、以後はそれが再利用されます。XPRESS でリストを定義した場合は、フォームを表示するたびに新しいリストが作成されます。

XML オブジェクト言語とそれに対応する XPRESS

次の表は、基本的な XML オブジェクトと、それに対応する XPRESS 式 (存在する場合) を示しています。

表 7-1 基本的な XML オブジェクトとそれに対応する XPRESS 式

XML オブジェクト言語	XPRESS 言語
<String>cat</String>	<s>cat</s>
<Integer>10</Integer>	<i>10</i>
<Boolean>true</Boolean>	<i>1</i>
<Boolean>false</Boolean>	<i>0</i>
<null/>	<null/>
<Map>	<map>
<MapEntry key='name' value='neko' />	<s>name</s>
<MapEntry key='ID' value='123' />	<s>neko</s>
</Map>	<s>ID</s>
	<i>123</i>
	</map>
<List>	<list>
<String>cat</String>	<s>cat</s>
<String>dog</String>	<s>dog</s>
<integer>673</Integer>	<i>673</i>
</List>	</list>
<Long>123456789</Long>	なし
<Date>20020911 09:15:00</Date>	なし

XML オブジェクト内で XPRESS ステートメントを使用することはできません。

XPRESS での XML オブジェクトの使用

式の記述が許可される場所であれば、XPRESS 内で XML オブジェクトを使用できます。次の例では、呼び出されたメソッドの引数としてマップが渡されます。

```
<invoke name='printTheMap'>
  <ref>mapPrinter</ref>
  <Map>
  </Map>
</invoke>
```

2.0 より前のリリースの XPRESS では、すべての XML オブジェクトを <o> 要素で囲む必要がありました。この制約は適用されなくなりましたが、XPRESS を含む古いファイルでは、このような設定をまだ見かけることがあります。

XPRESS の代わりに使用される XML オブジェクト言語

XML オブジェクト言語と XPRESS はどちらもフォーム内のリストを表現する方法を提供しますが、リストが長く、静的なデータを含んでいる場合は、XML オブジェクト構文を使用するほうが XPRESS より効率的です。リストはメモリ内に 1 回作成され、参照のたびにそれが再利用されます。一方、XPRESS のリスト構文は、参照のたびに評価され、毎回新しいリストが作成されます。

次の表に示される情報のリストを作成するときは、通常、XML オブジェクト言語が使用されます。

表 7-2 情報リストでの XML の用途

情報リストの種類	使用される場所
マシン名	フォーム
ビジネスサイト	フォーム
承認者名	ワークフロー

XML オブジェクト言語と XPRESS でのリストの表現

XML オブジェクト言語と XPRESS は、どちらもフォーム内のリストを表現する方法を提供します。

XPRESS によるリストの表現

XPRESS でリストを表現するときは、<list> 要素を使用します。<list> 要素には、任意の XPRESS 式を含めることができます。

注	計算される要素を含むリストで使用できるのは、<list> XPRESS 要素のみです。<list> 要素を使用すると、その要素が含まれるフォームの実行速度が低下する場合があります。リストに大量の要素が含まれる場合を除き、このパフォーマンス低下に気が付くことはほとんどありません。フォームでの <list> の使用は許容範囲内であり、よく使用されます。

次の例では、XPRESS リストで文字列定数 <s> を使用していますが、<invoke> 要素または <concat> 要素を使用して、リスト要素を動的に作成することもできます。

例

```
<list>
  <s>cat</s>
  <s>dog</s>
</list>
```

XML オブジェクト言語によるリストの表現

XML オブジェクト言語でリストを表現するときは、<List> 要素を使用します。<List> 要素に含めることができるのは、その他の XML オブジェクトのみです。次の例の <List> 要素の内容は、<String> 要素です。

例

```
<List>
  <String>cat</String>
  <String>dog</String>
</List>
```

両タイプの構文を使用したフォームの例

次のフォームには、XML オブジェクト構文と XPRESS の両方によって定義されたリストを含むフィールドが組み込まれています。

```

<Form>

  <Field name='department'>
    <Display class='Select'>
      <Property name='allowedValues'>
        <List>
          <String>Engineering</String>
          <String>Marketing</String>
          <String>Sales</String>
        </List>
      </Property>
    </Display>
  </Field>

  <Field name='department2'>
    <Display class='Select'>
      <Property name='allowedValues'>
        <expression>
          <list>
            <s>Engineering</s>
            <s>Marketing</s>
            <s>Sales</s>
          </list>
        </expression>
      </Property>
    </Display>
  </Field>

</Form>

```

department フィールドの `allowedValues` リストは、`<List>` を使用して作成される静的なリストとして定義されています。このフォームを何回使用する場合でも、作成されるリストは1つのみです。反対に、**department2** フィールドの `allowedValues` リストは、`<list>` 式を使用して定義されています。このフォームを使用するたびに、新しいリストが作成されます。

XML オブジェクト構文と XPRESS によるマップオブジェクトの定義

XML オブジェクト構文と XPRESS のどちらを使用しても、マップオブジェクトを動的に作成できます。XPRESS の `<map>` 要素の使用は、XML オブジェクト言語の `<Map>` 要素と `<MapEntry>` 要素の使用に似ています。これらの要素の違いは、`<map>` の内容を式を使用して計算できることです。一方、`<Map>` 要素を使用して定義できるのは、静的なマップのみです。

注	マップは、<invoke> 式を使用して呼び出すメソッドの引数として使用される場合があります。たとえば、FormUtil クラスの一部のメソッドは、引数としてマップを必要とします。FormUtil メソッドでのマップ引数の使用の詳細については、「 FormUtil メソッド 」の章を参照してください。

XPRESS によるマップの表現

XPRESS の <map> 要素の内容は、名前と値のペアの式です。偶数番号の式はマップキーを定義し、奇数番号の式はマップ値を定義します。キー式の評価が NULL となる場合は、そのエントリは無視されます。

XPRESS の <map> 要素を使用することで、java.util.HashMap オブジェクトを動的に作成することができます。

```
<map>
  <s>name</s>
  <s>Jeff</s>
  <s>phone</s>
  <s>338-1818</s>
</map>
```

XML オブジェクト構文によるマップオブジェクトの定義

XML オブジェクト構文を使用して、マップオブジェクトを次のように定義できます。

```
<Map>
  <MapEntry key='name' value='Jeff' />
  <MapEntry key='phone' value='338-1818' />
</Map>
```


HTML 表示コンポーネント

この章では、Identity Manager の HTML 表示コンポーネントライブラリについて説明します。HTML 表示コンポーネントは、フォームのカスタマイズで使用されます。フォームのカスタマイズについては、「[Identity Manager フォーム](#)」を参照してください。

この章の内容

ここで説明する内容は次のとおりです。

- [HTML コンポーネントとは](#)
- [コンポーネントクラス](#)
- [コンテナクラス](#)
- [コンポーネントサブクラス](#)

HTML 表示コンポーネント

フォームを設計するときは、ここで説明する HTML コンポーネントを使用します。フォームを作成するには、Identity Manager のフォーム XML 言語 (フォーム) を使用して、HTML 表示コンポーネントを表現します。この言語は実行時に解釈され、必要なコンポーネントが作成されます。これを使用することで、追加の Java 開発をほとんど、または一切行わずに、新しいページを動的に作成できるので、カスタマイズが大幅に簡略化されます。

HTML コンポーネントとは

HTML 表示コンポーネントは、HTML テキストの文字列を生成する Java クラスのインスタンスです。各表示コンポーネントは次の要素を持ちます。

- クラス名 (Display 要素のクラス属性によってフィールドに定義)。この名前は、コンポーネントクラスを識別します。コンポーネントクラスは、コンポーネントの基本動作を決定し、コンポーネントが認識するプロパティセットを定義します。
- 1 つまたは複数のプロパティ (Property 要素によってフィールドに定義)。プロパティは、フィールドの動作と外観をさらに定義します。

表示コンポーネントの指定

表示コンポーネントは、次のように指定します。

```
<Field name='Name'>
    <Display class='Class'>
        <Property name='Name' value='Value' />
    </Display>
</Field>
```

HTML コンポーネントに関するページプロセッサの要件

HTML コンポーネントを実装するフォームには、次のページプロセッサ要件が適用されます。

非表示パラメータ

ほとんどのコンポーネントは、HTML フォームから渡されるパラメータの名前に対応した名前を持ちます。Identity Manager では、一般的な用途のために、いくつかのパラメータ名が予約されています。これらの名前をコンポーネント名に使用しないでください。

表 8-1 非表示パラメータ

予約されている名前	説明
id	編集するオブジェクトの ID が格納されます
command	フォームの送信に使用されるボタンの値が格納されます
activeControl	フォームで最後にアクティブであったコンポーネントの名前を格納します
message	ページの上部に表示される情報メッセージを格納できます
error	ページの上部に表示されるエラーメッセージを格納できます

コンポーネントクラス

HTML コンポーネントは、さまざまな方法で組み合わせることができる独立オブジェクトです。関連するコンポーネントはクラスにまとめられます。コンポーネントクラスには、次の 2 つの主要グループがあります。

- **基本コンポーネントクラス** - 1 つの値を表示、編集するためのコンポーネント
- **コンテナクラス** - 1 つまたは複数のコンポーネントを格納できるコンポーネント

基本コンポーネントクラス

共通コンポーネントクラスには、1 つの値を表示、編集するためのコンポーネントが含まれます。これらのコンポーネントについては、「[基本コンポーネント](#)」の節を参照してください。

コンテナクラス

コンテナクラスは、特定の方法で視覚的に構成されたコンポーネントの集合を定義します。コンテナクラスを作成すると、通常は **HTML** の `table` タグが生成されます。単純なコンテナは、コンポーネントを縦または横に並べることができます。より複雑なコンテナは、より複雑にコンポーネントを配置でき、コンポーネントの周囲に装飾を加えることができる場合もあります。

コンテナ自体がコンポーネントであるため、どのコンテナも別のコンテナに含めることができます。このメカニズムを利用することで、複雑なページレイアウトを作成できます。たとえば、多くのページはタイトルを持ち、その後に編集フィールドのリスト、さらにフォーム送信ボタンの行を持ちます。このようなレイアウトを作成するには、垂直整列を使用して `Panel` コンポーネントを作成し、`Label`、`EditForm`、および `ButtonRow` コンポーネントを含めます。`EditForm` コンポーネント自体にも、多数のサブコンポーネントが含まれます。`ButtonRow` は、垂直整列を使用する `Panel` コンポーネントで、`Button` コンポーネントのリストがここに含まれます。

BorderedPanel

項目を配置するための 5 つの領域 (東西南北と中央) を定義します。北と南の領域に配置されたコンポーネントは、水平に並べられます。それ以外の領域に配置されたコンポーネントは、垂直に並べられます。

次のプロパティーがあります。

- `eastWidth` - 東の領域の幅を指定します
- `westWidth` - 西の領域の幅を指定します

ButtonRow

ボタン配置用のデフォルトオプションを設定します。`Panel` コンポーネントを拡張します。

- `pad` - ボタン行と隣接コンポーネントの間のどこに空白を挿入するかを指定します。指定できるのは、上下の間隔 (`top`、`bottom`) です。値が `NULL` の場合は、間隔はなくなります。デフォルト値は `top` です。
- `divider` - 境界バーを水平の線で表示するか、空白行として表示するかを指定します。`true` に設定すると、境界バーは水平線として表示されます (たとえば、`<hr>`) (ブール型)。

EditForm

この表示コンポーネントは、フォームをブラウザに表示するときのデフォルトの表示クラスです。

フォームコンポーネントは2つの列に配置され、左がタイトル、右がコンポーネントとなります。タイトルには吹き出しヘルプを含めることができます。複数のコンポーネントを1つの行に連結できます。

よく編集されるプロパティは、title、subTitle、および adjacentTitleWidth です。

```
<Form name='Default User Form' help='account/modify-help.xml'>
  <Display class='EditForm'>
    <Property name='titleWidth' value='120'>
    <Property name='adjacentTitleWidth' value='60'>
  </Display>
```

EditForm には、これ以外に次のプロパティがあります。

- adjacentTitleWidth - 隣接するフィールドのタイトルの幅を指定します。このプロパティが定義されていない場合は、デフォルト値のゼロが適用されます。adjacentTitleWidth の値をゼロに設定すると、列のタイトルの大きさは自動的に変更されます。ゼロ以外の値を設定した場合は、隣接する列 (たとえば、2 番目と 3 番目の列) のタイトルの幅は adjacentTitleWidth の値に設定されます。
- componentTableWidth - **EditForm** の幅を指定します (ピクセル単位)。指定しない場合は、デフォルト値の 400 ピクセル、または **EditForm** の defaultComponentTableWidth グローバルプロパティの値が適用されます。
- noAlternatingRowColors - **EditForm** の行を同じ色で表示するかどうかを指定します。noAlternatingRowColors を **true** に設定すると、**EditForm** のすべての行が同じ色で表示されます。指定しない場合は、デフォルト値の false が適用されます。

Panel

もっとも基本的なコンテナを定義します。**Panel** は、単純な直線リストに子要素を表示します。

次のプロパティがあります。

- horizontal - **true** に設定すると、コンポーネントが水平に配置されます (ブール型)。
- horizontalPad - 水平配置コンポーネントの周囲の、テーブルのセルパディングの属性を、ピクセル単位で指定します。
- verticalPad - コンポーネント間に挿入する空白行の数を指定します (ブール型)。

デフォルトの配置は垂直ですが、水平に設定することができます。

Selector

単一値または複数値のフィールド (それぞれ **Text**、**ListEditor** コンポーネントに相当) と、その下の検索フィールドを設定します。検索を実行すると、検索フィールドの下に結果が表示され、値フィールドに結果が取り込まれます。

その他のコンテナコンポーネントとは異なり、**Selector** は値を持ちます (**search** の結果が反映されるフィールド)。このコンポーネントには、通常、検索条件フィールドが含まれます。**Selector** は、検索結果の内容を表示するためのプロパティを実装します。

次のプロパティがあります。

- **fixedWidth** - コンポーネントの幅を固定にするかどうかを指定します (**Multiselect** と同じ機能) (ブール型)。
- **multivalued** - 値のデータ型が **List** であるか、**String** であるかを指定します。このプロパティの値は、**ListEditor** または **Text** フィールドのどちらで値を表示するかを決定します (ブール型)。
- **allowTextEntry** - 用意されているリストから値を選択しなければならないか、手動で入力できるかを指定します (ブール型)。
- **valueTitle** - **value** コンポーネントに使用するラベルを指定します (**String**)。
- **pickListTitle** - **picklist** コンポーネントに使用するラベルを指定します (**String**)。
- **pickValues** - **picklist** コンポーネントで使用する値を指定します。NULL に設定した場合は、選択リストは表示されません (**List**)。
- **pickValueMap** - 選択リスト内の値の表示ラベルのマップ (**Map** または **List**)。
- **searchLabel** - 入力テキストフィールドの横のボタンに、指定のラベルを付けます。値を指定しない場合は、デフォルト値の「...」が適用されます。
- **sorted** - 選択リストの値をソートするかどうかを指定します。複数値の順序を指定しない場合も、値はソートされます (ブール型)。
- **clearFields** - 「クリア」ボタンをクリックしたときに、リセットするフィールドのリストを指定します (**List**)。

次のプロパティは、複数値のコンポーネントでのみ有効です。

- **ordered** - 値の順序が重要であるかどうかを指定します (ブール型)。
- **allowDuplicates** - リストに重複値を含めることができるかどうかを指定します (ブール型)。
- **valueMap** - リスト内の値の表示ラベルのマップを指定します (**Map**)。

次のプロパティは、単一値のコンポーネントでのみ有効です。

- **nullLabel** - NULL 値として解釈される値のラベルを指定します (**String**)。

SimpleTable

コンポーネントをグリッドに配置します。オプションとして、上部に列のタイトル行も設定できます。

次のプロパティーがあります。

- `columns` - 列の見出しを定義します。通常はメッセージキーのリストですが、単純な文字列を定義することもできます (**List**)。
- `rows` - テーブルのセルを定義します。各セルは、コンポーネントでなければなりません (**List**)。
- `columnCount` - 列見出しのリストが指定されていない場合に、列の数を指定します。
- `border` - テーブルの境界線の幅を指定します。値を 0 に設定すると、境界線は非表示になります。
- `noItemsMessage` - 行がない場合に、テーブルに表示するメッセージを指定します。

TabPanel

次に示されるタブの行を表示する、タブ付きパネルの表示に使用されます。デフォルトでは、タブは水平に並べられます。

次のプロパティーがあります。

- `leftTabs` - `true` に設定すると、上部マージンにではなく、左マージンにそろえてタブが配置されます (**ブール型**)。
- `border` - `true` に設定すると、タブの下のメインパネルの周囲に境界線を表示します (**ブール型**)。
- `renderTabsAsSelect` - `true` に設定すると、タブの代わりに選択ドロップダウンとしてタブを表示します。これは、ブラウザの水平スクロールが必要になるほどタブの数が多いフォームで便利です。タブの左マージン配置と組み合わせて使用しないでください。
- `tabAlignment` - ページの内容に相対的な、タブの位置を決定します。有効な値は、`left` (左、デフォルト設定)、`top` (上)、`right` (右)、`bottom` (下)、`center` (中央)、および `middle` (中間) です。

```
<Field name='MainTabs'>
  <Display class='TabPanel'>
    <Property name='leftTabs' value='false' />
    <Property name='tabAlignment' value='left' />
  </Field>
```

Row

水平配置に対応したパネルの作成に使用されます。

SortingTable

列の見出しを指定して内容をソートできるテーブルの作成に使用されます。このテーブルの内容は、子コンポーネントによって定義されます。列ごとに1つの子コンポーネントを作成してください (columns プロパティで定義)。通常は、列は **FieldLoop** に格納されます。

テーブルセルの表示時に、このコンポーネントには、子コンポーネントの align、valign、および width プロパティが適用されます。

次のプロパティがあります。

- **pageButtonAlign** - ページの内容に相対的な、ボタンの位置を決定します。有効な値は、**left** (左)、**right** (右)、**bottom** (下)、および **center** (中央) です。デフォルト値は、**right** です。
- **sortEnable** - **true** に設定すると、列のソートが有効になります (ブール型)。
- **sortURL** - 列のソートを選択したときに **Identity Manager** が送信する URL を指定します。列のソートを設定しない場合は、**HtmlPage** の **_postURL** が使用されます (**String**)。
- **sortURLParams** - **sortURL** とともに渡されるパラメータを指定します (**String**)。
- **sortColumn** - 現在ソート基準として使用されている列の番号を指定します。デフォルトでは、この値は最初の列に設定されます (**Integer**)。
- **sortOrder** - ソート順序を指定します。指定できる値には、**asc** (昇順) と **desc** (降順) があります。デフォルト値は **asc** です (**String**)。
- **linkEnable** - 最初の列をリンク付きに指定して、テーブルを生成するかどうかを指定します (ブール型)。
- **linkURL** - リンクの生成時に **Identity Manager** がリンクする URL を指定します。指定しない場合は、そのコンポーネントが使用される **HtmlPage** の送信 URL がデフォルト値として適用されます (**String**)。
- **linkURLArguments** - リンク URL に含める引数を指定します。
- **linkColumn** - **linkURL** 属性によって指定されるリンク生成で使用する列の番号を指定します (**Integer**)。
- **linkParameter** - リンク行 **id** の値を持つ送信データパラメータの名前を指定します。デフォルト値は **id** です。
- **selectEnable** - チェックボックスの列を、複数選択テーブルの左マージンにそろえて表示するかどうかを指定します。**true** に設定すると、**Identity Manager** はチェックボックスの列を表示します (ブール型)。

- columns - テーブル列の見出しのリストを指定します (文字列の List)。
- pageSize - テーブルに同時に表示する _pageSize の最大エントリ数を指定します。_pageSize より多くのエントリが存在する場合は、インタフェース要素で結果のページを切り替えることができます。_pageSize に 1 (デフォルト設定) を下回る値を設定すると、すべてのエントリが一度に表示されます (Integer)。
- useSavedPage - pageSize の値が 0 より大きい場合にテーブルをソートすると、現在の HTTP セッションのソートテーブルページが <fieldName>_currentPage 属性に保存されます。_useSavedPage プロパティは、現在のページを HTTP セッションから取得して表示するかどうかを指定します。このプロパティの値を XPRESS 式の結果にすることで、SortingTable コンポーネントが含まれる JSP に戻るときに、現在のページをいつ呼び出しなおすかをフォームまたはビューで制御できます (ブール型)。

たとえば、SortingTable コンポーネントが編集可能な項目を含むクエリ結果を表示する場合、ユーザーが結果テーブルの項目を編集したあとに、編集後の項目を含む結果ページが必ず表示されるようにするには、0 より大きな値を指定します。

WizardPanel

いくつかの子コンポーネントの 1 つ (通常は EditForms) の表示に使用されます。コンポーネント間の移動には、ウィザード形式の「次へ」および「戻る」ボタンが使用されます。

次のプロパティがあります。

- button - 子コンポーネントの値をボタン行に表示する、子コンポーネントの location プロパティの値を指定します (String)。
- nextLabel - 「次へ」 ボタンに表示するラベルを指定します。デフォルト値は、Next です (String)。
- prevLabel - 「戻る」 ボタンに表示するラベルを指定します (String)。
- cancelLabel - 「キャンセル」 ボタンに表示するラベルを指定します (String)。
- okLabel - 「OK」 ボタンに表示するラベルを指定します (String)。
- noOk - 「OK」 ボタンを非表示にするかどうかを指定します (ブール型)。
- alwaysOk - true に設定すると、「OK」 ボタンが表示されます (ブール型)。
- noCancel - true に設定すると、「キャンセル」 ボタンは表示されません (ブール型)。
- topButtons - true に設定すると、ページの下部ではなく、ページの上部にボタンが表示されます (ブール型)。
- noButtons - true に設定すると、すべてのボタンが非表示になります (ブール型)。

コンポーネントサブクラス

すべてのコンポーネントは、ほとんどのコンポーネントに共通するプロパティを定義した `Component` クラスを拡張します。また、一部のコンポーネントは `Container` クラスを拡張し、別のコンポーネントを格納できるようにします。

各 `Component` サブクラスは、コンポーネントの基本クラスより詳細にコンポーネントの特性を指定する、数多くのプロパティを定義します。たとえば、`Label` コンポーネントは `font` プロパティをサポートします。このプロパティは、ラベルの表示に適用されるフォントを指定します。

表記上の規則

プロパティ名は、常に小文字から始まり、隣接する単語との区切りにはキャメルケース (2 番目以降の複合単語の頭文字を大文字で表記) が使用されます。アクセスメソッド名はプロパティ名の大文字表記で、`get` または `set` のプレフィックスが付けられます。たとえば、`font` というプロパティには、`getfont` および `setfont` メソッドを利用して、`Java` からアクセスできます。

各プロパティのデータ型は多様です。各プロパティの説明を参照してください。次の表は、プロパティ値のデータ型を表現する用語を示しています。

データ型

次の表は、コンポーネントのプロパティで使えるデータ型を示しています。

表 8-2 HTML コンポーネントプロパティのデータ型

データ型	説明
null	プロパティが値を持たないことを表します
String	もっとも一般的なデータ型です。String 値は、通常は Java の String クラスのインスタンスで表現されます。一部のコンポーネントは、任意のクラスの値です。これは、強制的に、文字列に toString メソッドを適用することを意味します。 特に明記されていない限り、すべてのプロパティのデータ型は String であると考えてください。 例:<String>Hello World</String>
文字列の List	値が、1 つまたは複数の文字列のリストであることを表します。Java では、この値は常に List クラスのインスタンスとして実装されます。また、リストを構成する要素は、String クラスのインスタンスです。 例： <pre><List> <String>choice one</String> <String>choice two</String> </List></pre>

基本コンポーネントクラス

Component クラスは、すべての HTML コンポーネントの基本クラスです。このクラスには、ほとんどのコンポーネントに共通するプロパティが含まれます。Component クラスのすべてのプロパティが、すべてのサブクラスに関連するわけではありません。たとえば、Component は、値の制約リストを含めることができる `allowedValues` プロパティを定義します。このプロパティは、Select や MultiSelect のように、値を編集できるサブクラスにのみ関連します。一方で、Container クラスは、編集可能な値を直接表すことは、ほとんどありません。このため、コンポーネントの値に関連するプロパティとは無関係です。一部のプロパティは、コンポーネントが特定の Container クラスに含まれる場合にのみ関係します。

name

フィールドの内部名を指定します。すべての編集コンポーネントは名前を持ちます。この名前は、通常、ページに表示されるその他すべてのコンポーネント名と重複しません。多くの場合は、`name` はビュー属性へのパスを表す文字列です。

コンテナコンポーネントは名前を必要としません。また、割り当てられているすべての名前は無視されます。コンポーネントを Java で構築する場合、コンポーネント名がアプリケーションによって定義されます。XML フォームからコンポーネントを構築する場合は、コンポーネント名はフォームの Field 要素の名前から継承されます。フィールド名は、フォームで使用されるビューオブジェクト内のパス式です。

例

```
<Field name ='global.firstname'>
```

`name` 属性がユーザービューの特定の属性をどのように参照するかについての詳細は、「[Identity Manager のビュー](#)」を参照してください。

title

(省略可能) フィールドの外部名を指定します。タイトルは、1つの列にタイトル、別の列にコンポーネントを持つ HTML テーブルを作成する `EditForm` コンテナで頻繁に使用されます。

コンポーネントは、独自のタイトルを表示しません。タイトルの表示はコンテナによって制御されます。多くのコンテナはタイトルを無視します。

例

```
<Property name='title' value='FirstName' />
<Property name='title'>
  <expression>
    <concat>
      <s>Edit User: </s>
      <ref>waveset.accountId</ref>
    </concat>
  </expression>
</Property>
```

この例では、フィールドのタイトルの一部が、ユーザーの **Identity Manager** アカウント ID から動的に取得されます。

value

編集コンポーネントは、NULL 値にも設定できる値を持ちます。多くの場合、値はビューの属性に基づいて、**Identity Manager** によって自動的に設定されます。一部のコンポーネントでは、現在のビューの内容を無視して値を設定できます。この値は、NULL 値にすることもできます。

Component クラスでは、任意の **Java** オブジェクトを値として指定できます。サブクラスは、割り当て時、または **HTML** の生成時に、強制的に特定のデータ型を値に適用します。コンポーネントの値は、ほとんどの場合、常に **String** オブジェクト、または文字列を含む **List** オブジェクトです。コンポーネントのデータ型の詳細については、「[データ型](#)」の節を参照してください。

ほとんどのコンテナクラスは値を持ちません。割り当てた値は無視されます。ただし、一部のコンテナには値を設定できます (たとえば、**TabPanel**、**WizardPanel**)。

XML フォームからコンポーネントを構築するときは、値は、通常、基本となるビューオブジェクトへのパスを示す、コンポーネントの **name** から継承されます。そのビューオブジェクトには、編集の対象となるすべての値が含まれます。

例

```
<Property name= 'value' value='false' />
```

allowedValues

コンポーネントで利用できる値のオプションリストを指定します。指定した場合、そのコンポーネントで選択できる値は、このリストに含まれる値に限定されます。コンポーネントが値の制限をサポートする場合は、利用できる値のリストはここに格納されます。この値は常にリストであり、その値は通常、文字列です。XML フォームからプロパティを設定するときに便利のように、利用できる値をカンマ区切りのリストとして指定することもできます。

例

```
<Property name='allowedValues' value= 'Mon, Tue, Wed, Thurs, Fri' />
<Property name='allowedValues' />
  <expression>
    <call name='DaysOfWeek' />
  </expression>
</Property>
```

primaryKey

このプロパティは、SortingTable コンテナでのみ認識されます。SortingTable コンテナは、コンポーネントをテーブル形式にまとめます。各列には、同じクラスのコンポーネントが配置されます。SortingTable を使用することで、任意の列の値に基づいて行をソートできます。通常、ソート順序は、列の各コンポーネントの value によって決定されます。しかし、コンポーネントの値がソートに適さない場合や、比較に適さない場合もあります。このようなときは、それに代わる数値ソートキーを指定できます。

required

true に設定した場合は、フォームの送信前に、フィールドに値を指定しなければなりません。このコンポーネントを EditForm で使用する場合は、保存前にフィールドに値を指定しなければならないことをユーザーに示すために、コンポーネントの後に赤のアスタリスク (*) が表示されます。必須スキーママップ属性を選択した場合は (つまり、値を true に設定した場合は)、そのフィールドは常に必須フィールドとなります。

プロパティの値は、true または false です。

例

```
<Property name='required ' value='true ' />
```

noNewRow

true に設定すると、**Identity Manager** ページの前のフィールドの横にフィールドが表示されます。設定しない場合、または false に設定した場合は、フィールドは前のフィールド直下の新規行に表示されます。デフォルト値は false です。

このブール型プロパティは、フィールドが EditForm 表示クラスを使用するフォームに指定されている場合にのみ認識されます。通常、EditForm は各コンポーネントを新しい行に表示し、左の列にタイトル、右の列にコンポーネントを示します。複数のコンポーネントを同じ行に連結することで、表示領域を節約できます。コンポーネントにタイトルも設定されている場合は、そのコンポーネントと前のコンポーネントの間に、強調表示されないテキストとしてタイトルが表示されます。

次の値があります。

```
value='true ' | 'false '
```

例

```
<Property name='noNewRow ' value='true ' />
```

location

コンテナが複数の表示領域を定義し、特定の領域にコンポーネントを追加しなければならない場合に使用されます。一部のコンテナでは、location プロパティに値を割り当てることでコンポーネントの配置を制御できます。たとえば、BorderedPanel コンテナは、東西南北と中央の 5 つの異なる表示領域をサポートします。

location プロパティで認識される値は、コンテナによって定義されます。場所を割り当てない場合、または認識されない場所名を割り当てた場合は、コンテナはコンポーネントをデフォルトの場所に配置します。

help

ユーザーがフィールドの目的を理解できるように、画面に表示する説明テキストを指定します。ほとんどの **Identity Manager** ページでは、このコンポーネントを設定すると、コンポーネントタイトルの横に <icon> アイコンが表示されます。マウスカーソルをこのアイコンに重ねると、左マージンにヘルプテキストが表示されます。

プロパティの値には、表示するリテラルテキスト、またはメッセージカタログキーを指定できます。リテラルテキストには **HTML** マークアップを含めることができます。

カスタムフォームにヘルプを追加する方法の詳細については、「[Identity Manager フォーム](#)」の章の「[フォームへのガイダンスヘルプの追加](#)」を参照してください。

inlineHelp

Identity Manager ページのコンポーネントの下に表示されるテキストを指定します。

プロパティの値には、表示するリテラルテキスト、またはメッセージカタログキーを指定できます。リテラルテキストには **HTML** マークアップを含めることができます。

command

コンポーネントの変更時に送信するコマンドを指定します。ユーザーが値を変更すると、フォーム出力が再計算されます。

このプロパティは、通常、Button コンポーネントで使用されます。一部のコンポーネントでは、変更が加えられた場合に、アプリケーションがその変更に応じてページを再生成できるように、そのコンポーネントが含まれる **HTML** フォームをただちに送信する必要があります。command プロパティに **NULL** 以外の値を指定することで、この動作を設定できます。

command プロパティを設定してコンポーネントを変更すると、フォームが送信され、command という追加の非表示パラメータも送信されます。この非表示パラメータの値は、command プロパティの値です。

command は、ビューに加えられた変更をシステムがどのように処理するかを指定します。command プロパティは、次のいずれかの値をとります。

表 8-3 command プロパティの値

値	説明
Save	編集内容を保存します。
Cancel	編集内容を破棄します。
Recalculate	ページを再生成します。
SaveNoValidate	編集内容を保存しますが、フォームの検証は行いません。

フォームでは、**command** はよく **Recalculate** に設定されるため、この設定のために短い構文が用意されています。Display 要素は action という属性を持ちますが、これを **true** に設定した場合、command プロパティを **Recalculate** に設定した場合と同じ効果が得られます。

```
<Display class='Select' action='true'>
```

onClick

値を設定すると、その値は、このコンポーネントで生成される input 要素の onClick 属性の値として割り当てられる **JavaScript** を持つものと見なされます。すべてのコンポーネントが onClick プロパティをサポートするわけではありません。

このプロパティが使用されることはあまりありません。使用する場合は、生成される HTML に関する詳しい知識が必要です。このプロパティを使用する場合は、通常、onClick の値から呼び出す **JavaScript** 関数を定義する Javascript コンポーネントが、ページに必要です。

例

```
<Property name='onClick' value="Uncheck(this.form,
'resourceAccounts.selectAll');" />
```

注

フォームがリポジトリに格納されると、**Identity Manager** は常に引用符 (一重) で属性値を囲みます。属性値内で引用符を使用するときは、それを `'` に置き換えます。このエスケープを使用しない場合は、XPRESS の s 式で文字列を表現します。

```
<Property name='onClick'>
    <s>Uncheck(this.form, 'resourceAccounts.selectAll'); </s>
</Property>"
```

onChange

これは command コンポーネントに似ています。このコンポーネントの値には、フィールドの変更時に実行する、任意の **JavaScript** ステートメントを指定できます。

すべてのコンポーネントが onChange プロパティをサポートするわけではありません。

このプロパティが使用されることはあまりありません。使用する場合は、生成される HTML に関する詳しい知識が必要です。このプロパティを使用する場合は、通常、onChange の値から呼び出す **JavaScript** 関数を定義する Javascript コンポーネントが、ページに必要です。

nowrap、align、width、valign、および colspan

ほとんどのコンテナは、HTML の table タグで囲んで、サブコンポーネントを配置しています。各コンポーネントに生成される HTML は、通常は td タグに含まれます。一部のコンテナは、nowrap、align、width、および colspan の各プロパティを認識し、テーブルセルのタグを生成するときにそれらを使用します。これらのコンポーネントを使用することで、コンテナ内のコンポーネントの配置や大きさを調整できます。

- nowrap - 一部のコンポーネントについて、長い文字列のテキストが含まれる場合にどのように表示するかを指定します。nowrap の値を **false** に設定した場合、または値を指定しなかった場合は、ブラウザはコンポーネントの表示時にテキストを複数の行に分割することがあります。nowrap を **true** に設定すると、ブラウザはコンポーネントテキストの 1 行での表示を試みます。
- align - あまり使用されません。フォーム上の要素の横方向の位置を調整します。指定できる値は、**left** (左)、**right** (右)、および **center** (中央) です。
- valign - あまり使用されません。コンポーネントの縦方向の位置を指定します。指定できる値は、**top** (上)、**bottom** (下)、および **middle** (中間) です。
- colspan - 非推奨です。

例

```
<Property name= 'width' value='3' />
<Field name='Start Day' prompt='Day' nowrap='true' />
```

htmlFormName

コンポーネントが表示される HTML の <FORM> タグの名前属性を設定するときに使用されます。これにより、コンポーネントで使用される JavaScript 関数が、適切な HTML フォームを確実に参照します。デフォルト値が mainform であるため、このプロパティは、mainform 以外のフォームにコンポーネントを表示する場合にのみ使用されます。

例

```
<Property name='htmlFormName' value='endUserNavigation'>
```

基本コンポーネント

BackLink

前のページに戻るリンクを表示します。このコンポーネントは、ブラウザの「戻る」ボタンと同じように機能します。ただし、このリンクはページ上の使いやすい場所に配置できます。

この表示コンポーネントには、次のプロパティがあります。

- `text` - リンクのテキストを指定します。テキストを指定しない場合は、デフォルト値の **Back** が適用されます。

例

```
<Field name='back'>
    <Display class='BackLink'>
        <Property name='value' value='previous page' />
    </Display>
</Field>
```

Button

ボタンを表示します。ボタンは通常、そのボタンが含まれるフォームの送信に使用されますが、任意の **JavaScript** を実行するように定義することもできます。

この表示コンポーネントには、次のプロパティがあります。

- `name` - ユーザーがこのボタンをクリックしたときに送信されるパラメータの名前を指定します。このプロパティは省略可能です。指定しなかった場合は、デフォルト値の `command` が適用されます。
- `value` - ユーザーがこのボタンをクリックしたときに送信されるパラメータの値を指定します。
- `label` - ボタンに表示されるテキストを指定します。
- `command` - `name` パラメータとともに送信するオプション値 (たとえば、**Save**、**Cancel**、**Recalculate**) を指定します。
- `postURL` - フォームの代替送信先 **URL** を指定します。この値は、**JSP** に指定されている **URL** に優先して適用されます。
- `hiddenID` - フォーム送信データに含まれる `id` パラメータのオプション値を指定します。

例

```
<Display class ='Button'>
```

```
<Property name ='label' value ='Change Password'/>
<Property name ='value' value ='Recalculate'/>
</Display>
```

Checkbox

チェックボックスを表示します。チェックボックスを選択すると、そのボックスの値は true となります。選択されていないボックスの値は false となります。

この表示コンポーネントには、次のプロパティがあります。

- **label** - (省略可能) チェックボックスの右に表示されるラベルを指定します。これはコンポーネントの横に表示されますが、タイトル列には表示されません。
- **leftLabel** - ラベルがチェックボックスの左に表示されるように指定します。
- **checkAll** - このチェックボックスを「すべてを選択」チェックボックスとして機能させる場合に使用されます。この値は、その他のチェックボックスにも反映されます。プロパティの値は、HTML ページのその他のチェックボックスの名前と一致する正規表現です。
- **uncheck** - 同期された一連のチェックボックスの「すべてを選択」チェックボックスを表す、別のチェックボックスフィールドの名前を指定します。これを設定すると、このチェックボックスの選択状態が変わるたびに、「すべてを選択」チェックボックスの選択が解除されます。
- **syncCheck** - このプロパティが設定されているチェックボックスフィールドの値との同期を保つ、別のチェックボックスフィールドの名前を指定します。これを設定すると、このチェックボックスの値が変更されるたびに、同期されているチェックボックスに同じ値が設定されます。
- **syncUncheck** - このプロパティが設定されているチェックボックスフィールドの値が false に変更された場合に同期を保つ、別のチェックボックスフィールドの名前を指定します。これを設定すると、このチェックボックスの値が false に変更されるたびに、同期されているチェックボックスの値も false (未選択) に設定されます。
- **syncCheckAllTo** - このプロパティが設定されているチェックボックスフィールドの値が false に変更された場合に、正規表現と一致するすべての「すべてを選択」チェックボックスとの同期が維持されます。このプロパティの値は、1 つまたは複数の「すべてを選択」チェックボックスを表す正規表現です。
- **syncUncheckAll** - このプロパティが設定されているチェックボックスフィールドの値が false に変更された場合に同期を保つ、別のチェックボックスフィールドの名前を指定します。これを設定すると、このチェックボックスの値が false に変更されるたびに、同期されているチェックボックスの値も false (未選択) に設定されます。

- syncCheckTo - このプロパティーが設定されているチェックボックスフィールドの値は、正規表現と一致するすべてのチェックボックスと同期を保ちます。このプロパティーが設定されているチェックボックスフィールドの値が変更されるたびに、同期されているチェックボックスに同じ値が設定されます。このプロパティーの値は正規表現です。
- value - チェックボックスの状態を決定します。値が論理 **true** の場合は、チェックマークが表示されます。

例

```
<Field name='accounts[NT].passwordExpired'>
    <Display class='Checkbox'>
        <Property name='title' value='Password is Expired' />
    </Display>
</Field>
```

DatePicker

ユーザーが、カレンダーを表示するアプレットを使用して、日付を指定できるようにします。フィールドは、カレンダーアイコンの形式で表示されます。このアイコンをクリックすると、別のウィンドウにカレンダーアプレットが呼び出されます。

次のプロパティーがあります。

- format - 日付を表示するための日付フォーマットを指定します。値には、y、M、またはdの任意のフォーマット文字を使用する、Java形式の日付フォーマット文字列を指定できます。また、ISOフォーマット (yyyy-MM-dd) を指定する値 iso、またはロケールを反映したフォーマット (ロケール用の Java デフォルト) を指定する値 local も指定できます。設定しなかった場合は、「MM/dd/yyyy」のフォーマットが適用されます。
- multiField - 日付要素ごとに、独立した入力フィールドを表示するかどうかを指定します。値を設定しなかった場合、または **false** に設定した場合は、適切にフォーマットされた日付テキストを入力するための1つのテキストフィールドが使用されます。
- value - カレンダーで、現在の日付として強調表示する日付を指定します。日付は、Date オブジェクトまたは String オブジェクトのいずれかから解析できます。

例

```
<Field name='ExpireDate'>
    <Display class='DatePicker'>
        <Property name='title' value='Set Password Expire date' />
        <Property name='format' value='iso' />
    </Display>
</Field>
```

```
</Display>
</Field>
```

FileUpload

ユーザーがファイルを選択し、それをサーバーにアップロードするためのテキストフィールドと「参照」ボタンを表示します。ファイル(ユーザーオブジェクトや設定オブジェクトなど)から **Identity Manager** にデータをインポートするときは、このコンポーネントを使用します。このコンポーネントは、Text コンポーネントがサポートするすべてのプロパティをサポートします。

Html

このコンポーネントを使用することで、HTML ページに格納されているフォームフィールドやその他のコンポーネントに、JavaScript など任意の HTML マークアップを挿入できます。

このコンポーネントには、html という 1 つのプロパティがあります。これは、ページに表示する文字列の指定に使用されます。

例

```
<Display class='Html'>
  <Property name='html'>
    <concat>
      <s><![CDATA[<div class="DashAlrtMsgTxt">]]></s>
      <ref>loginWarning</ref>
      <s><![CDATA[&nbsp;<a href='']]></s>
      <s>user/changePassword.jsp</s>
      <s><![CDATA['>]]></s>
      <message name='UI_USER_MAIN_CLICK_HERE_INTRO' />
      <s><![CDATA[</a>]]></s>
      <message name='UI_USER_MAIN_CLICK_HERE_REMAINDER' />
      <s><![CDATA[</div>]]></s>
    </concat>
  </Property>
</Display>
```


HtmlPage

ここでは、ルート HTML ページについて説明します。このコンポーネントには、任意の HTML とブラウザ JavaScript を含めることができます。次のプロパティがあります。

- `commentScripts` - JavaScript 用に出力される `<script>` タグを、コメント内に入れるかどうかを指定します。
- `title` - ページのタイトルを指定します。String または Message のデータ型でも指定できますが、通常は String として指定されます。
- `postUrl` - メインフォームの送信時に Identity Manager が送信する URL を指定します。
- `messages` - 表示する情報メッセージを指定します。
- `comments` - ページに含める特別なコメントを指定します。このプロパティは、通常、GenericEditForm および FormConverter メソッドで例外が検出された場合に使用されます。
- `focussedFieldName` - 最初に選択されるフィールドの名前を指定します。通常は NULL に設定されます。このプロパティの値は、最初のテキストフィールドとして、またはテキストフィールドが存在しない場合に最初のフィールドとして計算されます。
- `activeControl` - 最後にアクティブであると認識されるフォームフィールドの名前を指定します (String)。

InlineAlert

エラー、警告、成功、または情報のアラートボックスを表示します。このコンポーネントは、通常はページの上部に配置されます。タイプが InlineAlert\$AlertItem の子コンポーネントを定義することで、1 つのアラートボックスに複数のアラートを表示できます。

この表示コンポーネントには、次のプロパティがあります。

- `alertType` - 表示するアラートの種類を指定します。このプロパティは、適用するスタイルと使用する画像を決定します。指定できる値は、`error` (エラー)、`warning` (警告)、`success` (成功)、および `info` (情報) です。このプロパティのデフォルト値は `info` です。このプロパティは、InlineAlert でのみ有効です。
- `header` - アラートボックスに表示するタイトルを指定します。これは、文字列またはメッセージのいずれかのオブジェクトです。このプロパティは、InlineAlert または InlineAlert\$AlertItem で有効です。
- `value` - 表示するアラートメッセージを指定します。この値は、文字列またはメッセージのいずれかのオブジェクトです。このプロパティは、InlineAlert または InlineAlert\$AlertItem で有効です。

- `linkURL` - アラートの下部に表示するオプション **URL** を指定します。このプロパティは、`InlineAlert` または `InlineAlert$AlertItem` で有効です。
- `linkText` - `linkURL` のテキストを指定します。これは、文字列またはメッセージのいずれかのオブジェクトです。このプロパティは、`InlineAlert` または `InlineAlert$AlertItem` で有効です。
- `linkTitle` - `linkURL` のタイトルを指定します。これは、文字列またはメッセージのいずれかのオブジェクトです。このプロパティは、`InlineAlert` または `InlineAlert$AlertItem` で有効です。

単一のアラートメッセージの例

```
<Field>
  <Display class='InlineAlert'>
    <Property name='alertType' value='warning' />

    <Property name='header' value='Data not Saved' />
    <Property name='value' value='The data entered is not yet saved.
      Please click Save to save the information.' />

  </Display>
</Field>
```

複数のアラートメッセージの例

`alertType` は、`InlineAlert` プロパティ内のみに定義します。その他のプロパティは `InlineAlert$AlertItems` に定義できます。

```

<Field>
  <Display class='InlineAlert'>
    <Property name='alertType' value='error' />
  </Display>
</Field>
  <Display class='InlineAlert$AlertItem'>
    <Property name='header' value='Server Unreachable' />
    <Property name='value' value='The specified server could not
    be contacted. Please view the logs for more information.' />
    <Property name='linkURL' value='viewLogs.jsp' />
    <Property name='linkText' value='View logs' />
    <Property name='linkTitle' value='Open a new window with
    the server logs' />
  </Display>
</Field>
<Field>
  <Display class='InlineAlert$AlertItem'>
    <Property name='header' value='Invalid IP Address' />
    <Property name='value' value='The IP address entered is
    in an invalid subnet. Please use the 192.168.0.x subnet.' />
  </Display>
</Field>
</Field>

```

JavaScript

ページに事前にフォーマットされている **JavaScript** を挿入するときに使用されます。これは、コンポーネントで `onClick` または `onChange` プロパティを使用し、カスタム **JavaScript** 関数を呼び出す場合に便利です。

`name` プロパティの設定は必須ではありませんが、XML フォームからコンポーネントを作成する場合は、設定するようにしてください。フィールドをループさせたり、含めたりする機能などを使用するときに、同じスクリプトを含む複数の **JavaScript** コンポーネントをページに追加できます。同じ名前の **JavaScript** コンポーネントは、HTML の生成時に 1 回だけ取り込まれます。

例

```
<Display class='Javascript'>
  <Property name='script'>
    <String>
function setTextFromSelect(sel, textFieldName) {
  if ( sel == null || sel.inchange ) return;
  sel.inchange = true;
    var textField = sel.form.elements[textFieldName];
  if ( textField == null ) return;
    textField.value = sel.value;
  sel.selectedIndex = 0;
  sel.inchange = false;
} // setTextFromSelect(sel, textFieldName)
    </String>
  </Property>
  <Property name='noNewRow' value='true' />
</Display>
```

コンポーネントには、JavaScript テキストを格納できる script という拡張プロパティがあります。

Label

テキストの文字列を表示します。

この表示コンポーネントには、次のプロパティがあります。

- value - 表示するテキストを定義します。この値は、文字列、または文字列のリストのいずれかです。値がリストの場合は、リストに含まれる各文字列が別々の行に表示されます。
- leftPad - ラベルの左に挿入する空白文字の数を指定します。
- pad - ラベルの左右に挿入する空白文字の数を指定します。
- rightPad - ラベルの右に挿入する空白文字の数を指定します。

注	空白文字の数を指定しない場合は、デフォルト設定の leftPad=2、rightPad=2 が適用されます。

```
<Field>
  <Display class='Label'>
    <Property name='title' value='Account ID' />
    <Property name='value'>
      <ref>waveset.accountId,/ref>
    </Property>
  </Display>
</Field>
```

- font - フォントスタイルを指定します。指定できる値は、Identity Manager のインストールディレクトリ内の styles/style.css ファイルに定義されているいずれかのスタイル名です。
- color - ラベルの色を指定します。色の値の指定には、HTML の標準色フォーマット (#xxxxxx) を使用します。

Link

ページにリンクを配置します。

次のプロパティがあります。

- URL - ターゲット URL (Uniform Resource Locator) を指定します。
- imageURL - (省略可能) リンクの右に表示されるアイコンまたは画像に URL を指定します。
- imageURL2 - (省略可能) 最初の画像の右に表示されるアイコンまたは画像に URL を指定します。
- hoverText - マウスイカーソルを最初の画像または 2 番目の画像に合わせたときに表示されるテキストを指定します。
- id - (省略可能) リンクの id クエリー引数として含められる値を指定します。
- arguments - (省略可能) クエリー引数として含められる名前と値のペアのセットを指定します。

- extraURL-(省略可能) ベース URL と引数の後に含められる、追加の URL フラグメントを指定します。

例

```
<Field>
  <Display class='Link'>
    <Property name='name' value='Request
      Group Access' />
    <Property name='URL'
      value='user/processLaunch.jsp?newView=true'>
    <Property name='id' value='Group Request
      Process' />
  </Display>
</Field>
```

注	Link コンポーネントは、<map> 要素を使用して名前と値のペアを渡すことができる、フォーム内の唯一の場所です。次の例の <map> 要素には、 String をブール型の値に、また、 String を List にマッピングするためのペアがいくつか含まれています。

```
<invoke class='com.waveset.ui.FormUtil'  
name='getOrganizationsDisplayNames'>  
  <ref>:display.session</ref>  
  <map>  
    <s>filterVirtual</s>  
    <o><Boolean>true</Boolean></o>  
    <s>current</s>  
    <list>  
      <ref>original.orgParentName</ref>  
    </list>  
    <s>excluded</s>  
    <list><ref>orgName</ref></list>  
  </map>  
</invoke>
```

LinkForm

メニューと同様に、リンクの箇条書きリストを表示します。

NameValueTable

このコンポーネントは、名前と値のペアを 2 列の簡易テーブルに表示するときに使用されます。このコンポーネントは、含まれるデータを直接表示します。

データはいくつかの形式で指定できます。

- フラットリスト - リストには、要素 0 が名前、要素 1 が値、要素 2 が名前のような形式で、名前と値のペアが含まれます。
- マップ - マップ内のエントリは、アルファベット順に出力されます。
- **GenericObject** - オブジェクトはフラット化され、マップとして出力されます。

プロパティーには、true に設定した場合に値を含まない行を非表示にする `_hideEmptyRows` があります。

MultiSelect

複数の選択項目を持つテキストボックスを表示します。これは、1 つのボックスに含まれる定義済みの値セットを選択ボックスに移動できる、2 つの部分から構成されるオブジェクトです。左のボックスの値は `allowedValues` プロパティーで指定します。多くの場合、この値は `FormUtil.getResources` などの **Java** メソッドを呼び出すことで動的に取得されます。複数選択ボックスの右側部分に表示される値には、フィールド名によって特定される関連ビュー属性の現在値が適用されます。

この 2 部構成のオブジェクトのフォームタイトルは、`availabletitle` プロパティーと `selectedtitle` プロパティーによって設定されます。

`MultiSelect` コンポーネントがアプレットを使用しないように設定するときは、`noApplet` プロパティーを **true** に設定します。

注

Safari ブラウザが稼動するシステムで **Identity Manager** を実行する場合は、`MultiSelect` コンポーネントを使用するすべてのフォームで、`noApplet` オプションの設定をカスタマイズしてください。このオプションを次のように設定します。

```
<Display class='MultiSelect'>
    <Property name='noApplet' value='true' />
    ...
```

この表示コンポーネントには、次のプロパティがあります。

- `availableTitle` - 表示されるボックスのタイトルを指定します。
- `selectedTitle` - 選択されたボックスのタイトルを指定します。
- `ordered` - テキストボックスの項目リスト内で、選択している項目を上下に移動できることを定義します。値を `true` に設定すると、選択している項目を上下に移動させるための追加ボタンが表示されます。
- `allowedValues` - 複数選択ボックスの左側に関連付ける値を指定します。この値は、必ず文字列のリストとして指定してください。**注:** このボックスへの値の取り込みには `<Constraints>` 要素を使用できますが、この方法はお勧めしません。
- `sorted` - 両方のボックスの値をアルファベット順にソートすることを指定します。
- `noApplet - MultiSelect` コンポーネントがアプレットとともに実装されるか、標準 HTML 選択ボックスのペアとともに実装されるかを指定します。デフォルトでは、アプレットを使用するように設定されます。これは、値の長いリストを処理する場合に効果的です。**Safari** ブラウザが稼動するシステムでのこのオプションの使用については、前述の注記を参照してください。
- `typeSelectThreshold - noApplet` プロパティが `true` に設定されている場合にのみ適用されます。このコンポーネントは、`allowedValue` リストの下に先行入力選択ボックスを表示するかどうかを制御します。左の選択ボックスのエントリ数が、このプロパティによって定義されるしきい値に達すると、選択ボックスの下に追加のテキスト入力フィールドが表示されます。テキストフィールドに文字を入力するときに選択ボックスがスクロールされ、入力した文字と一致するエントリが存在する場合に、それが表示されます。たとえば、**w** と入力すると、名前が **w** から始まる最初のエントリまで、選択ボックスがスクロールされます。
- `width` - 選択しているボックスの幅を、ピクセル単位で指定します。デフォルト値は、150 です。
- `height` - 選択しているボックスの高さを、ピクセル単位で指定します。デフォルト値は、400 です。

例

```
<Field name='accounts[LDAP].LDAPDept' type='string'>
  <Display class='MultiSelect' action='true'>
    <Property name='title' value='LDAP Department' />
  </Display>
  <Constraints>
    <O>
      <List>
        <String>Sales</String>
        <String>Marketing</String>
        <String>International Sales</String>
      </List>
    </O>
  </Constraints>
</Field>
```

Radio

1 つまたは複数のラジオボタンの水平リストを表示します。ユーザーが一度に選択できるラジオボタンは1つだけです。コンポーネントの値が **NULL** の場合、または許可される値のいずれとも一致しない場合は、ボタンは選択されません。

この表示コンポーネントには、次のプロパティがあります。

- **title** - すべてのラジオボタンのタイトルを指定します。
- **labels** - ボタンラベルの代替リストを指定します。labels リストの長さは、allowedValues リストの値と同じ長さにしてください。代替ラベルは、値が暗号化されている場合に使用されます。たとえば、**H**、**M**、**S** などの文字コードを値として指定できますが、これは時、分、および秒のボタンラベルの識別に使用されるプロパティです。
- **allowedValues** - 各ボタンに割り当てる値を指定します。この値は、必ず文字列のリストとして指定してください。
- **value** - ボタンの値を指定します。この値は1つの文字列を受け付けます。設定しない場合は、ラベルと同じ値が適用されます。

例

```
<Field name='attributes.accountLockExpiry.unit'>
  <Display class='Radio'>
    <Property name='noNewRow' value='true' />
    <Property name='labels'>
      <List>
        <String>UI_TASKS_XML_SCHED_MINUTES</String>
        <String>UI_TASKS_XML_SCHED_HOURS</String>
        <String>UI_TASKS_XML_SCHED_DAYS</String>
        <String>UI_TASKS_XML_SCHED_WEEKS</String>
        <String>UI_TASKS_XML_SCHED_MONTHS</String>
      </List>
    </Property>
    <Property name='allowedValues'>
      <List>
        <String>minutes</String>
        <String>hours</String>
        <String>days</String>
        <String>weeks</String>
        <String>months</String>
      </List>
    </Property>
  </Display>
</Field>
```

SectionHead

text プロパティの値によって定義される、新しいセクション見出しを表示します。これは Label クラスを拡張したもので、font プロパティに大きな太字のテキストとなるスタイルを設定します。また、pad プロパティはゼロに設定され、デフォルトの 2 文字分の隙間は取り除かれます。長いフォームを、目立つラベルで複数のセクションに分割する場合に使用されます。

この表示コンポーネントのプロパティは、表示するテキストを指定する text のみです。

例

```
<Field>

  <Display class='SectionHead'>

    <Property name='text' value ='Calculated Fields' />

  </Display>

</Field>
```

Select

1 つの項目を選択できるリストボックスを表示します。このリストボックスの値は、allowedValues プロパティで指定します。

この表示コンポーネントには、次のプロパティがあります。

- allowedValues - リストボックスに表示される、選択できる値のリストを表示します。
- allowedOthers - これを設定すると、allowedValues リストに指定されていない初期値が許容され、リストに追加されます。
- autoSelect - true に設定すると、フィールドの初期値が NULL の場合に、allowedValues リストの最初の値が自動的に選択されます。
- multiple - true に設定すると、複数の値を選択できるようになります。
- nullLabel - どの値も選択していない場合に、リストボックスの最上部に表示するテキストを指定します。
- optionGroupMap - セレクタが、<optgroup> タグを使用して、グループ内のオプションを表示できるようになります。マップのキーがグループラベルとなり、要素が選択可能なリスト項目になるように、マップをフォーマットしてください。値を表示するには、それらの値が allowedValues のメンバーである必要があります。
- size - (省略可能) 表示する行の最大数を指定します。行数がこの値を超える場合は、スクロールバーが追加表示されます。

- `sorted=true` に設定すると、リストの値がソートされます。
- `valueMap` - **raw** 値と表示値をマップします。

このコンポーネントは、`command` プロパティと `onChange` プロパティをサポートします。

例

```
<Field name='city' type='string'>
  <Display class='Select'>
    <Property name='title' value='City' />
    <Property name='allowedValues'>
      <List>
        <String>Austin</String>
        <String>Portland</String>
        <String>New York</String>
      </List>
    </Property>
  </Display>
</Field>
```

Text

通常のテキスト入力ボックスを表示します。

この表示コンポーネントでよく使用されるプロパティは次のとおりです。

- `size` - テキスト入力ボックスに表示される文字数を指定します。入力ボックスのサイズは、ボックス内のテキストの長さに応じて再計算されます。
- `notrim` - **HTML** フォームから送信されるテキストをトリミングするかどうかを指定します。`true` に設定すると、空白文字が削除されません。空白文字を維持するには、このオプションを設定します。
- `noTranslate` - **true** に設定すると、メッセージキーの値が代替値としてではなく、そのまま表示されます。デフォルトは **false** です。
- `maxLength` - テキストボックスで編集できる文字列の最大長を指定します。

- `multiValued=true` に設定すると、値を追加、削除できるように、「追加」ボタンと「削除」ボタンを持つテキストボックスが表示されます。
- `secret` - 入力したテキストの代わりに、アスタリスク (****) が表示されます。このオプションは、パスワードフィールドで頻繁に利用されます。
- `readOnly` - 読み取り専用のテキストを表示します。ユーザーがこのテキストを編集することはできません。このプロパティは、たとえば、管理者がユーザーアカウントを作成または編集するときに、リソース属性に関する必要情報を表示できるようにする場合などに使用されます。
- `submitOnEnter` - このプロパティが設定され、Text フィールドが選択されると、ユーザーが **Enter** キーを押したときに、プロパティ値として指定されているコマンドを使用してフォームが送信されます。次の例では、ユーザーが「保存」ボタンをクリックした場合と同様に、フォームが送信されます。

例

```
<Field name='variables.identityID'>
  <Display class='Text'>
    <Property name='required'>
      <Boolean>true</Boolean>
    </Property>
    <Property name='title' value='Identity ID' />
    <Property name='size' value='32' />
    <Property name='maxLength' value='128' />
    <Property name='submitOnEnter' value='Save' />
  </Display>
</Field>
```

TextArea

複数行のテキスト入力ボックスを表示します。

この表示コンポーネントには、次のプロパティがあります。

- `rows` - テキスト領域の行数を指定します (**Integer**)。
- `columns` - テキスト領域の列数を指定します (**Integer**)。

- `readOnly` - テキスト入力ボックスに、読み取り専用テキストを表示します。`true` に設定すると、このコンポーネントの境界線は表示されなくなります (ブール型)。
- `format - setValue()` の動作を制御し、`getPostData()` から返されるオブジェクトのタイプを決定します (**String**)。
- `sorted - true` に設定すると、テキスト領域の行をソートできるようになります。この機能は、自由形式のテキストではなく、選択項目のリストをテキスト領域に表示する場合に便利です (ブール型)。
- `noTrim` - **HTML** フォームから送信されるテキストをトリミングするかどうかを指定します。デフォルトの設定では、空白文字は削除されます。空白文字を維持するには、この値を `true` に設定します。

例

5 つの表示行を持ち、70 文字で行を折り返すテキストボックスを表示するには、次のように設定します。

```
<Field name='Description'>
  <Display class='TextArea'>
    <Property name='rows' value='5' />
    <Property name='columns' value='70' />
  </Display>
</Field>
```

定義されている表示行を超えてテキストを入力すると、テキスト領域にスクロールバーが追加表示されます。

フォームとプロセスのマッピング

この付録は、Identity Manager で使用されるフォームおよびワークフロープロセスと、それに対応するシステム名の一覧を示しています。Identity Manager の統合開発環境 (Identity Manager IDE) については、『Identity Manager 配備ツール』の「Identity Manager IDE の使用」を参照してください。

フォームのマッピング

次の表は、各フォームのシステム名と、製品インタフェースに表示される名前を示しています。

「システム名」列には、フォームのシステム名が示されます。

「マップ対象」列には、Identity Manager IDE でのフォームの識別に使用され、Identity Manager の「デバッグ」ページに表示される名前が示されます。

表 A-1 フォームのシステム名と製品インタフェース名

システム名	マップ対象
accessApprovalList	アクセス承認リスト
accessReviewDeleteConfirmation	アクセスレビュー削除確認フォーム
accessReviewDashboard	アクセスレビューダッシュボード
accessReviewSummary	アクセスレビュー概要
accessReviewDetail	アクセスレビュー詳細
accessScanForm	アクセススキャンフォーム
LDAP ChangeLog ActiveSync グループ作成フォーム	LDAP グループ作成フォーム
LDAP ChangeLog ActiveSync 組織作成フォーム	LDAP 組織作成フォーム
LDAP ChangeLog ActiveSync 組織単位作成フォーム	LDAP 組織単位作成フォーム

表 A-1 フォームのシステム名と製品インタフェース名 (続き)

システム名	マップ対象
LDAP ChangeLog ActiveSync 人物作成フォーム	LDAP 人物作成フォーム
LDAP ChangeLog ActiveSync グループ更新フォーム	LDAP グループ更新フォーム
LDAP ChangeLog ActiveSync 組織更新フォーム	LDAP 組織更新フォーム
LDAP ChangeLog ActiveSync 組織単位更新フォーム	LDAP 組織単位更新フォーム
LDAP ChangeLog ActiveSync 人物更新フォーム	LDAP 人物更新フォーム
LDAP Listener ActiveSync グループ作成フォーム	LDAP グループ作成フォーム
LDAP Listener ActiveSync 組織作成フォーム	LDAP 組織作成フォーム
LDAP Listener ActiveSync 組織単位作成フォーム	LDAP 組織単位作成フォーム
LDAP Listener ActiveSync 人物作成フォーム	LDAP 人物作成フォーム
LDAP Listener ActiveSync グループ更新フォーム	LDAP グループ更新フォーム
LDAP Listener ActiveSync 組織更新フォーム	LDAP 組織更新フォーム
LDAP Listener ActiveSync 組織単位更新フォーム	LDAP 組織単位更新フォーム
LDAP Listener ActiveSync 人物更新フォーム	LDAP 人物更新フォーム
UserEntitlementForm	UserEntitlementForm
Windows Active Directory ActiveSync コンテナ作成フォーム	Windows Active Directory コンテナ作成 フォーム
Windows Active Directory ActiveSync グループ作成フォーム	Windows Active Directory グループ作成 フォーム
Windows Active Directory ActiveSync 組織単位作成フォーム	Windows Active Directory 組織単位作成 フォーム
Windows Active Directory ActiveSync ユーザー作成フォーム	Windows Active Directory ユーザー作成 フォーム
Windows Active Directory ActiveSync コンテナ更新フォーム	Windows Active Directory コンテナ更新 フォーム
Windows Active Directory ActiveSync グループ更新フォーム	Windows Active Directory グループ更新 フォーム
Windows Active Directory ActiveSync 組織単位更新フォーム	Windows Active Directory 組織単位更新 フォーム
Windows Active Directory ActiveSync ユーザー更新フォーム	Windows Active Directory ユーザー更新 フォーム
accountOwnerSelection	アカウント所有者選択フォーム

表 A-1 フォームのシステム名と製品インタフェース名 (続き)

システム名	マップ対象
anonymousUserMenu	匿名ユーザーメニュー
changeAnswers	ユーザーの秘密質問の回答変更フォーム
changeCapabilities	ユーザー機能変更フォーム
changeMyPassword	マイパスワード変更フォーム
changePassword	ユーザーパスワード変更フォーム
changePasswordSelection	ユーザー選択フォーム
confirmDeletes	削除確認
deprovisionUser	プロビジョニング解除フォーム
disableUser	無効化フォーム
editArgument	引数編集
editField	フィールド編集
editForm	フォーム編集
editRule	規則編集
enableUser	有効化フォーム
endUserChangePassword	パスワード変更フォーム
endUserForm	エンドユーザーフォーム
endUserLaunchList	エンドユーザー起動リスト
endUserMenu	エンドユーザーメニュー
endUserResetPassword	ユーザーパスワードリセットフォーム
endUserTaskList	エンドユーザータスクリスト
endUserTaskResults	エンドユーザータスク結果
endUserWorkItemEdit	エンドユーザー作業項目編集
endUserWorkItemList	エンドユーザー作業項目リスト
findAccountOwner	アカウント所有者検索フォーム
findUser	ユーザー検索フォーム
findUserResults	ユーザー結果検索フォーム
listForms	リストフォーム
listRules	リスト規則
loadForm	デフォルトユーザーフォーム

表 A-1 フォームのシステム名と製品インタフェース名 (続き)

システム名	マップ対象
loginChangePassword	有効期限切れログインフォーム
loginResetPassword	ユーザーパスワードリセットフォーム
renameUser	ユーザー名変更フォーム
reprovisionForm	デフォルトユーザーフォーム
resetPassword	ユーザーパスワードリセットフォーム
resetPasswordSelection	ユーザー選択フォーム
selfDiscovery	自己検索
userForm	タブ付きユーザーフォーム
viewUserForm	タブ付きビューユーザーフォーム
enableUser	有効化フォーム
endUserChangePassword	パスワード変更フォーム
endUserForm	エンドユーザーフォーム

プロセスのマッピング

「システム名」列には、プロセスのシステム名が示されます。

「マップ対象」列には、Identity Manager IDE でのプロセスの識別に使用され、Identity Manager の「デバッグ」ページに表示される名前が示されます。

表 A-2 プロセスのシステム名と製品インタフェース名

システム名	マップ対象
abortAccessReview	アクセスレビューレポート
accessReviewScan	アクセススキャン
changeResourceAccountPassword	アカウントリソースパスワード変更
changeUserPassword	ユーザーパスワード変更
createResourceGroup	グループリソース作成
createResourceObject	オブジェクトリソース作成
createResourceOrganization	組織リソース作成
createResourceOrganizationalUnit	組織単位リソース作成

表 A-2 プロセスのシステム名と製品インタフェース名 (続き)

システム名	マップ対象
createResourcePerson	人物リソース作成
createResourceUser	ユーザーリソース作成
createUser	ユーザー作成
deleteAccount	アカウントリソース削除
deleteResourceGroup	グループリソース削除
deleteResourceObject	オブジェクトリソース削除
deleteResourceOrganization	組織リソース削除
deleteResourceOrganizationalUnit	組織単位リソース削除
deleteResourcePerson	人物リソース削除
deleteResourceUser	ユーザーリソース削除
deleteUser	ユーザー削除
disableUser	ユーザー無効化
enableUser	
handleNativeChangeToAccountAttributes	
lockUser	ユーザーロック
manageResource	リソース管理
manageRole	規則管理
questionLogin	質問ログイン
renameUser	ユーザー名変更
resetUserPassword	ユーザーパスワードリセット
unlinkResourceAccountsFromUser	ユーザーとのアカウントリソースリンク解除
unlockUser	ユーザーロック解除
updateResourceGroup	グループリソース更新
updateResourceObject	オブジェクトリソース更新
updateResourceOrganization	組織リソース更新
updateResourceOrganizationalUnit	組織単位リソース更新
updateResourcePerson	人物リソース更新

索引

A

accountInfo 属性 311
accounts 属性 304
Action ワークフローコンポーネント 23
Activity ワークフローコンポーネント 23
addDeferredTask ワークフローサービス 49
add 関数 432
align 表示コンポーネント 500
allowedValues 表示コンポーネント 496
and 関数 434
AND 結合 23
AND 分割 23
Anonymous User Menu Form 109
appendAll 関数 449
append 関数 448
approveProvision ワークフローサービス 79
attrName 38
auditableAttributesList 38
auditNativeChangeToAccountAttributes ワークフローサービス 80
audit ワークフローサービス 51
authenticateUserCredentials ワークフローサービス 82
authorize ワークフローサービス 53, 55

B

BackLink 表示コンポーネント 501
block 関数 457
BorderedPanel 表示コンポーネント 486
break 関数 458
buildDns メソッド 240
buildDn メソッド 239
bulkReprovision ワークフローサービス 81
ButtonRow 表示コンポーネント 486
Button 表示コンポーネント 501

C

callResourceMethod メソッド 238
call 関数 463
changeResourceAccountPassword ワークフローサービス 82
Checkbox 表示コンポーネント 502
checkDeProvision ワークフローサービス 83
checkinObject ワークフローサービス 56
checkinView メソッド 360
checkinView ワークフローサービス 56
checkoutObject ワークフローサービス 57
checkoutView ワークフローサービス 58
checkStringQualityPolicy メソッド 243
cleanupResult ワークフローサービス 83

D

cmp 関数 [434](#)
colspan 表示コンポーネント [500](#)
command 表示コンポーネント [498](#)
concat 関数 [442](#)
cond 関数 [459](#)
containsAll 関数 [450](#)
containsAny 関数 [451](#)
contains 関数 [450](#)
controlsAtLeastOneOrganization メソッド [244](#)
createResourceObject ワークフローサービス [84](#)
createView メソッド [360](#)
createView ワークフローサービス [60](#)

D

DatePicker 表示コンポーネント [503](#)
defarg 関数 [463](#)
Default
要素 [142](#)
deferred 属性 [319, 411](#)
defun 関数 [463](#)
defvar 関数 [462](#)
deleteResourceAccount ワークフローサービス [85](#)
deleteResourceObject ワークフローサービス [85](#)
deleteUser ワークフローサービス [87](#)
deProvision ワークフローサービス [86](#)
Derivation ステートメント [190](#)
Derivation 要素、フィールド [143](#)
disableUser ワークフローサービス [61](#)
Disable 要素、フィールド [141](#)
disable ワークフローサービス [88](#)
display 属性 [317](#)
div 関数 [432](#)
DN 文字列、作成 [201](#)
dolist 関数 [459](#)
downcase 関数 [442](#)
Dynamic Tabbed User Form [222](#)

E

EditForm 表示コンポーネント [486](#)
enableUser ワークフローサービス [62](#)
enable ワークフローサービス [89](#)
eq 関数 [435](#)
Expansion ステートメント [190](#)
Expansion 要素、フィールド [144](#)

F

FileUpload 表示コンポーネント [504](#)
filterdup 関数 [451](#)
filternull 関数 [452](#)
findUser ワークフローサービス [63](#)
FormUtil メソッド [184, 216](#)
オブジェクトタイプによるリストのフィルタリング [272](#)

G

GenericObject クラス [288, 289, 293](#)
getApprovals ワークフローサービス [90](#)
getObjectNames メソッド [245](#)
getObject メソッド [245](#)
getObject ワークフローサービス [64](#)
getObj 関数 [466](#)
getOrganizationsDisplayNames メソッド [247](#)
getProperty ワークフローサービス [65](#)
getResourceObjects メソッド [252](#)
getResources メソッド [249](#)
getRoles メソッド [255](#)
getSubordinates メソッド [258](#)
getUnassignedApplications メソッド [256](#)
getUnassignedResources メソッド [260](#)
getUsers メソッド [261](#)
getViewForm ワークフローサービス [67, 68](#)

getView ワークフローサービス 66

get 関数 452, 466

global 属性 309

gte 関数 436

gt 関数 436

GUID 属性 411

H

hasCapabilities メソッド 268

hasCapability メソッド 267

help

表示コンポーネント 497

プロパティ 211

helpKey プロパティ 211

HTML 表示コンポーネント、「表示コンポーネント」を参照

I

Identity Manager

XPRESS との統合 419

オブジェクトワークフロー 24

パラメータ 382

indexOf 関数 442, 453

InlineAlert 表示コンポーネント 505

inlineHelp 表示コンポーネント 498

insert 関数 453

instanceOf 関数 469

invoke 関数 470

isFalse 関数 436

isNull 関数 437

isTrue 関数 437

i 関数 430

J

Java

クラス、インスタンスとしての HTML 表示コンポーネント 484

クラス、式の最適化 155

式 470

メソッド、ワークフローアクションからの呼び出し 426

JavaScript

式 470

表示コンポーネント 507

フォームへの挿入 216

L

Label 表示コンポーネント 508

length 関数 454

lh コマンド、XML 構文の検証 427

LinkForm 表示コンポーネント 512

Link 表示コンポーネント 509

listResourceObjects メソッド 262

list 関数 431

location 表示コンポーネント 497

lockOrUnlock ワークフローサービス 90

logattr テーブル 38

lte 関数 438

ltrim 関数 442

lt 関数 438

M

match 関数 443

mod 関数 432

MultiSelect 表示コンポーネント 512

mult 関数 432

N

NameValueTable 表示コンポーネント [512](#)
 name 表示コンポーネント [494](#)
 ncmp 関数 [439](#)
 neq 関数 [439](#)
 new 関数 [470](#)
 noNewRow 表示コンポーネント [497](#)
 notify ワークフローサービス [91](#)
 notnull 関数 [441](#)
 not 関数 [440](#)
 nowrap 表示コンポーネント [500](#)
 null 関数 [431](#)

O

onChange 表示コンポーネント [499](#)
 onClick 表示コンポーネント [499](#)
 or 関数 [440](#)
 OR 結合 [23](#)
 OR 分割 [23](#)

P

pad 関数 [443](#)
 Panel 表示コンポーネント [487](#)
 password ユーザービュー属性 [301](#)
 primaryKey 表示コンポーネント [496](#)
 print 関数 [472](#)
 provision ワークフローサービス [91](#)
 putmap 関数 [467](#)

Q

queryObjectNames ワークフローサービス [70](#)
 queryObjects ワークフローサービス [71](#)

queryReferencingRoles ワークフローサービス [72](#)
 questionLock ワークフローサービス [92](#)

R

Radio 表示コンポーネント [514](#)
 refreshView ワークフローサービス [72](#)
 ref 関数 [462](#)
 reject ワークフローサービス [92](#)
 removeAll 関数 [455](#)
 removeDeferred ワークフローサービス [73](#)
 removeProperty ワークフローサービス [74](#)
 remove 関数 [454](#)
 reProvision ワークフローサービス [94](#)
 required 表示コンポーネント [496](#)
 Row 表示コンポーネント [490](#)
 rtrim 関数 [444](#)
 rule 関数 [464](#)
 runResourceAction ワークフローサービス [94](#)

S

scheduler [390](#)
 scopingOrg オプション [269](#)
 script 関数 [471](#)
 SectionHead 表示コンポーネント [516](#)
 Selector 表示コンポーネント [488](#)
 select 関数 [460](#)
 Select 表示コンポーネント [516](#)
 setlist 関数 [468](#)
 setProperty ワークフローサービス [75](#)
 setvar 関数 [468](#)
 set 関数 [456](#), [465](#), [467](#)
 SimpleTable 表示コンポーネント [489](#)
 Solaris
 サポート [15](#)
 パッチ [15](#)

SortingTable 表示コンポーネント 490

split 関数 444

substr 関数 445

sub 関数 433

s 関数 431

T

Tabbed User Form 109

table タグ 486

TabPanel 表示コンポーネント 489

TaskDefinition オブジェクト

概要 19

パラメータ 19

TaskInstance オブジェクト 21

削除 22

testObject メソッド 266

testUser メソッド 267

TextArea 表示コンポーネント 518

Text 表示コンポーネント 517

title 表示コンポーネント 494

trace 関数 472

Transition ワークフローコンポーネント 23

trim 関数 447

U

unlockObject ワークフローサービス 75

unlockView ワークフローサービス 76

upcase 関数 447

update ワークフローサービス 96

V

Validation ステートメント 195

Validation 要素、フィールド 145

value 表示コンポーネント 495

W

waveset 属性

accountId 298

applications 298

attributes 298

correlationKey 298

createDate 299

creator 298

disabled 299

email 299

exclusions 299

id 300

lastModDate 300

lastModifier 300

locked 300

lockExpiry 300

organization 300

original 301

passwordExpiry 302

passwordExpiryWarning 302

questions 302

resources 303

roles 304

頻繁に使用される 297

while 関数 461

width 表示コンポーネント 500

WizardPanel 表示コンポーネント 491

workflowAuditAttrConds リスト、定義 38

workflowAuditAttrConds 属性 38

WorkItem

オブジェクト 21

管理表示機能の制限 27

タイプ 26

WSUser オブジェクト 296

X

XML

XPRESS の構文 [417, 418](#)
構文、検証 [427](#)
フォームの構造 [125](#)
XML オブジェクト言語
XPRESS [477](#)
プロパティ値の指定 [139](#)
マップオブジェクト [480](#)
リスト [479](#)
XPRESS
Derivation 要素と Expansion 要素 [190](#)
Identity Manager との統合 [419](#)
Java/Javascript 式 [470](#)
Java メソッドの呼び出し [426](#)
XML オブジェクト [478](#)
XML オブジェクト言語 [477](#)
値コンストラクタ [430](#)
値の生成 [424](#)
値の導出 [423](#)
演算式 [432](#)
オブジェクト式 [466](#)
概要 [417](#)
関数 [430](#)
関数式 [462](#)
繰り返し式 [457](#)
検証 [427](#)
検証式 [472](#)
構文 [417, 418](#)
式 [419](#)
条件式 [457](#)
データ型 [473](#)
デバッグ式 [472](#)
デフォルト値 [421](#)
トレース [428](#)
表記 [418](#)
フィールドの可視性 [420](#)
フォーム内での使用 [208](#)
ブロック式 [457](#)
変数式 [462](#)
マップオブジェクト [480](#)
文字列式 [442](#)
リスト [479](#)
リスト式 [448](#)

論理式 [434](#)
ワークフローアクション [426](#)
ワークフローの遷移条件 [425](#)
XPRESS のトレース [428](#)

あ

アイデンティティテンプレート [382](#)
アカウント関連ビュー [322](#)
アクション
手動 [21](#)
アクティビティ
ワークフロータスク [34](#)
値コンストラクタ式 [430](#)

い

委任承認者ビュー [331](#)

う

ウィザードフォーム [217](#)

え

演算式 [432](#)
エンドユーザーフォーム [111](#)
エンドユーザーメニューフォーム [108](#)

お

オブジェクト検索ビュー [341](#)
オブジェクト操作 [466](#)

か

ガイドヘルプ 210

カスタマイズしたフォームの検証 230

カレンダーアイコン、フォームへの追加 186

監査

ワークフロー 37

関数

add 432
 and 434
 append 448
 appendAll 449
 block 457
 break 458
 call 463
 cmp 434
 concat 442
 cond 459
 contains 450
 containsAll 450
 containsAny 451
 defarg 463
 defun 463
 defvar 462
 div 432
 dolist 459
 downcase 442
 eq 435
 filterdup 451
 filternull 452
 get 452, 466
 getObj 466
 gt 436
 gte 436
 i 430
 indexOf 442, 453
 insert 453
 instanceOf 469
 invoke 470
 isFalse 436
 isNull 437
 isTrue 437
 length 454
 list 431
 lt 438
 lte 438

ltrim 442
 match 443
 mod 432
 mult 432
 ncmp 439
 neq 439
 new 470
 not 440
 notnull 441
 null 431
 or 440
 pad 443
 print 472
 putmap 467
 ref 462
 remove 454
 removeAll 455
 rtrim 444
 rule 464
 s 431
 script 471
 select 460
 set 456, 465, 467
 setlist 468
 setvar 468
 split 444
 sub 433
 substr 445
 trace 472
 trim 447
 upcase 447
 while 461
 XPRESS 430

関数定義式 462

管理者ロールビュー 325

き

規則、フォーム内での使用 208

基本コンポーネントクラス 494

基本表示クラス 485

く

く

繰り返し式 [457](#)

グローバル登録 [413](#)

け

結合ワークフロー遷移 [23](#)

権限名 [99](#)

検証式 [427](#), [472](#)

こ

コンテナ [166](#)

コンテナ表示クラス [486](#)

コンテナフィールド [146](#)

コンポーネントクラス [494](#)

さ

再プロビジョニングビュー [373](#)

作業項目

ビュー [289](#), [397](#)

表示と変更 [397](#)

リストビュー [404](#)

サブクラス、コンポーネント [492](#)

サポート

Solaris [15](#)

参照、選択的 [221](#)

し

式 [417](#)

XPRESS [419](#)

検証 [427](#)

時刻計算、有効化 [37](#)

手動アクション [21](#)

WorkItem のタイプ [26](#)

認証のタイプ [26](#)

例 [25](#)

条件式 [457](#)

承認フォーム [111](#)

す

スケーラブルフォーム [220](#), [221](#), [222](#), [225](#)

せ

セクション見出し、フォームへの追加 [186](#)

セッションワークフローサービス [48](#)

設定オブジェクト [22](#)

遷移条件、ワークフロー [425](#)

遷移、ワークフロー [27](#)

選択的な参照 [221](#)

そ

属性

「ビュー属性」も参照

accountInfo [311](#)

accounts [304](#)

deferred [319](#), [411](#)

display [317](#)

global [309](#)

logattr テーブルへの格納 [38](#)

password [301](#)

waveset [296](#)

workflowAuditAttrConds [38](#)

オブジェクト [289](#)

登録 [413](#)

ビューの登録 [413](#)

メタビュー [292](#)

ユーザービュー [292](#)

ワークフロー監査の収集対象 37
 属性の登録 413
 組織ビュー 347
 その他のワークフロー 24

た

タイプ名 97
 タスクスケジュールビュー 390
 タブ付きフォーム 217

ち

チェックボックス、作成 170
 調整状態ビュー 368
 調整ビュー 361
 調整ポリシービュー 362

て

定義
 workflowAuditAttrConds リスト 38
 データ型
 XPRESS 473
 表示コンポーネント 493
 テキストフィールド 166
 デバッグ
 式 472
 ユーザービュー 321
 デフォルト
 フィールド値 421
 ワークフロープロセス 24

に

認証のタイプ、手動アクション 26

は

パス式 288, 290
 パスワードビュー 353
 パスワードフィールドの移動 227
 ハッシュマップ、作成 206

ひ

ビュー
 deferred 属性 411
 アカウント関連ビュー 322
 委任承認者ビュー 331
 オブジェクト検索ビュー 341
 拡張 413
 管理者ロールビュー 325
 再プロビジョンビュー 373
 作業項目ビュー 397
 作業項目リストビュー 404
 説明 284
 組織ビュー 347
 代表的な 286
 タスクスケジュールビュー 390
 調整状態ビュー 368
 調整ビュー 361
 調整ポリシービュー 362
 パス式 290
 パスワードビュー 353
 フォームとの統合 285
 プロセスビュー 358
 プロビジョン解除ビュー 333
 無効化ビュー 337
 有効化ビュー 339
 ユーザー機能変更ビュー 330
 ユーザーの秘密質問の回答変更ビュー 328
 ユーザーパスワードのリセットビュー 376
 ユーザー名変更ビュー 370
 ユーザー、「ユーザービュー」を参照
 リソースオブジェクトビュー 385
 リソースビュー 379
 ロールビュー 387

ロック解除ビュー 394
ワークフローとの統合 286
ビュー属性 112, 285
登録 413
ビューハンドラ 285
表示コンポーネント
align 500
allowedValues 496
BackLink 501
Button 501
Checkbox 502
colspan 500
command 498
DatePicker 503
help 497, 498
Html 504
HtmlPage 505
JavaScript 507
Label 508
Link 509
location 497
MultiSelect 512
name 494
noNewRow 497
nowrap 500
onChange 499
onClick 499
primaryKey 496
Radio 514
required 496
SectionHead 516
Select 516
SimpleTable 489
Text 517
TextArea 518
title 494
value 495
width 500
概要 483
基本クラス 485
基本コンポーネントクラス 494
コンテナクラス 486
サブクラス 492
データ型 493
非表示パラメータ 484

表記上の規則 492
ページプロセッサの要件 484
表示コンポーネントに関するページプロセッサの要件 484

ふ

フィールド、「フォーム、フィールド」を参照
フィールド値の生成 424
フィールド値の導出 423
フォーム
Derivation 規則と Expansion 規則 190
Display 要素 162
Javascript 216
値の計算 207
ウィザード 217
ガイドダンス (ヘルプ) 210
概要 102
カスタマイズ 114
カスタマイズの概要 115
カレンダーアイコン 186
検証 216, 230
構造 125
コンポーネント
概要 127
フッター 132
ヘッダー 127
本文 128
コンポーネントの位置 189
作成に関するガイドライン 154
サンプル 231, 232
使用されるページ 106
スケーラブル 220, 221, 222, 225
セクション見出し 186
タブ付き 217
タブ付きユーザーフォーム 227
動作 111
ハッシュマップ 206
非表示コンポーネント 189
ビューとの統合 285
評価 112

フィールド

- 値の計算 [148](#)
- 値の生成 [424](#)
- 値の導出 [150, 423](#)
- 可視性 [420](#)
- コンポーネント [133](#)
- 再計算 [153](#)
- 式の最適化 [154](#)
- 定義 [134](#)
- デフォルト値の計算 [421](#)
- 名前の定義 [134](#)
- 非表示 [148, 207](#)
- 表示プロパティ [137](#)
- 無効化 [147, 206](#)
- フィールドの参照 [160](#)
- フォーム名にマップされるシステム名 [521](#)
- 別のフォームの参照 [159](#)
- 編集 [105, 162](#)
- 変数の作成 [133](#)
- ユーザー作成 [220](#)
- ユーザービュー [112](#)
- ユーザービューとの統合 [288](#)
- ユーザー編集 [220](#)
- リスト [169](#)
- リソースメソッドの呼び出し [159](#)
- リンクの追加 [189](#)
- 例 [104, 106](#)

- フォームジェネレータ [288](#)
- フォーム内の非表示コンポーネント [189](#)
- フォームの編集 [162](#)
- 複数リソースの編集 [221](#)
- プレフィックス表記 [418](#)
- プロセスビュー [358](#)
- ブロック式 [457](#)
- プロビジョン解除ビュー [333](#)
- プロビジョンワークフローサービス [76](#)
- 分割ワークフロー遷移 [23](#)

へ

- ヘッダー、フォーム [127](#)

ヘルプ

- カタログ [211](#)
- フォームへの追加 [210](#)

編集フィールド [146](#)

変数

- 定義 [462](#)

変数、フォームでの作成 [133](#)

ほ

ボタン

- コマンドの値 [164](#)
- 作成 [162](#)
- 整列 [165](#)
- デフォルト名の無効化 [163](#)
- ラベルの割り当てと変更 [163](#)

ま

マップオブジェクト [480](#)

む

無効化ビュー [337](#)

め

メソッド

- 「FormUtil メソッド」も参照
- buildDn [239](#)
- buildDns [240](#)
- callResourceMethod [238](#)
- checkStringQualityPolicy [243](#)
- controlsAtLeastOneOrganization [244](#)
- getObject [245](#)
- getObjectNames [245](#)
- getOrganizationsDisplayNames [247](#)
- getResourceObjects [252](#)

- getResources 249
- getRoles 255
- getSubordinates 258
- getUnassignedApplications 256
- getUnassignedResources 260
- getUsers 261
- hasCapabilities 268
- hasCapability 267
- listResourceObjects 262
- testObject 266
- testUser 267
- 呼び出しによるリストへのデータの取り込み 196

も

- 文字列操作 442

ゆ

- 有効化
 - 時刻計算 37
- 有効化ビュー 339
- ユーザー機能変更ビュー 330
- ユーザー作成フォーム 220
- ユーザーの秘密質問の回答変更ビュー 328
- ユーザーパスワードのリセットビュー 376
- ユーザービュー
 - metaView 属性 292
 - アカウント関連のユーザービューネームスペース 293
 - アカウントタイプの参照 291
 - 概要 284, 288
 - 属性 292
 - デバッグ 321
 - フォームとの統合 112, 288
 - ワークフローとの統合 289
- ユーザーフォームライブラリ 231, 232
- ユーザー編集フォーム 220
- ユーザー名変更ビュー 370
- ユーザーワークフロー 24

ら

- ラジオボタン、作成 171
- ラベルフィールド、作成 185

り

- リスト
 - XML オブジェクト言語 479
 - XPRESS 479
 - 切り替え 290
 - 計算 291
 - 操作 169
 - 単一選択、作成 171
 - データの取り込み 174
 - 複数選択、作成 172
 - 別の表示値セット 173
 - メソッドの呼び出しによるデータの取り込み 196
- リスト操作 448
- リソース
 - アカウント、フィルタリング 181
 - オブジェクト名 197
 - 固有の登録 414
 - 属性 381
 - オーバーライド 305
 - メソッド、フォームからの呼び出し 159
- リソースオブジェクトビュー 385
- リソーステーブルユーザーフォーム 224
- リソースの差分フェッチ 221
- リソースビュー 379
- リンク、フォームへの追加 189

ろ

- ロールビュー 387
- ロギング、有効化と無効化 230
- ロック解除ビュー 394
- 論理式 434

わ

ワークフロー

「ワークフロープロセス」も参照

Java [426](#)

TaskDefinition オブジェクト [19](#)

 パラメータ [19](#)

アクション [426](#)

アプリケーションの追加 [38](#)

エンジン [22](#)

概要 [18](#)

組み込み変数 [46](#)

コンポーネント [23](#)

手動アクション [21](#)

進行状況の追跡 [35](#)

設定プロパティ [39](#)

説明 [18](#)

遷移

 作成 [27](#)

 条件 [425](#)

タスク [34](#)

ツールボックス

 デフォルトアクティビティ [30](#)

ビューとの統合 [286](#)

ユーザービューとの統合 [289](#)

リポジトリオブジェクト [22](#)

ワークフロー監査

収集される情報 [37](#)

ワークフローサービス

addDeferredTask [49](#)

approveProvision [79](#)

audit [51](#)

auditNativeChangeToAccountAttributes [80](#)

authenticateUserCredentials [82](#)

authorize [53, 55](#)

bulkReprovision [81](#)

changeResourceAccountPassword [82](#)

checkDeProvision [83](#)

checkinObject [56](#)

checkinView [56](#)

checkoutObject [57](#)

checkoutView [58](#)

cleanupResult [83](#)

createResourceObject [84](#)

createView [60](#)

deleteResourceAccount [85](#)

deleteResourceObject [85](#)

deleteUser [87](#)

deProvision [86](#)

disable [88](#)

disableUser [61](#)

enable [89](#)

enableUser [62](#)

findUser [63](#)

getApprovals [90](#)

getObject [64](#)

getProperty [65](#)

getView [66](#)

getViewForm [67, 68](#)

lockOrUnlock [90](#)

notify [91](#)

provision [91](#)

queryObjectNames [70](#)

queryObjects [71](#)

queryReferencingRoles [72](#)

questionLock [92](#)

refreshView [72](#)

reject [92](#)

removeDeferred [73](#)

removeProperty [74](#)

reProvision [94](#)

runResourceAction [94](#)

setProperty [75](#)

unlockObject [75](#)

unlockView [76](#)

update [96](#)

呼び出しの構造 [47](#)

ワークフロープロセス

「ワークフロー」も参照

TaskInstance オブジェクト [21](#)

概要 [19](#)

カスタマイズ [29](#)

更新 [28](#)

デフォルト [24](#)

本稼働環境での編集 [28](#)

ワークフロープロパティの設定 [39](#)

