Service Registry 3.1 管理ガイド

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

Part No: 820–0535
2007 年 2 月

070830@18135

目次

はじめに

この『Service Registry 3.1 管理ガイド』では、Service Registry (「レジストリ」) のインストール後の設定方法やレジストリに付属する管理ツールの使用方法について説明します。また、レジストリデータベースのバックアップや復元など、その他の管理作業についても説明します。

Service Registry は ebXML レジストリです。これは、標準および拡張可能なメタデータで記述されたすべての型の電子的なコンテンツを管理する、連携されたレジストリおよびリポジトリです。サービス指向アーキテクチャー (Service Oriented Architecture、SOA) およびその他のコンテンツとメタデータについて、連携されてセキュリティー保護された情報管理を提供します。ebXML Registry 3.0 および UDDI 3.0 のレジストリプロトコルをサポートします。

対象読者

この『管理ガイド』の対象読者は、レジストリをインストール、アンインストール、および管理する必要のある方、およびレジストリのコンテンツを Web コンソールを使わずに一括作成する必要のある方です。

読者には、使用するオペレーティングシステム (Solaris ™ オペレーティング環境、Linux、または HP-UX) における UNIX ® コマンドシェル環境の基本に習熟していることが期待されます。

お読みになる前に

このマニュアルをお読みになる前に、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』の説明に従ってレジストリをインストールする必要があります。

Service Registry は Sun Java Enterprise System (「Java ES」) のコンポーネントであり、特定のネットワーク上やインターネット環境に分散配置された企業アプリケーションをサポートするソフトウェアインフラストラクチャーです。`http://docs.sun.com/coll/1286.2` にある Java ES のマニュアルを熟読されることをお勧めします。

管理作業の中には、次の各仕様の基本概念に習熟していることが要求されるものもあります。

- 『ebXML Registry Information Model Version 3.0』
- 『ebXML Registry Services and Protocols Version 3.0』

内容の紹介

このマニュアルの内容は次のとおりです。

第 1 章では、Service Registry のインストール後の設定方法やその他の管理作業の実行方法について説明します。

第 2 章では、管理ツールの使用方法について説明します。

Service Registry のマニュアルセット

Service Registry のマニュアルセットは `http://docs.sun.com/coll/1314.2` から入手可能です。Service Registry について学ぶには、次の表に記載されたマニュアルを参照してください。

表 P–1　Service Registry のマニュアル

マニュアルタイトル	内容
『Service Registry 3.1 リリースノート (UNIX 版)』	既知の問題を含む、Service Registry の最新情報を記載しています。
『Service Registry 3.1 管理ガイド』	Service Registry のインストール後の設定方法やレジストリに付属する管理ツールの使用方法について説明します。また、その他の管理作業の実行方法についても説明します。
『Service Registry 3.1 ユーザーズガイド (2006Q4)』	Service Registry の Web コンソールによる Service Registry の検索方法やデータの発行方法について説明します。
『Service Registry 3.1 開発ガイド』	JAXR (Java API for XML Registries) による Service Registry の検索方法やデータの発行方法について説明します。

関連マニュアル

Service Registry をインストールすると、それは Sun Java System Application Server に配備されます。Application Server の管理方法については、『Sun Java System Application Server Enterprise Edition 8.2 管理ガイド』を参照してください。

Java ES のマニュアルセットでは、配備計画やシステムのインストールについて説明しています。システムマニュアルの URL は、`http://docs.sun.com/coll/1286.2` です。Java ES の概要を把握するには、これらのマニュアルを次の表に記載されている順番で参照してください。

表 P–2　Java Enterprise System のマニュアル

マニュアルタイトル	内容
『Sun Java Enterprise System 5 リリースノート (UNIX 版)』 『Sun Java Enterprise System 5 リリースノート (Windows 版)』	既知の問題を含む、Java ES の最新情報を記載しています。さらに、各コンポーネントには、Release Notes Collection (`http://docs.sun.com/coll/1315.2`) に記載されている独自のリリースノートがあります。
『Sun Java Enterprise System 5 技術の概要』	Java ES の技術的、概念的な基礎について概説します。コンポーネント、アーキテクチャー、プロセス、機能についても説明しています。
『Sun Java Enterprise System Deployment Planning Guide 』	Java ES に基づく企業向け配備ソリューションの計画と設計について概説します。配備計画と配備設計に関する基本的な概念や原則について説明し、ソリューションのライフサイクルについて説明し、Java ES に基づくソリューションを計画する際に参考になる高レベルの例や方針を提供します。
『Sun Java Enterprise System 5 インストール計画ガイド』	Java ES 配備のハードウェア、オペレーティングシステム、ネットワークの各側面に対する実装仕様を作成する際に役立つ情報を提供します。コンポーネント間の依存関係など、インストール計画時や設定計画時に解決すべき問題について説明します。
『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』 『Sun Java Enterprise System 5 インストールガイド (Windows 版)』	Java ES のインストール手順について説明します。また、インストール後に各コンポーネントを設定する方法や各コンポーネントが正しく動作しているか確認する方法についても説明します。
『Sun Java Enterprise System 5 インストールリファレンス (UNIX 版)』	設定パラメータに関する追加情報や設定計画時に使用するワークシートを提供するほか、Solaris オペレーティングシステムおよび Linux オペレーティング環境でのデフォルトのディレクトリやポート番号など、各種リファレンス情報の一覧表も提供します。

表 P–2 Java Enterprise System のマニュアル (続き)

マニュアルタイトル	内容
『Sun Java Enterprise System 5 アップグレードガイド (UNIX 版)』	Java ES 5 の以前にインストールしたバージョンからのアップグレード手順について説明します。
『Sun Java Enterprise System 5 Upgrade Guide for Microsoft Windows 』	
『Sun Java Enterprise System 5 監視ガイド (UNIX 版)』	各製品コンポーネントで Monitoring Console を設定する手順、および Monitoring Console を使用してリアルタイムデータを表示して監視規則を作成する手順を説明します。
『Sun Java Enterprise System 用語集』	Java ES のマニュアルで使用されている用語を定義します。

Java ES とそのコンポーネントに関するすべてのマニュアルの URL は、`http://docs.sun.com/prod/entsys.5` です。

デフォルトのパスとファイル名

次の表は、このマニュアルで使用されているデフォルトのパス名とファイル名について説明したものです。

表 P–3 デフォルトのパスとファイル名

プレースホルダ	説明	デフォルト値
ServiceRegistry-base	Service Registry のベースインストールディレクトリを表します。	Solaris OS の場合: `/opt/SUNWsrvc-registry` Linux および HP-UX システムの場合: `/opt/sun/srvc-registry`
RegistryDomain-base	Service Registry 用の Application Server ドメインや Service Registry データベースが格納されるディレクトリを表します。	Solaris OS の場合: `/var/opt/SUNWsrvc-registry` Linux および HP-UX システムの場合: `/var/opt/sun/srvc-registry`
Ant-base	Ant ツールの Java ES バージョンが格納されるディレクトリを表します。	Solaris OS の場合: `/usr/sfw/bin` Linux および HP-UX システムの場合: `/opt/sun/share/bin`

表記上の規則

このマニュアルでは、次のような字体や記号を特別な意味を持つものとして使用します。

表 P–4 表記上の規則

字体または記号	意味	例
AaBbCc123	コマンド名、ファイル名、ディレクトリ名、画面上のコンピュータ出力、コード例を示します。	.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。 machine_name% you have mail.
AaBbCc123	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区別して示します。	machine_name% **su** Password:
aabbcc123	変数を示します。実際に使用する特定の名前または値で置き換えます。	ファイルを削除するには、rm *filename* と入力します。
『』	参照する書名を示します。	『コードマネージャー・ユーザーズガイド』を参照してください。
「」	参照する章、節、ボタンやメニュー名、強調する単語を示します。	第 5 章「衝突の回避」を参照してください。 この操作ができるのは、「スーパーユーザー」だけです。
\	枠で囲まれたコード例で、テキストがページ行幅を超える場合に、継続を示します。	sun% **grep '^#define ** **XV_VERSION_STRING'**

コード例は次のように表示されます。

- C シェル

  machine_name% **command y|n** [*filename*]

- C シェルのスーパーユーザー

  machine_name# **command y|n** [*filename*]

- Bourne シェルおよび Korn シェル

  $ **command y|n** [*filename*]

- Bourne シェルおよび Korn シェルのスーパーユーザー

```
# command y|n [filename]
```

[ ] は省略可能な項目を示します。上記の例は、*filename* は省略してもよいことを示しています。

| は区切り文字 (セパレータ) です。この文字で分割されている引数のうち 1 つだけを指定します。

キーボードのキー名は英文で、頭文字を大文字で示します (例: Shift キーを押します)。ただし、キーボードによっては Enter キーが Return キーの動作をします。

ダッシュ (-) は 2 つのキーを同時に押すことを示します。たとえば、Ctrl-D は Control キーを押したまま D キーを押すことを意味します。

コマンド例のシェルプロンプト

次の表は、デフォルトのシステムプロンプトとスーパーユーザープロンプトを示したものです。

表 P–5 シェルプロンプト

シェル	プロンプト
UNIX および Linux システムの C シェル	machine_name%
UNIX および Linux システムの C シェルのスーパーユーザー	machine_name#
UNIX および Linux システムの Bourne シェルおよび Korn シェル	$
UNIX および Linux システムの Bourne シェルおよび Korn シェルのスーパーユーザー	#
Microsoft Windows のコマンド行	C:\

記号の表記規則

次の表では、このマニュアルで使用する記号について説明します。

表 P–6 記号の表記規則

記号	説明	例	意味
[ ]	省略可能な引数およびコマンドオプションを含みます。	`ls [-l]`	-l オプションは必須ではありません。
{ | }	必須コマンドオプションの選択肢を含みます。	`-d {y|n}`	-d オプションには、引数 y または引数 n を指定する必要があります。
${ }	変数の参照を示します。	`${com.sun.javaRoot}`	com.sun.javaRoot 変数の値を参照します。
-	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 キーを押して離したあと、後続のキーを押します。
→	グラフィカルユーザーインターフェイスでのメニュー項目の選択を示します。	「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。

マニュアル、サポート、およびトレーニング

Sun のサービス	URL	内容
マニュアル	http://jp.sun.com/documentation/	PDF 文書および HTML 文書をダウンロードできます。
サポートおよびトレーニング	http://jp.sun.com/supportraining/	技術サポート、パッチのダウンロード、および Sun のトレーニングコース情報を提供します。

◆ ◆ ◆ 第 1 章

Service Registry の設定およびセットアップ

この章では、Service Registry のインストール後の設定方法およびその他の管理作業の実行方法について説明します。

この章の内容は次のとおりです。

Service Registry の設定

レジストリのデフォルトのプロパティー設定を使って Service Registry のインストール後の設定を実行する方法については、『Sun Java Enterprise System 5 インストールガイド (UNIX 版)』を参照してください。カスタムプロパティー設定を使用するには、設定を実行する前に、ファイル *ServiceRegistry-base*`/install/install.properties` または *ServiceRegistry-base*`/install/install.properties.template` のコピーを作成して編集します。

レジストリを設定するには、`root` としてログインしても、スーパーユーザーになっても、root 以外のユーザーとしてログインしてもかまいません。

セキュリティーの観点から、root 以外のユーザーとしてレジストリを設定することをお勧めします。手順については、20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」を参照してください。

ServiceRegistry-base の場所は、Solaris OS では `/opt/SUNWsrvc-registry`、Linux および HP-UX システムでは `/opt/sun/srvc-registry` です。

注 – Service Registry を設定する前に、Sun Java System Application Server (「Application Server」) をインストールおよび設定しておく必要があります。Service Registry の設定プロセスは、レジストリを Application Server ドメイン内にインストールします。

Application Server はデフォルトの場所にインストールすることをお勧めします。Application Server をデフォルト以外の場所にインストールした場合には、Service Registry を設定する前に 25 ページの「非デフォルト Service Registry インストールに対する Application Server の設定」の手順に従ってください。

`install.properties` ファイルには、一連の変更可能なプロパティー設定が含まれています。表 1–1 に記載されたプロパティーは、設定プロセスによって使用されるものです。各プロパティー名には、接頭辞 `registry.install.` (末尾はピリオド) が付加されます。これらのプロパティーのいくつかは、レジストリ用に作成された Application Server ドメインにデフォルト以外のポートを設定します。その他の設定プロパティーについては、このマニュアルの別の部分で説明します。

Service Registry の複数のインスタンスを、root ユーザーまたは root 以外のユーザーとして設定できます。その場合は、`registry.install.dataHome` プロパティーがレジストリインスタンスごとに一意になるように変更してください。サーバー上で同時に複数のインスタンスを実行できるようにするには、インスタンスごとに一意となるように 8 つのポートプロパティーをさらに変更してください。複数の root 以外のインスタンスのために、インスタンスごとに一意になるように `registry.install.CACertDir` の値を変更してもかまいません。

表 1–1　Service Registry 設定プロパティー

プロパティー名	説明	デフォルトのプロパティー値
`DomainName`	Application Server ドメイン名	`registry`
`ServerInstanceHost`	Service Registry にアクセス可能なデフォルトのホスト名	`localhost`
`ServerInstancePort`	Service Registry Application Server HTTP ポート	`6480`
`ServerInstanceSecurePort`	Service Registry Application Server HTTPS ポート	`6443`

表 1–1 Service Registry 設定プロパティー (続き)

プロパティー名	説明	デフォルトのプロパティー値
ServerJMSPort	Service Registry Application Server Message Queue ポート	6484
ServerIIOPPort	Service Registry Application Server IIOP ポート	6485
ServerIIOPSecurePort	Service Registry Application Server IIOP セキュアポート	6486
ServerIIOPMutualAuthPort	Service Registry Application Server IIOP 相互認証ポート	6487
AdministrationJMXPort	Service Registry Application Server JMX ポート	6488
AdministrationPort	Service Registry Application Server 管理サーバーポート	6489
AdministratorUserID	Application Server 管理サーバーへのアクセス時に使用するユーザー名	admin
AdministratorPassword	Application Server 管理サーバーへのアクセス時に使用するパスワード	root として設定する場合、12345678 root 以外のユーザーとして設定する場合、なし
ApplicationServerKeystorePassword	Application Server キーストアへのアクセス時に使用するパスワード	root として設定する場合、12345678 root 以外のユーザーとして設定する場合、なし
RegistryServerKeystorePassword	Service Registry キーストアへのアクセス時に使用するパスワード	root として設定する場合、12345678 root 以外のユーザーとして設定する場合、なし

表 1–1 Service Registry 設定プロパティー (続き)

プロパティー名	説明	デフォルトのプロパティー値
clientDatabase	Service Registry データベースが、組み込まれた状態で実行するか、ネットワークサーバーモードで実行するかを決定します (ネットワークサーバーモードの場合は true)	false
RequireDatabaseAuthentication	Service Registry データベースへのアクセスにユーザー認証を要求するかどうかを決定します	false
DatabaseUserID	ユーザー認証が必要な場合、Service Registry データベースへのアクセスに使用するユーザー ID	APP
DatabasePassword	ユーザー認証が必要な場合、Service Registry データベースへのアクセスに使用するパスワード	root として設定する場合、app123 root 以外のユーザーとして設定する場合、なし
backupDir	Service Registry のバックアップに使用されるディレクトリ。通常はコメントアウトされています。別の場所を指定するには、コメントを削除します。	root として設定する場合、*RegistryDomain-base*/3.0/backup root 以外のユーザーとして設定する場合、$HOME/srvc-registry/3.0/backup
dataHome	Service Registry データが格納されるディレクトリ	root として設定する場合、*RegistryDomain-base* root 以外のユーザーとして設定する場合、$HOME/srvc-registry
CACertDir	Application Server ドメインが信頼すべき追加の証明書を格納するディレクトリ	root として設定する場合、*ServiceRegistry-base*/install/cacerts root 以外のユーザーとして設定する場合、$HOME/srvc-registry/cacerts

▼ インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する

始める前に　次に示す手順は、root としてログインするか、スーパーユーザーになることを前提とします。

1. ディレクトリ *ServiceRegistry-base*/install に移動します。
2. ファイル install.properties をセキュリティー保護された場所にコピーします。
   次のようなコマンドを使用します。
   ```
   cp install.properties $HOME/hidden_dir/sr.properties
   ```
3. プロパティーファイルのアクセス権を書き込み可能に変更します。
   次のようなコマンドを使用します。
   ```
   chmod 600 $HOME/hidden_dir/sr.properties
   ```
4. ファイル内の変更可能なプロパティーを編集します。
   たとえば、すべてのパスワードをデフォルト値から変更することをお勧めします。
5. アクセス権を元の読み取り専用の値に戻します。
   次のようなコマンドを使用します。
   ```
   chmod 400 $HOME/hidden_dir/sr.properties
   ```
6. *ServiceRegistry-base*/install ディレクトリで、変更した install.properties ファイルの場所を指定して次のコマンドを実行します (すべてを 1 行で入力)。
   次のようなコマンドを使用します。

   Ant-base **/ant -f build-install.xml -Dinstall.properties=$HOME/hidden_dir/sr.properties configure**

   ant コマンドが動作するには、JAVA_HOME 環境変数が設定されている必要があります。通常はこの変数を次の値に設定します。

   ```
   /usr/jdk/entsys-j2se
   ```

   レジストリ設定プロセスは、*RegistryDomain-base*/domains/${registry.install.DomainName} に Application Server ドメインを作成します。デフォルトのドメイン名は、registry です。その後、設定プロセスは、そのドメインを起動し、レジストリを配備したあと、その実行中のドメインをそのまま起動しておきます。

   レジストリ設定プロセスは、レジストリデータベースとサーバーキーストアをディレクトリ *RegistryDomain-base*/3.0 内にインストールします。このディレクトリは

レジストリをアンインストールしても削除されませんが、これは、そのデータベースを将来のリリースにおいても使用できるよう保護するためです。このディレクトリを削除するかどうかや、削除する場合のタイミングは、管理者によって制御されます。

RegistryDomain-base の場所は、Solaris OS では `/var/opt/SUNWsrvc-registry`、Linux および HP-UX システムでは `/var/opt/sun/srvc-registry` です。

レジストリ設定プロセスによって *ServiceRegistry-base*`/install/cacerts` という名前のディレクトリが作成され、Application Server ドメインが信頼すべき追加の証明書が格納されます。

7 `ant configure` コマンドの出力にエラーが含まれていないか確認します。

エラーが存在しない場合には、Web コンソールまたは管理ツールを使い始めることができます。

▼ インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する

始める前に　これらの手順は、`root` ではなく通常のユーザーとしてログインしていて、さらに別のユーザーが `root` としてログインして Service Registry をインストール済みの場合を前提とします。

1 ディレクトリ *ServiceRegistry-base*`/install` に移動します。

2 ファイル `install.properties.template` をホームディレクトリ内の場所にコピーし、名前を変更します。

次のようなコマンドを使用します。

```
cp install.properties.template $HOME/sr.properties
```

3 テキストエディタでそのファイルのコピーを開き、変更可能なプロパティーを必要に応じて編集します。

次のプロパティーにパスワード値を指定してください。これらは、テンプレートファイルでは空の状態です。

```
registry.install.AdministratorPassword=
registry.install.ApplicationServerKeystorePassword=
registry.install.RegistryServerKeystorePassword=
```

4 ほかのユーザーが読み取れないようにファイルのアクセス権を変更します。

次のようなコマンドを使用します。

```
chmod 400 $HOME/sr.properties
```

5 *ServiceRegistry-base*/install ディレクトリで、変更したファイルの場所を指定して次のようなコマンドを実行します (すべてを 1 行で入力)。

Ant-base/**ant -f build-install.xml -Dinstall.properties=$HOME/sr.properties configure**

同様のコマンドを使用して、たその他の設定ターゲットを必要に応じ実行します。

レジストリ設定プロセスは、$HOME/srvc-registry/domains/${registry.install.DomainName} に Application Server ドメインを作成します。デフォルトのドメイン名は、registry です。

レジストリ設定プロセスは、レジストリデータベースとサーバーキーストアをディレクトリ $HOME/srvc-registry/3.0 内にインストールします。

レジストリ設定プロセスによって $HOME/srvc-registry/cacerts という名前のディレクトリをされ、Application Server ドメインが信頼すべき追加の証明書が格納されます。

6 ant configure コマンドの出力にエラーが含まれていないか確認します。

エラーが存在しない場合には、Web コンソールまたは管理ツールを使い始めることができます。

Service Registry 用 Application Server ドメインの管理

Service Registry の設定プロセスはデフォルトで、registry という名前の Application Server ドメインを作成し、そこに Service Registry Web アプリケーションを配備します。このドメインは、*RegistryDomain-base*/domains/registry ディレクトリ内に存在します。

この場所は、Application Server ドメインのデフォルトの場所である、/var/opt/SUNWappserver/domains (Solaris OS)、/var/opt/sun/appserver/domains (Linux および HP-UX システム) とは異なります。

注 – registry ドメインでは Service Registry 以外のアプリケーションを実行しないことをお勧めします。

---

registry ドメインを管理するには、Application Server 管理コンソール (「管理コンソール」) を使用できます。管理コンソールを使えば、ドメインの起動と停止、サーバーログの表示など、各種の管理作業を行えます。詳細については、22 ページの「Application Server 管理コンソールを使用する」を参照してください。

また、サーバーログを直接確認することもできます。このログはファイル *RegistryDomain-base*/domains/registry/logs/server.log に格納されています。

管理コンソールのほかに、asadmin コマンドを使って registry ドメインを管理することも可能です。ただし、このドメインはデフォルトの場所に存在しないため、--domaindir オプションをサポートする asadmin コマンドを使用する際には、そのオプションを指定する必要があります。--domaindir オプションの引数は *RegistryDomain-base*/domains です。

asadmin コマンドの --passwordfile オプションを使う場合、レジストリドメインの管理者パスワードのファイルとコピーが必要です。このようなファイルを作成するには、build-install.xml ファイルの generate.password.file ターゲットを使用します。ファイルは *RegistryDomain-base*/3.0/data/security/pw.txt です。

registry ドメインでは、デフォルトの Application Server ドメインである domain1 との間で衝突が発生しないように、一連の非デフォルトポートが使用されます。これらの Service Registry ポート値は、IANA (Internet Assigned Numbers Authority) に登録されます。表 1–2 に、それらのポートの一覧と説明を示します。詳細については、『Sun Java System Application Server Enterprise Edition 8.2 管理ガイド』の「Application Server のポート」を参照してください。

表 1–2　Service Registry ドメインのデフォルトのポート

ポート値	説明
6480	HTTP ポート
6443	HTTPS over SSL
6484	Message Queue ポート
6485	IIOP ポート
6486	IIOP SSL ポート
6487	IIOP 相互認証ポート
6488	JMX ポート
6489	Application Server ドメイン管理ポート

▼ Application Server 管理コンソールを使用する

1. **Web** ブラウザで **URL** https://*hostname*:6489/ にアクセスします。
   hostname は、Application Server と Service Registry が動作しているシステムです。
2. 提供された証明書を受け入れます。
   ログインページが表示されます。

3 ログインページの「ユーザー名」フィールドに「admin」と入力します。
レジストリの設定時に registry.install.AdministratorUserID プロパティーのデフォルト値を変更した場合は、そのときに指定した値を入力します。

4 「パスワード」フィールドに **Application Server** 管理者パスワードを入力します。レジストリ設定時に registry.install.AdministratorPassword プロパティーに指定した値を使用します。デフォルトは 12345678 です。

5 「ログイン」をクリックします。

参照 管理コンソールの使用方法の詳細については、管理コンソールのオンラインヘルプまたは『Sun Java System Application Server Enterprise Edition 8.2 管理ガイド』を参照してください。

Service Registry ロギングレベルの変更

Service Registry のロギングレベルを変更するには、管理コンソールのオンラインヘルプの手順に従います。「追加プロパティー」領域で指定するプロパティーは org.freebxml.omar です。

特定の Service Registry サブコンポーネントのロギングを変更するには、次のファイルを参照してください:
RegistryDomain-base/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes/log4j.properties 。このファイルに記載されているサブコンポーネントであればいずれも指定可能です。文字列 log4j.logger を含めないでください。たとえば、org.freebxml.omar.server と指定することで、サーバー呼び出しのログをとれます。

▼ レジストリ用 Application Server ドメインを停止および再起動する

レジストリの設定プロセスは、レジストリの配備先となる Application Server ドメインを起動します。特定の管理作業を実行し終わったら、そのドメインを停止および再起動する必要があります。そうした作業の例としては、26 ページの「レジストリドメイン用の Java 仮想マシン (JVM) の設定」および27 ページの「管理者の作成」が挙げられます。

ドメインを再起動する必要がある場合には、管理コンソールからユーザーにその旨が通知されます。管理コンソールを使えばその作業を実行できます。asadmin コマンドを使用している場合には、Ant タスクを使ってドメインを停止および起動できます。

1 **Service Registry** インストールディレクトリに移動します。
cd *ServiceRegistry-base*/**install**

2 次のコマンドを実行します (すべてを 1 行で入力)。

Ant-base`/ant -f build-install.xml -Dinstall.properties=` *props-file* `appserver.domain.bounce`

ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

`appserver.domain.bounce` のターゲットがドメインを停止し、そのあと再起動します。

`build-install.xml` ファイルには、レジストリ用ドメインの停止と起動を別個に行うための Ant ターゲットも含まれています。このドメインを停止するには、Ant ターゲット `appserver.domain.stop` を使用します。このドメインを起動するには、Ant ターゲット `appserver.domain.start` を使用します。

▼ レジストリ用ドメイン内の信頼できる証明書にルート証明書を追加する

この作業を行うと、Application Server `registry` ドメイン内の信頼できる証明書リストが拡張されます。

この作業は、ユーザーがサードパーティーの証明書を使用し、そのサードパーティーのルート認証局 (CA) の証明書が Application Server の truststore 内にまだ存在しない場合にのみ実行します。レジストリで発行された証明書のみを使用する場合は、この作業を実行しないでください。

必要となる CA の証明書がすでに使用可能であるかどうかを確認するには、次のように `build-install.xml file` の `list.cacerts` ターゲットを使用できます。

Ant-base`/ant -f build-install.xml -Dinstall.properties=`*props-file* `list.cacerts`

ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

1 サポートするルート証明書をダウンロードします。

ルート証明書を提供するサイトを次に示します。

- `http://www.entrust.net/developer/`
- `http://www.geotrust.com/resources/root_certificates/`

- http://www.thawte.com/roots/
- www.verisign.com/support/roots.html

2 必要に応じて、unzip コマンドを使ってダウンロードしたアーカイブから .cer ファイルを取り出します。

注 – ファイルによっては、サフィックスが .der になっています。

---

3 .cer ファイルを、install.properties ファイルのコピーで registry.install.CACertDir プロパティーに指定されたディレクトリへコピーします。

通常、この値は、root として設定した場合は *ServiceRegistry-base*/install/cacerts で、root 以外のユーザーとして設定した場合は $HOME/srvc-registry/cacerts です。

4 ディレクトリ *ServiceRegistry-base*/install に移動します。

5 次のコマンドを実行します (すべてを 1 行で入力)。

Ant-base/**ant -f build-install.xml -Dinstall.properties=** *props-file* **install.cacerts**

このコマンドは、registry.install.CACertDir プロパティーで指定されたディレクトリ内で見つかったすべての証明書を、Application Server ドメインの truststore 内にインストールします。

ここでも、list.cacerts ターゲットを使えば、証明書が正しくインストールされたか確認できます。

6 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

非デフォルト Service Registry インストールに対する Application Server の設定

Application Server のデフォルトのインストール場所は、Solaris OS の場合は /opt/SUNWappserver/appserver 、Linux および HP-UX システムの場合は /opt/sun/appserver です。Application Server を別の場所にインストールした場合、Service Registry を設定する前に install.properties のコピーを編集してください。

▼ install.properties ファイルのコピーを編集する

1 ディレクトリ *ServiceRegistry-base*/install に移動します。

2. ファイル `install.properties` をセキュリティー保護された場所にコピーし、名前を変更します。

   次のようなコマンドを使用します。

   ```
   cp install.properties $HOME/hidden_dir/sr.properties
   ```

3. プロパティーファイルのアクセス権を変更して書き込み可能にします。

   次のようなコマンドを使用します。

   ```
   chmod 600 $HOME/hidden_dir/sr.properties
   ```

4. テキストエディタでファイルを開きます。

5. プロパティー `appserver.root.dir` のコメントアウトされた定義を見つけます。

6. コメント文字 (#) を削除し、そのプロパティー定義を **Application Server** の実際の場所で置き換えます。

7. ファイルを保存して閉じます。

次の手順　続いて、15 ページの「Service Registry の設定」の手順に従います。

レジストリドメイン用の Java 仮想マシン (JVM) の設定

Service Registry が正しく動作するには、次の条件が必要です。

- レジストリが外部 Web サイトにアクセス可能である
- レジストリ用の Application Server ドメインに、使用可能なメモリーが十分にある

任意のレジストリオブジェクト内に格納できる `ExternalLink` オブジェクトは、そのレジストリオブジェクトに関連付けられた外部 URL を指定します。任意の `Service` オブジェクト内に格納できる `ServiceBinding` も、外部 URL を指すことが可能です。ユーザーが `ExternalLink` および `ServiceBinding` オブジェクトを作成するためには、Service Registry が URL の妥当性検査を行える必要があり、それには外部の Web サイトへのアクセスが必要です。レジストリがファイアウォールの背後に配備されている場合には、こうしたアクセスを可能にするプロキシを設定する必要があります。

プロキシ設定を行うには、Service Registry が配備されている Application Server ドメインの Java 仮想マシン (JVM) オプションとして、Web プロキシのホストとポートを指定する必要があります。

また、レジストリでメモリー不足となる可能性もあります。この問題の発生を防ぐには、レジストリ用の Application Server ドメインで使用可能なメモリーを増やすように JVM オプションを設定します。

レジストリ用の JVM オプションを設定するには、次の手順に従います。

▼ レジストリドメイン用の JVM オプションを設定する

1. 22 ページの「Application Server 管理コンソールを使用する」の説明に従って Application Server 管理コンソールにログインします。
2. 「設定」ノードを開きます。
3. サーバーノード `server-config (Admin Config)` を開きます。
4. 「JVM 設定」をクリックします。
5. 「JVM オプション」タブをクリックします。
6. 「JVM オプションを追加」をクリックします。
7. テキストフィールドに次のように入力します (すべてを 1 行で入力)。

-Dhttp.proxyHost=*hostname.domainname* **-Dhttp.proxyPort=8080 -Dhttp.nonProxyHosts=localhost**

   ポートの値は通常 8080 ですが、実際の値がこれとは異なっている場合には、その正しい値を指定します。

8. 「JVM オプションを追加」を再度クリックします。
9. テキストフィールドに次のように入力します (すべてを 1 行で入力)。

   -XX:MaxPermSize=128m

10. 「保存」をクリックします。
11. 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

管理者の作成

Service Registry 管理ツールで行う作業の中には、管理者として登録されているユーザーのみが行えるものもあります。さらに、管理者は、ほかのユーザーが送信したオブジェクトに対するライフサイクル変更 (承認など) の実施を要求されることもあります。

また、管理者は、デフォルトのアクセス制御ポリシー (ACP) も変更できます。ただし、ACP の記述は現時点では手動による処理であり、OASIS の XACML (eXtensible Access Control Markup Language) の知識が必要になります。詳細については、

『ebXML RIM 3.0』の第 9 章「Access Control Information Model」、特に 9.7.6 節から 9.7.8 節にかけてのサンプルを参照してください。ebXML RIM 3.0 仕様を入手する方法については、7 ページの「お読みになる前に」を参照してください。

▼ 管理者を作成する

自分自身を管理者として登録するには、次の手順に従います。

1. 『**Service Registry 3.1** ユーザーズガイド **(2006Q4)**』の「ユーザーアカウントの作成」の説明に従ってユーザー登録を行うか、**54** ページの「add user」の説明に従って管理ツールの `add user` を使って自分自身をユーザーとして追加します。

2. 登録に **Web** コンソールを使用した場合は、次のように `User` オブジェクトの一意の識別子を取得します。

   a. **Web** コンソールを使って、**User** に設定したオブジェクト型で基本クエリーを実行します。

   b. 「詳細」リンクをクリックして、レジストリによって作成された `User` オブジェクトを表示します。

   c. 「一意の識別子」フィールドの値を記録するか、コピーしてファイルにペーストします。

   `add user` コマンドを使用した場合は、`users` コマンドを使ってユーザーのリストを取得してから、自分のユーザー名の識別子の値をコピーします。

3. ディレクトリ *RegistryDomain-base*`/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes` に移動します。

4. ファイル `omar.properties` をテキストエディタで開きます。

5. プロパティー `omar.security.authorization.registryAdministrators` の定義を見つけます。

6. そのプロパティー定義を編集し、縦棒 **(|)** と手順 **2** でコピーした一意の識別子の文字列を追加します。

   プロパティー定義はすべて 1 行で入力する必要があります。また、定義内に空白が含まれていてはいけません。編集後のプロパティー定義は、次のようになります (すべて1 行に入る)。

   ```
   omar.security.authorization.registryAdministrators=
   urn:freebxml:registry:predefinedusers:registryoperator|
   urn:uuid:77f5c196-79de-4286-8483-8d80def3583b
   ```

7 omar.properties ファイルを保存して閉じます。

8 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

次の手順 別の管理者を追加する場合、omar.properties ファイルを編集する必要はありません。管理ツールまたは Web コンソールのいずれかを使ってユーザーを追加でき、Web コンソールを使ってそれらのユーザーを管理者として分類できます。

ユーザー登録を実行できるユーザーの指定

デフォルトでは、Service Registry URL にアクセスできるユーザーであれば誰でもユーザー登録ウィザードを使ってユーザー登録を実行できます。登録されたユーザーは誰でもコンテンツをレジストリに発行できます。

ファイル omar.properties 内でプロパティー omar.security.selfRegistration.acl を定義すれば、指定されたユーザーのみがユーザー登録を実行できるようにこの機能を制限できます。

このタスクの完了後は、ウィザードを使って登録を行えるユーザーだけが、プロパティーの定義によって指定された名と姓を持つようになります。それ以外のユーザーが登録を行おうとして、ユーザー登録ウィザードの手順 3 で「ユーザー認証の詳細」フォームに情報を入力してから「次へ」ボタンをクリックすると、エラーメッセージが表示されます。エラーメッセージの内容は「ユーザー登録に失敗しました。」です。このメッセージの次には UserNotFoundException を報告する行が表示されます。

omar.security.selfRegistration.acl プロパティーが omar.properties に存在しない場合、あるいは空の文字列として定義されている場合は、登録されている任意のユーザーはコンテンツをレジストリに発行できます。

▼ ユーザー登録を制限する

1 ディレクトリ *RegistryDomain-base*/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes に移動します。

2 ファイル omar.properties をテキストエディタで開きます。

3 次のプロパティーの定義を追加します。

```
omar.security.selfRegistration.acl
```

このプロパティー定義はファイル内のどこにでも追加できます。プロパティー定義の論理的な位置は、接頭辞 omar.security を持つその他のプロパティーが定義されている領域内です。

次に示す例のように、ユーザー登録の実行を承認されているユーザーの名と姓をコンマで区切ったリストとして、プロパティー値を定義します。

```
omar.security.selfRegistration.acl=Vijay Patel, Jane Doe,
```

4 omar.properties ファイルを保存して閉じます。

5 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

レジストリオブジェクトのバージョン管理の有効化

デフォルトでは、レジストリオブジェクトのバージョン管理は無効になっています。バージョン管理を有効にすると、オブジェクトのいずれかの属性が変更されるたびに、オブジェクトの新しいバージョンが作成されます。バージョン管理の有効化は管理者のタスクです。

▼ レジストリオブジェクトのバージョン管理を有効にする

1 ディレクトリ *RegistryDomain-base*/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes に移動します。

2 ファイル omar.properties をテキストエディタで開きます。

3 プロパティー omar.server.lcm.VersionManager.versionableClassList の定義を見つけます。

デフォルトでは、このプロパティーには値が設定されていません。

```
omar.server.lcm.VersionManager.versionableClassList=
```

4 オブジェクトの変更時にレジストリによって新しいバージョンを作成する任意のオブジェクトを指定します。複数のオブジェクトは縦棒 (|) で区切ります。

次に例を示します。

```
omar.server.lcm.VersionManager.versionableClassList=Service|Organization
```

例として、ファイル内のプロパティー設定のコメントにされたコピーを使用します。

5 omar.properties ファイルを保存して閉じます。

6 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

WSDL Cataloger の無効化

デフォルトでは、WSDL ファイルが ExtrinsicObject オブジェクトとしてレジストリに発行されると、レジストリは ebXML Registry Profile for Web Services によって定義されている機能を使用して、オブジェクトのメタデータを作成します。この機能は無効にできます。

▼ WSDL Cataloger を無効にする

1 ディレクトリ *RegistryDomain-base*/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes に移動します。

2 ファイル omar.properties をテキストエディタで開きます。

3 プロパティー omar.server.cms.classMap.urn\:oasis\:names\:tc\:ebxml-regrep\:profiles\:ws\:wsdl\:cat の定義を見つけます。

4 定義の前にコメント文字 (#) を挿入して、このプロパティー定義をコメントアウトします。

5 omar.properties ファイルを保存して閉じます。

6 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

Web コンソールの設定

管理者は、設定ファイルを編集することにより、Web コンソールの表示方法の一部をカスタマイズできます。

ここでは次の作業について説明します。

- 32 ページの「定義済みクエリーの追加」
- 34 ページの「デフォルトクエリーの変更」

- 35 ページの「Classification Scheme の非表示」
- 36 ページの「検索結果の表示の設定」

Web コンソールの使用の詳細については、『Service Registry 3.1 ユーザーズガイド (2006Q4)』を参照してください。

定義済みクエリーの追加

Service Registry には定義済みクエリーがいくつか含まれていますが、これらは、Web コンソール上の「検索」フォームの「定義済みクエリーを選択」ドロップダウンリスト内に表示されます。管理者として、対象レジストリインストールに固有の新しいクエリーを、そのドロップダウンリストに追加できます。

▼ 定義済みクエリーを追加する

1. **Web** コンソールを使って `AdhocQuery` オブジェクトをレジストリに発行します。

   クエリーに指定する名前と説明は、定義済みクエリーのドロップダウンリストに表示されます。次のように、クエリーの SQL 文では、ユーザーが入力するデータを対になった一重引用符で囲んで、プレースホルダを指定します。

   ```
   select * from registryobject where id = ''$lid''
   ```

2. `AdhocQuery` オブジェクトの一意の識別子と **SQL** 文内の任意のプレースホルダの一意の識別子を記録するか、コピーしてファイルにペーストします。

3. ディレクトリ *RegistryDomain-base*`/3.0/jaxr-ebxml` に移動します。

4. テキストエディタでファイル `registry-browser-config.xml` を開きます。

   必要に応じて変更をバックアウトできるように元のファイルのコピーを作成します。

5. 次の形式のエントリを `registry-browser-config.xml` ファイルに追加します。**SQL** 文内の各プレースホルダに対応する `Parameter` 要素を指定します。

   ```
   <Query>
     <AdhocQueryRef id="unique-identifier"/>
     <Parameter parameterName="$placeholder-name" datatype="string">
       <rim:Name>
         <rim:LocalizedString xml:lang="en" charset="UTF-8"
           value="parameter-name-in-en-locale"/>
         <rim:LocalizedString xml:lang="fr" charset="UTF-8"
           value="parameter-name-in-fr-locale"/>
       </rim:Name>
       <rim:Description>
         <rim:LocalizedString xml:lang="en" charset="UTF-8"
   ```

```
        value="parameter-description-in-en-locale"/>
      <rim:LocalizedString xml:lang="fr" charset="UTF-8"
        value="parameter-description-in-fr-locale"/>
    </rim:Description>
  </Parameter>
  ...
</Query>
```

unique-identifier は AdhocQuery オブジェクトの一意の識別子です。

各パラメータの parameterName 属性は、クエリーの SQL 文内のプレースホルダから取り込んでください。

datatype 属性には、次のいずれかの値を指定できます。

- string: パラメータが検索フォーム内でテキストフィールドとして表示されます。
- taxonomyElement: パラメータが検索フォーム内でドロップダウンリストとして表示されます。taxonomyElement データ型を指定すると、Name 要素および Description 要素のあとに次のような SlotList 要素が追加されます。

  ```
  <rim:SlotList>
    <rim:Slot name="domain">
      <rim:ValueList>
        <rim:Value>
        classification-scheme-or-concept-id
        </rim:Value>
      </rim:ValueList>
    </rim:Slot>
  </rim:SlotList>
  ```

  classification-scheme-or-concept-id は、Concept (または Subconcept) をドロップダウンリストに表示する Classification Scheme または Concept の一意の識別子です。該当する Classification Scheme がレジストリにまだ存在しない場合は、発行してください。

  スロット名は「domain」にしてください。
- boolean: パラメータが検索フォーム内でチェックボックスとして表示されます。

datatype が string または boolean である場合は、defaultValue 属性を Parameter 要素に追加して、検索フォームに表示されるデフォルト値を指定することもできます。

サポートするすべてのロケールのために、各パラメータの名前と説明に対してローカライズされた文字列値を指定します。検索フォーム内のパラメータのラベルには、現在のロケールの *parameter-name* が表示されます。

registry-browser-config.xml ファイル内の既存エントリを参考として使用します。

6 registry-browser-config.xml ファイルを保存して閉じます。

7 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

デフォルトクエリーの変更

「定義済みクエリーを選択」ドロップダウンリストにデフォルトとして表示されるクエリーは、ユーザーが名前、説明、および分類でレジストリオブジェクトを検索できる基本クエリーです。

管理者は、このデフォルトを実際の環境に適したクエリーに変更できます。たとえば、32 ページの「定義済みクエリーの追加」の手順に従って、デフォルトクエリーをレジストリに追加した新しい定義済みクエリーにすることもできます。この変更を行うには、設定ファイルのプロパティーを編集します。

▼ デフォルトクエリーを変更する

1 ディレクトリ *RegistryDomain-base*/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes に移動します。

2 ファイル jaxr-ebxml.properties をテキストエディタで開きます。

3 プロパティー jaxr-ebxml.thin.defaultQueryPanel の定義を見つけます。デフォルトでは、このプロパティーは次のようにコメントアウトされています。

```
#jaxr-ebxml.thin.defaultQueryPanel=
```

4 コメント文字 (#) を削除します。

5 次の例のように、デフォルトにするクエリーの論理識別子を指定することにより、このプロパティーの値を設定します。

```
jaxr-ebxml.thin.defaultQueryPanel=urn:oasis:names:tc:ebxml-regrep:query:MyQuery
```

6 jaxr-ebxml.properties ファイルを保存して閉じます。

7 23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」の手順に従います。

Classification Scheme の非表示

Classification Scheme のツリー構造は、Web コンソールの次の領域に表示されます。

- 「検索」メニュー領域の「Classification ノードの選択」をクリックするとき、またはある種のレジストリオブジェクトの Concept を選択する必要があるときに表示される「ClassificationScheme/Concept セレクタ」ウィンドウ
- 「探索」メニュー領域

管理者は、Service Registry のユーザーが Classification Scheme を使用できないようにする場合に、そのClassification Scheme を非表示にできます。Classification Scheme を非表示にするには、設定ファイル内にプロパティーを定義します。

▼ Classification Scheme を非表示にする

1 ディレクトリ *RegistryDomain-base*`/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes` に移動します。

2 ファイル `jaxr-ebxml.properties` をテキストエディタで開きます。

3 次の構文を使ってプロパティー `jaxr-ebxml.registryBrowser.ConceptsTreeModel.hiddenSchemesList` を設定します。プロパティー定義はすべて **1** 行で入力してください。また、定義内に空白が含まれていてはいけません。

```
jaxr-ebxml.registryBrowser.ConceptsTreeModel.hiddenSchemesList=
class-scheme-id1|class-scheme-id2|...
```

非表示にする各 Classification Scheme の論理識別子を指定します。複数の識別子を指定する場合は、次の例のように縦棒 (|) を使って識別子を区切ります。

```
jaxr-ebxml.registryBrowser.ConceptsTreeModel.hiddenSchemesList=
urn:oasis:names:tc:ebxml-regrep:classificationScheme:StatusType|
urn:oasis:names:tc:ebxml-regrep:profile:ws:classificationScheme:BindingType
```

4 `jaxr-ebxml.properties` ファイルを保存して閉じます。

5 23 ページの「レジストリ用 **Application Server** ドメインを停止および再起動する」の手順に従います。

検索結果の表示の設定

デフォルトでは、Web コンソールには 1 つのクエリーに対して一度に 25 件の検索結果が表示されます。25 件を超える検索結果が返された場合、ユーザーは追加の検索結果ページを表示できます。管理者は、各ページに表示される検索結果の数を変更できます。

Web コンソールの検索結果領域には、デフォルトでは特定の列が表示されます。オブジェクトごとに、オブジェクトの型、名前、説明、バージョン、およびバージョンコメントが表示されます。一部のオブジェクト型には、デフォルト以外の表示が設定されています。たとえば、`ServiceBinding` オブジェクトでは、バージョン情報の代わりにエンドポイントが表示に含まれます。管理者は、選択したオブジェクト型についてデフォルト以外のデータを表示するように設定データを追加できます。

これらの作業を実行するには、設定ファイルを編集します。

▼ 検索結果の表示内の行数を設定する

1. ディレクトリ *RegistryDomain-base*`/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes` に移動します。
2. ファイル `jaxr-ebxml.properties` をテキストエディタで開きます。
3. プロパティー `omar.client.thinbrowser.numSearchResults` の定義を見つけます。

   ```
   omar.client.thinbrowser.numSearchResults=25
   ```

4. 値「**25**」を希望する値に変更します。
5. `jaxr-ebxml.properties` ファイルを保存して閉じます。
6. **23** ページの「レジストリ用 **Application Server** ドメインを停止および再起動する」の手順に従います。

▼ 検索結果領域の列を設定する

オブジェクト型に「検索結果」領域の列を設定できます。列にはオブジェクトの属性が表示されます。

1. ディレクトリ *RegistryDomain-base*`/3.0/jaxr-ebxml` に移動します。
2. テキストエディタでファイル `registry-browser-config.xml` を開きます。

3 エントリを registry-browser-config.xml ファイルに追加するか、既存のエントリを編集します。次の形式を使用します。

この例では、Service オブジェクトについてデフォルト以外の表示を設定しています。

```
<ObjectTypeConfig
    className="org.freebxml.omar.client.xml.registry.infomodel.ServiceImpl"
    id="urn:oasis:names:tc:ebxml-regrep:ObjectType:RegistryObject:Service">
  <SearchResultsConfig>
    <SearchResultsColumn columnClass="java.lang.Object"
      columnHeader="Object Type" columnWidth="25" editable="false"
      method="getObjectType"/>
    <SearchResultsColumn columnClass="java.lang.Object"
      columnHeader="Name" columnWidth="25" editable="true" method="getName"/>
    <SearchResultsColumn columnClass="java.lang.Object"
      columnHeader="Description" columnWidth="30" editable="true"
      method="getDescription"/>
    <SearchResultsColumn columnClass="java.lang.Object"
      columnHeader="Status" columnWidth="15" method="getStatusAsString"/>
    <SearchResultsColumn columnClass="java.lang.Object"
      columnHeader="Version" columnWidth="5" method="getVersionName"/>
  </SearchResultsConfig>
</ObjectTypeConfig>
```

registry-browser-config.xml ファイルには、ObjectTypeConfig 要素のための構文が用意されています。ファイル内の既存の要素を例として使用します。これらの要素によって、レジストリオブジェクトについてのデフォルトの表示と、ExternalLink、ExtrinsicObject、および ServiceBinding の各オブジェクトについてのデフォルト以外の表示が設定されます。

設定可能な列の最大数は 30 です。

SearchResultsColumn 要素については次のとおりです。

- columnClass 属性の値は常に java.lang.Object です。
- columnHeader 属性の値は、Web コンソールのリソースバンドルファイル内のメッセージへのキーです。これらのファイルは、ディレクトリ *registryDomain-base*/domains/registry/applications/j2ee-modules/soar/WEB-INF/classes/org/freebxml/omar/client/ui/thin/ に格納されています。たとえば、columnHeader の値に Object Type と入力すると、Web コンソールのバッキング Bean は WebResourceBundle クラスを使ってそのキーを持つメッセージを検索します。WebResourceBundle は、メッセージを見つけられない場合、キーを小文字に変換して再度検索します。それでも見つからない場合は、メッセージの値を "???"+key+"???" に設定し、リソースバンドルエントリが見つからないという警告メッセージをログにとります。このため、新しくローカライズされた columnHeader 値を追加するには、このディレクトリに格納されている ResourceBundle ファイルに新しいメッセージキーを入力してください。
- columnWidth 属性は、Web コンソールには使用されません。

- `editable` 属性は、Web コンソールには使用されません。
- ほとんどの場合、ebXML Registry Information Model Version 3.0 仕様のクラス属性から `method` 属性のメソッド名を推測できます (詳細については、7 ページの「お読みになる前に」を参照)。`getStatusAsString` メソッドは、`RegistryObjectImpl` 実装クラスで見つけることができます。ただし、Service Registry の今回のリリースに API のマニュアルは付属しません。

`omar.client.xml.registry.infomodel` クラス名ごとに `ObjectTypeConfig` 要素を複数設定することはできません。

4. `registry-browser-config.xml` ファイルを保存して閉じます。
5. **23 ページの「レジストリ用 Application Server ドメインを停止および再起動する」**の手順に従います。
6. 再構成の結果を確認するため、**Web** コンソールの「検索」メニューまたは「探索」メニューを使って、変更した列を持つオブジェクトを表示します。

Service Registry の再インストール

Service Registry のアンインストールと再インストールを行う必要がある場合、再インストールを実行する前に次の作業を実行します。

- 保持するデータがレジストリデータベースに含まれる場合、39 ページの「Java DB データベースの管理」の説明に従ってデータベースをバックアップします。
- レジストリ用 Application Server ドメインを停止したあと、このドメインを削除します。このドメインを削除しなかった場合、再インストールされたレジストリのインストール後設定が失敗します。

Service Registry データベースを再インストールする必要がある場合 (データベースが破損した場合など) は、39 ページの「Service Registry データベースを再インストールする」の手順に従います。データベースを再インストールする前にアンインストールする必要はありません。

▼ レジストリ用 Application Server ドメインを停止および削除する

1. ディレクトリ *ServiceRegistry-base*`/install` に移動します。
2. 次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `appserver.domain.delete`

ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

このターゲットは、ドメインを停止したあと、ドメインを削除します。

▼ Service Registry データベースを再インストールする

このタスクでは、登録済みユーザーを含む既存のデータベースコンテンツが削除され、デフォルトのデータベースが再作成されます。

1. ディレクトリ *ServiceRegistry-base*`/install` に移動します。
2. 次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `install.db`
3. 次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `appserver.domain.stop export.registryOperatorCert install.cacerts appserver.domain.start`

Java DB データベースの管理

このレジストリは、Java DB データベースを使用します。Java DB は、Apache Software Foundation のオープンソースリレーショナルデータベースプロジェクトの商用リリースです。Apache プロジェクトでは Derby と呼ばれています。

デフォルトでは、このデータベースはディレクトリ *RegistryDomain-base*`/3.0/data/registry/soar/` に格納されています。データベースのバックアップは、バックアップの日付を含んだサブディレクトリ名 (たとえば、`20060419-004759`) とともに、ディレクトリ *RegistryDomain-base*`/3.0/backup/` に格納されています。

デフォルトでは、Java DB データベースは組み込みモードで実行されます。これは、Service Registry と同じ JVM で実行されることと、1 つのクライアント (Service Registry) のみから接続が可能であることを意味します。リモート接続はできません。Java DB が組み込みモードで実行される場合、データベースをバックアップできるのはデータベースの停止中のみです (オフラインバックアップ)。手順については、41 ページの「データベースのオフラインバックアップを実行する」を参照してください。

レジストリドメインの実行中にデータベースをバックアップすることができる必要がある場合 (オンラインバックアップ)、Java DB データベースをネットワークサーバーモードで実行してください。その手順については、42 ページの「組み込みモードからネットワークサーバーモードに切り替える」および 44 ページの「データベースのオンラインバックアップを実行する」を参照してください。

ネットワークサーバーモードで実行される場合、Java DB は、よく知られているクライアント/サーバー構成で複数のクライアント接続を受け入れることができます。たとえば、Service Registry と SQL クライアントの両方が同時に Java DB と通信できます。 ネットワークサーバーモードで実行されるとき、Java DB は、デフォルト値が 1527 のデータベースポートを使用します。Service Registry などのクライアントは、このポートを使ってデータベースと通信します。

Java DB がネットワークサーバーモードで実行されている場合、Service Registry はネットワーククライアントモードで実行されます。

ネットワークサーバーモードで実行されるデータベースは、認証されたクライアントのみが使用できるようにパスワードで保護する必要があります。組み込みモードで実行する場合にも同様の方法でデータベースを保護できますが、そのような保護は必須ではありません。

デフォルトでは、ファイル *ServiceRegistry-base* `/install/install.properties` 内のプロパティーは、Java DB が組み込みモードで実行されるように設定されます。表 1–3 は、それらの設定を示しています。

表 1–3　Java DB のデフォルトのプロパティー設定

プロパティー設定	説明
`registry.install.clientDatabase=false`	組み込みモードを有効にします
`registry.install.RequireDatabaseAuthentication=false`	データベース認証を必要としません
`registry.install.DatabaseUserID=APP`	ユーザー ID を `APP` (使用されていない) に設定します
`registry.install.DatabasePassword=app123`	パスワードを `app123` (使用されていない) に設定します

Java DB の詳細については、the Java DB web siteの Java DB に関するマニュアルを参照してください。

▼ データベース認証を要求する

デフォルトでは、データベース認証は必須ではありません。データベースを組み込みモードで実行しているときに認証を要求することは可能です。また、データベースをネットワークサーバーモードで実行中には認証を必須とする必要があります。

1. **Service Registry** インストールディレクトリに移動します。

   cd *ServiceRegistry-base*/**install**

2. テキストエディタで `install.properties` ファイルのコピーを開きます。

3. `registry.install.RequireDatabaseAuthentication` プロパティーの設定を `false` から `true` に変更します。

4. `registry.install.DatabaseUserID` プロパティーの設定を編集します。

   組み込みモードでは、この値は `APP` または空のいずれかです。

   ```
   registry.install.DatabaseUserID=APP

   registry.install.DatabaseUserID=
   ```

   ネットワークサーバーモードでは、この値を `APP` にしてください。

5. `registry.install.DatabasePassword` プロパティーの設定を編集します。

   パスワードは 6 文字以上にしてください。デフォルト値は `app123` です。

6. ファイルを保存して閉じます。

7. レジストリ用 **Application Server** ドメインを停止および再起動します。そのためには、次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base/**ant -f build-install.xml Dinstall.properties=** *props-file* **appserver.domain.bounce**

   ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

▼ データベースのオフラインバックアップを実行する

データベースを組み込みモードで実行している場合は、オフラインバックアップを実行してください。

1. **Service Registry** インストールディレクトリに移動します。

   cd *ServiceRegistry-base*/**install**

2 次のコマンドを実行します (すべてを 1 行で入力)。

Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `appserver.domain.stop`

ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

3 次のコマンドを実行します (すべてを 1 行で入力)。

Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `backup.db`

4 次のコマンドを実行してドメインを再起動します (すべてを 1 行で入力)。

Ant-base`/ant -f build-install.xml Dinstall.properties=`*props-file* `appserver.domain.start`

▼ 組み込みモードからネットワークサーバーモードに切り替える

組み込みモード (デフォルト) からネットワークサーバーモードに切り替えるには、`registry.install.clientDatabase` プロパティーの設定を `false` から `true` に変更し、データベース認証を要求します。

このプロパティーを編集してから、データベースリソースを再作成し、Application Server を停止および再起動します。

1 **Service Registry** インストールディレクトリに移動します。

`cd` *ServiceRegistry-base*`/install`

2 テキストエディタで `install.properties` ファイルのコピーを開きます。

3 プロパティー `registry.install.clientDatabase` の設定を `false` から `true` に変更します。

4 `registry.install.RequireDatabaseAuthentication` プロパティーの設定を `false` から `true` に変更します。

5 必要に応じて、`registry.install.DatabaseUserID` プロパティーの設定を編集します。

ネットワークサーバーモードでは、この値を `APP` にしてください。

6. `registry.install.DatabasePassword` プロパティーの設定を編集します。
   どのような長さでも有効です。デフォルト値は `app123` です。

7. ファイルを保存して閉じます。

8. データベース接続プールとそれに関連付けられたリソースを再作成します。そのためには、次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `appserver.jdbcResource.update`

   ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

9. レジストリ用 Application Server ドメインを停止および再起動します。そのためには、次のコマンドを実行します (すべてを 1 行で入力)。

   Ant-base`/ant -f build-install.xml Dinstall.properties=` *props-file* `appserver.domain.bounce`

10. Java DB データベースを起動します。そのためには、次のコマンドを実行します。

    `asadmin start-database --dbhome` *database-directory*

    デフォルトでは、このコマンドによってデータベースとそのログファイルはカレントディレクトリに格納されます。`--dbhome` オプションを使用して、データベースの場所を指定します (通常、これは *RegistryDomain-base* `/3.0/data/registry/soar`)。

次の手順

データベースがネットワークサーバーモードで実行中の場合は、データベースのオンラインバックアップを実行できます。

あとで組み込みモードに戻す場合は同じ手順に従いますが、次の点が異なります。

- `registry.install.clientDatabase` プロパティーの設定を `true` から `false` に変更します。
- データベース認証の要求を停止する場合は、`registry.install.RequireDatabaseAuthentication` プロパティーの設定を `true` から `false` に変更します。
- データベース認証を要求し続ける場合は、必要に応じてユーザー ID とパスワードを変更します。`registry.install.DatabaseUserID` の値は `APP` または空にしてください。`registry.install.DatabasePassword` の値はどのような長さにしてかまいません。
- 手順 10 を実行しないでください。データベースを個別に起動する必要はありません。

▼ データベースのオンラインバックアップを実行する

1 **Service Registry** インストールディレクトリに移動します。

cd *ServiceRegistry-base***/install**

2 `install.properties` ファイルのコピー内のプロパティー `registry.install.clientDatabase` が `true` に設定されていることを確認します。

3 次のコマンドを実行します (すべてを **1** 行で入力)。

*Ant-base***/ant -f build-install.xml Dinstall.properties=** *props-file* **backup.db**

ここで、*props-file* は、19 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root として設定する」または 20 ページの「インストール後の設定のあとでカスタムプロパティーを使って Service Registry を root 以外のユーザーとして設定する」で編集した `install.properties` ファイルのコピーのパス名です。

◆ ◆ ◆ 第 2 章

2 管理ツールの使用

この章では、Service Registry の管理ツールの使用方法について説明します。

この章の内容は次のとおりです。

管理ツールについて

Service Registry 管理ツールは、レジストリへの関連付けの追加やレジストリからのオブジェクトの削除といった一般的な管理作業を行うための、単純なコマンド行インタフェースを提供します。

このツールは次の 2 つのモードのいずれかで動作します。

- バッチモードでは、ツールのコマンド行で 1 つまたは複数のコマンドを指定します。
- 対話型モードでは、ツールの対話型シェル内でコマンドを入力します。

`ls` や `rm` などのいくつかのコマンドでは、ファイルやフォルダの操作に用いられる有名な UNIX® コマンドの名前と動作の両方を模倣しています。その他のコマンドについては、対応する UNIX コマンドはありません。

管理ツールの起動

管理ツールを起動するには、admin-tool.jar ファイルを実行します。

java *java-options* **-jar** *ServiceRegistry-base*/**lib/admin-tool.jar [***admin-tool-options***]...**

通常、java コマンドはディレクトリ /usr/jdk/entsys-j2se/bin にあります。

ServiceRegistry-base の場所は、Solaris OS では /opt/SUNWsrvc-registry、Linux および HP-UX システムでは /opt/sun/srvc-registry です。

ツールの起動時に表示される警告は無視しても安全です。

管理ツールを終了するには、quit コマンドを使用します。

バッチモード

管理ツールをバッチモードで実行するには、管理ツール起動時のコマンド行で -command オプションを指定します。

たとえば、次のコマンドでは ls コマンドが実行されます。

java -jar *ServiceRegistry-base*/**lib/admin-tool.jar -command "ls *.html"**

管理ツールは、コマンドとその応答を画面上にエコーし、コマンドの実行が完了すると処理を終了します。

シェルにとって特別な意味を持つ文字は、必ず正しくエスケープするようにしてください。

対話型モード

管理ツールを対話型モードで実行するには、コマンド行で -command 以外の任意のオプションを指定するか、オプションを指定せずに、管理ツールシェルを起動します。

java -jar *ServiceRegistry-base*/**lib/admin-tool.jar**

管理ツールは、次のプロンプトを表示し、ユーザーからの入力を待ちます。

```
admin>
```

管理ツールのコマンド行オプション

管理ツールは、47 ページの「形式」にリストされ、47 ページの「オプション」で説明されているコマンド行オプションを認識します。

形式

[-alias *alias*] [-command *commands*] [- debug] [-help] [-keypass *keypass*] [-localdir *localdir*] [-locale *locale*] [- registry *url*] [-root *locator* [-create]] [-sqlselect *SQL-statement*] [-verbose]

オプション

-alias　キーストア内のユーザーの証明書にアクセスする際に使用するエイリアス。ユーザー登録時に使用したエイリアスを指定します。管理ツールを使ってデータをレジストリに発行する場合、このオプションは必須です。

-command　対話型シェルからコマンドを取得しないで実行する管理ツールコマンドシーケンス。複数のコマンドを区切るには、セミコロン (;) を使用します。*commands* に quit コマンドを含める必要はありません。コマンド区切り文字でないセミコロンを使用する必要がある場合は、セミコロンの前に円記号を付けます。

```
\;
```

管理ツールを実行しているシェルが、円記号をもう 1 つの円記号でエスケープすることを要求する可能性があります。

```
\\;
```

コマンドに空白が含まれている場合、ツールがそのコマンドシーケンスを複数のコマンド行パラメータとしてではなく 1 つのパラメータとして処理できるように、そのシーケンス全体を一重引用符または二重引用符で囲みます。実行中のシェルでセミコロンもシェルコマンドの区切り文字として解釈される場合、複数の管理ツールコマンドのシーケンスは常に引用符で囲む必要があります。

-create　-root オプションで指定されたRegistryPackage オブジェクトとそのすべての親 RegistryPackage オブジェクトを、必要に応じて作成します。このオプションが有効になるのは、管理ツールの実行ユーザーがオブジェクト作成権限を持っている場合だけです。

-debug　デバッグ時に役立つ追加情報を出力します。

-help　これらのオプションの一覧を表示します。

-keypass　キーストア内のユーザーの証明書にアクセスする際に使用するパスワード。ユーザー登録時に使用したパスワードを指定します。管理ツールを使ってデータをレジストリに発行する場合、このオプションは必須です。

`-localdir`	ローカルファイルシステム内のベースディレクトリ。ローカルファイルシステム内のファイルに関係するコマンドの場合に指定します。
`-locale`	エラーメッセージや状態メッセージ用のリソースバンドルを選択する際に使用するロケール (`en`、`fr` など)。デフォルトは、Java 仮想マシン (JVM) によって決定されます。
`-registry`	接続先となる ebXML レジストリの URL。デフォルトは `http://localhost:6480/soar/registry/soap` です。
`-root`	リポジトリを `RegistryPackage` オブジェクトのツリーとして扱うコマンドにおいてベースとして使用される `RegistryPackage` のロケータ (`/registry/userData` など)。なお、こうしたツリー内の各オブジェクトには、ほかの `RegistryObject` オブジェクトや `RegistryPackage` オブジェクトが含まれます。デフォルトは、すべてのユーザーのデータ用に定義された `RegistryPackage` である、`/registry/userData` です。
`-sqlselect`	レジストリオブジェクトを選択するための *SQL-statement* を実行します。この文は `select` で始まる完全な SQL 文にするようにしてください。この SQL 文は、引用符で囲む必要がありますが、末尾にセミコロンを付ける必要はありません。このオプションを指定したあとで、引数なしの `select` コマンドを使用した場合、*SQL-statement* 以外の引数を持つ `select` コマンドを使用するまでその *SQL-statement* が実行されます。
`-v` | `-verbose`	状態メッセージの冗長出力を指定します。

注 – `--help` オプションの出力には、このリリースでサポートされていないオプションが 2 つ表示されます。`-class` と `-property` です。

管理ツールを使用した、コンテンツのレジストリへの発行

管理ツールコマンドには、コンテンツをレジストリに発行できるものがあります。たとえば、 `cp`、`import` などです。また、 `rm` コマンドでは、レジストリからコンテンツを削除できます。これらのコマンドを使用できるようにするには、追加の手順を実行してください。

▼ コンテンツをレジストリに発行できるようにする

1 『**Service Registry 3.1** ユーザーズガイド **(2006Q4)**』の「ユーザーアカウントの作成」の説明に従ってユーザー登録を行います。

指定したユーザー名とパスワードに加えて、ダウンロードした PKCS12 証明書の格納場所を覚えておきます。

2 管理ツールを起動します。

```
java -jar ServiceRegistry-base/lib/admin-tool.jar
```

3 keystoreMover コマンドを実行して、**PKCS12** 証明書を **JKS** キーストアにエクスポートします。詳細については、**66** ページの「keystoreMover」を参照してください。

通常は、コマンド例で示される 4 つのオプションのみ指定する必要があります。

4 管理ツールを停止します。

```
quit
```

5 再度、管理ツールを起動します。今回は次のようにオプションを指定します。

```
java -Djaxr-ebxml.security.storetype=JKS \
-Djaxr-ebxml.security.keystore=security/filename \
-Djaxr-ebxml.security.storepass=ebxmlrr \
-jar ServiceRegistry-base/lib/admin-tool.jar -alias alias -keypass password
```

ここで、*filename* は証明書ファイルの名前で、通常は keystore.jks です。場所 security/*filename* は、ディレクトリ $HOME/soar/3.0/jaxr-ebxml からの相対パスです。*alias* と *password* の値は、ユーザーアカウントを作成したときに指定した値です。

入力内容を保存するには、このコマンドを実行するスクリプトを作成します。

管理ツールの機能

ここでは、管理ツールの次の特徴について説明します。

権限

管理ツール使用時に実行可能なアクションは、ツール起動時に指定されたキーエイリアスとパスワードに対応するユーザーに許可されているアクションだけです。自らが所有しないオブジェクトを変更するコマンドを実行できるのは、管理者のロールを持つユーザーのみです。詳細については、27 ページの「管理者の作成」を参照してください。

例外の表示

管理ツールを使えば、コマンド失敗時に長いスタックトレースが表示されるのを回避できます。

コマンドが失敗した場合、管理ツールは、スタックトレースの 1 行目を出力したあと、次のメッセージを表示します。

```
An error occurred when executing the function.  Use the show exception
command to view messages.
```

より詳しい情報が必要な場合には、続けて `show exception` コマンドを実行して、完全なスタックトレースを確認します。

`show exception` コマンドは常に、直前のコマンドに対するスタックトレースを表示します。

レジストリオブジェクトの特定

レジストリオブジェクトの特定方法としてもっともよく使用されるのは、名前による特定です。ただし、通常は `registry` ルートから `RegistryPackage` までのパスによって `RegistryPackage` オブジェクトを特定します。たとえば、`/registry/userData` は、`userData` `RegistryPackage` へのパスです。

一部の名前一致では、ワイルドカードがサポートされています。1 文字に一致させるには疑問符 (`?`) を使用します。0 個以上の文字に一致させるにはアスタリスク (`*`) を使用します。

`cd` や `chown` など、一部のコマンドでは、URN (Uniform Resource Name) によるオブジェクトの特定がサポートされています。ただし、その場合、先頭の `urn:` まで含めて指定する必要があります。たとえば、
`urn:uuid:2702f889-3ced-4d49-82d1-e4cd846cb9e4` は有効な URN です。

chown コマンドおよび cp コマンドでは別の方法もサポートされています。その方法では、先行する users コマンドで表示された一覧内の特定の User を、*%number* を使って参照します。

一部のコマンドでは、空白を含む名前を入力できます。その場合、名前全体を二重引用符で囲むか、名前に含まれる各空白の直前に円記号を付けます。

select コマンドでは、SQL のワイルドカードがサポートされています。複数の文字に一致させるにはパーセント記号 (%) を、1 文字に一致させるにはアンダースコア (_) を、それぞれ使用します。

ロケールの名前指定への影響

RegistryObject (または RegistryPackage) は、異なるロケールに関連付けられた複数の名前を持つことができます。

ユーザーが指定したパスやオブジェクト名は、現在のロケールだけに基づいて評価されます。複数の名前を持つレジストリオブジェクトを名前で選択しようとした場合、ユーザーが入力した名前の一致対象となるのは、そのレジストリオブジェクトの名前の 1 つの選択肢 (現在のロケールにもっとも近いロケールを持つ選択肢) のみです。そのレジストリオブジェクトの複数の名前すべてが一致対象になるわけではありません。

たとえば、現在の RegistryPackage に、異なるロケールに関連付けられた 2 つの名前を持つ 1 つのメンバーオブジェクトがあるとします。具体的には、red が en (英語) ロケールに、rouge が fr (フランス語) ロケールに、それぞれ関連付けられています。現在のロケールが en である場合、コマンド ls rouge を実行してもそのメンバーオブジェクトは表示されませんが、ロケールが fr (またはそのバリアントの 1 つ) である場合には表示されます。

大文字と小文字の区別

管理ツールが認識するコマンド名やリテラルパラメータでは、大文字、小文字の区別はありません。たとえば、ls、Ls、LS はすべて同等です。

値が指定されたオプションは、そのオプションを使用するコードにそのまま渡されます。

管理ツールコマンドの使用

以下では、利用可能なコマンドについて説明します。

各コマンドの形式、オプション説明、およびオペランド説明は、次の文字体裁規約に従って記述されています。

- 斜体は、コマンド実行時に実際の値に置き換える、オプション引数またはオペランドを示します。
- オプションまたはオペランドの選択肢は、大括弧 ({ }) で囲まれます。ユーザーは、この括弧内のオプションまたはオペランドのいずれか 1 つを選択する必要があります。複数のオプションまたはオペランドは、縦棒 (|) で区切られます。
- 省略可能な 1 つ以上のオプションまたはオペランドは、角括弧 ([ ]) で囲まれます。
- 1つの オプションまたはオペランドに続く省略記号 (. . .) は、その引数またはオペランドを繰り返し指定できることを示します。

上記以外はすべて、コマンド実行時に含める必要のあるリテラルテキストです。

add association

`Association` オブジェクトをレジストリに追加します。

形式

add association -type *association-type sourceURN targetURN*

説明

add association コマンドは、指定されたタイプの Association オブジェクトをレジストリに追加します。

次のタイプのいずれかを使用できます。

- AccessControlPolicyFor
- AffiliatedWith (Subconcept として EmployeeOf と MemberOf を持つ)
- Contains
- ContentManagementServiceFor
- EquivalentTo
- Extends
- ExternallyLinks
- HasFederationMember
- HasMember
- Implements
- InstanceOf
- InvocationControlFileFor (Subconcept として CatalogingControlFileFor と ValidationControlFileFor を持つ)
- OffersService
- OwnerOf
- RelatedTo
- Replaces
- ResponsibleFor
- SubmitterOf
- Supersedes
- Uses

オプション

-type　Association オブジェクトのタイプ。

オペランド

sourceURN　ソースオブジェクトの URN。

targetURN　ターゲットオブジェクトの URN。

例

次のコマンド (すべてを 1 行で入力) は、指定された URN を持つ 2 つのオブジェクト間に RelatedTo 関係を作成します。

```
admin> add association -type RelatedTo
urn:uuid:ab80d8f7-3bea-4467-ad26-d04a40045446
urn:uuid:7a54bbca-2131-4a49-8ecc-e7b4ac86c4fd
```

add user

特定のユーザーをレジストリに追加します。

形式

add user [-edit] [- load *pathname*] [-firstname *string*] [-lastname *string*]
[-middleName *string*] -alias *string* -keypass *string* [-post1. -type *string*]
[- post1.city *string*] [-post1.country *string*] [-post1.postalcode *string*]
[-post1.stateOrProvince *string*] [-post1.street *string*] [-post2.streetNumber *string*]
[-post2. -type *string*] [- post2.city *string*] [-post2.country *string*]
[-post2.postalcode *string*] [-post2.stateOrProvince *string*] [-post2.street *string*]
[-post2.streetNumber *string*] [-post3. -type *string*] [- post3.city *string*]
[-post3.country *string*] [-post3.postalcode *string*] [-post3.stateOrProvince *string*]
[-post3.street *string*] [-post3.streetNumber *string*] [-telephone1. -type *string*]
[-telephone1.areaCode *string*] [-telephone1.countryCode *string*]
[-telephone1. -extension *string*] [-telephone1.number *string*]
[-telephone1. -URL *string*] [-telephone2. -type *string*] [-telephone2.areaCode *string*]
[-telephone2.countryCode *string*] [-telephone2. -extension *string*]
[-telephone2.number *string*] [-telephone2. -URL *string*] [-telephone3. -type *string*]
[-telephone3.areaCode *string*] [-telephone3.countryCode *string*]
[-telephone3. -extension *string*] [-telephone3.number *string*]
[-telephone3. -URL *string*] [-email1. -type *string*] [- email1.address *string*]
[-email2. -type *string*] [-email12address *string*] [-email3. -type *string*]
[-email3.address *string*]

説明

add user コマンドは、User オブジェクトを追加します。User オブジェクトには通常、PostalAddress、TelephoneNumber、EmailAddress の各オブジェクトが少なくとも 1 つずつ含まれます。

ユーザーに関する情報を指定するには、それらの情報をコマンド行から直接入力するか、あるいは -load オプションを使ってそれらの情報を含む Java プロパティーファイルを指定します。情報オプションと -load オプションは、コマンド行に指定された順番で評価されます。たとえば、いくつかのプロパティーをコマンド行から指定し、その他のプロパティーをプロパティーファイルから読み込んだあとで、そのプロパティーファイルの情報を後続のコマンド行オプションで上書きする、といったことも可能です。

新規ユーザーごとにアドレス、電話番号、および電子メールアドレスを最大 3 つずつ指定できます。それ以上必要な場合も、あとで Web コンソールや JAXR を使って追加できます。

住所、電話番号、電子メールアドレスのいずれかを指定する際には、タイプの値を指定してください。たとえば、-emailType OfficeEmail のように指定します。

コマンド行では、すべてのユーザーで必要となる基本情報のいくつかに対して短形式オプション (-fn など) が使用できるようになっています。ただし、そうした情報をプロパティーファイルで指定する場合には、長形式を使用してください。たとえば、-email1.address、-emailAddress 、-email のいずれかを使用して、ユーザーの 1 つ目の電子メールアドレスをコマンド行から指定できます。ただし、1 つ目の電子メールアドレスをプロパティーファイルで指定する場合には、email1.address= を使用してください。ユーザーの 2 つ目の電子メールアドレスに対するオプションは 1 つしか用意されていないため、コマンド行では -email2.address を、プロパティーファイル内では email2.address= を、それぞれ使用してください。

-edit オプションを指定した場合、新しいユーザーの情報を編集できるように、管理ツールによってエディタが起動されます。詳細については、オプション説明を参照してください。

このコマンドは、ホームディレクトリのファイル $HOME/soar/3.0/jaxr-ebxml/security/keystore.jks に新しいユーザーの証明書キーストアを作成します。ツールを root として実行している場合、ホームディレクトリは / または /root です。

注 – --load の読み込み対象または -edit の編集対象となるプロパティーファイルでは、その他のすべての Java プロパティーファイルと同じく、IS0-8859-1 文字セットを使用します。ISO-8859-1 に含まれない文字をプロパティーファイル内で表現する方法の詳細については、java.util.Properties.load(InputStream) のドキュメントを参照してください。

オプション

-edit
これを指定すると、新しいユーザーの情報を編集できるように、管理ツールによってエディタが起動されます。ツールによってエディタが起動されるのは、ほかのコマンド行パラメータが評価されたあとです。したがって、コマンド行またはプロパティーファイル内に指定された情報の評価結果に基づいて編集作業が開始されます。編集プログラムが正常に終了しないと、コマンドは続行されません。管理ツールは、set editor コマンド (73 ページの「set」 を参照) で指定されたエディタを起動します。これはデフォルトでは vi エディタになっています。

注 – このリリースでは、-edit は、emacsclient および NetBeans™ コマンド bin/runide.sh --open と組み合わせて使用できますが、動作は保証されておらず、vi と組み合わせた場合に正常に動作するかどうかは、明らかになっていません。

-load
ユーザーのプロパティーが格納された Java プロパティーファイルを指定します。そのプロパティー名は、add user コマンドの長形式オプション (lastName、post1.type など) と同一です。

-fn | -firstName
ユーザーの名を指定します。

-ln | -lastName
ユーザーの姓 (名字) を指定します。姓は必須です。コマンド行、プロパティーファイル内のいずれかで指定してください。

-mn | -middleName
ユーザーのミドルネームを指定します。

-alias
キーストア内のユーザーの証明書にアクセスする際に使用するエイリアス。 このオプションは必須です。エイリアスの文字長は 3 文字以上でなければなりません。

-keypass
キーストア内のユーザーの証明書にアクセスする際に使用するパスワード。 このオプションは必須です。パスワードの文字長は 6 文字以上でなければなりません。

-postalType | -post1.type
1 つ目の PostalAddress のタイプ。このタイプは必須です。コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列 (Office、Home など) です。

-city | -post1.city
1 つ目の PostalAddress の市。

-country | -post1.country
1 つ目の PostalAddress の国。

-postalCode | -postcode | - zip | -post1.postalcode
1 つ目の PostalAddress の郵便番号。

-stateOrProvince | -state | - province | -post1.stateOrProvince
1 つ目の PostalAddress の州または都道府県。

-street | -post1.street
1 つ目の PostalAddress の町名。

-streetNumber | -number | - -post1.streetNumber
1 つ目の PostalAddress の番地。

-post2.type
2 つ目の PostalAddress のタイプ。2 つ目の PostalAddress を指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列 (Office、Home など) です。

-post2.city
2 つ目の PostalAddress の市。

-post2.country
2 つ目の PostalAddress の国。

-post2.postalcode
2 つ目の PostalAddress の郵便番号。

-post2.stateOrProvince
2 つ目の PostalAddress の州または都道府県。

-post2.street
2 つ目の PostalAddress の町名。

-post2.streetNumber
2 つ目の PostalAddress の番地。

-post3.type
3 つ目の PostalAddress のタイプ。3 つ目の PostalAddress を指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列 (Office、Home など) です。

-post3.city
3 つ目の PostalAddress の市。

-post3.country
3 つ目の PostalAddress の国。

-post3.postalcode
3 つ目の PostalAddress の郵便番号。

-post3.stateOrProvince
3 つ目の PostalAddress の州または都道府県。

-post3.street
3 つ目の PostalAddress の町名。

-post3.streetNumber
3 つ目の PostalAddress の番地。

-phoneType | -telephone1.type
1 つ目の TelephoneNumber のタイプ。このタイプは必須です。コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。Beeper、FAX、HomePhone、MobilePhone、または OfficePhone。

-areaCode | -telephone1.areaCode
1 つ目の TelephoneNumber の市外局番。

-countryCode | -telephone1.countryCode
1 つ目の TelephoneNumber の国番号。

-extension | -telephone1.extension
1 つ目の TelephoneNumber の内線。

-number | -telephone1.number
1 つ目の TelephoneNumber の電話番号サフィックス。国番号や市外局番は含みません。この番号は必須です。コマンド行、プロパティーファイル内のいずれかで指定する必要があります。

-URL | -telephone1.URL
1 つ目の TelephoneNumber の URL (この番号に電子的に電話をかけることのできる URL)。

-telephone2.type
2 つ目の TelephoneNumber のタイプ。2 つ目の TelephoneNumber を指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定して

ください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。Beeper、FAX、HomePhone、MobilePhone、またはOfficePhone。

-telephone2.areaCode
2つ目のTelephoneNumberの市外局番。

-telephone2.countryCode
2つ目のTelephoneNumberの国番号。

-telephone2.extension
2つ目のTelephoneNumberの内線。

-telephone2.number
2つ目のTelephoneNumberの電話番号サフィックス。国番号や市外局番は含みません。2つ目のTelephoneNumberを指定する場合、必須である番号を、コマンド行、プロパティーファイル内のいずれかで指定してください。

-telephone2.URL
2つ目のTelephoneNumberのURL(この番号に電子的に電話をかけることのできるURL)。

-telephone3.type
3つ目のTelephoneNumberのタイプ。3つ目のTelephoneNumberを指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。Beeper、FAX、HomePhone、MobilePhone、またはOfficePhone。

-telephone3.areaCode
3つ目のTelephoneNumberの市外局番。

-telephone3.countryCode
3つ目のTelephoneNumberの国番号。

-telephone3.extension
3つ目のTelephoneNumberの内線。

-telephone3.number
3つ目のTelephoneNumberの電話番号サフィックス。国番号や市外局番は含みません。3つ目のTelephoneNumberを指定する場合、必須である番号を、コマンド行、プロパティーファイル内のいずれかで指定してください。

-telephone3.URL
3つ目のTelephoneNumberのURL(この番号に電子的に電話をかけることのできるURL)。

-emailType | -email1.type
1つ目のEmailAddressのタイプ。このタイプは必須です。コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。HomeEmailまたはOfficeEmail。

`-emailAddress | -email | - email1.address`
1 つ目の電子メールアドレス。1 つ目の電子メールアドレスは必須です。

`-email2.type`
2 つ目の `EmailAddress` のタイプ。2 つ目の `EmailAddress` を指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。`HomeEmail` または `OfficeEmail`。

`-email2.address`
2 つ目の電子メールアドレス。

`-email3.type`
3 つ目の `EmailAddress` のタイプ。3 つ目の `EmailAddress` を指定する場合、必須であるタイプを、コマンド行、プロパティーファイル内のいずれかで指定してください。値は任意の文字列ですが、次の既知のタイプのいずれかを指定できます。`HomeEmail` または `OfficeEmail`。

`-email3.address`
3 つ目の電子メールアドレス。

例

次のコマンドは、ユーザーのホームディレクトリ内のファイル `JaneSmith.properties` から `User` のプロパティーを読み込みます。

```
admin> add user -load ~/JaneSmith.properties
```

次のコマンド (すべてを 1 行で入力) では、`User` オブジェクトの作成に最低限必要なプロパティーを指定しています。

```
admin> add user -ln Smith -postaltype Office -country US
-phonetype Office -number 333-3333 -emailtype OfficeEmail
-emailaddress JaneSmith@JaneSmith.com -alias 123 -keypass 123456
```

cd

`RegistryPackage` の場所を変更します。

形式

cd { *locator* | *URN*}

説明

`cd` コマンドは、指定されたパスまたは指定された URN を持つ `RegistryPackage` にディレクトリ (メタファー) を変更します。

(現在のロケールに対して) 同じパスを持つ RegistryPackage オブジェクトが複数存在する場合には、指定された URN に変更します。

オペランド

locator レジストリのルートからリポジトリ内の特定のオブジェクトまでの、複数のレジストリオブジェクト名から成るパス。各オブジェクト名の前にはスラッシュ (/) を付けます。

たとえば、registry RegistryPackage (それ自体はいずれの RegistryPackage のメンバーでもない) のメンバーである userData RegistryPackage のロケータは /registry/userData です。userData RegistryPackage のメンバーである folder1 RegistryPackage のロケータは /registry/userData/folder1 です。

URN RegistryPackage のURN。これは、urn: で始まる URN でなければいけません。

例

次のコマンドは、URN urn:uuid:92d3fd01-a929-4eba-a5b4-a3f036733017 を持つ RegistryPackage にディレクトリを変更します。

```
admin> cd urn:uuid:92d3fd01-a929-4eba-a5b4-a3f036733017
```

次のコマンドは、場所 /registry/userData/myData にディレクトリを変更します。

```
admin> cd /registry/userData/myData
```

chown

特定の RegistryObject の所有者を変更します。

形式

chown { *URN* | %*index*}

説明

chown コマンドは、先行する select コマンドで選択されたオブジェクトの所有者を、URN として指定されたユーザー、または先行する users コマンドで表示された一覧内の URN への参照として指定されたユーザーに変更します。

このコマンドを正常に実行できるのは、管理者のロールを持つユーザーだけです。

オペランド

URN URN によって指定された User オブジェクト。

%index 先行する users コマンドで表示された一覧内に含まれる User オブジェクトの URN への番号参照。

例

次のコマンドは、選択されたオブジェクトの所有者を、URN urn:uuid:26aa17e6-d669-4775-bfe8-a3a484d3e079 で指定されるユーザーに変更します。

```
admin> chown urn:uuid:26aa17e6-d669-4775-bfe8-a3a484d3e079
```

次のコマンドは、選択されたオブジェクトの所有者を、先行する users コマンドで番号 2 に対応しているユーザーに変更します。

```
admin> chown %2
```

関連項目

- 73 ページの「select」
- 75 ページの「users」

cp

ファイルとフォルダをレジストリにコピーします。

形式

cp [-owner {*URN* | % *index*}] [- exclude *pattern*]… [-include *pattern*]… *pattern*…

説明

cp コマンドは、フォルダおよびファイルをそれぞれ RegistryPackage オブジェクトおよび ExtrinsicObject オブジェクトとして、レジストリ内にコピーします。

ファイルやフォルダのコピー元となるローカルファイルシステム上のローカルディレクトリはデフォルトで、管理ツールの起動元である現在のディレクトリになります。-localdir オプションを使えば、管理ツール起動時にローカルディレクトリを変更できます。lcd コマンドを使えば、管理ツール起動後にローカルディレクトリを変更できます。show localdir コマンドを使えば、現在のローカルディレクトリの絶対パスを取得できます。

このコマンドは再帰的です。つまり、ユーザーが特定のディレクトリを指定すると、このコマンドはそのディレクトリの下にあるすべてのファイルとフォルダをコピーします。

オプション

`-owner` 引数の *URN* または *%index* に指定されたユーザーに、コピー先のレジストリオブジェクトの所有者を設定します。これらの引数については、`chown` コマンドの説明を参照してください。自分以外の所有者を指定するには、管理者のロールを持っている必要があります。

`-exclude` 指定されたパターンを名前に含むファイル以外のすべてのファイルをコピーします。ここで、*pattern* は、リテラル文字と、特殊文字のアスタリスク (*) (0 個以上の文字を表現) または疑問符 (?) (1 文字を表現) で構成されるパターンです。

このオプションは複数回指定可能です。

`-include` 指定されたパターンを名前に含むファイルをすべてコピーします。ここで、*pattern* は、リテラル文字と、特殊文字のアスタリスク (*) (0 個以上の文字を表現) または疑問符 (?) (1 文字を表現) で構成されるパターンです。

このオプションは複数回指定可能です。

オペランド

pattern コピー対象のファイルまたはフォルダ。リテラル文字と、特殊文字のアスタリスク (*) (0 個以上の文字を表現) または疑問符 (?) (1 文字を表現) で構成されるパターンを使って指定します。*pattern* は複数個指定可能です。

例

次のコマンドは、ディレクトリ `mydir` をレジストリにコピーし、その所有者として、先行する `users` コマンドの番号 4 に対応するユーザーを設定します。

```
admin> cp -owner %4 mydir
```

次のコマンドは、ディレクトリ `mydir` をレジストリにコピーします。ただし、文字列 `.z`、`.c` のいずれかで終わるファイルやディレクトリは除外されます。

```
admin> cp mydir -exclude \\.z -exclude \\.c
```

関連項目

- 68 ページの「lcd」
- 74 ページの「show」

echo

特定の文字列をエコーします。

形式

echo *string*

説明

echo コマンドは、指定された *string* を出力先にエコーします。このコマンドは、管理ツールをバッチモードで実行する場合に -command オプション内で指定すると、非常に便利です。

オペランド

string　　文字シーケンス。

例

次のコマンドは、「date」と ls コマンドの結果をログファイルに出力します。

```
java -jar admin-tool.jar -command "echo ”date”; ls" > admin.log
```

help

コマンドに関する情報を表示します。

形式

help [*command_name*]

説明

help コマンドは、利用可能なコマンドまたは指定された特定のコマンドに関する情報を表示します。

add や show など、サブコマンドを持つコマンドが指定された場合、help コマンドはそのサブコマンドに関する情報を表示します。

引数が指定されなかった場合、help コマンドは、すべてのコマンドの使用法に関する情報を表示します。

オペランド

command_name　　特定の管理ツールコマンドの名前。

例

次のコマンドは、すべてのコマンドの使用法に関する情報を表示します。

```
admin> help
```

次のコマンドは、lcd コマンドの使用法に関する情報を表示します。

```
admin> help lcd
```

次のコマンドは、add サブコマンドの使用法に関する情報を表示します。

```
admin> help add
```

import

レジストリオブジェクトを定義する XML ファイルをインポートします。

形式

import [{-a | --attach}*pathname*, *mimeType*, *id*]… *submitObjectsRequest*

説明

import コマンドは、 ebXML Registry Services and Protocols Version 3.0 の仕様で規定される SubmitObjectsRequest プロトコルに従った XML ファイルを送信することで、1 つ以上の新しいオブジェクトまたはリポジトリ項目をレジストリに追加します。

XML ファイルおよび指定した任意の添付ファイルは、レジストリに送られてそこで処理される SOAP メッセージのコンテンツを形成します。したがって、この処理は極めて低レベルであり、ebXML の仕様を熟知しているユーザーのために用意されています。

オプション

`-a|--attach` 付帯オブジェクトのリポジトリ項目としてファイルを添付します。*pathname* は、追加されるファイルのパス名です。*mimeType* には、ファイルの MIME タイプを指定します。*id* は、リポジトリ項目が属する付帯オブジェクトの一意の識別子です。このオプションは複数回指定可能です。

オペランド

submitObjectsRequest レジストリオブジェクトの定義を含む XML ファイル。

例

次のコマンドは、ファイル `MyRequest.xml` に定義されたオブジェクトのグループをインポートします。

```
admin> import MyRequest.xml
```

次のコマンド (すべてを 1 行で入力) は、付帯オブジェクトとそのリポジトリ項目であるイメージファイルをインポートします。

```
admin> import --attach chicken.jpg, image/jpeg, urn:bird:poultry:chicken
ChickenRequest.xml
```

keystoreMover

あるキーストア形式から別のキーストア形式へ 1 つ以上のキーをエクスポートします。

形式

keystoreMover [-sourceKeystoreType {JKS | PKCS12}] -sourceKeystorePath *pathname* -sourceKeystorePassword *password* [-sourceAlias *alias* [-sourceKeyPassword *password*]] [-destinationKeystoreType {JKS | PKCS12}] -destinationKeystorePath *pathname* -destinationKeystorePassword *password* [-destinationAlias *newAlias*] [-destinationKeyPassword *password*]

説明

レジストリの開発者インタフェースには JKS キーストアの使用が必要です。一方、Web コンソールには、Web ブラウザにインポート可能な PKCS12 または DER の証明書が必要です。

レジストリで生成された PKCS12 証明書を使用してユーザーアカウントを作成し、管理ツールを使用してコンテンツをレジストリに発行する場合は、keystoreMover コマンドを使用して証明書を JKS キーストアにエクスポートします。また、レジストリに対して開発者アプリケーションを実行するためにこのコマンドを使用することも可能です。

add user コマンドを使用してユーザーを作成し、そのユーザーが Web コンソールを使えるようにする場合は、このコマンドを使用して、add user によって作成された JKS キーストアを PKCS12 形式にエクスポートすることができます。

Web コンソールを使用してユーザーアカウントを作成する方法の詳細については、『Service Registry 3.1 ユーザーズガイド (2006Q4)』の「ユーザーアカウントの作成」を参照してください。管理ツールと連携して keystoreMover を使用する方法については、48 ページの「管理ツールを使用した、コンテンツのレジストリへの発行」を参照してください。add user コマンドの使用方法については、54 ページの「add user」を参照してください。

レジストリに関するアプリケーションの開発については、『Service Registry 3.1 開発ガイド』を参照してください。

オプション

-sourceKeystoreType	エクスポートするキーストアのタイプを指定します。引数は PKCS12 または JKS のいずれかにしてください。デフォルトは PKCS12 です。
-sourceKeystorePath	エクスポート元キーストアを含むファイルのパス名を指定します。 このオプションは必須です。通常、これはユーザーの作成時に作成された証明書ファイルのパス名です。
-sourceKeystorePassword	エクスポート元キーストアのパスワードを指定します。通常、これはユーザーの作成時に指定したパスワードです。 このオプションは必須です。
-sourceAlias	エクスポートするエイリアスを指定します。このオプションを指定しないと、キーストア内のすべてのエイリアスがエクスポートされます。Web コンソールからダウンロードされたキーストアに含まれるエイリアスは 1 つだけです。
-sourceKeyPassword	キーストアパスワードではなく、エイリアスに固有のパスワードを指定します。このオプションを指定しないと、パスワードはキーストアパスワードと同じになります (通常はこのようになる)。

-destinationKeystoreType	エクスポート先キーストアのタイプを指定します。引数は JKS または PKCS12 のいずれかにすることができます。デフォルトは JKS です。
-destinationKeystorePath	エクスポート先キーストアを含ませるファイルのパス名を指定します。 このオプションは必須です。通常、この引数は *HOME*/soar/3.0/jaxr-ebxml/security/keystore.jks です。ここで、 *HOME* はユーザーのホームディレクトリです。
-destinationKeystorePassword	エクスポート先キーストアのパスワードを指定します。この引数は必須です。このプロパティーのデフォルト値は ebxmlrr です。
-destinationAlias	エイリアス名を変更する場合、新しいエイリアス名を指定します。このオプションを指定しないと、新しいエイリアスの名前はエクスポート元の証明書のエイリアスと同じになります。
-destinationKeyPassword	キーストアパスワードではなく、エイリアスに固有のパスワードを指定します。このオプションを指定しないと、パスワードはキーストアパスワードと同じになります (通常はこのようになる)。

注 – すべてのパスワードは 6 文字以上にしてください。

例

次のコマンドは、ユーザーのホームディレクトリに存在する generated-key.p12 の証明書を、同様にユーザーのホームディレクトリに存在する soar/3.0/jaxr-ebxml/security/keystore.jks の JKS キーストアにエクスポートします。エクスポート元キーストアのパスワードは、ユーザーがレジストリに登録されたときに指定されたパスワードです。エクスポート先キーストアのパスワードは、ebxmlrr のデフォルト値です。コマンドはすべて 1 行で入力します。

```
admin> keystoreMover -sourceKeystorePath /home/myname/generated-key.p12
-sourceKeystorePassword mypass -destinationKeystorePath
/home/myname/soar/3.0/jaxr-ebxml/security/keystore.jks
-destinationKeystorePassword ebxmlrr
```

lcd

ローカルファイルシステム上の現在のディレクトリを変更します。

形式

lcd [*pathname*]

説明

lcd コマンドは、ローカルファイルシステム上の現在のローカルディレクトリを変更します。

引数が指定されなかった場合、lcd コマンドは、現在のディレクトリをユーザーのデフォルトホームディレクトリに変更します。

オペランド

pathname ディレクトリ名。絶対パス、相対パスのいずれを使ってもかまいません。

例

次のコマンドは、現在のローカルディレクトリを /usr/share ディレクトリに変更します。

```
admin> lcd /usr/share
```

次のコマンドは、ローカルファイルシステム上の現在のローカルディレクトリをユーザーのデフォルトホームディレクトリに変更します。

```
admin> lcd
```

ls

現在の RegistryPackage 内のオブジェクトを一覧表示します。

形式

ls [{*pattern* | *URN*}…]

説明

引数が指定されなかった場合、ls コマンドは、現在の RegistryPackage 内のオブジェクトを一覧表示します。*pattern* または *URN* が指定された場合、現在の RegistryPackage 内のオブジェクトのうち、(現在のロケールにおける) 名前または一意の識別子が *pattern* または *URN* に一致するものを一覧表示します。

オペランド

pattern リテラル文字と、特殊文字のアスタリスク (*) (0 個以上の文字を表現) または疑問符 (?) (1 文字を表現) で構成されるパターン。*pattern* は複数個指定可能です。

URN `urn:uuid:4a6741e7-4be1-4cfb-960a-e5520356c4fd` などの、`urn:` で始まる URN。*URN* は複数個指定可能です。この URN は、オブジェクトの論理識別子ではなく、一意の識別子である必要があります。

例

次のコマンドは、現在の `RegistryPackage` 内のすべてのオブジェクトを一覧表示します。

```
admin> ls
```

次のコマンドは、パターン `urn:bird:poultry:chicken` に一致する名前を持つか `urn:bird:poultry:chicken` を ID に持つすべてのオブジェクトを一覧表示します。

```
admin> ls urn:bird:poultry:chicken
```

次のコマンドは、パターン `*bird*` に一致する名前を持つすべてのオブジェクトを一覧表示します (`*bird*` が有効な ID である場合、このコマンドは、`*bird*` を ID に持つオブジェクトも一覧表示する)。

```
admin> ls *bird*
```

次のコマンドは、パターン `*bird*` に一致する名前を持つか、パターン `urn:bird:poultry:chicken` に一致する名前を持つか、`urn:bird:poultry:chicken` を ID に持つような、すべてのオブジェクトを一覧表示します。

```
admin> ls *bird* urn:bird:poultry:chicken
```

pwd

現在の `RegistryPackage` へのパスを表示します。

形式

pwd

説明

pwd コマンドは、現在の RegistryPackage への 1 つまたは複数のパスを、現在のロケールにもっともふさわしい名前を使って表示します。また、そのパスに対するロケールも表示します。

例

```
admin> pwd
(en_US) /registry/userData
```

quit

管理ツールを終了します。

形式

quit

説明

quit コマンドは管理ツールを終了します。

例

```
admin> quit
```

rm

特定の RegistryPackage からオブジェクトを削除します。

形式

rm [-d] [-r] { *pattern* | *URN*}…

説明

rm コマンドは、現在の RegistryPackage のメンバーオブジェクトのうち、現在のロケールにおける名前が *pattern* または *URN* で指定されたパターンに一致するものを削除します。

一致する RegistryObject が複数の RegistryPackage オブジェクトのメンバーになっている場合、このコマンドは、現在の RegistryPackage とそのオブジェクトとの間の関

連付けだけを削除します。そのオブジェクトがレジストリから削除されるのは、関連付けの削除によって、包含 RegistryPackage オブジェクトを含むあらゆる RegistryObject との関連付けを持たないオブジェクトが残される場合だけです。

一致するメンバーオブジェクト自身がほかのオブジェクトを含む RegistryPackage である場合には、-r、-d のいずれかのオプションが指定されない限り、そのオブジェクトが削除されたり、現在の RegistryPackage とそのメンバー RegistryPackage との間の関連付けが削除されたりすることはありません。

-d オプションと -r オプションが両方とも指定された場合は -d オプションが再帰的に適用されますが、これは、-r の選択対象となるすべてのオブジェクト (とそれらの関連付け) が、ほかの関連付けの有無にかかわらず削除されるようにするためです。

オプション

-d 現在の RegistryPackage と指定された RegistryPackage との間の関連付けを削除します。指定された RegistryPackage が削除されるのは、残りの関連付けが自身のメンバーオブジェクトに対するものだけである場合に限ります。削除された RegistryPackage のメンバーオブジェクトのうち、ほかの HasMember 関連付けのターゲットとして固定されていないものは、レジストリのルートのメンバーとしてアクセスできるようになります。

-r 指定された RegistryPackage オブジェクトとその配下にあるすべてのオブジェクトを削除します (ただし、ほかの関連付けを持つオブジェクトが存在する場合を除く)。

オペランド

pattern リテラル文字と、特殊文字のアスタリスク (*) (0 個以上の文字を表現) または疑問符 (?) (1 文字を表現) で構成されるパターン。*pattern* は複数個指定可能です。

URN urn:uuid:4a6741e7-4be1-4cfb-960a-e5520356c4fd などの、urn: で始まる URN。*URN* は複数個指定可能です。

例

次のコマンドは、文字列 "stat" を含むすべての RegistryPackage オブジェクトと、それらの配下にあるすべてのオブジェクトを削除します。

```
admin> rm -r *stat*
```

select

SQL の select 文を実行します。

形式

select [*SQL*]

説明

select コマンドは、コマンド全体を 1 つの SQL クエリーとして評価することで、指定されたオブジェクトを選択および一覧表示します。引数が指定されなかった場合、このコマンドは、先行する select コマンドまたは -sqlselect オプションによって選択されたすべてのオブジェクトを一覧表示します。

オペランド

SQL SQL の select 文 (ただし、先頭の select はすでにコマンド名として存在しているので含めない)。

例

次のコマンドは、レジストリ内のすべての ClassificationScheme オブジェクトを一覧表示します。

```
admin> select s.* from ClassificationScheme s
```

set

特定のプロパティーの値を設定します。

形式

set *property value*

説明

set コマンドは、管理ツールシェルの特定のプロパティーの値を設定します。

ツールがサポートするプロパティーと値は、次のとおりです。

set debug {true | on | yes | false | off | no}

デバッグメッセージの出力を有効または無効にします。

set editor *string*

対話型エディタ起動時に管理ツールが使用するコマンドを設定します。UNIX および Linux システムでのデフォルト値は `/bin/vi` です。

set verbose {true | on | yes | false | off | no}

コマンド実行時における冗長メッセージの出力を有効または無効にします。

オペランド

property 次のプロパティーのいずれかです。`debug`, `editor`, `verbose`.

value 指定されたプロパティーのサポートされている値。詳細については、「説明」を参照してください。

例

次のコマンドは、エディタをデフォルトの `/bin/vi` ではなく、`/usr/bin/vi` に設定します。

```
admin> set editor /usr/bin/vi
```

次のコマンドは、デバッグをオンにします。

```
admin> set debug true
```

次のコマンドは、冗長な出力をオフにします。

```
admin> set verbose off
```

show

特定のプロパティーの値を表示します。

形式

show [*property*]

説明

`show` コマンドは、管理ツールシェルの特定のプロパティーの値を表示します。

引数が指定されなかった場合、このコマンドはすべてのプロパティーの値を表示します。

このコマンドがサポートするプロパティーは、次のとおりです。

`debug` デバッグ出力が有効になっているかどうか。

`editor` 対話型エディタ起動時に管理ツールが使用するエディタ。

`exception` 直前に実行されたコマンドで発生した例外スタックトレース (存在する場合のみ)。

`localdir` ローカルファイルシステム上の現在のディレクトリ。このプロパティーを設定するには `lcd` コマンドを使用します。詳細については、68 ページの「`lcd`」を参照してください。

`locale` 現在のロケール。

`verbose` 冗長出力が有効になっているかどうか。

オペランド

property 現在の値を表示すべきプロパティー。プロパティー `exception` と `locale` については、表示は可能ですが、`set` コマンドによる値の設定は行えません。

例

次のコマンドは、直前のコマンドで発生した例外を表示します。

```
admin> show exception
```

users

現在の `User` オブジェクトを一覧表示します。

形式

`users`

説明

`users` コマンドは、現在レジストリ内に存在している `User` オブジェクトを一覧表示します。

出力の書式は次のとおりです。

%index: *URN lastname, firstname middlename*

出力内の *index* は数値ですが、chown コマンドまたは cp コマンドを実行する際にこの数値をパーセント記号 (%) も含めて使用すれば、特定のユーザーを参照することができます。*lastname*、*firstname*、および *middlename* は、ユーザーの姓、名、およびミドルネームです。

例

次のコマンドは、現在のユーザーを表示します。

```
admin> users
%0:  urn:freebxml:registry:predefinedusers:registryoperator  Operator, Registry
%1:  urn:freebxml:registry:predefinedusers:registryguest  Guest, Registry
%2:  urn:freebxml:registry:predefinedusers:farrukh  Najmi, Farrukh Salahudin
%3:  urn:freebxml:registry:predefinedusers:nikola  Stojanovic, Nikola
%4:  urn:uuid:799cc524-b7cd-4e51-8b34-d93b79ac52de  User, Test
%5:  urn:uuid:85428d8e-1bd5-473b-a8c8-b9d595f82728  Parker, Miles
```

関連項目

- 61 ページの「chown」
- 62 ページの「cp」

索引