



# BEA WebLogic Server™

## BEA WebLogic Server パートナーズ ガイド

BEA WebLogic Server バージョン 7.0  
マニュアルの日付 : 2002 年 6 月  
改訂 : 2004 年 4 月 12 日

## 著作権

Copyright © 2002 BEA Systems, Inc. All Rights Reserved.

## 限定的権利条項

本ソフトウェアおよびマニュアルは、**BEA Systems, Inc.** 又は日本ビー・イー・エー・システムズ株式会社（以下、「**BEA**」といいます）の使用許諾契約に基づいて提供され、その内容に同意する場合にのみ使用することができ、同契約の条項通りにのみ使用またはコピーすることができます。同契約で明示的に許可されている以外の方法で同ソフトウェアをコピーすることは法律に違反します。このマニュアルの一部または全部を、**BEA** からの書面による事前の同意なしに、複写、複製、翻訳、あるいはいかなる電子媒体または機械可読形式への変換も行うことはできません。

米国政府による使用、複製もしくは開示は、**BEA** の使用許諾契約、および FAR 52.227-19 の「Commercial Computer Software-Restricted Rights」条項のサブパラグラフ (c)(1)、DFARS 252.227-7013 の「Rights in Technical Data and Computer Software」条項のサブパラグラフ (c)(1)(ii)、NASA FAR 補遺 16-52.227-86 の「Commercial Computer Software--Licensing」条項のサブパラグラフ (d)、もしくはそれらと同等の条項で定める制限の対象となります。

このマニュアルに記載されている内容は予告なく変更されることがあり、また **BEA** による責務を意味するものではありません。本ソフトウェアおよびマニュアルは「現状のまま」提供され、商品性や特定用途への適合性を始めとする（ただし、これらには限定されない）いかなる種類の保証も与えません。さらに、**BEA** は、正当性、正確さ、信頼性などについて、本ソフトウェアまたはマニュアルの使用もしくは使用結果に関していかなる確約、保証、あるいは表明も行いません。

## 商標または登録商標

**BEA**、**Jolt**、**Tuxedo**、および **WebLogic** は **BEA Systems, Inc.** の登録商標です。**BEA Builder**、**BEA Campaign Manager for WebLogic**、**BEA eLink**、**BEA Manager**、**BEA WebLogic Commerce Server**、**BEA WebLogic Enterprise**、**BEA WebLogic Enterprise Platform**、**BEA WebLogic Express**、**BEA WebLogic Integration**、**BEA WebLogic Personalization Server**、**BEA WebLogic Platform**、**BEA WebLogic Portal**、**BEA WebLogic Server**、**BEA WebLogic Workshop**、および **How Business Becomes E-Business** は、**BEA Systems, Inc** の商標です。

その他の商標はすべて、関係各社がその権利を有します。

## BEA WebLogic Server パートナーズ ガイド

パート番号	マニュアルの日付	ソフトウェアのバージョン
なし	2004年4月12日	BEA WebLogic Server バージョン 7.0

---

# 目次

## このマニュアルの内容

対象読者.....	v
e-docs Web サイト.....	vi
このマニュアルの印刷方法.....	vi
関連情報.....	vi
サポート情報.....	vii
表記規則.....	viii

## 1. WebLogic Server の配布

WebLogic Server の新リリースを配布するための ISV ライセンスのアップグレード.....	1-2
BEA Partner Program への登録.....	1-2
パートナー開発キットのインストール.....	1-3
ISV ライセンスのインストール.....	1-3
手順 1: ISV ライセンスをインストールする準備.....	1-4
手順 2: ライセンスデータの抽出と WebLogic Server ファイルへのリンク.....	1-5
手順 3: WebLogic Server ライセンスの更新.....	1-6
次の手順: アプリケーションと WebLogic Server のコンフィグレーション.....	1-6
ファイルの配布.....	1-7

## 2. WebLogic Server コンフィグレーション ファイルのカスタマイズ

config.xml ファイルのカスタマイズ.....	2-1
アプリケーション リソースのプリコンフィグレーション.....	2-2
アプリケーション コンポーネントのデプロイメント.....	2-3
コンフィグレーションの例.....	2-4
ドメインのコンフィグレーション.....	2-5
基本的なサーバの設定.....	2-5
JDBC 要件.....	2-6

---

アプリケーション コンポーネント.....	2-8
起動クラス.....	2-9
互換性セキュリティに対するファイルのカスタマイズ.....	2-9

### **3. JDBC プロファイリング MBean の使用**

JDBC プロファイリングの有効化 .....	3-1
JDBC プロファイルへのアクセス .....	3-3

---

# このマニュアルの内容

このマニュアルでは、独立ソフトウェアベンダ（ISV）ライセンスを取得およびインストールして、BEAのコアテクノロジーとアプリケーションをまとめて1つの製品として配布する方法について説明します。また、BEA WebLogic Server™とアプリケーションをまとめるための開発技術も紹介します。

このマニュアルの構成は次のとおりです。

- 第1章「WebLogic Server の配布」では、ISVライセンスの取得およびインストール方法と、配布に含める必要のある WebLogic Server ファイルについて説明します。
- 第2章「WebLogic Server コンフィグレーション ファイルのカスタマイズ」では、パートナーおよび ISV が、アプリケーションとともに配布する WebLogic Server コンフィグレーション ファイルに対して行う一般的な変更事項について説明します。
- 第3章「JDBC プロファイリング MBean の使用」では、JDBC プロファイリングを有効にする方法および使用方法について説明します。

## 対象読者

このマニュアルは、BEA WebLogic Server のコアテクノロジーを使用するカスタムアプリケーションの作成に関心がある独立ソフトウェアベンダ（ISV）とその他の開発者を対象としています。BEA WebLogic Server プラットフォーム、WebLogic Server マニュアルセットに収められている他のマニュアルの内容、および Java プログラミング言語に読者が精通していることを前提として書かれています。

---

# e-docs Web サイト

BEA 製品のドキュメントは、BEA の Web サイトで入手できます。BEA のホームページで [製品のドキュメント] をクリックするか、または WebLogic Server 製品ドキュメント ページ (<http://edocs.beasys.co.jp/e-docs/index.html>) を直接表示してください。

## このマニュアルの印刷方法

Web ブラウザの [ファイル | 印刷] オプションを使用すると、Web ブラウザからこのマニュアルを一度に 1 章ずつ印刷できます。

このマニュアルの PDF 版は、WebLogic Server の Web サイトで入手できます。PDF を Adobe Acrobat Reader で開くと、マニュアルの全体（または一部分）を書籍の形式で印刷できます。PDF を表示するには、WebLogic Server ドキュメントのホームページを開き、[ドキュメントのダウンロード] をクリックして、印刷するマニュアルを選択します。

Adobe Acrobat Reader は Adobe の Web サイト (<http://www.adobe.co.jp>) で無料で入手できます。

## 関連情報

BEA の Web サイトでは、WebLogic Server の全マニュアルを提供しています。以下の BEA WebLogic Server のマニュアルには、WebLogic Server を拡張する方法を理解するための関連情報が含まれています。

- BEA WebLogic Server のマニュアル（オンラインで入手可能）
  - 管理者ガイド
  - プログラミング ガイド

- 
- WebLogic Server API
  - Sun Microsystems, Inc. の Java サイト (<http://java.sun.com/>)

BEA WebLogic Server と Java の詳細については、<http://edocs.beasys.co.jp/e-docs/> の参考文献を参照してください。

## サポート情報

BEA のドキュメントに関するユーザからのフィードバックは弊社にとって非常に重要です。質問や意見などがあれば、電子メールで [docsupport-jp@beasys.com](mailto:docsupport-jp@beasys.com) までお送りください。寄せられた意見については、WebLogic Server のドキュメントを作成および改訂する BEA の専門の担当者が直に目を通します。

電子メールのメッセージには、ご使用のソフトウェアの名前とバージョン、およびドキュメントのタイトルと日付をお書き添えください。本バージョンの BEA WebLogic Server について不明な点がある場合、または BEA WebLogic Server のインストールおよび動作に問題がある場合は、BEA WebSupport ([www.bea.com](http://www.bea.com)) を通じて BEA カスタマサポートまでお問い合わせください。カスタマサポートへの連絡方法については、製品パッケージに同梱されているカスタマサポート カードにも記載されています。

カスタマサポートでは以下の情報をお尋ねしますので、お問い合わせの際はあらかじめご用意ください。

- お名前、電子メール アドレス、電話番号、ファクス番号
- 会社の名前と住所
- お使いの機種とコード番号
- 製品の名前とバージョン
- 問題の状況と表示されるエラー メッセージの内容

---

# 表記規則

このマニュアルでは、全体を通して以下の表記規則が使用されています。

表記法	適用
[Ctrl] + [Tab]	複数のキーを同時に押すことを示す。
<i>斜体</i>	強調または書籍のタイトルを示す。
等幅テキスト	コード サンプル、コマンドとそのオプション、データ構造体とそのメンバー、データ型、ディレクトリ、およびファイル名とその拡張子を示す。等幅テキストはキーボードから入力するテキストも示す。 例： <pre>import java.util.Enumeration; chmod u+w * config/examples/applications .java config.xml float</pre>
<i>斜体の等幅テキスト</i>	コード内の変数を示す。 例： <pre>String <i>CustomerName</i>;</pre>
すべて大文字のテキスト	デバイス名、環境変数、および論理演算子を示す。 例： <pre>LPT1 BEA_HOME OR</pre>
{ }	構文の中で複数の選択肢を示す。

表記法	適用
[ ]	<p>構文の中で任意指定の項目を示す。</p> <p>例：</p> <pre>java utils.MulticastTest -n name -a address       [-p portnumber] [-t timeout] [-s send]</pre>
	<p>構文の中で相互に排他的な選択肢を区切る。</p> <p>例：</p> <pre>java weblogic.deploy [list deploy undeploy update]       password {application} {source}</pre>
...	<p>コマンドラインで以下のいずれかを示す。</p> <ul style="list-style-type: none"> <li>■ 引数を複数回繰り返すことができる。</li> <li>■ 任意指定の引数が省略されている。</li> <li>■ パラメータや値などの情報を追加入力できる。</li> </ul>
.	<p>コード サンプルまたは構文で項目が省略されていることを示す。</p> <p>.</p> <p>.</p> <p>.</p>



---

# 1 WebLogic Server の配布

開発したアプリケーションと BEA のコアテクノロジーをまとめて 1 つの製品として配布することができます。これにより、顧客側でアプリケーションと J2EE アプリケーション サーバの両方を購入、インストール、および保守する必要がなくなります。WebLogic Server にはプラグアンドプレイ環境が提供されているため、1 つの製品として配布することは開発した製品との統合には理想的な方法といえます。

WebLogic Server を配布するには、**ISV ライセンス**という特別なライセンスを取得およびインストールする必要があります。アプリケーションの開発や WebLogic Server のコンフィグレーションには **ISV ライセンス**は必要ありません。WebLogic Server に対して **ISV ライセンス**をインストールすると、サーバが変更され、ファイルが密接にリンクされます。配布キットには、これらの変更された WebLogic Server ファイルを含める必要があります。

配布可能な WebLogic Server を設定するには、以下の作業を行います。

- WebLogic Server の新リリースを配布するための **ISV ライセンス**のアップグレード
- BEA Partner Program への登録
- パートナ開発キットのインストール
- **ISV ライセンス**のインストール
- ファイルの配布

# WebLogic Server の新リリースを配布するための ISV ライセンスのアップグレード

旧バージョンの WebLogic Server が配布されている場合、次の手順を実行して既存の ISV ライセンスをアップグレードします。

1. 旧ライセンスが格納されている `isv.jar` ファイルを添付して、新しい ISV ライセンスのリクエストを `licensing@bea.com` 宛に電子メールで送信します。WebLogic Server 5.1 からアップグレードする場合は、旧ライセンスファイルを添付してください。
2. WebLogic Server 7.0 パートナ開発キットをインストールします。
3. 新しい `isv.jar` ファイルを BEA から受け取ったら、1-3 ページの「ISV ライセンスのインストール」の説明に従って新しい ISV ライセンスをインストールします。
4. インストーラを更新し、新しい WebLogic Server ファイルを収めます。

## BEA Partner Program への登録

BEA Systems, Inc. では Partner Program を通じて、独立系ソフトウェアベンダ (ISV) およびアプリケーション ソフトウェアプロバイダ (ASP) と協力関係を結んでいます。BEA Partner Program への登録を済ませていない場合は、

1. <http://edocs.beasys.co.jp/e-docs/wls/certifications/certifications/index.html> の「WebLogic Platform サポート対象のコンフィグレーション」を参照して、対象のプラットフォームが WebLogic Server で使用できることを確認します。
2. BEA Partner Program Web サイトにアクセスし、プログラムについて理解した上で登録します。このページには、<http://www.beasys.co.jp/BeaPortal/partners.jsp> でアクセスできます。

# パートナ開発キットのインストール

プログラムに登録すると、BEA の主要製品がすべて収録された CD コレクションが BEA より発送されます。パートナ開発キットが到着したら、CD からソフトウェアをインストールします。WebLogic Server のインストールについては、BEA ドキュメント Web サイト (<http://edocs.beasys.co.jp/e-docs/index.html>) にある『インストール ガイド』を参照してください。

**警告：** 配布可能な WebLogic Server のホストとして使用するコンピュータ上に BEA 製品が既にインストールされている場合は、パートナ開発キットをインストールする前に、**その時点の BEA\_HOME\license.bea ファイルをバックアップします**。BEA ホーム ディレクトリと license.bea ファイルの詳細については、『インストール ガイド』の「BEA ホーム ディレクトリ」を参照してください。

CD が到着する前に、BEA Download Center (<http://commerce.beasys.com>) から BEA のソフトウェアをダウンロードすることもできます。有効な WebSUPPORT アカウントをお持ちの場合、WebSUPPORT ログイン パスワードを使用してソフトウェアをダウンロードできます。

## ISV ライセンスのインストール

Partner Program への登録が完了すると、カスタマイズ済みの ISV ライセンスが、電子メールの添付ファイル (isv.jar) として BEA より送付されます。この節では、**WebLogic Server 7.0 専用**の ISV ライセンス ファイルのインストール方法について説明します。WebLogic Server のそれ以外のバージョン用の ISV ライセンスをインストールする場合は、そのソフトウェア バージョンに対応したインストール手順を参照してください。

ISV ライセンスのインストールは、3 つの主要な手順を経て行います。

- 手順 1: ISV ライセンスをインストールする準備
- 手順 2: ライセンス データの抽出と WebLogic Server ファイルへのリンク

- 手順 3: WebLogic Server ライセンスの更新

# 手順 1: ISV ライセンスをインストールする準備

ISV ライセンス ファイルをインストールする前に、次の作業を行います。

1. まだインストールしていない場合、1-3 ページの「パートナー開発キットのインストール」の説明に従って、**WebLogic Server 7.0** をインストールします。  
**BEA** インストーラが使用する **BEA** ホーム ディレクトリの場所に注意してください。**BEA** ホーム ディレクトリには、このプロセスの後の手順で更新される **license.bea** ファイルが置かれます。**BEA** ホーム ディレクトリと **license.bea** ファイルの詳細については、『インストールガイド』の「**BEA** ホーム ディレクトリ」を参照してください。
2. 電子メールで送付された **isv.jar** ファイルを、インストーラで使用する **BEA** ホーム ディレクトリにコピーします。
3. コマンド シェルを起動し、**BEA\_HOME** ディレクトリに移動します。**BEA\_HOME** は **BEA** ホーム ディレクトリの名前です。
4. 次のスクリプトのいずれかを実行して、環境変数を設定します。
  - **WL\_HOME**\server\bin\setWLSEnv.cmd (Windows システムの場合)
  - **WL\_HOME**/server/bin/setWLSEnv.sh (UNIX システムの場合)

**WL\_HOME** は **WebLogic Server** のインストール ディレクトリです。デフォルトでは、このディレクトリは **BEA** ホーム ディレクトリのすぐ下にあります。たとえば、UNIX の **bash** シェルでは、次のコマンドを入力します。

```
[me@workstation]$ . WL_HOME/server/bin/setWLSEnv.sh
```
5. 次のコマンドのいずれかを入力して、コンピュータの **CLASSPATH** 環境変数に **isv.jar** へのパスを追加します。
  - **set CLASSPATH=.\isv.jar;%CLASSPATH%** (Windows システムの場合)
  - **export CLASSPATH=./isv.jar:\$CLASSPATH** (UNIX システムの場合)

これで、**ISV** ライセンス データを抽出して **WebLogic Server** ファイルにリンクする準備ができました。

## 手順 2: ライセンス データの抽出と WebLogic Server ファイルへのリンク

ISV ライセンス データを抽出して WebLogic Server ファイルにリンクするには、BEA\_HOME ディレクトリに移動して次のコマンドのいずれかを入力します。

- `java -Xmx128m -Dbea.home=BEA_HOME -Dbea.jar=WL_HOME\server\lib\weblogic.jar install` (Windows システムの場合)
- `java -Xmx128m -Dbea.home=BEA_HOME -Dbea.jar=WL_HOME/server/lib/weblogic.jar install` (UNIX システムの場合)

BEA\_HOME は BEA ホーム ディレクトリの絶対パス名であり、WL\_HOME は WebLogic Server をインストールしたディレクトリの絶対パス名です。

**警告：** いったん開始した後でこのプロセスを中断しないでください。

コマンドを実行すると、ISV ライセンス データを格納した、BEA\_HOME\license\_isv.bea という名前のファイルが生成されます。また、WL\_HOME ディレクトリ内部のファイルが ISV ライセンスにリンクされます。これにより、WL\_HOME ディレクトリ内の指定されたファイルだけが、license\_isv.bea に抽出された ISV ライセンス データを使用できるようになります。

**注意：** 一部のプラットフォームと JDK では、「メモリ不足エラー」が発生する場合があります。このエラーに対処するには、(最大ヒープ サイズを MB 単位で設定する) -Xmx 引数の値を大きくして、コマンドをもう一度実行します。たとえば、デフォルトのヒープ サイズを 150MB に増やすには、-Xmx150m と指定します。

ISV ライセンスのインストールプロセスを完了するには、license\_isv.bea 内のデータを使用して WebLogic Server ライセンスを更新する必要があります。

## 手順 3: WebLogic Server ライセンスの更新

新しく生成された `license_isv.bea` ファイルを使用して `license.bea` ファイルを更新するには、`BEA_HOME` ディレクトリに移動して次のいずれかのコマンドを入力します。

- `UpdateLicense license_isv.bea` (Windows システムの場合)
- `sh UpdateLicense.sh license_isv.bea` (UNIX システムの場合)

`UpdateLicense` コマンドを実行すると、`license_isv.bea` ファイルが `license.bea` ファイルとマージされます。`UpdateLicense` を実行した後は、`license_isv.bea` ファイルを削除してもかまいません。

## 次の手順 : アプリケーションと WebLogic Server のコンフィグレーション

ISV ライセンスのインストールが終了したら、ISV ライセンスとリンクされた WebLogic Server を起動し、アプリケーションをデプロイし、サーバコンポーネントをコンフィグレーションします。詳細については、<http://edocs.beasys.co.jp/e-docs/index.html> にある以下のドキュメントを参照してください。

- 『管理者ガイド』の「WebLogic Server の起動と停止」
- Administration Console オンライン ヘルプ
- 『WebLogic JMX サービス プログラマーズ ガイド』。WebLogic Server 管理システムを使用した作業の詳細な情報とコード サンプルが記載されています。
- このマニュアルの残りの節。ISV に固有の開発上のヒントが記載されています。

## ファイルの配布

アプリケーションとともに **WebLogic Server** を配布する準備ができれば、1-6 ページの「手順 3: **WebLogic Server** ライセンスの更新」で指定した **BEA** ライセンスファイル (`BEA_HOME\license.bea`) と `WL_HOME\lib\weblogic.jar` ファイルがインストーラに収められていることを確認する必要があります。

指定したファイルを両方ともインストールしないと、アプリケーションに組み込まれた **WebLogic Server** は起動しません。

この `license.bea` と `weblogic.jar` の同じ組み合わせは、ライセンスにリンクされるすべてのインストールで使用できます。**WebLogic Server** のサイレントインストールプロセスの使用方法については、「サイレントモード インストールによる **WebLogic Server** のインストール」を参照してください。



---

## 2 WebLogic Server コンフィグレーション ファイルのカスタマイズ

WebLogic Server では、セキュリティ資格、デプロイ可能なリソースおよびアプリケーションのリストなどのコンフィグレーション情報は一連のコンフィグレーション ファイルに格納されます。

以下の節では、パートナーおよび ISV が、開発したアプリケーションとともに配布する WebLogic Server コンフィグレーション ファイルに対して行う一般的な変更事項について説明します。

- config.xml ファイルのカスタマイズ
- コンフィグレーションの例
- 互換性セキュリティに対するファイルのカスタマイズ

### config.xml ファイルのカスタマイズ

config.xml ファイルは、管理ドメインにあるすべての WebLogic Server のコンフィグレーション設定の大部分を定義します。たとえば、config.xml ファイルは、特定のドメインのすべての詳細情報を制御します。このような情報としては、サーバとクラスタの名前、数、およびコンフィグレーション、デプロイ可能なリソースおよびアプリケーションのリスト、デプロイ可能なリソースおよびアプリケーションのサーバおよびクラスタへのマッピングなどがあります。

通常は、Administration Console や weblogic.Admin ユーティリティなどの WebLogic Server ツールを使用して config.xml ファイルを編集することをお勧めします。ただし、パートナーがインストールをカスタマイズするために、このファイルを直接編集することが必要な場合もあります。

以下の節では、パートナーがインストール用に変更する可能性のある `config.xml` ファイルの要素について説明します。

- アプリケーション リソースのプリコンフィグレーション
- アプリケーション コンポーネントのデプロイメント

`config.xml` ファイルまたは管理ドメインの役割に精通していない場合は、以下のドキュメントを参照してください。

- 『管理者ガイド』の「WebLogic システム管理の概要」
- 『WebLogic Server クラスタ ユーザーズ ガイド』の「クラスタのコンフィグレーションとアプリケーションのデプロイメント」

`config.xml` を直接編集することに慣れていない場合は、『コンフィグレーション リファレンス』を参照してください。このリファレンス マニュアルには、`config.xml` を編集する際の規則や、このファイルの文書型定義 (DTD) の説明が記載されています。

# アプリケーション リソースのプリコンフィグレーション

パートナー アプリケーションは通常いくつかの WebLogic Server リソースに依存し、それらの各リソースは `config.xml` ファイルで定義されます。

リソース	<code>config.xml</code> 要素	注意
ドメイン	Domain	凝集された単位として動作するために、アプリケーションのコンポーネントをホストするすべての WebLogic Server は、単一の WebLogic Server 管理ドメイン内に置かれる必要がある。

リソース	config.xml 要素	注意
サーバ名および 接続情報	Server	1 つまたは複数の <b>WebLogic Server</b> 名、IP アドレス、およびポート番号にアクセスするよう、パートナアプリケーションをコンフィグレーションできる。アプリケーションで必要であれば、 <b>WebLogic Server</b> ドメインをハード コード化して、特定のサーバ名および接続ポートを使用できる。  コンフィグレーションをインストールする前に、IP アドレスをアプリケーション インストーラによって動的にコンフィグレーションし、config.xml に埋め込むことができる。
JDBC データソース	JDBCConnectionPool JDBCDataSource JDBCMultiPool JDBCTxDataSource	<b>WebLogic Server</b> をインストールするパートナアプリケーションは、アプリケーション データを管理するために <b>RDBMS</b> や他のデータストアもインストールする場合がよくある。開発した製品のインストーラがアプリケーションと一緒にデータストアをインストールする場合、インストールされた <b>WebLogic Server</b> をプリコンフィグレーションして、そのデータストアのデフォルトのデータソースおよび接続プールを設定できる。

## アプリケーション コンポーネントのデプロイメント

パートナアプリケーションは、config.xml に必要な要素を追加することによってインストールすることもできます。ただし、プリコンフィグレーションした **WebLogic Server** にアプリケーションをインストールする場合は、config.xml 設定とアプリケーション コンポーネント ファイル (.war、.jar、.html など) のインストール場所とを調整する必要があります。

次の表は、**WebLogic Server** 内にアプリケーション コンポーネントを事前にデプロイするために使用する要素についてまとめたものです。これらの要素がどのように実際のアプリケーション コンポーネント ファイルのインストール場所に対応するかを示す例については、2-4 ページの「コンフィグレーションの例」を参照してください。

コンポーネント	config.xml 要素	注意
起動クラス	StartupClass	WebLogic Server 起動クラスは、パートナーアプリケーションの他のコンポーネントに必要なリソースを初期化するために使用できる。
Web サーバ	WebServer	Web アプリケーションには、通常、ビジネス ロジックに加え、静的な .html コンテンツなどの標準 Web リソースが必要である。config.xml ファイルを使用して、アプリケーションの静的ファイルのデフォルトの場所をコンフィグレーションする。
Web アプリケーション	Application	EAR ファイルおよび WAR ファイルは、アプリケーションディレクトリまたは WebLogic Server ディレクトリの任意の場所に格納できる。起動時にアプリケーションをデプロイするには、config.xml 内から最終のインストール場所を参照する。

## コンフィグレーションの例

WebLogic Server Pet Store は、Sun Microsystems の Java Pet Store 1.3 デモに基づいています。Pet Store には、J2EE プラットフォームのさまざまな側面を示す 4 つのエンタープライズ アプリケーションと 1 つの Web アプリケーションが含まれます。

WebLogic Server Pet Store では、エンタープライズ アプリケーションをサポートするためにインストールをプリコンフィグレーションする方法の簡単な例が用意されています。WebLogic Server Pet Store の起動については、WebLogic Server ドキュメント Web サイトにある「サンプルとチュートリアル」を参照してください。

この節では、WebLogic Server Pet Store をコンフィグレーションする config.xml ファイルの主要な側面について説明します。

## ドメインのコンフィグレーション

config.xml ファイルの親要素である、<Domain> には、petstore ドメインのコンフィグレーション情報が記述されます。アプリケーションのすべてのサーバ、リソース、およびコンポーネントは、このドメイン内に定義されます。

```
<Domain
  Name="petstore"
  >
```

## 基本的なサーバの設定

WebLogic Server Pet Store は、petstoreServer という 1 つのサーバを使用します。このサーバの接続プロパティは、次のように Server 要素属性でプリコンフィグレーションされています。

```
<Server
  JavaCompiler="C:\bea\jdk131_02/bin/javac"
  ListenPort="7001"
  Name="petstoreServer"
  ServerVersion="7.0.0.0"
  IIOPEnabled="false"
  >
```

サーバ設定の残りの部分では、サーバのデフォルト Web サーバおよび SSL コンフィグレーションがコンフィグレーションされています。

# JDBC 要件

複数の JDBC 接続プールの使用を示すために、WebLogic Server Pet Store では 3 つの JDBC 接続プールが定義されています。また、Pet Store 内の各アプリケーションに 1 つずつ、全部で 4 つのデータソースが定義されています。JDBC 接続プールおよびデータソースはすべて、デフォルトの Pet Store サーバ petstoreServer にデプロイされます。

```
<JDBCConnectionPool
  CapacityIncrement="1"
  DriverName="com.pointbase.jdbc.jdbcUniversalDriver"
  InitialCapacity="1"
  MaxCapacity="10"
  Name="petstorePool"
  Password="petstore"
  Properties="user=petstore"
  RefreshMinutes="0"
  ShrinkPeriodMinutes="15"
  ShrinkingEnabled="true"
  Targets="petstoreServer"
  TestConnectionsOnRelease="false"
  TestConnectionsOnReserve="false"
  URL="jdbc:pointbase:server://localhost/demo"
/>
```

```
<JDBCConnectionPool
  CapacityIncrement="1"
  DriverName="com.pointbase.jdbc.jdbcUniversalDriver"
  InitialCapacity="1"
  MaxCapacity="10"
  Name="petstoreopcPool"
  Password="petstoreopc"
  Properties="user=petstoreopc"
  RefreshMinutes="0"
  ShrinkPeriodMinutes="15"
  ShrinkingEnabled="true"
  Targets="petstoreServer"
  TestConnectionsOnRelease="false"
  TestConnectionsOnReserve="false"
  URL="jdbc:pointbase:server://localhost/demo"
/>
```

```

<JDBCConnectionPool
  CapacityIncrement="1"
  DriverName="com.pointbase.jdbc.jdbcUniversalDriver"
  InitialCapacity="1"
  MaxCapacity="10"
  Name="petstoresupplierPool"
  Password="petstoresupplier"
  Properties="user=petstoresupplier"
  RefreshMinutes="0"
  ShrinkPeriodMinutes="15"
  ShrinkingEnabled="true"
  Targets="petstoreServer"
  TestConnectionsOnRelease="false"
  TestConnectionsOnReserve="false"
  URL="jdbc:pointbase:server://localhost/demo"
/>

```

```

<JDBCTxDataSource
  EnableTwoPhaseCommit="true"
  JNDIName="datasource-petstorePool"
  Name="PetstoreDataSource"
  PoolName="petstorePool"
  Targets="petstoreServer"
/>

```

```

<JDBCTxDataSource
  EnableTwoPhaseCommit="true"
  JNDIName="datasource-petstoreopcPool"
  Name="PetstoreOPCDataSource"
  PoolName="petstoreopcPool"
  Targets="petstoreServer"
/>

```

```

<JDBCTxDataSource
  EnableTwoPhaseCommit="true"
  JNDIName="datasource-petstoresupplierPool"
  Name="PetstoreSupplierDataSource"
  PoolName="petstoresupplierPool"
  Targets="petstoreServer"
/>

```

```

<JDBCTxDataSource
  EnableTwoPhaseCommit="true"
  JNDIName="jdbc/CatalogDataSource"

```

```
Name="CatalogDataSource"  
PoolName="petstorePool"  
Targets="petstoreServer"  
/>
```

# アプリケーション コンポーネント

Pet Store アプリケーションでは、`tour.war`、`petstore.ear`、`opc.ear`、`supplier.ear`、および `petstoreadmin.ear` という 5 つのアプリケーション コンポーネント ファイルがインストールされます。これらのコンポーネントはそれぞれ **WebLogic Server** サブディレクトリにインストールされ、`petstoreServer` にデプロイされます。たとえば **Windows** では、`tour.war` は次の要素を使用してデプロイされます。

```
<Application  
  Deployed="true"  
  Name="tour"  
  Path="C:/bea/weblogic700/samples/server/stage/petstore">  
  
  <WebAppComponent  
    Name="tour"  
    Targets="petstoreServer"  
    URI="tour.war"  
  />  
</Application>
```

アプリケーション コンポーネント パスの `c:\bea\wlserver700` の部分は **WebLogic Server** のインストール中に決定され、パスの残りの部分はハードコード化されていることに注意してください。必要であれば、アプリケーション インストーラによって、同じような方法で **WebLogic Server** に関連しないサブディレクトリにアプリケーション コンポーネントをインストールできます。

## 起動クラス

Windows システムでは、petstoreServer の起動後、Pet Store アプリケーションは起動クラスを使用してシステムの Web ブラウザを起動します。この起動クラスの定義は、ほとんどの場合、インストールされた config.xml ファイルにハードコード化され、ポート番号だけが WebLogic Server インストールプログラムによって動的に設定されます。

```
<StartupClass </Application>
  Arguments="port=7001"
  ClassName="com.bea.estore.startup.StartBrowser"
  FailureIsFatal="false"
  Name="StartBrowser"
  Targets="petstoreServer"
  Notes="On Windows, this class automatically starts a browser after the server
has finished booting."
  />
```

## 互換性セキュリティに対するファイルのカスタマイズ

**互換性セキュリティ**とは、WebLogic Server 6.x のセキュリティ コンフィグレーションを WebLogic Server 7.0 で実行する機能のことです。WebLogic Server を互換性セキュリティを使用して実行する場合は、配布キットに以下のものを含める必要があります。

- fileRealm.properties ファイル。デフォルト WebLogic Server セキュリティ レalm の ACL、グループ、およびセキュリティ プリンシパルを定義します。
- config.xml の以下の最小限の要素

```
<Domain Name="mydomain">
  <Security Name="mydomain" Realm="mysecurity" />
  <Realm Name="mysecurity" FileRealm="myrealm" />
  <FileRealm Name="myrealm" />
  <Server ListenPort="7001" Name="myserver">
  </Server>
```

</Domain>

アプリケーションをサードパーティのセキュリティレルム（たとえば Windows NT セキュリティレルムを使用したシングルサインオン）と統合する必要がある場合は、キャッシングレルムもコンフィグレーションする必要があります。

WebLogic Server セキュリティの詳細については、以下のドキュメントを参照してください。

- 『WebLogic Security の管理』の「互換性セキュリティの使い方」
- WebLogic Server ドキュメント Web サイトにある「セキュリティ」のページ
- 『コンフィグレーション リファレンス』。このリファレンス マニュアルには、`config.xml` を編集する際の規則や、このファイルの DTD の説明が記載されています。

---

## 3 JDBC プロファイリング MBean の使用

WebLogic Server 管理システムでは、Java Management Extension (JMX) と Managed Bean (MBean) を使用して、サーバをコンフィグレーションします。『WebLogic JMX サービスプログラマーズ ガイド』には、WebLogic Server MBean を使用するための詳細な情報、およびコード サンプルが記載されています。

BEA では、SQL 文、prepared statement、および JDBC 接続リークのマトリックを格納および解析するために使用できる複数の JDBC MBean を提供しています。以降の節では、JDBC プロファイリングを有効にする方法および使用方法について説明します。詳細については、以下の WebLogic Server MBean および関連するクラスの Javadoc を参照してください。

- JDBCConnectionPoolMBean
- JDBCConnectionPoolRuntimeMBean
- JDBCStatementProfile
- JDBCConnectionLeakProfile

### JDBC プロファイリングの有効化

SQL 文または接続リーク プロファイルを解析する前に、調査する接続プールのプロファイリングを有効にする必要があります。プロファイリングが有効になると、接続プールによってメトリックは、後で解析するために外部リポジトリに格納されます。

### 3 JDBC プロファイリング MBean の使用

---

アプリケーションは、`JDBCConnectionPoolMBean` を使用して JDBC プロファイリング オプションを有効および無効にします。標準接続プールプロパティの `get/set` メソッドを提供するのに加えて、`JDBCConnectionPoolMBean` はプロファイリングを有効および無効にする次のメソッドを提供します。

- `setConnLeakProfilingEnabled()` は、JDBC 接続リークのプロファイリングを有効または無効にします。接続リークは、接続プールからチェックアウトされても、`close()` メソッドによって返されない接続を示します。リークされた接続は、のちの接続要求の実行に使用できないため、接続リークプロファイルを解析することは重要です。
- `setSqlStmtProfilingEnabled()` は、SQL 文のプロファイリングを有効または無効にします。このタイプのプロファイリングが有効になると、接続プールによって SQL 文および文の実行時間と他のメトリックが格納されます。SQL 文のプロファイルを解析することによって、アプリケーションで最も時間のかかるクエリを調べることができます。
- `setSqlStmtParamLoggingEnabled()` は、prepared statement および callable statement のバインド パラメータのプロファイリングを有効または無効にします。文パラメータは非常に大きくなる可能性があるため、プロファイルに格納されるパラメータのサイズを制限するために、`setSqlStmtMaxParamLength()` を任意で使用できます。

WebLogic Server での MBean の取得については、「WebLogic Server MBean へのアクセス」を参照してください。次の例は、`JDBCConnectionPoolMBean` を取得し、すべてのプロファイリング オプションを有効にするアプリケーションを示します。この例では、各文パラメータについて最大 20 文字を格納します。

```
// 管理サーバの MBeanHome を取得する
...
JDBCConnectionPoolMBean mbean =
    (JDBCConnectionPoolMBean)home.getConfigurationMBean(poolName,
        "JDBCConnectionPoolConfig");
mbean.setConnLeakProfilingEnabled(true);
mbean.setSqlStmtParamLoggingEnabled(true);
mbean.setSqlStmtMaxParamLength(maxLen);
...
```

---

## JDBC プロファイルへのアクセス

必要なプロファイリング オプションを有効にしたら、`JDBCStatementProfile` クラスおよび `JDBCConnectionLeakProfile` クラスを使用して、格納されたメトリックを解析できます。これらのプロファイル クラスは共に、`JDBCConnectionPoolRuntimeMBean` を使用して簡単に取得できます。

`JDBCStatementProfile` は、接続プールの **SQL** 文と関連メトリック（およびオプションとして、バインド パラメータ）を格納します。

`JDBCConnectionLeakProfile` は、リークされた接続のスタック トレースを格納します。

すべてのプロファイルを一度に取得すると、非常に多くのリソースが使用される可能性があります。この理由により、通常、アプリケーションは指定した時間にプロファイルのサブセットのみを取得します。これを行うには、最初にストレージ内のプロファイルの合計数を調べてから、より小さいサブセットでプロファイルを取得します。

次の例は、プロファイルの数をより小さい単位に分ける簡単な方法を示します。

```
// 接続プールをホストするサーバの MBeanHome を取得する
. . .

// 「testPool」接続プールの JDBCRuntimeMbean を取得する

String poolName = "testPool";

JDBCConnectionPoolRuntimeMBean mbean =
    (JDBCConnectionPoolRuntimeMBean)home.getRuntimeMBean
        (poolName, "JDBCConnectionPoolRuntime");

JDBCConnectionLeakProfile[] profiles = null;

// 使用可能な prepared statement のキャッシュ プロファイルの合計数を取得する
int profileCount = mbean.getConnectionLeakProfileCount();

// プロファイルの profilesPerStep の数を要求する
int profilesPerStep = 10;

// プロファイル数 profileIndex で開始する
int profileIndex = 0;
```

### 3 JDBC プロファイリング MBean の使用

---

```
boolean done = (profileCount > 0);
while (!done) {
    // プロファイルを取得する
    profiles = mbean.getConnectionLeakProfiles(profileIndex,
        profilesPerStep);

    // 取得されたプロファイルを調査する
    for (int index = 0; index < profiles.length; index++) {
        // プール名を取得する
        String poolName = profiles[index].getPoolName();

        // スタック トレースを取得する
        String stackTrace = profiles[index].getStackTrace();
    }

    profileIndex = profileIndex + profilesPerStep - 1;

    // 取得したプロファイルの数が
    // 要求した数よりも少ない場合は終了
    done = (profiles.length < profilesPerStep);
}
```