入門ガイド

Sun ONE Application Server 7

Version 7

817-0599-10
2002 年 9 月

目 次

はじめに

この章では、『Sun™ Open Net Environment (Sun ONE) Application Server 7 入門ガイド』の内容について説明します。

この章には次の節があります。

- 本書について
- マニュアルの構成
- マニュアルの使用法
- マニュアルの表記規則
- 製品サポート

本書について

この『入門ガイド』は、Sun ONE Application Server を初めて使うユーザーを対象としています。この製品の開発機能や管理機能を使いこなすための基本的な情報を提供します。このマニュアルで説明している練習手順を一通りこなすには、通常は 1 ～ 2 時間で十分です。アプリケーションサーバーの使用や開発の経験は特に必要ありません。

このマニュアルでは、最初にアプリケーションサーバーについて説明してから、一般的な環境設定手順を説明します。次に、製品をまだインストールしていないユーザーのために評価用バージョンの基本的なインストール手順について説明します。さらに、アプリケーションサーバーインスタンスの起動と停止、アプリケーションの構築、配備、変更など、基本的な管理作業に共通する操作の手順を紹介します。また、サーバーポート番号の変更、仮想サーバーの作成、アプリケーションサーバーインスタンスの作成、管理ドメインの作成など、基本操作以外の共通管理作業についても説明します。

このマニュアルの最新版については、http://docs.sun.com/ に掲載されている Sun ONE Application Server 7 のマニュアルセットを参照してください。

マニュアルの構成

このマニュアルは次の章から構成されています。

- 第 1 章「Sun ONE Application Server 7 について」では、アプリケーションサーバーの概要について説明します。
- 第 2 章「Sun ONE Application Server の評価用 (Evaluation) バージョンのインストール」では、Windows、UNIX の両方の環境に製品をインストールする手順について説明します。この手順を実行すれば、Sun ONE Application Server を評価用に簡単にインストールできます。
- 第 3 章「環境の設定」では、PATH 環境変数にアプリケーションサーバーの `bin` ディレクトリを追加する手順について説明します。
- 第 4 章「アプリケーションサーバーの起動と停止」では、アプリケーションサーバー環境を起動し、正常な稼動を確認する手順について説明します。
- 第 5 章「サンプルアプリケーションの操作」では、簡単なサンプルアプリケーションを作成して配備する手順について説明します。アプリケーションが正しく動作することを確認したら、アプリケーションに変更を加え、配備し直します。
- 第 6 章「データベースとの接続の設定」では、サンプルアプリケーションを配備、実行する前に必要な JDBC 関連の設定をアプリケーションサーバー環境に設定する手順について説明します。
- 第 7 章「サンプルアプリケーションの配備と実行」では、サンプルアプリケーションの `build.xml` ファイルに対して Ant 機能を実行する手順について説明します。これにより、アプリケーションのソースコードのコンパイル、および EAR ファイルの再アセンブルを迅速に行えます。次に、新たに作成した EAR ファイルをアプリケーションサーバーに配備し、実行します。
- 第 8 章「サンプルアプリケーションの変更」では、アプリケーションサーバーの動的な配備機能について説明し、前章で配備したサンプルアプリケーションを実際に使用する方法を説明します。
- 第 9 章「サーバーのその他の機能」では、共通の管理作業の手順を説明します。
- 第 10 章「要約と参照情報」では、このマニュアルで説明した作業を振り返り、次に参照すべき情報について説明します。

マニュアルの使用法

このマニュアルは、PDF 形式または HTML 形式でも入手できます。次のサイトを参照してください。

`http://docs.sun.com`

次の表は、Sun ONE Application Server のマニュアルに記述されているタスクと概念を示しています。左側の列にタスクと概念、右側の列に参照するマニュアルを示します。

Sun ONE Application Server マニュアルの概要

情報の内容	参照するマニュアル
ソフトウェアおよびマニュアルの最新情報	リリースノート
サポート対象のプラットフォームと環境	プラットフォーム
アプリケーションサーバーの紹介。新機能、評価用バージョンのインストール、アーキテクチャの概要など	入門ガイド
Sun ONE Application Server とそのコンポーネント (サンプルアプリケーション、管理インタフェース、Sun ONE Message Queue など) のインストール	インストールガイド
Sun ONE Application Server 7 の Java オープンスタンダードモデルに準拠した J2EE アプリケーションの作成方法と実装方法。アプリケーション設計、開発ツール、セキュリティ、アセンブリ、配備、デバッグ、ライフサイクルモジュールの作成に関する情報など	開発者ガイド
Sun ONE Application Server 7 の Web アプリケーション向け Java オープンスタンダードモデルに準拠した J2EE アプリケーションの作成方法と実装方法。Web アプリケーションプログラミングの概念とタスクの説明、サンプルコード、実装のヒント、関連資料の紹介など	Web アプリケーション開発者ガイド
Sun ONE Application Server 7 の エンタープライズ Beans 向け Java オープンスタンダードモデルに準拠した J2EE アプリケーションの作成方法と実装方法。EJB プログラミングの概念とタスクの説明、サンプルコード、実装のヒント、関連資料の紹介など	Enterprise JavaBeans 開発者ガイド
Web サービス、RMI-IIOP、Sun ONE Application Server 7 上の J2EE アプリケーションにアクセスするその他のクライアントの作成方法	Developer's Guide to Clients
JDBC、JNDI、JTS、JMS、JavaMial、リソース、コネクタなどの J2EE 機能	Developer's Guide to J2EE Features and Services

Sun ONE Application Server マニュアルの概要 (続き)

情報の内容	参照するマニュアル
カスタム NSAPI プラグインの作成方法	NSAPI Developer's Guide
次の管理タスクの実行 • 管理インタフェースとコマンド行インタフェースの使用 • サーバーの作業環境の設定 • 管理ドメインの使用 • サーバーインスタンスの使用 • サーバーの稼動状況の監視およびログ記録 • Web サーバープラグインの設定 • Java Messaging Service の設定 • J2EE 機能の使用 • CORBA ベースのクライアント機能の設定 • データベース接続性の設定 • トランザクション管理の設定 • Web コンテナの設定 • アプリケーションの配備 • 仮想サーバーの管理	管理者ガイド
サーバー設定ファイルの編集	管理者用設定ファイルリファレンス
Sun ONE Application Server 7 運用環境のセキュリティの設定および管理。セキュリティに関する一般情報、証明書、SSL/TLS による暗号化など。Web コアベースのセキュリティについても解説	セキュリティ管理者ガイド
Sun ONE Application Server 7 で利用する J2EE CA コネクタのサービスプロバイダ実装の設定と管理。管理ツールである DTD に関する情報、および XML ファイルのサンプルを提供	J2EE CA Service Provider Implementation Administrator's Guide
Netscape Application Server バージョン 2.1 から新しい Sun ONE Application Server 7 プログラミングモデルへのアプリケーションの移行。Sun ONE Application Server に付属するオンラインバンクアプリケーションの移行サンプルなど	サーバーアプリケーションの移行および再配備

Sun ONE Application Server マニュアルの概要 (続き)

情報の内容	参照するマニュアル
Sun ONE Message Queue の使用	http://docs.sun.com に掲載されている Sun ONE Message Queue のマニュアルを参照

マニュアルの表記規則

ここでは、このマニュアル全体に適用される表記規則について説明します。

- 一般的な表記規則
- ディレクトリ名の表記規則

一般的な表記規則

このマニュアルに適用される一般的な表記規則は次のとおりです。

- **ファイルとディレクトリのパス**は、UNIX の形式で表記します (ディレクトリ名を「/」記号で区切って表記)。Windows バージョンでは、ディレクトリパスについては UNIX と同じですが、ディレクトリの区切り記号には「/」記号ではなく「¥」記号を使用します。
- URL は次のように表記されます。

  ```
  http://<server>.<domain>/<path>/<file>.html
  ```

  これらの URL で、`server` はアプリケーションを実行するサーバー名で、`domain` はユーザーのインターネットドメイン名、`path` はサーバー上のディレクトリの構造、`file` は個別のファイル名を示します。URL の斜体文字の部分は可変部分です。
- このマニュアルでは、**フォント**について次の規則を採用しています。
  - モノスペースフォントは、コード例、コードリスト、API および言語要素 (関数名、クラス名など)、ファイル名、パス名、ディレクトリ名、および HTML タグに使用します。
  - コード変数は、<monospace> のように表記されます。
  - 斜体文字は、変数および可変部分、およびリテラルに使われる文字に使います。
  - **太字**は、段落の先頭文字またはリテラルに使われる文字の強調に使います。

- このマニュアルでは、ほとんどのプラットフォームの**インストールルートディレクトリ**は、<install_dir> として表記されます。例外については、12 ページの「ディレクトリ名の表記規則」を参照してください。

  ほとんどのプラットフォームのデフォルトの <install_dir> は次のとおりです。

  - Solaris 8 のパッケージベースでない評価 (Evaluation) バージョン

    < ユーザーのホームディレクトリ >/sun/appserver7

  - Solaris にバンドルされていない、評価用以外のバージョン

    /opt/SUNWappserver7

  - Windows のすべてのインストール

    C:¥Sun¥AppServer7

  上記プラットフォームでは、<default_config_dir> と <install_config_dir> は <install_dir> と同じ意味です。これ以外の説明と例外については、12 ページの「ディレクトリ名の表記規則」を参照してください。

- このマニュアルでは、**インスタンスルートディレクトリ**は <instance_dir> と表記されます。これは、実際には次のディレクトリを示しています。

  <default_config_dir>/domains/<domain/instance>

- このマニュアルを通じて、特に明記のない限り、**UNIX 固有の表記**は、Linux オペレーティングシステムにも適用されます。

ディレクトリ名の表記規則

Solaris 8 および 9 のパッケージに含まれる製品のインストール、および Solaris 9 との一括インストールでは、アプリケーションサーバーのファイルは通常は複数のルートディレクトリにまたがって保存されます。ここでは、これらのディレクトリについて説明します。

- **Solaris 9 にバンドルのインストール**では、デフォルトのインストールディレクトリは次のように表記されます。

  - <install_dir> は /usr/appserver/ を示します。このディレクトリにはインストールイメージの静的な要素が保存されます。ユーティリティ、実行可能ファイル、およびアプリケーションサーバーを構成するライブラリは、すべてここに保存されます。

  - <default_config_dir> は /var/appserver/domains を示します。このディレクトリは、作成したドメインのデフォルトの保存場所です。

  - <install_config_dir> は /etc/appserver/ を示します。このディレクトリには、ライセンスやそのインストール用に設定した管理ドメインのマスターリストなど、インストール全体に適用される設定情報が保存されます。

- **Solaris 8 および Solaris 9 のパッケージベース**の評価用以外のアンバンドルのインストールでは、デフォルトのインストールディレクトリは次のように表記されます。
  - `<install_dir>` は `/opt/SUNWappserver7` を示します。このディレクトリにはインストールイメージの静的な要素が保存されます。ユーティリティ、実行可能ファイル、およびアプリケーションサーバーを構成するライブラリは、すべてここに保存されます。
  - `<default_config_dir>` は `/var/opt/SUNWappserver7/domains` を示します。このディレクトリは、作成したドメインのデフォルトの保存場所です。
  - `<install_config_dir>` は `/etc/opt/SUNWappserver7/` を示します。このディレクトリには、ライセンスやそのインストール用に設定した管理ドメインのマスターリストなど、インストール全体に適用される設定情報が保存されます。

製品サポート

ご使用のシステムに問題が発生した場合は、次のいずれかの方法でカスタマサポートにお問い合わせください。

- 次のオンラインサポート Web サイトをご利用ください。

  `http://www.sun.com/supportraining/`

- 保守契約を結んでいるお客様の場合は、専用ダイヤルをご利用ください。

事前に次の情報を準備してください。テクニカルサポートスタッフが問題解決のお手伝いする上で、この情報が役立ちます。

- 問題が発生した箇所や動作への影響など、問題の具体的な説明
- マシン機種、OS バージョン、および、問題の原因と思われるパッチやそのほかのソフトウェアなどの製品バージョン
- 問題を再現するための具体的な手順の説明
- エラーログやコアダンプ

第 1 章

Sun ONE Application Server 7 について

この章では次の項目について説明します。

- Sun ONE Application Server について
- 製品ラインの概要
- Sun ONE Application Server の新機能
- 製品の主要コンポーネント
- アーキテクチャの概要

Sun ONE Application Server について

Sun ONE Application Server は、電子商取引アプリケーションと多種多様なサーバー、クライアント、デバイス向け Web サービスを開発、配備、管理するための堅牢な J2EE プラットフォームを提供しています。主要な機能として、トランザクション管理、パフォーマンス、スケーラビリティ、セキュリティ、エンタープライズアプリケーションの統合などの機能があります。Sun ONE Application Server は、高度なスケーラビリティを備えたアプリケーションサーバー技術を実現したいという Sun の専門家たちの 6 年以上におよぶ開発努力の成果です。本製品により、JavaServer Pages (JSP™)、Java Servlet、Enterprise Java Beans (EJB™) テクノロジを利用した堅牢なアプリケーションを迅速に開発できます。本製品のテクノロジは、小規模な部門アプリケーションからエンタープライズ規模のミッションクリティカルなサービスまで、広い範囲のビジネス要件をカバーしています。

製品ラインの概要

Sun ONE Application Server 7 は J2EE 1.3 仕様に準拠したアプリケーションサーバーで、最新の Java Web サービス標準や、標準の HTTP サーバープログラミング機能にも対応しています。本稼働環境、開発環境の両方のさまざまなニーズに対応できるように、このアプリケーションサーバーには次の 3 種類が用意されています。

- Platform Edition
- Standard Edition
- Enterprise Edition

Platform Edition

Platform Edition は、Sun ONE Application Server 7 製品ラインの中核となります。本稼動製品は無料で、J2EE 1.3 仕様に準拠した高性能かつ小型の実行環境を提供します。この環境は、基本的な配備作業や、サードパーティ製のアプリケーションの埋め込みにも適しています。Web サービスにも対応しており、Sun ONE Web Server 製品および Sun ONE Message Queue 製品ですでに定評のあるテクノロジが搭載されています。

Platform Edition は、1 つのアプリケーションサーバーインスタンス、つまり Java プラットフォーム上の 1 つの仮想マシン (「Java 仮想マシン」または「JVM」) にしか配備できません。Platform Edition は多層配備のトポロジには対応していますが、Web サーバー層プロキシはロードバランスを行いません。Platform Edition で管理ユーティリティを使用できるのはローカルクライアントだけです。

2003 年初頭以降、Sun ONE Application Server 7 の Platform Edition は Solaris 9 に統合される予定です。

Standard Edition

このマニュアルでは、Standard Edition について主に扱います。Standard Edition は、Platform Edition に拡張されたリモート管理機能を加えた製品です。Standard Edition では、拡張された管理機能、リモート環境でのコマンド行の使用、および Web ベースの管理のすべてを利用できます。また、Web サーバー層プロキシによる Web アプリケーショントラフィックの分配機能も備えています。Standard Edition では、1 台のマシンに複数のアプリケーションサーバーインスタンス (JVM) を設定できます。

Enterprise Edition

Enterprise Edition では、最も要求の厳しい J2EE ベースアプリケーションにも適用できるように、可用性の向上、ロードバランス機能、およびクラスタリング機能によりコアアプリケーションサーバープラットフォームの機能をさらに高めています。Enterprise Edition では Standard Edition の管理機能も拡張され、複数マシンへの複数インスタンスの配備に対応しています。

クラスタリング機能により、アプリケーションサーバーインスタンスのクローンを簡単にグループとして設定し、クライアント要求をそのグループ間でロードバランスを調整できます。Enterprise Edition は、外部のロードバランサと、Web 層ベースのプロキシのロードバランスの両方に対応しています。Enterprise Edition には、HTTP セッション、ステートフルセッション Bean インスタンス、および Java Message Service (JMS) リソースフェイルオーバーが含まれます。Enterprise Edition の HA 持続性ストアには、データベースの高可用性を実現する「Always On」という特許を取得したテクノロジが採用されています。

製品の詳細については、Sun Microsystems の Web サイトで Sun ONE Application Server のページを参照してください。

Sun ONE Application Server の新機能

Sun ONE Application Server 7 のコアには、開発および運用作業の効率を良くするためのさまざまな新機能が搭載されています。

これらの新機能は、上記 3 つの Edition 製品に搭載されています。

- 開発機能
- 運用機能

開発機能

Sun ONE Application Server 7 では、次の開発機能を利用できます。

- Java 2 Enterprise Edition 1.3 との完全な互換性
- JavaServer Pages (JSP) 1.2 および Servlet 2.3 のサポート
- Enterprise JavaBeans (EJB) 2.0 テクノロジ
- メッセージ駆動型 Beans (MDB)
- Java 2 Platform、Standard Edition (J2SE™ プラットフォーム) 1.4

- 統合された Java Web サービス
- 最新の Sun ONE Studio の統合
- 動的な (「ホット」) 配備と再読み込み
- 大幅に拡張されたコンテナ管理の持続性 (CMP) に対応
- 設定が簡単な XML ベースのサーバー設定
- ライフサイクルリスナークラス (洗練された起動クラスと停止クラス)
- 統合された J2EE アプリケーション検証ユーティリティ
- 豊富なサンプルアプリケーション
- Ant のビルド機能の統合
- 依存性の少ない簡単なインストール (別の Web サーバーは不要)

運用機能

Sun ONE Application Server 7 では、次の運用機能を利用できます。

- 統合された高性能 HTTP サーバーと Web コンテナ
- 定評のある JMS プロバイダを統合
- Web アプリケーション用の仮想 HTTP サーバーに対応
- インストールイメージごとの複数の管理ドメイン (独立したアプリケーションサーバー設定。ベータ版では使用不可)
- Web ベースの管理
- すべての機能をリモート実行できるコマンド行インタフェースによるリモート監視
- Java Authentication and Authorization Service (JAAS) に基づくプラグインによる認証
- 改善されたロギング

製品の主要コンポーネント

Sun ONE Application Server 7 の Standard Edition には、次のコンポーネントが含まれています。

- アプリケーションサーバーコア
- Sun ONE Message Queue 3.0.1
- Sun ONE Studio 4, Enterprise Edition for Java (試用ライセンス)
- PointBase データベースサーバーおよび Type 4 JDBC ドライバ
- Java 2 Software Development Kit, Standard Edition 1.4_02

ダウンロード可能な評価用バージョンには、Sun ONE Studio IDE が含まれたバージョンと、含まれていないバージョンの 2 種類があります。

CD ROM で配布される Evaluation バージョンには、Sun ONE Studio IDE が含まれています。

上記すべてのコンポーネントは評価用バージョンに含まれ、製品の一部として同時にインストールされます。

アプリケーションサーバーコア

アプリケーションサーバーコアは、J2EE 1.3 に準拠したアプリケーションサーバーおよび HTTP サーバーで、そのまま配備可能です。

- 実行時制御、プロセス制御、スレッド管理などのコアランタイム
- Sun ONE Web Server による HTTP サーバー
- J2EE 1.3 Web コンテナと EJB コンテナ
- 持続性レイヤ
- すべて Java で記述された JTS ベースのトランザクションマネージャ
- Object Request Broker (ORB)
- SNMP エージェント (ORB の後に存在)
- HTTP プロキシプラグイン (SNMP エージェントの後に存在)
- J2EE ベリファイアユーティリティ
- Apache Ant タスクと Sun のオプションタスク
- 管理コンポーネント
    - Web ベースの管理サーバー

- すべての機能をリモート実行できるコマンド行インタフェース

Sun ONE Message Queue 3.0.1

Sun ONE Message Queue 3.0.1 はアプリケーションサーバーに統合されていますが、単独の製品としても入手可能です (従来は iPlanet Message Queue と呼ばれていました)。このコンポーネントはすべて Java で記述されていて、JMS クライアントと MDB の両方に対応する堅牢な JMS プロバイダとして機能します。

Sun ONE Studio 4, Enterprise Edition for Java

アプリケーションサーバーの評価用バージョンには、Sun ONE Studio 4, Enterprise Edition for Java IDE の試用ライセンスが含まれています。これは、従来の Forte for Java 3, Enterprise Edition for Java の後継製品です。

PointBase データベースサーバーおよび Type 4 JDBC ドライバ

付属のサンプルアプリケーションの操作と JDBC ベースのアプリケーション開発に対応するために、アプリケーションサーバーにはリレーショナルデータベース PointBase Server 4.2 が含まれています。JDBC API に対応した PointBase の Type 4 ドライバ (「JDBC ドライバ」) は、インストール時に設定されます。テーブルも作成され、JDBC に依存するすべてのサンプルアプリケーションの内容が設定されます。付属する PointBase では、データベースの容量が 5M バイトに制限されています。

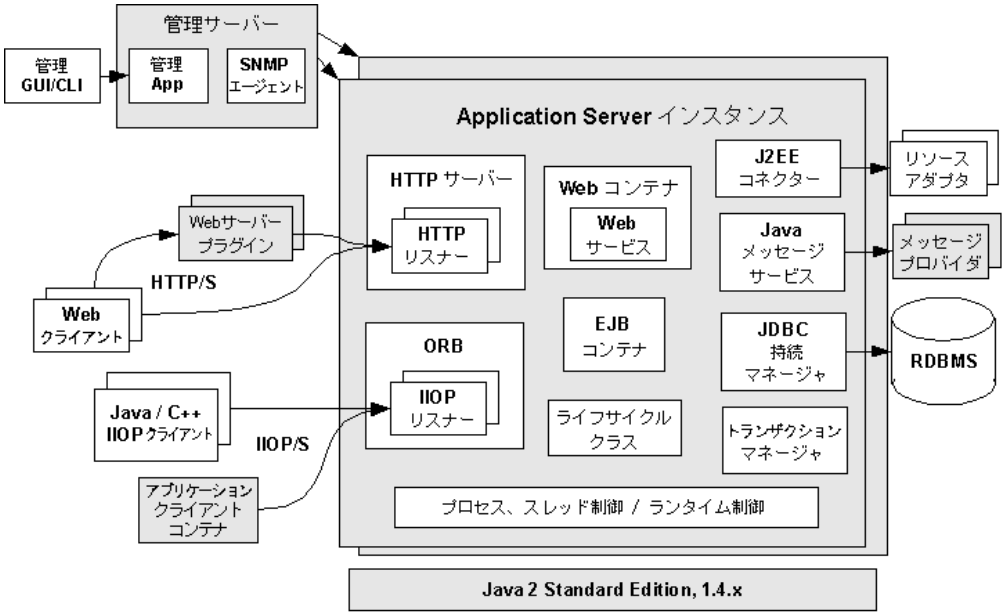
Java 2 Software Development Kit, Standard Edition 1.4_02

Sun ONE Application Server は、J2SE 1.4 プラットフォームで向上した性能および機能を利用しているため、J2SE 1.4.x, SDK が必要です。

アーキテクチャの概要

Sun ONE Application Server 7 の配備は、多数のアプリケーションサーバーインスタンスと 1 つの管理サーバーから構成され、オプションとして 1 つまたは複数の Web サーバー層プロキシプラグインが含まれます。

Sun ONE Application Server の配備



アプリケーションサーバーの配備の中心は、アプリケーションサーバーインスタンスです。アプリケーションサーバーの各インスタンスには、J2EE 1.3 Web コンテナと EJB コンテナが含まれます。定評のある高性能 HTTP サーバーは Web コンテナの前段に配備され、EJB コンテナは内蔵の ORB に支えられています。バックエンドシステムにアクセスできるので、アプリケーションは J2EE コネクタアーキテクチャとサードパティのリソースアダプタ、JMS と内蔵の JMS プロバイダーまたはサードパーティ製のプロバイダ、および一般的なサードパーティ製の JDBC ドライバを自由に組み合わせて使用できます。分散トランザクションの範囲内であれば、バックエンドシステムへのアクセスは、すべてが Java で記述された内蔵のトランザクションマネージャを使って管理できます。

管理サーバーには、コア管理アプリケーションと SNMP エージェントが格納されています。リモート管理機能は、管理サーバーを通じて実行されます。コマンド行ベースと Web ブラウザベースの管理クライアントは、どちらも HTTP または HTTPS を介して管理サーバーに直接アクセスします。HTTPS を使用する場合は、セキュリティ保護されたアクセスが可能です。

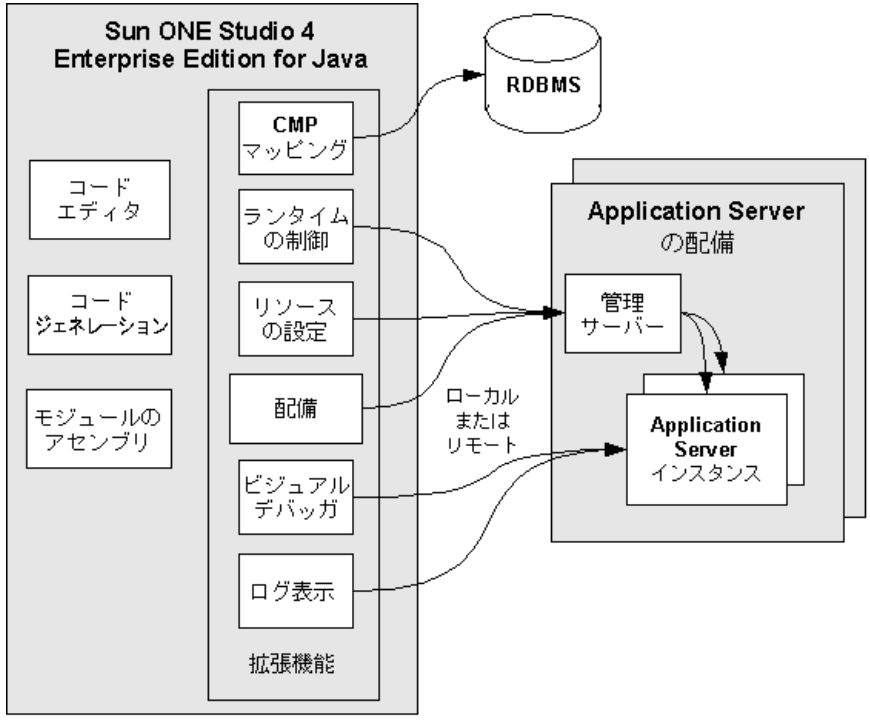
Web サーバープロキシプラグインにより、1 つまたは複数のファイアウォールで保護された非武装ゾーン (DMZ) に設置された 1 つまたは複数の Web サーバーにもアプリケーションサーバーを配備できます。プラグインは、インターネットから受信した HTTP または HTTPS トラフィックを、フロントエンドの Web サーバーがバックエンドのアプリケーションサーバー層に置かれている 1 つまたは複数のアプリケーションサーバーに配信するときに使われます。

多様なクライアントアプリケーションが、アプリケーションサーバーに配備されたビジネスサービスにアクセスできます。Web サービスとブラウザベースのクライアントは、HTTP または HTTPS のいずれかを使って Java Web サービスと J2EE Web アプリケーションにアクセスできます。Java アプリケーションクライアントは、スタンドアロンとして配備できるだけでなく、標準のアプリケーションクライアントコンテナ内にも配備できます。また、Java RMI-IIOP テクノロジ (Java Remote Method Invocation over Internet Inter-ORB Protocol テクノロジ) を使って、アプリケーションサーバーに配備された EJB にもアクセスできます。C++ 言語のクライアントも、Java IDL/IIOP を使って EJB にアクセスできます。

開発の統合

Sun ONE Application Server 7 の重要な機能の 1 つに、統合開発環境 (IDE) による Sun ONE Studio 4, Enterprise Edition for Java との密接な統合があげられます。アプリケーションサーバーには、Sun ONE Application Server を使用する開発者の生産性をさらに向上するために、Sun ONE Studio の主要機能を拡張する統合モジュールが用意されています。

Sun ONE Application Server の統合開発環境



Sun ONE Application Server の統合モジュールは、次の表に示す Sun ONE Studio 4, Enterprise Edition for Java の主要開発機能を拡張します。

主要開発機能

Studio の機能	Sun ONE Application Server による拡張
CMP マッピング	開発者は、データベーステーブルを検索して関連するテーブルを選び、コンテナ管理持続性 (CMP) EJB テクノロジを自動的に生成できる
サーバーランタイムの制御	開発者は、リモートとローカル両方のアプリケーションサーバーを簡単に登録し、アプリケーションサーバーのインスタンスを起動および停止できる
リソースの設定	アプリケーションを配備する前に、開発者は登録されているアプリケーションサーバーに J2EE リソースを登録できる。Studio では、JDBC リソースと接続プール、JMS リソース、その他の各種リソースを設定できる

主要開発機能 (続き)

Studio の機能	Sun ONE Application Server による拡張
アプリケーションの配備	開発者は、登録されているアプリケーションサーバーのリストからサーバーを選び、Sun ONE Application Server 7 でサポートされている動的な (「ホット」) 配備・再配備機能を利用できる
デバッグとログの表示	ローカルとリモート両方のアプリケーションサーバーインスタンスで、配備されているアプリケーションを簡単にデバッグできる。アプリケーションサーバーの手動設定は必要ない アプリケーションのデバッグ時に、開発者は Studio でもサーバーイベントログファイルの内容を確認できる

配備トポロジ

Sun ONE Application Server 7 の Platform Edition と Standard Edition は、単一のマシンと複数のマシンの多層配備に対応しています。これらのエディションは、アプリケーションサーバーのクラスタリングには対応していません。Standard Edition では、同じ Web サーバーインスタンスで受信する HTTP または HTTPS トラフィックを中間層の複数のアプリケーションサーバーに分配することで、Web 層のサポートを拡張しています。Platform Edition と Standard Edition は、プラグインによるロードバランスには対応していません。

Platform Edition では各管理ドメインに対し 1 つのアプリケーションサーバーインスタンス (1 つの JVM プロセス) しか設定できませんが、Standard Edition では、1 つの管理ドメインに複数のアプリケーションサーバーインスタンスを設定できます。

Enterprise Edition では、複数層、複数マシンへの配備、およびクラスタリングによるアプリケーションサーバーの配備をサポートしています。また、Web サーバープラグインにより Web 層でのロードバランス、およびアプリケーショントラフィックの分配をサポートしています。さらに Enterprise Edition では、アプリケーションサーバーインスタンスのクローン作成、およびクラスタリングに関連する管理作業も可能です。

第 2 章

Sun ONE Application Server の評価用 (Evaluation) バージョンのインストール

製品のインストールが完了している場合は、第 3 章「環境の設定」に進みます。

Solaris 9 環境で、Solaris 9 のインストール時にアプリケーションサーバーを同時にインストールした場合は、32 ページの「Solaris 9 インストール後の設定」に記載されている Solaris インストール後の作業手順に従ってアプリケーションサーバーの環境を設定してください。

製品を初めてインストールする場合は、この章の説明に従って操作します。この章では、製品の評価目的用のバージョンを対話モードでインストールする方法について説明します。評価用バージョン以外の製品をインストールするか、あるいはサイレントモードでインストールする方法については、『Sun ONE Application Server 7 インストールガイド』を参照してください。

Sun ONE Application Server 7 のインストールは、ユーザーロールや使用する方法 (コマンド行、GUI、サイレントモード) によって異なります。この章では、インストールの種類と、評価目的用のバージョンを GUI モードでインストールする方法について説明します。運用または開発用に実際の製品をインストールする準備ができたら、『インストールガイド』のインストール手順を参照してください。

この章では次の項目について説明します。

- インストールについて
- 評価用バージョンのインストール
- Sun ONE Application Server のアンインストール
- Solaris 9 インストール後の設定

インストールについて

『Sun ONE Application Server 7 インストールガイド』に従ってセットアッププログラムを実行する前に、この章でインストールの種類と方法を確認してください。ユーザーロールに基づいてインストールの種類を選択し、インストール環境に最適のインストール方法を選択します。

製品の種類

Sun ONE Application Server には、評価用バージョンと、それ以外のバージョンがあります。評価用バージョンのインストールでは、ユーザーの入力は最小限ですみます。

評価目的以外のバージョンでは、実際の運用または開発用に製品をインストールできます。また、コンポーネントを指定してインストールできます。このマニュアルでは、評価用バージョンのインストールについて説明します。

評価用バージョンのインストールでは、次のコンポーネントがインストールされます。

- Sun ONE Application Server 7 Core Server およびユーティリティ
- Sun ONE Message Queue 3.0.1
- 入門ガイド
- サンプルアプリケーション
- Java 2 Software Development Kit, Standard Edition 1.4_02
- PointBase データベースサーバーおよび Type 4 JDBC ドライバ
- Sun ONE Studio 4, Enterprise Edition for Java (試用ライセンス)

  Sun ONE Studio 4, Enterprise Edition for Java を含む評価用バージョンをインストールした場合は、このコンポーネントもインストールされます。

インストール方法

評価用バージョンでは、次のいずれかの方法でアプリケーションサーバーをインストールできます。

- グラフィカルインタフェースによるインストールでは、対話型のダイアログを使います。

- コマンド行による対話型のインストールでは、グラフィカルインタフェースと同様の手順でインストールを行いますが、グラフィック表示に対応したディスプレイは必要ありません。たとえば、telnet からリモートサーバーにアクセスしている状況でも、この方法であれば対話モードで製品をインストールできます。この方法でインストールするときは、`-console` オプションを指定してインストーラを起動します。
- サイレントモード (パラメータ駆動モード) によるインストールでは、用意されているパラメータファイルに基づいて製品のインストールをスクリプトで実行します。この方法で製品をインストールする場合は、途中の操作は必要ありません。

このマニュアルではグラフィカルインタフェースによるインストールについて説明しますが、コマンド行による対話モードのインストールも簡単に実行できます。

サイレントモードによるインストールの詳細については、『インストールガイド』を参照してください。

評価用バージョンのインストール

この節には次の項目があります。

- インストールの準備
- Solaris 環境で必要なパッチ
- セットアッププログラムの実行

評価用バージョンの Sun ONE Application Server には、標準的なものと Sun ONE Studio 4.0 が付属したものがあります。どちらをダウンロードするかは、ユーザーが選択できます。

インストールの準備

システム要件は次の表のとおりです。

オペレーティングシステム	アーキテクチャ	最小メモリー容量	推奨メモリー容量	最小ディスク容量	推奨ディスク容量
Sun Solaris 8 SPARC 版、Sun Solaris 9 SPARC 版	32 ビット / 64 ビット	Sun ONE Studio なしの場合、128M バイト Sun ONE Studio ありの場合、512M バイト	512M バイト	250M バイト	500M バイト
Microsoft Windows • 2000 Advanced Server、SP2+ • 2000 Server、SP2+ 以上 • 2000 Professional、SP 2 以上 • Windows XP Professional	Intel 32 ビット	Sun ONE Studio なしの場合、128M バイト Sun ONE Studio ありの場合、256M バイト	Sun ONE Studio なしの場合、256M バイト Sun ONE Studio ありの場合、512M バイト	250M バイト	500M バイト

Solaris 環境で必要なパッチ

Solaris 8 システムでは、次の Solaris パッチをインストールする必要があります。

109326-06 以上

108827-26 以上

110934 (パッケージに含まれる製品をインストールする場合のみ)

上記パッチは、次の SunSolve のパッチ検索ページから入手できます。

http://sunsolve.sun.com

Solaris 8 を使用する場合は、Sun 推奨パッチクラスタをインストールすることをお勧めします。このクラスタには、上記の 3 種類のパッチが含まれています。このパッチは、http://sunsolve.sun.com/ の「Recommended and Security Patches」から入手できます。

システム要件を満たしている環境では、アプリケーションサーバーのインストールを開始できます。

セットアッププログラムの実行

次に説明する手順では、Sun ONE Application Server 7 をグラフィカルインタフェースモードでインストールします。コマンド行インタフェースのインストールでも、操作手順は同じです。GUI ベースのインストール画面の代わりにテキストベースの情報がコンソールに表示されます。

次の手順は、すべてのプラットフォームで実行できます。

注 Sun ONE Application Server 7 をインストールするには、Windows マシンの管理権限が必要です。1 台の Windows マシンに複数の Sun ONE Application Server をインストールすることはできません。

1. インストール用の圧縮ファイルを一時ディレクトリに展開 (解凍) します。
2. ファイルの展開先ディレクトリに移動します。

注 UNIX 環境では、Sun ONE Application Server をリモートインストールする場合は、インストール先マシンの表示設定を有効にします。

3. コマンドプロンプトに次のコマンドを入力します。

   ```
   #./setup
   ```

   コマンド行によるインストールでは、次のように入力します。

   ```
   #./setup -console
   ```

   Windows 環境では、エクスプローラを使ってファイルの展開先フォルダを開き、`setup` ファイルをダブルクリックするという方法もあります。

   インストールインタフェースが表示されます。
4. 「Welcome」画面の内容を確認し、「Next」をクリックします。
5. ライセンス契約の内容を確認し、契約内容に同意するラジオボタンを選択して、「Next」をクリックします。

   インストールを続行するには、ライセンス契約に同意する必要があります。
6. インストールディレクトリのパスを入力します。(...) をクリックしてディレクトリをブラウズすることもできます。

ディレクトリが存在しない場合は、「Create New Directory?」ダイアログボックスが表示されます。「Create Directory」をクリックします。「Choose New」をクリックして既存のディレクトリを選択することもできます。

7. 「Server Configuration Information」ダイアログボックスに次の情報を入力します。
   - Admin User: サーバーの管理者の名前です (`admin` など)。
   - Admin User's Password: 管理サーバーにアクセスするパスワードです。8 文字以上の文字列を入力します。確認のため、その下のテキストボックスにも同じパスワードを入力します。
   - Admin Server Port: 管理サーバーにアクセスするポート番号です。

     デフォルトのポート番号が表示されます (現在のマシン上で使用されていなければ、たとえば `4848` など)。必要に応じてデフォルトのポート番号を変更します。「Next」をクリックすると、インストールプログラムにより、指定のポート番号が有効かつ使用可能であるかどうかがチェックされます。
   - HTTP Server Port: デフォルトサーバーインスタンスにアクセスするポート番号です。

     デフォルトのポート番号が表示されます (現在のマシン上で使用されていなければ、たとえば `1024` など)。必要に応じてデフォルトのポート番号を変更します。「Next」をクリックすると、インストールプログラムにより、指定のポート番号が有効かつ使用可能であるかどうかがチェックされます。

注 このインストールプログラムは使用中のポートを検出し、現在使われていないポートの番号を自動的に表示します。標準の設定では、UNIX 環境または Windows 環境のルートとして実行している場合、HTTP サーバーと管理サーバーのデフォルトポート番号は、それぞれ 80 と 4848 です。UNIX 環境のルート以外で実行している場合は、HTTP サーバーのデフォルトポート番号は 1024 です。システムでこれらのデフォルトポートが使用されている場合、インストールプログラムは別のポート番号を表示します。

---

8. 「Checking Disk Space」ダイアログボックスが表示されます。
9. 「Install Now」をクリックしてインストールを完了します。

   インストールの進捗インジケータバーが表示されます。

   インストールが終了すると、「Installation Complete」画面が表示されます。内容を確認し、「Finish」をクリックしてインストーラを終了します。

第 3 章「環境の設定」に進みます。

インストール時の問題を解決するには、『インストールガイド』に記載されているトラブルシューティングを参照してください。

Sun ONE Application Server のアンインストール

Sun ONE Application Server for Evaluation をアンインストールすると、すべてのコンポーネントが自動的にアンインストールされます。アンインストールするコンポーネントを個別に指定することはできません。

アンインストールが実行される前に、アンインストールプログラムにより実行中のプロセスがすべて検出され、停止されます。

UNIX、Windows のどちらのプラットフォームを使用している場合も、次の手順に従ってください。

1. Sun ONE Application Server のインストールディレクトリに移動します。
2. コマンドプロンプトに次のいずれかのコマンドを入力します。
   - GUI モード :

     ```
     ./uninstall
     ```

   - コマンド行モード :

     ```
     ./uninstall -console
     ```

3. 「Welcome」画面の内容を確認し、「Next」をクリックするか Enter キーを押して、アンインストールを続行します。
4. 「Ready to Uninstall」画面に、アンインストール可能なコンポーネントのリストが表示されます。確認後、「Next」をクリックするか Enter キーを押して続行します。

   Evaluation バージョンの Sun ONE Application Server では、アンインストールするコンポーネントを個別に指定することはできません。
5. 「Uninstall Now」をクリックするか、Enter キーを押します。

   アンインストールの進捗状況が表示されます。
6. 「Uninstall Summary」画面が表示されます。

   内容を確認し、「Exit」をクリックするか Enter キーを押してアンインストールプログラムを終了します。
7. インストールディレクトリに移動し、残ったファイルやディレクトリを手作業で削除します。

Solaris 9 インストール後の設定

Solaris 9 のインストール時に同時にインストールした Sun ONE Application Server 7 には、必要なライブラリ、実行可能ファイル、およびアプリケーションサーバーが必要とするその他のファイルだけが含まれています。インストール時にアプリケーションサーバーの設定は行われません。初期設定を行うには、アプリケーションサーバーの `asadmin` コマンド行インタフェース (CLI) を使う必要があります。`asadmin` コマンド行インタフェースのサブコマンド `create-domain` を実行すると、指定したディレクトリに管理ドメインが作成されます。管理ドメインは、管理サーバーの設定とアプリケーションサーバーインスタンスの設定から構成されます。最初の管理ドメインを作成するときは、そのドメインの管理サーバーと関連づける管理ユーザー名、パスワード、およびポート番号を指定します。

管理ドメインを作成すると、そのドメインに 1 つまたは複数のアプリケーションサーバーインスタンスを作成できるようになります。アプリケーションサーバーの各インスタンスには、HTTP サーバー、J2EE の Web コンテナと EJB コンテナ、およびアプリケーションサーバーのその他のファイルが保存されます。

管理ドメインとアプリケーションサーバーインスタンスについては、このマニュアルでさらに詳しく説明します。

管理ドメインの作成

標準の設定では、サブコマンド `create-domain` を実行して作成した新しい管理ドメインの設定は、`/var/appserver/domains/` ディレクトリに保存されます。この領域への書き込み権が割り当てられていないユーザーとしてログインした場合は、管理ドメインの作成時には別のディレクトリを指定する必要があります。ドメイン設定の保存場所を指定するには、サブコマンド `create-domain` に `--path` オプションを指定して実行します。

管理ドメインを作成する手順は次のとおりです。

1. ルートユーザー ID 以外の ID でログインしている場合、管理ドメインの作成に必要なアクセス権がその ID に割り当てられていないときは、システム管理者に連絡してドメインを作成してもらうか、管理ドメインを作成できる UNIX グループにユーザー ID を追加してもらいます。

注 **UNIX 環境でのユーザーのアクセス権**: ルートユーザー ID を使ってアプリケーションサーバーをインストールした場合、ルート以外のユーザーとしてアプリケーションサーバーとサンプルアプリケーションを操作するには注意が必要です。

ルート以外のユーザーが管理ドメインを作成、削除するには、ドメインの設定ファイルに対して書き込み権を持つ UNIX グループにそのユーザーの ID を追加する必要があります。デフォルトの設定では、UNIX グループ `asadmin` にはドメイン設定ファイルに対する書き込み権があります。

また、ルートユーザー ID を使うシステム管理者に連絡して、サブコマンド `create-domain` に `--sysuser` オプションを指定して、ユーザー ID を特定したドメインを作成してもらうこともできます。

ユーザー ID が特定された管理ドメインでは、その ID を使うユーザーは `asadmin` のサブコマンドを実行して新しいアプリケーションサーバーインスタンスを作成したり、そのインスタンスに対して様々な管理タスクを実行したりできます。この場合、管理ドメインの設定ファイルに対して書き込み権を持つ UNIX グループにユーザー ID を追加する必要はありません。このグループへの登録が必要なのは、管理ドメインを作成、削除するユーザーだけです。

2. パスに `/usr/sbin` ディレクトリが追加されていることを確認します。
3. コマンド行で次のコマンドを実行し、`domain1` という新しい管理ドメインを作成します。

```
asadmin create-domain --path <domain_config_dir> --adminport 4848
--adminuser admin --adminpassword password domain1
```

`<domain_config_dir>` には、管理ドメインの設定を保存するディレクトリを指定します。

`--adminport`、`--adminuser`、および `--adminpassword` オプションには、ドメインに定義する新しい管理サーバーの初期設定を指定します。

サブコマンド `create-domain` の実行をルートユーザーに代行してもらう場合は、`--sysuser` オプションを使って管理ドメインのファイルとディレクトリの保存先にユーザー ID に設定してもらいます。次に例を示します。

```
asadmin create-domain --sysuser ckamps --path <domain_config_dir>
--adminport 4848 --adminuser admin --adminpassword password
domain1
```

サブコマンド `create-domain` を実行すると、次のようなメッセージが表示されます。

```
ドメイン domain1 を作成しました
```

domain1 というドメイン名がすでに使われている場合は、別の名前を指定してサブコマンド create-domain を実行します。ドメイン名にはピリオドなどの文字も使用できます。ドメイン名が重複しないように、ログインユーザー名を含めることも可能です。次に例を示します。
ckamps.domain1.

サブコマンド create-domain を実行したときに次のようなエラーメッセージが表示される場合は、ドメイン設定ファイルに対するアクセス権がユーザー ID に割り当てられていないことを意味します。

ドメインを作成できません : domain1
problem locking store /etc/appserver/domains.lck (アクセス権がありません)

前述の手順に従って、システム管理者に連絡してユーザー ID を適切な UNIX グループに追加してもらうか、管理ドメインを作成してもらいます。

4. サブコマンド list-domains を実行し、アプリケーションサーバーのインストールに設定されているすべてのドメインをリスト表示します。これは読み取り専用のコマンドなので、ドメイン設定ファイルに対して書き込み権を持つ UNIX グループにユーザー ID が登録されていなくても実行できます。

```
asadmin list-domains
domain1 [<domain_config_dir>/domain1]
```

<domain_config_dir> は、新規作成した管理ドメインのデフォルトディレクトリ、またはサブコマンド create-domain の実行時に --path オプションで指定したディレクトリとなります。

アプリケーションサーバーインスタンスの作成

システム管理者が管理ドメインを作成すると、または自身で管理ドメインを作成すると、新しい管理ドメインにアプリケーションサーバーのインスタンスを作成できるようになります。専用の管理ドメインにアプリケーションを作成するには、ドメイン設定ファイルに対して書き込み権を持つ UNIX グループにユーザー ID が登録されている必要はありません。

アプリケーションサーバーのインスタンスを作成するには、サブコマンド create-instance を実行します。

```
asadmin create-instance --domain domain1 --instanceport 80 server1
```

domain1 はドメイン作成時に指定したドメイン名、80 はアプリケーションサーバーインスタンスの HTTP サーバーのポート番号、server1 はインスタンスの名前です。作業環境に応じて任意の値をオプションに指定します。番号が 1024 未満のポートにアクセスできるのはルートユーザーだけなので、ルート以外のユーザーは 1024 以上の値を指定する必要があります。

システムに定義したドメインが 1 つだけであれば、インスタンスの作成時に対象ドメインを指定する必要はありません。

Solaris 9 のインストール時に同時にインストールしたアプリケーションサーバーの環境設定の詳細は、『Sun ONE Application Server 管理者ガイド』を参照してください。

アプリケーションサーバーの最初の管理ドメインとアプリケーションサーバーインスタンスを作成したら、第 4 章「アプリケーションサーバーの起動と停止」に進みます。

第 3 章

環境の設定

アプリケーションサーバーをインストールしたら、作業環境にアプリケーションサーバーの bin/ ディレクトリを設定します。この章では、PATH 環境変数に次のディレクトリを追加する方法について説明します。

```
<install_dir>/bin
```

アプリケーションサーバーを管理するコマンド行ユーティリティ asadmin を実行したり、asant ユーティリティにアクセスしてサンプルアプリケーションを操作したりする上で、必要な環境設定はこれだけです。

Solaris 9 のインストール時にアプリケーションサーバーを同時にインストールした環境で、asant ユーティリティにアクセスするには PATH 環境変数に /usr/appserver/bin ディレクトリを追加する必要があります。また、PATH に /usr/sbin が含まれない環境でコマンド行ユーティリティ asadmin にアクセスするには、PATH に /usr/appserver/bin を追加する必要があります。

注

一部の Windows 2000 環境では、PATH 環境変数の追加設定をしないと Windows の net コマンドを利用できません。アプリケーションサーバーを起動、停止するには、この Windows ユーティリティを使えるようにしておく必要があります。

作業環境で net コマンドを使えるかどうかを調べる手順は、次のとおりです。

1. 「スタート」->「ファイル名を指定して実行」を選び、ダイアログボックスを開きます。
2. 「名前」フィールドに cmd と入力して「OK」をクリックします。
3. コマンドコンソールが起動するので、プロンプトに net と入力します。コマンドが見つからないときは、PATH 環境変数を編集し、<Windows install root>¥system32 ディレクトリを追加します。たとえば、次のようなディレクトリです。

   ```
   C:¥WINNT¥system32;
   ```

システム環境変数 PATH の設定がよくわからない場合は、詳細な変更手順について次項を参照してください。

環境変数の設定方法に問題ない場合は、必要に応じて設定してください。

PATH 変数を設定すると、asadmin コマンドを使用できるようになります。43 ページの「アプリケーションサーバーの起動と停止」に進んでください。コマンドを実行できない場合は、PATH の設定を調べて環境設定を変更し、もう一度 asadmin を実行してください。

環境変数の設定方法がよくわからない場合は、作業環境のプラットフォームに応じて次のいずれかの設定手順に従ってください。

- UNIX 環境での PATH 変数の設定
- Windows 環境での PATH 変数の設定

UNIX 環境での PATH 変数の設定

UNIX 環境では、ログイン時にアプリケーションサーバーの bin ディレクトリが自動的に PATH に追加されるように、このディレクトリをログインプロファイルに追加することをお勧めします。

PATH 環境を設定して環境を更新したら、コマンドプロンプトから asadmin コマンドを実行します。

実行結果が、次のように表示されます。

```
asadmin
Use "exit" to exit and "help" for online help
       asadmin>_
```

管理用インタフェースを終了するときは、コマンド行に exit と入力します。端末ウィンドウを開いたまま、43 ページの「アプリケーションサーバーの起動と停止」に進みます。

コマンドにアクセスできない場合は、PATH の設定を調べて環境設定を変更し、もう一度 asadmin を実行してください。

注

asadmin コマンドを実行すると、アプリケーションサーバーの管理用コマンド行インタフェースが起動します。引数を指定せずに asadmin を実行した場合は、コマンド行インタフェースは対話モードとなります。

asadmin コマンドプロンプトで help と入力すると、このコマンド行インタフェースがサポートしているすべてのサブコマンドがリスト表示されます。

asadmin コマンドは、サーバーの管理や監視を自動化するために事前に作成したコマンドファイルにも完全に対応しています。このマニュアルで使うコマンドは一部だけなので、『Sun ONE Application Server 管理者ガイド』に記載されているコマンド行インタフェースの解説をよく読むことをお勧めします。

Windows 環境での PATH 変数の設定

Windows 環境では、Windows のコントロールパネルを使ってシステム環境変数 PATH を変更することをお勧めします。手順は次のとおりです。

1. 「スタート」>「設定」>「コントロールパネル」を選びます。
2. 「コントロールパネル」が表示されるので、「システム」をクリックします。
3. 「詳細」タブを選び、「環境変数」をクリックします。

   「環境変数」ダイアログには、現在のユーザーアカウントに適用される環境変数と、システム全体に適用される環境変数がリスト表示されます。

   ここでは、現在のユーザーアカウントに適用される環境変数 PATH を変更します。
4. 表示されている PATH 変数を選んで「編集」をクリックします。PATH 変数が表示されないときは「新規」をクリックして新たに作成します。
5. 「変数値」フィールドに表示される値の最初に <install_dir>¥bin を追加します。

   たとえば、変数値の先頭に C:¥Sun¥AppServer7¥bin; を追加します。

6. 「OK」をクリックして「ユーザー変数の編集」ウィンドウを閉じます。

   入力したディレクトリパスが PATH 変数に追加されます。

7. 「OK」をクリックして変更を適用し、「環境変数」ウィンドウを閉じます。次に、「OK」をクリックして「システムのプロパティ」ウィンドウを閉じます。

8. コマンドコンソールを起動し、コマンド行からアプリケーションサーバーのコマンドにアクセスできることを確認します。

   a. 「スタート」>「ファイル名を指定して実行」を選びます。

   b. 「名前」フィールドに `cmd` と入力して「OK」をクリックします。

   c. コマンドコンソールが表示されたら、コマンドプロンプトに `asadmin` と入力します。

      `asadmin` コマンドを実行すると、次のようなメッセージが表示されます。

      ```
      C:¥>asadmin

      Use "exit" to exit and "help" for online help

      asadmin>
      ```

      結果がこのように表示されれば、このパス設定でアプリケーションサーバーのコマンド行ユーティリティにアクセスできます。

注

`asadmin` コマンドを実行すると、アプリケーションサーバーの管理用コマンド行インタフェースが起動します。引数を指定せずに `asadmin` を実行した場合は、コマンド行インタフェースは対話モードとなります。

`asadmin` コマンドプロンプトで `help` と入力すると、このコマンド行インタフェースがサポートしているすべてのサブコマンドがリスト表示されます。

`asadmin` コマンドは、サーバーの管理や監視を自動化するために事前に作成したコマンドファイルにも完全に対応しています。このマニュアルで使うコマンドは一部だけなので、『Sun ONE Application Server 管理者ガイド』に記載されているコマンド行インタフェースの解説をよく読むことをお勧めします。

---

9. 管理用インタフェースを終了するときは、コマンド行に `exit` と入力します。

   コマンドコンソールを開いたまま、43 ページの「アプリケーションサーバーの起動と停止」に進みます。コマンドにアクセスできないときは、PATH の設定を調べて環境設定を更新し、もう一度 `asadmin` を実行してください。

環境設定のトラブルシューティング

asadmin コマンドが認識されないときは、39 ページの手順 2 に記載されているコントロールパネルの説明に戻り、PATH の設定を確認してからもう一度 asadmin コマンドを実行してください。PATH の設定を変更したときは、asadmin コマンドの実行をテストする前に新しいコマンドコンソールを起動します。環境変数の変更は、新しいコマンドコンソールだけに適用されます。PATH 変数の設定に誤りがあると、次のようなメッセージが表示されます。

```
C:¥>asadmin

'asadmin' is not recognized as an internal or external command,
operable program or batch file.

C:¥>
```

「アプリケーションサーバーの起動と停止」に進みます。

第 4 章

アプリケーションサーバーの起動と停止

インストールが完了してもアプリケーションサーバーは自動的に起動されません。サンプルアプリケーションを配備したり、実行したりする前に、アプリケーションサーバーを起動し、問題なく稼動することを確認する必要があります。この章では、アプリケーションサーバーの起動に先立って、管理ドメイン機能について解説し、アプリケーションサーバー環境を構成するプロセスについて簡単に説明します。

この章では次の項目について説明します。

- 管理ドメイン機能
- アプリケーションサーバーのプロセス
- インストールディレクトリの構造
- アプリケーションサーバーの起動と停止に使用するツール

管理ドメイン機能

Sun ONE Application Server 7 には管理ドメインという機能が導入されています。この機能を使うことで、同じインストールイメージを再利用する独立した複数のアプリケーション実行時環境を定義できます。それぞれの管理ドメインは管理サーバーとして扱われ、1 つまたは複数のサーバーインスタンスの制御はこの管理サーバーによって行われます。管理ドメインの設定情報は、マシン上のどの場所にも配備できます。

独自にワークステーションを所有する開発者の多くは日常の開発業務に 1 つの管理ドメインを使いますが、複数ドメインを使用すると、開発用の共用サーバーと運用環境の両方に利点があります。共用の開発用サーバーでは、開発者ごとに管理ドメインを作成することで、サーバーマシンを共用する開発者別に使用領域を区分けできます。運用環境では、複数の管理ドメインを使うことによってシステム管理者は複数のインストールする必要はなくなり、独立したセキュリティの保護された実行時設定を定義できます。

次の例では、1 つの管理ドメインを使います。このドメインは、製品のインストール時に設定したものでも、Soralis 9 のインストール時に同時にアプリケーションサーバーをインストールし、その後の作業で設定したものでもかまいません。

アプリケーションサーバーのプロセス

アプリケーションサーバーを起動する前に、各管理ドメインの背後で実行されるプロセスについて説明します。

- アプリケーションサーバーデーモン
- アプリケーションサーバーウォッチドッグ
- メッセージキューブローカ
- メッセージキューブローカラッパー

アプリケーションサーバーデーモン

アプリケーションサーバーデーモン `appservd` は、アプリケーションサーバーの実行時に中核となるプロセスです。このプロセスには、組み込み型の HTTP サーバー、ORB、J2EE コンテナ、およびトランザクションマネージャや持続マネージャなどのサポートサブシステムが格納されます。また、それぞれの `appservd` には Java 仮想マシン (JVM) が格納されています。

各管理ドメインには、そのドメインの管理サーバーを格納する `appservd` プロセスが用意されています。このため、このプロセスは管理サーバーインスタンスと呼ばれています。アプリケーションサーバーの個々のインスタンスを表すその他の `appservd` プロセスの制御および管理は、管理サーバーによって行われます。

注

UNIX 環境の appservd プロセス : Windows 環境ではアプリケーションサーバーのインスタンスごとに 1 つの `appservd` プロセスが起動しますが、UNIX 環境ではインスタンスごとに 2 つの `appservd` プロセスが起動します。

UNIX 環境で起動する 2 つの `appservd` プロセスのうち、1 つは「基本」プロセスと呼ばれ、もう 1 つは「ワーカー」プロセスと呼ばれます。ワーカープロセスはアプリケーションからの要求を実際に処理し、基本プロセスは全体を制御するコントローラとして機能します。アプリケーションサーバーの将来のリリースでは、アプリケーションサーバーインスタンスごとにワーカープロセス数を定義するオプションを用意する予定です。今回のリリースでは、アプリケーションサーバーインスタンスに定義できるワーカープロセスは 1 つだけです。

アプリケーションサーバーウォッチドッグ

アプリケーションサーバーウォッチドッグ `appservd-wdog` は、アプリケーションサーバーデーモンを監視するネイティブ言語プロセスです。アプリケーションサーバーデーモンに障害が発生すると、対応するウォッチドッグがそのデーモンを再起動します。それぞれのアプリケーションサーバーデーモンには、対応するウォッチドッグプロセスが個別に割り当てられています。

メッセージキューブローカ

デフォルトの設定では、アプリケーションサーバーの各インスタンスには Sun ONE Message Queue Message Broker プロセスが割り当てられます。メッセージブローカプロセスは、アプリケーションサーバーの JMS で中心的な役割を果たします。

インストール時に作成されるアプリケーションサーバーインスタンスは 1 つだけなので、アプリケーションサーバーの起動時に開始されるメッセージブローカプロセスも 1 つだけです。

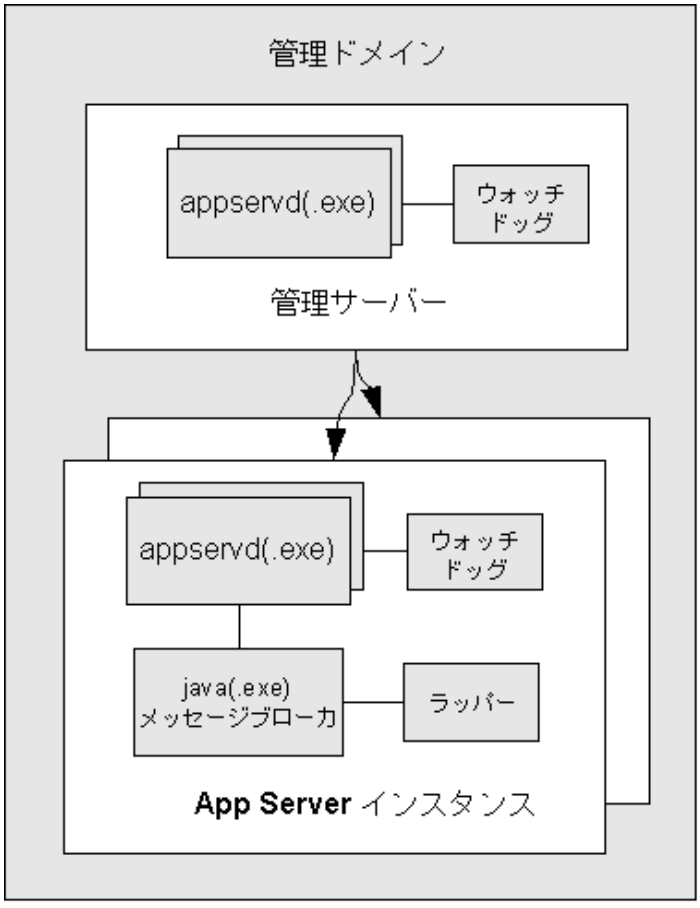
アプリケーションサーバーのインスタンスを新たに定義すると、それに対応するメッセージブローカプロセスが自動的に新規作成されます。アプリケーションサーバーインスタンスを作成した後で、そのインスタンスに対応するメッセージブローカプロセスを無効にできます。これは、アプリケーションに JMS の機能を使用させない場合や、MQSeries のようにアプリケーションサーバーに含まれないメッセージサービスを使用する場合に有効です。

メッセージキューブローカラッパー

このプロセスはメッセージブローカプロセスに付随する軽量のラッパーです。ラッパープロセスはメッセージブローカプロセスの再起動を制御します。

各プロセスの関連については次の図を参照してください。

アプリケーションサーバーのプロセス



インストールディレクトリの構造

アプリケーションサーバーが 1 つのルートディレクトリパスにインストールされるか、あるいは複数のルートディレクトリパスにまたがってインストールされるかは、インストールするアプリケーションサーバー製品の形態によって異なります。

- Solaris 9 でのバンドル版のインストール
- Solaris 8 および Solaris 9 のパッケージからのインストール
- Windows 環境でのインストールと Evaluation バージョンのインストール

Solaris 9 でのバンドル版のインストール

Solaris 環境で Solaris 9 のインストール時にパッケージベースのアプリケーションサーバーを同時にインストールすると、複数のルートディレクトリにまたがって製品がインストールされます。

Solaris 9 に含まれるアプリケーションサーバーがインストールされるディレクトリは次のとおりです。

- `/usr/appserver` には、インストールイメージの静的な要素が保存されます。ユーティリティ、実行可能ファイル、およびアプリケーションサーバーを構成するライブラリは、すべてここに保存されます。パッチのインストールや、製品のアップグレードを行なった場合を除き、この領域に保存された内容が変化することはありません。

  このディレクトリには次のサブディレクトリがあります。

  - `bin/` には実行可能ファイルとユーティリティが保存され、その一部は `/usr/bin` からのシンボリックリンクになっています。
  - `include/` にはレガシーヘッダーファイルが保存されます。
  - `lib/` にはネイティブライブラリと Java ライブラリが保存されます。
- `/etc/appserver/` には、ライセンスやそのインストール用に設定した管理ドメインのマスターリストなど、インストール全体に適用される設定情報が保存されます。
- `/var/appserver/domains` は、作成した管理ドメインが保存されるデフォルト領域です。

  Solaris 9 の一部としてアプリケーションサーバーをインストールしても、ドメインは作成されません。このため、最初のドメインを作成するまでこのディレクトリは存在しません。作成した管理ドメインは、システム上の任意の場所に移動できます。この領域は、ドメイン作成時のデフォルトの保存場所に過ぎません。

Solaris 8 および Solaris 9 のパッケージからのインストール

デフォルトの設定では、Solaris のパッケージに含まれるアプリケーションサーバーをインストールすると、一括インストールの場合と同様に複数のルートディレクトリにまたがって製品がインストールされます。

- `/opt/SUNWappserver7` には、インストールイメージの静的な要素が保存されます。ユーティリティ、実行可能ファイル、およびアプリケーションサーバーを構成するライブラリは、すべてここに保存されます。パッチのインストールや、製品のアップグレードを行なった場合を除き、この領域に保存された内容が変化することはありません。
- `/etc/opt/SUNWappserver7/config` には、ライセンスやそのインストール用に設定した管理ドメインのマスターリストなど、インストール全体に適用される設定情報が保存されます。
- `/var/opt/SUNWappserver7/domains` は、作成した管理ドメインが保存されるデフォルト領域です。

  Solaris 9 の一部としてアプリケーションサーバーをインストールしても、ドメインは作成されません。このため、最初のドメインを作成するまでこのディレクトリは存在しません。作成した管理ドメインは、システム上の任意の場所に移動できます。この領域は、ドメイン作成時のデフォルトの保存場所に過ぎません。

Windows 環境でのインストールと Evaluation バージョンのインストール

Windows 環境では、アプリケーションサーバーは 1 つのディレクトリパスにインストールされます。また、Solaris のパッケージに含まれないアプリケーションサーバーの Evaluation バージョンも 1 つのディレクトリパスにインストールされます。たとえば、次のようなディレクトリとなります。

- `c:¥Sun¥AppServer7` は、Windows 環境でのインストールディレクトリです。
- `<home_dir>/sun/appserver7` は、Solaris のパッケージに含まれない Evaluation バージョンを Solaris 環境にインストールした場合のインストールディレクトリです。

いずれの場合も、インストールディレクトリ内に `config/` ディレクトリと `domains/` ディレクトリが作成されます。

ディレクトリ名の表記規則

このマニュアルでは、アプリケーションサーバーのインストールディレクトリに含まれる主要ディレクトリを次の規則に従って表記します。

- <install_dir> は、ユーティリティ、実行可能ファイル、アプリケーションサーバーを構成するライブラリなど、インストールイメージの静的な要素を保存するディレクトリを意味します。

  この領域は読み取り専用です。パッチのインストールや、製品のアップグレードを行なった場合を除き、この領域に保存された内容が変化することはありません。

  たとえば、次のディレクトリがこれにあたります。

  - /usr/appserver
  - /opt/SUNWappserver7
  - <home_dir>/sun/appserver7/
  - c:¥Sun¥AppServer7

- <install_config_dir> は、ライセンスや管理ドメインのマスターリストを保存するディレクトリを意味します。

  たとえば、次のディレクトリがこれにあたります。

  - /etc/appserver/config
  - /etc/opt/SUNWappserver7/config
  - <home_dir>/sun/appserver7/config
  - c:¥Sun¥AppServer7¥config

- <domain_config_dir> は、作成した管理ドメインが保存されるディレクトリを意味します。

  たとえば、次のディレクトリがこれにあたります。

  - /var/appserver/domains
  - /var/opt/SUNWappserver7/domains
  - <home_dir>/sun/appserver7/domains
  - c:¥Sun¥AppServer7¥domain

アプリケーションサーバーの起動と停止に使用するツール

次に、アプリケーションサーバーを起動してみましょう。ここでは、管理サーバーの起動と、最初に設定したアプリケーションサーバーインスタンスの起動を行います。アプリケーションサーバーのインスタンスは、事前に設定した管理ドメインに定義されています。

アプリケーションサーバーは、次のいずれかの方法で起動します。

- コマンド行インタフェースの使用 (UNIX および Windows)
- 管理コンソールの使用 (UNIX および Windows)
- Windows のプログラムグループの使用
- Windows サービスの使用

コマンド行インタフェースの使用

オペレーティングシステムに関係なく、コマンド行インタフェース `asadmin` を使用すると、アプリケーションサーバー全体、特定の管理ドメイン、およびアプリケーションサーバーの個々のインスタンスの起動と停止を行えます。起動と停止に関連する `asadmin` のサブコマンドは次の表のとおりです。

起動と停止に関連するコマンド

サブコマンド	説明
`start-domain`	特定の管理ドメインに定義されたアプリケーションサーバーインスタンスおよび管理サーバーを起動する
`stop-domain`	特定の管理ドメインに定義されたアプリケーションサーバーインスタンスおよび管理サーバーを停止する
`start-instance`	特定のアプリケーションサーバーインスタンスを起動する。ローカルまたはリモートどちらのモードでも実行可能。ローカルモードで実行する場合は、管理サーバーが稼動している必要はない
`stop-instance`	特定のアプリケーションサーバーインスタンスを停止する。このサブコマンドの実行に関しては `start-instance` と同様

start-domain と stop-domain の使用

アプリケーションサーバーが稼動している状態であれば、次のコマンドを使用して管理サーバーだけでなく、最初に設定したドメインのアプリケーションサーバーインスタンスを停止できます。

```
asadmin stop-domain --domain domain1
```

domain1 は、アプリケーションサーバーのインストール時に定義した管理ドメインの名前です。

処理が完了すると、次のようなメッセージが表示されます。

```
asadmin stop-domain --domain domain1
インスタンス domain1:server1 が停止しました
ドメイン domain1 が停止しました。
```

注

Windows 環境では : Windows 環境では、ドメインの起動時にデスクトップにコマンドウィンドウが表示されます。この読み取り専用のコマンドウィンドウには、アプリケーションサーバーインスタンスのイベントログファイルの内容が表示されます。このログファイルは、関連するアプリケーションサーバーインスタンスが動作している間はデスクトップに表示されたままです。最初に設定したサーバーを起動すると、別のコマンドウィンドウが表示され、そこにイベントメッセージが表示されます。サーバーインスタンスが問題なく起動すると、数秒後に起動が成功したことを知らせるメッセージが表示されます。

起動メッセージを表示するコマンドウィンドウが表示されない場合 : 一部の Windows 2000 環境には Windows の net コマンドが含まれていないので、操作環境でこのコマンドを使用できることを確認してください。コマンドプロンプトから net コマンドを実行できない場合は、修正が必要となります。詳しくは第 3 章「環境の設定」を参照してください。

同様に、次のコマンドを実行して、最初に設定した管理ドメインを起動できます。

```
asadmin start-domain --domain domain1
```

処理が完了すると、次のようなメッセージが表示されます。

```
asadmin start-domain --domain domain1
インスタンス domain1:admin-server が起動しました
インスタンス domain1:server1 が起動しました
ドメイン domain1 が起動しました。
```

注 **asadmin のローカルモードでの実行 :** asadmin ユーティリティのほとんどのサブコマンドでは、実行時に対象管理サーバー、および適切な管理者のユーザー名とパスワードを指定する必要があります。ただし、一部のサブコマンドは、ローカルモードだけ、あるいはローカルモードとリモートモードの両方で実行できます。start-domain サブコマンドは、管理サーバーの起動だけでなく特定の管理ドメインに定義されているすべてのアプリケーションサーバーインスタンスの起動に関与しています。サブコマンド start-instance および stop-instance は、ローカルモードまたはリモートモードのどちらでも実行できます。リモートモードで使用できるオプション --user および --password を指定せずに実行した場合は、サブコマンドはローカルモードで実行したものとして解釈されます。

--local=true オプションまたは --local オプションを指定することで、サブコマンドを強制的にローカルモードで実行することもできます。

日常の作業では、管理サーバーの起動と停止を何度も繰り返すことは少ないので、ドメイン全体に適用される start|stop-domain コマンドよりも、インスタンスの起動と停止に適用されるサブコマンドを使用する場合のほうが多くなります。次の項では、アプリケーションサーバーの個々のインスタンスをコマンド行から起動および停止する方法について説明します。

start-instance と stop-instance の使用

管理サーバーの稼動状態に関係なくアプリケーションサーバーの特定のインスタンスを停止するには、次のコマンドを実行します。

```
asadmin stop-instance server1
```

server1 は、アプリケーションサーバーのインスタンス名を意味します。複数の管理ドメインが存在する環境では、stop-instance コマンドの実行時に管理ドメインの名前を指定する必要があります。

次に例を示します。

```
asadmin stop-instance --domain domain1 server1
```

アプリケーションサーバーの特定のインスタンスをローカルモードで起動するには、次のコマンドを実行します。

```
asadmin start-instance server1
```

インスタンスをリモートモードで起動または停止する場合は、start-instance コマンドおよび stop-instance コマンドに対象となる管理サーバー、および管理ユーザーの名前とパスワードを指定します。どちらのコマンドも、パラメータを指定せずに実行すると使用方法が表示されます。使用方法についてさらに詳細な情報が必要なときは、--help オプションを指定してサブコマンドを実行します。

注 **インスタンスを起動および停止するスクリプトについて** : UNIX と Windows のいずれの環境でも、インスタンスを起動および停止するスクリプトは `<domain_config_dir>/domain1/<instance name>/bin/` ディレクトリに保存されます。スクリプトファイルの名前は、それぞれ `startserv(.bat)` および `stopserv(.bat)` です。

このスクリプトを実行すると、UNIX 環境ではアプリケーションサーバーの実際のプロセスが直接起動および停止するのに対し、Windows 環境では対応する Windows サービスだけが起動および停止します。`asadmin` コマンドは、PATH 環境変数を設定するだけで使用できる場合が多いので、通常はインスタンスレベルのスクリプトを実行するよりも、サブコマンド `asadmin start-instance` および `stop-instance` を使うほうが便利です。

ヒント **管理サーバーとアプリケーションサーバーインスタンスの関係** : 最初に管理サーバーを起動せずにアプリケーションサーバーのインスタンスを起動することができます。管理サーバーの起動と、それに関連するインスタンスの起動は、相互に依存していません。ただし、管理サーバーが稼動していない状態では、アプリケーションサーバーインスタンスに対する管理作業は実行できません。管理サーバーが稼動していない状態でアプリケーションサーバーインスタンスを実行しても、アプリケーションは正常にロードされ、要求も正しく処理されます。

プロセス起動後、アプリケーションサーバーが正常に起動したかどうか確認する方法がいくつかあります。正常に起動した場合は、そのメッセージがアプリケーションサーバーのイベントログウィンドウに表示されます。ここでは、アプリケーションサーバーの起動状態を確認するためのその他の方法について説明します。

デスクトップに表示されるサーバーインスタンスのイベントログのほかに、次のいずれかの方法でサーバーが正常に起動したことを確認できます。

- サーバーイベントログファイルの確認
- アプリケーションサーバーインスタンスの HTTP サーバーへのアクセス

サーバーイベントログファイルの確認

デフォルトの設定では、アプリケーションサーバーインスタンスのイベントログファイルはデスクトップに表示されますが、管理サーバーのイベントログは表示されません。通常は管理サーバーからの出力を確認する必要がないためです。しかし、問題が発生し、その原因を探る必要があるときは、次の手順で管理サーバーのイベントログを表示できます。

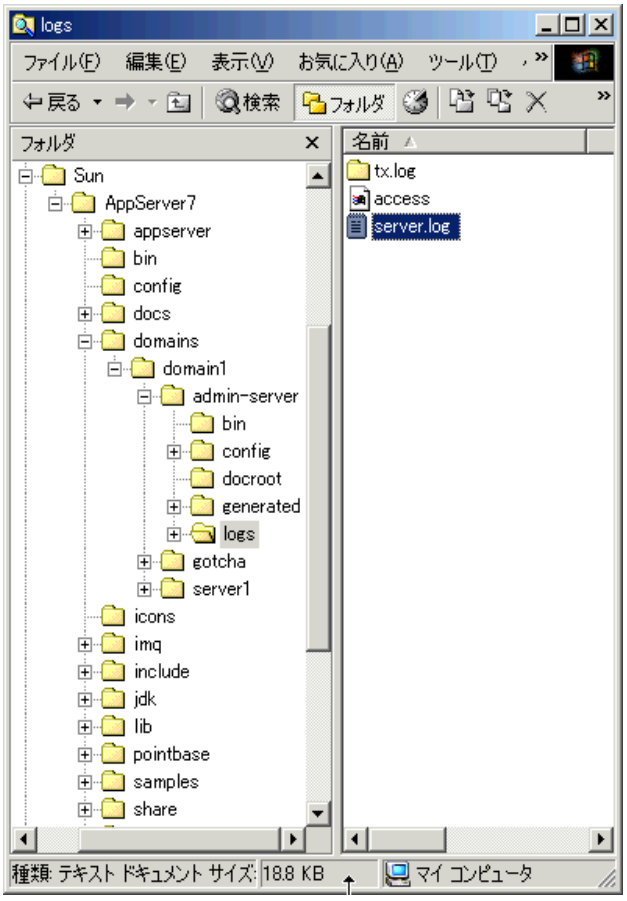
1. Windows でエクスプローラを起動し、管理イベントログファイルが保存されているディレクトリに移動します。

   ```
   <domain_config_dir>¥domain1¥admin-server¥logs¥
   ```

   たとえば、次のようなディレクトリです。

   ```
   c:¥Sun¥AppServer7¥domains¥domain1¥admin-server¥logs¥
   ```

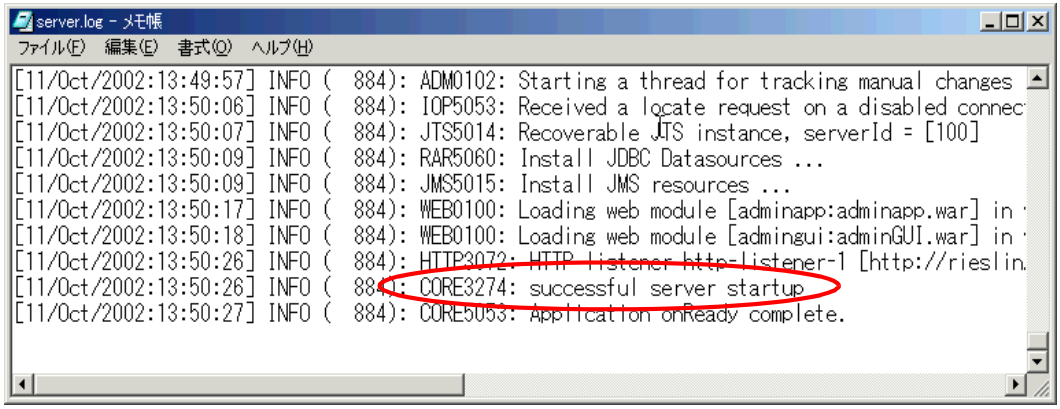
イベントログが保存されるディレクトリ



2. server.log ファイルをダブルクリックしてエディタで開きます。

   ログファイルの最後に successful server startup という情報が記録されています。

ログファイルの内容



正常な起動を示すメッセージが表示されない場合、管理サーバーが起動プロシージャを終了する前にイベントログファイルを開いてしまった可能性があります。ログファイルを閉じてから開き直し、最新のイベントメッセージを確認してください。

アプリケーションサーバーの通常操作では、管理サーバーが正常に起動されたかどうかを頻繁に確認する必要はありません。主な作業対象は、J2EE 開発作業の対象となるアプリケーションサーバーインスタンスのほうです。

ヒント **ドメインのディレクトリとインスタンスのディレクトリ**：管理サーバーのログファイルにアクセスするときに、domains/ というディレクトリと、それに関連する domains1/ というディレクトリが Windows エクスプローラに表示されていました。domains1/ ディレクトリには、製品のインストール時に作成された管理ドメインが格納されています。管理ドメインを新規作成する方法については後述しますが、新たに作成したドメインの設定は、デフォルトでは domains/ ディレクトリに保存されます。

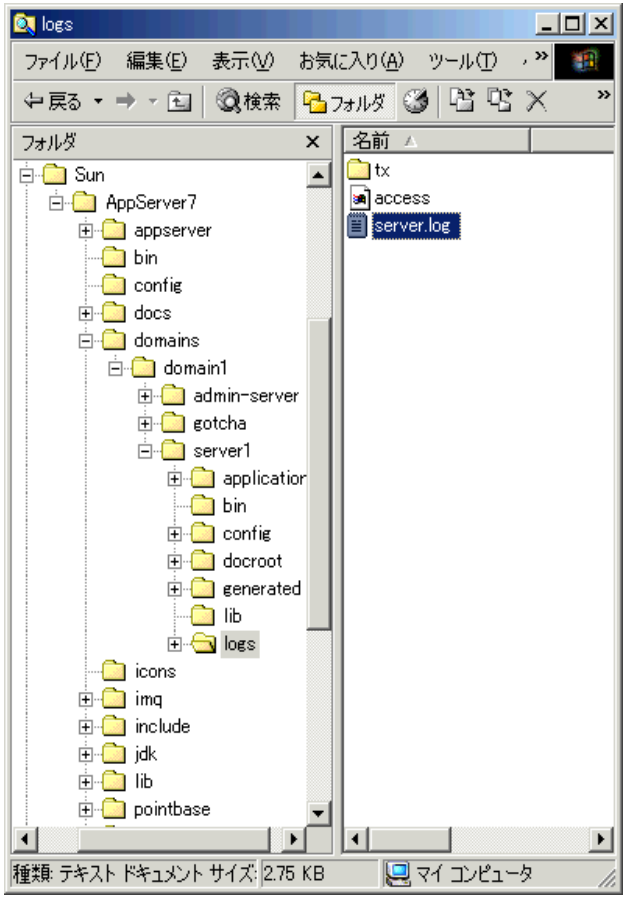
<domain_config_dir>/domain1 ディレクトリには、admin-server ディレクトリと同じ階層の server1 というディレクトリが含まれています。このディレクトリには、製品のインストール時に設定したアプリケーションサーバーインスタンスの設定情報、ログファイル、および配備したアプリケーションが保存されています。アプリケーションサーバーのインスタンスを追加すると、domains/domain1 ディレクトリ内に server1 や admin-server と同じ階層の新しいディレクトリが作成されます。インスタンスには、作成時に任意の名前をつけることができます。

デフォルトの設定では、アプリケーションサーバーインスタンスのイベントログファイルの内容はデスクトップに表示されます。これらのイベントログファイルには、実際には次の場所に保存されています。

アプリケーションサーバーインスタンスのログ格納ディレクトリを開き、`server.log` ファイルを開きます。

```
<domain_config_dir>¥domain1¥server1¥logs¥server.log
```

ログファイル



アプリケーションサーバーインスタンスのログファイルを開くと、デスクトップ上のイベントログウィンドウの表示内容と同じメッセージが表示されます。

ヒント **イベントログの属性**: イベントログファイルに記録されるログレコードの構造について説明します。それぞれのレコードには、重要度のあとに固有のメッセージ ID が記録されます。メッセージ ID は、サブシステム ID の略号とメッセージ番号から構成されています。この情報は、参考文献で詳細な情報を調べるときに便利です。

ヒント **HTTP サーバーのアクセスログ**: 内蔵の HTTP サーバーのアクセスログは、デフォルトの設定ではサーバーのイベントログと同じディレクトリに保存されます。このファイルに記録される情報は、HTTP サーバーに関する問題の解決に役立ちます。また、アプリケーションサーバーのインスタンスに送られる HTTP 要求のアクティビティを追跡する場合にも便利です。HTTP アクセスログファイルは管理サーバーによっても管理されていますが、通常の開発作業ではこのファイルを開く必要はほとんどありません。

アプリケーションサーバーインスタンスの HTTP サーバーへのアクセス

Web ブラウザからアプリケーションサーバーインスタンスの HTTP サーバーリスナーにアクセスすることで、インスタンスが正常に起動したかどうかを簡単に確認できます。アプリケーションサーバーのインスタンスを新規作成したときは、そのアプリケーションサーバーが起動したことをこの方法で素早く確認できます。

ブラウザで次のディレクトリに移動します。

```
http://<host name>:<port number>
```

<port number> は、インストール時に指定した HTTP サーバーのポート番号を意味します。HTTP サーバーのデフォルトのポート番号は 80 ですが、これはインストール時に使っていたポートによって異なります。

ヒント HTTP サーバーのポート番号がわからないときは、アプリケーションサーバーインスタンスの設定ファイルを調べます。

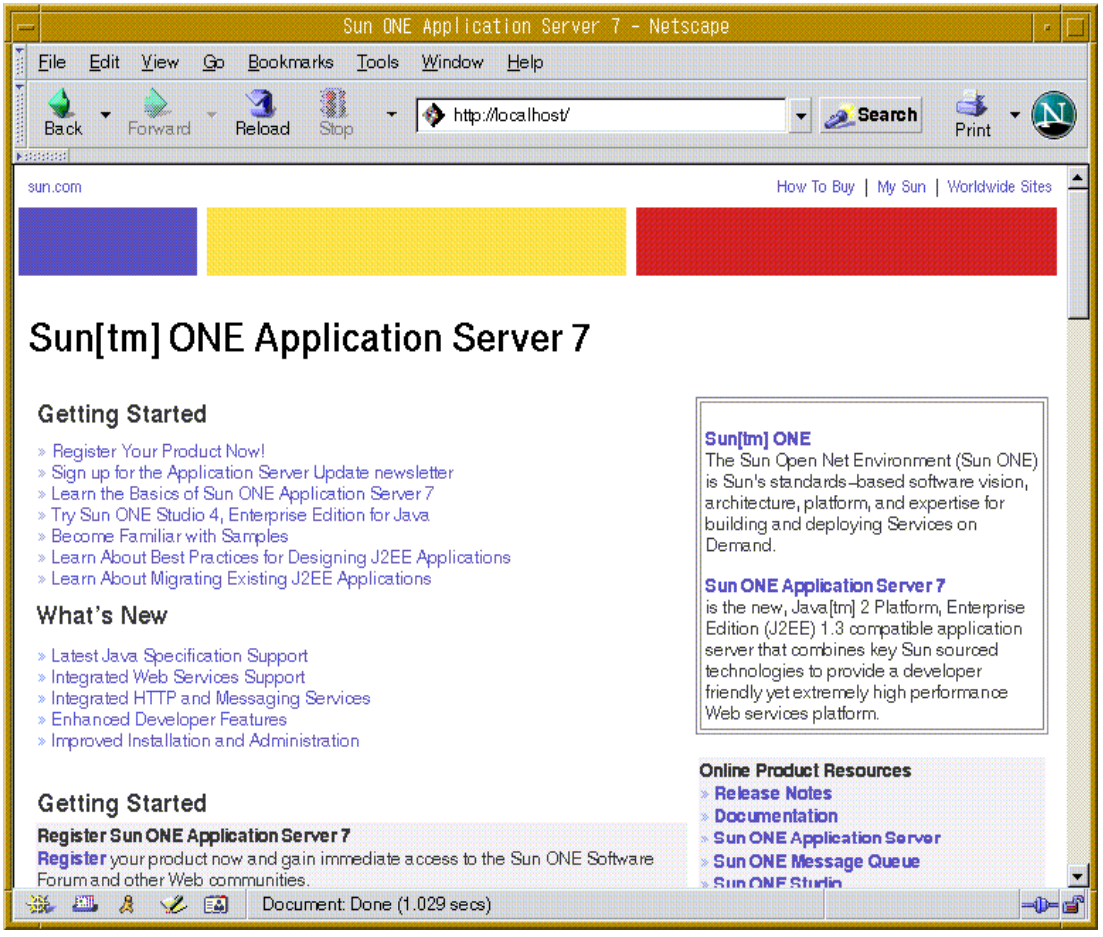
1. <domain_config_dir>/domain1/server1/config/ ディレクトリに移動して、server.xml ファイルをエディタで開きます。
2. 次の項目を確認します。

```
http-listener id="http-listener-1" address="0.0.0.0"
port="80"...
```

この例では、使用中の HTTP ポートの番号は 80 です。

アプリケーションサーバーのインスタンスが正常に稼動している状態では、HTTP サーバーのデフォルトのトップページがブラウザに表示されます。

HTTP サーバーのトップページ



ヒント

HTTP サーバーのトップページ: HTTP サーバーのトップページは、`index.html` という HTML ファイルです。このファイルは、アプリケーションサーバーインスタンスのデフォルトのドキュメントディレクトリに保存されています。アプリケーションサーバーインスタンスのデフォルトのドキュメントルートは、そのインスタンスの `server.xml` 設定ファイルに設定されています。インストールが完了すると、インスタンスのドキュメントルートである `server1` は `<domain_config_dir>/domain1/server1/docroot/` に設定されます。トップページはこの場所にあります。

アプリケーションサーバーのインスタンスは HTTP サーバーとしても十分に機能するので、静的なコンテンツだけでなく CGI 実行可能ファイルやサーバー側のインクルードファイルから生成される動的なコンテンツに対応するように設定できます。詳細は、『Sun ONE Application Server 管理者ガイド』を参照してください。

アプリケーションサーバーを完全に停止するには、63 ページの「Windows のプログラムグループの使用」に記載されている手順に従って Windows プログラムグループから「Sun ONE Application Server」メニューにアクセスします。

次の項では、Windows サービスを使ってアプリケーションサーバーを起動する方法について説明します。

管理コンソールの使用

管理サーバーが稼動している状態であれば、Web ベースの管理コンソールからもアプリケーションサーバーインスタンスを起動および停止できます。

1. 管理コンソールを起動します。
   - Windows 環境では、「スタート」-> 「プログラム」-> 「Sun Microsystems」-> 「Sun ONE Application Server 7」-> 「Start Admin Console」を選ぶと、Web ベースの管理コンソールを簡単に起動できます。

     「Start Admin Console」を選ぶとデフォルトのブラウザが起動し、製品のインストール時に設定した場所に格納されている管理サーバーのコンソールが表示されます。
   - UNIX 環境では、ブラウザを開いて管理サーバーのコンソールアプリケーションが格納されている場所を指定します。

管理サーバーのデフォルトのポート番号はインストール時に 4848 に設定されます。このポートが使用中の場合、あるいはポート番号に別の値を指定した場合は、ポート番号も指定します。

次に例を示します。

```
http://localhost:4848
```

2. インストール時に設定した管理ユーザー名とパスワードを入力して管理コンソールにサインインします。

ヒント　**管理サーバーのポート番号がわからない場合**：管理サーバーの HTTP サーバーポート番号がわからない場合は、管理サーバーの設定ファイルを調べます。

1. `<domain_config_dir>/domain1/admin-server/config/` ディレクトリに移動して、`server.xml` ファイルをエディタで開きます。
2. `http-listener id="http-listener-1" address="0.0.0.0" port="4848"...` という項目を確認します。

この例では、使用中の HTTP ポートの番号は 4848 です。

ヒント　**ユーザー名またはパスワードがわからない場合**：インストール時に指定した管理ユーザー名がわからない場合は、`admin` というユーザー名で試してください。これは、インストール時にサーバーの設定ダイアログに表示されるデフォルトのユーザー名です。

それでもユーザー名を特定できないときは、次のファイルの内容を確認します。

```
<domain_config_dir>/domain1/admin-server/config/admpw
```

このファイルには管理者のユーザー名とパスワードが記録されています。ただし、パスワードは暗号化されています。

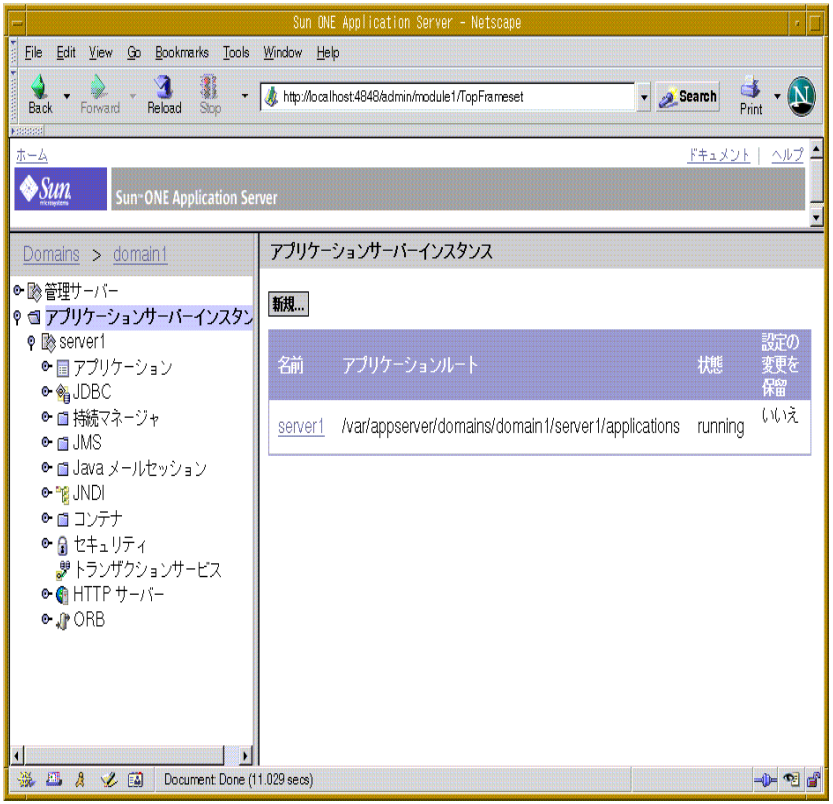
管理ユーザーのパスワードがわからない場合は、管理サーバーをアンインストールしてから、インストールし直す必要があります。

管理ユーザーのパスワードがわからないときは、`asadmin` のサブコマンド `delete-domain` を使って管理ドメインを削除し、新しい管理ユーザーパスワードを設定して新しいドメインを作成できます。

ヒント　**ポートにアクセスできない場合**：管理サーバーの管理コンソールアプリケーションに接続しようとしても拒否される場合は、管理サーバーが稼動していない可能性があります。サーバーが稼動していない原因を調べるには、この章の最初に戻って起動手順と管理サーバーログファイルの内容を確認してください。

正しく認証されると、管理コンソールの最初の画面が表示されます。

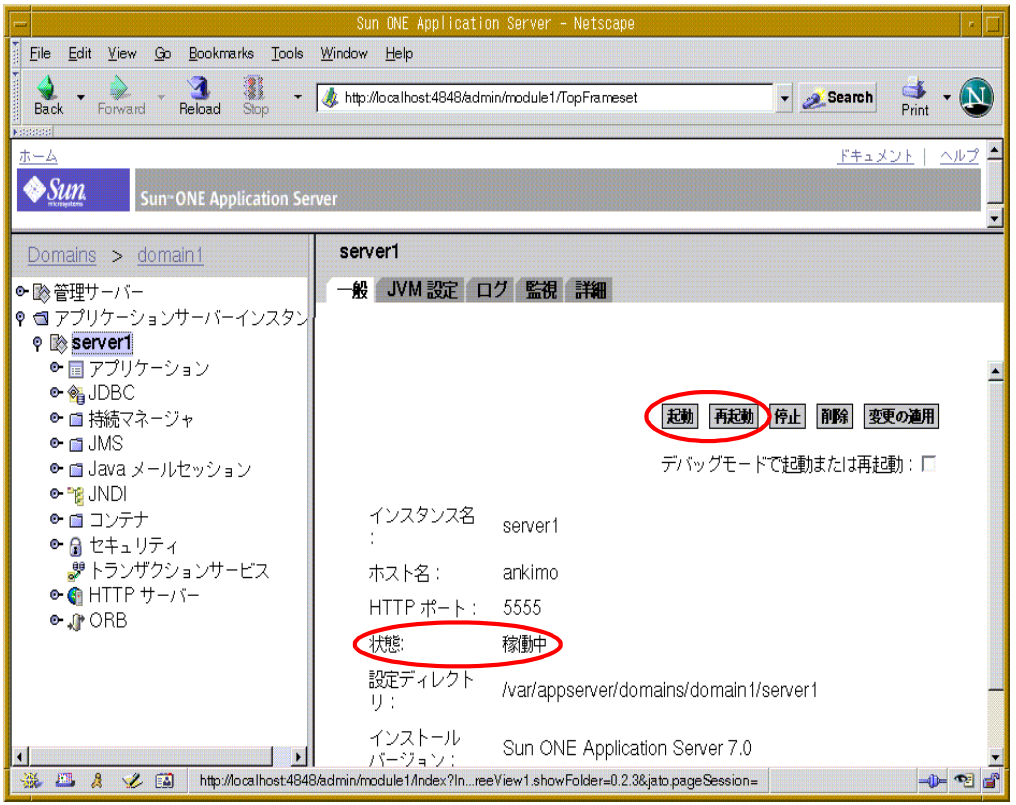
管理コンソール



注 **管理サーバーのノードとアプリケーションサーバーインスタンスのノード**: 管理コンソールには、管理サーバーのノード (Admin Server) と `server1` というアプリケーションサーバーインスタンスのノードが表示されます。この 2 つのノードは、`<domain_config_dir>/domain1/` ディレクトリの下の 2 つのディレクトリに相当します。管理コンソールの主な目的は、各インスタンスの `config/server.xml` ファイルに記録されているサーバーの設定情報を簡単に管理できるようにすることです。

3. `server1` ノードを選ぶと、起動および停止機能を使用できます。

Server 1 のノード



アプリケーションサーバーインスタンスの状態を確認します。状態は「稼動中」または「停止中」のいずれかです。

4. アプリケーションサーバーインスタンスの状態に応じて、「起動」または「停止」をクリックすると、そのインスタンスを起動または停止できます。

サンプルアプリケーションを配備して実行する前に、第 5 章「サンプルアプリケーションの操作」に記載されているサンプルアプリケーションに関する簡単な説明を参照してください。

Windows のプログラムグループの使用

Windows 環境では、Windows のプログラムグループからアプリケーションサーバーを簡単に起動できます。

1. 「スタート」-> 「プログラム」-> 「Sun Microsystems」-> 「 Sun ONE Application Server 7」-> 「Start Application Server」を選びます。

   「Start Application Server」を選ぶと、起動プロセスの状態を示す最初のコマンドウィンドウが表示されます。

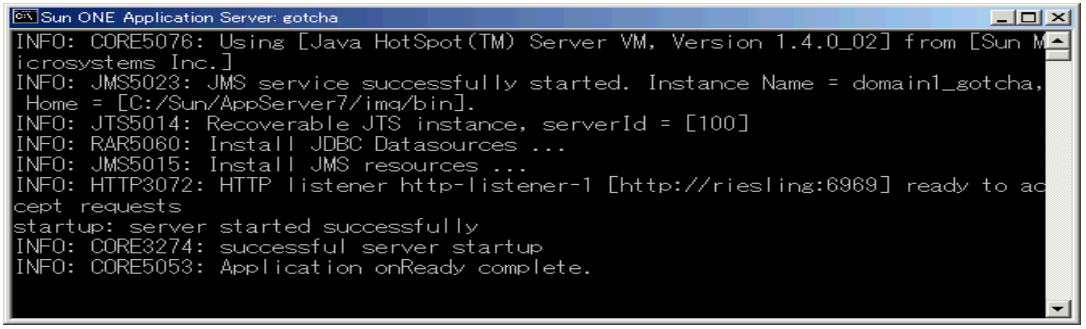
   次にもう 1 つのコマンドウィンドウが表示されます。この読み取り専用のコマンドウィンドウには、アプリケーションサーバーインスタンスのイベントログファイルの内容が表示されます。関連するアプリケーションサーバーインスタンスの実行中は、常にデスクトップに表示されます。最初に設定したサーバーを起動すると、別のコマンドウィンドウが表示され、そこにイベントメッセージが表示されます。サーバーインスタンスが問題なく起動すれば、数秒後に起動が成功したことを知らせるメッセージが表示されます。

   ```
   INFO: CORE3274: successful server startup
   ```

注 **起動メッセージを表示する別のウィンドウが表示されない場合**: 一部の Windows 2000 環境には Windows の `net` コマンドが含まれていないので、操作環境でこのコマンドを使用できることを確認してください。コマンドプロンプトから `net` コマンドを実行できない場合は、修正が必要となります。詳しくは 37 ページの「環境の設定」を参照してください。

   アプリケーションサーバーインスタンスのコマンドウィンドウには起動の状態が表示されます。下の図を参照してください。ウィンドウのタイトルは `server1` となっています。これはアプリケーションサーバーのインスタンス名です。

アプリケーションサーバーインスタンスのコマンドウィンドウ



注

実際に実行される処理：Windows のプログラムグループから「Start Application Server」を選ぶと、コマンド行インタフェースレベルでは `asadmin` コマンドの `start-appserv` サブコマンドが実行されます。このサブコマンドは、管理サーバーと最初に設定したアプリケーションサーバーインスタンスの両方を起動します。後ほどアプリケーションサーバーのインスタンスを追加することになりますが、その状態で「Start Application Server」を実行すると、新たに追加したインスタンスも起動します。

最初に設定したサーバーを起動すると、デスクトップ上のコマンドウィンドウにはイベントログの内容が表示されます。アプリケーションサーバーインスタンスのイベントログの内容をデスクトップに表示するかどうかは、インスタンスごとに設定できます。この機能を無効にする方法については後述します。

2. アプリケーションサーバーの起動中に、「Windows タスクマネージャ」を開いて起動プロセスが完了することを確認します。

   アプリケーションサーバーの通常操作ではこの作業を行う必要はありません。これは、アプリケーションサーバーの起動時に実際に行われている処理を確認する際に必要なだけです。

   「Windows タスクマネージャ」を起動する手順は次のとおりです。

   a. タスクバーの何も表示されていない場所を右クリックし、「タスクマネージャ」を選びます。

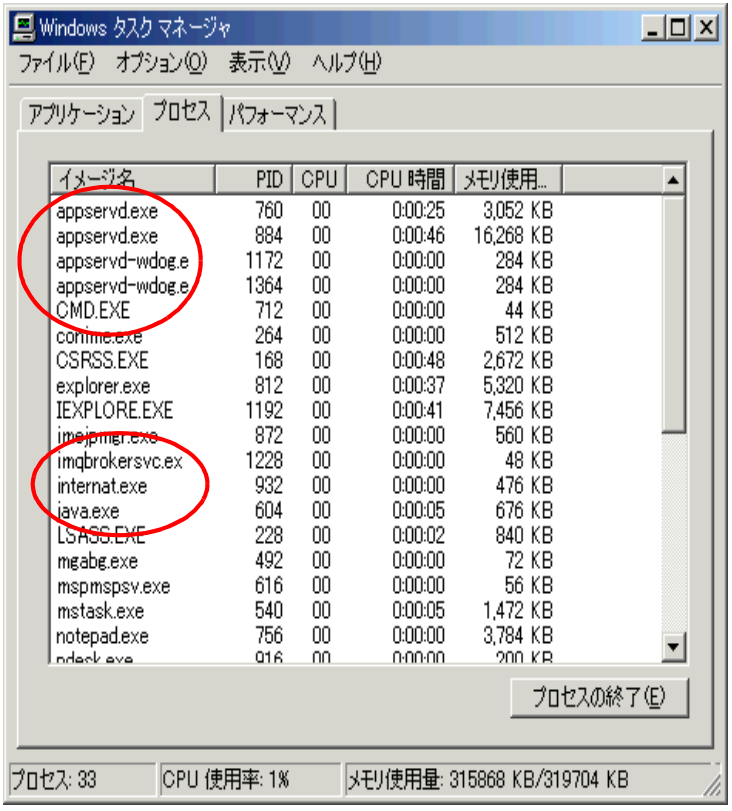
      Ctrl、Alt、Delete の各キーを同時に押して「タスクマネージャ」を選ぶか、Ctrl、Shift、Esc の各キーを同時に押しても、「Windows タスクマネージャ」を起動できます。

   b. 「Windows タスクマネージャ」が起動したら、「プロセス」タブをクリックしてシステムで実行中のすべてのプロセスを表示します。

c. 「イメージ名」列のタイトルをクリックすると、プロセスがアルファベット順にソートされます。

次の図では、最初に設定したアプリケーションサーバーの環境が 6 つのプロセスから構成されていることがわかります。

アプリケーションサーバーのプロセス

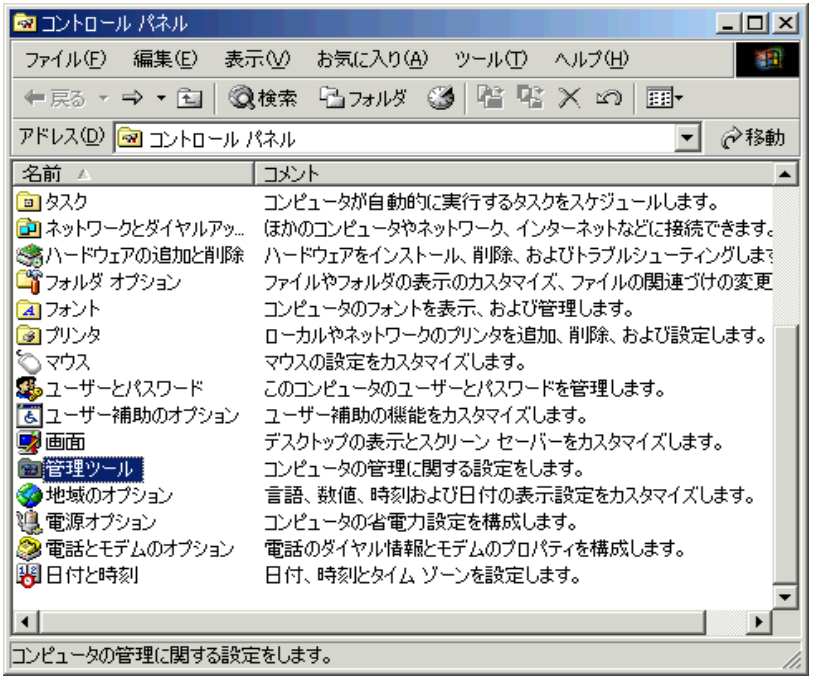


Windows サービスの使用

アプリケーションサーバーをインストールすると、管理サーバー、および最初に設定した server1 アプリケーションサーバーインスタンスの起動と停止を制御する複数の Windows サービスが定義されます。この項では、Windows サービスを使ってアプリケーションサーバーのプロセスを制御する方法について説明します。

1. 「スタート」-> 「設定」-> 「コントロールパネル」を選びます。
2. 「コントロールパネル」が表示されたら、「管理ツール」をダブルクリックします。

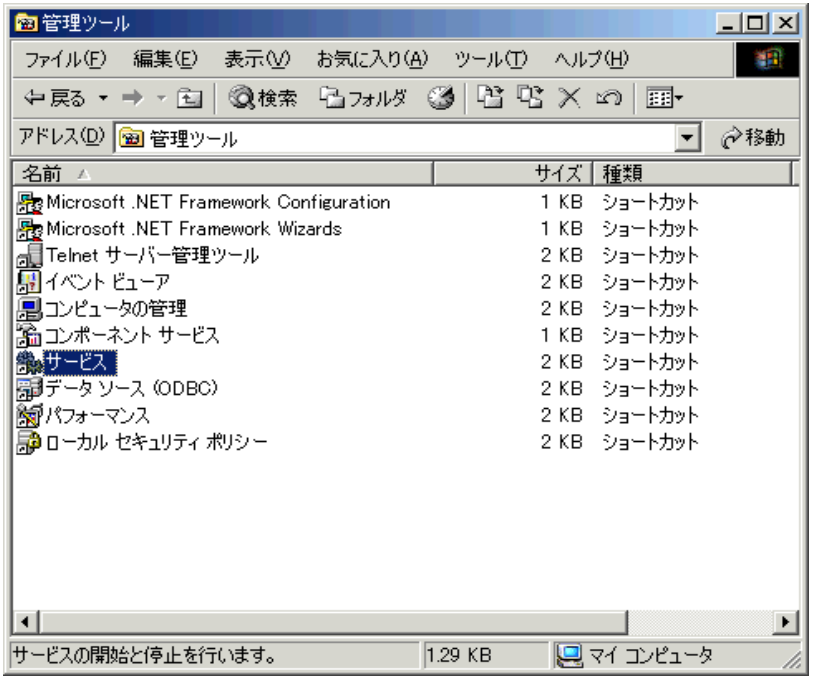
コントロールパネル



3. 「サービス」をダブルクリックして、システムにインストールされているサービスを表示します。

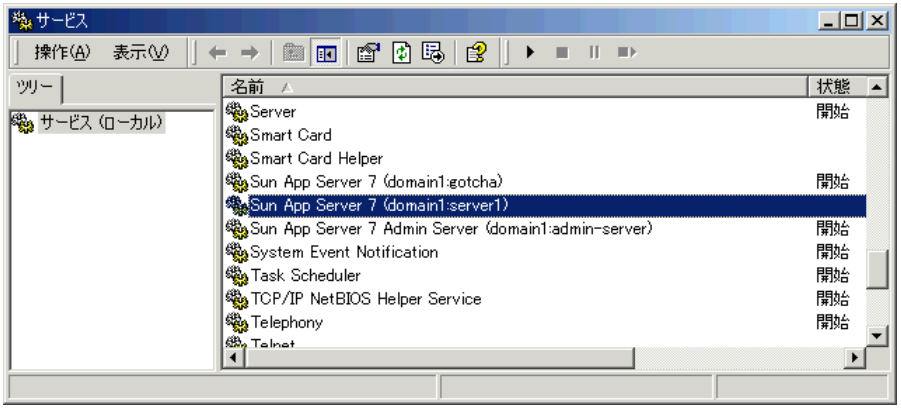
   システムに定義されているすべてのサービスが表示されます。

「管理ツール」ダイアログボックス



4. 「Sun Application Server」が表示されるまでスクロールします。

「サービス」ウィンドウ



管理サーバーの Windows サービスと、最初に設定したアプリケーションサーバーインスタンス `server1` の Windows サービスは異なります。前の練習で起動した管理サーバーとアプリケーションサーバーインスタンスが稼動中であれば、「状態」列に表示される各サービスの状態は「開始」となります。

5. 64 ページの手順 a で説明した方法で「Windows タスクマネージャ」を起動し、システム上で実行中のプロセスを確認してみましょう。
6. 「サービス」ウィンドウで管理サーバーとして「Sun Application Server 7」を選び、右クリックして「停止」を選びます。

   関連するプロセスが終了することを「Windows タスクマネージャ」で確認します。
7. アプリケーションサーバーインスタンス `server1` についても同じ手順を繰り返します。

   アプリケーションサーバーインスタンス `server1` を停止すると、デスクトップにイベントログウィンドウが表示されなくなります。
8. 管理サーバーとアプリケーションサーバーのインスタンスを停止し、Windows サービスを使って起動し直します。サービスを右クリックし、今度は「停止」ではなく「開始」を選びます。

   「Windows タスクマネージャ」を起動しておけば、関連するプロセスが開始されることを確認できます。

   アプリケーションサーバーインスタンス `server1` のサービスを開始すると、そのインスタンスのイベントログウィンドウがデスクトップに表示されます。

ヒント **Windows サービスの作成** : アプリケーションサーバーのインスタンスを作成または削除すると、それに対応する Windows サービスも作成または削除されます。作成直後の Windows サービスの「スタートアップの種類」は、「手動」に設定されています。つまり、Windows を再起動したときにそのサービスは自動的に起動されません。必要に応じて「スタートアップの種類」を「自動」に変更してください。こうすることにより、Windows を再起動したときにアプリケーションサーバーに関連するプロセスが自動的に開始されます。変更手順は次のとおりです。

1. サービスを選んで右クリックし、「プロパティ」を選びます。
2. 「スタートアップの種類」で「自動」を選び、「OK」をクリックします。

次に、コマンド行インタフェース `asadmin` を使ってアプリケーションサーバーを起動および停止します。

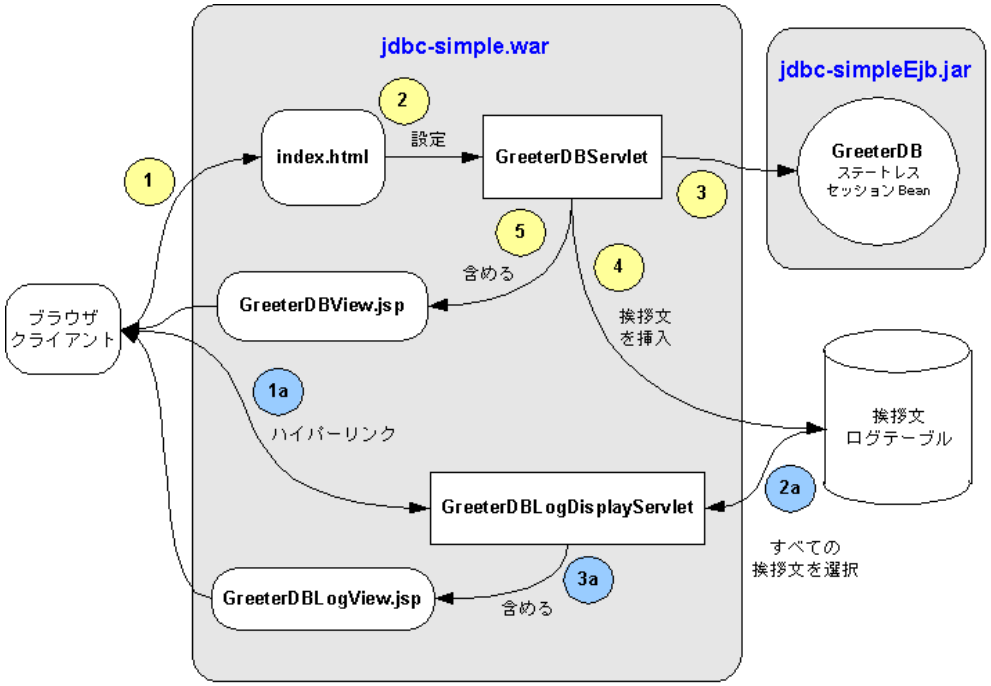
第 5 章

サンプルアプリケーションの操作

このマニュアルで使うサンプルアプリケーションは、Web アプリケーションモジュールと、ステートレスセッション Bean を 1 つ含む EJB モジュールから構成されています。Web モジュールと EJB モジュールは、どちらも Enterprise Application Archive (EAR) ファイルに格納されています。よりシンプルな「Hello World」アプリケーションに似ていますが、このサンプルアプリケーションが出力する挨拶文は入力内容と日時によって異なります。すべての挨拶文はデータベーステーブルに保存されているので、過去に作ったすべての挨拶文をリスト表示することもできます。

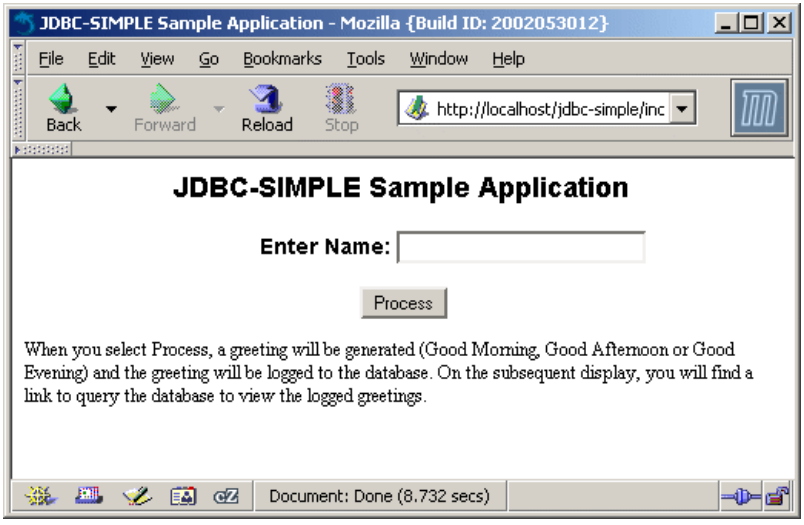
次の図では、挨拶文の表示に関するアプリケーション上のやり取りは番号 (1 ～ 5) で示されています。挨拶文ログの表示に関わるやり取りは文字つきの番号 (1a ～ 3a) で示されています。

サンプルアプリケーション



アプリケーションを起動すると、最初の Web ページが表示されます。このページでは、次の挨拶文ページに表示される名前を入力します。

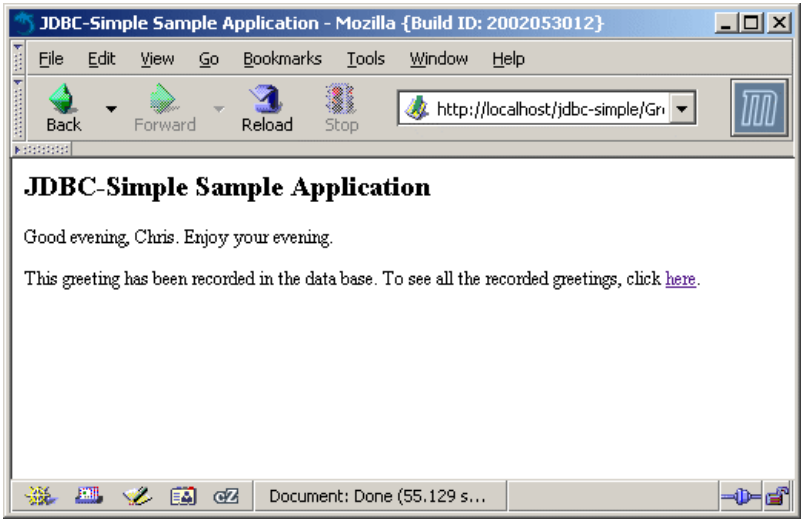
最初の Web ページ



名前を入力して「Process」ボタンをクリックすると、挨拶文を示す次のページが表示されます。この裏では、`GreeterDBServlet` がステートレスセッション Bean にアクセスし、「morning」、「afternoon」、「evening」から適切な項目を決定しています。

`GreeterDBServlet` は、サーブレットに渡す要求オブジェクトに適切な挨拶文を設定し、結果を表示するように JSP を設定します。

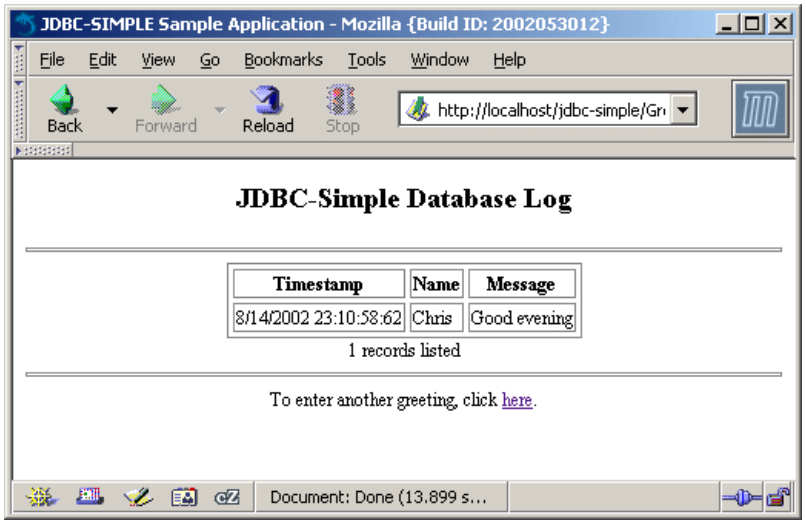
表示された挨拶文



挨拶文が表示されます。

GreeterDBLogDisplayServlet は、過去に作成したすべての挨拶文をリスト表示する場合は、挨拶文のデータベーステーブルですべての要素を選択し、結果を表示するように JSP を設定します。

データベースログの内容



第 6 章「データベースとの接続の設定」では、サンプルアプリケーションを配備し実行できるように、アプリケーションサーバーのデータベースを設定する方法について説明します。

第 6 章

データベースとの接続の設定

挨拶文をデータベースに記録するときに、サンプルアプリケーションでは Java[tm] Database Connectivity (JDBC[tm]) API を使っています。この章では、サンプルアプリケーションの配備と実行の前に、アプリケーションサーバー環境に JDBC 関連の設定を定義する方法について説明します。

JDBC の設定には、JDBC ドライバの設定、JDBC 接続プールの設定、およびアプリケーションが使用する JDBC リソースの登録が含まれます。このマニュアルで使うサンプルアプリケーションのデフォルトの設定では、データベースとして PointBase Server が使われます。PointBase Server は、どの Sun ONE Application Server 7 製品にも含まれています。ただし、Solaris 9 のインストールと同時にインストールしたアプリケーションサーバーには PointBase Server は含まれません。アプリケーションサーバーのインストールディレクトリに `<appserver_install_dir>/pointbase/` というディレクトリが存在すれば、PointBase Server はすでにインストールされているので、次に説明する手順に従って JDBC 接続プールとリソースオブジェクトを定義できます。

アプリケーションサーバーのインストールに PointBase Server が含まれていないときは、JDBC 接続プールとリソースの設定手順に進む前に、90 ページの「PointBase Server のインストールと設定」に記載されている手順に従って PointBase Server 環境を設定する必要があります。

この章では次の項目について説明します。

- JDBC ドライバの設定
- JDBC 接続プールの定義
- JDBC リソースの定義
- PointBase Server データベースの起動

JDBC ドライバの設定

JDBC を利用するアプリケーションを使うには、事前にアプリケーションサーバーのクラスパスに JDBC ドライバを設定しておく必要があります。

アプリケーションサーバーと PointBase を同時にインストールした場合や、PointBase のインストール時に設定が完了している場合など、アプリケーションサーバーインスタンスの環境にすでに PointBase Type 4 JDBC ドライバが設定されているのであれば、JDBC ドライバの設定手順を実行する必要はありません。

ヒント

クラスパスへの JDBC ドライバの追加 : 管理コンソールを使えば、アプリケーションサーバーのクラスパスに JDBC ドライバを簡単に追加できます。

1. コンソールを開いてサーバーインスタンスのノードを選び、「JVM 設定」タブの「パス設定」タブを開きます。
2. 「パス設定」タブでは、適切な JDBC ドライバに合わせて「クラスパスのサフィックス」の設定を `.zip` ファイルまたは `.jar` ファイルに変更します。Type 2 ドライバを使うときは、適切なネイティブライブラリに合わせて「ネイティブライブラリのパスサフィックス」の設定を変更します。
3. 変更を保存し、サーバーインスタンスの「一般」タブに戻ります。
4. 「変更の適用」をクリックし、アプリケーションサーバーを再起動して変更を有効にします。

Type 4 ドライバのアーカイブを次のディレクトリにドロップしても設定できます。

```
<domain_config_dir>/domain1/server1/lib/
```

インスタンスの `lib` ディレクトリに含まれるすべての Java ライブラリは、サーバーインスタンスを再起動したときにサーバーのクラスパスの最後に自動的に追加されます。

ヒント

設定済みの PointBase データベース : サンプルアプリケーションがどのように PointBase を利用しているかについては、サンプルアプリケーションのマニュアルを参照してください。

JDBC 接続プールの定義

サンプルアプリケーションを実行する前に、適切な JDBC 接続プールを定義する必要があります。この接続プールは、PointBase データベースサーバーと、サンプルアプリケーションで作成される JDBC 参照を接続プールの定義に関連付ける JDBC リソースにマッピングされます。

注

サーバーインスタンスの設定 : J2EE アプリケーションが利用するリソース、およびその他のサーバーの設定は、`server.xml` というファイルに記録されます。このファイルは、次のディレクトリに保存されています。

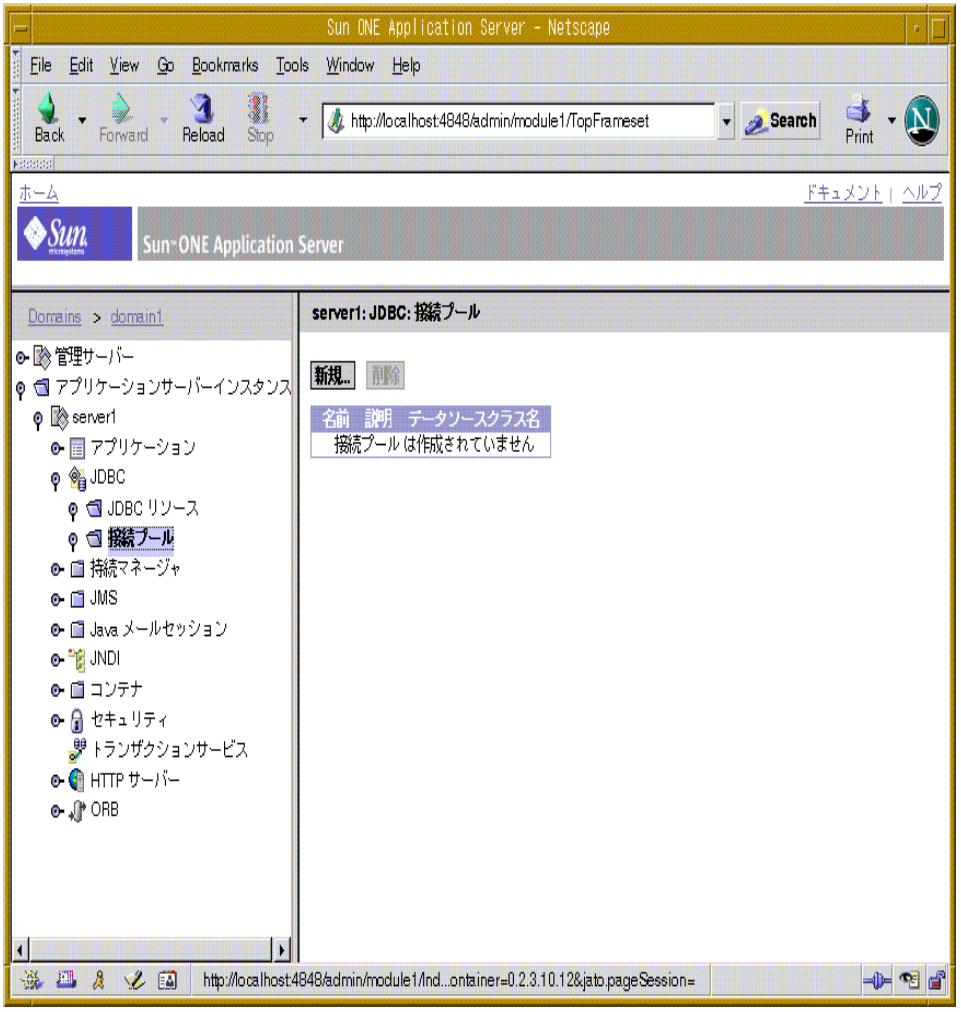
```
<domain_config_dir>/domain1/server1/config/
```

このファイルの内容を手動で変更することもできますが、管理コンソールまたは `asadmin` コマンド行インタフェースを通じて別のユーザーがファイルの内容を更新している可能性もあるので注意してください。`server.xml` ファイルの内容を手動で変更したときは、アプリケーションサーバーインスタンスを再起動して変更を適用する必要があります。

---

1. Web ブラウザから管理コンソールにアクセスします。アプリケーションサーバーが稼動していないときは、起動する必要があります。
2. アプリケーションサーバーインスタンス `server1` を展開して JDBC ノードを表示します。接続プールノードをクリックし、「新規」をクリックして新しい接続プールを定義します。

データソースのノード

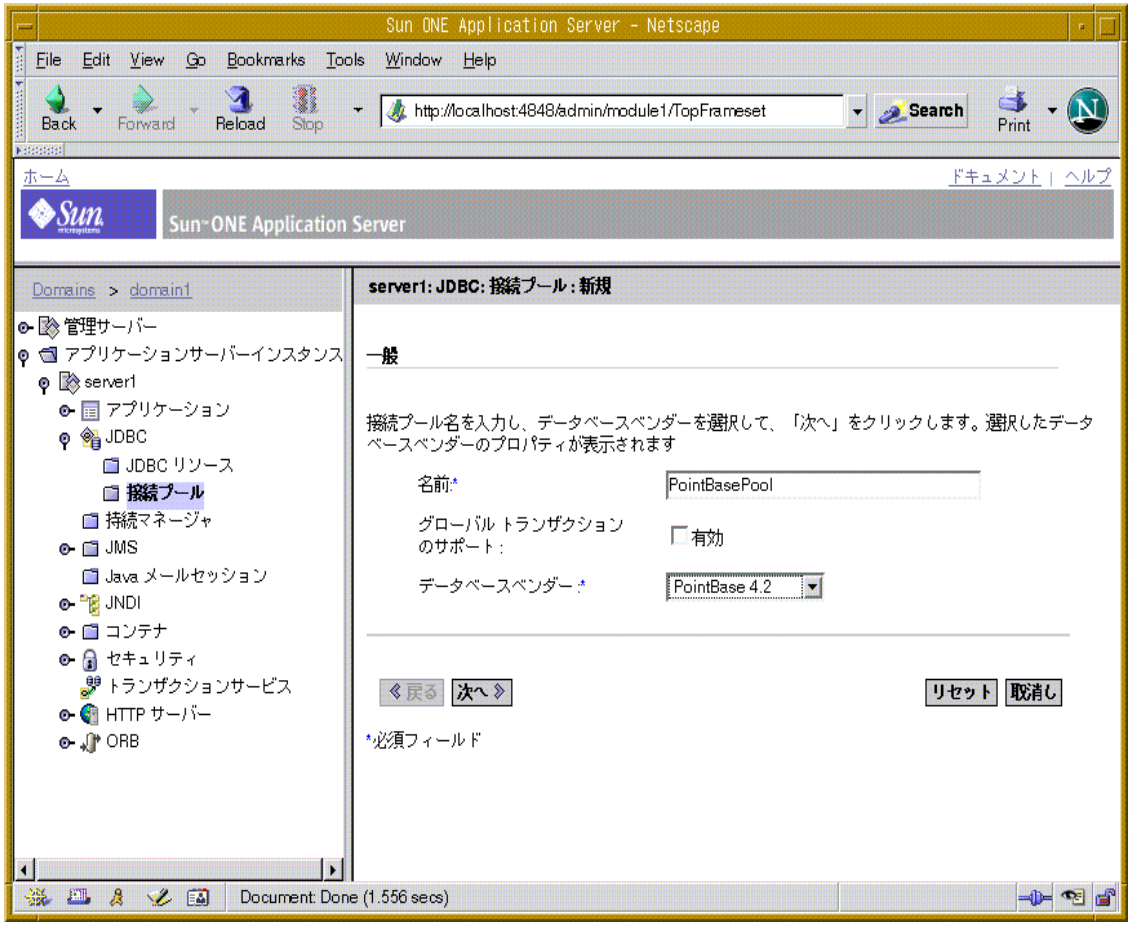


3. 接続プール新規作成ウィザードの最初の画面が表示されるので、データベースのプロパティ設定を次の表のとおりに入力し、「次へ」ボタンをクリックします。

データベースのプロパティ設定

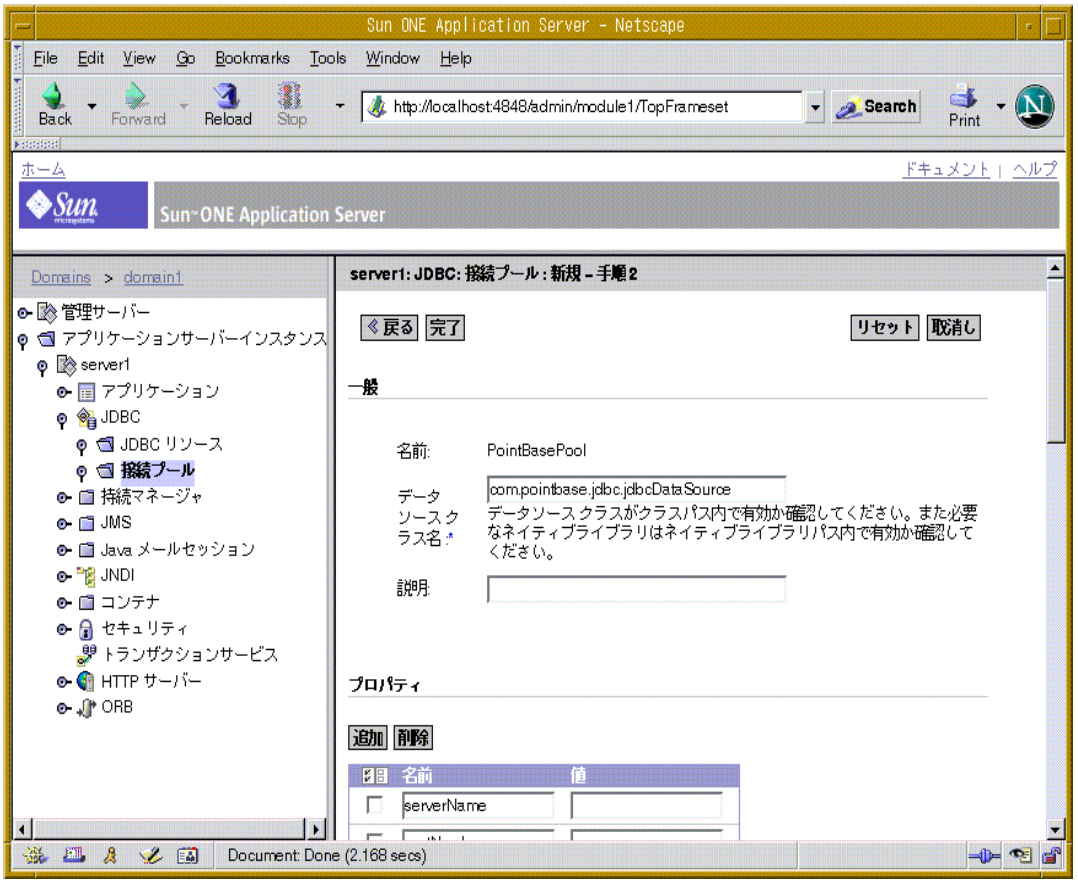
フィールド	値
Name	`PointBasePool`
Database vendor	`Pointbase 4.2` を選択

データベース接続プールの設定



次の画面が表示されます。

接続プールの設定



4. 「プロパティ」が表示されるまでスクロールし、データベースのプロパティ設定を次の表のとおりに入力します。

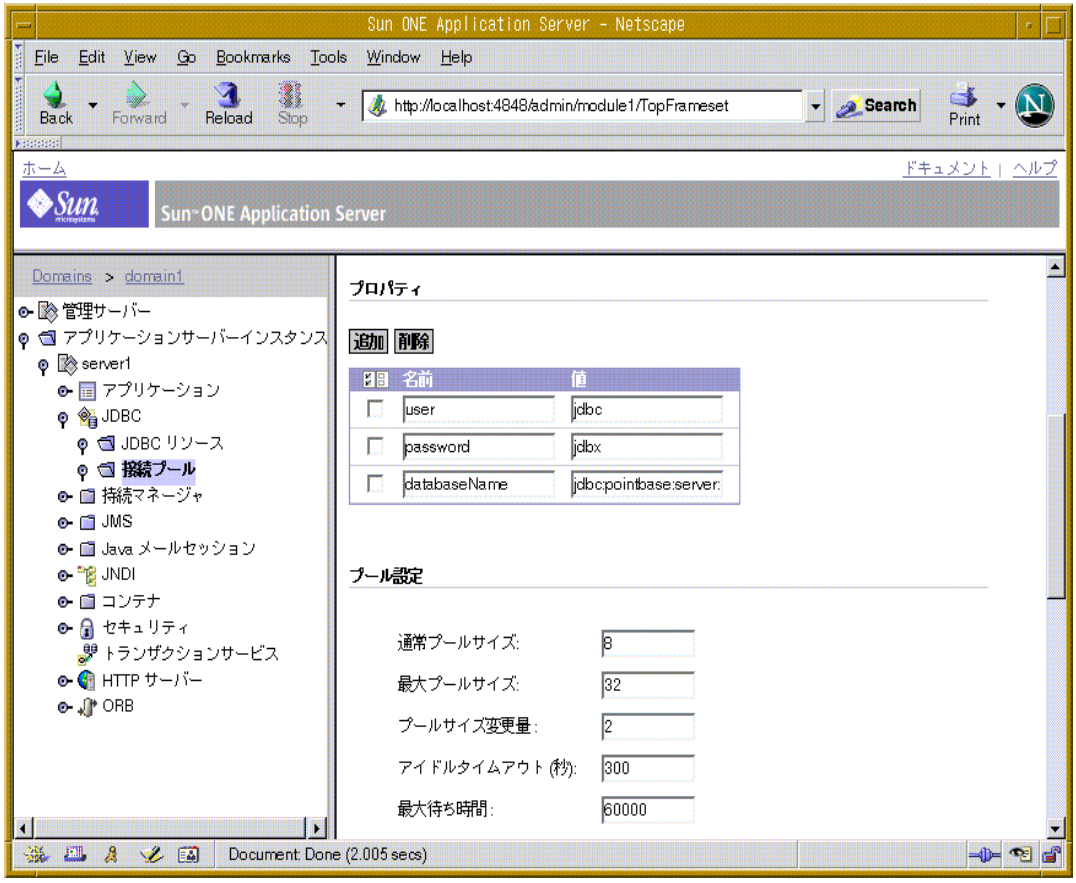
データベースのプロパティ設定

プロパティ	値
user	`jdbc`
password	`jdbc`
dataBaseName	`jdbc:pointbase:server://localhost/sun-appserv-samples`

注 **接続プールのプロパティとその他のデータベースの使用** : PointBase Server は `dataBaseName` プロパティに対応していますが、このプロパティに対応していないデータベースもあります。たとえば、Oracle RDBMS では、JDBC ドライバが Oracle RDBMS に接続するときに `URL` プロパティを使って URL を指定しています。具体的にどのプロパティに対応しているかを確認するには、JDBC ドライバのマニュアルを参照してください。サンプルアプリケーションで Oracle JDBC ドライバを使う例については、「JDBC リソースの定義」に記載されている Oracle Type 4 JDBC ドライバの使用に関する注を参照してください。

これ以外の不要プロパティが表示されるときは、そのプロパティの左に表示されるチェックボックスを選んでから「削除」ボタンをクリックして削除します。

接続プールのプロパティ

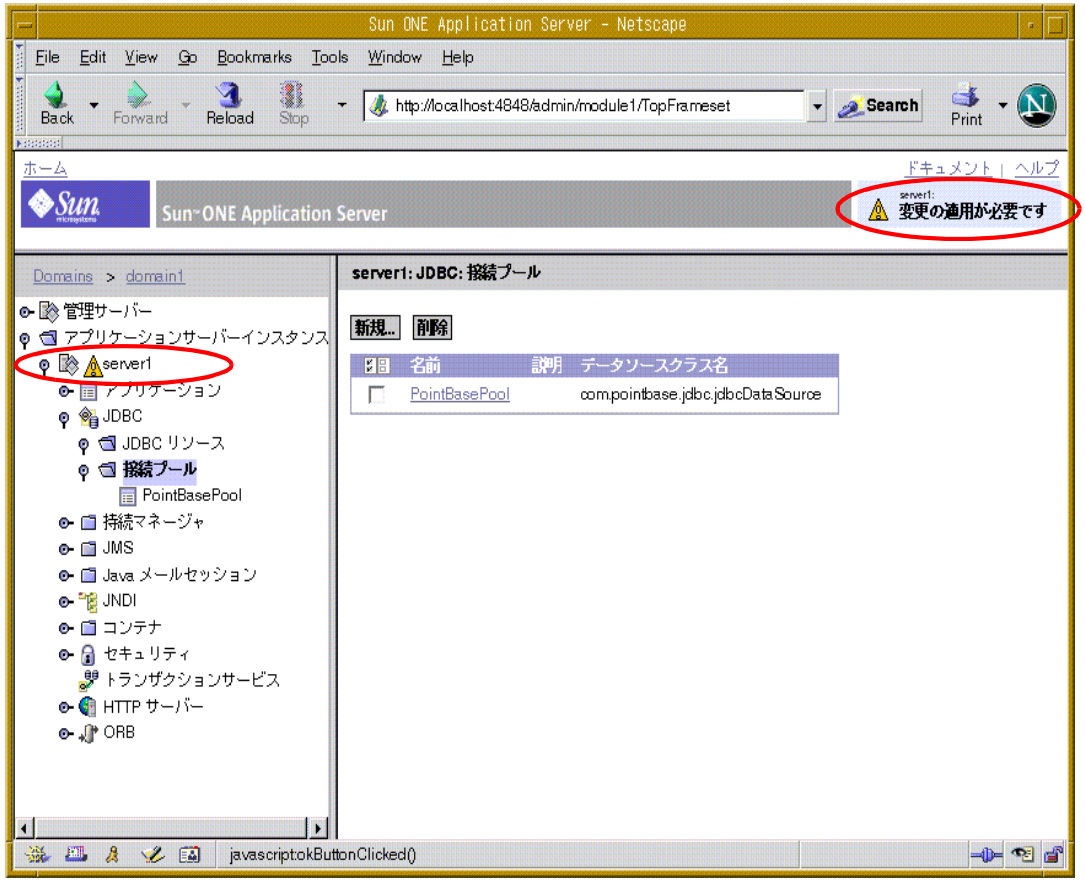


5. 「完了」ボタンをクリックしてウィザードを終了します。

   JDBC 接続プールウィザードを使ってサーバーの設定に変更を加えても、管理サーバーはアクティブな設定が変更されていないものとして認識します。このため、変更の適用が必要であることを示す警告アイコンが管理コンソールに表示されます。

   変更を適用する前に、次の手順に進んでサンプルアプリケーションの JDBC リソースを定義します。JDBC リソースを定義したら、接続プールと JDBC リソースの変更を同時に適用します。

変更の適用

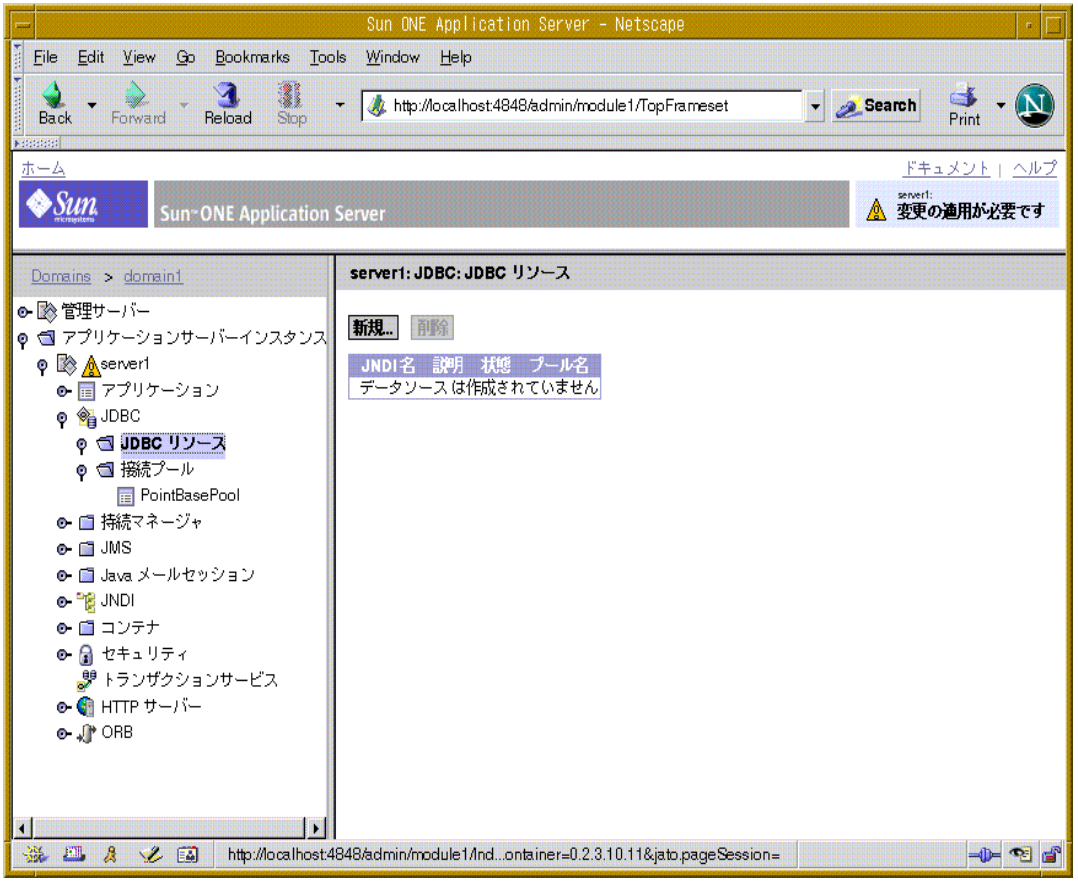


JDBC リソースの定義

JDBC 接続プールの定義を作成できたので、次に JDBC リソースを定義して接続プールのエントリと関連づけます。

1. JDBC のノードに含まれる JDBC Resources というノードを選び、「新規 ...」をクリックして新しいリソースエントリを定義します。

データソースのノード



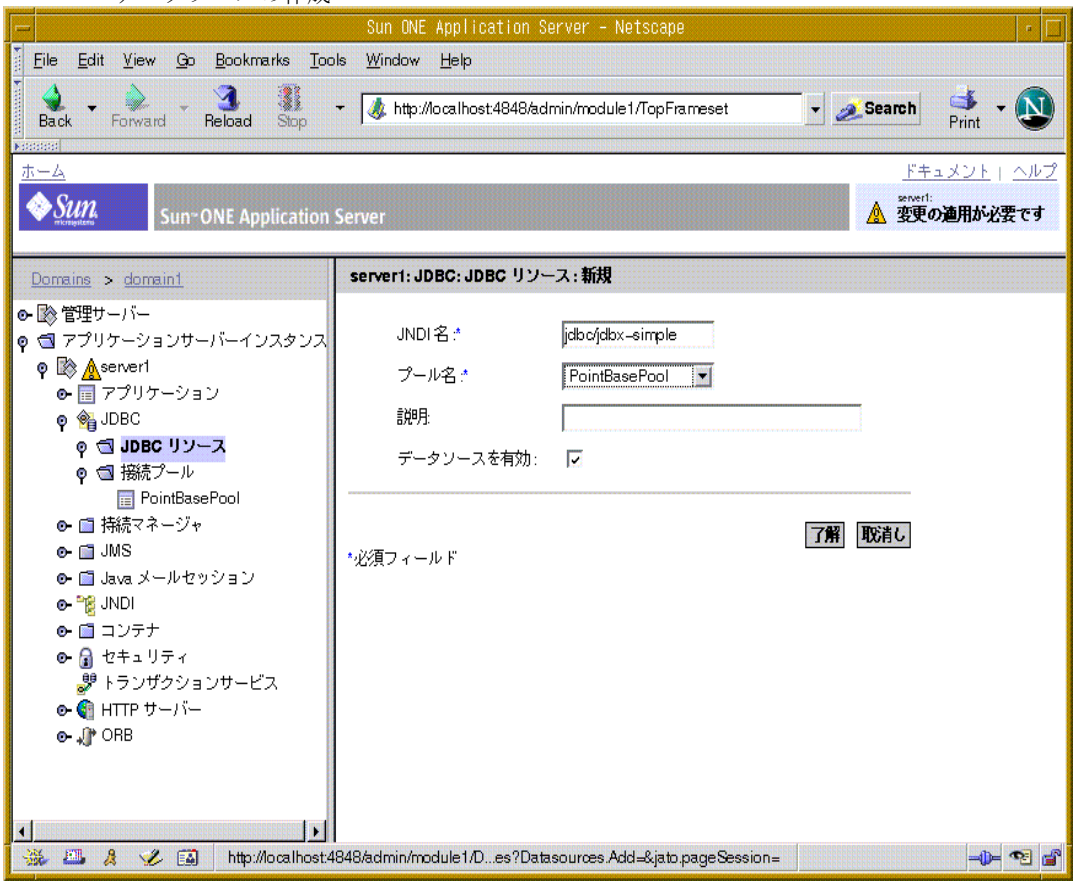
サンプル Web アプリケーションの `sun-web.xml` ファイルは JNDI 名 `jdbc/jdbc-simple` を参照しますが、この名前の JDBC リソースはまだサーバーの設定として保存されていないため、新しい JDBC リソースを定義する必要があります。

2. データベースのプロパティ設定を次の表のとおりに入力し、「了解」をクリックしてリソースを定義します。

データベースのプロパティ設定

フィールド	値
JNDI Name	`jdbc/jdbc-simple`
Pool Name	`PointBasePool` を選択

データソースの作成

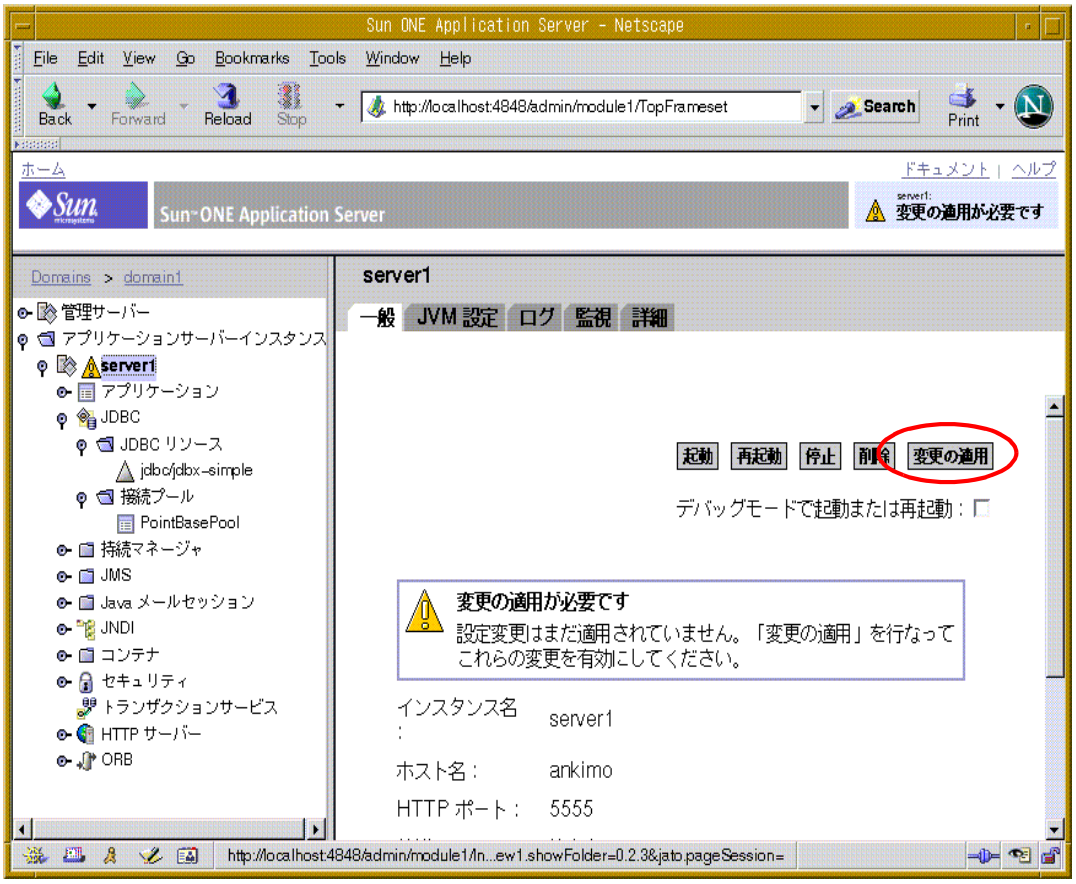


3. JDBC 接続プールとサンプルアプリケーションのリソースの両方を定義できました。次に、変更を適用してアプリケーションサーバーのインスタンスに認識させます。

   a. `server1` のノードを選びます。

b. 「変更の適用」をクリックし、JDBC 接続プールとリソースの変更を適用します。

変更の適用



変更が適用されたことを示すメッセージが表示されます。この場合、サーバーインスタンスを再起動する必要はありません。

注

変更の適用時に行われる処理 : JDBC 接続プールやリソースなどのオブジェクトを新たに定義したり、これらのオブジェクトに変更を加えたりすると、アプリケーションサーバーインスタンスの設定領域にあるバックアップディレクトリ内の `server.xml` ファイルに設定情報が保存されます。たとえば、次のような場所にあります。

```
<domain_config_dir>/domain1/server1/config/backup/server.xml
```

さらに変更を加えて保存すると、バックアップ領域内の `server.xml` ファイルも更新されます。

適用前の変更をアプリケーションサーバーのインスタンスに認識させるには、「変更の適用」をクリックしてバックアップ領域内の `server.xml` ファイルを次のような通常の設定ディレクトリに強制的にコピーする必要があります。

```
<domain_config_dir>/domain1/server1/config/server.xml
```

設定ディレクトリに適用した変更の一部はアプリケーションサーバーインスタンスによって直ちに認識されますが、それ以外の変更を認識させるにはサーバーインスタンスを再起動する必要があります。変更の適用にサーバーの再起動が必要な場合は、管理コンソールにメッセージが表示されます。

注 **アプリケーションで Oracle Type 4 JDBC ドライバを使う場合** : アプリケーションで Oracle Type 4 JDBC ドライバを使う場合、別の RDBMS も簡単に使えます。

1. 提供される SQL ファイルを使って Oracle RDBMS にテーブルを作成します。

```
<install_dir>/samples/jdbc/simple/src/sql/jdbc-simple-ora.sql
```

2. 次のいずれかの方法で Oracle Type 4 ドライバをサーバーのクラスパスに追加します。

a) アプリケーションサーバーインスタンスの次の `lib/` ディレクトリにドライバライブラリをドロップする

```
<domain_config_dir>/domain1/server1/lib/
```

このディレクトリに含まれるすべてのライブラリは、再起動時にサーバーのクラスパスの最後に自動的に追加されます。

b) アプリケーションサーバーインスタンスの 「JVM 設定」-> 「パス設定」を開き、「クラスパスのサフィックス」にドライバライブラリを追加します。

3. Oracle Type 4 ドライバ用の JDBC 接続プールオブジェクトを作成します。

```
Datasource Classname = oracle.jdbc.pool.OracleDataSource
```

接続プールのプロパティは次のとおりです。

URL: `jdbc:oracle:thin:@$ORACLE_SERVER:1521:$ORACLE_SID`

対象となるデータベースに合わせて `$ORACLE_SERVER` と `$ORACLE_SID` を適切な値に置き換える必要があります。次に例を示します。

```
jdbc:oracle:thin:@localhost:1521:orcl
```

User: $USER (対象となるデータベースに合わせて設定)

Password: $PASSWORD (対象となるデータベースに合わせて設定)

4. JDBC リソース `jdbc/jdbc-simple` を PointBasePool 接続プールから新たに定義した Oracle 接続プールにマッピングし直します。

5. 変更を適用し、アプリケーションサーバーを再起動します。

6. アプリケーションを実行し直します。

PointBase Server データベースの起動

サンプルアプリケーションを実行する前に、PointBase Server データベースを起動する必要があります。

アプリケーションサーバーのインストールを他のユーザーと共用している場合、または PointBase がインストールされている領域に対する書き込み権がシステムユーザー ID に割り当てられていない場合は (たとえば、UNIX システムのルートユーザーがアプリケーション サーバーをインストールしたときは、ルートユーザー以外の ID に書き込み権が与えられていないことがあります)、事前に構築されている PointBase データベースと PointBase Server の起動スクリプトの専用コピーを作成する必要があります。

PointBase のインストールを他のユーザーと共用していない場合は、86 ページの「PointBase Server データベースの起動」に進んでください。

データベースの専用コピーの作成

PointBase Server の専用コピーを作成する手順は次のとおりです。

1. 書き込み権のあるディレクトリの下に `pointbase/` という名前のディレクトリを作成します。
2. すでに内容が設定されている PointBase データベースのファイルをサンプルアプリケーションのインストールディレクトリから専用の PointBase ディレクトリにコピーします。

   内容が設定されているデータベースのファイルを PointBase の専用インストールにコピーするには、次のディレクトリに含まれるファイルをコピーします。

   `<appserver_install_dir>/samples/pointbase/databases/*`

   から次のディレクトリにコピーします。

   `<personal_pointbase_dir>/pointbase/databases/`

   これにより、`sun-appserv-samples$1.wal` および `sun-appserv-samples.dbn` というファイルが専用の PointBase ディレクトリにコピーされます。
3. 次のファイルを PointBase のインストールディレクトリから `<personal_pointbase_dir>/pointbase/` にコピーします。

ファイル	コメント
StartServer.sh または StartServer.bat	PointBase Server が待機するポートの番号を記録した起動スクリプト
pointbase.ini	データベースファイルの場所を指定した初期化ファイル

アプリケーションサーバーのインストール時に PointBase をインストールした場合は、これらのファイルは次のディレクトリに保存されています。

```
<appserver_install_dir>/pointbase/server/
```

PointBase の Evaluation バージョンをダウンロードしてインストールした場合は、これらのファイルは次のディレクトリに保存されています。

```
<pointbase_install_dir>/tools/server/
```

4. スクリプトファイル StartServer.sh または StartServer.bat を編集し、実行中の他の PointBase Server と重複しないポートの番号を設定します。

ヒント **デフォルト設定以外のポート番号の使用** : PointBase Server が待機するポートの番号をデフォルト値の 9092 から変更するときは、それに合わせて JDBC 接続プールのプロパティも変更する必要があります。この変更を加えないと、アプリケーションサーバーが PointBase Server の専用コピーにアクセスできません。

JDBC 接続プールに設定されている dataBaseName プロパティを次のように変更します。

```
jdbc:pointbase:server://localhost/sun-appserv-samples
```

から次のように変更します。

```
jdbc:pointbase:server://localhost:<port>/sun-appserv-samples
```

<port> は、StartServer スクリプトに設定されているポート番号です。

dataBaseName プロパティにポート番号を追加する手順は次のとおりです。

1. アプリケーションサーバーの管理コンソールを開いて PointBasePool のノードを選びます。
2. 下にスクロールして「プロパティ」ボタンをクリックし、接続プールのプロパティを編集します。
3. dataBaseName プロパティの値を適切なポート番号に変更し、「了解」をクリックして変更を保存します。
4. 変更を適用してアプリケーションサーバーのインスタンスを再起動し、接続プールに加えた変更を認識させます。

これで、専用ディレクトリから PointBase Server を起動できるようになります。

PointBase Server の起動

データベースサーバーは、次のいずれかの方法で簡単に起動できます。

- Windows 環境

  アプリケーションサーバーのインストール時に PointBase をインストールした場合

  「スタート」-> 「プログラム」-> 「Sun Microsystems」-> 「Sun ONE Application Server 7」-> 「Start PointBase」

  ダウンロードした PointBase を個別にインストールした場合

  <pointbase_install_dir>/tools/server/startserver.bat を実行

- UNIX 環境

  アプリケーションサーバーのインストール時に PointBase をインストールした場合

  <install_dir>/pointbase/server/StartServer.sh を実行

  ダウンロードした PointBase を個別にインストールした場合

  <pointbase_install_dir>/tools/server/startserver.sh を実行

このスクリプトを実行すると、コマンドウィンドウまたは UNIX の端末プロンプトに次のメッセージが表示されます。

```
Server started, listening on port 9092, display level: 0 ...
>
```

サーバーを停止するには、プロンプトに quit を入力します。

ヒント　**コンソールウィンドウの必要性** : UNIX 環境で PointBase Server をバックグラウンドで実行するには、StartServer.sh スクリプトを編集し、コマンド行の最後に /noconsole オプションを追加します。

```
.../java -classpath ./lib/pbserver42RE.jar
com.pointbase.net.netServer /port:9092 /noconsole
```

サーバーをバックグラウンドで実行するには、StartServer.sh コマンドの最後に & を追加します。

```
./StartServer.sh &
[1] 26418
root@canteloupe-{/opt/appserverBeta/pointbase/server}:
```

サーバーの起動と停止の詳細については、次のディレクトリに保存されている PointBase Server のマニュアルを参照してください。

<pointbase_install_dir>/docs/

サンプルアプリケーションを使用してみる場合は、第 7 章「サンプルアプリケーションの配備と実行」に進みます。

PointBase Server のインストールと設定

Solaris 9 のインストール時にアプリケーションサーバーを同時にインストールした場合、またはアプリケーションサーバーのインストール時にインストールするコンポーネントとして PointBase を選ばなかった場合は、PointBase はインストールされません。

PointBase がインストールされ、サンプルアプリケーションで使えるように設定されているかどうかを調べるには、次のディレクトリでアプリケーションサーバーのインストールを確認します。

```
<appserver_install_dir>/pointbase/
```

このディレクトリが存在するときは、アプリケーションサーバーのインストール時に PointBase のインストールも完了しています。このマニュアルの 74 ページの「JDBC ドライバの設定」に戻ってデータベースの設定を続けてください。

PointBase をインストールし、サンプルアプリケーションで使えるように設定する手順は次のとおりです。

1. PointBase Server と Client のダウンロードとインストール

2. PointBase データベースのサンプルファイルのコピー

3. アプリケーションサーバーのクラスパスへの PointBase Type 4 JDBC ドライバの追加

1. PointBase Server と Client のダウンロードとインストール

1. http://www.pointbase.com にアクセスし、PointBase の Evaluation バージョンをダウンロードします。
2. PointBase Server と Client をシステムにインストールします。

   PointBase Client には PointBase Type 4 JDBC ドライバが含まれます。

2. PointBase データベースのサンプルファイルのコピー

Sun ONE Application Server 7 には、サンプルアプリケーション用に事前に内容が設定された PointBase データベースのファイルが含まれています。PointBase のインストールを他のユーザーと共用しない場合は、このデータベースファイルをサンプルのディレクトリから PointBase のインストールディレクトリにコピーする必要があります。

PointBase のインストールを他のユーザーと共用する場合は、後述する方法で専用の PointBase 実行時環境を作成する必要があります。

この場合は、「3. アプリケーションサーバーのクラスパスへの PointBase Type 4 JDBC ドライバの追加」に進んで次の手順を参照してください。

内容が設定されているデータベースのファイルを PointBase の専用インストールにコピーするには、次のディレクトリに含まれるファイルをコピーします。

```
<appserver_install_dir>/samples/pointbase/databases/*
```

から次のディレクトリにコピーします。

```
<pointbase_install_dir>/databases/
```

これにより、`sun-appserv-samples$1.wal` および `sun-appserv-samples.dbn` というファイルが PointBase のインストールディレクトリにコピーされます。

3. アプリケーションサーバーのクラスパスへの PointBase Type 4 JDBC ドライバの追加

1. PointBase Type 4 JDBC ドライバを PointBase のインストールディレクトリからアプリケーションサーバーインスタンスの `lib/` ディレクトリにコピーします。たとえば、次のようなディレクトリです。

   ```
   .../domains/domain1/server1/lib/
   ```

   JDBC ドライバは、`<pointbase_install_dir>/lib/` というディレクトリにあります。ドライバの名前は `pbclientnn.jar` で、`nn` は PointBase のバージョンを意味します。

2. アプリケーションサーバーを再起動してドライバを認識させます。

アプリケーションサーバー側の設定で、「クラスパスのサフィックス」フィールドに PointBase ドライバの場所を指定することもできます。

1. 管理コンソールを開いてアプリケーションサーバーインスタンスの「JVM 設定」-> 「パス設定」を表示し、変更を加えます。

「クラスパスのサフィックス」フィールドに PointBase JDBC ドライバがすでに設定されているときは、新たにインストールしたドライバへのパスに変更します。

2. 「クラスパスのサフィックス」フィールドの設定を変更したら、変更を適用し、アプリケーションサーバーのインスタンスを再起動します。

これで PointBase のインストールと設定が完了しました。74 ページの「JDBC ドライバの設定」に戻ってデータベースの設定を続けてください。

第 7 章

サンプルアプリケーションの配備と実行

JDBC 接続プール、およびアプリケーションに必要なリソースエントリを定義したら、今度はアプリケーションを配備します。

サンプルアプリケーションには、生成済みの EAR ファイルのコピーが含まれますが、Ant の機能とサンプルに含まれる `build.xml` ファイルを使ってアプリケーションのソースコードをコンパイルし、EAR ファイルを最初から再アセンブルすることもできます。新たに作成したこの EAR ファイルをアプリケーションサーバーに配備し、アプリケーションを実行します。

- アプリケーションのコンパイルと再アセンブル
- サンプルアプリケーションの配備
- サンプルアプリケーションを監視する準備
- アプリケーションの実行

アプリケーションのコンパイルと再アセンブル

アプリケーションサーバーのインストールを他のユーザーと共用している場合、またはアプリケーションサーバーがインストールされている領域に対する書き込み権がシステムユーザー ID に割り当てられていない場合は、サンプルアプリケーションを専用ディレクトリにコピーする必要があります。手順については、94 ページの「サンプルアプリケーションの専用コピーの作成」を参照してください。それ以外の場合は 94 ページの「アプリケーションのコンパイルと再アセンブル」に進みます。

サンプルアプリケーションの専用コピーの作成

アプリケーションサーバーのインストールを他のユーザーと共用している場合、またはアプリケーションサーバーがインストールされている領域に対する書き込み権がシステムユーザー ID に割り当てられていない場合は、ユーザー ID に書き込み権が割り当てられている領域に次のディレクトリをコピーします。

```
<install_dir>¥samples
```

このマニュアルで説明するサンプルアプリケーションに対して Ant の build 機能を使うときは、次のファイルに記録されている `com.sun.aas.installRoot` というプロパティがアプリケーションサーバーのインストールパスに設定されていることを確認してください。

```
<personal_samples_dir>/samples/common.properties
```

Solaris 9 と同時にアプリケーションサーバーをインストールした場合を除き、このプロパティはアプリケーションサーバーのインストール時に自動的に設定されます。

Solaris 9 環境の `/usr/appserver/samples` からサンプルアプリケーションをコピーする場合は、`common.properties` ファイルの内容を次のように変更します。

```
com.sun.aas.installRoot=/usr/appserver
```

このプロパティの変更は、このマニュアルで説明するレベルでの Ant の使用には適していますが、サンプルアプリケーションに対して Ant の機能を十分に活用するには、`common.properties` ファイルの別のプロパティにも変更が必要です。このマニュアルで説明するすべての練習が終わったら、サンプルアプリケーションのマニュアルで「Using Ant with the Samples」の項を参照し、`common.properties` ファイルのその他のプロパティについて詳細を確認してください。

このマニュアルで `<install_dir>/samples/` と表記されるディレクトリは、サンプルアプリケーションの専用コピーが保存されているディレクトリに置き換えて解釈する必要があります。

アプリケーションのコンパイルと再アセンブル

1. アプリケーションサーバーの `bin` ディレクトリが環境に設定されていることを確認します。

   この手順については、37 ページの「環境の設定」を参照してください。

2. コマンド行を使って、サンプルアプリケーション `jdbc-simple` のソースディレクトリに移動します。

   ```
   cd <install_dir>/samples/jdbc/simple/src
   ```

3. コマンド行で、引数を指定せずに Ant のラッパースクリプト `asant`（`.bat`）を実行し、Java ソースファイルをコンパイルして、J2EE WAR、EJB JAR、EAR の各ファイルをアセンブルします。

   ```
   asant
   ```

ヒント **asant について**：`asant` ユーティリティは、Apache Ant メインクラスを呼び出すラッパースクリプトに過ぎません。ラッパースクリプトは、まずアプリケーションサーバー環境に関連する適切なクラスパスを設定します。たとえば、適切な JDK、およびアプリケーションサーバーに含まれるカスタム Ant タスクを記録した JAR ファイルが環境に設定されます。独自にコピーした Ant を使うときは、専用の Ant 環境に同じクラスパスが設定されているかどうか、`asant(bat)` ファイルの内容を確認します。

ヒント **asant の機能**：`make` コマンドがローカルな Makefile を自動的に処理するのと同様に、`asant` ユーティリティを実行すると、実行したディレクトリから `build.xml` というファイルを自動的に検出します。また、`make` と同様に Ant もデフォルトターゲットという概念を使用しています。アプリケーションサーバーに含まれるサンプルアプリケーションについて言えば、core というデフォルトターゲットによってサンプルアプリケーションが完全に再ビルドされます。これ以外に一般的なターゲットには compile と deploy があります。詳細については、サンプルアプリケーションのマニュアルを参照してください。

4. 次のディレクトリで `jdbc-simple.ear` というファイルを探し、EAR ファイルが作成されたことを確認します。

   ```
   cd ../assemble/ear
   ```

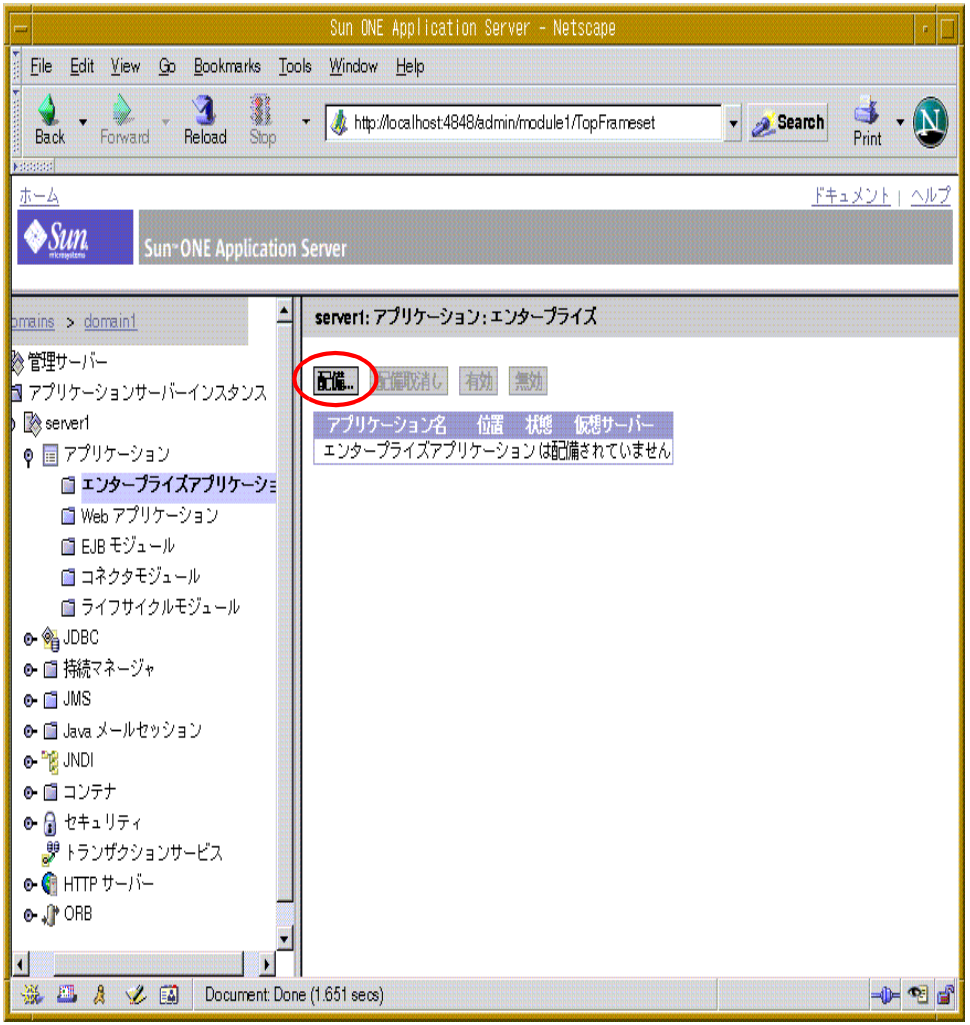
   作成された `jdbc-simple.ear` というファイルはこのディレクトリに保存されます。

   これでアプリケーションを最初からアセンブルできました。次に、管理コンソールを使ってこのアプリケーションを配備します。

サンプルアプリケーションの配備

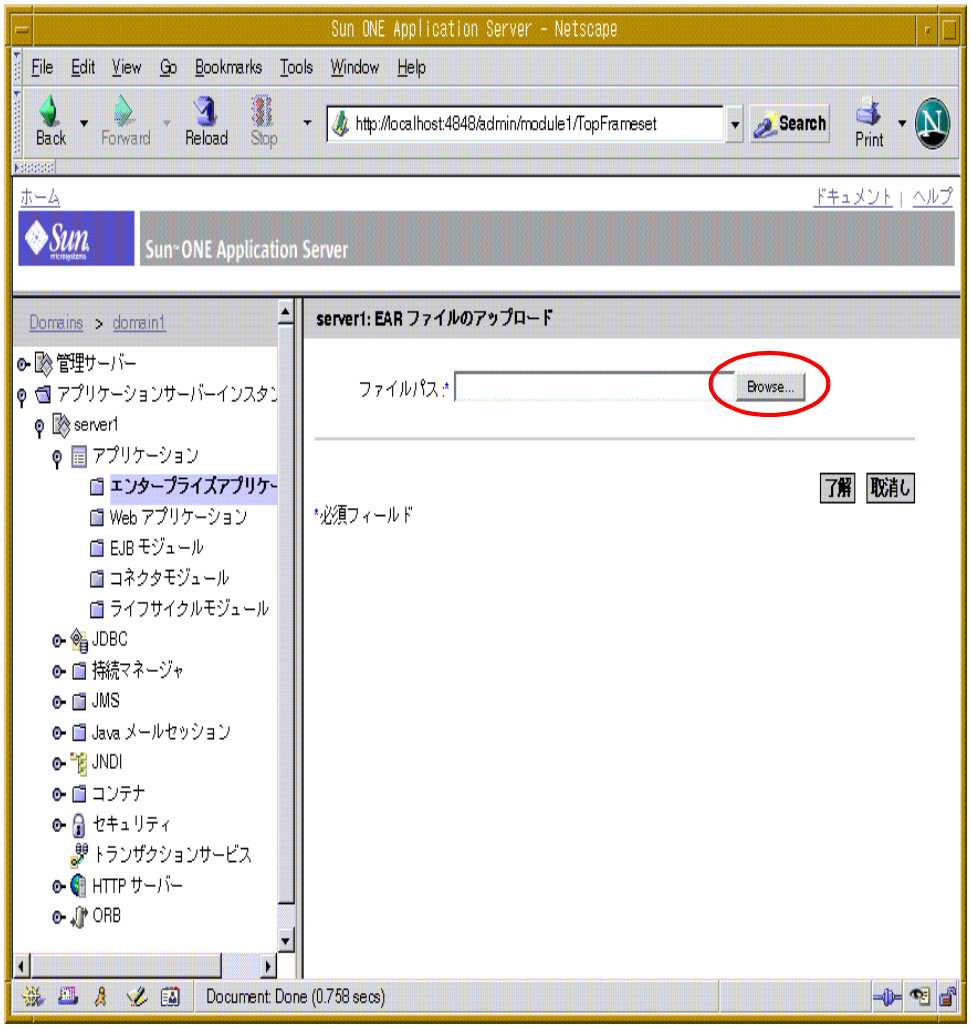
1. 管理コンソールを開き、アプリケーションサーバーインスタンス `server1` の中の Applications のノードを開きます。
2. Enterprise アプリケーションのフォルダを選びます。
3. 「配備 ...」ボタンをクリックします。

サンプルアプリケーションの配備



4. 「Browse」ボタンをクリックし、ファイルを指定するダイアログボックスを表示します。

ファイルの選択

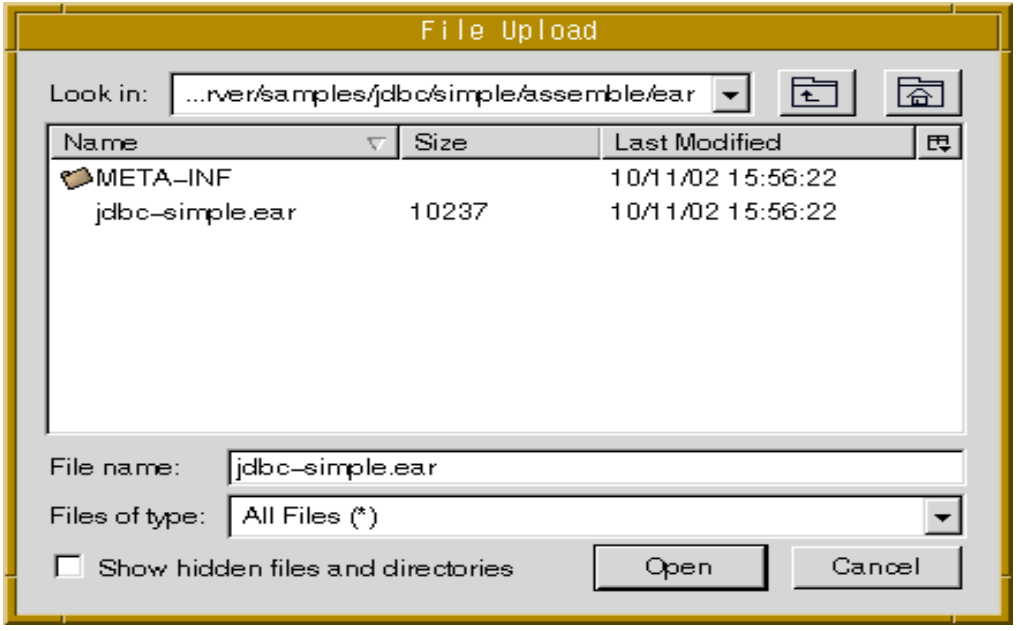


5. 次のディレクトリに移動し、前項でアセンブルした `jdbc-simple.ear` ファイルを選択します。

   `<install_dir>/samples/jdbc/simple/assemble/ear/`

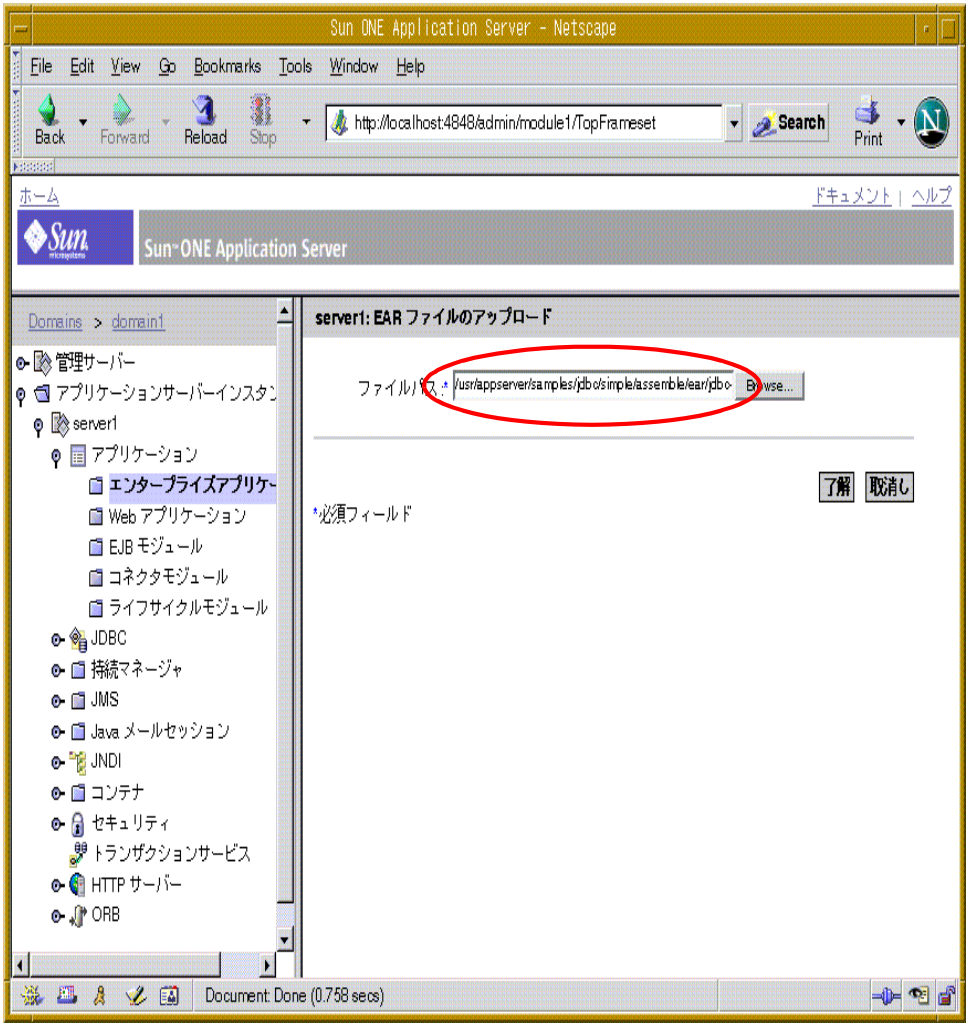
6. 「開く」をクリックして、ファイルの選択を確定します。

ファイルのアップロード



7. 「了解」をクリックして先に進みます。

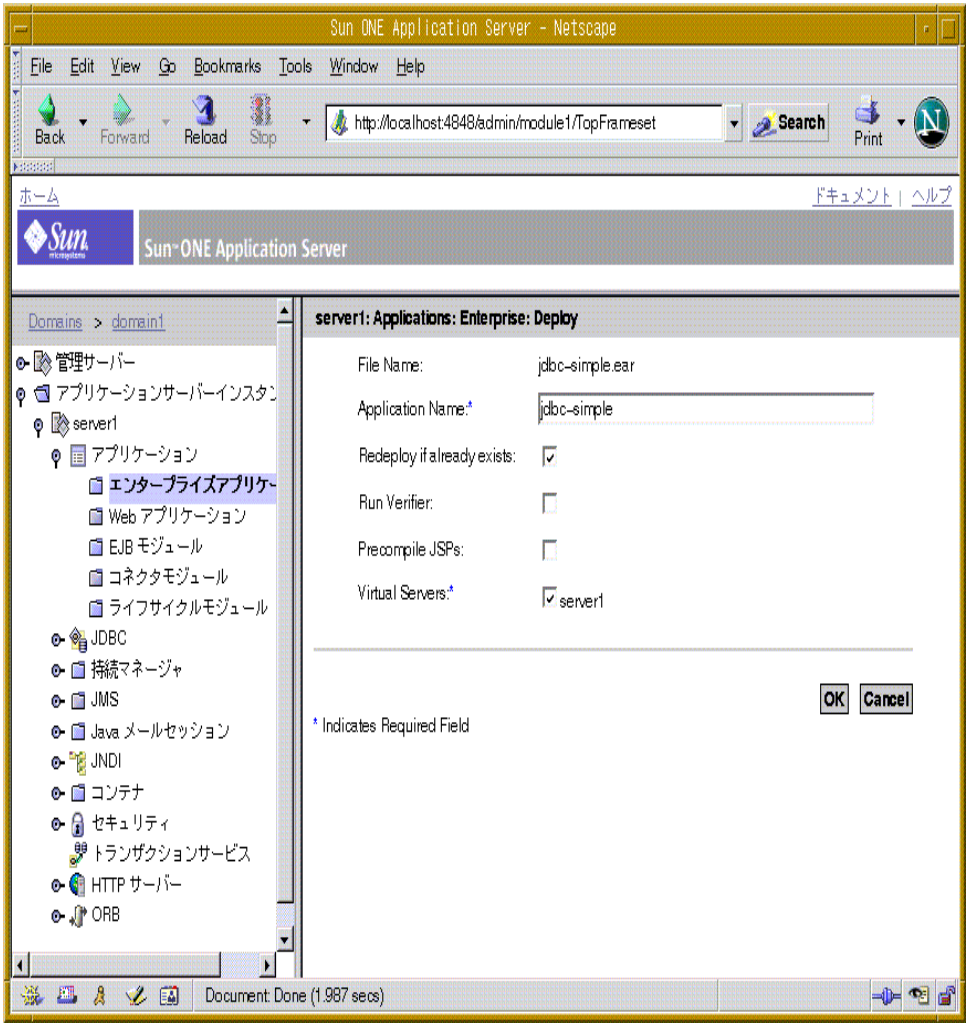
配備ファイルのアップロード



「了解」をクリックしてアプリケーションを配備する前に、「ベリファイア実行」と「JPS をプリコンパイル」の 2 つのチェックボックスを確認します。これは、配備プロセスの一部として J2EE アプリケーションベリファイアを実行するかどうかを指定するオプションです。サンプルアプリケーションで `asant verify` を実行してベリファイアを使用することもできます。「JPS をプリコンパイル」オプションを有効にすると、配備プロセスの最中に JSP ファイルがコンパイルされます。JSP をコンパイルする分だけ配備プロセスが長引きますが、JSP に最初にアクセスするのにかかる時間はずっと短くなります。

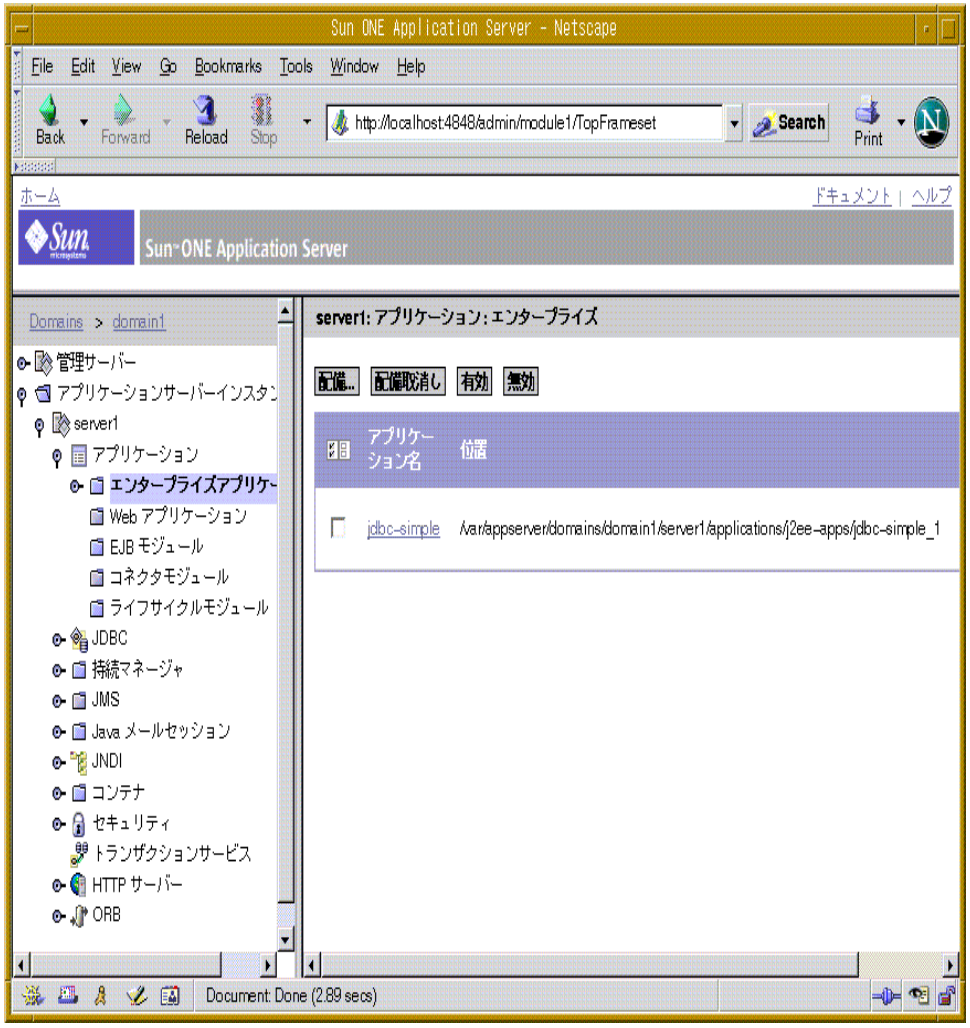
8. 「了解」をクリックしてアプリケーションを配備します。

jdbc-simple アプリケーション



9. 配備プロセスが完了すると、次のページが表示されます。

配備プロセスの完了


Sun ONE Application Server - Netscape
File Edit View Go Bookmarks Tools Window Help
Back Forward Reload Stop
http://localhost:4848/admin/module1/TopFrameset
Search
Print
ホーム
ドキュメント | ヘルプ
Sun-ONE Application Server
Domains > domain1
管理サーバー
アプリケーションサーバーインスタン
server1
アプリケーション
エンタープライズアプリケー
Web アプリケーション
EJB モジュール
コネクタモジュール
ライフサイクルモジュール
JDBC
持続マネージャ
JMS
Java メールセッション
JNDI
コンテナ
セキュリティ
トランザクションサービス
HTTP サーバー
ORB
server1: アプリケーション: エンタープライズ
配備...
配備取消し
有効
無効
アプリケーション名
位置
jdbc-simple
/var/appserver/domains/domain1/server1/applications/j2ee-apps/jdbc-simple_1
Document: Done (2.89 secs)

注

配備時に行われる処理 : アプリケーションを正しく配備するには、管理サーバーが稼動している必要があります。配備するアプリケーションを受け取った管理サーバーは、対象となるサーバーインスタンスの `server.xml` ファイルにアプリケーションを登録し、対象インスタンスのアプリケーション配備領域にアプリケーションをインストールします。

アプリケーションに EJB が含まれて入れば (EJB が存在すれば)、アプリケーションを初めて配備するときに、管理サーバーはすべての EJB についてスタブとスケルトンを自動的に生成します。同じアプリケーションの 2 回目以降の配備では、EJB のインタフェースに変更があったかどうかが配備インフラストラクチャによって検出されるので、初回の配備と比較して EJB の処理はずっと高速になります。これは、初回の配備にかかる時間の多くが EJB のスタブとスケルトンの生成にあてられているためです。EJB のインタフェースが変更されていない場合は、再配備時にスタブとスケルトンは再生成されません。この機能を「スマート再配備」と呼んでいます。

EAR の配備が完了すると、アプリケーションが次のディレクトリに展開されます。

```
<domain_config_dir>/domain1/server1/applications/j2ee-apps/jdbc-simple_1
```

最後の `_1` は、そのアプリケーションを配備した回数を意味しています。再配備を繰り返すたびに、この番号は増分されます。

個別に配備される Web モジュールと EJB モジュールは、`applications/j2ee-modules` ディレクトリに展開されます。

10. 次の方法で変更を適用し、配備したアプリケーションをアプリケーションサーバーのインスタンスに認識させます。
    a. `server1` のノードを選びます。
    b. 「変更の適用」をクリックします。

       この場合、サーバーインスタンスを再起動する必要はありません。

注

モジュールの配備 : Sun ONE Application Server 7 は、EAR ファイルの配備だけでなく、WAR モジュールと EJB JAR モジュールの配備にも対応しています。モジュールの配備は、Web 専用アプリケーションで特に便利です。管理コンソールの Applications ノードには Web Apps フォルダと EJB フォルダが含まれています。モジュールは個別に配備され、各フォルダに保存されます。各モジュールは分離されており、EAR ベースの各アプリケーションと、個別に配備された WAR および EJB JAR の各モジュールとでは、アプリケーションサーバーのインスタンスが使用するクラスローダの一時領域がそれぞれ異なります。

ヒント **コマンド行インタフェースによる配備** : 管理コンソールからだけでなく、asadmin ユーティリティを使ってコマンド行からアプリケーションを配備することもできます。詳細を確認するには、次のコマンドを実行します。

```
asadmin deploy --help
```

また、アプリケーションサーバーには Ant のカスタムタスクが含まれています。これは、Ant の build.xml ファイルを使ったアプリケーション配備プロセスを容易にするためのタスクです。サンプルアプリケーションのすべての build.xml ファイルは deploy ターゲットに対応しています。詳細については、各サンプルアプリケーションと『Using Ant with the Application Server』を参照してください。

これでアプリケーションを配備できました。アプリケーションを実行する前に、アプリケーションがサーバーログファイルに出力する内容を表示できるように環境を設定します。

サンプルアプリケーションを監視する準備

サンプルアプリケーションを実行する前に、アプリケーションとアプリケーションサーバーランタイムの両方から出力される内容を表示できるように準備します。アプリケーションから stdout と stderr に出力される内容は、デフォルトの設定ではアプリケーションサーバーのイベントログにリダイレクトされます。このため、サーバー側のアプリケーションとアプリケーションサーバーインフラストラクチャの両方の実行をこのイベントログで監視できます。

- UNIX 環境でのアプリケーション出力とログの表示
- Windows 環境でのアプリケーション出力とログの表示
- 管理コンソールによるログの表示

UNIX 環境でのアプリケーション出力とログの表示

UNIX 環境でのログファイルの監視には、通常は tail -f コマンドが使用されます。

1. server1 インスタンスのログディレクトリに移動します。

   ```
   cd <domain_config>/domain1/server1/logs/
   ```

2. ログファイルに対して tail を実行します。

   ```
   tail -f server.log
   ```

106 ページの「管理コンソールによるログの表示」に進みます。

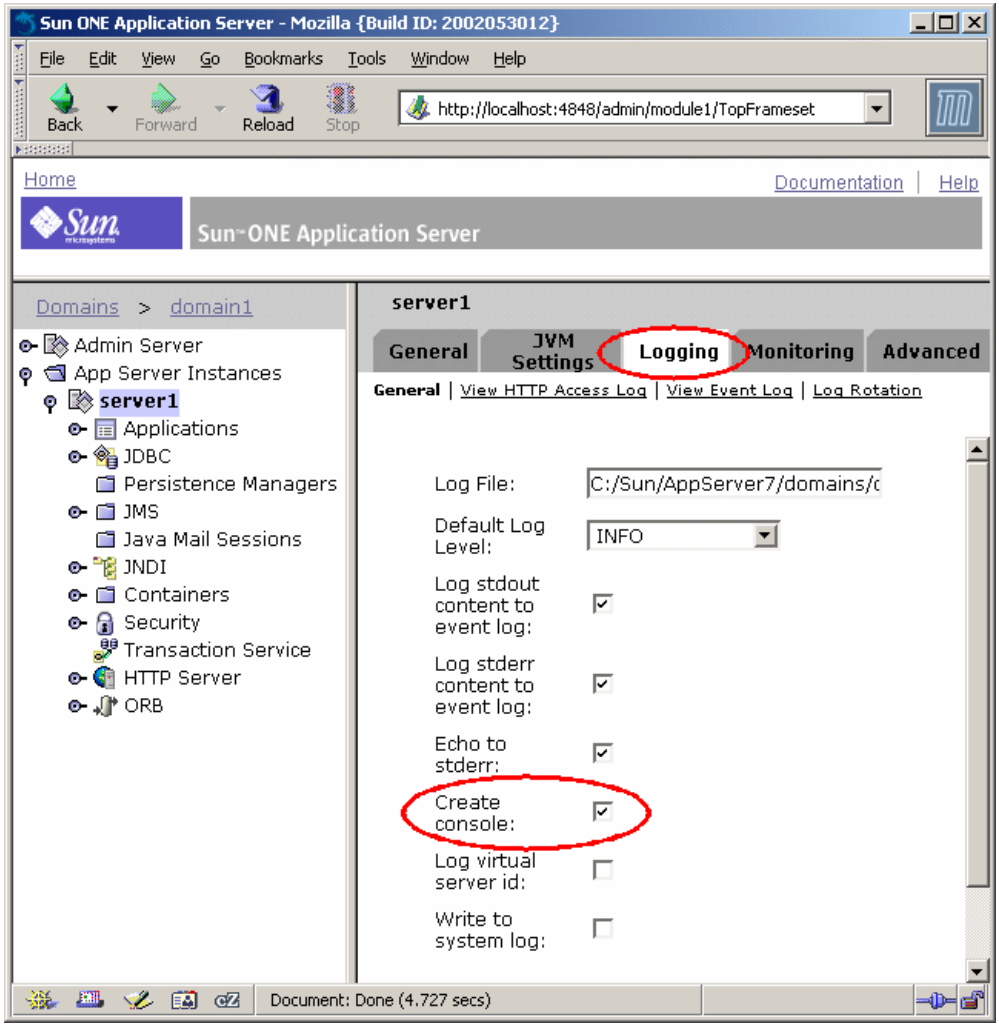
Windows 環境でのアプリケーション出力とログの表示

アプリケーションサーバーのインスタンスを使用してアプリケーションをテストする場合、Windows のデスクトップでサーバーのイベントログ情報を確認できます。表示される情報は、アプリケーションから `stdout` および `stderr` への出力内容、例外とサーバーイベントに関するメッセージなどです。

すでに説明したように、デフォルトの設定ではアプリケーションサーバーインスタンスのイベントログ情報はデスクトップに表示されます。

イベントログデータをデスクトップに表示しないようにするには、管理コンソールで設定を変更します。アプリケーションサーバーインスタンスの「ログ」タブをクリックして「一般」ペインにアクセスすると、「Create console/ コンソールを作成」というオプションが表示されます。これは、インスタンスのアプリケーションサーバーログをデスクトップ上のコマンドウィンドウに表示するかどうかを指定するためのオプションです。

「Create console/ コンソールを作成」オプション

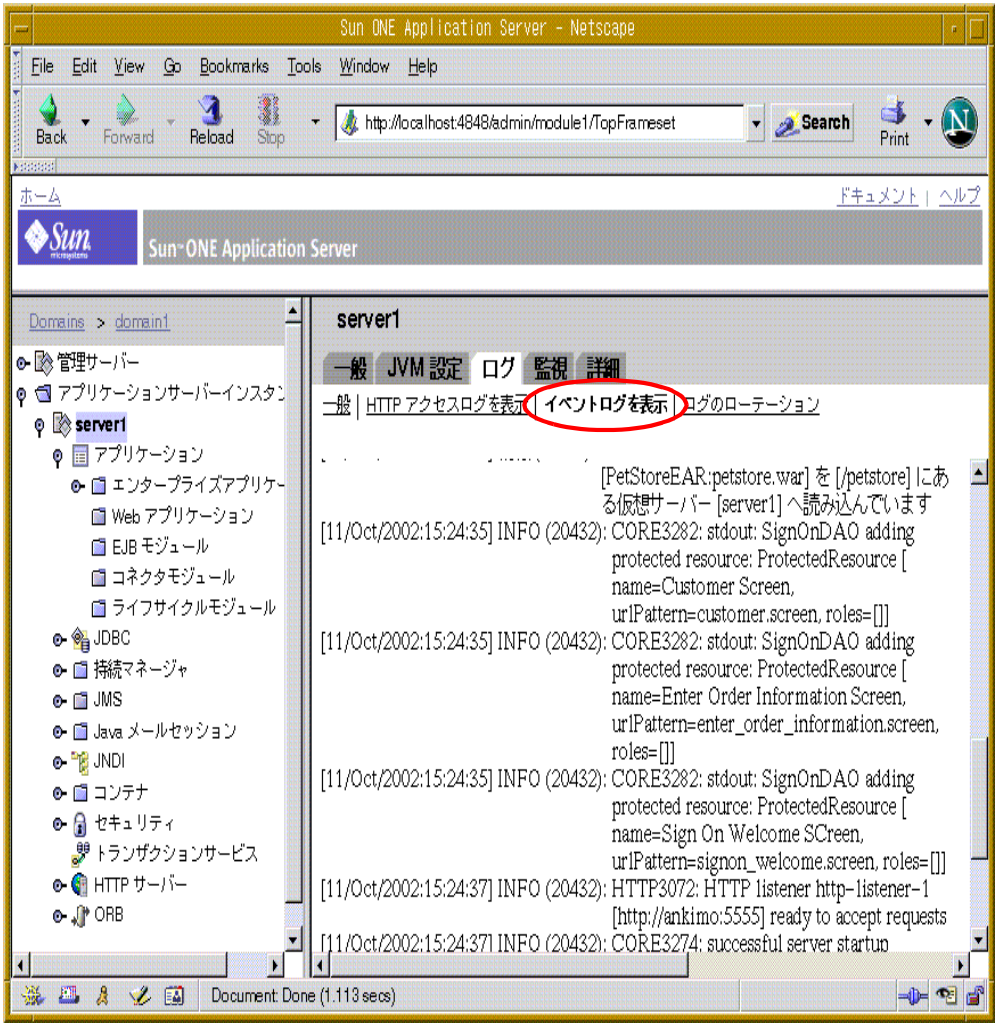


管理コンソールによるログの表示

サーバーインスタンスのログファイルを管理コンソール上で確認することもできます。

1. 管理コンソールを開きます。
2. `server1` のノードを選びます。
3. 「ログ」タブを開き、「イベントログを表示」というリンクをクリックします。

「イベントログを表示」ペイン



「了解」ボタンをクリックするとログの表示が更新されます。

25 項目より多くのログを表示するには、「表示するイベント数」に表示する項目数を入力し、「了解」をクリックして表示を更新します。たとえば「200」と入力し、どのように表示されるかを確認してください。

注 **HTTP アクセスログの表示 :**「ログ」タブには「HTTP アクセスログを表示」というリンクがあります。このリンクをクリックすると、そのサーバーインスタンスの HTTP アクセスログが表示されます。アクセスログのファイル名は access です。デフォルトの設定では、アクセスログファイルはサーバーイベントログと同じディレクトリに保存されます。

```
<domain_config_dir>/domain1/server1/logs/
```

これでサーバーのログファイルを監視できるようになりました。次に、データベースを起動し、アプリケーションを実際に実行します。

アプリケーションの実行

サンプルアプリケーションを実行する手順は、次のとおりです。

1. まず、サーバーイベントログファイルの内容を表示し、アプリケーションからの出力を確認できるようにします。
2. ブラウザを起動して次の URL にアクセスし、名前を入力して「Process」をクリックします。

   ```
   http://localhost:<port>/jdbc-simple
   ```

   「Process」ボタンをクリックすると、アプリケーションからの出力がサーバーのイベントログファイルに書き込まれるので、これを確認します。また、アプリケーションサーバーによって JSP ソースファイルの最初のコンパイルが行われるため、処理に時間がかかることにも注意してください。

注 **サーバーのイベントログに記録されるアプリケーションからの出力**:
アプリケーションが stdout または stderr 、あるいはその両方に情報を書き込むと、同じ情報が INFO レベルのメッセージとしてサーバーインスタンスのイベントログファイルにも記録されます。このメッセージの ID は、それぞれ CORE3282 (stdout) および CORE3283 (stderr) となります。サンプルアプリケーション jdbc-simple の実行中は、アプリケーションから多数のメッセージが生成され、サーバーのイベントログに記録されます。次に、出力されるメッセージの一部を引用します。

たとえば、次のようなメッセージが記録されます。

```
INFO: CORE3282: stdout: GreeterDBServlet is executing...

INFO: CORE3282: stdout: Retrieving JNDI initial
context...

INFO: CORE3282: stdout: - Retrieved initial context
successfully

INFO: CORE3282: stdout: Looking up dbGreeter bean home
interface...

INFO: CORE3282: stdout: - Looking up:
java:comp/env/ejb/jdbc-simple

INFO: CORE3282: stdout: - Looked up the EJB successfully
```

---

3. 挨拶文が表示されたら、表示されるリンクをクリックして過去に作成されたすべての挨拶文のログを表示します。

   2 つ目の JSP ソースファイルのコンパイルが行われるため、ここでも処理に時間がかかります。

4. サンプルアプリケーションを実行して、2 回目の応答がどれだけ速いかを確認します。

   JSP ファイルがすでにコンパイルされているので、2 回目の実行は初回時よりずっと速くなります。

トラブルシューティング

サンプルアプリケーションの実行に関する一般的な問題を、次の表にまとめます。

表 7-1 トラブルシューティング

状態	考えられる原因	解決法
最初のページにアクセスできない	アプリケーションサーバーが稼動していない	アプリケーションサーバーが稼動していることを確認する
	URL に別のポート番号を指定している	HTTP サーバーの正しいポート番号を指定する
		詳細については、43 ページの「アプリケーションサーバーの起動と停止」を参照
挨拶文は表示されるが、過去の挨拶文の数が「0」と表示される	`jdbc/jdbc-simple` が定義されていないため、JNDI 検索に失敗している	JDBC リソースが正しく定義されていることを確認する
	JDBC への接続に失敗している	JDBC 接続プールのプロパティが解説手順に記載されている設定と一致することを確認する
	データベースが稼動していない	86 ページの「PointBase Server データベースの起動」を参照
メインページを表示しようとすると 404 エラーとなる	アプリケーションが配備されていない	96 ページの「サンプルアプリケーションの配備」を参照

問題を解決したら、必ずアプリケーションサーバーのログファイルを確認してください。また、HTTP アクセスログファイルの内容を確認すれば、HTTP 要求が正しくアプリケーションサーバーに送られているかどうかを確認できます。

第 8 章「サンプルアプリケーションの変更」に進みます。この章では、アプリケーションサーバーが動的再配備と動的再読み込みにどのように対応しているかについて説明します。

第 8 章

サンプルアプリケーションの変更

開発者の生産性を向上させるために、Sun ONE Application Server 7 にはいくつかの動的な配備機能が用意されています。この章ではこれらの機能について説明し、前章で配備したサンプルアプリケーションを使っていくつかの練習を行います。

- 動的配備と動的再読み込み
- アプリケーションコンポーネントの変更

動的配備と動的再読み込み

開発サイクルの短縮のために、Sun ONE Application Server 7 には次の機能が用意されています。

- 動的配備と動的再読み込み
- スマート再配備
- 動的再読み込み

動的配備と動的再読み込み

J2EE アプリケーションを動的に配備したり、再読み込みしたりできるようにしたことで、対象となるアプリケーションサーバーインスタンスを再起動しなくても、アプリケーションの初回配備やその後の再配備が行えます。再配備時に、アプリケーションはアプリケーションサーバーによって完全に再読み込みされます。この機能は、EAR レベルの配備だけでなく、Web モジュールや EJB JAR モジュールの個別配備にも適用されます。動的配備と動的再読み込みには、サーバーの設定を変更する必要はありません。

注 **運用環境での動的再配備**：動的再配備の主な目的は開発時間を短縮することにありますが、アプリケーションを再配備する前に次の手順を実行すれば、運用環境でもこの機能を利用できます。

1. アプリケーションへの新しい要求を別のアプリケーションサーバーインスタンスにリダイレクトします。

2. すでに実行されているセッションを終了できるようにします。

アプリケーションに関連するすべてのセッションの終了を確認するには、アプリケーションサーバーに組み込まれている監視機能を利用できますが、送られてくる要求のリダイレクトには、外部のロードバランサなどが必要になるかもしれません。

スマート再配備

スマート再配備によって、EJB ベースのアプリケーションを迅速に再配備できます。アプリケーションを再配備すると、前回の配備以降にインタフェースが変更された EJB のスタブクラスとスケルトンクラスだけが再生成されます。配備に要する時間の多くはスタブとスケルトンの生成にあてられているため、スマート再配備によって待ち時間を大幅に短縮できます。この機能を使用するのに、サーバーの設定を変更する必要はありません。スマート再配備は、すべての再配備作業に適用されます。

動的再読み込み

動的再読み込みは、アプリケーションの完全な再アセンブルと再配備、およびアプリケーションサーバーの再起動を行わなくても、アプリケーションサーバーが個々のクラスファイル、Web コンテンツの変更、および配備記述子の変更を読み込み直す機能です。クラスファイルと配備記述子の変更を動的に再読み込みするために、サーバーインスタンスは `.reload` という特別なファイルを監視しています。このファイルは、アプリケーションまたは個々の配備モジュールの配備領域の最上位層に保存されます。開発者による編集や make 機能の実行によって `.reload` ファイルに変更が加えられると、サーバーインスタンスはアプリケーションまたは個々の配備モジュールを完全に読み込み直します。この機能を利用するには、アプリケーションサーバーのインスタンスごとに明示的に有効にする必要があります。監視によるオーバーヘッドが加わるため、この機能は開発環境だけに適しています。

動的再読み込みを有効にしなくても、アプリケーションサーバーの Web コンテナは静的なコンテンツや JavaServer Page (JSP) などの Web コンテンツを監視し、変更の有無を確認します。変更があった場合は、次回のアクセス時に Web コンテナはファイルを自動的に読み込みし直します。

アプリケーションサーバーに関する著述では、「ホット配備」という用語を目にすることがあります。多くの場合、「ホット配備」は動的配備、動的再配備、および動的再読み込みを意味します。ただし、あるアプリケーションサーバーがこれらの機能すべてに対応しているかどうかを調べるには、その製品のマニュアルをよく読む必要があります。

アプリケーションコンポーネントの変更

この項では、サンプルアプリケーション `jdbc-simple` に含まれる次のコンポーネントに変更を加えます。

- Web コンテンツの再配備と再読み込み
  - サーブレットクラスファイルの動的再配備
  - 静的コンテンツの動的再読み込み
  - JSP ファイルの動的再読み込み
- EJB の再配備と再読み込み
  - EJB 実装クラスファイルの動的再配備
  - EJB 実装クラスファイルの動的再読み込み

サーブレットクラスファイルの動的再配備

ここでは、サーブレットクラスファイルの変更、アプリケーションの再アセンブル、およびアプリケーションの動的再配備を実際に行うことで、アプリケーションサーバーの動的再配備機能について説明します。動的再配備では、アプリケーションの再起動は必要ありません。

1. サーブレットのソースコードディレクトリに移動します。たとえば、次のようなディレクトリです。

   ```
   <install_dir>/samples/jdbc/simple/src/samples/jdbc/simple/servlet/
   ```

2. `GreeterDBServlet.java` ファイルを編集します。`System.out.println("¥nGreeterDBServlet is executing...");` という行を探し、文字列の先頭に「MODIFIED」を追加します。

   次に例を示します。`System.out.println("¥nMODIFIED GreeterDBServlet is executing...");`.

3. 変更を保存します。
4. サンプルを再コンパイルし、再アセンブルします。

a. サンプルアプリケーションの src ディレクトリに戻ります。

b. オプションを指定せずに asant を実行し、アプリケーションの再コンパイルと再アセンブルを行います。

   ここで再コンパイルされるソースファイルは 1 つだけで、残りの Java ソースファイルは再コンパイルされません。

5. 管理コンソールを使ってサンプルアプリケーションを再配備します。

   a. 管理サーバーが稼動していることを確認します。稼動していないときは、次のコマンドを実行すれば簡単に起動できます。

      ```
      asadmin start-instance admin-server
      ```

   b. 管理コンソールを開きます。

   c. server1 のノードに含まれる Applications のノードを開き、Enterprise Apps フォルダを開いて「配備」をクリックします。

   d. 69 ページの「サンプルアプリケーションの操作」に記載されている「アプリケーションの配備」で実行した手順に従って、アプリケーションを配備します。

      EJB のインタフェースは変更されていないので、再配備は速やかに完了します。これは、アプリケーションサーバーのスマート再配備機能によるものです。

6. サーバーのイベントログファイルを確認しながらアプリケーションを再実行します。

   次の stdout メッセージの先頭には、「MODIFIED」という単語が表示されています。

   ```
   INFO: CORE3282: stdout: MODIFIED GreeterDBServlet is executing...
   ```

ヒント **アプリケーションの再読み込みに関するメッセージ**: アプリケーションが完全に再読み込みされたことは、サーバーのイベントログファイルからも確認できます。サンプルアプリケーションの再配備時に、対象となるサーバーインスタンスのイベントログには次のメッセージが表示されます。

```
INFO: CORE5022: All ejbs of [jdbc-simple] were unloaded
successfully!

INFO: LOADER5010: All ejbs of [jdbc-simple] loaded
successfully!

INFO: CORE3276: Installing a new configuration

INFO: WEB0100: Loading web module
[jdbc-simple:jdbc-simple.war] in virtual server
[server1]

INFO: WEB0100: Loading web module [default-web-module]
in virtual server [server1]

INFO: CORE3280: A new configuration was successfully
installed

INFO: WEB4004: Closing web application environment for
virtual server [server1]
```

この練習では、サーブレットクラスファイルが変更されたアプリケーションを再配備することで、アプリケーションサーバーインスタンスの再起動を必要としない動的再配備機能のメリットを確認しました。

静的コンテンツの動的再読み込み

静的なコンテンツを動的に再読み込みする手順は、次のとおりです。

1. サンプルアプリケーション `jdbc-simple` を実行し、メインページのタイトルとして表示される文字列を確認します。
2. アプリケーションサーバーインスタンスのアプリケーション配備領域にある Web モジュールのディレクトリに移動します。

   ```
   <domain_config_dir>/domain1/server1/applications/j2ee-apps/jdbc-
   simple_n/jdbc-simple_war/
   ```

3. index.html ファイルを開き、次のように <TITLE> タグの直後に「MODIFIED」という文字列を追加します。

   ```
   <head>
   <title>MODIFIED JDBC-SIMPLE Sample Application</title>
   </head>
   ```

4. 変更を保存します。
5. アプリケーションを再実行し、新しいタイトルを確認します。

   Web ページのタイトルが変更されていない場合は、Shift キーを押しながらブラウザの「更新」ボタンをクリックすると、コンテンツがアプリケーションサーバーから再読み込みされます。

JSP ファイルの動的再読み込み

JavaServer Page (JSP) ファイルを動的に再読み込みする手順は、次のとおりです。

1. サンプルアプリケーション jdbc-simple を実行し、名前を入力して「Process」ボタンをクリックします。

   表示されるページの内容を確認します。このページを表示している JSP を変更します。

2. アプリケーションサーバーインスタンスのアプリケーション配備領域にある Web モジュールのディレクトリに移動します。

   ```
   <domain_config_dir>/domain1/server1/applications/j2ee-apps/jdbc-simple_n/jdbc-simple_war/
   ```

3. GreeterDBView.jsp ファイルを開き、次のように <TITLE> タグの直後に「MODIFIED」という文字列を追加します。

   ```
   <head>
   <title>MODIFIED JDBC-SIMPLE Sample Application</title>
   </head>
   ```

4. 変更を保存します。
5. アプリケーションを再実行し、ブラウザウィンドウに表示される新しいタイトルを確認します。

   変更を検出した Web コンテナは JSP ファイルを再コンパイルするため、変更後のアプリケーションの最初の実行には時間がかかります。

6. アプリケーションを再実行します。JSP のコンパイルが完了しているので、今回は速やかに実行されます。

ヒント **Web コンテンツの動的再読み込み** : 静的コンテンツと JSP の動的再読み込みでは、アプリケーションサーバーインスタンスの動的再読み込み機能を有効にする必要がありませんでした。デフォルトの設定では、Web コンテナは Web コンテンツを動的に読み込みし直します。Web コンテナが JSP ファイルを再読み込みしないように設定するには、Web アプリケーションの `sun-web.xml` ファイルの設定を変更します。これは、変更および配備した JSP ソースファイルを誤って実行しないようにする場合に便利です。

EJB 実装クラスファイルの動的再配備

サーブレットクラスファイルを変更したのと同じ方法で、EJB 実装クラスファイルに変更を加え、アプリケーションサーバーを再起動せずに動的再配備を行なって、アプリケーションを簡単にテストしてみましょう。

1. EJB のソースコードディレクトリに移動します。たとえば、次のようなディレクトリです。

   ```
   <install_dir>/samples/jdbc/simple/src/samples/jdbc/simple/ejb/
   ```

2. `GreeterDBBean.java` ファイルを編集します。

   `System.out.println("GreeterDB EJB is determining message...");` という行を探し、文字列の先頭に「MODIFIED」を追加します。

   次に例を示します。`System.out.println("MODIFIED GreeterDB EJB is determining message...");`.

3. 変更を保存します。
4. サンプルを再コンパイルし、再アセンブルします。
   a. サンプルアプリケーションの `src` ディレクトリに戻ります。
   b. オプションを指定せずに `asant` を実行し、アプリケーションを再コンパイルし、再アセンブルします。ここで再コンパイルされるソースファイルは 1 つだけで、残りの Java ソースファイルは再コンパイルされません。
5. 管理コンソールを使ってサンプルアプリケーションを再配備します。
   a. 管理サーバーが稼動していることを確認します。稼動していないときは、次のコマンドを実行すれば簡単に起動できます。

      ```
      asadmin start-instance admin-server
      ```

   b. 管理コンソールを開きます。
   c. Enterprise Apps フォルダを開いて「配備」をクリックします。
   d. 前の項と同じ手順でアプリケーションを配備します。

EJB のインタフェースは変更されていないので、再配備は速やかに完了します。

6. サーバーのイベントログファイルを確認しながら、アプリケーションを再実行します。

   次の `stdout` メッセージの先頭には、「MODIFIED」という単語が表示されています。

   ```
   INFO: CORE3282: stdout: MODIFIED GreeterDB EJB is determining
   message...
   ```

この練習では、EJB 実装クラスファイルが変更されたアプリケーションを再配備することで、アプリケーションサーバーインスタンスの再起動を必要としない動的再配備機能のメリットを確認しました。

EJB 実装クラスファイルの動的再読み込み

ここでは、動的再読み込み機能について説明します。クラスを動的に読み込み直す利点は、動的な再配備による利点より複雑に考えられがちですが、開発時間の短縮という観点から言えば、動的再配備よりも動的再読み込みのほうが効果的です。個々のファイルをビルド領域から配備領域にコピーするプロセスを make 機能、または Java コンパイラのクラスファイル送信先設定を使って自動化することで、動的再読み込み機能のメリットを引き出すことができます。

この練習の前半では、`GreeterDBBean's` 実装クラスに新たな変更を加えます。EJB を再コンパイルしたら、それをアプリケーションの配備領域に手動でコピーします。次に、`.reload` という特別なファイルを作成し、サンプルアプリケーションを再実行します。`.reload` ファイルを作成することで、アプリケーションの配備領域に 1 つまたは複数のファイルが新たにコピーされたこと、およびアプリケーション全体の再読み込みが必要であることをサーバーインスタンスに認識させることができます。アプリケーションサーバーのインスタンスがアプリケーション全体を再読み込みするときに、メモリーに残されているすべてのアプリケーションコンポーネントはフラッシュされ、すべてが再読み込みされます。また、アプリケーションレベルの再読み込み時には、そのアプリケーションと関連するすべてのユーザーセッションもフラッシュされます。

変更したクラスファイルをコピーして `.reload` ファイルを作成するだけで、変更後のアプリケーションを直ちにテストできるようになります。

動的再読み込み機能を利用すれば、アプリケーションを再アセンブルしたり、あらためて再配備したりする必要はありません。

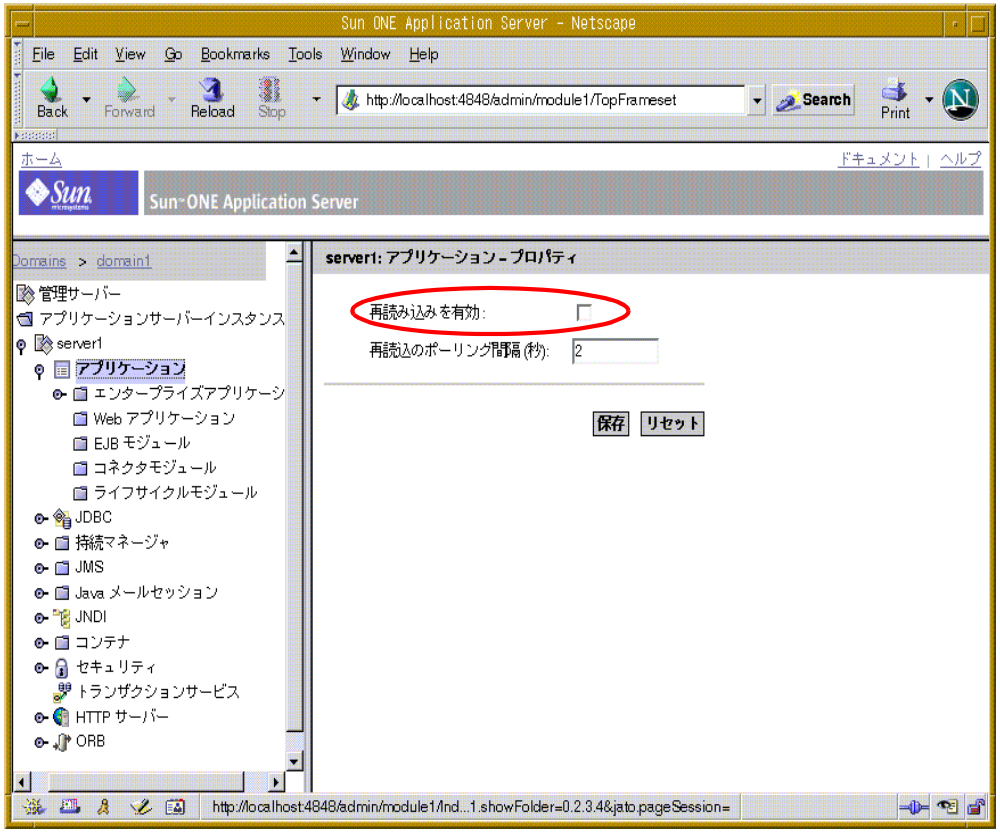
動的再読み込みの有効化

クラスの動的再読み込みを実際に行う前に、動的再読み込み機能を有効にする必要があります。

動的再読み込み機能を有効にすると、`.reload` ファイルの存在や変更が頻繁に確認されるため、オーバヘッドが増大します。運用環境でこの機能を有効にすることはお勧めできません。

1. 管理コンソールを開きます。
2. `server1` インスタンスのノードを選びます。
3. Applications フォルダを選びます。
4. 「再読み込みを有効」チェックボックスをクリックしてチェックマークをつけます。
5. 「保存」をクリックします。
6. サーバーインスタンス `server1` を選んで「変更の適用」をクリックし、サーバーインスタンスを再起動します。

「再読み込みを有効」オプション



次に、EJB 実装クラスを変更し、再コンパイルして手動で配備領域にコピーします。

EJB 実装クラスの変更

EJB 実装クラスを変更する手順は次のとおりです。

1. EJB のソースコードディレクトリに移動します。たとえば、次のようなディレクトリです。

   `<install_dir>/samples/jdbc/simple/src/samples/jdbc/simple/ejb/`

2. `GreeterDBBean.java` ファイルを編集します。`System.out.println("MODIFIED GreeterDB EJB is determining message...");` という行を探し、文字列の先頭に「MODIFIED AGAIN」を追加します。

   次に例を示します。`System.out.println("MODIFIED AGAIN GreeterDB EJB is determining message...");`.

3. 変更を保存します。

再コンパイルと配備領域へのコピー

アプリケーションを再コンパイルし、それを配備領域にコピーする手順は次のとおりです。

1. サンプルアプリケーションの `src` ディレクトリに戻ります。
2. ターゲット名に compile を指定して `asant` を実行し、変更した Java ソースファイルを再コンパイルします。ターゲット名に compile を指定するのは、アプリケーションが完全に再アセンブルされるのを防ぐためです。

   ```
   asant compile
   ```

   ここで変更されるソースファイルは 1 つだけで、残りの Java ソースファイルは再コンパイルされません。

3. 新たに生成されたクラスをローカル環境の `build/classes/` ディレクトリから対象となるアプリケーションサーバーインスタンスのアプリケーション配備領域に手動でコピーします。

   新たに生成された EJB 実装クラスは、次のディレクトリに保存されています。

   `<install_dir>/samples/jdbc/simple/build/classes/samples/jdbc/simple/ejb/GreeterDBBean.class`

4. このファイルを次のディレクトリにコピーします。

   `<domain_config_dir>/domain1/server1/applications/j2ee-apps/jdbc-simple_n/jdbc-simpleEjb_jar/samples/jdbc/simple/ejb/`

.reload ファイルの作成

アプリケーションの配備ディレクトリのルートに .reload という名前の空のファイルを作成します。

- Windows 環境では、メモ帳を使ってテキストファイルを新規作成し、それを次のディレクトリに保存します。

  ```
  <domain_config_dir>/domain1/server1/applications/j2ee-apps/jdbc-simple_n/.reload
  ```

注 **メモ帳を使って .reload ファイルを作成できない場合** : メモ帳で .reload ファイルを保存する際は、「ファイルの種類」を「すべてのファイル」に設定する必要があります。通常の設定では、.reload というファイル名に拡張子 .txt が追加されてしまいます。

- UNIX 環境では、上記ディレクトリで touch .reload を実行するだけで空のファイルを作成できます。

アプリケーションの再実行

1. サーバーのイベントログファイルを確認しながらアプリケーションを再実行します。

   次の stdout メッセージの先頭には、「MODIFIED AGAIN」という文字列が表示されています。

   ```
   INFO: CORE3282: stdout: MODIFIED AGAIN GreeterDB EJB is determining messa..."
   ```

2. println() ステートメントを変更して、EJB 実装クラスにさらに変更を加えます。EJP 実装クラスを再コンパイルしてコピーし、.reload ファイルにも新たな変更を加えてアプリケーションによる動的再読み込みを実行します。

ヒント

手動コピーの自動化：コマンド行インタフェースを使っている作業環境では、Ant の `build.xml` を使うか、あるいは Makefile のターゲット名に `update` などを指定することによって、開発作業領域で変更した `.class` ファイルをアプリケーションの配備領域に直接コピーするプロセスを自動化し、その上で `.reload` ファイルの修正にとりかかる方法が一般的です。

IDE を使っている環境では、`.class` ファイルがアプリケーションサーバーの配備領域に生成されるように IDE のコンパイラオプションを設定する方法が一般的です。このように設定しておけば、特定の Java ソースファイルを再コンパイルするたびに `.class` ファイルは配備領域に自動的にコピーされます。つまり、別のソース編集ウィンドウで `.reload` ファイルを修正するだけでアプリケーションの再読み込みが自動的に完了します。

これらの方法により、開発プロセス全体から再アセンブルと新たな再配備が除かれるため、編集、コンパイル、テストを繰り返す開発サイクルを大きく短縮することができます。動的な再読み込みには管理サーバーは関与しません。このため、配備したアプリケーションをこの方法でアップデートする場合は、管理サーバーを起動しておく必要がありません。

第 9 章「サーバーのその他の機能」に進みます。この章では、共通の管理作業について説明します。

第 9 章

サーバーのその他の機能

これまでに、アプリケーションの配備、再配備、および再読み込みの練習を通じてアプリケーションサーバーの基本機能について説明してきました。次の練習では、共通の管理タスクの実行方法について説明します。

- HTTP リスナーのポート番号の変更
- 新しい HTTP リスナーの追加
- 仮想サーバーの追加
- サーバーインスタンスの追加
- 管理ドメインの作成

HTTP リスナーのポート番号の変更

HTTP リスナーのポート番号がほかのサーバーと重複する場合、管理コンソールを使用して簡単に変更できます。

1. 管理コンソールを開きます。
2. アプリケーションサーバーインスタンスの HTTP サーバーフォルダと HTTP リスナーフォルダを展開します。
3. `http-listener-1` というリスナーをクリックします。
4. ポート番号を変更します。
5. 「保存」をクリックします。
6. `server1` インスタンスのノードを選びます。
7. 「変更の適用」をクリックします。サーバーの再起動は必要ありません。
8. ブラウザに新しいポート番号を指定し、HTTP サーバーにアクセスします。

新しい HTTP リスナーの追加

初期設定で使える HTTP リスナーは 1 つだけですが、簡単に別のリスナーを追加できます。次の練習では、SSL に対応していない新しい HTTP リスナーを作成します。

新しい HTTP リスナーを追加する手順は次のとおりです。

1. 管理コンソールを開きます。
2. HTTP サーバーフォルダを展開します。
3. 「HTTP リスナー」をクリックします。
4. 「新規」をクリックします。
5. 次の表のとおりに設定情報を入力します。

`Name`	new-listener
`IP Address`	0.0.0.0 (すべてのインタフェースで待機)
`Port`	任意のポートを選択
`Return Server Name`	そのマシンで有効な任意のホスト名またはインタフェース名

6. 「了解」をクリックします。
7. `server1` インスタンスのノードを選びます。
8. 「変更の適用」をクリックします。
9. サーバーインスタンスを再起動します。
10. ブラウザに両方のポート番号を指定し、HTTP サーバーにアクセスします。

    サーバーの再起動時に、サーバーインスタンスのイベントログに次のようなメッセージが記録されます。

```
INFO: HTTP3072: HTTP listener http-listener-1
[http://kampfire:88] ready to accept requests

INFO: HTTP3072: HTTP listener new-listener [http://kampfire:89]
ready to accept requests
```

仮想サーバーの追加

運用環境では、通常は 1 つの HTTP サーバーに複数ドメインのコンテンツと Web アプリケーションを持つように設定します。たとえば、同じ HTTP サーバーに別々の仮想サーバーを定義することで、www.abc.com ドメインと www.xyz.com ドメインの両方を簡単に混在させることができます。それぞれの仮想サーバーがどのトラフィックを処理するかは、HTTP トラフィックを受け取るインタフェース、またはポート番号、あるいは HTTP 要求を受け取るドメインの名前で決定されます。

この練習では、新たに定義した HTTP リスナーを使ってトラフィックを新しい仮想サーバーに配信します。

1. 管理コンソールを開きます。
2. HTTP サーバーフォルダを展開します。
3. 前述の手順に従って、新しい仮想サーバーに関連づける HTTP リスナーを新たに作成します。
4. 「仮想サーバー」をクリックします。
5. 「新規」をクリックします。
6. 次の表のとおりに設定情報を入力します。

ID	new-virtual-server
Hosts	受信 URL に指定するホスト名 (この例では、この仮想サーバーには一意のポート番号が割り当てられるので重要ではない)
MIME Types File	mime1
HTTP Listeners	1 つまたは複数の適切な HTTP リスナーを選択
Access Log	<domain_config_dir>/domain1/server1/logs/<access log file name>

7. 「了解」をクリックします。
8. server1 インスタンスのノードを選びます。
9. 「変更の適用」をクリックします。
10. サーバーインスタンスを再起動します。
11. ブラウザを起動し、新しい仮想サーバーと関連づけた HTTP リスナーポートにアクセスします。

サーバーインスタンスの追加

アプリケーションサーバーの設定をテストする上で、独立した作業領域が必要な場合は、アプリケーションサーバーの新しいインスタンスの定義が必要な場合があります。たとえば、Java 2 SDK (JDK) を新しいバージョンにアップグレードするときに、新しい JDK を試すと同時に従来の設定も残すような場合は、アプリケーションサーバーのインスタンスを新たに定義する必要があります。

サーバーインスタンスを追加する手順は次のとおりです。

1. 管理コンソールを開きます。
2. 「アプリケーションサーバーインスタンス」を選びます。
3. 「新規」をクリックします。
4. `my-new-server` のようなサーバーインスタンス名と、最初の HTTP リスナーのポート番号を入力します。
5. 「了解」をクリックします。
6. サーバーインスタンスのリストから新しいインスタンスを選びます。
7. 「変更の適用」をクリックします。
8. 「起動」をクリックします。
9. ブラウザを起動し、新しいサーバーインスタンスと関連づけた HTTP リスナーポートにアクセスします。
10. サーバーインストールの `domain1` ディレクトリに移動します。

    `<domain_config_dir>/domain1/`

    新たに作成したインスタンスのディレクトリが作成されていることを確認します。

    管理コンソールの「アプリケーションサーバーインスタンス」には、新しいサーバーインスタンスのノードが表示されます。新しいノードを開くと、作成したアプリケーションサーバーインスタンスのデフォルト設定を確認できます。

管理ドメインの作成

Solaris 9 のインストール時にアプリケーションサーバーを一括インストールした場合、すでに asadmin CLI を使って新しい管理ドメインを作成しているはずです。この場合は、最後の第 10 章「要約と参照情報」に進んでください。

それ以外の場合、次の手順に従って mydomain という新しいドメインを作成して起動し、メインを管理する管理サーバーにアクセスします。

ドメインの作成と削除に使うユーティリティは asadmin CLI だけなので、ドメインの新規作成にはコマンド行インタフェースを使用する必要があります。

1. コマンド行で次のコマンドを実行し、mydomain という名前の新しい管理ドメインを作成します。

   ```
   asadmin create-domain --adminport 9999 --adminuser admin
   --adminpassword password mydomain
   ```

   ポート番号、ユーザー名、パスワードのフィールドには、そのドメインの新しい管理サーバーの初期設定を指定します。

   このコマンドを実行すると、次のようなメッセージが表示されます。

   ```
   Created Domain mydomain successfully
   ```

   この時点で、新たに定義したディレクトリが表示されます。

   ```
   <domain_config_dir>/mydomain/
   ```

   domain1 というドメイン名がすでに使われている場合は、別の名前を指定してサブコマンド create-domain を実行します。ドメイン名にはピリオドなどの文字も使用できます。ドメイン名が重複しないように、ログインユーザー名を含めることも可能です。次に例を示します。ckamps.domain1

2. サブコマンド list-domains を実行し、アプリケーションサーバーのインストールに設定されているすべてのドメインをリスト表示します。

   次に例を示します。

   ```
   asadmin list-domains

   domain1 [<domain_config_dir>/mydomain]
   ```

   <domain_config_dir> は、新規作成した管理ドメインのデフォルトディレクトリ、またはサブコマンド create-domain の実行時に --path オプションに指定したディレクトリとなります。

ヒント **管理ドメインのデフォルトディレクトリの変更**: 新規作成した管理ドメインのデフォルトディレクトリを変更する場合、サブコマンド `create-domain` に `--path` オプションを指定します。管理ドメインの設定はこのディレクトリに書き込まれるので、書き込み権のあるディレクトリを指定する必要があります。

```
asadmin create-domain --path <domain_config_dir>
--adminport 4848 --adminuser admin --adminpassword
password mydomain
```

この例では、`<domain_config_dir>` で指定したディレクトリに `mydomain` という新しいディレクトリが作成されます。

3. 次のコマンドを実行して新しいドメインを起動します。

   ```
   asadmin start-domain --domain mydomain
   ```

   このコマンドは、新しいドメインの管理サーバーだけを起動します。

4. 新しいドメインの管理サーバーが起動すると、ブラウザでその管理サーバーの管理コンソールを開き、次の URL にアクセスできるようになります。

   ```
   http://localhost:9999
   ```

5. 管理サーバーの認証が完了したら、管理コンソールを使って新しいアプリケーションサーバーインスタンスを作成し、Web ブラウザからそのインスタンスの HTTP ポートにアクセスします。

   ドメインを停止するには、次のコマンドを実行します。

   ```
   asadmin stop-domain --domain mydomain
   ```

   すべての練習を完了したら、最後の第 10 章「要約と参照情報」に進みます。

第 10 章

要約と参照情報

このマニュアルでは、Sun ONE Application Server の主要機能について説明してきました。

- 主な機能とアーキテクチャ
- アプリケーションサーバーの制御
- J2EE アプリケーションの配備と実行
- JDBC によるデータベースへのアクセス
- J2EE アプリケーションの再コンパイルと再アセンブルを容易にする Ant ベースの build 機能の概要
- 製品のディレクトリ構造の概要
- 開発サイクルを効率化する動的な配備、再配備、再読み込み機能
- 仮想サーバーやアプリケーションサーバーインスタンスの新規作成などの共通管理タスク

更に詳細な情報を参照するには、目的に応じて次の情報を参照してください。

参照情報

目的	参照情報
最新の Java IDE と Sun ONE Application Server 7 を使った J2EE 開発を経験する	『Sun ONE Studio 4, Enterprise Edition for Java and Application Server Tutorial』を参照
J2EE を詳しく調べる	サンプルアプリケーションを参照
J2EE の設計および開発の最適事例を詳しく調べる	Java BluePrints を調べ、アプリケーションサーバーに付属するサンプルアプリケーション Java Pet Store および Smart Ticket を試す

参照情報 (続き)

目的	参照情報
特定の J2EE 機能が Sun ONE Application Server 7 にどのように実装されているかを詳しく調べる	アプリケーションサーバーのインストールに付属するサンプルアプリケーションを参照
Sun ONE Application Server 7 を使った一般的な開発事例を詳しく調べる	『Sun ONE Application Server 7 開発者ガイド』を参照
Sun ONE Application Server 7 の管理について詳しく調べる	『Sun ONE Application Server 7 管理者ガイド』を参照

索引

N

P

S

U

W

あ

い

う

か

き

く

け

こ

さ

し

す

せ

て

と

は