JRun セットアップ ガイド

JRun 3.0 for Windows®、UNIX、Linux™

Allaire Corporation

版権告知

© 2000, 2001 Allaire Corporation. All rights reserved.

このマニュアルとその中に記載されているソフトウェアは、ライセンス契約のもとに供給され、このライセンスの条項に従ってのみ使用または複製することができます。このマニュアルの内容は、情報の提供のみを目的としており、予告なしに変更することがあります。これについて、Allaire Corporation は一切責任を負いません。Allaire Corporation は、このマニュアルの誤りについて一切責任を負いません。

ライセンスによる許可がある場合を除いて、Allaire Corporation の事前の書面による許可なしに、この出版物の一部または全部の複製、検索システムへの保存、あるいは電子的、機械的な記録、または他のいかなる形態や手段による転送を行うことはできません。

ColdFusion および HomeSite は、連邦政府に認可された Allaire Corporation の登録商標です。 Allaire、Allaire Spectra、JRun Studio、JRun、<CF_Anywhere>、ColdFusion ロゴ、JRun ロゴ、および Allaire ロゴは、Allaire Corporation のアメリカ合衆国内およびその他の国々での商標です。Microsoft、Windows、Windows NT、Windows 95、Microsoft Access および FoxPro は、Microsoft Corporation の登録商標です。Java、JavaBeans、JavaServer、JavaServer Pages、JavaScript、JDK および Solaris は、Sun Microsystems Inc. の商標です。UNIX は、The Open Group の商標です。PostScript は、Adobe Systems Inc. の商標です。その他の製品および製品名は、各社の商標です。

このソフトウェアの著作権の一部は、Merant, Inc. に帰属します。1991-2001

部品番号 : AA-JJSET-RK-SP2, February 2001

目次

序章

始める前に

この章では、JRun のインストールの手順を概説し、インストールに必要なハードウェアおよびソフトウェアの条件を示します。また、JRun および Allaire の Web サイト、マニュアル、テクニカル サポートなどのリソースにアクセスする方法について説明します。

目次

JRun 製品の種類

JRun は、Sun Microsystems 社の最新のサーブレット /JSP および EJB 仕様に対応した Java アプリケーション サーバーです。JRun は、次のバージョンが出荷されています。

- Developer : Web アプリケーションや EJB の開発およびテストなど、非営利の使用を目的とした無料版。ライセンスなしで使用できます。Web の開発および EJB/JMS/JTA がサポートされます。無制限の数の Java Virtual Machines (JVM)、サーブレット および JSP の 3 つまでの同時接続、EJB の 3 つまでの同時接続がサポートされます。
- Professional : プロセッサ単位の価格体系となっており、営利目的で利用することができます。Web の開発のみがサポートされます。JVM 無制限と、サーブレット および JSP の無制限の同時接続がサポートされます。
- Advanced : プロセッサ単位の価格体系となっており、営利目的で利用することができます。Allaire ClusterCATS を使った HTTP ベースのロード バランスおよびフェールオーバなどが含まれています。Web の開発がサポートされ、JRun JDBC ドライバが含まれます。JRun Advanced では、無制限の JVM と、サーブレット、JSP、および EJB の無制限の同時接続がサポートされます。
- Enterprise : プロセッサ単位の価格体系となっており、企業クラスのアプリケーションを開発および使用することができます。Allaire ClusterCATS を使った HTTP ベースのロード バランスおよびフェールオーバなどが含まれています。Web の開発、EJB/JMS/JTA、および JRun JDBC データベース ドライバがサポートされます。また、無制限の JVM と、サーブレット、JSP、および EJB の無制限の同時接続がサポートされます。
- Studio : ライセンスごとの価格設定です。HomeSite HTML エディタをベースとする統合 JSP 開発環境です。JRun サーバーは含まれていません。

価格の最新情報については、国内総販売元である (株) シリウス に問い合わせてください。

JRun をインストールするためシステム 要件

ここでは、JRun をインストールするのに必要なハードウェアおよびソフトウェアの条件を示します。

ハードウェアの必要条件

JRun をフル インストールするには、最低限、次のハードウェアが必要です。

- 32 MB の RAM (64 MB を推奨)
- 20 MB の ハードディスク スペース (50 MB を推奨)

ソフトウェアの必要条件

JRun には次のソフトウェアが必要です (詳細は下記を参照)。

- Windows または UNIX を実行するシステム
- Netscape Communicator または Internet Explorer
- Java Runtime Environment (JRE) 1.1 (EJB、JTA、JMS には JDK バージョン 1.2.2 以降が必要)

オペレーティング システムの必要条件

JRun には、最低限、次のようなオペレーティング システムのバージョンが必要です。JVM および Web サーバーによっては、OS の必要条件をより厳密に指定している場合があります。

- Windows 95/98/NT/2000 (NT には SP 3 以降が必要)
- Solaris 2.6、2.7、8
- Red Hat Linux 6.x
- HP-UX 11.0
- IBM AIX 4.2、4.3
- SGI IRIX 6.5
- Compaq UNIX Tru64 4.0

インターネット ブラウザの必要条件

JRun には、JRun 管理コンソール (JMC)、JRun 環境設定を行う HTML ユーティリティ、および JRun と Web サーバーの接続が含まれています。JMC は Web ベースであるため、次の Web ブラウザのいずれかをインストールしている必要があります。

- Netscape Communicator Version 4.0 以降
- Internet Explorer Version 4.0 以降

Java の必要条件

次の表は、Java サーブレット、JSP ページ、および EJB を開発するために必要な Java ユーティリティの一覧です。表の内容は最小限の必要条件です。ただし、それぞれのユーティリティの最新版の使用をお勧めします。Java ユーティリティのベータ版をシステムで使用することはお勧めしません。

これらのユーティリティは、以下の URL から取得することができます。

http://java.sun.com/products/jdk/1.2/

JRun Java ソフトウェアの必要条件		
Java コンポーネント	**Windows**	**UNIX**
Java Runtime Environment (JRE)	JRE は JRun に含まれていますが、固有の JRE を使用することもできます。バージョンは 1.1.6 以降が必要です。EJB には、JDK を使用する必要があります。	JRE を取得する必要があります (Java JDK に含まれています)。バージョンは 1.1.6 以降が必要です。EJB には、JDK を使用する必要があります。
Java Virtual Machine (JVM)	JVM Version 1.2 が JRun に含まれていますが、Version 1.1.8 以降のバージョンを使用することもできます。	UNIX に対して最適化するには、JVM Version 1.1.8 以降を入手する必要があります。Version 1.2 を推奨します (HP/UX の場合は Version 1.2 が必須です)。
Java サーブレット	Java コンパイラ付属の Java JDK	Java コンパイラ付属の Java JDK
JavaServer Pages (JSP)	追加のソフトウェアをインストールする必要ありません。JRun は必要なツールをすべて含んでいます。	追加のソフトウェアをインストールする必要ありません。JRun は必要なツールをすべて含んでいます。
Enterprise JavaBeans (EJB)	JDK 1.2 以降	JDK 1.2 以降

次の表は、サポートする各プラットフォームの JRun に含まれているユーティリティの一覧表です。

JRun に含まれるユーティリティ			
ユーティリティ	**説明**	**Windows**	**UNIX**
Rhino	オープンソースの JavaScript インタプリタおよびコンパイラ	1.4	1.4
jikes	オープンソースの JavaServer Pages コンパイラ	1.06	1.06

サポートされている JVM

次の表は、JRun がサポートする JVM の一覧表です。最新のアップデートについては、Allaire に問い合わせてください。

サポートされている JVM

JVM	JDK	システム	ベンダの URL
Sun Microsystems 1.1.8	1.1.8	Windows NT/ x86	http://java.sun.com/
Sun Microsystems 1.2.1 & 1.2.2	1.2.2	Windows NT/ x86	http://java.sun.com/
Sun Microsystems 1.3	1.3	Windows NT/ x86	http://java.sun.com/
Sun Java HotSpot 1.0.1	1.2.2	Windows NT/ x86	http://java.sun.com/
Microsoft VM 2.02	1.1	Windows NT/ x86	http://www.microsoft.com/
Microsoft VM 4,0	1.1.4	Windows NT/ x86	http://www.microsoft.com/
IBM JDK 1.1.8	1.1.8	Windows NT/ x86	http://www.ibm.com/
IBM 1.1.8	1.1.8	Linux/i686	http://www.ibm.com/
Blackdown 1.1.7Bv3	1.1.7B	Linux/x86	http://java.blackdown.org/java-linux.html
Blackdown 1.2.2 rc3, 4	1.2.2	Linux/i386	http://java.sun.com/
Sun/Borland 1.2.2 RC2	1.2.2 RC2	Linux/i386	http://java.sun.com/
Sun 1.10.6	1.1.6	Solaris/Sparc	http://www.sun.com
Sun 1.2.1	1.2.1	SunOS/Sparc	http://www.sun.com
Sun 1.2.1_0	1.2.1	SunOS/Sparc	http://www.sun.com
Sun 1.2.1_04 ベータ版	1.2.1	SunOS/Sparc	http://www.sun.com

Web サーバーの必要条件

JRun を外部 Web サーバーに接続するのは、一般的なタスクです。Web サーバーは、JRun により次のプラットフォーム上でサポートされています。

Web サーバーの接続にサポートされているプラットフォーム								
プラットフォーム	**PWS**	**IIS 3.0/ 4.0**	**IIS 5.0**	**Apache 1.2.x/ 1.3.x**	**Fast Track 3.x**	**Netscape 2.x, 3.5, 3.6, 4.x (iPlanet)**	**Zeus**	**O'Reilly WebSite Pro 2.x, 3.0**
NT 4.0 Server/ Wkstn		X		X	X	X		X
Win 2K Pro/ Server			X	X	X	X		X
Win 95/98	X							X
RedHat Linux 6.x				X		X (4.x のみ)	X	
Solaris 2.6/ 2.7, 8				X	X	X	X	
HP/UX 11.0*				X	X	X	X	
SGI/IRIX 6.5				X	X	X	X	
AIX 4.2/4.3*				X	X	X	X	
Compaq UNIX Tru64 4.0*				X	X	X	X	

* JDK 1.2 をサポートする IRIX、AIX、HP/UX、Digital UNIX のすべてのバージョンがサポートされています。

JRun の旧バージョンからのアップグレード

JRun 3.0 にアップグレードすると EJB のサポート、サーブレット API のアップグレード、機能強化された GUI 管理ツールなど、多くの追加機能を利用することができます。これらの追加機能を完全に活用するためには、準備が必要な場合もあります。

メモ ベータ版の上に JRun 3.0 をインストールする場合には、インストールを続行する前に、ベータ版をアンインストールしてください。

このセクションでは JRun の旧バージョンから 3.0 にアップグレードする場合の問題点について説明します。

JSP、EJB、およびサーブレット仕様の以前のバージョンとの違いについては、Sun の仕様を参照してください。

JRun 2.3.x と 3.0 を同時に実行する

デフォルトでは、JRun のインストール スクリプトは JRun を C:\Program Files\Allaire\JRun (Windows) および /opt/jrun (UNIX) にインストールします。JRun の旧バージョンからアップグレードする場合、はじめにすべての JRun サーバーを停止しなければなりません。次の手順のいずれかを行います。

- 既存の JRun ディレクトリを新しい場所へ移動する。
- JRun の新しいバージョンを新しい場所へ移動する。

既存の JRun をアンインストールしない場合は、JRun を Windows 95/98/NT にインストールする前に、すべての JRun および Java のプロセスを中止します。この処理を行わないと、JRun Web サーバーの構成が正しく行われないことがあります。

メモ JRun 3.0 と 2.3.x を同じマシンにインストールすることはお勧めしません。

Windows NT で実行中のすべての JRun プロセスを中止するには

1. Control + ALT + Delete を押します。[Windows NT のセキュリティ] ウィンドウが表示されます。
2. [タスク マネージャ] をクリックします。Windows NT タスク マネージャが起動します。
3. [プロセス] タブを選択します。
4. [イメージ名] を選択します。
5. 次のプロセスを中止します。

   javaw.exe
   jrun.exe

管理

インタフェース

JRun を構成するために、JRun 2.3.x で使用していた Swing ベースの管理ユーティリティの代わりに、ブラウザ ベースのユーティリティ、JRun 管理コンソール (JMC) を使用します。JMC の使用方法については、第 3 章を参照してください。

プロパティ ファイル

JMC は JRun の初期設定および構成の値をプロパティ ファイルに保存します。デフォルトのインストールについて作成されるプロパティ ファイルの数が 75 以上から 10 未満に減りました。JRun プロパティ ファイルの詳細については、第 5 章を参照してください。

Web アプリケーション

サーブレットのバージョン 2.2 仕様を導入すると、Web アプリケーションおよび .war ファイルという概念が採用されます。Web アプリケーションの中の .class およびそれをサポートするファイルが、この仕様によって指定されているディレクトリ階層に公開されます。Web アプリケーションに含まれない個別のサーブレットも引き続きサポートされます。これらのサーブレットは </jrun_rootdirectory>/servlet ディレクトリに置かれます。デフォルトの Web アプリケーションは、その classpath の中にこのディレクトリを含みます。

縮小または廃止された機能

JRun 3.0 では Web アプリケーションから Enterprise JavaBeans まで、最新の仕様を導入しています。しかし、廃止または段階的に縮小されている機能もあります。ここでは、これらの機能について説明します。

CF_Anywhere

JRun では引き続き Allaire の ColdFusion markup language (CFML) のサブセットを使用するファイルを処理することができます。しかし、この機能は JRun のこれ以降のリリースではサポートされません。詳細については、『JRun Developer Center』を参照してください。

Server Side Includes (SSI)

SSI は、ダイナミックなコンテンツを作成するために以前は広く使用されていたので、JRun では主に古いインプリメンテーションをサポートするためにこの機能を使用します。現在では、JavaServer Pages (JSP) および Java サーブレット技術が SSI の代わりに用いられるようになり、機能的にも大幅に拡張されています。

Active Server Pages

JRun は ASP ページのサポートを廃止しました。

インストールの概要

ここでは、JRun のインストールおよび構成の基本的な手順を説明します。この手順は使用する Web サーバー、Web サーバーのバージョン、および Web サーバーのプラットフォームによって異なります。

JRun のインストールおよび構成の基本的な手順を以下の表に示します。

インストールの手順	
ステップ	**章**
1. JRun のインストール	**第 1 章**
2. JRun　が正常にインストールされていることを確認します。	**第 1 章**
3. Web サーバーが JRun と通信できるように構成します。	**第 2 章**
4. Web サーバーと JRun が通信していることを確認します。	**第 2 章**
5. JRun 管理コンソール (JMC) を使用して、JRun 構成を追加します。	**第 3 章**

配布された環境で JRun をインストールする場合、第 4 章を参照してください。

Java 製品の概要

ここでは、主要な Java 製品の最新バージョンの概要を示します。詳細については Sun の Web サイト http://java.sun.com を参照してください。

Java Platform

Java Platform は、Java 環境のアーキテクチャを定義します。Java 2 Platform には次のような 3 つのエディションがあります。

- Java 2 Platform, Standard Edition (J2SE)
- Java 2 Platform, Enterprise Edition (J2EE)
- Java 2 Platform, Micro Edition (J2ME)

Java 2 Platforms は次の Java Software Development Kit によって実装されます。

Java Software Development Kit

Java Software Developer Kit (SDK) は、Java Development Kit (JDK) とも呼ばれることも多くあります。これは Java Runtime Environment (JRE) のほかに、開発者が Java プラットフォーム向けのコンパイル、デバッグ、アプリケーション実行を行うのに使用するツールおよびコア クラスから成っています。Windows システムでは、JRE は SDK に含まれます。UNIX では、JRE は同じダウンロード ファイルには含まれません。SDK は使用許諾契約ごとに配布されるものではありません。

主なコンポーネント

- コンパイラおよびデバッガ
- Java Runtime Environment (JRE)
- Win32 パフォーマンス パック (オプション)
- Solaris ネーティブ スレッド パック (オプション)

バージョン

- JDK 1.0.x
- JDK 1.1
- J2 SDK 1.2.2 Standard Edition (Java 2 SDK ともいいます)
- J2 SDK 1.2.2 Enterprise Edition (Java 2 SDK ともいいます)

J2 SDK Enterprise Edition は、JSP、EJB、サーブレット などの高度なサービス向けに、SDK のサポートを追加しました。

Java Runtime Environment

Java 2 Runtime Environment (J2 JRE) は Java Virtual Machine (JVM) 仕様のインプリメンテーションであり、サポートする一連のクラスが付属しています。これには Java 2 プラットフォームのために作成されたプログラムを実行するために必要なすべての機能が含まれています。SDK とは異なり、開発者は使用許諾契約にもとづいて JRE を自由に配布することができます。

主なコンポーネント

- Java Virtual Machine (JVM)
- Java アプリケーション ランチャ
- ランタイム クラス ライブラリ
- Java プラグイン (ブラウザ用)
- Symantec JIT Compiler (Windows) または Sun JIT (UNIX)

バージョン

- JRE 1.1
- JRE 1.2.2

JVM はソフトウェアによる CPU のインプリメンテーションであり、コンパイルされた Java コードを実行するために設計されています。HP、Sun、Microsoft、Symantec をはじめとする多くの企業が独自の JVM を開発しています。Java Runtime Environment (JRE) という用語は、Sun の JVM 実装の Sun 固有の名前です。JVM を JRE と呼ぶベンダも多くあります。このマニュアルでは、JRE と JVM は同じものとして使用します。JRun でサポートされる JVM のリストは、5 ページの 「サポートされている JVM」にあります。

拡張サービス

Java は拡張可能な言語で、継続的に機能を広げています。ここでは、JRun がサポートするいくつかの拡張機能について説明します。JRun をインストールする場合、個々のコンポーネントについてインストールするかどうかを選択することができます。

サーブレット

サーブレットはダイナミック コンテンツを生成する Java Web コンポーネントです。JRun 3.0 は Sun の サーブレット仕様 2.2 に適合します。この仕様は 2.1 をもとに確立され、Web アプリケーションおよび Web アプリケーション アーカイブ (WAR) のサポートを含みます。JRun に サーブレット 2.2 を実装するには、JRE 1.1 以降が必要です。

JavaServer Pages (JSP)

Java ServerPages (JSP) は Java サーブレット API の拡張です。Java コードと HTML を組み合わせることにより、ダイナミックな Web ページを作成します。JRun 3.0 は、Sun の JSP 仕様 1.1 をサポートします。この仕様は 1.0 に基づいており、タグ拡張およびコンテナへのサポートを含んでいます。JRun に JSP 1.1 を実装するには、JRE 1.1 以降が必要です。

Enterprise JavaBeans

Enterprise JavaBeans (EJB) は、J2EE プラットフォームのためのソフトウェア アーキテクチャに基づいた、サーバー側にある分散型のコンポーネントです。JRun は Sun の Enterprise JavaBeans 1.1 仕様をサポートします。EJB 1.1 仕様では、1.0 の仕様に JTA や JMS など、開発および配布のための機能が追加されました。JRun に EJB を実装するには、JRE 1.2.2 以降が必要です。

開発者のためのリソース

(株) シリウスは、開発者の教育、ドキュメンテーション、テクニカル サポート、およびプロフェッショナル サービスのカスタマ サポートの標準を確立することを目指しています。弊社の Web サイトは、すべてのオンライン リソースにすばやくアクセスできるように設計されています。以下の表は、これらのリソースの場所を示しています。

Allaire のデベロッパ サービス	
リソース	**説明**
(株) シリウスのサイト www.sirius.co.jp	Allaire 社製品の国内総販売元。Allaire 製品に関する情報やパートナー、サポートなどの情報を掲載。尚、トレーニングやセミナーの申込を行なうことができる。
(株) シリウス JRun のサイト cfusion.sirius.co.jp/jrun/	JRun の詳細な製品情報および関連トピック
Allaire の Web サイト www.allaire.com	Allaire の製品およびサービスに関する一般的情報
JRun に関する情報 www.allaire.com/products/jrun/	JRun の詳細な製品情報および関連トピック
デベロッパ コミュニティ www.allaire.com/developer	オンライン ディスカッション グループ、Component Exchange、Resource Library、テクニカル ペーパーなど、JRun 開発の最先端を保持するために必要なすべてのリソース
JRun Dev Center www.allaire.com/developer/jrunreferencedesk/	サーブレット リソース、開発のヒント、記事、マニュアル、ホワイト ペーパーなどの便利な情報リソース
JRun サポート フォーラム forums.allaire.com/jrunconf	Allaire オンライン フォーラムに参加すると、経験豊富な JRun 開発者に質問したり、JRun に関するさまざまな問題についてメッセージを書き込み、回答を得ることができます。

JRun Documentation について

JRun のドキュメント セットには、以下のドキュメントが含まれます。

- リリース ノート
- JRun セットアップ ガイド

- JRun によるアプリケーション開発
- JRun サンプル ガイド
- Allaire ClusterCATS の使用方法 (JRun Enterprise に付属)
- JRun Studio 入門 (JRun Studio に付属)
- JRun 拡張設定ガイド (Web サイトから入手可能)
- JRun JBDC Drivers User's Guide and Reference

オンライン マニュアル

JRun のマニュアルはすべて、Adobe Acrobat (PDF) 形式のファイルとしてオンラインで提供されています。PDF 形式のファイルは JRun CD に含まれており、デフォルトでは JRun /docs ディレクトリにインストールされます。これらのファイルにアクセスするには、JRun 管理コンソール の [ようこそ] の画面で [製品マニュアル] のリンクをクリックします。

Adobe Acrobat ファイルは、Allaire の Web サイト www.allaire.com/documents からダウンロードすることができます。

マニュアルの表記規則

マニュアルをお読みになるときは、以下の表記規則に注意してください。

- 番号を付けられたステップは、手順を示します。
- コード例、ファイル名、および URL は、monospaced フォントで示します。
- 注記およびヒントは、**太字**で示します。
- 黒丸はオプションおよび機能のリストを示します。
- メニュー レベルは > 記号で区切り ます。

その他のリソース

このマニュアルで扱っている項目に関する詳細情報については、次のリソースを参照してください。

書籍

- 『Java Servlets』Karl Moss 著。 1999 年、McGraw Hill 発行。ISBN: 0071351884
- 『Core Servlets and JavaServer Pages』Marty Hall 著。2000 年、Prentice Hall 発行。ISBM: 0130893404
- 『Java Servlet:By Example』 Alan R. Williamson 著。1998 年、Manning Publications 発行。ISBN: 188477766X

- 『Java Servlet Programming』Jason Hunter、William Crawford 著。1998 年、O'Reilly & Associates 発行。ISBN: 156592391X
- 『Developing Java Servlets』James Goodwill 著。1999 年、Sams 発行。ISBN: 0672316005
- 『Inside Servlets:Server-Side Programming for the Java Platform』Dustin R. Callaway 著。1999 年、Addison-Wesley Pub. Co. 発行。ISBN: 0201379635
- 『Professional Java Server Programming』1999 年、Wrox Press Ltd. 発行。ISBN: 1861002777
- 『Mastering Enterprise JavaBeans and the Java 2 Platform, Enterprise Edition』Ed Roman 著。Wiley 発行。ISBN: 0471332291
- 『Enterprise JavaBeans』Richard Monson-Haefel 著。O'Reilly & Associates 発行。ISBN: 1565928695

オンライン リソース

- Java servlet API (http://java.sun.com/products/servlet)
- JavaServer Pages (http://java.sun.com/products/jsp)
- Servlet Source (http://www.servletsource.com)
- JSP Resource Index (http://www.jspin.com)
- ServerPages.com (http://www.serverpages.com)
- Enterprise JavaBeans (http://java.sun.com/products/ejb/)

JRun 関連のコンタクト先

テクニカル サポート

Allaire では、電話および Web ベースの種々のサポート オプションを提供しております。テクニカル サポート サービスの詳細については、http://www.allaire.com/support/ をご覧ください。

JRun Support Forum (http://forums.allaire.com) へは、いつでも投稿できます。

国内総販売元

(株) シリウス

電話 : 03-5562-4099
Fax: 03-5562-4070

http://www.sirius.co.jp/
E-mail:jrun@sirius.co.jp

第 1 章

JRun のインストール

この章では、JRun のインストール方法について説明します。この章で説明する手順を完了したら、第 2 章の説明に従って Web サーバーを JRun と通信できるように構成する必要があります。

目次

JRun のインストール

インストールでは、JRun アプリケーションをデフォルトで (無効にできます) ホストする 2 つの JRun Web Servers (JWS) がインストールされることにご注意ください。これらの JRun アプリケーションには、JRun Management Console (JMC) およびデモ アプリケーションがあります。これらのサーバーに割り当てられたポートがあることを確認する必要があります。デフォルトは、8000 (`admin` サーバー) および 8100 (`default` サーバー) です。

ここでは、JRun を次のシステムにインストールする方法について説明します :

- Windows 95/98/NT/2000 へのインストール 16 ページ
- UNIX および Linux へのインストール 25 ページ

Windows 95/98/NT/2000 へのインストール

ここでは、Windows 95/98/NT システムに JRun をインストールする方法について説明します。

JRun をインストールするには

1. JRun を Web サーバーに接続する場合には、Web サーバーを停止します。
2. 現在実行中の Windows アプリケーションをすべて終了します。
3. JRun インストール ファイル `setup.exe` を実行します。

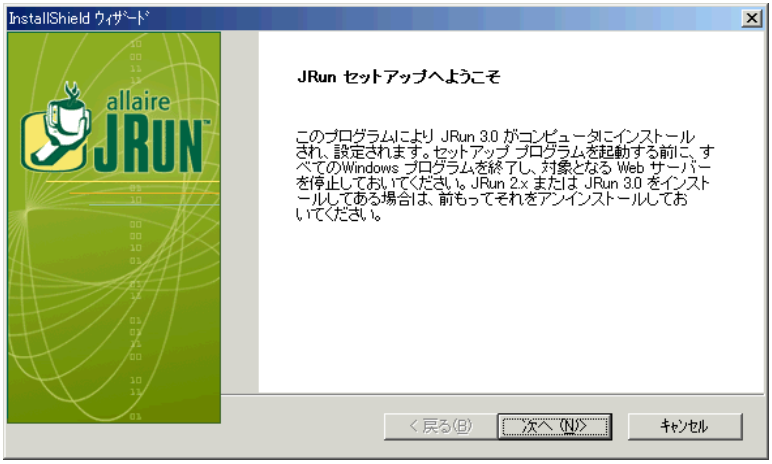
   JRun スプラッシュ画面が表示されます。

4. [インストール] セクションで、インストールするバージョンの JRun 3.0 をクリックします。

   メモ Enterprise JavaBeans (EJB) コンポーネントを使用するには、JDK 1.2.2 （単に JRE と異なる）以降が必要です。Sun JDK は、`http://java.sun.com` `http://java.sun.com/ /a` から入手してください。

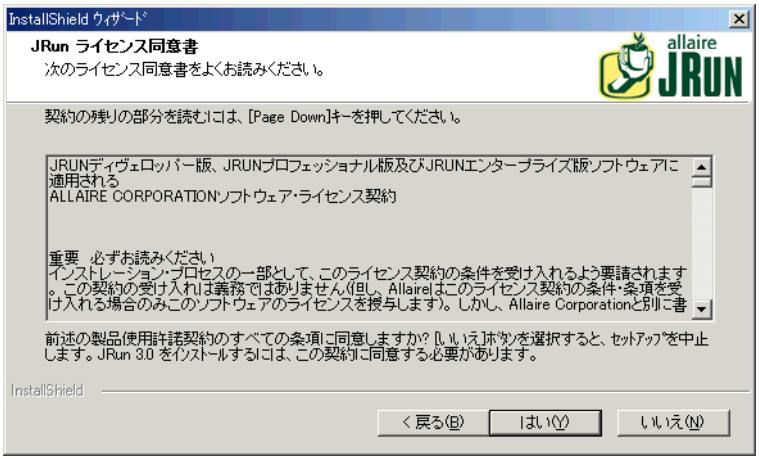
   JRun をインストールするには、JRE が必要です。インストールしていない場合には、[インストール] セクションの Java をクリックしてください。Sun J2 JRE バージョン 1.2.2 がインストールされます。その後、JRun インストールを再開します。

   [JRun セットアップへようこそ] ウィンドウが表示されます。



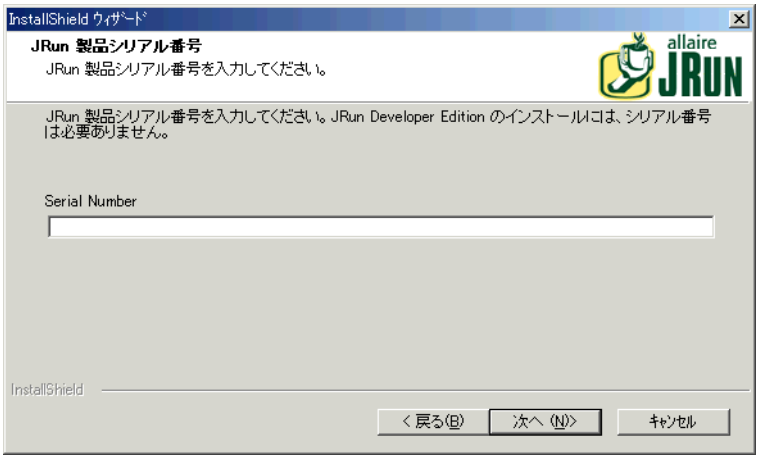
5. [次へ] をクリックします。

   [JRun ライセンス同意書] ウィンドウが表示されます。



6. JRun ライセンス契約に同意する場合は [はい] を、インストールを中止する場合は [いいえ] をクリックします。

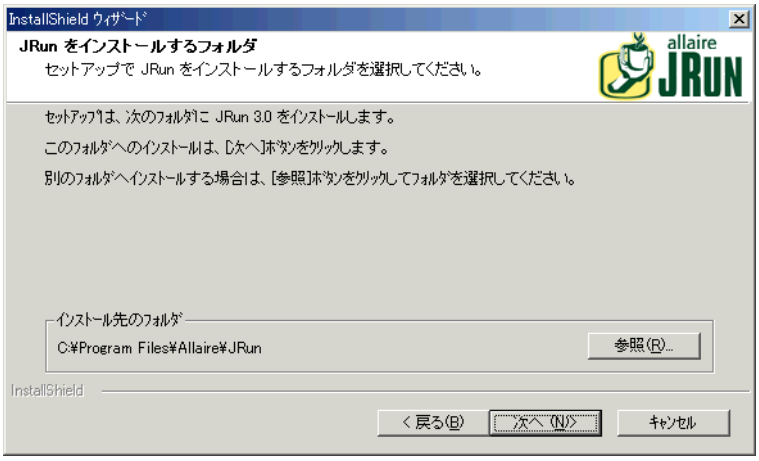
   [JRun 製品シリアル番号] ウィンドウが表示されます。



7. Allaire から提供されたシリアル番号を正確に入力して、[次へ] をクリックします。

   JRun Developers 版または評価版をインストールする場合は、このフィールドは空欄 (デフォルト) にします。JRun の旧バージョンからアップグレードする場合は、新しいシリアル番号を入力します。その後、旧 JRun バージョン 2.*x* のライセンスキーを入力するよう求められます。

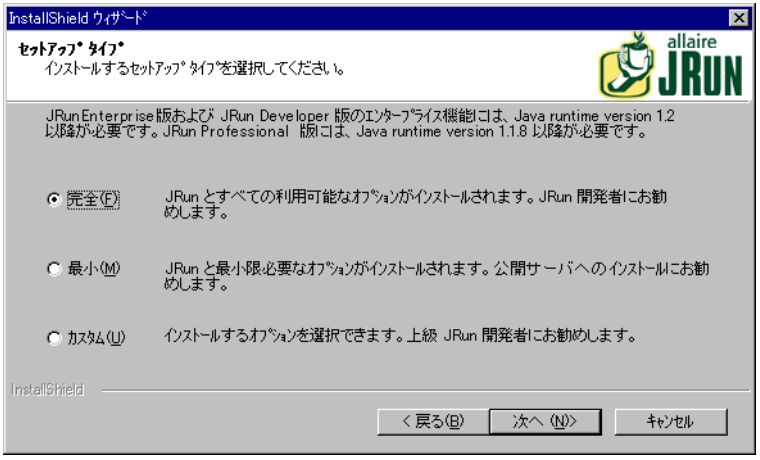
[JRun をインストールするフォルダ] ウィンドウが表示されます。



8. JRun をインストールするフォルダを選択し、[次へ] をクリックします。

メモ このマニュアルでは、このフォルダを <JRun_directory> と呼びます。

[セットアップタイプ] ウィンドウが表示されます。



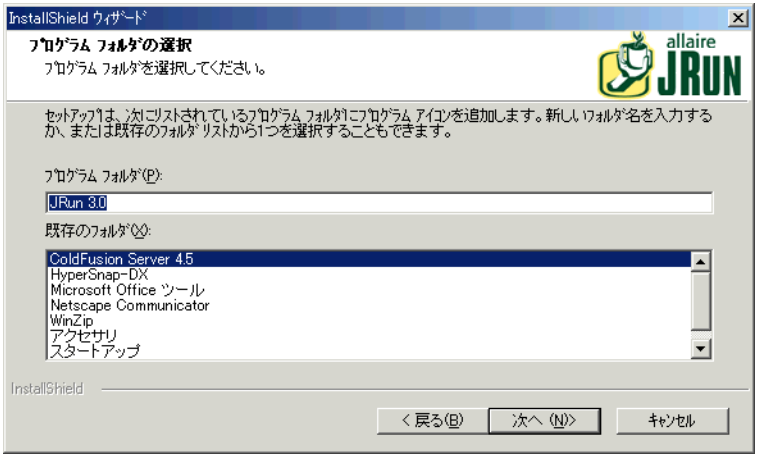
9. インストールの種類を選択し、[次へ] をクリックします。以下の表では、これらのオプションについて説明しています。

カスタム コンポーネントの選択	
オプション	**説明**
フル	使用可能なすべてのオプションをインストールします。サーブレット、JSP、EJB、JMS、JTA サポート、サンプル、およびマニュアルがインストールされます。一般的な JRun 開発者はこのオプションを選択することをお勧めします。
最小	最低限の必須オプションをインストールします。このインストールには サーブレット、JSP、EJB、および JMS サポートを含みます。マニュアルおよびサンプルはインストールされません。アプリケーションを展開するサーバーにインストールする場合は、このオプションを選択することをお勧めします。
カスタム	インストールするオプションを独自に選択することができます。経験豊富な JRun 開発者は、このオプションを選択することをお勧めします。

[カスタム] をクリックすると、[コンポーネントの選択] ウィンドウが表示されます。JRun Professional をご使用の場合は、EJB コンポーネントは利用できません。

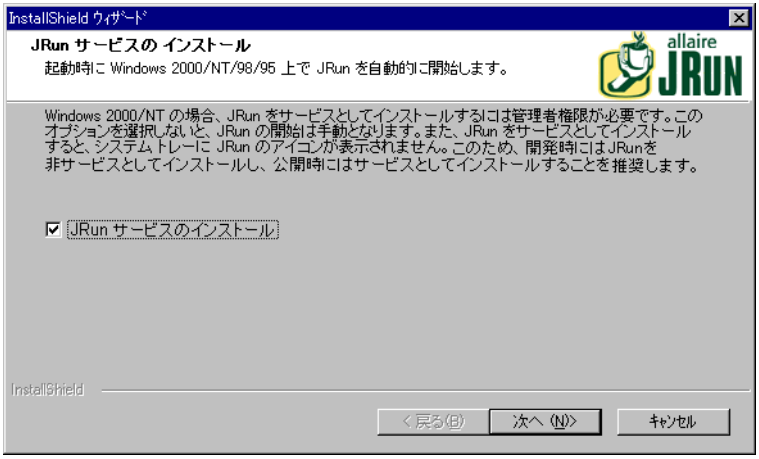
10. インストールする JRun コンポーネントを選択して、[次へ] をクリックします。

[プログラム フォルダの選択] ウィンドウが表示されます。



11. JRun をインストールするプログラム フォルダ名を選択し、[次へ] をクリックします。

    JRun は指定したファイルをインストールします。[JRun サービスをインストール] ウィンドウが表示されます。

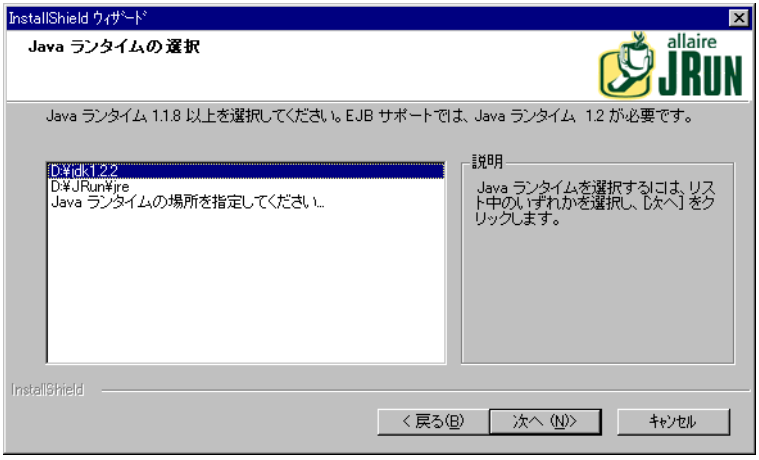


12. JRun サーバーを自動的に起動するかどうかを選択して、[次へ] をクリックします。

    [JRun サービスをインストール] を選択した場合には、マシンを起動したときに必ず JRun Admin Server と JRun default Server が起動します。NT では、これらのサーバーは NT サービスとしてインストールされるため、ユーザ プロセスではなく、シ

ステム プロセスとして実行されます。Windows 95/98 では、これらのサーバーは、Windows レジストリで参照することが可能で、再起動すると自動的に起動します。[JRun サービスをインストール] を選択しない場合には、JRun サーバーはアプリケーションとして実行され、手動で起動する必要があります。

[Java ランタイムの選択] ウィンドウが表示されます。

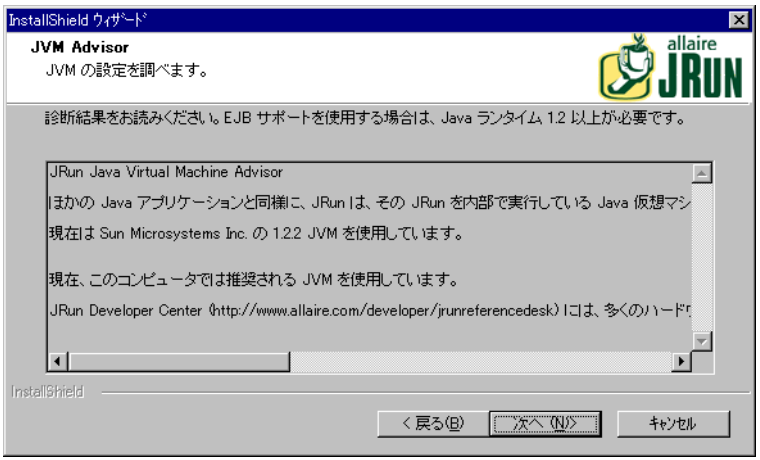


13. Java の実行時環境を選択し、[次へ] をクリックします。

メモ JRun の Enterprise JavaBeans コンポーネントを使用するには、JRE バージョン 1.2.2 以降を選択する必要があります。

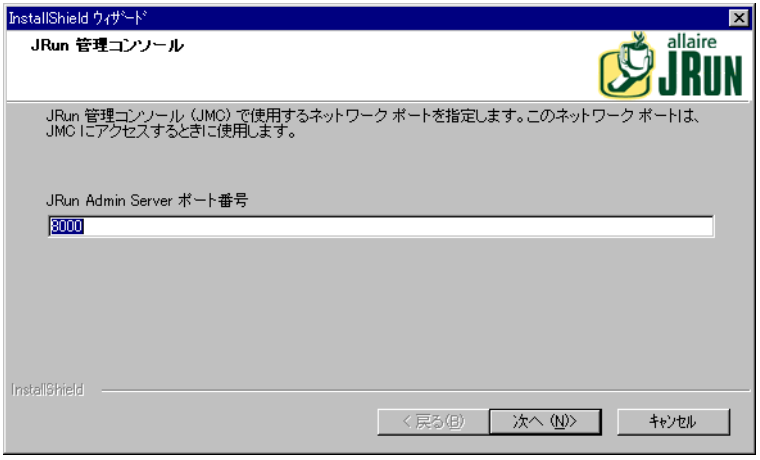
Sun の Java Runtime Environment (JRE) は JRun の Windows バージョンに含まれているので、別途 JRE を用意する必要はありません。JRE をインストールする場合は、いったんインストールをキャンセルし、JRun スプラッシュ画面で [Java Runtime Environment 1.2.2] のリンクをクリックします。JRE をインストールした後、スプラッシュ画面で ふたたび [インストール] で JRun を選択します。

JVM Advisor が表示されます。



14. JVM Advisor 内の情報が正しいかどうかを確認して、[次へ] をクリックします。

    [JRun 管理コンソール] の管理ポートウィンドウが表示されます。

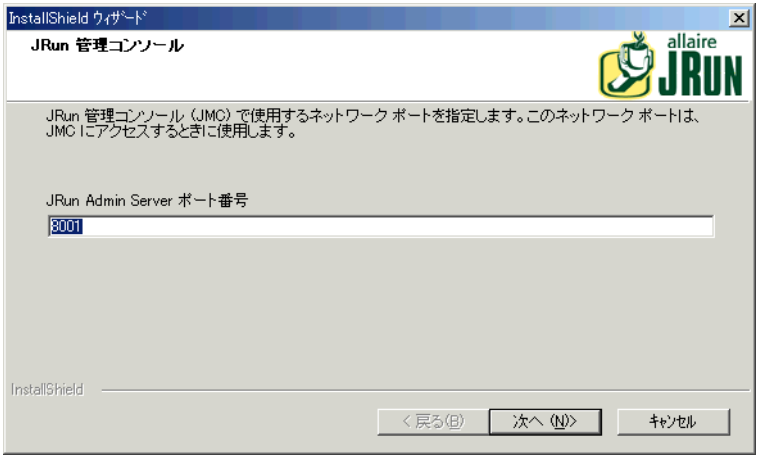


15. JRun Web サーバー上にある JRun の管理 Web アプリケーションへのアクセスに使用する一意のポート番号を入力し、[次へ] をクリックします。

    JRun Web サーバー (JWS) はこのポートで受信して、JRun 管理コンソール (JMC) へのアクセスを行います。既定のポート番号は 8000 です。

メモ 8100 から 8199 までのポート番号は選択しないでください。この範囲のポートは、JRun によって、default JRun サーバーの JWS に使用されます。

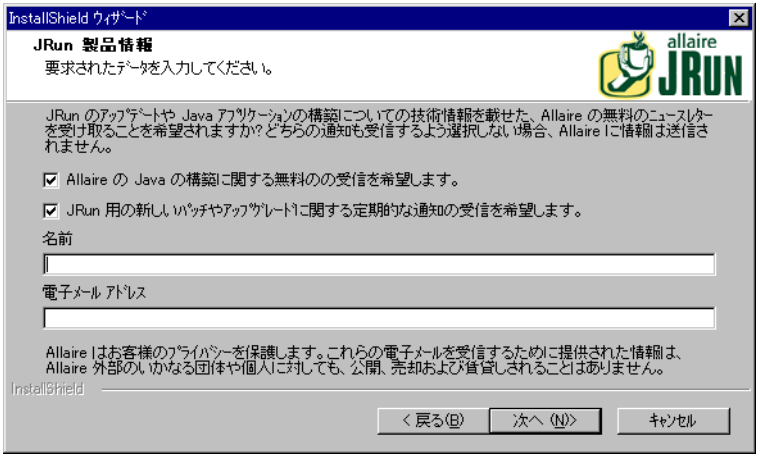
[JRun 管理コンソール] のパスワード ウィンドウが表示されます。



16. JRun 管理者のパスワード (admin) を入力および確認し、[次へ] をクリックします。

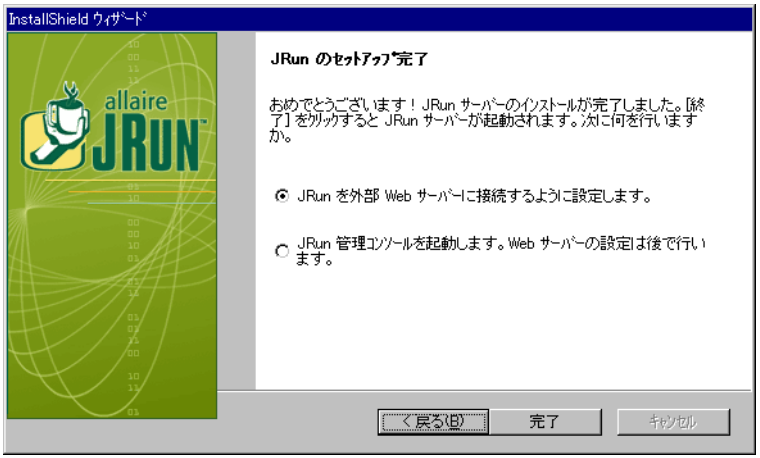
メモ パスワードにはスペースおよびアスタリスク (*) は使用できません。

JRun インストーラで必要なディレクトリの作成およびシステム ファイルの設定が完了すると、[製品情報] 画面が表示されます。



17. オプションで、表示されたフィールドに、お客様の名前と電子メール アドレスを入力します。Java アプリケーション開発に関する情報や JRun に関するメールを受信したい場合には、各チェックボックスを選択し、[次へ] をクリックします。

    [セットアップの完了] ウィンドウが表示されます。



18. JRun と外部 Web サーバー (Apache や IIS など) との接続を設定するには、最初のラジオ ボタンを選択し、[完了] をクリックします。JRun 管理コンソールが起動し、ログインするよう求めます。ログインすると、コネクタ ウィザードが表示されます。

    ログインおよび構成の終了については、27 ページの「 JRun 管理コンソールの起動 」を参照してください。

    外部 Web サーバーを後から設定するには、2 つめのラジオ ボタンを選択し、[次へ] をクリックします。[JRun 管理コンソール] が開きます。

UNIX および Linux へのインストール

ここでは、JRun を UNIX/Linux システムにインストールする方法について説明します。

JRun をインストールするには

1. Web サーバーを停止します。JRun を Web サーバーに接続するには、この作業が必要です。
2. 目的の Java Runtime Environment (JRE) がマシンにインストールされていることを確認してください。JSP/ サーブレットをサポートするには、JRE 1.1.6 以降が必要です。EJB をサポートするには、JRE 1.2 以降が必要です。Sun の JRE は、以下の Web サイトから取得することができます。

    http://java.sun.com

3. 次のコマンドを使用して、jr302u.sh ファイル、JRun インストール シェル スクリプトの実行許可を設定します :

   chmod 755 jr302u.sh

jr30h.sh	HP/UX
jr30l.sh	Linux
jr30s.sh	Solaris
jr30x.sh	Irix
jr30i.sh	AIX
jr30o.sh	Tru64
jr30g.sh	すべての UNIX プラットフォーム

4. 次のコマンドを使って JRun インストール スクリプトを実行します。:

   /bin/sh ./jr302u.sh

   ライセンス契約を読むよう求められます。

5. Enter を押して、ライセンス契約の各ページを表示します。

   ライセンス契約に同意するよう求められます。

6. 契約に同意する場合は [y] 、インストールを中止する場合は [n] を入力します。

   インストール先のディレクトリを入力するよう求められます。

7. JRun をインストールするディレクトリを入力します。このマニュアルでは、このディレクトリを <JRun_directory> と呼びます。既定値は、/opt/JRun です。

   実行するインストールの種類を選択するよう求められます。[Typical] または [Custom] のどちらかを選択できます。

   [Typical] を選択すると、すべてのコンポーネントがインストールされます。[Custom] を選択すると、次のオプションから選択するよう求められます。

   1. Servlet and Java ServerPages
   2. Enterprise Java Beans and Java Message Service
   3. All

8. インストールの種類を入力します。

   選択したインストールに必要なファイルがすべて解凍およびコピーされます。その後 JRE または JDK ディレクトリへの絶対パスを入力するよう求められます。

9. JRE/JDK の場所を指定します。通常、JRE/JDK は /usr/java にインストールされますが、システムによっては別の場所にインストールされる場合があります。

メモ JRE/JDK の 1.2 より前のバージョンを選択すると、JRun の Enterprise JavaBeans コンポーネントが正常に動作しません。

ライセンス キーを入力するよう求められます。

10. Allaire から提供されたライセンス キーを正確に入力します。

    JRun Developer または評価版をインストールする場合は、このフィールドは空欄 (デフォルト) にします。JRun の旧バージョンからアップグレードする場合は、新しいアップグレード キーを入力します。その後、旧 2.*x* のライセンス キーを入力するよう求められます。

    JRun 管理者のパスワードを入力するよう求められます。

11. パスワードを入力します。パスワードにはスペースおよびアスタリスク (*) は使用できません。

    ポート番号を入力するよう求められます。

12. JRun Web サーバー上にある JRun の管理 Web アプリケーションへのアクセスに使用する一意のポート番号を入力します。JRun Web サーバー (JWS) はこのポートで受信して、JRun 管理コンソール (JMC) へのアクセスを行います。既定のポート番号は 8000 です。

    メモ 8100 から 8199 までのポート番号は入力しないでください。この範囲のポートは、JRun によって、default JRun サーバーの JWS に使用されます。

    Java アプリケーション開発に関する情報や JRun に関する通知を受信するかどうかを指定するよう、求められます。

13. これらの情報を受信するかどうかを選択します。

    [yes] と入力すると、お客様の名前と電子メール アドレスを入力するよう求められます。

14. 名前と電子メール アドレスを入力します。

    JMC URL またはブラウザの demo URL を開くよう求められます。

    構成を継続し、JRun を外部 Web サーバーに接続するには、JMC の URL を開きます。設定の終了方法については、27 ページの「JRun 管理コンソールの起動」を参照してください。

    JMC を起動すれば、随時 JRun インプリメンテーションを構成することができます。詳細については、86 ページの「JRun 管理コンソール の開始」を参照してください。

JRun 管理コンソールの起動

JRun 管理コンソール (JMC) はブラウザ ベースのインターフェイスを持つ Web アプリケーションで、JRun の設定に使用します。JMC を使用するには、Netscape Communicator Version 4.0 以降、または Internet Explorer Version 4.0 以降が必要です。

メモ この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

JMC を起動するには

1. UNIX および Windows では、次の方法で JMC を起動することができます。

   - Web ブラウザで以下の URL を開きます :

     http://localhost:8000

   また、**Windows の場合のみ、**次の操作のいずれかを行います :

   - [**スタート**] > [**プログラム**] > [JRun 3.0] > [JRun **管理コンソール**] をクリックします。
   - システム トレイで [JRun] アイコンをダブルクリックし、[実行] をクリックします (JRun をアプリケーションとしてインストールした場合)。
   - *<JRun_directory>*/bin ディレクトリで次の DOS コマンドを入力します :

     jrun –admin

   JMC を初めて起動した場合は、JRun 使用許諾契約書が表示されます。

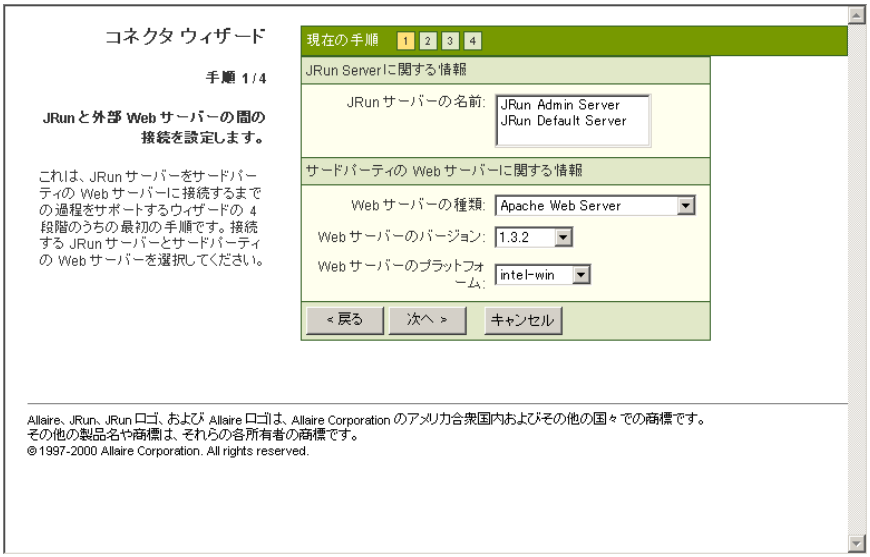
   JMC が表示されない場合は、37 ページの「トラブルシューティング」を参照してください。JRun のコマンドライン オプションの詳細については、第 3 章を参照してください。

2. JRun 使用許諾契約に同意します。使用許諾契約に 1 度同意すれば、2 度目からは表示されません。

   JMC のログインウィンドウが表示されます。

3. 表示されたフィールドにユーザ名とパスワードを入力し、[ログイン] をクリックします。規定のユーザ名は admin です。パスワードは、インストール手順で admin に対して作成したものを入力します。

   インストール実行中に JMC を起動した場合は、JRun コネクタ ウィザードが表示されます :



   セットアップの終了については、第 2 章 の「Web サーバーの設定」のセクションを参照してください :

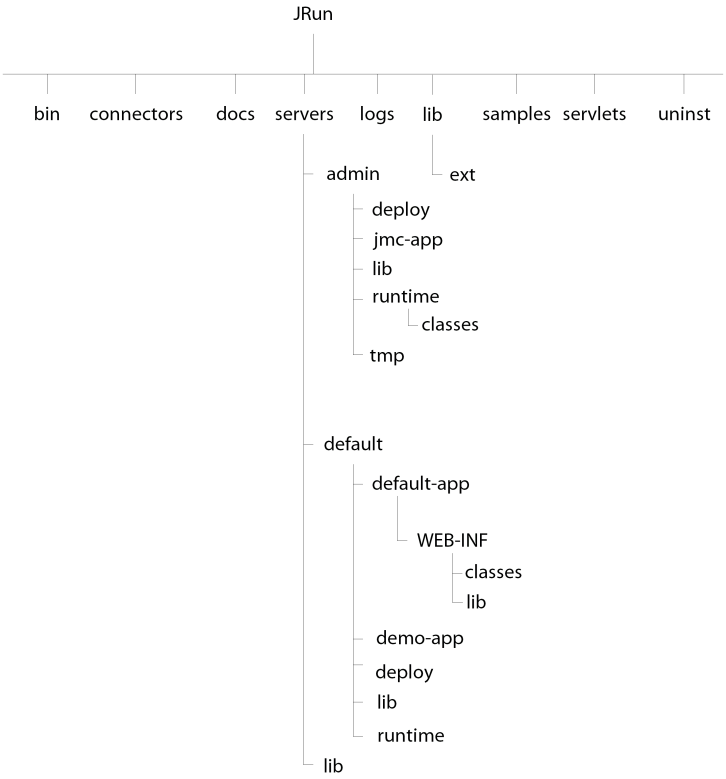
   - 43 ページの「 Apache の構成」
   - 48 ページの「 IIS 3.0/PWS の構成」
   - 52 ページの「 IIS 4.0/5.0 の構成」
   - 60 ページの「 Netscape/iPlanet の構成」
   - 68 ページの「 WebSite Pro の構成」
   - 76 ページの「 Java ベースの Web サーバーの構成」
   - 77 ページの「 Zeus Web サーバーの構成」

   インストール後に JMC を起動すると、JMC のメイン ウィンドウが [JRun クイックスタート プロダクト ツアー] ウィンドウとともに表示されます。

JMC を使った JRun の設定方法については、第 3 章 を参照してください。

JRun のディレクトリ構造

次の図は、JRun のディレクトリ構造を示しています :



デフォルトでは、/JRun ディレクトリは c:\Program Files\Allaire (Windows) または /opt (UNIX/Linux) の下に作成されます。

/JRun の内容は、次の表に示しています。ディレクトリ構造はインプリメンテーションによって異なるため、すべてのディレクトリおよびサブディレクトリが表示されているわけではありません。

JRun のサブディレクトリ

次の表は、/JRun ディレクトリ内のサブディレクトリを示しています。

ディレクトリ	説明
/bin	JRun の実行ファイルが含まれています。
/connectors	Web サーバーのコネクタ ファイルが含まれています。
/docs	JRun および Java サーブレット API の HTML 文書が含まれています。
/lib	すべての JRun アプリケーションの標準プロパティを定義する .jar ファイル、およびプロパティ ファイルが含まれています。
/lib/ext	servlet.jar や ejb.jar などの .jar ファイルを格納します。
/logs	JRun のログ ファイルが含まれています。
/samples	JRun のサンプル ファイルが含まれています。
/servers	JRun サーバーとそのアプリケーションが含まれています。
/servers/lib	すべての JRun サーバーがアクセスする .jar ファイルおよび .class ファイルが含まれています。ここに共有データベース ドライバおよび他の共有ファイルを格納しておくと便利です。このディレクトリには、タグ ライブラリが格納されます。
/servlets	規定の Web アプリケーションにアクセス可能な .class が格納されます。このディレクトリは下位互換性を目的として用意されています。新しいアプリケーションの .class ファイルは、サーブレット 2.2 の仕様で定義されている階層構造で構成しなければなりません。
/uninst	JRun のアンインストールに関する情報が含まれています。

JRun Admin Server のサブディレクトリ

以下の表は、/servers/admin ディレクトリ内のサブディレクトリを示しています。

ディレクトリ	説明
/servers/admin	JRun Admin Server を定義します。
/servers/admin/deploy	公開する Enterprise JavaBeans (EJBs) を格納します。公開が完了すると、起動時に EJB が runtime ディレクトリにコピーされます。
/servers/admin/jmc-app	JRun 管理コンソール (JMC) アプリケーションが含まれています。
/servers/admin/lib	admin サーバー内のすべてのアプリケーションがアクセスする .jar ファイルおよび .class ファイルが含まれています。
/servers/admin/runtime	EJB 実行時ディレクトリ
/servers/admin/tmp	該当する JRun サーバーにある各アプリケーションのテンポラリ サブディレクトリが含まれています。これらのテンポラリ ディレクトリを削除しないでください。

Default Server のサブディレクトリ

次の表は、/servers/default ディレクトリ内のサブディレクトリを示します。JRun サーバーを新規作成した場合には、これらのサブディレクトリはそのサーバーの一部となります。

ディレクトリ	説明
/servers/default	標準の JRun サーバーを定義します。
/servers/default/default-app	標準の JRun アプリケーションが含まれています。このアプリケーションを使って Java サーブレット および JSP を作成およびテストします。

ディレクトリ	説明
/servers/default/default-app/WEB-INF	ドキュメントのルート ディレクトリに含まれていない標準 アプリケーションに関連するすべてのリソースが含まれています。このディレクトリはアプリケーションのドキュメント ツリーには含まれません。つまり、このディレクトリに含まれるファイルにクライアントから直接にアクセスすることはできません。このディレクトリは、アプリケーション記述子 web.xml を含みます。
/servers/default/default-app/WEB-INF/classes	Web アプリケーションのサーブレットが使用する Java クラス ファイルが格納されます。
/servers/default/default-app/WEB-INF/jsp	アプリケーションの JSP 用の .class ファイルが格納されます。
/servers/default/default-app/WEB-INF/lib	アプリケーションが使用する beans およびその他のファイルが格納されます。これらのファイルは .jar ファイルに格納される場合もあります。
/servers/default/demo-app	JSP/servlet のサンプル アプリケーションが含まれています。
/servers/default/deploy	公開された Enterprise JavaBeans (EJBs) を格納します。公開された EJB は、起動時に runtime ディレクトリにコピーされます。
/servers/default/lib	標準サーバー内にあるすべてのアプリケーションがアクセスする .jar ファイルおよび .class ファイルが含まれています。
/servers/default/runtime	公開された EJB は、起動時に runtime ディレクトリにコピーされます。
/servers/default/runtime/classes	動的にロードされる EJB インプリメンテーションのためのクラス ファイルが含まれています。
/servers/default/tmp	該当する JRun サーバーにある各アプリケーションのテンポラリ サブディレクトリが含まれています。これらのテンポラリ ディレクトリを削除しないようにしてください。

JRun サーバーの使用方法

JRun は、JRun サーバーで他の機能を起動、停止、および実行するためのユーティリティを備えています。この後、JRun プラットフォームごとのユーティリティについて説明します。JRun サーバーについては、94 ページの「 JRun サーバーの設定」を参照してください。

Windows に関する検討事項

インストール時に、JRun を Windows システム上の NT サービスとして実行するように設定することができます。この場合、このサービスを無効にしない限り、NT システムを起動するたびに JRun が起動します。コントロール パネルからアクセスできる [サービス] ダイアログ ボックスでもここで説明する手順を行うことができます。サービスとして実行しない場合、JRun はアプリケーションとして実行されます。

また、ここで説明する Windows に関する手順は、スクリプトの実行が可能な JRun コマンド ライン ユーティリティによって実行することもできます。詳細については、96 ページの「 jrun コマンドの使用」を参照してください。

JRun サーバーの起動と停止

JRun サーバーを起動するには

- Windows:

  [スタート] > [プログラム] > [JRun 3.0] > [JRun_server_name] をクリックします。たとえば、JRun の admin サーバーを起動するには、[スタート] > [プログラム] > [JRun 3.0] > [JRun Admin Server] をクリックします。

  または

  JRun コマンド ライン ユーティリティの使用 :

  ```
  jrun –start <JRun_server_name>
  ```

  メモ JRun をサービスとしてインストールした場合、サービスがすでに実行中であるときに JRun をプログラム グループから起動しようとすると ([スタート] > [プログラム] > [JRun 3.0] を選択)、「JRun 異常終了」エラーが発生します。

- UNIX:

  JRun コマンド ライン ユーティリティの使用 :

  ```
  jrun –nohup –start <JRun_server_name>
  ```

  または

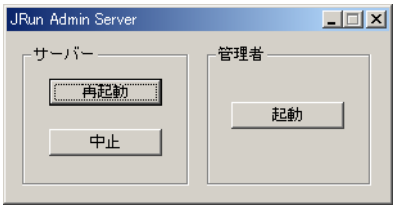
  ```
  jrun –start <JRun_server_name>
  ```

  nohup オプションにより、JRun サーバーはバックグラウンド プロセスとして起動します。

JRun サーバーを停止するには

- Windows:

  JRun サーバーをアプリケーションとして実行している場合は、システム トレイで JRun サーバーのアイコンをクリックすることで、JRun アプリケーション マネージャを開きます。



  次に [停止] ボタンをクリックします。JRun は該当する JRun サーバーのみを停止します。

  JRun を NT サービスとして実行している場合は、コントロール パネルからアクセスできる [サービス] ダイアログボックスを使って JRun サーバーを停止します。

- UNIX/Windows:

  JRun コマンド ライン ユーティリティの使用 :
  jrun -stop <JRun_server_name>

JRun サーバーを再起動するには

- Windows:

  JRun サーバーをアプリケーションとして実行している場合は、システム トレイで JRun サーバーのアイコンをクリックすることで、JRun アプリケーション マネージャを開きます。次に [再起動] ボタンをクリックします。JRun は該当する JRun サーバーだけを再起動します。

  JRun サーバーを NT サービスとして実行している場合は、コントロール パネルからアクセスできる [サービス] ダイアログボックスを使って JRun サーバーを停止してからふたたび起動します。

- UNIX/Windows:

  JRun コマンド ライン ユーティリティの使用 :
  jrun -restart <JRun_server_name>

  または

  JRun 管理コンソール (JMC) の左側ペインで [machine_name] をクリックします。また、右側ペインで、再起動する JRun サーバーの [Restart] リンクをクリックします。JRun は指定したサーバーを再起動します。JMC の使用方法については、27 ページの「 JRun 管理コンソールの起動」を参照してください。

メモ JMC で admin JRun サーバーを再起動することはできません。

JRun デモ アプリケーションの開始

JRun には、デモ アプリケーションから使用できるサンプルの EJB、Java サーブレット、JavaServer Pages (JSP)、および、サンプル タグ ライブラリが添付されています。ここでは、Windows および UNIX で JRun デモンストレーションを開始する方法について説明します。

メモ この手順は、default JRun サーバーが、JRun 付属の Web サーバー (JWS) が規定のポート (8100) で実行されていることを想定しています。JMC (admin JRun サーバー) に関連する JWS が動作する規定ポートは 8000 です。

運用環境では、JRun サーバーから demo-app の登録を取りやめ、ファイル システムから関連ファイルを削除します。Web アプリケーションの削除については、128 ページの「アプリケーションの削除」を参照してください。

サーブレット、タグ ライブラリ、JSP および EJB サンプルについては、『JRun サンプル ガイド』を参照してください。

JRun デモ アプリケーションを起動するには

1. JRun サーバーが実行されていない場合は、35 ページの「 JRun サーバーの起動と停止」の手順に従って default JRun server を起動します。
2. Web ブラウザで、次の URL を開きます。

   http://localhost:8100/demo/index.html

   または (Windows のみ)

   [スタート] > [プログラム] > [JRun 3.0] > [JRun Demo] をクリックします。

トラブルシューティング

ここでは、JRun のインスト ールに関連する一般的な問題の解決に役立つ情報を示します。

JRun のトラブルシューティングを行う際、<JRun_directory>/logs に置かれているログ ファイルをチェックすることで、追加情報を得ることができます。

ログイン ページが正しく開かない場合には

- 要求 URL のポート 番号を確認して ください。インスト ール実行中に JRun の標準設定 (8000) が上書きされた可能性があり ます。その場合は、その値を使用します。不明な場合は、</>JRun_directory>/servers/admin/local.properties ファイルの Web Services」セクション の [web.endpoint.main.port] の値をチェックします。

- プロパティ ファイルを変更したあと、JRun サーバーを再起動します。
- JRun Admin Server が実行されていることを確認してください。
    - Windows の場合は、システム トレイを確認してください。JRun をアプリケーションとしてインストールした場合には、[JRun Admin Server] アイコンが表示されます。JRun をサービスとしてインストールした場合には、コントロール パネルからアクセスできる [サービス] ダイアログ ボックスで JRun Admin Server サービスが実行されているかどうかを確認してください。
    - UNIX の場合は、/opt/jrun/bin の次のコマンド ライン ツールを使用して、サーバーが実行されているかどうかを確認してください。

        ```
        jrun –status admin
        ```
- /JRun/servers/admin ディレクトリおよびそのサブディレクトリの読み取りアクセス権があることを確認してください。
- JMC がポート 8000 で受信していることを確認します。JRun 3.0 を前のバージョンの上にインストールした場合、JRun がストレイ プロセスを検出したときに既定のポート番号を 8001 に増分した可能性があります。詳細については、16 ページの「 JRun のインストール」を参照してください。

デモ アプリケーションが正しく開かない場合には

- **デモ** アプリケーションにアクセスするときに正しいポートを指定したかどうかを確認してください。デモ アプリケーションは、Admin JRun サーバー (ポート 8000) ではなく、default JRun サーバー (ポート 8100) で実行されています。

    ```
    http://localhost:8100/demo/index.html
    ```

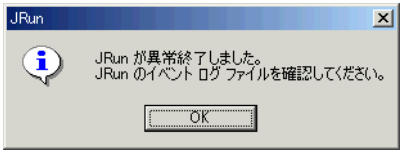
    default サーバーの規定のポートは 8100 です。しかし、インストール時にこのポート番号を使用する別のプロセスが検出されると、JRun は、空きポートが見つかるまでポート番号を増やしていきます。

    サーバーがどのポートで実行されているかは、JRun 管理コンソールの JRun Web サーバー パネルで確認できます。詳細については、115 ページの「 JWS の設定」を参照してください。
- default JRun サーバーが実行されていることを確認してください。
    - Windows の場合は、システム トレイを確認してください。JRun をアプリケーションとしてインストールした場合には、[default サーバー] アイコンが表示されます。JRun をサービスとしてインストールした場合には、コントロール パネルからアクセスできる [サービス] ダイアログ ボックスで JRun default Server サービスが実行されているかどうかを確認してください。
    - UNIX の場合は、/opt/jrun/bin の次のコマンド ライン ツールを使用して、サーバーが実行されているかどうかを確認してください。

        ```
        jrun –status default
        ```

- [ようこそ] ページの [アプリケーションの例] リンクをクリックすることで、デモアプリケーションを JMC から起動してみてください。

JRun サーバーを Windows で起動しようとしたときに、次のエラーが表示される場合には



JRun サーバーをアプリケーションとして起動しようとしている場合は、JRun がすでにサービスとして実行されていないかを確認してください。JRun サービスは、コントロール パネルからアクセスできる [サービス] ダイアログボックスを介して呼び出されます。一方、JRun アプリケーションは通常、プログラム グループまたは **[スタート]** メニューから起動されます。

JRun サーバーがすでに実行されていることを確認するには、システムトレイに [JRun] アイコンが表示されているかどうかをチェックします。サーバーがアプリケーションとして実行されている場合には、実行中のサーバーごとに 1 つのアイコンが表示されます。また、これらのサーバーがサービスとして実行されている場合には、コントロール パネルからアクセスできる [サービス] ダイアログボックスをチェックして、「JRun Admin Server」と「JRun default Server」があるかどうかを確認してください。

第 2 章

外部 Web サーバーの構成

この章では、Web サーバーと通信するために JRun を構成する方法、および JRun を使用してサーバーの構成を変更する方法について説明します。この手順の一部として、Web サーバー用の構成パラメータを設定し、JRun 管理コンソール (JMC) を使用して JRun と Web サーバーの接続を構成しなければならない場合があります。

特定のサーバーに関する JRun の構成手順については、該当するセクションを参照してください。

メモ JRun を使用してアプリケーションを開発するために、別の Web サーバーを用意する必要はありません。JRun をインストールすると JRun 独自の Web サーバーが提供されます。JRun をプラグインとして外部 Web サーバーに接続していない場合、この章は必要ありません。

目次

構成の概要

JRun 3.0 は、スタンドアロンの Java アプリケーション サーバーとしても、あるいは既存の Web サーバーに Java アプリケーションのサポートを追加するプラグイン モジュールとしても機能します。スタンドアロンの場合、統合された JRun Web Server (JWS) を使用して JRun は機能します。プラグイン モジュールの場合、接続ウィザードを実行して、JRun を外部 Web サーバーに接続します。

JRun は、多様な Web サーバーをサポートします。JRun と Web サーバーの接続を構成するための基本的な手順は、すべての Web サーバーにおいて同じですが、各 Web サーバーには固有の構成情報および設定があります。Web サーバー用に JRun を構成するための一般的な処理手順を次に示します。

JRun を外部 Web サーバーに接続するとき、Web サーバーの要求を処理する JRun サーバーを選択する必要があることに注意してください。アプリケーションがサーバーで導入されるため、たいていの場合、`admin` JRun サーバーではなく、`default` JRun サーバー (または作成したもの) に接続することになります。

JRun コネクタについての詳細および分散環境での JRun の設定方法については、159 ページの第 4 章「コネクタ」を参照してください。

JRun と外部 Web サーバーの接続を構成するには

1. Web サーバーを停止します。
2. 必要に応じて、JRun サーバーと通信するために Web サーバーを構成します。
3. JRun 管理コンソール (JMC) を起動します。
4. JRun コネクタ ウィザードを実行して、Web サーバーと default JRun サーバーの通信を容易にする JRun Connection Module (JCM) を作成します。
5. Web サーバーと default JRun サーバーを起動します。
6. JRun と Web サーバーの接続を確認します。

この後のセクションでは、JRun でサポートされている特定の Web サーバー用の手順について説明します。

JRun と外部 Web サーバーは、JRun コネクタ経由で接続されます。Allaire は、主要な Web サーバー用のネイティブ コネクタを用意していますが、JRun 側にも、サポートされていない Web サーバーで使用するためのコネクタ ソース コードが含まれています。このソース コードは、基本的な使用手順とともに `<JRun_directory>/connectors/src` に置かれています。詳細については、184 ページの「カスタム コネクタの作成」を参照してください。

Apache の構成

ここでは、Windows または UNIX で実行する Apache Web サーバーと通信するために JRun を構成する方法について説明します。高度な Web サーバー接続テクニックについては、184 ページの「カスタム コネクタの作成」を参照してください。

使用するオペレーティング システムに基づき、JRun は Apache Web サーバーによるサーブレットの実行に関して、Dynamic Shared Objects (DSO) モジュールおよび静的モジュールといった 2 つの方法をサポートしています。Apache と通信する JRun を構成する処理の一部として、特定のモジュールに関して Apache をコンパイルしなければならない場合があります。

Windows ベースのシステムの場合、Apache は DSO モジュールのみサポートします。DSO モジュールをセットアップするために構成手順を実行する必要はありません。

DSO は Apache の構築を容易にするため、Linux を含む UNIX ベースのシステムで Apache バージョン 1.3.x を実行する場合は、DSO を使用することをお勧めします。また、RedHat Linux 5.2 など一部のプラットフォームでは、DSO サポート付きの Apache があらかじめ構築されるため、Apache を再コンパイルする必要はありません。

UNIX バージョンの Apache では、すべて JRun を静的モジュールとしてサーバーにコンパイルすることができますが、静的モジュールは DSO をサポートしないシステムの場合にのみお勧めします。

この章では、JMC 内からのコネクタ ウィザードの起動方法について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

Apache と JRun を接続するには

1. Web サーバーを停止します。
2. **UNIX のみ** : システムで必要な場合は、DSO モジュールを構成します。

   この手順は、UNIX で Apache 1.3.x を実行する場合にのみ適用されます。

   - 次の Apache コマンドを実行して、Apache を DSO 用に構成します。

     ```
     ./configure --prefix=/user/local/apache --enable-rule=SHARED_CORE \
     --enable-module=so
     ```

   - make および make install スクリプトを使用して、Apache の再コンパイルおよびインストールを行います。
3. **UNIX のみ** : システムで必要な場合は、静的モジュールを構成します。静的モジュールは、DSO を使用していない場合にのみ構成します。

   1. JRun ソース ファイルを <JRun_directory>/connectors/apache/src から Apache の /src/modules/jrun ディレクトリにコピーします。

   静的モジュールの構成手順は、UNIX で Apache 1.2 を実行する場合と Apache 1.3.x を実行する場合で異なります。

Apache 1.2:

2. 次の行を src ディレクトリの Apache 構成ファイルに追加します。

   ```
   Module jrun_module modules/jrun/libjrun.a
   ```

3. Apache の src ディレクトリから configure スクリプトを実行して新しい Makefile を作成し、Apache の再コンパイルおよびインストールを行います。

Apache 1.3.x:

2. 次の Apache コマンドを実行して、JRun ライブラリを Apache サーバーに追加します。

   ```
   ./configure --prefix=/user/local/apache \
   --activate-module=src/modules/jrun/libjrun.a
   ```

   --prefix エントリおよびその他のエントリは、異なる場合があるので注意してください。有効なエントリは、--activate-module エントリです。

3. make および make install スクリプトを使用して、Apache の再コンパイルおよびインストールを行います。

4. Web ブラウザで 次の URL を開いて、JMC を起動します。

   ```
   http://localhost:8000
   ```

   Windows のみ : スタート > プログラム > JRun 3.0> JRun 管理コンソール を選択

   メモ この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

5. JRun admin として JMC にログインします。

6. アクセス バーの [**コネクタ ウィザード**] を選択します。

7. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
ステップ	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	Apache に接続する JRun サーバーを選択します。通常は、default サーバーを選択します。 JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [Apache Web サーバー] を選択します。
	Web サーバーのバージョン	Apache のバージョンを選択します。
	Web サーバーのプラットフォーム	Apache を実行しているプラットフォームを選択します。
手順 2	JRun サーバーの IP アドレス	Apache への接続に使用する JRun サーバーの IP アドレスを入力します。Apache と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバー コネクタ ポート	JRun サーバーが Apache との通信に使用するポートを指定します。このポートと Apache の HTTP ポートを混同しないでください。デフォルトは 51000 です。
手順 3	Apache の conf ディレクトリ	構成ファイル srm.conf および httpd.conf を含むディレクトリを指定します。JRun のディレクトリ リーダー を使用するには、[参照] をクリックします。
	DSO サポート	**UNIX:** JRun モジュールを Apache サーバーにコンパイルした場合は、[DSO サポート] を選択します。 **Windows:** [DSO サポート] をクリックします。

8. JRun コネクタのインストールが完了したら、Apache Web サーバーと default JRun サーバーを再起動します。

   default JRun サーバーがまだ起動されていない場合には、次の手順に従ってください。

   - Windows:

     JRun をアプリケーションとしてインストールした場合には、[スタート] > [プログラム] > [JRun 3.0] > [JRun default Server] を選択します。

     JRun をサービスとしてインストールした場合には、コントロール パネルからサービス コントロール ツールを開いて「JRun default Server」サービスを起動するか、次の JRun コマンドライン ユーティリティを使用します。

     ```
     jrun –start default
     ```

   - UNIX:

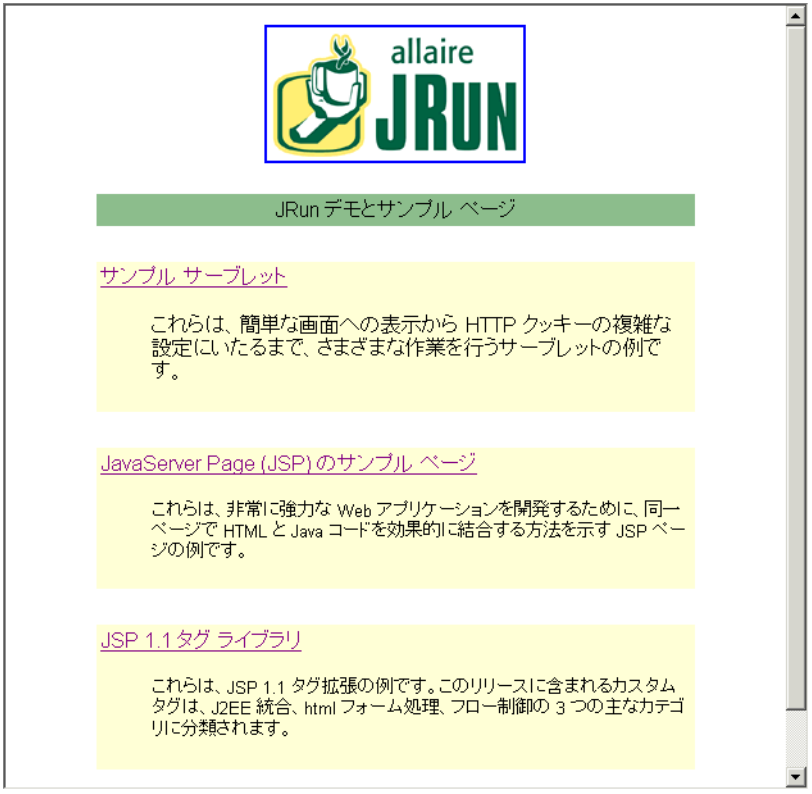
     JRun コマンド ライン ユーティリティの使用 :

     ```
     jrun –start default
     ```

9. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と Apache Web サーバーの接続を確認します。

   ```
   http://localhost:80/demo/index.html
   ```

   これは、Apache が既定のポート 80 で接続をリスニングしていることを想定しています。



デモ アプリケーションが実行される場合、JRun と Apache Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

Apache 構成ファイルへの変更

コネクタ ウィザードでは、Apache httpd.conf ファイルに JRun Settings セクションを追加して、いくつかの変更を加えます。この設定には、Windows システム上の JRun DLL を初期化する jrun.ini ファイルの設定が反映されます。一般的な JRun Settings セクションは、次のようになります。

```
# JRun Settings
# JRun - Comment out this line to disable DSO (ie you compiled module # into your
server.
LoadModule jrun_module "C:\Program Files\Allaire\JRun\connectors\
    apache\intel-win\mod_jrun136.dll"
<IfModule mod_jrun.c>
JRunConfig Verbose false
JRunConfig ProxyHost 127.0.0.1
JRunConfig ProxyPort 51000
```

```
JRunConfig Timeout 300
JRunConfig Mappings "C:\Program Files\Allaire\JRun\servers\
    default\local.properties"
</IfModule>
```

コネクタ ウィザードは、入力した内容に基づいて JRun local.properties ファイルも変更します。詳細については、81 ページの「 local.properties への変更」を参照してください。

JRun コネクタ ウィザードが使用するファイルを変更しないでください。ここでの説明は、情報提供のみを目的としています。

IIS 3.0/PWS の構成

ここでは、JRun を Windows 95/98/NT システムで実行する IIS 3.0 または Personal Web Server (PWS) に接続する方法について説明します。高度な Web サーバー接続テクニックについては、159 ページの第 4 章「コネクタ」を参照してください。

この章では、JMC 内からのコネクタ ウィザードの起動方法について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

JRun と IIS 3.0 または PWS を接続するには

1. Web サーバーを停止します。

   メモ IIS 3.0 の場合、Web サーバーを停止するには、コントロール パネルの Web サーバー ユーティリティを使用します。Microsoft 管理コンソール (MMC) を使用して Web サーバーを停止しないでください。

2. 次のいずれかの方法で JMC を起動します。

   - [**スタート**] > [**プログラム**] > [JRun 3.0] > [JRun **管理コンソール**] をクリックします。
   - Web ブラウザで、次の URL を開きます。

     http://localhost:8000

   この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

3. JRun 管理者として JMC にログインします。
4. アクセス バーの [**コネクタ ウィザード**] を選択します。

5. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	Web サーバーに接続する JRun サーバーを選択します。通常は、default サーバーを選択します。 JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [Internet Information Server] または [Personal Web Server] を選択します。
	Web サーバーのバージョン	ドロップダウン リストから Web サーバーのバージョンを選択します。
	Web サーバーのプラットフォーム	ドロップダウン リストから [intel-win] を選択します。
手順 2	JRun サーバーの IP アドレス	IIS/PWS への接続に使用する JRun サーバーの IP アドレスを入力します。IIS/PWS と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバーの接続ポート	JRun サーバーが IIS/PWS との通信に使用するポートを指定します。 このポートと IIS/PWS の HTTP ポートを混同しないでください。デフォルトは、51000 です。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 3	PWS または IIS の scripts ディレクトリ	IIS/PWS の /scripts ディレクトリの場所を指定します。JRun の ディレクトリ リーダー を使用するには、[参照] をクリックします。
	グローバル フィルタとしてのインストール	このチェックボックスを選択すると、JRun は ISAPI フィルタをインストールして、HTTP リクエストが サーブレットを実行しようとしているかどうかを検出します。

6. JRun コネクタのインストールが完了したら、IIS/PWS および default JRun サーバーを再起動します。

   メモ コネクタを PWS に対して実行した場合には、コンピュータを再起動する必要があります。

   default JRun サーバーがまだ起動されていない場合には、[スタート] > [プログラム] > [JRun 3.0] > [JRun Default Server] をクリックします。

7. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と IIS/PWS Web サーバーの接続を確認します。

   http://localhost:80/demo/index.html

   これは、IIS/PWS が既定のポート 80 で接続をリスニングしていることを想定しています。

デモ アプリケーションが実行される場合、JRun と IIS/PWS Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

構成ファイルへの変更

コネクタ ウィザードは、次のように Web サーバーの実装に変更を加えます。

- jrun.ini および jrun.dll ファイルを /inetpub/scripts ディレクトリに追加します。JRun は .ini ファイルを使用して、起動時に DLL を初期化します。DLL は、Web サーバーへの HTTP リクエストを阻止し、JRun が処理するために適切な HTTP リクエストを JRun に渡す ISAPI フィルタです。
- Windows のレジストリを変更します。ウィザードにより、jrun.dll をポイントする「Filter DLL」キーが追加されます。このキーの値は、jrun.dll の絶対パスです。キーは、HKEY_LOCAL_MACHINE/SYSTEM/CurrentControlSet/Services/W3SVC/Parameters/ に置かれています。

- [グローバル フィルタとしてインストールする] を選択した場合、コネクタ ウィザードは、MMC で構成可能な JRun コネクタ フィルタへのポインタを追加します。詳細については、57 ページの「JRun ISAPI フィルタの構成」を参照してください。
- JRun local.properties ファイルを更新します。詳細については、81 ページの「local.properties への変更」を参照してください。

JRun コネクタ ウィザードが使用するレジストリやファイルを変更しないでください。ここでの説明は、情報提供のみを目的としています。

IIS 4.0/5.0 の構成

JRun は、Windows NT の場合は Internet Information Server 4.0、Windows 2000 の場合は Internet Information Server 5.0 とともに使用するよう構成できます。IIS に接続するには、JSP またはサーブレットを実行するディレクトリに IIS の実行許可を設定してから、JRun コネクタ ウィザードを使用する必要があります。このセクションでは、以下について説明します。

- 52 ページの「JRun の IIS 4.0/5.0 への接続」
- 57 ページの「IIS 構成ファイルへの変更」
- 57 ページの「JRun ISAPI フィルタの構成」

この章では、JMC 内からのコネクタ ウィザードの起動方法について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

高度な Web サーバー接続テクニックについては、159 ページの第 4 章「コネクタ」を参照してください。

JRun の IIS 4.0/5.0 への接続

JRun と IIS 4.0/5.0 を接続するには

1. コントロール パネルのサービス コントロール管理ツールで、World Wide Web Publishing Service を停止します。

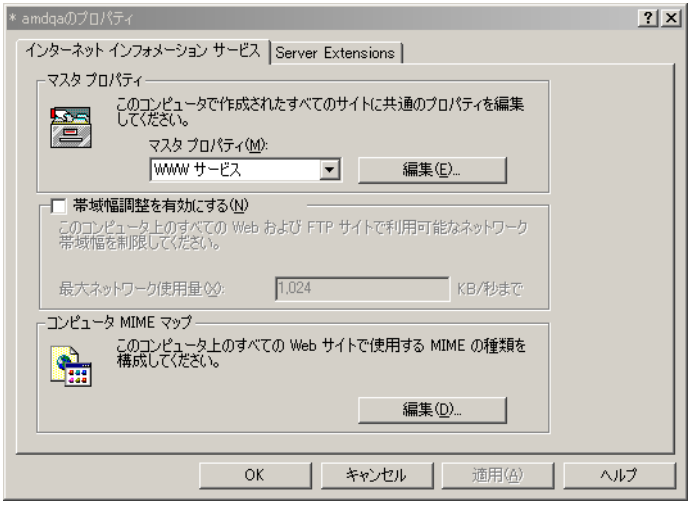
   メモ Windows NT では、Microsoft 管理コンソール (MMC) の Web サービス スナップインを使用してこの処理を行わないでください。

2. インターネット サービス マネージャを開きます。

   Windows NT: [スタート] > [プログラム] > [NT 4.0 Option Pack] > [Microsoft Internet Information Server] > [インターネット サービス マネージャ] を選択します。Microsoft 管理コンソール (MMC) が表示され、iis.msc スナップインが開きます。

Windows 2000: [**スタート**] > [**設定**] > [**コントロールパネル**] > [**管理ツール**] > [**インターネット サービス マネージャ**] をクリックします。インターネット サービス マネージャが開きます。

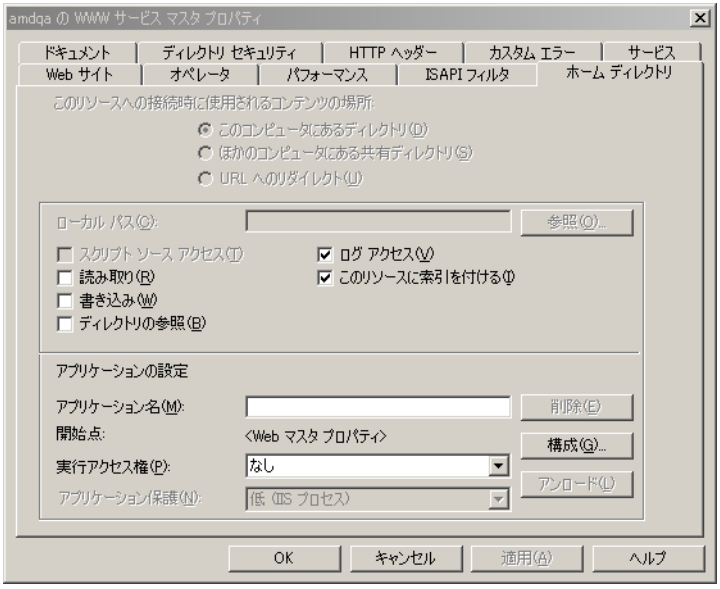
3. 特定の仮想 Web サーバーまたは Web サーバー全体でマウスの右ボタンをクリックし、[**プロパティ**] を選択します。[プロパティ] ウィンドウが表示されます。



4. [**マスタ プロパティ**] ドロップダウン リストボックスから、[**WWW サービス**] リストボックスを選択して、[編集] をクリックします。WWW Service Master Properties アプリケーションが起動します。
5. [ホーム ディレクトリ] タブを選択します。
6. [**ローカル パス**] フィールドで指定されているディレクトリの実行許可を設定します。[**ローカル パス**] に指定されているディレクトリがない場合、すべての仮想 Web サーバーに影響するグローバル プロパティが設定されます。

   Windows NT:[**アクセス許可**] で、[**実行**] をクリックします (スクリプトを含む)。

   Windows 2000:[**アクセス許可の実行**] ドロップダウン リストボックスで、[**スクリプト と 実行可能ファイル**] をクリックします。



7. [OK] をクリックして、変更を適用します。[継承優先] を変更するかどうかを尋ねられます。[OK] をクリックします。
8. 次のいずれかの方法で JMC を起動します。
   - **[スタート]** > **[プログラム]** > [JRun 3.0] > **[JRun 管理コンソール]** をクリックします。
   - Web ブラウザで、次の URL を開きます。

     http://localhost:8000

   この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。
9. JRun 管理者として JMC にログインします。
10. アクセス バーの **[コネクタ ウィザード]** を選択します。

11. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	IIS に接続する JRun サーバーを選択します。通常は、default サーバーを選択します。JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [Internet Information Server] を選択します。
	Web サーバーのバージョン	ドロップダウン リストから、[4.0] または [5.0] を選択します。
	Web サーバーのプラットフォーム	ドロップダウン リストから [intel-win] を選択します。
手順 2	JRun サーバーの IP アドレス	IIS への接続に使用する JRun サーバーの IP アドレスを入力します。IIS と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバーの接続ポート	JRun サーバーが IIS との通信に使用するポートを指定します。このポートと IIS の HTTP ポートを混同しないでください。デフォルトは 51000 です。
手順 3	IIS スクリプト ディレクトリ	IIS の /scripts ディレクトリの場所を指定します。JRun の ディレクトリ リーダー を使用するには、[参照] をクリックします。
	グローバル フィルタとしてインストールする	このチェックボックスを選択すると、JRun は ISAPI フィルタをインストールして、HTTP リクエストが サーブレット を実行しようとしているかどうかを検出します。

12. JRun コネクタのインストールが完了したら、IIS Web サーバーと default JRun サーバーを再起動します。

    default JRun サーバーがまだ起動されていない場合には、次の手順に従ってください。

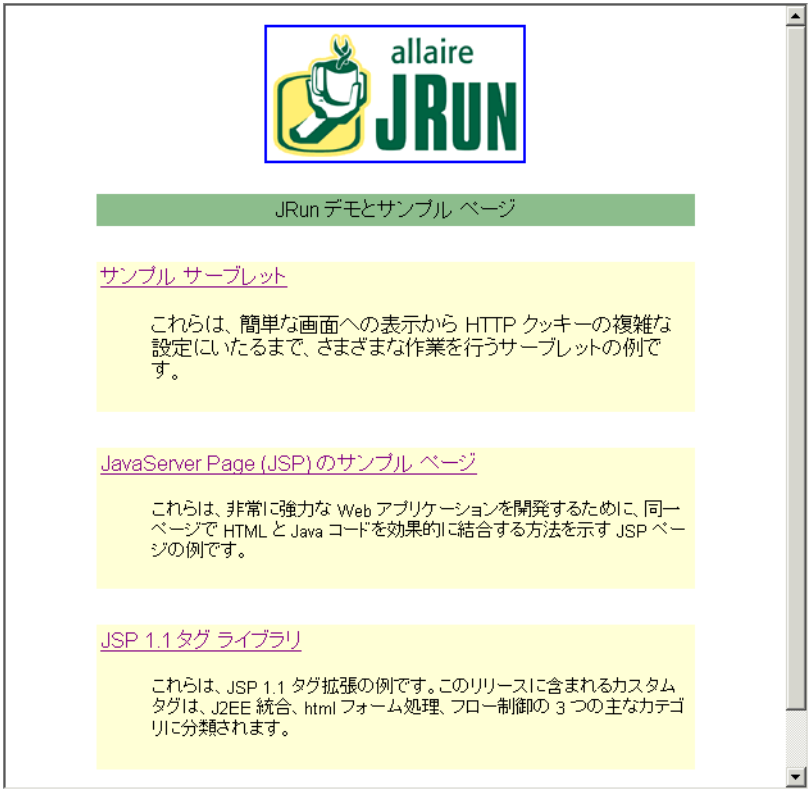
    - JRun をアプリケーションとしてインストールした場合には、[スタート] > [プログラム] > [JRun 3.0] > [JRun default Server] を選択します。
    - JRun をサービスとしてインストールした場合には、コントロール パネルのサービス コントロール管理ツールで「JRun default Server」サービスを起動するか、次の JRun コマンド ライン ユーティリティを使用します。

      ```
      jrun –start default
      ```

13. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と IIS Web サーバーの接続を確認します。

    ```
    http://localhost:80/demo/index.html
    ```

    これは、IIS が既定のポート 80 で接続をリスニングしていることを想定しています。



デモ アプリケーションが実行される場合、JRun と IIS Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

IIS 構成ファイルへの変更

JRun コネクタ ウィザードでは、次のように IIS の実装に変更を加えます。

- jrun.ini および jrun.dll ファイルを /inetpub/scripts ディレクトリに追加します。JRun は .ini ファイルを使用して、起動時に DLL を初期化します。DLL は、Web サーバーへの HTTP リクエストを阻止し、JRun が処理するために適切な HTTP リクエストを JRun に渡す ISAPI フィルタです。
- IIS の [アプリケーション マッピング] 設定で、.jsp を jrun.dll にマッピングします。
- インターネットサービスマネージャのメタベースにいくつかの変更を加えます。メタベースは、IIS の設定を格納する階層構造です。JRun がメタベースに加える変更には次の内容が含まれます。
    - jrun.dll の絶対パスを ScriptMaps パラメータに追加します。ScriptMaps は、ファイル名拡張子を処理用の DLL にマッピングします。
    - **[グローバル フィルタとしてインストールする]** を選択した場合には、コネクタ ウィザードは、**[JRun コネクタ フィルタ]** サービス オブジェクトおよびそれに関連するパラメータをメタベースの**フィルタ**オブジェクトに追加します。
- JRun local.properties ファイルを更新します。詳細については、81 ページの「 local.properties への変更」を参照してください。

JRun コネクタ ウィザードが使用するレジストリやメタベースを変更したり、ファイルを編集しないでください。ここでの説明は、情報提供のみを目的としています。

JRun ISAPI フィルタの構成

ISAPI フィルタは、IIS または PWS が HTTP リクエストを受信する場合に、イベントに応答することができます。JRun 接続ウィザードの実行中に **[グローバル フィルタとしてインストールする]** を選択すると、Web サーバーのメモリ内の他の ISAPI フィルタに追加される DLL をインストールします。jrun.dll の既定の場所は c:\inetpub\scripts\ です。

このセクションの説明はオプションです。JRun の既定の構成を使用する場合、JRun ISAPI フィルタに変更を加える必要はありません。ただし、他の ISAPI フィルタをインストールする場合には、変更が必要なこともあります。

JRun ISAPI フィルタの編集

[ISAPI フィルタ] ダイアログ ボックスを使用して、IIS 用の JRun ISAPI フィルタの名前および場所を追加、削除、および変更することができます。

JRun ISAPI フィルタを編集するには

1. Internet Service Manager を開きます。

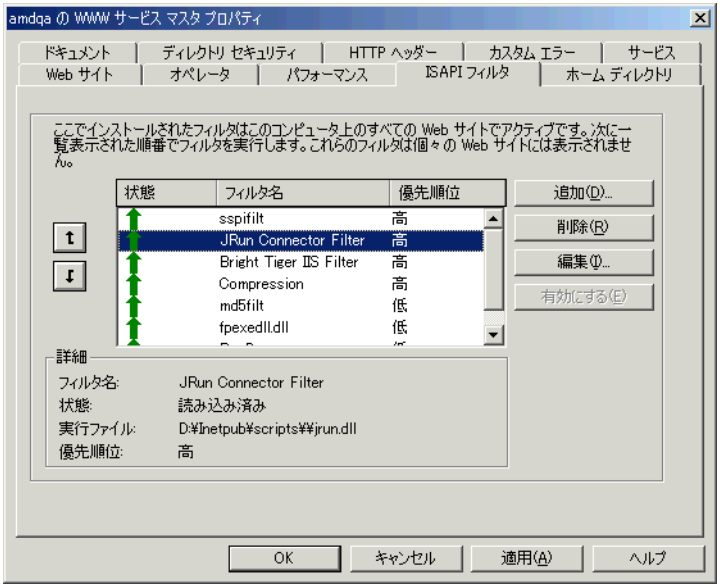
   Windows NT: [スタート] > [プログラム] > [NT 4.0 Option Pack] > [Microsoft Internet Information Server] > [インターネット サービス マネージャ] を選択します。Microsoft 管理コンソール (MMC) が表示され、iis.msc スナップインが開きます。

   Windows 2000: [スタート] > [設定] > [コントロールパネル] > [管理ツール] > [インターネット サービス マネージャ] をクリックします。インターネット サービス マネージャが開きます。

2. マシン名の上でマウスを右クリックして、[プロパティ] を選択します。[プロパティ] ウィンドウが表示されます。

3. [マスタ プロパティ] ドロップダウン リストボックスから、[WWW サービス] リストボックスを選択して、[編集] をクリックします。WWW Service マスタ プロパティ アプリケーションが起動します。

4. [ISAPI フィルタ] タブをクリックします。



5. フィルタを削除するには、使用可能な ISAPI フィルタの一覧から [JRun コネクタ フィルタ] を選択して、[削除] をクリックします。

6. JRun フィルタ DLL の名前または場所を編集するには、使用可能な ISAPI フィルタの一覧から [JRun コネクタ フィルタ] を選択して、[編集] をクリックします。

7. 新しい JRun フィルタを追加するには、[追加] をクリックして、新しいフィルタの場所を参照します。

   メモ 新しいフィルタを追加する前に、古いフィルタを削除する必要があります。

8. 変更を適用するには、[OK] をクリックします。
9. Web サーバーを再起動します。

JRun ISAPI フィルタへの優先順位付け

複数の ISAPI フィルタが同じイベント (または通知) に登録されている場合、それらのフィルタは IIS によって連続して呼び出されます。優先順位の高いフィルタは、優先順位の低いフィルタより先に実行されます。高、中、低といった優先レベルは、Internet Services Manager またはその他のメタベース エディタで変更することのできない読み取り専用のプロパティです。JRun の優先レベルは " 高 " です。

フィルタの優先レベルは変更できませんが、フィルタが他のフィルタと同じ優先レベルを共有する場合は、イベントに応答するフィルタの順位を指定することができます。JRun ISAPI フィルタの優先順位を変更するには、次の手順に従います。

JRun ISAPI フィルタの優先順位を変更するには

1. Internet Service Manager を開きます。

   Windows NT: [スタート] > [プログラム] > [NT 4.0 Option Pack] > [Microsoft Internet Information Server] > [インターネット サービス マネージャ] を選択します。Microsoft 管理コンソール (MMC) が表示され、iis.msc スナップインが開きます。

   Windows 2000: [スタート] > [設定] > [コントロールパネル] > [管理ツール] > [インターネット サービス マネージャ] をクリックします。インターネット サービス マネージャが開きます。

2. マシン名の上でマウスを右クリックして、  [プロパティ] を選択します。[プロパティ] ウィンドウが表示されます。
3. [マスタ プロパティ] ドロップダウン リストボックスから [プロパティ] を選択して、[編集] をクリックします。WWW Service マスタ プロパティアプリケーションが起動します。
4. [ISAPI フィルタ] タブをクリックします。
5. 使用可能な ISAPI フィルタの一覧から [JRun コネクタ フィルタ] を選択します。
6. 上向矢印をクリックして、JRun フィルタを一覧の他のフィルタの前に移動します。
7. [OK] をクリックして、変更を適用します。
8. Web サーバーを再起動します。

Netscape/iPlanet の構成

ここでは、Netscape Web サーバーの構成方法について説明します。高度な Web サーバー接続テクニックについては、159 ページの第 4 章「コネクタ」を参照してください。

メモ Netscape を構成する作業の一部として、Netscape Java Interpreter を有効にしなければならない場合があります。ただし、これは JRun コネクタ ウィザードを起動しなければわかりません。Java インタプリタを有効にする方法の詳細については、64 ページの「Java インタプリタの有効化」を参照してください。

この章では、JMC 内からのコネクタ ウィザードの起動方法について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

JRun と Netscape Web サーバーを接続するには

1. Web サーバーを停止します。
2. Web ブラウザで 次の URL を開いて、JMC を起動します。

   http://localhost:8000

   Windows のみ :

   [スタート] > [プログラム] > [JRun 3.0] > [JRun Management Console] をクリックします。

   メモ この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

3. JRun admin として JMC にログインします。
4. アクセス バーの [コネクタ ウィザード] を選択します。

5. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	Web サーバーに接続する JRun サーバーを選択します。通常は、default サーバーを選択します。 JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [Netscape Enterprise Server] または [Netscape FastTrack Server] を選択します。
	Web サーバーのバージョン	ドロップダウン リストから Web サーバーのバージョンを選択します。
	Web サーバーのプラットフォーム	Netscape Web サーバー を実行しているプラットフォームを選択します。
手順 2	JRun サーバーの IP アドレス	NES への接続に使用する JRun サーバーの IP アドレスを入力します。NES と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバーの接続ポート	JRun サーバーが NES との通信に使用するポートを指定します。このポートと NES の HTTP ポートを混同しないでください。デフォルトは 51000 です。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 3	Netscape の http-xxx ディレクトリ	https または httpd ディレクトリを指定します。このディレクトリは、通常、/Netscape/suitespot ディレクトリ内にあり、httpd-xxx または https-xxx という名前が付けられています。 JRun の ディレクトリ リーダー を使用するには、 [参照] をクリックします。
	ネイティブ コネクタまたは Java コネクタ	Netscape Web サーバーのネイティブ (標準) コネクタまたは Java コネクタを選択します。 ネイティブ コネクタは NSAPI を使用して、Web サーバーと通信します。通常は、このコネクタを使用してください。 Java コネクタは、Netscape の Server Applet API を実装する純粋な Java コネクタです。このコネクタは、ご使用のプラットフォーム用のネイティブ コネクタを利用できない場合にのみ使用します。このコネクタを選択する場合は、コネクタをインストールする前に、使用する Netscape サーバー用に Java を有効にする必要があります。Java インタプリタを有効にする方法の詳細については、64 ページの「 Java インタプリタの有効化」を参照してください。

6. JRun コネクタのインストールが完了したら、NES/iPlanet Web サーバーと default JRun サーバーを再起動します。

   default JRun サーバーがまだ起動されていない場合には、次の手順に従ってください。

   - Windows:

     JRun をアプリケーションとしてインストールした場合には、[スタート] > [プログラム] > [JRun 3.0 > JRun default Server] を選択します。

     JRun をサービスとしてインストールした場合には、コントロール パネルのサービス コントロール管理ツールで「JRun default Server」サービスを起動するか、次の JRun コマンドライン ユーティリティを使用します。

```
jrun -start default
```

- UNIX:

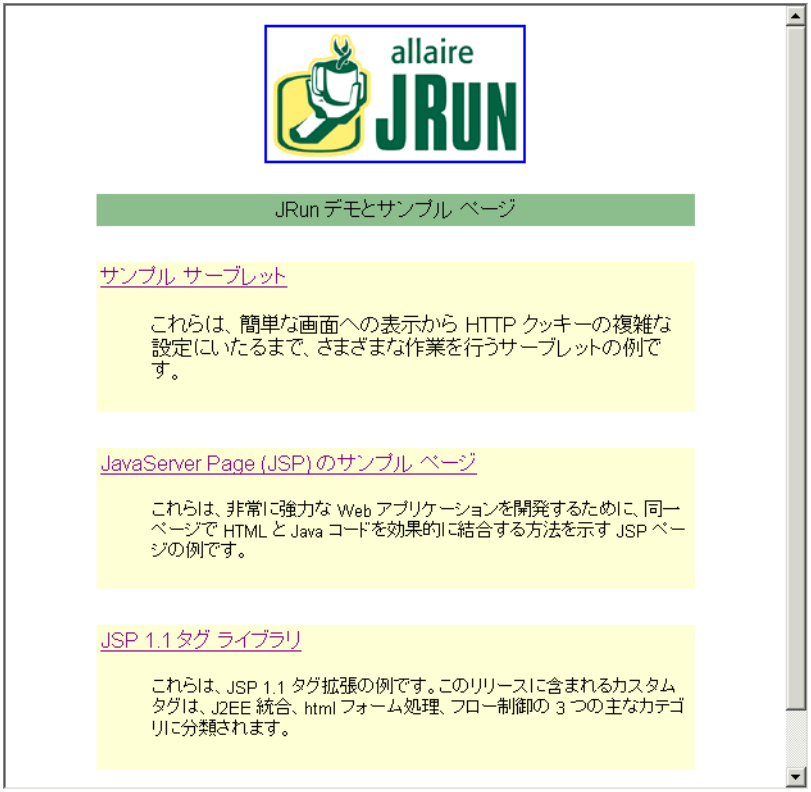
  JRun コマンド ライン ユーティリティの使用 :

  ```
  jrun -start default
  ```

7. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と NES/iPlanet Web サーバーの接続を確認します。

   ```
   http://localhost:80/demo/index.html
   ```

   これは、NES/iPlanet が規定ポート 80 で接続をリスニングしていることを想定しています。



デモ アプリケーションが実行される場合、JRun と NES/iPlanet Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

Java インタプリタの有効化

ネイティブ コネクタは、ほとんどのプラットフォームで使用することができます。Netscape Web サーバーでネイティブ コネクタを使用できる場合、追加の構成手順は必要ありません。ネイティブ コネクタを使用できない場合は、JRun コネクタを追加する前に、Netscape Web サーバーに組み込まれている Java インタプリタを有効にする必要があります。ここでは、Netscape 3.5.x 用の Java インタプリタを有効にする方法について説明します。

NES 3.5.x 用の Java インタプリタを有効にするには

1. Netscape Enterprise Administration Server を起動し、[Programs] メニューをクリックします。

2. [プログラム] の [Java] をクリックします。
3. [Activate the Java interpreter?] の [Yes] をクリックします。
4. JRun コネクタ ウィザードを使用して、JRun と Netscape の接続を構成します。この構成手順については、60 ページの「 Netscape/iPlanet の構成」で説明します。

NES 構成ファイルへの変更

JRun コネクタ ウィザードは、Netscape/Server4/https-<machine_name>/config/ ディレクトリの obj.conf ファイルを変更します。以下に説明する変更内容は、ネイティブ コネクタを使用する場合と Java コネクタ を使用する場合で異なります。

コネクタ ウィザードは、JRun local.properties ファイルも変更します。詳細については、81 ページの「 local.properties への変更」を参照してください。

JRun コネクタ ウィザードが使用するファイルを変更しないでください。ここでの説明は、情報提供のみを目的としています。

obj.conf ファイルを変更する場合、proxyhost 設定は、サーバー名ではなく IP アドレスにする必要があります。obj.conf ファイルのサンプルは、67 ページに記載されています。obj.conf ファイルの編集の詳細については、Netscape のマニュアルを参照してください。

ネイティブ コネクタ (最も一般的なコネクタ)

ネイティブ コネクタを使用して Netscape の Web サーバーに接続する場合、JRun は、次のように obj.conf ファイルに変更を加えます。

- JRun NSAPI フィルタ の初期化を追加して、初期化パラメータを設定します。NSAPI フィルタは、Web サーバーへの HTTP リクエストを阻止して、JRun が処理するために適切な HTTP リクエストを JRun に渡します。obj.conf ファイルの一般的な初期化は、次のようになります。

```
Init fn="load-modules" shlib="C:/JRun/connectors/nsapi/intel-win/
     jrun_nsapi35.dll" funcs="jruninit,jrunfilter,jrunservice"
Init proxyport="51000" verbose="false" proxyhost="127.0.0.1"
     timeout="300" rulespath="C:/JRun/servers/default/
     local.properties" fn="jruninit"
```

- JRun オブジェクト定義を追加します。オブジェクト定義は、特定のリソースに適用される命令のグループ化です。一般的な JRun オブジェクト定義は、次のようになります。

```
<Object name="jrun">
PathCheck fn="jrunfilter"
Service fn="jrunservice"
</Object>
```

- 次の命令を既定のオブジェクト定義に追加します。

```
NameTrans fn="jrunfilter"
```

NameTrans 命令は、URL をホスト マシン上の物理パスにマッピングします。ただし JRun においては、NameTrans 命令は、jrunfilter の部分パスを指定するため、サーバーは PathCheck 命令がその部分パスに一致するオブジェクト (この場合、jrun オブジェクト) を処理します。

- NES が JRun を無効にしないように、/servlet 用の URL からディレクトリへの既定のマッピング行をコメント アウトします。

```
#NameTrans fn="pfx2dir" from="/servlet"
    dir="C:/Netscape/Server4/docs/servlet" name="ServletByExt"
```

Java (非ネイティブ) コネクタ

Java (非ネイティブ) コネクタを使用して Netscape の Web サーバーに接続する場合、JRun は、次のように obj.conf ファイルに変更を加えます。

- init クラス パス行の前に jrun.jar を追加します。
- JRun オブジェクト定義を追加します。オブジェクト定義は、特定のリソースに適用される命令のグループ化です。Java コネクタを使用する場合の一般的な JRun オブジェクト定義は、次のようになります。

  ```
  <Object name="jrun">
  Service fn="java-run" class="com/allaire/jrun/connector/
      JRunConnector" proxyhost="127.0.0.1" proxyport="51000"
      timeout="300"
  </Object>
  ```

- NES が JRun を無効にしないように、/servlet 用の URL からディレクトリへの既定のマッピング行をコメント アウトします。

  ```
  #NameTrans fn="pfx2dir" from="/servlet"
      dir="C:/Netscape/Server4/docs/servlet" name="ServletByExt"
  ```

- 次の命令を既定のオブジェクト定義に追加します。

  ```
  NameTrans name="jrun" from="*.shtml" fn="assign-name"
  NameTrans name="jrun" from="*.jsp" fn="assign-name"
  NameTrans name="jrun" from="/servlet/*" fn="assign-name"
  ```

NameTrans 命令は、URL をホスト マシン上の物理パスにマッピングします。ただし JRun の場合、NameTrans 命令は、jrun オブジェクトを指定して関連するパターンを処理します。

サンプル obj.conf ファイル

次の例は、obj.conf ファイルのサンプルです。変更されたセクションは太字で示してあります。このファイルには、ネイティブ コネクタを使用する NES の実装に一般的な変更が含まれています。

サンプル obj.conf ファイル

```
# Netscape Communications Corporation - obj.conf
# Use only forward slashes in pathnames--backslashes can cause
# problems.See the documentation for more information.

Init fn=flex-init access="C:/Netscape/SuiteSpot/https-tford1 /logs/access" format.access="%Ses->client.ip%
- %Req->vars.auth-user% [%SYSDATE%] \"%Req->reqpb.clf-request%\" %Req->srvhdrs.clf-status% %Req-
>srvhdrs.content-length%"
Init fn=load-types mime-types=mime.types

Init fn="load-modules" shlib="C:/JRun/connectors/nsapi/intel-win/jrun_nsapi35.dll"
funcs="jruninit,jrunfilter,jrunservice"
Init proxyport="51000" verbose="false" proxyhost="127.0.0.1" timeout="300"
rulespath="C:/JRun/jsm-default/services/jse/properties/rules.properties" fn="jruninit"

<Object name="default">
Nametrans fn="jrunfilter"
NameTrans fn=pfx2dir from=/ns-icons dir="C:/Netscape/SuiteSpot/ns-icons"
NameTrans fn=pfx2dir from=/mc-icons dir="C:/Netscape/SuiteSpot/ns-icons"
NameTrans fn="pfx2dir" from="/help" dir="C:/Netscape/SuiteSpot/manual/https/ug"
NameTrans fn=document-root root="C:/Netscape/SuiteSpot/docs"
PathCheck fn=nt-uri-clean
PathCheck fn="check-acl" acl="default"
PathCheck fn=find-pathinfo
PathCheck fn=find-index index-names="index.html,home.html"
ObjectType fn=type-by-extension
ObjectType fn=force-type type=text/plain
Service method=(GET|HEAD) type=magnus-internal/imagemap fn=imagemap
Service method=(GET|HEAD) type=magnus-internal/directory fn=index-common
Service method=(GET|HEAD) type=*~magnus-internal/* fn=send-file
AddLog fn=flex-log name="access"
</Object>

<Object name=cgi>
ObjectType fn=force-type type=magnus-internal/cgi
Service fn=send-cgi
</Object>

<Object name="jrun">
PathCheck fn="jrunfilter"
Service fn="jrunservice"
</Object>
```

WebSite Pro の構成

ここでは、O'Reilly の WebSite Pro Web サーバーを構成する方法について説明します。JRun が WebSite Pro と通信するには、ここで説明するすべての構成手順を実行する必要があります。高度な Web サーバー接続テクニックの詳細については、159 ページの第 4 章「コネクタ」を参照してください。

構成手順を開始する前に、WebSite Pro CD から WebSite Pro をインストールし、O'Reilly の Web サイトから WebSite Pro の最新のパッチをインストールしていることを確認してください。次に、JRun をインストールし、WebSite Pro 用のインストールに関するすべての手順に従ってください。

JRun を WebSite Pro に接続するには、次の手順が必要です。

- 68 ページの「サーブレットを実行するための URL 接頭部のマッピング」
- 70 ページの「マルチ ホームおよび URL 接頭辞」
- 71 ページの「ファイル拡張子の JRun へのマッピング」
- 73 ページの「 WebSite Pro と通信するための JRun の構成」

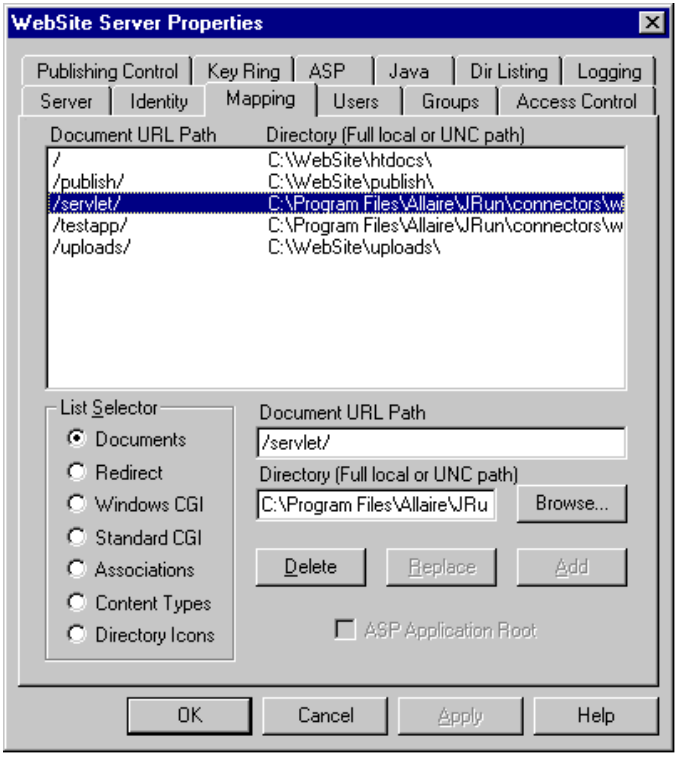
この章では、JMC 内からのコネクタ ウィザードの起動方法について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

サーブレットを実行するための URL 接頭部のマッピング

WebSite Pro では、**サーブレット** を実行するために特定の URL 接頭部をマッピングするなど、サーバーの構成を高度にカスタマイズすることができます。 たとえば、http://yourhost.com/servlet/SampleServlet 経由でサーブレットを実行するには、\servlet\ 用の Documents マッピングを WebSite Pro に追加する必要があります。

URL 接頭部を JRun にマッピングするには

1. WebSite Server Properties アプリケーションを起動します。
2. [マッピング] タブをクリックします。



3. [ドキュメント URL のパス] および [ディレクトリ] フィールドを編集します。これにより、\servlet\<servlet_name > URL 経由でサーブレットを実行することができます。

   たとえば、[ドキュメント URL のパス] フィールド内の \servlet\ マッピングは、[ディレクトリ] フィールド内の C:\Program Files\Allaire\JRun\connectors\wsapi\intel-win\jrun.isa\servlet\ をポイントすることも可能です。ファイル名を [ディレクトリ] フィールドの最後に手作業で入力する必要があります。

   WebSite Pro で複数の ID を使用する場合は、70 ページの「マルチ ホームおよび URL 接頭辞」に記載されているこのパスの設定に関する追加情報を参照してください。

4. 特定の **サーブレット**を URL にマッピングする場合は、ここで説明した手順と同じ手順を使用しますが、**サーブレット** 名をパスの最後に追加します。たとえば、次のように設定します。

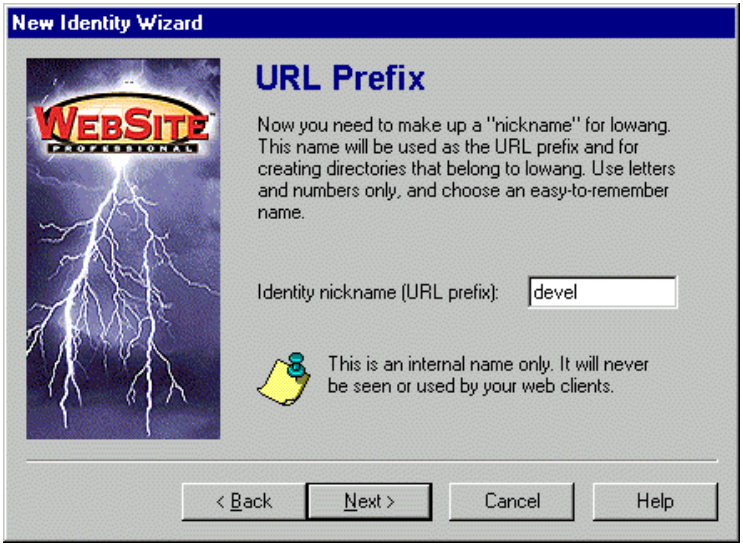
   C:\Program Files\Allaire\JRun\connectors\wsapi\intel-win\jrun.isa\servlet\SnoopServlet

5. [OK] をクリックします。

6. Web サーバーを再起動します。

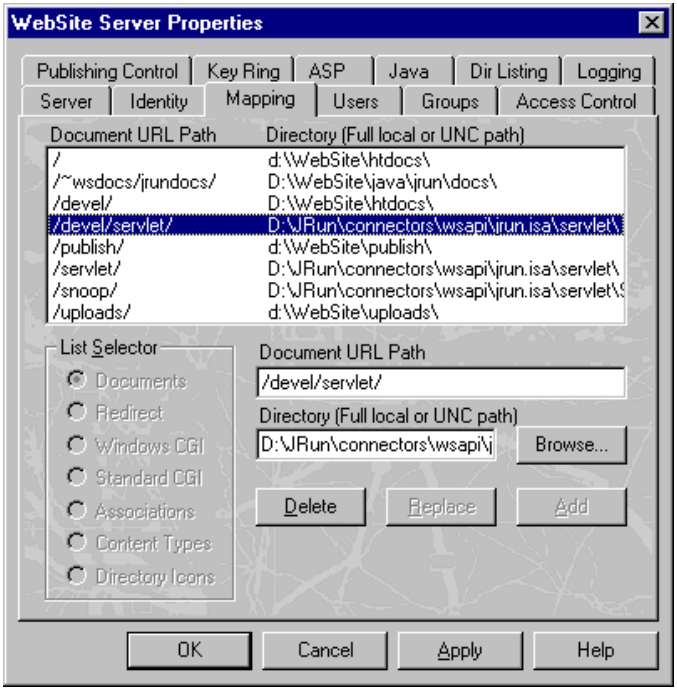
マルチ ホームおよび URL 接頭辞

WebSite Pro で複数の ID を使用する場合は、ID に割り当てられるニックネームを URL マッピングの前に追加する必要があります。ID を設定する際に、ニックネームを要求されます。



このニックネームは、[ID] タブの [URL 接頭部] フィールドにも表示されます。

このニックネームは、Document URL Path マッピングの前に追加されます。たとえば、**サーブレット** を実行するために /devel ID をマッピングするには、Document URL Path として /devel/servlet/ を入力します。



ファイル拡張子の JRun へのマッピング

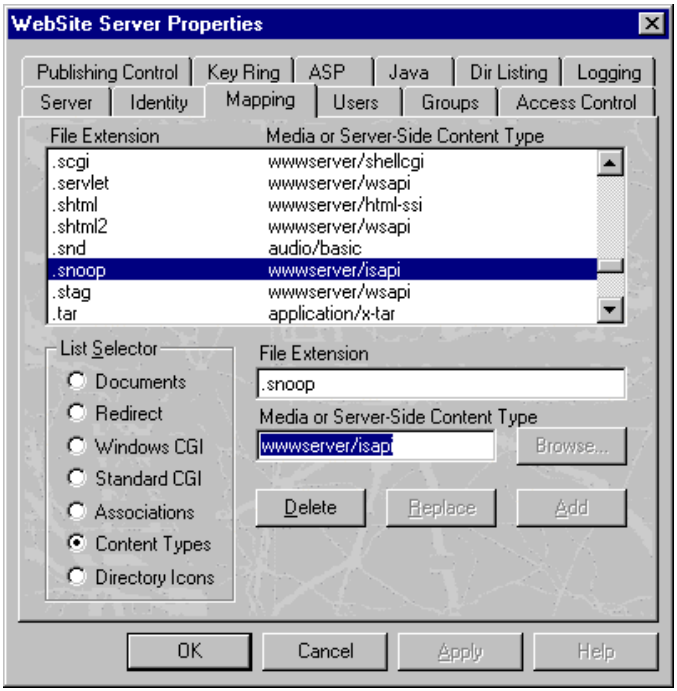
指定した拡張子で JRun を起動できるように WebSite サーバーを構成するには、2 つの手順が必要です。まず、その WebSite Server Properties アプリケーションを使用して、WebSite をセットアップする必要があります。次に、JRun 管理コンソール を使用して、そのマッピングを JRun に追加する必要があります。

ここでは、ファイル拡張子をマッピングするために WebSite を構成する手順について説明します。ファイル拡張子を JRun にマッピングする方法については、144 ページの「サーブレットへのリクエストマッピング」を参照してください。

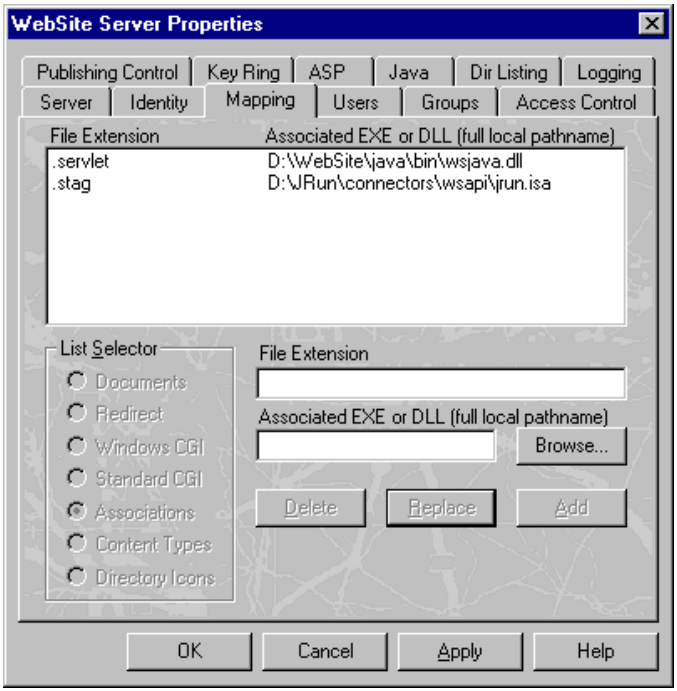
ファイル拡張子のマッピングを追加するには

1. WebSite Server Properties アプリケーションを起動します。
2. [マッピング] タブをクリックします。
3. [List Selector] ボックスの [Content Types] を選択して、wwwserver/isapi にマッピングする拡張子のエントリを追加します。

   次の図は、.snoop 拡張子の例を示しています。



4. [List Selector] ボックスの [Associations] を選択して、次の図に示すように .snoop を jrun.isa ファイルの場所にマッピングするエントリを追加します。



5. [OK] をクリックします。
6. Web サーバーを再起動します。

WebSite Pro と通信するための JRun の構成

ここでは、JRun と通信できるように WebSite Pro を構成する方法について説明します。

JRun と WebSite Pro を接続するには

1. Web サーバーを停止します。
2. 次のいずれかの方法で JMC を起動します。
   - [スタート] > [プログラム] > [JRun 3.0] > [JRun 管理コンソール] をクリックします。
   - Web ブラウザで、次の URL を開きます。

     http://localhost:8000

     この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。
3. JRun 管理者として JMC にログインします。
4. アクセス バーの [コネクタ ウィザード] を選択します。

5. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	WebSite Pro に接続する JRun サーバーを選択します。通常は、default サーバーを選択します。 JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [WebSite Pro] を選択します。
	Web サーバーのバージョン	ドロップダウン リストから WebSite Pro のバージョンを選択します。
	Web サーバーのプラットフォーム	WebSite Pro を実行しているプラットフォームを選択します。
手順 2	JRun サーバーの IP アドレス	WebSite Pro への接続に使用する JRun サーバーの IP アドレスを入力します。WebSite Pro と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバーの接続ポート	JRun サーバーが WebSite Pro との通信に使用するポートを指定します。このポートと WebSite Pro の HTTP ポートを混同しないでください。デフォルトは 51000 です。

6. JRun コネクタのインストールが完了したら、WebSite Pro Web サーバーと default JRun サーバーを再起動します。

   規定 JRun サーバーがまだ起動されていない場合には、次の手順に従ってください。

   - JRun をアプリケーションとしてインストールした場合には、[スタート] > [プログラム] > [JRun 3.0 > JRun default Server] を選択します。

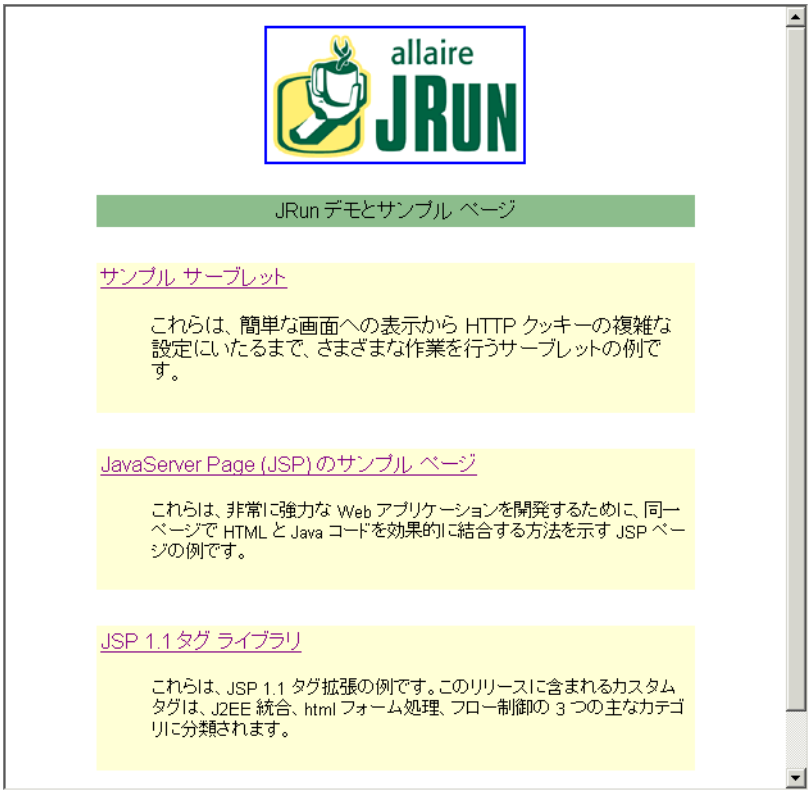
- JRun をサービスとしてインストールした場合には、コントロール パネルのサービス コントロール管理ツールで「JRun default Server」サービスを起動するか、次の JRun コマンドライン ユーティリティを使用します。

  ```
  jrun -start default
  ```

7. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と WebSite Pro Web サーバーの接続を確認します。

   ```
   http://localhost:80/demo/index.html
   ```

   これは、WebSite Pro が既定のポート 80 で接続をリスニングしていることを想定しています。



デモ アプリケーションが実行される場合、JRun と WebSite Pro Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

WebSite Pro 構成ファイルへの変更

JRun コネクタ ウィザードは、WebSite Pro Web サーバーの構成ファイルを変更します。本書で詳しく説明されていない設定に変更を加える場合は、WebSite Pro Server Properties インターフェイスを使用してください。

コネクタ ウィザードは、JRun local.properties ファイルも変更します。詳細については、81 ページの「 local.properties への変更」を参照してください。

Java ベースの Web サーバーの構成

JRun は、本書に特定のサーバーに関する手順が説明されているかどうかにかかわらず、大部分の Java Web サーバーと通信することができます。ここでは、この章の他のセクションで詳しく説明されていない Java ベースの Web サーバーを構成する方法について説明します。高度な Web サーバー接続テクニックの詳細については、159 ページの第 4 章「コネクタ」を参照してください。

JRun と Java ベースの Web サーバーを接続するには

1. jrun.jar ファイルを配布ディレクトリから Web サーバーの /lib/classes ディレクトリにコピーします。(アプリケーションによっては、jrun.jar をクラス パスに追加する必要があります。Web サーバーに付属するマニュアルを参照してください。)
2. 次のクラスをポイントする JRunConnector というエイリアスを作成します。

   allaire.jrun.connector.JRunConnector
3. 次のように 2 つの初期化パラメータを設定します。

   proxyhost=localhost
   proxyport=51000

   これらの値は既定値であり、別の値を使用する場合以外は明示的に設定する必要はありません。異なるポートを JRun に設定している場合は、ポート番号を変更する必要があります。
4. Web サーバーの管理インターフェイスを使用して、適切な HTTP リクエストを JRun にマッピングします。たとえば、次のように設定します。

   /servlet=JRunConnector
   *.jsp=JRunConnector
   /msservlets=JRunConnector

   これらのマッピングに一致するすべての要求は、JRun に転送されます。

サーブレットの実行用 CGI インターフェイスの構成

JRun は、CGI を使用する Web サーバー用の Perl コネクタをサポートしています。Perl コネクタを使用するために Web サーバーを構成するには、JRun と一緒に提供されている jrun.pl Perl スクリプトを使用します。このスクリプトは、<JRun_directory>/connectors/perl5/ にあります。

CGI スクリプトの実行に使用する任意のディレクトリに jrun.pl をコピーします。たとえば、CGI ディレクトリが cgi-bin の場合、次のように サーブレットを実行することができます。

http://host/cgi-bin/jrun.pl/servlet/SnoopServlet

この例では、/servlet/SnoopServlet が JRun に渡されて、JRun は サーブレットを呼び出します。サーブレット の出力は、CGI スクリプトによって返されます。

次に別の例を示します。

http://host/cgi-bin/jrun.pl/yourpage.jsp

この例では、JRun 上の /yourpage.jsp が呼び出され、結果が返されます。

jrun.pl スクリプトは、JRun サーバーとの接続方法を確認するために JRUNPROXY 環境変数を検索します。この値には、次の形式が使用されている必要があります。

<IP アドレス >:< ポート >

既定値は次のとおりです。127.0.0.1:51000

このスクリプトは、特定の環境向けにカスタマイズすることができます。JRun プロキシ アドレスを変更したり、スクリプトの該当するサブルーチンを修正してエラー応答を変更することができます。

Zeus Web サーバーの構成

Zeus Web サーバーを使用している場合は、JRun と接続するように Web サーバーを構成するために、以下の手順を実行してください。高度な Web サーバー接続テクニックの詳細については、159 ページの第 4 章「コネクタ」を参照してください。

この章では、JMC 内からのコネクタ ウィザードの起動について説明するため、コネクタ ウィザードを JRun インストールの一部として実行している場合には、この章を読む必要はありません。

JRun と Zeus を接続するには

1. 管理ログインを使用して Zeus にログインします。
2. [Module config] を選択します。
3. [Distributed] オプションをオンにします。
4. [Distributed] リンクを選択します。

5. [Java Servlets] オプションを次のように設定します。

   Servlet prefs: /servlet
   Servlet Server:ip_address:port_number

   ip_address を Web サーバーのホストの IP アドレスに設定します。Web サーバーと JRun が同じホスト マシン上にある場合は、ip_address を 127.0.0.1 に設定します。

   port_number を Zeus と JRun が通信するポートに設定します。既定値は、51000 です。

6. 次のいずれかの方法で JMC を起動します。
   - [スタート] > [プログラム] > [JRun 3.0] > [JRun 管理コンソール] をクリックします。
   - Web ブラウザで、次の URL を開きます。

     http://localhost:8000

   この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

7. JRun 管理者として JMC にログインします。
8. アクセス バーの [コネクタ ウィザード] を選択します。
9. 次の表に示すように、コネクタ ウィザードで必要な構成情報を指定します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 1	JRun サーバーの名前	Web サーバーに接続する JRun サーバーを選択します。通常は、default サーバーを選択します。 JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。
	Web サーバーの種類	ドロップダウン リストから [Zeus Web サーバー] を選択します。
	Web サーバーのバージョン	ドロップダウン リストから Web サーバーのバージョンを選択します。

JRun Connection Module の設定		
コネクタ ウィザードの手順	**パラメータ**	**説明**
手順 2	JRun サーバーの IP アドレス	Zeus への接続に使用する JRun サーバーの IP アドレスを入力します。Zeus と JRun サーバーの IP アドレスが同じであれば、既定値 127.0.0.1 をそのまま使用してください。
	JRun サーバーの接続ポート	JRun サーバーが Zeus との通信に使用するポートを指定します。このポートと Zeus の HTTP ポートを混同しないでください。デフォルトは 51000 です。

10. JRun コネクタのインストールが完了したら、Zeus Web サーバーと default JRun サーバーを再起動します。

    default JRun サーバーがまだ起動されていない場合には、次の手順に従ってください。

    - JRun をアプリケーションとしてインストールした場合には、[スタート] > [プログラム] > [JRun 3.0 > JRun default Server] を選択します。
    - JRun をサービスとしてインストールした場合には、コントロール パネルのサービス コントロール管理ツールで「JRun default Server」サービスを起動するか、次の JRun コマンドライン ユーティリティを使用します。

      ```
      jrun –start default
      ```

11. 次の URL を使用して JRun デモ アプリケーションを実行し、JRun と Zeus Web サーバーの接続を確認します。

    ```
    http://localhost:80/demo/index.html
    ```

    これは、Zeus が既定のポート 80 で接続をリスニングしていることを想定しています。

デモ アプリケーションが実行される場合、JRun と Zeus Web サーバーの接続は正常に構成されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

Zeus 構成ファイルへの変更

JRun コネクタ ウィザードでは、Zeus Web サーバーの構成ファイルを変更します。本書で詳しく説明されていない設定に変更を加える場合は、Zeus の管理インターフェイスを使用してください。

コネクタ ウィザードは、JRun local.properties ファイルも更新します。詳細については、81 ページの「 local.properties への変更」を参照してください。

local.properties への変更

Web サーバーの構成ファイルの他に、JRun コネクタ ウィザードは、次のように JRun サーバーの local.properties ファイルに変更を加えます。

- 次の行を local.properties の最後に追加して、JRun コネクタが正常にインストールされたことを示します。

  ranConnector=yes

- jcpservices セクションの jcp.endpoint.main.port を、コネクタ ウィザード実行時に指定したポートに設定します。これは、JRun サーバー コネクタ ポートで、プロキシ ポート ともいいます。

トラブルシューティング

ここでは、JRun の外部 Web サーバーへの接続に関連する多くの一般的な問題を解決するために役立つ情報を提供します。JRun インストールのトラブルシューティングについては、37 ページの「トラブルシューティング」を参照してください。

JRun のトラブルシューティングを行う際、<JRun_directory>/logs に置かれているログファイルをチェックすることで、追加情報を得ることができます。

JRun コネクタ ウィザードの使用

JRun コネクタ ウィザードを使用すると、Web サーバーとの接続を簡単に構成することができます。ここでは、ウィザードの実行中に発生する可能性がある一般的なエラーについていくつか説明します。
エラーの内容 :

「httpd.conf にアクセスできません」
「obj.confile をロードできません」
「JRun ISAPI フィルタ コピー時のエラー」
解決方法 :

- コネクタ ウィザードの手順 3 で入力した Web サーバーの構成ファイルへのパスが間違っているか、またはパスが入力されていない可能性があります。コネクタ ウィザードに戻り、正しいパスを入力してください。
- Web サーバーを停止して、コネクタ ウィザードに戻ってください。

JRun デモ アプリケーションのテスト

コネクタ ウィザードで Web サーバーと JRun とのコネクタをインストールした後、次の URL を使用して、JRun デモ アプリケーションを実行し、接続を確認します。

http://localhost:80/demo/index.html

ここでは、このデモ アプリケーションのテスト中に発生する可能性がある一般的なエラーについていくつか説明します。

HTTP エラー

"404 File Not Found" エラー
"The page cannot be found"
"500 Internal Server Error"
「JRun サーバーに接続できません」

解決方法 :

- JRun default Server が実行されていることを確認してください。特に指定しない限り、デモ アプリケーションはこのサーバーで実行されます。
    - Windows の場合は、システム トレイを確認してください。JRun をアプリケーションとしてインストールした場合には、JRun サーバー 3.0 アイコンが表示されます。JRun をサービスとしてインストールした場合には、コントロール パネルのサービス コントロール管理ツールで JRun Default Server サービスが実行されているかどうかを確認してください。
    - UNIX の場合は、/opt/bin の次のコマンドライン ツールを使用して、サーバーが実行されているかどうかを確認してください。

        jrun –status default
- default JRun サーバーがすでに実行されている場合には、コネクタ ウィザードを実行した後、再起動してください。
- 要求 URL のポート番号を確認してください。規定の HTTP ポートは 80 ですが、Web サーバーが別のポートをリスニングしている場合には、そのサーバーを URL に指定する必要があります。たとえば、Web サーバーが 8080 をリスニングしている場合にデモアプリケーションにアクセスするには、http://localhost:8080/demo/index.html と入力します。
- JRun コネクタ ウィザードの使用中に、Web サーバーを実行していないかどうか確認してください。
- /JRun/servers/default　ディレクトリおよびそのサブディレクトリの読み取りアクセス権があることを確認してください。

並行処理エラー

エラーの内容 :

"Too Many Concurrent requests"

"Reverting to Developer Edition" in JMC

解決方法 :

- ライセンスをアップグレードしてください。 無償ダウンロードと JRun Developer の最大同時接続数は 3 です。詳細については、2 ページの「 JRun 製品の種類」を参照してください。
- ベータ版または評価版の JRun を使用している場合、使用期限が切れるとこのメッセージが表示されます。この場合、最新の評価版または製品版にアップグレードしてください。

プロセス エラー

jrun コマンドを記述中にロックされ、システムに入れない場合

オンラインで子プロセスを作成するのではなく、-nohup オプション (UNIX のみ) を使って、サーバーのプロセスを新規作成してください。このオプションは、& と同じはたらきをします。たとえば、次のように設定します。

jrun -nohup 既定値

-nohup オプションを使っていないときにオンラインでロックされた場合には、Ctrl + z を押してからプロンプトで bg と入力することで、JRun サーバー プロセスをバックグラウンドに移動してください。

-start および -nohup オプションについては、96 ページの「 jrun コマンドの使用」を参照してください。

第 3 章

JRun 管理コンソール

JRun 管理コンソール (JMC) はブラウザベースのユーティリティです。JRun でさまざまな設定を構成するには、JMC を使用します。この章では、JMC の概要と、JMC で実行できる機能について説明します。

内容

JRun 管理コンソールの開始

JRun には、JRun 管理コンソール (JMC) というブラウザベースの Web アプリケーションが用意されています。このユーティリティを使用して、JRun 環境や、JRun と使用している Web サーバーの接続を設定することができます。デフォルトで、JMC は Admin JRun サーバー上で実行されます。

このセクションでは、JMC の開始方法と、コンソールの基本レイアウト、および機能について説明します。

メモ JMC は JSP ベースであるため、アクセスするには Netscape Communicator バージョン 4.0 以降、または Internet Explorer バージョン 4.0 以降のいずれかが必要です。

JMC を開始するには

1. JRun サーバーが実行されていない場合は、Admin で JRun server を開始します。JRun サーバーの開始については、35 ページの「JRun サーバーの起動と停止」を参照してください。
2. Web ブラウザで次の URL を開いて、JMC を開始します。

   http://localhost:8000

   または Windows のみ : **[スタート] > [プログラム] > [JRun 3.0] > [JRun 管理コンソール]** をクリックします。

   メモ この手順は、JRun が提供する Web サーバーを既定のポート (8000) で使用して、JMC に接続する場合を想定しています。

   JMC を初めて起動した場合は、JRun 使用許諾契約書が表示されます。JMC が表示されない場合は、37 ページの「トラブルシューティング」を参照してください。
3. JRun 使用許諾契約に同意します。使用許諾契約に 1 度同意すれば、2 度目からは表示されません。

JMC のログインウィンドウが表示されます。

4. 表示されたフィールドにユーザ名とパスワードを入力し、[ログイン] をクリックします。既定のユーザ名は admin です。admin 用のパスワードはインストール 時に設定した内容です。

JRun 管理コンソール ウィンドウは [ようこそ] ページを開いたときに表示されます。

このウィンドウには 2 つのペインがあります。左側ペインにはマシン レベルからの JRun オブジェクト階層がツリーで表示されます。右側ペインには、現在、ツリーで選択しているフォルダやオブジェクトの中身が表示されます。ペインの上にはアクセス バーがあります。このバーには、JRun サーバー固有ではないコマンドが表示されます。

admin 権限を持っていない場合は、アクセス バーにすべてのオプションは表示されず、またツリーにある全オブジェクトにアクセスすることもできません。

このツリーには Change Password のような機能や、serial_number のようなプロパティなどのオブジェクトがあります。

JRun 管理コンソールの使用

コンソールを使用する場合は、次の点に注意してください。

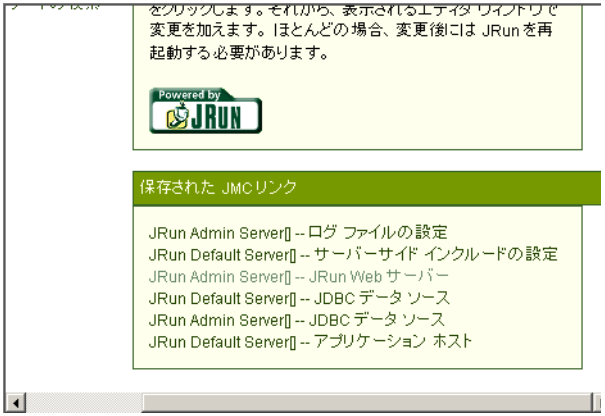
- 左側ペインに表示されているフォルダの内容をプレビューするには、フォルダの前にあるプラス (+) 記号をクリックします。
- 左側ペインで開かれているフォルダを閉じるには、マイナス (-) 記号をクリックします。
- 右側ペインにあるフォルダの内容を表示するには、目的のフォルダをクリックします。本書では、> 記号はフォルダ、サブフォルダ、ファイルやオブジェクトのレベルを表します。また、斜体は変数情報を表します。たとえば、「[*Select*

machine_name] > [*JRun_server_name*] > [*application_name*] > [**ログの設定**]」のように使用されます。

- オブジェクトを選択するには、目的のオブジェクトをクリックします。右側ペインに選択したオブジェクトのプロパティが表示されます。または、JRun により選択した機能が実行されます。プロパティを編集するには、右側ペインの [編集] ボタン、または [プロパティ] をクリックします。[エディタ] ウィンドウが表示されるので、そこで変更を行います。
- ほとんどの場合、JRun サーバーのプロパティに変更を加えた後は、その変更を有効にするために JRun サーバーを再起動する必要があります。

JMC のお気に入りの設定

JMC では、一般的に使われる編集ウィンドウのリストを [ようこそ] ページに追加することができます。たとえば、特定の Web アプリケーションのセッションの設定を頻繁に変更する場合、オブジェクト エクスプローラをいくつも表示しなくても、[ようこそ] ページで一度クリックするだけでそのページを取得できるように、リンクを追加することができます。



JMC の特定のパネルにリンクを追加するには、そのパネルの右下にある [ようこそページへ追加] をクリックします。JRun は、パネルが追加されたことを確認します。次回に [ようこそ] ページを表示すると、保存された JMC リンク セクションでそのリンクが表示されます。

JRun シリアル番号の設定

JRun シリアル番号は、実行中の JRun のバージョンを定義します。JRun Developer か無料の評価版をインストールした場合、シリアル番号は空欄です。JRun ライセンスを

アップグレードする場合は、JMC にあるシリアル番号を変更して制限を解除することができます。

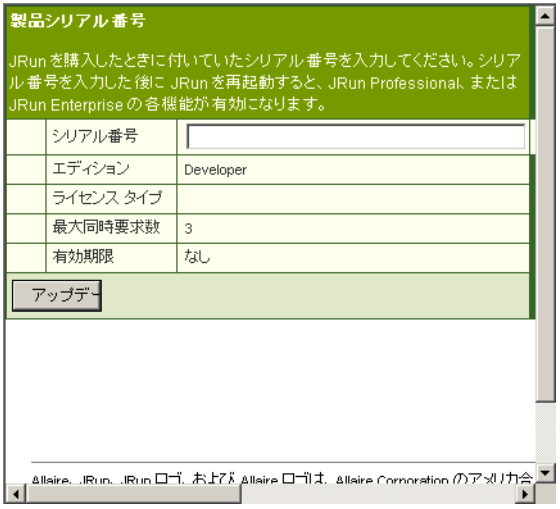
このセクションでは、JRun シリアル番号の変更方法について説明します。JRun ライセンス購入の詳細については、14 ページの「 JRun 関連のコンタクト先」を参照してください。

メモ [**シリアル番号**] プロパティを変更できるのは、admin としてログインしたユーザのみです。

シリアル番号を修正するには

1. アクセス バーで [**シリアル番号**] を選択します。アクセス バーは JMC ペイン上部に横長に表示されます。

   [製品シリアル番号] パネルが表示されます。



2. [**シリアル番号**] フィールドに、提供されたシリアル番号を正確に入力します。
3. 変更を適用するには、[**更新**] をクリックします。
4. JRun サーバーを再起動します。

JMC ユーザの管理

JMC には JMC ユーザを管理するためのユーティリティが用意されています。JRun アドミニストレータでは、このユーティリティを使って JRun サーバーによるアクセスを制限することができます。たとえば、ISP で使用している場合、個々のカスタマに専用の JRun サーバーと、このサーバーの設定のみにアクセスする権利を与えることができます。この場合、ユーザは JMC にある JRun サーバーのうち、アクセス権を与えられてい

るものだけを参照できます。このセクションでは、ユーザの追加と削除、ユーザ設定の変更などを行う方法について説明します。

現在のユーザに対する変更を行った場合、変更を有効にするには、いったんログアウトし、もう一度ログインする必要があります。また、別のユーザに対する変更を行った場合も、変更を行ったユーザがログアウトし、変更を加えられたユーザ自身がログインするまで、変更は有効になりません。

現在どのユーザとしてログインしているかを調べるには、JMC アクセス バーの左側で確認します。初めてログインしたときには、(admin) と表示されています。

新規 JMC ユーザの追加

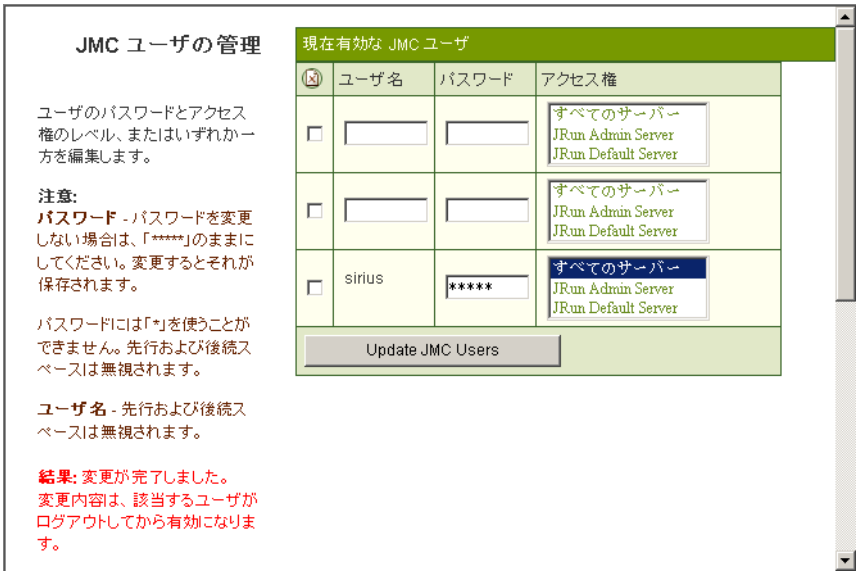
[JMC ユーザの管理] オプションを使用して、異なるレベルのアクセス権を持つユーザを JMC に追加することができます。個々のユーザに対し、JRun サーバーに部分的または完全なアクセス権を与えることができるほか、完全にアクセス不可能に設定することもできます。

メモ [JMC ユーザの管理] オプションにアクセスできるのは、admin としてログインしたユーザのみです。

新規ユーザを追加するには

1. アクセス バーで [JMC ユーザの管理] を選択します。

   右側ペインに [JMC ユーザの管理] パネルが表示されます。



2. [**ユーザ名**] フィールドに新規ユーザの名前を入力します。

メモ [**ユーザ名**] に空白文字を使用できます。たとえば、「Nick Danger」は有効な名前です。名前の前後につけられた空白文字は削除されます。

3. [**パスワード**] フィールドに新規ユーザのパスワードを入力します。新規ユーザのパスワードを入力する場合は、次の点に注意してください。
   - [JMC ユーザの管理] パネルに入力している間、パスワードはアスタリスクで表示され、他のユーザにはわからないようになっています。
   - パスワードは最低 1 文字です。
   - パスワードにスペースやアスタリスク (*) を使用することはできません。
4. 新規ユーザに対しアクセスを許可する JRun サーバーを、[**アクセス権**] **リスト**ボックスから選択します。複数の JRun サーバーを選択するには、最初のサーバーをクリックし、Ctrl キーを押したまま、残りの JRun サーバーを選択します。
5. 変更を適用するには、[JMC ユーザの更新] をクリックします。

JMC ユーザの設定の変更

[JMC ユーザの管理] オプションを使用して、ユーザのパスワードや、ユーザがアクセスできる JRun サーバーを変更することができます。admin ユーザに対する設定を [JMC ユーザの管理] オプションで変更することはできません。

メモ [JMC ユーザの管理] オプションにアクセスできるのは、admin としてログインしたユーザのみです。

ユーザ設定を変更するには

1. アクセス バーで [**JMC ユーザの管理**] を選択します。

   右側ペインに [JMC ユーザの管理] が表示されます。
2. JRun サーバーに対するユーザのアクセス権を変更するには、このユーザに対応する [**アクセス権**] リストボックスで、目的の JRun サーバーをクリックします。複数の JRun サーバーを選択するには、最初のサーバーをクリックし、Ctrl キーを押したまま、残りの JRun サーバーを選択します。
3. ユーザのパスワードを変更するには、[**パスワード**] フィールドでアスタリスクで表示されたパスワードを選択し、新しいパスワードを入力します。新規パスワードを入力する場合は、次の点に注意してください。
   - [JMC ユーザの管理] に入力する場合、パスワードはアスタリスクで表示され、他のユーザにはわからないようになっています。
   - パスワードは最低 1 文字です。
   - パスワードにスペースやアスタリスク (*) を使用することはできません。
4. 変更を適用するには、[JMC ユーザの更新] をクリックします。

これらの変更は、次にユーザがログインしたときに有効になります。

JMC ユーザの削除

[JMC ユーザの管理] オプションを使用して、admin 以外のどのようなユーザでも JRun から削除することができます。[JMC ユーザの管理] オプションにアクセスできるのは、admin としてログインしたユーザのみです。

メモ ユーザを削除する前に、これらのユーザが JMC からログアウトしていることを確認してください。

ユーザを削除するには

1. アクセス バーで [JMC ユーザの管理] を選択します。

   右側ペインに [JMC ユーザの管理] パネルが表示されます。

2. 削除するユーザの [ユーザの削除] チェックボックス  をオンにします。一度に複数のユーザを選択して、削除することができます。
3. 変更を適用するには、[JMC ユーザの更新] をクリックします。

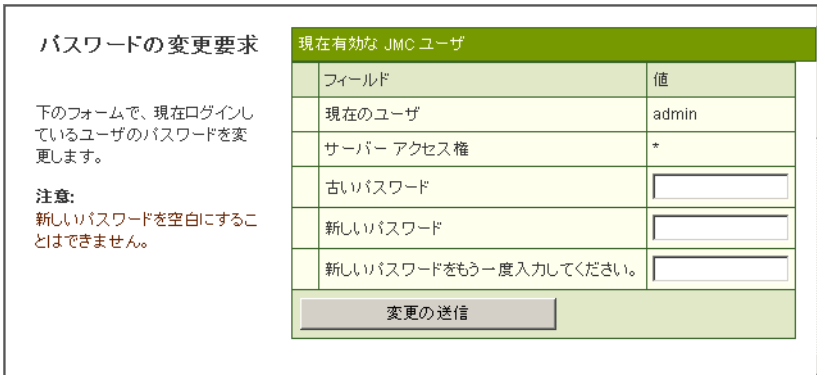
パスワードの変更

JMC の [パスワード の変更] オプション を使用して、現在ログインしているユーザのパスワード を変更することができます。admin としてログインしている場合は、[JMC ユーザの管理] オプション を使用すると 、他のどのユーザのパスワード でも 変更すること ができます。詳細については、92 ページの「 JMC ユーザの設定の変更」を参照してください。

自分のパスワードを変更するには

1. アクセス バーで **[パスワードの変更]** を選択します。

   右側ペインに [パスワード変更リクエスト] パネルが開き、現在ログインしているユーザとそのユーザの JRun サーバーに対するパーミッションが表示されます。



[サーバー アクセス権] フィールドにアスタリスク「*」が表示されている場合は、そのユーザがアクセス権をすべて持っていることを表します。

2. **[現在のパスワード]** フィールドに、変更前のパスワードを入力します。
3. **[新規パスワード]** フィールドに、新しいパスワードを入力し、**[新規 パスワード 再入力]** フィールドで新規パスワードを確認します。新規パスワードを入力する場合は、次の点を考慮してください。
   - **[パスワード変更リクエスト]** ウィンドウに入力している間、パスワードはアスタリスクで表示され、他のユーザにはわからないようになっています。
   - パスワードにスペースやアスタリスク (*) を使用することはできません。
   - パスワードは最低 1 文字です。
4. 変更を適用するには、[変更の送信] をクリックします。

JRun サーバーの設定

JRun サーバーは JRun アーキテクチャの核となります。JRun サーバーの機能は次のとおりです。

- Web アプリケーションを論理的にグループ化する方法の提供。JRun サーバーで実行することができる Web アプリケーションの数に制限はありません。
- JRun Connection Module を経由した、内部および外部 Web サーバーと Web アプリケーションとの接続。
- Web サーバーの安定性の維持。JRun サーバーはそれぞれ、独立したプロセスとして実行されます。JRun サーバーにより提供されるサービスもプロセスに組み込まれていません。
- Enterprise JavaBeans を使ったビジネス ロジックの実装。

JRun インストールは、次の 2 つの JRun サーバーをセットアップします。つまり、「default」と「admin」の 2 つです。JMC では、default サーバーは [JRun Default Server]、admin サーバーは [JRun Admin Server] と表示されます。

admin サーバーを使えば、**JRun 管理コンソール** アプリケーション (jmc-app) にアクセスできます。default サーバーは、ユーザがすぐに立ち上げて実行できるサーバーで、空の **デフォルト ユーザ アプリケーション** (default-app) と **JRun デモ** (demo-app) を含んでいます。

メモ Windows NT では、JRun サーバーは、NT サービスとして実行することも、あるいはアプリケーションとして実行することもできます。サービスとして実行されるときの JRun Admin Server や JRun default サーバーは、コンピュータが起動するたびに起動して、ユーザ処理としてでなくシステム処理として実行されます。アプリケーションとして実行されるときの JRun サーバーは、手動で開始させる必要があります。

JMC には [JRun サーバー] パネルがあり、[*machine_name*] > [*JRun_server_name*] を選択してアクセスします。

[JRun サーバー] パネルには、サーバーについての情報だけでなく、サーバーの状態も表示されます。このパネルを使用して、サーバーを再起動することもできます (96 ページの「JMC で JRun サーバーの再起動」で説明します)。

このセクションでは、JMC を使用した次の作業について説明します。

- 96 ページの「JMC で JRun サーバーの再起動」
- 96 ページの「jrun コマンドの使用」
- 100 ページの「JRun サーバーの追加および削除」
- 103 ページの「Java Virtual Machines の設定」
- 106 ページの「JRun サーバー イベント ログの設定」

JRun サーバーを使用した作業の詳細については、『JRun によるアプリケーションの開発』を参照してください。

JMC で JRun サーバーの再起動

JMC では、JRun サーバーを簡単な方法で再起動することにより、ユーザが JMC で行った大部分の変更内容を有効にすることができます。

UNIX や Windows では、コマンドラインから JRun サーバーを再起動することもできます。詳細については、96 ページの「jrun コマンドの使用」を参照してください。

また Windows では、コントロール パネルからアクセスできるサービス コントロール管理ユーティリティ (JRun サーバーをサービスとして実行している場合) から、またはシステム トレイ (JRun サーバーをアプリケーションとして実行している場合) から JRun サーバーを再起動することができます。

JMC で JRun サーバーを再起動するには

1. machine_name > JRun_server_name> Web Applications を選択します。

   [JRun サーバー] パネルが表示されます。

2. [サーバーの再起動] をクリックします。

   メモ この JMC から admin サーバーを再起動しないでください。デフォルトでは、そのサーバー上で JMC が実行されているからです。

jrun コマンドの使用

JRun には、Windows 環境と UNIX 環境の両方で使用できる、コマンドライン ユーティリティが用意されています。このセクションでは、このユーティリティのオプションについて説明します。

Windows では、<JRun_directory>/bin からコマンドライン ユーティリティを実行します。UNIX では、/opt/jrun/bin ディレクトリにコマンドライン ユーティリティがあります。jrun コマンド オプションをリストするには、コマンドラインに "jrun" (オプションなしの場合) と入力します。

このコマンドの構文は次のとおりです。

```
jrun -[info | version | admin]
jrun [-jrundir jrun_root_dir] -start | -stop | -restart [server_name]
jrun [-jrundir jrun_root_dir] -start [server_name] [-debug]
jrun -install NT_service_name server_name [-quiet]
jrun -remove NT_service_name [-quiet]
jrun -demo server_name
jrun -java java_prog -classpath classpath java_args class class_args
```

admin

jrun -admin

Windows (NT のみ) で JRun 管理コンソール を開始します。このコマンドにはパラメータを付けません。

console

jrun -console *options*

UNIX のみ java.System.out で指定する stdout/sterr 転送、および local.properties ファイルの java.System.err は、ログ ファイルへの出力ではなくコンソールへの出力なので、使用できません。また、screenlogger サービスを使用するときにこのオプションを使用します。

例

```
jrun -console -start default
```

demo

jrun -demo *server_name*

default サーバーを使用して、JRun デモ アプリケーションを開始します。**デモ** アプリケーションを別の JRun サーバーへ再公開する場合は、コマンドラインにそのサーバーを指定します。Windows NT のみ。

例

```
jrun -demo default
```

info

jrun -info

使用中の JRun インストールに関する情報を返します。

例

```
%> bash$ ./jrun -info
JRun 3.0 Service Pack 2
Version 3.02.2000
Developer Edition
```

Windows でこのオプションを使用する場合、コマンドラインから JRun を Java への入力として実行します。

例

```
java JRun -info
```

クラスパスに jrun.jar および servlet.jar が必要です。

install

jrun -install NT-service_name server_name -[quiet]

JRun を Windows NT サービスとしてインストールします (NT のみ)。server_name は、jvms.property ファイルにある JRun サーバーの 1 つでなければなりません。サーバーのルート ディレクトリの実際のパスを jvms.properties ファイルへ追加します。たとえば、次のようにします。

```
foo="C:/Program Files/Allaire/JRun/servers/foo"
```

例

```
% jrun -install "Foo Service" foo -quiet
```

-quiet オプションを使用すると、コマンドの成否にかかわらず、ダイアログ ボックスが表示されなくなります。JRun サーバーのインストールについては、100 ページの「 JRun サーバーの追加」を参照してください。

java

jrun -java java-prog -classpath path [java-args] class [class-args]

JRun 以外の Java アプリケーションを起動します。ディレクトリを指定するには、-classpath オプションを使用します。JRun により、.jar ファイルがすべて、このディレクトリに格納されています。

例

```
% jrun -java c:\jdk1.2.2\bin\java -classpath c:\JRun\lib JRun -start _
c:\JRun\servers\default
```

jrundir

jrun -jrundir *jrun_root_dir [command]*

jrundir オプションでは、JRun を起動するときにコマンドラインで JRun のルートディレクトリを指定できます。小売バージョンと OEM バージョンの両方を使用しているなどの場合で、1 つのマシンにインストールされた複数の JRun のインスタンスに違いをつけることもできます。Windows ユーザは 1 つのマシンに JRun の小売バージョンの 2 つのコピーをインストールできないことに注意してください。

JRun のインスタンスが 1 つだけインストールしてある場合、このオプションは必要ありません。

例

```
jrun -jrundir c:\oemcompany\servlet_engine start -admin
```

nohup

jrun -nohup [*JRun_server_name*] [-debug]

UNIX のみ。JRun サーバーをすべて開始します。特定の JRun サーバーのみを開始するには、JRun_server_name に該当するサーバー名を指定します。-nohup オプションは -start オプションと異なり、フォアグラウンドで子プロセスを作成する代わりにバックグラウンドでサーバー用に新規プロセスを作成します。-debugオプションは JRun を JRun Studio 用のデバッグ モードに設定します。-debug オプションを使用する場合、他の JRun サーバーを起動する前に admin JRun サーバーを起動する必要があります。

remove

jrun -remove NT-service-name [-quiet]

Windows NT のサービスである JRun サーバーを削除します (Windows NT のみ)。

例

```
% jrun -remove "JRun Default" -quiet
```

-quiet オプションを使用すると、コマンドの成否にかかわらず、ダイアログ ボックスが表示されなくなります。

restart

jrun -restart [JRun_server_name]

JRun サーバーをすべて再起動します。特定の JRun サーバーのみを再起動するには、JRun_server_name に該当するサーバー名を指定します。

start

jrun -start [*JRun_server_name*] [-debug]

JRun サーバーをすべて開始します。特定の JRun サーバーのみを開始するには、JRun_server_name に該当するサーバー名を指定します。-debug オプションは JRun を JRun Studio 用のデバッグ モードに設定します。-debug オプションを使用する場合、他の JRun サーバーを起動する前に admin JRun サーバーを起動する必要があります。

例

```
% jrun -start default -debug
```

-nohup オプション (UNIX のみ) は、オンラインで子プロセスを作成せずに、バックグラウンドでサーバーに新規プロセスを引き当てます。

status

jrun -status [JRun_server_name]

JRun サーバーすべてのステータスを表示します。特定の JRun サーバーのステータスのみを表示するには、JRun_server_name に該当するサーバー名を指定します。

stop

jrun -stop [JRun_server_name]

JRun サーバーをすべて停止します。特定の JRun サーバーのみを停止するには、JRun_server_name に該当するサーバー名を指定します。

version

```
jrun -version
```

使用中の JRun のバージョンを返します。

例 (UNIX)

```
%> bash$ ./jrun -version
3.02.2000
```

-version オプションを Windows で使用するには、コマンドラインから JRun を Java への入力として実行します。

例

```
java JRun -version
```

クラスパスに jrun.jar および servlet.jar が必要です。

JRun サーバーの追加および削除

JRun のデフォルトインストールには、admin と default の 2 つのサーバーが含まれます。これらの JRun サーバーには、サンプル アプリケーションが用意されており、サーバーをすばやく起動、実行するための方法が提供されています。これにより、品質保証、生産、開発環境を別々に維持したり、複数の Web サイトで公開するアプリケーションを開発するために JRun サーバーを追加することができます。

JRun サーバーを追加または削除する場合、それぞれのサーバーにより使用されるポートおよびそれらの機能に注意しておく必要があります。JRun のポートの使用法については、160 ページの「JRun ポートについて」を参照してください。

JRun サーバーの追加

新しい JRun サーバーを作成するには、デフォルト アプリケーションを含む default JRun サーバーのファイルとディレクトリ構造をコピーしてから、新しいサーバーに合わせてこのコピーを修正するのが最も簡単です。

JRun サーバーを追加するには

1. /servers/default ディレクトリを、/servers/foo のように新しいサーバー名を持つディレクトリにコピーします。
2. 新しいディレクトリ (/servers/foo) で、local.properties ファイルを開きます。local.properties ファイルには、名前やポート設定などの基本的なプロパティのほか、サーバーにインストールされているアプリケーションといった JRun サーバー固有の設定が格納されています。

   新規の local.properties ファイルを次のように変更します。

   i. サーバーに新しく付けた名前に合わせて、jrun.server.displayname プロパティを変更します。コードは、次の例のようになります。

```
## jvm properties
# was:jrun.server.displayname=Default Server
jrun.server.displayname=Foo Server
```

ii. [Web Application Settings] セクションにある default-app 以外のアプリケーションを削除します。default サーバーにアプリケーションが 1 つも追加されていない場合は、demo-app 以外は何も削除する必要はありません。

たとえば、各アプリケーションに次のようなセクションがあるとします。

```
jmc-app.rootdir=C:\\ProgramFiles\\Allaire\\JRun\\servers _
    \\admin\\jmc-app
jmc-app.class={webapp.service-class}
webapp.mapping./=jmc-app
```

そのアプリケーションで default-app を参照していない設定を削除します。

iii. 既定のアプリケーションだけを含むように、servlet.webapps プロパティをリセットします。上記の例にあるコード行は、次のようになります。

```
# was:servlet.webapps=default-app, demo-app
servlet.webapps=
```

iv. ポート設定を変更します。これらの設定はそれぞれ、ファイルの異なるセクションに保存されています。たとえば、次のように設定します。

```
web.endpoint.main.port=8101 #was:8100 (Web server port)
jcp.endpoint.main.port=51001 #was:51000 ( リスニング ポート )
control.endpoint.main.port=53001 #was:53000 ( コントロール ポート )
ejipt.classServer.port=2324 #was: 2323
ejipt.homePort=2334 #was 2333
```

メモ ポート設定を変更しない場合は、バインディング例外が起こることがあります。

3. local.properties ファイルを保存します。
4. /servers/foo から default-app 以外のすべてのアプリケーションにあるディレクトリを 削除します。default サーバーにアプリケーションが 1 つも追加されていない場合は、何も削除する必要はありません。
5. <*JRun_rootdir*>/lib ディレクトリで jvms.properties ファイルを開きます。JRun では、jvms.properties ファイルを使用して、インスタンス化するサーバーが決定されます。新規サーバーに対する行を追加します。Windows NT では、ファイルの内容が次のように表示されます。

```
admin=C:/Program Files/Allaire/JRun/servers/admin
default=C:/Program Files/Allaire/JRun/servers/default
foo=C:/Program Files/Allaire/JRun/servers/foo
```

行末にあるディレクトリ名や、行頭にあるサーバー名に注意してください。

6. 次の手順に従って、新規 JRun サーバーを開始します。

Windows の場合 :

1. <JRun_rootdir>/bin で、jrun-default.bat ファイルを jrun-foo.bat にコピーします。

ii. jrun-foo.bat を編集して、新規サーバーの名前を指定します。

```
@echo off
start jrun -start foo
```

iii. jrun-foo.bat ファイルを実行します。

iv. Windows NT では、必要に応じてコマンドライン ユーティリティを使用し、新しいサーバーを NT サービスとして追加することもできます。

```
jrun -install "Foo Service" foo -quiet
```

v. 以下のコマンドを使用して、サーバーを開始します。

```
jrun -start foo
```

UNIX の場合 :

/jrun/bin にある JRun コマンドライン ユーティリティを使用します。

```
jrun -start foo
```

6. JMC にログインしている場合は、ログアウトしてから Admin サーバーを再起動します。

次回にログインすると、JMC の左側ペインに新しい Foo Server が表示されます。このサーバーには default-app 以外の Web アプリケーションは何も入っていません。

JRun サーバーの削除

JRun サーバーの削除は十分に考慮してから行ってください。サーバーを削除する前に、必ず、該当するサーバーに入っている重要なファイルやアプリケーションをすべてバックアップしてください。このセクションでは、JRun サーバーの削除方法について説明します。

警告 JRun Admin Server、および Default Server を削除しないでください。これらのサーバーに入っているアプリケーションが失われないようにするためです。

JRun サーバーを削除するには

1. 削除する JRun サーバーを停止します。
2. サーバーのディレクトリ、およびすべてのサブディレクトリを削除します。たとえば、/servers にある /foo ディレクトリを削除します。
3. <JRun_rootdirectory/lib ディレクトリで jvms.properties ファイルを開き、このサーバーに対応する行を削除します。Windows NT では、ファイルの内容が次のように表示されます。

```
admin=C:/Program Files/Allaire/JRun/servers/admin
default=C:/Program Files/Allaire/JRun/servers/default
#foo=C:/Program Files/Allaire/JRun/servers/foo
```

4. このサーバーで使用されていたスタートアップ スクリプトをすべて削除します。

5. サーバーを NT のサービスとして追加していた場合、コマンドライン ユーティリティを使用して、このサービスを削除します。

   ```
   jrun -remove "Foo Server" -quiet
   ```

6. JMC にログインしている場合は、ログアウトしてから Admin サーバーを再起動します。

   次回ログインすると、JMC の左側ペインにこのサーバーは表示されません。

Java Virtual Machines の設定

Java Virtual Machine (JVM) は JRE ともよばれ、ソフトウェアで実装された CPU です。これには Java プラットフォームのために作成されたプログラムを実行するために必要なすべての機能が含まれています。

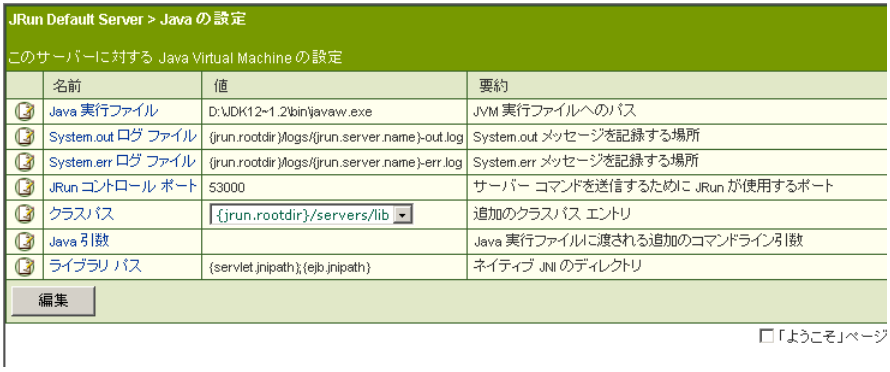
JRun サーバーはそれぞれ、JRun サーバーに対して全サーブレット、JSP ページ、および EJB を実行する 1 つの JVM と関連付けられています。このセクションの情報を使用し、各 JRun サーバーについて JVM を設定します。

[Java 設定] パネルを使用して、ログ ファイル出力の位置を設定できます。UNIX システムでは、[jrun] コマンドの -console オプションを使って、Java.System.err および java.System.out 出力をコンソールに転送することもできます。詳細については、96 ページの「 jrun コマンドの使用」を参照してください。

JVM の一般設定を編集するには

1. JRun 管理コンソールの左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Java の設定] を選択します。

   [Java の設定] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[Java 設定の編集] ウィンドウが表示されます。

3. 次の表の説明に従ってプロパティを編集します。

JVM プロパティ	
プロパティ	**説明**
Java 実行ファイル	ユーザの Java 仮想マシンへのパスを入力します。JVM を変更する場合は、[Java 引数] フィールドで、コマンドライン引数の変更が必要になることもあります。
System.out ログ ファイル	JVM の system.out メッセージのログ先の絶対パスを入力します。
System.err ログ ファイル	JVM の system.err メッセージのログ先の絶対パスを入力します。
JRun コントロール ポート	一意のポート番号を入力します。このポートの既定値は JRun インストール スクリプトにより決まります。JRun では、ステータス情報とシャットダウン情報のために、このポートが使用されます。
Java クラスパス (java.exe のみ)	クラスパスは、クラスを見つけるために Java プロセスにより検索されるディレクトリのリストです。このパスにクラスを追加するか、JRun によりデフォルトでクラスパスに追加されている <JRun_directory>/classes ディレクトリにクラスを保存します。 テキスト フィールドの Java クラスパスに追加するパスを入力します。この Java クラスパスは、現在の JRun サーバー内にあるサーブレットによって使用されます。 クラスパスの詳細については、『JRun によるアプリケーションの開発 』を参照してください。
クラスパス	JRun 自体が実行する必要のあるクラスおよび .jar ファイルの位置を入力します。ディレクトリを入力すると、そのディレクトリ内のすべての .jar ファイルがそのクラスパスに含まれます。
Java 引数	JRun により JVM が開始される場合に JRun から JVM 実行ファイルに渡されるコマンドライン引数をすべて入力します。
ライブラリ パス	ユーザのサーブレットで別のプログラミング言語 (C、C++ など) によるステートメントを使用する場合は、Java ネイティブ インタフェース (JNI) が入っているディレクトリを入力します。複数のディレクトリを指定する場合は、セミコロン (Windows の場合) かコロン (UNIX の場合) で区切ります。

4. 変更を適用するには、[更新] をクリックします。
5. JRun サーバーを再起動します。

Java コマンドライン オプションの使用

デフォルトでは、Windows 用 JRun と Solaris 用 JRun では JRE 1.2 を使用します。エディタで Java 実行ファイル フィールドに別の JVM を指定する前に、使用可能なコマンドライン オプションについて理解しておく必要があります。[Java 引数] フィールドにコマンドライン オプションを追加することもできます。一般的によく使用される 2 種類の JVM のコマンドライン オプションは次のとおりです。

Sun Microsystems JDK1.1.7b オプション

Sun Microsystems JDK1.1.7b
使用法 :java [-options] class
このオプションに入る値は次のとおりです。
-help このメッセージを出力します
-version ビルド バージョンを出力します
-v -verbose verbose モードをオンにします
-debug リモート JAVA デバッグを有効にします
-noasyncgc 非同期ガベージ コレクションができないようにします
-verbosegc ガベージ コレクションが始まったとき、メッセージを表示します
-noclassgc クラス ガベージ コレクションを無効にします
-ss<number> スレッドの最大ネイティブ スタック サイズを設定します
-oss<number> スレッドの最大 Java スタック サイズを設定します
-ms<number> Java ヒープ サイズの初期値を設定します
-mx<number> 最大 Java ヒープ サイズを設定します
-classpath < セミコロン (;) で区切られたディレクトリ >
クラスの検索先ディレクトリをリストします
-prof[:< ファイル >] .\java.prof、または .\<file> にプロファイル データを出力します
-verify 読み込むとき、クラスをすべて検証します
-verifyremote ネットワーク経由で読みこまれたクラスを検証します [既定値]
-noverify クラスの検証は行いません
-nojit JIT コンパイラを無効にします

Sun Microsystems JDK1.2 (Java 2 プラットフォーム)

使用法 :java [-options] class [args...]
(クラスを実行する場合)
または java -jar [-options] jarfile [args...]
(jar ファイルを実行する場合)
このオプションに入る値は次のとおりです。
-cp -classpath < セミコロン (;) で区切られたディレクトリおよび zip/jar ファイル >
アプリケーション クラスやリソースを検索するパスを設定します
-D<name>=<value>
システム プロパティを設定します
-verbose[:class|gc|jni]
verbose 出力を有効にします
-version 製品バージョンを表示します
-? -help このヘルプ メッセージを表示します

-X 非標準オプションに関するヘルプを表示します

Microsoft Command-Line Loader オプション

```
Microsoft (R) Command-line Loader for Java  Version 5.00.3155
Copyright (C) Microsoft Corp 1996-1999. All rights reserved.
使用法 :JView [options] <classname> [arguments]
オプション :
    /?              コマンド形式を表示します
    /cp <classpath>    クラス パスを設定します
    /cp:p <path>       クラス パスの前にパスを追加します
    /cp:a <path>       クラス パスの後ろにパスを追加します
    /n <namespace>     実行場所となるネームスペースを指定します
    /p              エラーが発生した場合、処理を終了する前に一時停止します
    /v              クラスすべてを検証します
    /d:<name>=<value>  システム プロパティを定義します
    /a              AppletViewer を実行します
    /vst            verbose スタック トレースを表示します ( デバッグ クラスが必要です )
クラス名 :
    実行される .CLASS ファイル
引数 :
    クラス ファイルに渡されるコマンドライン引数
```

JRun サーバー イベント ログの設定

JRun のログ記録メカニズムを使用して、それぞれの JRun サーバーについて、ログ ファイルの内容を制御することができます。ログを記録しておくと、Web サーバーのトラブルシューティングや負荷分散に便利です。このセクションでは、JMC を使用して JRun サーバーに関するイベント ログを設定する方法について説明します。

UNIX システムでは、[jrun] コマンドの -console オプションを使って、Java.System.err および java.System.out 出力をコンソールに転送することもできます。詳細については、96 ページの「 jrun コマンドの使用」を参照してください。

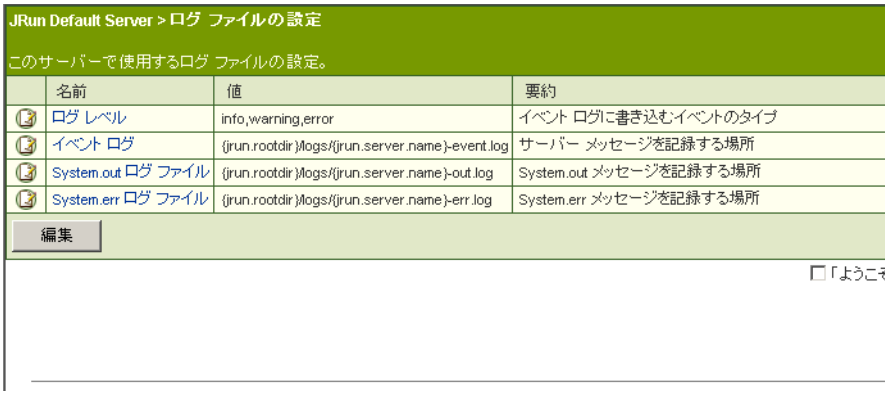
JVM のイベント ログの設定については、103 ページの「 Java Virtual Machines の設定」を参照してください。JRun アプリケーションのイベント ログの設定については、136 ページの「 JRun アプリケーション イベント ログの構成」を参照してください。

JRun ログ記録メカニズムの使用に関するその他の情報については、『JRun によるアプリケーションの開発』を参照してください。

JRun サーバーのログ設定を編集するには

1. JMC の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [**ログ ファイルの設定**] を選択します。

   [ログ ファイルの設定] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[ログ ファイルの設定] 編集ウィンドウが表示されます。
3. 次の表の説明に従って、右側ペインにプロパティを入力します。

JRun サーバー イベント ログ プロパティ

プロパティ	説明
ログレベル	ログ ファイルに追加するログレベルを選択します。既定では、Info、Error、および Warning が設定されています。他にも debug と metrics などのオプションがあります。
イベント ログ	ログ ファイルのパスと名前を設定します。既定値は {jrun.rootdir}/logs/{jrun.server.name}-event.log です。
System.out ログ ファイル	JRun サーバーの system.out メッセージのログ先の絶対パス名を入力します。
System.err ログ ファイル	JRun サーバーの system.err メッセージのログ先の絶対パス名を入力します。

4. [更新] をクリックして、変更を適用します。
5. JRun サーバーを再起動します。

JDBC データ ソースの設定

JMC を使用して、JDBC 準拠のデータ ソースの追加、削除、編集、およびテストを行うことができます。データ ソースの設定にこのインターフェイスを使用することで、サーブレットで同じ設定をハードコードする必要がなくなります。また、複雑なドライバおよび URL を埋める JDBC ウィザードも用意されています。

このセクションでは、次の項目について説明します。

- 108 ページの「JDBC データ ソースの設定」
- 110 ページの「JDBC データ ソースの編集」
- 111 ページの「よくある問題とその解決方法」

JMC にセットアップされているデータ ソースにアクセスする例については、『JRun によるアプリケーションの開発』を参照してください。

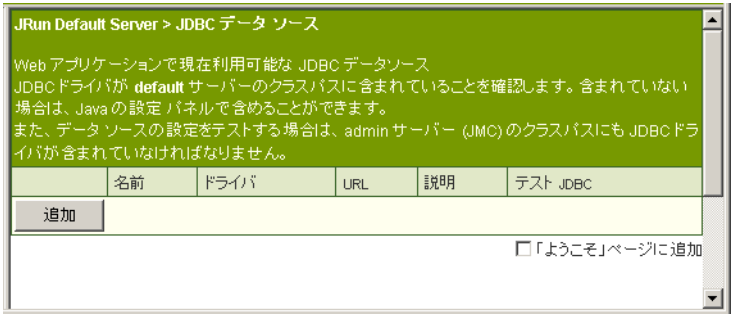
JDBC データ ソースの設定

JDBC ウィザードを使用して、Web アプリケーション用にデータ ソースを設定できます。

JDBC データ ソースを設定するには

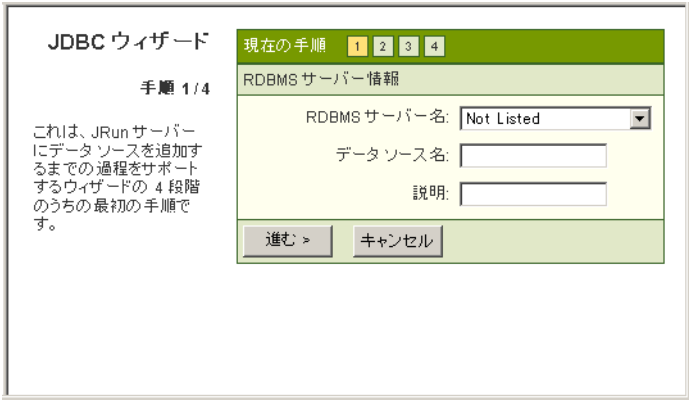
1. JMC の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [JDBC データ ソース] を選択します。

   [JDBC データ ソース] パネルが表示されます。



2. 右側ペインで、[**追加**] をクリックします。

   JDBC ウィザードの手順 1 が表示されます。



3. ドロップダウン リストから RDBMS Server Name または Not Listed を選択します。ウィザードの残りのフィールドは、選択したサーバーによって異なります。データベース名およびオプションでその説明を入力してください。JRun の Advanced 版または Enterprise 版のライセンスがあり、DB2 または Sybase を選択した場合、次のパネルで JRun JDBC ドライバか、またはベンダにより提供されたドライバのどちらを使用するか選択できます。
4. [**次へ**] をクリックします。
5. 次のパネルでは、JDBC ウィザードに必要な構成情報を指定し、[**次へ**] をクリックします。JDBC ウィザードの各ページでこの手順を繰り返します。手順の内容は、インストールしたドライバによって異なります。
6. 終われば [**完了**] をクリックします。
7. ドライバが JRun JDBC ドライバ以外の場合、そのデータベース ドライバの *.jar ファイルを選択された JRun サーバーのクラスパスと admin JRun サーバーのクラスパスの両方に追加します。*.jar ファイルを admin JRun サーバーのクラスパスに追加しなかった場合、JDBC データ ソース パネルの [**テスト**] ボタンは機能しませんが、ドライバは通常どおり機能するはずです。JRun サーバーのクラスパスの編集についての詳細は、103 ページの「 Java Virtual Machines の設定」を参照してください。
8. JRun サーバーを再起動します。
9. [**テスト**] ボタンをクリックし、データ ソースの接続テストを行います。エラーが発生した場合、111 ページの「よくある問題とその解決方法」を参照してください。

JDBC データ ソースの編集

このセクションの手順に従って、既存の JDBC データ ソースを編集できます。JDBC ウィザードを使用して設定した内容を変更できます。

JDBC データ ソースを編集するには

1. JMCの左側ペインで、[*machine_name*] > [*JRun_server_name*] > [JDBC データ ソース] を選択します。

   JDBC データ ソースパネルが表示されます。

2. 右側ペインで JDBC データ ソースの名前または名前の横にある編集アイコン  をクリックします。[JDBC データ ソース編集] ウィンドウが表示されます。

3. 次のテーブルで説明されているプロパティを編集します。

JDBC データ ソースの設定	
フィールド	**説明**
名前	JDBC データ ソース名を入力します。この名前は、データベースへの接続を確立するときに、サーブレットのコードで使用されます。このフィールドは省略できません。
表示名	JDBC データ　ソースの表示名を入力します。
ドライバ	JDBC　ドライバのクラス名を入力します。たとえば、sun.jdbc.odbc.JdbcOdbcDriver　のようにします。このフィールドは省略できません。
URL	データ ソースをポイントする URL を入力します。たとえば、"jdbc:odbc:fred" のように指定します。ここで、fred にはセットアップしたデータ ソース名が入ります。このフィールドは省略できません。
説明	この JDBC データ ソースの説明を入力します。この説明は、JRun JMC でのみ有効です。
プール	[プール] チェックボックスをオンにして、JDBC データ ソースの接続プールを使用できるようにします。パフォーマンスを上げるために、このオプションはぜひ使用してください。
タイムアウト (分)	JRun が、アクティブでない状態が継続しているデータ ソース接続を閉じるまでの継続時間を分単位で入力します。既定の設定は 30 分です。

JDBC データ ソースの設定	
フィールド	**説明**
間隔 (秒)	有効期間が満了したデータ ソース接続を閉じるまでの JRun による待機時間を秒単位で入力します。既定の設定は 30 秒です。
ユーザ名	データベースで認証が必要な場合に、ユーザ名を入力します。
パスワード	データベースで認証が必要な場合に、ユーザ名 に対応するパスワードを入力します。
ベンダ引数	ベンダー固有の引数に対する名前と値の組を入力します。たとえば、データベース ベンダーの中には、接続プール パラメータを設定する引数を渡せるものもあります。 書式は、name=value です。 詳細は、データベース ベンダーのマニュアルを参照してください。

4. JDBC データ ソースを削除するには、[**削除**] チェックボックスをクリックします。
5. 変更を適用するには、[**更新**] をクリックします。
6. JRun サーバーを再起動します。

よくある問題とその解決方法

設定したデータ ソースをテストしたときにエラーが発生した場合、JRun /logs ディレクトリの <*JRun_server_name*>-out.log ファイルを表示します。このセクションでは、log ファイルで見つかる可能性のあるエラーについて説明します。

No suitable driver

```
"SQLException: No suitable driver"
```

ドライバが見つからないか、またはドライバのクラスパスに誤りがある場合です。クラスパスが正しい *.jar ファイルを指しており、*.jar ファイルが存在することを確認します。

Access denied

```
"SQLException: Invalid authorization specification: Access denied for
user: 'nobody@' (Using password: YES)"
```

データベースに渡すユーザ名およびパスワード、またはデータベースの優先設定のどちらかに関する問題です。データベースのマニュアルを確認します。また、データ ソース

をサーブレットからテストしている場合、サーブレット コードを確認して、正しい名前とパスワードがデータベースに渡されているかどうかを確認します。

Naming Exception: <*datasource*> not found

作成したデータソースが JRun サーバーにより認識されない場合です。JMC に表示されることもありますが、サーバーを再起動してからでないとサーブレット コンテキストは認識しません。

JRun サーバーを再起動します。

```
c:\> jrun -restart default
```

Web サーバーの設定

JRun は、それぞれの JRun ごとに JRun Web サーバー (JWS) のインスタンスを作成し、それらを JRun 接続 モジュール (JCM) にリンクします。これは、インストール時には、Admin サーバーの JWS インスタンスおよび default サーバーの JWS インスタンスが実行されていることを表します。

新規の JRun サーバーを追加すると、JWS インスタンスがもう 1 つ作成されるので、リクエストへの対応をこの新規サーバー上ですぐに開始することができます。ただし、ほとんどの開発環境では、ニーズを満たすために外部 Web サーバーが必要になるため、JRun でそれらの外部 Web サーバーへの接続を行ないます。この接続は、JRun コネクタ ウィザードを使って作成されます。

メモ admin JRun サーバーの接続の設定を変更することにより、JRunサーバーでJRun管理コンソールへのアクセス方法を変更することができます。変更内容については、必要に応じて後で元に戻せるように、すべて記録しておきます。

次のセクションでは、JRun サーバーと JWS または外部 Web サーバーの間の接続を設定する方法を説明します。

並行処理の概要

JRun と JWS または外部 Web サーバーの間の接続を設定することにより、並行処理の設定が最適化されます。並行処理とは、HTTP 要求がプールされて分配される方式です。

忘れてならないのは、「同時要求」と「同時ユーザ」は、それぞれ異なる概念であるということです。今仮に 2000 個の同時要求をサポートする必要があると考えているとすれば、それは、実際に一度に作成される要求の数は 100 だけで、同時ユーザは 2000 人いるということである可能性があります。JRun は、各要求ごとにスレッドを 1 つ割り当てます。

一般的には、非常にボリュームの大きなインターネット サイトを実行していないかぎり、既定の並行処理の設定を変更しないでください。JRun には、Web サーバーへの接続に関する並行処理の設定値が 4 つあります。それは以下の設定値です。

- 待機スレッドのタイムアウト
- 最少スレッド カウント
- 最大アクティブ要求
- 最大同時要求

過去に Web サイトでトラフィックの負荷が頻繁に急増している環境では、最小スレッド数を多めに設定することにより、Web サイトでトラフィックの負荷が突然増えた場合でも、一連のスレッドを作成する必要はありません。あるいは、最小スレッド数を、同時要求について予想される安定した負荷状態に設定することもできます。たとえば、常時、200 個の同時要求がある場合は、最小スレッド数を 200 に設定する必要があります。

たとえば、Ｗｅｂ サイトの平均応答時間が RMI-CORBA- データベースという 3 段階のトランザクションが原因で遅れるのであれば、新しいを要求を拒否することなくスループットを維持するために、より多くの要求を受け入れるようにキューを大きくする必要があります。この場合は、最大同時要求数を、予想される要求数より大きな値に増やします。最大同時要求数の設定値は、リソースの安全弁の役割を果たします。

メモ JRun の既定の並行処理設定値を変更するにあたっては、トラフィックのパターンを事前に測定できるようにしてください。根拠もなく設定値を変更すると、リソースの無駄を起こしかねません。

並行処理の設定値は、[Web サーバーの編集] ウィンドウで編集します。JWS および外部 Web サーバーの並行処理設定値の編集については、115 ページの「 JWS の設定」「117 ページの「外部 Web サーバーの設定」を参照してください。

JRun コネクタ ウィザードの使用

JRun コネクタ ウィザードを使って、Web サーバーと JRun サーバーの間にリンクを設定することができます。この接続は、JRun 接続モジュールの形式となります (JCM)。JRun の標準インストールには、Default JRun サーバーを JWS へ接続する JCM が 1 つ含まれています。JRun サーバーに接続できる Web サーバーの数に制限はありません。

メモ ほとんどの場合、外部Ｗｅｂ サーバーの接続先には Default サーバーを選択します。JRun Admin Server はそれぞれ固有の Web サーバーを持ち、JRun インストールの管理にのみ使用されます。Default サーバーには、サーブレット、JSP、および Web アプリケーションを公開できます。JRun のデモ アプリケーションは Default サーバーで実行されます。

このセクションでは、JRun コネクタ ウィザードの一般的な使用手順について説明します。JRun サーバーを 特定の Web サーバーに接続する方法については、第 2 章 の次のセクションを参照してください。

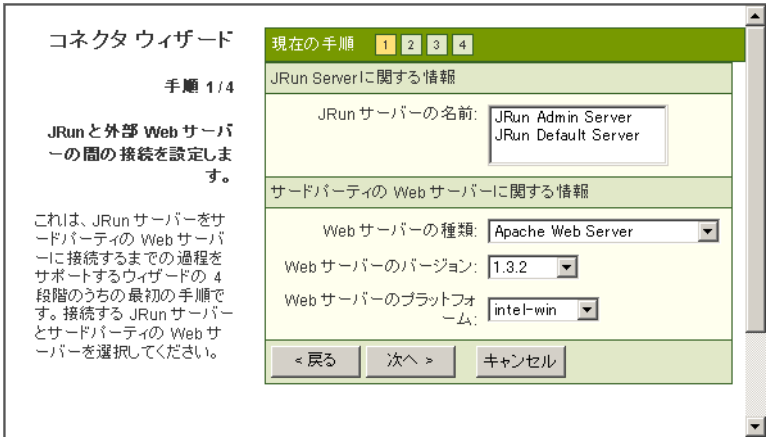
- 43 ページの「 Apache の構成」
- 48 ページの「 IIS 3.0/PWS の構成」
- 52 ページの「 IIS 4.0/5.0 の構成」

- 60 ページの「Netscape/iPlanet の構成」
- 68 ページの「WebSite Pro の構成」
- 77 ページの「Zeus Web サーバーの構成」

JRun コネクタ ウィザードを使用するには

1. JRun に接続する Web サーバーを停止します。
2. アクセス バーで **[コネクタ ウィザード]** リンクをクリックします。

   右側ペインに JRun コネクタ ウィザードが表示されます。



3. 各ステップで適切な情報を入力してから、[次へ] をクリックします。間違った場合は、[戻る] をクリックします。

   終了すると、JRun は、コネクタ ウイザードとの接続が正常に設定されたことを通知します。
4. Web サーバーを開始します。
5. JRun サーバーを再起動します。
6. Web サーバーで SnoopServlet をリクエストして、接続をテストします。

   http://localhost:80/demo/index.html

デモ アプリケーションが実行される場合、JRun と 外部 Web サーバーの接続は正常に設定されています。デモ アプリケーションが正常に実行されない場合は、81 ページの「トラブルシューティング」を参照してください。

JWS の設定

インストール処理の間、JRun により、2 種類の JRun Web サーバー (JWS) がセットアップされます。Admin JRun サーバーへあらかじめ接続された Web サーバーを使用して JMC 自体にアクセスする必要があるので、少なくとも最初のうちは、JWS のインスタンスが 1 つ必要です。また、JRun により JWS の 2 番目のインスタンスがインストールされ、default JRun サーバーに接続されます。

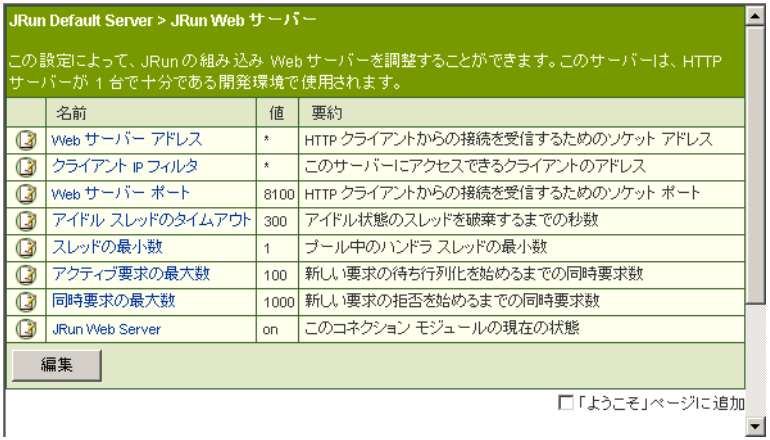
JWS はメモリ使用量の少ない Java Web サーバーです。現在 JWS は、SSL などの、開発環境で使用するのに適していないある種の機能をサポートしていません。

ここでは、JWS の JRun 接続モジュールの設定方法を説明します。使用している JRun サーバーが、外部 Web サーバーと接続されている場合は、117 ページの「外部 Web サーバーの設定」を参照して、JCM に対するエンドポイント プロパティを設定してください。

JWS のエンドポイント プロパティを編集するには

1. JRun 管理コンソール の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [JRun Web サーバー] を選択します。

   [Web サーバー] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[JRun Web サーバー] 編集ウィンドウが表示されます。

3. 次の表の説明に従ってプロパティを編集します。

JWS エンドポイント プロパティ	
プロパティ	**説明**
Web サーバー アドレス	JWS 上で、HTTP クライアントからの接続をリスニングしているソケットの IP アドレスを入力します。 サーバーが複数の IP アドレスを持っている場合は (マルチホーム)、JRun が JWS 上でバインドする相手側 IP アドレスのリストを入力します。 既定値は * で、この JWS はサーバー IP アドレスすべてにバインドされます。
クライアント IP フィルタ	JWS が応答する IP アドレスのリストを入力します。ここで指定されていない IP アドレスからのリクエストはすべて無視されます。 既定値は * で、JWS はすべての IP アドレスからのリクエストに応答します。
Web サーバー ポート	TCP ポート番号を入力します。この JWS により、このポートに届く HTTP リクエストが監視されます。 JRun Admin Server の JWS の既定値は 8000 です。JRun default Server の既定値は 8100 です。
待機スレッドのタイムアウト	JRun によりスレッドが破壊される前に、待機する時間を秒単位で指定します。このパラメータは、JWS が活動期間の後、何秒後に静止状態に戻るかを表します。スレッドが破壊されるたびに、少量のシステム リソースが解放されます。 JRun では、同時発生する要求を処理するために、Java スレッド メカニズムが使用されます。要求に対し個別に新しいスレッドを作成する代わりに、JRun では、新しい要求に対する準備の整ったハンドラ スレッドのプールが維持されます。このスレッド プールは Web サーバーにあり、要求の変化に合わせてサイズが変わります。最適な状況では、トラフィック負荷と Web サーバーの能力の間でプール パラメータのバランスが取れています。 既定値は 300 秒です。

JWS エンドポイント プロパティ	
プロパティ	**説明**
スレッドの最少数	スタートアップ時に初期プールで作成されるスレッドの数を指定します。システムの稼動に従って、スレッドが作成されたり、破壊されますが、ここで指定された最小値よりもプール サイズが小さくなることはありません。 既定値は 1 です。
アクティブ要求の最大数	この JWS が同時に処理できるアクティブ要求の最大数を入力します。この限度を超えた要求は (最大同時要求数まで) は、スレッドがそれらの要求を処理できるようになるまで遅延します。 JRun では、JWS 上の同時発生数を制限するために、主にこのパラメータが使用されます。既定値は 100 です。
同時要求の最大数	JWS が処理するか、あるいは処理のためにキューに入れる同時要求の最大個数を入力します。この最大数を超える要求はすべて捨てられます。 既定値は 1000 です。
JRun Web サーバー	使用する JWS のチェックボックスをオンにします。使用しない JWS のチェックボックスをオフにします。 JRun サーバーを 1 つ追加すると、JWS インスタンスが 1 つ作成され、この JRun サーバーに接続されるということに注意してください。リソースへの影響が最小である間は、JRun コネクタ ウイザードを使用して JRun サーバーが外部 Web サーバーへ正常に接続されている場合、当該 JWS をオフにすることができます。 Admin JRun サーバーの JMC アプリケーションについては、JWS をオフにしないでください。

4. 変更を適用するには、[**更新**] をクリックします。
5. JRun サーバーを再起動します。

外部 Web サーバーの設定

JRun には JWS が含まれていますが、開発環境においては、JRun サーバーを外部 Web サーバーと接続する必要がある場合もあります。外部 Web サーバーの接続については、113 ページの「 JRun コネクタ ウィザードの使用」に説明がありますが、結論を言えば、

JRun 接続モジュール (JCM) が、JRun サーバーと外部 Web サーバーの間の接続を管理します。

JRun サーバー(default など) は 1 つのモジュールに接続されます。接続できる Web サーバーの数に制限はありません。接続が確立されたら、このセクションの説明に従って JCM を調整します。

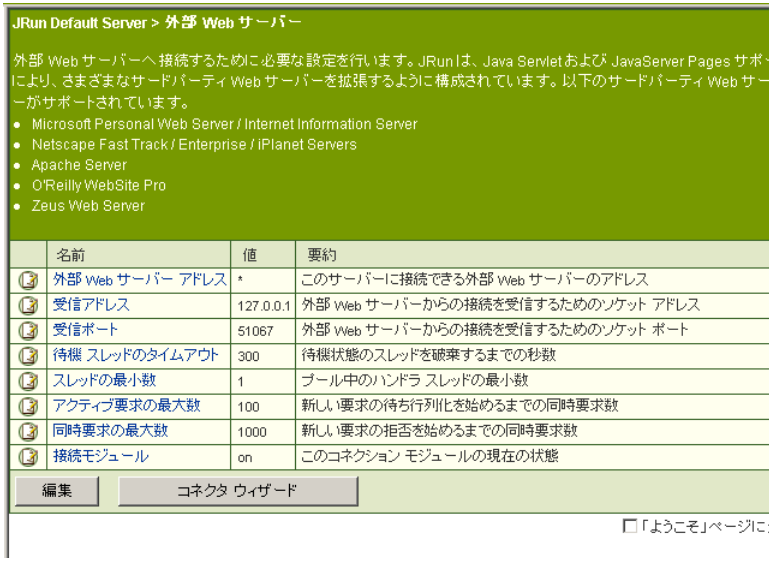
メモ JRun に組み込まれている JWS では、外部 Web サーバーの一部の設定を変更することはできますが、JMC からはできません。Web サーバーのマニュアルを参照してください。

外部 Web サーバーの JCM を設定するには

1. JRun 管理コンソール の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [**外部 Web サーバー**] を選択します。

   メモ コネクタ ウィザードを起動してこの JRun サーバーを外部 Web サーバーに接続していないと、すぐにそれを行うように促されます。

   [外部 Web サーバー] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。外部 Web サーバーの編集ウィンドウが表示されます。

3. 次の表の説明に従ってプロパティを編集します。

JRun 接続モジュールのプロパティ	
プロパティ	**説明**
外部 Web サーバー アドレス	IP アドレスのコンマ区切りの一覧を入力します。この JCM は、そのアドレスから JRun サーバーへと要求を渡すのみです。 既定値は * で、これによりこの JCM はすべての IP アドレスからの要求を受け入れます。
受信アドレス	外部 Web サーバーからの接続を受信するソケットの IP アドレスを入力します。使用しているサーバーに複数の IP アドレスがある (マルチホーミング) 場合には、この機能が便利です。 既定値は * で、これにより JRun はすべてのサーバー IP アドレスにバインドします。
受信ポート	外部 Web サーバーからの接続を受信するためにこの JCM が使用する一意のポート番号を入力します。 このポートと、外部 Web サーバーの HTTP ポートを混同しないでください。
待機スレッドのタイムアウト	JRun によりスレッドが破壊される前に、待機する時間を秒単位で指定します。このパラメータは、Web サーバーが活動期間の後、何秒後に静止状態に戻るかを表します。スレッドが破壊されるたびに、少量のシステム リソースが解放されます。 JRun では、同時発生する要求を処理するために、Java スレッド メカニズムが使用されます。要求に対し個別に新しいスレッドを作成する代わりに、JRun では、新しい要求に対する準備の整ったハンドラ スレッドのプールが維持されます。このスレッド プールは Web サーバーにあり、要求の変化に合わせてサイズが変わります。最適な状況では、トラフィック負荷と Web サーバーの能力の間でプール パラメータのバランスが取れています。 既定値は 300 秒です。
スレッドの最少数	起動時に初期プールで作成されるハンドラ スレッドの数を指定します。システムの稼動に従って、スレッドが作成されたり、破壊されますが、ここで指定された最小値よりもプール サイズが小さくなることはありません。 既定値は 1 です。

JRun 接続モジュールのプロパティ	
プロパティ	**説明**
アクティブ要求の最大数	JRun で同時に処理できる要求の最大数を指定します。ハンドラ スレッドで対応できるようになるまで、この制限を超える要求はすべて遅延します JRun では、外部 Web サーバーでの同時発生数を制限するために、主にこのパラメータが使用されます。 既定値は 100 です。
同時要求の最大数	Web サーバーで処理できる最大要求数を入力します。この最大数を超える要求はすべて捨てられます。 既定値は 1000 です。
接続モジュール	このチェックボックスを選択して JCM をオンにします。チェックをはずして JCM をオフにします。接続モジュールをオフにすると、JRun と外部 Web サーバーとの接続が切断されます。これにより、ユーザが JSP ページまたはサーブレットにアクセスしようとすると、エラーとなります。

4. 変更を適用するには、[**更新**] をクリックします。
5. JRun サーバーを再起動します。

Web アプリケーションの構成

Web アプリケーションには、サーブレット、JSP、スタティック ファイル、およびその他のリソースから構成されているものがあります。これらのリソースは、サーブレットに対応した任意の Web サーバーに公開できるように、あらかじめ定義されたディレクトリ構造に従って保存されます。

JRun には、JRun の実装にアプリケーションを追加するための手段が 2 種類用意されています。既存の Web アプリケーションの Web アプリケーション アーカイブ (.war) ファイルを公開するか、新規アプリケーションを作成することができます。

JRun サーバーに追加できる Web アプリケーションの数に制限はありません。標準インストールには、次のアプリケーションが含まれます。

- / にマッピングされた Admin JRun サーバーの JRun 管理コンソール アプリケーション
- /demo にマッピングされた default JRun サーバーの JRun Demo アプリケーション
- / にマッピングされた default JRun サーバーの Default User Application

JMC には [アプリケーション] パネルがあり、[machine_name] > [JRun_server_name] > [Web アプリケーション] を選択してアクセスします。

JMC では、[アプリケーション] パネルにある各アプリケーションについて、編集  および削除  機能へのクイック リンクが可能です。

このセクションでは、次の項目について説明します。

- 122 ページの「アプリケーションの作成」
- 123 ページの「アプリケーションの公開」
- 126 ページの「アプリケーションの編集」
- 128 ページの「アプリケーションの削除」
- 129 ページの「アプリケーション パスのマッピング」
- 130 ページの「アプリケーション ホストの作成」
- 132 ページの「アプリケーション パラメータの追加」
- 133 ページの「ファイル設定の変更」
- 135 ページの「 JSP コンパイラの構成」
- 136 ページの「 JRun アプリケーション イベント ログの構成」
- 137 ページの「 MIME タイプのマッピング」
- 138 ページの「セッション トラッキングの構成」

アプリケーションの構築および公開の詳細については、『JRun によるアプリケーションの開発』を参照してください。

アプリケーションの作成

空のアプリケーションを作成し、JMC を使用して JRun サーバーにそれを登録することができます。このプロセスにより、そのアプリケーションについて、ルート ディレクトリ、WEB-INF、WEB-INF/classes、および WEB-INF/lib ディレクトリからなる空のディレクトリ構造が作成されます。

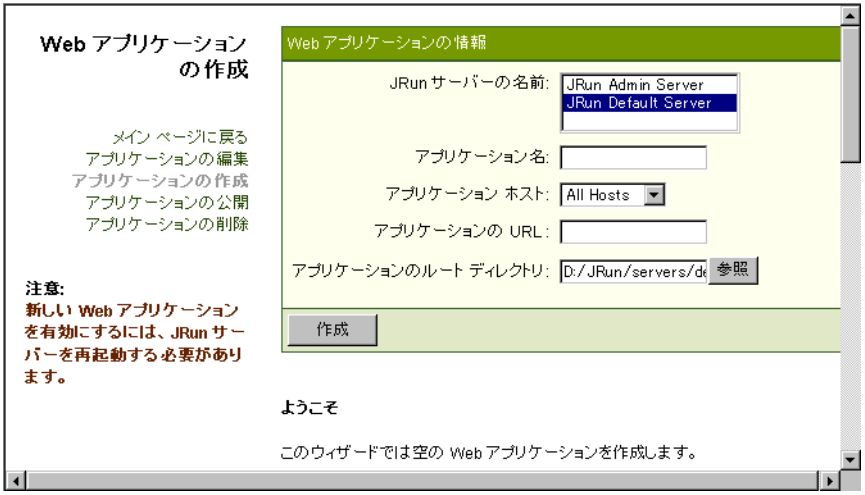
空のアプリケーションを追加するには

1. [*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] を選択します。

   [アプリケーション] パネルが表示されます。

2. [アプリケーションの作成] リンクをクリックします。

   [Web アプリケーションの作成] パネルが表示されます。



3. 次の表の説明に従ってプロパティを編集します。

Web アプリケーションの作成	
フィールド	**説明**
JRun サーバーの名前	この Web アプリケーションの公開先の JRun サーバーを選択します。
アプリケーション名	新規アプリケーションの名前を入力します。1 つの JRun サーバー内に、同じ名前を持つアプリケーションを複数入れることはできません。 名前が JMC とログ ファイルに表示されます。

Web アプリケーションの作成	
フィールド	**説明**
アプリケーション ホスト	マルチ ホーム環境でアプリケーションを実装する場合、ドロップダウン リストからホストを選択します。そうでない場合は、既定の設定になっている [すべてのホスト] を選択します。 アプリケーション ホストの詳細については、130 ページの「アプリケーション ホストの作成」を参照してください。
アプリケーションの URL	この Web アプリケーションにアクセスするためにクライアントが使用する URL 接頭部を入力します。複数のアプリケーションに対して同一のアプリケーション の URL を使用しないでください。
アプリケーションのルート ディレクトリ	Web アプリケーションの公開先のディレクトリを入力します。または、[参照] ボタンをクリックして JRun ディレクトリ リーダーを開いてください。これは、アプリケーション ファイルで利用するドキュメントのルート ディレクトリですこのディレクトリ構造が存在しない場合、JRun により作成されます。 同一名のファイルは上書きされるので、同じルート ディレクトリに複数のアプリケーションを保存しないでください。 既定の設定は、<JRun_directory>/servers/<server_name> です。

4. [作成] ボタンをクリックします。
5. JRun サーバーを再起動します。

アプリケーションの公開

JMC を使用して、アプリケーションの Web アプリケーション アーカイブ (.war) ファイルを既存の JRun サーバーに公開することができます。.war ファイルは JSP、サーブレット、イメージなど Web アプリケーションをサポートするファイルで構成されており、サーブレット 2.2 仕様によって定義された階層構造で構成されています。また、この構造には公開記述子ファイル web.xml も含まれています。

この機能を使用して、.war ファイルがなくても、サーブレット 2.2 仕様に準拠しているアプリケーションを登録することもできます。公開できるアプリケーションの最低要件は次のとおりです。

- ルート アプリケーション ディレクトリ (/foo-app など)
- /foo-app/WEB-INF ディレクトリ
- アプリケーションのルート ディレクトリの web.xml 公開記述子

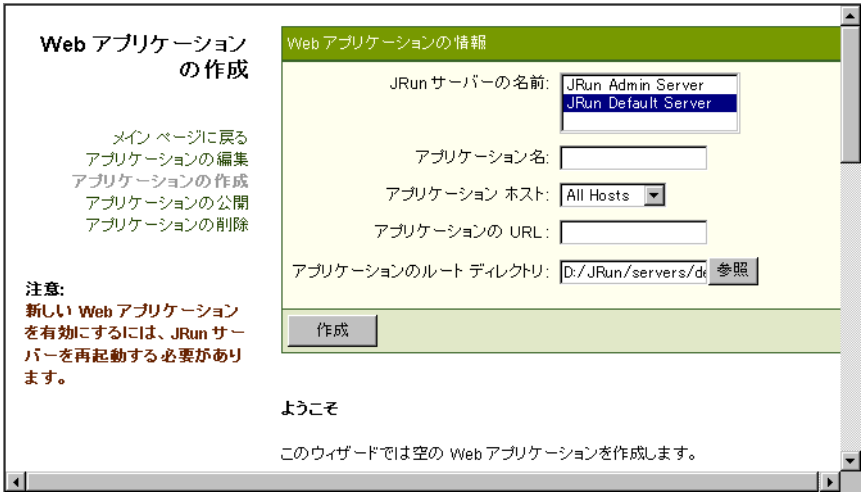
アプリケーションを公開するには

1. [*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] を選択します。

   [アプリケーション] パネルが表示されます。

2. [アプリケーションの公開] リンクをクリックします。または、[*machine_name*] > [*JRun_server_name*] を選択し、ページの上部にある [WAR 公開] リンクをクリックします。

   [Web アプリケーションの公開] パネルが表示されます。



3. 次の表の説明に従って、右側ペインにプロパティを入力します。

新規アプリケーション プロパティ	
プロパティ	**説明**
サーブレット War ファイルまたはディレクトリ	Web アプリケーションの公開前のディレクトリを入力します。または、[ブラウザ] をクリックして JRun のディレクトリ ブラウザを使用してください。たとえば、C:\temp\example.war のように入力します。 アプリケーションの .war ファイルの現在位置を指定することもできます。.war ファイルがない場合、アプリケーションの構造化ディレクトリ階層のルートを入力します。この階層は サーブレット 2.2 仕様に準拠している必要があります。
JRun サーバー名	このアプリケーションの公開先の JRun サーバーを選択します。
アプリケーション名	新規アプリケーションの名前を入力します。1 つの JRun サーバー内に、同じ名前を持つアプリケーションを複数入れることはできません。 名前が JMC とログ ファイルに表示されます。
アプリケーション ホスト	マルチ ホーム環境でアプリケーションを実装する場合、ドロップダウン リストからホストを選択します。そうでない場合は、既定の設定になっている [すべてのホスト] を選択します。 アプリケーション ホストの詳細については、130 ページの「アプリケーション ホストの作成」を参照してください。

新規アプリケーション プロパティ	
プロパティ	**説明**
アプリケーションの URL	この Web アプリケーションにアクセスするためにクライアントが使用する URL 接頭部を入力します。複数のアプリケーションに対して同一のアプリケーション URL を使用しないでください。
アプリケーションの公開ディレクトリ	Web アプリケーションの公開先のディレクトリを入力します。または、[参照] ボタンをクリックして JRun ディレクトリ リーダを開いてください。これは、アプリケーション ファイルで利用するドキュメントのルート ディレクトリですこのディレクトリ構造が存在しない場合、JRun により作成されます。 同一名のファイルは上書きされるので、同じルート ディレクトリに複数のアプリケーションを保存しないでください。 既定の設定は、<JRun_directory>/servers/<server_name>/<*application_name*> です。

4. [公開] をクリックします。

   JRun に接続している外部 Web サーバーとして WebSite Pro を使用している場合、WebSite サーバーのプロパティ アプリケーションで新規アプリケーションのマッピングを設定してください。このマッピングは、/servlet のマッピングと同一です。詳細については、68 ページの「サーブレットを実行するための URL 接頭部のマッピング」を参照してください。その他の Web サーバーについては、このマッピングを明示的に設定する必要はありません。

5. JRun サーバーを再起動します。新規アプリケーションが、[JRun_server] > [Web アプリケーション] の JMC の左側ペインに表示されます。

アプリケーションの編集

アプリケーションの設定は、公開した後に、JMC を使用して変更することができます。たとえば、アプリケーションを複数のホストにマッピングするには、アプリケーションのルート ディレクトリを変更するか、新しい URL をアプリケーションにマッピングします。

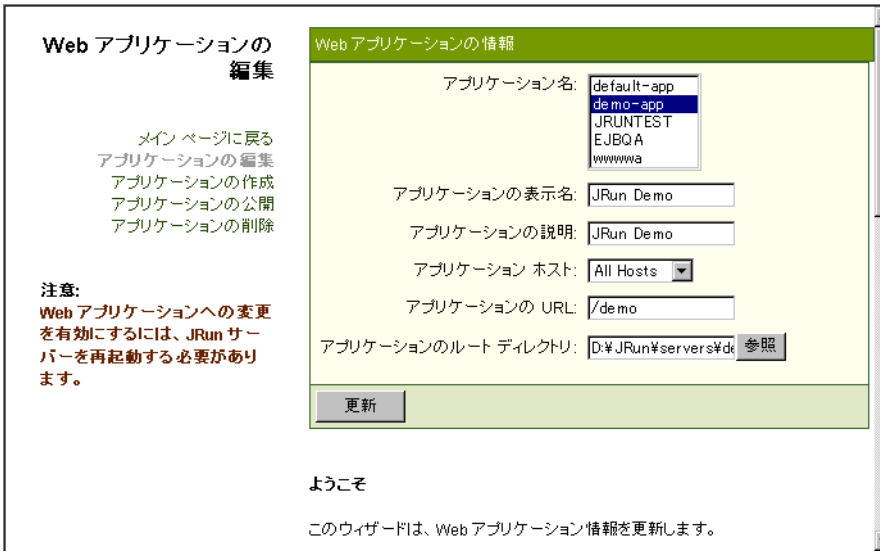
アプリケーションの設定を編集するには

1. [*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] を選択します。

   [アプリケーション] パネルが表示されます。

2. [アプリケーションの編集] リンクをクリックします。

[Web アプリケーションの編集] パネルが表示されます。



3. [アプリケーション 名] フィールドでアプリケーションを選択します。ほかのフィールドは、JRun により、アプリケーションの現在の設定が入れられます。
4. 次の表の説明に従って、右側ペインのプロパティを変更します。

アプリケーション プロパティの編集	
フィールド	**説明**
アプリケーションの表示名	アプリケーションの名前を変更します。1 つの JRun サーバー内に、同じ名前を持つアプリケーションを複数入れることはできません。 名前が JMC とログ ファイルに表示されます。
アプリケーションの説明	アプリケーションを特定するのに役立つように、アプリケーションの明細を入力、または変更します。このフィールドはオプションです。
アプリケーション ホスト	マルチ ホーム環境でアプリケーションを実装する場合、ドロップダウン リストからホストを選択します。そうでない場合は、既定の設定になっている [すべてのホスト] を選択します。 アプリケーション ホストの詳細については、130 ページの「アプリケーション ホストの作成」を参照してください。

アプリケーション プロパティの編集	
フィールド	**説明**
アプリケーション の URL	この Web アプリケーションにアクセスするためにクライアントが使用する URL 接頭部を変更します。1 つのアプリケーション ホスト内で、複数のアプリケーションに対して同一のアプリケーション の URL を使用しないでください。
アプリケーションの ルート ディレクトリ	Web アプリケーションの公開先のディレクトリを変更します。または、[参照] ボタンをクリックして JRun ディレクトリ リーダーを開いてください。これは、アプリケーション ファイルで利用するドキュメントのルート ディレクトリです 同一名のファイルは上書きされるので、同じルート ディレクトリに複数のアプリケーションを保存しないでください。

5. [更新] をクリックして、変更を適用します。
6. JRun サーバーを再起動します。

アプリケーションの削除

既存のアプリケーションはいずれもが、JMC を使用して JRun サーバーから削除することができます。

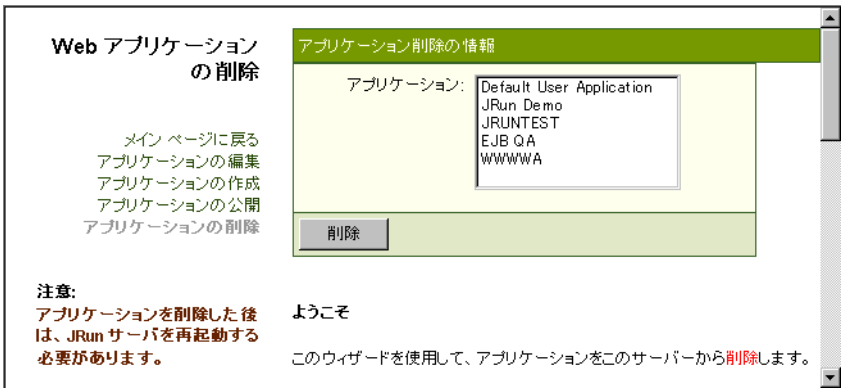
メモ JMC のアプリケーションを削除すると、そのアプリケーションの登録は取り消されますが、そのアプリケーションに関連するファイルは削除されません。

アプリケーションを削除するには

1. [*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] を選択します。

   [アプリケーション] パネルが表示されます。
2. [アプリケーションの削除] リンクをクリックします。

   [Web アプリケーションの削除] パネルが表示されます。



3. [**アプリケーション**] リストボックスに示されている利用可能なアプリケーションから、削除するアプリケーションを選択します。
4. [削除] をクリックします。
5. JRun サーバーを再起動します。

アプリケーション パスのマッピング

JRun では、アプリケーションの URL の一部を実際のディレクトリに変換するマッピングを作成することができます。たとえば、仮想パス /foo をハード ドライブ上の実際のパス c:/mydocs/temp に変換するマッピングを作成できます。このマッピングにより URL の長さを短縮し、クライアントから内部のディレクトリ構造を非表示にすることができます。この例では、c:/mydocs/temp に保管されたファイルを要求する場合、URL http://www.yourdomain.com/foo/<document_name> を入力します。

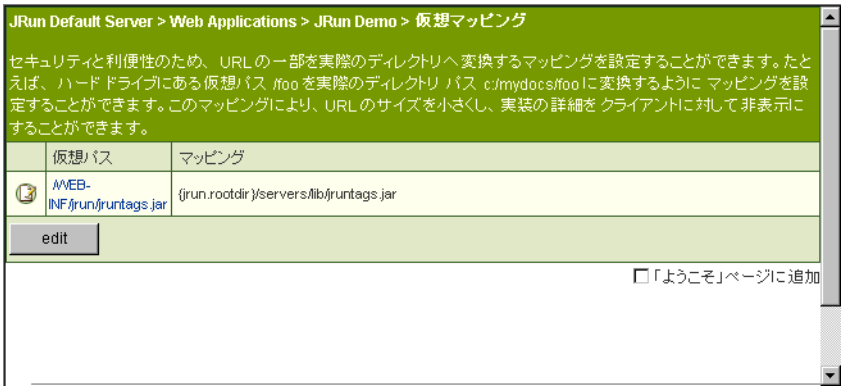
メモ JRun が外部リソースを参照するために内部的に使用する仮想マッピングがあらかじめ存在しています。このマッピングは、変更したり削除しないでください。

このセクションでは、独自のマッピングを JMC に追加する方法について説明します。

アプリケーション パスを編集するには

1. JMC の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web **アプリケーション**] > [*application_name*] > [**仮想マッピング**] を選択します。

   [仮想マッピング] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[仮想マッピング] 編集ウィンドウが表示されます。
3. [**仮想 パス**] フィールドに、実際のパスにマッピングする URL の部分を入力します。たとえば、/foo と指定します。
4. [マッピング] フィールドには、URL の **仮想パス**に置き換えられる部分を入力します。たとえば、c:/mydocs/temp と指定します。このフィールドでは JRun 変数を使用することができます。
5. パスを削除するには、[**削除**] チェックボックスをオンにします。
6. 変更を適用するには、[**更新**] をクリックします。
7. JRun サーバーを再起動すると、変更内容が反映されます。

アプリケーション ホストの作成

Web サイトを運営する場合、IP アドレスが 1 つだけの単一の Web サーバーで、複数の Web サイトのホスト名を設定する (www1.company.com、www2.company.com など) か、複数のドメインをホストする (www.yourdomain.com、www.mydomain.com) のが一般的です。このような運営方法は、マルチホーミングまたは仮想ホスティングと呼ばれます。

マルチホーミング環境では、Web アプリケーションにいくつかの問題があります。サーブレットの仕様によっては、1 つの Web アプリケーションに対して 1 つの Web サイトのホスト名を関連付けることができるだけです。さらに、サーブレットは、同一のホスト内の他のアプリケーションを参照するためには、サーブレット自身のコンテキスト情報を使用することができなければなりません。

異なる DNS ホスト名を使用して同じアプリケーションを呼び出せるようにするために、JRun では、アプリケーション ホストという構想を導入しています。アプリケーション ホストは、1 つのアプリケーションを、そのアプリケーションにアクセスすることができる DNS ホストをひとまとめにしてそれに対してマッピングするものです。

アプリケーション ホストによって、各アプリケーションは、任意の数の Web サイトのホスト名にマッピングすることができます。これが必要となるのは、1 つの Web サーバーの IP アドレスに対して複数の Web サイトのホスト名がマッピングされているマルチホーミング環境のみです。マルチホーミングを使用していない場合は、アプリケーション ホストを作成する必要はなく、アプリケーションを公開するときには、アプリケーション ホストの代わりに既定の設定を使用することができます。

アプリケーション ホストを作成して使用するのは、他段階の手順となります。

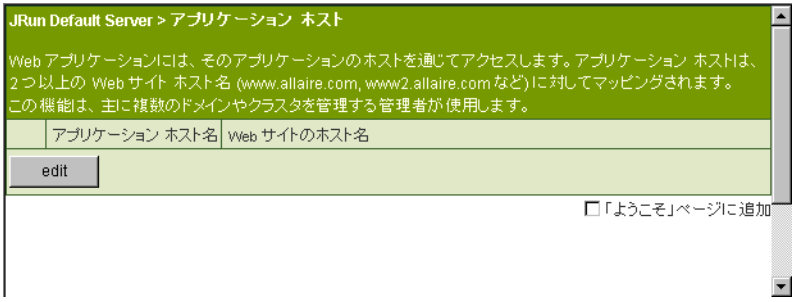
1. マルチホーミングをサポートするように DNS のエントリと Web サーバーを設定してください。詳細については、Web サーバーのマニュアルを参照してください。
2. アプリケーション ホストを作成して、任意の数の Web サイトのホスト名をそれに割り当てます。これの手順は次のとおりです。
3. アプリケーションを作成、または公開するときに、アプリケーションに対するアプリケーション ホストを選択します。これにより、アプリケーションにアクセスするために使用される DNS 名が 1 組として定義されます。詳細については、87 ページの「アプリケーションの作成」、または 88 ページの「アプリケーションの公開」をそれぞれ参照してください。

JMC の [アプリケーション ホスト] 編集ウィンドウを使用して、アプリケーション ホストによって複数のホストを 1 つのアプリケーションに結び付けてください。これにより、1 つのホストから Web サイトにアクセスした場合、そのホストに結び付けられているアプリケーションだけが表示されます。このセクションでは、新規アプリケーション ホストを作成する方法について説明します。

アプリケーション ホストを作成するには

1. JCM の左側ペインで、[machine_name] > [JRun_server_name] > **[アプリケーション ホスト]** を選択します。

   [アプリケーション ホスト] パネルが表示されます。



2. [編集] をクリックします。[アプリケーション ホスト] 編集ウィンドウが表示されます。

3. [アプリケーション ホスト名] フィールドにアプリケーション ホストの名前を入力します。この名前は重複してはならず、スペースまたは特殊文字を使用することもできません。
4. [Web サイト のホスト 名] フィールドにホストのコンマ区切りの一覧を入力します。

   完全修飾のホスト名をアプリケーション ホストに割り当てる必要はありません。たとえば、アプリケーションが内部ネットワークで使われる場合は、アプリケーション ホストは fred1 と fred2 に割り当てることができます。あるいは、アプリケーションが内部からも外部からもアクセス可能であるならば、fred1.allaire.com、fred2.allaire.com、fred1、および fred2 を割り当てることができます。
5. アプリケーション ホストを削除するには、[**削除**] チェックボックスをオンにします。
6. [更新] をクリックして、変更を適用します。
7. アプリケーションを作成、または公開するときに、[**アプリケーション ホスト**] トップダウン リストから新規ホストを選択します。詳細については、122 ページの「アプリケーションの作成」または 123 ページの「アプリケーションの公開」を参照してください。
8. すでにアプリケーションがある場合は、[アプリケーションの編集] 編集ウィンドウを使用して新規ホストを選択してください。詳細については、126 ページの「アプリケーションの編集」を参照してください。
9. JRun サーバーを再起動します。

アプリケーション パラメータの追加

JRun では、JMC を使用して実行時にアプリケーション パラメータを指定することができます。このパラメータには、ServletContext.getInitParameter() メソッドによりサーブレットでアクセスすることができます。その変数は、Web アプリケーション内の全サーブレットに渡され、サーブレットの init() メソッドで使用可能です。サーブレットがリロードされるまで、初期化パラメータを変更することはできません。

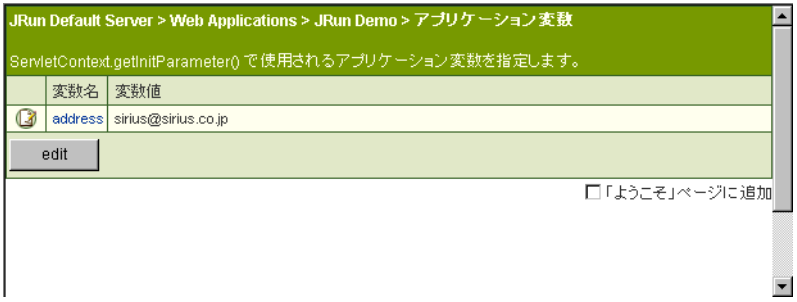
メモ 初期化パラメータを 1 つの サーブレットに渡すには、[サーブレット定義] 編集ウィンドウの [Init **引数**] フィールドを使用します。詳細については、141 ページの「サーブレットの定義」を参照してください。

初期化パラメータの使用例については、『JRun によるアプリケーションの開発』を参照してください。

アプリケーション変数を追加または変更するには

1. [*machine_name*] > [*JRun_server_name*] > [application_name] > [Web **アプリケーション**] > [**アプリケーション変数**] を選択します。

   [アプリケーション変数] パネルが表示されます。



2. [編集] をクリックします。[アプリケーション 変数] 編集ウィンドウが表示されます。
3. 新規のアプリケーション変数を追加するには、指定されたフィールドに**変数 名**と、この名前に関連する**変数 値**を入力します。たとえば、[**変数 名**] フィールドに address、[**変数 値**] フィールドに info@allaire.com と入力します。
4. アプリケーション変数を削除するには、[**削除**] チェックボックスをオンにします。
5. 変更を適用するには、[**更新**] をクリックします。
6. JRun サーバーを再起動します。

ここで、サーブレット コード内に ServletContext.getInitParameter("address") を呼び出すと、値 info@allaire.com が返されます。

ファイル設定の変更

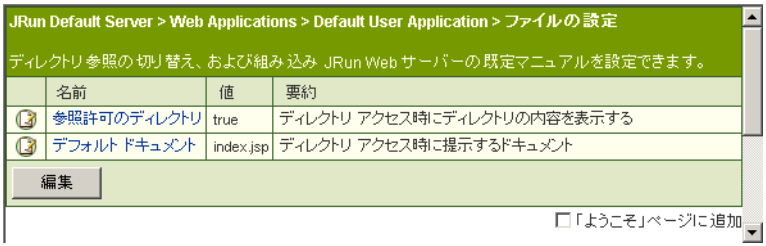
JRun のファイル設定では、ドキュメント名が URL で指定されていない場合に、JRun アプリケーションによって使用される既定のドキュメント名の順番が制御されます。また、JRun では、ディレクトリの参照を制御することもできます。

このセクションでは、JMC のファイル設定を変更する方法について説明します。

JRun アプリケーションのファイルの設定を編集するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [**ファイル 設定**] を選択します。

   [ファイルの設定] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[ファイルの設定] 編集ウィンドウが表示されます。
3. 次の表の説明に従ってプロパティを編集します。

ファイルのプロパティ	
プロパティ	**説明**
ディレクトリの参照許可	要求したファイルが存在せず、ディレクトリにデフォルトドキュメントもない場合に、ディレクトリー覧を表示するには、チェックボックスをオンにします (true)。 デフォルトドキュメントが見つからない場合に、ブラウザに「ファイルが見つかりません」というエラーを表示するには、チェックボックスをオフにします false。 既定値は true です。
デフォルトドキュメント	ページが URL で指定されていない場合は、アプリケーションによって使用されるデフォルトページのコンマ区切りの一覧を入力します。そのリストの順番が重要となります。 たとえば、URL にページのない要求が出された場合に、JRun が最初に index.jsp を使用して、次に index.html を探すようにするには、index.jsp,index.html と指定します。

4. 変更を適用するには、[更新] をクリックします。
5. JRun サーバーを再起動します。

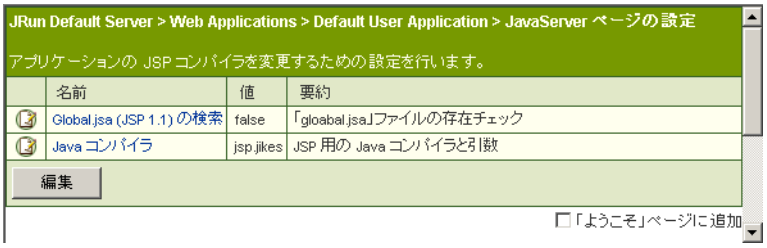
JSP コンパイラの構成

JRun では業界標準 JSP 1.1 仕様を含む JavaServer Pages (JSP) がサポートされています。このセクションで説明する内容に従って、JRun のアプリケーション レベルで JSP 設定を変更することができます。JSP コンパイラの詳細については、『JRun によるアプリケーションの開発』を参照してください。

アプリケーションの JSP プロパティを編集するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [JavaServer Pages] を選択します。

   [JavaServer Pages の設定] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[JavaServer Pages の設定] 編集ウィンドウが表示されます。

3. 次の表の説明に従ってプロパティを編集します。

JSP コンパイル プロパティ	
プロパティ	**説明**
Global.jsa の検索 (JSP 1.1)	JSP の処理中に、JRun に global.jsa ファイルを検索させる場合は、チェックボックスをオンにします (true) 。true の場合、Web サーバーにより、ディレクトリにある JSP ファイルに対する要求が初めて受信されたときに、JSP ファイルと同じディレクトリが検索されます。 global.jsa ファイルの詳細については、『JRun によるアプリケーションの開発 』を参照してください。 既定値は、チェックがオフの false です。
Java コンパイラ	JSP をコンパイルする場合は、外部 Java コンパイラへのパスを指定します。JRun のインプロセス コンパイルを使用する場合は、空白のままにしておきます。別のコンパイラが必要な場合は、適切な Java コンパイル文字列を入力してください。たとえば、次のように設定します。 D:\jdk1.1.7b\bin\javac -nowarn -classpath %c -d %d %f または jvc /cp:c %c /dest:%d %f ここで、 %c = classpath (java クラスパスが使用されます) %d = codepath (コンパイルされたクラス ファイルの保存場所) %f = filename 既定の設定は次のとおりです。 {jrun.rootdir}/bin/jikesw -classpath %c -d %d %f. 詳細については、『JRun によるアプリケーションの開発』を参照してください。

4. 変更を適用するには、[**更新**] をクリックします。
5. Web サーバーを再起動します。

JRun アプリケーション イベント ログの構成

JRun のログ記録メカニズムを使用して、ログ ファイルの内容を制御することができます。各アプリケーションにより、ログ ファイルに書き込まれるので、エラーの診断や、アプリケーションの保守ができます。

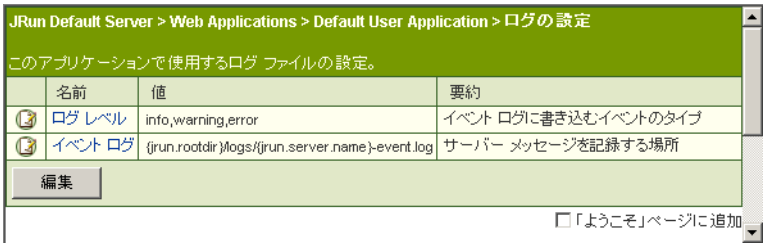
このセクションでは、JRun アプリケーションのイベント ログを構成する方法について説明します。JRun サーバーのイベント ログの構成については、106 ページの「JRun サーバー イベント ログの設定」を参照してください。

JRun ログ記録メカニズムの構成の詳細については、『JRun によるアプリケーションの開発』を参照してください。

アプリケーションのイベント ログを構成するには

1. JMC の左側ペインで、*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [ログ ファイルの設定] を選択します。

   [アプリケーション別ログ ファイルの設定] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[ログ設定エディタ] が表示されます。
3. 次の表の説明に従って、右側ペインにプロパティを入力します。

JRun アプリケーションのイベント ログのプロパティ	
プロパティ	**説明**
ログレベル	ログ ファイルに追加するログレベルをそれぞれ選択します。既定では、info、warning、error が設定されています。他にも debug と metrics のオプションがあります。
イベント ログ	ログ ファイルのパスと名前を設定します。既定値は {jrun.rootdir}/logs/{jrun.server.name}-event.log です。

4. 変更を適用するには、[更新] をクリックします。
5. JRun サーバーを再起動します。

MIME タイプのマッピング

JRun を使用すると、Web アプリケーション レベルで、特定のファイル拡張子と Multipurpose Internet Mail Extension (MIME) タイプを関連付けることができます。つまり、JRun を使用することにより、Web アプリケーション内で .html などの特定のファイル拡張子に対して要求をマッピングし、plain/text など特定の MIME タイプで応答を生成することができます。

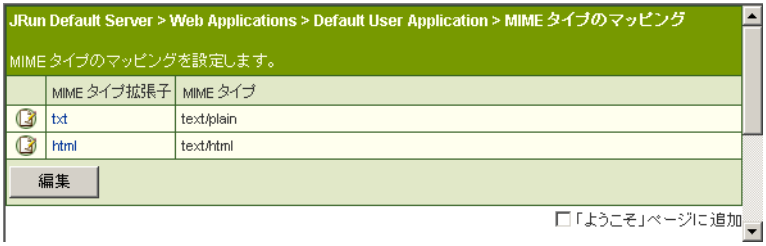
MIME タイプに基づいて、1 つのサーブレット、またはサーブレット チェーンを開始することもできます。詳細については、148 ページの「MIME タイプ フィルタリングでのサーブレットをチェーン化」を参照してください。

このセクションでは、ファイル拡張子と MIME タイプを関連付ける方法について説明します。

MIME タイプの関連付けを編集するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [MIME タイプのマッピング] を選択します。

   [MIME タイプのマッピング] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[MIME タイプのマッピング] 編集ウィンドウが表示されます。
3. [MIME タイプ 拡張子] フィールドに、MIME タイプと関連付けるファイル拡張子を入力します。たとえば、「html」と入力します。
4. [MIME タイプ] フィールドに、[MIME タイプ 拡張子] フィールドに入力した拡張子と関連付ける MIME タイプを指定します。
5. 関連付けを削除するには、[削除] チェックボックスをオンにします。
6. 変更を適用するには、[更新] をクリックします。
7. JRun サーバーを再起動します。

セッション トラッキングの構成

サーブレット 2.2 仕様には、サーブレットと JSP または EJB でセッション トラッキングに使用できる方法が用意されています。

- クッキー
- URL の書き換え
- 非表示のフォーム フィールド

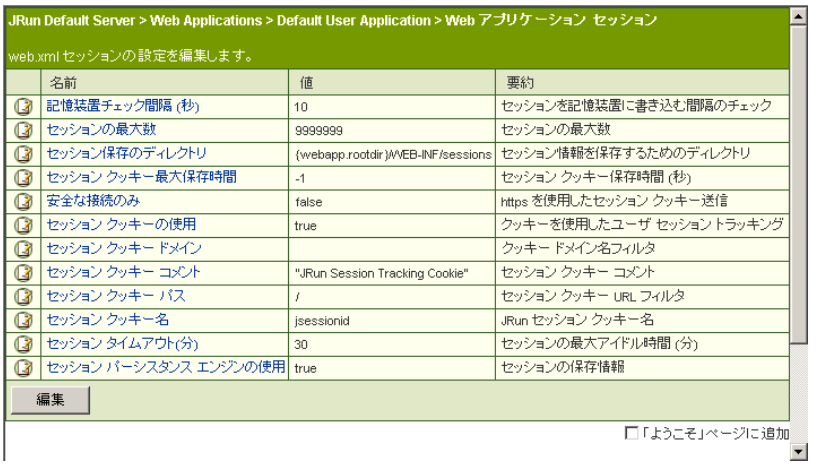
JRun の 2.2 サーブレット仕様の実装では、これらの手段がすべてサポートされていますが、JMC にはこれらのメソッドのうち最も一般的であるクッキーを構成するための簡単

な方法が用意されています。サーブレットにおけるセッション オブジェクトの使用の詳細については、『JRun によるアプリケーションの開発』を参照してください。

アプリケーションに関するセッション トラッキングのプロパティを編集するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [セッションの設定] を選択します。

   [Web アプリケーションセッション] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[Web アプリケーションセッション] 編集ウィンドウが表示されます。
3. 次の表の説明に従ってプロパティを編集します。

セッション トラッキングのプロパティ

プロパティ	説明
記憶領域チェックの時間間隔 (秒)	セッションが記憶装置に書き込まれる時間間隔を秒単位で指定します。既定の設定は 10 です。
セッションの最大数	古いセッションを開放するまでに、JRun がトラックするセッションの最大数を指定します。既定の設定は 9999999 です。
セッション保存のディレクトリ	セッションを格納するディレクトリの絶対パスを入力します。既定値は {webapp.rootdir}/WEB-INF/sessions です。

セッション トラッキングのプロパティ (続く)	
プロパティ	**説明**
セッション クッキー最大保存期間	セッション トラッキングに使用するために、ブラウザにより JRun クッキーが維持される時間を秒単位で指定します。 以下の数値には特別な意味があります。 –1 ブラウザの終了時に、クッキーが削除されます。 0 ブラウザにより、即座にクッキーが削除されます。 既定の設定は -1 です。
安全な接続のみ	クッキーが安全なプロトコル (https) のみを使用して送信されるように指定するには、チェックボックスをオンにします (true)。使用しているサーバーで安全なプロトコルがサポートされている場合のみ、このチェックボックスを使用します。 既定値は false です。
セッション クッキーのを使用	cookie を使用してユーザ セッションをトラックするには、チェックボックスをオンにします (true)。JRun のセッション トラッキングを無効にするには、チェックボックスをオフにします (false)。 既定値は true です。
セッション クッキー ドメイン	1 つのドメイン名を入力します。このドメインと一致するホストに対してのみクッキーが保管されます。特定の実装の詳細については、RFC 2109『HTTP State Management Mechanism』を参照してください。 既定では、このパラメータは空白になっているので、クッキーはすべてのドメインのホストに保管されます。
セッション クッキー コメント	JRun セッション クッキーに表示されるコメントを入力します。クッキーの目的を明確にするために、このコメントを使用します。 既定では、"JRun Session Tracking Cookie" と設定されています。
セッション クッキー パス	URL に対する制約を入力します。JRun からセッション クッキーが送られるのは、この URL で始まる要求に対してのみです。クッキーを設定したものと同じディレクトリやサブディレクトリを参照する URL も、クッキーを見ることができます。 既定の設定は / です。

セッション トラッキングのプロパティ (続く)	
プロパティ	**説明**
セッション クッキー名	JRun セッションクッキー名を入力します。既定値は、jsessionid です。
セッション タイムアウト (分)	最後のアクセスの後、セッションがそのまま残されている時間を分単位で指定します (セッション タイムアウト)。 既定の設定は 30 です。
セッション パーシスタンス エンジンの使用	JRun を終了して再起動した場合に、Java Serialization を使用してセッションを保存し、復元するには、このチェックボックスをオンにします (true)。セッション データは、JRun が適切に終了された場合のみ保存されます。 仮想マシンにセッション データを保管するだけならば、チェックボックスをオフにします (false)。これは、高可用性機能の 1 つです。 既定値は true です。

4. 変更を適用するには、[**更新**] をクリックします。
5. Web サーバーを再起動します。

サーブレットの構成

Web アプリケーションには、アプリケーションの機能を構成するサーブレットを必要な数だけ入れることができます。このセクションでは、サーブレットをアプリケーションに追加する方法、およびアプリケーション内のサーブレットの設定を変更する方法について説明します。

サーブレットの定義

サーブレットを追加、または登録する場合、JMC でそれを定義してから JRun サーバーを再起動し、コンテキスト パス内でサーブレットが認識されるようにする必要があります。このセクションでは、JMC を使用してサーブレットを追加し、サーブレットごとの設定を変更する方法について説明します。また、JRun サーバーを起動するときに、サーブレットをロードするかどうかを指定することも可能です。既定の設定では、サーブレットはプリロードされません。

サーブレットを定義するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [サーブレット 定義] を選択します。

   [サーブレット定義] パネルが表示されます。

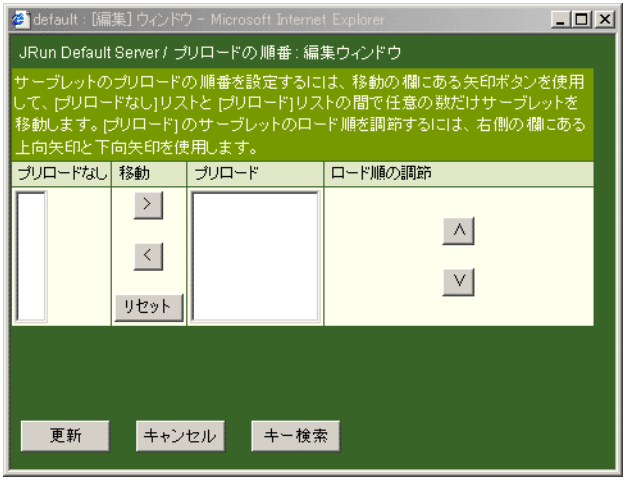
2. 右側ペインで、[編集] をクリックします。[サーブレット定義] 編集ウィンドウが表示されます。
3. 次の表で説明されているとおりにプロパティを入力します。

サーブレット構成のプロパティ	
プロパティ	**説明**
名前	サーブレット名を入力します。この値にはスペースや特殊文字を使用することはできません。たとえば、「DbFuncs」と入力します。 このフィールドは省略できません。
クラス名	サーブレットの完全修飾のクラス名を入力します。たとえば、サーブレット名が DbFuncs で、これが allaire.jrun.rds というパッケージに入っている場合、allaire.jrun.rds.DbFuncsServlet と入力します。このフィールドは省略できません。
表示名	JMC に表示されるサーブレットの短い名前を入力します。
説明	サーブレットの説明を入力します。このフィールドはオプションです。
小さいアイコン	このサーブレットを表す 16x16 ピクセルのアイコンの位置を指定します。この情報は、web.xml ファイルに保管されます。このプロパティは、所有するカタログのサーブレットをそれぞれのアイコンで表示する場合に使用するものです。このフィールドはオプションです。

サーブレット構成のプロパティ	
プロパティ	**説明**
大きいアイコン	このサーブレットを表す 32x32 ピクセルのアイコンの位置を指定します。この情報は、web.xml ファイルに保管されます。このプロパティは、所有するカタログのサーブレットをそれぞれのアイコンで表示する場合に使用するものです。このフィールドはオプションです。
Init 引数	このサーブレットに渡される初期化パラメータの一覧を入力します。 アプリケーション変数エディタを使用して、アプリケーション内にあるすべてのサーブレットに同じパラメータを渡すことができます。詳細については、132 ページの「アプリケーション パラメータの追加」を参照してください。このフィールドはオプションです。

4. サーブレットに初期化パラメータを渡す必要がない場合は、[Init 引数] フィールドにある InitParam=Value プレースホルダを削除します。
5. サーブレットの定義を削除するには、[削除] チェックボックスをオンにします。
6. 変更を適用するには、[更新] をクリックします。
7. サーバーの起動時にサーブレットをロードするかどうかを指定するには、[サーブレット定義] パネルの [プリロードの順番] をクリックします。

   [サーブレットのプリロードの順番] ダイアログ ボックスが表示されます。



矢印キーを使用して、[プリロードする] リストと [プリロードしない] リストの間でサーブレットを移動します。次に、[ロードの順番を調節] 矢印をクリックして、リスト内でサーブレットを上下に移動します。

8. 変更を適用するには、[更新] をクリックします。JRun により 1 から始まるプリロード番号がサーブレットそれぞれに割り当てられます。サーブレットは昇順にプリロードされます。
9. JRun サーバーを再起動します。

サーブレットを [サーブレット定義] 編集ウィンドウで変更した後で、このサーブレット名を変更するには、既存のサーブレットの定義をすべて削除し、再追加する必要があります。

サーブレットへのリクエストマッピング

JMC を使用して、サーブレットを URL パターンにマッピングすることができます。

JRun により、HTTP の要求が受信されると、サーブレット エンジンにより、定義済みコンテキスト パスと URL パターンが比較され、次にアプリケーション内にある登録済みサーブレットと URL パターンの次の部分が比較されます。最後に、残っているパス情報があれば、この情報がサーブレットに渡されて処理されます。パターンがどのコンテキスト パスにも一致しない場合、JRun により、既定のアプリケーションに要求が渡されます。

また、ファイル拡張子をサーブレットにマッピングし、サーブレットのチェーンを構成して、1 つずつ実行することも可能です。以下のセクションでは、JMC を使用してこれらのタスクを実行する方法について説明します。

JRun によるファイルの処理方法と、URL リクエスト を解析する方法の詳細については、『JRun によるアプリケーションの開発』を参照してください。

サーブレットの URL マッピング

JRun を使用して、サーブレットを URL 接頭部にマッピングすると、HTTP の要求からサーブレットへアクセスする方法を制御できるようになります。JMC を使用してできることは次のとおりです。

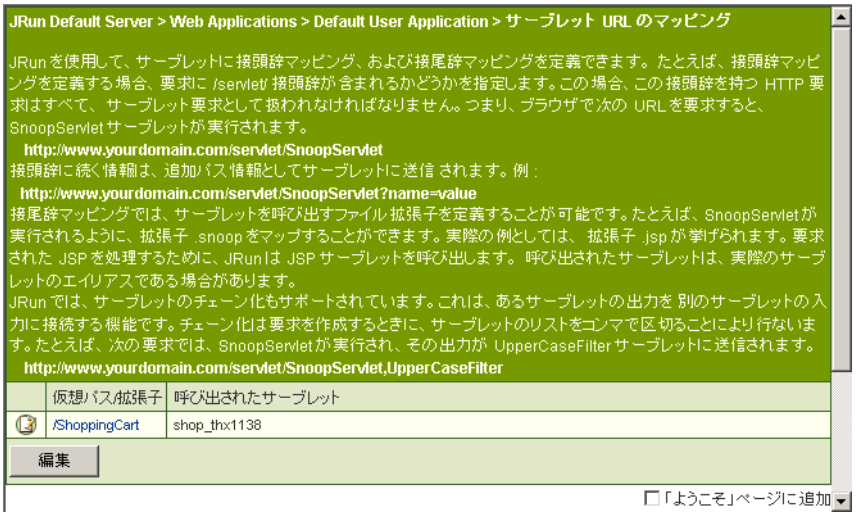
- 個々のサーブレットやサーブレット エイリアスに URL パターンをマッピングします。たとえば、要求を http://www.yourdomain.com/demo にマッピングし、JRunDemoServlet を開始することができます。
- URL パターンを JRun invoker サーブレットにマッピングします。たとえば、ユーザが http://www.yourdomain.com/ShoppingCart/<anyservlet> を要求し、/ShoppingCart が invoker にマッピングされると、<anyservlet> の値が invoker サーブレットに渡され、サーブレットとして実行されます。ただし、この方法は、Web アプリケーションがサーブレット仕様の対象になる前に設計されているので、お奨めできません。この方法でマップを行うと、リソース参照にエラーが発生します。

既定の設定では、/servlet 接頭部を含む HTTP の要求は http://www.yourdomain.com/demo/servlet/SimpleServlet と同様、invoker にマッピングされます。サーブレットにマップできる URL の数には制限はありません。

サーブレットの URL 接頭部へマッピングするには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [サーブレット URL のマッピング] を選択します。

   [サーブレット URL のマッピング] パネルが表示されます。



2. 右側ペインで、[編集] をクリックします。[サーブレット URL のマッピング] 編集ウィンドウが表示されます。

3. [**仮想 パス / 拡張子**] フィールドに、サーブレットを呼び出す URL 接頭部を入力します。たとえば、「/demo」や「/ShoppingCart」のように指定します。次のフィールドで指定されたサーブレットを、この Web アプリケーションのデフォルトサーブレットとするには、'/' を入力します。

4. [**呼び出されるサーブレット**] フィールドに、URL 接頭部により呼び出されるサーブレットを入力します。呼び出されるサーブレットは、実際のサーブレットのエイリアスである場合もあります。エイリアスの定義については、146 ページの「サーブレットのエイリアス設定」を参照してください。

5. サーブレットのマッピングを削除するには、[**削除**] チェックボックスをオンにします。

6. 変更を適用するには、[**更新**] をクリックします。

7. JRun サーバーを再起動します。

接尾部を持つファイル拡張子のマッピング

JRun では、どのようなファイル拡張子でも、サーブレットにマッピングすることができます。既定の設定では、拡張子 *.jsp が暗黙的に invoker サーブレットにマッピングされていますが、明示的なマッピングに上書きすることができます。

サーブレットのファイル拡張子をマッピングするには

1. [*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [**サーブレット URL のマッピング**] を選択します。

   [サーブレット URL のマッピング] パネルが表示されます。

2. [編集] をクリックします。[サーブレット URL のマッピング] 編集ウィンドウが表示されます。

3. [**仮想 パス / 拡張子**] フィールドに、サーブレットにマッピングする拡張子を入力します。この拡張子を使って要求されたすべてのファイルをサーブレットにマッピングするには、ワイルドカード文字 (*) を使用します。たとえば、「*.cfm」と入力します。

4. [**呼び出される サーブレット**] フィールドに、ファイル拡張子により呼び出されるサーブレットを入力します。このようなサーブレットは、実際のサーブレットのエイリアスである場合もあります。エイリアスの定義については、146 ページの「サーブレットのエイリアス設定」を参照してください。

5. 変更を適用するには、[**更新**] をクリックします。

6. JRun サーバーを再起動します。

サーブレットのエイリアス設定

[サーブレット URL のマッピング] 編集ウィンドウを使用して、エイリアスでサーブレットを参照することができます。これにより、サーブレットの実名など実装の詳細をユーザに対して非表示にすることができます。たとえば、ShoppingCart という名前のエイリアスに実名 shop_05022000 のサーブレットを参照させることができます。新しい名前を持つサーブレットの新しいバージョンを作成するには、1 か所のみ名前を変更します。

このサーブレットにエイリアスを設定する方法で、サーブレットのチェーンを参照することもできます。詳細については、147 ページの「エイリアスを使用したサーブレットのチェーン化」を参照してください。

このセクションでは、エイリアスを 1 つのサーブレットに割り当てる方法について説明します。

サーブレットに対するエイリアスを設定するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [**サーブレット URL のマッピング**] を選択します。

   [サーブレット URL のマッピング] パネルが表示されます。

2. 右側ペインで、[編集] をクリックします。[サーブレット URL のマッピング] 編集ウィンドウが表示されます。
3. [**仮想 パス / 拡張子**] フィールドに、サーブレットを指示するエイリアスを入力します。たとえば、「ShoppingCart」と入力します。
4. [**呼び出される サーブレット**] フィールドに、エイリアスにより参照されるサーブレット名を入力します。たとえば、「shop_05022000」と入力します。
5. サーブレットのエイリアスを削除するには、[**削除**] チェックボックスをオンにします。
6. 変更を適用するには、[**更新**] をクリックします。
7. JRun サーバーを再起動します。

サーブレットのチェーン化

JRun では、サーブレットのチェーン化がサポートされています。この機能を使用すると、あるサーブレットの出力を、別のサーブレットの入力として使用することができます。最も基本的なサーブレット チェーン化の方法は、リクエストを行うときに、サーブレットをコンマで区切ったリストとして、パス情報に追加することです。

たとえば、次の要求により、Servlet1 が実行され、その出力が Servlet2 サーブレットに送信されます。

http://www.yourdomain.com/servlet/Servlet1,Servlet2

この簡単なチェーン化メソッドでは、必要を満たさない場合もあります。JRun でのサーブレットのチェーン化には、このほかに次の 2 通りの方法があります。

- サーブレットのエイリアス設定
- MIME タイプのフィルタリング

後続の数セクションで、JRun でサーブレットをチェーン化する 2 通りの方法について説明します。

エイリアスを使用したサーブレットのチェーン化

JRun では、チェーンを形成するサーブレット リストの参照を 1 つの名前、つまりエイリアスで表すことができます。これはエイリアス設定と呼ばれます。

エイリアス設定を使ってサーブレットをチェーン化するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web **アプリケーション**] > [*application_name*] > [**サーブレット URL マッピング**] を選択します。

   サーブレット URL マップ パネルが表示されます。
2. 右側ペインで、[編集] をクリックします。[サーブレット URL マップの編集] ウィンドウが表示されます。

3. [**仮想パス / 拡張子**] フィールドで、サーブレットのチェーンを呼び出すエイリアスを指定します。たとえば、"/Samples" と入力します。
4. [**呼び出されるサーブレット**] フィールドに、実行順に並べられたサーブレットをコンマ区切りのリストとして入力します。

   たとえば、SnoopServlet からの出力を UpperCaseFilter に送信するチェーンを作成する場合は、"SnoopServlet, UpperCaseFilter" と入力します。
5. サーブレットのチェーンを削除するには、[**削除**] チェックボックスをオンにします。
6. 変更を適用するには、[**更新**] をクリックします。
7. JRun サーバーを再起動します。

MIME タイプ フィルタリングでのサーブレットをチェーン化

応答タイプの設定に加え、MIME タイプのマップを使用して、サーブレット チェーンを実行することもできます。しかし、エイリアスを使用した場合のように、要求にしたがってチェーン化するサーブレットを明確に指定するのではなく、サーブレットやサーブレットのチェーン実行のトリガとなる送信 MIME タイプを指定します。

たとえば、text/plain MIME タイプを UpperCaseFilter サーブレットにマップすると、このアプリケーションで text/plain というコンテンツ タイプで応答するサーブレットがすべて、UpperCaseFilter にチェーン化されます。コンテキスト タイプ text/plain で応答するその他のタイプの要求でも、UpperCaseFilter サーブレットが起動されるため注意が必要です。

JRun サーバーである応答の出力がフィルタされ、別のサーブレットやサーブレット チェーンに渡されるので、このプロセスはフィルタリングと呼ばれます。

MIME タイプ フィルタリングでサーブレットをチェーン化するには

1. JCM の左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] > [*application_name*] > [MIME タイプ のチェーン化] を選択します。

   [MIME タイプ チェーン化] パネルが表示されます。

2. 右側ペインで、[編集] をクリックします。

   [MIME タイプのチェーン化] 編集ウィンドウが表示されます。

3. [MIME タイプ] フィールドに、MIME タイプを xxx/yyy のフォームで入力します。たとえば、次のように設定します。

   text/vnd.wap.wml
   text/plain
   text/html
   image/gif
   image/jpg

4. [呼び出される サーブレット] フィールドには、MIME タイプにより起動されるサーブレット (またはエイリアス) の名前を入力します。このフィールドにはサーブレットのチェーンを、コンマで区切って入力することも可能です。たとえば、UpperCaseFilter, SpellCheckFilter と指定します。
5. MIME フィルタを削除するには、[削除] チェックボックスをオンにします。
6. 変更を適用するには、[更新] をクリックします。
7. JRun サーバーを再起動します。

SSI タグレットの使用

JRun には、Server-Side Include (SSI) タグレットを使用して、HTML ファイルに Java サーブレットを埋め込むオプションが用意されています。タグレットにより、SHTML ファイルで一意のタグをフレキシブルに定義し、実装することができます。

SSI は以前はダイナミックなコンテンツを作成するために広く使用されていましたが、JRun では古いインプリメンテーションにも対応できるようにするため、この機能も使用しています。従来のタグレットから Java Server Pages (JSP) と Java サーブレット テクノロジに代わり、SSI の機能が大幅に拡張されました。

このセクションでは、JRun で使用する SSI ターゲットの構成方法について説明します。SSI ターゲットの使用に関する詳細については、『JRun によるアプリケーションの開発』を参照してください。

SSI を構成するには

1. JMCの左側ペインで、[*machine_name*] > [*JRun_server_name*] > [Web アプリケーション] >[*application_name*] >[サーバーサイドインクルード]を選択します。

   [SSI 設定] パネルが表示されます。

2. [編集] をクリックします。[SSI 設定の編集] ウィンドウが表示されます。

3. 次の表の説明に従ってプロパティを入力します。

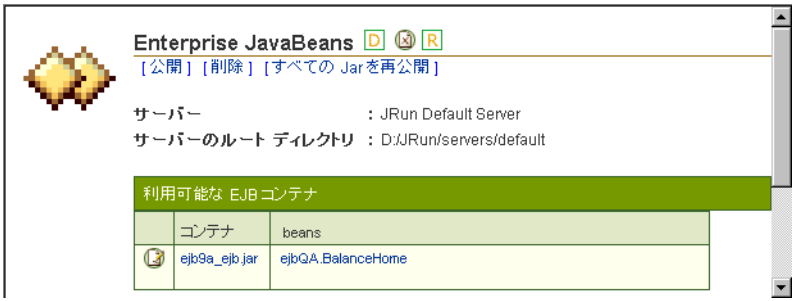
SSI プロパティ	
プロパティ	**説明**
タグレット名	タグレット名を入力する。たとえば、foo のように入力する。サーブレットを呼び出す必要がある場合は、Web ページで次のコードを使用する。 <foo></foo>
サーブレット マッピング	タグレットによって参照されるサーブレットを入力する。たとえば、SnoopServlet のように入力する。この結果、Web ページで foo タグレットを使用すると、SnoopServlet サーブレットが呼び出されるようになる。

4. SSI タグレットのマップを削除するには、[**削除**] チェックボックスをオンにします。
5. 変更を適用するには、[**更新**] をクリックします。
6. JRun サーバーを再起動します。

エンタープライズ アプリケーションの構成

JRun 3.0 では、Enterprise JavaBeans (EJB) および .ear ファイルの公開ができるようになりました。独自の EJB を開発し、ホームとリモート インタフェースを定義すると、公開の準備ができます。ただし、この「公開」という用語は、EJB の JRun に使用する場合に意味が若干異なるため、注意が必要です。サーブレットの場合は、コンパイル、テストを行ってから、最後に公開して配布します。EJB の場合は、コンパイル、テストを実行するための公開、テストを行ってから、最終的に公開して配布します。

JMC には [Enterprise JavaBeans] パネルがあり、[*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] を選択することによりアクセスできます。



このセクションでは、次のトピックについて説明します。

- 151 ページの「EJB の公開」
- 153 ページの「EJB の再公開」
- 153 ページの「EJB の削除」
- 154 ページの「EJB の構成」
- 155 ページの「EAR ファイルの公開」

JRun での Enterprise JavaBeans 使用の詳細については、『JRun によるアプリケーションの開発』を参照してください。

EJB の公開

JMC を使用して、JRun で公開する bean を作成します。JMC での公開では、次のアクションを実行します。

- 提供された .jar ファイルのリストにある EJB にホームとオブジェクト実装を生成します。
- 生成されたオブジェクトにスタブ クラスを作成します。
- deploy.properties ファイルが ejipt.isCompatible=true を指定している場合に限り、JDK 1.1 ベースのクライアントで使用する際に必要とされるスケルトンを作成します。
- deploy.properties のプロパティと現在の環境を使用して、runtime.properties を作成します。JRun では、runtime.properites ファイルを使用してランタイム環境を確立します。

JMC 公開ツールは、/deploy ディレクトリでのみ動作します。.jar ファイルなどのすべての入力が /deploy ディレクトリで使用可能であり、生成された出力がすべて /deploy ディレクトリに公開されるようにする必要があります。

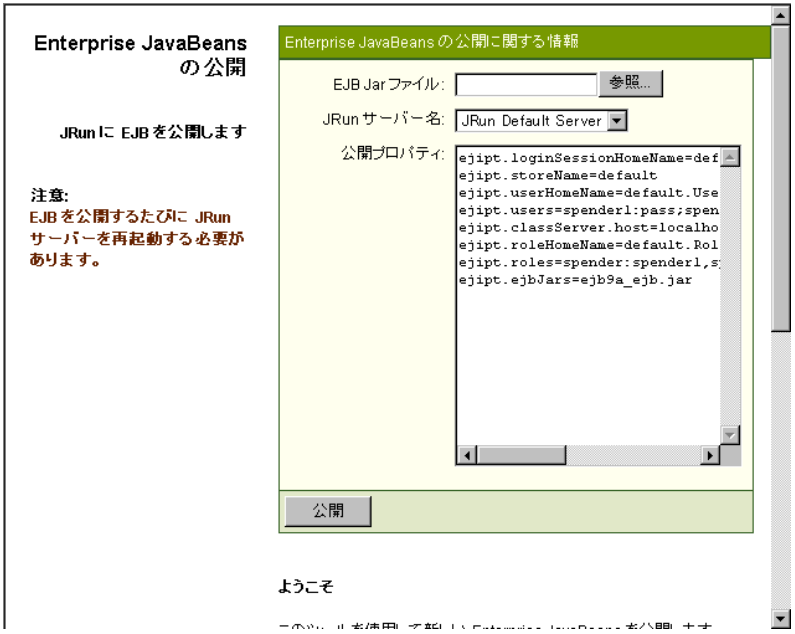
EJB を公開するには

1. [*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] を選択します。

   [Enterprise JavaBeans] パネルが表示されます。

2. ページ最上部にある [公開] をクリックします。また、[*machine_name*] > [*JRun_server_name*] をクリックしてから、ページ最上部の [EJB 公開] をクリックする方法もあります。

   [Enterprise JavaBeans の公開] パネルが表示されます。



3. 次の表の説明に従って、右側ペインにプロパティを入力します。

EJB の公開	
フィールド	**説明**
EJB Jar ファイル	EJB の .jar ファイルへのパスを入力するか [参照] をクリックして、JRun の ディレクトリ リーダー を使用する。
JRun サーバー名	EJB を公開するサーバーを選択する。
公開プロパティ	EJB のサーバー レベルの公開プロパティを編集して deploy.properties ファイルに格納する。name=value の組み合わせの変更、追加、削除が可能である。EJB を公開するときに JMC により変更を加えたファイルで deploy.properties ファイルが上書きされる。

4. [公開] をクリックします。
5. JRun サーバーを再起動します。

EJB の再公開

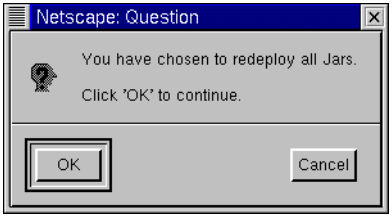
JMC の [Bean プロパティの編集] ウィンドウか、別の <bean_name>.properties ファイル変更ツールのいずれかを使用して bean のプロパティを変更する場合は、その都度、対象の EJB が組み込まれた .jar ファイルを再公開する必要があります。このセクションでは、JRun サーバー上にある .jars を一括して再公開する方法ついて説明します。

1. [*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] を選択します。

   [Enterprise JavaBeans] パネルが表示されます。

2. ページ最上部にある [すべての Jar を再公開] をクリックします。

   [OK] か [キャンセル] をクリックするように要求されます。



メモ すべての .jar ファイルをサーバーに再公開するときは、ファイルのサイズと数量に応じた時間がかかります。

3. [OK] をクリックします。

   JRun では、以前そのサーバで公開された .jar ファイルがすべて再公開されます。

EJB の削除

指定された .jar ファイルから EJB を削除する手順です。ファイル システムから実際にファイルが削除されるわけではありません。JRun サーバーへの登録だけが削除されます。

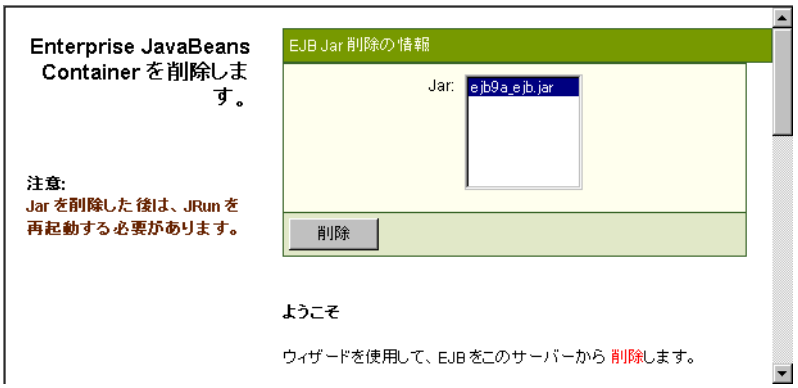
EJB を削除するには

1. [*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] を選択します。

   [Enterprise JavaBeans] パネルが表示されます。

2. ページ最上部にある [削除] をクリックします。

   [EJB コンテナ削除] パネルが表示されます。



3. [アプリケーション] リストボックスで削除する .jar ファイルを選択します。
4. [削除] をクリックします。
5. JRun サーバーを再起動します。

EJB の構成

JMC を使用すると、管理できる bean コンテキストの数を構成することができます。bean コンテキストを使用して、公開された bean のインスタンスのステータスに関する情報を取得します。コンテキストは、bean インスタンスの作成時に作成され、その bean インスタンスが存在する限り bean とともに保持され、他の bean インスタンスでは使用できません。コンテキストには、インスタンスのステータスの変更など、bean インスタンスに関する情報が格納されます。

利用できるコンテキストの数は、JMC の ejipt.maxContexts、ejipt.maxFreeContexts、および ejipt.minFreeContexts を設定することで管理できます。JMC は、この情報を beans の default.properties ファイルに書き込みます。JMC では他の bean プロパティも [Bean プロパティの編集] ウィンドウに表示されます。このセクションでは、このような bean プロパティを編集する方法について説明します。

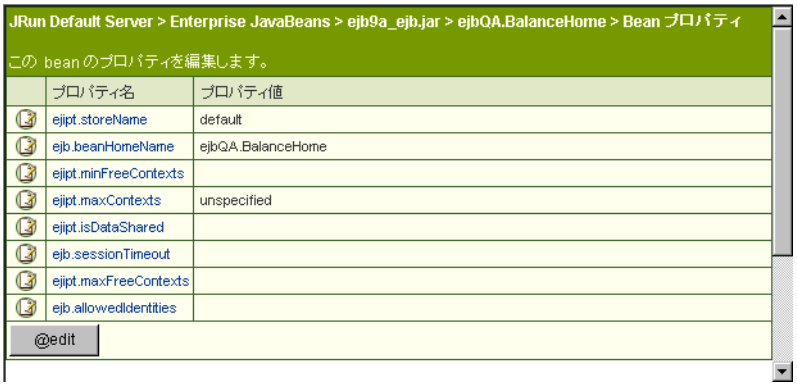
メモ 既存の bean プロパティを変更する場合は、bean の .jar ファイルの再公開が必要です。

bean コンテキスト プロパティの使用の詳細については、JRun JavaDocs ファイルに付属する「Ejipt Properties API」のマニュアルおよび『JRun によるアプリケーションの開発』を参照してください。

EJB 設定を構成するには

1. [*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] > [jar_file] > [bean] を選択します。

   [Bean プロパティ] パネルが右側ペインに表示されます。



2. [編集] をクリックします。[Bean プロパティの編集] ウィンドウが表示されます。
3. JRun Java Doc ファイルに付属する「Ejipt Properties API」マニュアルの説明にしたがってプロパティを編集します。
4. [更新] をクリックして、変更を適用します。
5. [*machine_name*] > [*JRun_server_name*] > [Enterprise JavaBeans] を選択します。

   [Enterprise JavaBeans] パネルが表示されます。
6. ページ最上部にある [すべての Jar を再公開] をクリックします。[OK] か [キャンセル] をクリックするように要求されます。

   メモ すべての .jar ファイルをサーバーに再公開するときは、ファイルのサイズと数量に応じた時間がかかります。
7. [OK] をクリックします。

   JRun では、以前そのサーバで公開された .jar ファイルがすべて再公開されます。
8. JRun サーバーを再起動します。

EAR ファイルの公開

.ear ファイル (Enterprise Application アーカイブ) には、ディレクトリ構造のすべてとエンタープライズ アプリケーションを定義するファイルのすべてが組み込まれます。.ear ファイルは、.jar ファイルの作成に使用したのと同じツールで作成します。J2EE アプリケーション公開の際に、JRun では .ear ファイルに格納された .war ファイルが変換され、指定 JRun サーバーに新しいアプリケーションが定義されます。JRun は、.ear ファイルに格納されたすべての EJB .jar ファイルも公開します。

JMC を使用して、J2EE アプリケーションを指定の環境にインストールします。.ear ファイルのインストールの際に、JMC によりサーバー固有パラメータのセットの構成、ディレクトリ構造の形成、JRun プロパティ ファイルの更新が行われます。

.ear ファイルには META-INF/application.xml 公開記述子が格納されている必要があります。JRun アプリケーション公開ユーティリティにはこの記述子から情報が提供されます。

.ear ファイルおよび J2EE アプリケーションの詳細については、『JRun によるアプリケーションの開発』を参照してください。

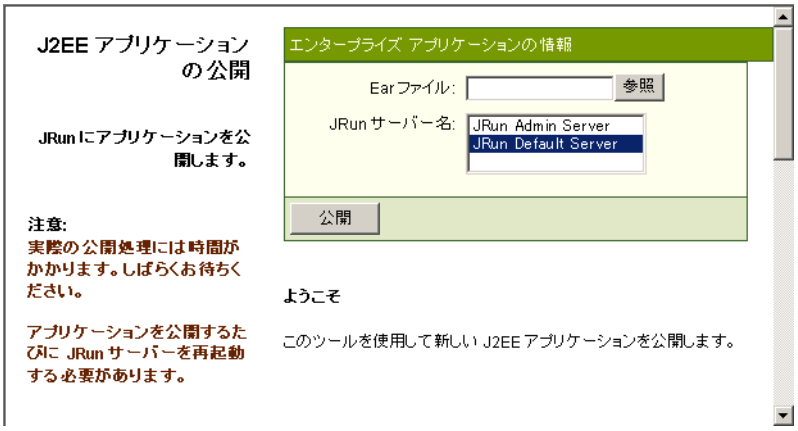
EAR ファイルを公開するには

1. [*machine_name*] > [*JRun_server_name*] を選択します。

   [JRun サーバー] パネルが表示されます。

2. [EAR 公開] をクリックします。

   [J2EE アプリケーションの公開] パネルが表示されます。



3. 次の表の説明に従って、右側ペインにプロパティを入力します。

.ear ファイルの公開	
フィールド	**説明**
EAR ファイル	.ear ファイルへのパスを入力するか [参照] をクリックして、JRun の ディレクトリ リーダー を使用する。
JRun サーバー名	.ear ファイルを公開する JRun サーバーを選択する。

4. [公開] をクリックします。
5. JRun サーバーを再起動します。

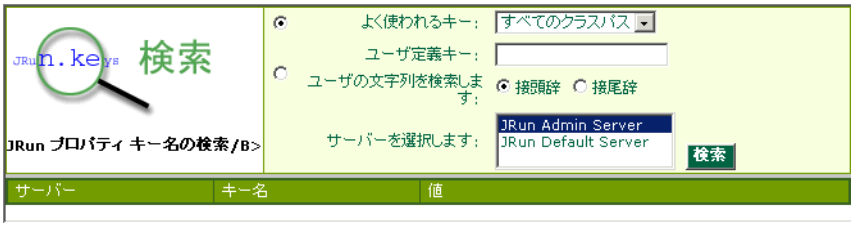
JMC キーの検索

キー検索機能を使用して、1 つ以上の JRun サーバー内で接頭辞や接尾辞により、キー (プロパティ名) を検索することができます。

キー検索を実行するには

1. アクセス バーの [**キー検索**] をクリックします。さまざまな JMC パネルにある [**キー検索**] ボタンをクリックすることもできます。

   右側ペインにキー検索ウィンドウが表示されます。



2. 一般的に使用されるキーを検索するには、[よく使われるキー] を選択し、ドロップダウン リストボックスからキーを選択します。
3. 追加したキーを検索するには、[**ユーザ 定義 キー**] を選択し、指定されたフィールドにキーを入力します。次に、適切なボタンをクリックして、このキーが接頭辞であるか、接尾辞であるかを指定します。
4. [**サーバーの 選択**] リストボックスで検索の対象となる JRun サーバーを選択します。複数の JRun サーバーを選択するときは、まず 1 つのサーバーをクリックし、Ctrl キーを押しながら、残りの JRun サーバーを選択します。
5. [検索] をクリックします。ウィンドウの下部ペインには検索の結果が表示されます。

ログアウト

変更の中には、JMC からログアウトして、変更内容を有効にするものもあります。

JMC からログアウトするには

1. アクセス バーの [**ログアウト**] をクリックします。

   これで JRun からログアウトできます。[ログイン] 画面が表示されます。

第 4 章

コネクタ

この章では、外部 Web サーバーに接続する場合の JRun アーキテクチャについて説明します。また、いくつかの一般的な事例と、分散環境で JRun を使用する場合に役立つセキュリティと要求のチェーン化などの問題についても取り上げます。

目次

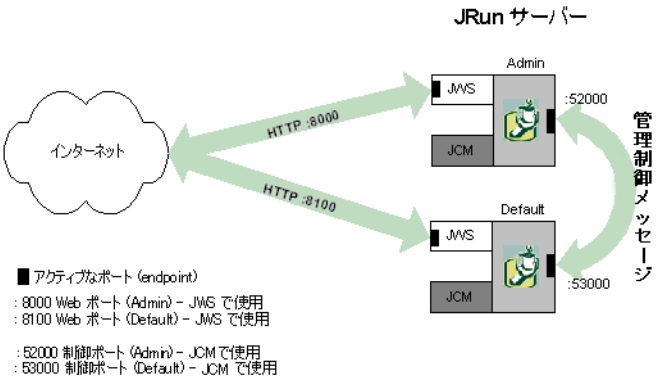
JRun ポートについて

JRun を管理する際に認識する必要がある、JRun サーバーで使用される既定のポートが 4 つあります。100 ページの「JRun サーバーの追加および削除」に記載されているように新しい JRun サーバーを追加するときには、これらのポートの認識が特に重要です。

各ポートは JRun サーバーごとに固有でなければなりません。これらのポートは以下のとおりです。

- 2 つの JWS ポート (admin および default JRun サーバー)
- 2 つの制御ポート (admin および default JRun サーバー)

次の図は、関連する既定のポート間の対話を示しています。この図は既定の JRun セットアップを想定しています。

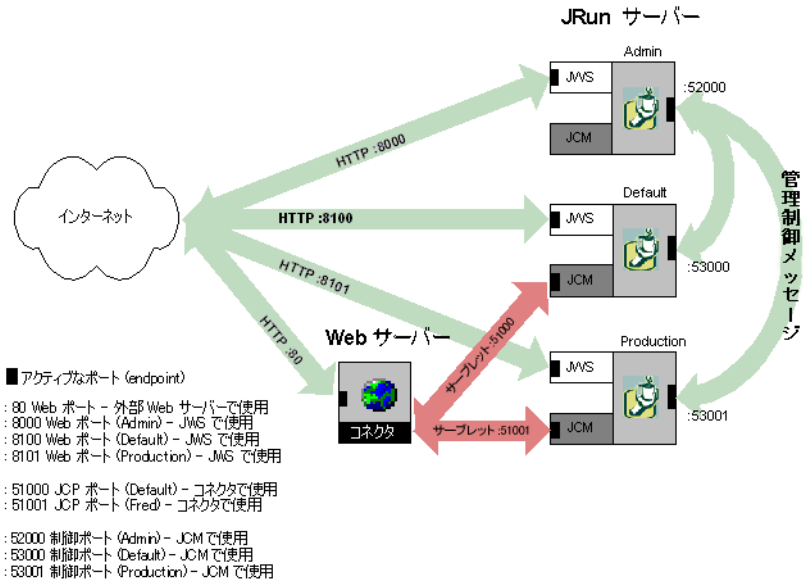


コネクタ ウィザードを実行して JRun を外部 Web サーバーに接続する場合、次のポートも認識する必要があります。

- JCP ポート (接続された JRun サーバーごとに 1 つ)

JCP ポートによって、JRun Connection Module (JCM) と Web サーバー コネクタとの接続が容易になります。

default および admin JRun サーバーのほかに JRun サーバーを追加する場合、その制御ポートおよび JWS ポートを認識する必要があります。追加した JRun サーバーに対してコネクタ ウィザードを実行した場合、その JCP ポートも認識する必要があります。次の図は、これらのポート間の対話を示しています。



次の表は、前に説明した各ポートと、EJB サーバーに使用されるその他のポートを示しています。また、JRun サーバーを新たに追加する際に各ポートに使用する推奨範囲も示しています。

ポート名 (local.properties 内)	説明	既定値 (JRun サーバーごと)	推奨範囲
web.endpoint.main.port	JRun Web サーバー (JWS) で使用されるポート。既定で、各 JRun サーバーに関連付けられた JWS があります。これはその JWS の HTTP ポートです。	admin: 8000 default: 8100	8101 - 8199
jcp.endpoint.main.port	JRun Connection Module (JCM) とも呼ばれる JRun コネクタプロキシ (JCP) に使用されるポート。このポートによって、JRun は外部 Web サーバーの JRun コネクタと通信できます。	コネクタ ウィザードを実行して JRun サーバーを外部 Web サーバーに接続しない場合、default サーバーはブランクになります。 デフォルトは 51000 です。	51001 - 51999

ポート名 (local.properties 内)	説明	既定値 (JRun サーバーごと)	推奨範囲
control.endpoint.main.port	admin JRun サーバーがほかの JRun サーバーに制御メッセージを送信するために使用するポート。	admin: 52000 default: 53000	53001 - 53999
ejipt.classServer.port	EJB エンジンがクライアントにクラスを配布するために使用するポート。	admin: 2423 default: 2323	2324 - 2422
ejipt.homePort	EJB ホーム オブジェクト用のポート。	admin: 2433 default: 2333	2334 - 2432

空きポートの検出

JRun インストール中に空きポートが検出されます。ただし、同じコンピュータに JRun サーバーを新たに追加する場合、空きポートに対するプログラミング的なスキャンが必要なこともあります。このセクションでは、単純な Java ユーティリティを使用してこの作業を実行する方法を説明します。

JRun には、一定範囲のポートをスキャンして使用可能なポートを返す GetControlPort ユーティリティが用意されています。GetControlPort は allaire.jrun.install パッケージに含まれています。GetControlPort を呼び出すには、クラスパスに *<jrun_root_dir>*/lib/install.jar を含める必要があります。

GetControlPort は 1 つのパブリック メソッドである scan() を持ちます。このメソッドは、範囲の上限と下限の 2 つの引数を取ります。これらの引数のデータ タイプは String です。String の配列で渡すこともできます (以下を参照)。GetControlPort は、該当範囲内の最初の空きポートを返します。

次に例を示します。

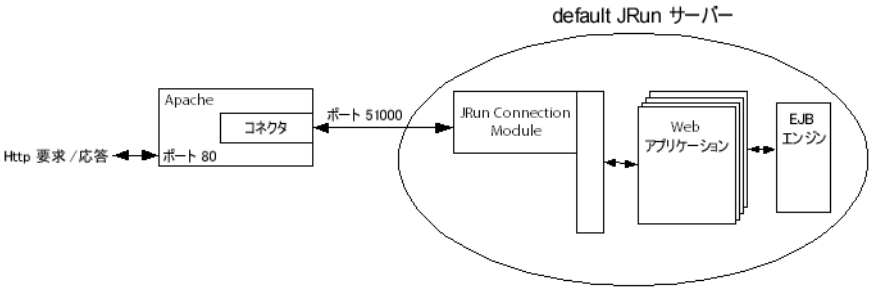
```
String[] portrange = new String[2];
String openport = new String;
portrange[0] = "8001";
portrange[1] = "8099";
GetControlPort gcp = new GetControlPort();
openport = gcp.scan(portrange);
out.println("The first open port between " + portrange[0] + " and " +
      portrange[1] + " is " + openport);
```

Web サーバー コネクタについて

ネイティブ サーバー接続モジュール、すなわち*コネクタ*は、特定の Web サーバー、ハードウェア アーキテクチャ、およびオペレーティング システムに対応してコンパイルさ

れています。たとえば、JRun で Netscape Application Server 用のコネクタを作成する場合は、NSAPI を使用し、JRun がサポートする各ハードウェア アーキテクチャおよびオペレーティング システムに合わせて作成します。JRun コネクタは、標準コネクタ サービス プロバイダ インタフェースを定義する Sun J2EE コネクタとは別のものです。

各 JRun サーバーに複数の Web サーバーを接続できます。通常の公開環境では、アプリケーションを処理する default JRun サーバーに、1 つの Web サーバーを接続します。次の図では、default サーバーに接続している Web サーバーの例を示しています。



アプリケーション リソースへの要求が発生すると、Web サーバー上のコネクタは、JRun サーバー内に存在する JRun 接続モジュールへのネットワーク接続を開きます。接続モジュールは透過的なコミュニケータ (伝達仲介者) としての役割を果たし、コネクタからの要求を変換して JRun サーバーに伝達します。JRun サーバーは要求を処理し、その結果を接続モジュール サービスに返します。

各 JRun サーバーは、それぞれ異なるネットワーク ポート番号を使用して、Web サーバーからの要求を受信します。たとえば、上の図では、default JRun サーバーはポート番号 51000 で受信します。ユーザは JRun サーバーと Web サーバー間の接続を構成する場合、このポート番号を指定する必要があります。

また、2 番目のパラメータである*バインド アドレス*を使用して、Web サーバーと JRun サーバー間の接続を定義することも可能です。バインド アドレスには、JRun サーバーが要求を受信できるネイティブ Web サーバーの IP アドレスを指定します。既定では、すべての JRun サーバーのバインド アドレスが "*" です。* は、JRun サーバーがすべての IP アドレスの Web サーバーから要求を受信することを表します。

Web サーバーのコンフィギュレーション ファイル内のコネクタのプロパティ

JRun コネクタのプロパティは、Web サーバー マシン上に格納されます。Web サーバーのコンフィギュレーション ファイルである jrun.ini には IIS インプリメンテーションのプロパティが、httpd.conf には Apache のプロパティが、そして obj.conf には Netscape/iPlanet のプロパティが含まれます。

これらのプロパティは接続モジュールの初期化を行います。その際に使用される設定では、接続モジュールが JRun サーバーを見つけて、どのサーバーに接続すべきかを判断する手助けを行います。以下の表に、Web サーバーのコンフィギュレーション ファイルに含まれる JRun プロパティを示します。

プロパティ	説明
jrun.rootdir	jrun.rootdir プロパティは、JRun のインストール先を指定します。たとえば、/opt/JRun や c:\Progra~1\Allaire\JRun のように指定します。
jvmlist	jvmlist プロパティは、この Web サーバーに接続される JRun サーバーの一覧を指定します。たとえば、default、admin、production、および qa のように指定します。
verbose	verbose プロパティは、JRun サーバーと Web サーバー間の通信に関する、より詳細なログ ファイル エントリを作成します。このログ ファイル エントリは、Web サーバーのログ ファイルに影響を与えます。既定値は false です。
proxyhost proxyport	proxyhost および proxyport プロパティは、JRun Connection Module (JCM) が接続されるネイティブ コネクタを指示します。JCM は、外部 Web サーバーからの要求を受信するサービスです。 proxyhost プロパティは、JRun を実行するコンピュータの IP アドレスを参照します。proxyport プロパティは、JRun が Web サーバーからの要求を受信するポートを参照します。Web サーバーは、proxyhost アドレス /proxyport の組み合わせを使用して、ソケットに要求を送信します。JRun をこのポート / アドレスの組み合わせで受信するように設定することも必要です (詳細については、167 ページの「 local.properties ファイル内のコネクタのプロパティ」を参照してください)。 コネクタ ウィザードを実行する場合、proxyhost および proxyport は JMC によって設定されます。ただし、同一コンピュータ上にない Web サーバーに接続するときは、場合によっては、これらのプロパティを手動で編集する必要があります。

プロパティ	説明
rulespath	rulespath プロパティは、local.properties ファイルの場所を指定します。JRun がリモート コンピュータにインストールされている場合、それらが同じファイル システムにあるときは、rulespath プロパティにリモート コンピュータへのフル パスを指定します。一方、異なるファイル システムにあるときは、local.properties ファイルのコピーをローカル ファイル システム上に作成します。local.properties ファイルには、JRun に渡す要求をネイティブ コネクタに指示するマッピング ルールが含まれています。
scriptpath	IIS のみに適用されます。scriptpath プロパティは、Web サーバー上の仮想 /scripts ディレクトリを指定します。
errorurl	errorurl プロパティは、カスタマイズしたエラー メッセージを指定します。このプロパティはオプションです。既定では、このプロパティはコメント化されています。エラー メッセージのカスタマイズの詳細については、『JRun によるアプリケーションの開発』の第 40 章を参照してください。

Web サーバーのコンフィギュレーション ファイルのサンプル

このセクションでは、Web サーバーを JRun に接続する場合の Web サーバーのプロパティの例を示し、Web サーバーのコンフィギュレーション ファイルのパラメータを具体的に説明します。ここで説明する例では、JRun と Web サーバーが同一のコンピュータ上にあるものとします。

Apache コンフィギュレーション ファイル

Apache Web サーバーと同一のコンピュータに JRun の標準インストールを行った場合、httpd.conf ファイルは次のようになります。

LoadModule ステートメントは、マルチ ホスティング環境での VirtualHost ディレクティブの外部に記述する必要があります。これは、LoadModule ステートメントの参照はグローバル レベルで 1 回だけ行うようにする必要があるためです。LoadModule ステートメントが必要となるのは、JRun を Dynamic Shared Objects (DSO) モジュールとして使用する場合に限られます。

また、Apache コンフィギュレーション ファイルでは、rulespath の代わりに Mappings を使用します。

```
LoadModule jrun_module136 "/opt/JRun/connectors/apache/intel-linux/mod_jrun.so"
<IfModule mod_jrun.c>
    JRunConfig jrun.rootdir "/opt/JRun/bin/.."
    JRunConfig jvmlist default
```

```
    JRunConfig Verbose false
    JRunConfig ProxyHost 127.0.0.1
    JRunConfig ProxyPort 51000
    JRunConfig Mappings "/opt/JRun/servers/default/local.properties"
</IfModule>
```

IIS コンフィギュレーション ファイル

IIS の場合、JRun は jrun.ini ファイルを使用して jrun.dll フィルタを初期化します。一般的な jrun.ini は以下のようになります。

```
jrun.rootdir=C:/PROGRA~1/Allaire/JRun
jvmlist=default
verbose=false
proxyhost=127.0.0.1
proxyport=51000
scriptpath=/scripts/jrun.dll
rulespath=C:/Progra~1/Allaire/JRun/servers/default/local.properties
#errorurl=<optionally redirect to this URL on errors>
```

Netscape/iPlanet コンフィギュレーション ファイル

Netscape/iPlanet Web サーバーの obj.conf ファイルでは、これらのパラメータは次のようになります。

```
Init jrun.rootdir="C:/PROGRA~1/Allaire/JRun" proxyport="51000" verbose="false"
proxyhost="127.0.0.1" jvmlist="default" rulespath="C:/Program Files/Allaire/JRun/
servers/default/local.properties" fn="jruninit"
```

local.properties ファイル内のコネクタのプロパティ

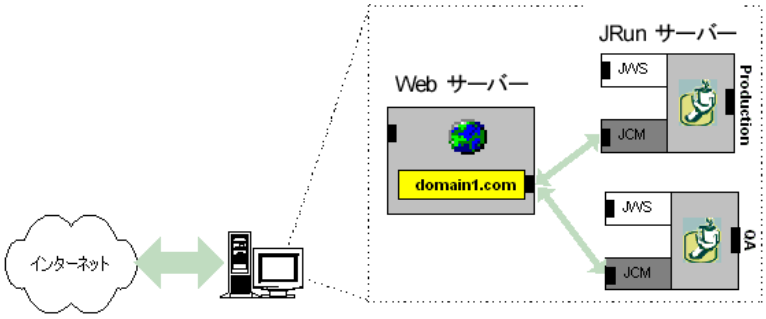
各 JRun サーバーの local.properties ファイルの JCP Services セクションには、JRun コネクタに影響を与えるプロパティが含まれます。以下の表に、これらのプロパティを示します。

プロパティ	説明
jcp.endpoint.main. interface	JMC の [外部 Web サーバー アドレス] フィールドに対応します。このプロパティを使用すると、JRun に接続可能な Web サーバーを制限できます。既定値は、すべての Web サーバー (*) です。JRun サーバーは、jcp.endpoint.main.interface プロパティと一致する Web サーバーからのみ要求を受け入れます。
jcp.endpoint.main. bindaddress	JMC の [受信アドレス] フィールドに対応します。このプロパティは、外部 Web サーバーからの要求を受信する JRun サーバーのアドレスを指定します。JRun と Web サーバーが同一のコンピュータ上にある場合は、127.0.0.1 に設定されます。
jcp.endpoint.main.port	JMC の [受信ポート] フィールドに対応します。このプロパティは、JRun サーバーが外部 Web サーバーからの接続を受信するポートを指定します。
timeout min.threads active.threads max.threads	JCP Services セクションの残りの 4 つのプロパティは、スレッドの設定を参照します。これらのプロパティは、JCP の構成に直接影響しません。これらのフィールドの詳細については、112 ページの「並行処理の概要」を参照してください。

1 つの Web サーバーへの複数の JRun サーバーの接続

1 つの JRun サーバーを 1 つの外部サーバーに接続するのは簡単です。ただし、外部 Web サーバーにさらに別の JRun サーバーを接続する場合は、特別な手順に従う必要があります。

たとえば、admin および default JRun サーバーのほかに、テスト / 品質管理用 JRun サーバーと実際の運用に使用される JRun サーバーが必要となるとします。この場合、複数の JRun サーバーを、同じコンピュータ上の 1 つの Web サーバーに接続して実行します。この事例を図に示すと、次のようになります。



このセクションでは、IIS や Apache など、1 つの Web サーバー、1 つの JRun プログラム、および複数の JRun サーバーが*すべて同一のコンピュータ*に必要であるとします。

設定の詳細

JRun は 3 つのファイルを使用して、1 台のコンピュータ上で 1 つの外部 Web サーバーに接続されている複数の JRun サーバーに対して設定を行います。それらのファイルは次のとおりです。

- Web サーバーのコンフィギュレーション ファイル (IIS の場合は jrun.ini、iPlanet/Netscape の場合は obj.conf、Apache の場合は httpd.conf など)
- jvms.properties
- 各 JRun サーバーの local.properties ファイル

これらの各ファイルでの変更については、以降のセクションで説明します。2 つ目の JRun サーバーを Web サーバーに追加する手順については、169 ページの「 Web サーバーの接続」を参照してください。

Web サーバーのコンフィギュレーション ファイル

JRun と Web サーバーが同一のコンピュータ上にある場合、Web サーバーのコンフィギュレーション ファイル内の jrun.rootdir および jvmlist プロパティによって、コネクタがすべての JRun サーバーと通信するように設定できます。次に例を示します。

```
JRunConfig jrun.rootdir "/opt/JRun/.."
JRunConfig jvmlist default,production,qa
```

JRun は jrun.rootdir プロパティを使用して jvms.properties ファイルを検索します。jvmlist プロパティは、jvms.properties の一覧に示されたサーバーを確認するために使用されます。

jvms.properties ファイル

jvms.properties ファイルによって、各 JRun サーバーのルート ディレクトリが JRun に提供されます。次に例を示します。

```
default=/opt/JRun/servers/default
production=/opt/JRun/servers/production
qa=/opt/JRun/servers/qa
```

各 JRun サーバーの local.properties ファイル

各 JRun サーバーのルート ディレクトリを取得した後、JRun は local.properties ファイルを検索することができます。JRun は各サーバーのファイルを読み取り、JCP の設定情報 (バインド アドレスとポート) を調べます。

```
## jcpservices
jcp.endpoint.main.bindaddress=127.0.0.1
jcp.endpoint.main.port=55555
```

JRun と Web サーバーが同一のコンピュータ上にない場合、JRun は、Web サーバーのコンフィギュレーション ファイル内の proxyhost、proxyport、および rulespath プロパティを使用して、各 JRun サーバーの local.properties ファイルを検索します。詳細については、170 ページの「単純な分散環境での JRun の実行」を参照してください。

複雑な分散環境で JRun をインストールする場合、JRun アーキテクチャとポートの使用についてよく理解しておくと役立ちます。詳細については、160 ページの「 JRun ポートについて」を参照してください。

Web サーバーの接続

複数の JRun サーバーを 1 つの Web サーバーに接続するには

1. 100 ページの「 JRun サーバーの追加」で説明されているように、新規の JRun サーバーをそれぞれ作成します。

   必ず、jvms.properties ファイルに新規 JRun サーバーのルート ディレクトリを手作業で加えて更新します。

2. 各 JRun サーバーのコネクタ ウィザードを実行し、同一の Web サーバーに接続する JRun サーバーをそれぞれ選択します。コネクタ ウィザードでは、必ず一意の JRun サーバー コネクタ ポートを入力します。この作業を行わないと、バインディング例外が発生します。コネクタ ウィザード使用の詳細については、42 ページの「構成の概要」を参照してください。

   コネクタ ウィザードを実行すると、接続モジュールがインストールされ、JRun サーバーの local.properties ファイルと Web サーバーのコンフィギュレーション ファイルが更新されます。

3. 各 JRun サーバーのコネクタ ウィザードを実行した後で、Web サーバーのコンフィギュレーション ファイルの jvmlist プロパティに新規 JRun サーバーの名前を追加

する必要があります (たとえば、コンフィギュレーション ファイルは、IIS の場合は jrun.ini、iPlanet/Netscape の場合は obj.conf、Apache の場合は httpd.conf です)。各サーバー名をカンマで区切ります。次に例を示します。

JRunConfig jvmlist default,production,qa

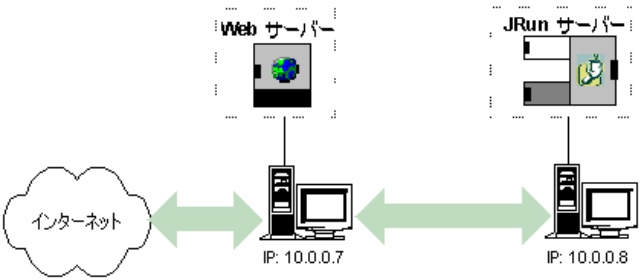
jvmlist プロパティにサーバーが複数存在すると、JRun が Web サーバーのコンフィギュレーション ファイル内の rulespath、proxyhost、および proxyport プロパティを無視することに注意してください。その代わり、JRun は、168 ページの「設定の詳細」で説明されているように、各サーバーの local.properties ファイルからこの情報を取得します。

4. Web サーバーを再起動します。
5. JRun サーバーを再起動します。

単純な分散環境での JRun の実行

単純な分散型構成では、1 台のコンピュータを Web サーバー専用とし、もう 1 台を JRun サーバー専用とします。これは、処理の負荷を複数のコンピュータに分散する一般的な方法です。

分散環境で JRun を実行する事例を以下の図に示します。



単純な分散環境でのインストール

単純な一対一の分散環境で JRun を構成する場合には、以下の手順をお勧めします。

JRun を一対一の分散環境にインストールするには

1. Web サーバー マシンと JRun 専用コンピュータの両方に JRun をインストールします。Web サーバー マシンにも JRun をインストールするのは、コネクタ ウィザードを実行できるようにするためです。
2. Web サーバー マシンで JRun コネクタ ウィザードを実行し、ローカル Web サーバーに接続します。コネクタ ウィザードを実行すると、Web サーバー フィルタお

よび Web サーバーのコンフィギュレーション ファイルがインストールされます。JRun のこのインスタンスは、サーブレット、JSP、または EJB の処理には使用しません。

a. コネクタ ウィザードで、ローカル Web サーバー、ローカル Web サーバーのスクリプトまたはコンフィギュレーション ディレクトリを選択します。

b. コネクタ ウィザードの [JRun サーバー IP アドレス] フィールドに、JRun を実行するコンピュータの IP アドレスを入力します。これによって、Web サーバーのコンフィギュレーション ファイル内の proxyhost プロパティが設定されます。

c. コネクタ ウィザードの [JRun サーバー コネクタ ポート] フィールドに、JRun コンピュータの JCP ポート番号を入力します。これによって、Web サーバーのコンフィギュレーション ファイル内の proxyport プロパティが設定されます。proxyport プロパティは、JRun が Web サーバーからの要求を受信するポートを参照します。

3. JRun マシン上で、local.properties ファイルを次のように変更します。

a. JRun コンピュータに、コネクタ ウィザードを実行したかのように認識させる必要があります。

- ファイルの最後に次の行を追加します。

```
ranConnector=yes
```

- jcp を servlet.services プロパティに追加し、JRun コネクタ プロキシを、実行するサービスの一覧に加えます。

```
servlet.services=jndi,jdbc,web,url,{servlet.webapps},jcp
```

b. JCP バインド アドレスを JRun サーバーのコンピュータの IP アドレスに設定します。次に例を示します。

```
jcp.endpoint.main.bindaddress=10.0.0.8
```

bindaddress はループバック アドレス (127.0.0.1) に設定できません。設定すると、Web サーバーは自身の Web サーバー上の JRun サーバーに接続しようとします。両方のサーバーが同一のコンピュータ上にある場合は、127.0.0.1 に設定できます。

4. Web サーバー マシン上で、Web サーバーのコンフィギュレーション ファイルを次のように変更します (このコンフィギュレーション ファイルは、IIS の場合は jrun.ini、iPlanet/Netscape の場合は obj.conf、Apache の場合は httpd.conf です)。

- Web サーバーのコンフィギュレーション ファイル内の rulespath プロパティを、JRun の local.properties ファイルを参照するように設定します。次に例を示します。

```
rulespath="Z:/Program Files/Allaire/JRun/servers/default/
    local.properties"
```

- jvmlist および jrun.rootdir プロパティをコメント化します。

```
#jrun.rootdir=C:/PROGRA~1/Allaire/JRun
```

```
#jvmlist=default
```

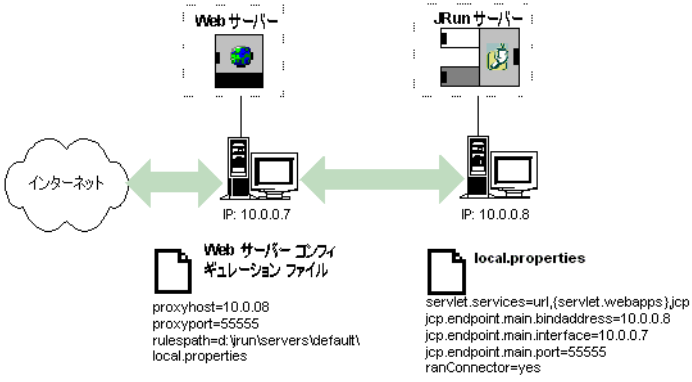
5. 2 台のコンピュータが異なるファイル システムを使用している場合 (1 台が Windows NT、もう 1 台が Linux を実行している場合など) は、local.properties ファイルを Web サーバー マシンにコピーして、Web サーバーのコンフィギュレーション ファイル内で、コピーしたローカル ファイルを参照する必要があります。

   メモ Web サーバー マシン上の local.properties ファイルは、rules セクションのサーブレット マッピング用にのみ使用されます。これらのマッピングがないと、JRun コンピュータに対する要求のマッピングが正しく行われません。したがって、ルール マッピングを変更しない限り、このファイルを継続的に更新する必要はありません。

6. 分散環境で、コンピュータ間のファイル システムが異なる場合に JSP を使用するときは、pathtrans プロパティを編集する必要があります。詳細については、175 ページの「分散環境での JSP の使用」を参照してください。
7. Web サーバー マシン上の JRun サーバーを使用不可にします (オプション)。Web サーバー マシンから JRun をアンインストールしないでください。また、コネクタを削除しないでください。
8. JRun サーバーを再起動します。
9. Web サーバーを再起動します。

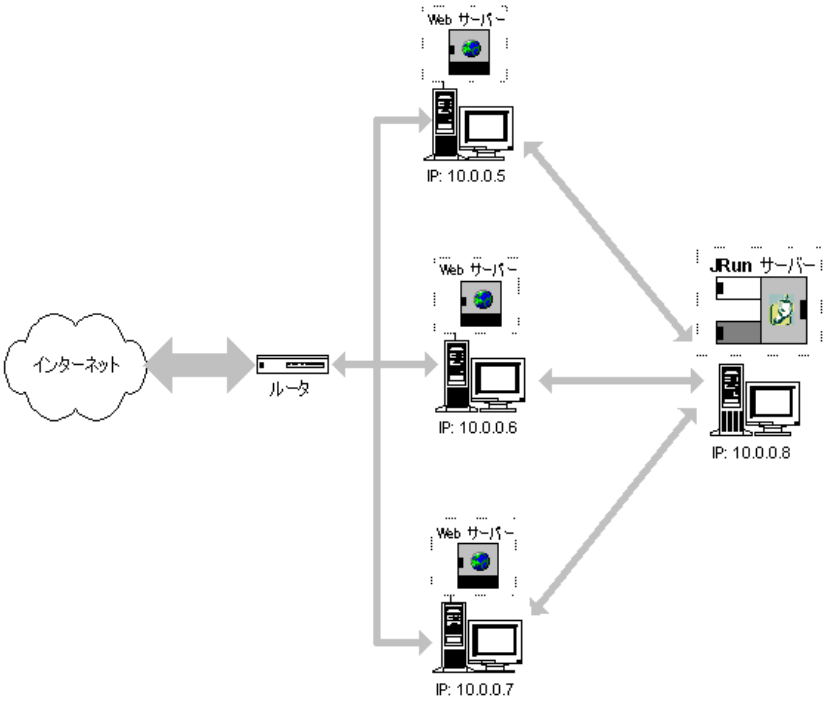
単純な分散環境の例

次の図は、2 台のコンピュータ上での 1 つの Web サーバーと 1 つの JRun サーバーの接続に必要な設定を示しています。ここでは、2 台のコンピュータが同一のファイル システム上にあると想定しています。JRun コンピュータは Web サーバー マシンの D ドライブにマッピングされています。このため、rulespath プロパティは D ドライブ上の local.properties ファイルを参照しています。



複雑な分散環境での JRun の実行

もう 1 つの一般的な事例としては、顧客が複数の Web サーバー マシンを使用している環境で、JRun サーバー専用のコンピュータをもう 1 台追加する場合が考えられます。分散環境で JRun を実行する事例を以下の図に示します。



複雑な分散型インストール

JRun サーバーを複数の Web サーバーに接続するには

1. 170 ページの「単純な分散環境での JRun の実行」の説明に従って、各 Web サーバー マシンを、JRun と通信するように設定します。
2. JRun コンピュータ上の local.properties の jcp.endpoint.main.interface プロパティを、* または JRun と接続する Web サーバーの IP アドレスの一覧 (カンマ区切り) に設定します。JRun は、このプロパティと一致しないコンピュータからの要求を受け付けません。

   上記の図では、以下のようになります。

   jcp.endpoint.main.interface=10.0.0.5,10.0.0.6,10.0.0.7

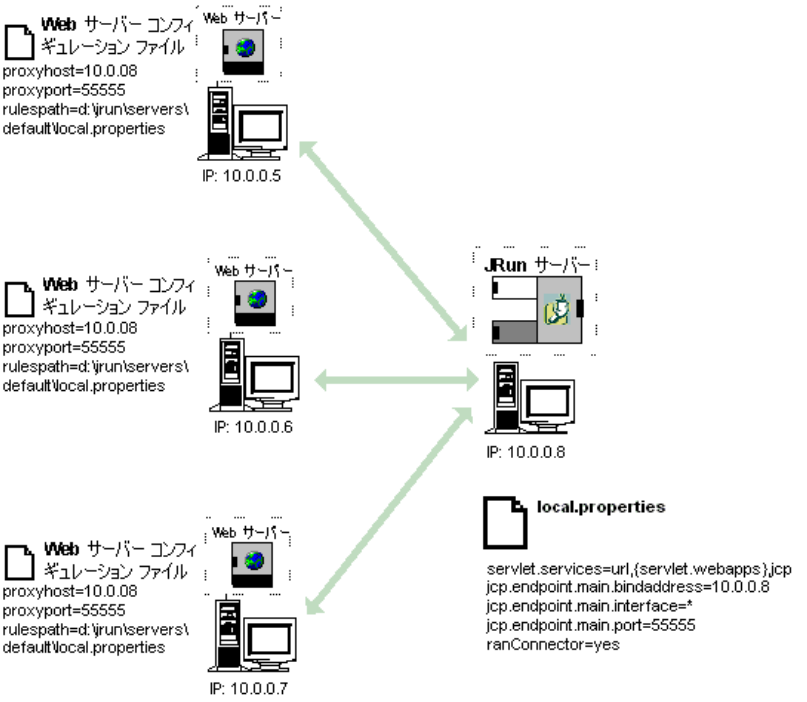
3. 各 Web サーバー コンフィギュレーション ファイルの proxyhost が、JRun を実行しているコンピュータの IP アドレスに設定されていることを確認します。また、これらのコンピュータの proxyport 設定は、JRun の local.properties ファイルの jcp.endpoint.main.port と同一である必要があります。
4. 複数のコンピュータが異なるファイル システムを使用している場合 (Web サーバーで IIS/Windows NT、アプリケーションの処理用に JRun/Linux を実行している場合など) は、local.properties ファイルを Web サーバー マシンにコピーし、Web サーバーのコンフィギュレーション ファイル内で、コピーしたローカル ファイルを参照する必要があります。

   メモ Web サーバー マシン上の local.properties ファイルは、rules セクションのサーブレット マッピング用にのみ使用されます。これらのマッピングがないと、JRun コンピュータに対する要求のマッピングが正しく行われません。したがって、ルール マッピングを変更しない限り、このファイルを継続的に更新する必要はありません。
5. 分散環境で、コンピュータ間のファイル システムが異なる場合に JSP を使用するときは、pathtrans プロパティを編集する必要があります。詳細については、175 ページの「分散環境での JSP の使用」を参照してください。
6. Web サーバー マシン上の JRun サーバーを使用不可にします (オプション)。Web サーバー マシンから JRun をアンインストールしないでください。また、コネクタを削除しないでください。
7. JRun サーバーを再起動します。
8. Web サーバーを再起動します。

複雑な分散環境の例

次の図は、別個のコンピュータ上での複数の Web サーバーと 1 つの JRun サーバーの接続に必要な設定を示しています。これは、複数のコンピュータが同一のファイル システムを使用しており、JRun サーバーの local.properties ファイルにアクセスできるように、ディレクトリ /jrun が各 Web サーバー マシンにマップされていることを想定しています。どのコンピュータも JRun サーバーに接続できるように、jcp.endpoint.main.interface が * に設定されていることに注意してください。これを 10.0.0.5、10.0.0.6、10.0.0.7 に設定することも簡単にできます。

以下の図は、Web サーバーのコンフィギュレーション ファイルおよび local.properties ファイルで必要な設定を示しています。



分散環境での JSP の使用

複数のコンピュータで異なるファイル システムを使用することがあります。ユーザが JSP の URL を要求したとき、Web サーバーのリアル パスは、実際にページを取得する " 本当の " パスではありません。Web サーバーの " リアル " パスを、JRun が要求への応答で使用するパスに変換する必要があります。

JRun の pathtrans プロパティを使用すると、要求を実際のパスにマッピング (パスを変換) することができます。global.properties ファイルのこれらのプロパティの既定値は以下のとおりです。

```
### pathtrans.properties
webapp.pathcount=0
webapp.path0.from=/virtualdir
webapp.path0.to=/realdir
```

この設定は、JRun サーバーと Web サーバーが同一のコンピュータ上にある場合に有効となります。ただし、分散環境で JRun サーバーが異なるコンピュータ上にあり、2 つ

のコンピュータが異なるファイル システムを使用している場合は、ユーザがマッピングを指定する必要があります。

pathtrans プロパティの編集

pathtrans プロパティを編集するには

1. JRun の global.properties の pathtrans セクションを JRun サーバーの local.properties にコピーします。ローカル Web サーバーの local.properties はルール マッピング用にのみ使用されます。この変更は、JRun をホストするコンピュータ上で行う必要があります。Web サーバー マシンの local.properties のコピーは変更しません。
2. webapp.pathcount をパス変換の回数 n だけ増分します。
3. webapp.path[*n*].from ディレクトリが Web サーバーを参照し、webapp.path[*n*].to ディレクトリが JRun コンピュータを参照するようにします。

   必要なすべての JSP を JRun コンピュータ上の webapp.path[*n*].to ディレクトリに格納する必要があります。

pathtrans の例

以下の例では、Linux 上の Apache ドキュメント ルート ディレクトリに JSP があります。pathtrans プロパティを使用して JRun に指示すると、/jsps 内の JSP に対する要求をすべて Apache /htdocs/jsps ディレクトリに再マッピングできます。以下の例ではディレクトリ名が同じですが、異なっていても問題はありません。

```
### pathtrans.properties
webapp.pathcount=1
webapp.path0.from=C:\\Inetpub\\wwwroot2\\jsps
webapp.path0.to=/usr/local/apache/htdocs/jsps
```

分散 JRun システムの保護

分散環境でシステムのセキュリティを備えるにあたって考慮するべきことは多々ありますが、このセクションでは JRun に関して行うべきことを説明します。

- **JWS のシャットダウン** JRun のデフォルトのインストールでは、dmin JRun サーバーを含め、それぞれの JRun サーバーに Web サーバーが設定されます。JWS のオフについては 177 ページの「JWS のオフ」を参照してください。
- **ホスト ベースの認証の設定** JRunでは、JRunサーバーと外部Webサーバー間の通信をハイジャックから守るための基本的なメカニズムが提供されています。この設定については、177 ページの「コネクタ用のホストベース認証」を参照してください。

JWS のオフ

JRun のインストールにより、default と admin の 2 つの JRun サーバー、および default と admin の 2 つの all-Java JRun Web Servers (JWS) が作成されます。デフォルトでは、これらの Web サーバーは、ポート 8000 およびポート 8100 のそれぞれで受信しようとします。たいていのシステム管理者は、ファイアウォールで入力ポートのアクセスを制限していますが、使用していないサービスをオフにしておくことに越したことはありません。このセクションでは、JRun サーバー用の JWS をオフにする方法を説明します。

使用されていない JRun サーバー用の JWS をオフにするには

1. JRun サーバーの local.properties ファイルを開きます。このファイルは、<*jrun_root_dir*>/servers/<*server_name*>/local.properties にあるはずです。
2. web サービスを servlet.services プロパティから削除します。たとえば、次のようになります。

   ```
   # was: servlet.services=jndi,jdbc,web,mail,url,{servlet.webapps},jcp
   servlet.services=jndi,jdbc,mail,url,{servlet.webapps},jcp
   ```

3. JRun サーバーを再起動します。

コネクタ用のホストベース認証

JRun を実行するマシンと Web サーバーを実行している別のマシン間で接続が作成されると、ネットワーク上の別の場所から許可されていないユーザーのアクセスを防止しておきたいものです。JRun では、JRun コネクタ用にホスト ベースの認証が提供されます。つまり、定義されたアドレスからのホストのみ、JRun サーバーに要求を送信するようにできます。

JMC の [外部 Web サーバー] パネルを使用して、IP アドレスが特定の JRun サーバーと交信できるように指定できます。現在のところ、SSL または他の暗号化技術を使用して外部 Web サーバーと JRun サーバー間のトラフィックを保護することはできません。

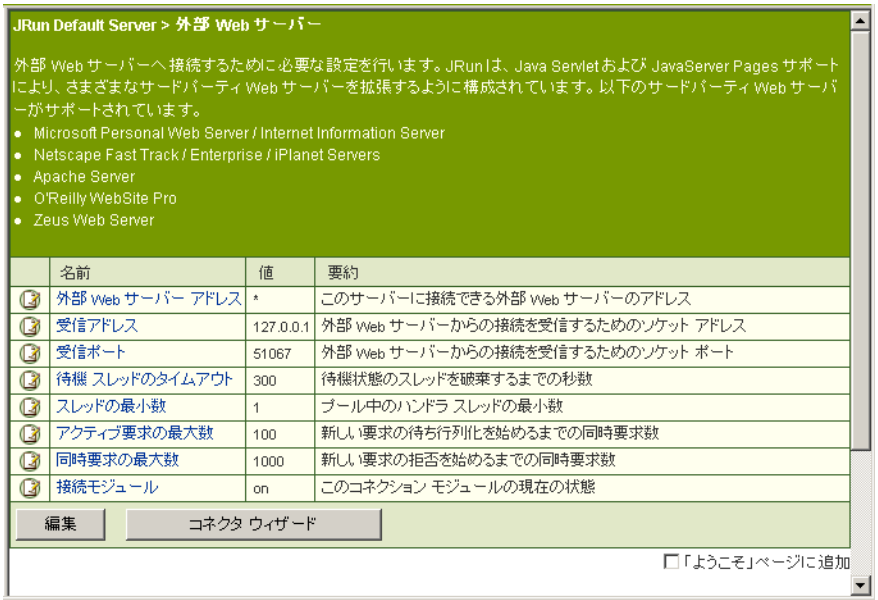
メモ すべての IP アドレスから要求を受信するのが JRun サーバーのデフォルトの設定です。

Web サーバーと JRun 間の接続をロックするには

1. JRun 管理コンソールの左ペインで、[*machine_name*] > [*JRun_server_name*] > [外部 Web サーバー] を選択します。

   メモ コネクタ ウィザードを実行して、この JRun サーバーを外部 Web サーバーに接続していない場合、ここで接続するように指示があります。

   [外部 Web サーバー] パネルが表示されます。



2. 右ペインで、[**外部サーバー アドレス**] フィールドをクリックします。[外部 Web サーバー編集] ウィンドウが表示されます。
3. カンマで区切った IP アドレスを入力します。これらのマシン上にある Web サーバーのみ JRun サーバーに要求を送信できます。すべての Web サーバーから JRun に要求を送信できるようにするには、* を入力します。
4. 変更を適用するには、[**更新**] をクリックします。
5. JRun サーバーを再起動します。

メモ ホスト ベースの認証による保護は、IP スプールまたは 他の *man-in-the-middle* テクニックを使用してのアタックを防止することはできません。JRun の後のバージョンでは、TLS/SSL がサポートされる予定です。

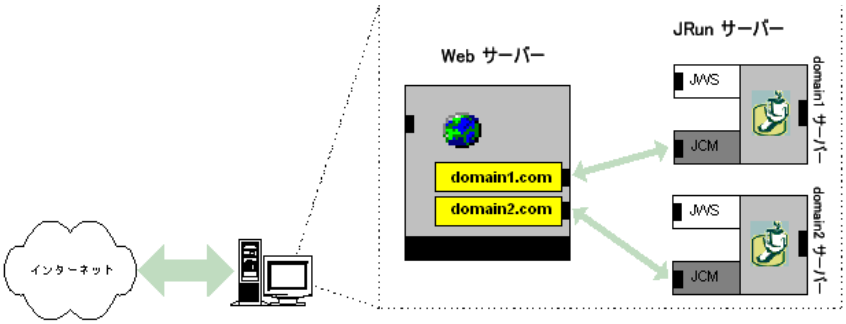
JRun でのマルチ ホスティング

1 台のコンピュータ上に複数の仮想ホストを設定する場合、JRun をこれらのホストに接続するには、特別な手順を実行する必要があります。ほとんどの環境では、以下の理由により、各仮想ホストごとに別個の JRun サーバーを作成します。

- 異なる仮想ホストの Web アプリケーションを別個の JVM によって処理できるようにするため。
- 1 つの JRun サーバーが停止しても、すべての Web アプリケーションに障害が起こらないようにするため。

- 開発者や顧客が、同じ名前を持つ Web アプリケーションを作成しないようにするため (ISP の場合)。

次の図は、1 台のコンピュータが 1 つの Web サーバーを実行する、単純なマルチホスティング構成を示しています。ここで、Web サーバーは 2 つの仮想ホストのホストであり、これらの仮想ホストはそれぞれ、独自の JRun サーバーを持っています。



このセクションでは、最もよく使用されている Web サーバーでマルチ ホスティングを行う方法について説明します。

Apache でのマルチ ホスティング

Apache 仮想ホストを設定し、各ホストに独自の JRun サーバーを作成するには、Apache のコンフィギュレーション ファイル (httpd.conf) の各 VirtualHost ディレクティブ内に JRun コンフィギュレーション ブロックを含めます。

次の一覧は、VirtualHost ディレクティブ内の JRun コンフィギュレーション情報の例を示しています。LoadModule は、VirtualHost ディレクティブの外部に記述する必要があります。これは、LoadModule ステートメントの参照は、グローバル レベルで 1 回だけ行うようにする必要があるためです。

```
LoadModule jrun_module136 "/opt/JRun/connectors/apache/intel-linux/mod_jrun.so"
<VirtualHost 127.0.0.1>
    ServerAdmin webmaster@localhost
    DocumentRoot /usr/local/apache/htdocs/localhost
    ServerName new-host
    ErrorLog logs/new-host-error_log
    CustomLog logs/new-host-access_log common

  # JRun Settings
    <IfModule mod_jrun.c>
        JRunConfig jrun.rootdir "/opt/JRun/bin/.."
        JRunConfig jvmlist new-host
        JRunConfig Verbose false
        JRunConfig ProxyHost 127.0.0.1
        JRunConfig ProxyPort 51001
```

```
        JRunConfig Mappings "/opt/JRun/servers/new-host/local.properties"
    </IfModule>
</VirtualHost>
```

LoadModule ステートメントが必要となるのは、JRun を Dynamic Shared Objects (DSO) モジュールとして使用する場合に限られます。JRun をスタティック モジュールとして設定した場合は、LoadModule ステートメントは必要ありません。

また、httpd.conf ファイルの編集に加えて、各仮想ホストの JRun サーバーを、一意のプロキシ ポートを使用するように設定する必要があります。

Apache でマルチ ホスティングを行うには

1. 新規 Apache 仮想ホストを作成します。
2. 100 ページの「JRun サーバーの追加」の説明に従って、新規 JRun サーバーを作成します。手順の説明上、新規 JRun サーバーを new-host と呼びます。

   各仮想ホストは、VirtualHost ディレクティブ内の ProxyPort プロパティと一致する、一意の JCP ポートを持つ必要があります。JRun ポートの使用の詳細については、160 ページの「JRun ポートについて」を参照してください。
3. local.properties の servlet.services プロパティから web サービスを削除して、new-host JWS を無効にします。

   ```
   # was:servlet.services=jndi,jdbc,{servlet.webapps},jcp,web
   servlet.services=jndi,jdbc,{servlet.webapps},jcp
   ```
4. admin JRun サーバーを再起動します。次に例を示します。

   ```
   % jrun -restart admin
   ```
5. Apache Web サーバーを再起動します。
6. new-host JRun サーバーを起動します。次に例を示します。

   ```
   % jrun -start new-host
   ```

IIS でのマルチ ホスティング

複数の仮想サーバーを設定できるのは IIS 4.0 だけです。IIS 3.0 および PWS は、複数の仮想サーバーをサポートしていません。

IIS でマルチ ホスティングを行うには

1. 新規仮想ホストのディレクトリを作成します。たとえば、c:\inetpub\newhost_root とします。
2. newhost Web サイトの新規 scripts ディレクトリを作成します。たとえば、c:\inetpub\newhost_root\scripts とします。
3. 『JRun セットアップ ガイド』の説明に従って、新規 JRun サーバーを作成します。手順の説明上、新規 JRun サーバーを new-host と呼びます。

各仮想ホストは、VirtualHost ディレクティブ内の ProxyPort プロパティと一致する、一意の JCP ポートを持つ必要があります。JRun ポートの使用の詳細については、160 ページの「 JRun ポートについて」を参照してください。

4. local.properties の servlet.services プロパティから web サービスを削除して、new-host JWS を無効にします。

```
# was:servlet.services=jndi,jdbc,{servlet.webapps},jcp,web
servlet.services=jndi,jdbc,{servlet.webapps},jcp
```

5. admin JRun サーバーを再起動します。次に例を示します。

```
% jrun -restart admin
```

6. MMC で、コンピュータ名を右クリックし、[**新規**] > [Web **サイト**] を選択して新規 Web サイトを作成します。たとえば、NewHost_Site を作成します。

7. JRun サーバーの scripts ディレクトリを参照する仮想 scripts ディレクトリ (たとえば、c:\inetpub\newhost_root\scripts) を、Web サイトを右クリックし、[**新規**] > [**仮想ディレクトリ**] を選択して作成します。エイリアスを newhost_scripts に設定します。この新規ディレクトリに実行権限を与えます。

8. JMC で、Web サーバーに対してコネクタ ウィザードを実行します。

   コネクタ ウィザードの手順 2 で、必ず接続ごとにコネクタ ウィザードに一意の JRun **サーバー コネクタ ポート** を入力してください。

   手順 3 で以下の操作を行います。

   - IIS Scripts **ディレクトリ** が新規の scripts ディレクトリを参照するようにします (たとえば、c:\inetpub\newhost_root\scripts)。
   - [**グローバル フィルタとしてのインストール**] チェック ボックスを *オフ* にして、JRun フィルタを仮想サイトの /scripts ディレクトリにのみ適用します。

   マルチ ホスティングをセットアップする際は、JRun フィルタが既にグローバルにインストールされていないことを確認してください。グローバルにインストールされている場合は、metaset ユーティリティを使用してアンインストールできます。metaset 使用の詳細については、『拡張設定ガイド』の関連するセクションを参照してください。

9. World Wide Web Publishing サービスを再起動します。

10. newhost JRun サーバーを起動します。次に例を示します。

```
% jrun -start newhost
```

Netscape でのマルチ ホスティング

Netscape では、仮想ホストごとに新規 Web サーバーのインスタンスが必要です。

Netscape でマルチ ホスティングを行うには

1. 仮想ホストごとに新規 Web サーバーのインスタンスを作成します。

2. 94 ページの「JRun サーバーの設定」の説明に従って、新規 JRun サーバーを作成します。手順の説明上、新規 JRun サーバーを new-host と呼びます。
3. local.properties の servlet.services プロパティから web サービスを削除して、new-host JWS を無効にします。

```
# was:servlet.services=jndi,jdbc,{servlet.webapps},jcp,web
servlet.services=jndi,jdbc,{servlet.webapps},jcp
```

4. JMC でコネクタ ウィザードを実行し、新規 JRun サーバーを新規の Netscape Web サーバー インスタンスに接続します。

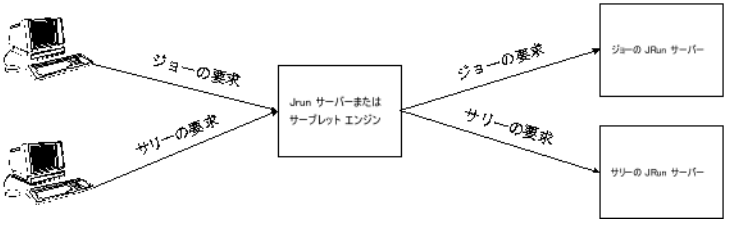
   コネクタ ウィザードの手順 2 で、必ず接続ごとにコネクタ ウィザードに一意の **JRun サーバー コネクタ ポート** を入力してください。
5. Netscape サーバーを再起動します。
6. admin JRun サーバーを再起動します。次に例を示します。

   % jrun -restart admin
7. fredserver JRun サーバーを起動します。次に例を示します。

   % jrun -start new-host

要求のチェーン化

JRunConnector サーブレットを使用すると、サーブレット エンジン インスタンスから別のサーブレット エンジン インスタンスへ渡される要求をチェーン化することができます。サーブレット エンジンを使用すると、JRun サーバーから別の JRun サーバーへ、または 非 JRun サーブレット エンジンから JRun サーバーへと要求を渡すことができます。



この機能は、複数の JRun サーバーにアクセスする中心点が必要な場合に役立ちます。

JRunConnector サーブレットを使用するには、次の手順を実行します。

- ターゲット JRun サーバーのホストおよびポート情報を確認します。
- 呼び出す側のサーバーの設定を定義します。
- ターゲット サーバーの設定を定義します。

ターゲット サーバーの設定の確認

まず、ターゲット JRun サーバーの設定を取得します。取得する必要のある設定には以下が含まれます。

- proxyhost: ターゲット JRun サーバーの IP アドレスです。
- proxyport: ターゲット JRun サーバー上で動作する JCP プロセスのポート番号です。これは、コネクタ ウィザードの実行時に指定したポート番号です。
- send-path-info (オプション): ブール値です。JRunConnector がすべての URI 情報を渡す (false) か、URI のパス情報のみを渡す (true) かを指定します。既定値は false です。URI およびパス情報の詳細については、『JRun によるアプリケーションの開発』を参照してください。

呼び出す側のサーバーの設定の定義

呼び出す側のサーバー (JRun 以外のサーブレット エンジンの場合もあります) 上で、以下の設定を作成します。

- JRunConnector のサーブレット定義。この定義には、proxyhost、proxyport、および (オプションの) send-path-info の初期化引数を含める必要があります。たとえば、次のように定義します。

```
Name:gotohr
Class name:allaire.jrun.connector.JRunConnector
Display name:Connect to HR server
Init arguments:
  proxyport=51001
  proxyhost=200.10.5.30
```

- サーブレット定義を呼び出す URL マッピング。たとえば、次のように定義します。

```
Virtual path/extension:/hr
Servlet invoked:gotohr
```

ターゲット JRun サーバーは、proxyhost/proxyport の組み合わせで一意に識別されます。ターゲット JRun サーバーごとに、サーブレット定義 /URL マッピングのペアを別個に指定します。

ターゲット JRun サーバーの設定の定義

前の例では、呼び出す側のサーバー上のサーブレット URL マッピングとして /hr を使用しました。ターゲット JRun サーバーは、次のような複数の方法でこのマッピングを処理できます。

- send-path-info が false である場合は、ターゲット サーバー上で次のいずれかを行います。

- 呼び出す側のサーバー上の URL マッピングと同じ名前で、Web アプリケーションを定義します。たとえば、呼び出す側のサーバーの URL マッピングが /hr である場合、ターゲット サーバーの Web アプリケーション マッピングは /hr とします。
- 呼び出す側のサーバーが使用する URL マッピングを含むマルチ パート URL マッピングを定義します。たとえば、呼び出す側のサーバー上のリクエスト URI に URL マッピングとして /hr が、パス情報として /ShowEmployees が含まれている場合、ターゲット サーバーでは URL マッピングを /hr/ShowEmployees と定義します。

- send-path-info が true である場合は、パス情報に含まれている情報で、ターゲット JRun サーバー上のサーブレットを正しく呼び出せることを確認する必要があります。

補足情報

JRunConnector はサーブレットであるため、RequestDispatcher を使用して別のサーブレットから呼び出すことができます。たとえば、呼び出す側のサーバー上でセキュリティや帯域幅の分析などの処理を行ってから、適切なターゲット JRun サーバーに転送する場合に利用できます。

カスタム コネクタの作成

コネクタは、JRun が Web サーバーとの通信の確立に使用するモジュールです。コネクタは、Web サーバーにインストールされると、すべてのサーブレットおよび JSP 要求を JRun に転送します。多くの Web サーバーでは、JMC を使用してコネクタ ウィザードを実行すれば、コネクタをインストールできます。

JRun には、カスタム Web サーバーやその他の特別なプラットフォームに使用するため、コネクタのソース コードが含まれています。基本的な使用に関する説明については、*<JRun のルート ディレクトリ >*/connectors/src/readme を参照してください。*<JRun のルート ディレクトリ >*/connectors/src ディレクトリには、サーバー独立モジュール、Apache、ISAPI、および NSAPI に対するサブディレクトリが含まれます。JRun は、サポートされるプラットフォームに適切なソース コード、make ファイル、ヘッダー ファイル、プロジェクト ファイル、および関連ファイルを提供します。

次の表は、*<JRun のルート ディレクトリ>*/connectors/src のサブディレクトリの概要を示します。

オープン ソース コネクタのディレクトリ	
ディレクトリ	**目次**
apache	Apache 用のプロジェクト ファイル、関連ファイル、ヘッダー ファイル、および make ファイルです。Apache 固有のソース コードは mod_jrun.c にあります。
connector	次のファイルを含みます。 • jrun_property.c: プロパティ ファイルを読み取るためのユーティリティ コードです。 • jrun_property.h: プロパティのデータ構造の定義です。 • jrun_proxy.c:JRun プロキシ プロトコルのインプリメンテーションです。 • jrun_proxy.h: コネクタ プロトコルの定義です。 • platform.c: 各種システム呼び出しの既定インプリメンテーションです。
isapi	ISAPI 用のプロジェクト ファイル、関連ファイル、ヘッダー ファイル、および make ファイルです。ISAPI 固有のソース コードは次のファイルに含まれています。 • extension.cpp: ISAPI 拡張機能のエントリ ポイントです。 • filter.cpp: ISAPI フィルタのエントリ ポイントです。 • interface.cpp: jrun_proxy.c によって呼び出されるメソッドです。
nsapi	NSAPI 用のプロジェクト ファイル、関連ファイル、ヘッダー ファイル、および make ファイルです。NSAPI 固有のソース コードは次のファイルに含まれています。 • extension.c: NSAPI のエントリ ポイントです。 • interface.c: jrun_proxy.c によって呼び出されるメソッドです。 • nwmain.c: Netware 固有のコードです。

次に、Apache および Netscape 用コネクタのコンパイル手順について説明します。

Apache 用のコネクタのコンパイル

サポートされていないプラットフォームで使用するため、カスタム Apache コネクタを作成できます。さらに、カスタム コネクタをコンパイルして、Apache に静的にリンクすることができます。

Apache 用のカスタム コネクタを作成するには

1. Apache のディレクトリに移動します。

   cd *apacheinstalldirectory*

2. src/modules/jrun ディレクトリを作成します。

   mkdir src/modules/jrun

3. src/modules/jrun ディレクトリに移動します。

   cd src/modules/jrun

4. 次のように、ファイルをコピーします。

   cp *jruninstalldirectory*/connectors/src/apache/*.c .
   cp *jruninstalldirectory*/connectors/src/apache/*.h .
   cp *jruninstalldirectory*/connectors/src/apache/Makefile.libdir .
   cp *jruninstalldirectory*/connectors/src/apache/Makefile.tmpl .
   cp *jruninstalldirectory*/connectors/src/connector/*.c .
   cp *jruninstalldirectory*/connectors/src/connector/*.h .

5. Apache のディレクトリに移動します。

   cd *apacheinstalldirectory*

6. configure ユーティリティを実行します。このユーティリティは、追加のサイト固有の引数を必要とします。

   ./configure --activate-module=src/modules/jrun/libjrun.a

7. 次のように、make コマンドを実行します。

   make
   make install

Netscape 用のコネクタのコンパイル

Netscape 用のカスタム コネクタを作成するには

1. *<JRun のルート ディレクトリ>*/connectors/src/nsapi ディレクトリに移動します。

   cd *<JRun のルート ディレクトリ >*/connectors/src/nsapi

2. 次のように、make ファイルを編集します。

   - 適切な INC*xy* = ../Servers/Netscape/*x.y*/include 行を変更し、正しい include ディレクトリ (たとえば、*Netscape のインストール ディレクトリ* /plugins/include) を参照するようにします。

- JRUN_LIBS 定義を変更し、使用する Netscape のバージョンのみが含まれるようにします (たとえば、JRUN_LIBS = libjrun_nsapi35.so)。

3. make ファイルを実行します。
   make
4. 出力を格納するためのディレクトリを作成します。
   mkdir ../../nsapi/custom
5. 出力をコピーします。
   cp –f libjrun_nsapi*.so ../../nsapi/custom

第 5 章

プロパティ ファイル

JRun では、プロパティ ファイルを使用して初期化および構成を行います。これらのファイルには、JRun で使用する構成可能な設定値の大部分が格納されています。

一般的な構成タスクのほとんどは、プロパティ ファイルへの書き込みを行う JRun 管理コンソール で実行することができますが、プロパティ ファイルを編集すると、JRun で使用可能な内部変数をよく理解できるだけでなく、JRun の設定の一部をより詳細に制御することができます。

この章では、プロパティ ファイルが持つ階層の性質と併せて、アクセスが容易であるというプロパティ ファイルの特徴を生かした JRun の構成方法について説明します。

目次

プロパティ ファイルの概要

プロパティ ファイルには、JRun の構成情報の大部分が格納されています。JRun は、起動時にプロパティ ファイル内の値を読み取り、再起動されるまで、それらの値をメモリ内に保持します。.JRun は、最初にローカル プロパティファイルを読み取り、次に残りのプロパティ ファイルを読み取ります。

以下の表は、プロパティ ファイルおよび JRun ディレクトリ ツリー内の各ファイルの格納場所を示したものです。

プロパティ ファイル		
ファイル名	**格納場所**	**説明**
descriptions.properties	\lib\	プロパティ ファイル内にある多くのオブジェクトの記述が格納されています。
global.properties	\lib\	最上位レベルの値が格納されています。既定では、ローカル プロパティ ファイルにはグローバル プロパティの値が格納されます。
local.properties	\servers\ <server_name>\	JRun サーバーごとのプロパティと、そのサーバー上のすべてのアプリケーションのプロパティが格納されます。グローバル プロパティは、このプロパティによって無効になります。
webapp.properties	\servers\ <server_name>\ <app_name>\WEB-INF\	必要に応じて、特定の Web アプリケーションのプロパティも格納されます。ローカル プロパティおよびグローバル プロパティは、このプロパティによって無効になります。JRun でこのファイルが作成されるのは、ユーザが JMC を使ってアプリケーション特有の設定値を設定した場合のみです。
jvms.properties	\lib\	該当する JRun のインストール先にあるすべての JRun サーバーの名前と絶対パスの一覧がリストされます。
pass.properties	\lib\	すべてのユーザについて、暗号化されたパスワードとアクセス許可の設定が格納されます。

プロパティ ファイル		
ファイル名	**格納場所**	**説明**
serial_number.properties	\lib\	JRun のシリアル番号を格納します。
users.properties	\lib\	ユーザと各ユーザのパスワードの一覧のほかに、グループへのユーザの割り当ておよびロールへのグループの割り当ての一覧が格納されます。
ejipt.properties	\lib\	EJB エンジンのホスト情報を格納します。
deploy.properties	\servers\<server_name>\deploy	サーバー レベルのプロパティを格納します。この情報は、Deploy ツールによって、使用する bean、ホスト名、データ ソース、および最大接続数を決定するために使用されます。
<bean_name>.properties	<Bean のインタフェースおよび実装ディレクトリ >	セキュリティおよび bean の説明情報を格納します。 bean の XML 記述子ファイルにも同じ情報を格納できます。
default.properties	<Bean のインタフェースおよび実装ディレクトリ >	この bean の内容に関する情報を格納します。

プロパティ ファイルのリロード

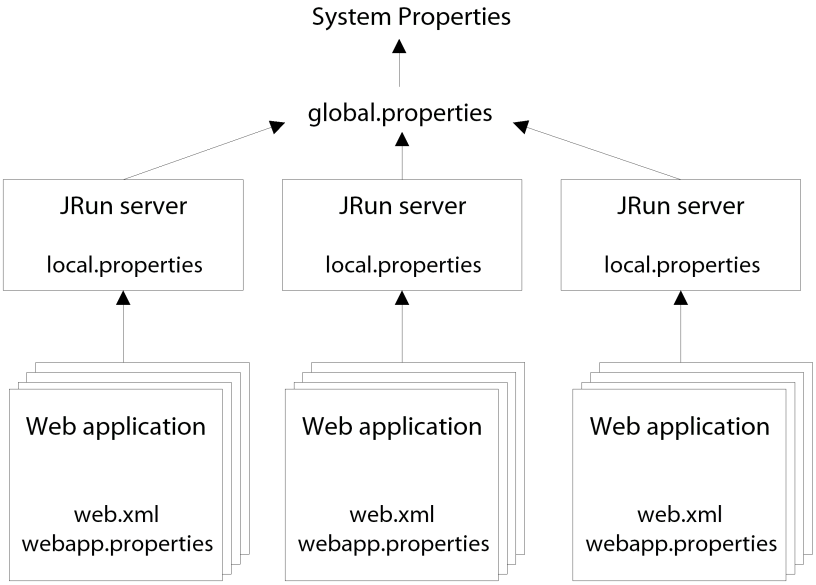
JRun プロパティ ファイルを変更すると、JRun サーバーを再起動する必要があります。例外は、`users.properties` ファイルで、JRun により自動的に再起動され、Web アプリケーションのユーザの追加、変更、削除が、リアルタイムに行われます。

プロパティ ファイルの階層について

JRun のプロパティ ファイルは、システム、グローバル、ローカル、およびアプリケーションという 4 層構造になっています。グローバル レベルでの設定値は、ローカル レベルでの設定値によって上書きされる場合があります。さらに、ローカル レベルでの設

定値は、アプリケーション レベルでの設定値によって上書きされる場合があります。システム レベルでの設定値が上書きされることはありません。ユーザが設定値を上書きしない場合には、上位レベルのプロパティ ファイルから下位レベルのプロパティ ファイルに値が継承されます。

次の図は、JRun の設定値の 4 つのレベルを示したものです。



システム レベルの設定値は、実行時に計算されます。これらの設定値は、プロパティ ファイルを編集しても、JMC を使用しても、上書きすることはできません。これには、次のような値があります。

jrun.server.rootdir
jrun.server.name
jrun.rootdir (UNIX のみ)

Windows 95/98/NT/2000 での jrun.rootdir システム設定値は、JRun のインストール ユーティリティによって設定され、Windows レジストリ内に格納されます。

グローバル プロパティは、global.properties ファイルと、一般的なプロパティ ファイル (pass.properties、jvms.properties、users.properties など) から構成されます。これらのファイル内にある設定値は、JRun のインストール先にあるすべての JRun サーバーに適用されます。たとえば、JRun のそれぞれのインストール先には、JMC にアクセスするためのパスワードが格納されているファイルが 1 つずつあります。

local.properties ファイルは、JRun のインストール先で JRun サーバー レベルでの設定を提供します。これらのファイルには、グローバル レベルおよびシステム レベルからプロパティが継承されますが、ファイル内にローカル値が含まれている場合は、継承されたプロパティが上書きされます。

webapp.properties ファイルは、Web アプリケーション レベルでの設定を提供します。これらの設定によって、ローカル レベルおよびグローバル レベルの設定は無効になります。JRun でこれらのファイルが作成されるのは、ユーザが JMC を使ってアプリケーション特有のプロパティを設定した場合のみです。

web.xml ファイルは、サーブレットおよび JSP の仕様により定義されます。アプリケーション レベルおよびコンテナ レベルの設定が提供されます。詳細については、適切な仕様を参照してください。

たとえば、次のような代入が行われているとします。

/default/local.properties	webapp=default_invoker
/admin/local.properties	webapp={default}
global.properties	webapp=invoker

結果 :

webapp 変数は、Admin サーバーの場合は invoker に設定され、default サーバーの場合は default_invoker に設定されます。

プロパティ ファイルの編集

ほとんどのプロパティは、JMC を使用して変更できます。しかし JRun プロパティは、編集ツールで設定する必要があります。テキスト エディタを使用し、JRun プロパティ ファイルをマニュアルで編集するか、または PropertyScript ユーティリティを使用して Java インタフェースを通して行うことができます (196 ページの「 PropertyScript の使用」を参照)。JRun のプロパティ ファイルを編集する場合は、いくつかの簡単な規則に従う必要があります。次のセクションでは、この規則について説明します。

構文

JRun プロパティ ファイルに変更を加えるには、次のような構文規則に従います。

- すべてのパラメータは、<parameter>=<value> によって設定されます。この構文では、等号 (=) の前後にスペースは入れません。
- 複数の値を区切るには、コンマまたはセミコロンのいずれかを使用します。
- いずれの行においても、行頭または行末にスペースまたはタブなどで余白を入れないでください。
- 変数は、中括弧 {} で囲みます。
- 行をコメントにするときには、ポンド記号 (#) を使用します。

編集

When editing a local.properties ファイルを編集する場合は、ファイルの編集前に関連する JRun サーバーを停止し、編集後にサーバーを再起動する必要があります。この操作によ

り、マニュアルによる変更後に、メモリに残っているものが local.properties ファイルに書き込まれるのを防ぐことができます。

また、JRun プロパティ ファイルを変更するときには、次のことに注意してください。

- プロパティ　ファイルを編集する前に、編集するプロパティ　ファイルのバックアップを作成します。
- テキスト エディタを使用して、*.properties という拡張子を持つ通常のテキスト ファイルとしてプロパティ ファイルを保存します。
- global.properties ファイルを編集する場合は、該当する JRun インストール先にあるすべての JRun サーバーを再起動します。

変数の使用法

JRun では、変数とプレースホルダーをよく使用します。変数とプレースホルダーは、プロパティ ファイルや JRun 管理コンソール (JMC) の中で中括弧 { }, によって示されます。

{variable} は、実行時には値に置き換えられます。変数と定数は、次のように同じ代入式の中で一緒に使用することができます。

jrun.services=scheduler,logging,monitor,{servlet.services},control

また次のように、変数と定数は同じ値の中で一緒に使用することもできます。

logging.filename={jrun.rootdir}/logs/{jrun.server.name}-event.log

最も頻繁に出てくる変数は、{jrun.rootdir} と {default} です。{default} は、実際には、変数というよりはプレースホルダーとして機能します。以下の表は、この 2 つの変数と、その他の変数について説明したものです。

変数の使用法	
変数	**説明**
{jrun.rootdir}	{jrun.rootdir} 変数は、インストール時に設定されるシステム変数です。UNIX システムでは、JRun サーバー プロセスを起動したときに、この値が計算されます。Windows 95/98/NT では、インストール時に、この値がシステム レジストリに一度だけ書き込まれます。 JRun サーバーに関連する JVM 実行可能プログラムを起動すると、JRun は、-D スイッチを指定して、関連する値を持つ jrun.rootdir を受け渡します。たとえば、次のように設定します。 java -D jrun.rootdir=c:\Allaire\JRun\
{default-app.rootdir}	JRun のすべての Web アプリケーションは、ルート ディレクトリを持ちます。この変数を使用すると、Web アプリケーションのルート ディレクトリを動的に指定することができます。 これは、アプリケーション ファイルで利用するドキュメントのルート ディレクトリです

変数の使用法	
変数	**説明**
{default}	{default} という値は、該当するパラメータが別のプロパティ ファイルで設定されていることを示します。最も一般的な使用法は、あるパラメータに global.properties の中で実際の値を設定し、local.properties の の中では、そのパラメータを {default} に設定する方法です。 ブランクを代入すると、変数の値がヌルに設定されます。local.properties 内に foo={default} という代入式があり、global.properties 内に foo= という代入式がある場合、foo にはヌルが代入されます。 local.properties と global.properties の両方で、同一のパラメータが {default} またはブランクに設定されていることは通常はありません。
その他の変数	プロパティ ファイル内では、動的なシステム変数をいくつか使用することができます。これらの変数には、{date}、{hour}、{day}、{month}、{year} などがあり、主にログ ファイルの出力設定やファイル名の指定で使用します。 オブジェクトを変数と混同しないように注意してください。たとえば、logging.dispatchLogger.events は、ディスパッチ ロガーによって割り当てられるイベントを、コンマで区切って指定したものです。これらのイベントは、一見システム変数のように見えますが、実際はオブジェクトであるので、一般的にこのような文字列は編集しません。既定値は次のとおりです。 {logging.infoevent}、{logging.debugevent}、{logging.warningevent}、{logging.errorevent}

PropertyScript の使用

PropertyScript ユーティリティを使用して、JRun プロパティ ファイルの設定を変更、追加、削除できます。PropertyScript では、JRun プロパティ ファイルを変更する指示語が含まれる、別のスクリプト ファイルが処理されます。

PropertyScriptクラスは、install.jarファイルのallaire.jrun.installパッケージの一部です。クラスパスに install.jar を含める必要があります。

PropertyScript の使用法

java –cp [*classpath*] allaire.jrun.install.PropertyScript *script–file* [*property–file*]

script-file は、指示語が含まれるスクリプト ファイルへのファイル システム パスです。スクリプト ファイルの作成の詳細については、197 ページの「スクリプト ファイルの作成」を参照してください。

property-file オプションとは、JRun プロパティ ファイルのことで、global.properties、local.properties、または jvms.properties などです。このオプションは、現在の JRun リリースでは抑制されており、必要はありません。script-file で JRun プロパティ ファイルを指定できます。

たとえば、次のようになります。

```
C:\>java -cp "c:\program files\allaire\jrun\lib\install.jar"
    allaire.jrun.install.PropertyScript
    "c:\program files\allaire\jrun\servers\default\script.txt"
```

スクリプト ファイルの作成

script-file オプションは、JRun プロパティ ファイルを変更するために PropertyScript によって使用される指示語すべてを含むテキスト ファイルを指します。スクリプト ファイルを作成する際、ファイル コマンドを少なくとも 1 つ、それから *filename* (referring to JRun のプロパティ ファイル) が必要です。各ファイル コマンドには、指示語および *filename* で実行するオプションをリストします。

スクリプト ファイルでは、次の構文に従います。

```
file filename
directive1
[directive2]
...
[file filename]
[directive1]
[directive2]
...
```

新規の file コマンドは、スクリプト ファイルの各プロパティ ファイル セクションの最初に置く必要があります。また、*filename* では、*.property ファイルへの完全なパスを指定する必要があります。

スクリプト ファイルの各指示語は、通常、コマンドおよび、キーと値のセットから成ります。キーは、JRun プロパティ ファイルのプロパティに一致します。次の例では、コマンドが add、キーが control.endpoint.main.port 、値が 53000 です。

```
add control.endpoint.main.port=53000
```

複数の値を持つことのできるキーが、値がスペースまたはカンマで区切られていることがあることに注意してください。

以下の表は、スクリプト ファイルの指示語とそれらのオプションについて説明したものです。

指示語	説明
add *key value*	プロパティ ファイルの最後に、新しいキーと値を追加します。既存のキーを追加すると、PropertyScript によりキーと値が上書きされます。
replace *key value*	指定のキーの値を別の値に置き換えます。存在しないキーに値を置き換えようとすると、PropertyScript により指示語は add として扱われます。
delete *key*	指定のキーとその値を削除します。
clear *key*	指定のキーの値をすべて削除し、キーを残します。
append *key value*	指定のキーの値の最後に別の値を追加します。値の最初に区切り記号 (通常カンマ) を指定する必要があります。 jcp を servlet.services キーに追加する例 : `append servlet.services ,jcp` キーによっては、区切り記号にカンマではなくスペースが使用されることがあります。このような場合、append_space 指示語を使用します。
append_space *key value*	指定のキーの最後に、スペースとそれに続けて値を追加します。区切り記号にスペースが使用される java.args キーを変更するときに便利です。
token_remove *key value*	指定の値 (1 つ) を検索し、その値と続く区切り記号 (スペースまたはカンマ) を削除します。
ejb_append jrun.services *value*	global.properties ファイルのみで jrun.services キーを変更するのに使用します。このメソッドにより、リストの最後に表示される {servlet_services} キーをリストの最後に確実に表示させることができます。このキーはすべての他のサービスの後に開始させる必要があります。

指示語	説明
unplug [servlet|ejb]	サーブレットまたは JRun の EJB コンポーネント機能を削除します。ユーザにコンポーネントの追加および消去の許可を与えるメンテナンスを提供する場合使用できます。それ以外の場合、インストール ファイルの変更により、意図しないコンポーネントのインストールを防ぐことができます。 unplug ではコンポーネントをアンインストールせずに使用不可にすることに注意してください。
comment *key text-string*	指定のキーの上の行に *text-string* から成るコメント行を追加します。
adduser *username password*	新規の JMC ユーザを追加します。UnixCrypt を使用して、PropertyScript が暗号化されます。この指示語を使用するのは、pass.properties ファイルを変更する場合のみです。 この指示語によって、新規の Web アプリケーション ユーザが users.properties ファイルに追加されるわけではないことに注意してください。Web アプリケーション ユーザを追加する場合は、『JRun によるアプリケーションの開発』の PropertyFileAuthentication クラスについての説明を参照してください。
port *key min,max*	インストール中に JRun が admin サーバーをアクセス可能にしようとするポートの範囲を設定します。min には最小ポート数が、max には最大ポート数が割り当てられます。PropertyScript では、この範囲のすべてのポートで試行されます。 各 JRun サーバーの local.properties ファイルのポート指示語を必ず使用します。 admin サーバーのデフォルトの範囲は 8000 から 8099 です。default サーバーのデフォルトの範囲は 8100 から 8199 です。

スクリプト ファイル サンプル

ここでは、スクリプト ファイルの例を紹介します。この例では、新規の Web アプリケーション の MyStocks を追加し、default JRun サーバーの local.properties ファイルの demo-app を削除します。最後の行では、default JRun サーバー用に 8080 と 8089 のうち使用可能なポートに JRun Web Server のポートを変更します。

```
file c:\program files\allaire\jrun\servers\default\local.properties
```

```
add webapp.mapping./mystocks /mystocks
add /mystocks.rootdir C:\\Mycompany\\servers\\default\\mystocks
add /mystocks.class {webapp.service-class}
token_remove servlet.webapps demo-app
append servlet.webapps ,/mystocks
comment servlet.webapps #Added mystocks app, removed demo
delete demo-app.rootdir
delete demo-app.class
delete webapp.mapping./demo
port web.endpoint.main.port 8080,8089
```

索引