

# JRun セット アップ ガイド

Windows®、UNIX、および  
Linux™ 用 JRun 3.1

# 版權告知

© 2000, 2001 Allaire Corporation. All rights reserved.

本書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。本書の内容は、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、本書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、米国における 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、米国および各国における Allaire Corporation 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、各所有者に帰属する商標です。

この製品には RSA Data Security からライセンス供与されたコードが含まれています。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

# 目次

<b>はじめに</b> .....	<b>ix</b>
JRun 製品のラインナップ .....	x
JRun をインストールするためのシステム要件 .....	xi
ハードウェアの必要条件 .....	xi
ソフトウェアの必要条件 .....	xi
JRun のバージョン 2.3.x からのアップグレード .....	xv
JRun 2.3.x と 3.x の同時実行 .....	xv
管理 .....	xvi
縮小または廃止された機能 .....	xvii
Java 製品の概要 .....	xviii
Java Platform 版 .....	xviii
Java Software Development Kit .....	xviii
Java Runtime Environment .....	xix
拡張サービス .....	xx
開発者リソース .....	xxi
JRun 文書の概要 .....	xxii
印刷およびオンライン文書セット .....	xxii
オンライン文書 .....	xxii
その他のリソース .....	xxiii
お問い合わせ先 .....	xxv
<b>第 1 章 JRun のインストール</b> .....	<b>1</b>
JRun のインストール .....	2
以前のバージョンの JRun のインストール .....	2
Windows 95/98/NT/2000 へのインストール .....	3
UNIX および Linux へのインストール .....	12
JRun 管理コンソールの起動 .....	15
JRun のディレクトリ構造 .....	18
admin JRun サーバーのサブディレクトリ .....	19
default JRun サーバーのサブディレクトリ .....	19

JRun サーバーの使用方法 .....	21
Windows に関する検討事項 .....	21
JRun サーバーの起動と停止 .....	21
JRun デモ アプリケーションの開始 .....	23
インストールのトラブルシューティング .....	25
JMC のエラー .....	25
デモ アプリケーションのエラー .....	26
Windows での JRun のエラー .....	27
<b>第 2 章 JRun の外部 Web サーバーへの接続 .....</b>	<b>29</b>
接続の概要 .....	30
Apache の接続 .....	31
Raven と JRun の併用 .....	35
Apache コンフィギュレーション ファイルへの変更 .....	35
IIS 3.0/PWS の接続 .....	36
コンフィギュレーション ファイルへの変更 .....	39
IIS 4.0/5.0 の接続 .....	39
JRun の IIS 4.0/5.0 への接続 .....	40
IIS コンフィギュレーション ファイルへの変更 .....	44
JRun ISAPI フィルタの構成 .....	44
Netscape/iPlanet への接続 .....	47
Java インタプリタの有効化 .....	50
NES コンフィギュレーション ファイルへの変更 .....	51
サンプル obj.conf ファイル .....	53
WebSite Pro への接続 .....	54
サブレットを実行するための URL 接頭辞のマッピング .....	54
マルチホームおよび URL 接頭辞 .....	56
ファイル拡張子の JRun へのマッピング .....	57
WebSite Pro と通信するための JRun の構成 .....	58
WebSite Pro コンフィギュレーション ファイルへの変更 .....	60
Java ベースの Web サーバーの接続 .....	61
サブレットの実行用 CGI インターフェイスの構成 .....	62
Zeus Web サーバーの接続 .....	63
Zeus コンフィギュレーション ファイルへの変更 .....	65
local.properties への変更 .....	66
コネクタのトラブルシューティング .....	66
JRun コネクタ ウィザードの使用 .....	66
JRun デモ アプリケーションのテスト .....	67

<b>第 3 章 JRun 管理コンソール</b> .....	<b>69</b>
JRun 管理コンソールの開始 .....	70
JRun 管理コンソール .....	72
JMC のお気に入りの設定 .....	73
JRun シリアル番号の設定 .....	74
JMC ユーザの管理 .....	75
新規 JMC ユーザの追加 .....	75
JMC ユーザの設定の変更 .....	77
JMC ユーザの削除 .....	78
パスワードの変更 .....	79
JRun サーバーの設定 .....	80
JMC での JRun サーバーの管理 .....	81
JMC での JRun サーバーの再起動 .....	82
jrun コマンドの使用 .....	83
JRun サーバーの追加と削除 .....	86
Java Virtual Machine の設定 .....	91
JWS での SSL (Secure Socket Layer) の使用 .....	94
JRun サーバー イベント ログの設定 .....	108
JDBC データ ソースの設定 .....	109
新しい JDBC データ ソースの追加 .....	110
JDBC データ ソースの編集 .....	111
よくある問題とその解決方法 .....	113
Web サーバーの設定 .....	114
並行処理の概要 .....	114
JRun コネクタ ウィザードの使用 .....	115
JWS の設定 .....	117
外部 Web サーバーの設定 .....	119
Web アプリケーションの構成 .....	122
既定のアプリケーション .....	122
アプリケーション パネル .....	123
アプリケーションの作成 .....	124
アプリケーションの公開 .....	125
アプリケーションの編集 .....	128
アプリケーションの削除 .....	130
アプリケーション パスのマッピング .....	131
アプリケーション ホストの作成 .....	132
アプリケーション パラメータの追加 .....	134
サーブレット ディレクトリの追加 .....	135
ファイル設定の変更 .....	138
JSP コンパイラの構成 .....	139
JRun アプリケーション イベント ログの構成 .....	141
MIME タイプのマッピング .....	142
セッション トラッキングの構成 .....	143

サブレットの構成 .....	146
サブレットの定義 .....	146
サブレットへの要求マッピング .....	148
サブレットのエイリアス設定 .....	151
サブレットのチェーン化 .....	152
SSI タグレットの使用 .....	155
エンタープライズ アプリケーションの構成 .....	156
EJB の公開 .....	156
EJB の再公開 .....	158
EJB の削除 .....	159
EJB の構成 .....	160
EAR ファイルの公開 .....	161
ログ ファイルビューアの使用 .....	163
JMC キーの検索 .....	166
ログアウト .....	166
<b>第 4 章 コネクタについて .....</b>	<b>167</b>
JRun ポートについて .....	168
空きポートの検出 .....	170
Web サーバー コネクタについて .....	172
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 173	
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 .....	174
local.properties ファイル内のコネクタのプロパティ .....	175
1 つの Web サーバーへの複数の JRun サーバーの接続 .....	176
設定の詳細 .....	176
Web サーバーの接続 .....	178
単純な分散環境での JRun の実行 .....	179
単純な分散環境でのインストール .....	179
単純な分散環境の例 .....	181
複雑な分散環境での JRun の実行 .....	182
複雑な分散型インストール .....	182
複雑な分散環境の例 .....	183
分散環境での JSP の使用 .....	185
pathtrans プロパティの編集 .....	185
pathtrans の例 .....	185
分散 JRun システムの保護 .....	186
JWS のオフ .....	186
コネクタのホストベース認証 .....	186

---

JRun でのマルチホスティング .....	188
Apache でのマルチホスティング .....	189
IIS でのマルチホスティング .....	190
Netscape でのマルチホスティング .....	191
要求のチェーン化 .....	192
ターゲット サーバーの設定の確認 .....	192
呼び出す側のサーバーの設定の定義 .....	193
ターゲット JRun サーバーの設定の定義 .....	193
補足情報 .....	193
カスタム コネクタの作成 .....	194
Apache 用のコネクタのコンパイル .....	195
Netscape 用のコネクタのコンパイル .....	196
<b>第 5 章 プロパティ ファイル .....</b>	<b>197</b>
プロパティ ファイルの概要 .....	198
プロパティ ファイルの再ロード .....	199
プロパティ ファイルの階層について .....	200
プロパティ ファイルの編集 .....	202
構文 .....	202
編集 .....	202
変数の使用法 .....	202
PropertyScript の使用法 .....	204
PropertyScript の使用法 .....	204
スクリプト ファイルの作成 .....	204
サンプル スクリプト ファイル .....	207





# はじめに

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、文書、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も説明します。

## 目次

- JRun 製品のラインナップ ..... X
- JRun をインストールするためのシステム要件 ..... xi
- JRun のバージョン 2.3.x からのアップグレード ..... xv
- Java 製品の概要 ..... xviii
- 開発者リソース ..... xxi
- JRun 文書の概要 ..... xxii
- その他のリソース ..... xxiii
- お問い合わせ先 ..... xxv

## JRun 製品のラインナップ

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。次の表は、JRun のエディションの一覧です。

版	説明	価格
Developer 版	Web 開発および EJB/JMS/JTA をサポートしており、JRun JDBC ドライバを含みます。無制限の JVM (Java Virtual Machine) 数、およびサーブレット、JSP、EJB (Enterprise JavaBeans) の 3 つの同時接続数が許可されています。	Web アプリケーションおよび EJB の非営利目的の開発/テスト用のライセンスであり、無償で入手できます。アプリケーションの公開を目的とした使用は許可されていません。
Professional 版	Web 開発のみをサポートしています。JRun Professional 版では、JVM 数、およびサーブレット /JSP (JavaServer Pages) の同時接続数はいずれも無制限です。	営利目的の公開用のライセンスで、CPU 単位のライセンスとなります。
Advanced 版	Allaire ClusterCATS を使用する HTTP ベースのロード バランス機能およびフェイルオーバー ソフトウェアが含まれています。Web 開発をサポートしており、JRun JDBC ドライバを含みます。JRun Advanced 版では、JVM 数、およびサーブレット /JSP の同時接続数はいずれも無制限です。	営利目的の公開用のライセンスで、CPU 単位のライセンスとなります。
Enterprise 版	Allaire ClusterCATS を使用する HTTP ベースのロード バランス機能およびフェイルオーバー ソフトウェアが含まれています。Web 開発、EJB、JMS (Java Messaging Service)、JTA (Java Transaction API)、および JRun JDBC ドライバをサポートしています。JRun Enterprise 版では、JVM 数、サーブレット、JSP、および EJB の同時接続数はいずれも無制限です。	営利目的の公開用のライセンスで、CPU 単位のライセンスとなります。
Studio 版	HomeSite HTML エディタに基づいた統合 JSP 開発環境です。JRun サーバーは含まれていません。	価格はライセンス単位です。

最新の価格情報については、国内総販売元である (株)アイ・ティ・フロンティアにお問い合わせください (株式会社シリウスは、2001 年 4 月に株式会社アイ・ティ・フロンティアに社名変更いたしました)。

## JRun をインストールするためのシステム要件

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 の完全インストールを行うには、次のハードウェアが最低限必要となります。

- 32 MB の RAM (64 MB を推奨)
- 30 MB のハードディスク容量 (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 (詳細については次を参照)。

- 「オペレーティングシステムの必要条件」 xi ページ
- 「インターネットブラウザの必要条件」 xi ページ
- 「Java の必要条件」 xii ページ
- 「Web サーバーの必要条件」 xiv ページ
- 「JRun JDBC ドライバのデータベース必要条件」 xv ページ (JRun の Developer 版、Advanced 版、および Enterprise 版のみ)

### オペレーティングシステムの必要条件

JRun には、最低限、次に示す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、必要条件をより厳密に指定される場合があります。

- Windows 95/98/NT/2000 (NT には Service Pack 3 以降が必要)
- Solaris 2.6、2.7、8
- Red Hat Linux 6.x、7.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、次のいずれかの Web ブラウザをインストールする必要があります。

- 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 サブレット、JSP ページ、および EJB の開発に必要な Java ユーティリティの一覧です。この表には最低限の必要条件が示されています。ただし、各ユーティリティの最新版を使用してください。また、Java ユーティリティのベータ版は、システムでは使用しないでください。

### メモ

JVM 1.1.8 を使用している場合は、[xiii ページ](#)の「JVM 1.1.8 に関する特別な検討事項」を参照してください。

Java の最新バージョンは、<http://java.sun.com> から入手できます。

Java コンポーネント	Windows	UNIX
Java Virtual Machine (JVM) Java Runtime Environment (JRE) と呼ばれることもあります。	JRE 1.3 は JRun に含まれていますが、個別にインストールする必要があります。独自の JRE をインストールできますが、1.1.6 以降である必要があります (1.1.8 以降を推奨)。EJB については、JDK 1.2.2 以降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)。EJB については、JDK 1.2.2 以降を使用する必要があります。1.1.6 以降のバージョンが必要です。UNIX に対して最適化するには、JVM Version 1.1.8 以降が必要です。Version 1.2 を推奨します (HP/UX の場合は Version 1.2 が必須です)。Solaris の場合は、1.3 JDK に含まれている HotSpot サーバーを推奨します。
Java サブレット	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はありません。JRun には必要なツールがすべて含まれています。	追加のソフトウェアをインストールする必要はありません。JRun には必要なツールがすべて含まれています。
Enterprise JavaBeans (EJB)	JDK 1.2.2 以降	JDK 1.2.2 以降

### サポートされている JVM

JRun インストールの JVM Advisor には、JRun によってサポートされている JVM の一覧が記載されています。JRun でサポートされていない JVM がない場合は、JVM をインストールしてから、JRun のインストールを再び実行する必要があります。

Windows ユーザの場合は、JRE 1.3 は JRun CD に含まれています。ダウンロードすることもできます。Java コンポーネントの詳細については、[xviii ページ](#)の「Java 製品の概要」を参照してください。

## JVM 1.1.8 に関する特別な検討事項

JVM 1.1.8 を使用している場合は、次の特別な検討事項に注意してください。

- JVM のメモリを 64 MB 以上に設定します。これを設定するには、次の引数を JVM の引数リストに追加します。  
`-mx64m`  
この引数を追加すると、ヒープ サイズを増やすことができます。必要に応じて、この値を 64 以上に設定できます。JRun をインストールしたら、この引数を次の 2 通りの方法で追加できます。
  - 引数を、JMC の [Java の設定] パネルの [Java 引数] フィールドに追加します。詳細については、[91 ページの「Java Virtual Machine の設定」](#)を参照してください。
  - `global.properties` ファイルにある `java.args` プロパティを編集します。プロパティファイルの編集については、[第 5 章](#)を参照してください。
- JRE を 1.1.8 からそれ以降にアップグレードする場合は、プロパティ ファイルにある `java.args` プロパティから次の引数を削除する必要があります。  
`-Djava.naming.factory.initial=allaire.jrun.ContextFactory`  
この引数は、次の 2 とおりの方法で削除できます。
  - JMC にある [Java の設定] パネルを使用します。詳細については、[91 ページの「Java Virtual Machine の設定」](#)を参照してください。
  - `global.properties` ファイルにある `java.args` プロパティを編集します。

## Web サーバーの必要条件

JRun の外部 Web サーバーへの接続は共通の作業です。次の表は、各プラットフォームでサポートされている Web サーバーの種類を示します。

Web サーバー コネクタで使用可能なプラットフォーム								
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/ Workstation		X		X	X	X		X
Windows 2000 Professi onal/ Server			X	X	X	X		X
Windows 95/98	X							X
Red Hat Linux 6.x、7.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	
IBM AIX 4.2/4.3*				X	X	X	X	
Compaq UNIX Tru64 4.0*				X	X	X	X	

\* JDK 1.2 以降対応の 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 JDBC ドライバのデータベース必要条件

JRun の Developer 版、Advanced 版、および Enterprise 版に含まれている JRun JDBC ドライバは、次のデータベースプラットフォームをサポートしています。

- Sybase 11
- Sybase 12 ASE
- SQL Server 7
- SQL Server 2000
- DB2 UDB 7.1 (Windows NT、IBM AIX、Sun Solaris 対応)
- DB2 UDB 6.X (Windows NT、IBM AIX、Sun Solaris 対応)

JRun JDBC ドライバの使用方法の詳細については、『JRun JDBC Drivers User's Guide and Reference』を参照してください。独自のデータベースドライバを JRun で使用することもできます。

## JRun のバージョン 2.3.x からのアップグレード

JRun 3.1 にアップグレードすると、EJB のサポート、更新されたサーブレット API、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できます。これらの追加機能を活用するために、準備が必要な場合もあります。

---

### メモ

ベータ版の上に JRun 3.1 をインストールする場合は、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。

---

JRun と 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 3.1 と JRun 3.0 の違いについては、『JRun Version 3.1 機能および移行ガイド』を参照してください。

## JRun 2.3.x と 3.x の同時実行

既定では、JRun のインストールスクリプトは JRun を C:\Program Files\Allaire\JRun (Windows) と /opt/jrun (UNIX) にインストールします。JRun のバージョン 2.3.x を同時に実行する場合は、最初に JRun サーバーをすべて停止する必要があります。次の手順のいずれかを行います。

- 既存の JRun ディレクトリをほかの場所に移動する。
- JRun の新しいバージョンを新しい場所にインストールする。

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 2000 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 サーバーの設定が正しく行われなことがあります。

インストールの際は、JRun サービス用の固有のポートを選択します。詳細については、[168 ページの「JRun ポートについて」](#)を参照してください。

---

## メモ

JRun 3.x 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/2000 で実行中の JRun プロセスを中止するには

- 1 Ctrl + 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 マネージャが表示されます。
- 3 [プロセス] タブをクリックします。
- 4 [イメージ名] によってソートします。
- 5 次の名前プロセスをすべて中止します (それぞれ複数の場合があります)。

```
javaw.exe  
jrun.exe
```

## 管理

### ユーザ インターフェイス

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) を使用して JRun を構成します。詳細については、[第 3 章](#)を参照してください。

### プロパティ ファイル

JMC には、JRun のプロパティ ファイルを変更するためのグラフィカル インターフェイスが用意されています。これらのファイルは、サーバーおよび Web アプリケーションの初期化および構成に使用されます。既定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 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

Java サーブレット 2.2 仕様の導入部に、Web アプリケーションと .war ファイルの概念についての説明があります。Web アプリケーション内のクラスと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 アプリケーションの一部でない個別のサーブレットは、*JRun* のルート ディレクトリ / *servlet* ディレクトリに配置することによって、その使用が引き続きサポートされます。既定の Web アプリケーションでは、そのクラスパス内にこのディレクトリが含まれます。



## 縮小または廃止された機能

JRun 3.x には、Web アプリケーションから Enterprise JavaBeans まで、最新の仕様が導入されています。ただし、廃止された機能または段階的に廃止される機能もあります。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 の次回からのリリースではサポートされません。詳細については、「JRun 開発者センター」を参照してください。

### サーバー側インクルード (SSI)

SSI は、以前はダイナミック コンテンツの作成に広く使用されていました。JRun では主に古い実装をサポートする目的で使用します。現在では、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 (ASP)

JRun は ASP のサポートを廃止しました。

## 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。JRun を実行するための Java の必要条件については、[xii ページの「Java の必要条件」](#)を参照してください。Java の最新バージョンの詳細については、Sun の Web サイト <http://java.sun.com> を参照してください。

## Java Platform 版

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 3 つの版があります。

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

Java 2 Platform 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) は通常、Java Development Kit (JDK) と呼ばれます。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に使用するツールとコアクラスから構成されています。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。

## SDK の主なコンポーネント

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

## SDK のバージョン

- JDK 1.0.x
- JDK 1.1.x
- J2 SDK version 1.2.2, Standard Edition および Enterprise Edition
- J2 SDK version 1.3, Standard Edition および Enterprise Edition

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 Runtime Environment (JRE) は Java Virtual Machine (JVM) 仕様の実装であり、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 プラットフォーム用に作成されたプログラムを実行する場合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基づいて JRE を自由に配布できます。

### JRE の主なコンポーネント

- Java Virtual Machine
- Java アプリケーションランチャ
- 実行時クラスライブラリ
- Java Plug-in (ブラウザ用)
- Java HotSpot Runtime (1.3 以降)

### JRE のバージョン

- JRE 1.1.x
- Java 2 Runtime Environment, Standard Edition, バージョン 1.2.2
- Java 2 Runtime Environment, Standard Edition, バージョン 1.3

JVM はソフトウェアによる CPU の実装であり、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。Hewlett Packard、Sun、Microsoft、Symantec などの多くの企業が独自の JVM を開発しています。*Java Runtime Environment* という用語は、Sun の JVM 実装の Sun 固有の名前です。ただし、JVM を JRE と呼ぶベンダも多くあります。本書では、JRE と JVM は同じものとして使用します。JRun インストールの JVM Advisor には、JRun によってサポートされている JVM の一覧が記載されています。

## 拡張サービス

Java は拡張可能な言語であり、継続的に機能を拡張しています。ここでは、JRun で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際に、コンポーネントごとにインストールするかどうかを選択できます。

## サーブレット

サーブレットは動的コンテンツを生成する Java Web コンポーネントです。JRun 3.x は Sun のサーブレット 2.2 仕様に準拠しています。この仕様は 2.1 に基づいて確立され、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) のサポートが含まれています。JRun にサーブレット仕様を実装するには、JRE 1.1.6 以降が必要です。

## JavaServer Pages

JavaServer Pages 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、動的な Web ページを作成します。JRun 3.x は、Sun の JSP 1.1 仕様をサポートしています。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が含まれています。JRun に JSP 1.1 を実装するには、JRE 1.1.6 以降が必要です。

## Enterprise JavaBeans

Enterprise JavaBeans 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、サーバー側の分散型コンポーネントです。JRun は Sun の Enterprise JavaBeans 1.1 仕様をサポートしています。EJB 1.1 仕様では、1.0 の仕様に JTA や JMS などの開発および公開強化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。

## 開発者リソース

(株)アイ・ティ・フロンティア(株式会社シリウスは、2001年4月に株式会社アイ・ティ・フロンティアに社名変更いたしました)では、開発者の教育、テクニカルサポートなどのサービスによりカスタマサポートを充実させております。以下にご紹介するWebサイトでは、すべてのオンラインリソースにすばやくアクセスできます。次の表に、このようなオンラインリソースにアクセスできるWebサイトのURLを示します。

リソース	説明
(株)アイ・ティ・フロンティア JRun のサイト <a href="http://cfusion.sirius.co.jp/jrun/">http://cfusion.sirius.co.jp/jrun/</a>	JRun の詳細な製品情報および関連トピック
開発者コミュニティ <a href="http://www.allaire.com/developer/">www.allaire.com/developer/</a>	JRun による開発に必要な最先端の情報を提供する、オンライン ディスカッション グループ、知識ベース、技術文書などのあらゆるリソース
JRun 開発者センター <a href="http://www.allaire.com/developer/jrunreferencedesk/">www.allaire.com/developer/jrunreferencedesk/</a>	開発のヒント、記事、文書、ホワイト ペーパーに関する情報サイト
JRun サポート フォーラム <a href="http://forums.allaire.com/jrunconf/">http://forums.allaire.com/jrunconf/</a>	Allaire オンライン フォーラムでは豊かな経験を持つ JRun 開発者と連絡をとり、JRun に関連した数多くのトピックについてメッセージを書き込んだり、回答を得ることができます。

## JRun 文書の概要

JRun 文書は、JSP 開発者、サーブレット 開発者、EJB クライアント 開発者、EJB 開発者、システム管理者を含むすべての JRun ユーザにサポートを提供することを目的としています。印刷物で提供されている場合でも、オンラインの場合でも、必要な情報を速やかに探し出せるように構成されています。JRun オンライン文書には、HTML 形式と Adobe Acrobat ファイル形式があります。

## 印刷およびオンライン文書セット

JRun 文書セットには、次の文書があります。

文書	説明
『JRun セットアップ ガイド』	JMC を使用した JRun のインストール、構成、および管理について説明します。
『JRun によるアプリケーションの開発』	Java サーブレット、JSP、および EJB から構成されるアプリケーションの開発方法について説明します。
『JRun サンプル ガイド』	サーブレット、JSP、および EJB のコード サンプルおよびサンプル アプリケーションを提供します。
『JRun タグ ライブラリ リファレンス』	JRun タグ ライブラリの JSP カスタム タグについて説明します。
『JRun 拡張設定ガイド』	ISP、ISV、および OEM カスタマ用の、JRun のインストール、使用、設定に関する情報があります。
『JRun JSP クイック リファレンス』	JSP のディレクティブ、アクション、およびスクリプト要素の簡単な説明と構文が記載されています。
『JRun Version 3.1 機能および移行ガイド』	JRun バージョン 3.1 の機能と、既存のアプリケーションをバージョン 3.1 に移行する方法について説明します。
『JRun タグ ライブラリ クイック リファレンス』	JRun タグ ライブラリの JSP カスタム タグの簡単な説明と構文について記載されています。

## オンライン文書

Allaire 社では、JRun の全文書のオンライン版を Adobe Acrobat (PDF) ファイルで提供しています。PDF ファイルは JRun CD-ROM に含まれ、既定では JRun /docs ディレクトリにインストールされます。JRun 管理コンソールのトップ ページにある製品の文書へのリンクをクリックすると、これらの PDF ファイルにアクセスできます。

また、これらの PDF ファイルは、Allaire 社の Web サイト <http://www.allaire.com/documents> からダウンロードすることもできます。

## その他のリソース

本書で扱っているトピックの詳細については、次のリソースを参照してください。

### 書籍

---

#### サーブレットと JavaServer Pages

『Java Server Pages Application Development』	Scott M. Stirling 他著、 Sams 刊、2000 年、 ISBN: 067231939X
『Java Servlets』	Karl Moss 著、 McGraw Hill 刊、1999 年、 ISBN: 0071351884
『Java Servlet Programming (Second Edition)』	Jason Hunter、William Crawford 著、 O'Reilly & Associates 刊、2001 年、 ISBN: 0596000405
『Core Servlets and Java Server Pages』	Marty Hall 著、 Prentice Hall 刊、2000 年、 ISBN: 0130893404
『Inside Servlets: Server-Side Programming for the Java Platform (Second Edition)』	Dustin R. Callaway 著、 Addison-Wesley 刊、2001 年、 ISBN: 0201709066
『Web Development with JavaServer Pages』	Duane K. Fields、Mark A. Kolb 著、 Manning Publications Company 刊、2000 年、 ISBN: 1884777996

---

#### Enterprise JavaBeans

『Mastering Enterprise JavaBeans and the Java 2 Platform, Enterprise Edition』	Ed Roman 著、 John Wiley & Sons 刊、1999 年、 ISBN: 0471332291
『Enterprise JavaBeans』	Richard Monson-Haefel 著、 O'Reilly & Associates 刊、2000 年、 ISBN: 1565928695
『Applying Enterprise JavaBeans: Component-Based Development for the J2EE Platform』	Vlada Matena、Beth Stearns 著、 Addison-Wesley Pub Co 刊、2000 年、 ISBN: 0201702673

---

---

**Enterprise Java プログラミング**

『Server-Based Java Programming』	Ted Neward 著、 Manning Publications Company 刊、2000 年、 ISBN: 1884777716
『Professional Java Server Programming J2EE Edition』	Danny Ayers 他著、 Wrox Press 刊、2000 年、 ISBN: 1861004656
『Designing Enterprise Applications with the Java 2 Platform, Enterprise Edition』	Nicholas Kassem 著、 Addison-Wesley 刊、2000 年、 ISBN: 0201702770 ( <a href="http://java.sun.com/j2ee/download.html#blueprints">http://java.sun.com/j2ee/download.html#blueprints</a> から無料でダウンロードできます。)
『Building Java Enterprise Systems with J2EE』	Paul Perrone、Venkata S.R. "Krishna" .R. Chaganti 著、 Sams 刊、2000 年、 ISBN: 0672317958
『J2EE: A Bird's Eye View (e-book)』	Rick Grehan 著、 Fawcette Technical Publications 刊、2001 年、 ISBN: B00005BAZV

---

## オンライン リソース

Java Servlet API	<a href="http://java.sun.com/products/servlet">http://java.sun.com/products/servlet</a>
JavaServer Pages API	<a href="http://java.sun.com/products/jsp">http://java.sun.com/products/jsp</a>
Enterprise JavaBeans API	<a href="http://java.sun.com/products/ejb/">http://java.sun.com/products/ejb/</a>
Java 2 Standard Edition API	<a href="http://java.sun.com/products/jdk/1.3/docs/api/index.html">http://java.sun.com/products/jdk/1.3/docs/api/index.html</a>
Servlet Source	<a href="http://www.servletsource.com">www.servletsource.com</a>
JSP Resource Index	<a href="http://www.jspin.com">www.jspin.com</a>
Server Side	<a href="http://www.theserverside.com">www.theserverside.com</a>
Dot Com Builder	<a href="http://dcb.sun.com">http://dcb.sun.com</a>
Servlet Forum	<a href="http://www.servletforum.com">www.servletforum.com</a>

---



## お問い合わせ先

### 販売元

株式会社アイ・ティ・フロンティア  
シリウス事業部

電話 : 03-5562-4099

Fax : 03-5562-4070

<http://cfusion.sirius.co.jp/jrun/>

E-mail : [jrunsales@sirius.co.jp](mailto:jrunsales@sirius.co.jp)

(株式会社シリウスは、2001年4月に株式会社アイ・ティ・フロンティアに社名変更いたしました)

### テクニカル サポート

Allaire 社では、電話および Web による幅広いサポート オプションを提供しています。テクニカルサポート サービスについては、<http://www.allaire.com/support/> をご覧ください。

JRun サポート フォーラム (<http://forums.allaire.com>) へは、いつでも投稿できます。



# 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 ..... 2
- JRun 管理コンソールの起動 ..... 15
- JRun のディレクトリ構造 ..... 18
- JRun サーバーの使用方法 ..... 21
- JRun デモ アプリケーションの開始 ..... 23
- インストールのトラブルシューティング ..... 25

## JRun のインストール

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。

インストールの際、既定では JRun アプリケーションをホスティングする 2 つの JRun Web Server (JWS) がインストールされることを知っておいてください (これを無効にすることができます)。これらの JRun アプリケーションには、JRun 管理コンソール (JMC) およびデモ アプリケーションが含まれています。これらのサーバーに割り当て可能なポートがあることを確認してください。既定値は、8000 (admin サーバー) および 8100 (default サーバー) です。

JRun のインストールおよび構成の基本的な手順を次の表に示します。

手順	章
1 JRun をインストールします。	<a href="#">第 1 章</a>
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。	<a href="#">第 1 章</a>
3 Web サーバーが JRun と通信できるように構成します。	<a href="#">第 2 章</a>
4 Web サーバーと JRun が通信していることを確認します。	<a href="#">第 2 章</a>
5 JRun 管理コンソール (JMC) を使用して、JRun 構成を追加します。	<a href="#">第 3 章</a>

JRun を分散型環境にインストールしている場合は、[第 4 章](#)を参照してください。

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
- UNIX と Linux

## 以前のバージョンの JRun のインストール

JRun CD には JRun Version 3.1 のほかに、JRun Version 3.0 SP2a も含まれています。このため、現在使用しているアプリケーションが新しいバージョンの JRun と互換性がない場合は、以前のバージョンの JRun をインストールできます。詳細については、『JRun Version 3.1 機能および移行ガイド』を参照してください。

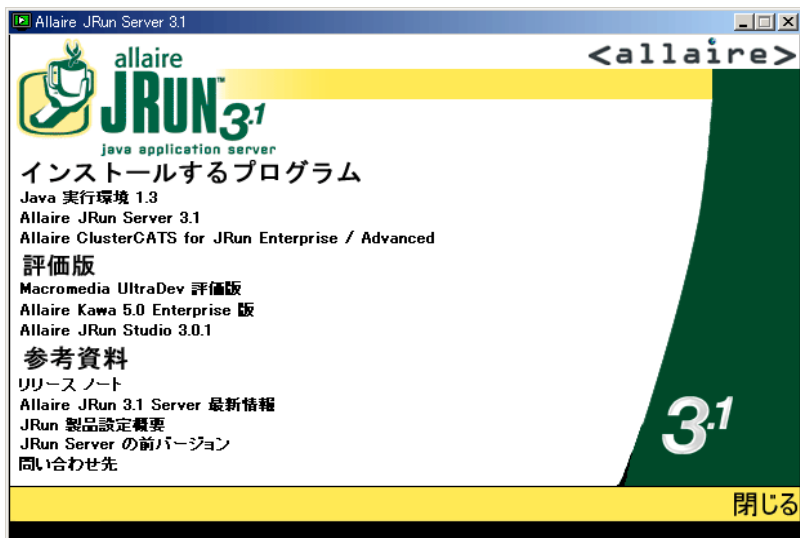
## Windows 95/98/NT/2000 へのインストール

ここでは、Windows 95/98/NT/2000 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は、その前に 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。JRun を CD からインストールしている場合、スプラッシュ画面は自動的に表示されます。

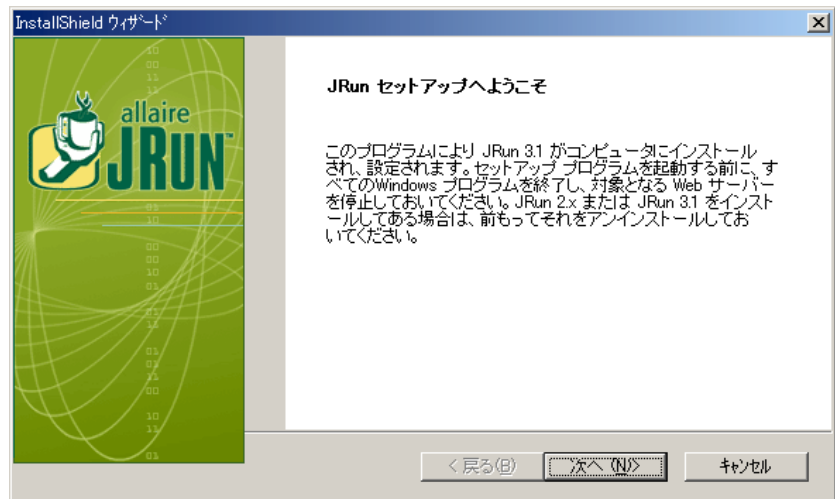


JRun をインストールするには、少なくとも JRE が必要です。JRE がない場合は、[Java Runtime Environment 1.3] をクリックすると、Sun JRE 1.3 をインストールできます。JRE のインストールが終了したら、JRun のインストールを再開します。

以前のバージョンの JRun (Version 3.02 SP2a) をインストールするには、[JRun Server の前バージョン] リンクをクリックします。この後のインストールプロセスは同じです。

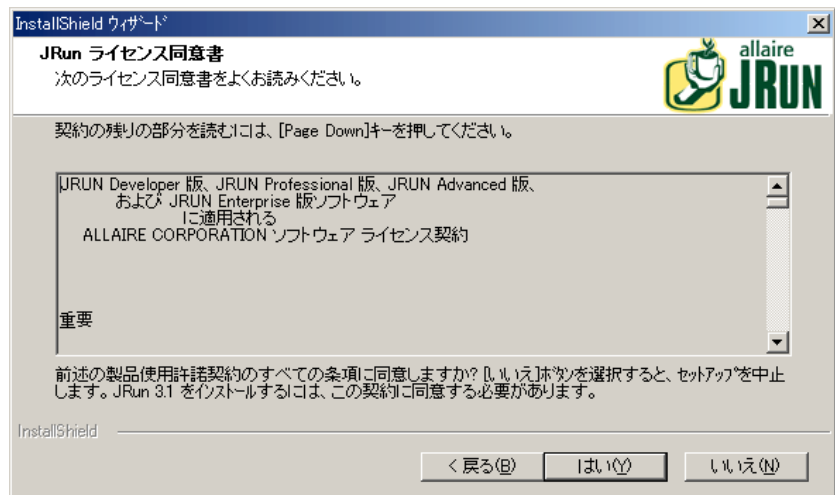
- 4 [Allaire JRun Server 3.1] をクリックします。

[JRun セットアップへようこそ] ウィンドウが表示されます。



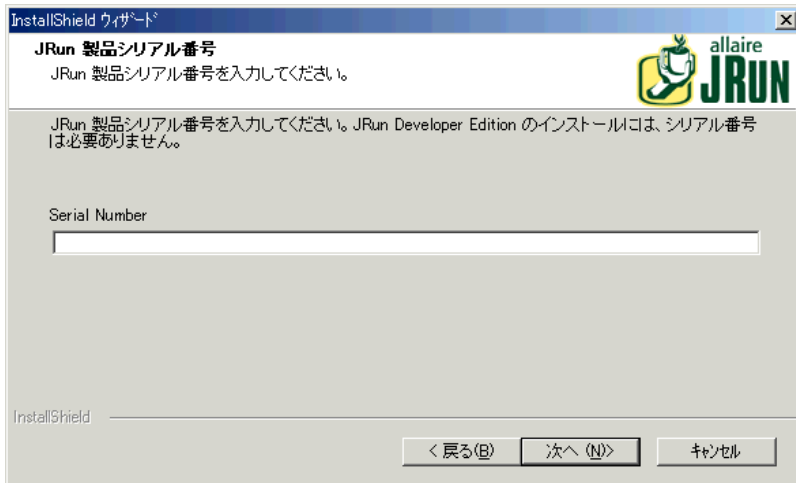
- 5 [次へ] をクリックします。

[JRun ライセンス同意書] ウィンドウが表示されます。



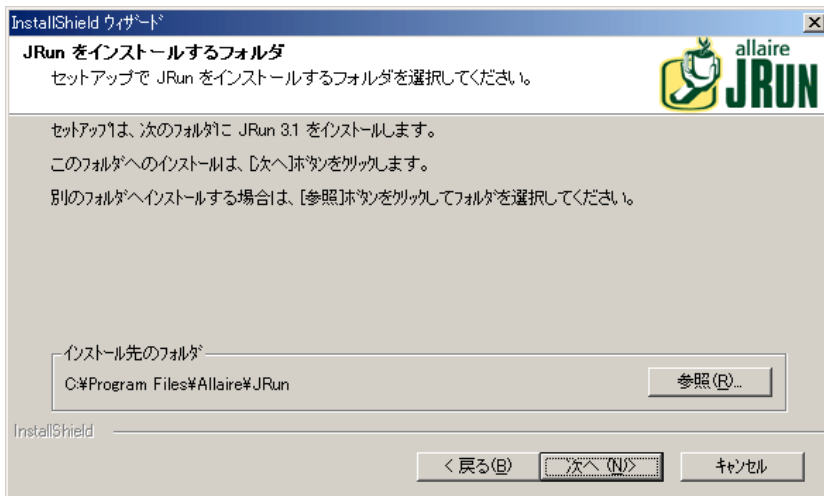
- 6 JRun ライセンス同意書に同意する場合は [はい] を、インストールを中止する場合は [いいえ] をクリックします。

[はい] をクリックすると、[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 Developer 版または評価バージョンをインストールする場合、このフィールドは空欄 (既定の設定) 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に要求されます。

[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。

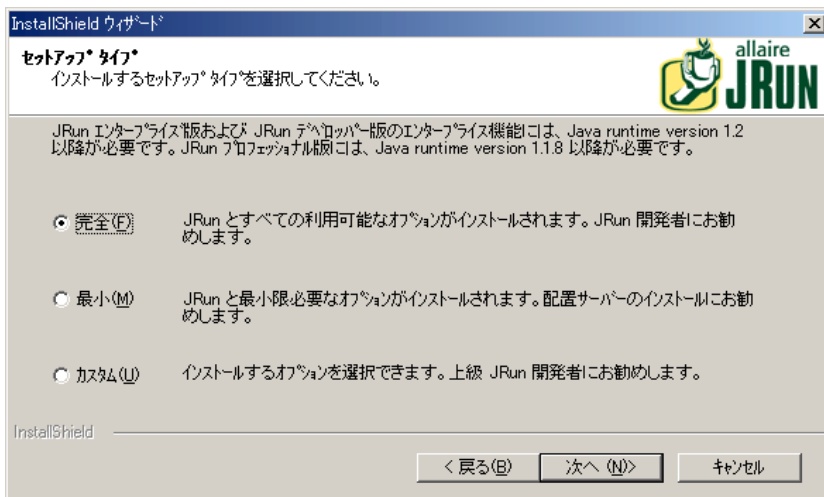


- 8 JRun をインストールするフォルダを選択し、[次へ]をクリックします。

### メモ

本書ではこのフォルダを *JRun* のルート ディレクトリと呼んでいます。

[セットアップ タイプ] ウィンドウが表示されます。

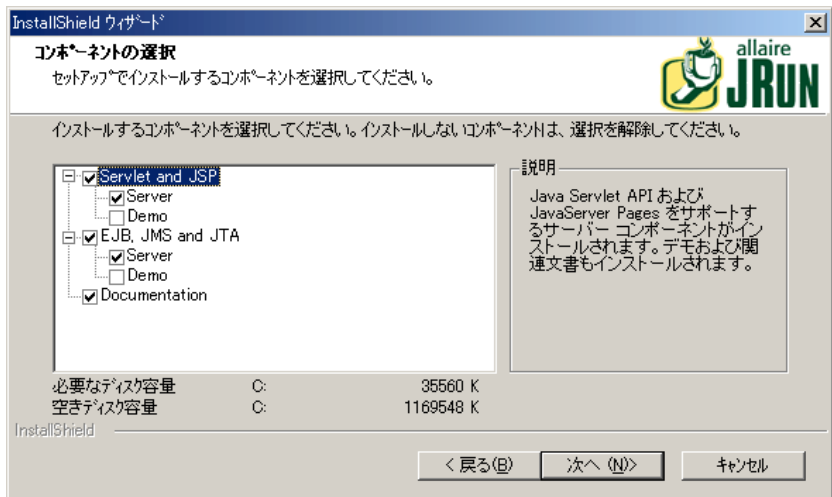


- 9 インストールの種類を選択して、[次へ]をクリックします。次の表では、オプションについて説明しています。

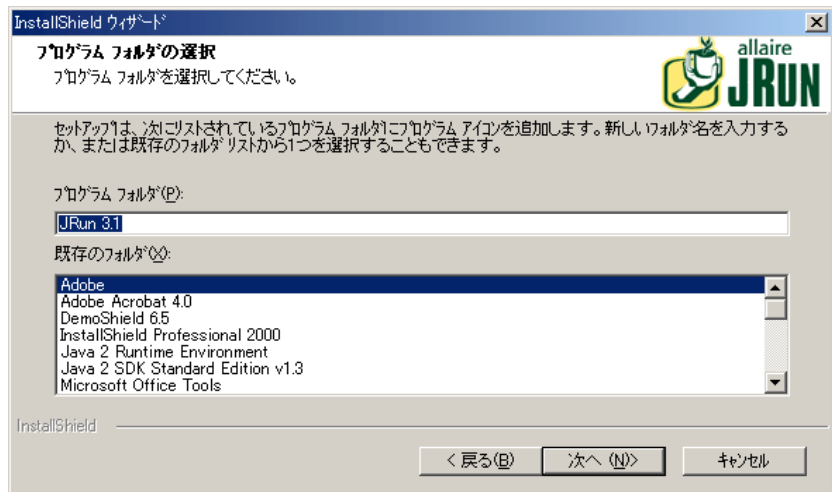
オプション	説明
完全	使用可能なすべてのオプションをインストールします。サーブレット、JSP、EJB、JMS、JTA サポート、サンプル、および文書がインストールされます。一般の JRun 開発者はこのオプションを選択することをお勧めします。
最小	最低限の必須オプションをインストールします。このインストールにはサーブレット、JSP、EJB、および JMS サポートが含まれます。文書およびデモアプリケーションはインストールされません。アプリケーションを公開するサーバーにインストールする場合は、このオプションを選択することをお勧めします。
カスタム	インストールするオプションを独自に選択できます。経験豊富な JRun 開発者は、このインストール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 版をご使用の場合、EJB コンポーネントは利用できません。

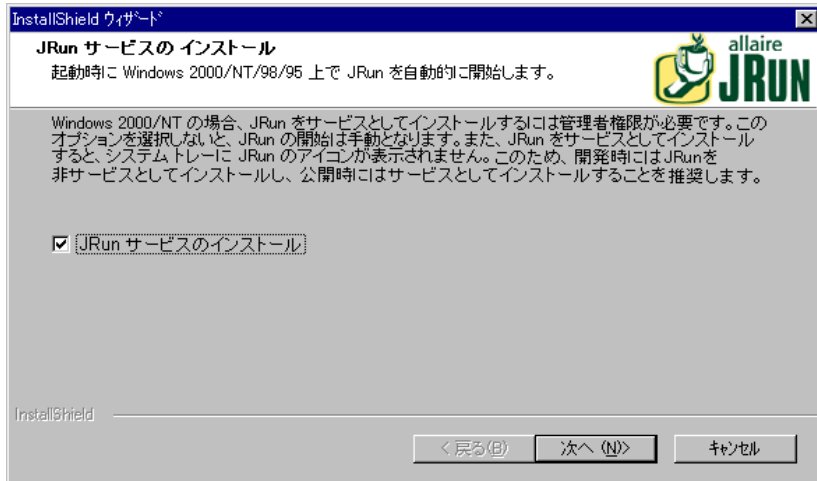


インストールするコンポーネントを選択して、[次へ]をクリックします。  
[プログラム フォルダの選択]ウィンドウが表示されます。



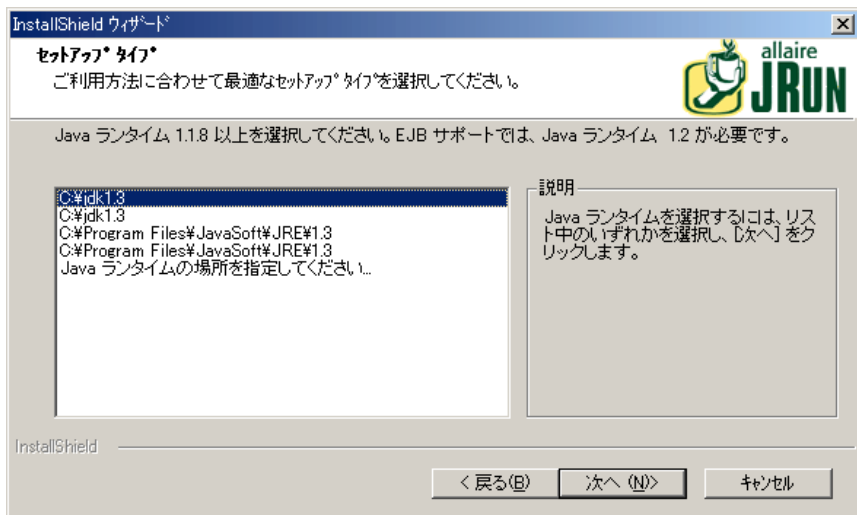
- 10 JRun をインストールするプログラム フォルダ名を選択し、[次へ]をクリックします。

- 11 指定したファイルがインストールされます。[JRun サービスのインストール]ウィンドウが表示されます。



- 12 JRun サーバーを NT サービスとしてインストールするかどうかを選択し、[次へ] をクリックします。このボックスをオフにすると、JRun サーバーはアプリケーションとして実行されます。NT サービスと Windows アプリケーションの違いについては、21 ページの「Windows に関する検討事項」を参照してください。

[セットアップ タイプ] ウィンドウが表示されます。



- 13 Java の実行時環境を選択して、[次へ] をクリックします。

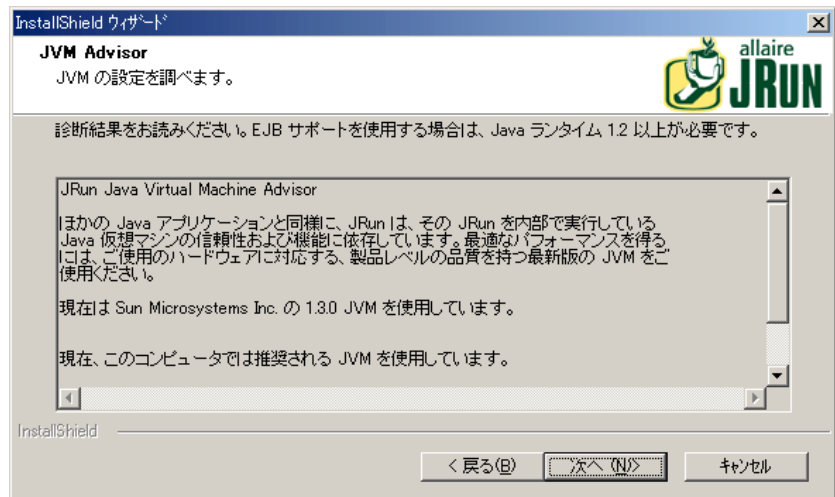
## メモ

EJB を公開するには、JRE だけでなく JDK 1.2 以降も必要です。最新の JDK は、<http://java.sun.com> からダウンロードできます。

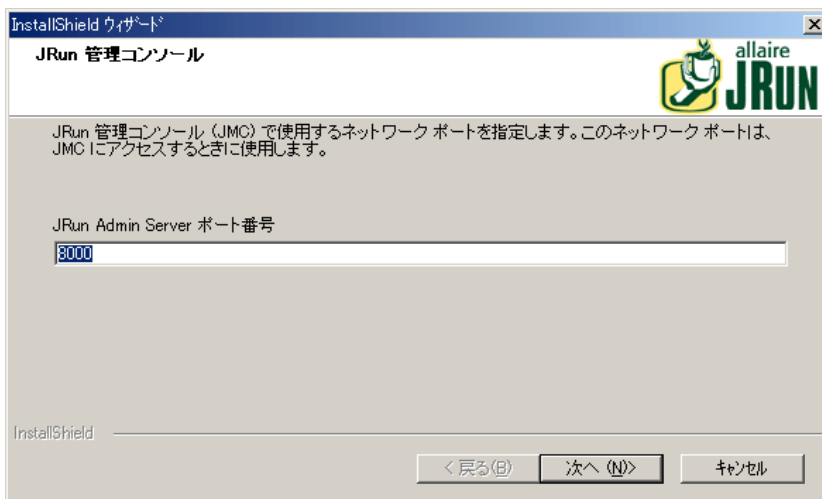
Sun の JRE は、JRun の Windows バージョンに含まれているので、別途 JDK または JRE を用意する必要はありません。JRun から提供された JRE をインストールするには、インストールをキャンセルして、JRun スプラッシュ画面の [Java Runtime Environment 1.3] リンクをクリックします。JRE をインストールしたら、このインストール手順を再開します。

JVM 1.1.8 を使用している場合は、インストールが終了したら [xiii ページの「JVM 1.1.8 に関する特別な検討事項」](#) を参照してください。

[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固有のポート番号を入力し、[次へ]をクリックします。

JRun Web Server (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。推奨する範囲は 8000 ~ 8099 です。

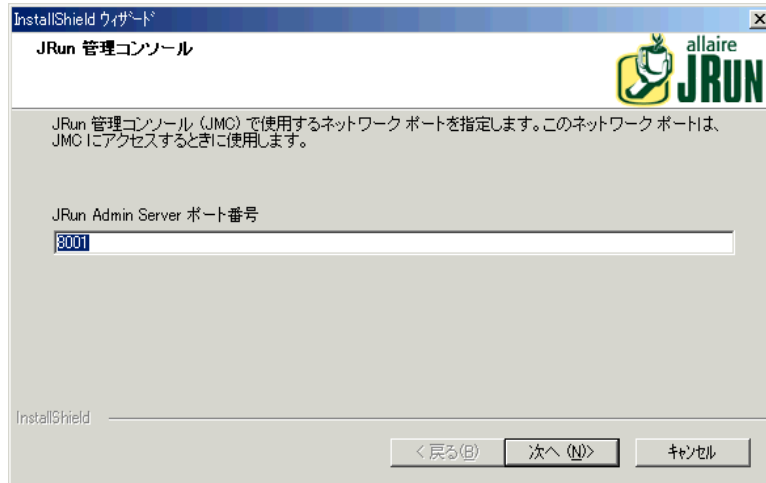
---

#### メモ

8100 ~ 8199 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、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。

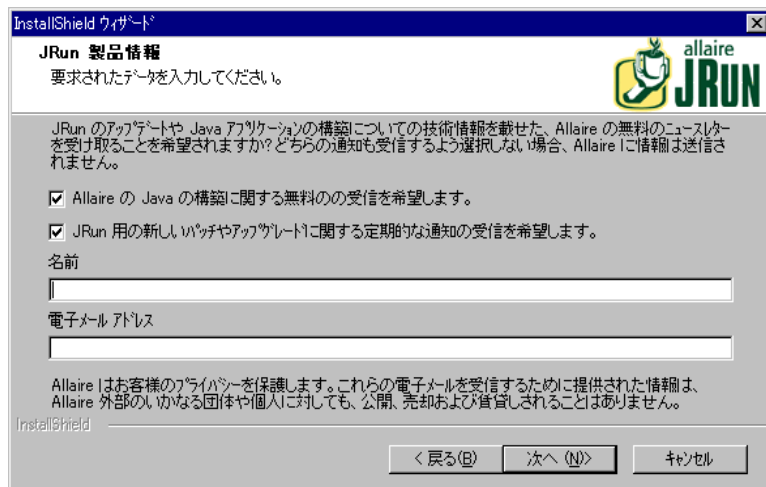


16 JRun 管理者のパスワード (admin) を入力して確認し、[次へ] をクリックします。

## メモ

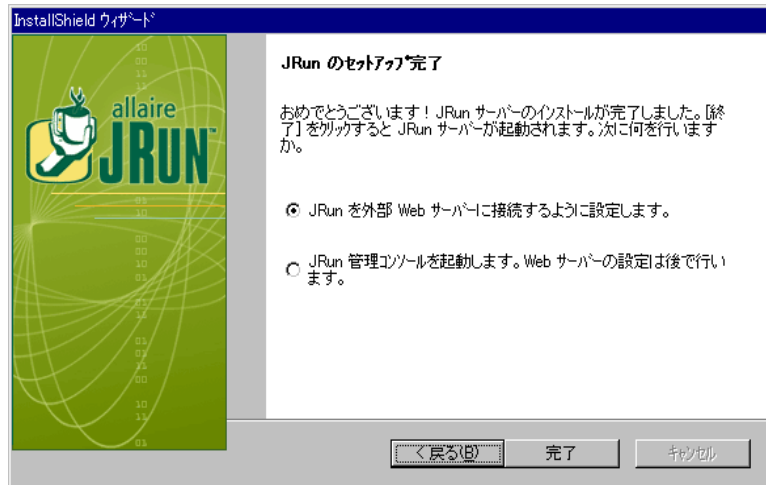
パスワードにはスペースとアスタリスク (\*) は使用できません。

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、[JRun 製品情報] 画面が表示されます。



- 17 オプションでます。Java アプリケーション開発に関する情報や JRun に関する通知の受信が必要な場合は、各チェックボックスをオンにして [次へ] をクリックします。

[JRun のセットアップ完了] 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 ボタンを選択して [終了] をクリックします。JRun 管理コンソールが起動して、ログインするように要求します。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、[15 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 ボタンを選択して [完了] をクリックします。[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 必要な JRE がコンピュータ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、次の Web サイトから取得できます。

<http://java.sun.com>

- 3 次のコマンドを使用して、`jrun-31-unix-jp.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 jrun-31-unix-jp.sh
```
- 4 次のコマンドを使用して、JRun インストール スクリプトを実行します。  

```
% /bin/sh ./jrun-31-unix-jp.sh
```

ライセンス同意書を読むように要求されます。
- 5 **Enter** キーを押して、ライセンス同意書の各ページを表示します。  
ライセンス同意書に同意するように要求されます。
- 6 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に要求されます。
- 7 JRunをインストールするディレクトリを入力します。本書ではこのディレクトリを *JRun* のルート ディレクトリと呼んでいます。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に要求されます。[標準] または [カスタム] のいずれかを選択できます。  
  
[標準]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。[カスタム]を選択すると、次のオプションが表示されます。
  1. サーブレットおよび Java ServerPages
  2. Enterprise JavaBeans および Java Message Service
  3. すべて
- 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に要求されます。
- 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 EJB 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に要求されます。

- 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 版または評価バージョンをインストールする場合、このフィールドは空欄 (既定の設定) 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。以前の 2.x のライセンス キーを入力するように要求されます。  
JRun 管理者のパスワードを入力するように要求されます。

- 11 パスワードを入力します。パスワードにはスペースとアスタリスク (\*) は使用できません。  
ポート番号を入力するように要求されます。
- 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固有のポート番号を入力します。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。推奨する範囲は 8000 ~ 8099 です。

---

#### メモ

8100 ~ 8199 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、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に要求されます。

- 13 これらの情報を受信するかどうかを選択します。

[Yes] を選択した場合は、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 を開くように要求されます。

引き続き設定を行い、JRun を外部 Web サーバーに接続するには、JMC の URL にアクセスします。設定の完了については、[15 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 を起動して、随時 JRun 実装を設定することができます。詳細については、[70 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

JMC は、JRun の設定に使用するブラウザ 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す。JMC を使用するには、Netscape Communicator 4.0 以降、または Internet Explorer 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 を起動できます。

- Web ブラウザで次の URL を開きます。

`http://localhost:8000`

また、Windows の場合には、次のいずれかの操作を実行して JMC を起動できます。

- [スタート] > [プログラム] > [JRun 3.1] > [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* のルート ディレクトリ/bin ディレクトリで次の DOS コマンドを入力します。

```
% jrun -admin
```

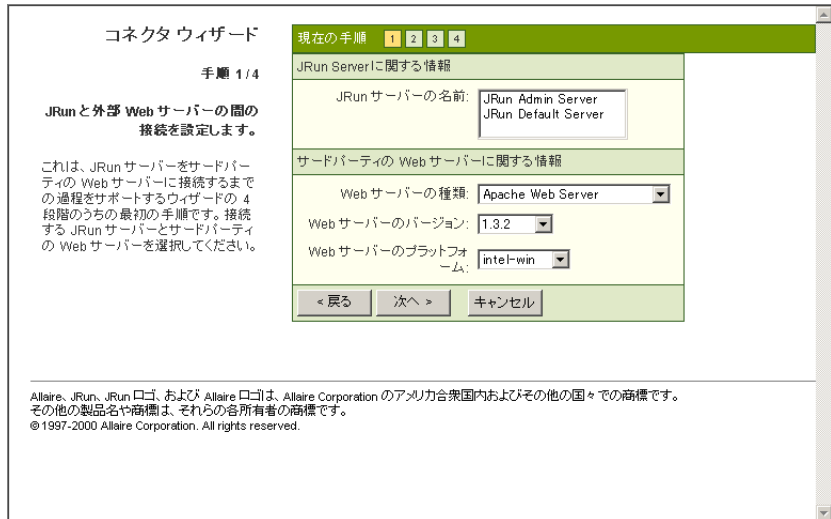
JMC が表示されない場合は、[25 ページの「インストールのトラブルシューティング」](#)を参照してください。JRun のコマンドライン オプションの詳細については、[83 ページの「jrun コマンドの使用」](#)を参照してください。

JMC のログイン ウィンドウが表示されます。



- 2 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ました。

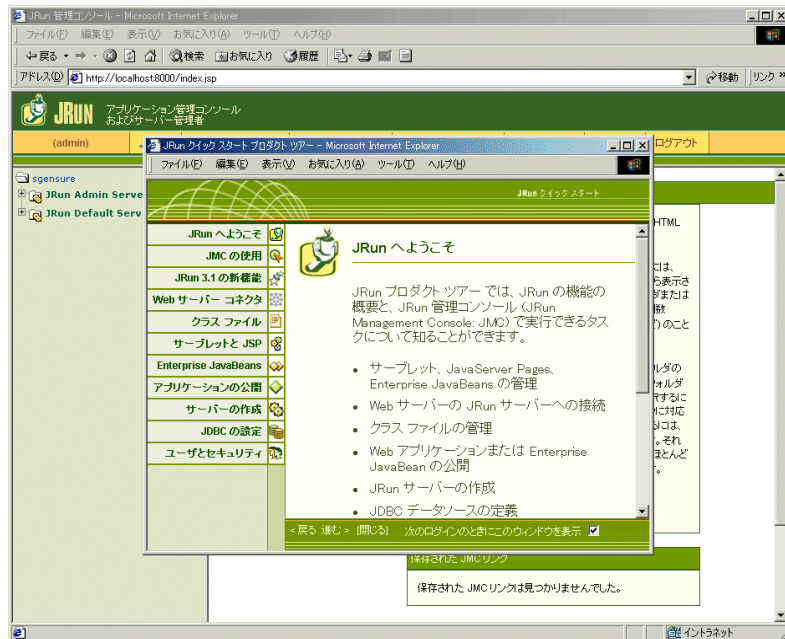
インストール実行中に JMC を起動すると、JRun コネクタ ウィザードが表示されます。



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。

- 「Apache の接続」 31 ページ
- 「IIS 3.0/PWS の接続」 36 ページ
- 「IIS 4.0/5.0 の接続」 39 ページ
- 「Netscape/iPlanet への接続」 47 ページ
- 「WebSite Pro への接続」 54 ページ
- 「Java ベースの Web サーバーの接続」 61 ページ
- 「Zeus Web サーバーの接続」 63 ページ

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手前に表示されます。



JMC を使用した 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

既定では、/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。

次の表は、/JRun の内容を示します。ディレクトリ構造は実装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。

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既定の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 ファイルが含まれています。
/logs	JRun のログ ファイルが含まれています。
/pointbase	組み込み型 Pointbase データベースのファイルが含まれています。JRun のカスタム タグ サンプルおよび EJB サンプルでは、このデータベースを使用します。
/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/ admin	admin JRun サーバーの実行に必要なファイルや、JRun 管理コンソール (JMC) の Web アプリケーションおよび rds-app 用のファイルが含まれています。
/servers/ default	default JRun サーバーの実行に必要なファイルが含まれています。default、demo、および invoice のサンプルアプリケーションも含まれています。
/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やその他の共有ファイルを格納しておくとう便利です。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	既定の 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配列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。

## admin JRun サーバーのサブディレクトリ

次の表では、/servers/admin ディレクトリ内のサブディレクトリについて説明しています。

ディレクトリ	説明
/servers/admin	admin JRun サーバーを定義します。
/servers/admin/deploy	公開する EJB を格納します。公開が完了すると、EJB は起動時に runtime ディレクトリにコピーされます。
/servers/admin/jmc-app	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一時サブディレクトリが含まれています。このディレクトリを削除しないでください。

## default JRun サーバーのサブディレクトリ

次の表では、/servers/default ディレクトリ内のサブディレクトリについて説明しています。JRun サーバーを新規作成した場合、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	default JRun サーバーを定義します。
/servers/default/default-app	既定の 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用して Java サーブレット、JSP、および EJB を作成してテストします。
/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既定のアプリケーションに関連する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 ツリーの一部ではありません。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アクセスすることはできません。このディレクトリには、アプリケーション記述子 web.xml が含まれています。
/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 用のクラス ファイルが格納されます。

ディレクトリ	説明
/servers/default/default-app/WEB-INF/lib	アプリケーションが使用する Bean 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/サーブレット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 EJB 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/invoice-app	fax cover sheet generator および invoice generator を示す invoice demo アプリケーションが含まれています。invoice generator には 3 つの異なるバージョンが含まれており、それぞれ異なるコーディングルールがあります (素の JSP、JavaBeans、および JSP カスタム タグ)。この Web アプリケーションには、使用上の注意とカラーのコード表示が含まれています。
/servers/default/lib	default 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 ファイルおよびクラス 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 実装のためのクラス 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一時サブディレクトリが含まれています。これらの一時ディレクトリを削除しないでください。

## JRun サーバーの使用方法

JRun は、JRun サーバーでほかの機能を起動、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。このセクションは、さまざまな JRun プラットフォームに対応したこれらのユーティリティについて説明します。詳細については、[80 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

Windows NT または 2000 を実行している場合は、インストール時に、JRun サーバーをサービスまたはアプリケーションとして実行するように設定できます。サービスを選択すると、サービスを無効にしない限り、NT システムを開始するたびに JRun サーバーが起動されます。サービスは、ユーザプロセスとしてではなく、システムプロセスとして実行されます。[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マネージャユーティリティを使用して、JRun サーバーの起動、停止、再起動を行うこともできます。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。

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。

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、[83 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、次のいずれかの手順を実行します。**

- Windows

[スタート]>[プログラム]>[JRun 3.1]>[JRun サーバー名]をクリックします。たとえば、admin JRun サーバーを起動するには、[スタート]>[プログラム]>[JRun 3.1]>[admin JRun サーバー]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を使用します。

```
% jrun -start JRun サーバー名
```

---

#### メモ

JRun をサービスとしてインストールした場合、サービスの実行中に JRun をプログラムグループから起動しようとする ([スタート]>[プログラム]>[JRun 3.1]を選択)、 「JRun が異常終了しました」というエラーが発生します。

---

- UNIX

次のいずれかの JRun コマンドライン ユーティリティを使用します。

```
% jrun -nohup JRun サーバー名
```

または

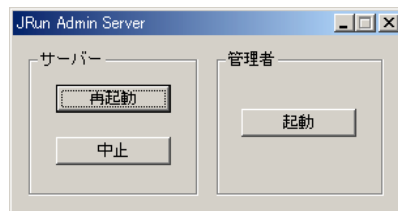
```
% jrun -start JRun サーバー名
```

nohup オプションを使用すると、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、次のいずれかの手順を実行します。**

- 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、システムトレイにある JRun サーバーのアイコンをダブルクリックして、JRun アプリケーションマネージャを開きます。



次に、[中止] 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 コントロール マネージャ ユーティリティを使用して、JRun サーバーを停止します。

- UNIX/Windows

JRun コマンドライン ユーティリティを使用します。

```
% jrun -stop JRun サーバー名
```

**JRun サーバーを再起動するには、次のいずれかの手順を実行します。**

- 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、システムトレイにある JRun サーバーのアイコンをクリックして、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 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 コントロール マネージャ ユーティリティを使用して、JRun サーバーを停止して、再起動します。



- UNIX/Windows

JRun コマンドライン ユーティリティを使用します。

```
% jrun -restart JRun サーバー名
```

または

JMC の左側ペインで [マシン名] 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 [再起動] 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 の使用方法の詳細については、[第 3 章](#)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、既定のポート (8100) にある JRun 供給の Web サーバー上で実行されていることを想定しています。JMC に関連する JWS が (admin JRun サーバ上で) 動作する既定のポートは 8000 です。

実際の運用環境では、JRun サーバーから demo-app の登録を解除し、ファイルシステムにある関連するファイルを削除する必要があります。詳細については、[130 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、[21 ページ](#)の「[JRun サーバーの起動と停止](#)」の手順に従って default JRun サーバーを起動します。

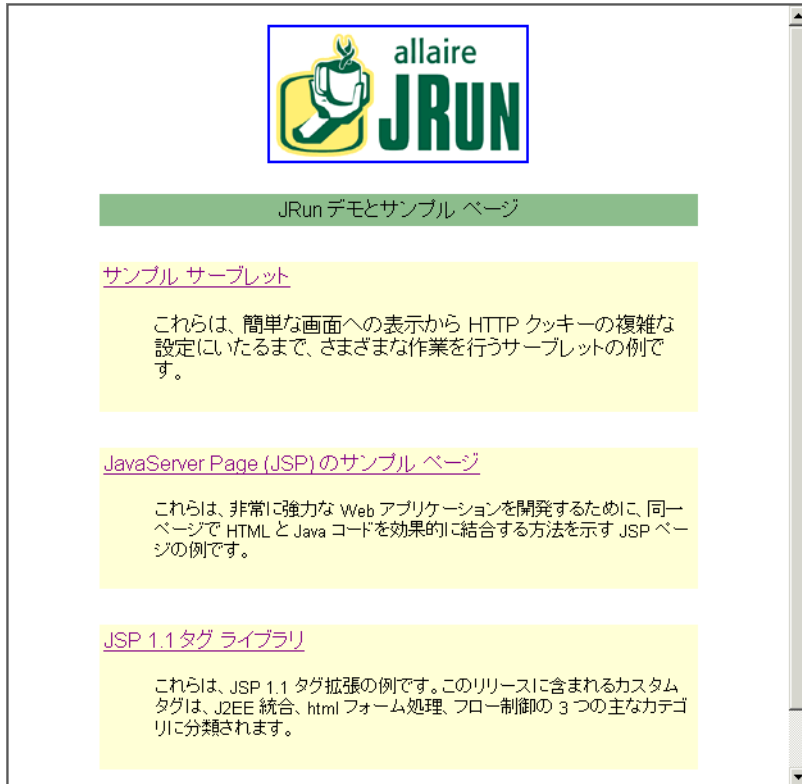
- 2 Web ブラウザで、次の URL を開きます。

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

または (Windows のみ)

[スタート] > [プログラム] > [JRun 3.1] > [JRun Demo] をクリックします。

JRun のデモとサンプルのページが表示されます。



## インストールの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* のルート ディレクトリ/logs にあるログ ファイルをチェックすると、詳細情報を得ることができます。

## JMC のエラー

### ログイン ページが正しく開かない

- 要求 URL のポート番号を確認してください。admin JRun サーバーの既定のポート (8000) を、インストール中に上書きしてしまった可能性があります。その場合は、その新しい値を使用します。不明な場合は、*JRun* のルート ディレクトリ/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。JRun 3.1 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、2 ページの「JRun のインストール」を参照してください。
- JRun 3.1 を前のバージョンの上にインストールする場合は、必ず、新しいバージョンをインストールする前に、前のバージョンのすべてのファイルを削除してください。特に、jrun\_jsp.jar や jrunadmin.jar などの、前の JAR ファイルに注意してください。前のファイルを完全に削除するには、Web サーバーや JRun サーバーを停止する必要があります。これを行わないと、JMC を開く時に「Incompatible object argument for function call (関数呼び出しに互換性のないオブジェクト引数があります)」というエラーが発生することがあります。
- プロパティファイルを変更したら、手作業で、または JMC を使用して、JRun サーバーを再起動します。
- JRun admin 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は、admin JRun サーバーの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マネージャユーティリティをチェックして、admin JRun サーバーのサービスが実行されているかどうかを確認します。
  - UNIX の場合は、/opt/jrun/bin で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。

## デモ アプリケーションのエラー

### デモ アプリケーションが開かない

- 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`

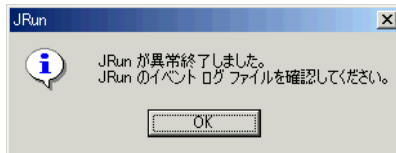
**default JRun** サーバーの既定のポートは 8100 です。しかし、インストール時にこのポート番号を使用する別のプロセスが検出されると、**JRun** は、空きポートが見つかるまでポート番号を増やしていきます。**default JRun** サーバーの既定のポートを、インストール中に書き替えてしまった可能性があります。その場合は、その新しい値を使用します。不明な場合は、*JRun* のルート ディレクトリ `/servers/default/local.properties` ファイルの「**Web Services**」セクションの `web.endpoint.main.port` の値をチェックします。

**default** サーバーがどのポートで実行されているかは、**JMC** の **JRun Web** サーバーパネルで確認できます。詳細については、[117 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は、**default** サーバーの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 マネージャ ユーティリティをチェックして、**default JRun** サーバーのサービスが実行されているかどうかを確認します。
  - **UNIX** の場合は、`/opt/jrun/bin` で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] リンクをクリックして、デモ アプリケーションを **JMC** から起動してみてください。

## Windows での JRun のエラー

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



JRun サーバーをアプリケーションとして起動する場合は、JRun がすでにサービスとして実行されていないかどうかを確認してください。[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マネージャユーティリティを使用して JRun サービスを起動します。ただし、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] メニューから起動します。

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 の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。さらに、サービスコントロールマネージャユーティリティで、それらのサービスがサービスとして実行されているかどうかをチェックし、admin JRun サーバーおよび default JRun サーバーを探します。

### ログオフすると Windows NT サービスが停止する

Sun 1.3 JRE および IBM 1.3 JVM には、ユーザが Windows NT/2000 からログオフすると、NT サービスとしてインストールした JRun が停止するというバグが含まれています。JRun サポート チームでは、この問題の回避策について説明した知識ベースを作成しています

(<http://www.allaire.com/Handlers/index.cfm?ID=19697&Method=Full>)。

このバグは、Sun JDK 1.3.1 ベータ リリースで修正されています。



## 第 2 章

# JRun の外部 Web サーバーへの 接続

この章では、JRun を Web サーバーと接続する方法と、JRun を使用して Web 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、JMC を使用して JRun を Web サーバーに接続しなければならない場合があります。

特定の Web 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 を外部サーバーとのプラグインとして接続しない場合は、この章を読む必要はありません。

---

### 目次

- [接続の概要](#) ..... 30
- [Apache の接続](#) ..... 31
- [IIS 3.0/PWS の接続](#) ..... 36
- [IIS 4.0/5.0 の接続](#) ..... 39
- [Netscape/iPlanet への接続](#) ..... 47
- [WebSite Pro への接続](#) ..... 54
- [Java ベースの Web サーバーの接続](#) ..... 61
- [サーブレットの実行用 CGI インターフェイスの構成](#) ..... 62
- [Zeus Web サーバーの接続](#) ..... 63
- [local.properties への変更](#) ..... 66
- [コネクタのトラブルシューティング](#) ..... 66

## 接続の概要

JRun 3.1 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、既存の Web サーバーに Web 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、JRun は、統合 JRun Web Server (JWS) を使用して機能します。プラグイン モジュールの場合は、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場合は、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、ほとんどの場合、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) を接続することになります。

JRun コネクタについての詳細、分散環境での JRun の設定方法については、[第 4 章](#)を参照してください。

### JRun を外部 Web サーバーに接続するには

- 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 JMC を開始します。
- 4 JRun コネクタ ウィザードを実行して、Web サーバーと default JRun サーバー間の通信を容易にする JRun 接続モジュール (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 コネクタは、外部 Web サーバーへの要求を阻止し、Web サーバーに渡すか JRun で処理するかを決めるフィルタです。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* のルート ディレクトリ/`connectors/src` にあります。詳細については、[194 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 の接続

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。Web サーバーの高度な接続方法については、[第 4 章](#)を参照してください。

JRun では、使用するオペレーティングシステムに基づいて、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) モジュールとスタティックモジュールという 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 ベースのシステムでは、Apache は DSO モジュールのみをサポートします。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。Red Hat Linux 5.2 など一部のプラットフォームでは、DSO サポート付きであらかじめビルドされた Apache が提供されるため、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、JRun をスタティックモジュールとしてサーバーにコンパイルできますが、これは 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。

ここでは、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します。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は、このセクションを読む必要はありません。

詳細については、[35 ページ](#)の「[Raven と JRun の併用](#)」を参照してください。

### 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、スタティックモジュールを構成します。DSO を使用していない場合にスタティックモジュールを構成する場合のみ、この手順(および詳細手順)を実行します。

a JRun ソース ファイルを *JRun* のルート ディレクトリ/`connectors/apache/src` から Apache の `/src/modules/jrun` ディレクトリにコピーします。

スタティック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 :

b 次の行を `/src` ディレクトリの Apache コンフィギュレーションファイルに追加します。

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

c 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 :

d 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。

e `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
```

また、Windows の場合は、次の操作によって JMC を起動することもできます。

[スタート] > [プログラム] > [JRun 3.1] > [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。

---

5 JRun 管理者として 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に必要な構成情報を指定します。

手順	パラメータ	説明
手順 1	JRun サーバー名	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。 admin JRun サーバーは固有の 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 サーバーのコネクタポート	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。既定値は 51000 です。
手順 3	Apache の conf ディレクトリ	コンフィギュレーションファイル srm.conf および httpd.conf が含まれている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	<b>UNIX</b> : 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。 <b>Windows</b> : [DSO サポート] を選択します。

- 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は、次の手順を実行します。

- Windows :

JRun をアプリケーションとしてインストールした場合は、[スタート]>[プログラム]>[JRun 3.1]>[default JRun サーバー]をクリックします。

JRun をサービスとしてインストールした場合は、[コントロールパネル]からサービス コントロール マネージャ ユーティリティを開いて「default JRun 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 ユーティリティを使用します。

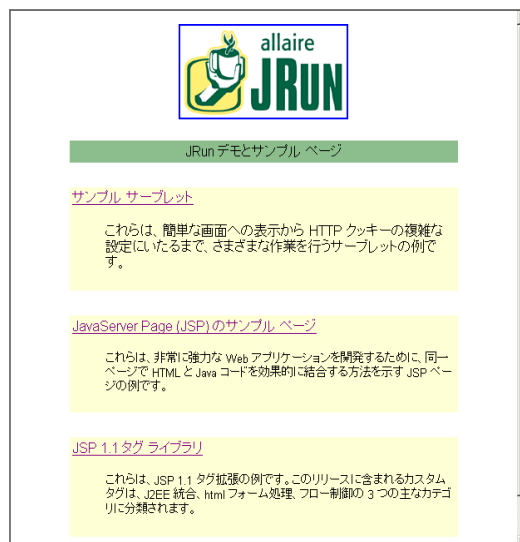
```
% jrun -start default
```

- 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
```

この手順は、Apache が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



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しく構成されています。デモ アプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。

## Raven と JRun の併用

Raven は SSL を使用できる 市販の Apache Web サーバー製品です。Raven を JRun に接続するには、このセクションで説明する手順に従う必要があります。この手順を使用しないと、コネクタ ウィザードの手順 3 で既定の Apache conf ディレクトリを入力した後、「httpd.conf にアクセスできません」というエラーが発生する場合があります。

コンフィギュレーション ファイルについては、Raven では Apache 標準の httpd.conf ファイルではなく httpsd.conf という名前のファイルが使用されます。コネクタ ウィザードでは httpd.conf というファイルを想定しています。そのため、JRun コネクタを使用して JRun を Apache に接続するには、コンフィギュレーションファイルの名前を変更する必要があります。

### JRun を Raven に接続するには

- 1 httpsd.conf の名前を httpd.conf に変更します。
- 2 [31 ページの「Apache の接続」](#)の手順に従って、コネクタ ウィザードを実行します。
- 3 httpd.conf の名前を httpsd.conf に戻します。

## Apache コンフィギュレーション 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 the following 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 ウィザードでは、入力した内容に基づいて JRun local.properties ファイルも変更されます。詳細については、[66 ページの「local.properties への変更」](#)を参照してください。

JRun コネクタ ウィザードで使用するファイルを変更しないでください。

## IIS 3.0/PWS の接続

このセクションでは、Windows 95/98/NT システムで実行する IIS 3.0 または Personal Web Server (PWS) に JRun を接続する方法について説明します。Web サーバーの高度な接続方法については、第 4 章を参照してください。

ここでは、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します。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は、このセクション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 3.1] > [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 コネクタウィザードで次の構成情報を指定します。

コネクタ ウィザード の手順	パラメータ	説明
手順 1	JRun サーバー名	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。 admin JRun サーバーは固有の 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 サーバーのコネクタポート	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。既定値は 51000 です。
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オンにすると、JRun に ISAPI フィルタがインストールされ、HTTP 要求でサーブレットを実行しようとしているかどうかを検出されます。

- 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。

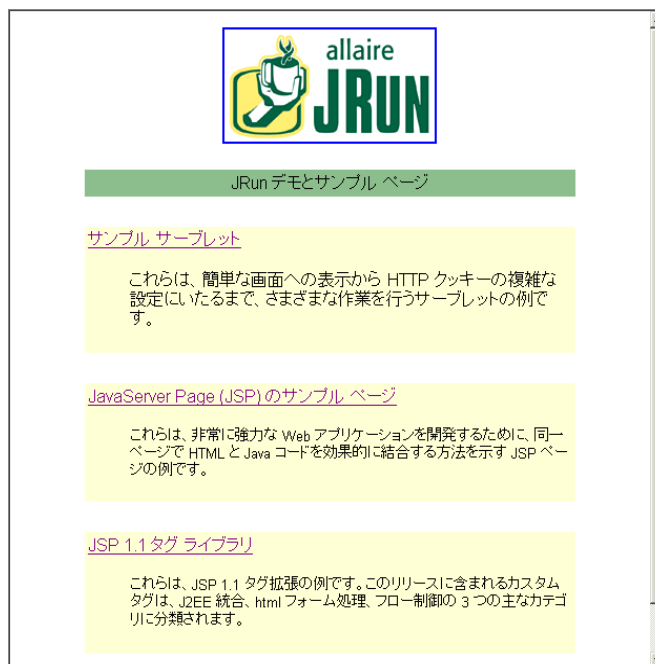
default JRun サーバーが起動されていない場合は、[スタート]>[プログラム]>[JRun 3.1]>[default JRun サーバー]をクリックします。

- 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`

この手順は、IIS/PWS が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



デモ アプリケーションが実行される場合は、JRun と IIS/PWS Web サーバーの接続は正しく構成されています。デモ アプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。



## コンフィギュレーション ファイルへの変更

コネクタ ウィザードによって、Web サーバーは次のように変更されます。

- `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 では `.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 要求を阻止し、JRun の処理に適切な HTTP 要求を 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] を選択すると、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、[44 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、[66 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。

## IIS 4.0/5.0 の接続

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に JRun を構成できます。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、次のことを説明します。

- [「JRun の IIS 4.0/5.0 への接続」 40 ページ](#)
- [「IIS コンフィギュレーションファイルへの変更」 44 ページ](#)
- [「JRun ISAPI フィルタの構成」 44 ページ](#)

ここでは、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します。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は、このセクションを読む必要はありません。

Web サーバーの高度な接続方法については、[第 4 章](#)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 マネージャ ユーティリティで、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] をクリックします。

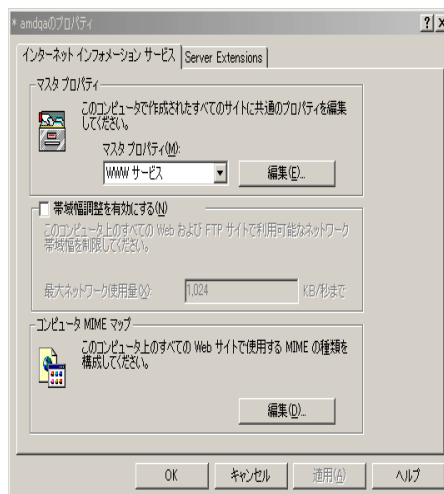
Microsoft 管理コンソール (MMC) が表示され、iis.msc スナップインが開きます。

Windows 2000 : [スタート] > [プログラム] > [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。

プロパティのウィンドウが表示されます。



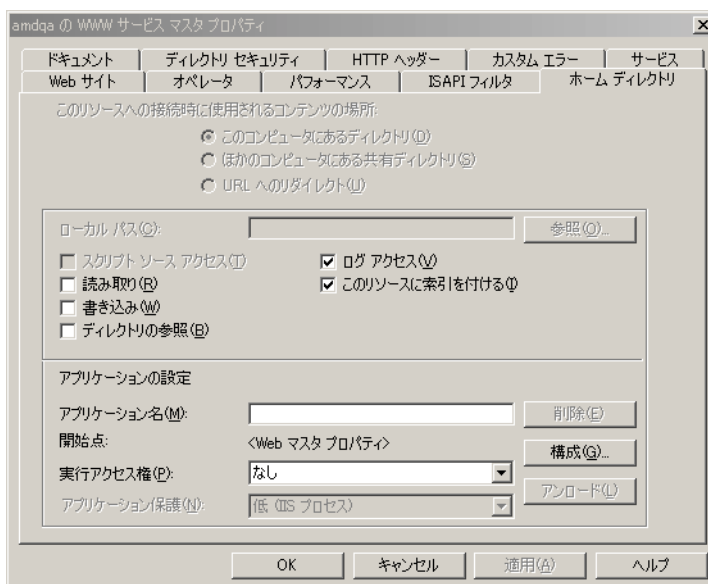
- 4 [マスタプロパティ] ドロップダウン リスト ボックスから [WWW サービス] を選択して、[編集] をクリックします。

WWW サービス マスタプロパティ アプリケーションが表示されます。

- 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] フィールドに指定されているディレクトリの実行許可を設定します。ディレクトリが指定されていない場合は、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を設定します。

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

Windows 2000 : [実行アクセス権] ドロップダウン リスト ボックスで、[スクリプトおよび実行可能ファイル] をクリックします。



- 7 [OK] をクリックして、この設定内容を適用します。[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 3.1] > [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 パーの [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。

コネクタ ウィザード の手順	パラメータ	説明
手順 1	JRun サーバー名	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。 admin JRun サーバーは固有の 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 サーバーのコネクタ ポート	JRun サーバーで 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。既定値は 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 フィルタとしてのインストール	このチェック ボックスをオンにすると、JRun に ISAPI フィルタがインストールされ、HTTP 要求でサーブレットを実行しようとしているかどうかを検出されます。

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は、次の手順を実行します。

- JRun をアプリケーションとしてインストールした場合は、[スタート] > [プログラム] > [JRun 3.1] > [default JRun サーバー] をクリックします。
- JRun をサービスとしてインストールした場合は、[コントロール パネル] から サービス コントロール マネージャ ユーティリティを開いて「default JRun 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。

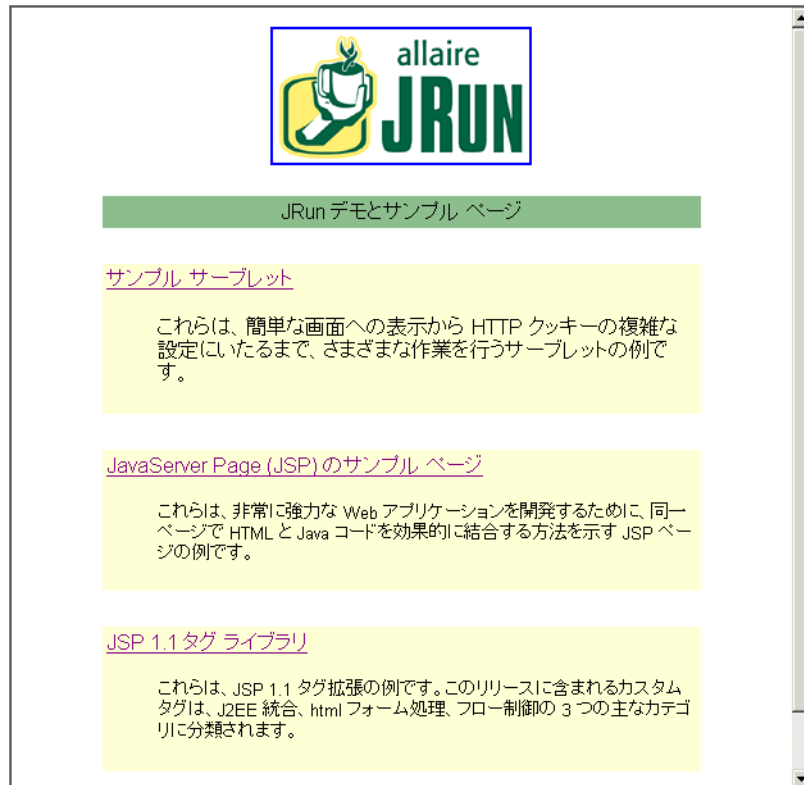
```
% jrun -start default
```

- 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`

この手順は、IIS が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



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しく構成されています。デモアプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。

## IIS コンフィギュレーション 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 の実装を変更します。

- `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 では `.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 要求を阻止し、JRun が処理するために適切な HTTP 要求を 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 によってメタベースに加えられる変更を次に示します。
  - `jrun.dll` の絶対パスを `ScriptMaps` パラメータに追加します。`ScriptMaps` によって、ファイル名拡張子が処理用の DLL にマッピングされます。
  - [グローバル フィルタとしてインストールする] を選択した場合は、コネクタ ウィザードによって、[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がメタベースのフィルタ オブジェクトに追加されます。
- JRun `local.properties` ファイルを更新します。  
詳細については、[66 ページの「local.properties への変更」](#)を参照してください。

JRun コネクタ ウィザードで使用され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。

## JRun ISAPI フィルタの構成

IIS または PWS で HTTP 要求を受信する場合に、ISAPI フィルタでイベントに応答することができます。JRun コネクタ 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ほかの ISAPI フィルタに追加される DLL がインストールされます。`jrun.dll` ファイルの既定の場所は `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 フィルタを変更する必要はありません。ただし、ほかの ISAPI フィルタをインストールする場合は、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の追加、削除、および変更を行うことができます。

### JRun ISAPI フィルタを編集するには


- 1 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] をクリックします。

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] > [プログラム] > [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 マネージャが表示されます。

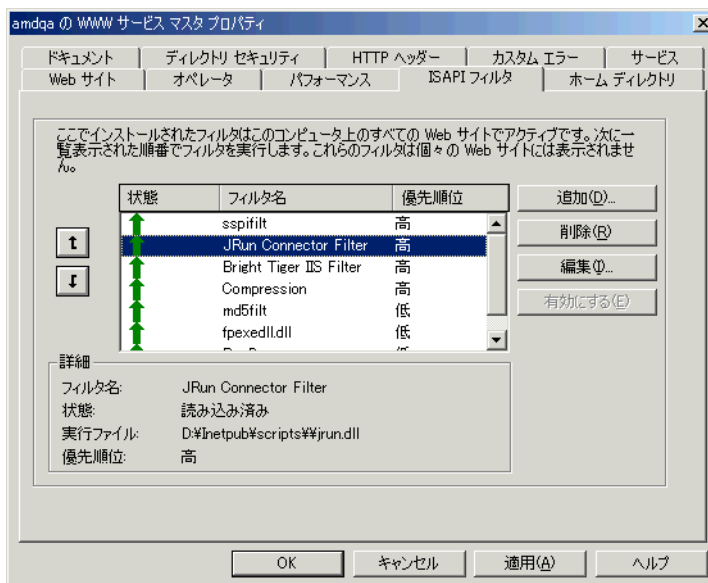
- 2 マシン名  の上でマウスを右クリックして、[プロパティ] を選択します。

プロパティの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 サービス マスタプロパティアプリケーションが表示されます。

- 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 Connector Filter] 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 Connector Filter] 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、フィルタがほか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、次の手順を実行します。

### JRun ISAPI フィルタの優先順位を変更するには

- 1 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] をクリックします。  
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] > [プログラム] > [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 マネージャが表示されます。
- 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。  
プロパティの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 サービス マスタプロパティ アプリケーションが表示されます。



- 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 Connector Filter] を選択します。
- 6 上向矢印をクリックして、JRun フィルタを一覧のほかのフィルタの前に移動します。
- 7 [OK] をクリックして、この変更を適用します。
- 8 Web サーバーを再起動します。

## Netscape/iPlanet への接続

このセクションでは、Netscape Web サーバーの構成方法について説明します。Web サーバーの高度な接続方法については、[第 4 章](#)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、[50 ページの「Java インタプリタの有効化」](#)を参照してください。

---

ここでは、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します。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は、このセクション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 の場合には、[スタート]>[プログラム]>[JRun 3.1]>[JRun 管理コンソール]をクリックして JMC を起動することもできます。

---

### 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 管理者として JMC にログインします。
- 4 アクセス パーの [コネクタ ウィザード] を選択します。

- 5 コネクタウィザードで次の構成情報を指定します。

コネクタ ウィザード の手順	パラメータ	説明
手順 1	JRun サーバー名	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 JRun サーバーを選択します。admin JRun サーバーは固有の 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	[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 サーバーのコネクタポート	JRun サーバーで 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。既定値は 51000 です。
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 (既定) 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。詳細については、 <a href="#">50 ページの「Java インタプリタの有効化」</a> 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は、次の手順を実行します。

- Windows :

JRun をアプリケーションとしてインストールした場合は、[スタート]>[プログラム]>[JRun 3.1]>[default JRun サーバー]をクリックします。

JRun をサービスとしてインストールした場合は、[コントロールパネル]からサービス コントロール マネージャ ユーティリティを開いて「default JRun 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。

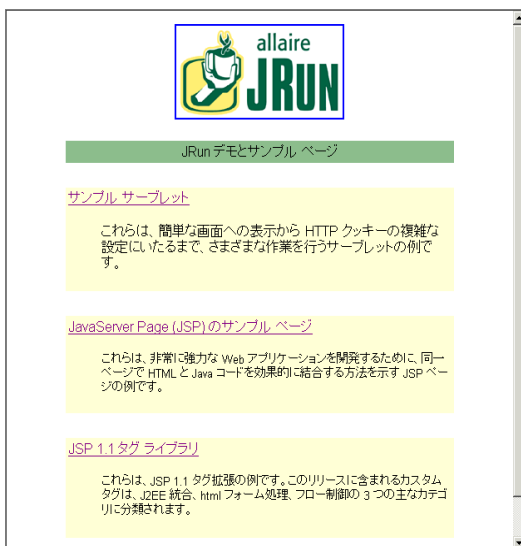
```
% jrun -start default
```

- 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 NES/iPlanet Web サーバーの接続を確認します。

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

この手順は、NES/iPlanet が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



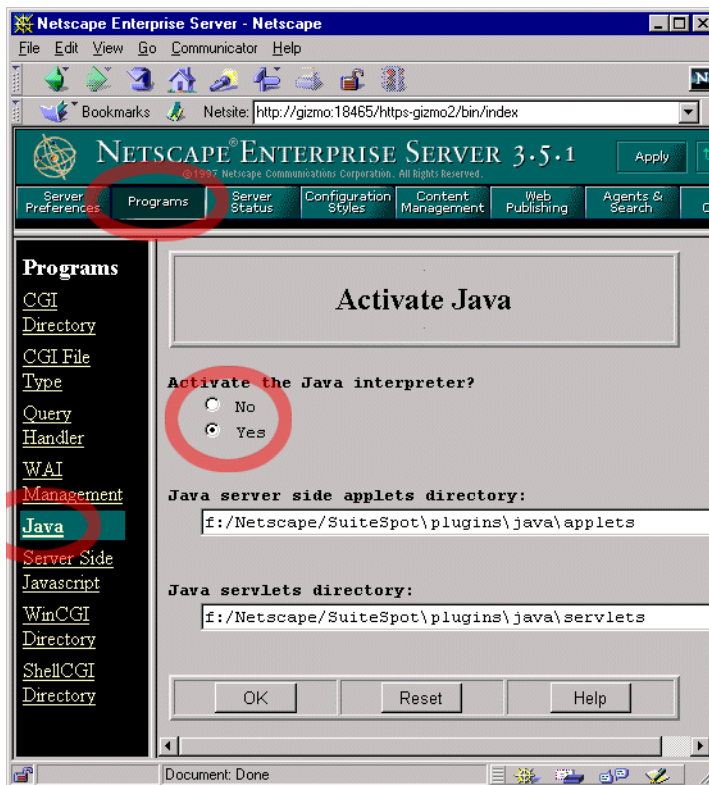
デモ アプリケーションが実行されている場合は、JRun と NES/iPlanet Web サーバーの接続は正しく構成されています。デモ アプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。

## 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できます。コネクタが「使用可能である」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 [Programs] 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、[47 ページの「Netscape/iPlanet への接続」](#)で説明します。

## NES コンフィギュレーション ファイルへの変更

JRun コネクタ ウィザードでは、Netscape/Server4/https- マシン名/config/ ディレクトリの `obj.conf` ファイルを変更します。次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 ウィザードでは、JRun `local.properties` ファイルも変更します。詳細については、[66 ページの「local.properties への変更」](#)を参照してください。

JRun コネクタ ウィザードで使用するファイルを変更しないでください。このセクションの説明は、情報だけを提供しています。

`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、[53 ページ](#)で説明しています。`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、`obj.conf` ファイルは JRun によって次のように変更されます。

- 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 要求を阻止して、JRun での処理に適切な HTTP 要求を 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ディレクティブのグループ化です。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>
```
- 次のディレクティブを既定のオブジェクト 定義に追加します。

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

`NameTrans` ディレクティブは、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` ディレクティブは、`jrunfilter` の部分パスを指定するため、サーバーは `PathCheck` ディレクティブ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 で JRun が無効にされないように、`/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化します。

```
#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、obj.conf ファイルには JRun によって次の変更が加えられます。

- init クラスパスの前に 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ディレクティブのグループ化です。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 で JRun が無効にされないように、/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化します。

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

- 次のディレクティブ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 ディレクティブは、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 ディレクティブは、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
# バス名にはフォワード スラッシュを使用します。バックスラッシュを使用すると
# エラーが発生します。詳細については、マニュアルを参照してください。

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">
Namedtrans 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 への接続

このセクションでは、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 と WebSite Pro が通信するには、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。Web サーバーの高度な接続方法については、[第 4 章](#)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た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、次の手順を実行します。

- 「[サブレットを実行するための URL 接頭辞のマッピング](#)」 [54 ページ](#)
- 「[マルチホームおよび URL 接頭辞](#)」 [56 ページ](#)
- 「[ファイル拡張子の JRun へのマッピング](#)」 [57 ページ](#)
- 「[WebSite Pro と通信するための JRun の構成](#)」 [58 ページ](#)

ここでは、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します。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は、このセクションを読む必要はありません。

## サブレットを実行するための URL 接頭辞のマッピング

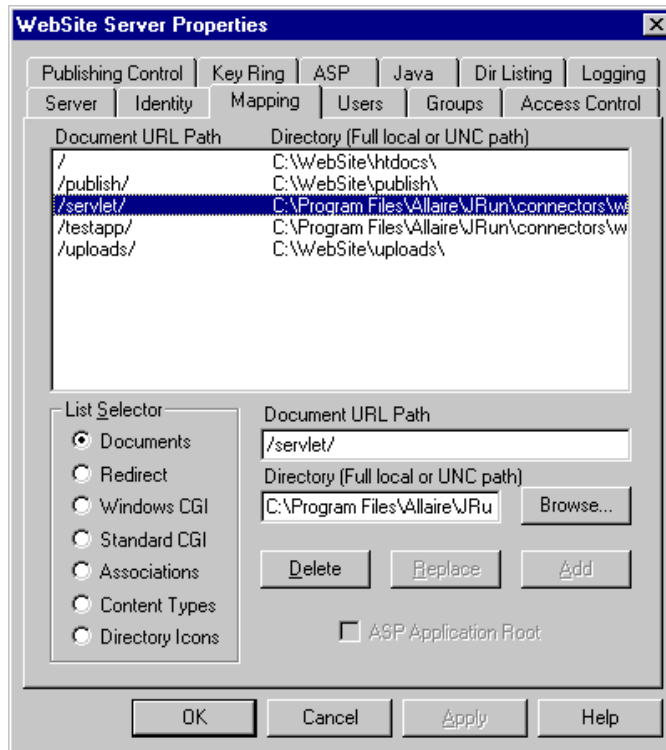
WebSite Pro では、サブレットを実行するために URL 接頭辞をマッピングするなど、サーバーの構成を高度にカスタマイズすることができます。たとえば、`http://yourhost.com/servlet/SampleServlet` 経由でサブレットを実行するには、`¥servlet¥` 用の Documents マッピングを WebSite Pro に追加する必要があります。

### URL 接頭辞を 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 [Mapping] タブをクリックします。



- 3 [Document URL Path] および [Directory] フィールドを編集します。この手順により、`¥servlet¥` サブレット名 URL 経由でサブレットを実行できます。

たとえば、[Document URL Path] フィールド内の `¥servlet¥` マッピングで、[Directory] フィールド内の `C:¥ProgramFiles¥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 [Directory] 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、[56 ページの「マルチホームおよび URL 接頭辞」](#) で説明し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。

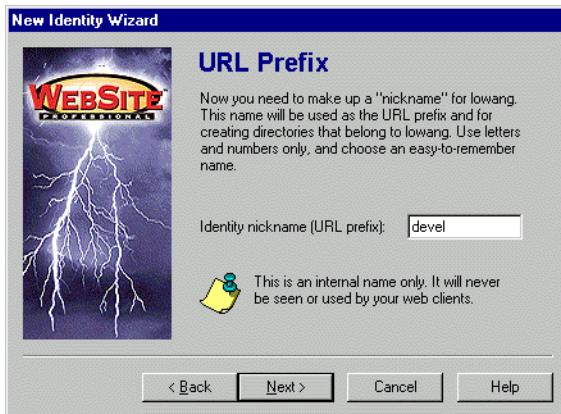
- 4 特定のサブレットを URL にマッピングする場合は、ここで説明した手順と同じ手順を使用しますが、サブレット名をパスの最後に追加します。次に例を示します。

```
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。

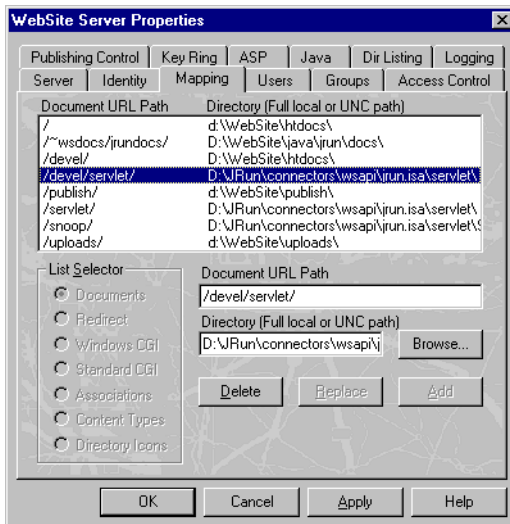
## マルチホームおよび URL 接頭辞

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 接頭辞] 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] に「/devel/servlet/」と入力します。



## ファイル拡張子の JRun へのマッピング

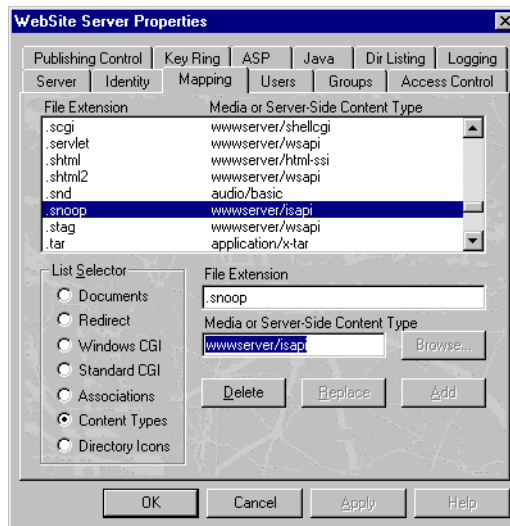
指定した拡張子で JRun を起動できるように WebSite サーバーを構成するには、2つの手順が必要です。最初に、WebSite Server Properties アプリケーションを使用して、WebSite をセットアップする必要があります。次に、JRun 管理コンソールを使用して、そのマッピングを JRun に追加します。

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、[148 ページの「サーブレットへの要求マッピング」](#)を参照してください。

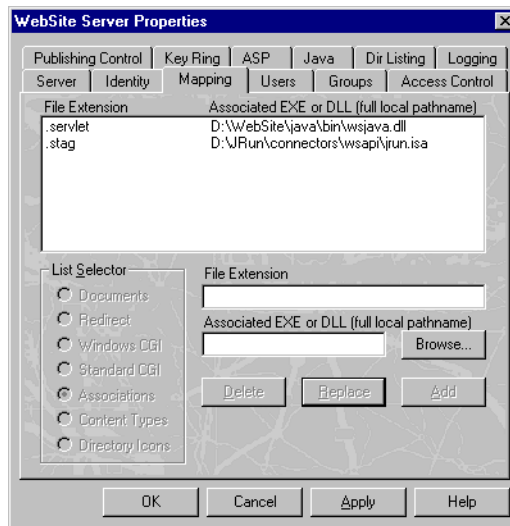
### ファイル拡張子のマッピングを追加するには

- 1 WebSite Server Properties アプリケーションを起動します。
- 2 [Mapping] 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TRIESを追加します。

次の図は、.snoop 拡張子の例を示します。



- 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 3.1] > [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 コネクタウィザードで次の構成情報を指定します。

コネクタ ウィザードの 手順	パラメータ	説明
手順 1	JRun サーバー名	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。 admin JRun サーバーは固有の 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 サーバーのコネクタポート	JRun サーバーで 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。既定値は 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。

default JRun サーバーがまだ起動されていない場合は、次の手順を実行します。

- JRun をアプリケーションとしてインストールした場合は、[スタート]>[プログラム]>[JRun 3.1]>[default JRun サーバー]をクリックします。
- JRun をサービスとしてインストールした場合は、[コントロール パネル]からサービス コントロール マネージャ ユーティリティを開いて「default JRun 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。

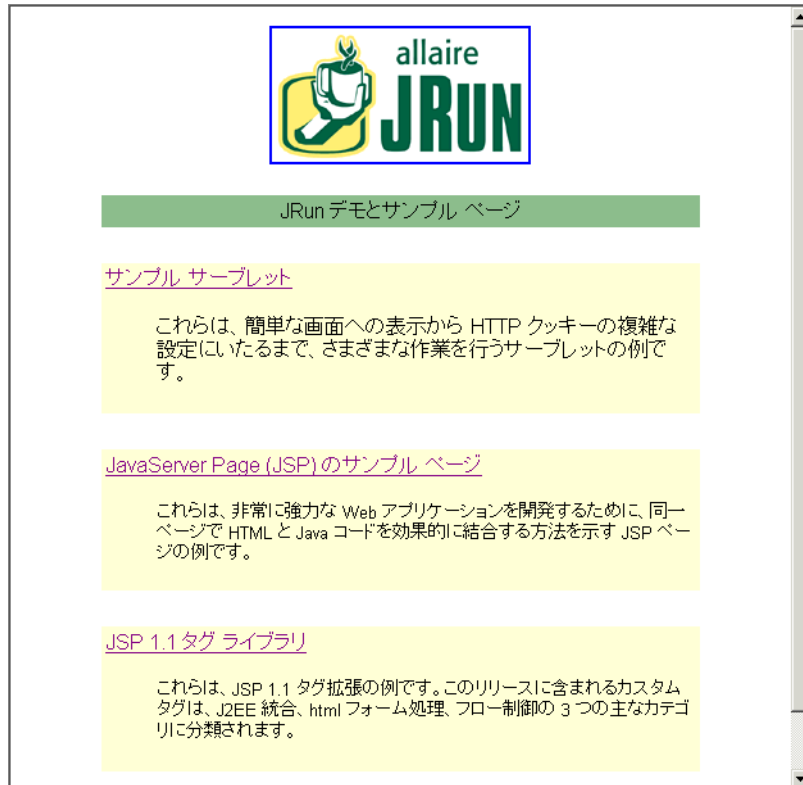
```
% jrun -start default
```

- 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`

この手順は、WebSite Pro が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



デモ アプリケーションが実行されている場合は、JRun と WebSite Pro Web サーバーの接続は正しく構成されています。デモ アプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。

## WebSite Pro コンフィギュレーション ファイルへの変更

JRun コネクタ ウィザードでは、WebSite Pro Web サーバーのコンフィギュレーション 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を変更する場合は、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 ウィザードでは、JRun `local.properties` ファイルも変更します。詳細については、66 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 サーバーの接続

JRun では、本書で特定のサーバーに関する手順を説明しているかどうかにかかわらず、大部分の Java Web サーバーと通信できます。ここでは、この章のほかのセクションで詳しく説明していない 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。Web サーバーの高度な接続方法については、[第 4 章](#)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 次に示す 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。JRun に異なるポートを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 要求を JRun にマッピングします。次に例を示します。

```
/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 に用意されている `jrun.pl` Perl スクリプトを使用します。このスクリプトは、*JRun* のルート ディレクトリ `/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 からサーブレットが呼び出されます。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` スクリプトによって、JRun サーバーとの接続方法を確認するために `JRUNPROXY` 環境変数が検索されます。この値には、次の形式を使用します。

```
IP_address:port
```

既定値は `127.0.0.1:51000` です。

このスクリプトは、特定の環境向けにカスタマイズできます。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できます。



## Zeus Web サーバーの接続

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、次の手順を実行します。Web サーバーの高度な接続方法については、[第 4 章](#)を参照してください。

ここでは、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します。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は、このセクション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` には Web サーバーのホストの IP アドレスを設定します。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` には Zeus と JRun が通信するポートを設定します。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 3.1] > [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に必要な構成情報を指定します。

コネクタ ウィザード の手順	パラメータ	説明
手順 1	JRun サーバー名	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。 admin JRun サーバーは固有の 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 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。
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 サーバーのコネクタ ポート	JRun サーバーで 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。既定値は 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は、次の手順を実行します。

- JRun をアプリケーションとしてインストールした場合は、[スタート]>[プログラム]>[JRun 3.1]>[default JRun サーバー]をクリックします。
- JRun をサービスとしてインストールした場合は、[コントロール パネル]からサービス コントロール マネージャ ユーティリティを開いて「default JRun 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。

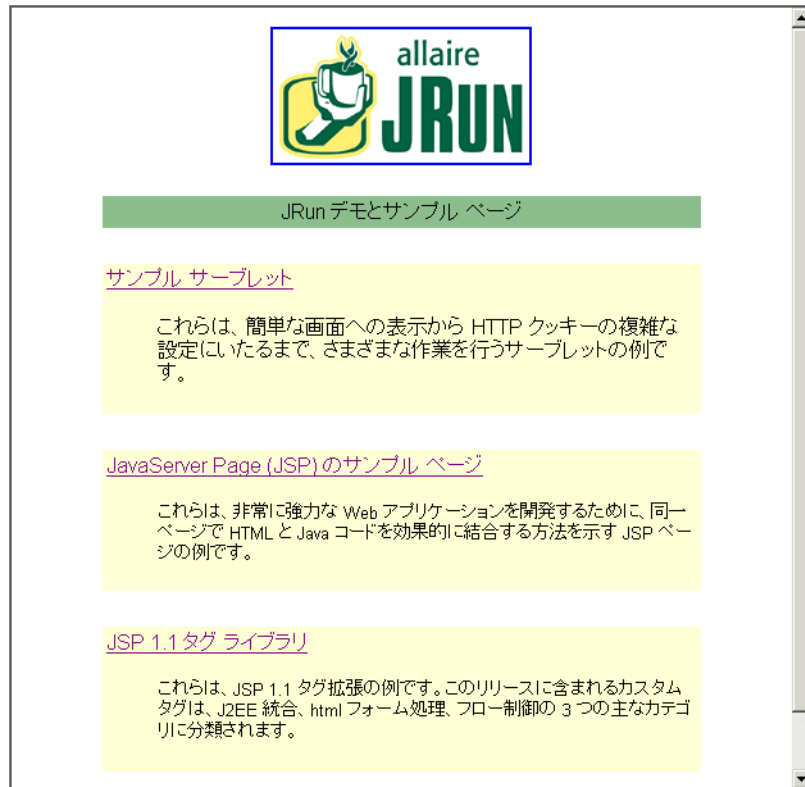
```
% jrun -start default
```

- 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`

この手順は、Zeus が既定のポート 80 で接続を受信していることを想定しています。

JRun のデモとサンプルのページが表示されます。



デモ アプリケーションが実行されている場合は、JRun と Zeus Web サーバーの接続は正しく構成されています。デモ アプリケーションが正しく実行されない場合は、66 ページの「コネクタのトラブルシューティング」を参照してください。

## Zeus コンフィギュレーション ファイルへの変更

JRun コネクタ ウィザードでは、Zeus Web サーバーのコンフィギュレーション 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を変更する場合は、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 ウィザードでは、JRun `local.properties` ファイルも更新します。詳細については、66 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 サーバーのコンフィギュレーションファイルのほかに、JRun コネクタ ウィザードでは、次のように JRun サーバーの local.properties ファイルも変更します。

- 次の行を local.properties の最後に追加して、JRun コネクタが正しく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 ウィザードの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呼ばれます。

## コネクタの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する一般的な問題の解決に役立つ情報を提供します。詳細については、[25 ページの「インストール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ときは、*JRun* のルート ディレクトリ/logs にあるログファイルをチェックすると、詳細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：

「obj.conf file 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 サーバーのコンフィギュレーション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 を Raven に接続しているときに次のエラーが発生した場合

「httpd.conf にアクセスできません」

httpd.conf ファイルの名前を一時的に変更する必要があります。詳細については、[35 ページの「Raven と JRun の併用」](#)を参照してください。

## 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ら、次の URL を使用して、JRun デモ アプリケーションを実行して、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

#### エラーの内容：

「404 ファイルが見つかりません」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：

- 既定では、デモ アプリケーションは default JRun サーバーで実行されます。次の手順で、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は、JRun サーバー 3.1 の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は、[コントロールパネル]のサービスコントロール マネージャユーティリティ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ます。  
`% jrun -status default`
- default JRun サーバーがすでに実行されている場合は、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。既定の HTTP ポートは 80 ですが、Web サーバーが別のポートで受信している場合は、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 で受信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を確認してください。
- `/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」  
JMC での「Reverting to Developer Edition」

### 解決方法：

- ライセンスをアップグレードします。JRun の Developer 版の最大同時接続数は 3 です。ライセンスをアップグレードする必要があります。詳細については、[x ページの「JRun 製品のラインナップ」](#)を参照してください。
- ベータ版または評価バージョンの JRun を使用している場合は、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、最新の評価バージョン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 コマンドの記述中にロックされ、システムに入れない場合は、次の手順を実行してください。**

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) を使用して、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、`&` と同じ機能を持っています。次に例を示します。

```
% jrun -nohup default
```

`-nohup` オプションを使用していないときにオンラインでロックされた場合は、`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` オプションの構文については、[83 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 では、JMC を使用してさまざまな設定を構成します。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。

### 目次

• JRun 管理コンソールの開始 .....	70
• JRun シリアル番号の設定 .....	74
• JMC ユーザの管理 .....	75
• JRun サーバーの設定.....	80
• JDBC データ ソースの設定 .....	109
• Web サーバーの設定.....	114
• Web アプリケーションの構成 .....	122
• サブレットの構成 .....	146
• エンタープライズ アプリケーションの構成.....	156
• ログ ファイル ビューアの使用.....	163
• JMC キーの検索 .....	166
• ログアウト.....	166

## 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 の環境や、JRun と Web サーバーの接続を設定できます。既定では、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 4.0 以降、または Internet Explorer 4.0 以降のいずれかが必要です。

---

### JMC を開始するには

- 1 admin JRun サーバーが実行されていない場合は起動します。詳細については、[21 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 を起動します。

`http://localhost:8000`

または (Windows の場合のみ)、[スタート] > [プログラム] > [JRun 3.1] > [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。ポートは、インストール時に選択しました。

---

JMC が表示されない場合は、[25 ページの「インストールのトラブルシューティング」](#)を参照してください。

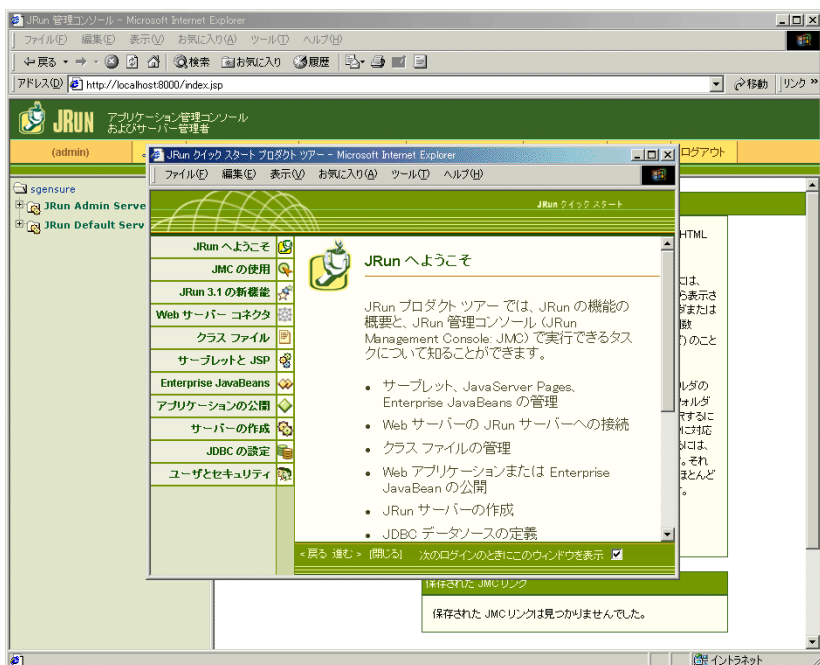


JMC のログイン ウィンドウが表示されます。



- 3 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ました。

JMC ウィンドウに「JRun へようこそ」ページが表示され、前面に [JRun クイックスタート プロダクト ツアー] が表示されます。



JMCには2つのペインが表示されます。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、ツリー内で現在選択されているフォルダまたはオブジェクトの内容が表示されます。ペインの上部にはアクセスバーがあります。このバーには、JRun サーバー専用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が表示されないか、またはツリーにあるすべてのオブジェクトにアクセスすることができません。

このツリーには、パスワード変更 (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表示されます。

## JRun 管理コンソール

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表示するには、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、[ **マシン名** ] > [ **JRun サーバー名** ] > [ **アプリケーション名** ] > [ **ログの設定** ] のように選択します。
- 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、選択した機能がJRunによって実行されます。プロパティを編集するには、右側ペインの [編集] ボタンをクリックするか、またはそのプロパティをクリックします。表示されるエディタウィンドウで変更します。
- ほとんどの場合、JRun サーバーのプロパティの変更後は、JRun サーバーを再起動してその変更を有効にする必要があります。

## JMC のお気に入りの設定

JMC では、一般的に使用される 編集ウィンドウ の一覧を「ようこそ」ページに追加できます。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、そのパネルの右下隅にある [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] セクションに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 版や無料の評価バージョンをインストールしている場合、[シリアル番号] は空欄になります。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新しい機能の制限を解除できます。

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、[xxv ページ](#)の「お問い合わせ先」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ています。  
[製品シリアル番号] パネルが表示されます。

製品シリアル番号	
JRun を購入したときに付いていたシリアル番号を入力してください。シリアル番号を入力した後に JRun を再起動すると、JRun Professional 版、JRun Advanced 版、または JRun Enterprise 版 の各機能が有効になります。	
シリアル番号	<input type="text"/>
2x シリアル番号 (アップグレードが必要)	<input type="text"/>
エディション	Developer
ライセンス タイプ	<input type="text"/>
最大同時要求数	3
有効期限	なし
<input type="button" value="アップデート"/>	
<input type="checkbox"/> 「ようこそ」ページを追加	

- 2 [シリアル番号] フィールドに、Allaire 社から提供されたシリアル番号を正確に入力します。
- 3 [アップデート] をクリックして、変更を適用します。
- 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 管理者は、このユーティリティによって JRun サーバーからのアクセスを制限できます。たとえば、ISP で使用している場合、各顧客に、専用の JRun サーバーと、そ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に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は、(admin) と表示されます。

## 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 に追加できます。個々のユーザに対し、JRun サーバーへの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。

### JMC ユーザの管理

ユーザのパスワードとアクセス権のレベル、またはいずれか一方を編集します。

**注意:**  
**パスワード** - パスワードを変更しない場合は、「\*\*\*\*\*」のままにしてください。変更するとそれが保存されます。

パスワードには「\*」を使うことができません。先行および後続スペースは無視されます。

**ユーザ名** - 先行および後続スペースは無視されます。

**結果:** 変更が完了しました。  
変更内容は、該当するユーザがログアウトしてから有効になります。

現在有効な JMC ユーザー			
✖	ユーザ名	パスワード	アクセス権
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	すべてのサーバー JRun Admin Server JRun Default Server
<input type="checkbox"/>	<input type="text"/>	<input type="text"/>	すべてのサーバー JRun Admin Server JRun Default Server
<input type="checkbox"/>	sirius	*****	すべてのサーバー JRun Admin Server JRun Default Server

JMC ユーザの更新

- 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] には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。名前の最初と最後に付いている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は、アスタリスクによるパスワードの非表示は行われません。
  - パスワードは 1 文字以上の長さにしてください。
  - パスワードにはスペースやアスタリスク (\*) を使用できません。
- 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 サーバーをクリックします。
- 5 変更を適用するには、[JMC ユーザの更新] ボタンをクリックします。

## JMC ユーザの設定の変更

JMC の [ユーザの管理] のオプションを使用すると、ユーザのパスワードや、ユーザがアクセス権を持つ JRun サーバーを変更できます。JMC の [ユーザの管理] のオプションで **admin** ユーザに対する設定を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] パネルが表示されます。
- 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 サーバーをクリックします。
- 3 ユーザのパスワードを変更するには、[パスワード] フィールドの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] に新しいパスワードを入力している間は、アスタリスクによるパスワードの非表示は行われません。
  - パスワードは 1 文字以上の長さにしてください。
  - パスワードにはスペースやアスタリスク (\*) を使用できません。
- 4 変更を適用するには、[JMC ユーザの更新] ボタンをクリックします。  
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 の [ユーザの管理] オプションを使用すると、**admin** 以外であればどのユーザも JRun から削除できます。JMC の [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できます。
- 3 変更を適用するには、[JMC ユーザの更新] ボタンをクリックします。



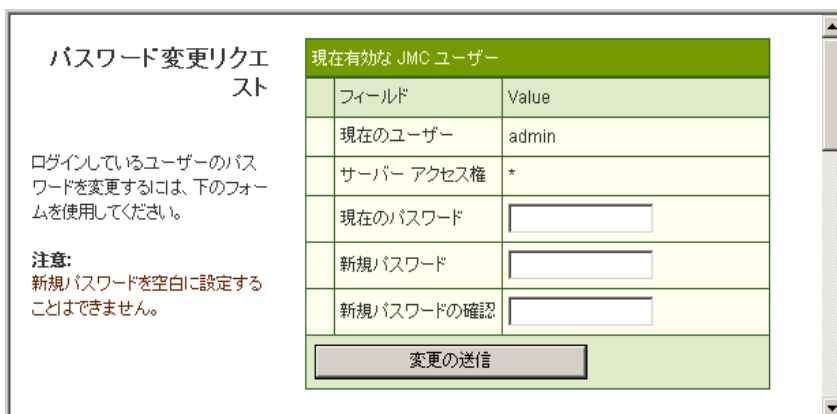
## パスワードの変更

JMC の [パスワードの変更] オプションを使用すると、現在ログインしているユーザのパスワードを変更できます。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によって、ほかのどのユーザのパスワードでも変更できます。詳細については、77 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] パネルが表示され、現在ログインしているユーザとそのユーザの JRun サーバーに対するアクセス権が表示されます。



現在有効な JMC ユーザー	
フィールド	Value
現在のユーザー	admin
サーバー アクセス権	*
現在のパスワード	<input type="password"/>
新規パスワード	<input type="password"/>
新規パスワードの確認	<input type="password"/>

変更の送信

[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、そのユーザがすべてのサーバに対するアクセス権を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、[新規パスワードの確認] フィールドにもう一度入力します。

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] 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、ほか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 1 文字以上の長さにしてください。

- 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 アーキテクチャの中心的役割を担います。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 サーバーで実行できる Web アプリケーションの数に制限はありません。
- JRun 接続モジュールを経由した、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 サーバーによって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 を使用したビジネス ロジックの実装

JRun インストールでは、default JRun サーバーと admin JRun サーバーという 2 つの JRun サーバーをセットアップします。JMC ではそれぞれ、[default JRun サーバー] および [admin JRun サーバー] と表示されます。

---

### メモ

Windows NT では、JRun サーバーを NT サービスとしても、あるいはアプリケーションとしても実行できます。詳細については、[21 ページの「Windows に関する検討事項」](#)を参照してください。

---

JRun サーバーの既定の Web アプリケーションの一覧については、[122 ページの「既定のアプリケーション」](#)を参照してください。

このセクションでは、次のトピックについて説明します。

- 「[JMC での JRun サーバーの再起動](#)」 [82 ページ](#)
- 「[jrun コマンドの使用](#)」 [83 ページ](#)
- 「[JRun サーバーの追加と削除](#)」 [86 ページ](#)
- 「[Java Virtual Machine の設定](#)」 [91 ページ](#)
- 「[JRun サーバー イベント ログの設定](#)」 [108 ページ](#)

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 での JRun サーバーの管理

JMC での JRun サーバー管理には JRun サーバーで実行する内容によって 2 種類の方法があります。このセクションでは、これらの方法について説明します。

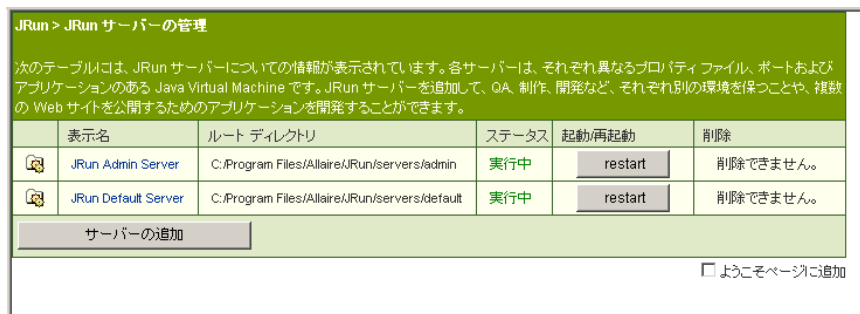
### [JRun サーバーの管理] パネルへのアクセス

JMC には [JRun サーバーの管理] パネルがあり、JRun サーバーの追加、削除、および再起動を実行できます。また、このパネルには JRun サーバーのステータスやそのルート ディレクトリが表示されます。

#### [JRun サーバーの管理] パネルにアクセスするには

- 1 JMC の左側ペインで、[マシン名] をクリックします。

[JRun サーバーの管理] パネルが表示されます。

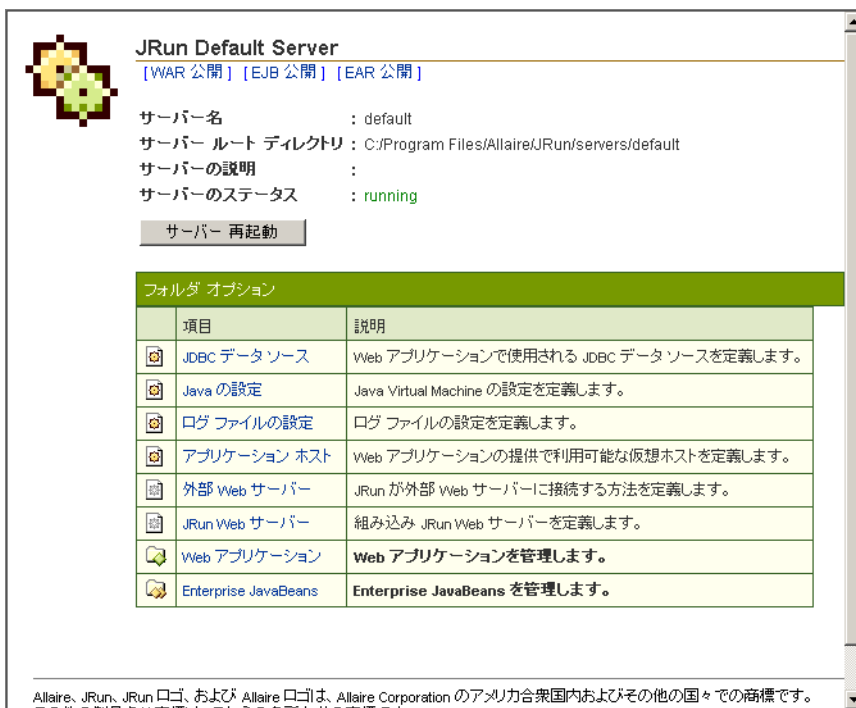


### JRun サーバー パネルへのアクセス

JMC には個別の JRun サーバーのパネルもあります。JRun サーバー パネルには、サーバーについての情報と現在のステータスが表示されます。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 (82 ページの「JMC での JRun サーバーの再起動」を参照)。

## 個別の JRun サーバー パネルにアクセスするには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] をクリックします。  
JRun サーバー パネルが表示されます。



## JMC での JRun サーバーの再起動

JMC には、JRun サーバーを簡単に再起動する方法があります。JMC での変更内容を有効にするときに再起動が必要な場合があります。また、テキスト エディタでプロパティ ファイルを変更したときや、接続ウィザードの実行後にも、JRun サーバーの再起動が必要です。

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、[83 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]のサービスコントロールマネージャユーティリティ (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 (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もできます。

---

**メモ**

JMC から `admin` サーバーを再起動しないでください。これは、JMC が `admin` サーバー上で実行されているためです。

---

**JMC で JRun サーバーを再起動するには**

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] をクリックします。  
JRun サーバー パネルが表示されます。
- 2 [サーバー再起動] をクリックします。

## 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* のルート ディレクトリ `/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 オプションの一覧を表示するには、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 のルート ディレクトリ] -start | -stop | -restart
      [JRun サーバー名]
jrun [-jrundir JRun のルート ディレクトリ] -start [JRun サーバー名] [-debug]
jrun -install NTサービス名 サーバー名 [-quiet]
jrun -remove NTサービス名 [-quiet]
jrun -demo JRun サーバー名
jrun -java java_prog -classpath classpath java_args class class_args
```

Windows で、jrun コマンドを起動したときに「Could not find *JRun* のルート ディレクトリ `/jvms.properties file`」などのエラーが表示された場合は、`/bin` ディレクトリがシステムパスにあることを確認するか、ディレクトリを明示してコマンドを実行してください。

-info や -version などのオプションの詳細については、『JRun 拡張設定ガイド』を参照してください。

**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。local.properties ファイルで java.System.out および java.System.err によって指定される stdout/sterr 転送を不可にし、これらをログ ファイルではなくコンソールに出力します。また、screenlogger サービスの利用時にこのオプションを使用します。次に例を示します。

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

**demo**

```
jrun -demo サーバー名
```

default サーバーで JRun デモ アプリケーションを開始します。別の JRun サーバーにデモ アプリケーションを再公開する場合は、コマンド ラインに JRun サーバーを指定します。Windows NT のみ。次に例を示します。

```
% jrun -demo default
```

**install**

```
jrun -install NTサービス名 サーバー名 [-quiet]
```

JRun を Windows NT サービスとしてインストールします (NT のみ)。サーバー名は jvms.property ファイルにあるいずれかの JRun サーバー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 ファイルに追加します。次に例を示します。

```
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の詳細については、[86 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 によって、すべての JAR ファイルが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
```

**nohup**

```
jrun -nohup [JRun サーバー名] [-debug]
```

UNIX のみ。すべての JRun サーバーを開始します。特定のサーバーのみを開始するには、*JRun* サーバー名に該当するサーバー名を指定します。**-nohup** オプションは **-start** とは異なり、フォアグラウンド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用の新規プロセスを作成します。**-debug** オプションによって、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は、JRun サーバーを開始する前に、最初に **admin JRun** サーバーを開始する必要があります。

**remove**

```
jrun -remove NTサービス名 [-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 サーバー名]
```

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

**start**

```
jrun -start [JRun サーバー名] [-debug]
```

```
jrun -jrundir JRun のルート ディレクトリ -start [JRun サーバー名]  
[-debug]
```

すべての JRun サーバーを開始します。特定の JRun サーバーのみを開始するには、*JRun* サーバー名に該当するサーバー名を指定します。**-debug** オプションによって、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は、他の JRun サーバーを開始する前に、最初に **admin JRun** サーバーを開始する必要があります。

例:

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

**-jrundir** オプションは、OEM で使用します。詳細については、『拡張設定ガイド』を参照してください。

**status**

```
jrun -status [JRun サーバー名]
```

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

**stop**

```
jrun -stop [JRun サーバー名]
```

すべての JRun サーバーを停止します。特定の JRun サーバーのみを停止する場合は、*JRun* サーバー名に該当するサーバー名を指定します。

## JRun サーバーの追加と削除

既定の JRun インストールには、**admin** と **default** の 2 つのサーバーが含まれています。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して実行するための方法が提供されています。このセクションでは、JMC を使用して、JRun サーバーの追加と削除を行う方法について説明します。手作業による JRun サーバーの追加と削除の詳細については、『拡張設定ガイド』を参照してください。

JRun サーバーの追加や削除を行う場合は、それぞれのサーバーに固有のポートが必要であることを注意してください。詳細については、[168 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

JRun サーバーの追加には、次のような利点があります。

- 多くの企業で、複数の Web アプリケーションから構成されるサイトを構築していますが、複数の JRun サーバーを使用すると、これらの Web アプリケーションを分離できます。いずれかの JRun サーバーがクラッシュしたり再起動が必要になっても、ほかのサーバーやそれらのアプリケーションは実行を継続できます。
- 新しい JRun サーバーを追加すると、サーバーレベルでクラスパス、データソース、および EJB 設定値を定義できるようになります。つまり、各 Web アプリケーションレベルで定義できるようになります。
- また、ユーザのタイプによって JRun サーバーを論理的に分ける場合もあります。たとえば、生産、品質管理、および admin JRun サーバーを使用すると、開発のワークフローを構築する上で役立ちます。

---

**メモ**

JRun サーバーを追加する場合は、JMC に **admin** としてログインする必要があります。

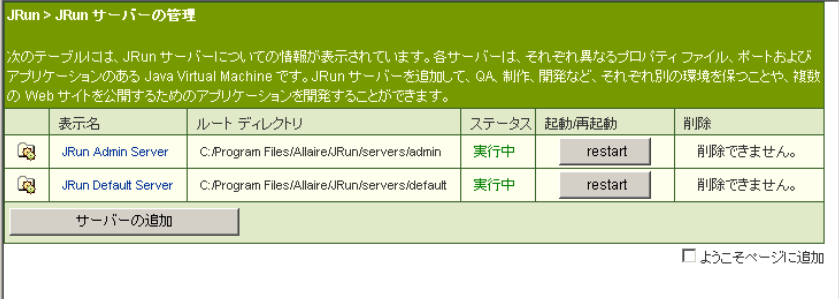
---



## 新しい JRun サーバーを追加するには

- 1 JMC の左側ペインで、[マシン名] をクリックします。

[JRun サーバーの管理] パネルが表示されます。このパネルには、現在の JRun サーバー、ステータス、およびルート ディレクトリが表示されます。



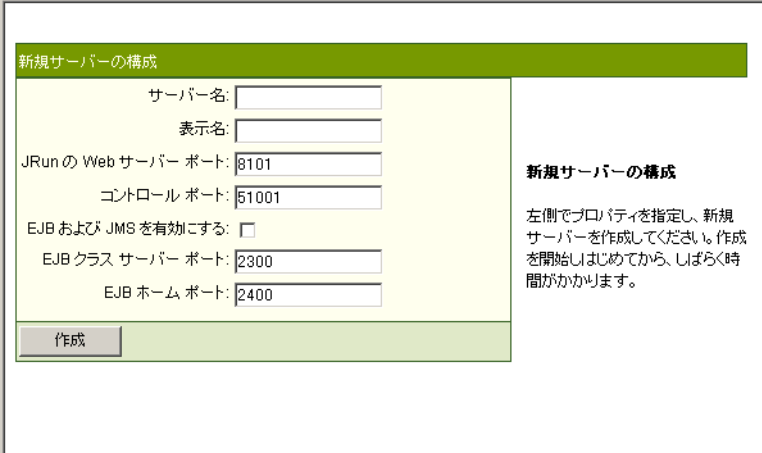
	表示名	ルート ディレクトリ	ステータス	起動/再起動	削除
	JRun Admin Server	C:\Program Files\Allaire\JRun\servers\admin	実行中	restart	削除できません。
	JRun Default Server	C:\Program Files\Allaire\JRun\servers/default	実行中	restart	削除できません。

サーバーの追加

ようこそページを追加

- 2 [サーバーの追加] ボタンをクリックします。

[新規サーバーの構成] パネルが表示されます。



新規サーバーの構成

サーバー名:

表示名:

JRun の Web サーバー ポート:

コントロール ポート:

EJB および JMS を有効にする:

EJB クラス サーバー ポート:

EJB ホーム ポート:

作成

**新規サーバーの構成**

左側でプロパティを指定し、新規サーバーを作成してください。作成を開始し始めてから、しばらく時間がかかります。

## メモ

[新規サーバーの構成] パネルを拡張すると、フィールドをさらに追加したり、新しいサーバーの `local.properties` ファイルにプロパティを追加できます。詳細については、『拡張設定ガイド』を参照してください。

- 3 次の表の説明に従ってフィールドを編集します。

フィールド	説明
サーバー名	新しい JRun サーバーの名前を入力します。たとえば、「admin」や「default」と入力します。この名前には、空白文字は使用できません。
表示名	新しい JRun サーバーの、わかりやすい名前を入力します。この名前は JMC の左側ペインに表示されます。
JRun の Web サーバーポート	<p>固有のポート番号を入力します。推奨する番号の範囲は、8101 ~ 8199 です。JRun の既定値は、この範囲の中で使用可能な一番小さい数字になります。ほかの JRun サーバーで使用していないポートを選択してください。</p> <p>これは、新しい JRun サーバーに関連付けられている JRun Web Server (JWS) の TCP ポート番号です。JWS では、このポートで HTTP 要求が受信されます。</p> <p>admin JRun サーバーの JWS の既定値は 8000 です。default JRun サーバーの JWS の既定値は 8100 です。</p> <p>JRun ポートの詳細については、<a href="#">168 ページの「JRun ポートについて」</a>を参照してください。</p>
コントロールポート	<p>固有のポート番号を入力します。推奨する番号の範囲は、51001 ~ 51999 です。JRun の既定値は、この範囲の中で使用可能な一番小さい数字になります。ほかの JRun サーバーで使用していないポートを選択してください。</p> <p>これは、admin JRun サーバーで、ほかの JRun サーバーへの制御メッセージの送信に使用されるポートです。</p> <p>JRun ポートの詳細については、<a href="#">168 ページの「JRun ポートについて」</a>を参照してください。</p>
EJB および JMS を有効にする	新しい JRun サーバーの EJB および JMS のサービスをオンにする場合に、このチェックボックスをオンにします。既定値はオフです。




フィールド	説明
EJB クラス サーバー ポート	<p>固有のポート 番号を入力します。推奨する番号の範囲は 2300 ~ 2399 です。JRun の既定値は 2300 です。ほかの JRun サーバーで使用されていないポート を選択してください。</p> <p>これは、EJB エンジンでクライアント へのクラスの送信に使用するポート です。</p> <p>別の JRun サーバーの EJB クラス サーバー ポートを確認するには、その JRun サーバーの local.properties ファイルの ejb.ejpt.classServer.port プロパティを調べてください。</p> <p>JRun ポート の詳細については、<a href="#">168 ページの「JRun ポートについて」</a>を参照してください。</p>
EJB ホーム ポート	<p>固有のポート 番号を入力します。推奨する番号の範囲は、2400 ~ 2499 です。JRun の既定値は 2400 です。ほかの JRun サーバーで使用されていないポート を選択してください。</p> <p>これは、EJB ホーム オブジェクト のポート です。</p> <p>別の JRun サーバーの EJB ホーム ポートを確認するには、その JRun サーバーの local.properties ファイルの ejb.ejpt.homePort プロパティを調べてください。JRun ポート の詳細については、<a href="#">168 ページの「JRun ポートについて」</a>を参照してください。</p>

#### 4 [作成] ボタンをクリックします。

JRun によって新しいサーバーが作成され、[JRun サーバーの管理] パネルに戻ります。サーバー リストに新しいサーバーが表示されます。また、新しい JRun サーバーごとに新しい JWS が作成されます。新規サーバーおよびその JWS は停止します。

**JRun > JRun サーバーの管理**

次のテーブルには、JRun サーバーについての情報が表示されています。各サーバーは、それぞれ異なるプロパティファイル、ポートおよびアプリケーションのある Java Virtual Machine です。JRun サーバーを追加して、QA、制作、開発など、それぞれ別の環境を保つことや、複数の Web サイトを公開するためのアプリケーションを開発することができます。

	表示名	ルート ディレクトリ	ステータス	起動/再起動	削除
	JRun Admin Server	C:\Program Files\Allaire\JRun\servers\admin	実行中	restart	削除できません。
	JRun Default Server	C:\Program Files\Allaire\JRun\servers/default	停止	start	remove
	Junko	C:\Program Files\Allaire\JRun\servers/Junko	停止	start	remove

サーバーの追加

サーバーが追加されました。

ようこそページに追加

- 5 [start] ボタンをクリックして、新しい JRun サーバーを開始します。  
新しい JRun サーバーの `local.properties` ファイルのスケルトンが作成されます。`local.properties` ファイルに明示されていない必須プロパティは、`global.properties` ファイルから継承されます。新しい JRun サーバーには、既定の Web アプリケーションは含まれていません。
- 6 新しい JRun サーバーを使用するには、そのサーバーの Web アプリケーションを作成する必要があります。詳細については、124 ページの「アプリケーションの作成」を参照してください。

## JRun サーバーの削除

JRun サーバーの削除は十分に注意して行ってください。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。JRun サーバーの削除によって、そのサーバーに関連する JWS も削除されます。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。

### メモ

admin JRun サーバーおよび default JRun サーバーは削除できません。




### JRun サーバーを削除するには

- 1 次のコマンドラインで、削除する JRun サーバーを停止します。  
`% jrun -stop サーバー名`
- 2 JMC の左側ペインで、[マシン名]をクリックします。

[JRun サーバーの管理] パネルが表示されます。このパネルには、現在の JRun サーバ、ステータス、およびルート ディレクトリが表示されます。

**JRun > JRun サーバーの管理**

次のテーブルには、JRun サーバーについての情報が表示されています。各サーバーは、それぞれ異なるプロパティファイル、ポートおよびアプリケーションのある Java Virtual Machine です。JRun サーバーを追加して、QA、制作、開発など、それぞれ別の環境を保つことや、複数の Web サイトを公開するためのアプリケーションを開発することができます。

	表示名	ルート ディレクトリ	ステータス	起動再起動	削除
	JRun Admin Server	C:\Program Files\Allaire\JRun\servers\admin	実行中	restart	削除できません。
	JRun Default Server	C:\Program Files\Allaire\JRun\servers/default	停止	start	remove
	Junko	C:\Program Files\Allaire\JRun\servers/Junko	停止	start	remove

サーバーの追加

サーバーが追加されました。

ようこそページに追加

- 3 削除する JRun サーバーの隣にある [remove] ボタンをクリックします。[remove] ボタンが無効になっている場合は、その JRun サーバーが停止していることを確認してください。

サーバーの削除を確認するプロンプトが表示されます。

- 4 [はい、サーバーを削除します] ボタンをクリックします。

JRun は、そのサーバーおよびその JWS を削除します。そのサーバーのファイルは削除されません。

## Java Virtual Machine の設定

Java Virtual Machine (JVM) は JRE とも呼ばれ、ソフトウェアに実装された CPU です。これには、Java プラットフォーム用に作成されたプログラムを実行する場合に必要なすべての機能が含まれています。

各 JRun サーバーは、その JRun サーバーに対するすべてのサーブレット、JSP、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 の設定] 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。また、UNIX システムでは、`jrun` コマンドの `-console` オプションを使用して、コンソールに `Java.System.err` および `java.System.out` 出力を転送できます。詳細については、[83 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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Java の設定] をクリックします。

[Java の設定] パネルが表示されます。

名前	値	要約
Java 実行ファイル	D:\jdk12-1.2\bin\javaw.exe	JVM 実行ファイルへのパス
System.out ログ ファイル	{jrun.rootdir}\logs\{jrun.server.name}-out.log	System.out メッセージを記録する場所
System.err ログ ファイル	{jrun.rootdir}\logs\{jrun.server.name}-err.log	System.err メッセージを記録する場所
JRun コントロール ポート	53000	サーバー コマンドを送信するために JRun が使用するポート
クラスパス	{jrun.rootdir}\servers\lib	追加のクラスパス エントリ
Java 引数		Java 実行ファイルに渡される追加のコマンドライン引数
ライブラリ パス	{servlet.jpipath}\{ejb.jpipath}	ネイティブ JNI のディレクトリ

編集

「ようこそ」ページに追加

- 2 右側ペインで、[編集] をクリックします。

Java の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フィールドを編集します。

プロパティ	説明
Java 実行ファイル	Java Virtual Machine 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があります。 JVM をバージョン 1.1.8 から 1.2 以降にアップグレードする場合は、[Java 引数] フィールドから次の引数を削除する必要があります。 -Djava.naming.factory.initial=allaire.jrun.ContextFactory
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名を入力します。
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名を入力します。
JRun コントロール ポート	固有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 スクリプトによって決まります。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一覧です。クラスをこのパスに追加するか、JRun によって既定でクラスパスに追加されている <i>JRun のルート ディレクトリ/classes</i> ディレクトリにクラスを保存します。 テキスト フィールドに、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ブレットによって使用されます。 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があるクラスおよび 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 実行可能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に、Java ネイティブ インターフェイス (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 [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。

## Java コマンドライン オプションの使用

既定では、Windows 用 JRun と Solaris 用 JRun では JRE 1.2 を使用します。エディタを使用して Java Executable 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 (さらに、[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 非同期ガーベッジ コレクションができないようにします。
- verbosegc ガーベッジ コレクションが開始されると、メッセージを表示します。
- noclassgc クラス ガーベッジ 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>  
クラスの検出先ディレクトリをリストします。
- prof[:<file>] .%java.prof または .%<file> にプロファイル データを出力します。
- verify 読み込み時にすべてのクラスを検証します。
- verifyremote ネットワーク経由で読み込まれたクラスを検証します [default]。
- noverify クラスを検証しません。
- 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	エラー発生時に、処理を終了する前に一時停止します。
/v	すべてのクラスを検証します。
/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)。

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

## JWS での SSL (Secure Socket Layer) の使用

JRun では、JRun Web サーバー (JWS) の強化のため、Secure Socket Layer (SSL) プロトコルをサポートしています。

---

### メモ

JWS で SSL を有効にするには、Sun の JDK バージョン 1.2.2 以降を使用してください。

---

多くの SSL 設定値は JRun の `global.properties` ファイルの設定値を使用していますが、JMC には、各 JWS について、SSL の作成、検証、およびテスト用のインターフェイスが用意されています。このセクションでは、JWS で SSL を有効にする方法について説明します。



## JRun の SSL の制限

JRun SSL 実装では、JRun に接続されているかどうかにかかわらず、外部 Web サーバーで SSL を有効にすることはできません。SSL 実装は組み込み型の JWS にのみ適用できます。Web サーバーでの SSL の使用については、Web サーバーのマニュアルを参照してください。

JRun SSL 実装を使用する場合の重要な制限事項は次の 2 つです。

- JRun SSL 実装では、ブラウザと外部 Web サーバー間のソケットは保護されません。多くの商用 Web サーバーでは、この組み込みをサポートしています。
- JRun SSL 実装では、外部 Web サーバーと JRun のコネクタ間のソケットは保護されません。

## JWS 用の SSL のセットアップ

JWS 用に SSL をセットアップするには、いくつかの手順があります。ここではその手順について説明します。その後のセクションでは、より複雑なタスクについて詳しく説明します。手順の一部はオプションで、認証局が発行した検証済みサーバー証明書を使用するのか、JRun によって作成および保持されている非認証証明書を使用するのかによって、扱いが異なります。オプションの手順には、「(オプション)」と示します。

- 1 証明書署名要求と非認証サーバー証明書を作成します。

証明書署名要求 (CSR、Certificate Signing Request) および JMC を使用した非認証サーバー証明書を作成します。CSR はエンコードされたプレーンテキストファイルで、公開鍵、組織名および所在地、URL および主要な要求から構成されています。オプションで、CSR を認証局 (CA、Certificate Authority) に提出することもできます。CA からサーバー証明書が届いたら JRun サーバーにインストールします。JRun によって作成される非認証の証明書を使用することもできます。これは、既定で JRun サーバーにインストールされており、CA による検証が不要です。JRun サーバーのキーストアに、非承認の証明書が保存されます。この証明書には秘密鍵が含まれています。

[98 ページの「CSR および非認証のサーバー証明書の作成」](#)を参照してください。

- 2 CSR を提出します (オプション)。

CA に CSR を提出します。このとき、CSR の始まりと終わりを示す BEGIN 行および END 行を加えてください。JRun では、SSL ウィザードの [手順 1](#) で CSR を作成し、[手順 3](#) で指定した場所に保存します。サーバー証明書のサンプルを次に示します。

```
-----BEGIN CERTIFICATE-----
```

```
MIIBCTCBtAIBADBPMQswuLCBJbmMxRzBFBgNVBAStPnd3dy5RmxcmlkYTEYMBYGCy1UEChMPRXl1cyBvb3IuUaGUgV2ViMRQwEgYDVQDDFA3d3cuZXR3Lm51dBcMAOGCSqGSIb3DQEBAQUAA0sAMEgCQQCeojtjnHqg0GTxp+XZ56RaSe1iZwPumXjU6Sx7v1FdXzsY1oLQa090Jtnu1wsQRHh0yDS+45oncjKm1zCG/IZAgMBAAGgADANBgkqhkiG9w0BAQQFAANBAFBj9g-NiUh8YwPrFGntgf4miUd/wqUshptjJy4PjdsD3ugyCSqGSIb3DQEBAQUAA0sAMEgCQQCeojtjnHqg0GTxp+XZ56RaSe1iZwPumXjU6Sx7zc3VyYw5jZXMGkEMpV1MxOTk3MB4XDTAxMDMwNzAwMDAwMFOXTAxMDesadFLKJfXwZda9gpyvtC1fbSzJfwYQ8JLS20rNVC4XGhb4Hq4GQAX+5w7Sm0RInxDJBsdfd aVgZxA==
```

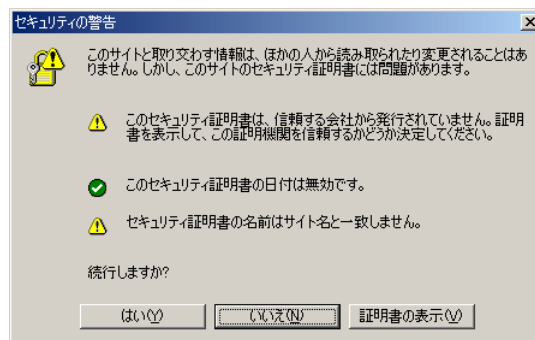
```
-----END CERTIFICATE-----
```

CSR を提出する場合は、電子メール アドレスや連絡先などの追加登録情報についても入力する必要があります。国名や一般名称などの入力情報は、SSL ウィザードで入力した情報と必ず一致させてください。

CA からは通常、電子メールの一部として SSL サーバー 証明書が返されます。これは、**Secure Server ID**、サーバー ID、またはトライアル サーバー ID と呼ばれることがあります。この手順を実行する場合は、**手順 4** も実行してください。**手順 3** を実行する必要がある場合もあります。

CA の例としては、Verisign 社 (<http://www.verisign.com>) のものがあります。Verisign 社は無料で試用 Secure Server ID を提供しており、これを使用して SSL 実装を試すことができます。試用の ID を継続的に使用する場合は、これを登録して料金を払う必要があります。

作成した CSR を CA に提出していない場合に JWS に安全なリソースを要求すると、ブラウザに次のダイアログ ボックスが表示されます。このダイアログ ボックスでは、非認証のサーバー証明書を受け取ることができます。



CSR を CA に提出している場合は、ブラウザで CA が確認され、それが信頼のおける権限であれば、ユーザに透過的に安全な接続が生成されます。

### 3 CA ルートを設定します (オプション)。

Secure Server ID (Verisign からの) やほかの試用サーバー証明書を使用している場合は、Web ブラウザにテスト CA ルートをインストールしなければならない場合があります。その手順については、利用した CA に確認してください。この手順は、CSR を CA に提出した場合のみ必要になります。

### 4 サーバー証明書を認証します (オプション)。

CA の電子メールからサーバー証明書を抽出し、JRun サーバーにインストールします。この手順は、CSR を CA に提出した場合にのみ必要になります。詳細については、**104 ページ**の「**サーバー証明書の認証 (オプション)**」を参照してください。

### 5 JRun サーバーを再起動します。

JRun サーバーにサーバー証明書を追加した後、SSL 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。

- 6 サーバーをテストします。

Web ブラウザを開き、要求に HTTPS プロトコルを使用して安全なリソースを要求します。

JMC では、一度新しい証明書が作成されて正しくインストールされると、[SSL 管理] パネルで新しい証明書をテストするリンクが作成されます。次の図に示すように、このリンクは [SSL 管理] パネルの [現在の証明書] の説明内にあります。

	SSL Certify	SSL 証明書鍵の認定
	SSL Delete	SSL 証明書鍵の削除
現在の証明書		
テスト リンク	この証明書の所有者	
	sgensure CN=sgensure OU=qa O=allaire L=newton ST=mass C=US	CN=sgensure OU=qa O=allaire L=newton ST=mass C=US
	シリアル番号	3b2fdc17
	証明書の有効	Tue Jun 19 19:11:19 EDT 2001 until: Mon Sep 17 19:11:19 EDT 2001
	証明用の指紋	MD5: D1:F3:FC:3B:19:81:22:49:1C:08:38:B0:66:8A:61:CD SHA1: E5:0E:08:10:07:2C:23:B6:A5:D8:69:54:6F:69:7B:4E:EB:77:AC:AC
	この証明書の所有者	
	証明書の発行元:	

- 7 保護されていないポートを無効にします (オプション)。

一度 JWS の SSL を設定してテストすると、保護されていないポートを無効にして、HTTPS トラフィックによって保護されているポート上のみで要求を行う場合があります。詳細については、[107 ページの「保護されていないポートの無効化 \(オプション\)」](#)を参照してください。

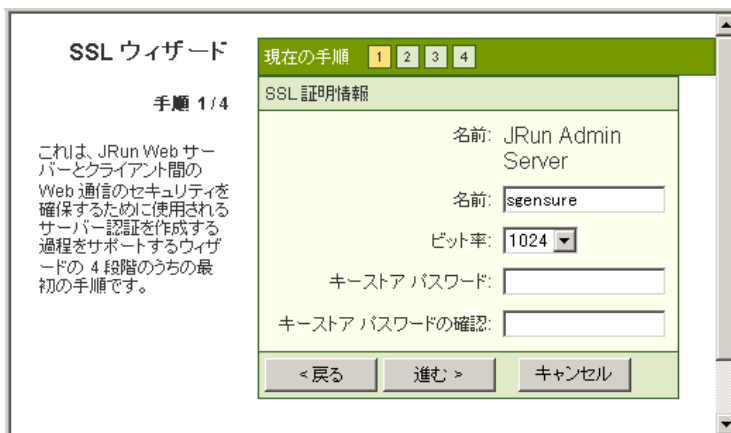
## CSR および非認証のサーバー証明書の作成

### CSR および非認証のサーバー証明書を作成するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [SSL] をクリックします。  
[SSL 管理] パネルが表示されます。現在のサーバーについて証明書がない場合、このパネルには [SSL Create] オプションのみが表示されます。



- 2 [SSL Create] をクリックして、新しい CSR や非認証のサーバー証明書を作成します。  
[SSL ウィザード] の手順 1 が表示されます。



## 3 次の表の説明に従ってフィールドを編集します。

フィールド	説明
名前	<p>サーバーの一般名を入力します。JRun では装置名が既定値となります。インターネットに接続している場合は、DNS 名を使用します。たとえば、www.servername.com などを使用します。</p> <p>一般名を入力する場合は、次のことに注意してください。</p> <ul style="list-style-type: none"><li>• 一般名は、多くの場合、ホスト名とドメイン名から構成します (www.domain.com や domain.com など)。</li><li>• 一般名は安全なリソースにアクセスする場合にユーザが要求する Web アドレスと同じものにする必要があります。</li><li>• 一般名はそのホスト名に固有のものになります。たとえば、www.domain.com のサーバー証明書を取得した場合に、このサーバー証明書を使用して secure.domain.com の安全な要求を行うとエラーになります。</li><li>• 複数の CSR を作成して、1つのドメイン内の複数のホストに対する複数の Secure ID を要求することもあります。JRun サーバーは1つの証明書をサポートするため、複数の安全なホストをサポートするには、複数の JRun サーバーが必要です。</li><li>• 安全なサーバーをローカルネットワーク上で使用する場合、一般名は1単語またはサーバー名 (maximus など) にすることができます。</li></ul>
ビット率	<p>ドロップダウンリストからビット率を選択します。ビット率が高いほど、暗号は強化されます。ただし、ビット率を高くするほど転送の暗号解除に必要な計算が増え、パフォーマンスが低下します。</p>
キーストアパスワード	<p>キーストアのパスワードを入力します。パスワードは6文字以上にする必要があり、大文字と小文字を区別します。</p>
キーストアパスワードの確認	<p>キーストアのパスワードを再入力します。</p>

- 4 [進む] をクリックします。

[SSL ウィザード] の手順 2 が表示されます。

**SSL ウィザード** 現在の手順 1 2 3 4

手順 2 / 4  
フィールドに記入後、[次へ] をクリックして手順 3 に進んでください。

組織情報:

組織名:

組織内の部署:

市町村名:

都道府県名:

国名/地域名: アメリカ合衆国

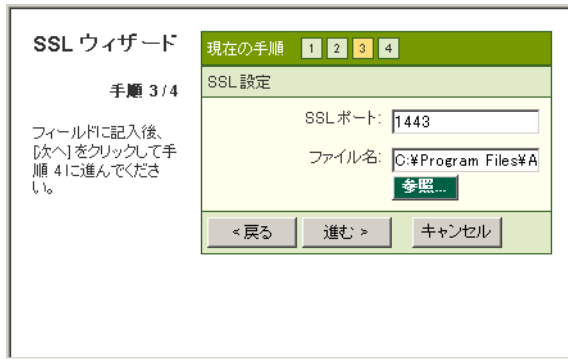
< 戻る    進む >    キャンセル

- 5 次の表の説明に従ってフィールドを編集します。

フィールド	説明
組織名	会社名を入力します。たとえば、「Allaire」と入力します。
組織内の部署	会社の課名または部門名を入力します。たとえば、「QA」と入力します。
市町村名	会社の住所の市町村名を入力します。たとえば、「Newton」と入力します。
都道府県名	会社の住所の都道府県名を入力します。アメリカ合衆国の場合は州の完全な名前を入力してください（たとえば、「MA」ではなく「Massachusetts」と入力します）。
国名/地域名	ドロップダウンリストから会社の住所の国名を選択します。既定値は「アメリカ合衆国」です。

- 6 [進む] をクリックします。

[SSL ウィザード] の手順 3 が表示されます。



- 7 次の表の説明に従ってフィールドを編集します。

フィールド	説明
SSL ポート	SSL トラフィックで使用するポート番号を入力します。既定値は 443 です。root レベルのアクセスがないシステムの場合は、1024 よりも大きいポートを選択しなければなりません。この場合、JRun の既定値は 1443 になります。Web サーバーなどほかのプロセスがポート 443 を使用している場合は、1443 が使用されます。このポートが使用中または使用不能である場合は、2443 などが使用されます。
ファイル名	ファイルへのパスと CSR の名前を入力するか、[参照] をクリックして場所を選択します。その場所に CSR が保存されます。

- 8 [進む] をクリックします。

JRun では、指定した場所に CSR が作成され、[SSL ウィザード] の手順 4 にこの場所が表示されます。

SSL ウィザード
現在の手順 1 2 3 4

手順 4 / 4

生成された CSR

ファイル名: C:\Program Files\Allaire\JRun\JRun Admin Server.txt

CSR には、次の情報が含まれています。

発行先: sgensure

ビット率: 1024

組織名: allaire

組織内の部署: qa

市町村名: newton

都道府県名: mass

国名: US

CSR

```

-----BEGIN NEW CERTIFICATE REQUEST-----
MIIBqDCCARECAQAwDELMAkGA1UEBhMCVYVWxJAUBoNjYBAgTDW1hc3NhY2h1c2Y0dHMxZDZANBgNV
BAcTBm5ld3RvbjEDMA4GA1UEChMHYXN5YyYyZTEuMAkGA1UECmMcWEsETAPBgNVBAMTCHNzW5z
dXJlMIGfMA0GCSqGSIb3DQEBQUAA4GNADCBiQKg0DdnW0feHtXGtw+Umca8th8HyT/ETqPC+3P
FRkPWCS44x0wT+PcBreHYuLk8P0nSJ06xSN6XDAeeed74+XssoUSEDZkaGwM/P4qLL481iaGaJh
IvD86C0C0s1N6+r8RwMJCNTE1YD90o3HkrC1rIFuds7Y0GC2xtkbs74CUskzjzWIDAQABoAAwDQYJ
KoZihvcNAQEEOADgYEAbTcJLLrtFOL4KnFmR&EVOF7L1IQaI7JOPSKUYAzcd0IvDLezn0P1/XmM
BAR0IDYhh87TCR8aptf4VLC/7Y71IzUaI+5mkhgz7BtizCFNDITPAYsukYVwpatHx+PJU0afQuSg+
XCaw4s/1JcJnXVnKHvB2bNF4ccD24PXYP4R7fjY=
-----END NEW CERTIFICATE REQUEST-----

```

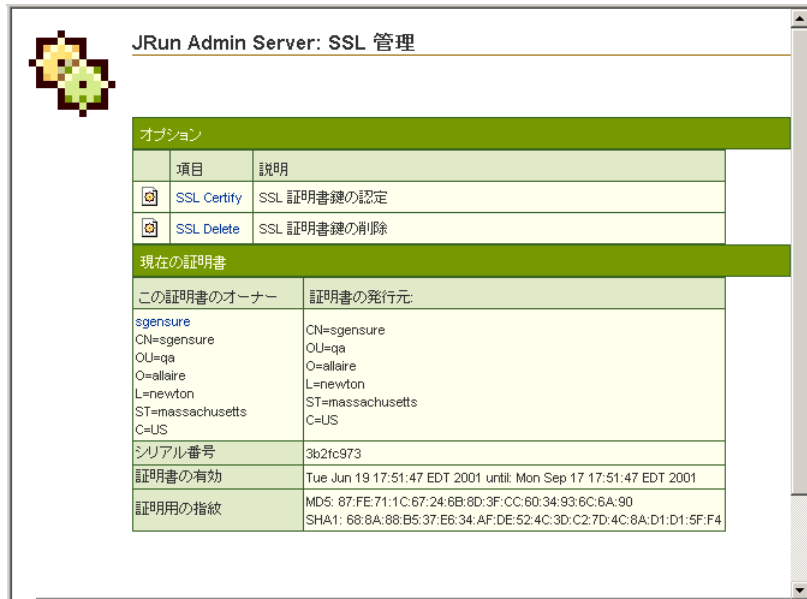
< 戻る      完了 >



メモ:  
この認証要求を使用して、Verisign などの会社で署名を受けることができます。未認証の証明書が作成され、有効になりました。



- 9 表示内容が正しいことを確認し、[完了] をクリックします。

[SSL 管理] パネルが表示されます。2 つの新しいリンクの下に現在の証明書の詳細が表示されます。JWS では JRun サーバーあたり 1 つの証明書のみをサポートしているため、既存の証明書を削除しないと、新しい証明書を作成できません。



オプション	
項目	説明
 SSL Certify	SSL 証明書鍵の認定
 SSL Delete	SSL 証明書鍵の削除

現在の証明書	
この証明書のオーナー	証明書の発行元:
sgensure CN=sgensure OU=qa O=allaire L=newton ST=massachusetts C=US	CN=sgensure OU=qa O=allaire L=newton ST=massachusetts C=US
シリアル番号	3b2fc973
証明書の有効	Tue Jun 19 17:51:47 EDT 2001 until: Mon Sep 17 17:51:47 EDT 2001
証明用の指紋	MD5: 67:FE:71:1C:67:24:6B:8D:3F:CC:60:34:93:6C:6A:90 SHA1: 68:8A:88:B5:37:E6:34:AF:DE:52:4C:3D:C2:7D:4C:8A:D1:D1:5F:F4

非認証の証明書 (秘密鍵が含まれている) が作成され、キーストアに保存されます。証明書の作成に使用したパスワードを使用した場合にのみ、キーストア内の証明書にアクセスできます。このパスワードは [SSL ウィザード] の手順 1 で入力しました。

JRun で、設定されたキーストアに証明書を保存する手順は次のとおりです。キーストアにはサーバー証明書が保存され、そのセキュリティが維持されます。各 JRun サーバーには独自のキーストアがあり、JRun サーバーは証明書を 1 つずつサポートしています。

- 10 CA に CSR を提出する場合は、95 ページの「CSR を提出します (オプション)」の手順 2 に戻ります。それ以外の場合で、JRun によって作成される非認証の証明書を使用するには、96 ページの「JRun サーバーを再起動します。」の手順 5 から開始してください。

## サーバー証明書の認証 (オプション)

一度 JMC で CSR を作成して CA に提出し、CA から 検証済みのサーバー証明書が返されたら、これを JRun サーバーにインストールするか、または認証する必要があります。

### メモ

CSR を CA に提出していないのに検証済みのサーバー証明書を受領した場合は、この手順を実行しないでください。JRun によって作成された非認証の証明書を使用する場合、この証明書はすでに JRun サーバーにインストールされ、JRun サーバーのキーストアに保存されています。

### JRun サーバーにサーバー証明書をインストールするには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [SSL] をクリックします。  
[SSL 管理] パネルが次のように表示されます。



**JRun Admin Server: SSL 管理**

オプション	
項目	説明
 SSL Certify	SSL 証明書鍵の認定
 SSL Delete	SSL 証明書鍵の削除

**現在の証明書**

この証明書のオーナー	証明書の発行元:
sgensure CN=sgensure OU=qa O=allaire L=newton ST=massachusetts C=US	CN=sgensure OU=qa O=allaire L=newton ST=massachusetts C=US
シリアル番号	3b2fc973
証明書の有効	Tue Jun 19 17:51:47 EDT 2001 until: Mon Sep 17 17:51:47 EDT 2001
証明用の指紋	MD5: 87:FE:71:1C:67:24:6B:8D:3F:CC:60:34:93:6C:6A:90 SHA1: 68:8A:88:B5:37:E6:34:AF:DE:52:4C:3D:C2:7D:4C:8A:D1:D1:5F:F4

- 2 [SSL Certify] リンクを選択します。

[SSL 証明書の認証] パネルが次のように表示されます。

SSL 証明書の認証

名前: JRun Admin Server

ストアパスワード:

証明書:

ようこそ

このフォームを使用して、Verisign などのプロバイダから SSL 証明書の認証を受けてください。

- 3 [ストアパスワード] フィールドにキーストアパスワードを入力します。
- 4 テキスト エディタでサーバー証明書を開きます。これは CA から送信されたものです。
- 5 BEGIN 行と END 行を含めた、サーバー証明書のテキスト全体をコピーします。

- 6 証明書の内容を [証明書] フィールドに貼り付けます。  
パネルは次の例のように表示されます。

SSL 証明書の認証

名前: JRun Admin Server

ストアパスワード:

証明書:

```
gNVBAMTCGFj
aG1sbGVzMIGfMAOGCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCj9RhoR2
Swwte0nkvvC
He41zGJdZrRiLcxFRd6NnbLADeQfMvA5JhQ9dut1tCF18HzMR8g3iZf
/XnBkqKnJ3F
q0686XOj kot+PRsb/2w5ZDhtS+48hUcj qU9bmy/mv+oQ6HORzIzBwF
wIDAQABoAAw
DQYJKoZIhvcNAQEEBQAdgYEAdfH9SpHK80sc7G8BoRRAYexzUoUmjn
d5TVMOS4kK2
T/H54pRt1WuHOQJW7t1q2O6rNb+/r6HhziqtTua7p3QdKuocetUHsob
QT6hb2OIEEt
hIBb8nnNbUSRS/1LXrjESMjdzj+a1o87F6fd7u0eF18=
-----END NEW CERTIFICATE REQUEST-----
```

挿入

ようこそ

このフォームを使用して、Verisign などのプロバイダから SSL 証明書の認証を受けてください。

- 7 [挿入] をクリックします。  
非認証の JRun によって作成されたサーバー証明書が、キーストア内の検証済みのサーバー証明書に置き換えられ、[SSL 管理] パネルに戻ります。
- 8 96 ページの「CA ルートを設定します (オプション)」の手順 3 に戻ります。

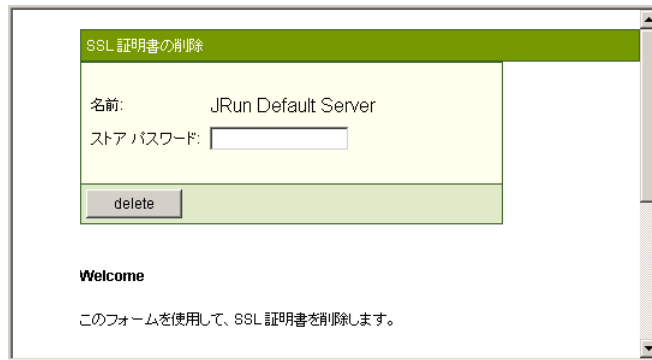
## 既存の証明書の削除

1 つの JWS でサポートする証明書は 1 つだけです。新しい CSR や非認証のサーバー証明書を作成する場合は、既存のサーバー証明書を削除する必要があります。これを実行するには、このセクションの手順を実行してください。

### 既存のサーバー証明書を削除するには

- JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [SSL] をクリックします。  
[SSL 管理] パネルが表示されます。
- [SSL Delete] リンクを選択します。この段階で作成済みのサーバー証明書がない場合、使用可能なリンクは [SSL Create] のみになります。

[SSL 証明書の削除] パネルが次のように表示されます。



- 3 [ストア パスワード] フィールドにキーストアのパスワードを入力し、[delete] をクリックします。

キーストアから証明書が削除され、[SSL 管理] パネルに戻ります。これで、新しい CSR や非認証のサーバー証明書を作成できます。

## 保護されていないポートの無効化 (オプション)

JWS で JWS に対する SSL を有効にすると、ほかのポートへのアクセスをブロックすることによって保護されているポート上のトラフィックだけを許可することがあります。既定では、[JRun Web サーバー] パネルの [Web サーバー ポート] フィールドに指定したポートを介して通常の HTTP 要求が許可されます。これは `local.properties` ファイルに `web.endpoint.main.port` プロパティとして保存されます。たとえば、ポート 8100 は default JRun サーバーの既定値です。

このセクションでは、保護されていないポートを閉じて、SSL が有効なポートからのみ JWS の要求を受け入れる方法について説明します。

### 保護されていないポートを無効にするには

- 1 テキスト エディタでその 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* のルート ディレクトリ `/servers/サーバー名/local.properties` に存在しています。
- 2 次の行をコメント化します (ポート番号は JRun サーバーの設定によって異なります)。  
`web.endpoint.main.port=8100`  
たとえば、次のようにコメント化します。  
`#web.endpoint.main.port=8100`
- 3 `local.properties` プロパティ ファイルを保存して、閉じます。
- 4 JRun サーバーを再起動します。

## 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すると、それぞれの JRun サーバーのログ ファイルの内容を制御できます。ログを記録しておく、Web サーバーのトラブルシューティングやロード バランスに便利です。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。

また、UNIX システムでは、jrun コマンドの `-console` オプションを使用して、コンソールに `Java.System.err` および `java.System.out` 出力を転送できます。詳細については、83 ページの「[jrun コマンドの使用](#)」を参照してください。

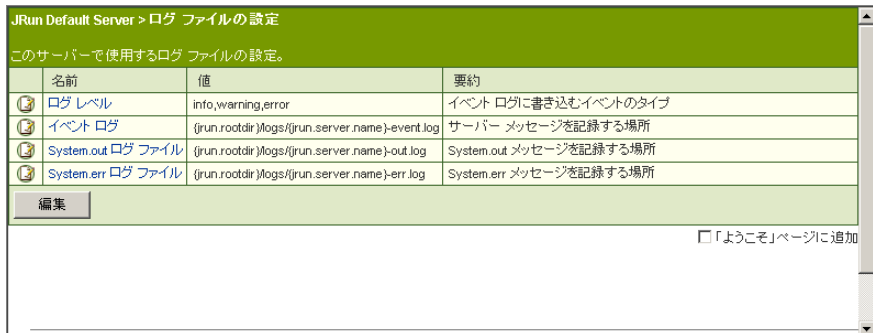
JVM のイベント ログの設定については、91 ページの「[Java Virtual Machine の設定](#)」を参照してください。JRun アプリケーションのイベント ログの設定については、141 ページの「[JRun アプリケーション イベント ログの構成](#)」を参照してください。

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、[マシン名] > [JRun サーバー名] > [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、右側ペインにプロパティを入力します。

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記録レベルをそれぞれ選択します。既定では Info、Error、および Warning が設定されています。ほかにも 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。
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 変更を適用するには、[更新] ボタンをクリックします。
- 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、およびテストを実行できます。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化する必要がなくなります。また、複雑なドライバおよび URL を指定する 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。

- 「[新しい JDBC データ ソースの追加](#)」 [110 ページ](#)
- 「[JDBC データ ソースの編集](#)」 [111 ページ](#)
- 「[よくある問題とその解決方法](#)」 [113 ページ](#)

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。

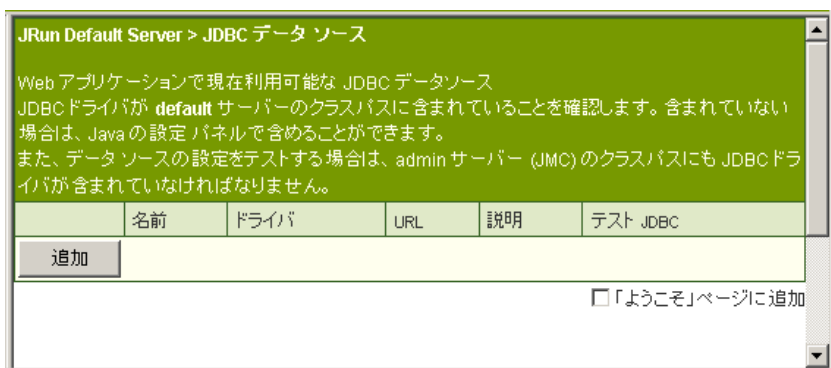
## 新しい JDBC データ ソースの追加

JDBC ウィザードを使用して、Web アプリケーション用のデータ ソースを追加します。

### 新しい JDBC データ ソースを追加するには

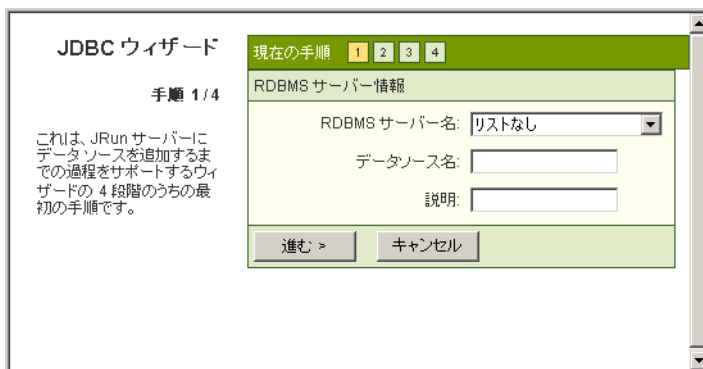
- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [JDBC データ ソース] をクリックします。

[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] ボタンをクリックします。

右側ペインに 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 サーバー名または [リストなし] を選択します。表示されるウィザードのフィールドは、選択したデータベース サーバーによって異なります。データベース名と説明 (オプション) を入力します。

JRun の Developer 版、Advanced 版、または Enterprise 版のライセンスを所有している場合は、次のパネルで JRun JDBC ドライバか、またはベンダから供給されたドライバのどちらを使用するか選択できます。

JRun JDBC ドライバによってサポートされているデータベースの一覧については、[xv ページの「JRun JDBC ドライバのデータベース必要条件」](#)を参照してください。JRun JDBC ドライバの使用方法については、『JRun JDBC Drivers User's Guide and Reference』を参照してください。

- 4 [進む] をクリックします。
- 5 次のパネルで JDBC ウィザードに必要な構成情報を指定し、[進む] ボタンをクリックして次の手順に進みます。JDBC ウィザードの各ページについて、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 終了したら、[完了] ボタンをクリックします。
- 7 JRun JDBC ドライバを除くすべてのドライバについて、選択した 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、データベースドライバの JAR ファイルを追加します。


admin JRun サーバーのクラスパスに JAR ファイルを追加しない場合、[JDBC データ ソース] パネルの [test] ボタンは機能しませんが、ドライバは正常に機能します。JRun サーバーのクラスパスの編集の詳細については、[91 ページの「Java Virtual Machine 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 [test] ボタンをクリックして、データ ソースの接続をテストします。エラーが発生する場合は、[113 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 ソースを編集したり、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、[マシン名] > [JRun サーバー名] > [JDBC データ ソース] をクリックします。  
[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 ソースの名前をクリックするか、名前の隣にある編集アイコン  をクリックします。

JDBC データ ソースの編集ウィンドウが開きます。

- 3 次の表の説明に従ってフィールドを編集します。

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。このフィールドは必須です。
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」と入力します。このフィールドは必須です。
URL	データ ソースを示す 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。このフィールドは必須です。
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)	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。既定値は 5 分です。
間隔 (秒)	有効期間が終了した場合にデータ ソース接続を閉じるまでの待機時間を秒単位で入力します。既定値は 60 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。 入力の形式は <i>name=value</i> です。 詳細については、データベース ベンダ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] ボタンをクリックします。
- 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

新しいデータ ソースをテストしたときにエラーが発生する場合は、JRun/logs ディレクトリの *JRun* サーバー名-out.log ファイルを表示してください。このセクションでは、この log ファイルで発生する可能性のある一般的な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ります。クラスパスが正しい JAR ファイルを指しており、その 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か、またはデータベース自体の優先設定のいずれ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てください。また、サーブレットからのデータソースをテストしている場合、サーブレット コードを確認して、データベースに正しい名前とパスワードが渡されているかどうかを確認します。

### Naming Exception: *datasource* not found

作成した新しいデータ ソースが JRun サーバーによって認識されません。JMC に表示される場合もありますが、サーバーを再起動しないとサーブレット コンテキストによって認識されません。

JRun サーバーを再起動します。

```
% jrun -restart default
```

## Web サーバーの設定

JRun では、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、**admin** サーバーの JWS インスタンスと **default** サーバーの JWS インスタンスの 2 つが実行されていることを表します。

新しい 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、この新規サーバー上で要求への対応をすぐに開始できます。ただし、ほとんどの開発環境においては、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 ではそれらの外部 Web サーバーとの接続を制御します。この接続は、JRun コネクタウィザードを使用して作成できます。

---

### 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 管理コンソールへのアクセス方法を変更できます。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、各 JRun サーバーとその JWS、または外部 Web サーバーの間の接続を設定する方法について説明します。

## 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。並行処理では、HTTP 要求をプールして分配する方式を定義します。

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。2000 個の同時要求をサポートする必要がある場合、本当は 2000 人の同時ユーザがいる状態を考えているのであって、この状態で一度に作成される要求は 100 個だけかもしれません。JRun は、各要求にスレッドを 1 つ割り当てます。

非常にボリュームの大きいインターネットサイトを実行していない限り、通常は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。それは次の設定値です。

- アイドルスレッドのタイムアウト
- スレッドの最小数
- 最大アクティブ要求
- 最大同時要求

Web サイトで頻繁にトラフィックの負荷が急増するような環境では、最小スレッド数を多めに設定すれば、Web サイトのトラフィックの負荷が急増しても一連のスレッドを作成する必要がありません。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も設定できます。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 サーバーの平均応答時間が 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、新しい要求を拒否せずに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前に、トラフィックのパターンを実際に観察できることを確認します。不用意に設定値を変更すると、リソースを浪費する場合があります。

---

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 や外部 Web サーバーの並行処理設定値の編集については、それぞれ [117 ページ](#)の「[JWS の設定](#)」および [119 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 ウィザードを使用すると、Web サーバーと JRun サーバーの間にリンクを設定できます。この接続は、JRun 接続モジュール (JCM) の形式になります。JRun の標準インストールには、default JRun サーバーを JWS に接続する 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

多くの場合、外部 Web サーバーは default JRun サーバーに接続します。admin JRun サーバーは固有の 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 アプリケーションの公開が可能です。default サーバーで JRun デモ アプリケーションを実行します。

---

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。

- [「Apache の接続」 31 ページ](#)
- [「IIS 3.0/PWS の接続」 36 ページ](#)
- [「IIS 4.0/5.0 の接続」 39 ページ](#)
- [「Netscape/iPlanet への接続」 47 ページ](#)
- [「WebSite Pro への接続」 54 ページ](#)
- [「Zeus Web サーバーの接続」 63 ページ](#)

## 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。

コネクタ ウィザード

現在の手順 1 2 3 4

手順 1/4

JRun と外部 Web サーバーの接続を設定します。

これは、JRun サーバーをサードパーティの Web サーバーに接続するまでの過程をサポートするウィザードの 4 段階のうちの最初の手順です。接続する JRun サーバーとサードパーティの Web サーバーを選択してください。

JRun Serverに関する情報

JRun サーバーの名前: JRun Admin Server  
JRun Default Server

サードパーティの Web サーバーに関する情報

Web サーバーの種類: Apache Web Server

Web サーバーのバージョン: 1.3.2

Web サーバーのプラットフォーム: intel-win

< 戻る 次へ > キャンセル

- 3 各手順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 ウィ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 を要求して、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、66 ページの「コネクタ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

インストール処理時に、JRun によって 2 種類の JRun Web サーバー (JWS) がセットアップされます。admin JRun サーバーに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ため、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。

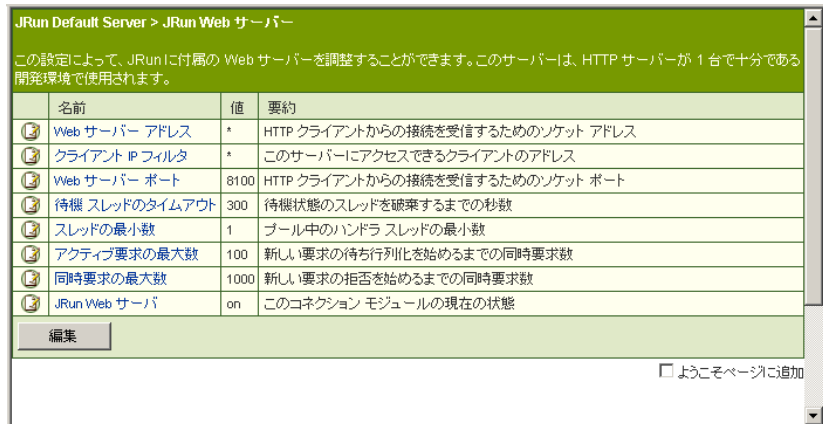
JWS はメモリ使用量の少ない Java Web サーバーです。JWS は実際の運用環境での使用には適していません。

ここでは、JWS の JRun 接続モジュールの設定方法について説明します。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、[119 ページの「外部 Web サーバーの設定」](#)を参照して、JCM のエンドポイント 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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [JRun Web サーバー] をクリックします。

[JRun 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フィールドを編集します。

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受信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 サーバーが JWS 上でバインドする相手側 IP アドレスの一覧を入力します。  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 アドレスの一覧を入力します。ここで指定されていない IP アドレスからの要求はすべて無視されます。  既定値は * で、JWS はすべての IP アドレスからの要求に応答します。
Web サーバー ポート	TCP ポート番号を入力します。この JWS では、このポートで HTTP 要求が受信されます。  admin JRun サーバーの JWS の既定値は 8000 です。default JRun サーバーの JWS の既定値は 8100 です。
アイドルスレッド のタイムアウト	JRun によりスレッドが破棄されるまでのアイドル状態の待機時間を秒単位で指定します。このパラメータには、JWS がビジー状態になってから何秒後に静止状態に戻るのかを指定します。スレッドが破棄されるたびに、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を使用します。JRun では要求に対して個別に新しいスレッドを作成する代わりに、新しい要求に使用できるハンドラ 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 サーバーに対してさまざまな要求が行われるたびに、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とれています。  既定値は 300 秒です。
スレッドの 最小数	スタート アップ時に初期プールで作成されるスレッド数を入力します。システムの稼動に従ってスレッドが作成されたり破棄されますが、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 (同時要求の最大数まで)、スレッドがそれらの要求を処理できるまで遅延します。  このパラメータは、JWS 上の同時発生数を制限するための JRun の主要なメカニズムです。既定値は 100 です。



プロパティ	説明
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破棄されます。 既定値は 1000 です。
JRun Web Server	この JWS を使用する場合は、このチェック ボックスをオンにします。この JWS を使用しない場合は、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 ウィザードを使用して JRun サーバーが外部 Web サーバーに正しく接続されている場合に、当該 JWS をオフにできます。  admin JRun サーバーの JMC アプリケーションの場合は、JWS をオフにしないでください。

- 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。

## 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 サーバーとの接続については、[115 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、最終的には JRun サーバーと外部 Web サーバー間の接続は、JRun 接続モジュール (JCM) によって管理されます。

JRun サーバー (default サーバーなど) 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、このセクションでの説明に従って JCM を調整します。

### メモ

外部 Web サーバーの一部の設定の変更は、JRun に組み込まれている JWS では可能ですが、JMC からは変更できません。Web サーバーの文書を参照してください。

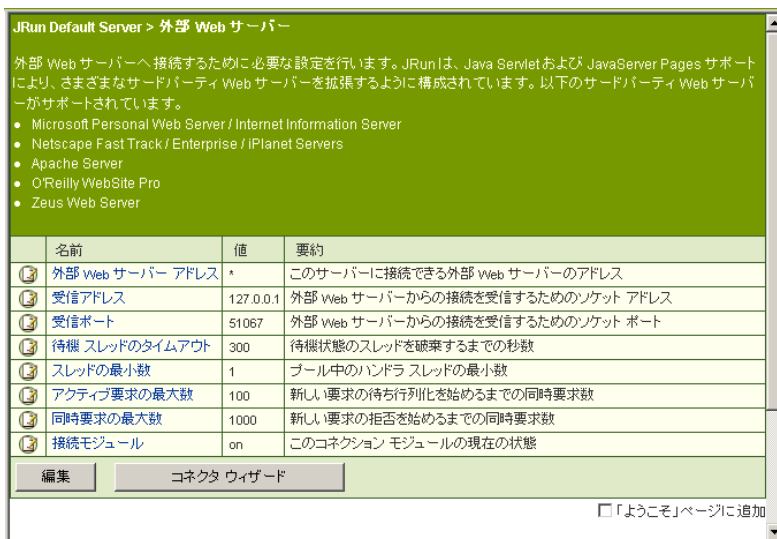
## 外部 Web サーバーの JCM を設定するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [外部 Web サーバー] をクリックします。

### メモ

外部 Web サーバーに JRun サーバーを接続する際にコネクタ ウィザードが実行されていないと、コネクタ ウィザードを実行するように要求されます。

[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フィールドを編集します。

プロパティ	説明
外部 Web サーバー アドレス	IP アドレスを入力します (複数の場合はカンマで区切ります)。この JCM は、このアドレス一覧からの要求のみを JCM に渡します。 既定値は * です。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 アドレスがある場合 (マルチホーミング) には、この機能が便利です。 既定値は * です。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。

プロパティ	説明
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 が使用する固有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。
アイドル スレッドの タイムアウト	JRun によりスレッドが破棄される前に、待機する時間を秒単位で指定します。このパラメータでは、Web サーバーがビジー状態になってから何秒後に静止状態に戻るのかを指定します。スレッドが破棄されるたびに、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を使用します。JRun では要求に対して個別に新しいスレッドを作成する代わりに、新しい要求に使用できるハンドラ 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 サーバーに対してさまざまな要求が行われるたびに、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とれています。 既定値は 300 秒です。
スレッドの最小数	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棄されますが、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	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。このパラメータは、外部 Web サーバー上の同時発生数を制限するための JRun の主要メカニズムです。 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破棄されます。 既定値は 1000 です。
接続モジュール	この JCM を使用する場合は、このチェック ボックスをオンにします。この JCM を使用しない場合は、チェック ボックスをオフにします。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。そのため、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、エラーが発生します。

- 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。

## 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、静的な HTML ファイル、画像など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ているディレクトリ構造に従って配置します。

JMC によって JRun の実装にアプリケーションを追加する方法は 2 種類あります。JMC の使用により、次のことが実行できます。

- 公開オプションによる、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) ファイルの作成および公開
- 作成オプションによって JSP やサーブレットを配置できる空のアプリケーションの新規作成

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。Web アプリケーションの記述の詳細と、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。

## 既定のアプリケーション

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。

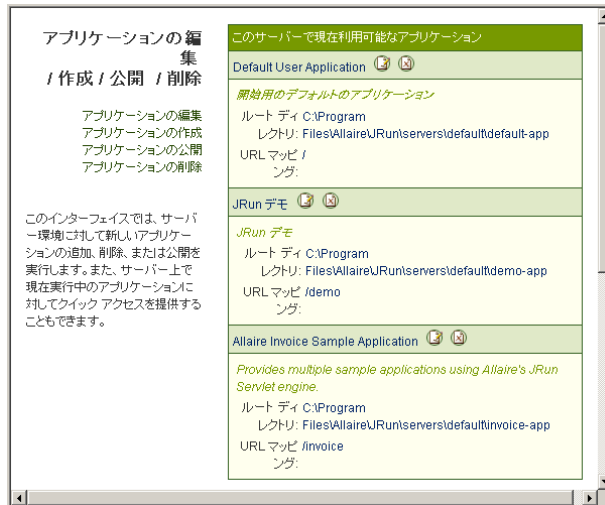
- / にマッピングされた admin JRun サーバーの JRun 管理コンソール アプリケーション (jmc-app)
- /demo にマッピングされた default JRun サーバーの JRun Demo アプリケーション (demo-app)
- / にマッピングされた default JRun サーバーの Default User Application (default-app)。default-app は、すばやく起動して実行できます。



また、JRun には次の Web アプリケーションも含まれています。

- default JRun サーバーの Invoice アプリケーション (invoice-app)
- admin JRun サーバーの RDS アプリケーション (web-rds)

## アプリケーション パネル

アプリケーション パネルでは、JMC での Web アプリケーションとともに一般的なタスクを実行できます。アプリケーション パネルにアクセスするには、[マシン名] > [JRun サーバー名] > [Web アプリケーション] をクリックします。



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、編集  および削除  機能へのクイック リンクが用意されています。

このセクションでは、次の項目について説明します。

- 「アプリケーションの作成」 [124 ページ](#)
- 「アプリケーションの公開」 [125 ページ](#)
- 「アプリケーションの編集」 [128 ページ](#)
- 「アプリケーションの削除」 [130 ページ](#)
- 「アプリケーションパスのマッピング」 [131 ページ](#)
- 「アプリケーションホストの作成」 [132 ページ](#)
- 「アプリケーションパラメータの追加」 [134 ページ](#)
- 「サーブレットディレクトリの追加」 [135 ページ](#)
- 「ファイル設定の変更」 [138 ページ](#)
- 「JSPコンパイラの構成」 [139 ページ](#)
- 「JRunアプリケーションイベントログの構成」 [141 ページ](#)
- 「MIMEタイプのマッピング」 [142 ページ](#)
- 「セッショントラッキングの構成」 [143 ページ](#)

## 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 サーバーにそのアプリケーションを登録できます。このプロセスにより、そのアプリケーションに対して、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 JMC の左側ペインで、[ マシン名 ] > [ JRun サーバー名 ] > [ Web アプリケーション ] をクリックします。  
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。

**Web アプリケーションの作成**

メイン ページに戻る  
アプリケーションの編集  
アプリケーションの作成  
アプリケーションの公開  
アプリケーションの削除

**注意:**  
新しい Web アプリケーションを有効にするには、JRun サーバーを再起動する必要があります。

**Web アプリケーションの作成**

Web アプリケーションの情報

JRun サーバーの名前:

アプリケーション名:

アプリケーション ホスト:

アプリケーションの URL:

アプリケーションのルート ディレクトリ:  [参照](#)

ようこそ

このウィザードでは空の Web アプリケーションを作成します。

- 3 次の表の説明に従ってフィールドを編集します。

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。 この名前は 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は、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、既定値である [All Hosts] 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、 <a href="#">132 ページの「アプリケーションホストの作成」</a> 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ときクライアントが使用する URL 接頭辞を入力します。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。このディレクトリ構造が存在しない場合は、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、 <i>JRun</i> のルート ディレクトリ/ <i>servers/</i> サーバー名です。

- 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 では、アプリケーションの WAR ファイルを既存の JRun サーバーに公開できます。WAR ファイルは JSP、サーブレット、画像など Web アプリケーションをサポートするファイルが、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。

さらに、この機能を使用すると、サーブレット 2.2 仕様に準拠していれば `.war` ファイルがなくてもアプリケーションを登録できます。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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] をクリックします。

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] リンクをクリックするか、または [マシン名] > [JRun サーバー名] をクリックしてページの上にある [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。

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	公開する Web アプリケーションがある場所のパスを入力します。または、[参照] をクリックして JRun のディレクトリ リーダを使用します。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。 この名前は 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、既定値である [All Hosts] を選択します。 詳細については、 <a href="#">132 ページの「アプリケーション ホストの作成」</a> を参照してください。
アプリケーション URL	この Web アプリケーションにアクセスするときにクライアントが使用する URL 接頭辞を入力します。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 リーダーを開きます。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。このディレクトリ構造が存在しない場合、JRun によって作成されます。 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 のルート ディレクトリ/servers/サーバー名/アプリケーション名です。

- 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は、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、[54 ページの「サーブレットを実行するための URL 接頭辞のマッピング」](#)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 サーバー] > [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 JMC の左側ペインで、[ マシン名 ] > [ JRun サーバー名 ] > [ Web アプリケーション ] をクリックします。

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。

The screenshot shows the 'Web アプリケーションの編集' (Edit Web Application) panel in the JRun management console. The panel is divided into two main sections: a sidebar on the left and a main form on the right.

**Sidebar (Left):**

- Web アプリケーションの編集 (Edit Web Application)
- メイン ページに戻る (Return to Main Page)
- アプリケーションの編集 (Edit Application)
- アプリケーションの作成 (Create Application)
- アプリケーションの公開 (Publish Application)
- アプリケーションの削除 (Delete Application)

**Main Form (Right):**

**Web アプリケーションの情報 (Web Application Information)**

アプリケーション名:  (Dropdown menu with options: default-app, demo-app, JRUNTEST, EJBQA, wwwwwa)

アプリケーションの表示名:

アプリケーションの説明:

アプリケーション ホスト:  (Dropdown menu)

アプリケーションの URL:

アプリケーションのルート ディレクトリ:  [参照](#)

**ようこそ**

このウィザードは、Web アプリケーション情報を更新します。

- 3 [ アプリケーション名 ] リスト ボックスからアプリケーションを選択します。ほかのフィールドは、JRun によって、アプリケーションの現在の設定が指定されます。

- 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。

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。 この名前は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	アプリケーションの識別に役立つように、アプリケーションの説明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は、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、既定値である [All Hosts] 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、 <a href="#">132 ページの「アプリケーション ホストの作成」</a> 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ときクライアントが使用する URL 接頭辞を変更します。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 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。 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

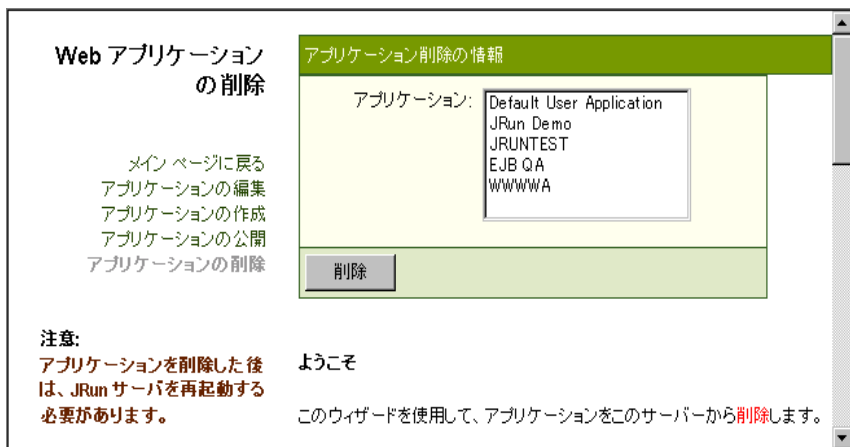
既存のどのアプリケーションも、JMC を使用して JRun サーバーから削除できます。

### 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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] をクリックします。  
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 ボックスに表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] 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、内部のディレクトリ構造をクライアントに対して非表示にできます。この例では、`c:/mydocs/temp` に保存されたファイルを要求する場合、URL `http://www.yourdomain.com/foo/` ドキュメント名を入力します。ほかのディレクトリのマッピングの詳細については、[135 ページの「サーブレット ディレクトリの追加」](#)を参照してください。

### メモ

外部リソースを参照するために JRun で内部的に使用する仮想マッピングがあらかじめ用意されています。このマッピングを変更したり削除しないでください。

### 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、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [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。  
仮想マッピングの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、JRun によって 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`」と入力します。このフィールドでは JRun 変数を使用できます。
- 5 パスを削除するには、[削除する] 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 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関連付けることができる Web サイトのホスト名は 1 つだけです。さらに、同一のホスト内のほかのアプリケーションを参照するために、サブレットがサブレット自身のコンテキスト情報を使用できる必要があります。

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 ホストという概念を導入しています。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できる一連の DNS ホスト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を、任意の数の Web サイトのホスト名にマッピングできます。アプリケーション ホストは、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でのみ必要になります。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できます。

仮想ホスト用のアプリケーション ホストの作成および使用手順は次のとおりです。

- 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 サーバーを設定します。詳細については、ご使用の 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、任意の数の Web サイトのホスト名を割り当てます。この手順は次のとおりです。
- 3 アプリケーションの作成または公開時に、そのアプリケーションのアプリケーション ホストを選択します。これにより、そのアプリケーションへのアクセスに使用される DNS 名が 1 組として定義されます。詳細については、[124 ページの「アプリケーションの作成」](#) または [125 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 にあるアプリケーション ホストの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バインドします。これによって、1 つのホストから Web サーバーにアクセスすると、そのホストにバインドさ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 JMC の左側ペインで、[ マシン名 ] > [ JRun サーバー名 ] > [ 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 ホストの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] 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 ホスト名] フィールドにホストのカンマ区切りリストを入力します。  
アプリケーション ホストに完全修飾ホスト名を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用される場合は、アプリケーション ホストを 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と外部からアクセス可能な場合は、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、そのアプリケーション ホストの [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] ドロップダウン リストから新規ホストを選択します。詳細については、[124 ページの「アプリケーションの作成」](#) または [125 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 既存のアプリケーションについては、アプリケーションの編集の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、[128 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できます。これらの変数は、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。サーブレットが再ロードされるまでは、初期化パラメータを変更できません。

### 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、サーブレット定義の編集ウィンドウの [Init 引数] フィールドを使用します。詳細については、[146 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の追加または変更を行うには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [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の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、そのアプリケーション変数の [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] ボタンをクリックします。
- 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で `ServletContext.getInitParameter("address")` を呼び出すと、値 `info@allaire.com` が返されます。



## サーブレット ディレクトリの追加

設定によっては、Web アプリケーションからアクセス可能な JRun 外部のディレクトリにサーブレットを保存することがあります。このセクションでは、JRun のファイル構造外にディレクトリを追加する方法について説明します。サーブレットはこのディレクトリからロードされ、サーブレット コンテナによって実行されます。この方法を使用すると、WAR ファイルディレクトリ構造を、たとえば、既存のプロジェクト構造にマッピングできます。

---

### メモ

このセクションで説明するオプションには、default JRun サーバーにのみ適用されるオプションもありますが、それ以外はすべての JRun サーバーに適用できます。

---

default JRun サーバーには2つのサーブレットディレクトリがあり、ここに JRun によってロードされたサーブレットクラスファイルが保存されます。これらのディレクトリは、次のとおりです。

- サーブレット 2.2 仕様によって定義されるディレクトリである、*JRun* ルートディレクトリ/*servers/default/default-app/WEB-INF/classes*。  
このディレクトリは、JRun サーバーに追加されているすべての Web アプリケーションに適用できます。
- JRun 2.3.3 ディレクトリ構造をシミュレートするディレクトリの、*JRun* ルートディレクトリ/*servlets*。  
このディレクトリは default JRun サーバーにのみ適用され、下位互換性維持のために追加されました。

サーブレットの保存が可能な別のディレクトリの追加方法はいくつかあります。

- 新しい Web アプリケーションを作成します (任意の JRun サーバー)。
- JRun サーバーのクラスパスにディレクトリを追加します (任意の JRun サーバー)。
- `webapp.properties` ファイルを使用します (default JRun サーバーのみ)。

これらの方法にはそれぞれ特徴があります。次のセクションでこれについて説明します。

## 新しい Web アプリケーションの作成による新しいサーブレット ディレクトリの追加

新しい Web アプリケーションの作成は、望ましいサーブレットディレクトリの追加方法です。これは、Java サーブレット 2.2 仕様に記述されている概要に準拠しています。これは最も移植性の高い方法であり、この方法によって、仕様 2.2 に準拠しているコンテナに Web アプリケーションを公開できます。Web アプリケーション作成の詳細については、124 ページの「アプリケーションの作成」を参照してください。これは、default JRun サーバーだけでなく、すべての JRun サーバーに適用できます。

## JRun のクラスパスへの新しいサーブレットのディレクトリの追加

ファイルシステムにあるいずれのディレクトリについても、JRun サーバーのクラスパスに追加できます。そのディレクトリ内のすべての JAR およびクラス ファイルは、追加後、JRun によって起動可能になります。これは、default JRun サーバーだけでなく、すべての JRun サーバーに適用できます。

---

### メモ

クラスパスからロードされたサーブレットは、変更すると再ロードされません。このため、サーブレット クラス ファイルを変更したときは常に、JRun サーバーを再起動する必要があります。

---

### JRun サーバーのクラスパスにサーブレット ディレクトリを追加するには

- 1 JMC を開きます。
- 2 [マシン名] > [JRun サーバー名] > [Java の設定] をクリックします。  
[Java の設定] パネルが表示されます。
- 3 [クラスパス] フィールドを選択します。  
Java の設定の編集ウィンドウが表示されます。
- 4 サーブレット ディレクトリを既存のクラスパスの引数に追加し、[更新] ボタンをクリックします。ディレクトリパスでは、必ずフォワード スラッシュを使用してください。たとえば、`c:\servletpost` ディレクトリを追加するには、次のようにします。  

```
{jrun.rootdir}/servers/lib  
{jrun.server.rootdir}/lib  
c:/servletpost
```
- 5 JRun サーバーを再起動します。

## webapp.properties ファイルを使用した新しいサーブレット ディレクトリの追加

JRun には、default-app アプリケーションのマッピングを変更できる `webapp.properties` ファイルがあります。このセクションでは、既存のアプリケーション マッピングに新しいサーブレット ディレクトリを追加する方法について説明します。これらの手順は、default JRun サーバーの default-app から起動されるサーブレットにのみ適用できます。

---

### メモ

この方法は、JRun 固有の `webapp.properties` ファイルを使用するため、3 つの中で最も移植性の低い方法です。

---

## webapp.properties ファイルを使用してサーブレット ディレクトリを追加するには

- 1 すべての JRun サーバーを停止します。  
手順の一部として、webapp.properties ファイルを編集します。すべての JRun サーバーを停止することによって、ユーザが行った変更が、JRun がメモリ内に保存しているサーバー設定値に上書きされることを防ぎます。
- 2 *JRun* のルート ディレクトリ/servers/default/default-app/WEB-INF/webapp.properties ファイルをテキスト エディタで開きます。
- 3 次の行を追加します。  
`webapp.path-mapping./WEB-INF/moreservlets=d:/servlets`

---

### メモ

この例では、追加するサーブレットのディレクトリは `d:¥servlets` です。  
`webapp.path-mapping` 内のフォワード スラッシュは意図的に使用しています。

---

- 4 `webapp.classpath` プロパティに `/WEB-INF/moreservlets` を追加します。次に例を示します。  
`webapp.classpath=/WEB-INF/servlets;/WEB-INF/classes;/WEB-INF/lib;/WEB-INF/jsp;/WEB-INF/moreservlets`
- 5 `webapp.properties` ファイルを保存します。
- 6 JRun サーバーを再起動します。
- 7 これで、`default-app` 内のほかのサーブレットにアクセスするのと同様に、`http://ホスト名/servlet/ServletName` のような URL を使用して `d:¥servlets` 内のサーブレットにアクセスできます。

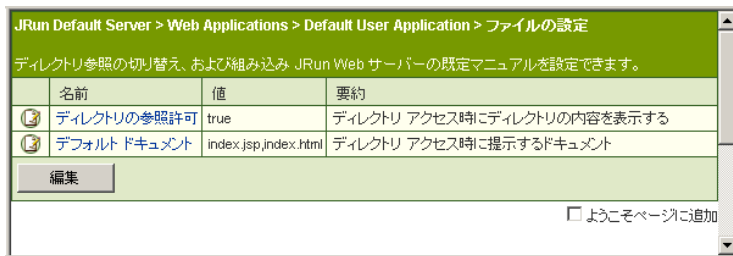
## ファイル設定の変更

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も制御できます。

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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [ファイルの設定] をクリックします。  
[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フィールドを編集します。

プロパティ	説明
ディレクトリの参照許可	<p>要求したファイルが存在せず、ディレクトリに既定のドキュメントもない場合に、ディレクトリ一覧を表示するには、チェックボックスをオン (true) にします。</p> <p>既定の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 (false) にします。</p> <p>既定値は true です。</p>
デフォルトドキュメント	<p>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既定のページのカンマ区切りリストを入力します。そのリストの順番が重要となります。</p> <p>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。</p>

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

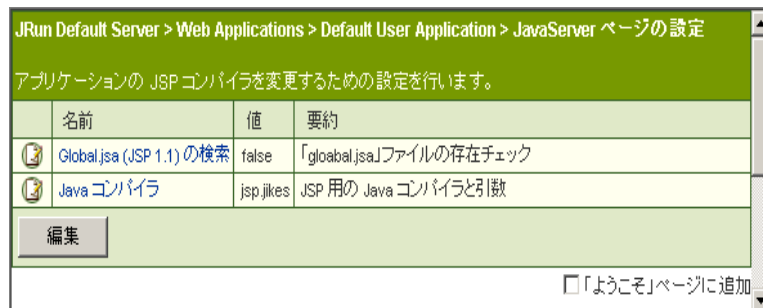
## JSP コンパイラの構成

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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [JavaServer Pages] を選択します。

[JavaServer ページの設定] 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 ページの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フィールドを編集します。

プロパティ	説明
global.jsa (JSP 1.1) の検索	<p>JSP の処理時に、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 (true) にします。true の場合、ディレクトリにある JSP ファイルに対する要求が Web サーバーによって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。</p> <p>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。</p> <p>既定値は、チェック ボックスがオフの false です。</p>
Java コンパイラ	<p>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 コンパイラを使用する場合は、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。次に例を示します。</p> <pre>D:%jdk1.1.7b%bin%javac -nowarn -classpath %c -d %d %f</pre> <p>または</p> <pre>jvc /cp:c %c /dest:%d %f</pre> <p>ここで、</p> <ul style="list-style-type: none"> <li>%c = classpath (Java クラスパスが使用されます)</li> <li>%d = codepath (コンパイルされたクラス ファイルの保存場所)</li> <li>%f = filename</li> </ul> <p>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。</p>

- 4 [更新] をクリックして、変更を適用します。
- 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できます。各アプリケーションによってログ 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に利用できます。

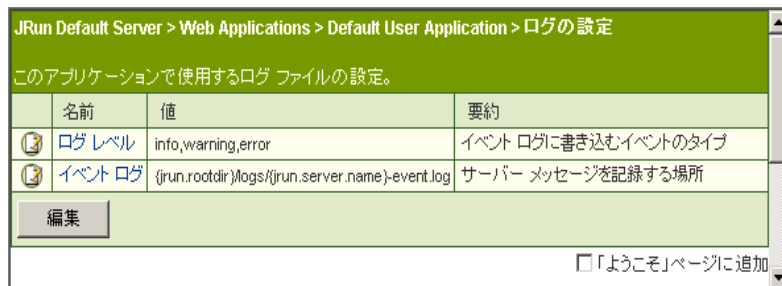
このセクションでは、JRun アプリケーションのイベント ログを構成する方法について説明します。詳細については、108 ページの「JRun サーバー イベント ログの設定」を参照してください。

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 イベント ログを構成するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [ログの設定] を選択します。

アプリケーション別の [ログの設定] 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。

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記録レベルをそれぞれ選択します。既定では、info、error、warning が設定されています。ほかにも 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] ボタンをクリックします。
- 5 JRun サーバーを再起動します。

## MIME タイプのマッピング

JRun では、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。

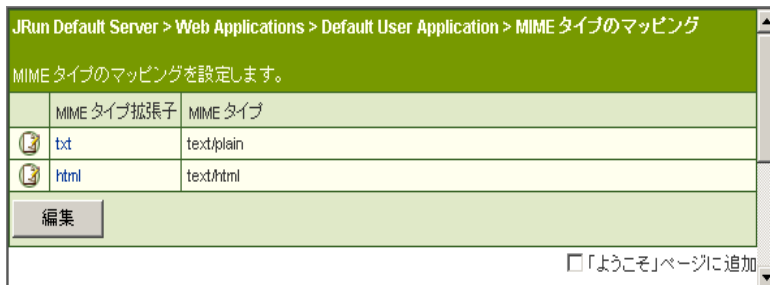
MIME タイプに基づいて、1つのサーブレット、またはサーブレット チェーンを開始することもできます。詳細については、153 ページの「MIME タイプフィルタリングでのサーブレットのチェーン化」を参照してください。

このセクションでは、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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [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の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、HTML などの MIME タイプと関連付けるファイル拡張子を入力します。
- 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] ボタンをクリックします。
- 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。

### アプリケーションのセッション トラッキング プロパティを編集するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [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。

web.xml セッションの設定を編集します。

名前	値	要約
記憶装置チェック間隔 (秒)	10	セッションを記憶装置に書き込む間隔のチェック
セッションの最大数	9999999	セッションの最大数
セッション保存のディレクトリ	{webapp.rootdir}/WEB-INF/sessions	セッション情報を保存するためのディレクトリ
セッションクッキー最大保存時間	-1	セッションクッキー保存時間 (秒)
安全な接続のみ	false	https を使用したセッションクッキー送信
セッションクッキーの使用	true	クッキーを使用したユーザセッショントラッキング
セッションクッキードメイン		クッキードメイン名フィルタ
セッションクッキーコメント	"JRun Session Tracking Cookie"	セッションクッキーコメント
セッションクッキーパス	/	セッションクッキー URL フィルタ
セッションクッキー名	jsessionid	JRun セッションクッキー名
セッションタイムアウト (分)	30	セッションの最大アイドル時間 (分)
セッションバースタンスエンジンの使用	true	セッションの保存情報

編集

「ようこそ」ページに追加

- 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フィールドを編集します。

プロパティ	説明
記憶装置チェック 間隔 (秒)	セッションが記憶装置に書き込まれる間隔を指定します。 既定の設定は 10 です。
セッションの 最大数	古いセッションが記憶プロバイダにスワップされるまでメモリに保持され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。  JRun では、LRU アルゴリズムを使用して、最近使用されていないセッションのトラックを保持し、セッション パーシスタンス メカニズムを使用してセッション データを記憶領域 (既定ではファイル) に対して直列化します。  セッションがスワップされた後で再度要求されると、これをメモリに再度読み込んで使用します。この設定値はチェック時にセッションによって使用されるメモリのサイズを保持します。
セッション保存の ディレクトリ	セッションを保存するディレクトリの絶対パスを入力します。 既定値は次のとおりです。 <code>{webapp.rootdir}/WEB-INF/sessions</code>
セッション クッキー 最大保存時間	セッション トラッキングに使用するために、ブラウザによって JRun クッキーが保持される時間を秒単位で指定します。 次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 ブラウザによって、すぐにクッキーが削除されます。 既定値は -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、このチェック 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にのみ、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 クッキーの使用	クッキーを使用してユーザ 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存されます。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存されます。
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。

プロパティ	説明
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一ディレクトリまたはサブ ディレクトリを参照する URL は、そのクッキーを参照できます。既定の設定は / です。
セッション クッキー名	JRun セッション クッキーの名前を入力します。 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)。既定値は 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にのみ保存されます。  仮想マシンにセッション データを保存するだけならば、チェック ボックスをオフにします (false)。これは、可用性の高い機能です。 既定値は true です。

- 4 変更を適用するには、[更新] ボタンをクリックします。
- 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と、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

サーブレットの追加または登録を行う場合、JMC でこれを定義してから 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を認識させる必要があります。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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サーブレット定義] を選択します。

[サーブレット定義] パネルを表示します。



- 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。

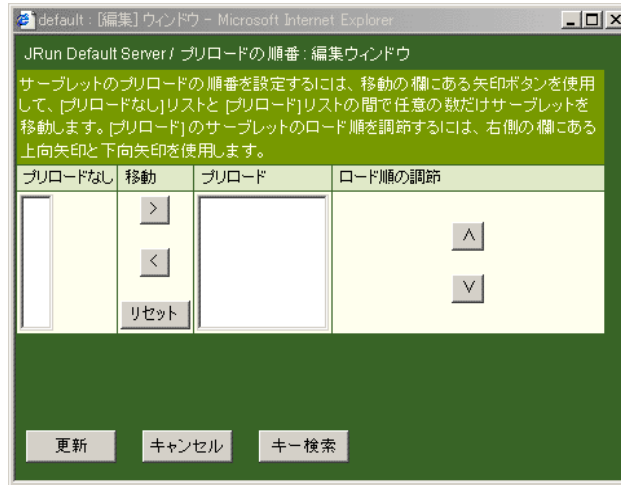
- 3 次の表の説明に従ってプロパティを入力します。

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。このフィールドは必須です。
クラス名	サーブレットの完全修飾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。このフィールドは必須です。
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 ファイルに保存されます。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。
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 ファイルに保存されます。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、 <a href="#">134 ページの「アプリケーションパラメータの追加」</a> 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] ボタンをクリックします。

- 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 [プリロードの順番の設定] をクリックします。

[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] リストと [プリロードなし] リストの間でサーブレットを移動します。次に、[ロード順の調節] 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 [更新] ボタンをクリックして、変更を適用します。JRun により 1 から始まるプリロード番号が各サーブレット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の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。

## サーブレットへの要求マッピング

JMC を使用すると、URL パターンにサーブレットをマッピングできます。

JRun で HTTP の要求を受信すると、サーブレット エンジンによって、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、要求は既定のアプリケーションに渡されます。

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。次のセクションでは、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 サーバーによるファイルの処理方法と、URL 要求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 を使用すると、サーブレットを URL 接頭辞にマッピングすることによって、HTTP の要求からサーブレットにアクセスする方法を制御できるようになります。JMC の使用により、次のことが実行できます。

- 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` にマッピングして `JRunDemoServlet` を開始できます。
- 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、お勧めできません。この方法でマッピングすると、リソース参照にエラーが発生します。

既定の設定では、`/servlet` 接頭辞を含む HTTP の要求は

`http://www.yourdomain.com/demo/servlet/SimpleServlet` と同様に、invoker にマッピングされます。サーブレットにマッピングできる URL の数に制限はありません。

### URL 接頭辞にサーブレットをマッピングするには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サーブレット URL のマッピング] をクリックします。  
[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] フィールドに、サーブレットを呼び出す URL 接頭辞を入力します。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既定サーブレットとするには、「/」を入力します。
- 4 [呼び出されたサーブレット] フィールドに、URL 接頭辞によって呼び出す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。エイリアスを定義するには、[151 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] ボタンをクリックします。
- 7 JRun サーバーを再起動します。

## 接尾辞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できます。既定では、拡張子 \*.jsp は暗黙的に invoker サーブレットにマッピングされていますが、明示的にマッピングすることによって書き換えることができます。

### 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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サーブレット URL のマッピング] をクリックします。  
[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。この拡張子を使用し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 [呼び出されたサーブレット] フィールドに、ファイル拡張子によって呼び出すサーブレットを入力します。このようなサーブレットは、サーブレットのエイリアスである場合もあります。[151 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] ボタンをクリックします。
- 6 JRun サーバーを再起動します。



## サブレットのエイリアス設定

サブレットの URL マッピングの編集ウィンドウを使用して、エイリアスでサブレットを参照できます。これにより、サ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できます。たとえば、**ShoppingCart** という名前のエイリアスに **shop\_05022001** という実名のサ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。

同様に、このサブレットにエイリアスを設定する方法でサブレットのチェーンも参照できます。詳細については、[152 ページの「エイリアスを使用したサ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つのサブレットに割り当てる方法について説明します。

### サブレットに対してエイリアスを設定するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サブレット URL のマッピング] をクリックします。  
[サブレット 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。  
サブレット URL のマッピングの編集ウィンドウが表示されます。
- 3 [仮想パス / 拡張子] フィールドに、サ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 [呼び出されたサブレット] フィールドに、エイリアスによって参照されるサ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] ボタンをクリックします。
- 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ほかのサーブレットの入力として使用できます。最も基本的なサーブレットチェーン化の方法は、要求を行うときにサーブレットを、カンマ区切りリストとしてパス情報に追加することです。

たとえば、次の要求によって Servlet1 が実行され、その出力が Servlet2 に送信されます。

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

この簡単なチェーン化方法では実用的で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 種類の方法があります。

- 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

後続のセクションで、JRun でサーブレットをチェーン化するこれら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、1 つの名前、つまりエイリアスで、チェーンを形成するサーブレットリストを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用してサーブレットをチェーン化するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サーブレット URL のマッピング] をクリックします。  
[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 [呼び出されたサーブレット] フィールドに、実行順に並べられたサーブレットをカンマ区切り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] ボタンをクリックします。
- 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のチェーン化

応答タイプの設定だけではなく、MIME タイプのマッピングを使用しても、サーブレット チェーンを実行できます。しかし、エイリアスを使用する場合のように、要求に基づいてどのサーブレットをチェーン化するか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 チェーンの実行をトリガする送信 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 にマッピングすると、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。

ある応答の出力が JRun サーバーでフィルタリングされ、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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [MIME タイプのチェーン化] をクリックします。

[MIME タイプのチェーン化] 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の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` の形式で入力します。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 [呼び出されたサーブレット] フィールドには、MIME タイプにより呼び出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カ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] ボタンをクリックします。
- 7 JRun サーバーを再起動します。

### MIME タイプの例: WML

JRun を WML (Wireless Markup Language) ページを読み込むように設定するには、JMC を使用して次の拡張子を MIME タイプにマッピングします。

拡張子	MIME タイプ
bmp	image/bmp
wbmp	image/vnd.wap.wbmp
wmls	text/vnd.wap.wmlscript
wml	text/vnd.wap.wml

Web アプリケーションの web.xml ファイルへのエントリは、次のようになります。

```
<mime-mapping>
  <extension>bmp</extension>
  <mime-type>image/bmp</mime-type>
</mime-mapping>
<mime-mapping>
  <extension>wbmp</extension>
  <mime-type>image/vnd.wap.wbmp</mime-type>
</mime-mapping>
<mime-mapping>
  <extension>wmls</extension>
  <mime-type>text/vnd.wap.wmlscript</mime-type>
</mime-mapping>
<mime-mapping>
  <extension>wml</extension>
  <mime-type>text/vnd.wap.wml</mime-type>
</mime-mapping>
```

## SSI タグレットの使用

JRun には、サーバー側インクルード (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 ファイルで固有のタグをフレキシブルに定義し、実装できます。

SSI は、以前はダイナミック コンテンツの作成に広く使用されていました。JRun では主に古い実装をサポートする目的で使用します。現在では、JavaServer Pages (JSP) および Java サーブレット テクノロジーがタグレット付きの SSI の代わりに使用されるようになり、機能的にも大幅に拡張されています。

このセクションでは、JRun で使用する SSI タグレットの構成方法について説明します。SSI タグレッ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。

### SSI を構成するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Web アプリケーション] > [アプリケーション名] > [サーバー側インクルード] を選択します。

[サーバー側インクルードの設定] パネルが表示されます。



- 2 [編集] をクリックします。  
サーバー側インクルードの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。

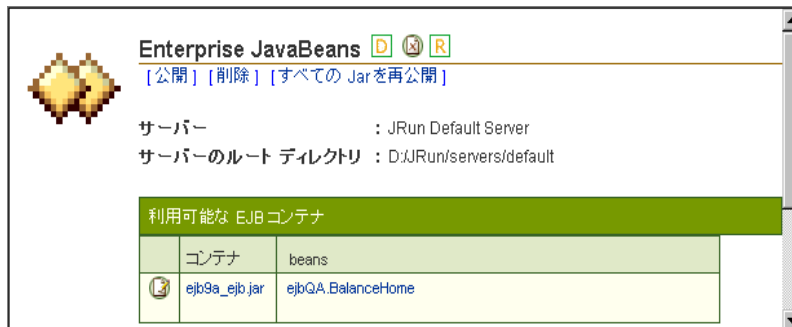
プロパティ	説明
タグレット名	タグレット名を入力します。たとえば、「foo」と入力します。サーブレットを呼び出す場合は、Web ページで次のコードを使用します。 <foo></foo>
サーブレット マッピング	タグレットによって参照されるサーブレットを入力します。たとえば、「SnoopServlet」と入力します。この結果、<foo>タグレットを Web ページで使用すると、SnoopServlet サーブレットが呼び出されます。

- 4 SSI タグ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。
- 5 変更を適用するには、[更新] ボタンをクリックします。
- 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 では、Enterprise JavaBeans (EJB) および EAR ファイルの公開をサポートしています。EJBを開発し、ホームとリモート インターフェイスを定義すると、公開の準備ができます。ただし、この「公開」という用語は、JRun の EJB に使用する場合と意味が若干異なるので注意が必要です。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および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、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] をクリックしてアクセスします。



このセクションでは、次のトピックについて説明します。

- 「EJB の公開」 156 ページ
- 「EJB の再公開」 158 ページ
- 「EJB の削除」 159 ページ
- 「EJB の構成」 160 ページ
- 「EAR ファイルの公開」 161 ページ

JRun での Enterprise JavaBeans 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 ファイルの一覧にある 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場合に必要とな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のプロパティを使用して、`runtime.properties` ファイルを作成します。`runtime.properties` ファイルは、実行時環境を確立するために JRun によって使用されます。
- `ejb-jar.xml` ファイルは変更されません。JMC ではその代わりに `deploy.properties` ファイルを変更して、公開者または管理者が、開発者によって設定された `ejb-jar.xml` 内のプロパティを上書きできます。

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は `/deploy` ディレクトリで使用可能でなければならないので、生成されたすべての出力は `/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。

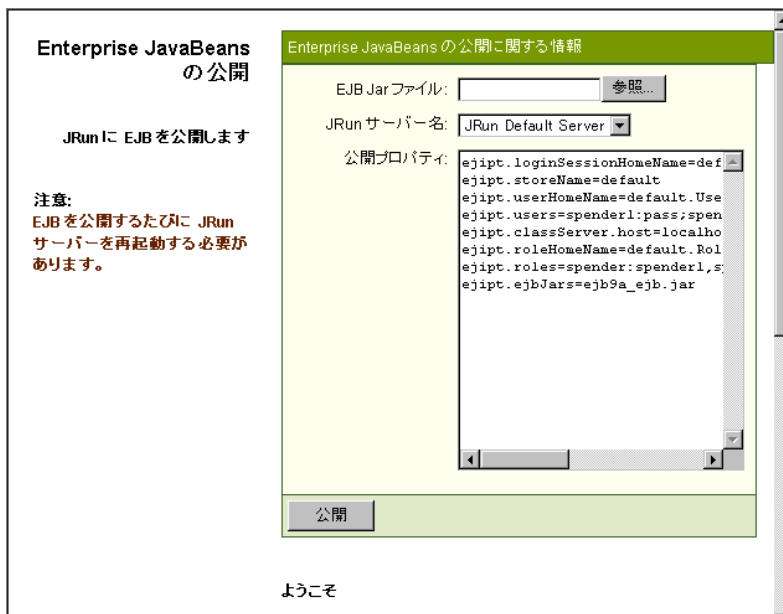
## EJB を公開するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] をクリックします。

[Enterprise JavaBeans] パネルが表示されます。

- 2 ページ最上部にある [公開] リンクをクリックします。または、[マシン名] > [JRun サーバー名] 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。

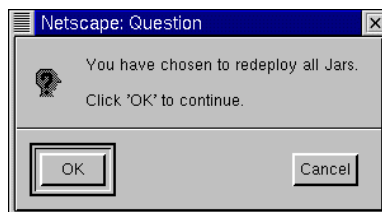
フィールド	説明
EJB Jar ファイル	EJB の JAR ファイルへのパスを入力するか、[参照] をクリックして、JRun の ディレクトリ リーダーを使用します。
JRun サーバー名	EJB を公開するサーバーを選択します。
公開プロパティ	<p>deploy.properties ファイルに保存されている EJB のサーバーレベルの公開プロパティを編集します。name=value の組み合わせの変更、追加、削除が可能です。EJB を公開するときに、変更を加えたファイルで JMC によって deploy.properties ファイルが上書きされます。</p> <p>JMC 公開ツールでは ejb-jar.xml ファイルは変更されません。開発者によって設定された ejb-jar.xml 内のプロパティを公開者または管理者によって上書きできるようにする deploy.properties ファイルがあります。</p>

- 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、*bean\_name.properties* ファイルを変更するほかのツールを使用して Bean のプロパティを変更する場合は、その EJB が含まれている 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 JAR を一括して再公開する方法について説明します。

- JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] をクリックします。  
[Enterprise JavaBeans] パネルが表示されます。
- ページ最上部にある [すべての Jar を再公開] リンクをクリックします。  
[OK] または [Cancel] をクリックするように要求するプロンプトが表示されます。





### メモ

すべての JAR ファイルのサーバーへの再公開に要する時間は、ファイルのサイズと個数によって異なります。

- 3 [OK] をクリックします。

JRun では、以前 JRun サーバーで公開された JAR ファイルがすべて再公開されます。

## EJB の削除

JAR ファイルから EJB を削除するには、次の手順を実行します。この手順では、ファイルがファイルシステムから実際に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。

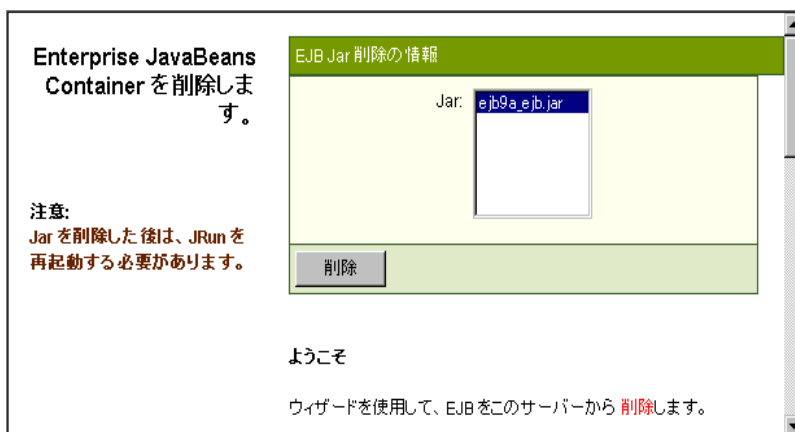
### EJB を削除するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] をクリックします。

[Enterprise JavaBeans] パネルが表示されます。

- 2 ページ最上部にある [削除] リンクをクリックします。

[Enterprise JavaBeans コンテナを削除します。] 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] ボタンをクリックして、JAR ファイルを削除します。
- 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設定できます。Bean コンテキストは、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の取得に使用します。コンテキストは Bean インスタンスの作成時に作成され、Bean インスタンスが存在する間はその Bean に関連付けられ、ほかの Bean インスタンスで使用することはできません。コンテキストには、Bean インスタンスについて、インスタンスのステートが変化したかどうかを示す情報などが記録されます。

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` プロパティを設定することによって管理できます。JMC では、この情報を Bean の `default.properties` ファイルに書き込みます。JMC では、ほかの 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

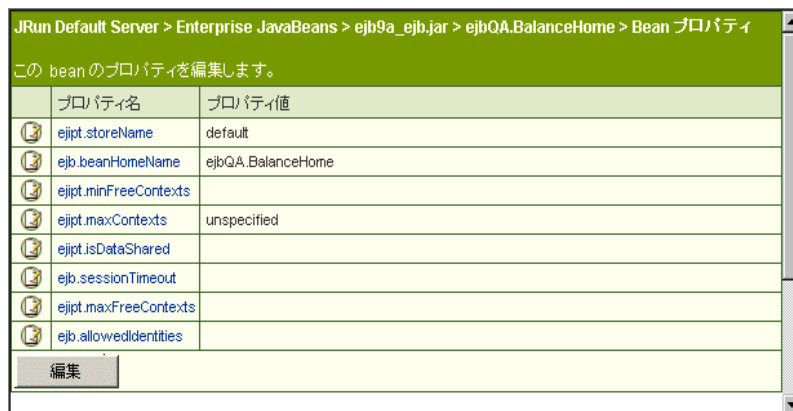
既存の Bean プロパティを変更する場合は、Bean の JAR ファイルの再公開が必要です。

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「Ejpt Properties API」のマニュアルや、『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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] > [Jar ファイル] > [Bean] をクリックします。

[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 JavaDocs ファイルに付属する『Ejpt Properties API』マニュアルの説明に従ってフィールドを編集します。
- 4 [更新] をクリックして、変更を適用します。
- 5 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [Enterprise JavaBeans] をクリックします。  
[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] リンクをクリックします。  
[OK] か [Cancel] をクリックするように要求するプロンプトが表示されます。

---

#### メモ

すべての JAR ファイルのサーバーへの再公開に要する時間は、ファイルのサイズと個数によって異なります。

---

- 7 [OK] をクリックします。  
JRun では、以前 JRun サーバーで公開された 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 (Enterprise Application アーカイブ) ファイルには、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すべてのファイルが組み込まれています。EAR ファイルは、JAR ファイルと同じツールを使用して作成します。J2EE アプリケーション公開時に、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、指定の環境に J2EE アプリケーションをインストールします。EAR ファイルのインストール時に、JMC によって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。この公開記述子から、JRun アプリケーション公開ユーティリティに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 JMC の左側ペインで、[マシン名] > [JRun サーバー名] をクリックします。  
[JRun Default Server] パネルが表示されます。



**JRun Default Server**  
[WAR 公開] [EJB 公開] [EAR 公開]

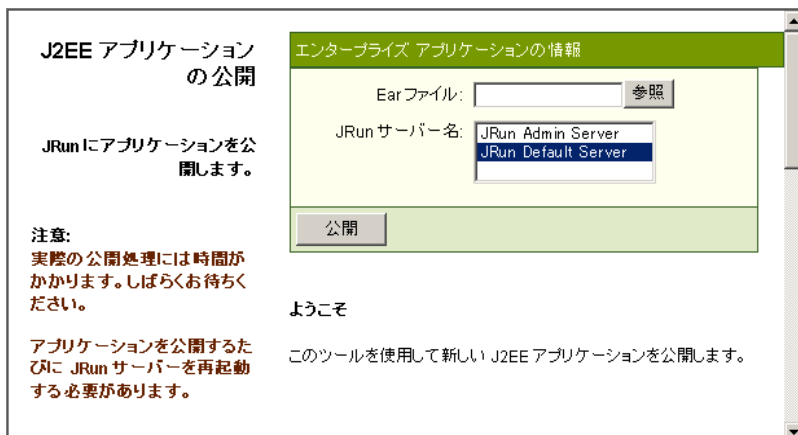
サーバー名 : default  
サーバー ルート ディレクトリ : C:/Program Files/Allaire/JRun/servers/default  
サーバーの説明 :  
サーバーのステータス : running

サーバー 再起動

フォルダ オプション	
項目	説明
JDBC データ ソース	Web アプリケーションで使用される JDBC データ ソースを定義します。
Java の設定	Java Virtual Machine の設定を定義します。
ログ ファイルの設定	ログ ファイルの設定を定義します。
アプリケーション ホスト	Web アプリケーションの提供で利用可能な仮想ホストを定義します。
外部 Web サーバー	JRun が外部 Web サーバーに接続する方法を定義します。
JRun Web サーバー	組み込み JRun Web サーバーを定義します。
Web アプリケーション	Web アプリケーションを管理します。
Enterprise JavaBeans	Enterprise JavaBeans を管理します。

Allaire, JRun, 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。

- 2 [EAR 公開] リンクをクリックします。  
[J2EE アプリケーションの公開] パネルが表示されます。



**J2EE アプリケーションの公開**

JRun にアプリケーションを公開します。

注意:  
実際の公開処理には時間がかかります。しばらくお待ちください。

アプリケーションを公開するたびに JRun サーバーを再起動する必要があります。

ようこそ  
このツールを使用して新しい J2EE アプリケーションを公開します。

エンタープライズ アプリケーションの情報

Ear ファイル:  参照

JRun サーバー名: JRun Admin Server  
JRun Default Server

公開

- 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。

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 リーダーを使用します。
JRun サーバー名	EAR ファイルの公開先の JRun サーバーを選択します。

- 4 [公開] ボタンをクリックして、EAR ファイルを公開します。
- 5 JRun サーバーを再起動します。

## ログ ファイル ビューアの使用

JRun にはログ ファイルビューアが組み込まれており、これを使用すると、JMC 内から JRun ログ ファイル内のエントリを表示できます。このログ ファイルビューアでは交換されたログ ファイル (default-event\_1.log など) を確認しません。現在のログ ファイルのみを読み込みます。

ログ ファイルの情報の量は、ログ ファイルの設定によって異なります。JRun サーバーレベルでログ レベルの設定値を変更する場合は、[108 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。アプリケーションレベルでログ レベルの設定値を変更する場合は、[141 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。

### JMC に JRun ログ ファイルを表示するには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [ログ ビューア] をクリックします。

[default ログ ファイルの表示] パネルが表示されます。

default ログ ファイルの表示

表示するエントリ数:

ログ タイプ:

開始行:

検索:

大文字小文字を区別して検索:

逆の順に表示:

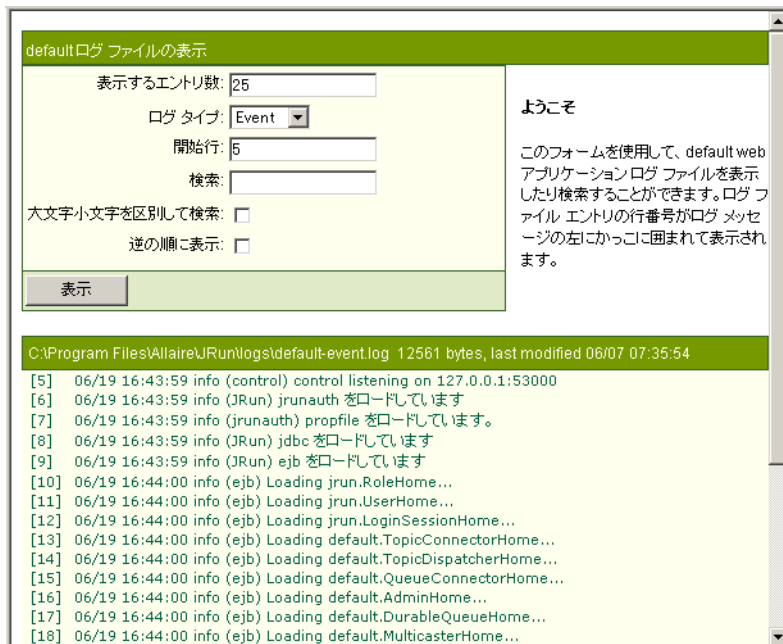
ようこそ

このフォームを使用して、default web アプリケーションログ ファイルを表示したり検索することができます。ログ ファイル エントリの行番号がログ メッセージの左にかっこに囲まれて表示されます。

- 2 次の表の説明に従ってフィールドを編集します。

フィールド	説明
表示するエントリ数	表示するログ ファイルの行数を指定します。ログ ファイルビューアには、この行数の最新のログ エントリが表示されます。最大行数は 99999 です。ログ ファイル全体を表示する場合は、このフィールドは空白にしてください。既定値は 25 です。
ログ タイプ	表示する JRun ログ ファイルのタイプを選択します。既定値は Event です。
開始行	(オプション) ログ ファイル表示を開始する行の番号を指定します。行番号は、ログ ファイルビューアの左側に [行番号] と表示されます。
検索	(オプション) 表示するログ ファイルのエントリにある文字列を入力します。ログ ファイルビューアには、この文字列が含まれているエントリのみが表示されます。この後、対応する行番号をクリックし、ログ ファイル内のその行にジャンプできます。
大文字小文字を区別して検索	(オプション) [検索] フィールドで大文字と小文字を区別するには、このチェック ボックスをオンにします。既定値はオフです。
逆の順に表示	(オプション) このログ ファイルでの最新のログ エントリを先頭に表示するには、このチェック ボックスをオンにします。既定値はオフです。

- 3 [表示] をクリックしてログ ファイルを表示します。  
JMC パネルにログ タイプのログ ファイルが表示されます。



- 4 ログ アクティビティを更新するには、[表示] ボタンをクリックします。
- 5 ログ ファイルビューアの設定値をリセットするには、JMC の左側パネルにある [ログ ビューア] を再度クリックします。

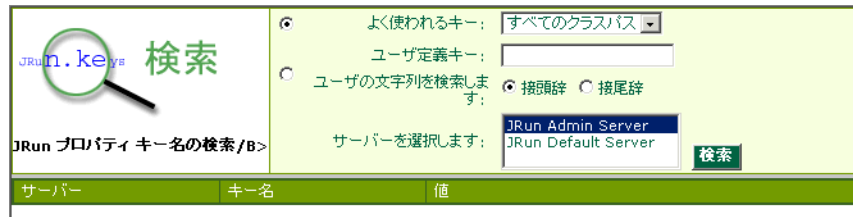
## JMC キーの検索

キー検索機能を使用して、接頭辞や接尾辞によって、1 つ以上の JRun サーバー内でキー (プロパティ名) を検索できます。

### キー検索を実行するには

- 1 JMC のアクセス バーの [キー検索] をクリックします。各 JMC パネルの [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。



サーバー	キー名	値
------	-----	---

- 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。

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、このキーが接頭辞または接尾辞のどちらであるかを指定します。

- 3 [サーバーを選択します] 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 サーバーをクリックします。

- 4 [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

変更内容を有効にするために JMC のログアウトが必要な場合があります。

### JMC からログアウトするには

- 1 JMC のアクセス 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

# コネクタについて

この章では、外部 Web サーバーに接続する場合の JRun 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も取り上げます。

### 目次

- JRun ポートについて..... 168
- Web サーバー コネクタについて..... 172
- 1 つの Web サーバーへの複数の JRun サーバーの接続..... 176
- 単純な分散環境での JRun の実行..... 179
- 複雑な分散環境での JRun の実行..... 182
- 分散環境での JSP の使用..... 185
- 分散 JRun システムの保護..... 186
- JRun でのマルチホスティング..... 188
- 要求のチェーン化..... 192
- カスタム コネクタの作成..... 194

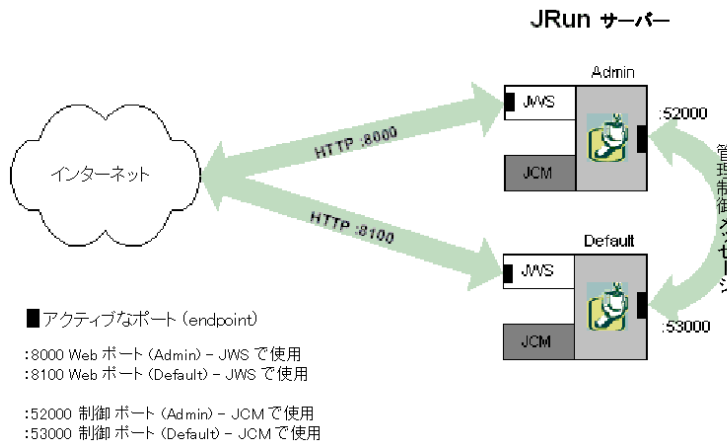
## JRun ポートについて

JRun サーバーで使用される既定のポートは4つあり、JRun を管理する際に認識する必要があります。新しい JRun サーバーを追加するときは、これらのポートの認識が特に重要です。[86 ページの「JRun サーバーの追加と削除」](#)を参照してください。

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。これらのポートは次のとおりです。

- 2つの JWS ポート (admin JRun サーバーおよび default JRun サーバー)。これらのポートから JWS への HTTP 要求を受信します。
- 2つの制御ポート (admin JRun サーバーおよび default JRun サーバー)。これらのポートによって、管理コンソールに対する管理メッセージの送受信を行います。たとえば、JRun サーバーへの開始コマンドや終了コマンドがあります。

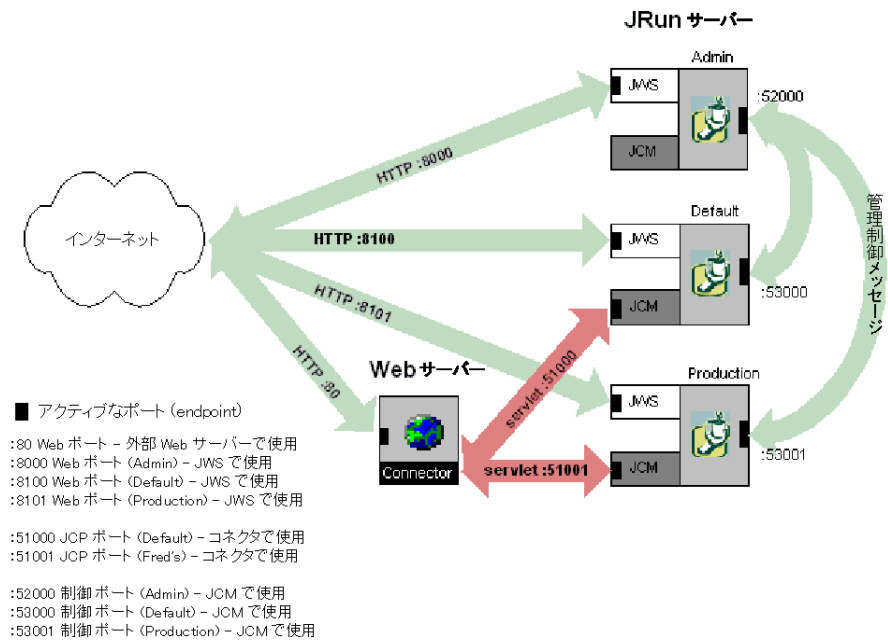
次の図は、関連する既定のポート間の対話を示します。この図は既定の JRun セットアップを想定しています。



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、認識する必要があるもう1つのポートは、JRun コネクタプロキシ (JCP) ポートです。JCP は、Web サーバーから渡される JRun への要求を処理するモジュールです。このモジュールは JRun 接続モジュール (JCM) と呼ばれます。JCP ポートによって、JRun 接続モジュール (JCM) と Web サーバーコネクタとの接続が容易になります。接続された JRun サーバーごとに1つの JCP ポートを使用します。

default JRun サーバーおよび admin JRun サーバーのほかに JRun サーバーを追加する場合は、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。コネクタ ウィザードを実行して新しい JRun サーバーを接続した場合、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ます。また、JRun サーバーを追加する際に各ポートに使用する推奨範囲も示します。

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	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 接続モジュール (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 コネクタと通信することができます。	コネクタ ウィザードを使用して JRun サーバーを外部 Web サーバーに接続しない場合、既定値は空白になります。 既定値は 51000 です。	51001 - 51999
control.endpoint.main.port	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	2300 - 2399
ejipt.homePort	EJB ホーム オブジェクト用のポート	admin: 2433 default: 2333	2400 - 2499

## 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、コンピュータに JRun サーバーを新たに追加する場合は、プログラミングによって空きポートをスキャンできます。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` のルートディレクトリ `/lib/install.jar` を含める必要があります。

`GetControlPort` は、コマンドラインまたは Java アプリケーションから呼び出すことができます。

## コマンド ラインからの GetControlPort の使用

次に、コマンド ラインから GetControlPort を呼び出す場合の使用方法を示します。

```
% java [-classpath classpath] allaire.jrun.install.GetControlPort min  
      max [output_file]
```

たとえば、51000 ~ 51999 の範囲で最初の空きポートを探すには、次のようにします。

```
% java allaire.jrun.install.GetControlPort 51000 51999
```

## Java アプリケーションからの GetControlPort の使用

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。エラーになった場合、GetControlPort から -1 が返されます。

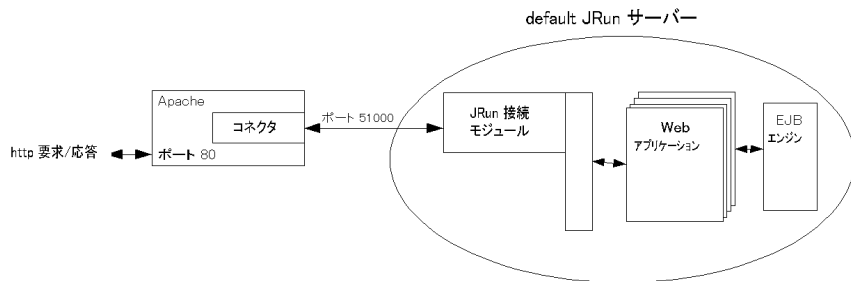
例:

```
String[] portrange = new String[2];  
String openport = new String();  
portrange[0] = "51000";  
portrange[1] = "51999";  
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

ネイティブの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 インターフェイスを定義する 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ー (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。上の図では、default JRun サーバーはポート番号 51000 で受信します。JRun サーバーと Web サーバー間の接続を構成する場合、ユーザは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できます。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 アドレスは「\*」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 実装のプロパティが、`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、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できるような設定を使用します。次の表は、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
<code>jrun.rootdir</code>	<code>jrun.rootdir</code> 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、 <code>/opt/JRun</code> や <code>c:\%Progra~1\Allaire\JRun</code> のように指定します。
<code>jvmlist</code>	<code>jvmlist</code> 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、 <code>default</code> 、 <code>admin</code> 、 <code>production</code> 、 <code>qa</code> のように指定します。
<code>verbose</code>	<code>verbose</code> 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 <code>false</code> です。
<code>proxyhost</code> <code>proxyport</code>	<p><code>proxyhost</code> および <code>proxyport</code> プロパティは、JRun 接続モジュール (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。</p> <p><code>proxyhost</code> 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。<code>proxyport</code> 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、<code>proxyhost</code> アドレス/<code>proxyport</code> の組み合わせを使用して、ソケットに要求を送信します。</p> <p>JRun をこのポートと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、<a href="#">175 ページの「local.properties ファイル内のコネクタのプロパティ」</a>を参照してください。</p> <p>コネクタ ウィザードを実行する場合、<code>proxyhost</code> および <code>proxyport</code> 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作業で編集する必要があります。</p>

プロパティ	説明
rulespath	<p>rulespath プロパティは、local.properties ファイルの場所を指定します。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。</p> <p>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 コンピュータへの絶対パスを指定する必要があります。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 システム上に作成する必要があります。</p> <p>同じファイル システムを共有していても、ローカル コンピュータに local.properties ファイルをコピーする必要がある場合もあります。</p>
scriptpath	<p>IIS のみ適用されます。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。</p>
errorurl	<p>オプションの 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。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』を参照してください。</p>

## 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 サーバーと 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ります。

**LoadModule** ステートメントが必要となるのは、JRun を DSO モジュールとして使用する場合だけです。

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 では `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` は次のようになります。

```
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 サーバーの一般的な `obj.conf` ファイル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

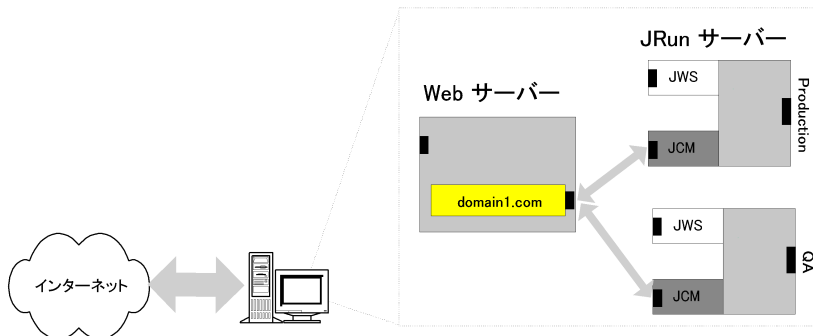
各 JRun サーバーの `local.properties` ファイルの JCP Services セクションには、JRun コネクタに影響を与えるプロパティが含まれます。次の表は、これらのプロパティを示します。

プロパティ	説明
<code>jcp.endpoint.main.interface</code>	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、 <code>jcp.endpoint.main.interface</code> 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。
<code>jcp.endpoint.main.bindaddress</code>	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。
<code>jcp.endpoint.main.port</code>	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。
<code>timeout</code> <code>min.threads</code> <code>active.threads</code> <code>max.threads</code>	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 の構成に直接影響を与えません。これらのフィールドの詳細については、 <a href="#">114 ページの「並行処理の概要」</a> 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、特別な手順を実行する必要があります。

たとえば、admin JRun サーバーおよび default JRun サーバーのほかに、テスト/品質管理用の QA JRun サーバーと実際の運用に使用される Production JRun サーバーが必要であるとします。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上に必要であると想定します。

### 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、[178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 `jmplist` 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。

たとえば、Apache コンフィギュレーションファイルは次のようになります。

```
JRunConfig jrun.rootdir "/opt/JRun/.."
JRunConfig jmplist default,production,qa
```

`jrun.rootdir` プロパティを使用して、`javms.properties` ファイルを検索します。`jmplist` プロパティは、`javms.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。

## javms.properties ファイル

`javms.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すると、JRun では `local.properties` ファイルを検索できるようになります。各サーバーのファイルが読み取られ、JCP の設定情報 (バインド アドレスとポート) が調べられます。

```
## 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 では 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` ファイルが検索されます。詳細については、[179 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ことが役立ちます。詳細については、[168 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

このセクションでは、複数の JRun サーバーを同一コンピュータ上の 1 つの Web サーバーに接続する方法を説明します。

### 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 86 ページの「JRun サーバーの追加」で説明しているように、新規の JRun サーバーをそれぞれ作成します。

必ず、`javms.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。各 JRun サーバーのコネクタ ウィザードでは、必ず固有の 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 ウィザード使用方法の詳細については、30 ページの「接続の概要」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。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
```

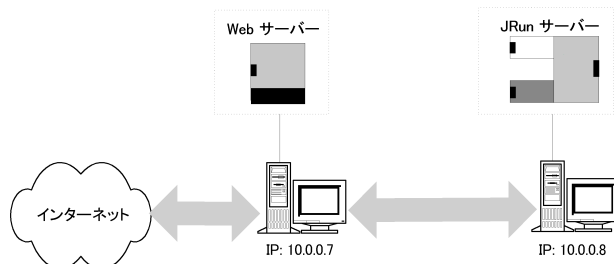
`jvmlist` プロパティにサーバーが複数存在すると、JRun では Web サーバーのコンフィギュレーション ファイル内の `rulespath`、`proxyhost`、および `proxyport` プロパティが無視されることに注意してください。その代わりに、176 ページの「設定の詳細」で説明しているように、JRun によって各サーバーの `local.properties` ファイルからこの情報が取得されます。

- 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。

次の図は、単純な分散環境でのインストールを示します。



## 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 を構成する場合は、次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。

- 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 `jrunit.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 `jrunit.rootdir` プロパティをコメント化します。次に例を示します。

```
#jrunit.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 ファイルの `rulespath` プロパティ内で、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。ファイルシステムを共有していても、`local.properties` ファイルをコピーする必要がある場合もあります。

---

### 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、Web サーバー コンピュータ 上のこのファイルを継続的に更新する必要はありません。

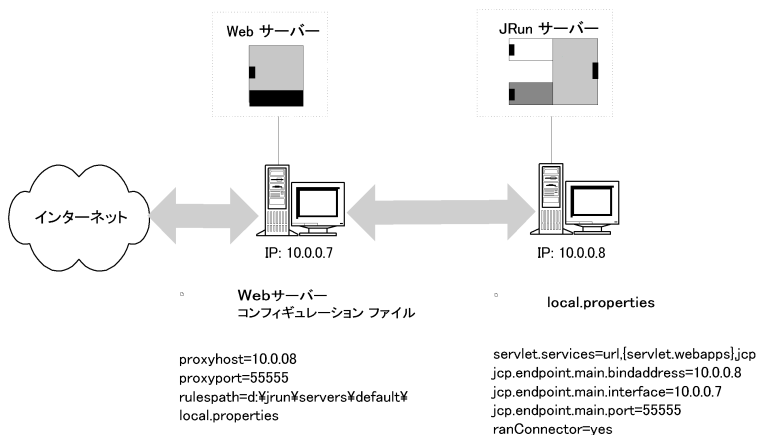
---

`local.properties` を Web サーバーにコピーする場合は、ファイル全体の階層を維持します。たとえば、ファイルが JRun コンピュータの `c:\JRun\servers\default` に保存されている場合は、空の `/JRun` ディレクトリと、`/servers` および `/default` サブディレクトリを作成して `local.properties` ファイルを保存します。その階層内のほかのすべてのファイルをコピー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、[185 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 サーバーを使用不能にします (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。

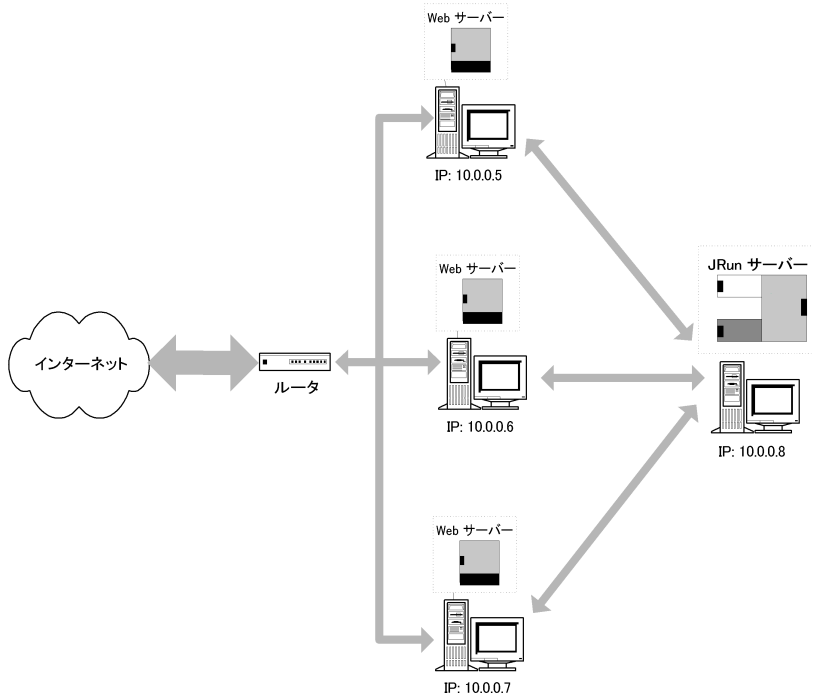
## 単純な分散環境の例

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ことを想定しています。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 台追加する場合があります。次の図は、この複雑な分散環境を示します。



## 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**

- 1 179 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 `jsp.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 アドレスのカンマ区切りリストに設定します。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。

上の図の場合は、次の設定になります。

```
jsp.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。

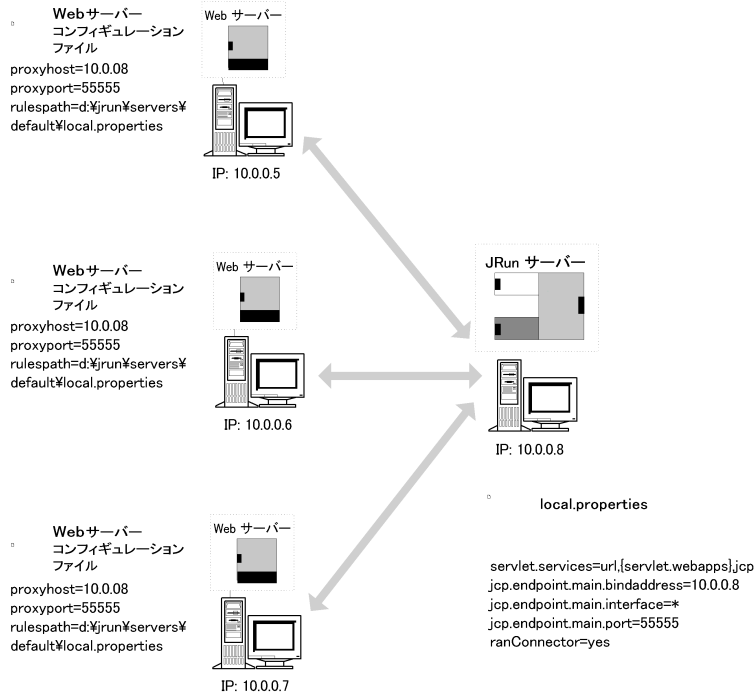
---

- 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、[185 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 サーバーを使用不能にします (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 マシンにマッピング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。

次の図は、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 サーバーの実際のパスは、実際にページを取得する実際のパスではありません。Web サーバーの実際のパスを、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次のとおりです。

```
### 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 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 をホスティング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

次の例では、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。次の例ではディレクトリ名が同じですが、異なっ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

分散環境でセキュリティを実装する場合には検討事項がいくつかありますが、このセクションでは、ユーザが実行できる次の JRun 固有のアクションについて説明します。

- **admin JWS をシャットダウンします。**既定の JRun インストールでは、admin JRun サーバーなどの JRun サーバーごとに Web サーバーがセットアップされます。詳細については、[186 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をセットアップします。**JRun には、JRun サーバーと外部 Web サーバー間の通信を他者から防御する基本的なメカニズムが用意されています。この設定については、[186 ページの「コネクタのホストベース認証」](#)を参照してください。

## JWS のオフ

JRun をインストールすると、default と admin の 2 つの JRun サーバーと、default と admin の 2 つの all-Java JRun Web Servers (JWS) が作成されます。既定では、これらの Web サーバーはそれぞれポート 8000 と 8100 で応答します。たい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が制限されますが、使用していないサービスはオフにすることをお勧めします。このセクションでは、JRun サーバーの JWS をオフにする方法について説明します。

### 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 のルートディレクトリ/servers/JRun サーバー名/local.properties にあるはずですが。
- 2 servlet.services プロパティから Web サービスを削除します。次に例を示します。

```
# 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を作成したら、認証されていないユーザがネットワーク上の別の場所から JRun サーバーにアクセスできないようにする必要があります。これを行うために、JRun には、JRun コネクタ用のホストベース認証が用意されています。これにより、アドレスの定義済みセットのホストだけが、JRun サーバーに要求を送信できるようになります。

JRun 管理コンソール (JMC) の [外部 Web サーバー] パネルを使用すると、特定の JRun サーバーと通信可能な IP アドレスを指定できます。現在、外部 Web サーバーと JRun サーバーの間のトラフィックを保護するための SSL やその他の暗号化テクノロジーを使用することはできません。

---

### メモ

既定の設定では、JRun サーバーはすべての IP アドレスからの要求を受け付けます。

---

## Web サーバーと JRun 間の接続をロックするには

- 1 JMC の左側ペインで、[マシン名] > [JRun サーバー名] > [外部 Web サーバー] を選択します。

### メモ

外部 Web サーバーに JRun サーバーを接続する際に、コネクタ ウィザードをまだ実行していない場合は、コネクタ ウィザードを実行するように要求されます。

[外部 Web サーバー] パネルが表示されます。

JRun Default Server > 外部 Web サーバー

外部 Web サーバーへ接続するために必要な設定を行います。JRun は、Java Servlet および JavaServer Pages サポートにより、さまざまなサードパーティ Web サーバーを拡張するように構成されています。以下のサードパーティ Web サーバーがサポートされています。

- Microsoft Personal Web Server / Internet Information Server
- Netscape Fast Track / Enterprise / iPlanet Servers
- Apache Server
- O'Reilly WebSite Pro
- Zeus Web Server

	名前	値	要約
	外部 Web サーバー アドレス	*	このサーバーに接続できる外部 Web サーバーのアドレス
	受信アドレス	127.0.0.1	外部 Web サーバーからの接続を受信するためのソケット アドレス
	受信ポート	51067	外部 Web サーバーからの接続を受信するためのソケット ポート
	待機 スレッドのタイムアウト	300	待機状態のスレッドを破棄するまでの秒数
	スレッドの最小数	1	プール中のハンドラ スレッドの最小数
	アクティブ要求の最大数	100	新しい要求の待ち行列化を始めるまでの同時要求数
	同時要求の最大数	1000	新しい要求の拒否を始めるまでの同時要求数
	接続モジュール	on	このコネクション モジュールの現在の状態

編集      コネクタ ウィザード

「ようこそ」ページに追加

- 2 右側ペインで、[外部 Web サーバー アドレス] フィールドをクリックします。外部 Web サーバーの編集ウィンドウが表示されます。
- 3 IP アドレスを入力します (複数の場合はカンマで区切ります)。これらのマシン上にある Web サーバーのみが JRun サーバーに要求を送信できます。\* を入力すると、すべての Web サーバーが JRun に要求を送信できるようになります。
- 4 変更を適用するには、[更新] ボタンをクリックします。
- 5 JRun サーバーを再起動します。

### メモ

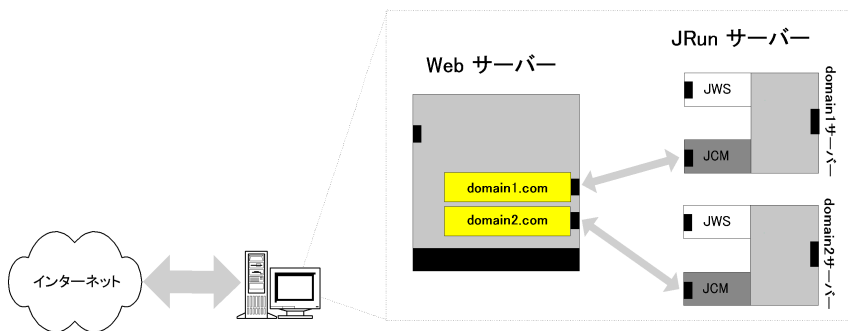
ホストベースの認証による保護では、IP のなりすましや、ほかの 中間一致攻撃 (*man-in-the-middle*) を防止できません。

## 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に、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、次の理由により、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 台のコンピュータで 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ます。ここで、Web サーバーは 2 つの仮想ホストを持っており、これらの各仮想ホストは独自の JRun サーバーを持っています。



このセクションでは、一般的な 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ます。LoadModule ステートメントは、グローバルレベルで 1 回だけ参照できるので、`VirtualHost` ディレクティブの外部に記述する必要があります。

```
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 newhost
    ErrorLog logs/newhost-error_log
    CustomLog logs/newhost-access_log common

    # JRun の設定
    <IfModule mod_jrun.c>
        JRunConfig jrun.rootdir "/opt/JRun/bin/.."
        JRunConfig jvmlist newhost
        JRunConfig Verbose false
        JRunConfig ProxyHost 127.0.0.1
        JRunConfig ProxyPort 51001
        JRunConfig Mappings "/opt/JRun/servers/newhost/local.properties"
    </IfModule>
</VirtualHost>
```

LoadModule ステートメントが必要となるのは、JRun を DSO モジュールとして使用する場合だけです。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` ファイルの編集のほかに、各仮想ホストの JRun サーバーを、固有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 [86 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 `newhost` 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、固有の 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、[168 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、`newhost 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 newhost JRun サーバーを起動します。次に例を示します。  

```
% jrun -start newhost
```

## 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 Microsoft 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 [86 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 `newhost` 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、固有の 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、[168 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、`newhost 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 Microsoft 管理コンソール (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 ウィザードに固有の JRun サーバー コネクタ ポートを入力してください。



手順 3 で次の操作を行います。

- 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 フィルタがすで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` コマンド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』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 [86 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 `newhost` 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、`newhost 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 ウィザードに固有の 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。

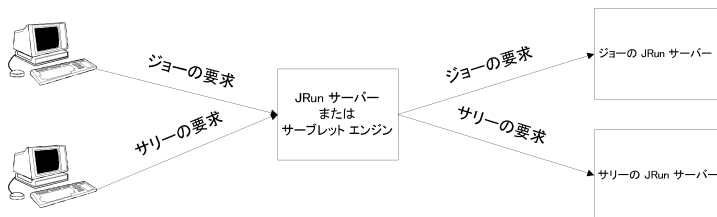
```
% jrun -restart admin
```

- 7 `newhost` JRun サーバーを起動します。次に例を示します。

```
% jrun -start newhost
```

## 要求のチェーン化

JRunConnector サブレットを使用すると、サブレット エンジン インスタンスから別のサブレット エンジン インスタンスに渡される要求をチェーン化できます。サブレット 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 サブレット エンジンから JRun サーバーに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。次の表では、それらの設定について説明します。

プロパティ	説明
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	JRunConnector によってすべての URI 情報を渡す (false) か、URI のパス情報のみを渡す (true) かを指定するブール値です (オプション)。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、次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 サーバーでは、次に示す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` の場合は、ターゲット サーバー上で次のいずれかの作業を行います。
  - 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要求 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` の場合は、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すための情報がパス情報にすべて含まれていることを確認する必要があります。

## 補足情報

JRunConnector はサーブレットなので、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 サーバーに要求を転送する場合に利用することができます。

## 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、ユーザの Web サーバーまたはほかの特殊なプラットフォームで使用するための、接続ソース 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/connector/src/readme を参照してください。JRun のルート ディレクトリ/connector/src ディレクトリには、サーバー独立モジュールのサブディレクトリと、Apache、ISAPI、および NSAPI のサブディレクトリが含まれています。JRun には、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 ファイル、プロジェクト ファイル、および関連ファイルが用意されています。

次の表は、JRun のルート ディレクトリ/connector/src のサブディレクトリの概要を示します。

ディレクトリ	内容
apache	Apache 用のプロジェクト ファイル、関連ファイル、ヘッダ 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	次のファイルが含まれています。 <ul style="list-style-type: none"> <li>• 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。</li> <li>• jrun_property.h : プロパティのデータ構造の定義です。</li> <li>• jrun_proxy.c : JRun プロキシ プロトコルの実装です。</li> <li>• jrun_proxy.h : コネクタ プロトコルの定義です。</li> <li>• platform.c : 各種システム呼び出しの既定の実装です。</li> </ul>
isapi	ISAPI 用のプロジェクト ファイル、関連ファイル、ヘッダ 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。 <ul style="list-style-type: none"> <li>• extension.cpp : ISAPI 拡張機能のエントリ ポイントです。</li> <li>• filter.cpp : ISAPI フィルタのエントリ ポイントです。</li> <li>• interface.cpp : jrun_proxy.c によって呼び出されるメソッドです。</li> </ul>
nsapi	NSAPI 用のプロジェクト ファイル、関連ファイル、ヘッダ 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。 <ul style="list-style-type: none"> <li>• extension.c : NSAPI のエントリ ポイントです。</li> <li>• interface.c : jrun_proxy.c によって呼び出されるメソッドです。</li> <li>• nwmain.c : Netware 固有のコードです。</li> </ul>

次のセクションでは、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用のカスタム Apache コネクタを作成できません。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もできます。

### 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。たとえば、次のようになります。  
`% ./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 次の方法で `make` ファイルを編集します。
  - 適切な `INCxy = ../Servers/Netscape/x.y/include` 行を変更し、正しい `include` ディレクトリ (たとえば、`netscapeinstalldirectory/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り理解できるだけでなく、JRun の設定の一部を細かく制御することもできます。

この章では、プロパティファイルが持つ階層の性質と、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 ..... 198
- プロパティファイルの再ロード ..... 199
- プロパティファイルの階層について ..... 200
- プロパティファイルの編集 ..... 202
- PropertyScript の使用法 ..... 204

## 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 `local.properties` ファイルを読み込みます。

次の表では、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について説明しています。

ファイル名	格納場所	説明
<code>descriptions.properties</code>	¥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。
<code>global.properties</code>	¥lib¥	最上位レベルの値が格納されます。このファイルの設定は、コンテナにあるすべての JRun サーバーに影響を与えます。既定では、 <code>local.properties</code> ファイルには <code>global.properties</code> の値が格納されます。
<code>local.properties</code>	¥servers¥ サーバー名¥	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。 <code>global.properties</code> は、このプロパティによって無効になります。
<code>webapp.properties</code>	¥servers¥ サーバー名¥ アプリケーション名 ¥WEB-INF¥	必要に応じて、Web アプリケーションのプロパティが格納されます。 <code>local.properties</code> および <code>global.properties</code> 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 を使用し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。
<code>jvms.properties</code>	¥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です。
<code>pass.properties</code>	¥lib¥	JRun JMC ユーザのための暗号化パスワードと許可の設定が格納されます。
<code>serial_number.properties</code>	¥lib¥	JRun のシリアル番号が格納されます。
<code>users.properties</code>	¥lib¥	Web アプリケーション 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。



ファイル名	格納場所	説明
<code>ejipt.properties</code>	<code>¥lib¥</code>	EJB エンジンのホスト情報が格納されます。
<code>deploy.properties</code>	<code>¥servers¥</code> サーバー名 <code>¥deploy</code>	サーバー レベルのプロパティが格納されます。この情報は、Deploy ツールによって、公開する 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。
<code>bean_name.properties</code>	Bean のインターフェイスおよび実装ディレクトリ	セキュリティおよび Bean の説明情報が格納されます。 Bean の XML 記述子ファイルにも同じ情報を格納できます。
<code>default.properties</code>	Bean のインターフェイスおよび実装ディレクトリ	この Bean の内容に関する情報が格納されます。

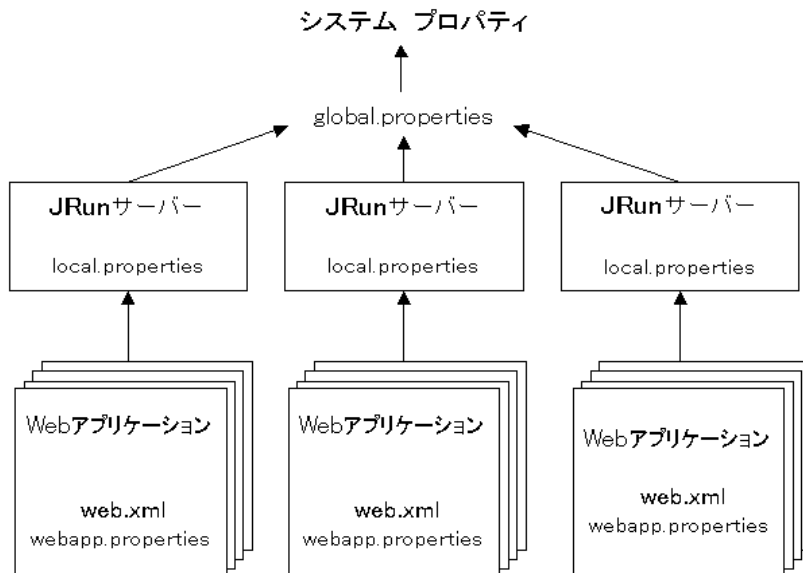
## プロパティ ファイルの再ロード

JRun プロパティ ファイルを変更した場合は、JRun サーバーを再起動する必要があります。ただし、`users.properties` ファイルは唯一の例外です。このファイルは動的に再ロードされるので、Web アプリケーションのユーザをリアルタイムで追加、変更、または削除できます。

## 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の4階層から構成されています。グローバルレベルの設定値は、ローカルレベルの設定値によって上書きされる場合があります。さらに、ローカルレベルの設定値は、アプリケーションレベル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つのレベルを示します。



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の編集やJMCの使用によって上書きすることはできません。次のような設定値があります。

```
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`、`javms.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 の各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が無効になります。JRun でこれらのファイルが作成されるのは、ユーザが JMC を使用し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 の仕様によって定義されます。このファイルには、アプリケーションレベルとコンテナレベルの両方の設定があります。詳細については、該当する仕様を参照してください。

たとえば、次のように割り当てられているとします。

```
/default/local.properties  webapp=default_invoker
/admin/local.properties    webapp={default}
global.properties          webapp=invoker
```

結果：

`webapp` プロパティは、`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

JMC を使用すると、ほとんどのプロパティを変更できます。ただし、一部の JRun プロパティは、エディタ ツールを使用して設定しなければなりません。JRun プロパティ ファイルは、テキスト エディタを使用して手作業で編集することもできますが、PropertyScript ユーティリティを使用して、Java インターフェイスから編集することもできます (204 ページの「PropertyScript の使用法」を参照)。JRun のプロパティ ファイルを編集する場合は、いくつかの簡単なルールに従う必要があります。次のセクションでは、このルールについて説明します。

### 構文

JRun プロパティ ファイルを変更する場合は、次の構文ルールを使用します。

- パラメータは `parameter=value` のように設定します。この構文では、等号 (=) の前後にスペースは入れません。
- 値はカンマまたはセミコロンで区切ります。
- 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空白を入れないでください。
- 変数は中かっこ {} で囲みます。
- 行をコメント化するときには、番号記号 (#) を使用します。

### 編集

`local.properties` ファイルを編集する場合は、その前に関連する JRun サーバーを停止し、編集が終了したら再起動します。これは、`local.properties` ファイルを手作業で変更した後、JRun サーバーによってメモリにある内容が `local.properties` ファイルに書き込まれるのを防ぐために行います。

JRun プロパティ ファイルを編集する場合は、次の点にも注意してください。

- プロパティ ファイルを編集する前に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をよく使用します。変数とプレースホルダは、プロパティ ファイルや JRun 管理コンソール (JMC) 内で中かっこ {} によって示されます。

{variable} は、実行時に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できます。

```
jrun.services=scheduler,logging,monitor,{servlet.services},control
```

また次のように、変数と定数を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、実際は変数ではなくプレースホルダとして機能します。次の表では、この2つの変数と、その他の変数について説明しています。

変数	説明
<code>{jrun.rootdir}</code>	<p>インストール時のシステム変数セット。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。</p> <p>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、<code>-D</code> スイッチを指定して、関連する値を持つ <code>jrun.rootdir</code> を受け渡します。次に例を示します。</p> <pre>java -D jrun.rootdir=c:%Allaire%JRun%</pre>
<code>{default-app.rootdir}</code>	<p>JRun のすべての Web アプリケーションには、ルート ディレクトリがあります。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できます。</p> <p>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。</p>
<code>{default}</code>	<p><code>{default}</code> 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 <code>global.properties</code> 内で実数値を設定し、<code>local.properties</code> 内では、そのパラメータを <code>{default}</code> に設定する方法です。</p> <p>空白を代入すると、変数の値が <code>null</code> に設定されます。<code>local.properties</code> 内に <code>foo={default}</code> という代入式があり、<code>global.properties</code> 内に <code>foo=</code> という代入式がある場合、<code>foo</code> には <code>null</code> が代入されます。</p> <p>通常は、<code>local.properties</code> と <code>global.properties</code> の両方で、同じパラメータが <code>{default}</code> または空白に設定されていることはありません。</p>
その他の変数	<p>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できます。これらの変数には、<code>{date}</code>、<code>{hour}</code>、<code>{day}</code>、<code>{month}</code>、<code>{year}</code> 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。</p> <p>オブジェクトと変数を混同しないように注意してください。たとえば、<code>logging.dispatchLogger.events</code> 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、カンマで区切って指定したものです。これらのイベントは、システム変数のように見えますが、実際はオブジェクトなので、通常は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。</p> <pre>{logging.infoevent},{logging.debugevent}, {logging.warningevent},{logging.errorrevent}</pre>

## PropertyScript の使用法

PropertyScript ユーティリティを使用すると、JRun プロパティファイルにある設定の変更、追加、および削除を行うことができます。PropertyScript は、JRun プロパティファイルを変更するディレクティブが含まれている、個別のスクリプトファイル进行处理します。

PropertyScript クラスは、install.jar ファイルにある `allaire.jrun.install` パッケージの一部です。クラスパスには、`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、ディレクティブが含まれているスクリプトファイルへのファイルのシステムパスです。スクリプトファイルの作成については、[204 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` オプションでは、`global.properties`、`local.properties`、`javms.properties` などの JRun プロパティファイルを指示します。このオプションは現在のリリースの JRun で推奨されていないため、必要ありません。`script-file` には JRun プロパティファイルを指定します。

例:

```
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` オプションでは、JRun プロパティファイルの変更時に PropertyScript によって使用されるすべてのディレクティブが含まれるテキストファイルを指示します。スクリプトファイルの作成時には、少なくとも1つの `file` コマンドと `filename` (JRun プロパティファイルを参照する) を含める必要があります。各 `file` コマンドの下には、前に示してある `filename` に対して実行するディレクティブおよびオプションを列挙します。

スクリプトファイルの構文は、次のようになります。

```
file filename
directive1
[directive2]
...
[file filename]
[directive1]
[directive2]
...
```

スクリプト ファイル内のプロパティ ファイルのセクションごとに、新しい `file` コマンドを指定する必要があります。 `filename` (ファイル名) には、 `*.property` ファイルへの絶対パスを指定する必要があります。

通常、スクリプト ファイル内の各ディレクティブは、1つのコマンドと、1組のキーと値のペアから成り立っています。キーは、JRun プロパティ ファイル内のプロパティに対応します。

次の例では、`add` がコマンド、`control.endpoint.main.port` がキー、`53000` が値です。

```
add control.endpoint.main.port=53000
```

複数の値を持つキーもあります。この場合、値はスペースまたはカンマで区切られることがあります。

次の表では、スクリプト ファイルのディレクティブとそのオプションについて説明しています。

ディレクティブ	説明
<code>add</code> キー 値	新しいキーと値を、プロパティ ファイルの最後に追加します。既存のキーを追加すると、該当するキーと値が PropertyScript によって上書きされます。
<code>replace</code> キー 値	キーの値を新しい値に置き換えます。存在しないキーの値を置き換えようとする、このディレクティブは PropertyScript によって <code>add</code> として処理されます。
<code>delete</code> キー	キーとその値を削除します。
<code>clear</code> キー	キーのすべての値を削除しますが、キーは削除しません。
<code>append</code> キー 値	キーの最後の値の後に、値を追加します。値の先頭には、区切り文字 (通常はカンマ) を指定する必要があります。 たとえば、 <code>jcp</code> を <code>servlet.services</code> キーに追加する場合は、次のように指定します。 <code>append servlet.services ,jcp</code> 一部のキーはカンマでなくスペースによって区切られます。この場合は、 <code>append_space</code> ディレクティブを使用します。
<code>append_space</code> キー 値	キーの最後の値の後にスペースを入れてから、値を追加します。スペースを区切り文字として使用する <code>java.args</code> キーを変更する場合に役立ちます。
<code>token_remove</code> キー 値	キーの中から指定した値を検索し、その値および後に続く区切り文字 (スペースまたはカンマ) を削除します。
<code>ejb_append</code> <code>jrun.services</code> 値	<code>global.properties</code> ファイル内の <code>jrun.services</code> キーを変更する場合にのみ使用します。 <code>servlet_services</code> はほかのすべてのサービスが開始してから開始する必要があるため、このメソッドを使用して <code>{servlet_services}</code> キーを一覧の最後に表示します。

ディレクティブ	説明
unplug [servlet ejb]	<p>サーブレットまたは EJB のいずれかの機能を JRun から削除します。ユーザがコンポーネントを追加および削除できるようにする場合に使用します。また、インストールファイルを編集して、不要なコンポーネントがインストールされないようにすることもできます。</p> <p>unplug によってコンポーネントが無効になりますが、実際にはアンインストールされません。</p>
comment キー テキスト文字列	<p>テキスト文字列から構成されるコメント行を、指定したキーの上の行に追加します。</p>
adduser ユーザ名 パスワード	<p>新規 JMC ユーザを追加します。PropertyScript では、UnixCrypt を使用してパスワードが暗号化されます。このディレクティブは、pass.properties ファイルを変更する場合にのみ使用します。</p> <p>このディレクティブを使用して、新規の Web アプリケーション ユーザを users.properties ファイルに追加することはできません。Web アプリケーション ユーザを追加する方法については、『JRun によるアプリケーションの開発』にある「PropertyFileAuthentication クラス」の説明を参照してください。</p>
port キー 最小値, 最大値	<p>インストール時に JRun で設定される、admin サーバーのアクセス可能ポート 範囲を設定します。最小値は最小のポート 番号、最大値は最大のポート 番号です。</p> <p>PropertyScript では、この範囲内にあるすべてのポートが試行されます。</p> <p>ポート ディレクティブは、JRun サーバーの local.properties ファイルごとに指定します。</p> <p>admin サーバーの既定範囲は 8000 ~ 8099 です。default サーバーの既定範囲は 8100 ~ 8199 です。</p>



## サンプル スクリプト ファイル

次に、スクリプト ファイルの一例を示します。この例では、**default JRun** サーバーの **local.properties** ファイルに対して、新しい Web アプリケーション、「**MyStocks**」を追加し、**demo-app** を削除します。最後の行は、**default JRun** サーバーの **JRun Web** サーバーのポートを、**8080 ~ 8089** の範囲で使用可能なポートに変更します。

```
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 # mystocks app が追加され、demo が削除されました。
  delete demo-app.rootdir
  delete demo-app.class
  delete webapp.mapping./demo
  port web.endpoint.main.port 8080,8089
```



# 索引

## 記号

/admin 19  
/bin 18  
/connectors 18  
/default 19  
/docs 18  
/ext 18  
/lib 18  
/logs 18  
/samples 18  
/servers 18  
/servlets 18  
/servlets ディレクトリ 135  
/uninst 18  
{default} 203  
{jrun.rootdir} 203  
2.3 からのアップグレード xv

## A

Active Server Pages xvii  
admin JRun サーバー  
開始 85  
既定の Web アプリケーション 122  
起動 83  
説明 80  
admin オプション 83  
admin、パスワードの変更 79  
Allaire xxv  
お問い合わせ先 xxv  
テクニカル サポート xxv  
Allaire Spectra  
文書、概要 xxii  
Apache  
DSO モジュール 31  
カスタム コネクタの  
コンパイル 195  
コンフィギュレーション  
ファイルのサンプル 174

接続 31  
変更 35  
モジュール 31

## B

Bean  
「Enterprise JavaBeans」を参照  
Bean コンテキスト 160  
[Bean プロパティ] パネル 160  
Bean プロパティファイル 199

## C

CF\_Anywhere xvii  
CGI インターフェイス 62  
console オプション 84

## D

default JRun サーバー  
開始 85  
既定の Web アプリケーション  
122  
説明 80  
[default ログ ファイルの表示]  
パネル 163  
default.properties  
説明 199  
default-app 19  
サブレット ディレクトリの  
追加 135  
マッピング 136  
demo オプション 84  
demo-app 20, 23  
deploy.properties  
EJB の公開 156  
説明 199  
descriptions.properties  
説明 198  
Dynamic Shared Objects  
(DSO) 31

## E

EAR ファイル  
公開 162  
説明 161  
EJB  
「Enterprise JavaBeans」を参照  
EJB の再公開 158  
ejipt.properties  
説明 199  
Enterprise Application アーカイブ  
「EAR ファイル」を参照  
159  
Enterprise JavaBeans  
Bean コンテキストの説明 160  
公開 156  
再公開 158  
削除 159  
サポート xx  
設定 160  
[Enterprise JavaBeans コンテナを  
削除します。] パネル 159  
[Enterprise JavaBeans の公開]  
パネル 157  
[Enterprise JavaBeans] パネル 156  
errorurl 174

## G

GetControlPort 171  
getInitParameter 134  
global.jsa ファイル 140  
global.properties 198

## H

HTML ページ、既定の使用順 138  
HTTP 要求 114  
httpd.conf 174  
トラブルシューティング 66  
変更 35

- I**
- IIS**
- 「Internet Information Server」を参照
  - init() メソッド 134
  - install オプション 84
  - Internet Information Server 3.0
    - グローバル フィルタ 39
    - 接続 36
    - 変更 39
    - レジストリ変更 39
  - Internet Information Server 4.0/5.0
    - ISAPI フィルタ 45
    - グローバル フィルタ 44
    - コンフィギュレーション ファイルのサンプル 175
    - 接続 39, 40
    - 設定 40
    - フィルタの順位付け 46
    - 変更 44
    - マッピング 44
    - メタベースの変更 44
  - invoice アプリケーション 20
  - invoice-app 20
  - invoker サーブレット 149
  - iPlanet
    - 「NES/iPlanet」を参照
  - ISAPI
    - カスタム コネクタ 194
  - ISAPI フィルタ
    - 順位付け 46
    - トラブルシューティング 66
    - 編集 45
- J**
- [J2EE アプリケーションの公開] パネル 162
  - J2EE アプリケーション、公開 161
  - JAR ファイル、公開 156
  - Java
    - Java platform xviii
    - Java Runtime Environment xix
    - Software Development Kit xviii
    - 概要 xviii
  - Java Virtual Machine
    - JVM 1.1.8 に関する検討事項 xiii
    - アップグレード 92
    - サポートされている xii
    - 設定 91
    - 説明 xix
  - Java アーカイブ
    - 「JAR ファイル」を参照
  - Java インタプリタ
    - NES/iPlanet の有効化 50
  - java オプション 84
  - [Java の設定] パネル 91
  - Java ベースの Web サーバー
    - 接続 61
  - JavaServer Pages (JSP)
    - コンパイラ 139
    - サポート xx
    - 分散環境 185
  - [JavaServer ページの設定] パネル 139
  - JCM
    - 「JRun 接続モジュール」を参照
  - JCP services 175
  - JCP ポート 168
  - JDBC データ ソース
    - 一般的なエラー 113
    - 接続プール 112
    - 説明 109
    - ドライバの設定 111
    - 引数の受け渡し 112
  - [JDBC データ ソース] パネル 110, 111
  - JDBC データ ソースのプール 112
  - JMC
    - 「JRun 管理コンソール」を参照
  - JMC キーの検索 166
  - JMC のお気に入りの設定 73
  - JMC の使用 72
  - JMC のパスワードの変更 79
  - JMC ユーザの管理 75
  - JMC ユーザの削除 78
  - JMC ユーザの設定の変更 77
  - JMC ユーザの追加 75
  - jmc-app 19
  - JRE
    - 「Java Runtime Environment」を参照
  - /servlets ディレクトリ 135
  - JRun 2.3
    - 3.x との実行 xv
    - アップグレード xv
    - サーブレット xvi
    - 縮小された機能 xvii
    - 相違点 xv, 135
  - JRun Advanced 版
    - 説明 x
  - JRun Connection Module
    - IIS 3.0 の使用 37
  - JRun Developer 版
    - 説明 x
  - JRun Enterprise 版
    - 説明 x
  - JRun Professional 版
    - 説明 x
  - JRun Studio 版 x
  - JRun Web サーバー
    - JCM の設定 117
    - エンドポイント プロパティ 117
    - 開始/停止 119
    - 設定 117
    - 説明 117
    - ポート 168
  - [JRun Web サーバー] パネル 117
  - JRun 管理コンソール
    - お気に入りの設定 73
    - 開始 70
    - コマンド ラインから開始 83
    - 使用 72
    - ショートカットの追加 73
    - 説明 72
    - 必要条件 15
    - 開く 15
    - ユーザの管理 75
    - ユーザの削除 78
    - ユーザの追加 75
    - ユーザの編集 77
    - ログイン 71
  - JRun 管理コンソール アプリケーション 19
  - JRun コネクタ ウィザード
    - 「コネクタ ウィザード」を参照
  - JRun コネクタ フィルタ
    - 削除 45
    - 順位付け 46
    - 編集 45
  - jrun コマンド
    - admin オプション 83
    - console オプション 84
    - demo オプション 84
    - install オプション 84
    - java オプション 84
    - jrundir オプション 85
    - remove オプション 85
    - restart オプション 85
    - start オプション 85
    - status オプション 86
    - stop オプション 86
    - 使用 83
  - jrun コマンドの使用 83
  - JRun サーバー
    - JMC で再起動 82
    - NT サービスとしてインストール 84
    - status 86
    - イベント ログ記録 108
    - 管理 81
    - 既定のアプリケーション 122
    - 起動と停止 21

- コマンドラインから開始 85
  - コマンドラインから停止 86
  - 再起動 85
  - 削除 85,90
  - 使用 21
  - 新規プロセスの作成 68,85
  - ステータス 81
  - 設定 80
  - 説明 80
  - 追加 86
  - プロセスのバックグラウンドへの移動 68
  - プロパティ 201
  - 分散 179
  - 分散環境 176
  - マルチホーム機能 133
  - [JRun サーバー] パネル 82
    - ear ファイルの公開 162
    - 説明 81
  - [JRun サーバーの管理] パネル 87, 81
  - JRun サーバーの起動 21
  - JRun サーバーの再起動 22
    - JMCで 82
  - JRun サーバーの削除 90
  - JRun サーバーの作成 86
  - JRun サーバーの消去 90
  - JRun サーバーの設定 80
  - JRun サーバーの追加 86
  - JRun サーバーの停止 22
  - JRun 製品のラインナップ x
  - JRun 接続モジュール 172
    - Apache の使用 33
    - IIS 4.0/5.0 の使用 42
    - NES/iPlanet の使用 48
    - PWS の使用 37
    - WebSite Pro の使用 59
    - Zeus の使用 64
    - 開始/停止 121
    - 外部 Web サーバーの設定 119
    - 構成の概要 30
  - JRun 接続モジュール (JCM)
    - JWS の設定 117
  - JRun のエディション x
  - JRun のコンポーネント 6
  - jrundll 175
    - IIS 3.0/PWS 39
    - IIS 4.0/5.0 44
  - jrundll.ini 175
    - IIS 3.0/PWS 39
    - IIS 4.0/5.0 44
  - JRunConnector サブレット 192
  - jrundir オプション 85
  - JSP
    - 「JavaServer Pages」を参照
  - JSP エンジン
    - 外部 Java コンパイラの使用 140
    - 設定 139
  - JSP のコンパイル 140
  - JVM
    - 「Java Virtual Machines」を参照
    - JVM のアップグレード 92
    - JVM の設定 91
  - javlist 173
  - javms.properties
    - JRun サーバーを NT サービスとしてインストール 84
    - 説明 198
  - JWS
    - 「JRun Web サーバー」を参照
    - JWS の開始 119
    - JWS の停止 119
- ## L
- local.properties
    - コネクタのプロパティ 175
    - 説明 198
    - 変更 66
- ## M
- MIME タイプ
    - WML 154
    - 関連付けの編集 142
    - サブレットのチェーン化 153
    - マッピング 142
  - [MIME タイプのマッピング] パネル 142
  - Multipurpose Internet Mail Extension
    - 「MIME タイプ」を参照
- ## N
- NameTrans ディレクティブ
    - Java コネクタ変更 52
  - NES/iPlanet
    - Java インタプリタの有効化 50
    - Java コネクタ 52
    - オブジェクト定義 51
    - カスタム コネクタのコンパイル 196
    - コンフィギュレーション ファイルのサンプル 175
    - サンプル obj.conf 53
    - 接続 47
    - ネイティブ コネクタ 51
    - ネイティブ コネクタまたは Java コネクタの選択 48
    - 変更 51
- ## O
- obj.conf 175
    - Java コネクタ 52
    - サンプル ファイル 53
    - トラブルシューティング 66
    - ネイティブ コネクタ 51
    - 変更 51
- ## P
- pass.properties 198
  - Personal Web Server
    - 接続 36
    - 変更 39
  - PropertyScript 204
    - サンプル スクリプト 207
    - 使用法 204
    - スクリプト ファイルの作成 204
  - proxyhost 173
  - proxyport 173
  - PWS
    - 「Personal Web Server」を参照
- ## Q
- quiet オプション
    - install オプションあり 84
    - remove オプションあり 85
- ## R
- remove オプション 85
  - restart オプション 85
  - rulespath 174
  - runtime.properties 157
- ## S
- scriptpath 174
- ## Netscape Enterprise Server
- 「NES/iPlanet」を参照
  - nohup オプション 68,85
  - NSAPI
    - カスタム コネクタ 194
  - NSAPI フィルタ 48
    - トラブルシューティング 66
    - ネイティブ コネクタ 51
  - NT サービス
    - JRun サーバーのインストール 84
    - アプリケーションとの相違 21, 80
    - 開始 85
    - 再起動 85
    - 削除 85
    - ステータス 86
    - 停止 86

## SDK

「Software Developer Kit」を参照  
serial\_number.properties 198

## SnoopServlet

「デモアプリケーション」を参照

## Software Development Kit xviii

コンポーネント xviii

バージョン xviii

## SSI

「サーバー側インクルード」を参  
照

## SSL 94

CSRの作成 98

概要 95

サーバー証明書の認証 104

証明書の削除 106

証明書のサンプル 95

制限 95

保護されていないポートの無効  
化 107

[SSLウィザード] 98

[SSL証明書の認証]パネル 105

start オプション 85

status オプション 86

stop オプション 86

## U

## URL

接頭辞 149

パスのマッピング 131

URL 接頭辞 129

users.properties 198

## W

## WAR ファイル

「Webアプリケーション」を参照

公開 125

説明 125

## Web アプリケーション

EARファイルの公開 162

EJBの公開 156

JSPコンパイラ 139

アプリケーションURLの変  
更 129

アプリケーションパスのマッピ  
ング 131, 135

アプリケーションホストの変  
更 129

アプリケーションの説明の変  
更 129

アプリケーションのルートディ  
レクトリの変更 129

アプリケーション名の変更 129

イベントログ 141

エンタープライズアプリケー  
ション 156

公開 125

公開の最低要件 125

削除 130

作成 124

説明 122

追加 124

パラメータの追加 134

標準インストール 122

標準マッピング 122, 135

ファイル設定 138

プロパティ 201

編集 128

[Webアプリケーションセッション  
]パネル 143

Webアプリケーションの公開 125

[Webアプリケーションの公開]  
パネル 126

Webアプリケーションの削除 130

[Webアプリケーションの削除]  
パネル 130

Webアプリケーションの作成 124

[Webアプリケーションの作成]  
パネル 124

Webアプリケーションの消去 130

Webアプリケーションの設定の変  
更 128

Webアプリケーションの追加 124

Webアプリケーションの登録取  
消 130

Webアプリケーションの編集 128

[Webアプリケーションの編集]  
パネル 128

## Web サーバー

「外部 Web サーバー」を参照

ネットワークポート 172

バインドアドレス 172

複数 176

要求の処理 172

## Web サイト ホスティング

複数の Web サイト 132

## Web サイトのトラフィック 114

## Web ページ

既定の使用順 138

## webapp.properties 198

サブレット マッピングの変  
更 136

## WebSite Pro

URL 接頭辞 56

URL 接頭辞のマッピング 54

アプリケーションの公開 127

接続 58

設定 54

ファイル拡張子 57

変更 60

マルチホーム機能 56

## Windows レジストリ

IIS 3.0/PWS 39

Wireless Markup Language、MIME  
タイプ

## WML

「Wireless Markup Language」を  
参照

## Z

## Zeus Web サーバー

接続 63

変更 65

## あ

アップグレード、JRun 74

## アプリケーション

invoice 20

JMC 19

「Webアプリケーション」も参照

既定値 19

デモ 20

アプリケーション URL、変更 129

アプリケーションパス、  
マッピング 131, 135

アプリケーションパネル 123

アプリケーションパラメータ、  
追加 134

## アプリケーション ホスト

公開時の選択 127

削除 133

作成 132

変更 129

[アプリケーションホスト]  
パネル 133

アプリケーションホストの  
作成 132

アプリケーションホストの  
追加 132

アプリケーションの説明

変更 129

アプリケーションのルートディレ  
クトリ、変更 129

## アプリケーション変数

削除 134

追加 134

[アプリケーション変数]

パネル 134

アプリケーション名、変更 129

## い

イベント ログ  
 JRun サーバー 108  
 アプリケーション 141  
 インストール  
 JRE がない場合 9  
 JRun のコンポーネント 6  
 UNIX と Linux 12  
 Windows 3  
 必要条件 xi

## え

エラー  
 404 見つかりません 26, 67  
 500 Internal Server Error 67  
 reverting to Developer  
 Edition 68  
 Too Many Concurrent  
 requests 68  
 エンタープライズ アプリケーシ  
 ョン、構成 156  
 エンドポイント プロパティ  
 JWS 117  
 外部 Web サーバー 119

## お

オブジェクト定義 51  
 オンライン プロセス 68

## か

開始パラメータ 134  
 開始、JMC 70  
 外部 Web サーバー  
 Apache 31  
 CGI の使用 62  
 IIS 3.0 36  
 Java ベースの Web サーバー 61  
 JCM の設定 119  
 NES/iPlanet 47  
 PWS 36  
 WebSite Pro 54  
 Zeus 63  
 エンドポイント プロパティ 119  
 概要 30, 114  
 カスタム コネクタ 194  
 コンフィギュレーション ファイ  
 ルのサンプル 174  
 接続 30  
 [外部 Web サーバー] パネル 120,  
 187  
 コネクタ  
 概要 172  
 ポート  
 概要 168

拡張子、サーブレットへの  
 マッピング 150  
 カスタム コネクタ 194  
 Apache 195  
 Netscape/iPlanet 196  
 仮想パス 149  
 JWS 用の作成 131  
 サーブレットの URL 149  
 仮想ホスティング 133  
 [仮想マッピング] パネル 131  
 関連付け、MIME タイプ 142

## き

キーストア 103  
 既定アプリケーション 122  
 既定ディレクトリ  
 説明 19  
 既定のドキュメント、使用順 138

## く

クッキー 143

## け

キー、検索 166

## こ

コネクタ  
 local.properties 175  
 カスタム 194  
 「コネクタ ウィザード」も参照  
 プロパティ 173  
 コネクタ ウィザード 115  
 Apache の使用 33  
 IIS 3.0 の使用 37  
 IIS 4.0/5.0 40  
 IIS 4.0/5.0 の使用 42  
 NES/iPlanet の使用 48  
 PWS の使用 37  
 WebSite Pro の使用 59  
 Zeus の使用 64  
 概要 30  
 トラブルシューティング 66  
 メタベースの変更 44  
 コンテキスト  
 Bean 160  
 サーブレット 146  
 コンテキストパス  
 URL マッピング 148  
 サーブレットの登録 146

## さ

サーバー  
 「JRun Web サーバー」を参照  
 「JRun サーバー」を参照

「外部 Web サーバー」を参照  
 サーバー コネクタ ポート  
 local.properties 66  
 サーバー側インクルード  
 (SSI) 155, xvii  
 [サーバー側インクルードの設定]  
 パネル 155  
 サーバー管理コンソール 87  
 サーバーのディレクトリ 19  
 サービス  
 「NT サービス」を参照  
 サーブレット  
 CGI の実行 62  
 invoker 149  
 URL マッピング 149  
 エイリアス設定 151  
 コンテキストパス 146  
 サポート xx  
 初期化パラメータ 147  
 接尾辞のマッピング 150  
 チェーン化 152  
 定義 146  
 ディレクトリの追加 135  
 登録 146  
 名前の変更 148  
 プリロードの順番 148  
 要求の URL のマッピング 148  
 [サーブレット URL のマッピング]  
 パネル 149  
 サーブレットディレクトリの追  
 加 135  
 サーブレット マッピング  
 default-app 136  
 [サーブレット定義] パネル 146  
 サーブレットによるフィルタリング  
 出力 152  
 サーブレットのエイリアス設  
 定 151  
 サーブレットのチェーン化 152  
 エイリアス使用 152  
 MIME タイプ 153  
 サポートされている JVM xii  
 参照 138

## し

システム要件 xi  
 縮小された機能 xvii  
 証明書  
 インストール 104  
 作成 98  
 認証 104  
 例 95  
 証明書署名要求 95  
 作成 98

ショートカット、「ようこそ」ページに追加 73  
 初期化パラメータ  
   サブレット 147  
   追加 134  
 証明書  
   削除 106  
   作成 95  
 シリアル番号  
   設定 74  
 シリアル番号の設定 74  
 [新規サーバーの構成] パネル 87  
   拡張 87

## す

スタートアップ時のロード 148  
 スタートアップ、サブレット、プリロードの順番 148  
 ステートの管理 143  
 スレッド 114

## せ

制御ポート 168  
 製品のラインナップ x  
 セキュリティ  
   JMC ユーザの管理 75  
   ディレクトリ構造の非表示 131  
   パスワードの変更 77,79  
   分散環境 186  
 セッショントラッキング 143  
   編集 143  
 接続プール 112  
 接尾辞 150

## た

タグレット 155

## つ

ツリー構造 18

## て

ディレクトリ  
   サブレットディレクトリの追加 135  
 ディレクトリ構造 18  
 ディレクトリ参照  
   許可 138  
 データソース  
   「JDBC データソース」を参照  
 データベース  
   「JDBC データソース」を参照  
 データベースアクセス、JDBC 109

テクニカルサポート、お問い合わせ先 xxv  
 デバッグ オプション  
   -nohup オプションあり 85  
   -start オプションあり 85  
 デモ アプリケーション 20  
   起動 23  
   トラブルシューティング 67  
 stderr、転送 84  
 stdout、転送 84

## と

ドキュメント、既定の使用順 138  
 トラブルシューティング  
   JRun が異常終了しました 27  
   JRun サーバープロセス 68  
   reverting to Developer Edition 68  
   アプリケーション イベント ログ 141  
   インストール 25  
   エラー 404 26,67  
   エラー 500 67  
   コネクタウィザード 66  
   デモ アプリケーション 67  
   同時ユーザ数の超過 68  
   ログインできない 25

## に

認証  
   ホストベース 186  
 認証局 95

## ね

ネイティブコネクタ  
 NES/iPlanet 51  
 サンプル obj.conf 53

## は

バインドアドレス 172  
 パスワード  
   admin の変更 79  
   ユーザの変更 77  
 パス、仮想 149  
 バックグラウンドプロセス 68

## ひ

必要条件  
   Java xii  
   JVM xii  
   ソフトウェア xi  
   ハードウェア xi

## ふ

ファイル拡張子  
   接頭辞へのマッピング 150  
 ファイル設定、変更 138  
 ファイルの関連付け、MIME タイプ 142  
 ファイルの索引、指定順 138  
 [ファイルの設定] パネル 138  
 プール要求 114  
 フォルダ 18  
 負荷管理 114  
 複数の Web サイトのホスト 132  
 プリロードの順番の設定 148  
 プリロードの順番、設定 148  
 プロキシポート  
   ローカルプロパティ 66  
 プロセス  
   バックグラウンドでの起動 68  
 プロパティ  
   local 201  
   PropertyScript の使用法 204  
   webapp 201  
   グローバル 201  
 プロパティファイル  
   PropertyScript の使用法 204  
   階層、図 200  
   概要 198  
   構文 202  
   説明 198  
   編集 202  
   変数 202  
 プロパティファイルの編集 202, 204  
 文書  
   オンライン xxii

## へ

並行処理  
   JRun Web サーバー 118  
   Too Many Concurrent requests 68  
   外部 Web サーバー 121  
   概要 114  
 変更、シリアル番号 74  
 変数  
   プロパティファイル 202

## ほ

ポート 170  
   空きの検出 170  
   保護されていないものの無効化 107  
 保護されていないポートの無効化 107



## ホスト

「アプリケーション ホスト」を  
参照

ホスト ヘッダ 132

保存された JMC リンク 73

## ま

マルチホーム機能 132

マルチホスティング 188

Apache 189

IIS 190

Netscape/iPlanet 191

## め

メタベース

変更 44

## も

モジュール

Apache 31

## ゆ

ユーザ

管理 75

削除 78

設定の変更 77

追加 75

パスワードの変更 77

ユーザの削除 78

## よ

要求 114

要求のチェーン化 192

「ようこそ」ページ

図 71

リンクの追加 73

「ようこそ」ページへの追加 73

## ら

ライセンスキー 74

## り

リソース

オンライン xxiv

書籍 xxiii

## る

ルート ディレクトリ、変更 129

## ろ

ログ

JMC でログ ファイルを表  
示 163

JRun サーバー イベント 108

## JVM 92

アプリケーションの  
イベント 141

出力の転送 84

プロパティ 203

ログアウト 166

ログ ファイルビューア 163

[ログ ファイルの設定] パネル 108

ログイン 71

[ログの設定] パネル 141

