



Red Hat JBoss Enterprise Application Platform 7.1

Handbuch zum Einstieg

Zur Verwendung mit der Red Hat JBoss Enterprise Application Platform 7.1

Red Hat JBoss Enterprise Application Platform 7.1 Handbuch zum Einstieg

Zur Verwendung mit der Red Hat JBoss Enterprise Application Platform 7.1

Rechtlicher Hinweis

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution-Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Zusammenfassung

Dieses Handbuch bietet grundlegende Informationen, um den Benutzern den Einstieg in die Red Hat JBoss Enterprise Application Platform 7.1 zu erleichtern.

Inhaltsverzeichnis

KAPITEL 1. EINFÜHRUNG	4
1.1. ÜBER DIE RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7	4
1.2. ÜBER DAS HANDBUCH ZUM EINSTIEG	4
KAPITEL 2. VERWALTEN DER JBOSS EAP	5
2.1. HERUNTERLADEN UND INSTALLIEREN DER JBOSS EAP	5
2.1.1. Installationsvoraussetzungen	5
2.1.2. Herunterladen der JBoss EAP	5
2.1.3. Installieren der JBoss EAP	5
2.2. STARTEN UND ANHALTEN DER JBOSS EAP	6
2.2.1. Starten der JBoss EAP	6
Starten der JBoss EAP als Standalone-Server	6
Starten der JBoss EAP in einer Managed Domain	7
2.2.2. Anhalten der JBoss EAP	7
Anhalten einer interaktiven Instanz der JBoss EAP	7
Anhalten einer Hintergrund-Instanz der JBoss EAP	7
2.3. JBOSS EAP MANAGEMENT	8
2.3.1. Verwaltungsbenutzer	8
2.3.1.1. Hinzufügen eines neuen Verwaltungsbenutzers	8
2.3.1.2. Nicht-interaktives Ausführen des Dienstprogramms Benutzer-Hinzufügen	9
Erstellen eines neuen Benutzers, der zu mehreren Gruppen gehört	10
Geben Sie eine alternative Eigenschaften-Datei an	10
2.3.2. Verwaltungs-Schnittstellen	10
2.3.2.1. Management CLI	10
Starten des Management CLI	11
Verbinden mit einem laufenden Server	11
Hilfe anzeigen	11
Management CLI beenden	11
Systemeinstellungen ansehen	11
Systemeinstellungen aktualisieren	12
Server starten	12
2.3.2.2. Management-Konsole	12
2.3.3. Konfigurationsdateien	13
2.3.3.1. Konfigurationsdateien des Standalone-Servers	13
2.3.3.2. Konfigurationsdateien der Managed Domain	14
2.3.3.3. Sichern der Konfigurationsdaten	14
2.3.3.4. Konfigurationsdatei-Snapshots	15
Erstellen eines Snapshots	15
Liste aller Snapshots	15
Löschen eines Snapshots	16
Starten des Servers mit einem Snapshot	16
2.3.3.5. Eigenschaften-Austausch	16
Verschachtelte Ausdrücke	17
Eigenschaften-Austausch auf Deskriptor-Basis	17
2.4. NETZWERK- UND PORT-KONFIGURATION	18
2.4.1. Schnittstellen	18
2.4.1.1. Standard Schnittstellen-Konfigurationen	19
2.4.1.2. Konfigurieren von Schnittstellen	19
Hinzufügen einer Schnittstelle mit einem NIC Wert	20
Hinzufügen einer Schnittstelle mit mehreren bedingten Werten	20
Aktualisieren eines Schnittstellenattributs	20

Hinzufügen einer Schnittstelle zu einem Server in einer Managed Domain	20
2.4.2. Socket-Bindungen	21
2.4.2.1. Management Ports	21
2.4.2.2. Standard Socket-Bindungen	21
Standalone-Server	22
Managed Domain	22
2.4.2.3. Konfigurieren von Socket-Bindungen	23
2.4.2.4. Port Offsets	24
2.4.3. IPv6 Adressen	25
Konfigurieren von JVM-Stack für IPv6-Adressen	25
Aktualisieren der Schnittstellen-Deklarationen für IPv6 Adressen	25
KAPITEL 3. ENTWICKLUNG VON ANWENDUNGEN, DIE DIE JBOSS EAP VERWENDEN	27
3.1. ÜBERBLICK	27
3.2. EINRICHTEN DER ENTWICKLUNGSUMGEBUNG	27
3.3. VERWENDUNG VON QUICKSTART-BEISPIELEN	27
3.3.1. Über Maven	27
3.3.2. Verwendung von Maven mit Quickstarts	28
3.3.3. Download und Ausführung der Quickstarts	28
3.3.3.1. Download der Quickstarts	28
3.3.3.2. Ausführen der Quickstarts im JBoss Developer Studio	28
3.3.3.3. Ausführen der Quickstarts von der Befehlszeile aus	36
3.4. GEHEN SIE DIE QUICKSTART-BEISPIELE DURCH	36
3.4.1. Erkunden Sie den Quickstart helloworld	36
Voraussetzungen	37
Untersuchen Sie die Verzeichnisstruktur	37
Untersuchen Sie den Code	37
3.4.2. Erkunden Sie den numberguess-Quickstart	38
Voraussetzungen	39
Untersuchen Sie die Konfigurationsdateien	39
3.4.2.1. Gehen Sie den JSF-Code durch	40
3.4.2.2. Gehen Sie die Klassendateien durch	41
ANHANG A. REFERENZMATERIAL	46
A.1. SERVER-LAUFEIT ARGUMENTE	46
A.2. DIENSTPROGRAMM-ARGUMENT FÜR BENUTZER-HINZUFÜGEN	50
A.3. SCHNITTSTELLEN-ATTRIBUTE	51
A.4. SOCKET-BINDUNG ATTRIBUTE	53
A.5. STANDARD SOCKET-BINDUNGEN	54

KAPITEL 1. EINFÜHRUNG

1.1. ÜBER DIE RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7

Die Red Hat JBoss Enterprise Application Platform 7 (JBoss EAP) ist eine Middleware-Plattform, die auf offenen Standards basiert und mit der Java Enterprise Edition 7 Spezifikation konform geht.

Die JBoss EAP enthält eine modulare Struktur, die es ermöglicht, Dienste erst bei Bedarf zu aktivieren und so den Systemstart zu beschleunigen.

Die Management-Konsole und das Management Command-Line Interface (CLI) machen das Bearbeiten von XML-Konfigurationsdateien überflüssig und ermöglichen die Verwendung von Skripten und das Automatisieren von Aufgaben.

Die JBoss EAP bietet zwei Betriebsmodi für JBoss EAP Instanzen: Standalone-Server oder Managed Domain. Der Betriebsmodus Standalone-Server steht für die Ausführung der JBoss EAP als einzelne Server-Instanz. Der Betriebsmodus Managed Domain ermöglicht die Verwaltung mehrerer JBoss EAP Instanzen von einem einzelnen Kontrollpunkt aus.

Des Weiteren beinhaltet die JBoss EAP APIs und Development Frameworks zur schnellen Entwicklung sicherer und skalierbarer Java EE Anwendungen.

1.2. ÜBER DAS HANDBUCH ZUM EINSTIEG

Der Zweck dieses Handbuchs besteht darin, Ihnen möglichst schnell zum Einsatz der JBoss EAP zu verhelfen. Es umfasst [administrative](#) Aufgaben wie Installation, Management und Konfiguration der JBoss EAP. Dieses Handbuch hilft zudem den [Entwicklern](#) beim Schreiben von Java EE 7 Anwendungen unter Verwendung der JBoss EAP Quickstarts.

Weitere Informationen finden Sie in der vollständigen [JBoss EAP-Dokumentation](#).

KAPITEL 2. VERWALTEN DER JBOSS EAP

2.1. HERUNTERLADEN UND INSTALLIEREN DER JBOSS EAP

Dieses Handbuch bietet eine grundlegende Anleitung zum Herunterladen und Installieren von JBoss EAP unter Verwendung der Plattform-unabhängigen Zip-Installation.

Weitere Informationen finden Sie im [Installationshandbuch](#), darunter auch eine Anleitung zur Installation von JBoss EAP über die grafische Installation oder über RPM-Paketinstallation.

2.1.1. Installationsvoraussetzungen

Stellen Sie vor der Installation der JBoss EAP sicher, dass die folgenden Voraussetzungen erfüllt wurden.

Allgemeine Voraussetzungen

- Ihr System wird gemäß der [von JBoss EAP 7 unterstützten Konfigurationen](#) unterstützt.
- Ihr System ist hinsichtlich der von Red Hat herausgegebenen Updates und Errata auf dem neusten Stand.

Voraussetzungen für die Zip-Installation

- Der Benutzer, der die JBoss EAP ausführen wird, hat Lese- und Schreibzugriff auf das Installationsverzeichnis.
- Das gewünschte Java Development Kit wurde installiert
- Für Hewlett-Packard HP-UX wurde ein **unzip** Dienstprogramm installiert.
- Für Windows Server wurden die **JAVA_HOME** und **PATH** Umgebungsvariablen eingestellt.

2.1.2. Herunterladen der JBoss EAP

Die JBoss EAP Zip-Datei ist im Red Hat Kundenportal verfügbar. Die Installation per Zip-Datei ist unabhängig von der Plattform.

1. Melden Sie sich beim [Red Hat Kundenportal](#) an.
2. Klicken Sie auf **Downloads**.
3. Klicken Sie auf **Red Hat JBoss Enterprise Application Platform** in der Liste **Produkt Downloads**.
4. Wählen Sie im Dropdown-Menü **Version** die Version **7.1** aus.
5. Gehen Sie zu **Red Hat JBoss Enterprise Application Platform 7.1.0** auf der Liste und klicken Sie auf **Download**.

2.1.3. Installieren der JBoss EAP

Wenn die JBoss EAP Zip-Installationsdatei heruntergeladen ist, kann sie durch Entpacken des Paketinhalts installiert werden.

1. Verschieben Sie gegebenenfalls die Zip-Datei auf den Server und an den Speicherort, an dem die JBoss EAP installiert werden soll.



ANMERKUNG

Der Benutzer, der die JBoss EAP ausführt, muss Lese- und Schreibzugriff auf dieses Verzeichnis haben.

2. Entpacken Sie das Zip-Archiv.

```
$ unzip jboss-eap-7.1.0.zip
```



ANMERKUNG

Wählen Sie beim Windows Server per Rechtsklick auf die Zip-Datei **Alle Entpacken**.

Das durch Entpacken des Zip-Archivs erstellte Verzeichnis ist das Verzeichnis der obersten Ebene für die JBoss EAP Installation. Dieses wird als **EAP_HOME** bezeichnet.

2.2. STARTEN UND ANHALTEN DER JBOSS EAP

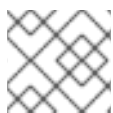
2.2.1. Starten der JBoss EAP

JBoss EAP wird von Red Hat Enterprise Linux, Windows Server, Oracle Solaris und Hewlett-Packard HP-UX und kann entweder als Standalone-Server oder im Betriebsmodus Managed Domain ausgeführt werden. Der genaue Befehl zum Start von JBoss EAP hängt von der zugrundeliegenden Plattform und dem gewünschten Betriebsmodus ab.

Server werden zunächst im angehaltenen Modus gestartet und nehmen erst Anfragen an, wenn alle erforderlichen Dienste gestartet wurden. Zu diesem Zeitpunkt werden Server in einem normalen Ausführungsmodus versetzt und können beginnen, Anfragen anzunehmen.

Starten der JBoss EAP als Standalone-Server

```
$ EAP_HOME/bin/standalone.sh
```



ANMERKUNG

Benutzen Sie für Windows Server das **EAP_HOME\bin\standalone.bat** Skript.

Dieses Startup-Skript benutzt die Datei **EAP_HOME/bin/standalone.conf** oder **standalone.conf.bat** für Windows Server, um Standardpräferenzen, wie JVM-Optionen, einzustellen. Sie können die Einstellungen in dieser Datei anpassen.

JBoss EAP benutzt standardmäßig die **standalone.xml** Konfigurationsdatei, kann aber auch unter Verwendung einer anderen gestartet werden. Weitere Details zu verfügbaren Standalone-Konfigurationsdateien und deren Gebrauch finden Sie unter [Konfigurationsdateien des Standalone-Servers](#).

Eine vollständige Liste aller verfügbaren Startup-Script-Argumente und ihrer Zwecke finden Sie im **--help** Argument oder im Abschnitt [Server-Runtime-Argumente](#).

Starten der JBoss EAP in einer Managed Domain

Der Domain-Controller muss vor den Servern in allen Servergruppen in der Domain gestartet werden. Benutzen Sie dieses Skript um zuerst den Domain-Controller und dann jeden zugehörigen Host-Controller zu starten.

```
$ EAP_HOME/bin/domain.sh
```



ANMERKUNG

Verwenden Sie für Windows Server das Skript **EAP_HOME\bin\domain.bat**.

Dieses Startup-Skript benutzt die Datei **EAP_HOME/bin/domain.conf** oder **domain.conf.bat** für Windows Server, um Standardpräferenzen, wie JVM-Optionen, einzustellen. Sie können die Einstellungen in dieser Datei anpassen.

JBoss EAP benutzt standardmäßig die **host.xml** Host-Konfigurationsdatei, kann aber auch unter Verwendung einer anderen gestartet werden. Details zu verfügbaren Konfigurationsdateien der Managed Domain und deren Gebrauch finden Sie im Abschnitt [Konfigurationsdateien der Managed Domain](#).

Beim Einrichten einer Managed Domain müssen zusätzliche Argumente in das Startup Skript übergeben werden. Eine vollständige Liste aller verfügbaren Startup Skript Argumente und ihrer Zwecke finden Sie im **--help** Argument oder im Abschnitt [Server-Runtime-Argumente](#).

2.2.2. Anhalten der JBoss EAP

Wie Sie die JBoss EAP anhalten können, hängt davon ab, wie Sie sie gestartet haben.

Anhalten einer interaktiven Instanz der JBoss EAP

Drücken Sie **Ctrl+C** im Terminal, in dem die JBoss EAP gestartet wurde.

Anhalten einer Hintergrund-Instanz der JBoss EAP

Benutzen Sie das Management CLI, um eine Verbindung mit der laufenden Instanz herzustellen und den Server herunterzufahren.

1. Starten Sie das Management CLI.

```
$ EAP_HOME/bin/jboss-cli.sh --connect
```

2. Geben Sie den **shutdown** Befehl.

```
Herunterfahren
```



ANMERKUNG

Bei der Ausführung in einer Managed Domain müssen Sie zum Herunterfahren den Hostnamen angeben, indem Sie das **--host** Argument mit dem **shutdown** Befehl verwenden.

2.3. JBOSS EAP MANAGEMENT

JBoss EAP verwendet eine vereinfachte Konfiguration mit einer Konfigurationsdatei pro Standalone-Server oder Managed Domain. Die Standard-Konfiguration für einen Standalone-Server ist in der **`EAP_HOME/standalone/configuration/standalone.xml`** Datei gespeichert und die Standard-Konfiguration für eine Managed Domain ist in der Datei **`EAP_HOME/domain/configuration/domain.xml`** gespeichert. Ergänzend ist die Standard-Konfiguration für einen Host-Controller in der Datei **`EAP_HOME/domain/configuration/host.xml`** gespeichert.

JBoss EAP kann über das Befehlszeilen-Management, die Management-Konsole auf Web-Basis, das Java-API oder das HTTP-API konfiguriert werden. Änderungen mittels dieser Management-Schnittstellen bleiben automatisch bestehen und die XML Konfigurationsdateien werden durch das Management API überschrieben. Die Management-Befehlszeile und die Managementkonsole stellen die bevorzugten Methoden dar, und es ist nicht empfehlenswert, die XML Konfigurationsdateien manuell zu bearbeiten.

2.3.1. Verwaltungsbutzer

Die Standard JBoss EAP-Konfiguration bietet lokale Authentifizierung, damit ein Benutzer ohne Authentifizierung auf die Management-Befehlszeile auf dem lokalen Host zugreifen kann.

Sie müssen jedoch mindestens einen administrativen Benutzer erstellen, um remote auf die Management-Befehlszeile zuzugreifen oder die Managementkonsole zu verwenden, die als Remotezugriff betrachtet wird, auch wenn der Traffic vom Localhost kommt. Wenn Sie versuchen, auf die Management-Konsole zuzugreifen, bevor Sie einen Benutzer hinzugefügt haben, wird Ihnen eine Fehlermeldung angezeigt.

Wurde JBoss EAP über die grafische Installation installiert, wird während dem Installationsprozess ein Management-Benutzer erstellt.

Dieser Leitfaden umfasst die einfache Benutzerverwaltung für JBoss EAP über das Script **`add-user`**, ein ein Dienstprogramm für das Hinzufügen neuer Benutzer zur Eigenschaften-Datei für sofort verfügbare Authentifizierung.

Erweiterte Optionen zur Authentifizierung und Autorisierung, wie LDAP oder Role-Based Access Control (RBAC) finden Sie im Abschnitt [Core Management Authentifizierung](#) des JBoss EAP-Handbuchs *Sicherheitsarchitektur*.

2.3.1.1. Hinzufügen eines neuen Verwaltungsbutzers

1. Führen Sie das Skript des **`add-user`** Dienstprogramms aus und folgen Sie den Aufforderungen.

```
$ EAP_HOME/bin/add-user.sh
```



ANMERKUNG

Verwenden Sie für Windows Server das Script **`EAP_HOME\bin\add-user.bat`**.

2. Drücken Sie **ENTER** um die Standardoption **a** zum Hinzufügen eines Verwaltungsbutzers zu wählen.
Dieser User wird der *ManagementRealm* hinzugefügt und autorisiert, Management-

Vorgänge über die Managementkonsole oder die Befehlszeile auszuführen. Die andere Auswahl, **b**, fügt einen Benutzer der *ApplicationRealm* hinzu, die für Anwendungen verwendet wird und keine besonderen Rechte bietet.

3. Geben Sie die gewünschten Angaben für Benutzernamen und Passwort ein. Sie werden dann aufgefordert, das Passwort zu bestätigen.



ANMERKUNG

Benutzernamen können nur die folgenden Zeichen enthalten, in beliebiger Anzahl und Reihenfolge:

- Alphanumerische Zeichen (a-z, A-Z, 0-9)
- Striche (-), Punkte (.), Kommas (,) @-Zeichen
- Backslash (\)
- Gleichheitszeichen (=)

Standardmäßig erlaubt JBoss EAP schwache Passwörter, gibt jedoch eine Warnung aus.

Weitere Informationen zum Ändern dieses Standardverhaltens finden Sie im Abschnitt [Einstellen von Passwortbeschränkungen im Dienstprogramm zum Hinzufügen von Benutzern](#) des *JBoss EAP Konfigurationshandbuchs*.

4. Geben Sie eine Komma-getrennte Liste von Gruppen ein, denen der Benutzer zugehören soll. Wenn Sie nicht möchten, dass der Benutzer einer Gruppe zugehört, drücken Sie **ENTER** um das Feld frei zu lassen.
5. Überprüfen Sie die Informationen und geben Sie **yes** zur Bestätigung ein.
6. Bestimmen Sie, ob dieser Benutzer eine entfernte JBoss EAP Server-Instanz darstellt. Geben Sie für einen einfachen Verwaltungsbenutzer **no** ein.
Ein Benutzertyp, der u.U. zur *ManagementRealm* hinzugefügt werden muss, repräsentiert eine andere Instanz der JBoss EAP, die sich authentifizieren können muss um Mitglied eines Clusters werden. Ist dies der Fall, so antworten Sie mit **yes** auf diese Aufforderung und Ihnen wird ein secret Hash-Wert als Benutzerpasswort gegeben, der einer anderen Konfigurationsdatei hinzugefügt werden muss.

Benutzer können auch nicht-interaktiv erstellt werden, indem Parameter an das **add-user** Skript übergeben werden. Dieser Ansatz wird für gemeinsam genutzte Systeme nicht empfohlen, weil die Passwörter in Protokoll- und Verlauf-Dateien sichtbar sind. Weitere Informationen finden Sie unter [Nicht-interaktives Ausführen des Dienstprogramms Benutzer-Hinzufügen](#).

2.3.1.2. Nicht-interaktives Ausführen des Dienstprogramms Benutzer-Hinzufügen

Sie können das **add-user** Skript nicht-interaktiv ausführen, indem Sie Argumente an der Befehlszeile übergeben. Es muss mindestens Benutzername und Passwort angegeben werden.



WARNUNG

Dieser Ansatz wird für gemeinsam benutzte Systeme nicht empfohlen, weil die Passwörter in Protokoll- und Verlauf-Dateien sichtbar sind.

Erstellen eines neuen Benutzers, der zu mehreren Gruppen gehört

Der folgende Befehl fügt einen Management-Benutzer hinzu, **mgmtuser1**, mit den Gruppen **guest** und **mgmtgroup**.

```
$ EAP_HOME/bin/add-user.sh -u 'mgmtuser1' -p 'password1!' -g
'guest,mgmtgroup'
```

Geben Sie eine alternative Eigenschaften-Datei an

Standardmäßig sind Informationen zu Benutzern und Gruppen, die mittels des **add-user** Skripts erstellt wurden, in den Eigenschaften-Dateien gespeichert, die sich im Konfigurationsverzeichnis des Servers befinden.

Benutzerinformationen sind in den folgenden Eigenschaften-Dateien gespeichert:

- **EAP_HOME/standalone/configuration/mgmt-users.properties**
- **EAP_HOME/domain/configuration/mgmt-users.properties**

Gruppeninformationen sind in den folgenden Eigenschaften-Dateien gespeichert:

- **EAP_HOME/standalone/configuration/mgmt-groups.properties**
- **EAP_HOME/domain/configuration/mgmt-groups.properties**

Diese Standardverzeichnisse und Namen der Eigenschaften-Dateien können ersetzt werden. Der folgende Befehl fügt einen neuen Benutzer hinzu und gibt einen anderen Namen und Speicherort für die Eigenschaften-Dateien des Benutzers an.

```
$ EAP_HOME/bin/add-user.sh -u 'mgmtuser2' -p 'password1!' -sc
'/path/to/standaloneconfig/' -dc '/path/to/domainconfig/' -up
'newname.properties'
```

Der neue Benutzer wurde der Benutzereigenschaften-Datei unter **/path/to/standaloneconfig/newname.properties** und **/path/to/domainconfig/newname.properties** hinzugefügt. Beachten Sie, dass diese Dateien bereits bestehen müssen; ansonsten wird Ihnen ein Fehler angezeigt.

Eine vollständige Liste aller verfügbaren **add-user** Argumente und ihrer Zwecke finden Sie im **--help** Argument oder im Abschnitt [Argumente für das Dienstprogramm Benutzer-Hinzufügen](#).

2.3.2. Verwaltungs-Schnittstellen

2.3.2.1. Management CLI

Das Management Command-Line Interface (CLI) ist ein Befehlszeilen-Verwaltungstool für JBoss EAP.

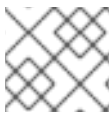
Benutzen Sie das Management CLI zum Starten und Anhalten von Servern, zum Bereitstellen und Deinstallieren von Anwendungen, zum Konfigurieren von Systemeinstellungen und zum Ausführen weiterer administrativer Aufgaben. Das Ausführen von Vorgängen im Batch-Modus ermöglicht mehrere Aufgaben als Gruppe auszuführen.

Viele allgemeine Terminal-Befehle sind verfügbar, wie **ls**, **cd** und **pwd**. Das Management CLI unterstützt auch die Vervollständigung mit der Tab-Taste.

Detaillierte Informationen zur Verwendung der Management-Befehlszeile einschließlich Befehle und Operationen, Syntax und Ausführung im Stapelmodus finden Sie im JBoss EAP [Leitfaden zur Management-Befehlszeile](#).

Starten des Management CLI

```
$ EAP_HOME/bin/jboss-cli.sh
```



ANMERKUNG

Verwenden Sie für Windows Server das Script **EAP_HOME\bin\jboss-cli.bat**

Verbinden mit einem laufenden Server

```
verbinden
```

Sie können auch das Management CLI starten und über den Befehl **EAP_HOME/bin/jboss-cli.sh --connect** Befehl in einem Schritt verbinden.

Hilfe anzeigen

Benutzen Sie folgenden Befehl für allgemeine Hilfe

```
Hilfe
```

Verwenden Sie die Flag **--help** bei einem Befehl, um Anweisungen zur Verwendung dieses Befehls zu erhalten. Führen Sie z.B. folgenden Befehl aus, um Informationen zur Verwendung von **deploy** zu erhalten.

```
deploy --help
```

Management CLI beenden

```
Beenden
```

Systemeinstellungen ansehen

Der folgende Befehl benutzt den **read-attribute** Vorgang um anzuzeigen, ob die Beispiel-Datenquelle aktiviert ist.

```
/subsystem=datasources/data-source=ExampleDS:read-attribute(name=enabled)
{
  "outcome" => "success",
  "result" => true
}
```

■
Bei der Ausführung in einer Managed Domain müssen Sie angeben, welches Profil aktualisiert werden soll, indem Sie dem Befehl `/profile=PROFILE_NAME` voranstellen.

```
/profile=default/subsystem=datasources/data-source=ExampleDS:read-attribute(name=enabled)
```

Systemeinstellungen aktualisieren

Der folgende Befehl benutzt den **write-attribute** Vorgang um die Beispiel-Datenquelle zu deaktivieren.

```
/subsystem=datasources/data-source=ExampleDS:write-attribute(name=enabled,value=false)
```

Server starten

Das Management CLI kann auch benutzt werden um Server zu starten und anzuhalten, wenn sie in einer Managed Domain ausgeführt werden.

```
/host=HOST_NAME/server-config=server-one:start
```

2.3.2.2. Management-Konsole

Die Management-Konsole ist ein Verwaltungs-Tool auf Web-Basis für die JBoss EAP.

Verwenden Sie die Management-Konsole für das Starten und Anhalten von Servern, zum Bereitstellen und De-Installieren von Anwendungen, für die Feinabstimmung von Systemeinstellungen und die Durchführung permanenter Modifikationen an der Server-Konfiguration. Die Management-Konsole kann auch für administrative Aufgaben wie Live-Benachrichtigungen eingesetzt werden, wenn Änderungen durch den aktuellen Benutzer einen Neustart oder ein erneutes Laden der Server-Instanz erfordern.

In einer Managed Domain können Server-Instanzen und Server-Gruppen in derselben Domain zentral von der Management-Konsole des Domain-Controllers aus verwaltet werden.

Für eine JBoss EAP Instanz, die auf einem Localhost ausgeführt wird und den standardmäßigen Management Port benutzt, kann über einen Web-Browser bei <http://localhost:9990/console/App.html> auf die Management-Konsole zugegriffen werden. Sie müssen sich mit einem Benutzer authentifizieren, der Zugriffsberechtigungen für die Management-Konsole hat.

Die Management-Konsole bietet folgende Tabs zur Navigation und Verwaltung Ihres JBoss EAP Standalone Servers oder Ihrer Managed Domain.

Home

Lernen Sie mehr über verschiedene allgemeine Konfigurations- und Verwaltungsaufgaben. Machen Sie sich mithilfe unserer Tour mit der JBoss EAP Management-Konsole vertraut.

Deployments

Hinzufügen, Entfernen und Aktivieren von Deployments. Zuweisung von Deployments zu Gruppen in einer Managed Domain.

Konfiguration

Konfigurieren Sie verfügbare Subsysteme, die gewisse Fähigkeiten bereitstellen, wie Web Services, Messaging oder Hochverfügbarkeit. Verwalten Sie Profile mit unterschiedlichen Subsystem Konfigurationen in einer Managed Domain.

Laufzeit

Lassen Sie sich Laufzeit-Informationen wie Server Status, JVM Verbrauch und Server Protokolle anzeigen. Verwalten Sie Ihre Hosts, Servergruppen und Server in einer Managed Domain.

Zugriffskontrolle

Weisen Sie unter Verwendung der Rollen-basierten Zugriffskontrolle den Benutzern und Gruppen bestimmte Rollen zu.

Patching

Patches auf Ihre JBoss EAP Instanzen anwenden.



ANMERKUNG

Eine Einführung in die Management-Konsole finden Sie über den Link **Einführung** auf der Homepage der Management-Konsole.

2.3.3. Konfigurationsdateien

2.3.3.1. Konfigurationsdateien des Standalone-Servers

Die Standalone-Konfigurationsdateien befinden sich im **EAP_HOME/standalone/configuration/** Verzeichnis. Für jedes der fünf vordefinierten Profile (*default*, *ha*, *full*, *full-ha*, *load-balancer*) existiert eine separate Datei.

Tabelle 2.1. Standalone-Konfigurationsdateien

Konfigurationsdatei	Verwendungszweck
standalone.xml	Diese Standalone-Konfigurationsdatei ist die Standardkonfiguration, die benutzt wird, wenn Sie Ihren Standalone-Server starten. Sie enthält alle Informationen über den Server, einschließlich Subsysteme, Networking, Deployment, Socket-Bindungen und andere konfigurierbare Details. Sie bietet nicht die zum Messaging oder zur Hochverfügbarkeit notwendigen Subsysteme.
standalone-ha.xml	Diese Standalone-Konfigurationsdatei umfasst alle Standard-Subsysteme und fügt die modcluster und jgroups Subsysteme für die Hochverfügbarkeit hinzu. Sie stellt nicht die zum Messaging notwendigen Subsysteme bereit.
standalone-full.xml	Diese Standalone-Konfigurationsdatei umfasst alle Standard-Subsysteme und fügt die messaging-activemq und iiop-openjdk Subsysteme hinzu. Sie stellt nicht die für Hochverfügbarkeit notwendigen Subsysteme bereit.
standalone-full-ha.xml	Diese Standalone-Konfigurationsdatei beinhaltet Support für jedes mögliche Subsystem, einschließlich der Subsysteme für Messaging und Hochverfügbarkeit.

Konfigurationsdatei	Verwendungszweck
standalone-load-balancer.xml	Diese Standalone-Konfigurationsdatei umfasst die mindest erforderlichen Subsysteme, um den eingebauten Front-End Load Balancer <code>mod_cluster</code> zum Load Balancing anderer JBoss EAP-Instanzen zu verwenden.

Der Start der JBoss EAP als Standalone-Server benutzt standardmäßig die **standalone.xml** Datei. Um die JBoss EAP mit einer anderen Konfiguration zu starten, benutzen Sie das **--server-config** Argument. Zum Beispiel

```
$ EAP_HOME/bin/standalone.sh --server-config=standalone-full.xml
```

2.3.3.2. Konfigurationsdateien der Managed Domain

Die Konfigurationsdateien der Managed Domain befinden sich im Verzeichnis **EAP_HOME/domain/configuration/**.

Tabelle 2.2. Konfigurationsdateien der Managed Domain

Konfigurationsdatei	Verwendungszweck
domain.xml	Dies ist die Hauptkonfigurationsdatei für eine Managed Domain. Nur der Domain-Master liest diese Datei. Diese Datei enthält die Konfigurationen für alle Profile (<i>default, ha, full, full-ha, load-balancer</i>).
host.xml	Diese Datei enthält spezifische Konfigurationsdetails für einen physischen Host in einer Managed Domain, wie Netzwerkschnittstellen, Socket-Bindungen, den Namen des Hosts und andere Host-spezifische Details. Die host.xml Datei umfasst alle Funktionen sowohl des host-master.xml , als auch des host-slave.xml , wie sie unten beschrieben sind.
host-master.xml	Diese Datei umfasst nur die Konfigurationsdetails, die zur Ausführung des Servers als Master Domain Controller notwendig sind.
host-slave.xml	Diese Datei umfasst nur die Konfigurationsdetails, die zur Ausführung des Servers als Managed Domain Host Controller notwendig sind.

Der Start der JBoss EAP in einer Managed Domain benutzt standardmäßig die **host.xml** Datei. Um die JBoss EAP mit einer anderen Konfiguration zu starten, benutzen Sie das **--host-config** Argument. Zum Beispiel

```
$ EAP_HOME/bin/domain.sh --host-config=host-master.xml
```

2.3.3.3. Sichern der Konfigurationsdaten

Um die JBoss EAP Serverkonfiguration später wieder herzustellen, sollten Elemente an den folgenden Speicherorten gesichert werden:

- **`EAP_HOME/standalone/configuration/`**
 - Sichern Sie das gesamte Verzeichnis um Benutzerdaten, Serverkonfiguration und Protokollierungseinstellungen für Standalone-Server zu speichern.
- **`EAP_HOME/domain/configuration/`**
 - Sichern Sie das gesamte Verzeichnis um Benutzer- und Profildaten, Domain und Host-Konfiguration, sowie Protokollierungseinstellungen für Managed Domains zu speichern.
- **`EAP_HOME/modules/`**
 - Sichern Sie jegliche benutzerdefinierten Module.
- **`EAP_HOME/welcome-content/`**
 - Sichern Sie jegliche benutzerdefinierten Willkommensinhalte.
- **`EAP_HOME/bin/`**
 - Sichern Sie jegliche benutzerdefinierten Skripts oder Start-Konfigurationsdateien.

2.3.3.4. Konfigurationsdatei-Snapshots

Um die Wartung und Verwaltung des Servers zu unterstützen erstellt die JBoss EAP zum Zeitpunkt der Inbetriebnahme eine mit Zeitstempel versehene Version der originalen Konfigurationsdatei. Durch Verwaltungs-Vorgänge erfolgte Konfigurations-Änderungen führen zum automatischen Backup der Ursprungs-Datei und zur Speicherung einer Arbeitskopie der Instanz für Referenz und Rollback. Zusätzlich können Konfigurations-Snapshots erstellt werden, welche Zeitpunkt-Kopien der Server-Konfiguration darstellen. Diese Snapshots können von Administratoren gespeichert und geladen werden.

Die folgenden Beispiele verwenden die **`standalone.xml`** Datei, aber derselbe Prozess gilt für **`domain.xml`** und **`host.xml`** Dateien.

Erstellen eines Snapshots

Benutzen Sie das Management CLI um einen Snapshot der aktuellen Konfigurationen zu erstellen.

```
:take-snapshot
{
    "outcome" => "success",
    "result" =>
    "EAP_HOME/standalone/configuration/standalone_xml_history/snapshot/2015102
    2-133109702standalone.xml"
}
```

Liste aller Snapshots

Verwenden Sie das Management CLI um alle erstellten Snapshots aufzulisten.

```
:list-snapshots
{
    "outcome" => "success",
```

```

    "result" => {
      "directory" =>
"EAP_HOME/standalone/configuration/standalone_xml_history/snapshot",
      "names" => [
        "20151022-133109702standalone.xml",
        "20151022-132715958standalone.xml"
      ]
    }
  }
}

```

Löschen eines Snapshots

Verwenden Sie das Management CLI um einen Snapshot zu löschen.

```
:delete-snapshot(name=20151022-133109702standalone.xml)
```

Starten des Servers mit einem Snapshot

Der Server kann mithilfe eines Snapshots oder einer automatisch gespeicherten Version der Konfiguration gestartet werden.

1. Gehen Sie zum **EAP_HOME/standalone/configuration/standalone_xml_history** Verzeichnis und identifizieren Sie den zu ladenden Snapshot oder die zu ladende Konfigurationsdatei.
2. Starten Sie den Server und weisen Sie auf die ausgewählte Konfigurationsdatei hin. Geben Sie den Dateipfad relativ zum Konfigurationsverzeichnis **EAP_HOME/standalone/configuration/** an.

```
$ EAP_HOME/bin/standalone.sh --server-
config=standalone_xml_history/snapshot/20151022-
133109702standalone.xml
```



ANMERKUNG

Wenn Sie in einer Managed Domain arbeiten, benutzen Sie stattdessen das **--host-config** Argument um die Konfigurationsdatei anzugeben.

2.3.3.5. Eigenschaften-Austausch

JBoss EAP ermöglicht Ihnen den Gebrauch von Ausdrücken an Stelle von literalen Werten um austauschbare Eigenschaften in der Konfiguration festzulegen. Ausdrücke verwenden das Format **\${PARAMETER:DEFAULT_VALUE}**. Ist der festgelegte Parameter eingestellt, so wird der Wert des Parameters verwendet. Andernfalls wird der angegebene Standardwert verwendet.

Unterstützte Quellen zum Auflösen der Ausdrücke sind Systemeigenschaften, Umgebungsvariablen und der Tresor. Nur für Deployments kann die Quelle aus Eigenschaften bestehen, die in einer **META-INF/jboss.properties** Datei im Deployment-Archiv aufgelistet sind. Für Deployment-Typen, die Sub-Deployments unterstützen, ist die Auflösung auf den Geltungsbereich aller Sub-Deployments bezogen, sofern die Eigenschaften-Datei im äußeren Deployment (z.B. EAR) ist. Ist die Eigenschaften-Datei im Sub-Deployment, so ist die Auflösung nur auf den Geltungsbereich dieses Sub-Deployments bezogen.

Das unten angeführte Beispiel aus der **standalone.xml** Konfigurationsdatei setzt die **inet-address** für die **public** Schnittstelle auf **127.0.0.1**, außer der **jboss.bind.address** Parameter ist festgelegt.

```
<interface name="public">
    <inet-address value="${jboss.bind.address:127.0.0.1}"/>
</interface>
```

Der **jboss.bind.address** Parameter kann beim Start der EAP als Standalone-Server mit folgendem Befehl festgelegt werden:

```
$ EAP_HOME/bin/standalone.sh -Djboss.bind.address=IP_ADDRESS
```

Verschachtelte Ausdrücke

Man kann Ausdrücke verschachteln, wodurch ein fortgeschrittener Gebrauch von Ausdrücken an Stelle von festen Werten möglich ist. Das Format eines verschachtelten Ausdrucks ist wie das eines normalen Ausdrucks, aber ein Ausdruck ist in einem anderen eingebettet, zum Beispiel:

```
${SYSTEM_VALUE_1}${SYSTEM_VALUE_2}}
```

Verschachtelte Ausdrücke werden rekursiv ausgewertet, damit zuerst der *innere* Ausdruck ausgewertet wird, und dann der *äußere* Ausdruck. Ausdrücke können auch rekursiv sein, wenn sich ein Ausdruck in einen anderen Ausdruck auflöst, der dann aufgelöst wird. Verschachtelte Ausdrücke sind überall zugelassen, wo Ausdrücke zugelassen sind, mit Ausnahme des Management CLI Befehls.

Ein Beispiel für eine mögliche Verwendung eines verschachtelten Ausdrucks wäre, wenn das in einer Datenquellen-Definition verwendete Passwort maskiert ist. Die Konfiguration für die Datenquelle könnte folgende Zeile haben:

```
<password>${VAULT::ds_ExampleDS::password::1}</password>
```

Der Wert von **ds_ExampleDS** könnte unter Verwendung eines verschachtelten Ausdrucks durch eine Systemeigenschaft (**datasource_name**) ersetzt werden. Die Konfiguration für die Datenquelle könnte stattdessen folgende Zeile haben:

```
<password>${VAULT::${datasource_name}::password::1}</password>
```

Die JBoss EAP würde zuerst den Ausdruck **\${datasource_name}** auswerten, dies anschließend am größeren Ausdruck eingeben und den daraus entstehenden Ausdruck auswerten. Der Vorteil dieser Konfiguration ist, dass der Name der Datenquelle von der festen Konfiguration abgewandelt ist.

Eigenschaften-Austausch auf Deskriptor-Basis

Gewöhnlich variiert die Anwendungskonfiguration (z.B. Verbindungsparameter der Datenquelle) zwischen Entwicklung, Test und Produktionsumgebung. Diese Abweichung ist manchmal in Build-System Skripts gespeichert, da die Java EE Spezifikation keine Methode zum Externalisieren dieser Konfigurationen enthält. Mit der JBoss EAP können Sie Eigenschaften-Austausch auf Deskriptor-Basis verwenden um Konfigurationen extern zu verwalten.

Eigenschaften-Austausch auf Deskriptor-Basis ersetzt Eigenschaften aufgrund von Deskriptoren, wodurch Sie Annahmen über die Umgebung aus der Anwendung und der

Build-Kette entfernen können. Umgebungs-spezifische Konfigurationen werden in den Deployment-Deskriptoren bestimmt, anstatt in Anmerkungen oder Build System Skripts. Sie können Konfigurationen in Dateien oder als Parameter an der Befehlszeile bereitstellen.

Es gibt mehrere Flags im **ee** Subsystem, die kontrollieren, ob Eigenschaften-Austausch angewendet wird.

JBoss-spezifischer Deskriptor-Austausch wird von der **jboss-descriptor-property-replacement** Flag kontrolliert und ist standardmäßig *aktiviert*. Wenn er aktiviert ist, können Eigenschaften in den folgenden Deployment-Deskriptoren ersetzt werden:

- **jboss-ejb3.xml**
- **jboss-app.xml**
- **jboss-web.xml**
- ***-jms.xml**
- ***-ds.xml**

Der folgende Management CLI Befehl kann verwendet werden, um Eigenschaften-Austausch in JBoss-spezifischen Deskriptoren zu aktivieren oder zu deaktivieren:

```
/subsystem=ee:write-attribute(name="jboss-descriptor-property-replacement",value=VALUE)
```

Java EE Deskriptor-Austausch wird vom **spec-descriptor-property-replacement** Flag kontrolliert und ist standardmäßig *deaktiviert*. Ist er aktiviert, so können Eigenschaften in den folgenden Deployment-Deskriptoren ersetzt werden:

- **ejb-jar.xml**
- **persistence.xml**
- **application.xml**
- **web.xml**

Der folgende Management CLI Befehl kann verwendet werden um Eigenschaften-Austausch in Java EE Deskriptoren zu aktivieren oder zu deaktivieren:

```
/subsystem=ee:write-attribute(name="spec-descriptor-property-replacement",value=VALUE)
```

2.4. NETZWERK- UND PORT-KONFIGURATION

2.4.1. Schnittstellen

JBoss EAP verweist in der ganzen Konfiguration auf benannte Schnittstellen. Dies ermöglicht der Konfiguration auf einzelne Schnittstellen-Deklarationen mit logischen Namen zu verweisen, anstatt bei jeder Verwendung die vollständigen Details der Schnittstelle zu erfordern.

Dies ermöglicht auch leichtere Konfiguration in einer Managed Domain, wo die Details von Netzwerkschnittstellen auf verschiedenen Rechnern variieren können. Jede Server-Instanz kann einer Gruppe von logischen Namen entsprechen.

Die Dateien **standalone.xml**, **domain.xml** und **host.xml** umfassen Schnittstellen-Deklarationen. Je nach verwendeter Standardkonfiguration gibt es mehrere vorkonfigurierte Schnittstellen-Namen. Die Schnittstelle **management** kann für alle Komponenten und Dienste verwendet werden, die die Managementebene benötigen, einschließlich des HTTP-Management-Endpunkts. Die Schnittstelle **public** kann für alle anwendungsbezogene Netzwerkkommunikation verwendet werden. Die Schnittstelle **unsecure** wird in der Standardkonfiguration für IIOP-Sockets verwendet. Die Schnittstelle **private** wird in der Standardkonfiguration für JGroups-Sockets verwendet.

2.4.1.1. Standard Schnittstellen-Konfigurationen

```
<interfaces>
  <interface name="management">
    <inet-address value="${jboss.bind.address.management:127.0.0.1}"/>
  </interface>
  <interface name="public">
    <inet-address value="${jboss.bind.address:127.0.0.1}"/>
  </interface>
  <interface name="private">
    <inet-address value="${jboss.bind.address.private:127.0.0.1}"/>
  </interface>
  <interface name="unsecure">
    <inet-address value="${jboss.bind.address.unsecure:127.0.0.1}"/>
  </interface>
</interfaces>
```

Die JBoss EAP bindet diese Schnittstellen standardmäßig an **127.0.0.1**, aber diese Werte können zur Runtime außer Kraft gesetzt werden, indem die entsprechende Eigenschaft eingestellt wird. Mit folgendem Befehl kann zum Beispiel die **inet-address** der **public** Schnittstelle beim Start der JBoss EAP als Standalone-Server festgelegt werden.

```
$ EAP_HOME/bin/standalone.sh -Djboss.bind.address=IP_ADDRESS
```

Alternativ können Sie den Switch **-b** in der Serverstart-Befehlszeile verwenden. Weitere Informationen zu Server-Startoptionen finden Sie unter [Server-Runtime-Argumente](#).



WICHTIG

Wenn Sie die Standard-Netzwerkschnittstellen oder Ports, die JBoss EAP verwendet, ändern, müssen Sie zudem Scripts ändern, die die modifizierten Schnittstellen oder Ports verwenden. Hierzu gehören JBoss EA-Dienst-Scripts, und zudem müssen Sie daran denken, die richtige Schnittstelle und den Port anzugeben, wenn Sie auf die Managementkonsole oder die Management-CLI zugreifen.

2.4.1.2. Konfigurieren von Schnittstellen

Netzwerkschnittstellen werden deklariert, indem ein logischer Name und Auswahlkriterien für die physische Schnittstelle angegeben werden. Die Auswahlkriterien können auf eine Wildcard-Adresse verweisen oder einen Satz von einem oder mehreren Merkmalen angeben, die eine Schnittstelle oder Adresse haben muss um eine gültige Übereinstimmung

darzustellen. Eine Liste aller verfügbaren Schnittstellen-Auswahlkriterien finden Sie im Abschnitt [Schnittstellenattribute](#).

Schnittstellen können mittels der Management-Konsole oder dem Management CLI konfiguriert werden. Unten finden Sie einige Beispiele für das Hinzufügen und Aktualisieren von Schnittstellen. Zuerst wird der Management CLI Befehl angezeigt, gefolgt von der entsprechenden Konfiguration XML.

Hinzufügen einer Schnittstelle mit einem NIC Wert

Fügen Sie eine neue Schnittstelle mit einem NIC Wert von **eth0** hinzu.

```
/interface=external:add(nic=eth0)
```

```
<interface name="external">
  <nic name="eth0"/>
</interface>
```

Hinzufügen einer Schnittstelle mit mehreren bedingten Werten

Neue Schnittstelle hinzufügen, die mit jeder Schnittstelle/Adresse auf dem korrekten Subnetz übereinstimmt, Multicast unterstützt und nicht Point-to-Point ist.

```
/interface=default:add(subnet-
match=192.168.0.0/16,up=true,multicast=true,not={point-to-point=true})
```

```
<interface name="default">
  <subnet-match value="192.168.0.0/16"/>
  <up/>
  <multicast/>
  <not>
    <point-to-point/>
  </not>
</interface>
```

Aktualisieren eines Schnittstellenattributs

Aktualisieren Sie den Standard **inet-address** Wert der **public** Schnittstelle unter Erhalt der **jboss.bind.address** Eigenschaft, damit dieser Wert zur Runtime eingestellt werden kann.

```
/interface=public:write-attribute(name=inet-
address,value="{jboss.bind.address:192.168.0.0}")
```

```
<interface name="public">
  <inet-address value="{jboss.bind.address:192.168.0.0}"/>
</interface>
```

Hinzufügen einer Schnittstelle zu einem Server in einer Managed Domain

```
/host=HOST_NAME/server-
config=SERVER_NAME/interface=INTERFACE_NAME:add(inet-address=127.0.0.1)
```

```
<servers>
  <server name="SERVER_NAME" group="main-server-group">
    <interfaces>
```



```

        <interface name="INTERFACE_NAME">
            <inet-address value="127.0.0.1"/>
        </interface>
    </interfaces>
</server>
</servers>

```

2.4.2. Socket-Bindungen

Socket-Bindungen und Gruppen von Socket-Bindungen ermöglichen Ihnen Netzwerk-Ports und ihre Beziehung zu den für Ihre JBoss EAP Konfiguration erforderlichen Netzwerk-Schnittstellen zu definieren. Eine Socket-Bindung ist eine benannte Socketkonfiguration. Eine Gruppe von Socket-Bindungen ist eine Sammlung von Socket-Bindung Deklarationen, die unter einem logischen Namen gruppiert sind.

Dies ermöglicht anderen Abschnitten der Konfiguration Socket-Bindungen anhand ihres logischen Namens zu referenzieren, anstatt bei jeder Verwendung die vollen Details der Socket Konfiguration zu erfordern.

Die Deklarationen für diese benannten Konfigurationen können in den **standalone.xml** und **domain.xml** Konfigurationsdateien gefunden werden. Ein Standalone-Server enthält nur eine Gruppe von Socket-Bindungen, während eine Managed Domain mehrere Gruppen enthalten kann. Sie können eine Gruppe von Socket-Bindungen für jede Servergruppe in der Managed Domain erstellen, oder eine Gruppe von Socket-Bindungen unter mehreren Servergruppen teilen.

Die von der JBoss EAP standardmäßig benutzten Ports sind abhängig davon, welche Gruppen von Socket-Bindungen benutzt werden und welche Anforderungen bei Ihren einzelnen Deployments bestehen.

2.4.2.1. Management Ports

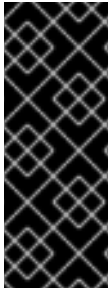
Management Ports wurden in der JBoss EAP 7 konsolidiert. Standardmäßig benutzt die JBoss EAP 7 Port **9990** sowohl für systemeigenes Management (vom Management CLI verwendet) und für HTTP Management (von Management-Konsole auf Web-Basis verwendet). Port **9999**, der als systemeigener Management Port in der JBoss EAP 6 benutzt wurde, wird nicht mehr verwendet, kann aber nach wie vor auf Wunsch aktiviert werden.

Ist HTTPS für die Management-Konsole aktiviert, so wird standardmäßig Port **9993** benutzt.

2.4.2.2. Standard Socket-Bindungen

JBoss EAP wird mit einer Gruppe von Socket-Bindungen für jedes der fünf vordefinierten Profile (*default*, *ha*, *full*, *full-ha*, *load-balancer*) geliefert.

Detaillierte Informationen zu den Socket-Bindungen, wie Standard-Ports und Beschreibungen, finden Sie im Abschnitt [Standard Socket-Bindungen](#).



WICHTIG

Wenn Sie die Standard-Netzwerkschnittstellen oder Ports, die JBoss EAP verwendet, ändern, müssen Sie zudem Scripts ändern, die die modifizierten Schnittstellen oder Ports verwenden. Hierzu gehören JBoss EA-Dienst-Scripts, und zudem müssen Sie daran denken, die richtige Schnittstelle und den Port anzugeben, wenn Sie auf die Managementkonsole oder die Management-CLI zugreifen.

Standalone-Server

Bei der Ausführung als Standalone-Server ist nur eine Gruppe von Socket-Bindungen pro Konfigurationsdatei definiert. Jede Standalone-Konfigurationsdatei (**standalone.xml**, **standalone-ha.xml**, **standalone-full.xml**, **standalone-full-ha.xml**, **standalone-load-balancer.xml**) definiert Socket-Bindungen für die Technologien, die vom entsprechenden Profil verwendet werden.

Die standardmäßige Standalone-Konfigurationsdatei (**standalone.xml**) gibt zum Beispiel die unten angeführten Socket-Bindungen an.

```
<socket-binding-group name="standard-sockets" default-interface="public"
port-offset="${jboss.socket.binding.port-offset:0}">
  <socket-binding name="management-http" interface="management"
port="${jboss.management.http.port:9990}"/>
  <socket-binding name="management-https" interface="management"
port="${jboss.management.https.port:9993}"/>
  <socket-binding name="ajp" port="${jboss.ajp.port:8009}"/>
  <socket-binding name="http" port="${jboss.http.port:8080}"/>
  <socket-binding name="https" port="${jboss.https.port:8443}"/>
  <socket-binding name="txn-recovery-environment" port="4712"/>
  <socket-binding name="txn-status-manager" port="4713"/>
  <outbound-socket-binding name="mail-smtp">
    <remote-destination host="localhost" port="25"/>
  </outbound-socket-binding>
</socket-binding-group>
```

Managed Domain

Bei der Ausführung in einer Managed Domain sind alle Gruppen von Socket-Bindungen in der **domain.xml** Datei definiert. Es gibt fünf vordefinierte Gruppen von Socket-Bindungen:

- **standard-sockets**
- **ha-sockets**
- **full-sockets**
- **full-ha-sockets**
- **load-balancer-sockets**

Jede Socket-Bindung Gruppe gibt Socket-Bindungen für die Technologien an, die vom zugehörigen Profil verwendet werden. Zum Beispiel definiert die **full-ha-sockets** Socket-Bindung Gruppe mehrere **jgroups** Socket-Bindungen, welche vom *full-ha* Profil für Hochverfügbarkeit verwendet werden.

```
<socket-binding-groups>
  <socket-binding-group name="standard-sockets" default-
```

```

interface="public">
    <!-- Needed for server groups using the 'default' profile -->
    <socket-binding name="ajp" port="${jboss.ajp.port:8009}"/>
    <socket-binding name="http" port="${jboss.http.port:8080}"/>
    <socket-binding name="https" port="${jboss.https.port:8443}"/>
    <socket-binding name="txn-recovery-environment" port="4712"/>
    <socket-binding name="txn-status-manager" port="4713"/>
    <outbound-socket-binding name="mail-smtp">
        <remote-destination host="localhost" port="25"/>
    </outbound-socket-binding>
</socket-binding-group>
<socket-binding-group name="ha-sockets" default-interface="public">
    <!-- Needed for server groups using the 'ha' profile -->
    ...
</socket-binding-group>
<socket-binding-group name="full-sockets" default-interface="public">
    <!-- Needed for server groups using the 'full' profile -->
    ...
</socket-binding-group>
<socket-binding-group name="full-ha-sockets" default-interface="public">
    <!-- Needed for server groups using the 'full-ha' profile -->
    <socket-binding name="ajp" port="${jboss.ajp.port:8009}"/>
    <socket-binding name="http" port="${jboss.http.port:8080}"/>
    <socket-binding name="https" port="${jboss.https.port:8443}"/>
    <socket-binding name="iiop" interface="unsecure" port="3528"/>
    <socket-binding name="iiop-ssl" interface="unsecure" port="3529"/>
    <socket-binding name="jgroups-mping" interface="private" port="0"
multicast-address="${jboss.default.multicast.address:230.0.0.4}"
multicast-port="45700"/>
    <socket-binding name="jgroups-tcp" interface="private" port="7600"/>
    <socket-binding name="jgroups-udp" interface="private" port="55200"
multicast-address="${jboss.default.multicast.address:230.0.0.4}"
multicast-port="45688"/>
    <socket-binding name="modcluster" port="0" multicast-
address="224.0.1.105" multicast-port="23364"/>
    <socket-binding name="txn-recovery-environment" port="4712"/>
    <socket-binding name="txn-status-manager" port="4713"/>
    <outbound-socket-binding name="mail-smtp">
        <remote-destination host="localhost" port="25"/>
    </outbound-socket-binding>
</socket-binding-group>
<socket-binding-group name="load-balancer-sockets" default-
interface="public">
    <!-- Needed for server groups using the 'load-balancer' profile -->
    ...
</socket-binding-group>
</socket-binding-groups>

```



ANMERKUNG

Die Socket Konfiguration für die Verwaltungs-Schnittstellen ist in der **host.xml** Datei des Domain-Controllers definiert.

2.4.2.3. Konfigurieren von Socket-Bindungen

Bei der Definition einer Socket-Bindung können Sie sowohl **port** und **interface** Attribute konfigurieren, als auch Multicast Einstellungen wie **multicast-address** und **multicast-port**. Details zu allen verfügbaren Socket-Bindung Attributen finden Sie im Abschnitt [Socket-Bindung Attribute](#).

Socket-Bindungen können über die Management-Konsole oder das Management CLI konfiguriert werden. Die folgenden Schritte helfen Ihnen beim Hinzufügen einer Socket-Bindungen Gruppe, beim Hinzufügen einer Socket-Bindung und beim Konfigurieren der Socket-Bindung Einstellungen über das Management CLI.

1. Hinzufügen einer neuen Gruppe von Socket-Bindungen. Beachten Sie, dass dieser Schritt bei der Ausführung als Standalone-Server nicht durchgeführt werden kann.

```
/socket-binding-group=new-sockets:add(default-interface=public)
```

2. Hinzufügen einer Socket-Bindung.

```
/socket-binding-group=new-sockets/socket-binding=new-socket-binding:add(port=1234)
```

3. Socket-Bindung ändern um eine andere als die Standard-Schnittstelle zu verwenden, welche von der Socket-Bindung Gruppe festgelegt wird.

```
/socket-binding-group=new-sockets/socket-binding=new-socket-binding:write-attribute(name=interface,value=unsecure)
```

Das folgende Beispiel zeigt, wie die XML Konfiguration aussehen könnte, nachdem die vorherigen Schritte durchgeführt wurden.

```
<socket-binding-groups>
  ...
  <socket-binding-group name="new-sockets" default-interface="public">
    <socket-binding name="new-socket-binding" interface="unsecure"
port="1234"/>
  </socket-binding-group>
</socket-binding-groups>
```

2.4.2.4. Port Offsets

Ein Port Offset ist ein numerischer Offset-Wert, der zu allen Port Werten hinzugefügt wurde, die in der Gruppe von Socket-Bindungen für diesen Server angegeben wurden. Dies ermöglicht dem Server die in seiner Gruppe von Socket-Bindungen angegebenen Port Werte zu erben, wobei ein Offset sicherstellt, dass er nicht mit einem anderen Server auf demselben Host in Konflikt steht. Wenn zum Beispiel der HTTP Port der Gruppe von Socket-Bindungen **8080** ist und ein Server ein Port Offset von **100** benutzt, dann ist sein HTTP Port **8180**.

Unten finden Sie ein Beispiel für die Einstellung eines Port Offsets von **250** für einen Server in einer Managed Domain unter Verwendung des Management CLI.

```
/host=master/server-config=server-two/:write-attribute(name=socket-binding-port-offset,value=250)
```

Port Offsets können für Server in einer Managed Domain verwendet werden, sowie für die Ausführung mehrerer Standalone-Server auf demselben Host.

Über die Eigenschaft **jboss.socket.binding.port-offset** können Sie beim Start eines Standalone-Servers einen Port-Offset übergeben.

```
$ EAP_HOME/bin/standalone.sh -Djboss.socket.binding.port-offset=100
```

2.4.3. IPv6 Adressen

JBoss EAP ist standardmäßig konfiguriert mit IPv4 Adressen zu laufen. Die Schritte unten helfen, die JBoss EAP zur Verwendung von IPv6 Adressen zu konfigurieren.

Konfigurieren von JVM-Stack für IPv6-Adressen

Aktualisieren der Startkonfiguration zur Bevorzugung von IPv6 Adressen.

1. Öffnen Sie die Start-Konfigurationsdatei.
 - Bei der Ausführung als Standalone-Server bearbeiten Sie die Datei **EAP_HOME/bin/standalone.conf** (oder **standalone.conf.bat** für Windows Server).
 - Bei der Ausführung in einer Managed Domain bearbeiten Sie die Datei **EAP_HOME/bin/domain.conf** (oder **domain.conf.bat** für Windows Server).
2. Setzen Sie die **java.net.preferIPv4Stack** Eigenschaft auf **false**.

```
-Djava.net.preferIPv4Stack=false
```

3. Fügen Sie die **java.net.preferIPv6Addresses** Eigenschaft an und setzen Sie sie auf **true**.

```
-Djava.net.preferIPv6Addresses=true
```

Das folgende Beispiel zeigt, wie die JVM-Optionen in der Start-Konfigurationsdatei nach den oben beschriebenen Änderungen aussehen könnte.

```
# Specify options to pass to the Java VM.
#
if [ "x$JAVA_OPTS" = "x" ]; then
    JAVA_OPTS="-Xms1303m -Xmx1303m -Djava.net.preferIPv4Stack=false"
    JAVA_OPTS="$JAVA_OPTS -
Djboss.modules.system.pkgs=$JBOSS_MODULES_SYSTEM_PKGS -
Djava.awt.headless=true"
    JAVA_OPTS="$JAVA_OPTS -Djava.net.preferIPv6Addresses=true"
else
```

Aktualisieren der Schnittstellen-Deklarationen für IPv6 Adressen

Die standardmäßigen Schnittstellen-Werte in der Konfiguration können auf IPv6 Adressen geändert werden. Der Management CLI Befehl unten setzt zum Beispiel die **management** Schnittstelle auf IPv6 Loopback-Adressen (::1).

```
/interface=management:write-attribute(name=inet-
address,value="{jboss.bind.address.management:[::1]}")
```

Das folgende Beispiel zeigt, wie die XML Konfiguration nach Ausführung des Befehls oben aussehen könnte.

```
<interfaces>
  <interface name="management">
    <inet-address value="{jboss.bind.address.management:[::1]}"/>
  </interface>
  ....
</interfaces>
```

KAPITEL 3. ENTWICKLUNG VON ANWENDUNGEN, DIE DIE JBOSS EAP VERWENDEN

3.1. ÜBERBLICK

Dieses Handbuch gibt Ihnen Information zum Einstieg in die Anwendungsentwicklung mithilfe des Red Hat JBoss Developer Studio und JBoss EAP 7 Quickstart-Beispielen.

Das Red Hat JBoss Developer Studio ist eine auf Eclipse basierende, integrierte Entwicklungsumgebung (Integrated Development Environment, IDE), die Plugins der JBoss Anwendungsentwicklung integriert. Das JBoss Developer Studio kann Ihnen bei der Entwicklung Ihrer Anwendung durch die Verfügbarkeit von JBoss-spezifischen Assistenten und durch die Fähigkeit, Anwendungen auf JBoss Servern bereitzustellen, helfen. Viele Quickstart-Codebeispiele werden mit der JBoss EAP 7 bereitgestellt um den Benutzern den Einstieg beim Schreiben von Anwendungen mittels verschiedener Java EE 7 Technologien zu erleichtern.

3.2. EINRICHTEN DER ENTWICKLUNGSUMGEBUNG

Es wird empfohlen, JBoss Developer Studio 11.0 oder höher für JBoss EAP 7.1 zu verwenden.

1. Laden Sie JBoss Developer Studio herunter und installieren sie es.
Anweisungen finden Sie unter [Installation des JBoss Developer Studio Stand-Alone über den Installer](#) im JBoss Developer Studio *Installationshandbuch*.
2. Richten Sie den JBoss EAP-Server in JBoss Developer Studio ein.
Anweisungen finden Sie unter [Verwendung der Runtime Detection zur Einrichtung von JBoss EAP über die IDE](#) im Leitfaden *Erste Schritte mit JBoss Developer Studio Tools guide*.

3.3. VERWENDUNG VON QUICKSTART-BEISPIELEN

Die Quickstart-Beispiele für JBoss EAP sind Maven-Projekte.

3.3.1. Über Maven

Apache Maven ist ein verteiltes Build-Automatisierungstool, das bei der Entwicklung von Java-Anwendungen für Erstellung, Verwaltung und Build von Software-Projekten verwendet wird. Maven verwendet Standard-Konfigurationsdateien namens Project Object Model (POM), die den Build-Prozess definieren und verwalten. POMs beschreiben die Modul- und Komponenten-Abhängigkeiten, die Build-Reihenfolge und die Ziele für resultierendes Projekt-Packaging und Ausgabe unter Verwendung einer XML-Datei. Dies stellt den korrekten und einheitlichen Build des Projekts sicher.

Maven erreicht dies unter Verwendung eines Repositorys. Ein Maven Repository speichert Java-Bibliotheken, Plugins und andere Build-Artefakte. Das standardmäßige öffentliche Repository ist das [Maven 2 Central Repository](#), aber Repositorys können innerhalb eines Unternehmens privat und intern sein, mit dem Ziel gemeinsame Artefakte unter den Entwicklungsteams zu teilen. Repositorys sind auch von Drittanbietern erhältlich. Weitere Informationen finden Sie im [Apache Maven](#) Projekt und im Handbuch [Einführung in Repositorys](#).

Die JBoss EAP beinhaltet ein Maven-Repository, das viele Anforderungen enthält, die Java EE Entwickler in der Regel für den Build von Anwendungen auf der JBoss EAP verwenden.

Weitere Informationen zur Verwendung von Maven mit JBoss EAP finden Sie unter [Verwendung von Maven mit JBoss EAP](#) im JBoss EAP *Entwickungsleitfaden*.

3.3.2. Verwendung von Maven mit Quickstarts

Die für Build und Deployment von Anwendungen auf der JBoss EAP 7 erforderlichen Artefakte und Abhängigkeiten werden auf einem öffentlichen Repository gehostet. Beim Start mit JBoss EAP 7 Quickstarts ist es nicht mehr nötig Ihre Maven **settings.xml** Datei zur Verwendung dieser Repositories beim Build von Quickstarts zu konfigurieren. Die Maven Repositories sind jetzt in den POM Dateien des Quickstart-Projekts konfiguriert. Diese Konfigurationsmethode soll den Einstieg in die Quickstarts erleichtern, wird jedoch im Allgemeinen nicht für Produktionsprojekte empfohlen, da sie Ihren Build verlangsamen kann.

Red Hat JBoss Developer Studio umfasst Maven, dies muss also nicht eigens heruntergeladen und installiert werden. Die Verwendung von JBoss Developer Studio Version 11.0 oder höher wird empfohlen.

Falls Sie die Maven-Befehlszeile für Build und Deployment Ihrer Anwendungen verwenden wollen, müssen Sie zuerst Maven vom [Apache Maven](#) Projekt entsprechend der Anleitung in der Maven-Dokumentation herunterladen und installieren.

3.3.3. Download und Ausführung der Quickstarts

3.3.3.1. Download der Quickstarts

Die JBoss EAP kommt mit einem umfassenden Satz an Quickstart-Codebeispielen, die den Benutzern beim Schreiben von Anwendungen mittels verschiedener Java EE 7 Technologien helfen sollen. Die Quickstarts können vom Red Hat Kundenportal heruntergeladen werden.

1. Melden Sie sich beim [Red Hat Kundenportal](#) an.
2. Klicken Sie auf **Downloads**.
3. Klicken Sie in der **Produkt Downloads** Liste auf **Red Hat JBoss Enterprise Application Platform**.
4. Wählen Sie **7.1** in der Auswahlliste **Version** aus.
5. Suchen Sie den Eintrag **Red Hat JBoss Enterprise Application Platform 7.1.0 Quickstarts** in der Tabelle und klicken Sie auf **Download**.
6. Speichern Sie die Zip-Datei im gewünschten Verzeichnis.
7. Entpacken Sie die Zip-Datei.

3.3.3.2. Ausführen der Quickstarts im JBoss Developer Studio

Sobald die Quickstarts heruntergeladen sind, können Sie ins JBoss Developer Studio importiert und in der JBoss EAP bereitgestellt werden.

Importieren der Quickstarts ins JBoss Developer Studio

Jeder Quickstart wird mit einer POM Datei geliefert, die seine Projekt- und Konfigurationsinformationen enthält. Mit dieser POM Datei können Sie den Quickstart einfach ins JBoss Developer Studio importieren.

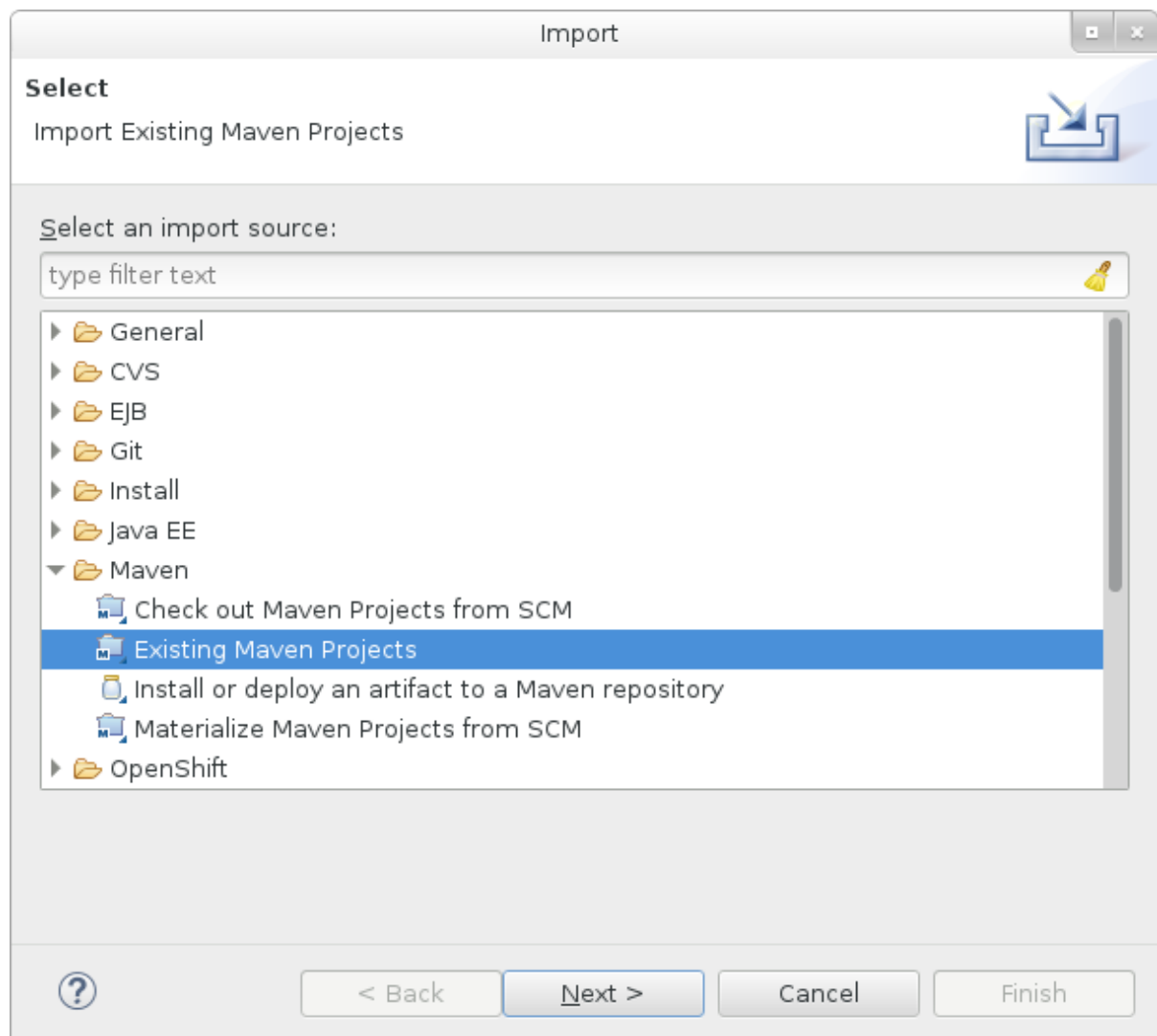


WICHTIG

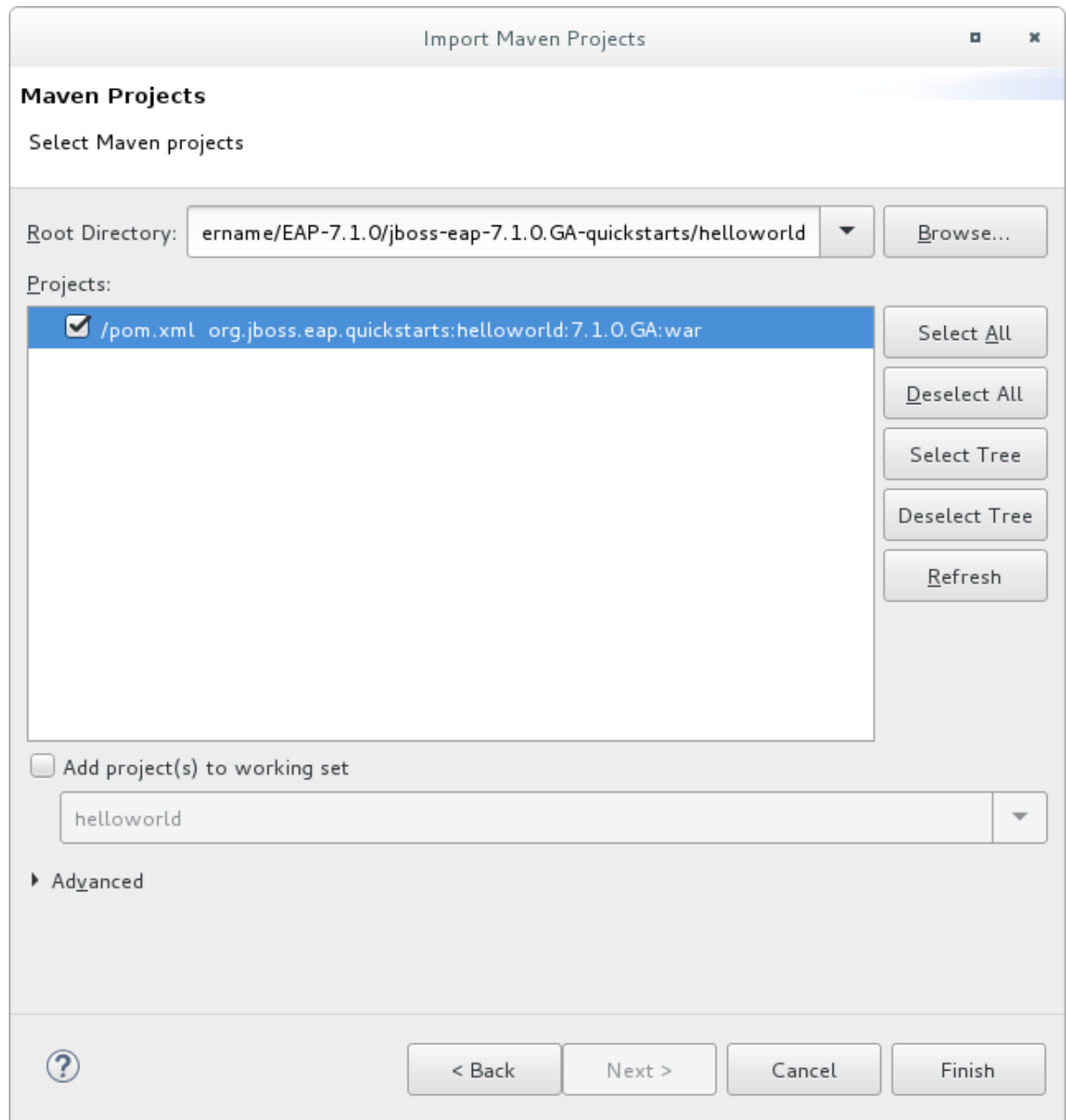
Befindet sich Ihr Quickstart-Projektordner beim Import ins JBoss Developer Studio innerhalb der IDE-Arbeitsfläche, so generiert IDE einen ungültigen Projektnamen und WAR-Archivnamen. Stellen Sie sicher, dass sich Ihr Quickstart-Projektordner außerhalb der IDE-Arbeitsfläche befindet, ehe Sie anfangen!

1. Starten Sie das JBoss Developer Studio.
2. Wählen Sie **Datei** → **Import**.
3. Wählen Sie **Maven** → **Bestehende Maven Projekte** und klicken Sie dann auf **Weiter**.

Abbildung 3.1. Import bestehender Maven Projekte



4. Gehen Sie zum gewünschten Quickstart-Verzeichnis (zum Beispiel **helloworld** Quickstart) und klicken Sie auf **OK**. Das **Projekte** Listenfeld wird mit der **pom.xml** Datei des gewählten Quickstart-Projekts bestückt.

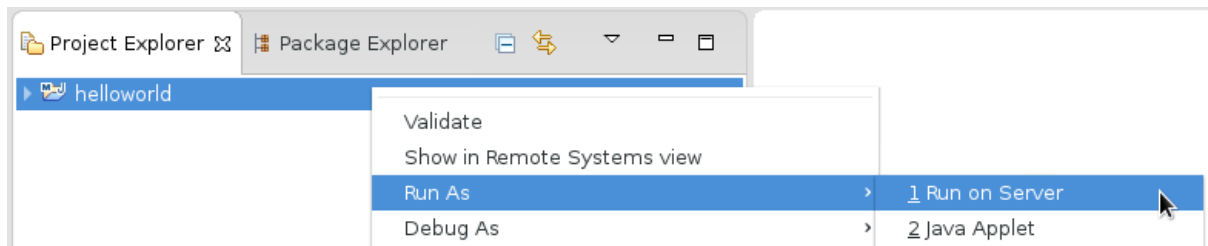
Abbildung 3.2. Auswahl von Maven-Projekten

5. Klicken Sie auf **Fertigstellen**.

Führen Sie den *helloworld* Quickstart aus

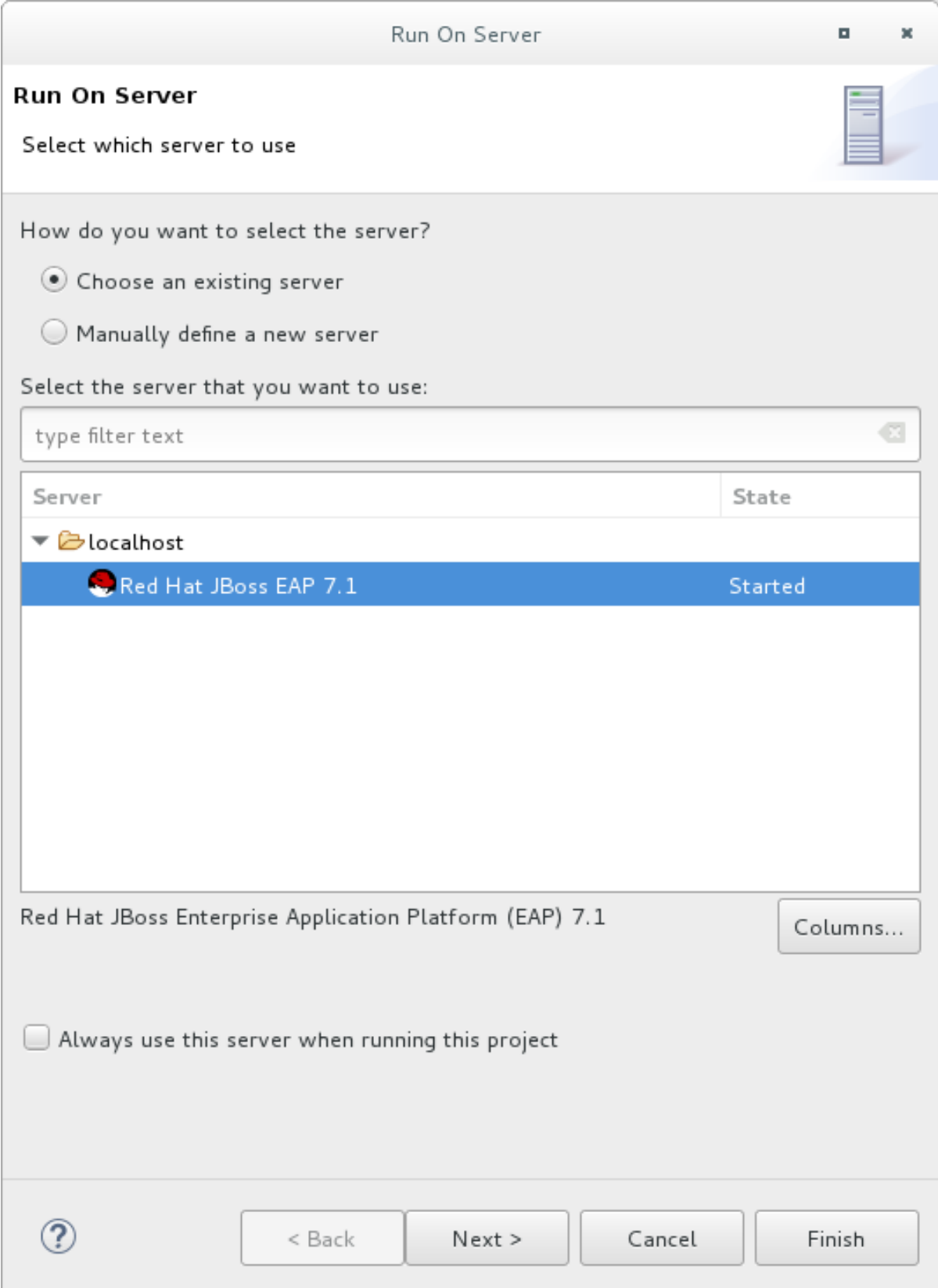
Durch die Ausführung des **helloworld** Quickstarts kann man ganz einfach prüfen, ob der JBoss EAP Server konfiguriert ist und ordnungsgemäß läuft.

1. Wenn Sie noch keinen Server definiert haben, fügen Sie den JBoss EAP-Server dem JBoss Developer Studio hinzu. Siehe [Verwendung der Runtime Detection zur Einrichtung von JBoss EAP über die IDE](#) im Leitfaden *Erste Schritte mit JBoss Developer Studio Tools*.
2. Klicken Sie rechts auf das Projekt **helloworld** in der Registerkarte **Project Explorer** und wählen Sie **Ausführen als** → **Ausführung auf Server** aus.

Abbildung 3.3. Ausführen als - Ausführen auf Server

3. Wählen Sie den JBoss EAP 7.1 Server aus der Serverliste aus und klicken Sie auf **Weiter**.

Abbildung 3.4. Ausführen auf Server



Run On Server

Select which server to use

How do you want to select the server?

☒ Choose an existing server

☐ Manually define a new server

Select the server that you want to use:

type filter text

Server	State
▼ localhost	
Red Hat JBoss EAP 7.1	Started

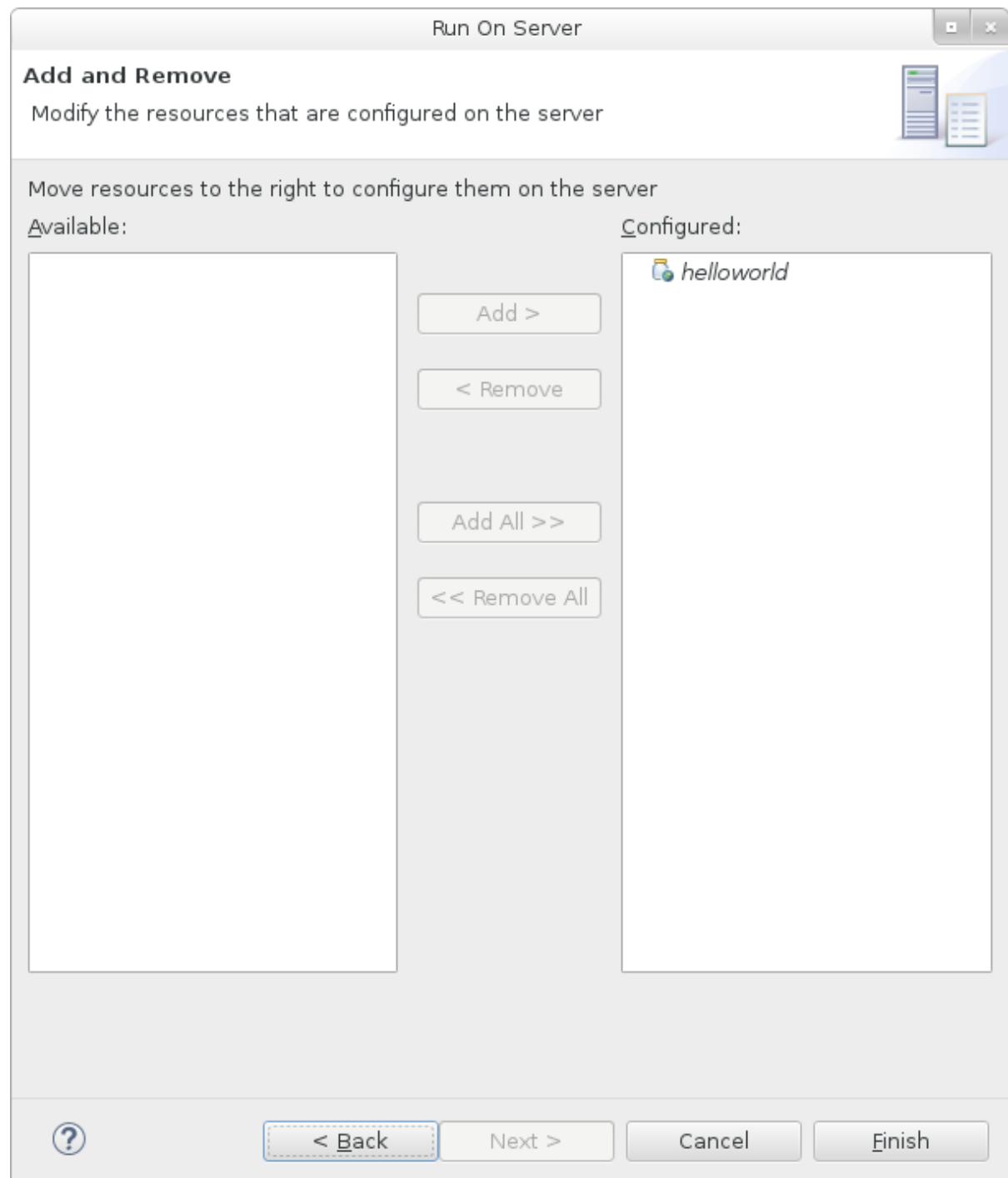
Red Hat JBoss Enterprise Application Platform (EAP) 7.1

Columns...

☐ Always use this server when running this project

? < Back Next > Cancel Finish

4. Der **helloworld** Quickstart ist bereits zur Konfiguration auf dem Server aufgelistet. Klicken Sie auf **Fertigstellen** um den Quickstart bereitzustellen.

Abbildung 3.5. Bearbeiten von auf dem Server konfigurierten Ressourcen

5. Überprüfen Sie die Ergebnisse.

- In der Registerkarte **Server** ändert sich der JBoss EAP 7.1 Serverstatus auf **Started**.
- Der **Konsole** Reiter zeigt Nachrichten zum JBoss EAP Server-Start und zum Deployment des **helloworld** Quickstarts.

```
WFLYUT0021: Registered web context: /helloworld
WFLYSRV0010: Deployed "helloworld.war" (runtime-name :
"helloworld.war")
```

- Die Anwendung **helloworld** finden Sie unter <http://localhost:8080/helloworld>, sie zeigt den Text **Hello World!** an.

Weitere Informationen zum **helloworld** Quickstart finden Sie unter [Den helloworld Quickstart erkunden](#).

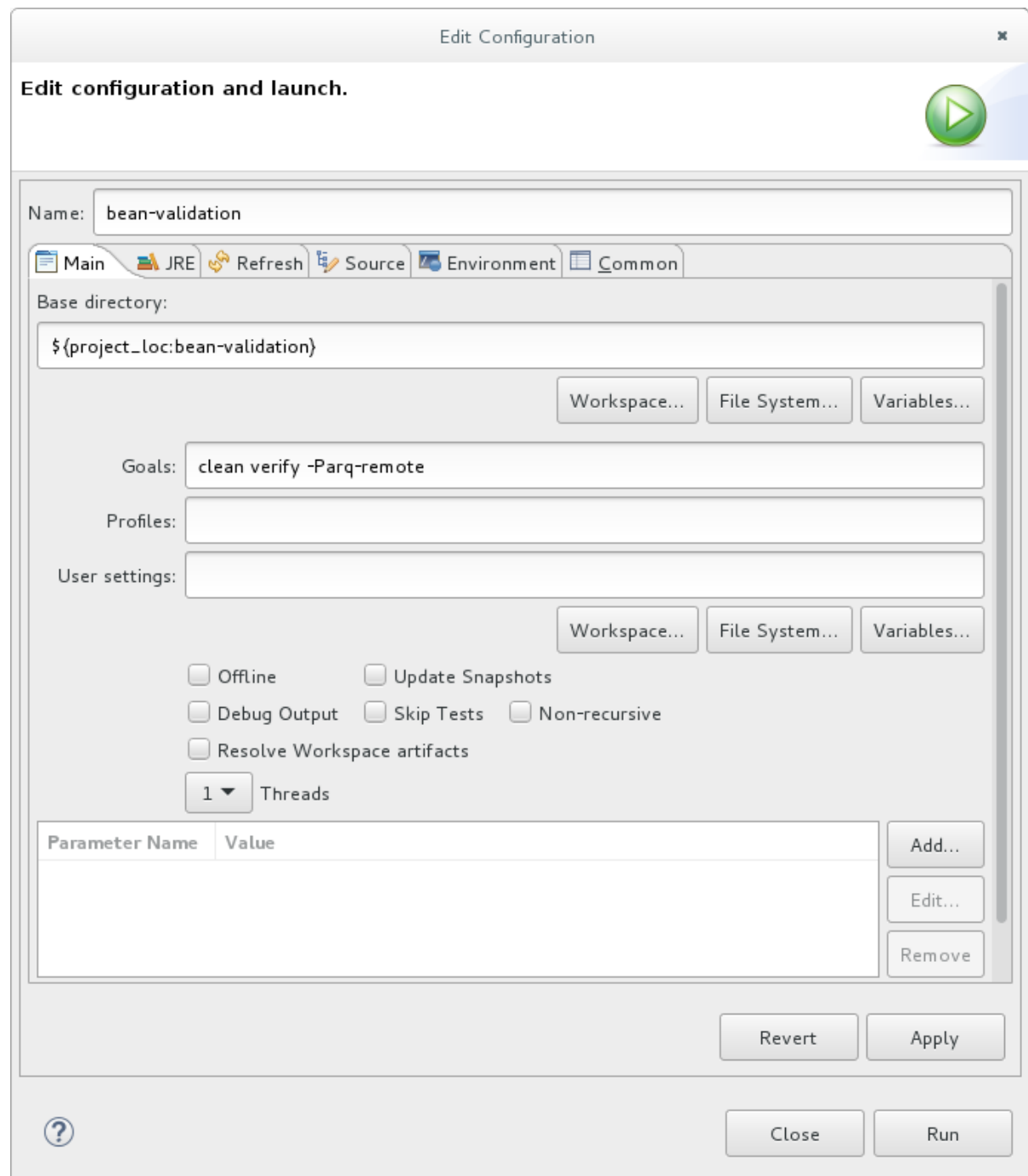
Führen Sie den *bean-validation* Quickstart aus

Einige Quickstarts, wie der **bean-validation** Quickstart, bieten keine Benutzerschnittstellen-Schicht und stellen stattdessen Arquillian-Tests bereit um ihre Funktionalität zu demonstrieren.

1. Importieren Sie den **bean-validation** Quickstart ins JBoss Developer Studio.
2. Klicken Sie auf der Registerkarte **Servers** rechts auf den Server und wählen Sie **Start** aus, um den JBoss EAP-Server zu starten. Wenn Sie keine Registerkarte **Servers** sehen oder noch keinen Server definiert haben, fügen Sie den JBoss EAP-Server dem JBoss Developer Studio hinzu. Siehe [Verwendung der Runtime Detection zur Einrichtung von JBoss EAP über die IDE](#) im Leitfaden *Erste Schritte mit JBoss Developer Studio Tools*.
3. Klicken Sie rechts auf das Projekt **bean-validation** auf der Registerkarte **Project Explorer** und wählen Sie **Ausführen als** → **Maven Build** aus.
4. Geben Sie Folgendes in das **Ziele** Eingabe-Feld ein und klicken Sie dann auf **Ausführen**.

```
clean verify -Parq-remote
```

Abbildung 3.6. Konfiguration bearbeiten



5. Überprüfen Sie die Ergebnisse.

Der **Konsole** Reiter zeigt die Ergebnisse des **bean-validation** Arquillian-Tests:

```

-----
T E S T S
-----
Running
org.jboss.as.quickstarts.bean_validation.test.MemberValidationTest
Tests run: 5, Failures: 0, Errors: 0, Skipped: 0, Time elapsed:
2.189 sec

Results :

Tests run: 5, Failures: 0, Errors: 0, Skipped: 0

```

```
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
```

3.3.3.3. Ausführen der Quickstarts von der Befehlszeile aus

Build und Bereitstellung von Quickstarts von der Befehlszeile aus sind mit Maven einfach. Wenn Sie Maven noch nicht installiert haben, gehen Sie zum [Apache Maven](#) Projekt um es herunterzuladen und zu installieren.

Eine **README.md** Datei im root Verzeichnis des Quickstarts enthält allgemeine Informationen über Systemanforderungen, über die Konfiguration von Maven, über das Hinzufügen von Benutzern und über die Ausführung von Quickstarts.

Jeder Quickstart enthält zudem seine eigene **README.md** Datei, die spezifische Anweisungen und Maven-Befehle zur Ausführung des Quickstarts bereitstellt.

Führen Sie den *helloworld* Quickstart von der Befehlszeile aus

1. Überprüfen Sie die **README.md** Datei im root-Verzeichnis des *helloworld* Quickstarts.
2. Starten Sie den JBoss EAP-Server.

```
$ EAP_HOME/bin/standalone.sh
```

3. Gehen Sie zum *helloworld* Quickstart-Verzeichnis.
4. Build und Deployment des Quickstarts über den in der **README.md** Datei angegebenen Maven-Befehl.

```
$ mvn clean install wildfly:deploy
```

5. Die Anwendung *helloworld* ist nun unter <http://localhost:8080/helloworld> verfügbar und zeigt den Text **Hello World!** an.

3.4. GEHEN SIE DIE QUICKSTART-BEISPIELE DURCH

3.4.1. Erkunden Sie den Quickstart *helloworld*

Der **helloworld** Quickstart demonstriert, wie sich ein einfaches Servlet auf JBoss EAP bereitstellen lässt. Die Geschäftslogik ist in einem Dienst verkapselt, der als Contexts and Dependency Injection (CDI) Bean bereitgestellt und in ein Servlet eingesetzt wird. Dieser Quickstart ist der Startpunkt, über den Sie sicherstellen können, dass Sie den Server korrekt konfiguriert und gestartet haben.

Detaillierte Anweisungen zum Erstellen und Bereitstellen dieses Quickstarts über die Befehlszeile finden Sie in der Datei **README.html** am Ursprung des Quickstart-Verzeichnisses **helloworld**. Dieses Thema zeigt Ihnen, wie Sie das Red Hat JBoss Developer Studio zur Verwendung des Quickstarts nutzen können und geht davon aus, dass Sie das Red Hat JBoss Developer Studio installiert haben, Maven konfiguriert haben und den Quickstart **helloworld** importiert und erfolgreich ausgeführt haben.

Voraussetzungen

- Installieren Sie JBoss Developer Studio. Anweisungen finden Sie unter [Installation des JBoss Developer Studio Stand-Alone über den Installer](#) im JBoss Developer Studio *Installationshandbuch*.
- Führen Sie den Quickstart **helloworld** aus. Anweisungen finden Sie unter [Ausführen der Quickstarts im JBoss Developer Studio](#).
- Prüfen Sie, dass der Quickstart **helloworld** erfolgreich in JBoss EAP bereitgestellt wurde, indem Sie den Webbrowser öffnen und über <http://localhost:8080/helloworld> auf die Anwendung zugreifen.

Untersuchen Sie die Verzeichnisstruktur

Den Code für den **helloworld** Quickstart finden Sie im Verzeichnis **QUICKSTART_HOME/helloworld/**. Der **helloworld** Quickstart besteht aus einem Servlet und einer CDI-Bean. Er umfasst zudem die Datei **beans.xml** im Verzeichnis **WEB-INF/** der Anwendung mit Versionsnummer 1.1 und einen **bean-discovery-mode all**. Diese Marker-Datei identifiziert den WAR als Bean-Archiv und teilt JBoss EAP mit, in dieser Anwendung nach Beans zu suchen und die CDI zu aktivieren.

Das Verzeichnis **src/main/webapp/** enthält die Dateien für den Quickstart. Alle Konfigurationsdateien für dieses Beispiel befinden sich im Verzeichnis **WEB-INF/** in **src/main/webapp/**, einschließlich der Datei **beans.xml**. Das Verzeichnis **src/main/webapp/** umfasst zudem eine Datei **index.html**, die eine einfache Meta-Aktualisierung verwendet, um den Browser des Benutzers zum Servlet weiterzuleiten, das sich unter <http://localhost:8080/helloworld/HelloWorld> befindet. Für den Quickstart ist keine **web.xml** Datei erforderlich.

Untersuchen Sie den Code

Die Paket-Deklaration und -Importe sind aus diesen Auflistungen ausgeschlossen. Die vollständige Auflistung finden Sie im Quellcode des Quickstarts.

1. Gehen Sie den **HelloWorldServlet** Code durch.
Die Datei **HelloWorldServlet.java** befindet sich im Verzeichnis **src/main/java/org/jboss/as/quickstarts/helloworld/**. Dieses Servlet sendet die Informationen zum Browser.

Beispiel: HelloWorldServlet Klassencode

```

42 @SuppressWarnings("serial")
43 @WebServlet("/HelloWorld")
44 public class HelloWorldServlet extends HttpServlet {
45
46     static String PAGE_HEADER = "<html><head>
<title>helloworld</title></head><body>";
47
48     static String PAGE_FOOTER = "</body></html>";
49
50     @Inject
51     HelloService helloService;
52
53     @Override
54     protected void doGet(HttpServletRequest req,
55     HttpServletResponse resp) throws ServletException, IOException {
56         resp.setContentType("text/html");
57         PrintWriter writer = resp.getWriter();

```

```

57         writer.println(PAGE_HEADER);
58         writer.println("<h1>" +
helloService.createHelloMessage("World") + "</h1>");
59         writer.println(PAGE_FOOTER);
60         writer.close();
61     }
62
63 }

```

Tabelle 3.1. HelloWorldServlet Details

Zeile	Hinweis
43	Sie müssen nur die Annotation @WebServlet hinzufügen und eine Zuordnung zu einer URL, über die auf das Servlet zugegriffen wird, bereitstellen.
46-48	Jede Webseite benötigt korrekt geformtes HTML. Dieser Quickstart verwendet statische Strings, um den Mindest-Header und Footer-Output zu schreiben.
50-51	Diese Zeilen fügen die HelloService CDI-Bean ein, die die tatsächliche Nachricht erstellt. Wird die API von HelloService nicht geändert, können wir so die Implementierung von HelloService später ändern, ohne die Ansichtsebene zu ändern.
58	Diese Zeile ruft den Dienst ab, um die Nachricht "Hello World" zu erstellen und sie über die HTTP-Anfrage auszugeben.

2. Gehen Sie den **HelloService** Code durch.

Die Datei **HelloService.java** befindet sich im Verzeichnis **src/main/java/org/jboss/as/quickstarts/helloworld/**. Dieser Dienst gibt einfach eine Nachricht aus. Es ist kein XML und keine Annotations-Registrierung erforderlich.

Beispiel: HelloService Klassencode

```

public class HelloService {

    String createHelloMessage(String name) {
        return "Hello " + name + "!";
    }

}

```

3.4.2. Erkunden Sie den numberguess-Quickstart

Der **numberguess** Quickstart zeigt Ihnen, wie Sie eine einfache, nicht persistente Anwendung erstellen und in JBoss EAP bereitstellen können. Informationen werden über eine JSF-Ansicht angezeigt und die Geschäftslogik in zwei CDI-Beans eingekapselt. Im **numberguess** Quickstart haben Sie zehn Versuche, eine Zahl zwischen 1 und 100 zu erraten. Nach jedem Versuch wird Ihnen mitgeteilt, ob Sie zu hoch oder zu niedrig lagen.

Den Code für den Quickstart **numberguess** finden Sie im Verzeichnis **QUICKSTART_HOME/numberguess/**, wobei **QUICKSTART_HOME** das Verzeichnis ist, in das Sie die JBoss EAP Quickstarts heruntergeladen und in das Sie diese entpackt haben. Der **numberguess** Quickstart besteht aus Beans, Konfigurationsdateien und Facelet-Ansichten (JSF) und ist als WAR-Modul gepackt.

Detaillierte Anweisungen zum Erstellen und Bereitstellen dieses Quickstarts über die Befehlszeile finden Sie in der Datei **README.html** am Ursprung des Quickstart-Verzeichnisses **helloworld**. Die folgenden Beispiele nutzen Red Hat JBoss Developer Studio, um den Quickstart auszuführen.

Voraussetzungen

- Installieren Sie JBoss Developer Studio. Anweisungen finden Sie unter [Installation des JBoss Developer Studio Stand-Alone über den Installer](#) im JBoss Developer Studio *Installationshandbuch*.
- Führen Sie den Quickstart **numberguess** aus. Anweisungen finden Sie unter [Ausführen der Quickstarts im JBoss Developer Studio](#); ersetzen Sie in der Anleitung **helloworld** durch **numberguess**.
- Prüfen Sie, dass der Quickstart **numberguess** erfolgreich in JBoss EAP bereitgestellt wurde, indem Sie den Webbrowser öffnen und über <http://localhost:8080/numberguess> auf die Anwendung zugreifen.

Untersuchen Sie die Konfigurationsdateien

Alle Konfigurationsdateien für dieses Beispiel finden Sie im Verzeichnis **QUICKSTART_HOME/numberguess/src/main/webapp/WEB-INF/** des Quickstarts.

1. Untersuchen Sie die Datei **faces-config.xml**.
Dieser Quickstart verwendet Version JSF 2.2 des Dateinamens **faces-config.xml**. Eine standardisierte Version von Facelets ist der Standard-Ansichtshandler in JSF 2.2 und benötigt daher keine Konfiguration. Diese besteht nur aus dem Root-Element und ist eine einfache Marker-Datei, die darauf hinweist, dass JSF in der Anwendung aktiviert sein sollte.

```
<faces-config version="2.2"
  xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://xmlns.jcp.org/xml/ns/javaee
    http://xmlns.jcp.org/xml/ns/javaee/web-facesconfig_2_2.xsd">

</faces-config>
```

2. Gehen Sie die Datei **beans.xml** durch.
Die Datei **beans.xml** umfasst die Versionsnummer 1.1 und den **bean-discovery-mode all**. Diese Datei ist eine Marker-Datei, die WAR als Bean-Archiv identifiziert und JBoss EAP mitteilt, in dieser Anwendung nach Beans zu suchen und die CDI zu aktivieren.

```
<beans xmlns="http://xmlns.jcp.org/xml/ns/javaee"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="
    http://xmlns.jcp.org/xml/ns/javaee
```

```

    http://xmlns.jcp.org/xml/ns/javaee/beans_1_1.xsd"
    bean-discovery-mode="all">
</beans>

```



ANMERKUNG

Dieser Quickstart benötigt keine Datei `web.xml`.

3.4.2.1. Gehen Sie den JSF-Code durch

JSF verwendet die Dateierweiterung `.xhtml` für Quelldateien, liefert die gerenderten Ansichten jedoch über die Erweiterung `.jsf`. Die Datei `home.xhtml` befindet sich im Verzeichnis `src/main/webapp/`.

Beispiel: JSF Quellcode

```

19<html xmlns="http://www.w3.org/1999/xhtml"
20 xmlns:ui="http://java.sun.com/jsf/facelets"
21 xmlns:h="http://java.sun.com/jsf/html"
22 xmlns:f="http://java.sun.com/jsf/core">
23
24 <head>
25 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"
26 />
27 <title>Numberguess</title>
28
29 <body>
30 <div id="content">
31 <h1>Guess a number...</h1>
32 <h:form id="numberGuess">
33
34 <!-- Feedback for the user on their guess -->
35 <div style="color: red">
36 <h:messages id="messages" globalOnly="false" />
37 <h:outputText id="Higher" value="Higher!"
38     rendered="#{game.number gt game.guess and game.guess ne 0}" />
39 <h:outputText id="Lower" value="Lower!"
40     rendered="#{game.number lt game.guess and game.guess ne 0}" />
41 </div>
42
43 <!-- Instructions for the user -->
44 <div>
45 I'm thinking of a number between <span
46 id="numberGuess:smallest">#{game.smallest}</span> and <span
47 id="numberGuess:biggest">#{game.biggest}</span>. You have
48 #{game.remainingGuesses} guesses remaining.
49 </div>
50
51 <!-- Input box for the users guess, plus a button to submit, and reset
52 -->
53 <!-- These are bound using EL to our CDI beans -->
54 <div>
55 Your guess:
56 <h:inputText id="inputGuess" value="#{game.guess}"

```

```

56 required="true" size="3"
57 disabled="#{game.number eq game.guess}"
58 validator="#{game.validateNumberRange}" />
59 <h:commandButton id="guessButton" value="Guess"
60   action="#{game.check}"
61   disabled="#{game.number eq game.guess}" />
62 </div>
63 <div>
64 <h:commandButton id="restartButton" value="Reset"
65 action="#{game.reset}" immediate="true" />
66 </div>
67 </h:form>
68
69 </div>
70
71 <br style="clear: both" />
72
73 </body>
74</html>

```

Die folgenden Zeilennummern entsprechen denen, die bei Ansicht der Datei in JBoss Developer Studio angezeigt werden.

Tabelle 3.2. JSF Details

Zeile	Hinweis
36-40	Dies sind die Nachrichten, die dem User gesendet werden können: "Higher!" und "Lower!"
45-48	Wenn der Benutzer rät, verringert sich der Bereich der Zahlen, die geraten werden können. Der Satz ändert sich, um sicherzustellen, dass der Benutzer den Zahlenbereich eines gültigen Rateversuchs kennt.
55-58	Dieses Eingabefeld ist mit einer Bean-Eigenschaft verbunden, die einen Wertausdruck verwendet.
58	Eine Validator-Bindung wird verwendet, um sicherzustellen, dass der Benutzer nicht versehentlich eine Zahl außerhalb des gültigen Ratebereichs eingibt. Ohne diesen Validator könnte der Benutzer einen Rateversuch für eine ungültige Zahl verbrauchen.
59-61	Der Benutzer benötigt einen Weg, den Rateversuch zum Server zu senden. Hier binden wir die Aktionsmethode an die Bean.

3.4.2.2. Gehen Sie die Klassendateien durch

Alle **numberguess** Quickstart-Quelldateien finden Sie im Verzeichnis **QUICKSTART_HOME/numberguess/src/main/java/org/jboss/as/quickstarts/numberguess/**. Die Paket-Deklaration und -Importe sind aus diesen Auflistungen ausgeschlossen. Die vollständige Auflistung finden Sie im Quellcode des Quickstarts.

1. Gehen Sie den **Random.java** Qualifikator-Code durch

Ein Qualifikator wird verwendet, um Unklarheiten zwischen den zwei Beans zu vermeiden, die beide für eine Injektion auf Grundlage ihres Typs in Frage kommen. Weitere Informationen zu Qualifikatoren finden Sie unter [Einen Qualifikator verwenden, um eine unklare Injektion zu lösen](#) im JBoss EAP *Entwicklerleitfaden*. Der Qualifikator **@Random** wird zur Injektion einer Zufallszahl verwendet.

```
@Target({ TYPE, METHOD, PARAMETER, FIELD })
@Retention(RUNTIME)
@Documented
@Qualifier
public @interface Random {

}
```

2. Gehen Sie den **MaxNumber.java** Qualifikator-Code durch
Der **@MaxNumber Qualifikator** wird zur Injektion der erlaubten Höchstzahl verwendet.

```
@Target({ TYPE, METHOD, PARAMETER, FIELD })
@Retention(RUNTIME)
@Documented
@Qualifier
public @interface MaxNumber {

}
```

3. Gehen Sie den **Generator.java** Code durch
Die Klasse **Generator** erstellt die Zufallszahl über eine Producer-Methode und stellt die mögliche Höchstzahl über dieselbe Methode zur Verfügung. Diese Klasse ist auf Anwendungsebene ausgelegt, damit nicht jedes Mal eine andere Zufallszahl erstellt wird.

```
@SuppressWarnings("serial")
@ApplicationScoped
public class Generator implements Serializable {

    private java.util.Random random = new
    java.util.Random(System.currentTimeMillis());

    private int maxNumber = 100;

    java.util.Random getRandom() {
        return random;
    }

    @Produces
    @Random
    int next() {
        // a number between 1 and 100
        return getRandom().nextInt(maxNumber - 1) + 1;
    }

    @Produces
    @MaxNumber
    int getMaxNumber() {
```

```

        return maxNumber;
    }
}

```

4. Gehen Sie den **Game.java** Code durch

Die auf Sitzungsebene ausgelegte Klasse **Game** ist der primäre Zugangspunkt für die Anwendung. Sie ist dafür verantwortlich, das Spiel einzurichten oder zurückzusetzen, den Rateversuch des Benutzers zu erfassen und zu validieren und dem Benutzer über eine **FacesMessage** Feedback zu geben. Sie verwendet eine Post-Construct-Lifecycle-Methode, um das Spiel zu initialisieren, indem eine Zufallszahl aus der Bean **@Random Instance<Integer>** abgerufen wird.

Beachten Sie die Annotation **@Named** in der Klasse. Diese Annotation ist nur erforderlich, wenn die Bean über Expression Language (EL), in diesem Fall **#{game}**, für eine JSF-Ansicht verfügbar gemacht werden soll.

```

@SuppressWarnings("serial")
@Named
@SessionScoped
public class Game implements Serializable {

    /**
     * The number that the user needs to guess
     */
    private int number;

    /**
     * The users latest guess
     */
    private int guess;

    /**
     * The smallest number guessed so far (so we can track the valid
     guess range).
     */
    private int smallest;

    /**
     * The largest number guessed so far
     */
    private int biggest;

    /**
     * The number of guesses remaining
     */
    private int remainingGuesses;

    /**
     * The maximum number we should ask them to guess
     */
    @Inject
    @MaxNumber
    private int maxNumber;

    /**
     * The random number to guess

```

```

    */
    @Inject
    @Random
    Instance<Integer> randomNumber;

    public Game() {
    }

    public int getNumber() {
        return number;
    }

    public int getGuess() {
        return guess;
    }

    public void setGuess(int guess) {
        this.guess = guess;
    }

    public int getSmallest() {
        return smallest;
    }

    public int getBiggest() {
        return biggest;
    }

    public int getRemainingGuesses() {
        return remainingGuesses;
    }

    /**
     * Check whether the current guess is correct, and update the
     * biggest/smallest guesses as needed. Give feedback to the user
     * if they are correct.
     */
    public void check() {
        if (guess > number) {
            biggest = guess - 1;
        } else if (guess < number) {
            smallest = guess + 1;
        } else if (guess == number) {
            FacesContext.getCurrentInstance().addMessage(null, new
FacesMessage("Correct!"));
        }
        remainingGuesses--;
    }

    /**
     * Reset the game, by putting all values back to their defaults,
     * and getting a new random number. We also call this method
     * when the user starts playing for the first time using
     * {@linkplain PostConstruct @PostConstruct} to set the initial
     * values.
     */

```



```

@PostConstruct
public void reset() {
    this.smallest = 0;
    this.guess = 0;
    this.remainingGuesses = 10;
    this.biggest = maxNumber;
    this.number = randomNumber.get();
}

/**
 * A JSF validation method which checks whether the guess is
 * valid. It might not be valid because there are no guesses left,
 * or because the guess is not in range.
 */
public void validateNumberRange(FacesContext context,
    UIComponent toValidate, Object value) {
    if (remainingGuesses <= 0) {
        FacesMessage message = new FacesMessage("No guesses
left!");
        context.addMessage(toValidate.getClientId(context),
message);
        ((UIInput) toValidate).setValid(false);
        return;
    }
    int input = (Integer) value;

    if (input < smallest || input > biggest) {
        ((UIInput) toValidate).setValid(false);

        FacesMessage message = new FacesMessage("Invalid
guess");
        context.addMessage(toValidate.getClientId(context),
message);
    }
}
}

```

ANHANG A. REFERENZMATERIAL

A.1. SERVER-LAUFZEIT ARGUMENTE

Das Anwendungs-Server Startup-Skript akzeptiert Argumente und Switches zur Runtime. Dies ermöglicht dem Server unter alternativen Konfigurationen zu starten, als in den **standalone.xml**, **domain.xml** und **host.xml** Konfigurationsdateien definiert sind.

Alternative Konfigurationen können einen Server-Start mit einem alternativen Set von Socket-Bindungen oder einer sekundären Konfiguration beinhalten.

Auf die Liste verfügbarer Parameter kann durch Eingabe des Hilfs-Switches **-h** oder **--help** beim Startup zugegriffen werden.

Tabelle A.1. Runtime Switches und Argumente

Argument oder Switch	Betriebsmodus	Beschreibung
--admin-only	Standalone	Setzt den laufenden Servertyp auf ADMIN_ONLY . Dies führt zur Öffnung administrativer Schnittstellen und zur Annahme von Management-Anfragen, aber nicht zum Start anderer Runtime-Dienste oder zur Annahme von Endnutzer-Anfragen. Beachten Sie, dass empfohlen wird, stattdessen --start-mode=admin-only zu verwenden.
--admin-only	Domain	Setzt den laufenden Host-Controller Typ auf ADMIN_ONLY , damit Verwaltungs-Schnittstellen geöffnet werden und Management-Anfragen angenommen werden, nicht jedoch Server gestartet werden oder, falls dieser Host-Controller der Master für die Domain ist, eingehende Verbindungen von Slave Host-Controllern angenommen werden.
-b=<value>, -b <value>	Standalone, Domain	Setzt die jboss.bind.address , die zur Konfiguration der Bindungs-Adresse für die öffentliche Schnittstelle verwendet wird. Dies ist standardmäßig 127.0.0.1 , wenn kein Wert angegeben ist. Das Festlegen der Bindungsadresse für andere Schnittstellen können Sie dem Eintrag -b<interface>=<value> entnehmen.
-b<interface>=<value>	Standalone, Domain	Setzt die Systemeigenschaft jboss.bind.address.<interface> auf den angegebenen Wert, beispielsweise bmanagement=IP_ADDRESS

Argument oder Switch	Betriebsmodus	Beschreibung
--backup	Domain	Hält eine Kopie der persistenten Domänenkonfiguration vor, auch wenn dieser Host nicht der Domain Controller ist.
-c=<config>, -c <config>	Standalone	Name der zu verwendenden Server-Konfigurationsdatei. Der Standard ist standalone.xml .
-c=<config>, -c <config>	Domain	Name der zu verwendenden Server-Konfigurationsdatei. Der Standard ist domain.xml .
--cached-dc	Domain	Ist der Host nicht der Domain Controller und kann er den Domain Controller zur Bootzeit nicht kontaktieren, über eine lokal gecachte Kopie der Domänenkonfiguration booten.
--debug [<port>]	Standalone	Debug-Modus mit einem optionalen Argument aktivieren um den Port anzugeben. Das funktioniert nur, wenn dieser vom Start-Skript unterstützt wird.
-D<name>[=<value>]	Standalone, Domain	Stellen Sie eine Systemeigenschaft ein.
--domain-config=<config>	Domain	Name der zu verwendenden Server-Konfigurationsdatei. Der Standard ist domain.xml .
-h, --help	Standalone, Domain	Hilfenachricht anzeigen und beenden.
--host-config=<config>	Domain	Name der zu verwendenden Host-Konfigurationsdatei. Der Standard ist host.xml .
--interprocess-hc-address=<address>	Domain	Adresse, an der der Host-Controller auf Kommunikation vom Prozess-Controller lauschen soll.
--interprocess-hc-port=<port>	Domain	Port, an dem der Host-Controller auf Kommunikation vom Prozess-Controller lauschen soll.

Argument oder Switch	Betriebsmodus	Beschreibung
--master-address=<address>	Domain	Setzt die Systemeigenschaft jboss.domain.master.address auf den angegebenen Wert. Bei einer Standard-Slave-Host-Konfiguration wird dies verwendet, um die Adresse des Master-Host-Controllers zu konfigurieren.
--master-port=<port>	Domain	Setzt die Systemeigenschaft jboss.domain.master.port auf den angegebenen Wert. In einer standardmäßigen Slave Host-Controller Config konfiguriert dies gewöhnlich den Port, der für systemeigene Management-Kommunikation vom Master Host-Controller verwendet wird.
--read-only-server-config=<config>	Standalone	Name der zu verwendenden Server-Konfigurationsdatei. Diese unterscheidet sich von --server-config und -c darin, dass die Originaldatei nie überschrieben wird.
--read-only-domain-config=<config>	Domain	Name der zu verwendenden Domain-Konfigurationsdatei. Diese unterscheidet sich von --domain-config und -c darin, dass die ursprüngliche Datei nie überschrieben wird.
--read-only-host-config=<config>	Domain	Name der zu verwendenden Host-Konfigurationsdatei. Diese unterscheidet sich von --host-config darin, dass die ursprüngliche Datei nie überschrieben wird.
-P=<url>, -P <url>, --properties=<url>	Standalone, Domain	Systemeigenschaften von der gegebenen URL laden.
--pc-address=<address>	Domain	Adresse, an der der Prozess-Controller auf Kommunikation von Prozessen lauscht, die er kontrolliert.
--pc-port=<port>	Domain	Port, an dem der Prozess-Controller auf Kommunikation von Prozessen lauscht, die er kontrolliert.
-S<name>[=<value>]	Standalone	Stellen Sie eine Sicherheitseigenschaft ein.
-secmgr	Standalone, Domain	Führt den Server mit installiertem Sicherheits-Manager aus.

Argument oder Switch	Betriebsmodus	Beschreibung
--server-config=<config>	Standalone	Name der zu verwendenden Server-Konfigurationsdatei. Der Standard ist standalone.xml .
--start-mode=<mode>	Standalone	<p>Setzt den Startmodus des Servers. Diese Option kann nicht mit --admin-only kombiniert verwendet werden. Gültige Werte sind:</p> <ul style="list-style-type: none"> • normal: Der Server startet normal. • admin-only: Der Server öffnet nur administrative Schnittstellen und nimmt Management-Anfragen an, startet jedoch keine anderen Laufzeit-Dienste und akzeptiert keine Endbenutzeranfragen. • suspend: Der Server startet im pausierten Modus und verarbeitet erst Anfragen, wenn er wieder ausgeführt wird.
-u=<value>, -u <value>	Standalone, Domain	Setzt die Systemeigenschaft jboss.default.multicast.address , die zur Konfiguration der Multicast-Adresse in den Socket-Bindungs-Elementen in den Konfigurationsdateien verwendet wird. Dies ist standardmäßig 230.0.0.4 , wenn kein Wert angegeben wird.
-v, -V, --version	Standalone, Domain	Anwendungs-Server-Version anzeigen und beenden.



WARNUNG

Die mit der JBoss EAP gelieferten Konfigurationsdateien sind darauf ausgerichtet, das Verhalten der Switches zu handhaben, z.B. **-b** und **-u**. Wenn Sie Ihre Konfigurationsdateien dahingehend ändern, dass die vom Switch kontrollierten Systemeigenschaften nicht mehr verwendet werden, dann wird es keine Auswirkung haben, wenn man sie zum Startbefehl hinzufügt.

A.2. DIENSTPROGRAMM-ARGUMENT FÜR BENUTZER-HINZUFÜGEN

Die folgende Tabelle beschreibt die verfügbaren Argumente für das **add-user.sh** oder **add-user.bat** Skript, ein Dienstprogramm für das Hinzufügen neuer Benutzer zur Eigenschaften-Datei für sofort verfügbare Authentifizierung.

Tabelle A.2. Befehlsargument für Benutzer Hinzufügen

Befehlszeilenargument	Beschreibung
-a	Erstellt einen neuen Benutzer im Anwendungs-Realm. Wird es ausgelassen, so wird standardmäßig ein Benutzer im Management-Realm erstellt.
-dc <value>	Das Domain-Konfigurationsverzeichnis, das die Eigenschaften-Datei enthalten soll. Wird es ausgelassen, ist das Standard-Verzeichnis EAP_HOME/domain/configuration/ .
-sc <value>	Ein alternatives Standalone-Server-Verzeichnis, das die Eigenschaften-Datei enthalten soll. Wird es ausgelassen, so ist das Standard-Verzeichnis EAP_HOME/standalone/configuration/ .
-up, --user-properties <value>	Der Name der Eigenschaften-Datei für alternative Benutzer. Das kann ein absoluter Pfad sein oder ein Dateiname, der in Verbindung mit dem -sc oder -dc Argument benutzt wird, welches das alternative Konfigurationsverzeichnis angibt.
-g, --group <value>	Eine Komma-getrennte Liste von Gruppen, die diesem Benutzer zuzuweisen sind.
-gp, --group-properties <value>	Der Name der Eigenschaften-Datei für alternative Gruppen. Das kann ein absoluter Pfad sein oder ein Dateiname, der in Verbindung mit dem -sc oder -dc Argument benutzt wird, welches das alternative Konfigurationsverzeichnis angibt.
-p, --password <value>	Das Passwort des Benutzers.
-u, --user <value>	Der Name des Benutzers. Benutzernamen können nur die folgenden Zeichen enthalten, in beliebiger Anzahl und Reihenfolge: <ul style="list-style-type: none"> • Alphanumerische Zeichen (a-z, A-Z, 0-9) • Striche (-), Punkte (.), Kommas (,) @-Zeichen • Backslash (\) • Gleichheitszeichen (=)

Befehlszeilenargument	Beschreibung
-r, --realm <value>	Der Name des Realms, das die Verwaltungs-Schnittstellen sichert. Wird es ausgelassen, so ist ManagementRealm der Standard.
-s, --silent	Führen Sie das add-user Skript ohne Ausgabe an die Konsole aus.
-e, --enable	Benutzer aktivieren.
-d, --disable	Benutzer deaktivieren.
-cw, --confirm-warning	Warnung im interaktiven Modus automatisch bestätigen.
-h, --help	Gebrauchsinformationen für das add-user Skript anzeigen.
-ds, --display-secret	Den geheimen Wert im nicht interaktiven Modus drucken.

A.3. SCHNITTSTELLEN-ATTRIBUTE



ANMERKUNG

Attributnamen in dieser Tabelle werden so aufgelistet, wie sie im Managementmodell erscheinen, beispielsweise bei Verwendung einer Management-CLI. Siehe die Schema-Definitionsdatei unter **EAP_HOME/docs/schema/wildfly-config_5_0.xsd**, um die Elemente zu betrachten, wie sie in der XML erscheinen, da sich dies vom Managementmodell unterscheiden kann.

Tabelle A.3. Schnittstellen-Attribute und Werte

Schnittstellen-Elemente	Beschreibung
beliebig	Element, das anzeigt, dass ein Teil der Auswahl-Kriterien für eine Schnittstelle die Erfüllung mindestens einer, jedoch nicht unbedingt aller verschachtelten Kriterienkataloge sein sollte.

Schnittstellen-Elemente	Beschreibung
any-address	Leeres Element, das anzeigt, dass die Sockets, die diese Schnittstelle verwenden, an eine Wildcard-Adresse gebunden sein sollten. Es wird die IPv6-Wildcard-Adresse (::) verwendet, außer die java.net.preferIPv4Stack Systemeigenschaft ist auf "true" gesetzt, in welchem Falle die IPv4-Wildcard-Adresse (0.0.0.0) verwendet wird. Falls ein Socket auf einem Dualstack-Rechner an eine IPv6 Anylocal-Adresse gebunden ist, so kann es sowohl IPv6- als auch IPv4-Datenverkehr empfangen; falls es an eine IPv4 (IPv4-mapped) Anylocal-Adresse gebunden ist, so kann es nur IPv4-Datenverkehr empfangen.
inet-address	Entweder eine IP-Adresse in IPv6 oder IPv4 punktierter Dezimalschreibweise oder ein Hostname, der zu einer IP-Adresse aufgelöst werden kann.
link-local-address	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob eine mit ihm assoziierte Adresse Link-Local ist oder nicht.
loopback	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob es sich um eine Loopback-Schnittstelle handelt oder nicht.
loopback-address	Eine Loopback-Adresse, die u.U. nicht auf der Loopback-Schnittstelle des Rechners konfiguriert ist. Unterscheidet sich vom Netzadressen-Typ darin, dass der gegebene Wert selbst dann verwendet wird, wenn kein NIC gefunden werden kann, dem die IP-Adresse zugeordnet ist.
multicast	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob Multicast unterstützt wird oder nicht.
name	Der Name der Schnittstelle
nic	Der Name einer Netzwerk-Schnittstelle (z.B. eth0, eth1, lo).
nic-match	Ein regulärer Ausdruck, mit dem die Namen der auf dem Rechner verfügbaren Netzwerk-Schnittstellen abgeglichen werden können, um eine zulässige Schnittstelle zu finden.
not	Element, das anzeigt, dass ein Teil der Auswahlkriterien für eine Schnittstelle die Nichterfüllung jeglicher verschachtelter Kriterienkataloge sein sollte.

Schnittstellen-Elemente	Beschreibung
point-to-point	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob es sich um ein Point-to-Point Interface handelt oder nicht.
public-address	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob es eine öffentlich routingfähige Adresse besitzt oder nicht.
site-local-address	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob eine mit ihm assoziierte Adresse Site-lokal ist oder nicht.
subnet-match	Eine Netzwerk-IP-Adresse und die Anzahl Bits im Netzwerk-Präfix der Adresse, geschrieben in <i>Slash-Notation</i> , z.B. 192.168.0.0/16 .
hoch	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob diese derzeit aktiv ist oder nicht.
virtuell	Leeres Element, das anzeigt, dass es Teil der Auswahlkriterien für eine Schnittstelle sein sollte, ob es sich um eine virtuelle Schnittstelle handelt oder nicht.

A.4. SOCKET-BINDUNG ATTRIBUTE



ANMERKUNG

Attributnamen in dieser Tabelle werden so aufgelistet, wie sie im Managementmodell erscheinen, beispielsweise bei Verwendung einer Management-CLI. Siehe die Schema-Definitionsdatei unter ***EAP_HOME/docs/schema/wildfly-config_5_0.xsd***, um die Elemente zu betrachten, wie sie in der XML erscheinen, da sich dies vom Managementmodell unterscheiden kann.

Tabelle A.4. Socket-Bindung Attribute

Attribut	Beschreibung
client-mappings	Gibt die Client Mappings für diese Socket-Bindung an. Ein Client, der mit diesem Socket verbindet, sollte die im Mapping angegebene Zieladresse verwenden, die der gewünschten Outbound Schnittstelle entspricht. Dies ermöglicht erweiterte Netzwerk-Topologien, die entweder Network Address Translation verwenden, oder Bindungen auf mehreren Netzwerk-Schnittstellen haben, um zu funktionieren. Jedes Mapping sollte in deklarierter Reihenfolge ausgewertet werden, wobei die erste erfolgreiche Übereinstimmung benutzt wird, um das Ziel zu bestimmen.

Attribut	Beschreibung
fixed-port	Wenn der Port-Wert fest bleiben soll, auch wenn numerische Offsets auf die anderen Sockets in der Socket-Gruppe angewendet werden.
Schnittstelle	Name der Schnittstelle, an die das Socket gebunden sein soll, oder bei Multicast-Sockets die Schnittstelle, auf der es lauschen soll. Dies sollte eine deklarierte Schnittstellen sein. Falls nicht Wert definiert, so wird der Wert des default-interface Attributs von der einschließenden Gruppe von Socket-Bindungen verwendet.
multicast-address	Multicast-Adresse, auf der das Socket Multicast-Datenverkehr empfangen soll. Falls diese nicht angegeben ist, wird das Socket nicht konfiguriert Multicast zu empfangen.
multicast-port	Port, an dem das Socket Multicast-Traffic empfangen soll. Muss konfiguriert sein, wenn multicast-address konfiguriert ist.
name	Der Name des Sockets. Dienste, die auf die Socket-Konfigurationsinformationen zugreifen müssen, werden es unter Verwendung dieses Namens finden. Dieses Attribut ist erforderlich.
Port	Nummer des Ports, an den das Socket gebunden sein soll. Beachten Sie, dass dieser Wert außer Kraft gesetzt werden kann, wenn ein Server ein Port-Offset anwendet um alle Port-Werte zu erhöhen oder zu verringern.

A.5. STANDARD SOCKET-BINDUNGEN

Die folgenden Tabellen zeigen die Standard-Socket-Bindungen für jede Socket-Bindungsgruppe an.

- [standard-sockets](#)
- [ha-sockets](#)
- [full-sockets](#)
- [full-ha-sockets](#)
- [load-balancer-sockets](#)

Tabelle A.5. standard-sockets

Socket-Bindung	Port	Beschreibung
ajp	8009	Apache JServ Protocol. Für HTTP-Clustering und Lastverteilung verwendet.
http	8080	Der Standard-Port für bereitgestellte Web-Anwendungen.

Socket-Bindung	Port	Beschreibung
https	8443	SSL-verschlüsselte Verbindung zwischen implementierten Web-Anwendungen und Clients.
management-http	9990	Wird verwendet für HTTP Kommunikation mit der Management-Ebene.
management-https	9993	Wird verwendet für HTTPS Kommunikation mit der Management-Ebene.
txn-recovery-environment	4712	Der JTA-Transaction-Recovery-Manager.
txn-status-manager	4713	Der JTA / JTS Transaction-Manager.

Tabelle A.6. ha-sockets

Socket-Bindung	Port	Multicast Port	Beschreibung
ajp	8009		Apache JServ Protocol. Für HTTP-Clustering und Lastverteilung verwendet.
http	8080		Der Standard-Port für bereitgestellte Web-Anwendungen.
https	8443		SSL-verschlüsselte Verbindung zwischen implementierten Web-Anwendungen und Clients.
jgroups-mping		45700	Multicast. Wird verwendet um erstes Mitglied in einem HA-Cluster zu finden.
jgroups-tcp	7600		Unicast Peer Discovery in HA-Clustern mittels TCP.
jgroups-udp	55200	45688	Multicast-Peer-Discovery in HA-Clustern mittels UDP.
management-http	9990		Wird verwendet für HTTP Kommunikation mit der Management-Ebene.
management-https	9993		Wird verwendet für HTTPS Kommunikation mit der Management-Ebene.

Socket-Bindung	Port	Multicast Port	Beschreibung
modcluster		23364	Multicast-Port für Kommunikation zwischen der JBoss EAP und dem HTTP-Lastverteiler.
txn-recovery-environment	4712		Der JTA-Transaction-Recovery-Manager.
txn-status-manager	4713		Der JTA / JTS Transaction-Manager.

Tabelle A.7. full-sockets

Socket-Bindung	Port	Beschreibung
ajp	8009	Apache JServ Protocol. Für HTTP-Clustering und Lastverteilung verwendet.
http	8080	Der Standard-Port für bereitgestellte Web-Anwendungen.
https	8443	SSL-verschlüsselte Verbindung zwischen implementierten Web-Anwendungen und Clients.
iiop	3528	CORBA-Dienste für JTS-Transaktionen und andere ORB-abhängige Dienste.
iiop-ssl	3529	SSL-verschlüsselte CORBA-Dienste.
management-http	9990	Wird verwendet für HTTP Kommunikation mit der Management-Ebene.
management-https	9993	Wird verwendet für HTTPS Kommunikation mit der Management-Ebene.
txn-recovery-environment	4712	Der JTA-Transaction-Recovery-Manager.
txn-status-manager	4713	Der JTA / JTS Transaction-Manager.

Tabelle A.8. full-ha-sockets

Name	Port	Multicast Port	Beschreibung
------	------	----------------	--------------

Name	Port	Multicast Port	Beschreibung
ajp	8009		Apache JServ Protocol. Für HTTP-Clustering und Lastverteilung verwendet.
http	8080		Der Standard-Port für bereitgestellte Web-Anwendungen.
https	8443		SSL-verschlüsselte Verbindung zwischen implementierten Web-Anwendungen und Clients.
iiop	3528		CORBA-Dienste für JTS-Transaktionen und andere ORB-abhängige Dienste.
iiop-ssl	3529		SSL-verschlüsselte CORBA-Dienste.
jgroups-mping		45700	Multicast. Wird verwendet um erstes Mitglied in einem HA-Cluster zu finden.
jgroups-tcp	7600		Unicast Peer Discovery in HA-Clustern mittels TCP.
jgroups-udp	55200	45688	Multicast-Peer-Discovery in HA-Clustern mittels UDP.
management-http	9990		Wird verwendet für HTTP Kommunikation mit der Management-Ebene.
management-https	9993		Wird verwendet für HTTPS Kommunikation mit der Management-Ebene.
modcluster		23364	Multicast-Port für Kommunikation zwischen der JBoss EAP und dem HTTP-Lastverteiler.
txn-recovery-environment	4712		Der JTA-Transaction-Recovery-Manager.
txn-status-manager	4713		Der JTA / JTS Transaction-Manager.

Tabelle A.9. load-balancer-sockets

Name	Port	Multicast Port	Beschreibung
http	8080		Der Standard-Port für bereitgestellte Web-Anwendungen.
https	8443		SSL-verschlüsselte Verbindung zwischen implementierten Web-Anwendungen und Clients.
management-http	9990		Wird verwendet für HTTP Kommunikation mit der Management-Ebene.
management-https	9993		Wird verwendet für HTTPS Kommunikation mit der Management-Ebene.
mcmp-management	8090		Der Port für die Mod-Cluster Management Protocol (MCMP) Verbindung zur Übertragung von Lifecycle-Ereignissen.
modcluster		23364	Multicast-Port für Kommunikation zwischen der JBoss EAP und dem HTTP-Lastverteiler.

Revised on 2018-01-11 05:28:21 EST