



Red Hat JBoss Enterprise Application Platform 7.1

Guia de Introdução

Para uso com o Red Hat JBoss Enterprise Application Platform 7.1

Red Hat JBoss Enterprise Application Platform 7.1 Guia de Introdução

Para uso com o Red Hat JBoss Enterprise Application Platform 7.1

Nota Legal

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.

Resumo

Este guia contém informações básicas de introdução ao Red Hat JBoss Enterprise Application Platform 7.1.

Índice

CAPÍTULO 1. INTRODUÇÃO	4
1.1. SOBRE O RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7	4
1.2. SOBRE O GUIA DE INTRODUÇÃO	4
CAPÍTULO 2. ADMINISTRAÇÃO DO JBOSS EAP	5
2.1. BAIXANDO E INSTALANDO O JBOSS EAP	5
2.1.1. Pré-requisitos de Instalação	5
2.1.2. Baixando o JBoss EAP	5
2.1.3. Instalando o JBoss EAP	5
2.2. INICIANDO E ENCERRANDO O JBOSS EAP	6
2.2.1. Iniciando o JBoss EAP	6
Iniciando o JBoss EAP como um Servidor Autônomo	6
Iniciando o JBoss EAP em um Domínio Gerenciado	7
2.2.2. Encerrando o JBoss EAP	7
Encerrando uma Instância Interativa do JBoss EAP	7
Encerrando uma Instância do JBoss EAP em Segundo Plano	7
2.3. GERENCIAMENTO DO JBOSS EAP	8
2.3.1. Usuários de Gerenciamento	8
2.3.1.1. Adicionando um Usuário de Gerenciamento	8
2.3.1.2. Executando o Utilitário Add-User Sem Interatividade	9
Criando um Usuário Pertencendo a Múltiplos Grupos	10
Especificando um Arquivo de Propriedades Alternativas	10
2.3.2. Interfaces de Gerenciamento	10
2.3.2.1. CLI de Gerenciamento	10
Iniciando a CLI de Gerenciamento	11
Conectando-se a um Servidor em Execução	11
Exibindo Ajuda	11
Encerrando a CLI de Gerenciamento	11
Exibindo as Configurações do Sistema	11
Atualizando as Configurações do Sistema	12
Iniciando os Servidores	12
2.3.2.2. Console de Gerenciamento	12
2.3.3. Arquivos de Configuração	13
2.3.3.1. Arquivos de Configuração de Servidor Autônomo	13
2.3.3.2. Arquivos de Configuração de Domínio Gerenciado	14
2.3.3.3. Fazendo o Backup dos Dados de Configuração	14
2.3.3.4. Snapshots de arquivos de configuração	15
Tirando um Snapshot	15
Listando os Snapshots	15
Excluindo um Snapshot	16
Iniciando o Servidor com um Snapshot	16
2.3.3.5. Substituição de Propriedades	16
Expressões Aninhadas	17
Substituição de Propriedades Baseadas no Descritor	17
2.4. CONFIGURAÇÃO DE PORTA E REDE	18
2.4.1. Interfaces	18
2.4.1.1. Configurações de Interface Padrão	19
2.4.1.2. Configurando Interfaces	19
Adicionando uma Interface com um Valor NIC	20
Adicionando uma Interface com Diversos Valores Condicionais	20
Atualizando um Atributo da Interface	20

Adicionando uma Interface a um Servidor em um Domínio Gerenciado	20
2.4.2. Associações de Soquete	21
2.4.2.1. Portas de Gerenciamento	21
2.4.2.2. Associações de Soquete Padrão	21
Servidor Autônomo	21
Domínio Gerenciado	22
2.4.2.3. Configurando as Associações de Soquete	23
2.4.2.4. Deslocamentos de Porta	24
2.4.3. Endereços IPv6	24
Configurando a Pilha JVM para Endereços IPv6	24
Atualizando Declarações de Interface para Endereços IPv6	25
CAPÍTULO 3. DESENVOLVENDO APLICATIVOS USANDO O JBOSS EAP	26
3.1. VISÃO GERAL	26
3.2. CONFIGURANDO O AMBIENTE DE DESENVOLVIMENTO	26
3.3. USANDO OS EXEMPLOS DE INÍCIO RÁPIDO	26
3.3.1. Sobre o Maven	26
3.3.2. Usando Maven com os Inícios Rápidos	27
3.3.3. Baixando e Executando os Inícios Rápidos	27
3.3.3.1. Baixando os Inícios Rápidos	27
3.3.3.2. Executando os Inícios Rápidos no JBoss Developer Studio	27
3.3.3.3. Executando os Inícios Rápidos através da Linha de Comando	35
3.4. REVISE OS EXEMPLOS DE INÍCIO RÁPIDO	35
3.4.1. Explorando a Inicialização Rápida helloworld	35
Pré-requisitos	35
Examine a Estrutura do Diretório	36
Examine o Código	36
3.4.2. Explorando a Inicialização Rápida do numberguess (adivinhação de número)	37
Pré-requisitos	38
Examinando os Arquivos de Configuração	38
3.4.2.1. Examinando o código JSF	39
3.4.2.2. Examinação nos Arquivos de Classe	40
APÊNDICE A. MATERIAL DE REFERÊNCIA	45
A.1. ARGUMENTOS DE TEMPO DE EXECUÇÃO DO SERVIDOR	45
A.2. ARGUMENTOS PARA O UTILITÁRIO ADD-USER	48
A.3. ATRIBUTOS DE INTERFACE	50
A.4. ATRIBUTOS DA ASSOCIAÇÃO DE SOQUETE	52
A.5. ASSOCIAÇÕES DE SOQUETE PADRÃO	52

CAPÍTULO 1. INTRODUÇÃO

1.1. SOBRE O RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7

O Red Hat JBoss Enterprise Application Platform 7 (JBoss EAP) é uma plataforma de middleware construída com padrões abertos e em conformidade com a especificação Java Enterprise Edition 7.

O JBoss EAP oferece uma estrutura modular que permite a habilitação de serviços apenas quando necessária, melhorando a velocidade de inicialização.

O Console de Gerenciamento e a CLI (Interface de Linha de Comando) de Gerenciamento tornam as edições dos arquivos de configuração XML desnecessárias e agregam a habilidade de utilizar script e automatizar tarefas.

O JBoss EAP fornece dois modos de operação para as instâncias do JBoss EAP: o servidor autônomo ou o domínio gerenciado. O modo de operação do servidor autônomo representa o JBoss EAP em execução como uma instância de servidor único. O modo de operação do domínio gerenciado permite o gerenciamento de múltiplas instâncias do JBoss EAP a partir de um ponto de controle único.

Além disso, o JBoss EAP inclui estruturas e APIs para o desenvolvimento rápido de aplicativos Java EE seguros e escaláveis.

1.2. SOBRE O GUIA DE INTRODUÇÃO

Este guia tem como finalidade orientar os usuários na inicialização e execução rápida do JBoss EAP. Ele cobre tarefas [administrativas](#) como instalação, gerenciamento e configuração básica do JBoss EAP. Também auxilia os [desenvolvedores](#) a escrever aplicativos Java EE 7 usando os inícios rápidos do JBoss EAP.

Para saber mais, consulte o conjunto completo de [documentação do JBoss EAP](#).

CAPÍTULO 2. ADMINISTRAÇÃO DO JBOSS EAP

2.1. BAIXANDO E INSTALANDO O JBOSS EAP

Este guia fornece instruções básicas para o download e a instalação do JBoss EAP usando a instalação ZIP, que é independente de plataforma.

Consulte o [Guia de instalação](#) para obter mais detalhes, incluindo instruções para a instalação do JBoss EAP usando os métodos de instalação com instalador gráfico ou pacote RPM.

2.1.1. Pré-requisitos de Instalação

Verifique se os pré-requisitos a seguir são atendidos antes de instalar o JBoss EAP:

Pré-requisitos Comuns

- Seu sistema é suportado de acordo com [as configurações suportadas no JBoss EAP 7](#).
- Seu sistema deve estar em dia em relação às erratas e atualizações emitidas pela Red Hat.

Pré-requisitos da Instalação ZIP

- O usuário que executará o JBoss EAP deve ter acesso de leitura e escrita para o diretório de instalação.
- O kit de desenvolvimento Java desejado deve ter sido instalado.
- Para o HP-UX da Hewlett-Packard, um utilitário de **descompactação** deve ter sido instalado.
- Para o Windows Server, as variáveis de ambiente **JAVA_HOME** e **PATH** devem ter sido definidas.

2.1.2. Baixando o JBoss EAP

O arquivo ZIP do JBoss EAP está disponível no Portal do Consumidor Red Hat. A instalação do arquivo ZIP é independente da plataforma.

1. Faça o login no [Portal do Cliente Red Hat](#).
2. Clique em **Downloads**.
3. Clique em **Red Hat JBoss Enterprise Application Platform** na lista **Downloads de Produtos**.
4. No menu suspenso **Versão**, selecione **7.1**.
5. Localize o **Red Hat JBoss Enterprise Application Platform 7.1.0** na lista e clique no link **Download**.

2.1.3. Instalando o JBoss EAP

Depois que o arquivo ZIP de instalação do JBoss EAP foi baixado, ele pode ser instalado extraindo os conteúdos do pacote.

1. Se necessário, mova o arquivo ZIP para o servidor e o local onde o JBoss EAP deve ser instalado.

**NOTA**

O usuário que executará o JBoss EAP deve ter acesso de leitura e escrita para este diretório.

2. Extraia o arquivo ZIP.

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

**NOTA**

Para o Windows Server, clique com o botão direito do mouse no arquivo ZIP e selecione **Extrair Todos**.

O diretório criado com a extração do arquivo ZIP é o diretório de nível superior para a instalação do JBoss EAP. Ele é referido como **EAP_HOME**.

2.2. INICIANDO E ENCERRANDO O JBOSS EAP

2.2.1. Iniciando o JBoss EAP

O JBoss EAP é compatível com Red Hat Enterprise Linux, Windows Server, Oracle Solaris e Hewlett-Packard HP-UX e funciona em um servidor autônomo ou modo de operação de domínio gerenciado. O comando específico para iniciar o JBoss EAP depende da plataforma subjacente e do modo de operação desejado.

Os servidores são iniciados primeiramente em um estado suspenso e não aceitarão solicitações até que todos os serviços necessários tenham sido iniciados. Então, os servidores são colocados em um estado de operação normal e podem começar a aceitar solicitações.

Iniciando o JBoss EAP como um Servidor Autônomo

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

**NOTA**

Para o Windows Server, use o script **EAP_HOME\bin\standalone.bat**.

Esse script de inicialização usa o arquivo **EAP_HOME/bin/standalone.conf** (ou **standalone.conf.bat** para o Windows Server) para definir algumas preferências padrão, como opções JVM. Você pode personalizar as configurações desse arquivo.

O JBoss EAP usa o arquivo de configuração **standalone.xml** por padrão, mas pode ser iniciado usando um outro diferente. Para detalhes sobre os arquivos de configuração de servidor autônomo disponíveis e como usá-los, consulte a seção [Arquivos de Configuração](#)

de Servidor Autônomo.

Para uma lista completa de todos os argumentos do script de inicialização disponíveis e suas finalidades, utilize o argumento **--help** ou consulte a seção [Argumentos de tempo de execução do servidor](#).

Iniciando o JBoss EAP em um Domínio Gerenciado

O controlador de domínio deve ser iniciado antes dos servidores em qualquer um dos grupos de servidores no domínio. Use o script a seguir para iniciar primeiro o controlador de domínio e, depois, para cada controlador do host associado.

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



NOTA

Para o Windows Server, use o script **EAP_HOME\bin\domain.bat**.

Esse script de inicialização usa o arquivo **EAP_HOME/bin/domain.conf** (ou **domain.conf.bat** para o Windows Server) para definir algumas preferências padrão, como opções JVM. Você pode personalizar as configurações desse arquivo.

O JBoss EAP usa o arquivo de configuração de host **host.xml** por padrão, mas pode ser iniciado usando um outro diferente. Para detalhes sobre os arquivos de configuração de domínio gerenciado disponíveis e como usá-los, consulte a seção [Arquivos de Configuração de Domínio Gerenciado](#).

Ao configurar um domínio gerenciado, argumentos adicionais precisarão ser passados ao script de inicialização. Para uma lista completa dos argumentos disponíveis do script de inicialização e suas finalidades, use o argumento **--help** ou consulte a seção [Argumentos de tempo de execução do servidor](#).

2.2.2. Encerrando o JBoss EAP

A forma como você deve encerrar o JBoss EAP depende de como ele foi iniciado.

Encerrando uma Instância Interativa do JBoss EAP

Pressione **Ctrl+C** no terminal onde o JBoss EAP foi iniciado.

Encerrando uma Instância do JBoss EAP em Segundo Plano

Use a CLI de gerenciamento para conectar-se com a instância em execução e desligue o servidor.

1. Inicie a CLI de gerenciamento.

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

2. Emita o comando **shutdown**.

```
shutdown
```

**NOTA**

Quando estiver executando em um domínio gerenciado, você deve especificar o nome do host a ser desligado usando o argumento **--host** com o comando **shutdown**.

2.3. GERENCIAMENTO DO JBOSS EAP

O JBoss EAP usa uma configuração simplificada, com um arquivo de configuração por servidor autônomo ou domínio gerenciado. A configuração padrão para um servidor autônomo é armazenada no arquivo

EAP_HOME/standalone/configuration/standalone.xml, e a configuração padrão para um domínio padrão é armazenada no arquivo **EAP_HOME/domain/configuration/domain.xml**. A configuração padrão para um controlador de host é armazenada no arquivo **EAP_HOME/domain/configuration/host.xml**.

O JBoss EAP pode ser configurado usando a CLI de gerenciamento de linha de comando, o console de gerenciamento baseado na web, a API Java ou a API HTTP. As alterações feitas usando essas interfaces de gerenciamento persistem automaticamente, e os arquivos de configuração XML são sobrescritos pela API de gerenciamento. A CLI de gerenciamento e o console de gerenciamento são os métodos preferenciais. Não é recomendado editar os arquivos de configuração XML manualmente.

2.3.1. Usuários de Gerenciamento

A configuração padrão do JBoss EAP fornece autenticação local para que um usuário possa acessar a CLI de gerenciamento no host local sem necessidade de autenticação.

No entanto, você deve adicionar pelo menos um usuário de gerenciamento se desejar acessar a CLI de gerenciamento de forma remota ou usar o console de gerenciamento, que é considerado acesso remoto mesmo se o tráfego for originado no host local. Se você tentar acessar o console de gerenciamento antes de adicionar um usuário de gerenciamento, você receberá uma mensagem de erro.

Se o JBoss EAP for instalado usando o instalador gráfico, um usuário de gerenciamento será criado durante o processo de instalação.

Este guia aborda o gerenciamento de usuários simples do JBoss EAP com o uso do script **add-user**, que é um utilitário para adicionar novos usuários aos arquivos de propriedades para autenticação pronta.

Para ver opções mais avançadas de autenticação e autorização, como LDAP ou controle de acesso baseado em função (RBAC), consulte a seção [Autenticação de gerenciamento principal](#) do guia *Arquitetura de segurança* do JBoss EAP.

2.3.1.1. Adicionando um Usuário de Gerenciamento

1. Execute o script utilitário **add-user** e siga os prompts.

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

**NOTA**

Para o Windows Server, use o script **EAP_HOME\bin\add-user.bat**.

2. Pressione **ENTER** para selecionar a opção padrão **(a)** e adicionar um usuário de gerenciamento.
Esse usuário será adicionado ao *ManagementRealm* e será autorizado a realizar operações de gerenciamento usando o console de gerenciamento ou a CLI de gerenciamento. A outra opção **(b)** adiciona um usuário ao *ApplicationRealm*, que é usado para aplicativos e não fornece nenhuma permissão específica.
3. Insira o nome de usuário e a senha. Você será solicitado a confirmar a senha.



NOTA

Os nomes de usuário podem conter somente os seguintes caracteres, em qualquer quantidade e em qualquer ordem:

- Caracteres alfanuméricos (a-z, A-Z, 0-9)
- Hífen (-), ponto (.), vírgula (,), arroba (@)
- Barra invertida (\)
- Sinal de igual (=)

Por padrão, o JBoss EAP permite senhas fracas, mas emitirá um aviso.

Consulte a seção [Definição de restrições de senha do utilitário add-user](#) do *Guia de configuração* do JBoss EAP para obter detalhes sobre a alteração desse comportamento padrão.

4. Insira uma lista separada por vírgulas dos grupos aos quais o usuário pertence. Se você não deseja que o usuário pertença a grupo algum, pressione **ENTER** para deixar em branco.
5. Revise as informações e insira **yes** para confirmar.
6. Determine se este usuário representa uma instância do servidor remoto do JBoss EAP. Para um usuário de gerenciamento básico, insira **no**.
Um tipo de usuário que pode precisar ser adicionado ao *ManagementRealm* é o usuário representando uma outra instância do JBoss EAP, que deve ser capaz de autenticar-se para associar-se como um membro de um cluster. Se este for o caso, responda **yes** a este prompt e você receberá um valor secreto com hash representando a senha do usuário, que precisará ser adicionado a um arquivo de configuração diferente.

Os usuários também podem ser criados sem interatividade, passando os parâmetros ao script **add-user**. Esta abordagem não é recomendada nos sistemas compartilhados, pois as senhas ficarão visíveis ao fazer o log e nos arquivos de histórico. Para mais informações, consulte [Executando o Utilitário Add-User Sem Interatividade](#)

2.3.1.2. Executando o Utilitário Add-User Sem Interatividade

Você pode executar o script do **add-user** sem interatividade passando argumentos na linha de comando. No mínimo, o nome de usuário e a senha devem ser fornecidos.



ATENÇÃO

Esta abordagem não é recomendada nos sistemas compartilhados, pois as senhas ficarão visíveis ao fazer o log e nos arquivos de histórico.

Criando um Usuário Pertencendo a Múltiplos Grupos

O comando a seguir adiciona um usuário de gerenciamento, **mgmtuser1**, com os grupos **guest** e **mgmtgroup**.

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

Especificando um Arquivo de Propriedades Alternativas

Por padrão, as informações dos grupos e usuários criadas usando o script **add-user** são armazenadas nos arquivos de propriedades localizados no diretório de configuração do servidor.

As informações dos usuários são armazenadas nos seguintes arquivos de propriedades:

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

As informações dos grupos são armazenadas nos seguintes arquivos de propriedades:

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

Esses diretórios padrão e os nomes de arquivo de propriedades podem ser substituídos. O comando a seguir adiciona um novo usuário, especificando um nome e uma localização diferente para os arquivos de propriedades do usuário.

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

O novo usuário foi adicionado aos arquivos de propriedades do usuário localizados em **/path/to/standaloneconfig/newname.properties** e **/path/to/domainconfig/newname.properties**. Observe que esses arquivos já devem existir ou você encontrará um erro.

Para uma lista completa de todos os argumentos disponíveis do **add-user** e suas finalidades, use o argumento **--help** ou consulte a seção [Argumentos para o utilitário Add-User](#).

2.3.2. Interfaces de Gerenciamento

2.3.2.1. CLI de Gerenciamento

A CLI (interface de linha de comando) de gerenciamento é uma ferramenta de administração da linha de comando do JBoss EAP.

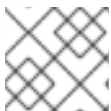
Use a CLI de gerenciamento para iniciar e encerrar servidores, implantar e cancelar implantação de aplicativos, configurar o sistema e desempenhar outras tarefas administrativas. As operações podem ser desempenhadas em modo batch, permitindo que múltiplas tarefas sejam executadas em grupo.

Vários comandos comuns do terminal estão disponíveis, tais como **ls**, **cd** e **pwd**. A CLI de gerenciamento também suporta o preenchimento automático através da tecla tab.

Para obter informações detalhadas sobre a utilização da CLI de gerenciamento, incluindo comandos e operações, sintaxe e execução em modo em lotes, consulte o [Guia da CLI de gerenciamento](#) do JBoss EAP.

Iniciando a CLI de Gerenciamento

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



NOTA

Para o Windows Server, use o script **EAP_HOME\bin\jboss-cli.bat**.

Conectando-se a um Servidor em Execução

```
connect
```

Ou você pode iniciar a CLI de gerenciamento e conectar-se em apenas uma etapa usando o comando **EAP_HOME/bin/jboss-cli.sh --connect**.

Exibindo Ajuda

Use o comando a seguir para ajuda em geral:

```
help
```

Use o sinalizador **--help** em um comando para receber instruções sobre o uso desse comando específico. Por exemplo, para receber informações sobre o uso de **deploy**, o seguinte comando é executado.

```
deploy --help
```

Encerrando a CLI de Gerenciamento

```
quit
```

Exibindo as Configurações do Sistema

O comando a seguir usa a operação **read-attribute** para especificar se a fonte de dados de exemplo está habilitada ou não.

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

■

Quando estiver executando em um domínio gerenciado, você deve especificar o perfil a ser atualizado usando `/profile=PROFILE_NAME` antes do comando.

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

Atualizando as Configurações do Sistema

O comando a seguir usa a operação **write-attribute** para desabilitar a fonte de dados de exemplo.

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

Iniciando os Servidores

A CLI de gerenciamento também pode ser usada para iniciar e encerrar os servidores quando estiver executando em um domínio gerenciado.

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

2.3.2.2. Console de Gerenciamento

O console de gerenciamento é uma ferramenta de administração baseada na web para o JBoss EAP.

Use o console de gerenciamento para iniciar e encerrar os servidores, implantar e cancelar a implantação de aplicativos, ajustar as configurações do sistema e realizar modificações persistentes à configuração do servidor. O console de gerenciamento também possui a habilidade de desempenhar tarefas administrativas, com notificações ativas quando qualquer alteração desempenhada pelo atual usuário exige que a instância do servidor seja reiniciada ou recarregada.

Em um domínio gerenciado, as instâncias do servidor e os grupos do servidor no mesmo domínio podem ser gerenciados centralmente pelo console de gerenciamento do controlador de domínio.

Para uma instância do JBoss EAP executando no host local e usando a porta de gerenciamento padrão, o console de gerenciamento pode ser acessado através de um navegador da web em <http://localhost:9990/console/App.html>. Você poderá fazer a autenticação com um usuário que tenha permissões para acessar o console de gerenciamento.

O console de gerenciamento fornece as seguintes guias para navegação e gerenciamento de seu servidor autônomo JBoss EAP ou domínio gerenciado.

Home

Aprenda como realizar várias configurações comuns e gerenciar tarefas. Faça um tour para familiarizar-se com o console de gerenciamento do JBoss EAP.

Implementações

Adicione, remova e possibilite implementações. Em um domínio gerenciado, distribua implementações para grupos de servidores.

Configuração

Configure subsistemas disponíveis que fornecem recursos como serviços web, mensagens ou alta disponibilidade. Em um domínio gerenciado, gerencie os perfis que contêm configurações de subsistemas diferentes.

Tempo de execução

Consulte informações de tempo de execução, como status do servidor, utilização de JVM e logs de servidores. Em um domínio gerenciado, gerencie seus hosts, grupos de servidores e serviços.

Controle de acesso

Atribua funções para usuários e grupos quando utilizar Controle de Acesso Baseado em Função (RBAC).

Patching

Aplique patches para as instâncias de seu JBoss EAP.



NOTA

Para fazer um tour do console de gerenciamento, clique no link **Faça um tour!** na página inicial do console de gerenciamento.

2.3.3. Arquivos de Configuração

2.3.3.1. Arquivos de Configuração de Servidor Autônomo

Os arquivos de configuração de servidor autônomo estão localizados no diretório **EAP_HOME/standalone/configuration/**. Existe um arquivo separado para cada um dos cinco perfis predefinidos (*default*, *ha*, *full*, *full-ha*, *load-balancer*).

Tabela 2.1. Arquivos de Configuração de Servidor Autônomo

Arquivo de Configuração	Finalidade
standalone.xml	Este arquivo de configuração contém a configuração padrão usada quando você inicia o seu servidor autônomo. Ele possui todas as informações sobre o servidor, incluindo subsistemas, sistemas de rede, implantações, associações e outros detalhes configuráveis. Ele não fornece os subsistemas necessários para o sistema de mensagens ou alta disponibilidade.
standalone-ha.xml	Este arquivo de configuração de servidor autônomo inclui todos os subsistemas padrão e adiciona os subsistemas mod_cluster e jgroups para alta disponibilidade. Ele não fornece os subsistemas necessários para o sistema de mensagens.
standalone-full.xml	Este arquivo de configuração de servidor autônomo inclui todos os subsistemas padrão e adiciona os subsistemas messaging-activemq e iiop-openjdk . Ele não fornece os subsistemas necessários para alta disponibilidade.
standalone-full-ha.xml	Este arquivo de configuração inclui suporte para todos os subsistemas possíveis, incluindo aqueles para o sistema de mensagens e alta disponibilidade.

Arquivo de Configuração	Finalidade
standalone-load-balancer.xml	Este arquivo de configuração de servidor autônomo inclui os subsistemas necessários mínimos para o uso do balanceador de carga front-end de mod-cluster integrado para balancear a carga de outras instâncias do JBoss EAP.

Por padrão, ao iniciar o JBoss EAP como um servidor autônomo, use o arquivo **standalone.xml**. Para iniciar o JBoss EAP com uma configuração diferente, use o argumento do **--server-config**. Por exemplo:

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

2.3.3.2. Arquivos de Configuração de Domínio Gerenciado

Os arquivos de configuração de domínio gerenciado estão localizados no diretório **EAP_HOME/domain/configuration/**.

Tabela 2.2. Arquivos de Configuração de Domínio Gerenciado

Arquivo de Configuração	Finalidade
domain.xml	Este é o principal arquivo de configuração para um domínio gerenciado. Apenas o domínio mestre é capaz de ler esse arquivo. Ele contém as configurações para todos os perfis (<i>default</i> , <i>ha</i> , <i>full</i> , <i>full-ha</i> , <i>load-balancer</i>).
host.xml	Este arquivo inclui detalhes de configuração específicos a um host físico em um domínio gerenciado, tais como interfaces de rede, associações, nome do host e outros detalhes. O arquivo host.xml inclui todos os recursos tanto do host-master.xml quanto do host-slave.xml , os quais estão descritos abaixo.
host-master.xml	Este arquivo inclui apenas os detalhes de configuração necessários para a execução de um servidor como o controlador de domínio mestre.
host-slave.xml	Este arquivo inclui apenas os detalhes de configuração necessários para a execução de um servidor como um controlador de host de domínio gerenciado.

Por padrão, ao iniciar o JBoss EAP em um domínio gerenciado use o arquivo **host.xml**. Para iniciar o JBoss EAP com uma configuração diferente, use o argumento **--host-config**. Por exemplo:

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

2.3.3.3. Fazendo o Backup dos Dados de Configuração

Para restaurar mais tarde a configuração do servidor do JBoss EAP, deve-se fazer o backup dos itens nas seguintes localizações:

- ***EAP_HOME/standalone/configuration/***
 - Faça o backup do diretório inteiro para salvar os dados do usuário, a configuração do servidor e as configurações de log para os servidores autônomos.
- ***EAP_HOME/domain/configuration/***
 - Faça o backup do diretório inteiro para salvar os dados do perfil e do usuário, a configuração do host e do domínio e as configurações de log para os domínios gerenciados.
- ***EAP_HOME/modules/***
 - Faça o backup de todos os módulos personalizados.
- ***EAP_HOME/welcome-content/***
 - Faça o backup de todos os conteúdos personalizados de boas vindas.
- ***EAP_HOME/bin/***
 - Faça o backup de todos os scripts personalizados ou dos arquivos de configuração de inicialização.

2.3.3.4. Snapshots de arquivos de configuração

Para auxiliar no gerenciamento e na manutenção do servidor, o JBoss EAP cria uma versão do arquivo de configuração original com um carimbo de data/hora no momento da inicialização. Todas as alterações de configuração adicionais realizadas pelas operações de gerenciamento obterão o backup automático do arquivo original e a preservação de uma cópia de trabalho da instância para referência e reversão. Além disto, também podem ser tirados snapshots da configuração, que são cópias pontuais da atual configuração do servidor. Esses snapshots podem ser salvos e carregados por um administrador.

Os exemplos a seguir usam o arquivo **standalone.xml**, mas o mesmo processo pode ser aplicado nos arquivos **domain.xml** e **host.xml**.

Tirando um Snapshot

Use a CLI de gerenciamento para tirar um snapshot das configurações atuais.

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

Listando os Snapshots

Use a CLI de gerenciamento para listar todos os snapshots que foram tirados.

```
:list-snapshots
{
```

```

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

```

Excluindo um Snapshot

Use a CLI de gerenciamento para excluir um snapshot.

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

Iniciando o Servidor com um Snapshot

O servidor pode ser iniciado usando um snapshot ou uma versão da configuração salva automaticamente.

1. Navegue até o diretório **EAP_HOME/standalone/configuration/standalone_xml_history** e identifique o snapshot ou o arquivo de configuração salvo a ser carregado.
2. Inicie o servidor e indique o arquivo de configuração selecionado. Forneça o caminho do arquivo relativo ao diretório de configuração, **EAP_HOME/standalone/configuration/**.

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



NOTA

Quando estiver executando em um domínio gerenciado, use o argumento **--host-config** em vez de especificar o arquivo de configuração.

2.3.3.5. Substituição de Propriedades

O JBoss EAP permite que você use expressões para definir propriedades substituíveis no lugar dos valores literais na configuração. As expressões usam o formato **\${PARAMETER:DEFAULT_VALUE}**. Se o parâmetro especificado estiver definido, então o valor do parâmetro será usado. Caso contrário, o valor padrão será usado.

As fontes compatíveis para a resolução de expressões são as propriedades do sistema, as variáveis do ambiente e o vault. Somente para implantações, as fontes podem ser as propriedades listadas em um arquivo **META-INF/jboss.properties** no arquivo de implantação. Para os tipos de implantação compatíveis com subimplantações, a resolução abrange todas as subimplantações se o arquivo de propriedades estiver na implantação externa (por exemplo, EAR). Se o arquivo de propriedades estiver na subimplantação, então a resolução abrange apenas essa subimplantação.

O exemplo abaixo do arquivo de configuração **standalone.xml** define o **inet-address** para a interface **pública** como **127.0.0.1**, a não ser que o parâmetro **jboss.bind.address** esteja definido.

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

O parâmetro **jboss.bind.address** pode ser definido iniciando o EAP como um servidor autônomo com o seguinte comando:

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

Expressões Aninhadas

As expressões podem ser aninhadas, permitindo um uso mais avançado delas no lugar dos valores fixos. O formato de uma expressão aninhada é o mesmo de uma expressão normal, com a diferença de que uma expressão é incorporada a outra, por exemplo:

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

As expressões aninhadas são avaliadas recursivamente, portanto a expressão *interna* é avaliada primeiro e, depois, a expressão *externa*. As expressões também podem ser recursivas, onde uma expressão determina a outra. As expressões aninhadas são permitidas onde expressões, em geral, são permitidas, com exceção dos comandos da CLI de gerenciamento.

Um exemplo de onde uma expressão aninhada pode ser usada é quando a senha usada em uma definição de fonte de dados está mascarada. A configuração para a fonte de dados deve ter a seguinte linha:

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

O valor de **ds_ExampleDS** poderia ser substituído por uma propriedade do sistema (**datasource_name**) usando uma expressão aninhada. A configuração para a fonte de dados poderia ter, em vez disto, a seguinte linha:

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

O JBoss EAP avaliaria primeiro a expressão **\${datasource_name}** e, então, a entraria na expressão maior e avaliaria a expressão resultante. A vantagem desta configuração é que o nome da fonte de dados é abstraído da configuração fixa.

Substituição de Propriedades Baseadas no Descritor

A configuração dos aplicativos (como os parâmetros de conexão da fonte de dados) tipicamente varia entre os ambientes de desenvolvimento, teste e produção. Essa variação é, às vezes, acomodada pelos scripts do sistema de compilação, já que a especificação Java EE não contém um método para externalizar essas configurações. Com o JBoss EAP, você pode usar a substituição de propriedades baseadas no descritor para gerenciar a configuração externamente.

A substituição das propriedades baseadas no descritor permite que você elimine suposições sobre o ambiente a partir do aplicativo e da cadeia de compilação. As configurações específicas ao ambiente podem ser especificadas nos descritores de implantação, em vez de serem especificadas nas anotações ou nos scripts do sistema de compilação. É possível fornecer a configuração nos arquivos ou como parâmetros na linha de comando.

Existem diversos sinalizadores no subsistema **EE** que controlam a aplicação da substituição de propriedades.

A substituição de descritores específicos ao JBoss é controlada pelo sinalizador **jboss-descriptor-property-replacement** e é *habilitada* por padrão. Quando habilitada, as propriedades podem ser substituídas nos seguintes descritores de implantação:

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

O comando da CLI de gerenciamento a seguir pode ser usado para habilitar ou desabilitar a substituição de propriedades nos descritores específicos ao JBoss:

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

A substituição de descritores do Java EE é controlada pelo sinalizador **spec-descriptor-property-replacement** e é *desabilitada* por padrão. Quando habilitada, as propriedades podem ser substituídas nos seguintes descritores de implantação:

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

O comando da CLI de gerenciamento a seguir pode ser usado para habilitar ou desabilitar a substituição de propriedades nos descritores do Java EE:

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

2.4. CONFIGURAÇÃO DE PORTA E REDE

2.4.1. Interfaces

O JBoss referencia as interfaces nomeadas por toda a configuração. Isto permite que a configuração reference declarações de interface individuais com nomes lógicos, em vez de exigir os detalhes completos da interface a cada uso.

Isto também permite uma configuração mais fácil em um domínio gerenciado, onde os detalhes da interface de rede podem variar em múltiplas máquinas. Cada instância do servidor pode corresponder a um grupo de nome lógico.

Todos os arquivos **standalone.xml**, **domain.xml** e **host.xml** incluem declarações de interface. Há diversos nomes de interfaces pré-configurados, dependendo de qual

configuração padrão é usada. A interface de **gerenciamento** pode ser utilizada para todos os componentes e serviços que exigem a camada de gerenciamento, incluindo o endpoint de gerenciamento HTTP. A interface **pública** pode ser utilizada para toda comunicação de rede relacionada ao aplicativo. A interface **não segura** é usada para soquetes IIOP na configuração padrão. A interface **privada** é utilizada por soquetes JGroups na configuração padrão.

2.4.1.1. Configurações de Interface Padrão

```
<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>
```

Por padrão, o JBoss EAP vincula essas interfaces com **127.0.0.1**, mas esses valores podem ser substituídos durante o tempo de execução ao definir a propriedade apropriada. Por exemplo, o **inet-address** da interface **pública** pode ser definido ao iniciar o JBoss EAP como um servidor autônomo com o comando a seguir:

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

Alternativamente, você pode utilizar opção **-b** na linha de comando de inicialização do servidor. Para mais informações sobre opções de inicialização de servidor, consulte [Argumentos de tempo de execução do servidor](#).



IMPORTANTE

Se você modificar as portas ou interfaces de rede padrão usadas pelo JBoss EAP, lembre-se de alterar também os scripts que usam as portas ou interfaces modificadas, que incluem os scripts de serviço do JBoss EAP. Lembre-se também de especificar a porta e a interface corretas ao acessar o console de gerenciamento ou a CLI de gerenciamento.

2.4.1.2. Configurando Interfaces

As interfaces de rede são declaradas ao especificar um nome lógico e os critérios de seleção para a interface física. Os critérios de seleção podem referenciar um endereço curinga ou especificar um conjunto de uma ou mais características que uma interface ou um endereço deve ter para obter uma correspondência válida. Para uma lista de todos os critérios de seleção da interface disponíveis, consulte a seção [Atributos de Interface](#).

As interfaces podem ser configuradas usando o console de gerenciamento ou a CLI de gerenciamento. Seguem abaixo alguns exemplos de adição e atualização de interfaces. O comando da CLI de gerenciamento é apresentado primeiro, seguido pela configuração XML correspondente.

Adicionando uma Interface com um Valor NIC

Adicione uma nova interface com o valor NIC de **eth0**.

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

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

Adicionando uma Interface com Diversos Valores Condicionais

Adicione uma nova interface que corresponda a qualquer interface ou endereço na sub-rede; se a interface estiver ativa, suportar multicast e não ser ponto a ponto.

```
/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>
```

Atualizando um Atributo da Interface

Atualize o valor **inet-address** padrão da interface **pública**, mantendo a propriedade **jboss.bind.address** para permitir que este valor seja definido durante o tempo de execução.

```
/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>
```

Adicionando uma Interface a um Servidor em um Domínio Gerenciado

```
/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. Associações de Soquete

As associações de soquete e os grupos de associação de soquete permitem que você defina as portas de rede e suas relações para as interfaces de rede necessárias para a sua configuração do JBoss EAP. Uma associação de soquete é uma configuração nomeada para um soquete. Um grupo de associação de soquete é uma coleção das declarações da associação de soquete que são agrupadas sob um nome lógico.

Isto permite que outras seções da configuração façam referência às associações de soquete pelos seus nomes lógicos, em vez de exigir os detalhes completos da configuração do soquete a cada uso.

As declarações para essas configurações nomeadas podem ser encontradas nos arquivos de configuração **standalone.xml** e **domain.xml**. Um servidor autônomo contém apenas um grupo de associação de soquete, enquanto um domínio gerenciado pode conter múltiplos grupos. Você pode criar um grupo de associação de soquete para cada grupo do servidor no domínio gerenciado ou compartilhar um grupo de associação de soquete entre os múltiplos grupos do servidor.

As portas usadas por padrão pelo JBoss EAP dependem dos grupos de associação de soquete que são usados e dos requisitos das suas implantações individuais.

2.4.2.1. Portas de Gerenciamento

As portas de gerenciamento foram consolidadas no JBoss EAP 7. Por padrão, o JBoss EAP 7 usa a porta **9990** para o gerenciamento nativo, usado pela CLI de gerenciamento, e para o gerenciamento HTTP, usado pelo console de gerenciamento baseado na web. A porta **9999**, que foi usada como a porta de gerenciamento nativo no JBoss EAP 6, não é mais usada, mas ainda pode ser habilitada se desejado.

Se o HTTPS é habilitado para o console de gerenciamento, então a porta **9993** é usada por padrão.

2.4.2.2. Associações de Soquete Padrão

O JBoss EAP é fornecido com um grupo de associação de soquete para cada um dos cinco perfis predefinidos (*default*, *ha*, *full*, *full-ha*, *load-balancer*).

Para mais informações sobre as associações de soquete padrão, tais como descrições e portas padrão, consulte a seção [Associações de Soquete Padrão](#).



IMPORTANTE

Se você modificar as portas ou interfaces de rede padrão usadas pelo JBoss EAP, lembre-se de alterar também os scripts que usam as portas ou interfaces modificadas, que incluem os scripts de serviço do JBoss EAP. Lembre-se também de especificar a porta e a interface corretas ao acessar o console de gerenciamento ou a CLI de gerenciamento.

Servidor Autônomo

Quando estiver executando como um servidor autônomo, apenas um grupo de associação de soquete é definido por arquivo de configuração. Cada arquivo de configuração de servidor autônomo (**standalone.xml**, **standalone-ha.xml**, **standalone-full.xml**, **standalone-full-ha.xml**, **standalone-load-balancer.xml**) define as associações de soquete para as tecnologias usadas por seu perfil correspondente.

Por exemplo, o arquivo de configuração de servidor autônomo padrão (**standalone.xml**) especifica as associações de soquete abaixo:

```
<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>
```

Domínio Gerenciado

Quando estiver executando em um domínio gerenciado, todos os grupos de associação de soquete são definidos no arquivo **domain.xml**. Existem cinco grupos de associação de soquete predefinidos:

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

Cada grupo de associação de soquete especifica as associações de soquete para as tecnologias usadas por seu perfil correspondente. Por exemplo, o grupo de associação de soquete **full-ha-sockets** define várias associações de soquete **jgroups**, as quais são usadas pelo perfil *full-ha* para alta disponibilidade.

```
<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>

```



NOTA

A configuração de soquete para as interfaces de gerenciamento é definida no arquivo **host.xml** do controlador de domínio.

2.4.2.3. Configurando as Associações de Soquete

Quando estiver definindo uma associação de soquete, você pode definir os atributos de **porta** e de **interface**, assim como as configurações de multicast, com **multicast-address** e **multicast-port**. Para detalhes sobre todos os atributos da associação de soquete disponíveis, consulte a seção [Atributos da Associação de Soquete](#).

As associações de soquete podem ser configuradas usando o console de gerenciamento ou a CLI de gerenciamento. As etapas a seguir passam pela adição de um grupo de associação de soquete, adição de uma associação de soquete e definição das configurações da associação de soquete usando a CLI de gerenciamento.

1. Adicione um novo grupo de associação de soquete. Observe que esta etapa não pode ser desempenhada quando executando como um servidor autônomo.

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

2. Adicione uma associação de soquete.

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

3. Altere a associação de soquete para usar uma interface que não seja a padrão, definida pelo grupo de associação de soquete.

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

O exemplo a seguir mostra como a configuração XML pode ficar depois que as etapas acima são concluídas.

```
<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. Deslocamentos de Porta

Um deslocamento de porta é um valor numérico de deslocamento adicionado a todos os valores de porta especificados no grupo de associação de soquete para aquele servidor. Isto permite que o servidor herde os valores de porta definidos no seu grupo de associação de soquete, com um deslocamento para garantir que não haja conflito entre os outros servidores no mesmo host. Por exemplo, se a porta HTTP do grupo de associação de soquete é **8080** e um servidor usa um deslocamento de porta de **100**, então a sua porta HTTP será **8180**.

Segue abaixo um exemplo de configuração de um deslocamento de porta de **250** para um servidor em um domínio gerenciado usando a CLI de gerenciamento.

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

Os deslocamentos de porta podem ser usados para servidores em um domínio gerenciado e para a execução de múltiplos servidores autônomos em um mesmo host.

Você pode passar em um deslocamento de porta quando iniciar um servidor autônomo utilizando a propriedade **jboss.socket.binding.port-offset**

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

2.4.3. Endereços IPv6

Por padrão, o JBoss EAP é configurado para executar usando endereços IPv4. As etapas abaixo mostram como configurar o JBoss EAP para executar usando endereços IPv6.

Configurando a Pilha JVM para Endereços IPv6

Atualize a configuração de inicialização para usar endereços IPv6.

1. Abra o arquivo de configuração de inicialização.

- Quando estiver executando como um servidor autônomo, edite o arquivo **EAP_HOME/bin/standalone.conf** (ou **standalone.conf.bat** para o Windows Server).
- Quando estiver executando em um domínio gerenciado, edite o arquivo **EAP_HOME/bin/domain.conf** (ou **domain.conf.bat** para o Windows Server).

2. Defina a propriedade **java.net.preferIPv4Stack** como **false**.

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

3. Acrescente a propriedade **java.net.preferIPv6Addresses** e a defina como **true**.

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

O exemplo a seguir mostra como as opções JVM no arquivo de configuração de inicialização podem ficar depois que as alterações acima são aplicadas.

```
# 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
```

Atualizando Declarações de Interface para Endereços IPv6

Os valores de interface padrão na configuração podem ser mudados para endereços IPv6. Por exemplo, o comando da CLI de gerenciamento abaixo define a interface de **gerenciamento** como endereço IPv6 de loopback (::1).

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

O exemplo a seguir mostra como a configuração XML deve ficar depois que o comando acima é executado.

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

CAPÍTULO 3. DESENVOLVENDO APLICATIVOS USANDO O JBOSS EAP

3.1. VISÃO GERAL

Este guia fornece informações sobre o desenvolvimento de aplicativos usando o Red Hat JBoss Developer Studio e exemplos de início rápido do JBoss EAP 7.

O Red Hat JBoss Developer Studio é um ambiente de desenvolvimento integrado (IDE) baseado em Eclipse que integra plug-ins do JBoss para o desenvolvimento de aplicativos. O JBoss Developer Studio pode ajudar no desenvolvimento de seu aplicativo com seus assistentes específicos ao JBoss e a habilidade de implantar aplicativos nos servidores do JBoss. Muitos exemplos de código de início rápido são fornecidos com o JBoss EAP 7 para auxiliar os usuários a começar a escrever os aplicativos usando diferentes tecnologias Java EE 7.

3.2. CONFIGURANDO O AMBIENTE DE DESENVOLVIMENTO

É recomendável usar o JBoss Developer Studio 11.0 ou posterior com o JBoss EAP 7.1.

1. Faça download e instale o JBoss Developer Studio.
Para obter instruções, consulte [Instalação do JBoss Developer Studio autônomo usando o instalador](#) no *Guia de instalação* do JBoss Developer Studio.
2. Configure o servidor do JBoss EAP no JBoss Developer Studio.
Para obter instruções, consulte [Uso de detecção de tempo de execução para configurar o JBoss EAP de dentro do IDE](#) no guia *Introdução às ferramentas do JBoss Developer Studio*.

3.3. USANDO OS EXEMPLOS DE INÍCIO RÁPIDO

Os exemplos de início rápido fornecidos com o JBoss EAP são projetos Maven.

3.3.1. Sobre o Maven

O Apache Maven é uma ferramenta de automação de compilação distribuída, usada no desenvolvimento de aplicativos Java para criar, gerenciar e compilar projetos de software. O Maven utiliza arquivos de configuração padrão chamados POM (Modelo de Objeto do Projeto) para definir os projetos e gerenciar o processo de compilação. Os arquivos POM descrevem o módulo e as dependências dos componentes, a ordem de compilação e as metas para a saída e o empacotamento do projeto resultante usando um arquivo XML. Isto garante que o projeto seja compilado de forma correta e uniforme.

O Maven é capaz de atingir isto usando um repositório. Um repositório do Maven armazena bibliotecas Java, plug-ins e outros artefatos de compilação. O repositório público padrão é o [Repositório Central do Maven 2](#), mas os repositórios podem ser privados e internos dentro de uma empresa com o objetivo de compartilhar artefatos comuns entre as equipes de desenvolvimento. Os repositórios também estão disponíveis através de terceiros. Para mais informações, consulte o projeto [Apache Maven](#) e o guia [Introduction to Repositories](#).

O JBoss EAP inclui um repositório do Maven que contém muitos dos requisitos que os desenvolvedores Java EE geralmente usam para compilar aplicativos no JBoss EAP.

Para obter mais informações sobre como usar o Maven com o JBoss EAP, consulte [Uso do Maven com o JBoss EAP](#) no *Guia de desenvolvimento* do JBoss EAP.

3.3.2. Usando Maven com os Inícios Rápidos

As dependências e os artefatos necessários para compilar e implantar aplicativos no JBoss EAP 7 são hospedados em um repositório público. Com o JBoss EAP 7, não é mais necessário configurar o arquivo do Maven `settings.xml` para usar esses repositórios durante a compilação dos inícios rápidos. Os repositórios do Maven agora são configurados nos arquivos POM dos projetos de início rápido. Este método de configuração é fornecido para facilitar a introdução dos inícios rápidos, no entanto, não é geralmente recomendado para projetos de produção, pois pode deixar a sua compilação mais lenta.

O Red Hat JBoss Developer Studio inclui o Maven, portanto não há necessidade de baixá-lo e instalá-lo separadamente. É recomendável usar o JBoss Developer Studio versão 11.0 ou posterior.

Se você planeja usar a linha de comando do Maven para compilar e implantar seus aplicativos, então você deve baixar primeiro o Maven através do projeto [Apache Maven](#) e instalá-lo seguindo as instruções fornecidas na documentação do Maven.

3.3.3. Baixando e Executando os Inícios Rápidos

3.3.3.1. Baixando os Inícios Rápidos

O JBoss EAP vem com um conjunto completo de exemplos de código de início rápido criado para ajudar os usuários a começar a escrever aplicativos usando as várias tecnologias Java EE 7. Os inícios rápidos podem ser baixados a partir do Portal do Cliente Red Hat.

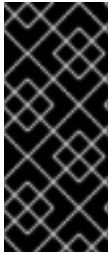
1. Faça o login no [Portal do Cliente Red Hat](#).
2. Clique em **Downloads**.
3. Na lista **Downloads de Produtos**, clique em **Red Hat JBoss Enterprise Application Platform**.
4. Selecione **7.1** no menu suspenso **Versão**.
5. Encontre a entrada **Red Hat JBoss Enterprise Application Platform 7.1.0 Quickstarts** na tabela e clique em **Download**.
6. Salve o arquivo ZIP no diretório desejado.
7. Extraia o arquivo ZIP.

3.3.3.2. Executando os Inícios Rápidos no JBoss Developer Studio

Depois que os inícios rápidos foram baixados, eles podem ser importados no JBoss Developer Studio e implantados no JBoss EAP.

Importando um Início Rápido para o JBoss Developer Studio

Cada início rápido é fornecido com um arquivo POM que contém o seu projeto e informações de configuração. Use este arquivo POM para importar com facilidade o início rápido para o Red Hat JBoss Developer Studio.

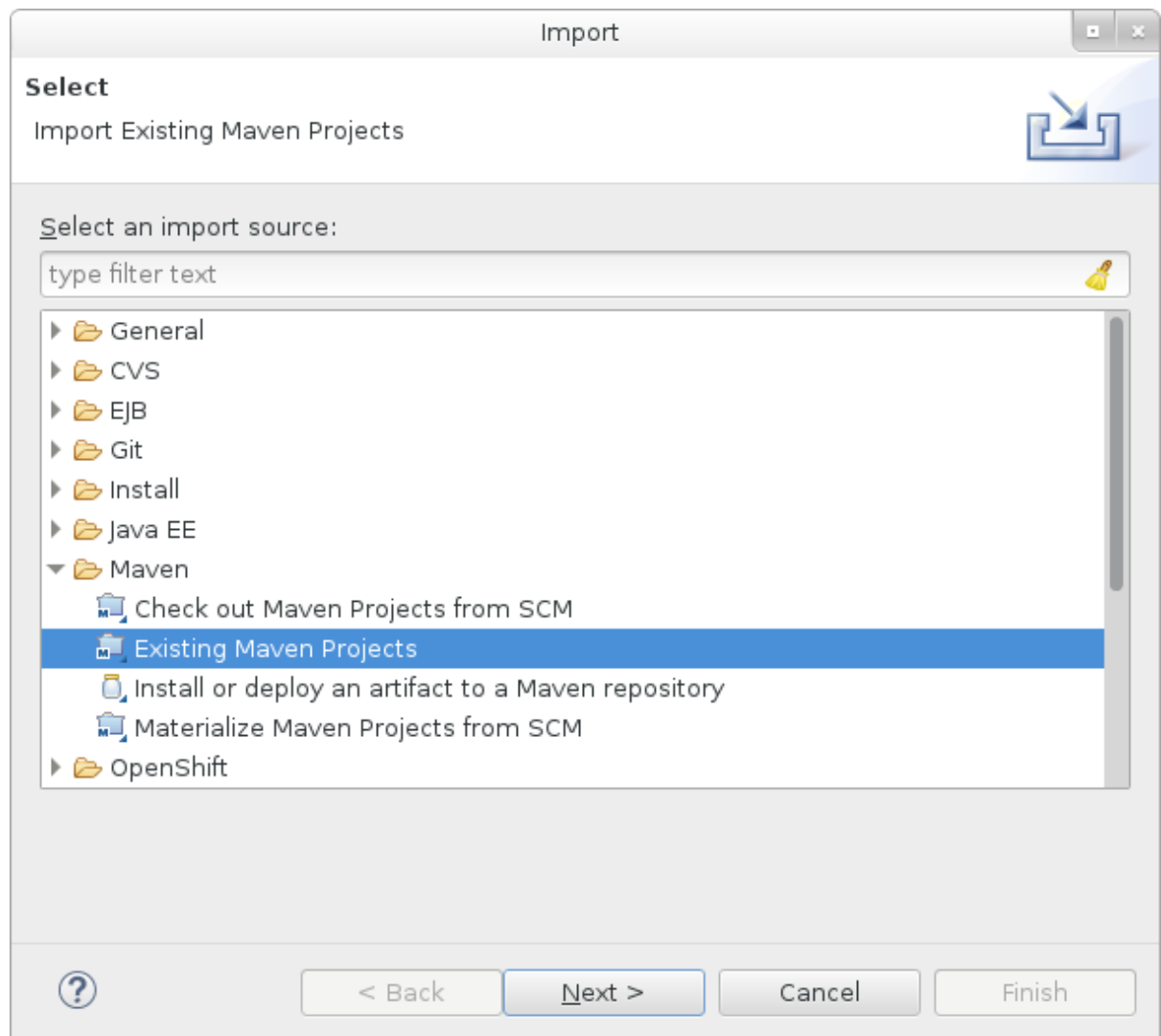


IMPORTANTE

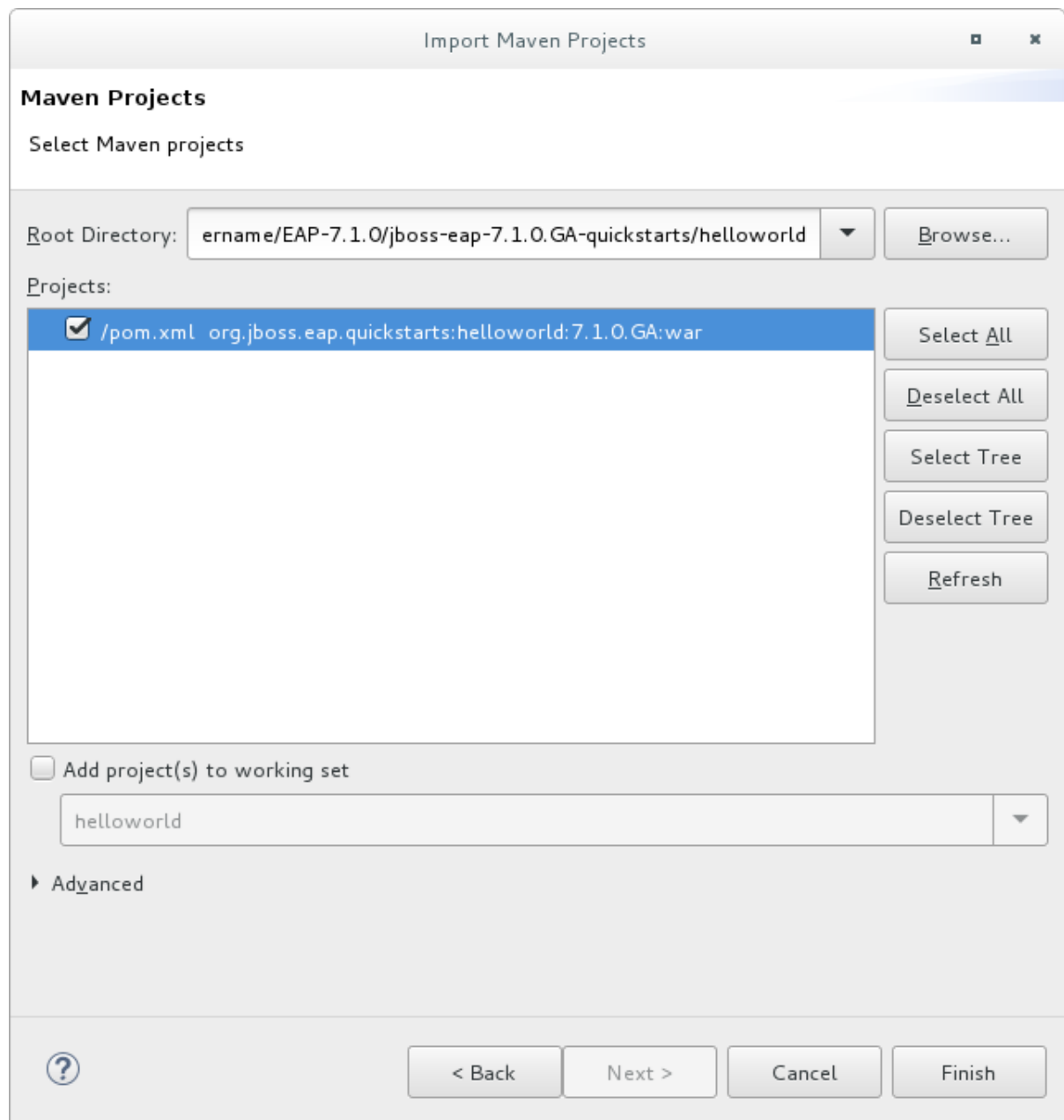
Caso a sua pasta do projeto de início rápido esteja localizada no espaço de trabalho do IDE, quando você importá-la no Red Hat JBoss Developer Studio, o IDE gera um nome de arquivo WAR e um nome de projeto inválido. Certifique-se antes de que a sua pasta do projeto de início rápido esteja localizada fora do espaço de trabalho do IDE.

1. Inicie o JBoss Developer Studio.
2. Selecione **Arquivo** → **Importar**.
3. Selecione **Maven** → **Projetos Maven Existentes** e clique em **Next**.

Figura 3.1. Importando Projetos Maven Existentes



4. Navegue até o diretório do início rápido desejado (por exemplo, o início rápido **helloworld**) e clique em **OK**. A caixa de listagem dos **Projetos** é preenchida pelo arquivo **pom.xml** do projeto de início rápido selecionado.

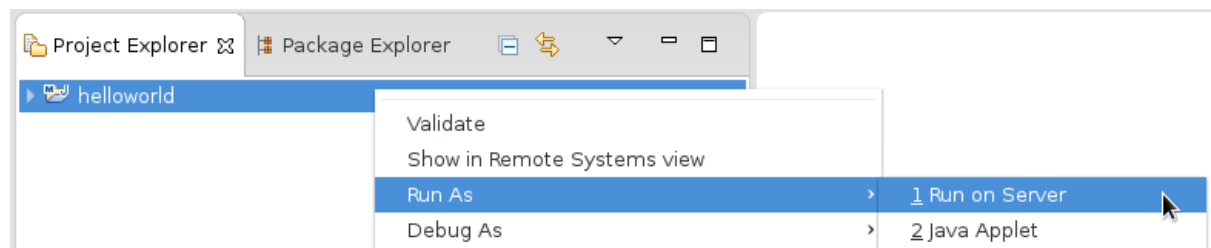
Figura 3.2. Selecionando Projetos Maven

5. Clique em **Concluir**.

Executando o Início Rápido *helloworld*

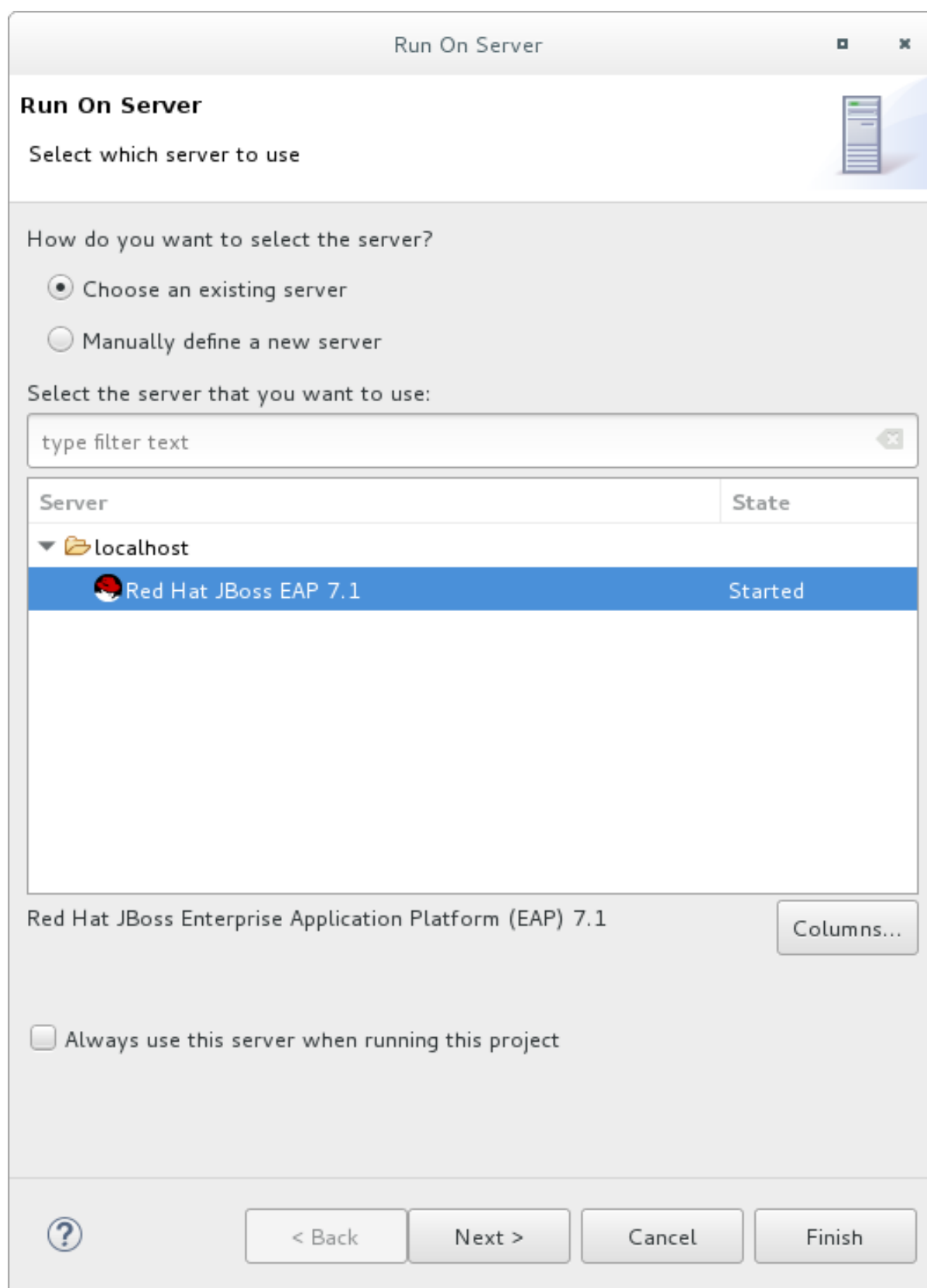
A execução do início rápido **helloworld** é uma maneira simples de verificar se o servidor do JBoss EAP está configurado e executando corretamente.

1. Se você ainda não definiu um servidor, adicione o servidor do JBoss EAP ao JBoss Developer Studio. Consulte [Uso da detecção de tempo de detecção para configurar o JBoss EAP de dentro do IDE](#) no guia *Introdução às ferramentas do JBoss Developer Studio*.
2. Clique com o botão direito do mouse no projeto **helloworld** na guia **Explorador de projeto** e selecione **Executar como** → **Executar no servidor**.

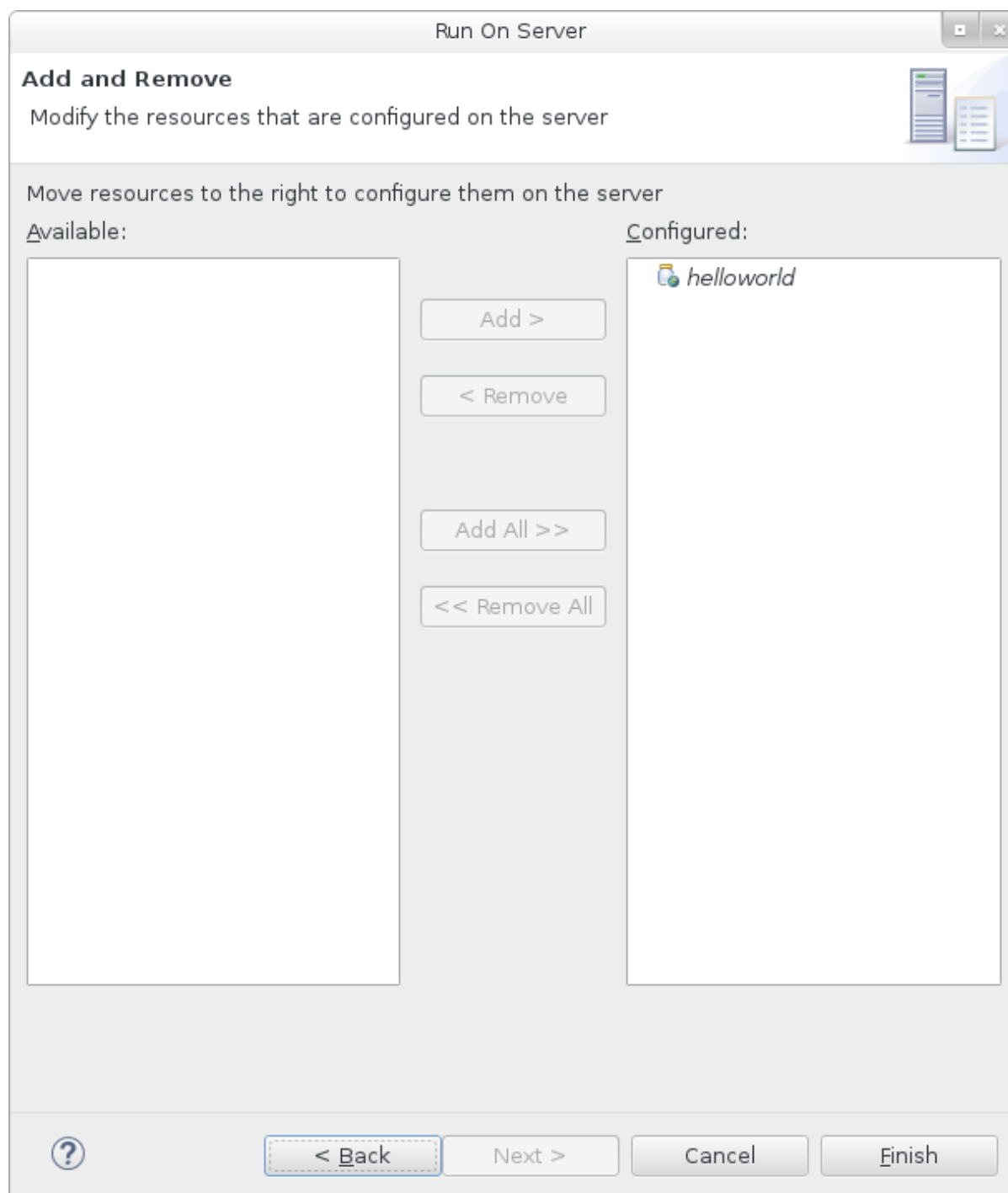
Figura 3.3. Executar Como - Executar no Servidor

3. Selecione o servidor do JBoss EAP 7.1 na lista de servidores e clique em **Próximo**.

Figura 3.4. Executar no Servidor



4. O início rápido **helloworld** já está listado para ser configurado no servidor. Clique em **Concluir** para implantar o início rápido.

Figura 3.5. Modificando os Recursos Configurados no Servidor

5. Verifique os resultados.

- Na guia **Servidor**, o status do servidor do JBoss EAP 7.1 muda para **Iniciado**.
- A guia **Console** mostra as mensagens detalhando a inicialização do servidor do JBoss EAP e a implantação do início rápido **helloworld**.

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

- O aplicativo **helloworld** está disponível em <http://localhost:8080/helloworld> e exibe o texto **Hello World!**.

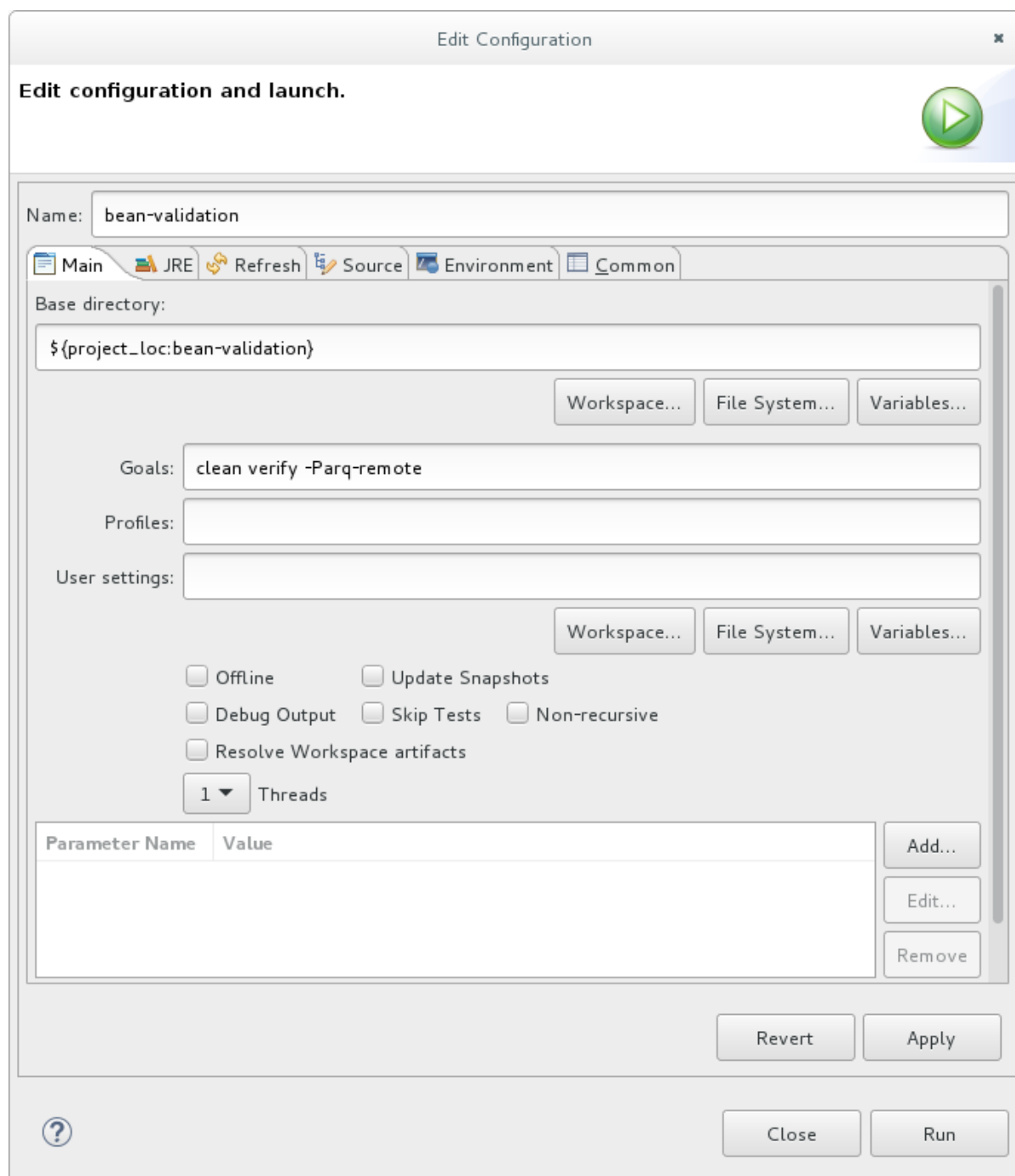
Para obter detalhes adicionais sobre o início rápido **helloworld**, consulte [Exploração do início rápido helloworld](#).

Executando o Início Rápido *bean-validation*

Alguns inícios rápidos, tais como o **bean-validation**, não fornecem a camada da interface do usuário e, em vez disto, fornecem testes Arquillian para demonstrar funcionalidade.

1. Importe o início rápido **bean-validation** no JBoss Developer Studio.
2. Na guia **Servidores**, clique com o botão direito no servidor e escolha **Iniciar** para iniciar o servidor do JBoss EAP. Se você não encontrar uma guia **Servidores** ou ainda não tiver definido um servidor, adicione o servidor do JBoss EAP ao JBoss Developer Studio. Consulte [Uso de detecção de tempo de execução para configurar o JBoss EAP de dentro do IDE](#) no guia *Introdução às ferramentas do JBoss Developer Studio*.
3. Clique com o botão direito do mouse no projeto **jboss-bean-validation** na guia **Explorador de projeto** e selecione **Executar como** → **Compilação do Maven**.
4. Insira o seguinte no campo de entrada **Metas** e clique em **executar**.

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

Figura 3.6. Editando a Configuração

5. Verifique os resultados.

A guia **Console** mostra os resultados dos testes Arquillian do **bean-validation**:

```

-----
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. Executando os Inícios Rápidos através da Linha de Comando

Você pode compilar e implantar facilmente os inícios rápidos através da linha de comando usando o Maven. Se você ainda não instalou o Maven, consulte o projeto [Apache Maven](#) para baixá-lo e instalá-lo.

O arquivo **README.md** é fornecido no diretório raiz dos inícios rápidos e contém informações gerais sobre as exigências do sistema, configuração do Maven, adição de usuários e execução de inícios rápidos.

Cada início rápido também contém o seu próprio arquivo **README.md** que fornece instruções específicas e comandos do Maven para executar esse início rápido.

Executando o Início Rápido *helloworld* através da Linha de Comando

1. Revise o arquivo **README.md** no diretório raiz do início rápido *helloworld*.
2. Inicie o servidor do JBoss EAP.

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

3. Navegue até o diretório de início rápido *helloworld*.
4. Compile e implante o início rápido usando o comando do Maven fornecido no arquivo **README.md** do início rápido.

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

5. O aplicativo *helloworld* está disponível agora em <http://localhost:8080/helloworld> e exibe o texto **Hello World!**.

3.4. REVISE OS EXEMPLOS DE INÍCIO RÁPIDO

3.4.1. Explorando a Inicialização Rápida *helloworld*

O início rápido **helloworld** mostra como implantar um servlet simples no JBoss EAP. A lógica comercial é encapsulada em um serviço, que é fornecido como um bean de contextos e injeção de dependência (CDI) e injetado no Servlet. Esse início rápido é um ponto de partida para garantir que você configurou e iniciou o servidor adequadamente.

Instruções detalhadas para compilação e implantação desse início rápido usando a linha de comando podem ser encontradas no arquivo **README.html** na raiz do diretório do início rápido **helloworld**. Este tópico mostra como usar o Red Hat JBoss Developer Studio para executar o início rápido e presume que você instalou o Red Hat JBoss Developer Studio, configurou o Maven e importou e executou com êxito o início rápido **helloworld**.

Pré-requisitos

- Instale o JBoss Developer Studio. Para obter instruções, consulte [Instalação do JBoss Developer Studio autônomo usando o instalador](#) no *Guia de instalação* do JBoss Developer Studio.
- Execute o início rápido **helloworld**. Para obter instruções, consulte [Execução de inícios rápidos no JBoss Developer Studio](#).
- Verifique se o início rápido **helloworld** foi implantado com êxito no JBoss EAP abrindo um navegador e acessando o aplicativo em <http://localhost:8080/helloworld>.

Examine a Estrutura do Diretório

O código do início rápido **helloworld** pode ser encontrado no diretório **QUICKSTART_HOME/helloworld/**. O início rápido **helloworld** é composto por um Servlet e um bean CDI. Ele também contém um arquivo **beans.xml** no diretório **WEB-INF/** do aplicativo que tem um número de versão 1.1 e um **bean-discovery-mode** de **all**. Esse arquivo de marcador identifica o WAR como um arquivo morto bean e informa ao JBoss EAP para procurar por beans no aplicativo e ativar a CDI.

O diretório **src/main/webapp/** contém os arquivos do início rápido. Todos os arquivos de configuração deste exemplo estão localizados no diretório **WEB-INF/** em **src/main/webapp/**, inclusive o arquivo **beans.xml**. O diretório **src/main/webapp/** também inclui um arquivo **index.html**, que usa uma atualização meta simples para redirecionar o navegador do usuário para o Servlet, que está localizado em <http://localhost:8080/helloworld/HelloWorld>. O início rápido não exige um arquivo **web.xml**.

Examine o Código

A declaração e importações do pacote foram excluídas destas listagens. A listagem completa está disponível no código de fonte da inicialização rápida.

1. Revise o código do **HelloWorldServlet**.

O arquivo **HelloWorldServlet.java** está localizado no diretório **src/main/java/org/jboss/as/quickstarts/helloworld/**. Este servlet envia as informações ao navegador.

Exemplo: código de classe do HelloWorldServlet

```

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();
58         writer.println(PAGE_HEADER);
59         writer.println("<h1>" +
helloService.createHelloMessage("World") + "</h1>");

```



```

59         writer.println(PAGE_FOOTER);
60         writer.close();
61     }
62
63 }

```

Tabela 3.1. Detalhes do HelloWorldServlet

Linha	Observação
43	Basta adicionar a anotação @WebServlet e fornecer um mapeamento para um URL usado para acessar o servlet.
46-48	Cada página da web precisa HTML corretamente formado. Essa inicialização rápida usa Sequências estáticas para gravar o resultado de rodapé e cabeçalho mínimo.
50-51	Estas linhas injetam o bean CDI do HelloService que gera a mensagem real. Contanto que a API do HelloService não seja alterada, esta abordagem permite alterar a implementação do HelloService em uma data posterior sem modificar a camada de exibição.
58	Esta linha chama o serviço para gerar a mensagem "Hello World" e gravá-la à solicitação HTTP.

2. Revise o código do **HelloService**.

O arquivo **HelloService.java** está localizado no diretório **src/main/java/org/jboss/as/quickstarts/helloworld/**. Este serviço simplesmente retorna uma mensagem. Não é necessário nenhum XML ou registro de anotação.

Exemplo: código de classe do HelloService

```

public class HelloService {

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

}

```

3.4.2. Explorando a Inicialização Rápida do **numberguess** (adivinhação de número)

O início rápido **numberguess** mostra como criar e implantar um aplicativo não persistente simples no JBoss EAP. As informações são exibidas usando uma exibição JSF e a lógica comercial é encapsulada em dois beans CDI. No início rápido **numberguess**, você tem dez tentativas para adivinhar um número entre 1 e 100. Depois de cada tentativa, você será informado se o número foi muito alto ou muito baixo.

O código do início rápido **numberguess** pode ser encontrado no diretório **QUICKSTART_HOME/numberguess/**, em que **QUICKSTART_HOME** é o diretório em que você

baixou e descompactou os inícios rápidos do JBoss EAP. O início rápido **numberguess** é composto por alguns beans, arquivos de configuração e exibições Facelets (JSF) e é empacotado como um módulo WAR.

Instruções detalhadas para compilação e implantação desse início rápido usando a linha de comando podem ser encontradas no arquivo **README.html** na raiz do diretório do início rápido **numberguess**. Os exemplos a seguir usam o Red Hat JBoss Developer Studio para executar o início rápido.

Pré-requisitos

- Instale o JBoss Developer Studio. Para obter instruções, consulte [Instalação do JBoss Developer Studio autônomo usando o instalador](#) no *Guia de instalação* do JBoss Developer Studio.
- Execute o início rápido **numberguess**. Para obter instruções, consulte [Execução de inícios rápidos no JBoss Developer Studio](#) e substitua **helloworld** por **numberguess** nas instruções.
- Verifique se o início rápido **numberguess** foi implantado com êxito no JBoss EAP abrindo um navegador e acessando o aplicativo neste URL:
<http://localhost:8080/numberguess>.

Examinando os Arquivos de Configuração

Todos os arquivos de configuração deste exemplo estão localizados no diretório **QUICKSTART_HOME/numberguess/src/main/webapp/WEB-INF/** do início rápido.

1. Examine o arquivo **faces-config.xml**.

Este início rápido usa a versão JSF 2.2 do nome de arquivo **faces-config.xml**. Uma versão padronizada do Facelets é o manipulador de exibição padrão no JSF 2.2 e, portanto, não requer configuração. Esse arquivo consiste em apenas o elemento raiz e é simplesmente um arquivo de marcador para indicar que o JSF deve ser habilitado no aplicativo.

```
<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. Examine o arquivo **beans.xml**.

O arquivo **beans.xml** contém um número de versão 1.1 e um **bean-discovery-mode** de **all**. Esse arquivo é um arquivo de marcador que identifica o WAR como um arquivo morto de bean e informa ao JBoss EAP para procurar por beans no aplicativo e ativar a CDI.

```
<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>
```

**NOTA**

Este início rápido não precisa de um arquivo **web.xml**.

3.4.2.1. Examinando o código JSF

O JSF usa a extensão de arquivo **.xhtml** para arquivos de fonte, mas fornece as exibições renderizadas com a extensão **.jsf**. O arquivo **home.xhtml** está localizado no diretório **src/main/webapp/**.

Exemplo: código-fonte do JSF

```

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 </head>
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}"
57   required="true" size="3"
58   disabled="#{game.number eq game.guess}"
59   validator="#{game.validateNumberRange}" />
60 <h:commandButton id="guessButton" value="Guess"
61   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>

```

Os seguintes números de linha correspondem àqueles vistos ao visualizar o arquivo no JBoss Developer Studio.

Tabela 3.2. Detalhes do JSF

Linha	Observação
36-40	Essas são as mensagens que podem ser enviadas ao usuário: "Maior!" e "Menor!"
45-48	Assim que o usuário vai adivinhando, a variedade de número de número que ele pode adivinhar vai reduzindo. Esta sentença é alterada para certificação de que um usuário sabe a variedade de número de uma adivinhação válida.
55-58	Este campo de entrada é vinculado a uma propriedade de bean usando uma expressão de valor.
58	Uma associação de validador é usada para garantir que o usuário não informe acidentalmente um número fora do intervalo para adivinhação. Se o validador não estivesse presente, o usuário poderia desperdiçar uma tentativa com um número fora dos limites.
59-61	Deve haver uma maneira do usuário enviar sua tentativa de adivinhação ao servidor. Aqui nós vinculamos um método de ação no bean.

3.4.2.2. Examinação nos Arquivos de Classe

Todos os arquivos de fonte do início rápido **numberguess** podem ser encontrados no diretório

QUICKSTART_HOME/numberguess/src/main/java/org/jboss/as/quickstarts/numberguess/. A declaração e as importações do pacote foram excluídas dessas listagens. A listagem completa está disponível no código-fonte do início rápido.

1. Revise o código do qualificador **Random.java**

Um qualificador é usado para remover a ambiguidade entre dois beans, ambos elegíveis para injeção com base no tipo. Para obter mais informações sobre

qualificadores, consulte [Uso de um qualificador para resolver uma injeção ambígua](#) no *Guia de desenvolvimento* do JBoss EAP. O qualificador **@Random** é usado para injetar um número aleatório.

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

}
```

2. Revise o código do qualificador **MaxNumber.java**

O **qualificador @MaxNumber** é usado para injetar o número máximo permitido.

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

}
```

3. Revise o código de **Generator.java**

A classe **Generator** cria um número aleatório por meio de um método produtor, expondo o número máximo possível por meio dele. Essa classe está no escopo do aplicativo, então você não obterá um aleatório diferente a cada vez.

```
@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. Revise o código de **Game.java**

A classe no escopo de sessão **Game** é o ponto de entrada primário do aplicativo. É responsável por configurar ou redefinir o jogo, capturar e validar a adivinhação do usuário e fornecer feedback para o usuário com um **FacesMessage**. Ela usa o método de ciclo de vida pós constructo para inicializar o jogo recuperando um número aleatório do bean **@Random Instance<Integer>**.

Observe a anotação **@Named** na classe. Essa anotação só é exigida quando você desejar tornar o bean acessível a uma exibição JSF usando a linguagem de expressão (EL), neste caso, **#{game}**.

```
@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
 * {@link PlainPostConstruct @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);
        }
    }
}
```


APÊNDICE A. MATERIAL DE REFERÊNCIA

A.1. ARGUMENTOS DE TEMPO DE EXECUÇÃO DO SERVIDOR

O script de inicialização do servidor do aplicativo aceita argumentos e opções. Isto permite que o servidor inicialize sob configurações alternativas em relação àquelas definidas nos arquivos de configuração **standalone.xml**, **domain.xml** e **host.xml**.

As configurações alternativas podem incluir a inicialização do servidor com um conjunto alternativo de associações de soquete ou uma configuração secundária.

A lista dos parâmetros disponíveis pode ser acessada usando a opção de ajuda **-h** ou **--help** durante a inicialização.

Tabela A.1. Argumentos e Opções em Tempo de Execução

Argumentos ou Opções	Modo de Operação	Descrição
--admin-only	Autônomo	Define o tipo de execução do servidor como ADMIN_ONLY . Isso fará com que ele abra interfaces administrativas e aceite solicitações de gerenciamento, mas ele não inicializará outros serviços de tempo de execução nem aceitará solicitações do usuário final. Observe que é recomendável usar --start-mode=admin-only como alternativa.
--admin-only	Domínio	Define o tipo de execução do controlador do host como ADMIN_ONLY , abrindo interfaces administrativas e aceitando solicitações de gerenciamento, mas não inicializa servidores ou aceita conexões de entrada de controladores do host subordinados, se esse controlador do host for o mestre do domínio.
-b=<value>, -b <value>	Autônomo, Domínio	Define a propriedade do sistema jboss.bind.address , que é usada na configuração do endereço de associação para a interface pública. Isso torna 127.0.0.1 o padrão se nenhum valor estiver especificado. Consulte a entrada -b<interface>=<value> para a configuração do endereço de associação de outras interfaces.
-b<interface>=<value>	Autônomo, Domínio	Define a propriedade do sistema jboss.bind.address.<interface> com o valor especificado. Por exemplo, -bmanagement=IP_ADDRESS
--backup	Domínio	Mantém uma cópia da configuração do domínio persistente mesmo se este host não for o controlador de domínio.

Argumentos ou Opções	Modo de Operação	Descrição
-c=<config>, -c <config>	Autônomo	Nome do arquivo de configuração do servidor para uso. O padrão é standalone.xml .
-c=<config>, -c <config>	Domínio	Nome do arquivo de configuração do servidor para uso. O padrão é domain.xml .
--cached-dc	Domínio	Se o host não é o controlador de domínio e não pode contactá-lo durante a inicialização, inicie usando uma cópia armazenada em cache localmente da configuração do domínio.
--debug [<port>]	Autônomo	Ative o modo de depuração com um argumento opcional para especificar a porta. Isto funcionará apenas se o script de inicialização suportá-lo.
-D<name>[=<value>]	Autônomo, Domínio	Define uma propriedade do sistema.
--domain-config=<config>	Domínio	Nome do arquivo de configuração do servidor para uso. O padrão é domain.xml .
-h, --help	Autônomo, Domínio	Exibe a mensagem de ajuda.
--host-config=<config>	Domínio	Nome do arquivo de configuração do host. O padrão é host.xml .
--interprocess-hc-address=<address>	Domínio	Endereço através do qual o controlador do host aguarda pela comunicação do controlador de processos.
--interprocess-hc-port=<port>	Domínio	Porta através da qual o controlador do host aguarda pela comunicação do controlador de processos.
--master-address=<address>	Domínio	Define a propriedade do sistema jboss.domain.master.address com o valor especificado. Em uma configuração padrão do controlador de host subordinado, isso é usado para configurar o endereço do controlador de host mestre.

Argumentos ou Opções	Modo de Operação	Descrição
--master-port=<port>	Domínio	Define a propriedade do sistema jboss.domain.master.port com o valor especificado. Em uma configuração padrão do controlador de host subordinado, isso é usado para configurar a porta usada para a comunicação de gerenciamento nativo pelo controlador de host mestre.
--read-only-server-config=<config>	Autônomo	Nome do arquivo de configuração do servidor para uso. Difere de --server-config e -c no sentido de que o arquivo original nunca é sobrescrito.
--read-only-domain-config=<config>	Domínio	Nome do arquivo de configuração do domínio para uso. Difere de --domain-config e -c no sentido de que o arquivo inicial nunca é sobrescrito.
--read-only-host-config=<config>	Domínio	Nome do arquivo de configuração do host para uso. Difere de --host-config no sentido de que o arquivo inicial nunca é sobrescrito.
-P=<url>, -P <url>, --properties=<url>	Autônomo, Domínio	Carrega propriedades do sistema a partir do URL fornecido.
--pc-address=<address>	Domínio	Endereço através do qual o controlador de processos aguarda pela comunicação dos processos que ele controla.
--pc-port=<port>	Domínio	Porta através da qual o controlador de processos aguarda pela comunicação dos processos que ele controla.
-S<name>[=<value>]	Autônomo	Define uma propriedade de segurança.
-secmgr	Autônomo, Domínio	Executa o servidor com um gerenciador de segurança instalado.
--server-config=<config>	Autônomo	Nome do arquivo de configuração do servidor para uso. O padrão é standalone.xml .

Argumentos ou Opções	Modo de Operação	Descrição
--start-mode=<mode>	Autônomo	<p>Define o modo de inicialização do servidor. Esta opção não pode ser usada em conjunto com --admin-only. Os valores válidos são:</p> <ul style="list-style-type: none"> normal: o servidor será inicializado normalmente. admin-only: o servidor somente abrirá interfaces administrativas e aceitará solicitações de gerenciamento, mas não inicializará outros serviços de tempo de execução nem aceitará solicitações do usuário final. suspend: o servidor será inicializado em modo suspenso e não responderá a solicitações até que seja retomado.
-u=<value>, -u <value>	Autônomo, Domínio	<p>Define a propriedade do sistema jboss.default.multicast.address, que é usada na configuração do endereço multicast dos elementos de associação de soquete nos arquivos de configuração. Isso torna 230.0.0.4 o padrão se nenhum valor for especificado.</p>
-v, -V, --version	Autônomo, Domínio	Exibe a versão do servidor do aplicativo.



ATENÇÃO

Os arquivos de configuração que são fornecidos com o JBoss EAP são configurados para manipular o comportamento das opções (por exemplo, **-be -u**). Se você alterar seus arquivos de configuração para não usar mais a propriedade do sistema controlada pela opção, sua inserção no comando de inicialização não terá efeito algum.

A.2. ARGUMENTOS PARA O UTILITÁRIO ADD-USER

A tabela a seguir descreve os argumentos disponíveis para a opção **add-user.sh** ou **add-user.bat**, que é um utilitário para a adição de novos usuários ao arquivo de propriedades para a autenticação inicial.

Tabela A.2. Argumentos do Comando Add-user

Argumento da Linha de Comando	Descrição
-a	Cria um usuário no realm do aplicativo. Se omitido, o padrão é criar um usuário no realm de gerenciamento.
-dc <value>	O diretório de configuração do domínio que conterá os arquivos de propriedades. Se omitido, o diretório padrão será EAP_HOME/domain/configuration/ .
-sc <value>	Um diretório alternativo de configuração do servidor autônomo que conterá os arquivos de propriedades. Se omitido, o diretório padrão será EAP_HOME/standalone/configuration/ .
-up, --user-properties <value>	O nome do arquivo alternativo de propriedades do usuário. Ele pode ser um caminho absoluto ou um nome de arquivo usado em conjunto com o argumento -sc ou -dc que especifica o diretório de configuração alternativo.
-g, --group <value>	Lista separada por vírgulas dos grupos a serem atribuídos a este usuário.
-gp, --group-properties <value>	O nome do arquivo alternativo de propriedades do grupo. Ele pode ser um caminho absoluto ou um nome de arquivo usado em conjunto com o argumento -sc ou -dc que especifica o diretório de configuração alternativo.
-p, --password <value>	Senha do usuário.
-u, --user <value>	O nome do usuário. Nomes de usuário podem conter somente os seguintes caracteres, em qualquer quantidade e em qualquer ordem: <ul style="list-style-type: none"> • Caracteres alfanuméricos (a-z, A-Z, 0-9) • Hífen (-), ponto (.), vírgula (,), arroba (@) • Barra invertida (\) • Sinal de igual (=)
-r, --realm <value>	Nome do realm usado para proteger as interfaces de gerenciamento. Se omitido, o padrão será ManagementRealm .
-s, --silent	Executa o script add-user sem saída para o console.
-e, --enable	Habilita o usuário.
-d, --disable	Desabilita o usuário.
-cw, --confirm-warning	Confirma avisos automaticamente no modo interativo.

Argumento da Linha de Comando	Descrição
-h, --help	Exibe informações de uso para o script add-user .
-ds, --display-secret	Imprime o valor secreto em modo não interativo.

A.3. ATRIBUTOS DE INTERFACE



NOTA

Os nomes de atributos nesta tabela são listados conforme aparecem no modelo de gerenciamento, por exemplo, ao usar a CLI de gerenciamento. Consulte o arquivo de definição de esquema localizado em **EAP_HOME/docs/schema/wildfly-config_5_0.xsd** para ver os elementos conforme aparecem no XML, pois podem existir diferenças com relação ao modelo de gerenciamento.

Tabela A.3. Valores e Atributos da Interface

Elementos da Interface	Descrição
qualquer	Elemento indicando que parte dos critérios de seleção para uma interface deve ser tal que atenda pelo menos um critério, mas não necessariamente todos, do conjunto aninhado.
any-address	Elemento vazio indicando que os soquetes usando esta interface devem ser associados a um endereço curinga. O endereço curinga IPv6 (::) será usado a não ser que a propriedade do sistema java.net.preferIPv4Stack seja definida como verdadeira e, neste caso, o endereço curinga IPv4 (0.0.0.0) é usado. Se um soquete estiver associado a um endereço anylocal IPv6 em uma máquina em pilha dupla, ele pode aceitar ambos os tráfegos, IPv6 e IPv4; mas, se estiver associado a um endereço anylocal IPv4 (IPv4-mapped), ele pode aceitar somente o tráfego IPv4.
inet-address	Seja um endereço IP em IPv6 ou uma anotação decimal com ponto IPv4, ou um nome de host que possa ser resolvido como um endereço IP.
link-local-address	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se um endereço associado a ele é link-local ou não.

Elementos da Interface	Descrição
loopback	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se trata-se de uma interface loopback ou não.
loopback-address	Endereço de loopback que pode não ser configurado na interface de loopback do computador. Difere-se do tipo de endereço inet em que o valor fornecido será usado mesmo que nenhum NIC que tenha o endereço IP associado a ele possa ser encontrado.
multicast	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se oferece suporte para multicast ou não.
name	O nome da interface.
nic	Nome de uma interface de rede (ex. eth0, eth1, lo).
nic-match	Expressão regular através da qual os nomes das interfaces de rede disponíveis na máquina podem ser correspondidos para encontrar uma interface aceitável.
not	Elemento indicando que parte dos critérios de seleção para uma interface deve ser tal que não atenda a critério algum do conjunto aninhado.
point-to-point	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se trata-se de uma interface ponto a ponto ou não.
public-address	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se possui um endereço publicamente roteável ou não.
site-local-address	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se um endereço associado a ele é site-local ou não.
subnet-match	Um endereço IP de rede e o número de bits no prefixo de rede do endereço, escrito em <i>notação slash</i> (por exemplo, 192.168.0.0/16).
up	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se está ativo no momento.
virtual	Elemento vazio indicando que parte dos critérios de seleção para uma interface deve ser se trata-se de uma interface virtual ou não.

A.4. ATRIBUTOS DA ASSOCIAÇÃO DE SOQUETE



NOTA

Os nomes de atributos nesta tabela são listados conforme aparecem no modelo de gerenciamento, por exemplo, ao usar a CLI de gerenciamento. Consulte o arquivo de definição de esquema localizado em **EAP_HOME/docs/schema/wildfly-config_5_0.xsd** para ver os elementos conforme aparecem no XML, pois podem existir diferenças com relação ao modelo de gerenciamento.

Tabela A.4. Atributos da Associação de Soquete

Atributos	Descrição
client-mappings	Especifica os mapeamentos de clientes para esta associação de soquete. Um cliente conectando a este soquete deve usar o endereço de destino especificado no mapeamento que corresponde a sua interface de saída desejada. Isto permite que as topologias de rede avançadas que usam conversão de endereço de rede ou possuem associações em múltiplas interfaces de rede funcionem. Cada mapeamento deve ser avaliado em ordem declarada, com a primeira correspondência bem-sucedida usada para determinar o destino.
fixed-port	Determina se o valor de porta deve manter-se fixo mesmo se os deslocamentos numéricos forem aplicados a outros soquetes no grupo de soquetes.
interface	Nome da interface com a qual o soquete deve ser associado ou, para soquetes multicast, a interface na qual ele deve aguardar por comunicação. Esta deve ser uma das interfaces declaradas. Se não estiver definido, o valor do atributo default-interface do grupo de associação de soquetes será usado.
multicast-address	Endereço multicast através do qual o soquete deve receber tráfego muticast. Se não especificado, o soquete não será configurado para receber multicast.
multicast-port	A porta pela qual o soquete deve receber o tráfego multicast. Ela deve ser configurada se o multicast-address for configurado.
name	Nome do soquete. Serviços com necessidade de acessar as informações de configuração do soquete o encontrarão usando este nome. Este atributo é necessário.
port	Número de porta com o qual o soquete deve ser associado. Observe que este valor pode ser substituído se os servidores aplicarem um deslocamento de porta para incrementar ou diminuir todos os valores de porta.

A.5. ASSOCIAÇÕES DE SOQUETE PADRÃO

As tabelas a seguir mostram as associações de soquete padrão para cada grupo de associação de soquetes.

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

Tabela A.5. standard-sockets

Associação de soquetes	Porta	Descrição
ajp	8009	Protocolo Apache JServ. Usado para clusterização HTTP e balanceamento de carga.
http	8080	Porta padrão para aplicativos web implantados.
https	8443	Conexão SSL criptografada entre aplicativos web implantados e clientes.
management-http	9990	Usado para comunicação HTTP com a camada de gerenciamento.
management-https	9993	Usado para comunicação HTTPS com a camada de gerenciamento.
txn-recovery-environment	4712	Gerenciador de recuperação de transação JTA.
txn-status-manager	4713	Gerenciador de transação JTA / JTS.

Tabela A.6. ha-sockets

Associação de soquetes	Porta	Porta Multicast	Descrição
ajp	8009		Protocolo Apache JServ. Usado para clusterização HTTP e balanceamento de carga.
http	8080		Porta padrão para aplicativos web implantados.
https	8443		Conexão SSL criptografada entre aplicativos web implantados e clientes.

Associação de soquetes	Porta	Porta Multicast	Descrição
jgroups-mping		45700	Multicast. Usado para descobrir a associação inicial em um cluster HA.
jgroups-tcp	7600		Descoberta unicast de máquinas em clusters HA usando TCP.
jgroups-udp	55200	45688	Descoberta multicast de máquinas em clusters HA usando UDP.
management-http	9990		Usado para comunicação HTTP com a camada de gerenciamento.
management-https	9993		Usado para comunicação HTTPS com a camada de gerenciamento.
modcluster		23364	Porta multicast para a comunicação entre o JBoss EAP e o balanceador de carga HTTP.
txn-recovery-environment	4712		Gerenciador de recuperação de transação JTA.
txn-status-manager	4713		Gerenciador de transação JTA / JTS.

Tabela A.7. full-sockets

Associação de soquetes	Porta	Descrição
ajp	8009	Protocolo Apache JServ. Usado para clusterização HTTP e balanceamento de carga.
http	8080	Porta padrão para aplicativos web implantados.
https	8443	Conexão SSL criptografada entre aplicativos web implantados e clientes.
iiop	3528	Serviços CORBA para transações JTS e outros serviços dependentes de ORB.
iiop-ssl	3529	Serviços CORBA criptografados por SSL.
management-http	9990	Usado para comunicação HTTP com a camada de gerenciamento.

Associação de soquetes	Porta	Descrição
management-https	9993	Usado para comunicação HTTPS com a camada de gerenciamento.
txn-recovery-environment	4712	Gerenciador de recuperação de transação JTA.
txn-status-manager	4713	Gerenciador de transação JTA / JTS.

Tabela A.8. full-ha-sockets

Nome	Porta	Porta Multicast	Descrição
ajp	8009		Protocolo Apache JServ. Usado para clusterização HTTP e balanceamento de carga.
http	8080		Porta padrão para aplicativos web implantados.
https	8443		Conexão SSL criptografada entre aplicativos web implantados e clientes.
iiop	3528		Serviços CORBA para transações JTS e outros serviços dependentes de ORB.
iiop-ssl	3529		Serviços CORBA criptografados por SSL.
jgroups-mping		45700	Multicast. Usado para descobrir a associação inicial em um cluster HA.
jgroups-tcp	7600		Descoberta unicast de máquinas em clusters HA usando TCP.
jgroups-udp	55200	45688	Descoberta multicast de máquinas em clusters HA usando UDP.
management-http	9990		Usado para comunicação HTTP com a camada de gerenciamento.
management-https	9993		Usado para comunicação HTTPS com a camada de gerenciamento.

Nome	Porta	Porta Multicast	Descrição
modcluster		23364	Porta multicast para a comunicação entre o JBoss EAP e o balanceador de carga HTTP.
txn-recovery-environment	4712		Gerenciador de recuperação de transação JTA.
txn-status-manager	4713		Gerenciador de transação JTA / JTS.

Tabela A.9. load-balancer-sockets

Nome	Porta	Porta Multicast	Descrição
http	8080		Porta padrão para aplicativos web implantados.
https	8443		Conexão SSL criptografada entre aplicativos web implantados e clientes.
management-http	9990		Usado para comunicação HTTP com a camada de gerenciamento.
management-https	9993		Usado para comunicação HTTPS com a camada de gerenciamento.
mcmp-management	8090		A porta da conexão do protocolo de gerenciamento de mod-cluster (MCMP) para transmissão de eventos de ciclo de vida.
modcluster		23364	Porta multicast para a comunicação entre o JBoss EAP e o balanceador de carga HTTP.

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

