Red Hat JBoss Enterprise Application Platform 7.1

Guide de démarrage

À utiliser avec Red Hat JBoss Enterprise Application Platform 7.1

Last Updated: 2018-01-11

Red Hat JBoss Enterprise Application Platform 7.1 Guide de démarrage

À utiliser avec Red Hat JBoss Enterprise Application Platform 7.1

Notice légale

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.

Résumé

Ce guide fournit des informations de base pour aider les utilisateurs à prendre en main Red Hat JBoss Enterprise Application Platform 7.1.

Table des matières

CHAPITRE 1. INTRODUCTION

1.1. RED HAT JBOSS ENTERPRISE APPLICATION PLATFORM 7

Red Hat JBoss Enterprise Application Platform 7 (JBoss EAP) est une plateforme middleware construite sur la base de normes ouvertes et conforme à la spécification Java Enterprise Edition 7.

JBoss EAP inclut une structure modulaire qui permet aux services d'être activés s'ils sont requis uniquement, améliorant ainsi la vitesse de démarrage.

La console de gestion et l'interface de ligne de commmande (CLI) rendent la modification des fichiers de configuration XML inutile et ajoutent la possibilité d'encoder et d'automatiser des tâches.

JBoss EAP fournit deux modes d'opération pour les instances JBoss EAP : serveur autonome ou domaine géré. Le mode d'opération serveur autonome correspond à l'exécution de JBoss EAP en tant qu'instance unique du serveur. Le mode d'opération de domaine géré permet la gestion d'instances multiples de JBoss EAP à partir d'un point de contrôle unique.

En outre, JBoss EAP comprend des frameworks de développement et des API pour développer rapidement des applications Java EE sécurisées et évolutives.

1.2. GUIDE DE DÉMARRAGE

Le but de ce guide est de vous expliquer l'utilisation de JBoss EAP de manière simple et rapide. Il aborde les tâches administratives de bases telles que l'installation, la gestion et la configuration de JBoss EAP. Ce guide aide également les développeurs à écrire des applications Java EE 7 à l'aide des guides de quickstart de JBoss EAP.

Pour en savoir plus, veuillez consulter la Suite de documentation JBoss EAP.

CHAPITRE 2. ADMINISTRATION DE JBOSS EAP

2.1. TÉLÉCHARGEMENT ET INSTALLATION DE JBOSS EAP

Ce guide fournit des instructions de base pour le téléchargement et l'installation de JBoss EAP à l'aide de l'installation ZIP, qui est indépendante de la plateforme.

Veuillez consulter le *Guide d'installation* pour plus de détails, dont des instructions sur l'installation de JBoss EAP à l'aide du programme d'installation graphique ou des méthodes d'installation de paquetage RPM.

2.1.1. Conditions préalables pour une installation

Vérifier que les conditions préalables suivantes sont bien remplies avant d'installer JBoss EAP.

Conditions préalables habituelles

- Votre système est pris en charge en accordance avec les configurations JBoss EAP 7 prises en charge.
- Votre système est à jour avec les mises à jour et les errata les plus récentes fournies par Red Hat.

Conditions préalables d'installation de ZIP

- L'utilisateur qui exécutera JBoss EAP possède des droits de lecture et d'écriture pour le répertoire d'installation.
- Le kit de développement Java désiré a été installé.
- Pour Hewlett-Packard HP-UX, un utilitaire **`unzip`** a été installé.
- Pour Windows Server, les variables d'environnement **`JAVA_HOME`** et **`PATH`** ont été installés.

2.1.2. Télécharger JBoss EAP

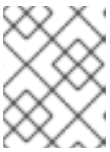
Le fichier ZIP JBoss EAP est disponible sur le Portail Client de Red Hat. Le fichier d'installation ZIP est indépendant de la plateforme.

1. Se connecter au Portail Clients de Red Hat.
2. Cliquer sur **Téléchargements**.
3. Cliquez sur **Red Hat JBoss Enterprise Application Platform** dans la liste des **Téléchargements de produits**.
4. Dans le menu déroulant **Version**, sélectionnez **7.1**.
5. Trouvez **Red Hat JBoss Enterprise Application Platform 7.1.0** dans la liste et cliquez sur le lien **Télécharger**.

2.1.3. Installer JBoss EAP

Une fois le fichier d'installation ZIP JBoss EAP téléchargé, il peut être installé en extrayant le contenu du paquet.

1. Si nécessaire, déplacez le fichier ZIP sur le serveur et l'emplacement sur lequel JBoss EAP doit être installé.

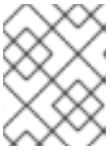


NOTE

L'utilisateur qui exécutera JBoss EAP doit avoir un accès lecture et écriture à ce répertoire.

2. Veuillez extraire l'archive ZIP.

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



NOTE

Pour le serveur Windows, effectuez un clic droit sur le fichier ZIP et sélectionnez **Tout extraire**.

Le répertoire créé en extrayant l'archive ZIP est le répertoire du niveau le plus élevé de l'installation JBoss EAP. Il est nommé ***EAP_HOME***.

2.2. DÉMARRAGE ET ARRÊT DU SERVICE JBOSS EAP 6 À PARTIR D'UN TERMINAL.

2.2.1. Démarrer JBoss EAP

JBoss EAP est pris en charge sur Red Hat Enterprise Linux, Windows Server, Oracle Solaris et Hewlett-Packard HP-UX, et s'exécute en tant que serveur autonome ou en mode d'exploitation de domaine géré. La commande spécifique pour démarrer JBoss EAP dépend de la plateforme sous-jacente et du mode d'exploitation souhaité.

Les serveurs démarrent dans un état suspendu et n'acceptent aucune requête tant que tous les services requis n'ont pas été démarrés. Les serveurs passent alors en mode d'exécution normale et peuvent commencer à accepter des requêtes.

Démarrez JBoss EAP comme un serveur autonome

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

NOTE

Pour Windows Server, utilisez le script ***EAP_HOME*\bin\standalone.bat**.

Ce script de démarrage utilise le fichier ***EAP_HOME*/bin/standalone.conf** (ou **standalone.conf.bat** pour Windows Server) pour définir certaines préférences par défaut, telles que les options JVM. Vous pouvez personnaliser les paramètres dans ce fichier.

JBoss EAP utilise le fichier de configuration **standalone.xml** par défaut, mais peut être démarré à l'aide d'un autre fichier. Pour plus de détails sur les fichiers de configuration autonomes disponibles et sur comment les utiliser, merci de consulter la section Fichiers de

configuration de serveur autonome.

Pour obtenir une liste complète de tous les arguments de script de démarrage disponibles ainsi que leur utilité, utiliser l'argument **--help** ou consulter la section Reference of Startup Arguments.

Démarrez JBoss EAP comme domaine géré
Le contrôleur de domaines doit être démarré avant les serveurs dans des groupes de serveurs du domaine. Utiliser ce script pour démarrer le contrôleur de domaine puis pour chaque contrôleur d'hôte associé.

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

NOTE

Pour Windows Server, utilisez le script ***EAP_HOME*\bin\domain.bat**.

Ce script de démarrage utilise le fichier ***EAP_HOME*/bin/domain.conf** (ou **domain.conf.bat** pour Windows Server) pour définir certaines préférences par défaut, telles que les options JVM. Vous pouvez personnaliser les paramètres dans ce fichier.

JBoss EAP utilise le fichier de configuration **host.xml** par défaut, mais peut être démarré à l'aide d'un autre. Pour plus de détails sur les fichiers de configuration de domaines gérés disponibles et sur la façon de les utiliser, merci de consulter la section Fichiers de configuration du domaine géré .

Lors de la configuration d'un domaine géré, des arguments supplémentaires devront être transmis au script de démarrage. Pour obtenir la liste complète de tous les arguments de script de démarrage disponibles, ainsi que leur utilité, utilisez l'argument **--help** ou consultez la section Server Runtime Arguments (Arguments de temps d'exécution du serveur).

2.2.2. Démarrage de JBoss EAP

La façon dont vous arrêtez JBoss EAP dépend de la façon dont il aura été démarré.

Arrêter une instance interactive de JBoss EAP
Appuyer sur **Ctrl+C** dans le terminal sur lequel JBoss EAP a été démarré.

Arrêter une instance d'arrière plan de JBoss EAP
Utiliser l'interface de ligne de commande de gestion pour se connecter à l'instance en cours et arrêter le serveur.

1. Lancer l'interface de ligne de commande de gestion.

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

2. Exécuter la commande **shutdown**.

   ```
   shutdown
   ```

NOTE

Pour une exécution dans un domaine géré, vous devez spécifier le nom d'hôte à arrêter à l'aide de l'argument **--host** avec la commande **shutdown**.

2.3. JBOSS EAP MANAGEMENT

JBoss EAP utilise une configuration simplifiée, avec un seul fichier de configuration par serveur autonome ou domaine géré. La configuration par défaut pour un serveur autonome est stockée dans le fichier ***EAP_HOME*/standalone/configuration/standalone.xml** et la configuration par défaut pour un domaine géré est stockée dans le fichier ***EAP_HOME*/domain/configuration/domain.xml**. De plus, la configuration par défaut pour un contrôleur d'hôte est stockée dans le fichier ***EAP_HOME*/domain/configuration/host.xml**.

JBoss EAP peut être configuré à l'aide de l'interface de ligne de commande de gestion, la console de gestion web, l'API JAVA ou l'API HTTP. Les modifications effectuées à l'aide de ces interfaces de gestion persistent automatiquement et les fichiers de configuration XML sont remplacés par l'API de gestion. L'interface de ligne de commande et la console de gestion sont les méthodes privilégiées. Par ailleurs, il n'est pas recommandé de modifier les fichiers de configuration XML manuellement.

2.3.1. Utilisateurs de gestion

La configuration JBoss EAP par défaut fournit une authentification locale, de sorte qu'un utilisateur puisse accéder à l'interface de ligne de commande de gestion sur l'hôte local sans nécessiter d'authentification.

Cependant, vous devez ajouter un utilisateur de gestion si vous souhaitez accéder à l'interface de ligne de commande de gestion à distance ou utiliser la console de gestion, slaquelle est considérée comme un accès distant, même si le trafic provient de l'hôte local. Si vous essayez d'accéder à la console de gestion avant d'ajouter un utilisateur de gestion, vous recevrez un message d'erreur.

Si JBoss EAP est installé à l'aide du programme d'installation graphique, un utilisateur de gestion est créé au cours du processus d'installation.

Ce guide traite de la gestion de base des utilisateurs pour JBoss EAP à l'aide du script **add-user**, qui est un utilitaire permettant d'ajouter des utilisateurs aux fichiers des propriétés pour une authentification prête à l'emploi.

Pour des options d'authentification et d'autorisation plus avancées, telles que LDAP ou le contrôle d'accès basé sur les rôles (RBAC), consultez la section Core Management Authentication (Authentification de gestion principale) du guide*Architecture de sécurité* de JBoss EAP.

2.3.1.1. Ajout d'un utilisateur de gestion

1. Exécuter le script utilitaire **add-user** et suivre les instructions.

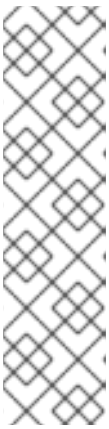
```
$ EAP_HOME/bin/add-user.sh
```

NOTE

Pour Windows Server, utilisez le script ***EAP_HOME*\bin\add-user.bat**.

2. Appuyer sur la touche **ENTRÉE** pour sélectionner l'option par défaut **a** pour ajouter un utilisateur de gestion.
   Cet utilisateur sera ajouté à *ManagementRealm* et sera autorisé à effectuer des opérations de gestion à l'aide de la console ou de l'interface de ligne de commande de gestion. L'autre option (**b**) ajoute un utilisateur à *ApplicationRealm* utilisé pour des applications et ne fournit aucune autorisation particulière.

3. Saisir le nom d'utilisateur et le nom de passe. On vous demandera de saisir le mot de passe une seconde fois pour confirmer.

   NOTE

   Les noms d'utilisateur ne peuvent contenir que les caractères suivants (quel que soit leur nombre et dans n'importe quel ordre) :

   - Caractères alphanumériques (a-z, A-Z, 0-9)
   - Tirets (-), points (.), virgules (,), arobase (@)
   - Barre oblique inverse (\)
   - Égal (=)

   Par défaut, JBoss EAP autorise les mots de passe faibles, mais émet un avertissement.

   Pour savoir comment modifier ce comportement par défaut, reportez-vous à la section Setting Add-User Utility Password Restrictions(Définition des restrictions de mot de passe de l'utilitaire Add-USer) du *Guide de configuration* de JBoss EAP.

4. Saisir une liste de groupes séparés par des virgules auxquels l'utilisateur appartient. Si vous souhaitez que l'utilisateur n'appartienne à aucun groupe, appuyez sur **ENTRÉE** pour laisser ce champ vide.

5. Vérifier les informations et saisir **yes** pour confirmer.

6. Déterminer si cet utilisateur représente une instance de serveur EAP à distance. Pour un utilisateur de gestion de base, saisir **no**.
   Un type d'utilisateur pouvant être ajouté au *ManagementRealm* est un utilisateur représentant une autre instance de JBoss EAP pouvant s'authentifier pour joindre en tant que membre d'un cluster. Si tel est le cas, répondre **yes** pour recevoir une valeur secrète hachée représentant le mot de passe de l'utilisateur, qui devra être ajouté à un fichier de configuration différent.

Les utilisateurs peuvent également être créés de manière non interactive en passant des paramètres au script **add-user**. Cette approche n'est pas recommandée sur des systèmes partagés, car les mots de passe seront visibles dans les journaux et fichiers d'historiques. Pour plus d'information, consulter Exécuter l'utilitaire Ajout d'utilisateur de manière non interactive.

2.3.1.2. Exécuter l'utilitaire Ajout d'utilisateur de manière non interactive

Vous pouvez exécuter le script **add-user** de manière non interactive en passant des arguments en ligne de commande. Le nom d'utilisateur et mot de passe doivent être fournis.

AVERTISSEMENT

Cette approche n'est pas recommandée sur des systèmes partagés, car les mots de passe seront visibles dans les fichiers de journalisation et d'historique.

Créer un utilisateur appartenant à plusieurs groupes
La commande suivante ajoute un utilisateur de gestion (**mgmtuser1**), avec les groupes **guest** et **mgmtgroup**.

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

Spécifier un fichier de propriétés alternatif
Par défaut, les informations de groupe et d'utilisateur créés à l'aide du script **add-user** sont stockés dans des fichiers de propriété situés dans le répertoire de configuration de serveur.

Les informations utilisateur sont stockées dans les fichiers de propriété suivants :

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

Les informations de groupe sont stockées dans les fichiers de propriété suivants :

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

Les noms de fichiers de propriété et de répertoires par défaut peuvent être remplacés. La commande suivante ajoute un nouvel utilisateur, indiquant un nom et emplacement différents pour les fichiers de propriété utilisateur.

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

Le nouvel utilisateur a été ajouté aux fichiers de propriétés utilisateur situés dans ***/path/to*/standaloneconfig/newname.properties** et ***/path/to*/domainconfig/newname.properties**. Veuillez noter que ces fichiers doivent déjà exister afin d'éviter une erreur.

Pour obtenir une liste complète de tous les arguments **add-user** disponibles ainsi que leur utilité, utiliser l'argument **--help** ou consulter la section Add-User Utility Arguments.

2.3.2. Interfaces de gestion

2.3.2.1. Interface CLI

L'interface en ligne de commande (CLI) est un outil d'administration en ligne de commande pour JBoss EAP.

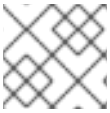
Utiliser l'interface CLI pour démarrer et stopper les serveurs, déployer et retirer les déploiements d'applications, configurer les paramètres du système, ou encore, effectuer d'autre tâches administratives. Les opérations peuvent être effectuées par mode lots, ce qui permet à plusieurs tâches d'ëtre exécutées en groupe.

Plusieurs commandes de terminal courantes sont disponibles, telles que **ls**, **cd** et **pwd**. L'interface CLI prend également en charge la saisie automatique avec la touche «Tab ».

Pour obtenir des informations supplémentaires sur la façon d'utiliser l'interface de ligne de commande de gestion, y compris les commandes et opérations, la syntaxe et l'exécution par lots, consultez le *Guide de l'interface de ligne de commande de gestion* de JBoss EAP.

Lancement de l'interface CLI

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



NOTE

Pour Windows Server, utilisez le script ***EAP_HOME*\bin\jboss-cli.bat**.

Connexion à un serveur en cours d'exécution

```
connecter
```

Ou vous pouvez lancer l'interface de ligne de commande de gestion et vous connecter, en une seule étape, à l'aide de la commande ***EAP_HOME*/bin/jboss-cli.sh --connect**.

Afficher l'aide
Utiliser la commande suivante pour afficher l'aide générale.

```
aide
```

Utilisez l'indicateur **--help** sur une commande pour obtenir des instructions sur l'utilisation de cette commande en particulier. Par exemple, pour obtenir des informations sur l'utilisation de **deploy**, la commande suivante est exécutée.

```
deploy --help
```

Quitter l'interface CLI

```
Quitter
```

Afficher les paramètres de système
La commande suivante utilise l'opération **read-attribute** pour afficher si l'exemple de source de donnée est activé.

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

En cas d'exécution dans un domaine géré, il convient de spécifier le profil à mettre à jour en faisant précéder la commande de **/profile=*PROFILE_NAME***.

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

Mettre à jour les paramètres système
La commande suivante utilise l'opération **write-attribute** pour désactiver l'exemple de source de donnée.

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

Démarrer les serveurs
L'interface CLI peut également être utilisée pour démarrer et arrêter les serveurs lors d'une exécution dans un domaine géré.

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

2.3.2.2. Console de gestion

La console de gestion est un outil administratif basé web pour la plateforme JBoss EAP.

Utilisez la console de gestion pour démarrer et arrêter des serveurs, déployer et annuler le déploiement des applications, régler les paramètres du système et apporter des modifications persistantes à la configuration du serveur. La console de gestion a également la capacité d'effectuer des tâches administratives, avec des notifications directes lorsque les modifications effectuées par l'utilisateur actuel exigent que l'instance du serveur soit redémarrée ou rechargée.

Dans un domaine géré, les instances de serveur et les groupes de serveurs d'un même domaine peuvent être gérés de façon centralisée à partir de la Console de gestion du contrôleur de domaine.

Pour une instance JBoss EAP exécutée sur l'hôte local à l'aide du port de gestion par défaut, la console de gestion peut être accédée par un navigateur web sur http://localhost:9990/console/App.html. Vous devrez vous authentifier avec un utilisateur possédant des permissions pour accéder à la console de gestion.

La console de gestion fournit les onglets de navigation et de gestion de votre serveur autonome JBoss EAP ou de votre domaine géré suivants :

Accueil

Apprenez à effectuer quelques tâches de gestion et quelques configurations de base. Prenez le temps de visiter le console de gestion JBoss EAP.

Déploiements

Ajouter, supprimer, et activer des déploiements. Dans un domaine géré, assigner des déploiements à des groupes de serveurs.

Configuration

Configurez les sous-systèmes disponibles, qui vous donnent des capacités de services web, messagerie, ou HA. Dans un domaine géré, gérez les profils qui contiennent plusieurs configurations de sous-systèmes.

Exécution

Voir les informations de runtime (exécution), comme le statut du serveur, l'utilisation de la JVM, et les journaux des serveurs. Dans un domaine géré, gérez vos hôtes, groupes de serveurs, et serveurs.

Contrôle des accès

Assignez des rôles à des groupes et à des utilisateurs lorsque vous êtes en contrôle d'accès basé rôle.

Correctifs

Appliquez des correctifs à vos instances JBoss EAP.

NOTE

Pour effectuer une visite guidée de la console de gestion, cliquez sur le lien **Take a Tour!** (Visite guidée) qui se trouve sur la page d'accueil de la console de gestion.

2.3.3. Fichiers de configuration

2.3.3.1. Fichiers de configuration du serveur autonome

Les fichiers de configuration autonomes se trouvent dans le répertoire ***EAP_HOME*/standalone/configuration/**. Il existe un fichier distinct pour chacun des cinq profils prédéfinis (*default*, *ha*, *full*, *full-ha*, *load-balancer*).

Tableau 2.1. Fichiers de configuration autonomes

Fichier de configuration	Utilité
standalone.xml	Ce fichier de configuration autonome est la configuration par défaut utilisée lorsque vous démarrez votre serveur autonome. Il contient toutes les informations sur le serveur, dont les sous-systèmes, les réseaux, les déploiements, les liaisons de socket et autres détails configurables. Il ne fournit pas les sous-systèmes nécessaires à la messagerie ou la haute disponibilité.
standalone-ha.xml	Ce fichier de configuration autonome inclut tous les sous-systèmes par défaut et ajoute les sous-systèmes **modcluster** et **jgroups** pour la haute disponibilité. Il ne fournit pas les sous-systèmes nécessaires pour la messagerie.
standalone-full.xml	Ce fichier de configuration autonome inclut tous les sous-systèmes par défaut et ajoute les sous-systèmes **messaging-activemq** et **iiop-openjdk**. Il ne fournit pas les sous-systèmes nécessaires à la haute disponibilité.
standalone-full-ha.xml	Ce fichier de configuration autonome inclut la prise en charge de chaque sous-système, dont les sous-systèmes de messagerie et la haute disponibilité.

Fichier de configuration	Utilité
standalone-load-balancer.xml	Ce fichier de configuration autonome inclut les sous-systèmes minimums nécessaires pour utiliser l'équilibreur de charge frontal mod_cluster intégré afin d'équilibrer la charge d'autres instances JBoss EAP.

Par défaut, le démarrage de JBoss EAP en tant que serveur autonome utilise le fichier **standalone.xml**. Pour démarrer JBoss EAP avec une configuration différente, utiliser l'argument **--server-config**. Par exemple,

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

2.3.3.2. Fichiers de configuration de domaines gérés

Les fichiers de configuration du domaine géré sont situés dans le répertoire ***EAP_HOME*/domain/configuration/**.

Tableau 2.2. Fichiers de configuration de domaines gérés

Fichier de configuration	Utilité
domain.xml	Il s'agit du fichier de configuration principal pour un domaine géré. Seul le maître du domaine lit ce fichier. Ce fichier contient les configurations pour tous les profils (*default*, *ha*, *full*, *full-ha*, *load-balancer*).
host.xml	Ce fichier contient les détails de configuration spécifiques à un hôte physique dans un domaine géré, tels que les interfaces réseau, les liaisons de sockets, le nom de l'hôte et d'autres détails spécifiques à l'hôte. Le fichier **host.xml** contient toutes les fonctionnalités de **hôte-master.xml** et **hôte-slave.xml** décrites ci-dessous.
host-master.xml	Ce fichier inclut uniquement les détails de configuration nécessaires pour exécuter un serveur en tant que contrôleur de domaine principal.
host-slave.xml	Ce fichier inclut uniquement les détails de configuration nécessaires pour exécuter un serveur en tant que contrôleur hôte de domaine géré.

Par défaut, le démarrage de JBoss EAP en tant que domaine géré utilise le fichier **host.xml**. Pour démarrer JBoss EAP avec une configuration différente, utiliser l'argument **--host-config**. Par exemple,

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

2.3.3.3. Sauvegarder les données de configuration

Afin de restaurer la configuration du serveur JBoss EAP, les éléments situés aux emplacements suivants doivet être sauvegardés :

- ***EAP_HOME*/standalone/configuration/**
  - Sauvegarder le répertoire complet pour enregistrer les données d'utilisateur, la configuration de serveur et les paramètres de journalisation pour les serveurs autonomes.
- ***EAP_HOME*/domain/configuration/**
  - Sauvegarder le répertoire complet pour enregistrer les données de profil et d'utilisateur, la configuration d'hôte et de domaine et les paramètres de journalisation pour des domaines gérés.
- ***EAP_HOME*/modules/**
  - Sauvegarder des modules personnalisés
- ***EAP_HOME*/welcome-content/**
  - Sauvegarder tout contenu de bienvenue personnalisé.
- ***EAP_HOME*/bin/**
  - Sauvegarder tout script personnalisé ou fichier de configuration de démarrage

2.3.3.4. Instantané du fichier de configuration

Pour aider à la maintenance et à la gestion du serveur, JBoss EAP crée une version horodatée du fichier de configuration original au moment du démarrage. Toutes les modifications de configuration supplémentaires suite aux opérations de gestion entraîneront la sauvegarde automatique du fichier d'origine, et une copie de travail de l'instance sera alors conservée en tant que référence et pour la restauration. De plus, les instantanés de configuration peuvent être effectués, représentant des copies à un moment précis de la configuration du serveur actuel. Ces instantanés peuvent être enregistrés et chargés par un administrateur.

Les exemples suivants utilisent le fichier **standalone.xml**, mais le même processus s'applique aux fichiers **domain.xml** et **host.xml**.

Effectuer un instantané
Utiliser l'interface CLI pour effectuer un instantané des configurations actuelles.

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

Lister les instantanés
Utiliser l'interface CLI pour énumérer tous les instantanés effectués.

```
:list-snapshots
{
```

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

Supprimer un instantané
Utiliser l'interface CLI pour supprimer un instantané.

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

Démarrer le serveur avec un instantané
Le serveur peut être démarré à l'aide d'un instantané ou d'une version de configuration automatiquement enregistrée.

1. Accédez au répertoire ***EAP_HOME*/standalone/configuration/standalone_xml_history** et identifiez l'instantané ou le fichier de configuration enregistré à charger.

2. Démarrez le serveur et pointez vers le fichier de configuration sélectionné. Transmettez le chemin d'accès du fichier relatif au répertoire de configuration (***EAP_HOME*/standalone/configuration/**).

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

NOTE

Pour une exécution dans un domaine géré, utiliser plutôt l'argument **--host-config** pour spécifier le fichier de configuration.

2.3.3.5. Remplacement de propriété

JBoss EAP vous autorise à utiliser des expressions pour définir des propriétés remplaçables à la place de valeurs littérales dans la configuration. Les expressions utilisent le format **${*PARAMETER*:*DEFAULT_VALUE*}**. Si le paramètre spécifié est défini, la valeur du paramètre sera utilisée. Sinon, la valeur par défaut fournie sera utilisée.

Les sources prises en charge pour résoudre les expressions sont les propriétés système, les variables d'environnement et le coffre-fort. Pour les déploiements uniquement, la source peut correspondre à des propriétés répertoriées dans un fichier **META-INF/jboss.properties** qui se trouve dans l'archive de déploiement. Pour les types de déploiement qui prennent en charge les sous-déploiements, la résolution est étendue à tous les sous-déploiements si le fichier de propriétés se trouve dans le déploiement externe (EAR, par exemple). Si le fichier de propriétés se trouve dans le sous-déploiement, la résolution est alors étendue uniquement à ce sous-déploiement.

L'exemple ci-dessous à partir du fichier de configuration **standalone.xml** définit le **inet-address** pour l'interface **public** à **127.0.0.1** à moins que le paramètre **jboss.bind.address** soit défini.

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

Le paramètre **jboss.bind.address** peut être défini au démarrage de EAP en tant que serveur autonome avec la commande suivante :

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

Expressions imbriquées
Les expressions peuvent être imbriquées, ce qui permet une utilisation plus avancée des expressions au lieu de valeurs fixes. Le format d'une expression imbriquée ressemble à celui d'une expression normale, mais une expression est imbriquée dans l'autre, comme par exemple :

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

Les expressions imbriquées sont évaluées de façon récursive, donc l'expression *inner* est d'abord évaluée, puis *outer*. Les expressions peuvent également être récursives, lorsque l'évaluation d'une expression entraîne une autre expression, qui est à son tour évaluée. Les expressions imbriquées sont autorisées partout où les expressions sont autorisées, à l'exception des commandes CLI de gestion.

Une expression imbriquée peut être utilisée si le mot de passe utilisé dans une définition de source de donnée est masqué. La configuration pour la source de donnée peut contenir la ligne suivante :

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

La valeur de **ds_ExampleDS** peut être remplacée avec une propriété de système (**datasource_name**) à l'aide d'une expression imbriquée. La configuration de la source de données peut contenir à la place la ligne suivante :

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

JBoss EAP doit tout d'abord évaluer l'expression **${datasource_name}**, puis l'ajouter à une plus grande expression et évaluer l'expression qui en résulte. L'avantage de cette configuration est que le nom de la source de données se soustrait de la configuration fixe

Remplacement de propriété basée descripteur
La configuration d'application (comme les paramètres de connexion de la source de données) varie généralement entre les environnements de développement, de test et de production. Cette variation est parfois adaptée par les scripts de création système, car la spécification Java EE ne contient pas de méthode pour externaliser ces configurations. Dans JBoss EAP, vous pouvez utiliser le remplacement de propriété basée sur le descripteur pour gérer la configuration en externe.

Le remplacement de propriété basée descripteur remplace les propriétés basées sur des descripteurs, ce qui vous permet de supprimer les hypothèses concernant l'environnement de l'application et la chaîne de construction. Les configurations spécifiques à

l'environnement peuvent être spécifiées dans les descripteurs de déploiement à la place des annotations ou des build system scripts. Vous pouvez fournir la configuration dans les fichiers ou en tant que paramètres en ligne de commande.

Il existe, dans le sous-système **ee**, plusieurs indicateurs qui contrôlent si le remplacement de propriété est appliqué.

Le remplacement de descripteur spécifique à JBoss est contrôlé par l'indicateur **jboss-descriptor-property-replacement** et *activé* par défaut. Une fois activées, les propriétés peuvent être remplacées dans les descripteurs de déploiement suivants :

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

La commande d'interface CLI suivante peut être utilisée pour activer ou désactiver le remplacement de propriété dans les descripteurs spécifiques à JBoss :

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

Le remplacement de descripteur Java EE est contrôlé par l'indicateur **spec-descriptor-property-replacement** et *désactivé* par défaut. Une fois activées, les propriétés peuvent être remplacées dans les descripteurs de déploiement suivants :

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

La commande d'interface CLI suivante peut être utilisée pour activer ou désactiver le remplacement de propriétés dans les descripteurs Java EE :

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

2.4. RÉSEAU ET CONFIGURATION DE PORT

2.4.1. Interfaces

JBoss EAP fait référence aux interfaces nommées dans la configuration. Cela permet à la configuration de faire référence aux déclarations d'interfaces individuelles par des noms logiques plutôt que de demander les informations complètes de l'interface à chaque utilisation.

Cela permet également une configuration simplifiée dans un domaine géré, où les informations d'interface de réseau peuvent varier selon les machines. Chaque instance de serveur peut correspondre à un groupe de noms logiques.

Les fichiers **standalone.xml**, **domain.xml** et **host.xml** incluent tous des déclarations d'interface. Il y a plusieurs noms d'interfaces préconfigurés, suivant la configuration utilisée par défaut. L'interface **management** peut être utilisée pour tous les composants et services nécessitant la couche de gestion, dont le point de terminaison de gestion HTTP. L'interface **public** peut être utilisée pour toutes les communications de réseau en lien avec les applications. L'interface **unsecure** est utilisée pour les sockets IIOP dans la configuration standard. L'interface **private** est utilisée pour les sockets de JGroups dans la configuration standard.

2.4.1.1. Configuration d'interface par défaut

```
<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>
```

Par défaut, JBoss EAP lie ces interfaces à **127.0.0.1**, mais ces valeurs peuvent être remplacées lors de l'exécution en définissant la propriété appropriée. Par exemple, l'**inet-address** de l'interface **public** peut être définie au démarrage de JBoss EAP en tant que serveur autonome par la commande suivante.

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

Sinon, vous pouvez utiliser l'argument **-b** dans la ligne de commande de démarrage du serveur. Pour plus d'informations sur les options de démarrage d'un serveur, voir Server Runtime Arguments.

IMPORTANT

Si vous modifiez les ports ou interfaces réseau par défaut utilisés par JBoss EAP, vous devez également vous rappeler de modifier les scripts qui utilisent ces interfaces ou ports modifiés, tels que les scripts de service JBoss EAP. Rappelez-vous également de spécifier le port et l'interface appropriés lorsque vous accédez à l'interface de ligne de commande ou à la console de gestion.

2.4.1.2. Configuration d'interfaces

Les interfaces de réseau sont déclarées en indiquant un nom logique et des critères de sélection pour une interface physique. Les critères de sélection peuvent faire référence à une adresse wildcard ou indiquer un ensemble d'une ou plusieurs caractéristiques qu'une interface ou adresse doit avoir afin d'obtenir une correspondance valide. Pour obtenir une

liste de tous les critères de sélection d'interface disponibles, consulter la section Interface Attributes.

Les interfaces peuvent être configurées à l'aide de la console de gestion ou l'interface CLI. Voici plusieurs exemples d'ajout et de mise à jour d'interfaces. La commande d'interface CLI est affichée en premier, suivie par le fichier XML de configuration correspondant.

Ajouter une interface avec une valeur NIC
Ajouter une nouvelle interface avec une valeur NIC de **eth0**.

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

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

Ajouter une interface avec plusieurs valeurs conditionnelles
Ajouter une nouvelle interface correspondant à une interface/adresse sur le sous-réseau approprié si l'interface/adresse est active, prend en charge les multi-diffusion et n'est pas une liaison point à point.

```
/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>
```

Mettre à jour un attribut d'interface
Mettre à jour la valeur **inet-address** par défaut de l'interface **publique**, en maintenant la propriété **jboss.bind.address** pour permettre à cette valeur d'être définie à l'exécution.

```
/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>
```

Ajouter une interface à un serveur dans un domaine géré

```
/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. Liaisons de sockets

Les liaisons de sockets et les groupes de liaisons de sockets vous permettent de définir les ports de réseau et leur relation aux interfaces de réseau requises pour votre configuration JBoss EAP. Une liaison de socket est une configuration nommée pour une socket. Un groupe de liaison de sockets est une collection de déclarations de liaisons de sockets regroupées sous un nom logique.

Cela permet à d'autres sections de la configuration de faire référence à des liaisons de sockets par leur nom logique plutôt que de demander des informations complètes sur la configuration des sockets à chaque utilisation.

Les déclarations de ces configurations nommées se trouvent dans les fichiers de configuration **`standalone.xml`** et **`domain.xml`**. Un serveur autonome contient uniquement un groupe de liaisons de sockets, alors qu'un domaine géré peut contenir plusieurs groupes. Vous pouvez créer un groupe de liaisons de sockets pour chaque groupe de serveurs dans le domaine géré, ou partager un groupe de liaisons de sockets entre plusieurs groupes de serveurs.

Les ports utilisés par défaut par JBoss EAP dépendent des groupes de liaisons de sockets utilisés, ainsi que des exigences de vos déploiements individuels.

2.4.2.1. Ports de gestion

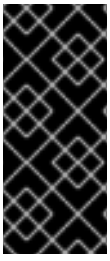
Les ports de gestion ont été consolidés dans JBoss EAP 7. Par défaut, JBoss EAP 7 utilise le port **9990** pour la gestion native, utilisée par l'interface de ligne de commande de gestion, et la gestion HTTP, utilisée par la console de gestion web. Le port **9999**, utilisé en tant que port de gestion natif dans JBoss EAP 6, n'est plus utilisé, mais peut être activé si nécessaire.

Si HTTPS est activé pour la console de gestion, alors le port **9993** est utilisé par défaut.

2.4.2.2. Liaisons de socket par défaut

JBoss EAP comprend un groupe de liaison de sockets pour chacun des cinq profils prédéfinis (*default*, *ha*, *full*, *full-ha*, *load-balancer*).

Pour des informations détaillées sur les liaisons de sockets par défaut, telles que les descriptions et ports par défaut, merci de consulter la section Default Socket Bindings.



IMPORTANT

Si vous modifiez les ports ou interfaces réseau par défaut utilisés par JBoss EAP, vous devez également vous rappeler de modifier les scripts qui utilisent ces interfaces ou ports modifiés, tels que les scripts de service JBoss EAP. Rappelez-vous également de spécifier le port et l'interface appropriés lorsque vous accédez à l'interface de ligne de commande ou à la console de gestion.

Serveur autonome
En cas d'exécution comme serveur autonome, un seul groupe de liaisons de sockets est

défini par fichier de configuration. Chaque fichier de configuration autonome (**standalone.xml**, **standalone-ha.xml**, **standalone-full.xml**, **standalone-full-ha.xml**, **standalone-load-balancer.xml**) définit les liaisons de sockets pour les technologies utilisées par son profil correspondant.

Par exemple, le fichier de configuration autonome (**standalone.xml**) spécifie les liaisons de sockets ci-dessous.

```
<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>
```

Domaine géré
En cas d'exécution dans un domaine géré, tous les groupes de liaisons de sockets sont définis dans le fichier **domain.xml**. Il existe cinq groupes de liaisons de sockets prédéfinis :

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

Chaque groupe de liaisons de sockets indique les liaisons de sockets pour les technologies utilisées par le profil correspondant. Par exemple, le groupe de liaisons de sockets **full-ha-sockets** définit plusieurs liaisons de sockets **jgroups**, utilisées par le profil*full-ha* pour la haute disponibilité.

```
<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>
```

NOTE

La configuration de socket pour les interfaces de gestion est définie dans le fichier **host.xml** du contrôleur de domaine.

2.4.2.3. Configuration de liaisons de socket

Lors de la définition d'une liaison de socket, vous pouvez configurer les attributs **port** et **interface**, ainsi que les paramètres de multidiffusion tels que**multicast-address** et **multicast-port**. Pour plus de détails sur tous les attributs de liaison de socket, voir la section Socket Binding Attributes section.

Les liaisons de socket peuvent être configurées à l'aide de la console de gestion ou l'interface CLI. Les étapes suivantes indiquent d'ajouter un groupe de liaison de socket et de configurer les paramètres de liaison de socket à l'aide de l'interface CLI.

1. Ajouter un nouveau groupe de liaisons de sockets. Merci de noter que cette étape ne peut pas être effectuée avec un serveur autonome.

   -

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

2. Ajouter une liaison de socket

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

3. Modifier les liaisons de sockets pour utiliser une interface autre que celle par défaut et définie par le groupe de liaisons de sockets.

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

L'exemple suivant montre à quoi la configuration XML peut ressembler après avoir effectué les étapes ci-dessus.

```
<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. Décalages (offset) de port

Un décalage de port (Port offset) est une valeur de décalage chiffré qui vient s'ajouter aux valeurs de port données par le groupe de liaisons de sockets pour ce serveur. Cela permet au serveur d'hériter des valeurs de port définies dans son groupe de liaisons de sockets, avec un décalage pour veiller à ce qu'il n'entre pas en conflit avec les autres serveurs du même hôte. Par exemple, si le port HTTP du groupe de liaisons de sockets est **8080** et si votre serveur utilise un port offset de **100**, son port HTTP sera**8180**.

Voici un exemple de définition de décalage de port de **250** pour un serveur dans un domaine géré à l'aide de l'interface CLI.

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

Les décalages de port peuvent être utilisés pour les serveurs dans un domaine géré et pour exécuter plusieurs serveurs autonomes sur le même hôte.

Vous pouvez indiquer un décallage de port quand vous démarrez un serveur autonome par la propriété **jboss.socket.binding.port-offset**.

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

2.4.3. Adresse IPv6

Par défaut, JBoss EAP est configuré pour exécuter avec des adresses IPv4. Les étapes ci-dessous montrent comment configurer JBoss EAP pour exécuter avec des adresses IPv6.

Configurer JVM Stack pour les adresses IPv6

Mettre à jour la configuration de démarrage pour privilégier les adresses IPv6.

1. Ouvrir le fichier de configuration de démarrage.
   - En cas d'exécution comme serveur autonome, modifiez le fichier ***EAP_HOME*/bin/standalone.conf** (ou **standalone.conf.bat** pour Windows Server).
   - En cas d'exécution dans un domaine géré, modifiez le fichier ***EAP_HOME*/bin/domain.conf** (ou **domain.conf.bat** pour Windows Server).
2. Définir la propriété **java.net.preferIPv4Stack** sur **false**.

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

3. Ajouter la propriété **java.net.preferIPv6Addresses** et la définir sur**true**.

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

L'exemple suivant montre à quoi les options JVM du fichier de configuration peuvent ressembler après avoir effectué les modifications mentionnées ci-dessus.

```
# 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
```

Déclarations d'interface de mise à jour pour les adresses IPv6
Les valeurs d'interface par défaut dans la configuration peuvent être modifiées en adresses IPv6. Par exemple, la commande d'interface CLI ci-dessous définit l'interface **management** à l'adresse de rebouclage IPv6 (**::1**).

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

L'exemple suivant montre à quoi la configuration XML peut ressembler après avoir effectué la commande ci-dessus

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

CHAPITRE 3. DÉVELOPPEMENT D'APPLICATIONS AVEC JBOSS EAP

3.1. VUE D'ENSEMBLE

Ce guide fournit des informations sur le démarrage d'applications de développement à l'aide de Red Hat JBoss Developer Studio et des exemples de quickstart de JBoss EAP 7.

Red Hat JBoss Developer Studio est un environnement de développement intégré (IDE) basé sur Eclipse et intégrant les plug-ins de développement d'application de JBoss. JBoss Developer Studio peut vous assister avec le développement de votre application, avec la disponibilité des assistants de JBoss et avec l'abilité de déployer des applications vers des serveurs JBoss EAP. Plusieurs exemples de codes de quickstart sont fournis avec JBoss EAP 7 pour aider les utilisateurs à écrire des applications avec les différentes technologies Java EE 7.

3.2. CONFIGURER L'ENVIRONNEMENT DE DÉVELOPPEMENT

Il est recommandé d'utiliser JBoss Developer Studio 11.0 ou version ultérieure avec JBoss EAP 7.1.

1. Téléchargez et installez JBoss Developer Studio.
   Pour plus d'informations, consultez la section Installing JBoss Developer Studio Stand-alone Using the Installer (Installation de JBoss Developer Studio Stand-alone à l'aide du programme d'installation) du *Guide d'installation* de JBoss Developer Studio.

2. Configurez le serveur JBoss EAP dans JBoss Developer Studio.
   Pour plus d'informations, consultez la section Using Runtime Detection to Set Up JBoss EAP from within the IDE (Utilisation de la détection d'exécution pour configurer JBoss EAP à partir de l'IDE) du guide *Getting Started with JBoss Developer Studio Tools* (Prise en main de JBoss Developer Studio Tools).

3.3. UTILISATION DES EXEMPLES DE QUICKSTART

Les exemples de Quickstart fournis avec JBoss EAP sont des projets Maven.

3.3.1. Maven

Apache Maven est un outil d'automation de builds distribuées, utilisé en développement Java pour créer, gérer et générer des projets de logiciels. Maven utilise des fichiers de configuration nommés Project Object Model, ou POM, pour définir des projets et gérer le processus de build. Les POM décrivent le module et les dépendance du composant, et les sorties sont sous forme de fichier XML. Cela garantit que le projet soit construit de façon correcte et uniforme.

Maven y parvient en utilisant un référentiel. Un référentiel Maven stocke les bibliothèques, plug-ins et autres artefacts de la build. Le référentiel public par défaut est Maven 2 Central Repository, mais les référentiels peuvent être privés et internes à une entreprise dans le but de partager des artefacts communs au sein d'équipes de développeurs. Les référentiels sont également disponibles auprès de tierces parties. Pour plus d'informations, voir le projet Apache Maven et le guide Introduction aux référentiels.

JBoss EAP inclut un référentiel Maven contenant les prérequis que les développeurs Java EE utilisent généralement pour créer des applications sur JBoss EAP.

Pour obtenir des informations supplémentaires sur la façon d'utiliser Maven avec JBoss EAP, consultez la section Using Maven with JBoss EAP (Utilisation de Maven avec JBoss EAP) du *Guide de développement* de JBoss EAP.

3.3.2. Utilisation de Maven avec des Quickstarts

Les artefacts et dépendances nécessaires pour créer et déployer des applications vers JBoss EAP 7 sont hébergés sur un référentiel public. En commençant par les quickstarts de JBoss EAP 7, il n'est plus nécessaire de configurer votre fichier **`settings.xml`** de Maven pour utiliser ces référentiels lors de la création des quickstarts. Les référentiels Maven sont désormais configurés dans les fichiers POM de projet de quickstart. Cette méthode de configuration est fournie pour faciliter le quickstart mais n'est généralement pas recommandée pour les projets de production car votre build peut être ralenti.

Red Hat JBoss Developer Studio inclut Maven. Il est donc inutile de le télécharger et de l'installer séparément. Il est recommandé d'utiliser JBoss Developer Studio version 11.0 ou ultérieure.

Si vous souhaitez utiliser la ligne de commande Maven pour créer et déployer vos applications, vous devez tout d'abord télécharger Maven à partir du projet Apache Maven et l'installer à l'aide des instructions fournies dans la documentation Maven.

3.3.3. Télécharger et exécuter les quickstarts

3.3.3.1. Télécharger les quickstarts

JBoss EAP contient une bonne série d'exemples de code quickstart (démarrage rapide) conçus pour aider les utilisateurs à commencer à rédiger des applications en utilisant différentes technologies Java EE 7. Les quickstarts peuvent être téléchargés à partir du portail client Red Hat.

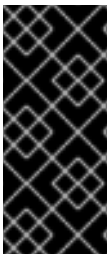
1. Se connecter au Portail Clients de Red Hat.
2. Cliquer sur **Téléchargements**.
3. À partir de la liste **Téléchargements de produits**, cliquer sur **Red Hat JBoss Enterprise Application Platform**.
4. Sélectionnez **7.1** dans le menu déroulant **Version**.
5. Recherchez l'entrée **Red Hat JBoss Enterprise Application Platform 7.1.0 Quickstarts** dans le tableau et cliquez ensuite sur **Download**.
6. Enregistrer le fichier ZIP sur le répertoire souhaité.
7. Veuillez extraire le fichier ZIP.

3.3.3.2. Exécuter les quickstarts dans JBoss Developer Studio

Une fois les quickstarts téléchargés, ceux-ci peuvent être importés dans JBoss Developer Studio et déployés vers JBoss EAP.

Importer un quickstart dans JBoss Developer Studio

Chaque quickstart est fourni avec un fichier POM qui contient des informations de projet et de configuration. À l'aide de ce fichier POM, vous pouvez facilement importer le quickstart dans JBoss Developer Studio.

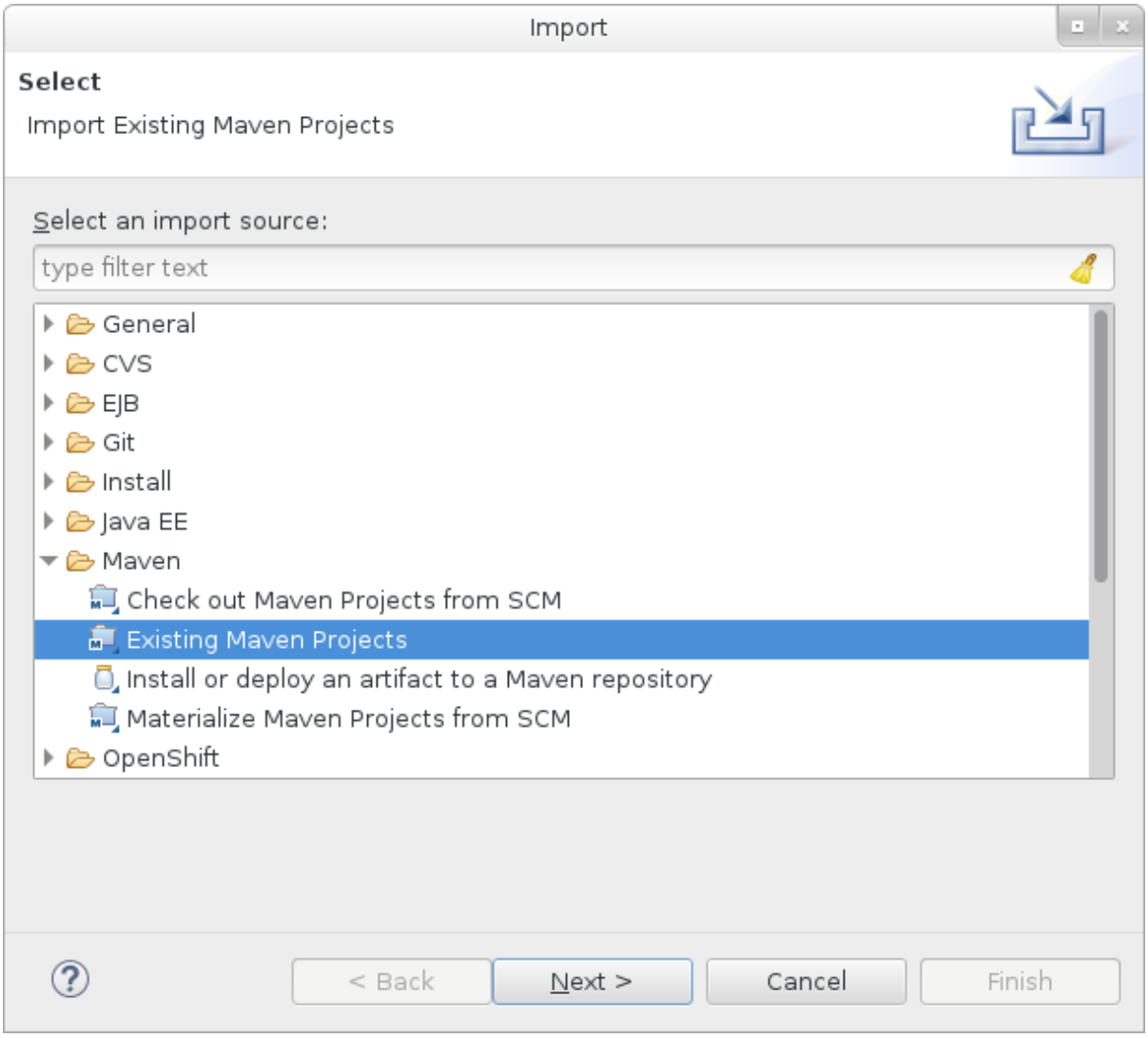


IMPORTANT

Si votre dossier de projet quickstart se situe dans l'espace de travail IDE, lorsque vous l'importez dans JBoss Developer Studio, l'IDE génère un nom de projet non valide et un nom d'archive WAR. N'oubliez pas que votre dossier de projet quickstart doit être situé en dehors de l'espace de travail IDE avant de commencer.

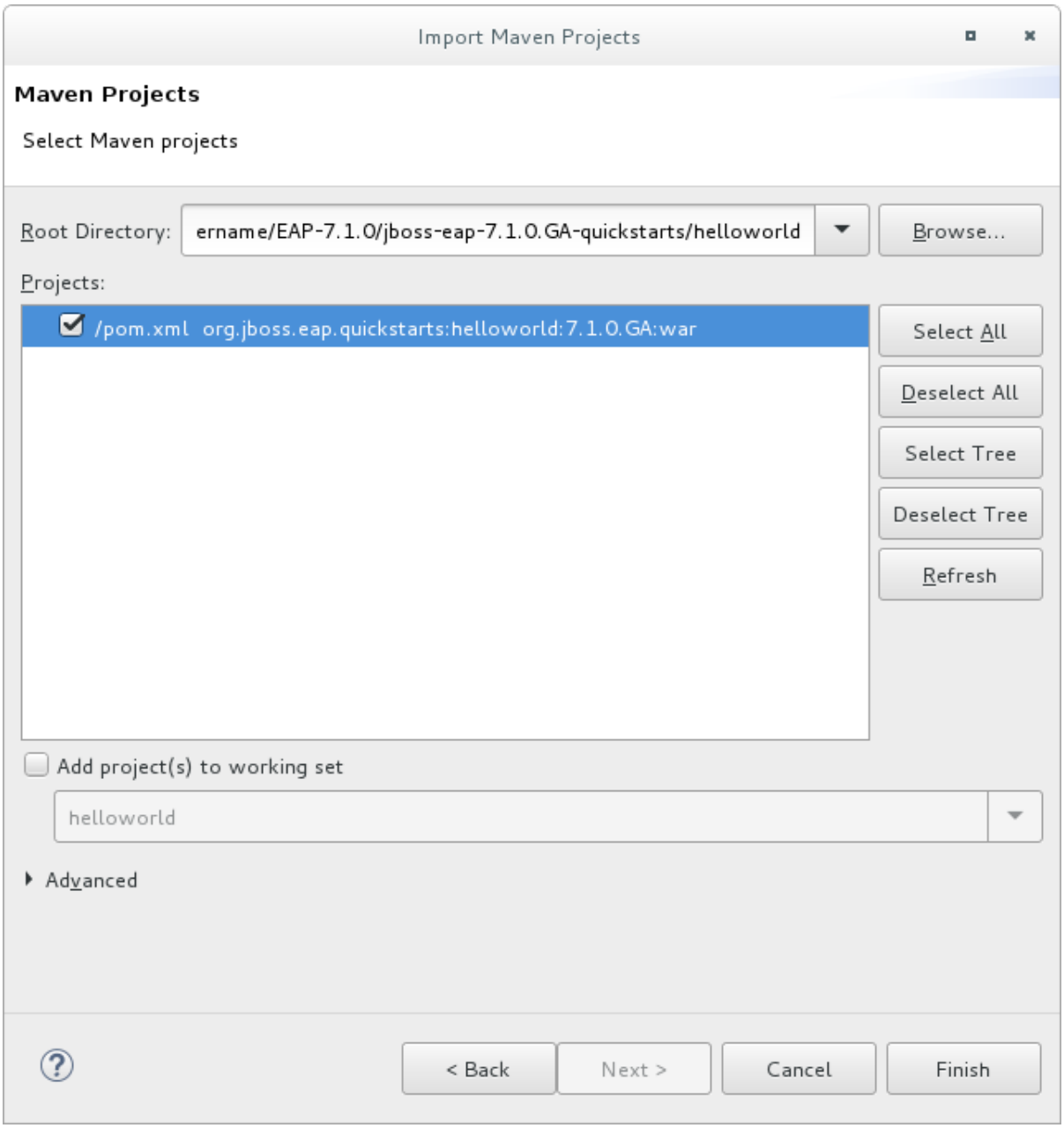
1. Démarrer JBoss Developer Studio.
2. Sélectionner **File** → **Import**.
3. Sélectionner **Maven** → **Projets Maven existants**, puis cliquer sur**Suite**.

Figure 3.1. Importer les projets Maven existants



4. Naviguer vers le répertoire de quickstart souhaité (par exemple le quickstart **`helloworld`**) puis cliquer sur**OK**. Le champ de la liste**Projects** se remplit avec le fichier **`pom.xml`** du projet de quickstart sélectionné.

Figure 3.2. Sélectionner les projets Maven



5. Cliquer sur **Finish**.

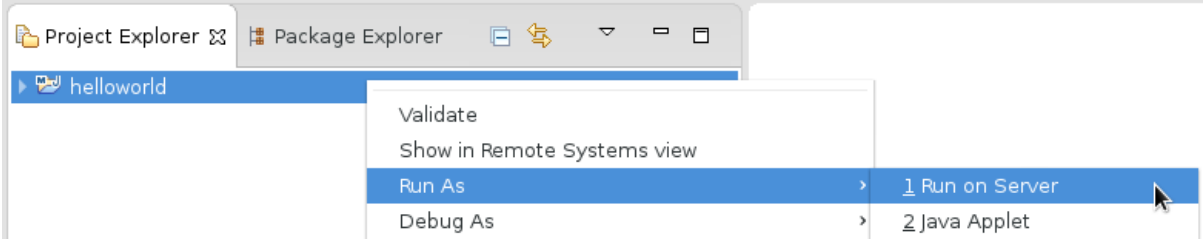
Exécuter le quickstart *helloworld*

Le quickstart **`helloworld`** est une manière simplifiée de vérifier que le serveur JBoss EAP est configuré et exécuté correctement.

1. Si vous n'avez pas encore défini de serveur, ajoutez le serveur JBoss EAP à JBoss Developer Studio. Reportez-vous à la section Using Runtime Detection to Set Up JBoss EAP from within the IDE (Utilisation de la détection d'exécution pour configurer JBoss EAP à partir de l'IDE) du guide *Getting Started with JBoss Developer Studio Tools* (Prise en main de JBoss Developer Studio Tools).

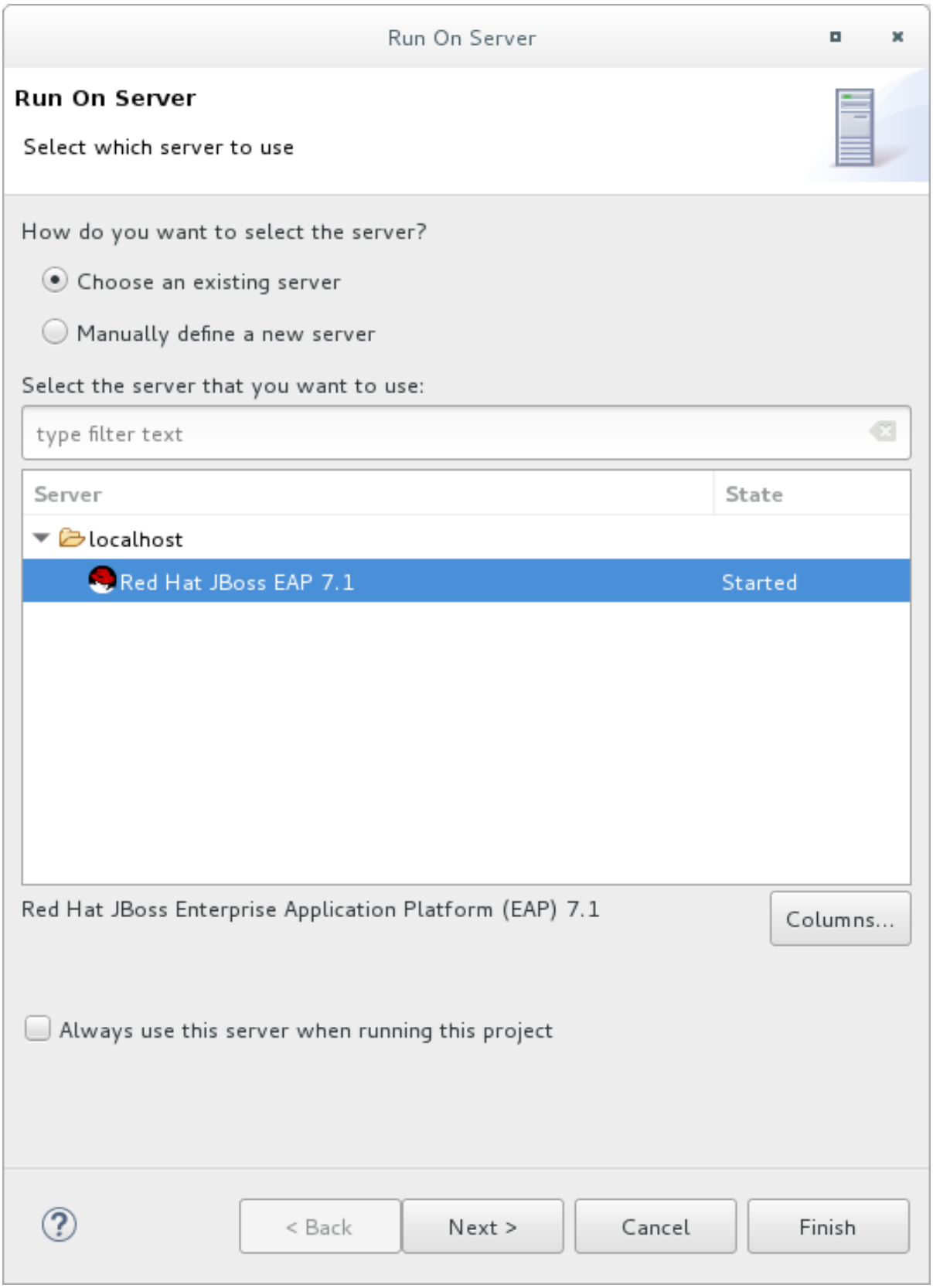
2. Cliquez avec le bouton droit de la souris sur le projet **helloworld** dans l'onglet **Project Explorer** et sélectionnez ensuite **Run As → Run on Server**.

Figure 3.3. Run As - Run on Server



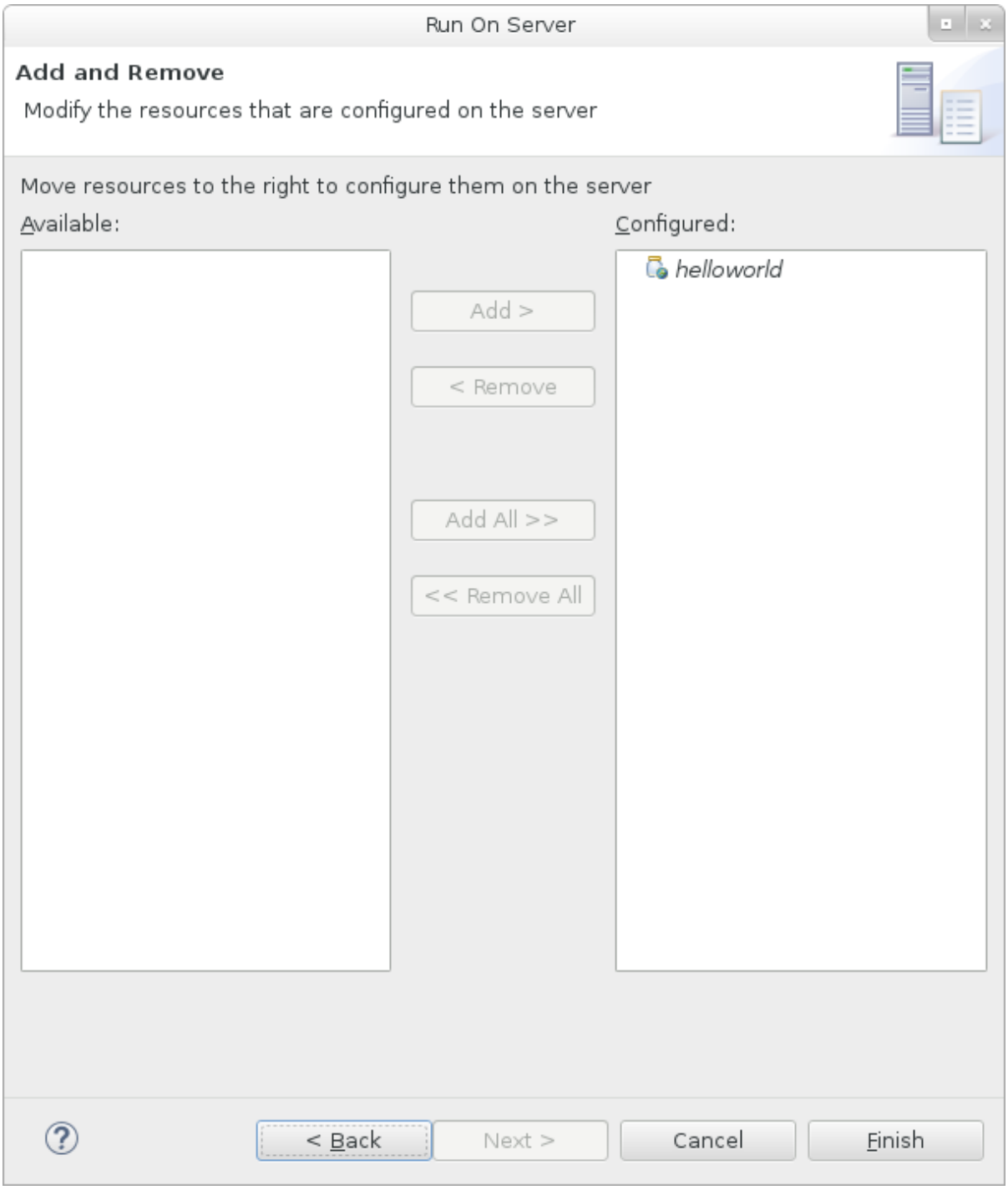
3. Sélectionnez le serveur JBoss EAP 7.1 dans la liste des serveurs, puis cliquez sur **Next**.

Figure 3.4. Run on Server



4. Le Quickstart **helloworld** est déjà répertorié comme devant être configuré sur le serveur. Cliquez sur **Finish** pour le déployer.

Figure 3.5. Modifier les ressources configurées sur le serveur



5. Vérifier les résultats.

   - Dans l'onglet **Server**, l'état du serveur JBoss EAP 7.1 est alors défini sur**Started**.
   - L'onglet **Console** montre des messages détaillant le démarrage du serveur JBoss EAP et le déploiement du quickstart **helloworld**.

     ```
     WFLYUT0021 : contexte web enregistré : /helloworld
     WFLYSRV0010 : "helloworld.war" déployé (runtime-name :
     "helloworld.war")
     ```

   - L'application **helloworld** est disponible à l'adresse http://localhost:8080/helloworld et affiche le texte**Hello World!**.

Pour plus d'informations sur le Quickstart **helloworld**, reportez-vous à la section Explore the helloworld Quickstart (Explorer le Quickstart Helloworld).

Exécuter le quickstart *bean-validation*

Certains quickstarts, comme **bean-validation** quickstart, ne fournissent pas de couche d'interface utilisateur, mais fournissent des tests Arquillian pour démontrer la fonctionnalité à la place.

1. Importer le quickstart **bean-validation** dans JBoss Developer Studio.

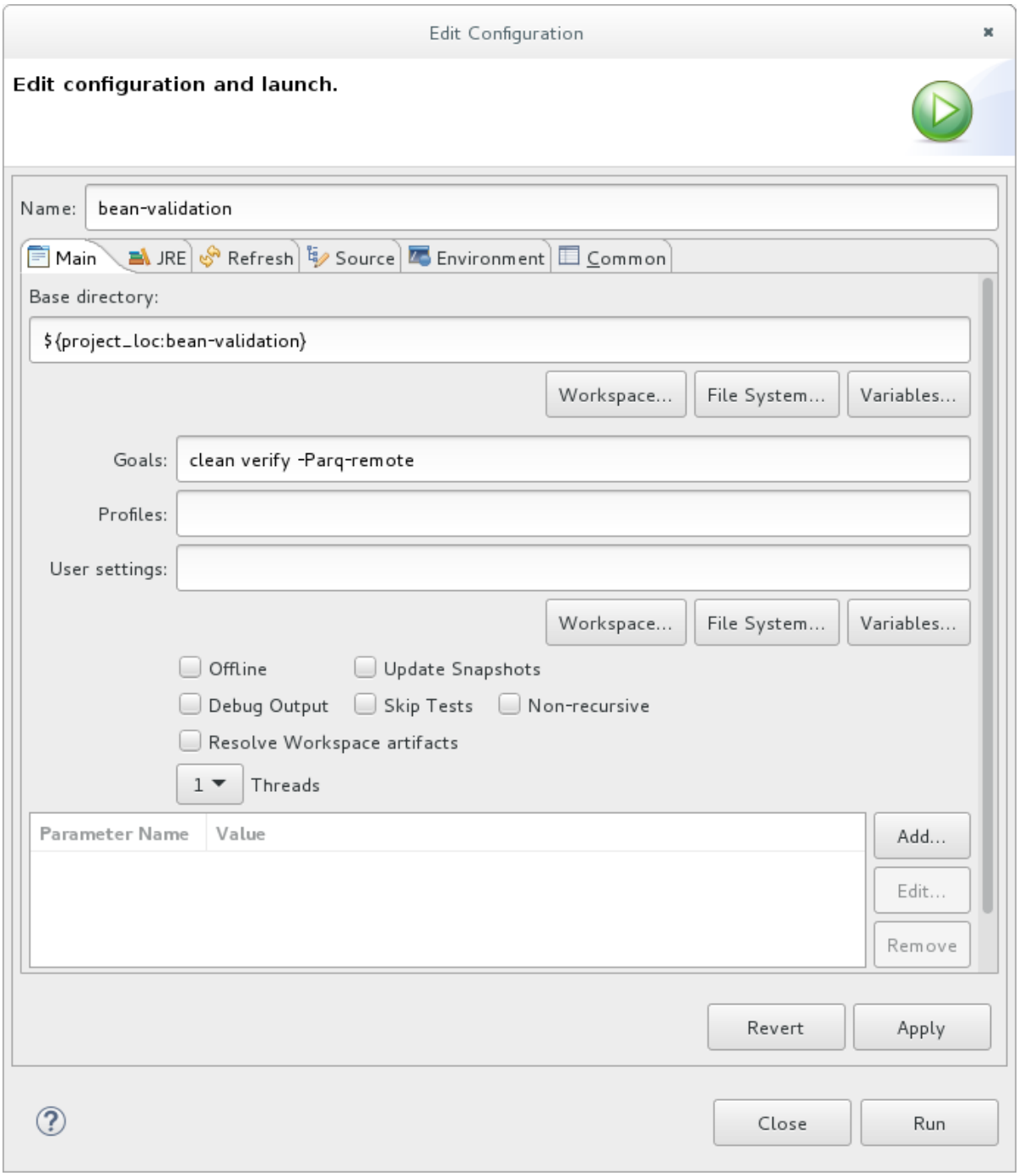
2. Dans l'onglet **Servers**, cliquez avec le bouton droit de la souris sur le serveur et sélectionnez ensuite **Start** pour démarrer le serveur JBoss EAP. Si l'onglet **Servers** n'est pas visible ou si vous n'avez pas encore défini de serveur, ajoutez le serveur JBoss EAP à JBoss Developer Studio. Reportez-vous à la section Using Runtime Detection to Set Up JBoss EAP from within the IDE (Utilisation de la détection d'exécution pour configurer JBoss EAP à partir de l'IDE) du guide *Getting Started with JBoss Developer Studio Tools* (Prise en main de JBoss Developer Studio Tools).

3. Cliquez avec le bouton droit de la souris sur le projet **bean-validation** dans l'onglet **Project Explorer** et sélectionnez ensuite **Run As** → **Maven Build**.

4. Saisir ce qui suit dans le champ de saisie **Goals** puis cliquer sur **Run**.

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

Figure 3.6. Modifier la configuration



5. Vérifier les résultats.
   L'onglet **Console** affiche les résultats des tests Arquillian**`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. Exécuter les Quickstarts à partir de la ligne de commande

Vous pouvez facilement créer et déployer des quickstarts à partir de la ligne de commande en utilisant Maven. Si vous n'avez pas encore installé Maven, voir le projet Apache Maven pour le télécharger et l'installer.

A **README.md** file is provided at the root directory of the quickstarts that contains general information about system requirements, configuring Maven, adding users, and running the quickstarts.

Chaque quickstart contient son propre fichier **README.md** fournissant les instructions spécifiques et les commandes Maven qui servent à exécuter ce quickstart.

Exécuter le quickstart *helloworld* à partir de la ligne de commandes

1. Vérifier le fichier **README.md** dans le répertoire racine du quickstart*helloworld*.
2. Démarrer le serveur JBoss EAP.

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

3. Navuguez dans le répertoire quickstart *helloworld*.
4. Construire et déployer le quickstart à l'aide de la commande Maven fournie dans le fichier **README.md** du quickstart.

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

5. L'application *helloworld* est désormais disponible à l'adresse http://localhost:8080/helloworld et affiche le texte**Hello World!**.

3.4. ANALYSE DES EXEMPLES QUICKSTART

3.4.1. Découvrir le Quickstart HelloWorld

Le Quickstart **helloworld** vous montre comment déployer un servlet simple sur JBoss EAP. La logique métier est encapsulée dans un service, qui est fourni en tant que bean CDI (Contexts and Dependency Injection) et injecté dans le servlet. Ce Quickstart constitue un point de départ pour être sûr que vous avez configuré et démarré correctement votre serveur.

Vous trouverez des informations détaillées sur la création et le déploiement de ce Quickstart à l'aide de la ligne de commande dans le fichier **README.html** situé à la racine du répertoire du Quickstart **helloworld**. Cette rubrique vous explique comment utiliser Red Hat JBoss Developer Studio pour exécuter le Quickstart. Nous partons du principe que vous avez installé Red Hat JBoss Developer Studio, configuré Maven, et importé et exécuté avec succès le Quickstart **helloworld**.

Conditions préalables

- Installez JBoss Developer Studio. Pour plus d'informations, consultez la section Installing JBoss Developer Studio Stand-alone Using the Installer(Installation de JBoss Developer Studio Stand-alone à l'aide du programme d'installation) du *Guide d'installation* de JBoss Developer Studio.
- Exécutez le Quickstart **helloworld**. Pour plus d'informations, reportez-vous à la section Run the Quickstarts in JBoss Developer Studio(Exécution des Quickstarts dans JBoss Developer Studio).
- Vérifiez que le Quickstart **helloworld** a été déployé correctement sur JBoss EAP en ouvrant un navigateur web et en accédant à l'application à l'adresse suivante : http://localhost:8080/helloworld.

Observer la structure du répertoire
Le code du Quickstart **helloworld** est disponible dans le répertoire ***QUICKSTART_HOME*/helloworld/**. Le Quickstart **helloworld** se compose d'un servlet et d'un bean CDI. Il contient également un fichier **beans.xml** dans le répertoire **WEB-INF/** de l'application, dont le numéro de version est 1.1 et le **bean-discovery-mode** est défini sur **all**. Ce fichier marqueur identifie le fichier WAR en tant qu'archive bean, et indique à JBoss EAP de rechercher des beans dans cette application et d'activer le bean CDI.

Le répertoire **src/main/webapp/** contient les fichiers nécessaires au Quickstart. Tous les fichiers de configuration de cet exemple sont situés dans le répertoire **WEB-INF/**, à l'intérieur de **src/main/webapp/**, y compris le fichier**beans.xml**. Le répertoire **src/main/webapp/** contient également un fichier**index.html**, qui utilise une redirection meta refresh simple pour rediriger le navigateur de l'utilisateur vers le servlet, situé à l'adresse suivante : http://localhost:8080/helloworld/HelloWorld. Aucun fichier**web.xml** n'est nécessaire pour le Quickstart.

Examiner le code
La déclaration de paquet et les importations ont été exclues de ces listes. La liste complète est disponible dans le code source du Quickstart.

1. Vérifiez le code **HelloWorldServlet**.
   Le fichier **HelloWorldServlet.java** se trouve dans le répertoire **src/main/java/org/jboss/as/quickstarts/helloworld/**. Ce servlet envoie les informations au navigateur.

   Exemple : code de classe 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,
HttpServletResponse resp) throws ServletException, IOException {
```

```
55          resp.setContentType("text/html");
56          PrintWriter writer = resp.getWriter();
57          writer.println(PAGE_HEADER);
58          writer.println("<h1>" +
helloService.createHelloMessage("World") + "</h1>");
59          writer.println(PAGE_FOOTER);
60          writer.close();
61      }
62
63 }
```

Tableau 3.1. Infos HelloWorldServlet

Ligne	Remarque
43	Il vous suffit simplement d'ajouter l'annotation **@WebServlet** et de fournir un mappage vers une URL utilisée pour accéder au servlet.
46-48	Chaque page web a besoin d'HTML formé correctement. Ce Quickstart utilise les Strings statiques pour écrire les sorties minimum de l'en-tête et du pied de page.
50-51	Ces lignes injectent le bean CDI HelloService qui génère le message proprement dit. À condition que l'API de HelloService ne soit pas modifié, cette méthode permettra de modifier l'implémentation de HelloService ultérieurement sans devoir changer la couche d'affichage.
58	Cette ligne appelle le service pour générer le message "Hello World", et l'écrire dans la requête HTTP.

2. Vérifiez le code **HelloService**.
   Le fichier **HelloService.java** se trouve dans le répertoire **src/main/java/org/jboss/as/quickstarts/helloworld/**. Ce service renvoie simplement un message. Aucun enregistrement d'annotations ou XML n'est requis.

   Exemple : code de classe HelloService

```
public class HelloService {

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

3.4.2. Découvrir le Quickstart numberguess

Le Quickstart **numberguess** vous montre comment créer et déployer une application non persistante simple sur JBoss EAP. Les informations sont affichées à l'aide d'une vue JSF et la logique métier est encapsulée dans deux beans CDI. Dans le Quickstart **numberguess**, vous

disposez de dix tentatives pour deviner un nombre entre 1 et 100. Après chaque tentative, le système vous indique si votre proposition est trop basse ou trop haute.

Le code du Quickstart **numberguess** est disponible dans le répertoire ***QUICKSTART_HOME*/numberguess/**, où ***QUICKSTART_HOME*** est le répertoire dans lequel vous avez téléchargé et décompressé les Quickstarts JBoss EAP. Le Quickstart **numberguess** se compose de plusieurs beans, fichiers de configuration et vues Facelets (JSF). Il est, en outre, empaqueté sous la forme d'un module WAR.

Vous trouverez des informations détaillées sur la création et le déploiement de ce Quickstart à l'aide de la ligne de commande dans le fichier **README.html** situé à la racine du répertoire du Quickstart **numberguess**. Dans les exemples suivants, Red Hat JBoss Developer Studio est utilisé pour exécuter le Quickstart.

Conditions préalables

- Installez JBoss Developer Studio. Pour plus d'informations, consultez la section Installing JBoss Developer Studio Stand-alone Using the Installer(Installation de JBoss Developer Studio Stand-alone à l'aide du programme d'installation) du *Guide d'installation* de JBoss Developer Studio.
- Exécutez le Quickstart **numberguess**. Pour plus d'informations, reportez-vous à la section Run the Quickstarts in JBoss Developer Studio(Exécution des Quickstarts dans JBoss Developer Studio) et remplacez **helloworld** par **numberguess** dans les instructions.
- Vérifiez que le Quickstart **numberguess** a été déployé correctement sur JBoss EAP en ouvrant un navigateur web et en accédant à l'application à l'adresse suivante : http://localhost:8080/numberguess.

Examiner les fichiers de configuration
Tous les fichiers de configuration de cet exemple se trouvent dans le répertoire ***QUICKSTART_HOME*/numberguess/src/main/webapp/WEB-INF/** du Quickstart.

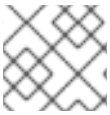
1. Examinez le fichier **faces-config.xml**.
   Ce Quickstart utilise la version JSF 2.2 du nom de fichier **faces-config.xml**. Une version normalisée de Facelets est le gestionnaire de vues par défaut dans JSF 2.2 ; aucune configuration n'est donc requise. Ce fichier se compose uniquement de l'élément racine. Il s'agit simplement d'un fichier marqueur utilisé pour indiquer que JSF doit être activé dans l'application.

   ```
   <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. Examinez le fichier **beans.xml**.
   Le fichier **beans.xml** contient le numéro de version 1.1 et le**bean-discovery-mode** est défini sur **all**. Il s'agit d'un fichier marqueur qui identifie le fichier WAR en tant qu'archive bean, et indique à JBoss EAP de rechercher des beans dans cette application et d'activer le bean 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>
```



NOTE

Ce Quickstart ne nécessite pas de fichier **web.xml**.

3.4.2.1. Examiner le code JSF

JSF utilise l'extension **.xhtml** pour les fichiers sources, mais génère les vues rendues avec l'extension **.jsf**. Le fichier**home.xhtml** se trouve dans le répertoire**src/main/webapp/**.

Exemple : code de source 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 <title>Numberguess</title>
27 </head>
28
29 <body>
30 <div id="content">
31  <h1>Guess a number...</h1>
32  <h:form id="numberGuess">
33
34  <!-- Feedback for the user on their guess -->
35 <div style="color: red">
36  <h:messages id="messages" globalOnly="false" />
37  <h:outputText id="Higher" value="Higher!"
38     rendered="#{game.number gt game.guess and game.guess ne 0}" />
39  <h:outputText id="Lower" value="Lower!"
40     rendered="#{game.number lt game.guess and game.guess ne 0}" />
41 </div>
42
43 <!-- Instructions for the user -->
44 <div>
45 I'm thinking of a number between <span
46 id="numberGuess:smallest">#{game.smallest}</span> and <span
47 id="numberGuess:biggest">#{game.biggest}</span>. You have
48 #{game.remainingGuesses} guesses remaining.
49 </div>
50
51 <!-- Input box for the users guess, plus a button to submit, and reset
-->
52 <!-- These are bound using EL to our CDI beans -->
```

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

Les numéros de ligne suivants correspondent à ceux qui sont affichés lors de la consultation du fichier dans JBoss Developer Studio.

Tableau 3.2. Infos JSF

Ligne	Remarque
36-40	Voici les messages qui peuvent être envoyés par l'utilisateur : "Higher!" et "Lower!"
45-48	Au fur et à mesure que l'utilisateur devine, l'étendue des nombres qu'ils devinent se rétrécit. Cette phrase change pour s'assurer qu'ils connaissent la portée d'une tentative valide.
55-58	Ce champ d'entrée est lié à une propriété de bean qui utilise une expression de valeur.
58	On utilise une liaison de validateur pour s'assurer que l'utilisateur ne mette pas le nombre qu'il devine en dehors de la limite. Si le validateur n'était pas présent, l'utilisateur peut deviner un nombre en dehors de la limite.
59-61	Il doit y avoir un moyen pour que l'utilisateur envoie le nombre qu'il devine au serveur. Ici, on associe une méthode d'action sur le bean.

3.4.2.2. Examiner les fichiers de classe

Tous les fichiers sources du Quickstart **numberguess** se trouvent dans le répertoire ***QUICKSTART_HOME*/numberguess/src/main/java/org/jboss/as/quickstarts/numberguess/**. La déclaration de paquetage et les importations ont été exclues de ces listes. La liste

complète est disponible dans le code source du Quickstart.

1. Vérification du code du qualificateur **Random.java**
   Un qualificateur est utilisé pour supprimer toute ambiguïté entre deux beans réunissant les conditions nécessaires pour une injection en fonction de leur type. Pour plus d'informations sur les qualificateurs, reportez-vous à la section Use a Qualifier to Resolve an Ambiguous Injection (Utilisation d'un qualificateur pour résoudre une injection ambiguë) du *Guide de développement* de JBoss EAP. Le qualificateur **@Random** est utilisé pour l'injection d'un nombre aléatoire.

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

}
```

2. Vérification du code du qualificateur **MaxNumber.java**
   Le **qualificateur @MaxNumber** est utilisé pour injecter le nombre maximum autorisé.

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

3. Vérification du code **Generator.java**
   La classe **Generator** crée le nombre aléatoire au moyen d'une méthode producteur, exposant ainsi le nombre maximum autorisé via celle-ci. Cette classe étant basée sur l'étendue de l'application, vous recevez à chaque fois le même nombre aléatoire.

```
@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. Vérification du code **Game.java**
   La classe **Game** basée sur la session constitue le point d'entrée principal de l'application. Elle est chargée de configurer ou de réinitialiser le jeu, de collecter et de valider les propositions de l'utilisateur, et de fournir des commentaires à l'utilisateur avec un **FacesMessage**. Elle utilise la méthode de cycle de vie postérieure à la construction (PostConstruct) pour initialiser le jeu en récupérant un nombre aléatoire du bean **@Random Instance<Integer>**.

   Notez l'annotation **@Named** dans cette classe. Cette annotation n'est nécessaire que si vous souhaitez que le bean soit accessible à une vue JSF en utilisant le langage d'expression (EL) ; dans ce cas **#{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
{@linkplain PostConstruct @PostConstruct} to set the initial
```

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

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

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

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

ANNEXE A. MATÉRIEL DE RÉFÉRENCE

A.1. ARGUMENTS DE TEMPS D'EXÉCUTION DU SERVEUR

Le script de démarrage du serveur d'applications accepte les arguments et s'active lors du runtime. Cela permet au serveur de démarrer dans une configuration autre que celle définie dans les fichiers de configuration **`standalone.xml`**, **`domain.xml`** et **`host.xml`**.

Les configurations alternatives peuvent signifier de démarrer le serveur avec un ensemble de liaisons de sockets alternatif ou une configuration secondaire.

La liste des paramètres disponibles est accessible en exécutant l'aide avec l'option **`-h`** ou **`--help`** lors du démarrage.

Tableau A.1. Arguments et options de Runtime

Argument ou Option	Mode d'opération	Description
--admin-only	Autonome	Définissez le type d'exécution du serveur sur **`ADMIN_ONLY`**. De cette manière, il ouvrira des interfaces d'administration et acceptera les demandes de gestion ; en revanche, il ne lancera pas d'autres services d'exécution et n'acceptera pas les demandes des utilisateurs finaux. Notez qu'il est recommandé d'utiliser **`--start-mode=admin-only`** à la place.
--admin-only	Domaine	Définir le type d'exécution du contrôleur hôte à **`ADMIN_ONLY`**, ce qui le fera ouvrir les interfaces administratives et il pourra ainsi accepter les requêtes de gestion, mais il ne pourra pas démarrer d'autres serveurs ou, si ce contrôleur hôte est le master du domaine, il pourra accepter les requêtes des contrôleurs hôte esclaves.
-b=<value>, -b <value>	Autonome, Serveur	Définissez la propriété système **`jboss.bind.address`**, utilisée dans la configuration de l'adresse de liaison pour l'interface publique. La valeur par défaut est **`127.0.0.1`** si aucune valeur n'est spécifiée. Reportez-vous à l'entrée **`-b<interface>=<value>`** pour définir l'adresse de liaison pour d'autres interfaces.
-b<interface>=<value>	Autonome, Serveur	Définissez la propriété système **`jboss.bind.address.<interface>`** sur la valeur donnée. Par exemple, **`-bmanagement=`*`IP_ADDRESS`***.

Argument ou Option	Mode d'opération	Description
--backup	Domaine	Conservez une copie de la configuration de domaine persistante même si cet hôte n'est pas le contrôleur de domaine.
-c=<config>, -c <config>	Autonome	Nommer le fichier de configuration du serveur à utiliser. La valeur par défaut est **`standalone.xml`**.
-c=<config>, -c <config>	Domaine	Nommer le fichier de configuration du serveur à utiliser. La valeur par défaut est **`domain.xml`**.
--cached-dc	Domaine	Si l'hôte n'est pas le contrôleur de domaine et ne peut pas contacter le contrôleur de domaine au démarrage, démarrez en utilisant une copie de la configuration de domaine mise en cache localement.
--debug [<port>]	Autonome	Active le mode de débogage par un argument en option qui indique le port. Ne fonctionne que si le script de lancement le supporte.
-D<name>[=<value>]	Autonome, Serveur	Définit une propriété système.
--domain-config=<config>	Domaine	Nommer le fichier de configuration du serveur à utiliser. La valeur par défaut est **`domain.xml`**.
-h, --help	Autonome, Serveur	Affiche le message d'assistance et sort.
--host-config=<config>	Domaine	Nom du fichier de configuration hôte à utiliser. La valeur par défaut est **`host.xml`**.
--interprocess-hc-address=<address>	Domaine	Adresse à laquelle le contrôleur hôte doit écouter la communication en provenance du contrôleur de processus.
--interprocess-hc-port=<port>	Domaine	Port sur lequel le contrôleur hôte doit écouter la communication en provenance du contrôleur de processus.

Argument ou Option	Mode d'opération	Description
--master-address=<address>	Domaine	Définissez la propriété système **jboss.domain.master.address** sur la valeur donnée. Dans une configuration de contrôleur hôte esclave par défaut, cette valeur est utilisée pour configurer l'adresse du contrôleur hôte maître.
--master-port=<port>	Domaine	Définissez la propriété système **jboss.domain.master.port** sur la valeur donnée. Dans une configuration de contrôleur hôte esclave par défaut, cette valeur est utilisée pour configurer le port utilisé dans le cadre de la communication de gestion native par le contrôleur hôte maître.
--read-only-server-config=<config>	Autonome	Nom du fichier de configuration du serveur à utiliser. Cela diffère de **--server-config** et **-c** en ce que le fichier d'origine n'est jamais écrasé.
--read-only-domain-config=<config>	Domaine	Nom du fichier de configuration du domaine à utiliser. Cela diffère de **--domain-config** et de **-c** en ce que le fichier de départ n'est jamais écrasé.
--read-only-host-config=<config>	Domaine	Nom du fichier de configuration de l'hôte à utiliser. Cela diffère de **--host-config** en ce que le fichier de départ n'est jamais écrasé.
-P=<url>, -P <url>, --properties=<url>	Autonome, Serveur	Télécharge les propriétés système de l'URL donné.
--pc-address=<address>	Domaine	Adresse à laquelle le contrôleur de processus doit écouter les communications en provenance des processus qu'il contrôle.
--pc-port=<port>	Domaine	Port sur lequel le contrôleur de processus doit écouter les communications en provenance des processus qu'il contrôle.
-S<name>[=<value>]	Autonome	Définit une propriété de sécurité.
-secmgr	Autonome, Serveur	Exécute le serveur avec un gestionnaire de sécurité installé.
--server-config=<config>	Autonome	Nommer le fichier de configuration du serveur à utiliser. La valeur par défaut est **standalone.xml**.

Argument ou Option	Mode d'opération	Description
--start-mode=<mode>	Autonome	Définissez le mode de démarrage du serveur. Cette option ne peut pas être utilisée avec **--admin-only**. Les valeurs acceptées sont les suivantes : • **normal** : le serveur va démarrer normalement. • **admin-only** : le serveur va ouvrir des interfaces d'administration et accepter les demandes de gestion. En revanche, il ne lancera pas d'autres services d'exécution et n'acceptera pas les demandes des utilisateurs finaux. • **suspend** : le serveur va démarrer en mode suspendu et ne satisfera pas les demandes tant qu'il n'aura pas été redémarré.
-u=<value>, -u <value>	Autonome, Serveur	Définissez la propriété système **jboss.default.multicast.address**, laquelle est utilisée pour configurer l'adresse de multidiffusion des éléments de liaison de sockets dans les fichiers de configuration. Elle est définie par défaut sur **230.0.0.4** si aucune valeur n'est spécifiée.
-v, -V, --version	Autonome, Serveur	Affiche la version du serveur d'applications et sort.

AVERTISSEMENT

Les fichiers de configuration compris dans JBoss EAP sont configurés pour gérer les comportements des options ; **-b** et **-u**, par exemple. Si vous modifiez vos fichiers de configuration afin de ne plus utiliser la propriété système contrôlée par l'option, son ajout à la commande de lancement sera alors sans effet.

A.2. ARGUMENTS DE L'UTILITAIRE ADD-USER

Le tableau suivant décrit les arguments disponibles pour le script **add-user.sh** ou **add-user.bat**, un utilitaire pour ajouter de nouveaux utilisateurs au fichier de propriété pour une authentification non courante.

Tableau A.2. Arguments pour la commande Add-User

Argument de ligne de commande	Description
-a	Crée un utilisateur dans le domaine d'application. S'il est omis, un utilisateur est créé par défaut dans le domaine de gestion.
-dc <value>	Le répertoire de configuration de domaine contiendra les fichiers de propriétés. S'il est omis, le répertoire par défaut est ***EAP_HOME*/domain/configuration/**.
-sc <value>	Autre répertoire de configuration de serveur autonome où seront stockés les fichiers de propriétés. S'il est omis, le répertoire par défaut est ***EAP_HOME*/standalone/configuration/**.
-up, --user-properties <value>	Nom de l'autre fichier de propriétés utilisateur. Il peut s'agir d'un chemin absolu ou d'un nom de fichier utilisé avec l'argument **-sc** ou **-dc** qui spécifie l'autre répertoire de configuration.
-g, --group <value>	Une liste séparée par des virgules de groupes à assigner à cet utilisateur.
-gp, --group-properties <value>	Nom d'un autre fichier de propriétés de groupe. Il peut s'agir d'un chemin absolu ou d'un nom de fichier utilisé avec l'argument **-sc** ou **-dc** qui spécifie l'autre répertoire de configuration.
-p, --password <value>	Mot de passe associé à l'utilisateur.
-u, --user <value>	Nom de l'utilisateur. Les noms d'utilisateur ne peuvent contenir que les caractères suivants (quel que soit leur nombre et dans n'importe quel ordre) : • Caractères alphanumériques (a-z, A-Z, 0-9) • Tirets (-), points (.), virgules (,), arobase (@) • Barre oblique inverse (\) • Égal (=)
-r, --realm <value>	Le nom du domaine utilisé pour sécuriser les interfaces de gestion. S'il est omis, la valeur par défaut sera **ManagementRealm**.
-s, --silent	Exécuter le script **add-user** sans sortie vers la console.
-e, --enable	Active l'utilisateur.
-d, --disable	Désactive l'utilisateur.

Argument de ligne de commande	Description
-cw, --confirm-warning	Confirme les avertissements automatiquement en mode interactif.
-h, --help	Affiche les informations d'utilisation du script **add-user**.
-ds, --display-secret	Imprimez la valeur secrète en mode non interactif.

A.3. ATTRIBUTS D'INTERFACE

NOTE

Les noms d'attribut affichés dans ce tableau sont répertoriés tels qu'ils apparaissent dans le modèle de gestion ; par exemple, lors de l'utilisation de l'interface de ligne de commande de gestion. Consultez le fichier de définition de schéma situé dans ***EAP_HOME*/docs/schema/wildfly-config_5_0.xsd** pour afficher les éléments tels qu'ils apparaissent dans le XML, dans la mesure où il peut y avoir des différences par rapport au modèle de gestion.

Tableau A.3. Attributs et valeurs d'interface

Élément d'interface	Description
any	Élément indiquant qu'une partie du critère de sélection d'une interface doit être qu'elle remplisse au moins un, mais pas forcément tous, les groupes de critères imbriqués.
any-address	Élément vide indiquant que les sockets qui utilisent cette interface doivent être liés à une adresse générique. L'adresse générique IPv6 (**::**) sera utilisée à moins que la propriété système **java.net.preferIPv4Stack** soit définie sur true, dans lequel cas, l'adresse générique (**0.0.0.0**) IPv4 sera utilisée. Si un socket est lié à une adresse anylocal IPv6 sur une machine dual-stack, il pourra accepter le trafic IPv6 et IPv4 ; si lié à l'adresse IPv4 anylocal (mappées IPv4), il ne peut accepter que le trafic IPv4.
inet-address	Soit une adresse IP en notation à points IPV6 ou IPV4, ou un nom d'hôte pouvant être résolu.
link-local-address	Élément vide indiquant qu'une partie du critère de sélection d'une interface devrait consister à savoir si oui ou non il a une adresse associée local-link.
loopback	Élément vide indiquant qu'une partie du critère de sélection d'une interface est de savoir s'il s'agit oui ou non d'une interface de loopback.

Élément d'interface	Description
loopback-address	Une adresse de loopback qui ne peut pas réellement être configurée sur l'interface de loopback de la machine. Diffère du type d'adresse inet car la valeur donnée sera utilisée même si aucune carte réseau possédant l'adresse IP associée ne peut être trouvée.
multicast	Élément vide indiquant qu'une partie du critère de sélection d'une interface doit être si oui ou non il y a un support multi-diffusion.
name	Nom de l'interface.
nic	Le nom d'une interface de réseau (e.g. eth0, eth1, lo).
nic-match	Une expression standard à laquelle faire correspondre les noms des interfaces de réseau disponibles sur la machine pour trouver une interface qui convienne.
not	Élément indiquant qu'une partie du critère de sélection d'une interface doit être qu'elle ne remplisse aucun des ensembles de critères imbriqués.
point-to-point	Élément vide indiquant qu'une partie du critère de sélection d'une interface doit être de savoir si elle a oui ou non une interface d'un point à un autre.
public-address	Élément vide indiquant qu'une partie du critère de sélection d'une interface doit être de savoir si elle a oui ou non une adresse publiquement routable.
site-local-address	Élément vide indiquant qu'une partie du critère de sélection d'une interface doit être ou non une adresse associée à son site-local
subnet-match	Adresse IP réseau et nombre de bits dans le préfixe réseau de l'adresse, sous la forme *slash notation* ; **`192.168.0.0/16`**, par exemple.
up	Élément vide indiquant qu'une partie du critère de sélection d'une interface est active ou non.
virtual	Élément vide indiquant qu'une partie du critère de sélection d'une interface doit être ou non une interface virtuelle.

A.4. ATTRIBUTS DE LIAISONS DE SOCKETS

NOTE

Les noms d'attribut affichés dans ce tableau sont répertoriés tels qu'ils apparaissent dans le modèle de gestion ; par exemple, lors de l'utilisation de l'interface de ligne de commande de gestion. Consultez le fichier de définition de schéma situé dans ***EAP_HOME*/docs/schema/wildfly-config_5_0.xsd** pour afficher les éléments tels qu'ils apparaissent dans le XML, dans la mesure où il peut y avoir des différences par rapport au modèle de gestion.

Tableau A.4. Attributs de liaisons de sockets

Attribut	Description
client-mappings	Spécifie les mappages de client pour cette liaison de socket. Un client qui se connecte à ce socket doit utiliser l'adresse de destination indiquée dans le mappage qui correspond à son interface de sortie désirée. Cela permet aux topologies de réseau avancées, qui utilisent une traduction d'adresses réseau ou qui ont des liaisons sur plusieurs interfaces de réseau, de fonctionner. Chaque mappage doit être évalué dans l'ordre déclaré, avec la première correspondance trouvée utilisée pour déterminer la destination.
fixed-port	Indique si la valeur de port doit rester fixe quand les décallages numériques sont appliqués aux autres sockets du groupe de sockets.
interface	Nom de l'interface à laquelle le socket doit être lié, ou, pour les sockets de multidiffusion, l'interface sur laquelle écouter. Cela doit correspondre à l'une des interfaces déclarées. Si non définie, la valeur de l'attribut **d'interface par défaut** du groupe de liaison de sockets englobant sera utilisée.
multicast-address	Adresse multidiffusion sur laquelle le socket doit recevoir le trafic multidiffusion. Si non spécifié, le socket ne sera pas configuré pour recevoir le trafic multidiffusion.
multicast-port	Port sur lequel le socket doit recevoir le trafic de multidiffusion. Il doit être configuré si **multicast-address** est configuré.
name	Le nom du socket. Les services ayant besoin d'accéder aux informations de configuration de socket pourront le faire par l'intermédiaire de son nom. Cet attribut est exigé.
port	Le numéro du port dans lequel le socket doit être lié. Notez que cette valeur peut être remplacée si les serveurs appliquent une valeur de décalage de port pour incrémenter ou décrémenter toutes les valeurs de port.

A.5. LIAISONS DE SOCKET PAR DÉFAUT

Les tableaux suivants présentent les liaisons de sockets par défaut pour chaque groupe de liaisons de sockets.

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

Tableau A.5. standard-sockets

Liaison de sockets	Port	Description
ajp	8009	Protocole Apache JServ. Utilisé pour le clustering HTTP et pour l'équilibrage des charges.
http	8080	Le port par défaut des applications déployées.
https	8443	Connexion cryptée-SSL entre les applications déployées et les clients.
management-http	9990	Utilisé pour les communications HTTP avec la couche de gestion.
management-https	9993	Utilisé pour les communications HTTPS avec la couche de gestion.
txn-recovery-environment	4712	Gestionnaire de recouvrement des transactions JTA.
txn-status-manager	4713	Gestionnaire des transactions JTA / JTS.

Tableau A.6. ha-sockets

Liaison de sockets	Port	Port multi-diffusion	Description
ajp	8009		Protocole Apache JServ. Utilisé pour le clustering HTTP et pour l'équilibrage des charges.
http	8080		Le port par défaut des applications déployées.
https	8443		Connexion cryptée-SSL entre les applications déployées et les clients.

Liaison de sockets	Port	Port multi-diffusion	Description
jgroups-mping		45700	Multidiffusion. Utilisée pour découvrir l'appartenance de groupe d'origine dans un cluster HA.
jgroups-tcp	7600		Découverte d'homoplogues unicastes dans les groupements HA avec TCP.
jgroups-udp	55200	45688	Découverte de paires multicast dans les groupements HA avec UDP.
management-http	9990		Utilisé pour les communications HTTP avec la couche de gestion.
management-https	9993		Utilisé pour les communications HTTPS avec la couche de gestion.
modcluster		23364	Port de multidiffusion pour la communication entre JBoss EAP 6 et l'équilibreur de charges HTTP.
txn-recovery-environment	4712		Gestionnaire de recouvrement des transactions JTA.
txn-status-manager	4713		Gestionnaire des transactions JTA / JTS.

Tableau A.7. full-sockets

Liaison de sockets	Port	Description
ajp	8009	Protocole Apache JServ. Utilisé pour le clustering HTTP et pour l'équilibrage des charges.
http	8080	Le port par défaut des applications déployées.
https	8443	Connexion cryptée-SSL entre les applications déployées et les clients.
iiop	3528	Services CORBA pour les transactions JTS et autres services dépendants-ORB.

Liaison de sockets	Port	Description
iiop-ssl	3529	Services CORBA cryptés-SSL.
management-http	9990	Utilisé pour les communications HTTP avec la couche de gestion.
management-https	9993	Utilisé pour les communications HTTPS avec la couche de gestion.
txn-recovery-environment	4712	Gestionnaire de recouvrement des transactions JTA.
txn-status-manager	4713	Gestionnaire des transactions JTA / JTS.

Tableau A.8. full-ha-sockets

Nom	Port	Port multi-diffusion	Description
ajp	8009		Protocole Apache JServ. Utilisé pour le clustering HTTP et pour l'équilibrage des charges.
http	8080		Le port par défaut des applications déployées.
https	8443		Connexion cryptée-SSL entre les applications déployées et les clients.
iiop	3528		Services CORBA pour les transactions JTS et autres services dépendants-ORB.
iiop-ssl	3529		Services CORBA cryptés-SSL.
jgroups-mping		45700	Multidiffusion. Utilisée pour découvrir l'appartenance de groupe d'origine dans un cluster HA.
jgroups-tcp	7600		Découverte d'homoplogues unicastes dans les groupements HA avec TCP.
jgroups-udp	55200	45688	Découverte de paires multicast dans les groupements HA avec UDP.

Nom	Port	Port multi-diffusion	Description
management-http	9990		Utilisé pour les communications HTTP avec la couche de gestion.
management-https	9993		Utilisé pour les communications HTTPS avec la couche de gestion.
modcluster		23364	Port de multidiffusion pour la communication entre JBoss EAP 6 et l'équilibreur de charges HTTP.
txn-recovery-environment	4712		Gestionnaire de recouvrement des transactions JTA.
txn-status-manager	4713		Gestionnaire des transactions JTA / JTS.

Tableau A.9. load-balancer-sockets

Nom	Port	Port multi-diffusion	Description
http	8080		Le port par défaut des applications déployées.
https	8443		Connexion cryptée-SSL entre les applications déployées et les clients.
management-http	9990		Utilisé pour les communications HTTP avec la couche de gestion.
management-https	9993		Utilisé pour les communications HTTPS avec la couche de gestion.
mcmp-management	8090		Port à utiliser pour le protocole MCMP (Mod-Cluster Management Protocol) en vue de la transmission d'événements de cycle de vie.
modcluster		23364	Port de multidiffusion pour la communication entre JBoss EAP 6 et l'équilibreur de charges HTTP.

Revised on 2018-01-11 05:43:14 EST