Red Hat JBoss Enterprise Application Platform 7.3

管理 CLI 指南

使用红帽 JBoss 企业应用平台管理 CLI 配置、使用和管理 JBoss EAP 的说明和命令示例。

Last Updated: 2023-03-27

Red Hat JBoss Enterprise Application Platform 7.3 管理 CLI 指南

使用红帽 JBoss 企业应用平台管理 CLI 配置、使用和管理 JBoss EAP 的说明和命令示例。

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

法律通告

摘要

本文档提供有关 JBoss EAP 管理 CLI 的一般信息。它包含许多示例，演示了如何使用 CLI 来管理和配置红帽 JBoss 企业应用平台。《配置指南》、配置消息传递和其他 JBoss EAP 文档提供了额外的 CLI 命令示例，以及使用管理 CLI 完成特定管理任务的详细信息。

目录

第 1 章 管理 CLI 概述 5

第 2 章 管理 CLI 入门 6
2.1. 启动管理 CLI 6
2.2. 连接到服务器 6
2.3. 获得帮助 6
2.4. 退出管理 CLI 7
2.5. 以非互动模式运行 7
传递命令 7
传递命令文件 8

第 3 章 管理 CLI 导航 9
3.1. 更改当前路径 9
3.2. 打印当前路径 9
3.3. 列出内容 9
3.4. 查看多页输出 10
浏览多页输出 10
搜索多页输出 11
3.5. 使用键盘导航快捷方式 11

第 4 章 创建和执行请求 13
构建操作请求 13
4.1. 显示资源值 14
包括运行时属性 15
递归读取子资源 15
排除默认值 16
resolve Expressions 17
4.2. 显示资源描述 17
4.3. 显示属性值 19
4.4. 更新属性 19
4.5. 取消定义属性 19
4.6. 显示操作名称 20
4.7. 显示操作描述 21
4.8. 使用特殊字符添加值 22
空白 22
引号 22
逗号 22
括号 23
大括号 23
方括号 23
diacritic Marks 23
4.9. 指定操作标头 23
4.10. 使用 IF-ELSE CONTROL FLOW 24
4.11. 使用 TRY-CATCH-FINALLY CONTROL FLOW 26
4.12. 使用 FOR-DONE CONTROL FLOW 26
4.13. 查询资源 27
4.14. 重定向输出 28
将输出重定向到文件 29
将输出重定向到命令 29

第 5 章 通过受管域使用管理 CLI 31
为子系统配置指定 Profile 31

为核心管理和运行时命令指定主机 31
为核心管理和运行时命令指定服务器 32

第 6 章 配置管理 CLI **34**
6.1. 属性替换 38
6.2. 创建别名 39
6.3. .JBOSSCLIRC 配置文件 40
.jbossclirc 文件示例 40
6.4. 使用变量 41
使用 Set 命令 41
使用 Unset 命令 41
使用 jbossclirc 文件 41
使用 Echo 命令 42
示例 42

第 7 章 管理 CLI 命令历史记录 **44**
查看管理 CLI 命令历史记录 44
清除管理 CLI 命令历史记录 44
启用管理 CLI 命令历史记录 44
禁用管理 CLI 命令历史记录 44

第 8 章 管理 CLI 日志 **45**
配置管理 CLI 日志 45

第 9 章 批处理 **46**
外部文件中的批处理命令 46

第 10 章 嵌入服务器以进行离线配置 **48**
启动嵌入式单机服务器 48
指定服务器配置 49
从仅限管理员模式开始 49
控制标准输出 49
引导超时 49
从空白配置开始 49
启动嵌入式主机控制器 50
指定主机控制器配置 50
控制标准输出 51
引导超时 51
使用管理 CLI 进行非修改类加载 51

第 11 章 如何... ..... **53**
11.1. 添加数据源 53
11.2. 添加扩展 53
11.3. 添加 JMS QUEUE 53
11.4. 添加 JMS 主题 53
11.5. 添加模块 53
11.6. 添加服务器 54
11.7. 添加服务器组 54
11.8. 添加系统属性 54
11.9. 克隆配置集 54
11.10. 创建层次结构配置集 55
11.11. 将应用程序部署到受管域 55
11.12. 将应用程序部署到单机服务器 55
11.13. 禁用所有应用程序 56
11.14. 显示活动用户 56

11.15. 显示 ATTACHMENT 的内容 56
11.16. 显示架构信息 57
11.17. 显示系统和服务器信息 57
11.18. 启用所有禁用的部署 58
11.19. 获取命令超时值 59
11.20. 重新加载主机控制器 59
11.21. 以管理员模式重新加载主机控制器 59
11.22. 重新加载服务器组中的所有服务器 59
11.23. 重新加载服务器 60
11.24. 重新加载单机服务器 60
11.25. 删除扩展 60
11.26. 删除模块 60
11.27. 重置命令超时值 61
11.28. 重启服务器组中的所有服务器 61
11.29. 重启服务器 62
11.30. 保存附件的内容 62
11.31. 设置命令超时值 63
11.32. 关闭主机控制器 63
11.33. 关闭服务器 63
11.34. 启动服务器组中的所有服务器 64
11.35. 启动服务器 64
11.36. 停止服务器组中的所有服务器 64
11.37. 停止服务器 65
11.38. 拍摄配置快照 65
11.39. 取消部署所有应用 65
11.40. 从受管域中取消部署应用 66
11.41. 从单机服务器取消部署应用 66
11.42. 更新主机名 66
11.43. 上传附件 66
11.44. 查看服务器日志 67

附录 A. 参考资料 **68**
A.1. 管理 CLI 启动参数 68
A.2. 管理 CLI 批处理模式命令 69
A.3. 管理 CLI 命令 70
A.4. 管理 CLI 操作 72
A.5. 资源属性详情 75

第 1 章 管理 CLI 概述

管理命令行界面(CLI)是 JBoss EAP 的命令行管理工具。

使用管理 CLI 启动和停止服务器、部署和取消部署应用、配置系统设置，以及执行其他管理任务。操作可以在批处理模式下执行，允许以组形式运行多个任务。

许多常见的终端命令可用，如 **ls**、**cd** 和 **pwd**。管理 CLI 也支持 tab 自动完成功能。

第 2 章 管理 CLI 入门

管理 CLI 包含在 JBoss EAP 分发中。启动管理 CLI 后，您可以连接到正在运行的服务器实例或受管域，以执行管理操作。

2.1. 启动管理 CLI

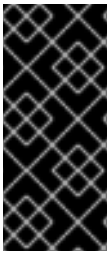
您可以通过运行随 JBoss EAP 提供的 **jboss-cli** 脚本来启动管理 CLI。

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

注意

对于 Windows Server，使用 ***EAP_HOME*\bin\jboss-cli.bat** 脚本启动管理 CLI。

有关使用 **--connect** 参数启动管理 CLI 和连接到服务器的详细信息，请参阅 连接到 服务器。



重要

jboss-cli 脚本将 **com.ibm.jsse2.overrideDefaultTLS** 属性设置为 **true**。如果您在使用 Elytron 配置的 SSL 时使用 IBM JDK 防止身份验证问题，则此设置非常重要。

如果您正在使用 IBM JDK 并使用其他方法启动 CLI 会话，请务必设置此属性；例如，以编程方式使用 ***EAP_HOME*/bin/client/jboss-cli-client.jar** 中的类。

有关所有可用 **jboss-cli** 脚本参数及其用途的完整列表，请使用 **--help** 参数，或参阅 管理 CLI 启动参数 部分。

2.2. 连接到服务器

您可以使用 **connect** 命令来连接正在运行的单机服务器或受管域。

```
connect
```

默认主机和端口配置是 **localhost:9990**。如果服务器正在侦听其他主机和端口，则需要将它们提供给 **connect** 命令。

```
connect 192.168.0.1:9991
```

您还可以启动管理 CLI，并使用 **--connect 参数（如有必要，使用 -- controller** 参数）连接服务器。

```
$ EAP_HOME/bin/jboss-cli.sh --connect --controller=192.168.0.1:9991
```

在 JBoss EAP 7.3 中，若要使用 **http-remoing** 协议进行连接，请使用：

```
connect http-remoting://192.168.0.1:9990
```

2.3. 获得帮助

管理 CLI 提供了多种方法，供您使用管理 CLI 获得帮助。

- 使用管理 CLI 查看有关 的一般帮助。

```
help
```

这为启动、导航和生成操作请求提供了详细帮助。

- 查看特定命令或操作的帮助信息：

```
help COMMAND_OR_OPERATION
```

这为特定命令或操作提供了用法、描述和参数。

例如：

  - 查看 **patch** 命令的帮助信息：

```
help patch
```

  - 查看 **patch** 命令 **应用** 操作的帮助：

```
help patch apply
```

  - 查看 Elytron **key-store** 资源 **add** 操作的帮助信息：

```
help /subsystem=elytron/key-store=?:add
```

- 查看当前上下文中可用命令列表：

```
help --commands
```

注意

需要连接到单机服务器或域控制器的命令不会显示在列表中，除非连接已经建立。

有关管理 CLI 命令列表，请参阅管理 CLI 命令部分。

2.4. 退出管理 CLI

您可以通过输入 **quit 命令退出** 管理 CLI。

```
quit
```

2.5. 以非互动模式运行

您可以在不启动管理 CLI 命令的情况下发出管理 CLI 命令，并与管理 CLI 交互。这可用于处理批量命令并从脚本执行命令。您可以 传递命令， 或者 传递包含命令到 **jboss-cli** 启动脚本的文件。

传递命令
您可以使用 **--command** 参数提供要执行的单个 CLI 命令。命令完成后，管理 CLI 将终止。

```
$ EAP_HOME/bin/jboss-cli.sh --connect --command="/interface=public:read-attribute(name=inet-address,resolve-expressions=true)"
```

提供的每个命令的输出会在执行时显示。

```
{
    "outcome" => "success",
    "result" => "127.0.0.1"
}
```

您还可以使用 **--commands** 参数提供要执行的 CLI 命令的逗号分隔列表。

传递命令文件

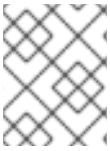
您可以使用 **--file** 参数传递要执行的 CLI 命令的文本文件，每个命令位于文件的单独行中。

```
$ EAP_HOME/bin/jboss-cli.sh --connect --file=/path/to/cli_commands.txt
```

文件中每个命令的输出会在执行时显示。

输出示例

```
{
    "outcome" => "success",
    "result" => "NORMAL"
}
helloworld.war
```



注意

您可以在 CLI 脚本中包含注释，以帮助理解和维护。注释以行首的井号(#)表示。在执行脚本时，JBoss EAP 会忽略您中包含的注释。

您可以使用 **--echo-command** 参数包含提示符和命令及输出。在通过将输出与执行的命令匹配来解决故障时，这非常有用。

```
$ EAP_HOME/bin/jboss-cli.sh --connect --file=/path/to/cli_commands.txt --echo-command
```

命令及其输出会在执行时显示。

使用命令选择的输出示例

```
[standalone@localhost:9990 /] :read-attribute(name=running-mode)
{
    "outcome" => "success",
    "result" => "NORMAL"
}
[standalone@localhost:9990 /] ls /deployment
helloworld.war
```

第 3 章 管理 CLI 导航

管理 CLI 中提供了许多常见的终端命令，例如 **ls** 用于 列出节点路径的内容，**cd** 可 更改节点路径，**pwd** 可 打印完整的节点路径。管理 CLI 也支持键盘快捷方式。

3.1. 更改当前路径

您可以使用 the **cd** 命令并提供所需的路径来更改到不同的节点路径。首次启动管理 CLI 时，它处于根级别(/)。

```
cd /subsystem=datasources
cd data-source=ExampleDS
```

3.2. 打印当前路径

您可以使用 **pwd** 命令打印当前节点的路径。首次启动管理 CLI 时，路径为 root 级别(/)。

```
cd /subsystem=undertow
cd server=default-server
pwd
```

以上示例使用 the **cd** 命令更改路径，然后将以下内容输出到控制台：

```
/subsystem=undertow/server=default-server
```

3.3. 列出内容

您可以使用 **ls** 命令列出特定节点路径的内容。如果路径以节点名称结尾，则也会列出该资源的属性。

以下示例浏览 **standard-sockets 套接字** 绑定组，然后列出其内容。

```
cd /socket-binding-group=standard-sockets
ls -l
```

```
ATTRIBUTE        VALUE                          TYPE
default-interface public                        STRING
name             standard-sockets               STRING
port-offset      ${jboss.socket.binding.port-offset:0} INT

CHILD                                MIN-OCCURS MAX-OCCURS
local-destination-outbound-socket-binding  n/a        n/a
remote-destination-outbound-socket-binding n/a        n/a
socket-binding                       n/a        n/a
```

通过指定 **ls** 命令的节点路径，可以从资源树层次结构中的任何位置获得相同的结果。

```
ls -l /socket-binding-group=standard-sockets
```

```
ATTRIBUTE        VALUE                          TYPE
default-interface public                        STRING
name             standard-sockets               STRING
```

```
port-offset       ${jboss.socket.binding.port-offset:0} INT

CHILD                                    MIN-OCCURS MAX-OCCURS
local-destination-outbound-socket-binding  n/a        n/a
remote-destination-outbound-socket-binding n/a        n/a
socket-binding                           n/a        n/a
```

您还可以使用 **--resolve-expressions** 参数将返回属性的表达式解析为服务器上对应的值。

```
ls -l /socket-binding-group=standard-sockets --resolve-expressions
```

```
ATTRIBUTE        VALUE            TYPE
default-interface public          STRING
name             standard-sockets STRING
port-offset      0                INT

CHILD                                    MIN-OCCURS MAX-OCCURS
local-destination-outbound-socket-binding  n/a        n/a
remote-destination-outbound-socket-binding n/a        n/a
socket-binding                           n/a        n/a
```

在本例中，port **-offset** 属性显示其解析值**0**，而不是表达式(**${jboss.socket.binding.port-offset:0}**)。

3.4. 查看多页输出

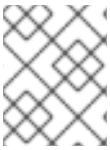
以交互模式运行管理 CLI 时，操作会导致多个输出页，命令处理器将在第一页面末尾暂停屏幕。这样，您可以一次页面浏览输出结果。在输出信息 **的末尾，输出的出现通过一行文本显示 --More(*NN*%)** 表示。

以下是一个管理 CLI 命令的示例，它提供了多个页面的输出：

```
/subsystem=undertow:read-resource(recursive=true)
{
    "outcome" => "success",
    "result" => {
      "default-security-domain" => "other",
      "default-server" => "default-server",
      "default-servlet-container" => "default",
      "default-virtual-host" => "default-host",
      "instance-id" => expression "${jboss.node.name}",
      "statistics-enabled" => false,
  Pre    "application-security-domain" => {"other" => {
           "enable-jacc" => false,
           "http-authentication-factory" => "application-http-authentication",
           "override-deployment-config" => false,
           "setting" => undefined
       }},
      "buffer-cache" => {"default" => {
           "buffer-size" => 1024,
           "buffers-per-region" => 1024,
--More(7%)--
```

浏览多页输出
当您遇到文本行表示输出更多时，您可以继续使用以下选项之一：

- 按 **Enter** 或向下箭头键通过输出一次一页。
- 按 Spacebar 或 **PgDn** 以跳到下一页输出。
- 按 **PgUp** 返回到输出的上一页。
- 按 **Home** 返回到输出的开头。
- 按 **End** 键跳到输出的最后一行。
- 键入 **q** 中断命令并退出。



注意

在 Windows 上，**PgUp**、**PgDn**、**home** 和 **End** 密钥以 Windows Server 2016 开头。其他操作系统没有问题。

搜索多页输出
您可以在多页输出中搜索文本。

1. 使用正斜杠(/)启动搜索。
2. 键入所需的文本并按 **Enter** 进行搜索。
   - 按 **n** 转到下一个匹配项。
   - 按 **N** 转到上一个匹配项。

您还可以使用上下箭头浏览搜索历史记录。

3.5. 使用键盘导航快捷方式

以交互模式运行管理 CLI 时，您可以使用键盘快捷方式快速编辑管理 CLI 命令。

注意

您还可以使用 Tab 键自动完成部分管理 CLI 命令或查看可用的选项。

您可以使用的键盘快捷方式因您正在运行的支持平台而异：

- Red Hat Enterprise Linux
- Windows Server
- Solaris

表 3.1. Red Hat Enterprise Linux 键盘导航快捷方式

Navigation	键盘快捷方式
保留一个单词	Alt+B 或 Ctrl+left箭头
对一个单词	Alt+F 或 Ctrl+right 箭头

Navigation	键盘快捷方式
行首	Ctrl+A 或 Home
行末尾	Ctrl+E 或结束
保留一个字符	Ctrl+B 或左箭头
右键一个字符	Ctrl+F 或左箭头

表 3.2. Windows Server Keyboard Nhortcuts

Navigation	键盘快捷方式
保留一个单词	Alt+B
对一个单词	Alt+F
行首	Ctrl+A 或 Home
行末尾	Ctrl+E 或结束
保留一个字符	Ctrl+B 或左箭头
右键一个字符	Ctrl+F 或左箭头

表 3.3. Solaris 键盘导航快捷方式

Navigation	键盘快捷方式
保留一个单词	Alt+B 或 Ctrl+left箭头
对一个单词	Alt+F 或 Ctrl+right 箭头
行首	Ctrl+A 或 Home
行末尾	Ctrl+E 或结束
保留一个字符	Ctrl+B 或左箭头
右键一个字符	Ctrl+F 或左箭头

第 4 章 创建和执行请求

JBoss EAP 配置呈现为可寻址资源的层次结构树，每种都提供自己的一组操作。管理 CLI 操作请求允许低级别与管理模式交互，并提供可控的方式来编辑服务器配置。

操作请求使用以下格式：

```
/NODE_TYPE=NODE_NAME:OPERATION_NAME(PARAMETER_NAME=PARAMETER_VALUE)
```

操作请求由三个部分组成：

地址

该地址指定要在其上执行操作的资源节点。*NODE_TYPE* 映射到元素名称，*NODE_NAME* 映射到配置 XML 中的该元素的 **name** 属性。资源树的每一级别都用斜杠(/)分隔。

操作名称

要在资源节点上执行的操作。它以冒号(**:**)前缀。

parameters

因操作而异的必选或可选参数集合。它们包含在括号（**()**）中。

构建操作请求

1. 确定地址
   您可以引用 XML 配置文件（**standalone.xml**、**domain.xml** 或 **host.xml**），以帮助确定所需的地址。您还可以使用 tab 自动完成功能来查看可用资源。

   以下是 root(/)级别上资源的几个常用地址。

   - **/deployment=*DEPLOYMENT_NAME*** - 部署配置.
   - **/socket-binding-group=*SOCKET_BINDING_GROUP_NAME*** - 套接字绑定配置.
   - **/interface=*INTERFACE_NAME*** - 接口配置.
   - **/subsystem=*SUBSYSTEM_NAME*** - 作为单机服务器运行时的子系统配置.
   - **/profile=*PROFILE_NAME*/subsystem=*SUBSYSTEM_NAME*** - 在受管域中运行时，适用于所选配置文件的子系统配置。
   - **/host=*HOST_NAME*** - 在受管域中运行时所选主机的服务器配置。

   以下地址用于 **ExampleDS** 数据源。

   ```
   /subsystem=datasources/data-source=ExampleDS
   ```

2. 确定操作
   每种资源类型的资源节点的可用操作都有所不同。您可以使用资源地址上的 **:read-operation-names** 操作来查看可用的操作。您还可以使用 tab 自动完成功能。

   使用 **:read-operation-description** 操作来获取资源特定操作的信息。

   以下操作（包含适当的参数后）将为 **ExampleDS** 数据源设置属性值：

   ```
   /subsystem=datasources/data-source=ExampleDS:write-attribute
   ```

3. 确定参数
   每个操作都有自己的一组可用参数。如果您在没有必要参数的情况下尝试执行操作，您会收到一条错误消息，指出参数无法 为**空**。

   使用逗号(,)分隔多个参数。如果某个操作没有任何参数，则括号是可选的。

   对资源使用 **:read-operation-description** 操作，在操作名称中传递，以确定该操作所需的参数。您还可以使用 Tab 补全来列出可用的参数。

   以下操作通过将 **enabled** 属性设置为 **false** 来禁用 **ExampleDS** 数据源。

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

输入之后，管理界面将在服务器配置上执行操作请求。根据操作请求，您将接收到包含结果以及操作结果或响应的输出。

禁用 **ExampleDS** 数据源的以下响应显示操作成功，需要重新加载服务器才能生效。

```
{
    "outcome" => "success",
    "response-headers" => {
        "operation-requires-reload" => true,
        "process-state" => "reload-required"
    }
}
```

您可以使用 **read-attribute** 操作来读取 **ExampleDS** 数据源的 **enabled** 属性的值。

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

以下响应显示操作成功，并且 **enabled** 值为 **false**。

```
{
    "outcome" => "success",
    "result" => false,
}
```

4.1. 显示资源值

您可以使用 **read-resource** 操作来查看资源的属性值。

```
:read-resource
```

您可以指定参数，以递归方式提供有关子资源的完整信息。您还可以指定参数，以包括运行时属性、解析表达式和包含别名。使用 **read-operation-description(name=read-resource)** 查看所有可用参数的 **read-resource** 的描述。

以下示例读取部署的属性：它包括部署名称、是否启用它的详细信息，以及它最后一次启用的时间。

```
/deployment=DEPLOYMENT_NAME:read-resource
{
    "outcome" => "success",
```

```
    "result" => {
        ...
        "enabled" => true,
        "enabled-time" => 1453929902598L,
        "enabled-timestamp" => "2016-01-27 16:25:02,598 EST",
        "name" => "DEPLOYMENT_NAME",
        "owner" => undefined,
        "persistent" => true,
        "runtime-name" => "DEPLOYMENT_NAME",
        "subdeployment" => undefined,
        "subsystem" => {
            "undertow" => undefined,
            "logging" => undefined
        }
    }
}
```

包括运行时属性

include-runtime 参数可用于检索运行时属性。

以下示例读取部署的属性：除了持久属性外，它还包含运行时属性，如部署状态和上次禁用的时间。

```
/deployment=DEPLOYMENT_NAME:read-resource(include-runtime=true)
{
    "outcome" => "success",
    "result" => {
        ...
        "disabled-time" => undefined,
        "disabled-timestamp" => undefined,
        "enabled" => true,
        "enabled-time" => 1453929902598L,
        "enabled-timestamp" => "2016-01-27 16:25:02,598 EST",
        "name" => "DEPLOYMENT_NAME",
        "owner" => undefined,
        "persistent" => true,
        "runtime-name" => "DEPLOYMENT_NAME",
        "status" => "OK",
        "subdeployment" => undefined,
        "subsystem" => {
            "undertow" => undefined,
            "logging" => undefined
        }
    }
}
```

您还可以在传递布尔值参数时使用 *not* 运算符(**!**)。例如：

- **:read-resource(include-runtime=false)** 可以输入为 **：read-resource(!include-runtime)**
- **:read-resource(include-runtime=true)** 可以输入为 **：read-resource(include-runtime)**

递归读取子资源

可以使用 **递归** 参数从子资源中以递归方式检索属性。

以下示例读取部署的属性：除了资源自己的属性外，它还以递归方式返回其子资源的属性，如 **undertow** 子系统配置。

```
/deployment=DEPLOYMENT_NAME:read-resource(recursive=true)
{
    "outcome" => "success",
    "result" => {
        ...
        "enabled" => true,
        "enabled-time" => 1453929902598L,
        "enabled-timestamp" => "2016-01-27 16:25:02,598 EST",
        "name" => "DEPLOYMENT_NAME",
        "owner" => undefined,
        "persistent" => true,
        "runtime-name" => "DEPLOYMENT_NAME",
        "subdeployment" => undefined,
        "subsystem" => {
            "undertow" => {
                "context-root" => "/test",
                "server" => "default-server",
                "virtual-host" => "default-host",
                "servlet" => undefined,
                "websocket" => undefined
            },
            "logging" => {"configuration" => undefined}
        }
    }
}
```

排除默认值

include-defaults 参数可用于在读取资源的属性时显示或隐藏默认值。默认情况下是 **true**，这意味着在使用 **read-resource** 操作时会显示默认值。

以下示例对 **undertow** 子系统使用了 **read-resource** 操作：

```
/subsystem=undertow:read-resource
{
    "outcome" => "success",
    "result" => {
        "default-security-domain" => "other",
        "default-server" => "default-server",
        "default-servlet-container" => "default",
        "default-virtual-host" => "default-host",
        "instance-id" => expression "${jboss.node.name}",
        "statistics-enabled" => false,
        "buffer-cache" => {"default" => undefined},
        "configuration" => {
            "filter" => undefined,
            "handler" => undefined
        },
        "server" => {"default-server" => undefined},
        "servlet-container" => {"default" => undefined}
    }
}
```

以下示例也对 **undertow** 子系统使用 **read-resource** 操作，但将 **include-defaults** 参数设置为 **false** ：现在，一些属性 （如启用了 **statistics** 和 **default-server** ）显示 **未定义** 的值，而不是默认值。

```
/subsystem=undertow:read-resource(include-defaults=false)
{
    "outcome" => "success",
    "result" => {
        "default-security-domain" => undefined,
        "default-server" => undefined,
        "default-servlet-container" => undefined,
        "default-virtual-host" => undefined,
        "instance-id" => undefined,
        "statistics-enabled" => undefined,
        "buffer-cache" => {"default" => undefined},
        "configuration" => {
            "filter" => undefined,
            "handler" => undefined
        },
        "server" => {"default-server" => undefined},
        "servlet-container" => {"default" => undefined}
    }
}
```

resolve Expressions

resolve-expressions 参数可用于将返回属性的表达式解析为服务器上对应的值。

带有表达式的属性，作为其值使用格式 **${*PARAMETER: DEFAULT_VALUE*}**。如需更多信息，请参阅 *《配置指南》* 中的属性替换。

以下示例读取部署的属性：**instance-id** 属性显示其解析值(**test-name**)，而不显示表达式(**${jboss.node.name}**)。

```
/subsystem=undertow:read-resource(resolve-expressions=true)
{
    "outcome" => "success",
    "result" => {
        "default-security-domain" => "other",
        "default-server" => "default-server",
        "default-servlet-container" => "default",
        "default-virtual-host" => "default-host",
        "instance-id" => "test-name",
        "statistics-enabled" => false,
        "buffer-cache" => {"default" => undefined},
        "configuration" => {
            "filter" => undefined,
            "handler" => undefined
        },
        "server" => {"default-server" => undefined},
        "servlet-container" => {"default" => undefined}
    }
}
```

4.2. 显示资源描述

您可以使用 **read-resource-description** 操作来描述资源及其属性。

```
:read-resource-description
```

您可以指定参数，以递归方式提供有关子资源的完整描述。您还可以指定参数，以包含资源操作和通知的详细信息。使用 **read-operation-description(name=read-resource-description)** 查看所有可用参数的描述 。

以下示例显示了缓冲区缓存的属性详细信息：

```
/subsystem=undertow/buffer-cache=default:read-resource-description
{
    "outcome" => "success",
    "result" => {
        "description" => "The buffer cache used to cache static content",
        "attributes" => {
            "buffer-size" => {
                "type" => INT,
                "description" => "The size of an individual buffer",
                "expressions-allowed" => true,
                "nillable" => true,
                "default" => 1024,
                "min" => 0L,
                "max" => 2147483647L,
                "access-type" => "read-write",
                "storage" => "configuration",
                "restart-required" => "resource-services"
            },
            "buffers-per-region" => {
                "type" => INT,
                "description" => "The numbers of buffers in a region",
                "expressions-allowed" => true,
                "nillable" => true,
                "default" => 1024,
                "min" => 0L,
                "max" => 2147483647L,
                "access-type" => "read-write",
                "storage" => "configuration",
                "restart-required" => "resource-services"
            },
            "max-regions" => {
                "type" => INT,
                "description" => "The maximum number of regions",
                "expressions-allowed" => true,
                "nillable" => true,
                "default" => 10,
                "min" => 0L,
                "max" => 2147483647L,
                "access-type" => "read-write",
                "storage" => "configuration",
                "restart-required" => "resource-services"
            }
        },
        "operations" => undefined,
        "notifications" => undefined,
        "children" => {}
    }
}
```

请参阅 Resource Attribute Details，以了解更多有关属性返回的字段的信息。

4.3. 显示属性值

您可以使用 **read-attribute** 操作来查看单个属性的当前值。

```
:read-attribute(name=ATTRIBUTE_NAME)
```

以下示例通过读取 **level** 属性来显示根日志记录器的日志级别：

```
/subsystem=logging/root-logger=ROOT:read-attribute(name=level)
{
    "outcome" => "success",
    "result" => "INFO"
}
```

使用 **read-attribute** 操作的一个优点是能够公开属性的当前运行时值。

```
/interface=public:read-attribute(name=resolved-address)
{
    "outcome" => "success",
    "result" => "127.0.0.1"
}
```

resolve **-address** 属性是 runtime 属性。在公共接口上使用 **read-resource** 操作时不会显示此属性，除非您通过了 **include-runtime** 参数。即使这样，它也会显示在资源的其余属性中。

您还可以使用 **include-defaults 和 resolve -expressions** 参数。有关这些参数的详情，请参阅显示资源值。

4.4. 更新属性

您可以使用 **write-attribute** 操作来更新资源的属性值。

```
:write-attribute(name=ATTRIBUTE_NAME, value=ATTRIBUTE_VALUE)
```

以下示例通过将 **scan-enabled** 属性设置为 **false** 来禁用部署扫描程序：

```
/subsystem=deployment-scanner/scanner=default:write-attribute(name=scan-enabled,value=false)
{"outcome" => "success"}
```

操作请求的响应显示它已成功。您还可以使用 **read-attribute** 操作来读取 **scan-enabled** 属性来确认结果，现在显示为 **false**。

```
/subsystem=deployment-scanner/scanner=default:read-attribute(name=scan-enabled)
{
    "outcome" => "success",
    "result" => false
}
```

4.5. 取消定义属性

您可以将属性的值设置为 **未定义**。如果此属性具有默认值，则将使用该值。

以下示例取消定义根日志记录器的 **level** 属性：

```
/subsystem=logging/root-logger=ROOT:undefine-attribute(name=level)
```

level 属性的默认值为 **ALL**。您可以看到在执行 **read-resource** 操作时会使用此默认值。

```
/subsystem=logging/root-logger=ROOT:read-resource
{
    "outcome" => "success",
    "result" => {
        "filter" => undefined,
        "filter-spec" => undefined,
        "handlers" => [
            "CONSOLE",
            "FILE"
        ],
        "level" => "ALL"
    }
}
```

若要查看资源但不读取默认值，您必须使用 **include-defaults** 参数设置为 **false**。现在，您可以看到 **level** 的值 **尚未定义**。

```
/subsystem=logging/root-logger=ROOT:read-resource(include-defaults=false)
{
    "outcome" => "success",
    "result" => {
        "filter" => undefined,
        "filter-spec" => undefined,
        "handlers" => [
            "CONSOLE",
            "FILE"
        ],
        "level" => undefined
    }
}
```

4.6. 显示操作名称

您可以使用 **read-operation-names** 列出给定资源的可用操作。

```
:read-operation-names
```

以下示例列出了可用于对部署执行的操作。

```
/deployment=DEPLOYMENT_NAME:read-operation-names
{
    "outcome" => "success",
    "result" => [
        "add",
        "deploy",
        "list-add",
        "list-clear",
```

```
        "list-get",
        "list-remove",
        "map-clear",
        "map-get",
        "map-put",
        "map-remove",
        "query",
        "read-attribute",
        "read-attribute-group",
        "read-attribute-group-names",
        "read-children-names",
        "read-children-resources",
        "read-children-types",
        "read-operation-description",
        "read-operation-names",
        "read-resource",
        "read-resource-description",
        "redeploy",
        "remove",
        "undefine-attribute",
        "undeploy",
        "whoami",
        "write-attribute"
    ]
}
```

使用 **read-operation-description** 操作 来显示操作描述。

4.7. 显示操作描述

您可以使用 **read-operation-description** 操作来显示资源的特定操作的描述。这也包括参数描述和所需的参数。

```
:read-operation-description(name=OPERATION_NAME)
```

以下示例提供了系统属性的 **add** 操作的描述和参数信息。

```
/system-property=SYSTEM_PROPERTY:read-operation-description(name=add)
{
    "outcome" => "success",
    "result" => {
        "operation-name" => "add",
        "description" => "Adds a system property or updates an existing one.",
        "request-properties" => {"value" => {
            "type" => STRING,
            "description" => "The value of the system property.",
            "expressions-allowed" => true,
            "required" => false,
            "nillable" => true,
            "min-length" => 0L,
            "max-length" => 2147483647L
        }},
        "reply-properties" => {},
        "read-only" => false,
```

```
        "runtime-only" => false
    }
}
```

4.8. 使用特殊字符添加值

有时在创建管理 CLI 请求时，您可能需要添加包含特殊字符的值。必须以特定的方式输入某些特殊字符，如管理 CLI 请求语法中使用的字符。

在很多情况下，以双引号(")括起这个值就足够了。如果您不确定您的特殊字符是否已正确接受，请务必在添加值后读取属性或资源，以验证是否已正确保存。

有关如何处理以下特殊字符的详情，请查看以下章节。

- 空白
- 引号
- 逗号
- 括号
- 大括号
- 方括号
- diacritic Marks

空白
默认情况下，空格与通过管理 CLI 添加的值分离。您可以在值中包含空格，方法是将值用双引号(")或大括号({})括起，或使用反斜杠(\)来转义。

```
/system-property=test1:add(value="Hello World")
/system-property=test2:add(value={Hello World})
/system-property=test3:add(value=Hello\ World)
```

这会将值设置为 **Hello World**。

引号
您可以在值中使用单引号(')，方法是将值用双引号(")括起，或者使用反斜杠(\)进行转义。以下示例将系统属性的值设置为 **server**。

```
/system-property=test1:add(value="server's")
/system-property=test2:add(value=server\'s)
```

您可以在值中使用双引号(")，方法是使用反斜杠(\)对其进行转义。根据值中引号的位置，您可能还需要将值用双引号(")括起。以下示例将系统属性的值设置为 **"quote"**。

```
/system-property=test1:add(value="\"quote\"")
```

逗号
您可以在值中使用逗号(,)，用双引号(")括起该值。

```
/system-property=test:add(value="Last,First")
```

这会将值设置为 **Last,First**。

括号
您可以将括号（ **()** ）**包含在**值中，方法是将值用双引号(**""**)或花括号(**{}**)括起，或使用反斜杠(\)来转义圆括号(\)。

```
/system-property=test1:add(value="one(1)")
/system-property=test2:add(value={one(1)})
/system-property=test3:add(value=one\(1\))
```

这会将该值设置为 **1(1)**。

大括号
您可以通过用双引号(**""**)括起值，将大括号(**{}**)包含在值中。

```
/system-property=test:add(value="{braces}")
```

这会将值设置为 **{braces}**。

方括号
您可以将方括号(**[]**)包含在值中，方法是用双引号(**""**)括起该值。

```
/system-property=test:add(value="[brackets]")
```

这会将值设置为 **[brackets]**。

diacritic Marks
在使用管理 CLI 添加值时，可以使用**dia** critic 标记（如 **ñ** 、 sudel 或 SAS）。

```
/system-property=test1:add(value=Año)
```

但是，不要将值包括在双引号(**""**)**中**。这可导致 diacritic 标记替换为问号(**?**)。如果值有需要维护的空格，则需要将值括在大括号(**{}**)中，或使用反斜杠(\)转义空格。

```
/system-property=test2:add(value={Dos años})
/system-property=test3:add(value=Dos\ años)
```

这会将该值设置为 **Dos años**。

4.9. 指定操作标头

您可以指定操作标头来控制操作执行方式的某些方面。可用的操作标头如下：

allow-resource-service-restart

是否重启需要重启的运行时服务以便操作的更改生效。默认值为 **false**。

警告

使用 **allow-resource-service-restart=true** 标头可能会破坏最终用户请求处理，直到重启所需的服务为止。

blocking-timeout

在回滚操作前，操作在其完成过程的任意点上应停止的最大时间（以秒为单位）。默认值为 **300** 秒。

角色

在做出访问控制决策时应使用的 RBAC 角色列表，而不是来自通常与调用操作的用户关联的角色。请注意，这只能用于减少调用者的权限，而不能提高权限。

rollback-on-runtime-failure

在对运行时服务应用更改时，是否应该恢复持久性配置更改。默认值为 **true**。

rollout

受管域部署的推出计划。如需更多信息，请参阅 JBoss EAP *配置指南中的使用* Rollout Plans 部分。

示例：使用操作标头部署应用

```
deployment deploy-file /path/to/DEPLOYMENT.war --headers={allow-resource-service-restart=true}
```

示例：使用操作标头删除资源

```
/subsystem=infinispan/cache-container=test:remove() {allow-resource-service-restart=true}
```

使用分号(;)来分隔多个操作标头。

4.10. 使用 IF-ELSE CONTROL FLOW

管理 CLI 支持 **if-其他** 控制流，它允许您根据条件选择要执行的命令和操作集合。**if** 条件是一个布尔值表达式，评估在 **关键字** 后指定的管理命令或操作的响应。

注意

如果不支持 **-其他语句，** 则使用嵌套。

表达式可以包含以下任一项目：

- 分组和优先级表达式的括号
- 条件运算符
  - 和 （**及&**）
  - 或(||)
- 比较运算符
  - 等于(==)

- 不等于(**!=**)
- 大于(**>**)
- 大于或等于(**>=**)
- 小于(**<**)
- 小于或等于(**<=**)
- 匹配正则表达式(**~=**)

重要

match 正则表达式(**~=**)运算符仅作为技术预览提供。技术预览功能不包括在红帽生产服务级别协议（SLA）中，且其功能可能并不完善。因此，红帽不建议在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

如需有关技术预览功能支持范围的信息，请参阅红帽客户门户网站中的技术预览功能支持范围。

以下示例使用 match 正则表达式(**~=**)运算符来检查 **features** 系统属性的值是否包含 **jgroups**。

```
if (result ~= ".*jgroups.*") of /:resolve-expression(expression=${features})
  echo Configuring JGroups
end-if
```

以下示例尝试读取系统属性 **test**。如果 **结果** **不成功** （这意味着该属性不存在），则系统属性将被添加并设为 **true**。

```
if (outcome != success) of /system-property=test:read-resource
    /system-property=test:add(value=true)
end-if
```

以上条件使用 **结果**，在执行 **关键字** 后 CLI 命令返回，如下所示：

```
/system-property=test:read-resource
{
    "outcome" => "failed",
    "failure-description" => "JBAS014807: Management resource '[(\"system-property\" => \"test\")]' not
found",
    "rolled-back" => true
}
```

以下示例通过检查服务器进程的启动类型（STANDAL**ONE 或 DOMAIN**）发出适当的管理 CLI 命令，以启用 **ExampleDS** 数据源。

```
if (result == STANDALONE) of /:read-attribute(name=launch-type)
    /subsystem=datasources/data-source=ExampleDS:write-attribute(name=enabled, value=true)
else
    /profile=full/subsystem=datasources/data-source=ExampleDS:write-attribute(name=enabled,
value=true)
end-if
```

使用 的管理 CLI 命令 **如果**可以在文件中指定 -**其他** 控制流，则在文件中单独一行执行每个命令。然后，您可以使用 **--file** 参数以非交互方式将 文件传递到 **jboss-cli** 脚本。

```
$ EAP_HOME/bin/jboss-cli.sh --connect --file=CLI_FILE
```

4.11. 使用 TRY-CATCH-FINALLY CONTROL FLOW

管理 CLI 提供了一个简单的 尝试**概括性** 控制流程。它由与 **try**、**catch** 和 **最终** 块对应的三组操作和命令组成。**捕获** 块 **和最终** 块是可选的，但其中至少应存在，并且只能指定一个捕获块。

控制流程从执行 尝试 批处理开始。如果 尝试 批处理成功完成，则将跳过 **捕获** 批处理并执行 **最终** 批处理。如果 尝试 批量失败，例如 **java.io.IOException**，**try- catch -finally** 控制流将立即终止，如果可用，则会执行 catch 批处理。**最后** 的批处理始终在控制流程的末尾执行，尝试 和 **捕获** 批处理是成功还是无法执行。

有四个命令定义了 try-catch-finally 控制流：

- 尝试 命令可以启动 尝试 的批处理。尝试 批处理将继续，直到遇到 **catch** 或 **last** 命令之一。
- **catch** 命令标记 尝试 批处理的末尾。然后，尝试 批处理退回，并且 **捕获** 批次启动。
- **最后**，命令标记 **捕获** 批处理或 尝试 批处理的末尾，并启动 **最终** 批处理。
- **end-try** 是指结束 **捕获或 最终 批处理并运行 try-catch** -finally 控制流程的命令。

以下示例创建或重新创建数据源并启用它：

```
try
/subsystem=datasources/data-source=myds:add(connection-url=CONNECTION_URL,jndi-
name=java:/myds,driver-name=h2)

catch
/subsystem=datasources/data-source=myds:remove
/subsystem=datasources/data-source=myds:add(connection-url=CONNECTION_URL,jndi-
name=java:/myds,driver-name=h2)

finally
/subsystem=datasources/data-source=myds:enable
end-try
```

4.12. 使用 FOR-DONE CONTROL FLOW

管理 CLI 支持 **for-done** 控制流，允许您迭代从操作返回的集合，并对集合中的每个项目执行命令。您可以在交互或非交互模式下将 **for-done** 语句与管理 CLI 搭配使用。**for-done** 语句使用以下语法：

```
for VARIABLE_NAME in OPERATION
  COMMANDS_TO_EXECUTE
done
```

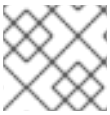
- 可以使用语法 **$ *VARIABLE_NAME* 在 *COMMANDS_TO_EXECUTE* 中使用*VARIABLE_NAME***。
- OPERATION ***必须*** 返回一个集合。

- ***COMMANDS_TO_EXECUTE*** 是要执行的命令列表，每个命令位于单独的行上。

以下示例迭代所有部署，并显示每项部署是启用或禁用的。

```
for deploymentName in :read-children-names(child-type=deployment)
  if (result == true) of /deployment=$deploymentName:read-attribute(name=enabled)
    echo $deploymentName is enabled.
  else
    echo $deploymentName is disabled.
  end-if
done
```

如果您想要丢弃当前 **的块** 而不执行命令，请输入 **done --discard**。



注意

不支持在批处理模式中使用 **for-done** 语句，或者将它们嵌套在其他 **for-done** 语句中。

4.13. 查询资源

JBoss EAP 管理 CLI 提供 **查询** 操作以查询资源。您可以使用 **:read-resource** 操作来读取资源的所有属性。如果只列出所选属性，您可以使用 **:query** 操作。

例如，要查看 **名称和 已启用** 的属性列表，请使用以下命令：

```
/deployment=jboss-modules.jar:query(select=["name","enabled"])
```

以下响应显示操作成功：**name** 和 **enabled** 属性列在 **jboss-modules.jar** 部署中。

```
{
    "outcome" => "success",
    "result" => {
        "name" => "jboss-modules.jar",
        "enabled" => true
    }
}
```

您还可以使用通配符来跨多个资源查询，例如，列出所有部署 **的名称和 启用** 的属性：

```
/deployment=*:query(select=["name","enabled"])
```

以下响应显示操作成功：列出了所有部署的 **name** 和 **enabled** 属性。

```
{
    "outcome" => "success",
    "result" => [
        {
            "address" => [("deployment" => "helloworld.war")],
            "outcome" => "success",
            "result" => {
                "name" => "helloworld.war",
                "enabled" => true
            }
```

```
        },
        {
            "address" => [("deployment" => "kitchensink.war")],
            "outcome" => "success",
            "result" => {
                "name" => "kitchensink.war",
                "enabled" => true
            }
        },
        {
            "address" => [("deployment" => "xyz.jar")],
            "outcome" => "success",
            "result" => {
                "name" => "xyz.jar",
                "enabled" => false
            }
        }
    ]
}
```

：**query** 操作也会过滤相关的对象。例如，若要查看 **enabled** 为 **true** 的部署的 **name** 和 **enabled** 属性值：

```
/deployment=*:query(select=["name","enabled"],where=["enabled","true"])
```

以下响应显示操作成功：列出了已启用为 **true** 的部署的 **name** 和 **enabled** 属性值。

```
{
    "outcome" => "success",
    "result" => [
        {
            "address" => [("deployment" => "helloworld.war")],
            "outcome" => "success",
            "result" => {
                "name" => "helloworld.war",
                "enabled" => true
            }
        },
        {
            "address" => [("deployment" => "kitchensink.war")],
            "outcome" => "success",
            "result" => {
                "name" => "kitchensink.war",
                "enabled" => true
            }
        }
    ]
}
```

4.14. 重定向输出

您可以将输出重定向到文件或 另一个命令，而不是从管理 CLI 操作打印输出到终端。

将输出重定向到文件

使用 > operator 将管理 CLI 操作的输出重定向到文件系统中的文件。

示例： 将 read-resource 输出写入一个文件

```
:read-resource > myfile.txt
```

使用 >> operator 重定向管理 CLI 操作的输出并将其附加到文件系统中的文件中。

示例： 将读取资源 输出附加到文件

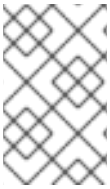
```
:read-resource >> myfile.txt
```

将输出重定向到命令

使用 | 运算符将管理 CLI 操作的输出重定向到 grep 命令，以搜索正则表达式匹配的输出。目前，gr ep 是 | 运算符唯一支持的命令。

示例：从 server.log 文件搜索输出

```
/subsystem=logging/log-file=server.log:read-log-file | grep Deployed
        "2018-03-06 09:48:02,389 INFO  [org.jboss.as.server] (management-handler-thread - 5)
WFLYSRV0010: Deployed \"jboss-helloworld.war\" (runtime-name : \"jboss-helloworld.war\")",
```



注意

不支持在同一命令中多次使用 grep 命令。

第 5 章 通过受管域使用管理 CLI

您可以使用管理 CLI 来配置和管理单机服务器和受管域。JBoss EAP 文档通常会显示用于单机服务器配置的管理 CLI 命令示例。如果您正在运行受管域，则通常需要调整 命令。以下小节介绍了如何为受管域配置更改单机服务器管理 CLI 命令。

为子系统配置指定 Profile

单机服务器子系统配置的管理 CLI 命令以 /subsystem=*SUBSYSTEM_NAME* 开头。对于受管域子系统配置，您必须通过使用 /profile=*PROFILE_NAME* /subsystem=*SUBSYSTEM_NAME* 启动命令来指定要配置的子系统。

示例：阅读日志记录子系统配置（单机服务器）

```
/subsystem=logging:read-resource
```

本例演示了如何读取单机服务器的 logging 子系统配置。

示例：阅读日志记录子系统配置（管理域）

```
/profile=default/subsystem=logging:read-resource
```

本例演示了如何读取受管域中 default 配置文件的 logging 子系统配置。

为核心管理和运行时命令指定主机

受管域的某些核心管理和运行时命令要求您通过使用 /host=*HOST_NAME* 启动命令来指定该命令应用到的主机。

示例：启用审计日志记录（单机服务器）

-

```
/core-service=management/access=audit/logger=audit-log:write-attribute(name=enabled,value=true)
```

本例演示了如何为单机服务器启用审计日志记录。

示例：启用审计记录（管理域）

```
/host=master/core-service=management/access=audit/logger=audit-log:write-
attribute(name=enabled,value=true)
```

本例演示了如何在受管域中为 master 主机启用审计日志记录。

注意

某些命令要求主机作为参数，例如 reload --host=*HOST_NAME*。如果您没有为这些命令指定主机，则会出现错误消息通知您需要 --host 参数。

为核心管理和运行时命令指定服务器

受管域的某些核心管理和运行时命令要求您通过使用 /host=*HOST_NAME* /server=*SERVER_NAME* 启动命令来指定适用于该主机和服务器的主机和服务器。

示例：显示部署（单机服务器）的运行时指标.

```
/deployment=test-application.war/subsystem=undertow:read-attribute(name=active-sessions)
```

本例演示了如何显示单机服务器部署的运行时指标。

示例：显示部署的运行时指标（管理域）

```
/host=master/server=server-one/deployment=test-application.war/subsystem=undertow:read-
attribute(name=active-sessions)
```

本例演示了如何显示部署到 master 主机上的 server-one 服务器的受管域部署的运行时指标。

第 6 章 配置管理 CLI

可以在其配置文件中自定义管理 CLI 的某些方面 jboss-cli.xml。此文件必须位于 *EAP_HOME*/bin 目录中，或者位于通过 jboss.cli.config 系统属性指定的自定义目录中。

以下元素可以在 jboss-cli.xml 文件中配置 ：

default-protocol

当控制器地址没有提供时要使用的默认协议。默认值为 remote+http。如果使用端口 9990 并且未指定协议，则该协议将默认自动进行远程移动，除非 将 use -legacy-override 属性设置为 false。

default-controller

如果不带任何参数执行 connect 命令，则配置要连接的控制器。如果管理 CLI 使用参数 --controller= 或 controller= 启动，则参数中指定的值会覆盖来自配置的 default-controller 定义。

- 协议 - 控制器的协议名称。如果未提供，则将使用 default-protocol 的值。
- Host - 控制器的主机名。默认值为 localhost。
- port - 用于连接到控制器的端口号。默认值为 9990。

controllers

您可以在 jboss-cli.xml 文件中定义连接控制器别名。例如：

```
<!-- The default controller to connect to when 'connect' command is executed w/o
arguments -->
<default-controller>
    <host>localhost</host>
    <port>9990</port>
</default-controller>
<!-- CLI connection controller aliases -->
<controllers>
    <controller name="ServerOne">
        <protocol>remoting</protocol>
        <host>192.168.3.45</host>
        <port>9990</port>
    </controller>
    <controller name="ServerTwo">
        <protocol>http-remoting</protocol>
```

```
        <host>192.168.3.46</host>
    </controller>
</controllers>
```

controller 元素的 name 属性应当用作 --controller= 参数的值。例如： --controller=ServerTwo。

validate-operation-requests

在将请求发送到控制器以执行之前，是否验证操作请求的参数列表。默认值为 true。

history

CLI 命令历史记录日志的配置。

- 启用 - 是否启用 历史记录。默认值为 true。
- file-name - 保存历史记录的文件名。默认值为 .jboss-cli-history。
- file-dir - 保存历史记录的目录。默认为用户的主目录。
- max-size - 历史记录文件中存储的命令的最大数量。默认值为 500。

resolve-parameter-values

在向控制器发送操作请求前，是否解析作为命令参数（或 operation 参数）指定的系统属性。默认值为 false。

connection-timeout

允许建立与控制器连接的时间（毫秒为单位）。默认值为 5000。

ssl

用于 SSL 的密钥存储和信任存储的配置。

警告

红帽建议显式禁用 SSLv2、SSLv3 和 TLSv1.0，以便在所有受影响的软件包中明确禁用 TLSv1.1 或 TLSv1.2。

- Vault - 库配置.如果未指定 代码 或 模块，则将使用默认的实施。如果指定了 代码，但不指定 模块，它将在 Picketbox 模块中查找指定的类。如果指定了模块和 代码，它将在" 模块 "指定的模块中查找由代码指定的类。
- key-store - 密钥存储.
- key-store-password - 密钥存储密码。
- alias - 别名.
- key-password - 密钥密码.
- trust-store - truststore。
- trust-store-password - truststore 密码。
- modify-trust-store - 如果设置为 true，则 CLI 将在收到未经识别的证书时提示用户，并允许它们存储在信任存储中。默认值为 true。

静默

是否将信息和错误消息写入终端。默认值为 false。

access-control

是否应根据授予用户的权限为当前用户筛选与管理相关的命令和属性。例如，如果为 true，则 Tab 补全将隐藏不允许该用户访问的命令和属性。默认值为 true。

echo-command

是否在输出中包含提示和命令，以非交互模式执行的命令。默认值为 false。

command-timeout

等待命令完成的最长时间（以秒为单位）。值 0 表示没有超时。默认情况下没有超时。

output-json

是否以纯 JSON 格式显示操作响应。默认情况下，操作响应以 DMR 格式显示。

color-output

是否根据日志消息输出类型以颜色打印 CLI 日志输出：可用的颜色为 black、蓝色、cyan、绿色、magenta、红色、白色 和 黄色。

- 启用 - 是否启用颜色输出。默认值为 true。
- error-color - 默认设置为 红色.
- warn-color - 默认至 黄色.
- success-color - 默认到 default，这是终端的默认前台颜色。
- required-color - 默认为 magenta。
- workflow-color - 默认为 绿色.
- prompt-color - 默认到 蓝色.

output-paging

显示输出页面后，管理 CLI 是否应该暂停，允许您浏览和搜索输出。如果此选项设为 false，则 立即打印整个输出。默认值为 true。

6.1. 属性替换

JBoss EAP 支持在管理 CLI 中使用预设元素和属性表达式。这些表达式将在执行 命令期间解析为其定义的值。

您可以替换以下属性的表达式：

- 操作地址部分（如节点类型或名称）
- 操作名称
- 操作参数名称
- 标头名称和值
- 命令名称
- 命令参数名称

默认情况下，管理 CLI 对每行执行属性替换，但参数或参数值除外。参数和参数值在服务器运行时解析。如果您需要在管理 CLI 中发生参数或参数值的属性替换，并将其解析的值发送到服务器，请完成以下步骤。

1. 编辑管理 CLI 配置文件： *EAP_HOME*/bin/jboss-cli.xml。
2. 将 resolve-parameter-values 参数设为 true （默认值为 false）。

```
<resolve-parameter-values>true</resolve-parameter-values>
```

此元素仅影响操作请求参数值和命令参数值。它不会影响其余命令行。这意味着命令行中存在的系统属性将在行解析期间解析，而不论 resolve-parameter-values 元素的值是什么，除非它是参数/参数值。

管理 CLI 命令中使用的系统属性值必须已经定义，以便能被解析。启动管理 CLI 实例时，您必须通过属性文件(--properties=*/path/to/file.properties*)或属性值对（-D*键=值*）。属性文件使用标准 *KEY=VALUE* 语法。

属性键在您的管理 CLI 命令中使用 ${*MY_VAR*} 语法表示，例如：

```
/host=${hostname}/server-config=${servername}:add(group=main-server-group)
```

有关其他 jboss-cli.xml 配置选项，请参阅 配置管理 CLI。

6.2. 创建别名

您可以使用 alias 命令在 CLI 会话期间为 CLI 命令 和操作定义别名。

以下示例创建一个名为 read_undertow 的新 CLI 命令别名，以使用 alias 命令读取 undertow 子系统中的资源：

```
alias read_undertow='/subsystem=undertow:read-resource'
```

注意

别名只能包含字母数字字符和下划线。

要测试 read_undertow 别名的创建，请在管理 CLI 中输入别名名称：

```
read_undertow
```

结果将是：

```
{
    "outcome" => "success",
    "result" => {
        "default-security-domain" => "other",
        "default-server" => "default-server",
        "default-servlet-container" => "default",
        "default-virtual-host" => "default-host",
        "instance-id" => expression "${jboss.node.name}",
```

```
        "statistics-enabled" => false,
        "buffer-cache" => {"default" => undefined},
        "configuration" => {
            "filter" => undefined,
            "handler" => undefined
        },
        "server" => {"default-server" => undefined},
        "servlet-container" => {"default" => undefined}
    }
}
```

要查看所有可用别名的列表，请使用 alias 命令：

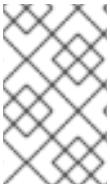
```
alias
```

结果将是：

```
alias read_undertow='/subsystem=undertow:read-resource'
```

要删除别名，请使用 unalias 命令：

```
unalias read_undertow
```



注意

别名存储在用户主文件夹中的 .aesh_aliases 文件中。

6.3. .JBOSSCLIRC 配置文件

JBoss EAP 包含运行时配置 。jbossclirc 文件可帮助您在启动新会话时初始化环境。此文件位于 *EAP_HOME*/bin/ 目录中。文件中提供的示例可用作用户特定环境设置的模板。jboss clirc 文件是存储全局 CLI 变量的理想选择。

jboss clirc 文件的内容是 CLI 支持的命令和操作的列表。启动新的管理 CLI 会话后，但在将控制权提供给用户之前，将执行此文件。如果有通过 --properties 参数指定的系统属性，则在设置属性后执行 .jbossclirc 文件。

.jbossclirc 文件示例

```
set console=/subsystem=logging/console-handler=CONSOLE
```

注意

使用 **--connect** 或 **-c** 参数时，将先执行 **jbossclirc**，然后再将客户端连接到服务器。

将按以下顺序检查是否存在 **.jbossclirc** 文件的位置：

1. 如果定义了系统属性 **jboss.cli.rc**，则其值将被视为文件的路径。
2. 用户工作目录由 **user.dir** 系统属性定义。
3. ***EAP_HOME*/bin** 目录。

6.4. 使用变量

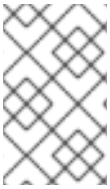
使用 Set 命令

您可以使用 **set** 命令将服务器模型的特定路径定义为变量。例如：

```
set s1=/host=master/server=server-one
```

这在受管域中很有用，因为您可以包含对主机和配置文件的引用，利用变量轻松地在不同服务器上复制脚本。例如：

```
$s1/subsystem=datasources/data-source=ExampleDS:test-connection-in-pool
```



注意

这些变量使用 **$** 引用。

使用 Unset 命令

您可以使用 **unset** 命令删除变量：

```
unset prod_db
```

使用 jbossclirc 文件

若要跨 CLI 会话使用 变量，您可以在 .jbossclirc 文件中包含这些变量。此文件位于 *EAP_HOME*/bin/ 目录中。

例如：

```
set s1=/host=master/server=server-one
set s2=/host=master/server=server-two
```

现在，重启管理 CLI 并发出一个 set 命令来检查可用变量：

```
set
```

输出将是：

```
s1=/host=master/server=server-one
s2=/host=master/server=server-two
```

这些变量可能显示在命令行的任何部分，并在命令行解析阶段得到解决。在本例中，prod _db 变量将解析到数据源：

```
$prod_db/statistics=jdbc:read-resource
```

使用 Echo 命令

使用 echo 命令检查变量的值：

```
echo $prod_db
```

输出将是：

```
/subsystem=datasources/data-source=ExampleDS
```

示例

以下一般示例演示了变量的显示位置，以及整个命令行可能由变量组成：

```
$prod_db:$op($param=$param_value)
$cmd --$param=$param_value
```

注意

这些变量可帮助您进行 CLI 脚本编写：

第 7 章 管理 CLI 命令历史记录

管理 CLI 具有在应用服务器安装中默认启用的命令历史记录功能。历史记录同时保留为活动 CLI 会话易失性内存中的记录，并附加到日志文件中，该文件自动保存在用户的主目录中，即 jboss -cli-history。默认情况下，此历史记录文件配置为记录最多 500 个 CLI 命令。可以在 *EAP_HOME*/bin/jboss-cli.xml 文件中自定义历史记录文件位置和最大历史记录条目。

history 命令本身将返回当前会话的历史记录，或者通过附加参数从会话内存中禁用、启用或清除历史记录。管理 CLI 还能够使用键盘的箭头键在命令和操作历史记录中来回原样。

查看管理 CLI 命令历史记录

显示自管理 CLI 启动或历史记录清除命令后存储在内存中的 CLI 命令历史记录。

```
history
```

清除管理 CLI 命令历史记录

从会话内存和保存到用户主目录的 .jboss-cli-history 文件中清除 CLI 命令的历史记录。

```
history --clear
```

启用管理 CLI 命令历史记录

在会话内存中和保存至用户主目录的 .jboss-cli-history 文件中记录 CLI 命令。

```
history --enable
```

禁用管理 CLI 命令历史记录

不要在会话内存中记录 CLI 命令，也不要记录保存在用户主目录中的 .jboss-cli-history 文件中。

```
history --disable
```

第 8 章 管理 CLI 日志

您可以在日志文件中捕获输出和其他管理 CLI 信息。默认情况下，禁用管理 CLI 日志。您可以使用 *EAP_HOME*/bin/jboss-cli-logging.properties 文件启用它和配置其他日志记录设置。

配置管理 CLI 日志

1. 编辑 *EAP_HOME*/bin/jboss-cli-logging.properties 文件。

2. 取消注释或添加以下行以启用日志记录：

```
# uncomment to enable logging to the file
logger.handlers=FILE
```

3. 将日志级别从 OFF 更改为所需的级别，如 INFO 或 ALL。

```
logger.org.jboss.as.cli.level=INFO
```

重新启动管理 CLI 后，输出将记录到 *EAP_HOME*/bin/jboss-cli.log 文件。

有关在日志记录属性文件中配置其他设置的详情，请参考 JBoss EAP *开发指南中的配置 logging.properties* 部分。

第 9 章 批处理

批处理允许按顺序分组多个操作请求，并作为一个单元一起执行。如果序列中的任何操作请求失败，则回滚整个操作组。

注意

批处理模式不支持条件语句。

1. 使用批处理管理 CLI 命令进入 批处理 模式。

```
batch
```

批处理模式由提示符中的哈希符号(#)表示。

2. 向批处理添加操作请求。

处于批处理模式后，请正常输入操作请求。操作请求按照输入的顺序添加到批处理中。

您可以编辑和重新排序批处理命令。您也可以在以后存储批量进行处理。有关可用于处理批处理的命令的完整列表，请参阅批处理模式命令。

3. 运行批处理。

输入整个操作请求序列后，使用 run-batch 命令运行批处理。

```
run-batch
```

输入的操作请求序列以批处理方式完成，并将结果打印到终端： 批处理成功执行。

外部文件中的批处理命令

经常运行的批处理命令可以存储在外部文本文件中，也可以通过将完整路径作为参数传递给 批处理 命令来加载，或者作为参数直接传递到 run-batch 命令。

您可以使用文本编辑器创建批处理命令文件，并将每个命令放置在自己的行中。

以下命令将以批处理模式加载 myscript.txt 文件。然后，可以编辑或删除来自此文件的命令。可以插入新的命令。此批处理会话所做的更改不会持久存在于 myscript.txt 文件中。

```
batch --file=myscript.txt
```

以下命令将立即运行存储在 myscript.txt文件中的批处理命令

```
run-batch --file=myscript.txt
```

输入的操作请求序列以批处理形式完成。

第 10 章 嵌入服务器以进行离线配置

您可以在管理 CLI 进程中嵌入 JBoss EAP 单机服务器或主机控制器流程。这可让您配置服务器，而无需它在网络中可见。此功能的常见用途是服务器的初始配置，例如在服务器在线之前管理与安全相关的设置或避免端口冲突。

这种通过管理 CLI 对 JBoss EAP 安装进行直接的本地管理不需要基于套接字的连接。您可以通过与远程 JBoss EAP 服务器交互的方式将管理 CLI 与嵌入式服务器配合使用。所有可用于管理远程服务器的标准管理 CLI 命令都可用。

启动嵌入式单机服务器

您可以使用管理 CLI 在本地启动单机服务器，以修改独立配置，而无需启动附加进程或打开网络套接字。

以下过程将启动管理 CLI，启动嵌入的单机服务器，修改配置，然后停止嵌入的服务器。

1. 启动管理 CLI。

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

2. 启动嵌入式单机服务器。

   传递 在 --std-out=echo 参数中将标准输出打印到终端。

```
embed-server --std-out=echo
```

3. 执行所需的操作。

```
/socket-binding-group=standard-sockets/socket-binding=management-http:write-attribute(name=port,value=9991)
```

4. 停止嵌入式服务器。

```
stop-embedded-server
```

   这会停止嵌入的服务器，并返回到管理 CLI 会话。如果也想退出管理 CLI 会话，您可以使用

quit 命令。

指定服务器配置

默认情况下，嵌入式服务器将使用 standalone.xml 配置文件。您可以使用 --server-config 参数来指定要使用的不同配置文件。

```
embed-server --server-config=standalone-full-ha.xml
```

从仅限管理员模式开始

默认情况下，嵌入式服务器以 管理员模式 启动，它将启动与服务器管理相关的服务，但不启动其他服务或接受最终用户请求。这对于服务器的初始配置非常有用。

您可以通过将 --admin-only 参数设置为 false，以正常运行模式启动嵌入式服务器。

```
embed-server --admin-only=false
```

您还可以使用 reload 命令更改正在运行的模式。

```
reload --start-mode=normal
```

控制标准输出

您可以控制如何处理嵌入式服务器的标准输出。默认情况下，标准输出将被丢弃，但您可以在服务器日志中找到输出。您可以传递 --std-out=echo，使服务器输出显示有管理 CLI 输出。

```
embed-server --std-out=echo
```

引导超时

默认情况下，si med-server 命令会无限期地阻止等待嵌入的服务器完全启动。您可以使用 --timeout 参数指定等待的时间（以秒为单位）。当嵌入式服务器到达可以通过 CLI 管理的点时，小于 1 的值将返回。

```
embed-server --timeout=30
```

从空白配置开始

启动嵌入的服务器时，您可以指定要以空配置开头。如果要使用管理 CLI 命令构建整个服务器配置，这非常有用。

```
embed-server --server-config=my-config.xml --empty-config
```

如果文件已存在，此命令将失败，这有助于避免意外删除配置文件。您可以通过传递 **--remove-existing** 参数来指定删除任何现有配置。

```
embed-server --server-config=my-config.xml --empty-config --remove-existing
```

启动嵌入式主机控制器

您可以使用管理 CLI 在本地启动主机控制器，以修改域和主机控制器配置，而无需启动其他进程或打开网络套接字。

嵌入式主机控制器不启动任何服务器。另外，启动嵌入式主机控制器时，您不能使用 **--admin-only** 参数。它始终会启动，就像它处于 仅限管理员 模式时一样。

以下过程将启动管理 CLI，启动嵌入式主机控制器，修改配置，然后停止嵌入的主机控制器。

1. 启动管理 CLI。

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

2. 启动嵌入式主机控制器。

传递 在 **--std-out=echo** 参数中将标准输出打印到终端。

```
embed-host-controller --std-out=echo
```

3. 执行所需的操作。

```
/host=HOST_NAME:write-attribute(name=name,value=NEW_HOST_NAME)
```

4. 停止嵌入式主机控制器。

```
stop-embedded-host-controller
```

指定主机控制器配置

默认情况下，嵌入式主机控制器将使用 **domain.xml** 作为域配置，**host.xml** 用于主机配置。您可以使用 **--domain-config** 和 **--host-config** 参数来指定要使用的不同配置文件。

```
embed-host-controller --domain-config=other-domain.xml --host-config=host-slave.xml
```

注意

根据您使用的替代配置文件，您可能需要在启动管理 CLI 时设置某些属性。例如，

```
$ EAP_HOME/bin/jboss-cli.sh -Djboss.domain.master.address=127.0.0.1
```

控制标准输出

您可以控制如何处理嵌入式主机控制器的标准输出。默认情况下，标准输出将被丢弃，但您可以在主机控制器的日志中找到输出。您可以传递 **--std-out=echo**，使主机控制器输出显示在管理 CLI 输出中。

```
embed-host-controller --std-out=echo
```

引导超时

默认情况下，**in p-host-controller** 命令会无限期地阻止等待嵌入的主机控制器完全启动。您可以使用 **--timeout** 参数指定等待的时间（以秒为单位）。当嵌入式主机控制器到达可以通过 CLI 管理的点时，值小于 1 时将返回。

```
embed-host-controller --timeout=30
```

使用管理 CLI 进行非修改类加载

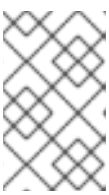
使用 ***EAP_HOME*/bin/jboss-cli.sh** 脚本启动管理 CLI，使用模块化类加载环境。如果使用 ***EAP_HOME*/bin/client/jboss-cli-client.jar** 在非模块化类加载环境中运行管理 CLI，您将需要指定 root JBoss EAP 安装目录。

1. 启动管理 CLI。

```
$ java -jar EAP_HOME/bin/client/jboss-cli-client.jar
```

2. 启动嵌入的服务器，指定 root 安装目录。

```
embed-server --jboss-home=/path/to/EAP_HOME
```



注意

若要嵌入主机控制器，可使用 **embed-host-controller** 命令。

嵌入逻辑将为服务器设置适当的模块化类加载环境。模块类加载程序的模块路径将具有一个元素： *EAP_HOME*/modules。

无论您以哪种方式启动管理 CLI，嵌入式服务器都将在模块化类加载环境中运行。

第 11 章 如何...

以下 CLI 命令和操作提供了有关如何完成某些任务的基本示例。 *具体步骤，请参阅《配置指南》、配置消息传递 或其他 JBoss EAP 文档的适当章节。*

除非另有指定，否则示例在作为单机服务器运行时适用。对命令使用 --help 参数以获得该命令的用法。使用 read-operation-description 获取资源特定操作的信息。

11.1. 添加数据源

```
data-source add --name=DATASOURCE_NAME --jndi-name=JNDI_NAME --driver-
name=DRIVER_NAME  --connection-url=CONNECTION_URL
```

11.2. 添加扩展

示例：添加新扩展到配置

```
/extension=EXTENSION_NAME:add
```

11.3. 添加 JMS QUEUE

```
jms-queue add --queue-address=QUEUE_NAME --entries=JNDI_NAME
```

11.4. 添加 JMS 主题

```
jms-topic add --topic-address=TOPIC_NAME --entries=JNDI_NAME
```

11.5. 添加模块

```
module add --name=MODULE_NAME --resources=PATH_TO_RESOURCE --
dependencies=DEPENDENCIES
```

其他资源

- *如需更多信息，请参阅模块和依赖项。*

重要

使用 模块 管理 CLI 命令添加和删除模块，仅作为技术预览提供。此命令不适合在受管域中使用，或在远程连接管理 CLI 时使用。在生产环境中，应当手动添加和删除模块。如需更多信息，请参阅 手动创建自定义模块并手动 删除 JBoss EAP 配置指南中的自定义模块部分 。

技术预览功能不包括在红帽生产服务级别协议（SLA）中，且其功能可能并不完善。因此，红帽不建议在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

如需有关技术预览功能支持范围的信息，请参阅红帽客户门户网站中的技术预览功能支持范围。

11.6. 添加服务器

示例：添加新服务器到受管域中的主机

```
/host=HOST_NAME/server-config=SERVER_NAME:add(group=SERVER_GROUP_NAME)
```

11.7. 添加服务器组

示例：在受管域中添加新服务器组

```
/server-group=SERVER_GROUP_NAME:add(profile=PROFILE_NAME, socket-binding-group=SOCKET_BINDING_GROUP_NAME)
```

11.8. 添加系统属性

```
/system-property=PROPERTY_NAME:add(value=PROPERTY_VALUE)
```

11.9. 克隆配置集

示例：克隆受管域中的配置集

```
/profile=PROFILE_TO_CLONE:clone(to-profile=NEW_PROFILE_NAME)
```

11.10. 创建层次结构配置集

示例：从其他配置集创建一个新配置集

```
/profile=NEW_PROFILE_NAME:add(includes=[PROFILE_1,PROFILE_2])
```

11.11. 将应用程序部署到受管域

示例：将应用程序部署到所有服务器组

```
deployment deploy-file /path/to/DEPLOYMENT.war --all-server-groups
```

示例：将应用程序部署到一个或多个服务器组

```
deployment deploy-file /path/to/DEPLOYMENT.war --server-
groups=SERVER_GROUP_1,SERVER_GROUP_2
```

11.12. 将应用程序部署到单机服务器

```
deployment deploy-file /path/to/DEPLOYMENT.war
```

11.13. 禁用所有应用程序

```
deployment disable /path/to/DEPLOYMENT.war
```

您可以使用 **deployment disable-all** 命令禁用所有部署。

```
deployment disable-all
```

11.14. 显示活动用户

示例：显示当前用户的命令

```
:whoami
```

示例：当前用户的输出

```
{
    "outcome" => "success",
    "result" => {"identity" => {
        "username" => "$local",
        "realm" => "ManagementRealm"
    }}
}
```

11.15. 显示 ATTACHMENT 的内容

您可以使用 附加 **display** 命令显示 从管理操作返回的附件的内容。这适用于返回 **attach -streams** 响应标头的任何管理操作。

例如，以下操作返回作为流附加的 **server.log** 文件：

```
/subsystem=logging/log-file=server.log:read-attribute(name=stream)
{
```

```
    "outcome" => "success",
    "result" => "f61a27c4-c5a7-43ac-af1f-29e90c9acb3e",
    "response-headers" => {"attached-streams" => [{
        "uuid" => "f61a27c4-c5a7-43ac-af1f-29e90c9acb3e",
        "mime-type" => "text/plain"
    }]}
}
```

您可以使用 附加 display 命令显示 从此操作返回的流的内容到控制台。

```
attachment display --operation=/subsystem=logging/log-file=server.log:read-attribute(name=stream)
```

这会将 server.log 文件的内容输出到控制台。

```
ATTACHMENT 3480a327-31dd-4412-bdf3-f36c94ac4a09:
2019-10-18 09:19:37,082 INFO  [org.jboss.modules] (main) JBoss Modules version 1.8.6.Final-redhat-00001
2019-10-18 09:19:37,366 INFO  [org.jboss.msc] (main) JBoss MSC version 1.4.5.Final-redhat-00001
2019-10-18 09:19:37,380 INFO  [org.jboss.threads] (main) JBoss Threads version 2.3.2.Final-redhat-1
2019-10-18 09:19:37,510 INFO  [org.jboss.as] (MSC service thread 1-1) WFLYSRV0049: JBoss EAP 7.3.0.GA (WildFly Core 10.0.0.Final-redhat-20190924) starting
...
```

11.16. 显示架构信息

显示 :product-info 命令的 schema 信息：

```
:read-operation-description(name=product-info)
```

要显示 schema 版本，在管理 CLI root 执行 ls 命令，并查找 management-*-version 值：

```
...
management-major-version=4
management-micro-version=0
management-minor-version=1
...
```

11.17. 显示系统和服务器信息

示例：显示系统和服务器信息的命令

```
:product-info
```

示例：系统和服务器信息的输出

```
{
    "outcome" => "success",
    "result" => [{"summary" => {
        "host-name" => "HOST_NAME",
        "instance-identifier" => "INSTANCE_ID",
        "product-name" => "JBoss EAP",
        "product-version" => "7.3.0.GA",
        "product-community-identifier" => "Product",
        "product-home" => "EAP_HOME",
        "standalone-or-domain-identifier" => "OPERATING_MODE",
        "host-operating-system" => "OS_NAME",
        "host-cpu" => {
            "host-cpu-arch" => "CPU_ARCH",
            "host-core-count" => CORE_COUNT
        },
        "jvm" => {
            "name" => "JAVA_VM_NAME",
            "java-version" => "JAVA_VERSION",
            "jvm-version" => "JAVA_VM_VERSION",
            "jvm-vendor" => "JAVA_VM_VENDOR",
            "java-home" => "JAVA_HOME"
        }
    }}]
}
```

同样，对于受管域，您可以显示特定 JBoss EAP 主机或服务器的信息：

```
/host=HOST_NAME:product-info
```

```
/host=HOST_NAME/server=SERVER_NAME:product-info
```

11.18. 启用所有禁用的部署

```
deployment enable DEPLOYMENT.war
```

您可以使用 **deployment enable-all** 命令启用所有部署。

```
deployment enable-all --server-groups=other-server-group
```

11.19. 获取命令超时值

示例：显示CLI命令超时值

```
command-timeout get
```

返回的值以秒为单位。值 **0** 表示没有超时。

11.20. 重新加载主机控制器

```
reload --host=HOST_NAME
```

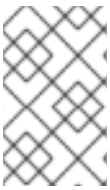
11.21. 以管理员模式重新加载主机控制器

```
reload --host=HOST_NAME --admin-only=true
```

11.22. 重新加载服务器组中的所有服务器

示例：在受管域中重新载入 Certain Server Group 中的所有服务器

```
/server-group=SERVER_GROUP_NAME:reload-servers
```



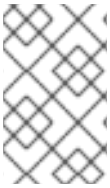
注意

若要重新加载处于暂停状态的服务器，可传递 **start-mode=suspend** 参数。

11.23. 重新加载服务器

示例：在受管域中重新载入服务器

```
/host=HOST_NAME/server-config=SERVER_NAME:reload
```



注意

若要重新加载处于暂停状态的服务器，请传递 **start-mode=suspend** 参数。

11.24. 重新加载单机服务器

```
reload
```

注意

若要以 **admin-only** 模式重新加载服务器，请传递 **--start-mode=admin-only** 参数。若要重新加载处于暂停状态的服务器，请传递 **--start-mode=suspend** 参数。

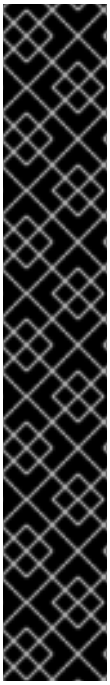
11.25. 删除扩展

示例：删除现有扩展

```
/extension=EXTENSION_NAME:remove
```

11.26. 删除模块

```
module remove --name=MODULE_NAME
```



重要

使用 模块 管理 CLI 命令添加和删除模块，仅作为技术预览提供。此命令不适合在受管域中使用，或在远程连接管理 CLI 时使用。在生产环境中，应当手动添加和删除模块。如需更多信息，请参阅 手动创建自定义模块并手动 删除 JBoss EAP 配置指南中的自定义模块部分 。

技术预览功能不包括在红帽生产服务级别协议（SLA）中，且其功能可能并不完善。因此，红帽不建议在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

如需有关技术预览功能支持范围的信息，请参阅红帽客户门户网站中的技术预览功能支持范围。

11.27. 重置命令超时值

示例：将命令超时重置为默认值

```
command-timeout reset default
```

示例：将命令超时重置为 CLI 配置提供的值

```
command-timeout reset config
```

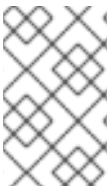
注意

CLI 配置提供的值可以在 EAP_HOME/bin/jboss-cli.xml 文件中设置，或者在启动管理 CLI 时通过 --command-timeout 参数传递。

11.28. 重启服务器组中的所有服务器

示例：重启受管域中 Certain 服务器组中的所有服务器

```
/server-group=SERVER_GROUP_NAME:restart-servers
```



注意

若要重新启动处于暂停状态的服务器，请传递 **start-mode=suspend** 参数。

11.29. 重启服务器

示例：重启受管域中的服务器

```
/host=HOST_NAME/server-config=SERVER_NAME:restart
```

注意

若要重新启动处于暂停状态的服务器，请传递 **start-mode=suspend** 参数。

11.30. 保存附件的内容

您可以使用 **附加 save** 命令将从管理操作返回的附件的内容保存到文件中。这适用于返回 **attach-streams** 响应标头的任何管理操作。

例如，以下操作返回作为流附加的 **server.log** 文件：

```
/subsystem=logging/log-file=server.log:read-attribute(name=stream)
{
    "outcome" => "success",
    "result" => "f61a27c4-c5a7-43ac-af1f-29e90c9acb3e",
    "response-headers" => {"attached-streams" => [{
        "uuid" => "f61a27c4-c5a7-43ac-af1f-29e90c9acb3e",
```

```
        "mime-type" => "text/plain"
    }]}
}
```

您可以使用 附加 save 命令将此操作返回的流的内容保存到文件中。

```
attachment save --operation=/subsystem=logging/log-file=server.log:read-attribute(name=stream) --
file=log-output.txt
```

这会将 server.log 文件的内容保存到 EAP_HOME/bin/log-output.txt。

11.31. 设置命令超时值

示例：将CLI命令的最长时间设置为 Wait，设置为 Complete

```
command-timeout set TIMEOUT_VALUE
```

该值以秒为单位设置。值 0 表示没有超时。

11.32. 关闭主机控制器

示例：关闭受管域中的主机控制器

```
shutdown --host=HOST_NAME
```

11.33. 关闭服务器

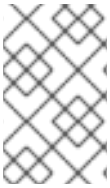
示例：关闭单机服务器

```
shutdown
```

***11.34.** 启动服务器组中的所有服务器*

***示例：在受管域中**启动 **Certain Server Group** 中的所有服务器*

```
/server-group=SERVER_GROUP_NAME:start-servers
```



注意

*要启动处于暂停状态的服务器，请传递 **start-mode=suspend** 参数。*

***11.35.** 启动服务器*

***示例：在受管域中**启动服务器*

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

注意

*要启动处于暂停状态的服务器，请传递 **start-mode=suspend** 参数。*

***11.36.** 停止服务器组中的所有服务器*

***示例：停止受管域中 Certain Server Group** 中的所有服务器*

```
/server-group=SERVER_GROUP_NAME:stop-servers
```

11.37. 停止服务器

示例：停止受管域中的服务器

```
/host=HOST_NAME/server-config=SERVER_NAME:stop
```

11.38. 拍摄配置快照

示例：获取当前配置的快照

```
:take-snapshot
```

11.39. 取消部署所有应用

示例：从受管域取消部署所有应用程序

```
deployment undeploy * --all-relevant-server-groups
```

示例：从独立域取消部署所有应用程序

```
deployment undeploy *
```

11.40. 从受管域中取消部署应用

示例：使用该 Deployment 从 All Server Groups 取消部署应用程序

```
deployment undeploy DEPLOYMENT.war --all-relevant-server-groups
```

示例：从特定服务器组取消部署应用程序

```
deployment undeploy DEPLOYMENT.war --server-groups=SERVER_GROUP_NAME
```

11.41. 从单机服务器取消部署应用

```
deployment undeploy DEPLOYMENT.war
```

11.42. 更新主机名

示例：更新受管域中的主机名称

```
/host=EXISTING_HOST_NAME:write-attribute(name=name,value=NEW_HOST_NAME)
reload --host=EXISTING_HOST_NAME
```

必须重新加载主机才能使更改生效。

11.43. 上传附件

您可以将本地文件作为附件上传到接受文件流的管理操作。例如，以下管理 CLI 命令使用 input-stream-index 选项将本地文件的内容上传到展开的部署：

```
/deployment=DEPLOYMENT_NAME.war:add-content(content=[{target-
path=/path/to/FILE_IN_DEPLOYMENT, input-stream-index=/path/to/LOCAL_FILE_TO_UPLOAD}]
```

有关将文件上传到部署的更多详细信息，请参阅《配置指南 https://access.redhat.com/documentation/en-us/red_hat_jboss_enterprise_application_platform/7.3/html-single/configuration_guide/#add_remove_content_deployment》中的将内容添加到展开的部署部分。

11.44. 查看服务器日志

```
/subsystem=logging/log-file=SERVER_LOG_NAME:read-log-file
```

附录 A. 参考资料

A.1. 管理 CLI 启动参数

下表列出了可以传递到 **jboss-cli** 脚本以启动管理 CLI 的参数：

表 A.1. 管理 CLI 参数

参数	描述
--bind	指定 CLI 将绑定到的地址。如果未提供任何服务，CLI 将根据需要自动选择一个。
--command	指定应在 CLI 会话中执行的单个命令或操作。CLI 将在 命令或 操作已执行后立即终止会话。
--command-timeout	等待命令完成的最长时间（以秒为单位）。值 **0** 表示没有超时。默认情况下没有超时。
--commands	指定应在 CLI 会话中执行的命令和操作的逗号分隔列表（必须不包含空格）。在执行了最后一个命令或命令失败后，CLI 会话将终止。
--connect, -c	指示 CLI 在启动时连接到控制器。这可避免稍后发出单独的 **连接** 命令。
--controller	启动时指定 **--connect** 选项时要连接的默认控制器主机、端口和协议，或者在管理 CLI 中不带参数发出 **connect 命令时连接** 的默认控制器主机、端口和协议。默认主机是 **localhost**，默认端口为 **9990**，默认协议为 **remote+http**。此外，也可提供包含主机、端口和协议信息的控制器别名。
--echo-command	在输出中包括以非交互模式执行的命令的提示符和 命令。
--error-on-interact	禁用在非互动模式中提供安全相关输入的提示。如果需要输入才能继续，CLI 进程将突然终止并显示错误。
--file	指定包含以非交互方式执行的命令和操作（每行一个）的文件路径。在执行了最后一个命令或命令或操作失败后，CLI 将终止。
--GUI	启动在命令行界面基础上构建的 GUI。此 GUI 不是与 JBoss EAP 管理 CLI 交互的受支持方式，应当仅用于查看配置并帮助构建 CLI 命令。
--help, -h	显示帮助消息.
--no-color-output	禁用 CLI 输出和提示颜色。
--no-local-auth	禁用本地身份验证机制，此机制允许 CLI 演示它正在本地执行到使用文件系统通过交换令牌进行管理的服务器。

参数	描述
--no-output-paging	禁用输出分页，即显示输出页面后管理 CLI 暂停时，您可以浏览和搜索输出。指定此选项时，将立即打印整个输出结果。
--output-json	以纯 JSON 格式显示操作响应。默认情况下，操作响应以 DMR 格式显示。
--password, -p	指定连接控制器时用于身份验证的密码。如果没有指定参数并且需要身份验证，则发出 **connect** 命令时会提示用户输入密码。
--properties	指定包含属性值对的属性文件的路径，以定义系统属性。属性文件使用标准 ***KEY=VALUE*** 语法。
--resolve-parameter-values	在向控制器发送操作请求之前，解析作为命令参数值或操作参数值指定的系统属性。
--timeout	指定等待连接成功的时间（毫秒）。默认值为 **5000**。
--user, -u	如果控制器需要用户身份验证，则指定用户名。如果没有指定参数并且需要身份验证，则会在发出 **连接** 命令时提示用户输入用户名。如果指定了用户，本地身份验证会自动禁用。
--version	显示应用服务器版本和环境信息。

A.2. 管理 CLI 批处理模式命令

此表提供了可用于管理 CLI 以处理批处理的命令列表。

表 A.2. 管理 CLI 批处理模式命令

命令名称	描述
clear-batch	从当前活动的批处理中删除所有现有的命令行。
discard-batch	丢弃当前活动的批处理并退出批处理模式。
edit-batch-line	通过提供用于编辑和编辑的命令的行号，编辑当前批处理中的行。例如：edit **-batch-line 2 data-source disable --name=ExampleDS**。

命令名称	描述
Holback-batch	推迟或存储当前批次。不带参数使用此命令可创建未命名的计费批处理。若要返回到此计费批处理，只需在 CLI 命令行中再次键入 **批处理**。只能有一个未命名的计费批处理。 您可以选择使用 hold **back_name** 参数提供要存储批处理的名称。若要返回到指定批处理，可将 Hol **back_name** 传递到 **batch** 命令。 使用 **batch -l** 命令查看所有计费批处理的列表。
list-batch	列出当前活动的批处理中的所有命令。
move-batch-line	通过指定您想要移动为第一个参数的行号，并将新位置作为第二个参数，重新排序批处理中的行。例如： **mo-batch-line 3** 1.
remove-batch-line	删除指定行中的 batch 命令。例如： **remove-batch-line 3**。
run-batch	运行当前活动的批处理。如果批处理成功执行，则批处理将被丢弃，CLI 将退出批处理模式。

A.3. 管理 CLI 命令

下表列出了管理 CLI 命令及其用途：要获得更多用法和参数详情，请在特定命令中使用 --help 参数。

表 A.3. 管理 CLI 命令

命令	描述
Alias	定义一个别名，格式为 ***NAME=VALUE***.如果未指定参数，则将显示别名列表。
batch	通过创建新批处理来启动批处理模式。如果存在未命名的回批，它将被重新激活。如果有 named 拥有回批，请通过指定 Hol **back_name 来**重新激活。
cd	更改指定路径.
clear	清除屏幕.
命令	允许您添加、删除和列出现有的通用类型命令。通用类型命令是分配给特定节点类型的命令，允许您执行可用于该类型实例的任何操作。它还可以修改由任何现有实例上类型公开的任何属性。
连接	在启动管理 CLI 时，使用指定的协议连接到指定主机和端口上的控制器。如果没有指定，则默认主机为 **localhost**，默认端口为 **9990**，默认协议为 **http**。

命令	描述
connection-factory	管理 **messaging-activemq** 子系统中的连接工厂。
connection-info	显示有关服务器当前连接的信息。
data-source	管理 datasource **s 子系统中的数据源** 配置。
Deployment deploy-file	部署应用.使用通配符**(*)**来部署所有应用程序。
Deployment disable	禁用预先存在的部署。
Deployment enable	启用预先存在的部署。
部署信息	显示有关单个部署或多个部署的信息。
部署取消部署	取消部署具有指定名称的应用。
deployment-overlay	管理部署覆盖。如果没有指定参数，则会列出所有现有的部署覆盖。
echo	将指定的文本输出到控制台。
echo-dmr	为 中作为参数传递的命令或操作构建 DMR 请求，并以 **toString（）** 格式回显。
help	显示帮助消息.可以与参数搭配使用，以显示特定命令或操作的帮助信息。使用 **--commands** 参数显示可用命令的列表。
history	显示内存中的 CLI 命令历史记录，并显示历史记录扩展是启用或禁用的状态。可以与 参数搭配使用，以清除、禁用和根据需要启用历史记录。
if	启动 if-else 控制流。
jdbc-driver-info	显示关于已安装 JDBC 驱动程序的信息。
jms-queue	管理 **messaging-activemq** 子系统中的 JMS 队列。
jms-topic	管理 **messaging-activemq** 子系统中的 JMS 主题。
ls	列出节点路径的内容。使用 **-l** 参数打印每行一个的结果。
module	添加和删除模块.请注意，这个命令仅作为技术预览提供。
patch	对服务器应用或回滚补丁。
PWD	显示当前工作节点的完整节点路径。

命令	描述
退出	终止命令行界面。
read-attribute	显示值，并根据参数显示受管资源属性的描述。
read-operation	显示指定操作的描述，或者列出所有可用的操作（如果没有指定）。
reload	将 **:reload** 操作请求发送到 server/domain controller，再等待控制器关闭连接，然后将控制返回给客户端。
rollout-plan	管理存储的推出计划.
run-batch	在批处理模式下，运行当前活动的批处理。虽然不在批处理模式中，可以将 与 **--file** 参数搭配使用，以批处理形式执行 文件的内容。
set	使用给定的名称使用指定的值初始化变量。
shutdown	将 **:shutdown** 操作请求发送到 server/domain controller，并等待控制器关闭连接。
Try	启动尝试catch-finally 控制流程.
unalias	删除指定的别名。
unset	删除具有指定名称的现有变量。
version	显示应用服务器版本和环境信息。
xa-data-source	在 **datasources** 子系统中管理 XA 数据源配置。

A.4. 管理 CLI 操作

下表列出了 root 级别(/)可用的管理 CLI 操作。特定资源的实际可用操作因资源而异，也取决于操作模式（单机服务器或受管域）。

操作通过冒号(:)调用。可以使用 read-operation-names 操作或使用冒号后使用 tab 自动完成来公开资源的可用操作。操作说明可使用 read-operation-description 操作显示。例如：

```
:read-operation-description(name=write-attribute)
```

表 A.4. 管理 CLI 操作

操作名称	描述
add-namespace	添加命名空间前缀映射到 namespace **属性** 的 map。
add-schema-location	添加到 **schema-locations** 属性的 map 的架构位置映射。
clean-obsolete-content	从内容存储库中清理不再引用的内容。
delete-snapshot	从快照目录删除服务器配置的快照。
full-replace-deployment	将之前上传的部署内容添加到可用内容列表中，替换运行时中相同名称的现有内容，并从可用内容列表中删除替换的内容。
list-add	添加条目到 list 属性。
list-clear	从 list 属性中清除所有条目。
list-get	从 list 属性获取条目。
list-remove	从 list 属性中删除条目。
list-snapshots	列出存储在快照目录中的服务器配置的快照。
map-clear	清除 map 属性中的所有条目。
map-get	从 map 属性获取条目。
map-put	添加条目到 map 属性。
map-remove	从 map 属性中删除条目。
product-info	返回当前服务器安装的摘要。
查询	查询资源。
read-attribute	显示所选资源的属性值。
read-attribute-group	显示所选组的属性值。
read-attribute-group-names	显示选定资源下所有属性组的名称。
read-children-names	显示给定类型下选定资源的所有子项的名称。
read-children-resources	显示有关给定类型的所有资源子代的信息。
read-children-types	显示选定资源下所有子项的类型名称。

操作名称	描述
read-config-as-xml	以 XML 格式显示当前配置。
read-operation-description	显示给定资源的操作详情。
read-operation-names	显示给定资源的所有可用操作的名称。
read-resource	显示资源的属性值，以及任何子资源的基本或完整信息。
read-resource-description	显示对资源属性、子项和操作的描述。
reload	通过关闭所有服务并重新启动来重新加载服务器。
reload-servers	重新加载当前在域中运行的所有服务器。
remove-namespace	从 **namespaces** 属性映射中删除命名空间前缀映射。
remove-schema-location	从 **schema-locations** 属性映射中删除架构位置映射。
replace-deployment	将运行时中的现有内容替换为新内容。新内容之前必须上传到部署内容存储库。
resolve-expression	接受表达式作为输入（或可解析为表达式的字符串），并根据本地系统属性和环境变量进行解析。
resolve-expression-on-domain	接受表达式作为输入（或可解析为表达式的字符串），并根据域中所有服务器上的本地系统属性和环境变量进行解析。
resolve-internet-address	取一组接口解析条件，在本地计算机上查找符合条件的 IP 地址；如果没有找到匹配的 IP 地址，则失败。
restart-servers	重新启动当前在域中运行的所有服务器。
恢复	在暂停的服务器中恢复正常操作。
restore-servers	恢复对 域中的所有服务器上的处理。
shutdown	通过调用 **System.exit(0)** 关闭服务器。
start-servers	启动受管域中当前未运行的所有已配置服务器。
stop-servers	停止当前在受管域中运行的所有服务器。
suspend	正常暂停服务器操作.所有当前请求都正常完成，但不会接受任何新请求。

操作名称	描述
suspend-servers	暂停 域中的所有服务器.所有当前操作都将完成，并且不允许新的操作。
take-snapshot	拍摄服务器配置的快照，并将它保存到快照目录中。
undefine-attribute	将所选资源的 属性值设置为 **未定义**。
upload-deployment-bytes	表示包含的字节数组中的部署内容应添加到部署内容存储库中。请注意，此操作不指示内容应部署到运行时。
upload-deployment-stream	表示应当将包含的输入流索引中的部署内容添加到部署内容存储库中。请注意，此操作不指示内容应部署到运行时。
upload-deployment-url	表示所包含 URL 上可用的部署内容应当添加到部署内容存储库中。请注意，此操作不指示内容应部署到运行时。
validate-address	检查具有指定地址的资源是否存在。
validate-operation	验证操作是否根据其描述有效。操作 **失败说明中将显示任何错误。**
whoami	返回当前经过身份验证的用户的身份。
write-attribute	设置所选资源的 属性值。

A.5. 资源属性详情

read-resource-description 操作显示资源的属性以及属性的详细信息。 下表列出了可能返回的字段，具体取决于它是否与 属性相关。

表 A.5. 资源属性详情

字段	描述
access-type	属性是只能读取、可读取和写入，还是指标。有效值为 **只读、读写 和指标。指标** 是一种只读属性，其中值不存储在持久配置中，并且可能会因为服务器上的活动而更改。
allowed	有效值列表。
alternatives	定义属性之间的独占关系。如果设置了此属性的值，则 **alternatives** 字段中列出的属性应当未定义，即使这些属性状态是必需的。
capability-reference	表示此属性的值指定由另一资源提供的指定能力的名称的动态部分。这表明该属性是对管理模型的另一个领域的引用。

字段	描述
default	如果未提供值，则用于属性的默认值。
description	属性的文本描述。
已弃用	此属性是否已弃用。它还提供了它在 中弃用的版本，以及弃用的原因。
expression-allowed	属性的值可以是表达式。
Max	数值属性的最大值。
max-length	**STRING**、LIST**或 BYTES** 类型的属性的最大长度。
Min	数字属性的最小值。
min-length	**STRING**、LIST**或 BYTES** 类型的属性的最小长度。
nillable	是否允许 属性具有定义的值。属性可以未定义，因为它不需要，或者因为此属性是必需的，但定义了替代方案。此字段可帮助用户轻松了解他们是否需要考虑出现未定义值的可能性。
必需	属性是否必须具有定义的值。如果为 **true**，则必须定义一个值或必须定义一个替代选项。如果为 **false**，则该值可能未定义。
Requires	表示，如果此属性具有定义的值，此列表中定义的属性也必须具有值。
restart-required	定义在执行 **write-attribute** 操作时必须重新启动哪些服务。此字段允许以下值： • **no-services** - 不得重启任何服务。 • 全部 **服务** - 必须重启所有服务。 • **resource-services** - 与资源关联的一些服务必须重启。 • **JVM** - 必须重启整个 JVM。
storage	属性的值是存储在持久配置文件中，还是仅只要资源在运行时才存在。该值可以是 **配置** 或 **运行时**。
type	属性值的类型。允许的值有 **BIG_DECIMAL**、**BIG_INTEGER**、**BOOLEAN**、**BYTES**、**DOUBLE**、**INT**、**LIST**、 **LONG**、**OBJECT**、**PROPERTY** 和 **STRING**。
value-type	定义类型为 **LIST** 或 **OBJECT** 的属性的额外类型信息。**LIST** 属性的 **INT 值类型** 类似于 Java **List<Integer>**。**OBJECT** 属性的 **STRING 类型** 与 Java **Map<String(Sring>)** 类似。如果 **OBJECT** 属性中的所有元素都不是同一类型，则 **value-type** 代表一个定义该对象的字段和值的完全定义复杂对象。

字段	描述
unit	属性值的单位（如果适用）。

修订到 2022 年 2 月 18 日：27:07 +1000