



**Red Hat AMQ 2021.Q3**

**使用 AMQ Spring Boot Starter**

用于 AMQ 客户端 2.10



## Red Hat AMQ 2021.Q3 使用 AMQ Spring Boot Starter

---

用于 AMQ 客户端 2.10

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

Copyright © 2021 | You need to change the HOLDER entity in the en-US/Using\_the\_AMQ\_Spring\_Boot\_Starter.ent file |.

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

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

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

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

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

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

The OpenStack<sup>®</sup> 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.

## 摘要

本指南描述了如何安装和配置库，运行实践示例，并将您的客户端与其他 AMQ 组件一起使用。

---

# 目录

<b>使开源包含更多</b> .....	<b>3</b>
<b>第 1 章 概述</b> .....	<b>4</b>
1.1. 主要特性	4
1.2. 支持的标准和协议	4
1.3. 支持的配置	4
1.4. 文档惯例	4
sudo 命令	4
文件路径	4
变量文本	4
<b>第 2 章 安装</b> .....	<b>5</b>
2.1. 先决条件	5
2.2. 使用 RED HAT MAVEN 存储库	5
2.3. 安装本地 MAVEN 存储库	5
<b>第 3 章 入门</b> .....	<b>7</b>
3.1. 先决条件	7
3.2. 运行 HELLO WORLD	7
运行示例	8
3.3. 其他示例	9
<b>第 4 章 CONFIGURATION</b> .....	<b>10</b>
4.1. 连接选项	10
4.2. 池选项	10
<b>附录 A. 使用您的订阅</b> .....	<b>12</b>
A.1. 访问您的帐户	12
A.2. 激活订阅	12
A.3. 下载发行文件	12
A.4. 为系统注册软件包	12
<b>附录 B. 使用红帽 MAVEN 存储库</b> .....	<b>14</b>
B.1. 使用在线存储库	14
将存储库添加到 Maven 设置中	14
在您的 POM 文件中添加软件仓库	15
B.2. 使用本地存储库	15
<b>附录 C. 将 AMQ BROKER 与示例搭配使用</b> .....	<b>17</b>
C.1. 安装代理	17
C.2. 启动代理	17
C.3. 创建队列	17
C.4. 停止代理	17



## 使开源包含更多

红帽承诺替换我们的代码、文档和网页属性中存在问题的语言。我们从这四个术语开始：master、slave、blacklist 和 whitelist。这些更改将在即将发行的几个发行本中逐渐实施。详情请查看 [CTO Chris Wright 信息](#)。

# 第 1 章 概述

AMQ Spring Boot Starter 是用于创建使用 AMQ 消息传递的基于 Spring 的应用的适配器。它提供了一个 Spring Boot starter 模块，供您构建独立的 Spring 应用。初学者使用 AMQ JMS 客户端通过 AMQP 1.0 协议进行通信。

AMQ Spring Boot Starter 是 AMQ 客户端的一部分，这是支持多种语言和平台的消息传递库套件。有关客户端的概述，请参阅 [AMQ 客户端概述](#)。有关此发行版本的详情，请参考 [AMQ Clients 2.10 发行注记](#)。

AMQ Spring Boot Starter 基于 [AMQP 1.0 JMS Spring Boot](#) 项目。

## 1.1. 主要特性

- 使用内置消息传递快速构建独立 Spring 应用程序
- 自动配置 JMS 资源
- 可配置 JMS 连接和会话池

## 1.2. 支持的标准和协议

- Spring Boot API 版本 2.2
- [Java Message Service](#) API 版本 2.0
- [高级消息队列协议\(AMQP\)](#)版本 1.0

## 1.3. 支持的配置

有关 [AMQ Spring Boot Starter 支持的配置](#)，请参阅红帽客户门户网站中的 [Red Hat AMQ 7 支持的配置](#)。

## 1.4. 文档惯例

### sudo 命令

在本文档中，**sudo** 用于任何需要 root 权限的命令。使用 **sudo** 时要小心，因为任何更改都可能影响整个系统。有关 **sudo** 的详情请参考 [使用 sudo 命令](#)。

### 文件路径

在这个文档中，所有文件路径都对 Linux、UNIX 和类似操作系统有效（例如 `/home/andrea`）。在 Microsoft Windows 中，您必须使用等效的 Windows 路径（例如 `C:\Users\andrea`）。

### 变量文本

本文档包含代码块，它们需要使用特定于环境的值替换。变量文本括在箭头大括号内，样式为圆形单空间。例如，在以下命令中，将 `<project-dir>` 替换为您的环境的值：

```
$ cd <project-dir>
```



## 第 2 章 安装

本章指导您完成在您的环境中安装 AMQ Spring Boot Starter 的步骤。

### 2.1. 先决条件

- 您必须有 [订阅](#) 才能访问 AMQ 发行文件和存储库。
- 要使用 AMQ Spring Boot Starter 构建程序，您必须安装 [Apache Maven](#)。
- 要使用 AMQ Spring Boot Starter，您必须安装 Java。

### 2.2. 使用 RED HAT MAVEN 存储库

配置您的 Maven 环境，以从红帽 Maven 存储库下载客户端库。

#### 流程

1. 将红帽存储库添加到您的 Maven 设置或 POM 文件。如需示例配置文件，请参阅 [第 B.1 节“使用在线存储库”](#)。

```
<repository>
  <id>red-hat-ga</id>
  <url>https://maven.repository.redhat.com/ga</url>
</repository>
```

2. 将库依赖关系添加到您的 POM 文件。

```
<dependency>
  <groupId>org.amqphub.spring</groupId>
  <artifactId>amqp-10-jms-spring-boot-starter</artifactId>
  <version>2.5.0.redhat-00001</version>
</dependency>
```

该客户端现在在 Maven 项目中可用。

### 2.3. 安装本地 MAVEN 存储库

作为在线存储库的替代选择，可以将 AMQ Spring Boot Starter 作为基于文件的 Maven 存储库安装到本地文件系统中。

#### 流程

1. [使用您的订阅](#) 下载 **AMQ 客户端 2.10.0 Spring Boot Starter Maven 存储库** zip 文件。
2. 将文件内容提取到您选择的目录中。  
在 Linux 或 UNIX 中，使用 **unzip** 命令提取文件内容。

```
$ unzip amq-clients-2.10.0-spring-boot-starter-maven-repository.zip
```

在 Windows 上，右键单击 .zip 文件并选择“**提取所有**”。

3. 配置 Maven，以使用提取的安装目录中 **maven-repository** 目录中的存储库。如需更多信息，请参阅 [第 B.2 节 “使用本地存储库”](#)。

## 第3章 入门

本章将引导您完成设置环境并运行简单消息传递程序的步骤。

### 3.1. 先决条件

- 若要构建示例，必须将 Maven 配置为 [使用红帽存储库](#) 或 [本地存储库](#)。
- 您必须有一个在 **localhost** 中侦听连接的消息代理。它必须启用匿名访问。如需更多信息，请参阅 [启动代理](#)。
- 您必须有一个名为 **example** 的队列。如需更多信息，请参阅 [创建队列](#)。

### 3.2. 运行 HELLO WORLD

Hello World 示例创建与代理的连接，发送一条包含问候语的消息到 **example** 队列，然后重新接收它。成功后，它会将收到的消息打印到控制台。

示例：发送并接收"Hello World!" - HelloWorld.java

```
package net.example;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.jms.annotation.EnableJms;
import org.springframework.jms.annotation.JmsListener;
import org.springframework.jms.core.JmsTemplate;

@EnableJms
@SpringBootApplication
public class HelloWorld implements CommandLineRunner {
    @Autowired
    private JmsTemplate jmsTemplate;

    public static void main(String[] args) {
        SpringApplication.run(HelloWorld.class, args);
    }

    @Override
    public void run(String... strings) throws Exception {
        sendMessage("Hello World!");
    }

    public void sendMessage(String text) {
        System.out.println(String.format("Sending '%s'", text));
        this.jmsTemplate.convertAndSend("example", text);
    }

    @JmsListener(destination = "example")
    public void receiveMessage(String text) {
```

```

    System.out.println(String.format("Received '%s'", text));
  }
}

```

## 运行示例

要编译并运行示例程序，请使用以下步骤：

## 流程

1. 创建新项目目录。这在以下步骤中被称为 **<project-dir>**。

2. 将示例列表复制到以下位置：

```
<project-dir>/src/main/java/net/example/HelloWorld.java
```

3. 使用文本编辑器创建新 **<project-dir>/pom.xml** 文件。在其中添加以下 XML：

```

<project>
  <modelVersion>4.0.0</modelVersion>

  <groupId>net.example</groupId>
  <artifactId>example</artifactId>
  <version>1.0.0-SNAPSHOT</version>

  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.5.0</version>
  </parent>

  <dependencies>
    <dependency>
      <groupId>org.amqphub.spring</groupId>
      <artifactId>amqp-10-jms-spring-boot-starter</artifactId>
      <version>2.5.0.redhat-00001</version>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>
</project>

```

4. 更改到项目目录，并使用 **mvn** 命令编译该程序。

```
$ mvn clean package
```

5. 使用 **java** 命令来运行该程序。

```
$ java -jar target/example-1.0.0-SNAPSHOT.jar
```

运行 Hello World 示例会导致以下控制台输出：

```
$ java -jar target/example-1.0.0-SNAPSHOT.jar

      _.-_._.-_._.-_._.-_._.-_._.-_._.-_._.-_._.-_._.-_._.-_
   /\  / ___ ' _ _ _ _ ( _ ) _ _ _ _ _ _ _ _ _ _ _ _ _ _ \
  ( ( ) \ _ _ | ' _ | ' _ | | ' _ V _ ` | \ \ \ \
   W _ _ ) | _ | | | | | | | | | ( | | ) ) ) )
   ' | _ _ | . _ | | | | | | _ \ , | / / / /
   =====|_|=====|_|_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_/_

:: Spring Boot ::      (v2.0.5.RELEASE)

[...]
2018-11-05 16:21:09.849 INFO 14805 --- [main] net.example.HelloWorld: Started
HelloWorld in 1.074 seconds (JVM running for 1.38)
Sending 'Hello World!'
Received 'Hello World!'
```

### 3.3. 其他示例

更多示例程序可从 [AMQP 1.0 JMS Spring Boot 项目](#) 和 [Spring 项目](#) 获取。

## 第 4 章 CONFIGURATION

在 **application-properties** 文件中可以使用以下选项来配置 Spring Boot 应用。

### 4.1. 连接选项

这些选项决定了 AMQ Spring Boot Starter 如何建立与远程 AMQP 对等点的连接。初学者使用 AMQ JMS 通过网络进行通信。如需更多信息，请参阅 [使用 AMQ JMS 客户端](#)。

#### **amqphub.amqp10jms.remoteUrl**

AMQ JMS 客户端用于建立新连接的连接 URI。

连接 URI 格式

```
amqp[s]://host:port[?option=value[&option2=value...]]
```

如需更多信息，请参阅 [使用 AMQ JMS 客户端 中的连接 URI](#)。

#### **amqphub.amqp10jms.username**

用于验证连接的用户名。

#### **amqphub.amqp10jms.password**

用于验证连接的密码。

#### **amqphub.amqp10jms.clientId**

应用到连接的客户端 ID。

#### **amqphub.amqp10jms.receiveLocalOnly**

如果启用，则调用 **receive** 并带有超时参数，只检查消费者的本地消息缓冲。否则，还会检查远程对等点，以确保没有可用的消息。它默认是禁用的。

#### **amqphub.amqp10jms.receiveNoWaitLocalOnly**

如果启用，调用 **receiveNoWait** 只检查消费者的本地信息缓冲。否则，还会检查远程对等点，以确保没有可用的消息。它默认是禁用的。

### 4.2. 池选项

这些选项决定了 AMQ Spring Boot Starter 如何缓存 JMS 连接和会话。初学者使用 AMQ JMS 池进行池化。如需更多信息，请参阅 [使用 AMQ JMS 池库](#)。

#### **amqphub.amqp10jms.pool.enabled**

控制是否启用池。它默认是禁用的。

#### **amqphub.amqp10jms.pool.maxConnections**

单个池的最大连接数。默认值为 1。

#### **amqphub.amqp10jms.pool.maxSessionsPerConnection**

每个连接的最大会话数。默认值为 500。负值会删除任何限制。  
如果超过限制，**createSession()** 会根据配置来阻断或抛出异常。

#### **amqphub.amqp10jms.pool.blockIfSessionPoolsFull**

如果启用，调用 **createSession()** 块直到会话在池中可用。它会被默认启用。  
如果没有可用的会话，则调用 **createSession()** 会抛出 **IllegalStateException**。

**amqphub.amqp10jms.pool.blockIfSessionPoolsFullTimeout**

被阻断调用 **createSession()** 前的时间以毫秒为单位抛出 **IllegalStateException**。默认值为 -1，即永远调用块。

**amqphub.amqp10jms.pool.connectionIdleTimeout**

当前没有贷款的连接前的时间（毫秒为单位）可以被从池中驱除。默认值为 30 秒。0 代表禁用超时。

**amqphub.amqp10jms.pool.connectionCheckInterval**

定期检查已过期连接的时间（毫秒为单位）。默认值为 0，表示禁用检查。

**amqphub.amqp10jms.pool.useAnonymousProducers**

如果启用，为所有对 **createProducer()** 的调用使用单个匿名 JMS **MessageProducer**。它会被默认启用。

在个别情况下，这种行为不可取。如果禁用，对 **createProducer()** 的每个调用都会产生新的 **MessageProducer** 实例。

**amqphub.amqp10jms.pool.explicitProducerCacheSize**

如果不使用匿名制作者，JMS **Session** 可以配置为使用显式目的地缓存特定数量的 **MessageProducer** 对象。当创建与缓存制作者不匹配的新制作者时，缓存中最旧的条目将被驱除。

**amqphub.amqp10jms.pool.useProviderJMSContext**

如果启用，使用底层 JMS 供应商的 **JMSContext** 类。它默认是禁用的。

在正常操作中，池使用自己的通用 **JMSContext** 实现来包装来自池中的连接，而不使用提供程序实施。般实施可能会限制供应商实施。但是，启用后，来自 **JMSContext** API 的连接不会由池管理。

## 附录 A. 使用您的订阅

AMQ 通过软件订阅提供。要管理您的订阅，请访问红帽客户门户中的帐户。

### A.1. 访问您的帐户

#### 流程

1. 转至 [access.redhat.com](https://access.redhat.com)。
2. 如果您还没有帐户，请创建一个帐户。
3. 登录到您的帐户。

### A.2. 激活订阅

#### 流程

1. 转至 [access.redhat.com](https://access.redhat.com)。
2. 导航到 **My Subscriptions**。
3. 导航到 **激活订阅** 并输入您的 16 位激活号。

### A.3. 下载发行文件

要访问 .zip、.tar.gz 和其他发布文件，请使用客户门户查找要下载的相关文件。如果您使用 RPM 软件包或 Red Hat Maven 存储库，则不需要这一步。

#### 流程

1. 打开浏览器并登录红帽客户门户网站 **产品下载页面**，网址为 [access.redhat.com/downloads](https://access.redhat.com/downloads)。
2. 查找 **INTEGRATION** 类别中的 **红帽 AMQ** 条目。
3. 选择所需的 AMQ 产品。此时会打开 **Software Downloads** 页面。
4. 单击组件的 **Download** 链接。

### A.4. 为系统注册软件包

要在 Red Hat Enterprise Linux 上安装此产品的 RPM 软件包，必须注册您的系统。如果您使用下载的发行文件，则不需要这一步。

#### 流程

1. 转至 [access.redhat.com](https://access.redhat.com)。
2. 进入 **Registration Assistant**。
3. 选择您的操作系统版本，再继续到下一页。
4. 使用您的系统终端中列出的命令完成注册。



有关注册您的系统的更多信息，请参阅以下资源之一：

- [Red Hat Enterprise Linux 7 - 注册系统并管理订阅](#)
- [Red Hat Enterprise Linux 8 - 注册系统并管理订阅](#)

## 附录 B. 使用红帽 MAVEN 存储库

本节论述了如何在您的软件中使用红帽提供的 Maven 存储库。

### B.1. 使用在线存储库

红帽维护一个中央 Maven 存储库，用于基于 Maven 的项目。如需更多信息，请参阅 [存储库欢迎页面](#)。

可以通过两种方式将 Maven 配置为使用 Red Hat 存储库：

- [将存储库添加到您的 Maven 设置中](#)
- [将软件仓库添加到您的 POM 文件](#)

#### 将存储库添加到 Maven 设置中

这种配置方法适用于您的用户拥有的所有 Maven 项目，只要您的 POM 文件不覆盖存储库配置并启用包含的配置集。

#### 流程

1. 找到 Maven **settings.xml** 文件。它通常位于用户主目录的 **.m2** 目录中。如果文件不存在，请使用文本编辑器创建该文件。

在 Linux 或 UNIX 中：

```
/home/<username>/.m2/settings.xml
```

在 Windows 中：

```
C:\Users\<username>\.m2\settings.xml
```

2. 在 **settings.xml** 文件的 **profiles** 元素中添加包含红帽存储库的新配置集，如下例所示：

示例：包含 Red Hat 软件仓库的 Maven **settings.xml** 文件

```
<settings>
  <profiles>
    <profile>
      <id>red-hat</id>
      <repositories>
        <repository>
          <id>red-hat-ga</id>
          <url>https://maven.repository.redhat.com/ga</url>
        </repository>
      </repositories>
      <pluginRepositories>
        <pluginRepository>
          <id>red-hat-ga</id>
          <url>https://maven.repository.redhat.com/ga</url>
          <releases>
            <enabled>true</enabled>
          </releases>
          <snapshots>
            <enabled>false</enabled>
          </snapshots>
        </pluginRepository>
      </pluginRepositories>
    </profile>
  </profiles>
</settings>
```

```

    </pluginRepository>
  </pluginRepositories>
</profile>
</profiles>
<activeProfiles>
  <activeProfile>red-hat</activeProfile>
</activeProfiles>
</settings>

```

有关 Maven 配置的更多信息，请参阅 [Maven 设置参考](#)。

在您的 POM 文件中添加软件仓库

要在项目中直接配置存储库，请在 POM 文件的 **repositories** 元素中添加一个新的条目，如下例所示：

示例：包含 Red Hat 软件仓库的 Maven pom.xml 文件

```

<project>
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.example</groupId>
  <artifactId>example-app</artifactId>
  <version>1.0.0</version>

  <repositories>
    <repository>
      <id>red-hat-ga</id>
      <url>https://maven.repository.redhat.com/ga</url>
    </repository>
  </repositories>
</project>

```

有关 POM 文件配置的更多信息，请参阅 [Maven POM 参考](#)。

## B.2. 使用本地存储库

红帽为其部分组件提供基于文件的 Maven 存储库。这些内容作为可下载存档提供，您可以提取到本地文件系统。

要将 Maven 配置为使用本地提取的存储库，请在 Maven 设置或 POM 文件中应用以下 XML：

```

<repository>
  <id>red-hat-local</id>
  <url>${repository-url}</url>
</repository>

```

**\${repository-url}** 必须是包含提取仓库本地文件系统路径的文件 URL。

表 B.1. 本地 Maven 存储库的 URL 示例

操作系统	文件系统路径	URL
Linux 或 UNIX	<code>/home/alice/maven-repository</code>	<code>file:/home/alice/maven-repository</code>

操作系统	文件系统路径	URL
Windows	<b>C:\repos\red-hat</b>	<b>file:C:\repos\red-hat</b>



运行完示例后，使用 **artemis stop** 命令停止代理。

```
$ <broker-instance-dir>/bin/artemis stop
```

2021-08-31 15:44:40 +1000 修订