AXIOM JDK

Axiom ActiveMQ
Руководство администратора

Axiom JDK I Май 2026

Copyright © 2019-2026 Все права защищены АО "АКСИОМ" (АКСИОМ)

Программное обеспечение АКСИОМ содержит программное обеспечение с открытым исходным кодом. Дополнительная информация о коде сторонних разработчиков доступна на сайте https://axiomjdk.ru/third_party_licenses. Для дополнительной информации о том, как получить копию исходного кода, можно обратиться по адресу info@axiomjdk.ru.

ДАННАЯ ИНФОРМАЦИЯ МОЖЕТ ИЗМЕНЯТЬСЯ БЕЗ ПРЕДВАРИТЕЛЬНОГО УВЕДОМЛЕНИЯ. АКСИОМ ПРЕДОСТАВЛЯЕТ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ "КАК ЕСТЬ" БЕЗ КАКИХ-ЛИБО ГАРАНТИЙ, АКСИОМ ПРЯМО ОТКАЗЫВАЕТСЯ ОТ ВСЕХ ПОДРАЗУМЕВАЕМЫХ ГАРАНТИЙ, ВКЛЮЧАЯ, НО НЕ ОГРАНИЧИВАЯСЬ ПОДРАЗУМЕВАЕМЫМИ ГАРАНТИЯМИ ТОВАРНОЙ ПРИГОДНОСТИ И ПРИГОДНОСТИ ДЛЯ ОПРЕДЕЛЕННОЙ ЦЕЛИ.

АКСИОМ НИ ПРИ КАКИХ ОБСТОЯТЕЛЬСТВАХ НЕ НЕСЕТ ОТВЕТСТВЕННОСТИ ЗА ЛЮБЫЕ КОСВЕННЫЕ, СЛУЧАЙНЫЕ, СПЕЦИАЛЬНЫЕ, ШТРАФНЫЕ ИЛИ КОСВЕННЫЕ УБЫТКИ, ИЛИ УБЫТКИ ОТ ПОТЕРИ ПРИБЫЛИ, ДОХОДА, ДАННЫХ ИЛИ ИСПОЛЬЗОВАНИЯ ДАННЫХ, ПОНЕСЕННЫЕ ВАМИ ИЛИ ЛЮБОЙ ТРЕТЬЕЙ СТОРОНОЙ, БУДЬ ТО В РЕЗУЛЬТАТЕ ДЕЙСТВИЯ ДОГОВОРА ИЛИ ДЕЛИКТА, ДАЖЕ ЕСЛИ АКСИОМ БЫЛО ПРЕДУПРЕЖДЕНО О ВОЗМОЖНОСТИ ТАКИХ УБЫТКОВ.

Использование любого программного продукта АКСИОМ регулируется соответствующим лицензионным соглашением, которое никоим образом не изменяется условиями данного уведомления. Программные продукты и фирменные наименования: Axiom JDK, Axiom JDK Pro, Axiom Runtime Container Pro, Axiom Linux, Libercat, Libercat Certified и АКСИОМ принадлежат АКСИОМ и их использование допускается только с разрешения правообладателя.

Товарный знак Linux® используется в соответствии с сублицензией от Linux Foundation, эксклюзивного лицензиата Линуса Торвальдса, владельца знака на всемирной основе. Java и OpenJDK являются товарными знаками или зарегистрированными товарными знаками компании Oracle и/или ее аффилированных лиц. Другие торговые марки являются собственностью их соответствующих владельцев и используются только в целях идентификации.

Содержание

1. Введение

Настоящий документ содержит сведения по настройке и администрированию программного обеспечения Axiom ActiveMQ. Данный документ предназначен для системных администраторов, разработчиков и команд по внедрению.

1.1. Axiom ActiveMQ в производственной среде

Axiom ActiveMQ — это брокер сообщений с открытым исходным кодом, написанный на Java. Он полностью соответствует стандартам JMS 1.1. Разработка и поддержка осуществляются Apache Software Foundation, лицензия — Apache. Он обеспечивает высокую доступность, масштабируемость, надежность, производительность и безопасность для корпоративных приложений обмена сообщениями.

JMS — это спецификация, позволяющая разрабатывать системы, основанные на обмене сообщениями. ActiveMQ выступает в качестве брокера сообщений, который находится между приложениями и позволяет им обмениваться данными асинхронным и надежным способом.

ActiveMQ разработан для обеспечения высокой доступности, масштабируемости, надежности, производительности и безопасности корпоративных приложений обмена сообщениями. Ниже перечислены некоторые из основных особенностей ActiveMQ.

- Соответствие стандарту JMS — ActiveMQ полностью соответствует стандарту JMS 1.1. Спецификация JMS предоставляет стандартный механизм для синхронной или асинхронной доставки сообщений, однократной доставки сообщений, обеспечения надежности сообщений для подписчиков и т. д.
- Возможности подключения — ActiveMQ поддерживает HTTP/S, многоадресную рассылку, SSL, STOMP, TCP, UDP, XMPP, предоставляя таким образом широкий спектр вариантов подключения и позволяя различным системам обмениваться данными, используя выбранные ими протоколы.
- Подключаемая архитектура — ActiveMQ позволяет выбирать механизм сохранения данных, а также предоставляет возможности для настройки безопасности аутентификации и авторизации в соответствии с потребностями приложения.
- Многоплатформенность — ActiveMQ предоставляет клиентские API для многих популярных языков, таких как Java, C, C++, .NET, Perl, PHP, Python, Ruby и др. Брокер ActiveMQ будет работать в JVM, но клиенты могут быть написаны на любом из поддерживаемых языков.
- Кластер брокеров ActiveMQ позволяет подготовить сеть брокеров для масштабируемости и

может поддерживать различные типы топологий.

- Богатый функционал — ActiveMQ предоставляет множество расширенных возможностей как для брокера, так и для клиентов, а также поддерживает Apache Camel.
- Простой интерфейс администрирования — консоль администрирования ActiveMQ проста в использовании, но при этом предоставляет множество мощных функций администрирования.

2. Системные требования и настройка среды

Установка производится в системах Unix/Linux/MacOSX/macOS.

Требования к программному обеспечению для работы с ActiveMQ:

- Требуется Java SE версии 1.6 или выше.
- Установить и загрузить ActiveMQ можно с веб-сайта Apache. Для дальнейшей работы нужно получить стабильную версию.
- Для сборки и тестирования примеров, входящих в состав ActiveMQ, требуется Apache Ant.

2.1. Установка JDK

Требуется JDK версии 1.6 или выше. Скачать можно по ссылке https://axiomjdk.ru/downloads/axiom-jdk . После установки JDK вы можете проверить правильность её настройки, открыв Терминал и введя следующую команду.

```
$ java -version
```

Вы увидите следующий результат.

```
openjdk version "1.8.0_492"
OpenJDK Runtime Environment (build 1.8.0_492-b09)
OpenJDK 64-Bit Server VM (build 25.492-b09, mixed mode)
```

Результат будет различаться в зависимости от установленной версии. Важно отметить, что в приведенном выше примере установлена Axiom JDK Pro, и ее версия — 1.8.

2.2. Установка ActiveMQ

Установить и загрузить ActiveMQ можно с веб-сайта по ссылке https://download.axiomjdk.ru/activemq/5.19.2/activemq-5.19.2-1-noarch-linux.deb . Для дальнейшей работы нужно получить стабильную версию.

Для дистрибутивов Unix/Linux/Cygwin загрузите `apache-activemq-5.19.2-bin.tar.gz` или последнюю версию. После загрузки распакуйте файл.

Пример пути

```
/Users/test_user/Documents/apache-activemq-5
```

2.3. Установка Apache Ant.

Apache Ant — это библиотека Java и инструмент командной строки, помогающий в разработке программного обеспечения.

Для сборки и тестирования примеров, входящих в состав ActiveMQ, требуется Apache Ant. Скачать данную библиотеку можно по ссылке https://ant.apache.org/bindownload.cgi .

Для работы Apache Ant версии 1.9.x требуется Java 5, а для версии 1.10.x — Java 8.

Для дистрибутивов Unix/Linux/Mac загрузите apache-ant-1.10.1-bin.tar.bz2 или последнюю версию. После загрузки распакуйте файл.

Пример пути

```
/Users/test_user/Documents/apache-ant-1.10.1
```

Наконец, обязательно настройте и добавьте переменную среды $ANT_HOME в переменную среды $PATH.

Откройте терминал и введите следующую команду.

```
$ cd
```

Теперь, используя vikоманду open.bash_profile

```
$ vi .bash_profile
```

Теперь добавьте в файл следующие строки. Нажмите Iклавишу, чтобы перейти в режим вставки.

```
#Apache Ant
export ANT_HOME=/Users/test_user/Documents/apache-ant-1.10.1
#Export to PATH
export PATH=$ANT_HOME/bin:$PATH
```

Переменная PATH может содержать и другие значения. Будьте осторожны при внесении изменений.

После этого нажмите ESCклавишу, введите текст :wqи нажмите клавишу Enter.Enter

Теперь проверьте, правильно ли установлен Ant, введя в терминале следующую команду.

```
$ ant -version
Apache Ant(TM) version 1.10.1 compiled on February 2 2017
```

Версия Ant может отличаться в зависимости от того, что вы скачали.

3. Установка и развертывание

3.1. Настройка инструментов сборки для работы с репозиторием

3.1.1. Maven

Ниже приведены настройки Maven для работы с репозиторием ActiveMQ.

Доверенный репозиторий может быть определен как внутри сборочного файла проекта `pom.xml`, так и внутри общего файла настроек Maven `settings.xml`. Определения в файле проекта имеют приоритет над определениями в файле настроек. При этом определения в файле настроек являются глобальными и доступны во всех проектах.

Для добавления доверенного репозитория в файл проекта `pom.xml` нужно в секции `<repositories>` добавить элемент `<repository>` следующим образом:

```
<project>
...
    <repositories>
        <repository>
            <id>trusted-repo</id>
            <name>Axiom Trusted Repository</name>
            <url>https://activemq.axiomjdk.ru/</url>
        </repository>
    ...
    </repositories>
...
</project>
```

Аналогичным образом репозиторий может быть определен в файле настроек `settings.xml`, находящимся в корне локального репозитория (обычно ~/.m2):

```
<settings>
    ...
    <profiles>
    ...
        <profile>
            <id>myprofile</id>
```

```
            <repositories>
                <repository>
                    <id>trusted-repo</id>
                    <name>Axiom Trusted Repository</name>
                    <url>https://activemq.axiomjdk.ru/</url>
                </repository>
            </repositories>
        </profile>
    ...
    </profiles>

    <activeProfiles>
    <activeProfile>myprofile</activeProfile>
    </activeProfiles>
    ...
</settings>
```

Поскольку для доступа к доверенному репозиторию требуется пройти процедуру аутентификации, необходимо в файл настроек `settings.xml` добавить предоставленные имя пользователя и пароль:

```
<settings>
    <servers>
        <server>
            <id>trusted-repo</id>
            <username>имя пользователя</username>
            <password>ваш токен</password>
        </server>
    </servers>
</settings>
```

Важно:

Обратите внимание на элемент `<id>` внутри элемента `<server>`, он должен совпадать с элементом `<id>` внутри элемента `<repository>`. В противном случае `maven` не сможет использовать предоставленные аутентификационные данные для доверенного репозитория.

В целях безопасности, мы рекомендуем использовать шифрование пароля, для более детальных инструкций ознакомьтесь с официальной документацией.

3.1.2. Gradle

Доверенный репозиторий может быть добавлен в секции `repositories` внутри сборочного файла проекта `build.gradle` (или иным способом, плагин, скрипт инициализации и т.п.).

```
repositories {
    ...
    maven {
        name "Axiom Trusted Repository"
        url "https://activemq.axiomjdk.ru/"
        credentials {
            username "имя пользователя"
            password "ваш токен"
        }
    }
}
```

В целях безопасности, мы рекомендуем использовать специальные переменные для хранения пароля (имени пользователя) и передавать их в качестве параметров командной строки.

Для использования артефактов ActiveMQ в проектах необходимо к имени группы добавлять `ru.axiomjdk`, например, для Maven:

```
<dependency>
    <groupId>ru.axiomjdk.org.activemq</groupId>
    <artifactId>activemq-core</artifactId>
    <version>6.2.5</version>
</dependency>
```

для Gradle:

```
implementation("ru.axiomjdk.org.activemq:activemq-core:6.2.5")
```

3.1.3. Развёртывание через самостоятельный JAR-файл

Вы можете упаковать ваше приложение в исполняемый JAR-файл, который содержит все зависимости и сервер приложения. Такой JAR запускается командой `java -jar <имя_файла>.jar`. Это простой и гибкий вариант развёртывания для небольших сервисов.

Если вы используете Maven, то для создания исполняемого файла JAR необходимо выполнить следующие шаги:

1. Создайте Maven-проект в используемой IDE.
2. Укажите правильную зависимость для создания JAR в файле `pom.xml`.

```
<packaging>jar</packaging>
```

3. Добавьте необходимые зависимости. Например:

```
<dependency>
    <groupId>ru.axiomjdk.org.activemq</groupId>
    <artifactId>activemq-core</artifactId>
    <version>6.2.5</version>
</dependency>
```

4. Запустите командную строку или терминал, перейти в папку с `pom.xml` и введите команду:

```
mvn clean package
```

или

```
./mvnw clean package
```

Если вы используете Gradle, то для создания исполняемого файла JAR необходимо выполнить следующие шаги:

1. Создайте Gradle-проект в используемой IDE.

```
mainClassName = "com.company.application.Main"

jar {
  manifest {
    attributes "Main-Class": "$mainClassName"
  }
```

2. Добавьте необходимые зависимости. Например:

```
implementation("ru.axiomjdk.org.activemq:activemq-core:6.2.5")
```

3. Откройте терминал или командную строку в корневом каталоге вашего проекта (где находится файл `build.gradle`) и выполните следующую команду:

```
./gradlew bootJar
```

Задача `bootJar` автоматически упаковывает ваше приложение и все его зависимости, а также встроенный контейнер сервлетов (например, Tomcat, Axiom Libercat, Jetty или Undertow) в один исполняемый файл JAR.

После завершения сборки исполняемый JAR-файл будет размещен в каталоге `build/libs` вашего проекта.

AXIOM JDK

Axiom ActiveMQ
Руководство администратора