Бинарный репозиторий для модулей JepRia

Оглавление

[Бинарный репозиторий для модулей JepRia 1](#_Toc453321290)

[1. Термины и сокращения 3](#_Toc453321291)

[2. Введение 3](#_Toc453321292)

[3. Артефакты репозитория 3](#_Toc453321293)

[4. Структура репозитория 4](#_Toc453321294)

[4.1 Идентификация проектов и библиотек 4](#_Toc453321295)

[4.1.1 Группы для библиотек JepRia 4](#_Toc453321296)

[4.1.2 Библиотеки одной сборки 4](#_Toc453321297)

[5. Snapshot-сборки 4](#_Toc453321298)

[6. Метаданные 5](#_Toc453321299)

[7. Пополнение репозитория 5](#_Toc453321300)

[7.1 Подключение новой библиотеки 5](#_Toc453321301)

[7.1.1 Поиск библиотеки на Maven-репозитории 5](#_Toc453321302)

[7.1.2 Определение идентификаторов библиотеки 5](#_Toc453321303)

[7.1.3 Поиск официального адреса библиотеки в Интернет 6](#_Toc453321304)

[7.1.4 Скачивание библиотеки 6](#_Toc453321305)

[7.1.5 Размещение библиотеки в бинарном репозитории 6](#_Toc453321306)

[7.1.6 Создание файла свойств 6](#_Toc453321307)

[7.1.7 Обновление корпоративного бинарного репозитория 6](#_Toc453321308)

[8. Ссылки 6](#_Toc453321309)

# Термины и сокращения

TBD

# Введение

Бинарный репозиторий, описываемый в настоящем документе, представляет собой хранилище версий библиотек и прочих артефактов, используемых при сборке библиотек и приложений JepRia, а также для их установки на сервера приложений.

Пользователями бинарного репозитория являются разработчики библиотек и приложений JepRia и специалисты отдела сопровождения.

Рисунок . Схема работы с бинарным репозиторием

Общая схема работы с бинарным репозиторием предполагает размещение репозитория (будем называть его *глобальным репозиторием*) в некотором общедоступном для разработчиков и специалистов отдела сопровождения месте (общий диск, SVN, «облако», …). При этом на компьютерах пользователей, как правило, размещается *локальный репозиторий* - локальное подмножество глобального репозитория (своего рода кэш), используемое конкретным пользователем.

Синхронизация локального репозитория с глобальным на данном этапе выполняется вручную – в одну сторону - фактически в локальный репозиторий по мере необходимости производится подкачка необходимых элементов глобального репозитория. В более простом варианте, когда размер глобального репозитория невелик, локальный репозиторий может представлять собой просто копию глобального.

# Артефакты репозитория

Артефакты репозитория представляют собой версии библиотек, часть из них представлена *snapshot-сборками* (см.5).

# Структура репозитория

За основу структуры репозитория взята структура репозитория Maven:

*groupId[0]*

*|*

*… --groupId[n]*

*|--artifactId*

*|--version*

*| `---artifactId-version.jar*

*|---metadata*

Артефакты репозитория размещаются в иерархической файловой системе согласно этой структуре.

Наличие метаданных, используемых в репозиториях Maven необязательно, при работе с репозиторием при помощи Ant метаданные никак не используются.

## Идентификация проектов и библиотек

Каждый проект в Maven идентифицируется парой: *groupId* и *artifactId*, а каждая библиотека – тройкой: *groupId* и *artifactId* и *version*. Например, проект *gwt-log* имеет идентификатор *com.allen-sauer.gwt.log.gwt-log*, а библиотека версии 3.2.1 имеет идентификатор *com.allen-sauer.gwt.log.gwt-log.gwt-log-3.2.1.jar.*

Для выбора идентификатора группы - groupId - в Maven не существует жёстких правил, однако есть то, что называется «хорошей практикой»: во избежание конфликта имён для groupId так же, как для пакетов Java, выбирают доменное имя организации или сайта проекта, записанное в обратном порядке. Стоит заметить, что не все проекты придерживаются «хорошей практики», например, log4j используют другие правила именования группы.

Но в любом случае для всех используемых проектов репозиторий *BinRepo* полностью воспроизводит

существующую идентификацию проектов и библиотек, а значит и структуру директорий репозитория Maven. Для библиотек JepRia используется упомянутая выше «хорошая практика».

### Группы для библиотек JepRia

Все библиотеки JepRia размещаются в подгруппах группы *com.technology.jep.jepria*. При этом:

* библиотеки сборки *jepria* – в группе *com.technology.jep.jepria.jepria,*
* библиотеки сборки *jepria-report* – в группе *com.technology.jep.jepria.jepria-report,*
* библиотека *jrcompiler* – в группе *com.technology.jep.jepria.jrcompiler,*
* библиотека *jepria-toolkit* – в группе *com.technology.jep.jepria.jepria-toolkit.*

### Библиотеки одной сборки

Библиотеки одной сборки создаются в одном акте сборки на основе одного и того же состояния исходников. Они представляются не одним, а несколькими отдельными jar-файлами, поскольку предназначаются для выполнения в разных средах и/или в разное время. Это такие библиотеки, как *jepria*, *jepria-ejb*, *jepria-auto*, *jepria-test*, ...

Для библиотек одной сборки в репозитории не создаётся отдельных групп, все они размещаются в одной группе, каждая – под своим именем. Например, все библиотеки сборки *jepria* версии 9.0.0 размещаются в группе *com.technology.jep.jepria.jepria* под именами: *gwt-theme-9.0.0.jar, jepria-9.0.0.jar, jepria-auto-9.0.0.jar, jepria-oc4j-9.0.0.jar, jepria-test-9.0.0.jar.*

# Snapshot-сборки

Характерная особенность snapshot-сборок состоит в том, что они отражают актуальное состояние текущей разработки, доработки или исправления какой-то ветки-версии, являются самыми последними сборками на ветке и поэтому, в отличие от сборок обычных версий, время от времени изменяются.

В одной группе репозитория одновременно может существовать несколько snapshot-сборок. Например, наряду со snapshot-сборкой последней разрабатываемой версии в той же группе репозитория может присутствовать snapshot-сборка для исправления ошибки в какой-то более ранней версии библиотеки.

# Метаданные

Для размещения различной сопроводительной информации в директории каждой версии используется файл свойств *artifactId-version.properties*. Обязательным свойством для него является директория проекта на сайте разработчика или на хостинге проектов (github и т.п.). Например, для библиотеки *log4j-1.2.17.jar* файл *log4j-1.2.17.properties* будет иметь обязательную строчку:

*HomePage=http://logging.apache.org/log4j/1.2/*

# Пополнение репозитория

Пополнение репозитория выполняется разработчиками в следующих случаях:

* При подключении новой библиотеки или новой версии библиотеки
* При выпуске очередной версии библиотеки JepRia или других системных библиотек, размещаемых в репозитории
* При создании snapshot-версии библиотеки JepRia или других системных библиотек, размещаемых в репозитории

## Подключение новой библиотеки

Подключение новой библиотеки выполняется при появлении у JepRia какой-либо новой зависимости.

При этом осуществляется следующий порядок действий:

1. Поиск библиотеки на Maven-репозитории
2. Определение идентификаторов библиотеки
3. Поиск официального сайта библиотеки в Интернет
4. Создание файла свойств
5. Скачивание библиотеки с официального сайта в Интернет
6. Размещение библиотеки в бинарном репозитории
7. Обновление локального (корпоративного) бинарного репозитория

### Поиск библиотеки на Maven-репозитории

Поиск нужной версии новой библиотеки выполняется на каком-либо из Maven-репозиториев (<http://mvnrepository.com/>, <http://nexus.emergya.es/nexus/content/repositories/public/>, …) с использованием доступных средств поиска (Google, локальный поиск репозитория, …).

### Определение идентификаторов библиотеки

Идентификаторы библиотеки (groupId, artifactId, version) проще всего определить по файлу *pom.xml* (см. ниже), доступного со страницы Maven-проекта. *Pom.xml* имеет такой вид:

<project>

<modelVersion>4.0.0</modelVersion>

<groupId>**com.oracle**</groupId>

<artifactId>**ojdbc14**</artifactId>

<version>**10.1.0.5.0**</version>

<description>Auto generated POM</description>

</project>

Значения *groupId*, *artifactId,* *version* находятся в одноимённых тегах файла *pom.xml*.

Идентификаторы библиотеки используются для определения её места в иерархии директорий репозитория. Например, в вышеприведённом случае файл библиотеки *ojdbc14-10.1.0.5.0.jar* согласно своим идентификаторам должен быть размещён в директории *%BIN\_REPO\_HOME% /com/oracle/ojdbc14/10.1.0.5.0.*

### Поиск официального адреса библиотеки в Интернет

В качестве официального адреса библиотеки может выступать:

* Адрес сайта организации или индивидуального разработчика библиотеки
* Адрес раздела сайта организации или индивидуального разработчика, посвящённого библиотеке
* Адрес проекта на сайте хостинга проектов, в рамках которого разрабатывается библиотека

### Скачивание библиотеки

Скачивание библиотеки выполняется строго по её официальному адресу в Интернет. Это делается в целях упрощения прохождения различных проверок/аудитов контролирующих органов/подразделений.

### Размещение библиотеки в бинарном репозитории

Полученная библиотека размещается в бинарном репозитории согласно своим идентификаторам.

### Создание файла свойств

Для подключаемой библиотеки создаётся файл свойств с идентификатором *artifactId-version.properties*.

Файл свойств (смотри структуру в разделе [Метаданные](#_Метаданные)) размещается в одной директории с библиотекой.

### Обновление корпоративного бинарного репозитория

В то время как работа с системными модулями ведётся с использованием внешнего репозитория, для работы с прикладными модулями используется внутренний корпоративный репозиторий, который является логической репликой внешнего, то есть внутренний репозиторий всегда обновляется из внешнего.

Изменение внешнего репозитория и синхронизация его с внутренним должно происходить одним логически неделимым действием, как правило, сразу же после изменения внешнего репозитория. При отсутствии возможностей автоматической синхронизации репозиториев необходимо выполнять их синхронизацию в ручном режиме.

# Ссылки

1. [Guide to naming conventions on groupId, artifactId and version](https://maven.apache.org/guides/mini/guide-naming-conventions.html)
2. [О структуре репозитория](http://maven.apache.org/plugins/maven-deploy-plugin/examples/deploying-in-legacy-layout.html)