Как собрать простейшую Java программу с помощью Maven

Статья написана для тех, кто умеет писать простейшие программы на java, но не умеет их собирать. Этим людям уже известно, что такое классы, что такое пакеты и зачем нужен public static main(String[] argv), но код без среды разработки они не запускали, да и не понимают кому и зачем это вообще может понадобиться.

Сразу скажу, что Java программиста, который не может собрать свою программу из консольки, на работу не возьмут, и это в общем более чем достаточная причина, чтобы научиться искусству обращения с системами сборки. Остальное детали, которым и посвящена статья.

Я принципиально не буду обсуждать в статье ничего, кроме сборки минимального HelloWorld. Также я постараюсь опустить все технические детали, которые можно опустить и подробно раскрыть всё, без понимания чего обойтись нельзя.

Для того, чтобы воспользоваться информацией из статьи нужно знать, что такое xml, переменные окружения, зачем нужна переменная окружения PATH и как пользоваться консолью.

Зачем собирать код без IDE

Первый вопрос, который возникает у начинающего разработчика — как в реальной жизни может пригодиться умение собирать код без среды разработки. Она ведь установлена и настроена у всех — это первое, что делает программист, когда приходит на проект.

Ответ простой — для того, чтобы нормально организовать рабочий процесс — код нужно регулярно собирать, проверять что он вообще компилируется, а потом, для того чтобы убедиться в работоспособности кода, прогонять тесты.

Можно, конечно, выделить специального человека, который будет раз в 15 минут запускать IDE и проводить описанные выше процедуры, но это безумие, такие вещи следует делать автоматически.

Уметь собирать код без среды разработки — суровая необходимость. Настолько суровая, что для решения этой задачи существует особый класс программного обеспечения, называемого системами сборки.

Что такое система сборки?

Система сборки это программа, которая собирает другие программы. На вход система сборки получает исходный код, а на выход выдаёт программу, которую уже можно запустить.

Чем она отличается от компилятора? Если коротко, то система сборки вызывает компилятор при своей работе, а компилятор о существовании системы сборки даже не подозревает.

Если более длинно, то сборка, помимо компиляции, включает в себя ещё целый спектр задач, для решения которых компилятор не пригоден от слова совсем.

Например, если программе для работы нужны какие-нибудь картинки, то положить их в директорию с программой — задача системы сборки. Если программе нужны сторонние библиотеки, то положить их в директорию с программой — задача системы сборки. Ну и так далее.

И да, если автоматическая сборка проекта настроена и работает, то нет нужды вручную конфигурировать IDE — современная среда разработки сгенерирует проект самостоятельно, достаточно показать ей где находится конфигурация для системы сборки.

Систем сборки для java по большому счёту 3 — ant, maven и gradle. ant отживает свой век, нынче его используют либо ретрограды, либо реально крутые чуваки, типа Антона Кекса, gradle пока удел хипстеров, а вот maven — стандарт индустрии. Уметь им пользоваться просто необходимо.

Как установить maven

Maven устанавливается просто копированием в нужную директорию — никакого инсталлера нет. Как и в случае с большинством консольных утилит для использования достаточно добавить директорию maven/bin в переменную окружения PATH.

То есть, если maven находится в d:/soft/maven, то в PATH надо добавить d:/soft/maven/bin

Ещё для работы maven потребует переменную JAVA\_HOME, которая указывает на JDK. Если JDK находится в C:/Program Files/Java/jdk1.8.0\_05, то именно такое значение нужно поместить в JAVA\_HOME. Добавлять bin в конец не нужно.

После этого можно попробовать написать в консоли

mvn *--version*

Если получится, значит maven установлен.

Как структурировать проект для maven

В терминологии maven совокупность файлов с исходным кодом программы, файлов настроек и всего такого называется проектом. Директория, в которой располагаются эти файлы, называется корневой директорией проекта. В дальшейшем я буду обозначать эту директорию как <project>

Как известно, язык программирования java навязывает программисту структуру директорий, которая диктует расположение файлов с классами. Напимер класс с полным именем com.app.HelloWorld должен находиться в файле com/app/HelloWorld.java и никак иначе.

Maven добавляет к этому ограничению ещё одно — исходный код должен находиться в директории <project>/src/main/java. То есть класс com.app.HelloWorld maven будет искать в <project>/src/main/java/com/app/HelloWorld.java

Вот как будет выглядеть этот самый HelloWorld

package com.app;

public class HelloWorld {

public static void main(String[] argv) {

System.out.println("Hello world");

}

}

Как сделать описание проекта

Но просто положить файлы в ожидаемую maven структуру директорий недостаточно. Для сборки проекта необходимо его описание в виде xml документа. Maven будет искать описание в корневой директории проекта в файле pom.xml .

Содержимое минимального файла pom.xml будет примерно следующим. Для прохождения туториала можно таким его и оставить, если ничего не поменять, код нормально соберётся.

<*project* xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-

v4\_0\_0.xsd">

*<modelVersion>*4.0.0*</modelVersion>*

*<groupId>*com*</groupId>*

*<artifactId>*hello-world*</artifactId>*

*<version>*1.0-SNAPSHOT*</version>*

*</project*>

Что тут было существенно важного и что надо запомнить? Запомнить надо вот эти три строки.

<groupId>com</groupId>

<artifactId>hello-world</artifactId>

<version>1.0-SNAPSHOT</version>

Эти три строки являются идентификатором программы для внешнего мира. Программа, она же результат работы по сборке проекта, в терминологии maven называется артефактом.

Непонятное слово артефакт используется здесь вместо понятного слова программа потому, что результатом работы системы сборки может быть не только собственно программа, но и библиотека или ещё что-нибудь эдакое. Комбинация параметров groupId, artifactId и version уникальна для каждого артефакта. Об уникальности этой комбинации должен позаботиться программист.

Технически, значения этих параметров могут быть любыми, но при их выборе строго рекомендуется соблюдать правила хорошего тона.

groupId

В groupId обычно находится название какого-нибудь пакета. Напомню, что название пакета согласно тем же правилам хорошего тона, должно быть именем веб-домена, владельцем которого является автор пакета, что позволяет худо-бедно обеспечить уникальность названия.

artifactId

В artifactId — строка с именем артефакта, которое придумывает его создатель. Как правило это какое-нибудь слово, иногда с разделителями в виде тире. Например hibernate-annotation-wat. artifactId должны быть уникальны в рамках groupId.

version

Ну и наконец version это версия артефакта, которую надо увеличивать при каждом более-менее значительном изменении. Версия обычно включает цифрры и буквы. Типа 1.0-SNAPSHOT

Как собрать проект

Теперь, когда структура файлов проекта соответствует ожидаемой, а его описание присутствует в файле pom.xml, для того, чтобы собрать проект, осталось только открыть консоль, сменить текущую директорию на директорию проекта и написать в консоли:

mvn compile

После того, как maven закончит свою работу, в директории проекта появится директория target, в которой будет находиться скомпилированный код.

Скомпилированный java код выглядит так же, как исходный, только вместо файлов с расширением java в директориях, соответствующих названиями пакетов, находятся файлы с расширением class, в которых будет уже не исходный код, а код, предназначенный для интерпретации джава-машиной.

То есть теперь этот код можно запустить.

Как запустить проект

Чтобы запустить скомпилированный код, нужно в консоли из этой же директории набрать

mvn exec:java -Dexec.mainClass="com.app.HelloWorld"

После того, как maven перестанет качать всякую дрянь из интернета, где-то перед здоровой табличкой с надписью BUILD SUCCESS, появится строчка Hello World.

Код отработал, всё прошло удачно.

Вот так собирают java программы с помощью системы сборки maven.

Итого

1. maven ищет код для сборки в директории **<project>src/main/java**.
2. Инструкции по сборке maven будет искать в **<project>/pom.xml**
3. Результат работы системы сборки называется артефактом.
4. От программиста требуется задать **groupId**, **artifactId** и **version**
5. Сборка осуществляется командой **mvn compile**
6. Скомпилированный java код выглядит так же, как исходный код, но вместо файлов с расширением java, там будут файлы с расширением class.
7. Запустить скомпилированную программу можно командой  
   **mvn exec:java -Dexec.mainClass="com.app.HelloWorld"**