СБЕРБАНК ТЕХНОЛОГИИ

СРЕДСТВА СБОРКИ ПРОЕКТОВ

Дата

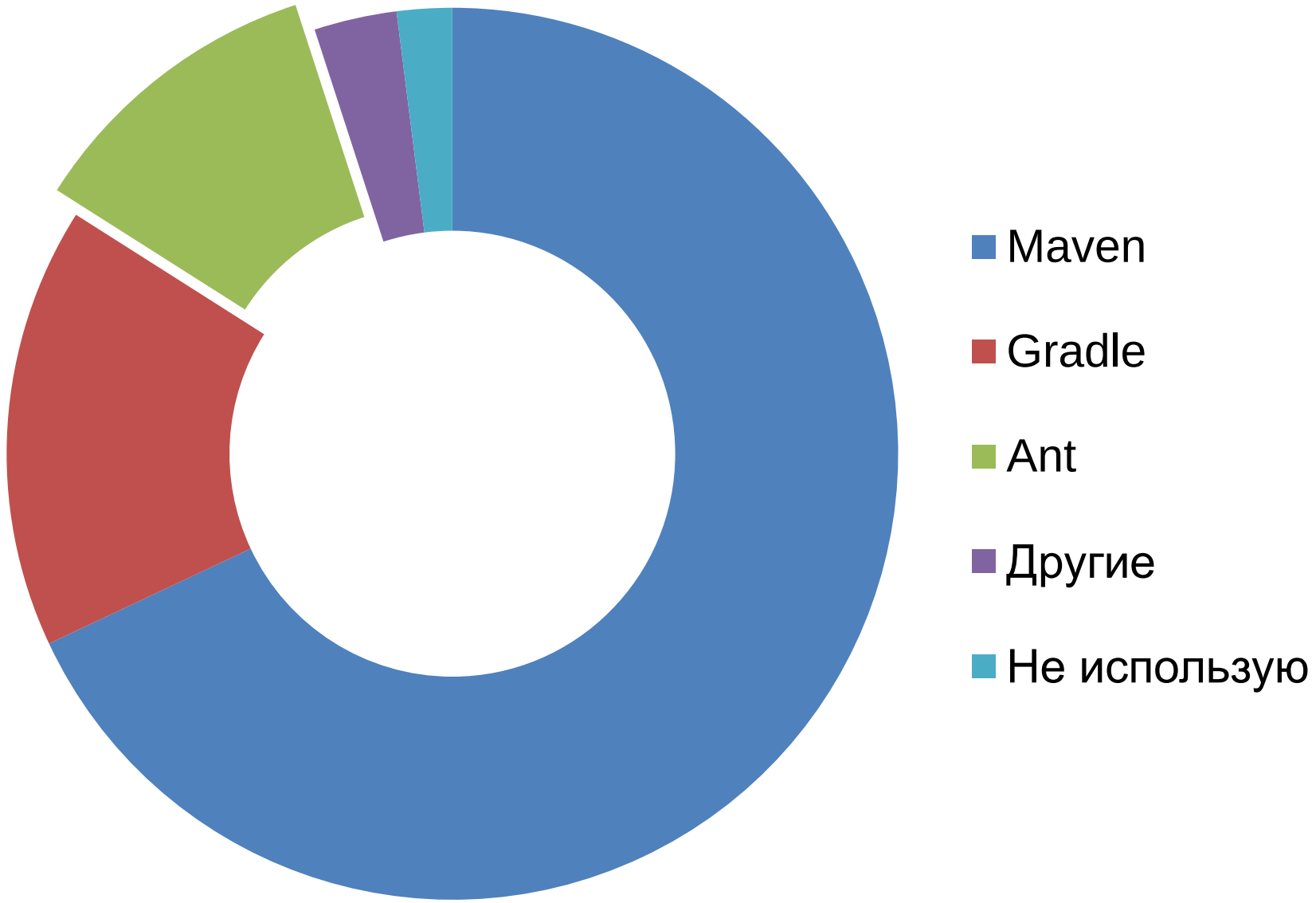
ЗАЧЕМ НУЖНЫ СРЕДСТВА СБОРКИ?

Основная цель – автоматизация и унификация процесса сборки проектов.

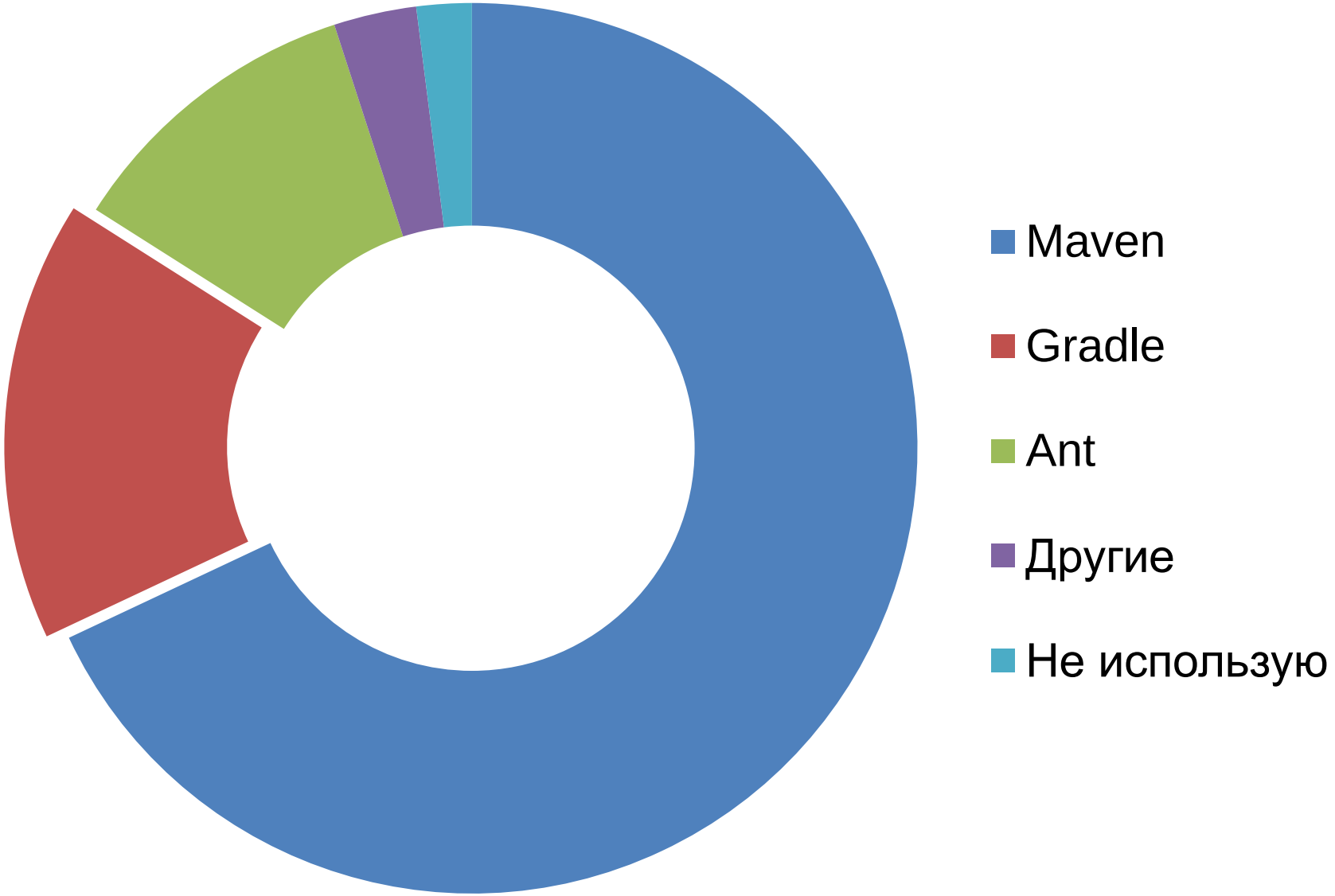
ТИПЫ СРЕДСТВ СБОРКИ

1. Ориентированные на задачи.
2. Ориентированные на продукт.

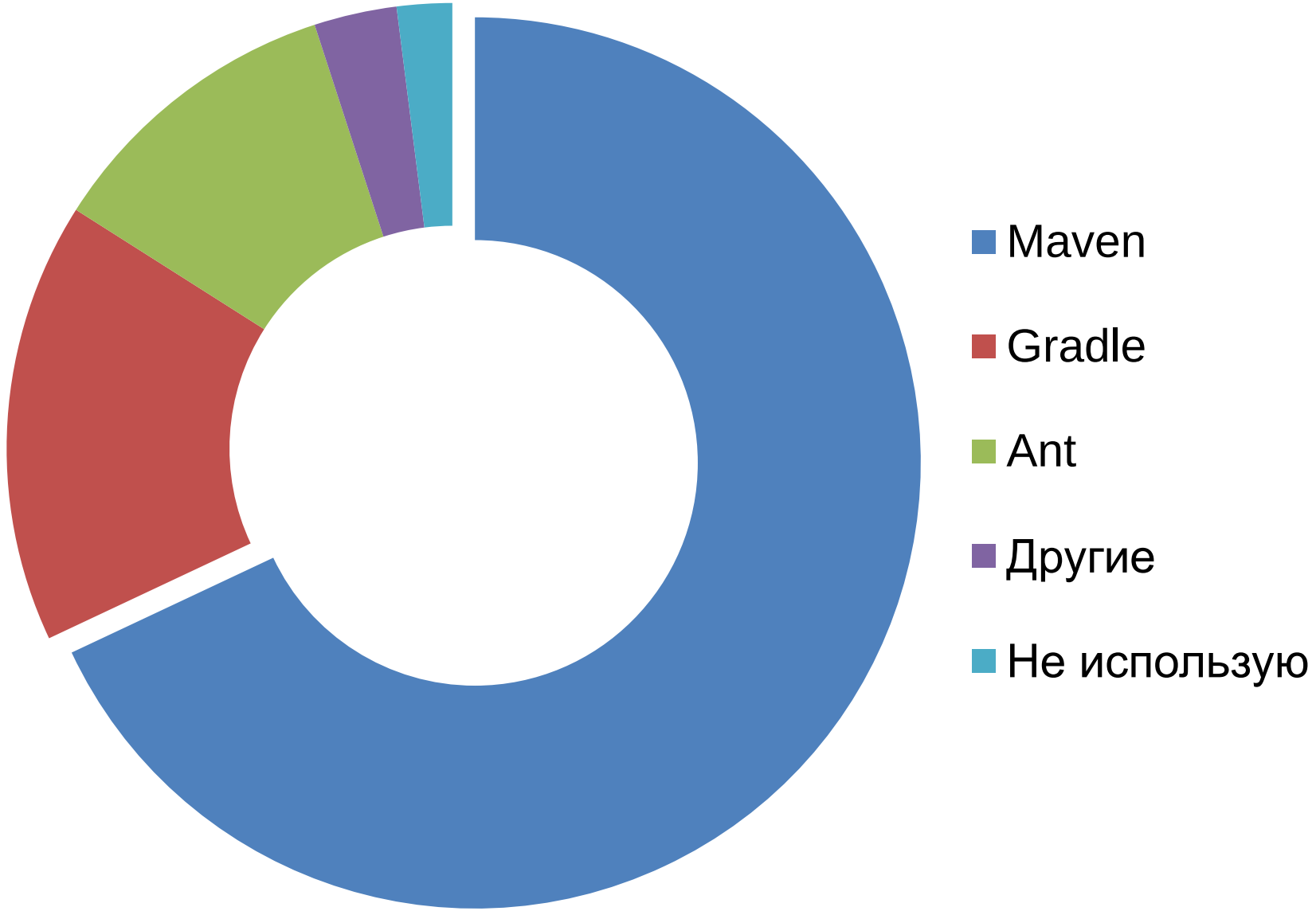
ПОПУЛЯРНЫЕ СРЕДСТВА СБОРКИ



ПОПУЛЯРНЫЕ СРЕДСТВА СБОРКИ



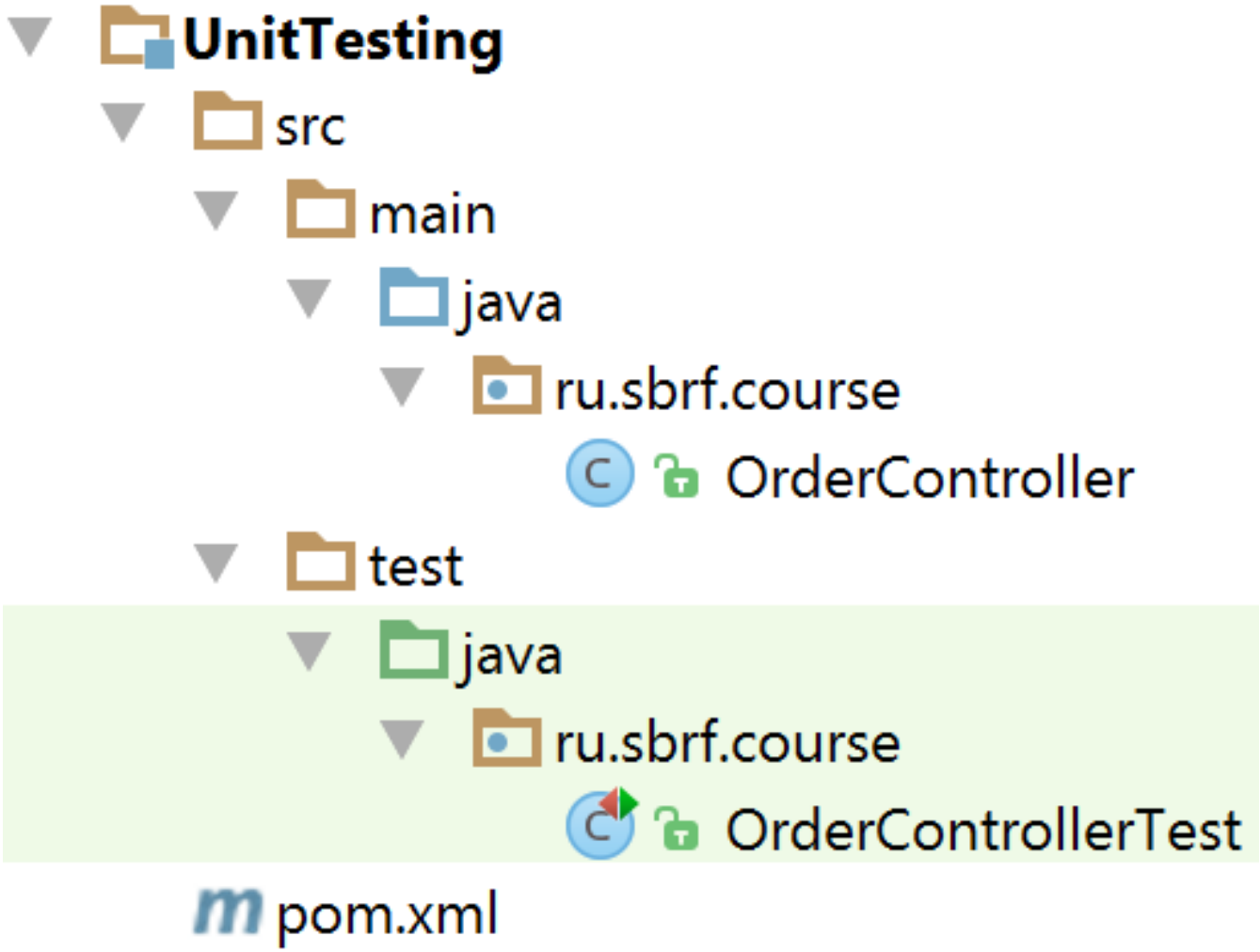
ПОПУЛЯРНЫЕ СРЕДСТВА СБОРКИ



Maven™ ПРЕИМУЩЕСТВА

- Упрощает процесс сборки.
- Предоставляет единых механизм сборки.
- Предоставляет хорошую информацию о проекте.
- Стимулирует использовать лучшие практики разработки.

Maven™ СТРУКТУРА ПРОЕКТА



Maven™ POM

```
<?xml version="1.0" encoding="UTF-8"?>
<project
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns="http://maven.apache.org/POM/4.0.0"
        xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>ru.sbrf.course</groupId>
    <artifactId>unittesting</artifactId>
    <packaging>jar</packaging>
    <version>1.0-SNAPSHOT</version>
</project>
```

Maven™ ПЕРВАЯ СБОРКА

Выполните: mvn package.

```
[INFO] -------------------------------------------------------------------
[INFO] Building unittesting 1.0-SNAPSHOT
[INFO] -------------------------------------------------------------------
…
-------------------------------------------------------
 T E S T S
-------------------------------------------------------
…
[INFO]
[INFO] --- maven-jar-plugin:2.3.2:jar (default-jar) @ unittesting ---
[INFO] Building jar: C:\Users\User\git\UnitTesting\target\unittesting-1.0-
SNAPSHOT.jar
[INFO] -------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] -------------------------------------------------------------------
…
```

Maven™ ДОБАВЛЯЕМ ЗАВИСИМОСТИ

```
<properties>
    <junit.version>4.12</junit.version>
    <slf4j.version>1.7.21</slf4j.version>
</properties>

<dependencies>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-api</artifactId>
        <version>${slf4j.version}</version>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>${junit.version}</version>
        <scope>test</scope>
    </dependency>
</dependencies>
```

Maven™ ОБЛАСТЬ ЗАВИСИМОСТИ

- *compile*
  Доступна во всех classpath текущего проекта и всех от него зависящих.

- *provided*
  Похожа на compile, но доступна только во время исполнения.

- *runtime*
  Потребуется во время исполнения, но не нужна для компиляции.

- *test*
  Требуется только для компиляции и исполнения тестов.

Maven™ РЕПОЗИТОРИИ

Все зависимости и собранные артефакты хранятся в репозиториях.

- *Внешние*.
  Внешние хранилища артефактов, доступных для скачивания.
  Пример: http://repo.maven.apache.org/maven2/

- *Локальный.*
  Ваша собственная копия. Представляет собой кэш внешних загрузок + собственные артефакты, которые ещё не были выпущены.
  Пример: %HOMEPATH%\.m2\repository

Maven™ ЖИЗНЕННЫЕ ЦИКЛЫ СБОРКИ

В Maven имеется 3 встроенных жизненных цикла сборки:

- *default*
  Компиляция, тестирование проекта и другие действия вплоть до передачи артефакта на хранение во внешний репозиторий.

- *clean*
  Очистка проекта.

- *site*
  Формирование документации.

Каждый цикл состоит из списка фаз, выполняемых последовательно до указанной.

Maven™ ФАЗЫ ЦИКЛА DEFAULT

Фаза	Описание
validate	Проверка проекта на корректность
generate-sources	Генерация исходного кода
generate-resources	Генерация ресурсов, включаемых в пакет
compile	Компиляция исходного кода
generate-test-sources	Генерация исходного кода для тестов
generate-test-resources	Создание ресурсов для тестирования
test-compile	Компиляция исходного кода
test	Исполнение тестов
package	Упаковка скомпилированного кода в распространяемый формат (jar, war, ear и др.)
integration-test	Выполнение интеграционных тестов
install	Копирование пакета в локальный репозиторий
deploy	Копирование пакета во внешний репозиторий

Maven™ ПЛАГИНЫ

- К каждой фазе цикла сборки привязываются вызовы специфичных команд плагинов.
- Тип собираемого пакета определяет список команд плагинов, привязанных к определённой фазе.
- Чтобы добавить свои команды в фазы сборки, необходимо добавить соответствующее сопоставление в pom.xml.

Maven™ ПЛАГИНЫ

```
<build>
   <plugins>
      <plugin>
         <groupId>org.codehaus.mojo</groupId>
         <artifactId>exec-maven-plugin</artifactId>
         <version>1.5.0</version>
         <executions>
            <execution>
               <phase>install</phase>
               <goals><goal>exec</goal></goals>
            </execution>
         </executions>
         <configuration>
            <executable>git</executable>
            <arguments><argument>status</argument></arguments>
         </configuration>
      </plugin>
   </plugins>
</build>
```

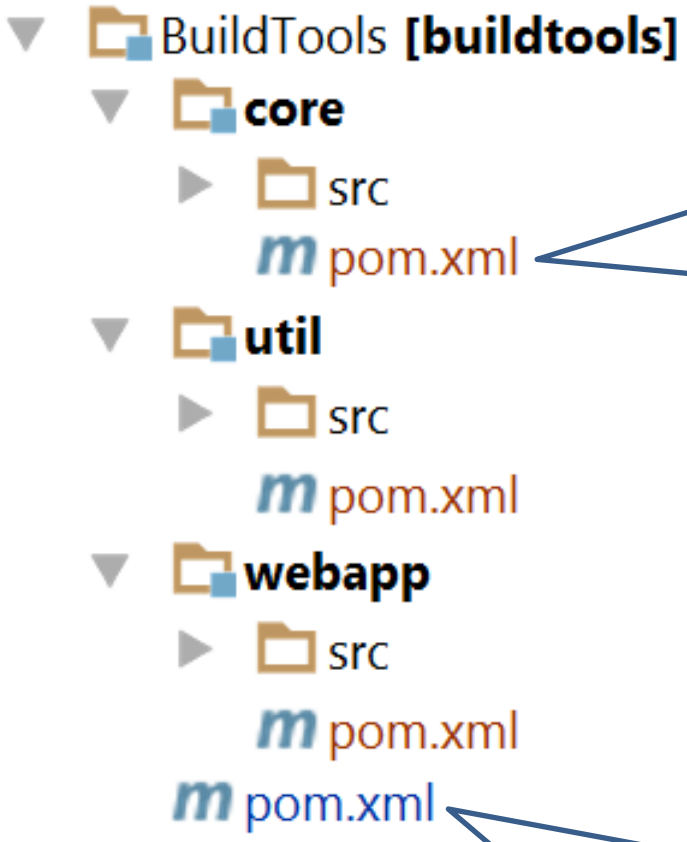
Maven™ модули

Модуль - артефакт, который агрегирует специфичную функциональность и распространяется самостоятельно.

Пример:

- slf4j-api.jar – фасад для работы с различными системами логирования;
- logback-classic.jar – реализация механизма логирования;
- ojdbc6.jar – драйверы JDBC для работы с СУБД Oracle.

Maven™ МНОГОМОДУЛЬНЫЙ ПРОЕКТ




```
<groupId>ru.sbrf.course.buildtools</groupId>
<artifactId>core</artifactId>
<packaging>jar</packaging>
<version>1.0-SNAPSHOT</version>
```

```
<groupId>ru.sbrf.course</groupId>
<artifactId>buildtools</artifactId>
<packaging>pom</packaging>
<version>1.0-SNAPSHOT</version>

<modules>
    <module>core</module>
    <module>util</module>
    <module>webapp</module>
</modules>
```

ИСПОЛЬЗУЕМЫЕ МАТЕРИАЛЫ

https://maven.apache.org/guides/getting-started/index.html

СПАСИБО!