

Основы фреймворка Maven.

Maven это фреймворк для автоматизации сборки и управления проектами на основе писания их структуры в файлах POM(Project Object Model) — подмножество языка XML.

Maven обеспечивает декларативную сборку проекта. В файлах описания проекта содержится его спецификация. Все задачи по обработке файлов, описанные в спецификации, Maven выполняет посредством их обработки последовательно встроенных и внешних плагинов.

Создание проекта с использованием Maven в IntelliJIDEA:

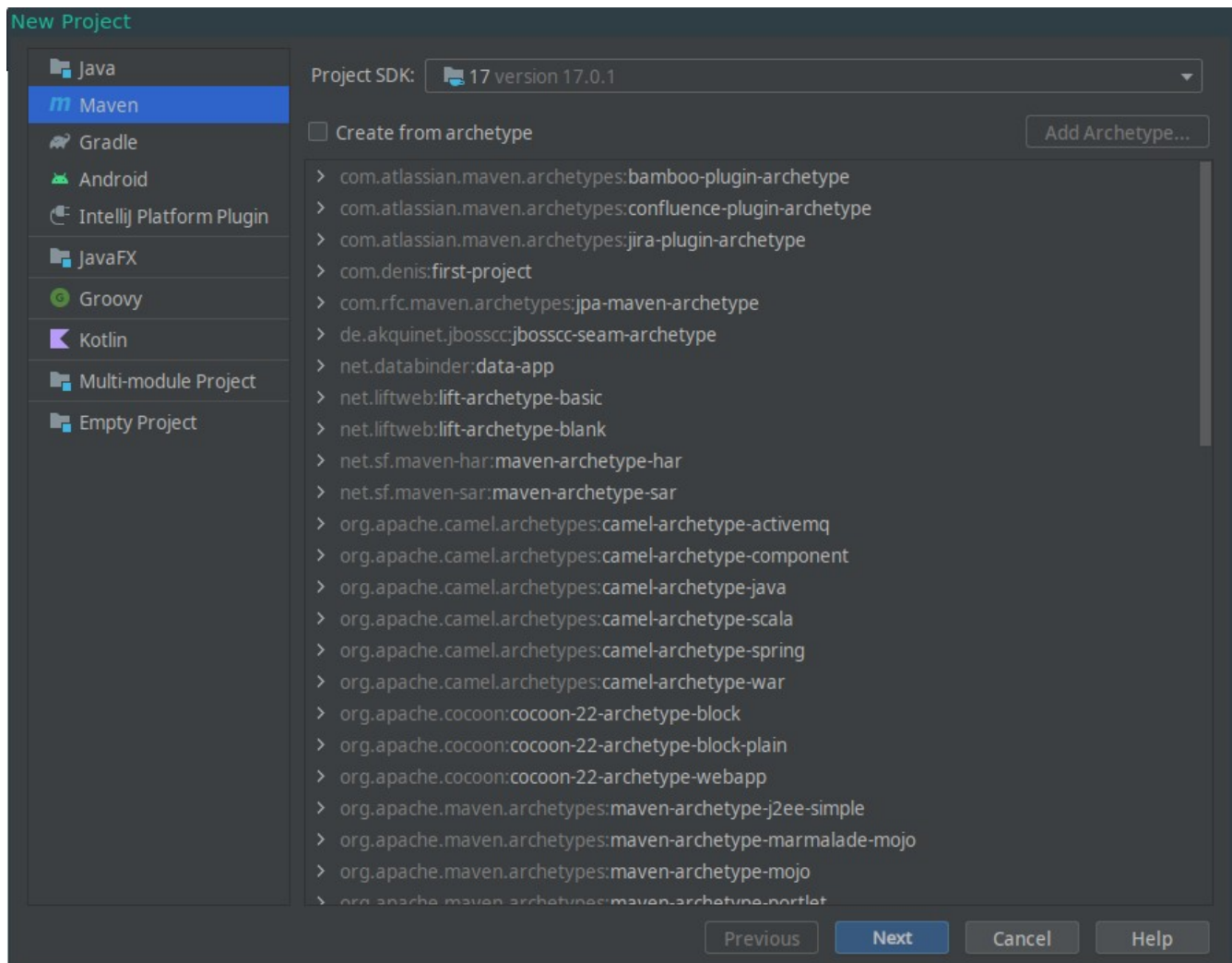


Рисунок 1: Мастер создания проектов Maven.

Выбираем Maven из предложенных вариантов слева. У меня есть возможность выбрать архетип проета, это набор инструментов для создания

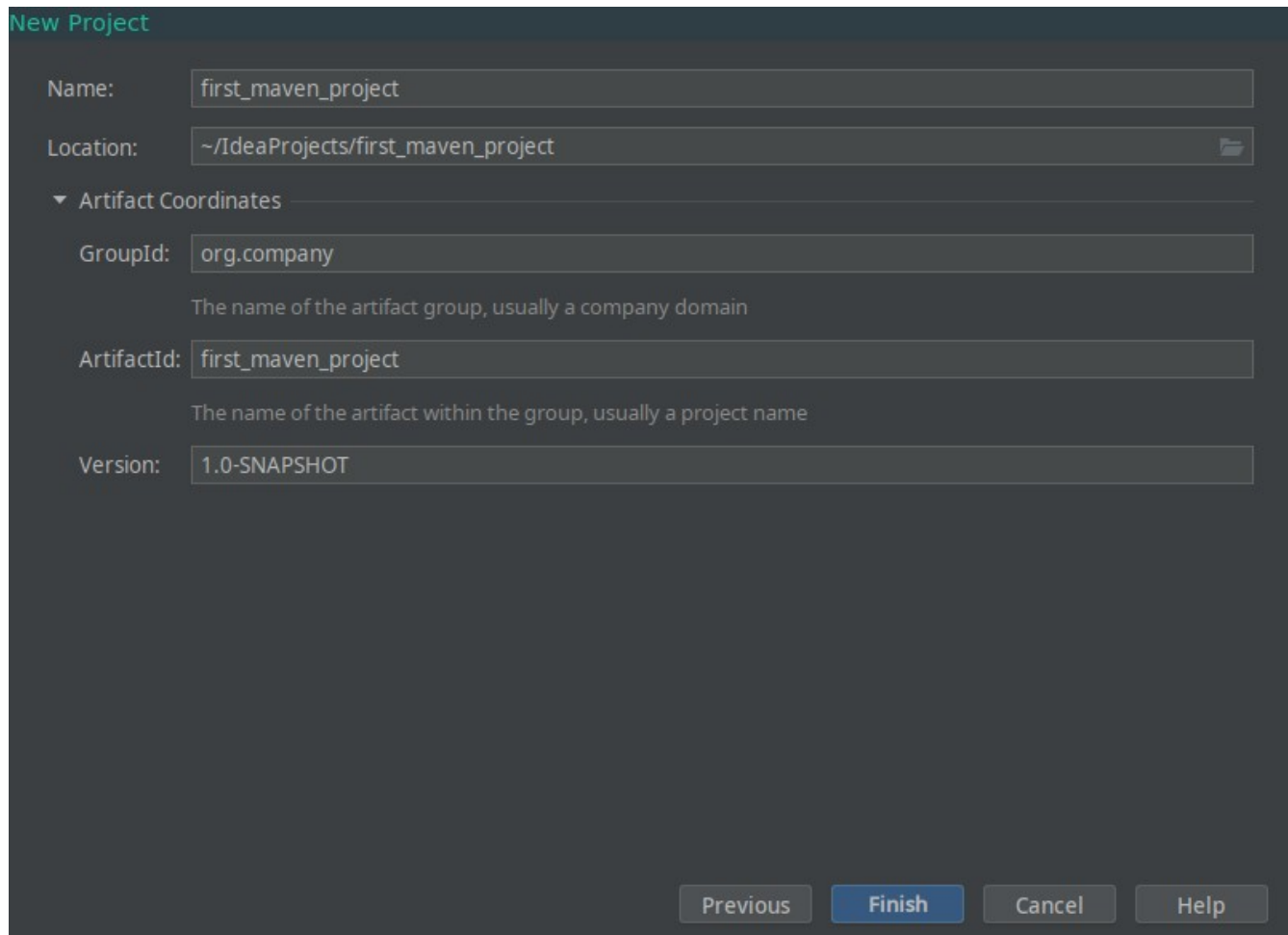
шаблонов проектов Maven, либо создать свой. Я не использую архетип и перехожу на следующий шаг.

Нас интересуют идентификационные данные `GroupId`, `ArtifactId`, `Version`.

`GroupId` идентифицирует мой проект среди других проектов. Идентификатор должен соответствовать правилам имени пакета Java. Если вкратце — обратное доменное имя. Как например : `com.company.app`.

`ArtifactId` — в моём случае название проекта. Maven оперирует так называемыми артефактами. Это приложения, плагины, архетипы и другие проекты.

`Version` — версия артефакта (проекта).
Подтверждаем данные кликом на кнопку «Finish».



The screenshot shows the 'New Project' dialog box in IntelliJ IDEA. The dialog has a title bar 'New Project' in green. It contains several input fields and buttons. The 'Name' field is set to 'first_maven_project'. The 'Location' field is set to '~/IdeaProjects/first_maven_project'. Under the 'Artifact Coordinates' section, the 'GroupId' is 'org.company' with a tooltip 'The name of the artifact group, usually a company domain'. The 'ArtifactId' is 'first_maven_project' with a tooltip 'The name of the artifact within the group, usually a project name'. The 'Version' is '1.0-SNAPSHOT'. At the bottom, there are four buttons: 'Previous', 'Finish' (highlighted in blue), 'Cancel', and 'Help'.

Рисунок 2: Мастер создания проектов Maven.

Информация для сборки проекта, поддерживаемого Maven, содержится в XML-файле с названием `pom.xml`. При запуске Maven проверяет, содержит ли конфигурационный файл все необходимые данные и все ли данные синтаксически правильно записаны.

По умолчанию, после создания проекта в рабочем пространстве открывается файл `pom.xml`.

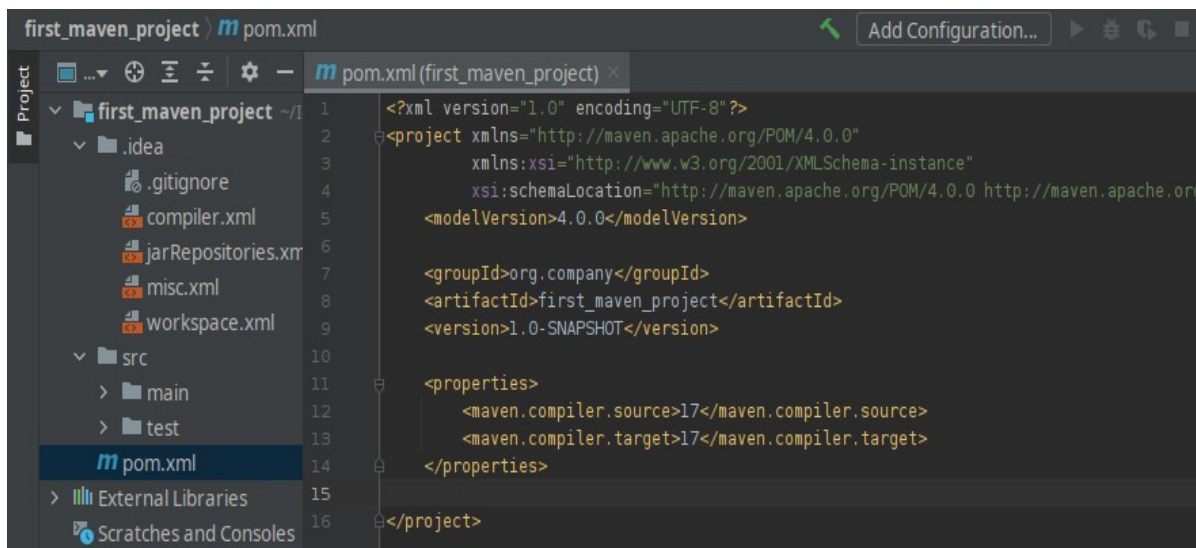


Рисунок 3: Файл проекта `pom.xml` с минимальными настройками.

версия модели для POM-ов Maven 2.x всегда 4.0.0

```
<modelVersion>4.0.0</modelVersion>
```

координаты проекта, то есть набор значений, который позволяет однозначно идентифицировать этот проект

```
<groupId>com.company</groupId>
<artifactId>first_maven_project</artifactId>
<version>1.0-SNAPSHOT</version>
```

Maven позволяет собирать проект из нескольких модулей. Каждый программный модуль включает свой проектный файл `pom.xml`. Один из проектных `pom.xml` файлов является корневым. Корневой `pom.xml` позволяет объединить все модули в единый проект. При этом в корневой проектный файл можно вынести общие для всех модулей свойства. А каждый модульный `pom.xml` должен включать параметры GAV (`groupId`, `artifactId`, `version`) корневого `pom.xml`.

Создадим модуль «interfaice» и добавим его в наш проект. Для этого правой кнопкой мыши нажимаем на имя проекта, выбираем пункт «New» и далее «Module». Нас встречает мастер создания модулей идентичный представленному на рисунке 1.

Нажимаем продолжить. Вводим имя модуля, родитель для него наш проект, следовательно, `pom.xml` файл проекта будет корневым для этого модуля.

New Module

Parent:

Name:

Location:

▼ Artifact Coordinates

GroupId:
The name of the artifact group, usually a company domain

ArtifactId:
The name of the artifact within the group, usually a module name

Version:

Рисунок 4: Мастер создания модулей maven.

В корневом файле `pom.xml` у нас появилась секция `<modules>` с модулем «`interface`». Создался модуль с собственным файлом `pom.xml` с секцией `<parent>`, где описываются родительские идентификаторы проекта: `groupId`, `artifactId`, `version`. И название модуля в собственной секции `artifactId`.

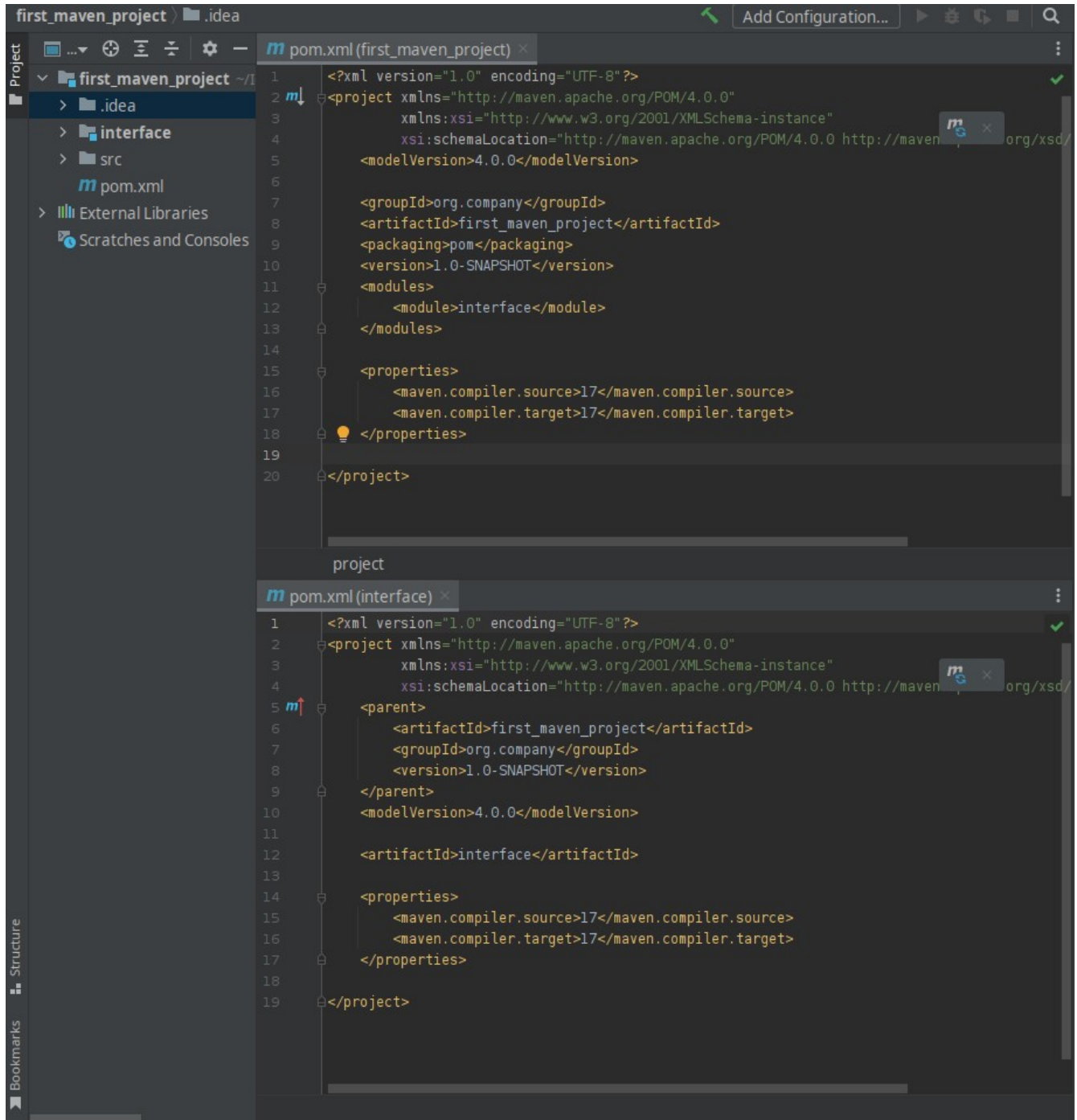
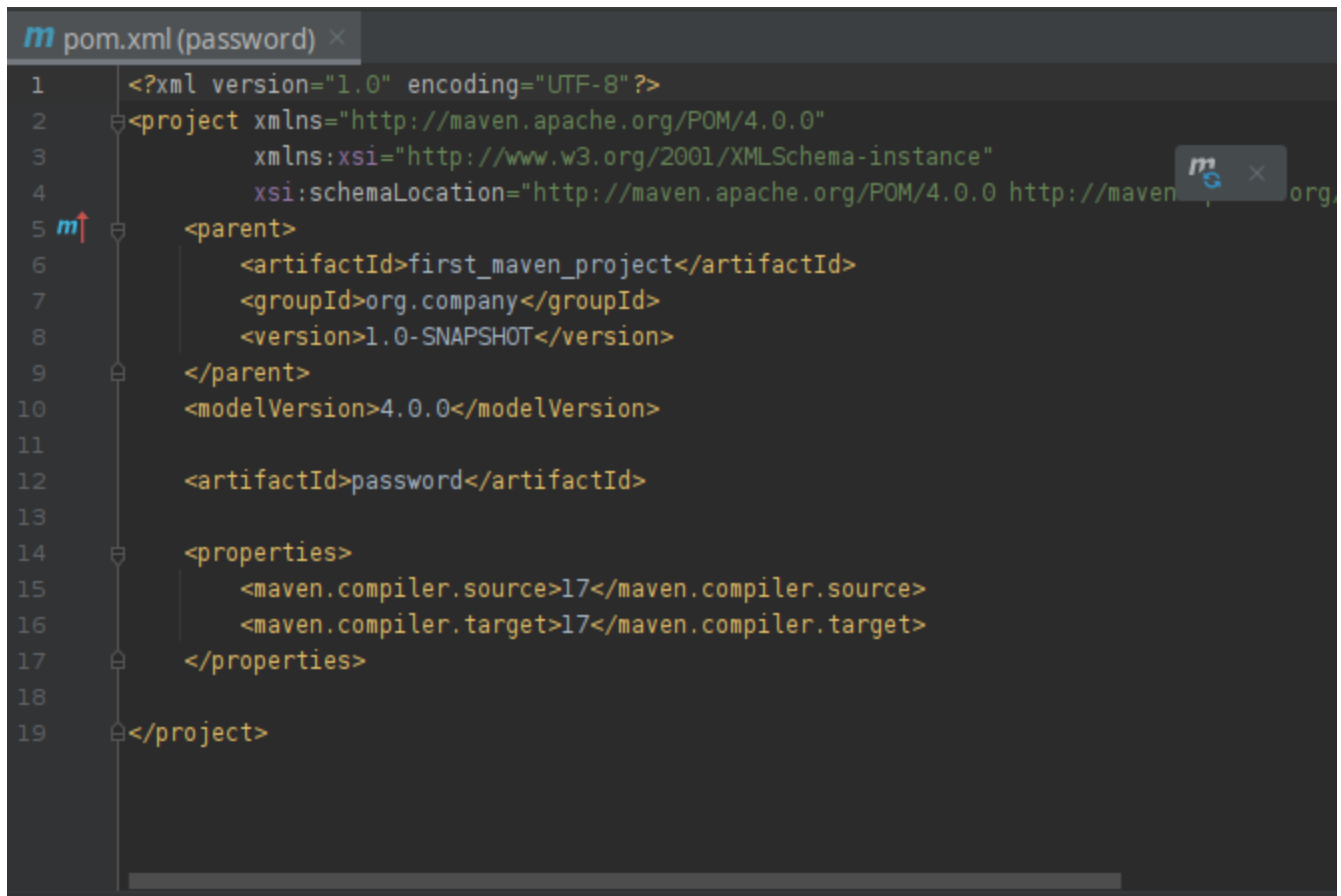


Рисунок 5: два файла `pom.xml` проекта и модуля.

Создадим, независимый от первого модуля, модуль «password». Таким же образом. По характеристикам он ничем не отличается от первого модуля.



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4     xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
5     <parent>
6         <artifactId>first_maven_project</artifactId>
7         <groupId>org.company</groupId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10    <modelVersion>4.0.0</modelVersion>
11
12    <artifactId>password</artifactId>
13
14    <properties>
15        <maven.compiler.source>17</maven.compiler.source>
16        <maven.compiler.target>17</maven.compiler.target>
17    </properties>
18
19 </project>
```

Рисунок 6: *pom.xml* модуля *password*.

Имеем два модуля зависящих от родительского pom.xml. Предположим, нам нужно, чтобы модуль «password» стал зависим от модуля «interface».

Для этого сохраняем каретку в pom.xml файле модуля «interface» в верхнем меню нажимаем на вкладку «Code» и выбираем «Generate», далее «Dependency». В открывшемся окне в первой вкладке прописываем artifactId модуля, который будет зависимым. Нажимаем добавить.

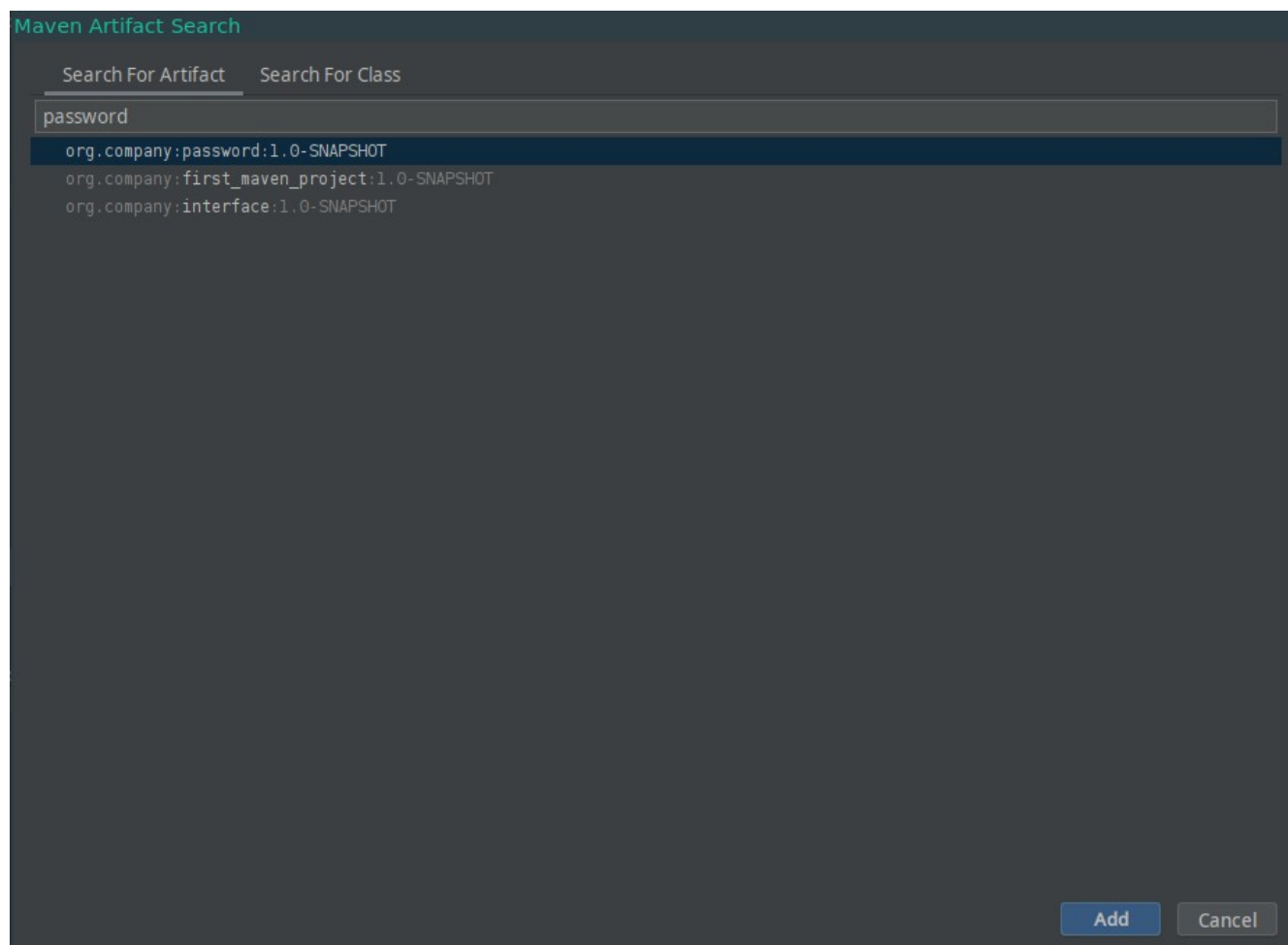
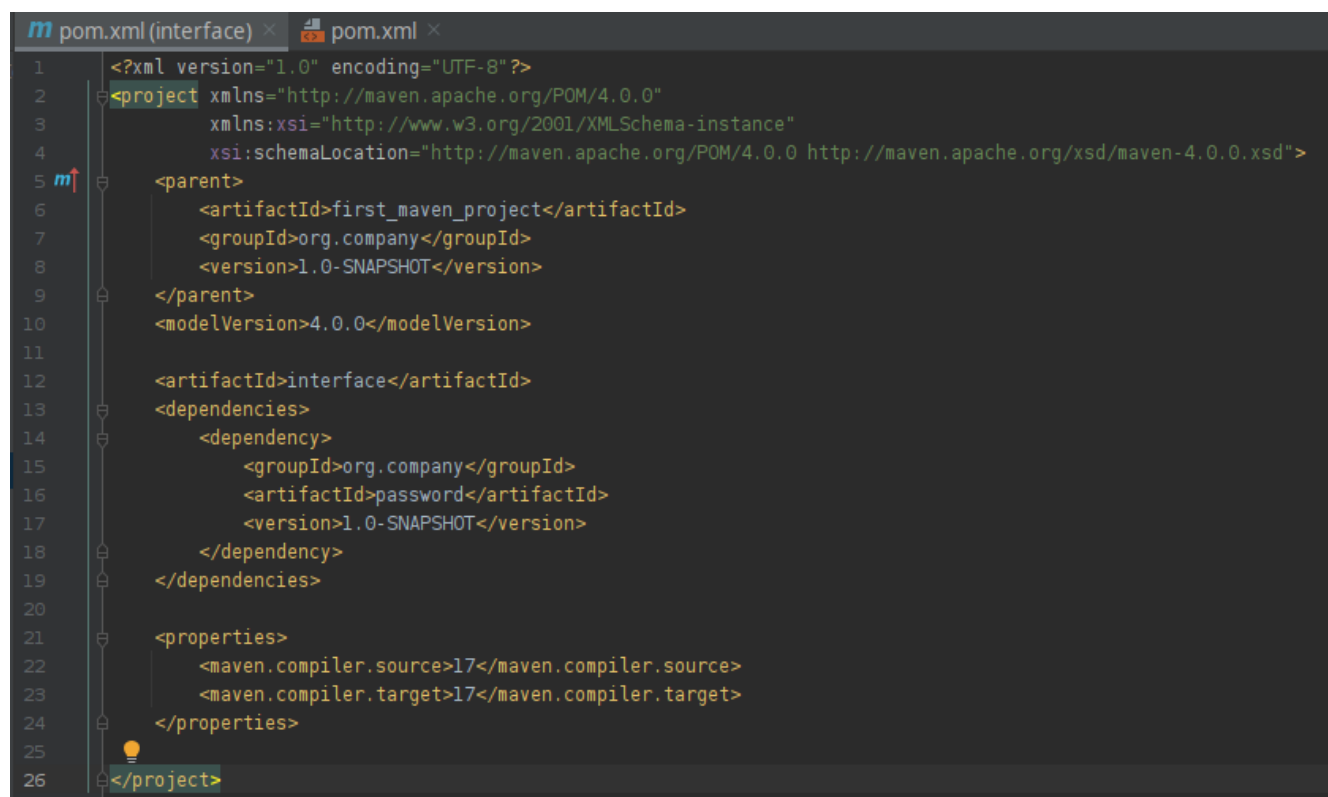


Рисунок 7: Выбирается зависимый модуль по artifactId.

Теперь файл pom.xml модуля «interface» хранить информацию о зависимости модуля «password».



```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5     <parent>
6         <artifactId>first_maven_project</artifactId>
7         <groupId>org.company</groupId>
8         <version>1.0-SNAPSHOT</version>
9     </parent>
10    <modelVersion>4.0.0</modelVersion>
11
12    <artifactId>interface</artifactId>
13    <dependencies>
14        <dependency>
15            <groupId>org.company</groupId>
16            <artifactId>password</artifactId>
17            <version>1.0-SNAPSHOT</version>
18        </dependency>
19    </dependencies>
20
21    <properties>
22        <maven.compiler.source>17</maven.compiler.source>
23        <maven.compiler.target>17</maven.compiler.target>
24    </properties>
25
26 </project>
```

Рисунок 8: pom.xml модуля interface имеет зависимость модуля password.