ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»
(СПбГУТ)

Институт магистратуры

Кафедра информационных управляющих систем

Дисциплина «Аспектно-ориентированное программирование»

Отчет

к лабораторной работе № 2

Выполнил:

Студент группы ИСТ-012м

Тарасенко А.А.

Принял:

Ст. преподаватель Антонов В.В.

Санкт-Петербург

2020

Задание: изучить фреймворк Maven и Gradle, а также их фазы жизненного цикла.

Ход работы:

Общее описание

Maven как и Gradle является системой автоматической сборки. Но Maven описывает структуру проектов на языке POM (Project Object Model) в XML-подобном формате; в то время как Gradle использует языки Groovy и Kotlin. В результате файл со структурой проекта, генерируемы Gradle получается гораздо короче и лаконичнее.

Жизненный цикл Gradle

В Gradle всё строится на двух базовых концепциях: на проектах (projects) и задачах (tasks).

Каждая сборка Gradle состоит из одного или более проектов.

Каждый проект состоит из одной или более задач. Задача – атомарная часть работы, выполняемой в сборке. Задачи могут требовать выполнения других задач – `dependsOn taskName [, task2Name, …]`.

В документации Gradle описаны три основные фазы: Initialization, Configuration и Execution – каждая из которых может быть дополнена по желанию самим разработчиком.

Инициализация – определяются проекты, которые будут участвовать в сборке.

Конфигурация – настройка объектов проекта; сборка скриптов всех проектов, являющихся частью запущенной сборки.

Исполнение – определяет список задач, созданных и настроенных на предыдущем этапе, которые должны быть выполнены; затем Gradle выполняют каждую из этих задач.

`settings.gradle`

```
println 'This is executed during the initialization phase.'
```

build.gradle

```
println 'This is executed during the configuration phase.'

task configured {
    println 'This is also executed during the configuration
phase.'
}

task testOne {
    doLast {
        println 'This is executed during the execution phase.'
    }
}

task testBoth {
     doFirst {
       println 'This is executed first during the execution
phase.'
     }
     doLast {
       println 'This is executed last during the execution
phase.'
     }
     println 'This is executed during the configuration phase as
well.'
}
```

При сборке будет следующий вывод:

```
> gradle testOne testBoth
This is executed during the initialization phase.

> Configure project :
This is executed during the configuration phase.
```

```
This is also executed during the configuration phase.
This is executed during the configuration phase as well.

> Task :testOne
This is executed during the execution phase.

> Task :testBoth
This is executed first during the execution phase.
This is executed last during the execution phase.

BUILD SUCCESSFUL in 0s
2 actionable tasks: 2 executed
```

На этапе инициализации запускается файл настроек (по умолчанию «settings.gradle»).

Таким образом код, помещенный в раздел `doFirst`, будет вызван на стадии инициализации; код в doLast будет выполнен на стадии исполнения; а остальной код будет выполнен на этапе конфигурации (кроме кода файла настроек, который выполняется на этапе инициализации).

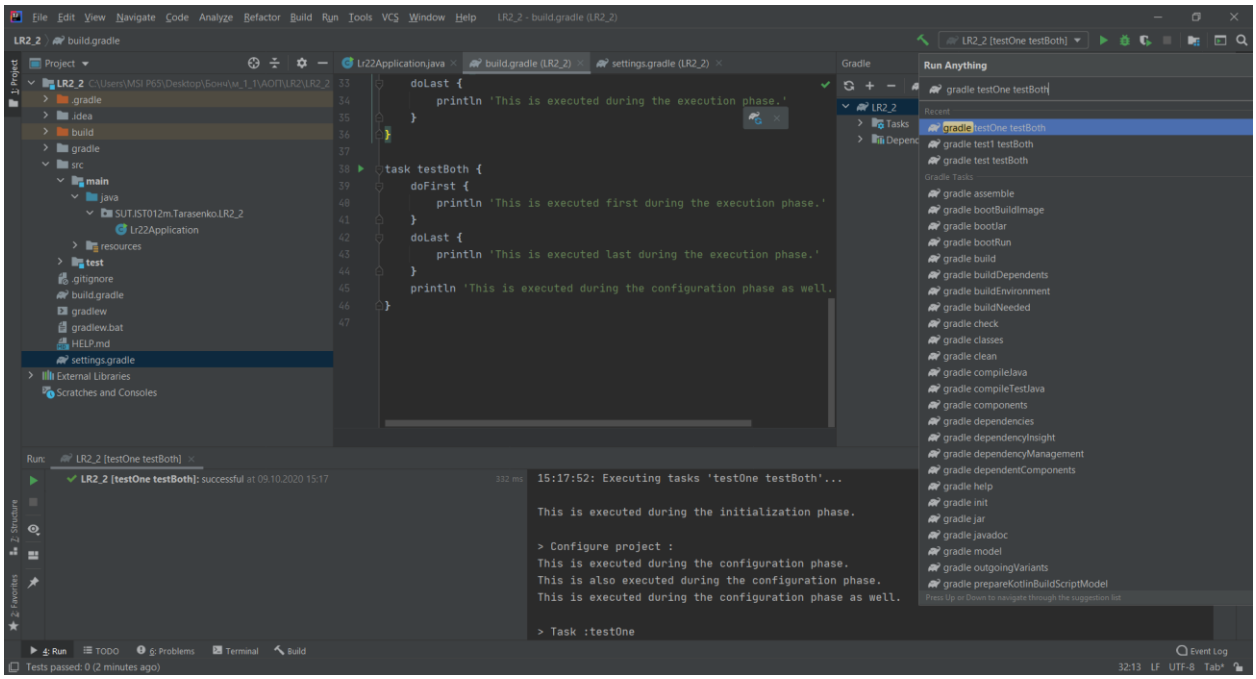

Рисунок 1 – пример запуска задач

Жизненный цикл Maven

У Maven есть три основных жизненных цикла: default, clean и site.
default – основной жизненный цикл, так как ответственен за развертывание проекта.
clean – очищает проект, удаляя файлы, сгененрированные при предыдущих сборках.
site – создаёт документацию проекта.

Каждый жизненный цикл состоит из набора фаз: default из 23-х, clean из 3-х, а site из 4-х.
Фазы выполняются в определённом порядке, это значит, что если запустить конкретную фазу, то выполнятся и все предшествующие ей.

У каждой фазы есть свой набор целей, который проверяется при выполнении фазы. Для просмотра полного списка целей, связанных с конкретной фазой, можно воспользоваться командой:
`mvn help:describe -Dcmd=PHASENAME`
Также в Maven существуют плагины – группы целей, однако они могут быть не привязаны к одной фазе.

```
<build>
    <plugins>
        <plugin>
            <artifactId>maven-failsafe-plugin</artifactId>
            <version>${maven.failsafe.version}</version>
            <executions>
                <execution>
                    <goals>
                        <goal>integration-test</goal>
                        <goal>verify</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
    </plugins>
</build>
```

Как видно, сам плагин привязан к фазе build и состоит из двух целей: запуск интеграционных тестов и проверка результатов тестов.

Всего существует два типа плагинов: плагины сборки – размещаются внутри блока build; и плагины отчётов – размещаются внутри блока reporting.

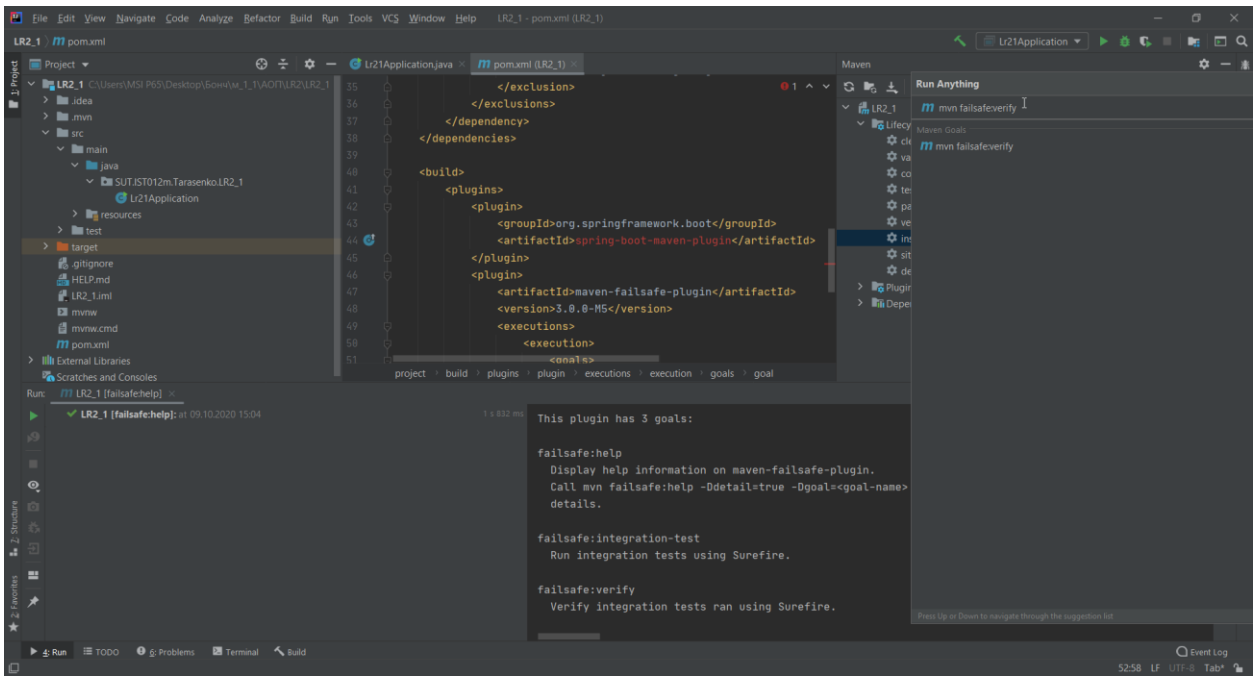

Рисунок 2 – пример выполнения определенной цели в плагине.

Ссылки

https://docs.gradle.org/current/userguide/build_lifecycle.html

https://www.baeldung.com/maven-goals-phases

https://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html#Lifecycle_Reference