Отчет по лабораторной работе 1

Подготовил: Костин Р., ИПБ-21

Часть 1

# Постановка задачи

В процессе выполнения лабораторной работы необходимо выполнить следующие задачи:

1. Создать исполняемый java класс, который выводит текстовую строку на экран.
2. Разработать два класса. Первый класс выводит строку из второго класса.
3. Создать jar-файл, хранящий элементы из предыдущего задания.

# Выполнение

## Структура проекта

*Структура 1. Структура проекта первого задания*

|-- src

| `--mypkg

| `-- FirstClass.java

`-- classes

`-- mypkg

`-- FirstClass.class

*Структура 2. Структура проекта второго задания*

|-- classes

|

|

|

|

|

|-- mypkg

|

|

`-- FirstClass.class

`-- mypkg2

`-- MyText.class

`-- src

|-- mypkg

| `-- FirstClass.java

|

`-- mypkg2

`-- MyText.java

*Структура 3. Структура проекта третьего задания*

|-- lab1.jar

|-- minifest.mf

|-- classes

|

|

|

|

|

|-- mypkg

|

|

`-- FirstClass.class

`-- mypkg2

`-- MyText.class

`-- src

|-- mypkg

| `-- FirstClass.java

|

`-- mypkg2

`-- MyText.java

## Задание 1

Сначала были созданы необходимые директории с помощью командной строки. Для этого нам понадобятся основные команды: mkdir (создание новой директории), cd <желаемый каталог> и cd .. (для смены каталога).

*Листинг 1. Создание директорий*

mkdir src mkdir classes mkdir mypkg

Далее был создан исполняемый класс FirstClass.

*Листинг 2. Листинг класса FirstClass*

package mypkg;

public class FirstClass{

public static void main(String args[]){ System.out.println("FirstClass");

}

}

Для компиляции файла в командной строке использовалась команда javac. Набирали соответствующую команду находясь в главной директории (в ней как раз располагались папки src и classes). Скомпилированный файл должен был лежать в папке с путем classes/mypkg.

*Листинг 3. Команда javac*

javac -classpath ./classes -d ./classes src/mypkg/FirstClass.java

Для запуска файла в командной строке использовалась команда java. Как и в прошлый раз, набирали команду находясь в главной директории.

*Листинг 4. Команда java*

java -classpath ./classes mypkg.FirstClass

При запуске скомпилированного файла в командной строке выводится строка "FirstClass".

## Задание 2

Наравне с уже созданными директориями mypkg была добавлена еще одна.

*Листинг 5. Создание директории*

mkdir mypkg2

В созданной директории был создан дополнительный класс MyText, хранящий текстовую строку "MyText". Для вывода нового сообщения класс FirstClass был изменен.

*Листинг 6. Листинг класса MyText*

package mypkg2; public class MyText{

public static String msg = "MyText";

}

*Листинг 7. Листинг класса FirstClass*

package mypkg; import mypkg2.MyText;

public class FirstClass{

public static void main(String args[]){ System.out.println(MyText.msg);

}

}

Далее с помощью команды javac данные файлы были скомпилированы. Файл FirstClass был запущен командой java. При запуске программы в командной строке выводится строка "MyText".

## Задание 3

Был создан файл manifest.mf, в котором указан главный исполняемый класс FirstClass. Обязательно в конце нового файла была оставлена пустая строка.

*Листинг 8. Листинг Manifest.mf*

Manifest-Version: 1.0

Created-By: 1.6.0\_19 (Sun Microsystems Inc.) Main-Class: mypkg.FirstClass

После этого была выполнена команда по сборке архива jar и запуск jar-файла.

*Листинг 9. Команда по сборке архива*

jar cvmf manifest.mf lab1.jar -C ./classes mypkg -C ./classes mypkg2

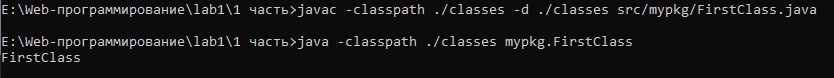
*Листинг 10. Команда запуска jar-файла*

java -jar lab1.jar

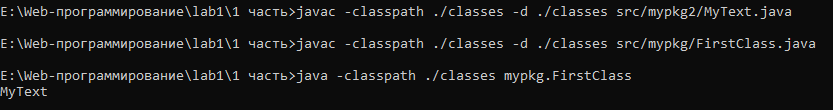
При запуске архива в командной строке выводится строка "MyText".

# Результаты выполнения

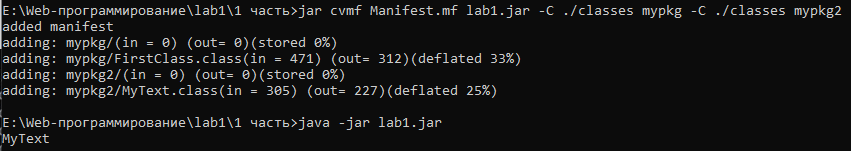
В результате выполнения первого задания был получен класс FirstClass. При запуске выводится строка:



В результате выполнения второго задания были получены классы MyText и FirstClass. При запуске FirstClass выводится строка:



В результате выполнения третьего задания был получен jar-файл lab1.jar. При запуске jar-файла выводится строка:



# Вывод

В результате выполнения лабораторной работы (часть 1) я узнал, как из командной строки осуществлять: компиляцию классов, запуск проектов, создание jar-файл.

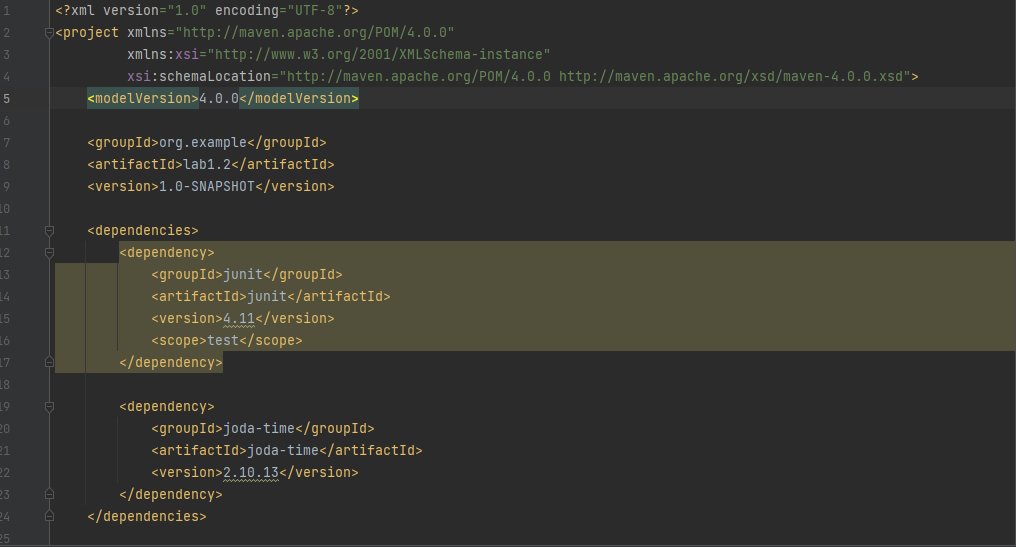
Часть 2

# Постановка цели и задачи

В процессе выполнения лабораторной работы необходимо сделать проект с помощью maven. Добавить в проект зависимость и продемонстрировать ее работу.

# Выполнение

При создании проекта проводим идентификацию и добавляем зависимости: JUnit и joda-time.



Далее добавляем класс App, который будет еще служить и точкой входа в нашу программу.

*Листинг 1. Листинг класса App*

package org.example;

import org.joda.time.LocalDateTime;

public class App {

public static void main(String[] args) {

LocalDateTime now = new LocalDateTime();

System.out.println("Current time: " + now);

}

}

После добавляем модульный тест AppTest.

*Листинг 2. Листинг модульного теста AppTest*

package org.example;

import static org.junit.Assert.assertTrue;

import org.junit.Test;

public class AppTest {

@Test

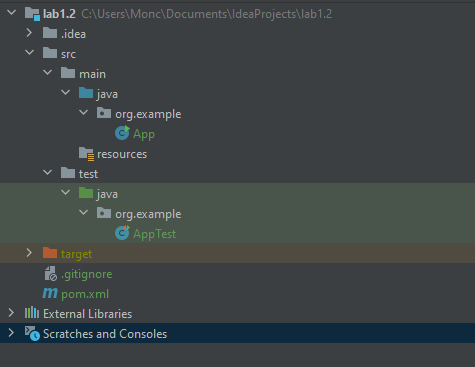
public void shouldAnswerWithTrue(){

assertTrue(true);

}

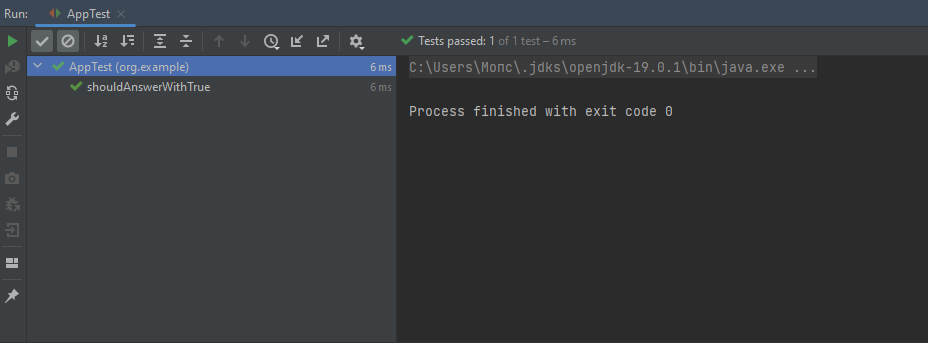
}

Структура нашего проекта теперь выглядит так:



# Результаты выполнения

Далее собираем наш проект и запускаем тест.



Тест успешно пройден.

# Вывод

В результате выполнения лабораторной работы (часть 2) я узнал, как создавать проекты с помощью maven: подключать зависимости и работать с модульными тестами.