Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра «ЭВМиС»

Отчёт по лабораторной работе №1

“Сборка Java-проекта с использованием Maven и Gradle. Создание и запуск первого Java-приложения”

Выполнил:

Студент 3-го курса

Группы МС-7

Мороз Е. В.

Проверил:

Николаюк-Ртищева М. В.

Брест, 2023

**Цель работы:** ознакомиться с основными принципами автоматической сборки проектов с использованием систем Maven и Gradle на примере простейшего проекта Java.

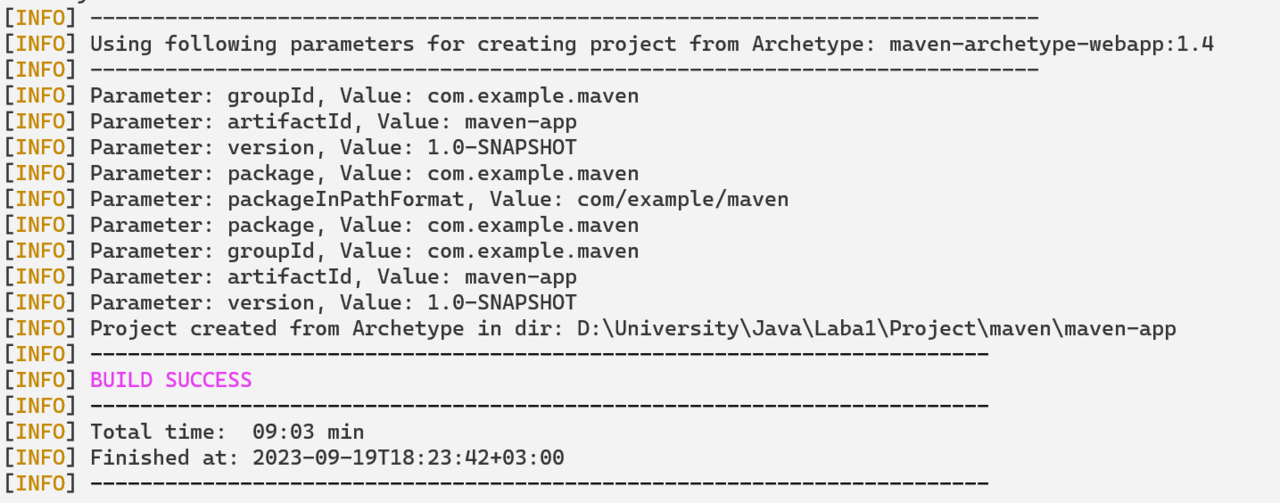
**Ход работы**

**1. Сборка Java-проекта с помощью Maven**

После установки Maven проверяется правильность установки, запустив в командной строке *mvn -v*. Результат выполнения команды:



Создается web-проект с помощью команды *mvn archetype:generate* с архетипом *org.apache.maven.archetypes:maven-archetype-webapp*. Результат выполнения команды:



В папку с созданной структурой добавляется главный java-файл Main.java (путь до файла: src/main/java). В Main.java вписан следующий код:

public class Main {

public static void main(String[] args) {

System.out.println("Hello, Java!");

}

}

В процессе генерации проекта создается файл pom.xml. Он содержит модель объекта проекта. Файл POM хранит информацию о проекте и различных деталях конфигурации, которые используются Maven для создания проекта. В файле также содержаться настройки, которые были указаны во время создания проекта:

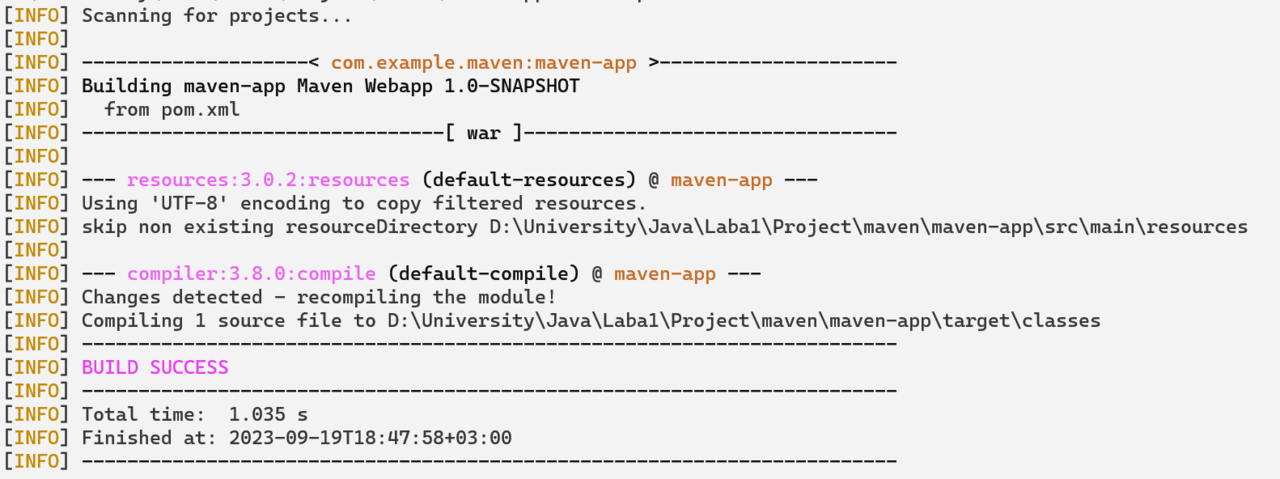
<groupId>com.example.maven</groupId>

<artifactId>maven-app</artifactId>

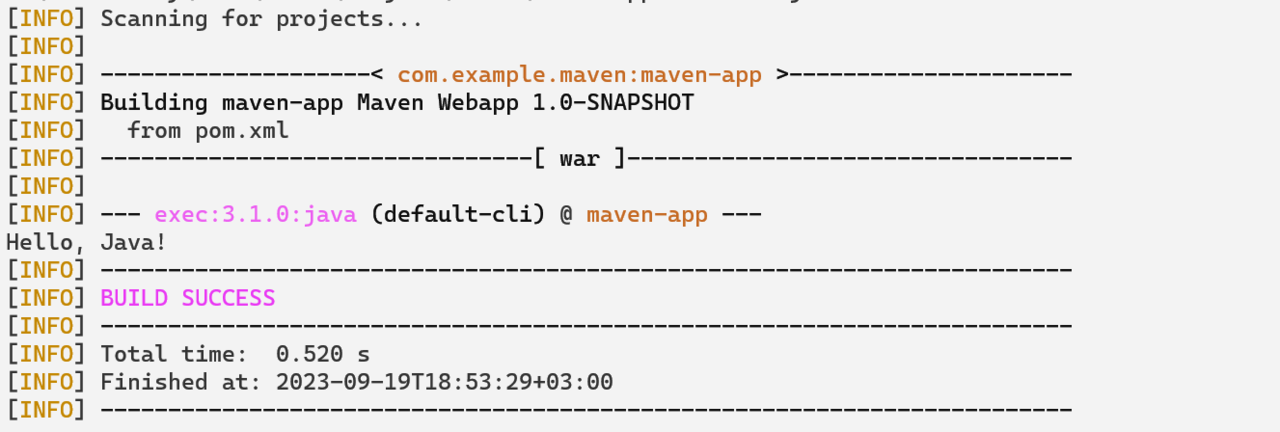
<version>1.0-SNAPSHOT</version>

<packaging>war</packaging>

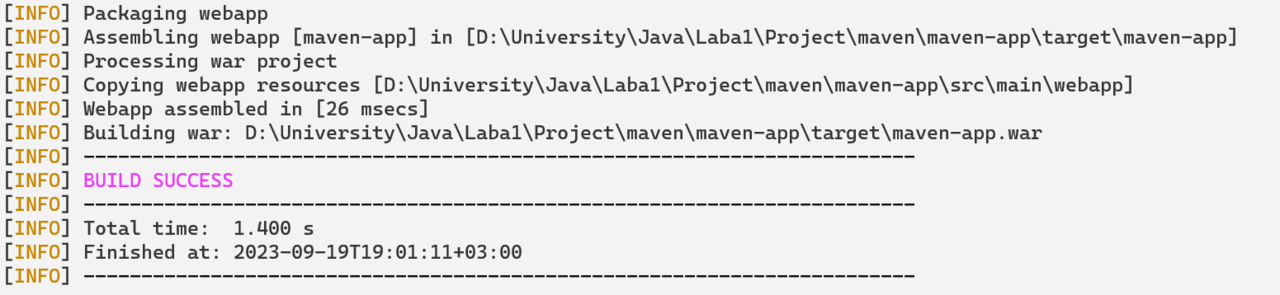
Сборка проекта происходит с помощью команды *mvn compile*. Результат выполнения команды:



Запуск проекта осуществляется с помощью команды *mvn exec:java -Dexec.mainClass="Main".* Результат выполнения команды:

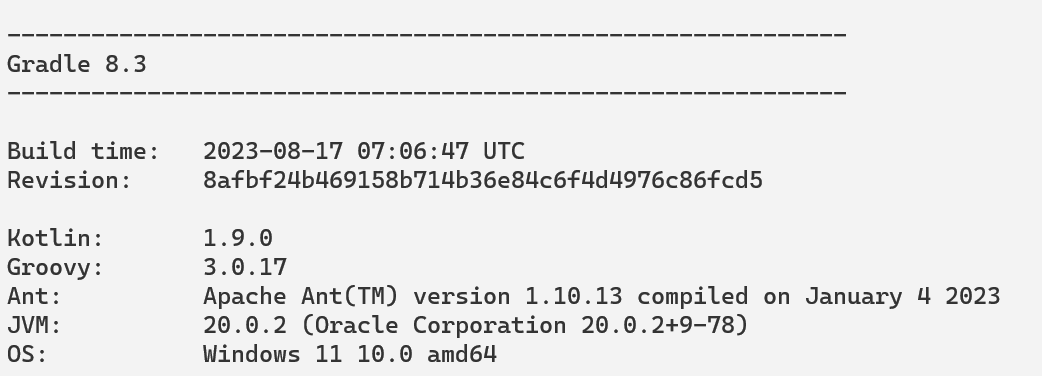


Упаковка скомпилированных классов в формат .jar с помощью команды *mvn package*. Эта команда проверяет корректность метаинформации о проекте, компилирует исходники, прогоняет тесты классов из предыдущего шага и упаковывает скомпилированные классы. Результат выполнения команды:



**2. Сборка Java-проекта с помощью Gradle**

После установки Gradle проверяется правильность установки, запустив в командной строке *gradle -v*. Результат выполнения команды:



Создается структура каталогов в корневой папке:

└── src

└── main

└── java

└── hello

В директории hello создается два класса: HelloWorld.java и Greeter.java. В HelloWorld добавляется внешняя библиотека для получения текущего времени и даты Joda Time. Содержимое файлов:

src/main/java/hello/HelloWorld.java

package hello;

import org.joda.time.LocalTime;

public class HelloWorld {

public static void main(String[] args) {

LocalTime currentTime = new LocalTime();

System.out.println("The current local time is: " + currentTime);

Greeter greeter = new Greeter();

System.out.println(greeter.sayHello());

}

}

src/main/java/hello/Greeter.java

package hello;

public class Greeter {

public String sayHello() {

return "Hello, world!";

}

}

В корневой папке проекта создается файл конфигурации build.gradle, который содержит компилируемую зависимость Joda Time, источники сторонних библиотек и имя JAR-файла. Содержимое файла:

apply plugin: 'java'

apply plugin: 'application'

dependencies {

implementation 'joda-time:joda-time:2.9.6'

}

repositories {

mavenLocal()

mavenCentral()

}

jar {

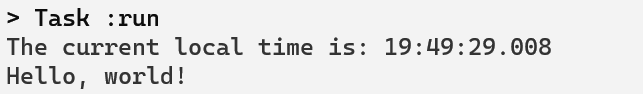
archivesBaseName = 'gs-gradle'

version = '0.1.0'

}

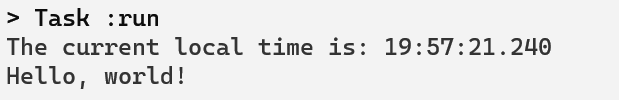
mainClassName = 'hello.HelloWorld'

Сборка и запуск программы с помощью команд *gradle build* и *gradle run* соответвенно. Результат выполнения команд:



После выполнения сборки, в главной директории проекта появляется папка *build*, которая содержит все собранные и скомпилированные данные, классы и библиотеки.

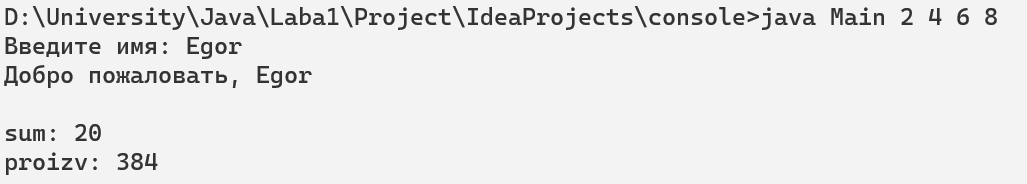
Для загрузки и инициализации wrapper-скриптов запускаем команду *gradle wrapper*. Gradle Wrapper позволяет не устанавливать и не настраивать Gradle определенной версии, чтобы собрать и запустить проект. Для выполнения задачи сборки необходимо выполнить команду *gradlew build*, а для выполнения и запуска проекта требуется запуск команды *gradlew run*. Результат выполнения команд:



**3. Индивидуальное задание**

Условие задания: приветствовать любого пользователя при вводе его имени через командную строку. Ввести целые числа как аргументы командной строки, подсчитать их суммы и произведения. Вывести результат на консоль.

Результат выполнения программы:



Код программы:

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

System.out.print("Введите имя: ");

Scanner in = new Scanner(System.in);

String name = in.nextLine();

System.out.println("Добро пожаловать, " + name);

int sum = 0;

int proizv = 1;

for(String i : args) {

sum += Integer.parseInt(i);

proizv \*= Integer.parseInt(i);

}

System.out.println("sum: " + sum);

System.out.println("proizv: " + proizv);

}

}

**Вывод:** в результате выполнения данной лабораторной работы, мною были освоены методы сборки java-проектом при помощи таких систем сборки: Maven и Gradle. Помимо этого, была написана простая программа с использование простых конструкций языка Java