Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

ПРАКТИЧНА РОБОТА № 1

З дисципліни «Інженерія програмного забезпечення»

на тему «Підготовка програмного проекту. Збірка проекту за допомогою Ant, Maven та Gradle»

Виконав:
студент групи ІО-42
Куліков М. М.
Залікова: 4214

Перевірив:
Ст. викладач кафедри ОТ
Васильєва М. Д.

Київ – 2025

Практична робота №1

Тема: «Підготовка програмного проекту. Збірка проекту за допомогою Ant, Maven та Gradle».

Мета: Отримання базових навичок з використання мови XML. Вивчення структури типового програмного проекту, форматів стандартних файлів опису проекту. Вивчення формату JAR. Здобуття навичок з використання засобів автоматизації процесу збірки програмних проектів на мові Java - Apache ANT (Another Neat Tool), Maven, Gradle.

Виконання роботи:

Завдання 1

1. Завдання до варіанту 2:
   a. Створити в каталозі out jar-архів з усіх файлів проекту з розширеннями java, js, html, htm. Скопіювати архів в корінь проекту.

Код програми:

Вміст файлу build.xml:

```
<project name="Template" default="signJAR" basedir=".">
<!-- ================== Application Properties ================== -->
  <property name="app.name" value="PR1" />
  <property name="app.version" value="0.1" />
  <property name="app.title" value="PR1" />
  <property name="app.author" value="Maksym Kulikov" />
  <property name="app.company" value="By Broniev" />
  <property name="sourceDir" value="src" />
  <property name="buildDir" value="out" />
  <property name="outputDir" value="${buildDir}/production/template" />
  <property name="resourceDir" value="src/res" />
  <property name="libDir" value="lib" />
<!-- ================== JAR Properties ================== -->
    <property name="jar.mainClass" value="com.lab111.TestMain" />
    <property name="jar.name" value="${app.name}.jar" />
    <property name="jar.keyStore" value="${basedir}/out/tempKey.store" />
    <property name="jar.keyPass" value="telpat" />
    <property name="jar.keyAlias" value="tempAlias" />
<!--  ================== Compilation Control Options ================== -->
   <property name="compile.debug"       value="true"/>
   <property name="compile.deprecation" value="false"/>
   <property name="compile.optimize"    value="true"/>
   <path id="compile.classpath">
     <fileset dir="${libDir}">
       <include name="*.jar"/>
    </fileset>
   </path>
```

```
<!-- ====================== Clean Target ================================== -->
   <target name="clean"
         description="Clean build dirs">
     <delete dir="${buildDir}" />
   </target>
<!-- ====================== Prepare Target ================================ -->
   <target name="prepare"
         depends="clean"
         description="Prepare build dirs">
     <mkdir dir="${buildDir}"/>
     <mkdir dir="${buildDir}/output"/>
     <mkdir dir="${buildDir}/web-apps"/>
   </target>
<!-- ====================== Compile Target ================================ -->
   <target name="compile"
         depends="prepare"
         description="Compile Java sources">
     <javac srcdir="${sourceDir}"
         destdir="${buildDir}/output"
           debug="${compile.debug}"
      deprecation="${compile.deprecation}"
        optimize="${compile.optimize}"
          target="1.8"
          source="1.8"
      includeantruntime="false">
       <classpath refid="compile.classpath"/>
     </javac>
     <copy  todir="${buildDir}/output">
       <fileset dir="${sourceDir}" excludes="**/*.java"/>
     </copy>
   </target>
<!-- ====================== Create JAR archive Target
================================== -->
   <target name="createJAR"
         depends="compile"
         description="Create JAR archive" >
      <jar destfile="${buildDir}/${jar.name}" basedir="${outputDir}">
           <manifest>
             <attribute name="Created-By" value="${app.author} - (${app.company})"/>
             <attribute name="Built-By" value="${user.name}"/>
             <attribute name="Main-Class" value="${jar.mainClass}"/>
             <section name="${app.name}">
               <attribute name="Specification-Title" value="${app.title}"/>
               <attribute name="Specification-Version" value="${app.version}"/>
               <attribute name="Specification-Vendor" value="${app.company}"/>
               <attribute name="Implementation-Title" value="${app.name}"/>
               <attribute name="Implementation-Version" value="${app.version}"/>
               <attribute name="Implementation-Vendor" value="${app.company}"/>
             </section>
           </manifest>
      </jar>
   </target>
<!-- ====================== Generate Key for JAR signing Target
================================== -->
  <target name="generateKey"
        description="Generates Key for JAR signing">
    <delete failonerror="false" file="${jar.keyStore}"/>
    <genkey keystore="${jar.keyStore}" alias="${jar.keyAlias}" storepass="${jar.keyPass}"
validity="720" keyalg="RSA">
```

```
      <dname>
        <param name="CN" value="${app.company}"/>
        <param name="OU" value="${app.title}"/>
        <param name="O"  value="${app.company}"/>
      </dname>
    </genkey>
  </target>
<!-- ================== FixStyle Target ================================== -->
    <target name="fixstyle"
          description="Fix Style in source code" >
      <fixcrlf srcdir="${basedir}"
            tab="remove"
            tablength="2"
            includesfile="fixstyle.list"
      />
   </target>
<!-- ================== JAR Singing Target ================================
-->
  <target name="signJAR"
        depends="createJAR,generateKey"
        description="Signing JAR archive">
      <exec dir="${buildDir}" executable="jarsigner">
        <arg line="-keystore ${jar.keyStore} -storepass ${jar.keyPass} ${jar.name} ${jar.keyAlias}"/>
      </exec>
      <delete file="${buildDir}/myKeystore"/>
  </target>
<!-- ================== Make zip Target ================================== -->
    <target name="make-zip-project"
          description="Zip all project tree from basedir">
      <tstamp/>
      <zip destfile="../${app.name}-${DSTAMP}-${TSTAMP}.zip"
         basedir="${basedir}" excludes="out/**"/>
    </target>
<!-- ================== generate doc by javadoc Target
================================== -->
      <target name="generate_javadoc"
        description="Zip all project tree from basedir">
            <javadoc sourcepath="${sourceDir}"
          destdir="doc" author="yes"
          version="yes" access="private">
            </javadoc>
      </target>
<!-- = 2 Variant = -->
  <target name="make-all-jar"
        description="Create JAR with all project files">
      <mkdir dir="${basedir}/out"/>
      <jar destfile="${basedir}/out/${app.name}-all.jar">
        <fileset dir="${basedir}"
              includes="**/*.java,**/*.js,**/*.html,**/*.htm"/>
      </jar>
      <copy file="${basedir}/out/${app.name}-all.jar"
          todir="${basedir}/"/>
  </target>
</project>
```

Результат виконання програми:

```
PS D:\Documents\Education\KPI\3 Term\IPZ\Projects\PR1\template> ant -f build.xml make-all-jar
Buildfile: D:\Documents\Education\KPI\3 Term\IPZ\Projects\PR1\template\build.xml

make-all-jar:
      [jar] Building jar: D:\Documents\Education\KPI\3 Term\IPZ\Projects\PR1\template\out\PR1-all.jar
     [copy] Copying 1 file to D:\Documents\Education\KPI\3 Term\IPZ\Projects\PR1\template

BUILD SUCCESSFUL
```

Рисунок 1 – результат виконання програми

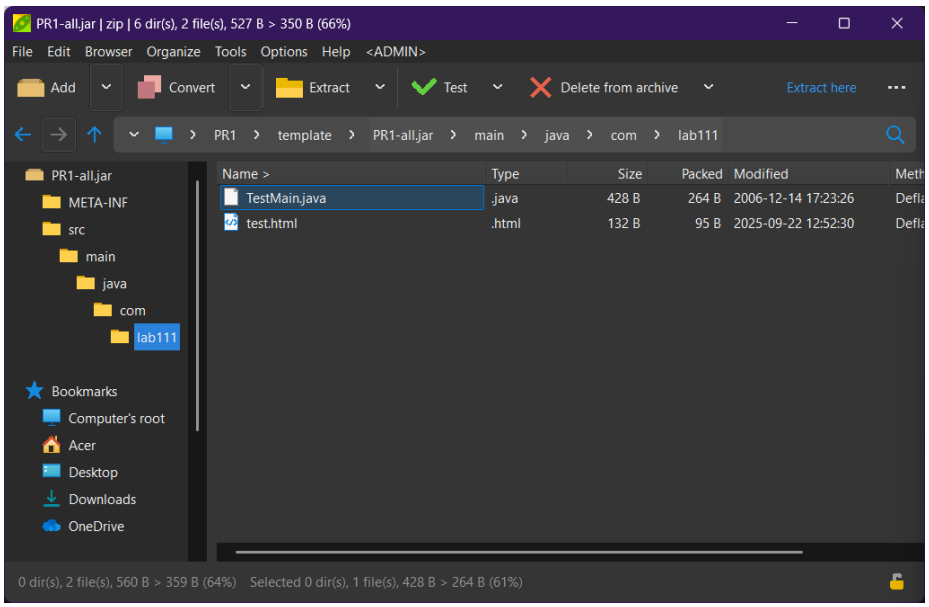

Рисунок 2 – файли .java та .html в створеному архіві

Завдання 2

1. Створіть новий Maven (або Gradle) проєкт в середовищі розробки IntelliJIDEA.
2. Для формування JSON використовуйте сторонню бібліотеку з Maven Central, наприклад, GSON. Додайте залежність для GSON у файл pom.xml (або build.gradle).
3. Налаштуйте проєкт так, щоб команда *mvn package* (*gradle jar* для gradle) створювала виконуваний FatJar файл з назвою lab1.jar.
4. При запуску команди java -jar lab1.jar, проєкт повинен виводити в консоль JSON об’єкт із вашим ім’ям, прізвищем та номером групи у форматі: *{“name”: “Maria”, “lastName”: “Vasylieva”, “group”: “IO-123”}.*

Код програми:

Вміст файлу pom.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         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>org.example</groupId>
    <artifactId>MavenBuild</artifactId>
    <version>A1</version>

    <properties>
        <maven.compiler.source>11</maven.compiler.source>
        <maven.compiler.target>11</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <dependency>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
            <version>2.13.0</version>
        </dependency>
    </dependencies>

    <build>
        <finalName>Lab1</finalName>
        <plugins>
            <!-- Maven JAR Plugin для створення JAR -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-jar-plugin</artifactId>
                <version>3.2.2</version>
                <configuration>
                    <archive>
                        <manifest>
                            <addClasspath>true</addClasspath>
                            <classpathPrefix>libs/</classpathPrefix>
                            <mainClass>org.example.Main</mainClass>
                        </manifest>
                    </archive>
                </configuration>
            </plugin>

            <!-- Maven Assembly Plugin для створення fat JAR -->
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-assembly-plugin</artifactId>
                <version>3.1.1</version>
                <configuration>
                    <descriptorRefs>
                        <descriptorRef>jar-with-dependencies</descriptorRef>
                    </descriptorRefs>
                    <archive>
                        <manifest>
```

```
                    <mainClass>org.example.Main</mainClass>
                </manifest>
            </archive>
            <appendAssemblyId>false</appendAssemblyId>
        </configuration>
        <executions>
            <execution>
                <id>make-assembly</id>
                <phase>package</phase>
                <goals>
                    <goal>single</goal>
                </goals>
            </execution>
        </executions>
    </plugin>
  </plugins>
 </build>
</project>
```

Вміст файлу Main.java:

```
package org.example;

import com.google.gson.Gson;

public class Main {
    public static void main(String[] args) {
        Person person = new Person("Maksym", "Kulikov", "IO-42");
        String json = new Gson().toJson(person);
        System.out.println(json);
    }
}
```

Вміст файлу Person.java:

```
package org.example;

public class Person {
    public String name;
    public String lastName;
    public String group;

    public Person() {}
    public Person(String name, String lastName, String group) {
        this.name = name;
        this.lastName = lastName;
        this.group = group;
    }
}
```

Результат виконання програми:

```
PS D:\Documents\Education\KPI\3 Term\IPZ\Projects\MavenBuild> cd target
PS D:\Documents\Education\KPI\3 Term\IPZ\Projects\MavenBuild\target> java -jar Lab1.jar
{"name":"Maksym","lastName":"Kulikov","group":"IO-42"}
```

Рисунок 3 – результат виконання програми

Висновки:

Отже, під час виконання практичної роботи було освоєно базові навички роботи з XML файлами практичним шляхом, а також було використано засоби автоматизації процесу збірку ПЗ на мові Java Apache ANT та Maven. У результаті, було створено дві програми, одна з яких створює .jar архів з усіма файлами проекту з розширеннями java, js, html, htm, а також програму, що виводить прізвище і ім’я з групою. Проблем під час виконання практичної роботи не виникало.