Университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки 09.03.04 Программная инженерия

Дисциплина «Методы и средства программной инженерии»

**Отчет**

По лабораторной работе №3

Вариант 44533

Выполнили:

*Барабанщиков А.Д.*

*Кочнев Р.Д.*

Преподаватель:

*Абузов Я.А.*

Санкт-Петербург, 2023 г.

Задание:

Написать сценарий для утилиты Apache Ant, реализующий компиляцию, тестирование и упаковку в jar-архив кода проекта из лабораторной работы №3 по дисциплине "Веб-программирование".

Каждый этап должен быть выделен в отдельный блок сценария; все переменные и константы, используемые в сценарии, должны быть вынесены в отдельный файл параметров; MANIFEST.MF должен содержать информацию о версии и о запускаемом классе.

**Сценарий должен реализовывать следующие цели (targets):**

* **compile** -- компиляция исходных кодов проекта.
* **build** -- компиляция исходных кодов проекта и их упаковка в исполняемый jar-архив. Компиляцию исходных кодов реализовать посредством вызова цели **compile**.
* **clean** -- удаление скомпилированных классов проекта и всех временных файлов (если они есть).
* **test** -- запуск junit-тестов проекта. Перед запуском тестов необходимо осуществить сборку проекта (цель **build**).
* **music** - воспроизведение музыки по завершению сборки (цель **build**).
* **team** - осуществляет получение из git-репозитория 4 предыдущих ревизий, их сборку (по аналогии с основной) и упаковку получившихся jar-файлов в zip-архив. Сборку реализовать посредством вызова цели **build**.

Код программы

<?xml version="1.0" encoding="UTF-8"?>

<project name="lab3" default="build">

<property file="build.properties"/>

<path id="classpath">

<fileset dir="${lib.dir}" includes="\*.jar"/>

<fileset dir="${main.dir}" includes="\*.java"/>

</path>

<path id="classpath.test">

<pathelement location="${junit}"/>

<pathelement location="${hamcrest}"/>

<pathelement location="${classes.dir}"/>

</path>

<target name="compile" depends="clean">

<echo message="\*\*\*\*\* COMPILING STARTED \*\*\*\*\*"/>

<mkdir dir="${classes.dir}"/>

<mkdir dir="${test.classes.dir}"/>

<javac srcdir="${src.dir}" destdir="${classes.dir}" classpathref="classpath"/>

<echo message="\*\*\*\*\* COMPILING COMPLETED \*\*\*\*\*"/>

</target>

<target name="build" depends="compile">

<echo message="\*\*\*\*\* BUILDING STARTED \*\*\*\*\*"/>

<copy todir="${build.dir}">

<fileset dir="${web.dir}"/>

</copy>

<copy todir="${build.dir.lib}">

<fileset dir="${lib.dir}"/>

</copy>

<copy todir="${classes.dir}">

<fileset dir="${resources.dir}"/>

</copy>

<jar destfile="${build.dir}/${ant.project.name}.jar">

<fileset dir="${classes.dir}"/>

<manifest>

<attribute name="Created-By" value="Voldemort"/>

<attribute name="Manifest-Version" value="1.0"/>

<attribute name="Main-Class" value="Main"/>

</manifest>

</jar>

<war destfile="${build.dir}/${ant.project.name}.war" webxml="${build.webxml}">

<fileset dir="${build.dir}"/>

<manifest>

<attribute name="Created-By" value="Voldemort"/>

<attribute name="Manifest-Version" value="1.0"/>

<attribute name="Main-Class" value="NoClass"/>

</manifest>

</war>

<echo message="\*\*\*\*\* BUILDING COMPLETED \*\*\*\*\*"/>

</target>

<target name="clean">

<echo message="\*\*\*\*\* CLEANING STARTED \*\*\*\*\*"/>

<delete dir="${build.dir}"/>

<delete dir="${junit.report.dir}"/>

<delete dir="${doc.dir}"/>

<echo message="\*\*\*\*\* CLEANING COMPLETED \*\*\*\*\*"/>

</target>

<target name="test" depends="build">

<echo message="\*\*\*\*\* TESTING STARTED \*\*\*\*\*"/>

<mkdir dir="${test.classes.dir}"/>

<mkdir dir="${junit.report.dir}"/>

<javac destdir="${test.classes.dir}" srcdir="${test.dir}" includeantruntime="false" encoding="utf-8">

<classpath refid="classpath.test"/>

</javac>

<junit printsummary="on" haltonfailure="true" haltonerror="true">

<classpath>

<path refid="classpath.test"/>

<pathelement location="${test.classes.dir}"/>

</classpath>

<batchtest fork="yes" todir="${junit.report.dir}">

<formatter type="xml"/>

<fileset dir="${test.dir}" includes="\*Test.java"/>

</batchtest>

</junit>

<echo message="\*\*\*\*\* TESTING COMPLETED \*\*\*\*\*"/>

</target>

<target name="scp" depends="build">

<echo message="\*\*\*\*\* SCP STARTED \*\*\*\*\*"/>

<exec executable="scp" failonerror="true">

<arg value="-P"/>

<arg value="${scp.port}"/>

<arg value="${build.dir}/${ant.project.name}.war"/>

<arg value="${scp.user}@${scp.host}:${scp.dir}"/>

</exec>

<echo message="\*\*\*\*\* SCP COMPLETED \*\*\*\*\*"/>

</target>

<target name="music" depends="build">

<echo message="--- MUSIC DONE ---"/>

<exec executable="cmd">

<arg value="/c"/>

<arg value="${musicPlayer} ${music}"/>

<arg value="-p"/>

</exec>

</target>

<target name="team">

<mkdir dir="team-builds"/>

<exec executable="cmd" outputproperty="orig-head">

<arg value="/c"/>

<arg value="git rev-parse HEAD"/>

</exec>

<antcall target="build-revision">

<param name="link" value="HEAD"/>

</antcall>

<antcall target="build-revision">

<param name="link" value="HEAD~1"/>

</antcall>

<antcall target="build-revision">

<param name="link" value="HEAD~1"/>

</antcall>

<antcall target="build-revision">

<param name="link" value="HEAD~1"/>

</antcall>

<exec executable="cmd">

<arg value="/c"/>

<arg value="git checkout ${orig-head}"/>

</exec>

<zip destfile="team-builds.zip">

<fileset dir="team-builds"/>

</zip>

<delete dir="team-builds"/>

</target>

<target name="build-revision">

<property name="link" value="${link}"/>

<exec executable="cmd" outputproperty="revision">

<arg value="/c"/>

<arg value="git rev-parse ${link}"/>

</exec>

<exec executable="cmd">

<arg value="/c"/>

<arg value="git checkout ${revision}"/>

</exec>

<antcall target="build"/>

<copy todir="team-builds/${revision}">

<fileset dir="${build.dir}" includes="\*.jar"/>

<fileset dir="${build.dir}" includes="\*.war"/>

</copy>

</target>

<target name="env">

<mkdir dir="${classes.dir}"/>

<javac srcdir="${main.dir}" destdir="${classes.dir}" classpathref="classpath" source="${compile.version}"

includeantruntime="false">

<compilerarg line="${vm.args}"/>

</javac>

<copy todir="${classes.dir}">

<fileset dir="${resources.dir}"/>

</copy>

<antcall target="build"/>

<exec executable="scp" failonerror="true">

<arg value="-P"/>

<arg value="${scp.port}"/>

<arg value="${build.dir}/${ant.project.name}.war"/>

<arg value="${scp.user}@${scp.host}:${scp.dir}"/>

</exec>

</target>

</project>

Выводы

Во время выполнения лабораторной работы мы изучили системы сборки приложений, написали собственные сценарии сборки с помощью утилиты Apache Ant. Также мы написали модульные тесты на основе библиотеки JUnit.