Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО» Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №3 по дисциплине «Основы программной инженерии» Вариант № 248310

Студенты:

Сиразетдинов А.Н., Шпинева У. С.

Группа: Р3216

Преподаватель:

Письмак А.Е.

г. Санкт-Петербург 2023/24

Задание	3
Сценарий для утилиты Ant	4
Полный код	
Вывод	

Задание

Написать сценарий для утилиты Apache Ant, реализующий компиляцию, тестирование и упаковку в јаг-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):

- 1. **compile** -- компиляция исходных кодов проекта.
- 2. **build** -- компиляция исходных кодов проекта и их упаковка в исполняемый jar-архив. Компиляцию исходных кодов реализовать посредством вызова цели **compile**.
- 3. **clean** -- удаление скомпилированных классов проекта и всех временных файлов (если они есть).
- 4. **test** -- запуск junit-тестов проекта. Перед запуском тестов необходимо осуществить сборку проекта (цель **build**).
- 5. **music** воспроизведение музыки по завершению сборки (цель **build**).
- 6. **native2ascii** преобразование native2ascii для копий файлов локализации (для тестирования сценария все строковые параметры необходимо вынести из классов в файлы локализации).
- 7. **alt** создаёт альтернативную версию программы с измененными именами переменных и классов (используя задание replace/replaceregexp в файлах параметров) и упаковывает её в jar-архив. Для создания jar-архива использует цель **build**.
- 8. **report** в случае успешного прохождения тестов сохраняет отчет junit в формате xml, добавляет его в репозиторий svn и выполняет commit.

Сценарий для утилиты Ant

```
basedir="."
        default="music"
        xsi:schemaLocation="antlib:org.apache.ivy.ant"
        xmlns:if="ant:if"
       xmlns:unless="ant:unless">
   cproperty file="build.properties"/>
   <path id="classpathRuntime">
       <fileset dir="${lib}" includes="*.jar"/>
   </path>
   <path id="test.classpath">
       <pathelement path="dist"/>
       <pathelement path="${build.classes}"/>
       <pathelement path="${build.tests}"/>
       <fileset dir="${ant.home}/lib" includes="*.jar" />
   </path>
   <target name="init">
       <tstamp/>
       <mkdir dir="${build}"/>
       <mkdir dir="${build.classes}"/>
      <mkdir dir="${build.tests}"/>
      <mkdir dir="${alt}"/>
      <mkdir dir="${altbuild}"/>
       <mkdir dir="${altbuild.classes}"/>
       <mkdir dir="${dist.lib}"/>
   </target>
   <target name="download-ivy" unless="skip.download">
       <mkdir dir="${ivy.jar.dir}"/>
src="https://repo1.maven.org/maven2/org/apache/ivy/ivy/${ivy.insta
usetimestamp="true"/>
   </target>
   <target name="install-ivy" depends="download-ivy"</pre>
description="--> install ivy">
       <path id="ivy.lib.path">
```

```
</path>
uri="antlib:org.apache.ivy.ant" classpathref="ivy.lib.path"/>
  </target>
   <target name="resolve dependencies" depends="install-ivy"</pre>
description="--> resolve dependencies">
       <ivy:retrieve log="quiet"/>
   </target>
   <target name="compile" depends="init, resolve dependencies"</pre>
description="--> compile the source">
       <javac includeantruntime="false"</pre>
              srcdir="${src}"
              destdir="${build.classes}">
           <classpath refid="classpathRuntime"/>
   </target>
   <target name="compile-tests" depends="compile" description="-->
       <javac includeantruntime="false"</pre>
              srcdir="${src.test}"
              destdir="${build.tests}">
           <classpath >
               <path refid="classpathRuntime"/>
               <path path="${src.test}"/>
               <pathelement location="${build.classes}"/>
           </classpath>
       </javac>
   </target>
   <target name="build" depends="compile" description="-->
building war">
       <ivy:retrieve</pre>
pattern="${dist.lib}/[artifact]-[revision].[ext]" conf="runtime"/>
webxml="${src.webapp.webxml}">
           <fileset dir="${src.webapp}">
           </fileset>
           <lib dir="${dist.lib}"/>
           <classes dir="${build.classes}"/>
           <manifest>
               <section name="Common">
                    <attribute name="Project-Name"
value="${project.name}"/>
```

```
<attribute name="Project-Version"</pre>
value="${project.version}"/>
               </section>
           </manifest>
       </war>
   </target>
   <target name="music" description="--> play music">
       <splash showduration="0" imageurl="${splash-gif}"/>
           <success source="${resource.music.success}"/>
       <antcall target="build"/>
   </target>
   <target name="native2ascii" description="Преобразование
       <native2ascii src="${src.main.resources}"</pre>
                     dest="${native2ascii.resources}"
   </target>
   <target name="alt" description=" создаёт альтернативную версию
       <copy todir="${alt}">
           <fileset dir="${src}"/>
       </copy>
       <replaceregexp match="${alt.regex}"</pre>
                       flags="${alt.regex.flags}"
                       replace="${alt.replace}">
       </replaceregexp>
       <antcall target="build">
           <param name="src" value="${alt}"/>
           <param name="src.main" value="${alt.main}"/>
           <param name="src.main.resources"</pre>
value="${alt.main.resources}"/>
           <param name="src.webapp" value="${alt.webapp}"/>
           <param name="src.webapp.webxml"</pre>
value="${alt.webapp.webxml}"/>
           <param name="src.test" value="${alt.test}"/>
           <param name="build" value="${altbuild}"/>
           <param name="build.classes"</pre>
value="${altbuild.classes}"/>
```

```
<param name="build.tests" value="${altbuild.tests}"/>
           <param name="dist.war.name"</pre>
value="${dist}/${ant.project.name}-alt.war"/>
       </antcall>
   </target>
   <target name="test" depends="build, compile-tests"</pre>
description="--> run tests">
       <mkdir dir="${dist.test}"/>
       <junitlauncher printsummary="true">
           <classpath refid="test.classpath"/>
           <testclasses outputdir="${dist.test}">
               <fileset dir="${build.tests}">
                    <include name="**/*.class"/>
               </fileset>
               <listener type="legacy-xml"</pre>
                          sendSysOut="true"
                          sendSysErr="true"/>
               <fork/>
           </testclasses>
       </junitlauncher>
   </target>
   <target name="report" depends="test" description="--> commit
       <delete dir="${dist.report}"/>
       <mkdir dir="${dist.report}"/>
       <junitreport todir="${dist.report}">
           <fileset dir="${dist.test}">
               <include name="TEST-*.xml"/>
           </fileset>
           <report format="frames" todir="${dist.report}"/>
       </junitreport>
       <exec dir="${svn.dir}" executable="svn"</pre>
if:true="${svn.use}">
           <arg line="add *"/>
       </exec>
       <exec dir="${svn.dir}" executable="svn"</pre>
if:true="${svn.use}">
           <arg line="commit -m ${svn.commit.name}</pre>
       </exec>
  </target>
   <target name="clean"</pre>
           description="clean up">
```

Полный код

https://github.com/Azat2202/OPI_lab_3.git

Вывод

В ходе выполнения лабораторной работе мы узнали про работу с системой сборки Ant и узнали про тестирование программ с помощью junit. Написали скрипт сборки проекта и базовые тесты.