**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №3

Вариант №51501

по дисциплине «**Основы программной инженерии**»

Автор: Баянов Равиль Динарович

Факультет: ПИиКТ

Группа: P3234

Преподаватель: Кулинич Ярослав Вадимович



Санкт-Петербург, 2024

Оглавление

[Задание 3](#_Toc165569823)

[Вариант 3](#_Toc165569824)

[Сценарий сборки проекта на утилите Ant 4](#_Toc165569825)

[Полный код проекта 5](#_Toc165569826)

[Список используемой литературы 6](#_Toc165569827)

[Вывод 7](#_Toc165569828)

# Задание

Написать сценарий для утилиты [Apache Ant](http://ant.apache.org/), реализующий компиляцию, тестирование и упаковку в jar-архив кода проекта из [лабораторной работы №3](https://se.ifmo.ru/courses/web#lab3) 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.

# Вариант

**Cценарий 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. **doc** - добавление в MANIFEST.MF MD5 и SHA-1 файлов проекта, а также генерация и добавление в архив javadoc по всем классам проекта.
6. **diff** - осуществляет проверку состояния рабочей копии, и, если изменения не касаются классов, указанных в файле параметров выполняет commit в репозиторий git.

# Сценарий сборки проекта на утилите Ant

|  |
| --- |
| 1. <**project** name="web3lab" 2. basedir="." 3. xmlns:ivy="antlib:org.apache.ivy.ant" 4. xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>" 5. xsi:schemaLocation="antlib:org.apache.ivy.ant"> 6. <**property** file="build.properties"/> 8. <**path** id="classpathRuntime"> 9. <**fileset** dir="${lib}" includes="\*.jar"/> 10. </**path**> 12. <**path** id="test.classpath" > 13. <**path** refid="classpathRuntime"/> 14. <**fileset** dir="${lib}" includes="\*\*/\*.jar"/> 15. <**pathelement** location="${build.tests}"/> 16. </**path**> 18. <**target** name="init"> 19. <**tstamp**/> 20. <**mkdir** dir="${build}"/> 21. <**mkdir** dir="${build.classes}"/> 22. <**mkdir** dir="${build.tests}"/> 23. <**mkdir** dir="${build.doc}"/> 24. <**mkdir** dir="${dist}"/> 25. <**mkdir** dir="${dist.lib}"/> 26. </**target**> 28. <**target** name="download-ivy" unless="skip.download"> 29. <**mkdir** dir="${ivy.jar.dir}"/> 30. <**get** src="<https://repo1.maven.org/maven2/org/apache/ivy/ivy/>${ivy.install.version}/ivy-${ivy.install.version}.jar" dest="${ivy.jar.file}" usetimestamp="true"/> 31. </**target**> 33. <**target** name="install-ivy" depends="download-ivy" description="(\*) install ivy"> 34. <**path** id="ivy.lib.path"> 35. <**fileset** dir="${ivy.jar.dir}" includes="\*.jar"/> 36. </**path**> 37. <**taskdef** resource="org/apache/ivy/ant/antlib.xml" uri="antlib:org.apache.ivy.ant" classpathref="ivy.lib.path"/> 38. </**target**> 40. <**target** name="resolve" depends="install-ivy" description="(\*) resolve dependencies"> 41. <**ivy:retrieve** log="quiet"/> 42. </**target**> 44. <**target** name="compile" depends="init, resolve" description="(\*) compile the source"> 45. <**javac** includeantruntime="false" 46. srcdir="${src}" 47. destdir="${build.classes}" 48. classpathref="classpathRuntime"/> 49. </**target**> 51. <**target** name="compile-tests" depends="compile" description="(\*) compile tests"> 52. <**javac** includeantruntime="false" 53. srcdir="${src.test}" 54. destdir="${build.tests}"> 55. <**classpath**> 56. <**path** refid="classpathRuntime"/> 57. <**path** path="${src.test}"/> 58. <**pathelement** location="${build.classes}"/> 59. </**classpath**> 60. </**javac**> 61. </**target**> 63. <**target** name="build" depends="compile" description="(\*) build war and jar file"> 64. <**ivy:retrieve** pattern="${dist.lib}/[artifact]-[revision](-[classifier]).[ext]" conf="compile,runtime"/> 65. <**war** destfile="${dist.war.name}" webxml="${src.webapp.webxml}"> 66. <**fileset** dir="${src.webapp}"> 67. <**include** name="\*\*/\*.\*"/> 68. </**fileset**> 69. <**lib** dir="${dist.lib}"/> 70. <**classes** dir="${build.classes}"/> 71. <**manifest**> 72. <**section** name="Main"> 73. <**attribute** name="Project-Name" value="${project.name}"/> 74. <**attribute** name="Project-Version" value="${project.version}"/> 75. </**section**> 76. </**manifest**> 77. </**war**> 78. <**jar** destfile="${dist.jar.name}" basedir="${build.classes}"> 79. <**manifest**> 80. <**section** name="Main"> 81. <**attribute** name="Project-Name" value="${project.name}"/> 82. <**attribute** name="Project-Version" value="${project.version}"/> 83. </**section**> 84. </**manifest**> 85. </**jar**> 86. </**target**> 88. <**target** name="test" depends="build, compile-tests" description="(\*) run tests"> 89. <**mkdir** dir="${dist.test}"/> 90. <**junitlauncher** printsummary="true"> 91. <**classpath**> 92. <**path** refid="test.classpath"/> 93. <**pathelement** location="${build.classes}"/> 94. <**pathelement** location="${build.tests}"/> 95. </**classpath**> 96. <**testclasses** outputdir="${build.tests}"> 97. <**fileset** dir="${build.tests}" includes="\*\*/\*.class" /> 98. <**listener** type="legacy-xml" sendSysOut="true" sendSysErr="true"/> 99. </**testclasses**> 100. </**junitlauncher**> 101. </**target**> 103. <**target** name="doc" depends="build" description="(\*) adding MD5 and SHA-1 project 104. files to MANIFEST.MF and generate javadoc"> 105. <**checksum** algorithm="MD5" file="${dist.jar.name}" property="md5"/> 106. <**checksum** algorithm="SHA-1" file="${dist.jar.name}" property="sha1"/> 107. <**jar** destfile="${dist.jar.name}" basedir="${build.classes}" update="true"> 108. <**manifest**> 109. <**attribute** name="Built-By" value="${user.name}"/> 110. <**attribute** name="Build-Date" value="${DSTAMP}"/> 111. <**attribute** name="MD5-Checksum" value="${md5}"/> 112. <**attribute** name="SHA1-Checksum" value="${sha1}"/> 113. </**manifest**> 114. </**jar**> 115. <**checksum** algorithm="MD5" file="${dist.war.name}" property="md5"/> 116. <**checksum** algorithm="SHA-1" file="${dist.war.name}" property="sha1"/> 117. <**war** destfile="${dist.war.name}" basedir="${build.classes}" update="true"> 118. <**manifest**> 119. <**attribute** name="Built-By" value="${user.name}"/> 120. <**attribute** name="Build-Date" value="${DSTAMP}"/> 121. <**attribute** name="MD5-Checksum" value="${md5}"/> 122. <**attribute** name="SHA1-Checksum" value="${sha1}"/> 123. </**manifest**> 124. </**war**> 125. <**javadoc** version="true" 126. author="true" 127. use="true" 128. destdir="${build.doc}" 129. sourcepath="${src.main}" 130. packagenames="\*" 131. classpath="${build.classes}" 132. classpathref="classpathRuntime" 133. windowtitle="${project.name} API"> 134. <**fileset** dir="${src.main}"> 135. <**include** name="\*\*/\*.java"/> 136. </**fileset**> 137. </**javadoc**> 138. </**target**> 140. <**target** name="diff" depends="compile" description="(\*) check work copy and commit in git repo"> 141. <**exec** executable="git" failonerror="true" outputproperty="git.status"> 142. <**arg** value="status"/> 143. <**arg** value="--porcelain"/> 144. </**exec**> 145. <**condition** property="modifications.allowed"> 146. <**not**> 147. <**contains** string="${git.status}" substring="M"/> 148. </**not**> 149. </**condition**> 150. <**condition** property="ignored.exist"> 151. <**contains** string="${git.status}" substring="${ignored.classes}"/> 152. </**condition**> 153. <**fail** if="modifications.allowed" message="No Modifications detected in classes"/> 154. <**fail** if="ignored.exist" message="Modifications detected in ignored classes: ${git.status}"/> 155. <**exec** executable="git" failonerror="true"> 156. <**arg** value="add"/> 157. <**arg** value="."/> 158. </**exec**> 159. <**exec** executable="git" failonerror="true"> 160. <**arg** value="commit"/> 161. <**arg** value="-m"/> 162. <**arg** value="Automated commit by Ant"/> 163. </**exec**> 164. </**target**> 166. <**target** name="clean" description="(\*) clean up"> 167. <**delete** dir="${build}"/> 168. <**delete** dir="${dist.lib}"/> 169. <**delete** dir="${dist}"/> 170. </**target**> 171. </**project**> |

# Полный код проекта

<https://github.com/RavvChek/lab3OPI>

# Список используемой литературы

* Клименков С. В., Цопа Е. А., Козин И. О. Конспект лекций по дисциплине “Основы программной инженерии”, 2021. Электронный ресурс / Режим доступа: <https://se.ifmo.ru/documents/10180/671657/%D0%9B%D0%B5%D0%BA%D1%86%D0%B8%D0%B8+%D0%BF%D0%BE+%D0%9E%D0%9F%D0%98+v1.3.3.pdf/50ce1e06-00d9-4900-be9c-a3316a746d6d>
* Хабр. Apache Ant – быстрый старт. Электронный ресурс / Режим доступа: <https://habr.com/ru/articles/323204/>
* Java-online.ru. Сборка Java приложения, ant. Электронный ресурс / Режим доступа: <https://java-online.ru/blog-ant.xhtml>

# Вывод

Во время выполнения данной лабораторной работы я узнал про работу системы сборки проектов Ant и написал сценарий сборки на ней. Также узнал более подробно про тестирование программ с помощью junit. Написал базовые тесты для своего проекта по веб-программированию.