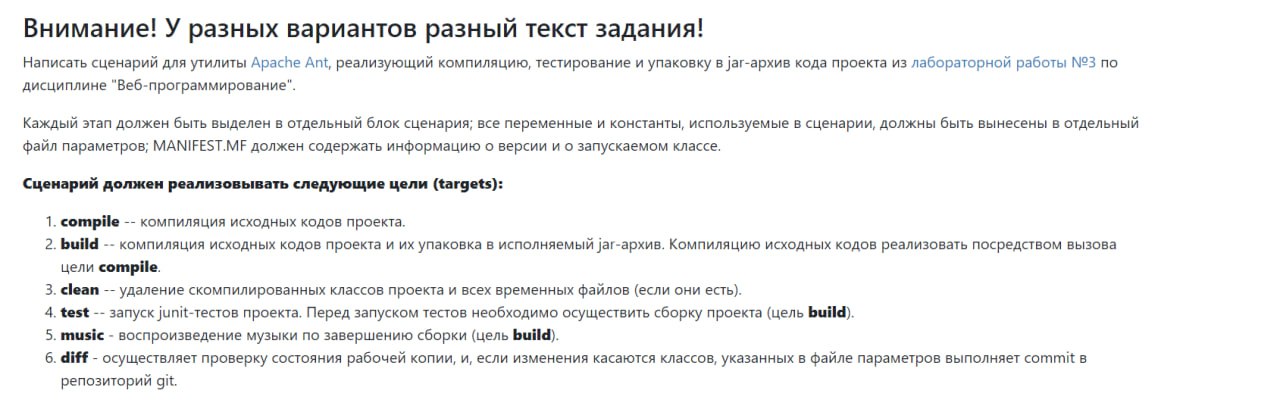
Задание:

Лабораторная работа №3



Выполнение:

Репозиторий с кодом: ---------------------------------------------

Файл – build.xml

<?xml version="1.0" encoding="UTF-8"?>  
<project name="lab3\_web" default="diff" basedir=".">  
 <!--  
 Добавить все переменные из файла  
 -->  
 <property file="build.properties"/>  
  
 <!--  
 Создать набор ссылок на внешние библиотеки, необходимые для компиляции  
 программы  
 -->  
 <path id="build.classpath">  
 <pathelement  
 location="${repo.local}/jakarta/platform/jakarta.jakartaee-web-api/9.1.0/jakarta.jakartaee-web-api-9.1.0.jar"/>  
 <pathelement  
 location="${repo.local}/jakarta/faces/jakarta.faces-api/3.0.0/jakarta.faces-api-3.0.0.jar"/>  
 <pathelement  
 location="${repo.local}/jakarta/servlet/jakarta.servlet-api/5.0.0/jakarta.servlet-api-5.0.0.jar"/>  
 <pathelement  
 location="${repo.local}/org/projectlombok/lombok/1.18.24/lombok-1.18.24.jar"/>  
 <pathelement  
 location="${repo.local}/org/postgresql/postgresql/42.5.1/postgresql-42.5.1.jar"/>  
 </path>  
 <!--  
 Создать набор ссылок на внешние библиотеки, необходимые для компиляции  
 тестов  
 -->  
 <path id="build.test.classpath">  
 <pathelement  
 location="${repo.local}/jakarta/platform/jakarta.jakartaee-web-api/9.1.0/jakarta.jakartaee-web-api-9.1.0.jar"/>  
 <pathelement  
 location="${repo.local}/jakarta/faces/jakarta.faces-api/3.0.0/jakarta.faces-api-3.0.0.jar"/>  
 <pathelement  
 location="${repo.local}/jakarta/servlet/jakarta.servlet-api/5.0.0/jakarta.servlet-api-5.0.0.jar"/>  
 <pathelement location="${repo.local}/junit/junit/4.12/junit-4.12.jar"/>  
 <pathelement  
 location="${repo.local}/org/hamcrest/hamcrest-core/1.3/hamcrest-core-1.3.jar"/>  
 <pathelement  
 location="${repo.local}/org/junit/jupiter/junit-jupiter-api/5.9.1/junit-jupiter-api-5.9.1.jar"/>  
 <pathelement  
 location="${repo.local}/org/opentest4j/opentest4j/1.2.0/opentest4j-1.2.0.jar"/>  
 <pathelement  
 location="${repo.local}/org/junit/platform/junit-platform-commons/1.9.1/junit-platform-commons-1.9.1.jar"/>  
 <pathelement  
 location="${repo.local}/org/apiguardian/apiguardian-api/1.1.2/apiguardian-api-1.1.2.jar"/>  
 <pathelement  
 location="${repo.local}/org/junit/jupiter/junit-jupiter-engine/5.9.1/junit-jupiter-engine-5.9.1.jar"/>  
 <pathelement  
 location="${repo.local}/org/junit/platform/junit-platform-engine/1.9.1/junit-platform-engine-1.9.1.jar"/>  
 <pathelement  
 location="${repo.local}/org/projectlombok/lombok/1.18.24/lombok-1.18.24.jar"/>  
 <pathelement  
 location="${repo.local}/org/postgresql/postgresql/42.5.1/postgresql-42.5.1.jar"/>  
 <pathelement  
 location="${repo.local}/org/checkerframework/checker-qual/3.5.0/checker-qual-3.5.0.jar"/>  
 <pathelement  
 location="${repo.local}/org/testng/testng/7.8.0/testng-7.8.0.jar"/>  
 <pathelement  
 location="${repo.local}/org/slf4j/slf4j-api/1.7.36/slf4j-api-1.7.36.jar"/>  
 <pathelement  
 location="${repo.local}/com/beust/jcommander/1.82/jcommander-1.82.jar"/>  
 <pathelement  
 location="${repo.local}/org/webjars/jquery/3.6.1/jquery-3.6.1.jar"/>  
 </path>  
  
 <!--  
 Добавить расширение в ApacheAnt чтобы использовать дополнительные команды  
 такие как <if>, <length>, <then>, <else>  
 -->  
 <taskdef resource="net/sf/antcontrib/antlib.xml">  
 <classpath>  
 <pathelement location="${path.lib}/ant-contrib-1.0b3.jar"/>  
 </classpath>  
 </taskdef>  
  
 <!--  
 Удалить результат прошлой сборкия  
 -->  
 <target name="clear">  
 <delete dir="${build.dir}"/>  
 </target>  
  
 <!--  
 Скачать из сети необходимые библиотеки для выполнения всего кода:  
 -->  
 <target name="get-deps" depends="clear"  
 description="Download all dependencies"  
 unless="maven.mode.offline">  
 <!--\* Создать папку для загрузки-->  
 <mkdir dir="${repo.local}"/>  
  
 <!--\* Создать папку с определенной библиотекой-->  
 <mkdir dir="${repo.local}/jakarta/platform/jakarta.jakartaee-web-api/9.1.0"/>  
 <!--\* Скачать .jar файл и поместить его с созданную папку-->  
 <get src="https://repo.maven.apache.org/maven2/jakarta/platform/jakarta.jakartaee-web-api/9.1.0/jakarta.jakartaee-web-api-9.1.0.jar"  
 dest="${repo.local}/jakarta/platform/jakarta.jakartaee-web-api/9.1.0/jakarta.jakartaee-web-api-9.1.0.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/jakarta/faces/jakarta.faces-api/3.0.0"/>  
 <get src="https://repo.maven.apache.org/maven2/jakarta/faces/jakarta.faces-api/3.0.0/jakarta.faces-api-3.0.0.jar"  
 dest="${repo.local}/jakarta/faces/jakarta.faces-api/3.0.0/jakarta.faces-api-3.0.0.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/jakarta/servlet/jakarta.servlet-api/5.0.0"/>  
 <get src="https://repo.maven.apache.org/maven2/jakarta/servlet/jakarta.servlet-api/5.0.0/jakarta.servlet-api-5.0.0.jar"  
 dest="${repo.local}/jakarta/servlet/jakarta.servlet-api/5.0.0/jakarta.servlet-api-5.0.0.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/junit/jupiter/junit-jupiter-api/5.9.1"/>  
 <get src="https://repo.maven.apache.org/maven2/org/junit/jupiter/junit-jupiter-api/5.9.1/junit-jupiter-api-5.9.1.jar"  
 dest="${repo.local}/org/junit/jupiter/junit-jupiter-api/5.9.1/junit-jupiter-api-5.9.1.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/opentest4j/opentest4j/1.2.0"/>  
 <get src="https://repo.maven.apache.org/maven2/org/opentest4j/opentest4j/1.2.0/opentest4j-1.2.0.jar"  
 dest="${repo.local}/org/opentest4j/opentest4j/1.2.0/opentest4j-1.2.0.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/junit/platform/junit-platform-commons/1.9.1"/>  
 <get src="https://repo.maven.apache.org/maven2/org/junit/platform/junit-platform-commons/1.9.1/junit-platform-commons-1.9.1.jar"  
 dest="${repo.local}/org/junit/platform/junit-platform-commons/1.9.1/junit-platform-commons-1.9.1.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/apiguardian/apiguardian-api/1.1.2"/>  
 <get src="https://repo.maven.apache.org/maven2/org/apiguardian/apiguardian-api/1.1.2/apiguardian-api-1.1.2.jar"  
 dest="${repo.local}/org/apiguardian/apiguardian-api/1.1.2/apiguardian-api-1.1.2.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/junit/jupiter/junit-jupiter-engine/5.9.1"/>  
 <get src="https://repo.maven.apache.org/maven2/org/junit/jupiter/junit-jupiter-engine/5.9.1/junit-jupiter-engine-5.9.1.jar"  
 dest="${repo.local}/org/junit/jupiter/junit-jupiter-engine/5.9.1/junit-jupiter-engine-5.9.1.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/junit/platform/junit-platform-engine/1.9.1"/>  
 <get src="https://repo.maven.apache.org/maven2/org/junit/platform/junit-platform-engine/1.9.1/junit-platform-engine-1.9.1.jar"  
 dest="${repo.local}/org/junit/platform/junit-platform-engine/1.9.1/junit-platform-engine-1.9.1.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/projectlombok/lombok/1.18.24"/>  
 <get src="https://repo.maven.apache.org/maven2/org/projectlombok/lombok/1.18.24/lombok-1.18.24.jar"  
 dest="${repo.local}/org/projectlombok/lombok/1.18.24/lombok-1.18.24.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/postgresql/postgresql/42.5.1"/>  
 <get src="https://repo.maven.apache.org/maven2/org/postgresql/postgresql/42.5.1/postgresql-42.5.1.jar"  
 dest="${repo.local}/org/postgresql/postgresql/42.5.1/postgresql-42.5.1.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 <mkdir dir="${repo.local}/org/checkerframework/checker-qual/3.5.0"/>  
 <get src="https://repo.maven.apache.org/maven2/org/checkerframework/checker-qual/3.5.0/checker-qual-3.5.0.jar"  
 dest="${repo.local}/org/checkerframework/checker-qual/3.5.0/checker-qual-3.5.0.jar"  
 usetimestamp="false"  
 ignoreerrors="true"/>  
 </target>  
  
 <!--  
 Скомпилировать весь код:  
 -->  
 <target name="compile" depends="get-deps" description="Compile the code">  
 <!--  
 \* Создать папку для результата компиляции кода проекта  
 -->  
 <mkdir dir="${build.outputDir}"/>  
 <!--  
 \* Скомпилировать весь код проекта в созданную папку  
 -->  
 <javac destdir="${build.outputDir}">  
 <!--  
 Что необходимо скомпилировать  
 -->  
 <src>  
 <pathelement location="${build.srcDir.0}"/>  
 </src>  
 <!--  
 Добавить ссылки на библиотеки  
 -->  
 <classpath refid="build.classpath"/>  
 </javac>  
  
 <!--  
 \* Создать папку для результата компиляции кода тестов  
 -->  
 <mkdir dir="${build.testOutputDir}"/>  
 <!--  
 \* Скомпилировать весь код тестов в созданную папку  
 -->  
 <javac destdir="${build.testOutputDir}">  
 <!--  
 Что необходимо скомпилировать  
 -->  
 <src>  
 <pathelement location="${build.testDir.0}"/>  
 </src>  
 <!--  
 Добавить ссылки на библиотеки и скомпилированный код проекта  
 -->  
 <classpath>  
 <path refid="build.test.classpath"/>  
 <pathelement location="${build.outputDir}"/>  
 </classpath>  
 </javac>  
 </target>  
  
 <!--  
 Запустить сборку проекта  
 -->  
 <target name="build" depends="compile"  
 description="Build main code and tests">  
 <!--  
 Создать папку для результатов сборки  
 -->  
 <mkdir dir="${build.dir}/${build.finalName}"/>  
 <!--  
 Скопировать в данную папку все файлы из ./webapp  
 -->  
 <copy todir="${build.dir}/${build.finalName}">  
 <!--  
 Добавить путь до ./webapp  
 -->  
 <fileset dir="${build.webappDir}"/>  
 </copy>  
  
 <!--  
 Создать папку для скомпилированного кода  
 -->  
 <mkdir dir="${build.dir}/${build.finalName}/WEB-INF/classes"/>  
 <!--  
 Скопировать скомпилированный код в данную папку  
 -->  
 <copy todir="${build.dir}/${build.finalName}/WEB-INF/classes">  
 <!--  
 Добавить путь до скомпилированного кода  
 -->  
 <fileset dir="${build.outputDir}"/>  
 </copy>  
  
 <!--  
 Создать папку для библиотек  
 -->  
 <mkdir dir="${build.dir}/${build.finalName}/WEB-INF/lib"/>  
 <!--  
 Скопировать в данную папку скачанные библиотеки  
 \* postgresql-42.5.1.jar  
 \* checker-qual-3.5.0.jar  
 -->  
 <copy file="${repo.local}/org/postgresql/postgresql/42.5.1/postgresql-42.5.1.jar"  
 todir="${build.dir}/${build.finalName}/WEB-INF/lib"/>  
 <copy file="${repo.local}/org/checkerframework/checker-qual/3.5.0/checker-qual-3.5.0.jar"  
 todir="${build.dir}/${build.finalName}/WEB-INF/lib"/>  
  
 <!--  
 Создать папку для манифеста  
 -->  
 <mkdir dir="${build.dir}/${build.finalName}/META-INF"/>  
 <!--  
 Создать файл манифеста в данную папку  
 -->  
 <manifest file="${build.dir}/${build.finalName}/META-INF/MANIFEST.MF">  
 <!--  
 Добавить в манифейс поле: кем собрано  
 -->  
 <attribute name="Built-By" value="${user.name}"/>  
 </manifest>  
  
 <!--  
 Создать .jar файл из всего приложения  
 -->  
 <jar destfile="${build.dir}/${build.finalName}.jar"  
 basedir="${build.dir}/${build.finalName}"  
 manifest="${build.dir}/${build.finalName}/META-INF/MANIFEST.MF"/>  
  
 <!--  
 Создать .war файл из всего приложения  
 -->  
 <war destfile="${build.dir}/${build.finalName}.war" compress="true"  
 webxml="src/main/webapp/WEB-INF/web.xml">  
 <lib dir="${build.dir}/${build.finalName}/WEB-INF/lib"/>  
 <classes dir="${build.outputDir}"/>  
 <fileset dir="${build.webappDir}" excludes="WEB-INF/web.xml"/>  
 </war>  
 </target>  
  
 <!--  
 Добавить звуковое уведомление при завершении сборки  
 -->  
 <target name="music" depends="build">  
 <sound>  
 <!--  
 Добавить ссылку на файл, который запуститься при успешной сборке  
 -->  
 <success source="${build.sound.success}"/>  
 <!--  
 Добавить ссылку на файл, который запуститься при неуспешной сборке  
 -->  
 <fail source="${build.sound.fail}"/>  
 </sound>  
 </target>  
  
 <!--  
 Запустить тесты  
 -->  
 <target name="test" depends="music">  
 <!--  
 Создать папку для сохранения результатов тестирования  
 -->  
 <mkdir dir="${test.reports}"/>  
 <!--  
 Запуст скомпилированных тестов  
 -->  
 <junit fork="true" printsummary="on">  
 <!--  
 Установка формата файлов как обычный текст  
 -->  
 <formatter type="plain"/>  
 <!--  
 Добавление ссылок на все файлы  
 -->  
 <classpath>  
 <!--  
 Ссылка на дополнительные библиотеки  
 -->  
 <path refid="build.test.classpath"/>  
 <!--  
 Ссылка на скомпилированный код проекта  
 -->  
 <pathelement location="${build.outputDir}"/>  
 <!--  
 Ссылка на скомпилированные тесты  
 -->  
 <pathelement location="${build.testOutputDir}"/>  
 </classpath>  
 <!--  
 Указание на файлы которые необходимо запустить  
 -->  
 <batchtest todir="${test.reports}">  
 <!--  
 Запустить все скомпилированные файлы тестов которые:  
 \* Начинаются с Test  
 \* Заканчиваются на Test  
 \* Заканчиваются на TestCase  
 \* Содержат Abstract и заканчиваются на Test  
 -->  
 <fileset dir="${build.testDir.0}">  
 <include name="\*\*/Test\*.java"/>  
 <include name="\*\*/\*Test.java"/>  
 <include name="\*\*/\*TestCase.java"/>  
 <exclude name="\*\*/\*Abstract\*Test.java"/>  
 </fileset>  
 </batchtest>  
 </junit>  
 </target>  
  
 <!--  
 Выполнить коммит определенных файлов, если указано свойство  
 -->  
 <target name="diff" depends="test" if="git.diff.files">  
 <if>  
 <!--  
 Если указан параметр, но не указаны файлы  
 \* то напечатает "Property "diff.files" is empty."  
 -->  
 <length string="${git.diff.files}" length="0" trim="true"  
 when="greater"/>  
 <then>  
 <!--  
 Выполнить комманду git diff [files] и записать результат в  
 переменную 'classes.change'  
 -->  
 <exec executable="git" outputproperty="classes.change">  
 <arg value="diff"/>  
 <arg value="${git.diff.files}"/>  
 </exec>  
  
 <if>  
 <!--  
 Если в переменной 'classes.change' что-то записано (не  
 пустая строка), следовательно есть изменения и их необходимо  
 закоммитить.  
  
 Иначе отобразить 'No changes in files. Nothing to commit.'  
 -->  
 <length string="${classes.change}" length="0" trim="true"  
 when="greater"/>  
 <then>  
 <!--  
 Выполнить комманду git add [files]  
 -->  
 <exec executable="git">  
 <arg value="add"/>  
 <arg value="${git.diff.files}"/>  
 </exec>  
 <!--  
 Выполнить комманду git commit -m "New commit"  
 -->  
 <exec executable="git">  
 <arg value="commit"/>  
 <arg value="-m"/>  
 <arg value="New commit"/>  
 </exec>  
 </then>  
 <else>  
 <echo>No changes in files. Nothing to commit.</echo>  
 </else>  
 </if>  
 </then>  
 <else>  
 <echo>Property "diff.files" is empty.</echo>  
 </else>  
 </if>  
 </target>  
  
</project>

Файл – build.properties

# Сыылка на путь к папке с библиотеками  
path.lib=./lib  
# Путь до папки в которую будут скачаны библиотеки  
repo.local=${user.home}/.m2/repository  
  
# Название папку в которой будет происходить сборка  
build.dir=target  
# Название итоговой сборки  
build.finalName=webapp  
# Путь до папки с основоным кодом проекта  
build.srcDir.0=src/main/java  
# Путь до папки с кодом тестов  
build.testDir.0=src/test/java  
# Путь до кода папки с веб-приложением  
build.webappDir=src/main/webapp  
# Путь до папки в которой будет сохранен скомпилированный код проекта  
build.outputDir=${build.dir}/classes  
# Путь до папки в которой будет сохранен скомпилированный код тестов  
build.testOutputDir=${build.dir}/test-classes  
# Путь до папки в которую будут сохранены результаты тестирования  
test.reports=${build.dir}/test-reports  
  
# Путь до файла, который будет проигрываться при успешной сборке  
build.sound.success=sounds/sample-3s.wav  
# Путь до файла, который будет проигрываться при неуспешной сборке  
build.sound.fail=sounds/sample-9s.wav  
  
# Какие файлы необходимо закоммитить при их изменении, через пробел  
git.diff.files=src/main/java/area\_zone/Area.java

Вывод:

В ходе выполнения данной лабораторной работы, мы познакомились с утилитой для автоматизации процесса сборки Apache Ant, а также научились тестировать код при помощи Junit-тестов.