

Laboratory 2: JUnit

Name : _____

Student Number : _____

Laboratory Time : _____

Objectives

- Creating JUnit Test Cases in Eclipse
- Creating JUnit Test Suites in Eclipse
- Running JUnit Test Cases and Suites in Eclipse

Preamble

JUnit is a unit-testing framework for Java. It provides a common and reusable structure that is required for developing automated and repeatable unit tests for Java classes. JUnit provides a base class called `TestCase` that can be extended to create series of tests for your classes, an assertion library that can be used to evaluate the results of the tests, and test drivers, both command line and GUI based, called `TestRunner` to run the test cases you create.

The recent version of the Eclipse JDT already has JUnit Plug-in built in to make creating and running test cases more convenient. The plug-in includes a wizard for assisting in creating testing a test case and test suite, and an environment for running them.

In this lab, you will learn how to set up a project for creating JUnit tests. Then you will create test cases and a test suite, and run them.

Grading Checklist

By the end of the laboratory session, you need to demonstrate to the TA that you can do the following tasks. The TA will check off the items below that you have completed and collect this cover page from you.

- JUnit library is in the project's build path
- Test case for Scrabble class has been created and asserts added to `initGame` method body
- The test case for Scrabble runs successfully (may find errors in Scrabble)
- Test case for the `PlayerActions` class has been created and asserts added to `calculateWordScore` method body
- The test case for `PlayerActions` runs successfully (may find errors in `PlayerActions`)
- Test suite created and works

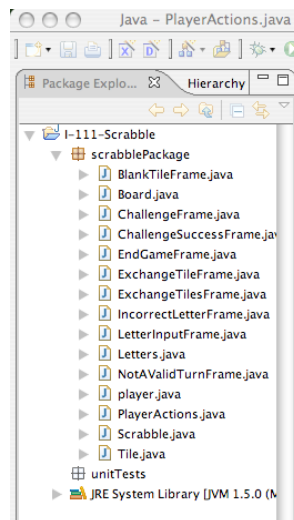
TA Initials: _____

Instructions for the Laboratory

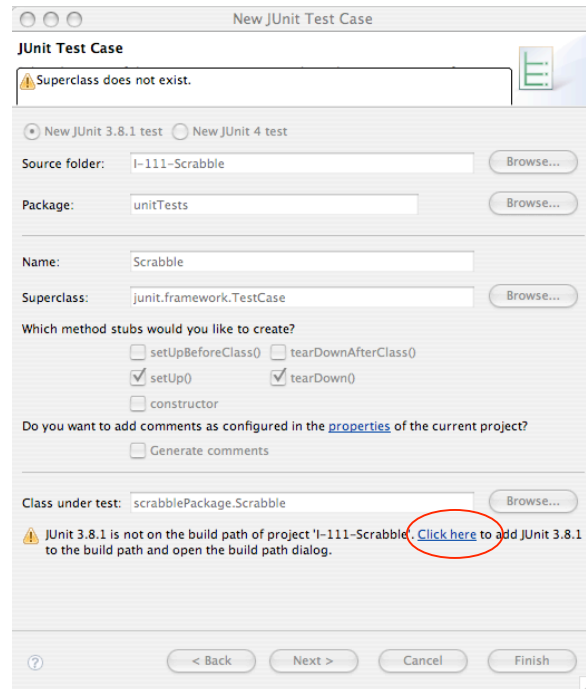
Task 1: Set up a new project and create JUnit Test Cases for the Scrabble class

For this task, you will set up a new project and include the provided classes in the project. Then you will create a test case to test the Scrabble class. In JUnit convention, a test class is created for every application class, and every non-trivial method is tested.

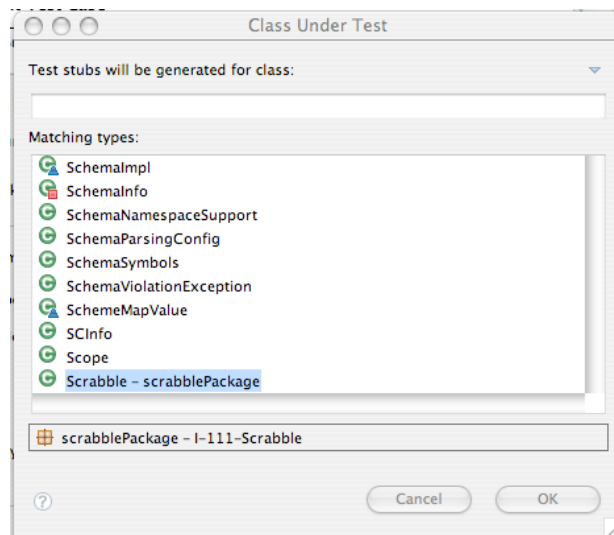
- a) Download and uncompress Scrabble.zip, which contains Scrabble.java and other Java files.
- b) Create a new Java project in Eclipse called inf111_JUnit.
- c) It is a good practice to separate the application source code from test case code.
 1. Import Scrabble.java and other files in the zip file into a default (unnamed) package. You can do this by right clicking on the project and then selecting Import → General → File System. You need to select the directory where you have Java files residing. You can use Eclipse's file filtering capabilities here.
 2. Create a new package and name it scrabblePackage (note the lowercase 's')
 3. Use the refactoring feature to move the java files from the default package into the scrabblePackage that you just created.
 4. Create a new package for test source code by right clicking on the project and selecting new package and name the package unitTests.



- d) Create a JUnit Test Case Class.
 1. First, to bring up the New JUnit Test Case Wizard, select the unitTests package. Then, select File → New → JUnit Test Case.
 2. Add JUnit library to the build path by clicking on the link at the bottom on the dialog box that pops up when you are creating the test case.



3. Name the class by putting the name ScrabbleTest in the name test box.
4. Select setUp() and tearDown() boxes to automatically generate skeleton of these methods. The setUp() and tearDown() methods are run before and after each test case is run.
5. For "Class under test", enter Scrabble. Eclipse should find Scrabble class in the scrabblePackage and should select it (Browse..). Then press Next.



6. Now you can select methods for which test method stubs will be created. Select initGame().
7. Press Finish. A new class, ScrabbleTest, that extends junit.framework.TestCase is created with generated method stubs.

- e) Implement a test case function in Scrabble (test class) to test the Scrabble class.
1. Create a new class variable, `scrabble` as a private variable of the type `Scrabble`.

```
Scrabble scrabble;
```
 2. Implement the `setUp()` method to initialize variables.

```
scrabble = new Scrabble();
```
 3. Implement the `tearDown()` method to clear the variable.

```
scrabble = null;
```
 4. Implement the `testInitGame()` to check that the method that initializes the game is working correctly.

```
scrabble.initGame();
assertNotNull(Scrabble.players);
assertFalse(Scrabble.isFirstTurn);
```
 5. Add in `testInitGame()` two assertions to validate that racks and letterbags (properties of `Scrabble`) are not null.

If you get a compilation error, be sure to import the required packages, including `junit.framework.Assert` package.

You can find more information on the `assert` API in JUnit JavaDoc at <http://www.junit.org/junit/javadoc/3.8.1/index.htm>

Task 2: Running `GUIEnvironmentTest` as a JUnit test case

In this task, you will run the test case in `ScrabbleTest` as a JUnit Test Case.

- a) Right click at `ScrabbleTest`, and select `Run` → `JUnit Test`.
- b) You should see the result of your test case in JUnit view. If the view does not appear, show the JUnit view by selecting `Window` → `Show View` → `Other` → `Java` → `JUnit`.

Task 3: Create JUnit Test Case for the `PlayerActions` class (You may want to skip this task if you are running behind on time)

- a) Now repeat Task 1, step d) to create a test case for the `calculateWordScore` method in the `PlayerActions` class. Implement a test case function in `PlayerActionsTest` to test the `PlayerActions` class.
 1. Create two new class variables, `scrabble` as a private variable of the type `Scrabble` and `player` as a private variable of the type `PlayerActions`.

```
Scrabble scrabble;
PlayerActions player;
```
 2. Implement the `setUp()` method to initialize variables.

```
scrabble = new Scrabble();
player = new PlayerActions();
```
 3. Implement the `tearDown()` method to clear the variable.

```
scrabble = null;
```

```
player = null;
```

4. Implement the `testCalculateWordScore()` to check that the `calculate` method is working correctly.

```
String[] word = {"T","E","S","T","I","G"};
//simulates placing the word on the board
for(int i = 0; i < word.length; i++){
    Scrabble.nextLetter = word[i];
    PlayerActions.placeTile(7, i+7);
}
final int score = PlayerActions.calculateWordScore();
assertEquals(score,16);
```

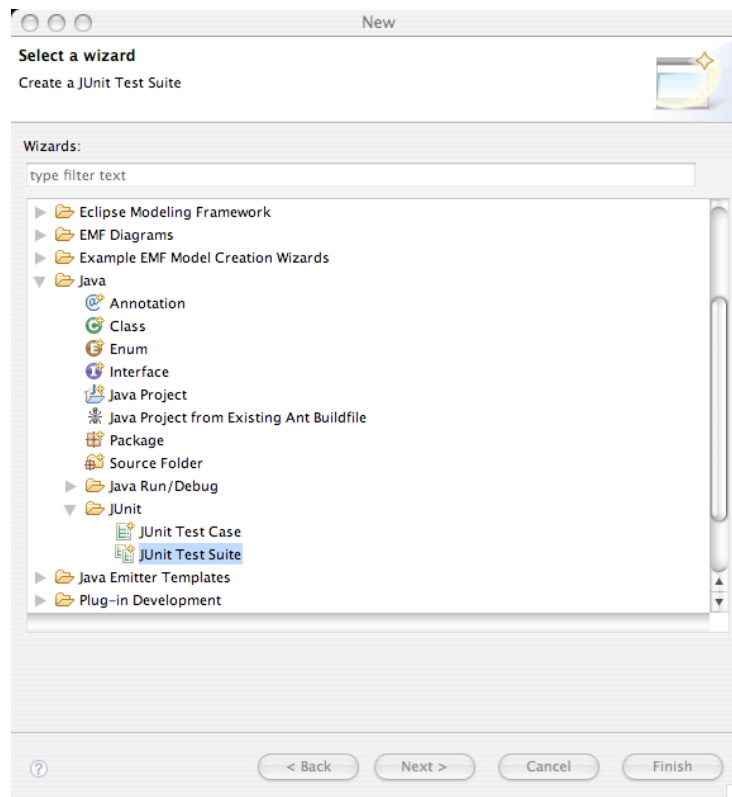
5. Run `PlayerActionsTest` as a JUnit Test Case (See Task 2).

If you get a compilation error, be sure to import the required packages, including `junit.framework.Assert` package.

Task 4: Creating a Test Case Suite

In this task, you will create a test case suite for the test cases created in Task 1 and Task3. A test case suite allows more convenient test case execution and will allow you to run all your test cases at once.

- a) Create a Test Suite by selecting the `unitTests` package, and then selecting `File` → `New` → `Other` → `Java` → `JUnit Test Suite`.

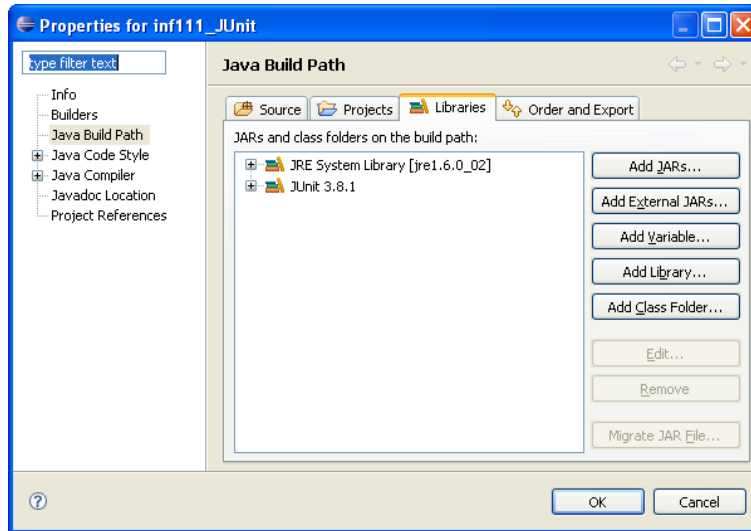


- b) In the next screen, name the Test Suite "ScrabbleAllTests", and include your test cases created in the Task 1 and Task 3 (ScrabbleTest and also PlayerActionsTest if you had time) into the test suite by selecting the check boxes in front of the appropriate classes. Then click Finish.
- c) Run the test suite by selecting the Test Suite class, and then Right Click → Run → JUnit Test

Appendix

You can also manually add the JUnit library, junit.jar, to the project's build path using the following steps.

- a) Right click the project and select Properties. The properties dialog box as shown below should appear.



- b) Select Java Build Path on the left panel and bring the Libraries tab forward.
- c) Click on "Add External JARs..." button.
- d) In the JAR Selection dialog box, find the plugins directory under Eclipse's installation directory (C:\Opt\eclipse). Then locate the JUnit folder. The current version of JUnit should be org.junit_3.8.1. From that directory, select junit.jar, as in the figure below.

