# Lab TDD 1: Setting up a TDD project

## Objectives

In this lab, you will set up a basic TDD project using Junit 4 to write some code for a simple calculator project.

The purpose of this lab is to get you familiar with the mechanics of setting up a project and the TDD workflow.

## Part One – Project Setup

The specification for the project is to produce a simple calculator that implements a calculator interface with the four standard arithmetic operations and a method that calculates a floating point quotient.

All of the operations take two integers as inputs and return an integer, except for the floating point division which returns a double.

### Step 1: Set up the project

1. Create a standard Java project with a package named “calc”
2. Create an interface called Calculator

Text

Description automatically generated

1. Create an implementation class called CalcImp that implements the Calculator interface.
2. The class methods should all have empty bodies, as shown on the next page.

Text

Description automatically generated

### Step 2: Add Junit

1. Add the Junit 4 library to the build path by selecting build path and selecting “Add libraries”

Graphical user interface, text, application

Description automatically generated

1. Graphical user interface, text, application

   Description automatically generatedSelect JUnit
2. Graphical user interface, text, application

   Description automatically generatedSelect Next and JUnit 4 then select finish
3. Graphical user interface, application, Word

   Description automatically generatedYou should see the library now as part of the project

### Step 4: Create a test class

1. Graphical user interface, text, application

   Description automatically generatedSelect new JUnit test case from the new menu.
2. JUnit will create a new test class to hold the test methods. In this case, you can name the test class whatever you want. In the example the imaginative name “TestClass” is used.
3. Chose all the fixture stubs to be autogenerated
4. Select the CalcImp as the class under test

Graphical user interface, text, application

Description automatically generated

1. This produces the class on the next page.

### A picture containing table Description automatically generatedPart 5: Run the tests

1. In the test class, right mouse click and select the ”run as JUnit test case” option.
2. Graphical user interface, text, application

   Description automatically generatedThe test output should show the one test method we have has failed. This is because the default “fail()” method throws a test fail exception.

### Part 6: Ignore the test

1. To get JUnit to skip the test, add the @Ignore annotation and rerun the tests
2. Notice that all the tests passed but it also notes that one test is skipped

Graphical user interface, application

Description automatically generated

### Part 7: Make the test past vacuously

1. Remove the @Ignore annotation and replace the fail() method with an output method that prints a message to the console
2. Also change the name of the test method to testOne()

Graphical user interface, text, application

Description automatically generated

**Save your lab because the next lab picks up from here.**