Simple Java Unit Testing with JUnit and Netbeans

What is Unit Testing

- Method of testing that verifies the individual units of the code is working properly (Wikipedia).
- Test the smallest unit in source code

Why Unit Testing

- Verifies if the unit is working offcourse! ☺
- Make sure the unit is working even after late changes in source code (regression test)
- Provides living documentation of how the units (e.g Method) works.

Unit Testing in Java

- 2 most used testing framework in java
  - Junit
  - TestNG
- This time we used Junit because its embedded in netbeans.

What is Junit

- Is a unit test framework in java
- Developed by Kent Beck and Erich Gamma
- Widely used and commonly become standard unit test framework
- Is part of xUnit family. xUnit is a ported Junit for various language.
  - PHPunit (PHP)
  - Nunit(.NET)

Simple JUnit

- Create test class and test case.
- Use an assert method for ensuring method output
  - assertEquals()
  - assertTrue()
  - assertNotNull()
- Can be invoked manually by running the test class or automated by using ant script

Junit in Netbeans

- You don’t need to load the jar into netbeans project.
- By default the jar is embedded in test libarary folder
- And also netbeans has test class and test case code generation menu

Lets Do The Code

- Lets start with heating up our Netbeans and create new java project.
- Make a simple class having both return valued and void method.
- Let the return valued method do simple process for example addition or substraction.
- Just print something in the void method.

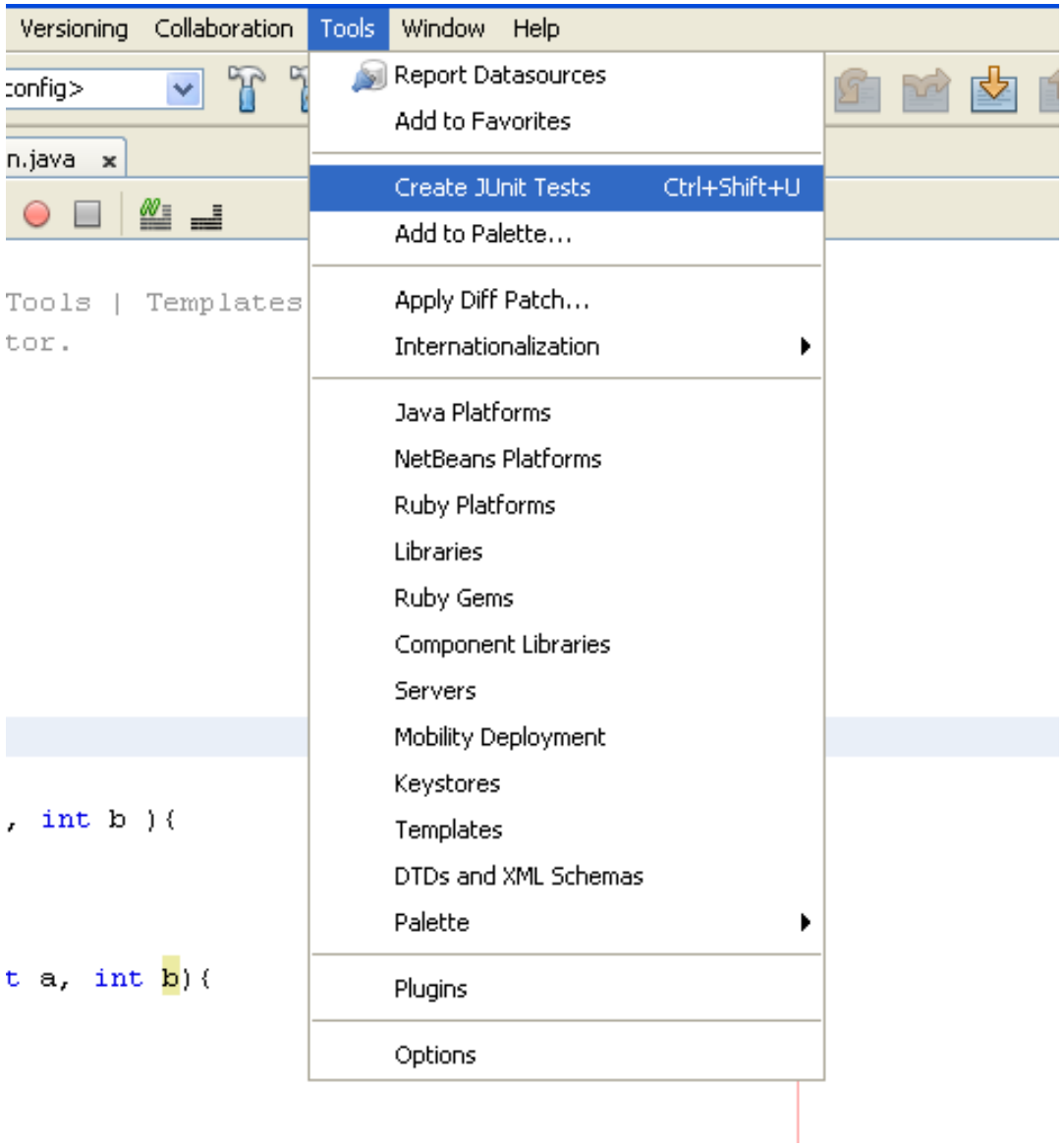
SimpleMath.java

```
11   */
10   * @author Kiki
11   */
12  public class SimpleMath {
13
14
15          public int doAddition(int a, int b ){
16
17              return a + b ;
18          }
19          public int doSubtraction(int a, int b){
20
21              return a / b;
22
23          }
24          public void printAddition(int a, int b){
25
26                  System.out.println("var1 = "+a+" , var2 = "+b+" " +
27                          "hasilnya adalah = "+doAddition(a, b));
28
29          }
30      }
```

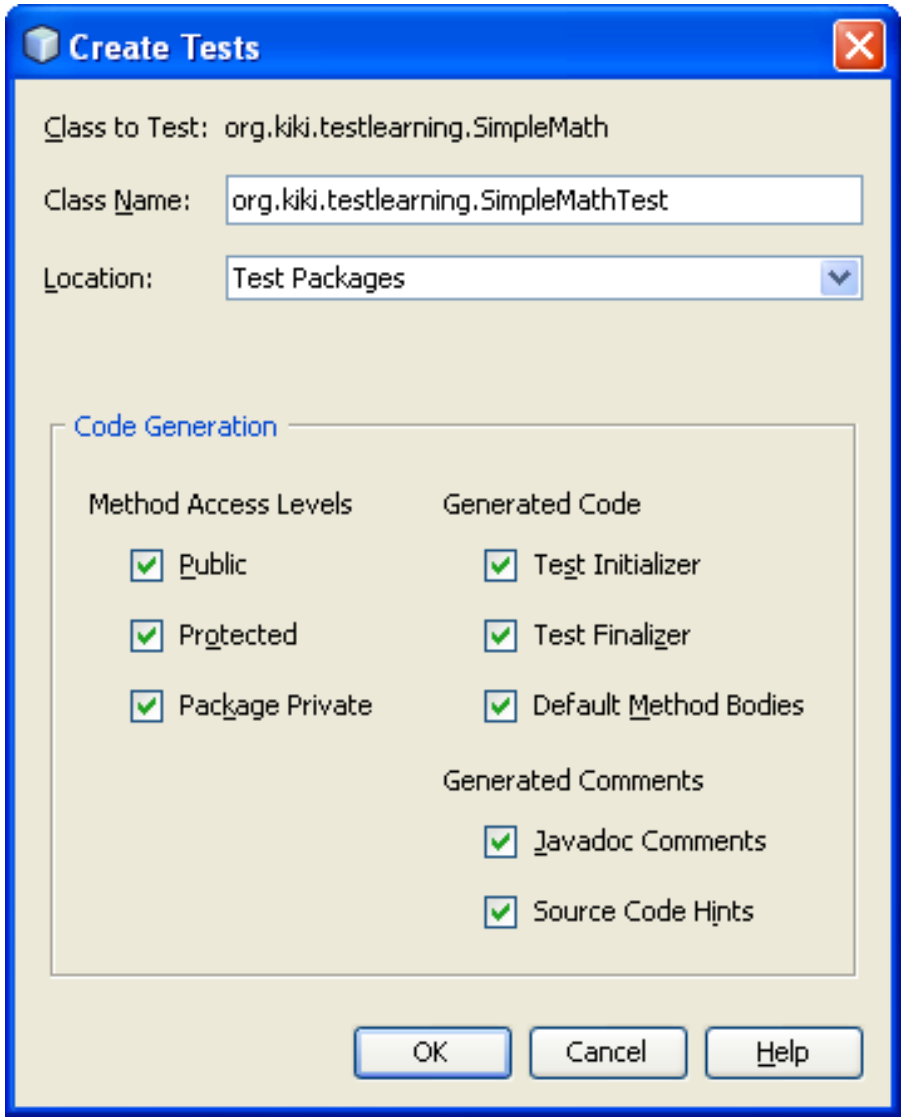
Create Unit Test

- Choose this menu in netbeans
  - Tools > Create Junit Test
- Or just simply press Ctrl + Shift + U.
- A window dialogue will appear, choose suitable options.
- Or you can leave it as is. Like I usually do ☺.
- Test case will automatically build inside the test package folder.

Unit Test Menu



Unit Test Window



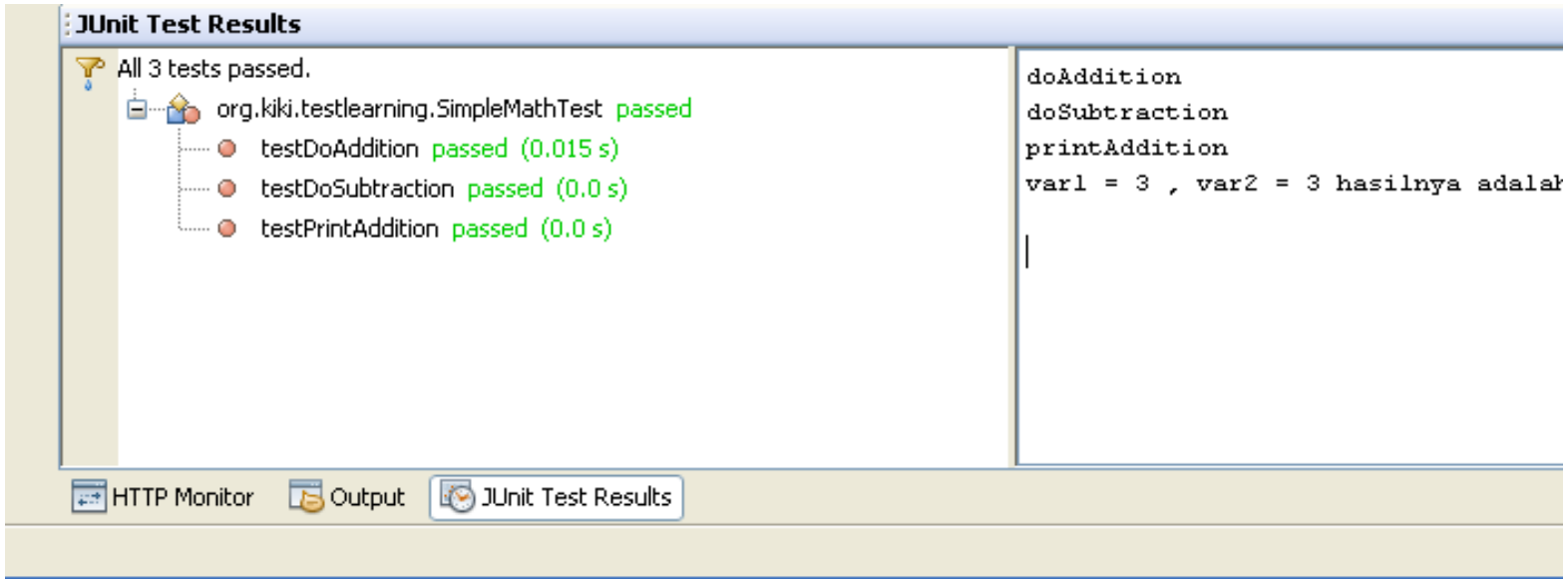
SimpleMathTest.java

```
41          * Test of doAddition method, of class SimpleMath.
42          */
43         @Test
44         public void testDoAddition() {
45             System.out.println("doAddition");
46             int a = 2;
47             int b = 2;
48             SimpleMath instance = new SimpleMath();
49             int expResult = 4;
50             int result = instance.doAddition(a, b);
51             assertEquals(expResult, result);
52             // TODO review the generated test code and remove the default call to fail.
53
54         }
55
56         /**
57          * Test of doSubtraction method, of class SimpleMath.
58          */
59         @Test
60         public void testDoSubtraction() {
61             System.out.println("doSubtraction");
62             int a = 3;
63             int b = 1;
64             SimpleMath instance = new SimpleMath();
65             int expResult = 3;
66             int result = instance.doSubtraction(a, b);
67             assertEquals(expResult, result);
68             // TODO review the generated test code and remove the default call to fail.
69
70         }
```

Unit Testing

- Assign the variable value for the test case.
- Remove the fail() method in return valued method test.
- Run the test class using Shift + F6.
- See the test result

Test Result


JUnit Test Results
All 3 tests passed.
org.kiki.testlearning.SimpleMathTest passed
testDoAddition passed (0.015 s)
testDoSubtraction passed (0.0 s)
testPrintAddition passed (0.0 s)
doAddition
doSubtraction
printAddition
var1 = 3 , var2 = 3 hasilnya adalah
HTTP Monitor
Output
JUnit Test Results