TDD

LAB ASSIGNMENTS

Lab 1:

Write a test case to check produceProduct() method from service layer. This method was dependent on DAO layer findProduct() method. Dependent object can be provided with Mockito. Use maven to include the supporting Mockito jars.

Steps:

1. Open GitHub URL (https://github.com/caprepo/softwareRepo)
2. Click **Day1-Lab-1.zip folder,** next page click download link. The zip file will be downloaded.
3. Unzip the folder
4. Go to Eclipse IDE, choose import ->General -> Existing Projects into Workspace
5. Under select root directory click browse button.
6. Go to the unzip folder location and choose the folder in the name of ProductApp.
7. Now the Product Application project will be imported in your local machine.
8. Once the project imported into Eclipse IDE, locate the src/test/java folder.
9. Under this folder, open the class ProductAppTestCase.
10. In this class you can find the below method

```
//Fill the method with proper test case
//conditions when produce product using mock object
@Test
public void when_produce_with_dao_dependency() throws InvalidProductQuantityException{
       //Your code goes here
}
```

11. Read the problem statement and then implement your code in the highlighted area.
12. Right click the class goto RunAs→Junit. If you get green color ribbon, it's a success test case. In case if you get red color ribbon, please re-do your work.

Conclusion:

From the above example, we understand that how to mock dependency object in unit test cases.

Note:

In case in the current workspace if you have ProductApp already, please remove the project. Then import the project in the workspace. If the same name of the project exists in the workspace, eclipse will not allow you to import the project.

Lab 2:

Write a parameterized test case in ProductApp to test findSquare() method.

Steps:

1. Open GitHub URL (https://github.com/caprepo/softwareRepo)
2. Click **Day1-Lab-Parameter.zip** **folder,** next page click download link. The zip file will be downloaded.
3. Unzip the folder
4. Go to Eclipse IDE, choose import ->General -> Existing Projects into Workspace
5. Under select root directory click browse button.
6. Go to the unzip folder location and choose the folder in the name of ProductApp.
7. Now the Product Application project will be imported in your local machine.
8. Once the project imported into Eclipse IDE, locate the src/test/java folder.
9. Open the test Class called **FindSquareParameterTestCase** class.
10. Add your code in the highlighted places.

```
package org.capgemini.test;
import org.capgemini.service.ProductService;
import org.capgemini.service.ProductServiceImpl;
import org.junit.Test;
import org.junit.runner.RunWith;
import org.junit.runners.Parameterized;

@RunWith(Parameterized.class)
public class FindSquareParameterTestCase {

	private ProductService productService=new ProductServiceImpl();

	private int input;
	private int expected;

	public FindSquareParameterTestCase(int input, int expected) {
		super();
		this.input = input;
		this.expected = expected;
	}

	//Add method which carries all the parameters.
	//Create a method with set of inputs and expected output.
	@Test
	public void test_findSquare_Method(){
		//Add your test logic here
	}


}
```

11. Read the problem statement and then implement your code in the highlighted area.
12. Right click the class goto RunAs→Junit. If you get green color ribbon, it's a success test case. In case if you get red color ribbon, please re-do your work.

Conclusion:

From the above example, we learnt parameterized test cases.

Note:

In case in the current workspace if you have ProductApp already, please remove the project. Then import the project in the workspace. If the same name of the project exists in the workspace, eclipse will not allow you to import the project.