**Including Unit Test Cases and Test Coverage Tools**

**Steps to configure Eclipse ready for writing test cases: -**

1. In your POM.xml file you need to include the SPRING BOOT TEST dependency. Search for it in google and add it. The below one I got from google.

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

</dependency>

1. You need to install JACOCO (Java Code Coverage). Or EclEMMA it is also a code coverage tool.
2. Also you need to install INFINITEST plugin from Eclipse market place. Which will help you to trigger immediate response of your test case status when you change some piece of code. (RED color status, Green Color Status, Blue Color Status).
3. You also need to include some favorites to your eclipse,
4. in MAC go to eclipse and click on PREFERENCES 🡪 Serach for favorites 🡪 Under java, Editor-- favorites will be there click on it.
5. In that NEW TYPE, it will open new window, on that click on browse, then search for **assert** of JUNIT type. (assert in org.junit). and add it. (assertion validation we can do inside the junit test case. For that we are adding it.).
6. Same way, Next you **add org.hamcrest – Machers** to your favorites. ( machers will help you to provide function like equalTo() etc – inside the assert statements. )

DONE with Setup.

**How to start writing test cases and include that with your Project: -**

1. Your code should be mavenized, then you will see a separate folder structure like src/test/java. This is the location where you will writing test cases for your java code.
2. Right cick on that path 🡪 new 🡪 others 🡪 search for test 🡪 you will be able to see two options like test case, test scenario under JUNIT. Select test case. Use right package and right test case name. then say ok.
3. You are having a class to implement the test cases.
4. Here is the example test case that I have written for,

**public** **class** ViewControllerTest {

@Test

**public** **void** testThatTestCanBeRun() **throws** Exception {

*assertThat*(**true**, *is*(*equalTo*(**true**)));

}

@Test

**public** **void** testThatDisplayHomeSendsBackHelloView() **throws** Exception {

ViewController viewController = **new** ViewController();

String view = viewController.displayHome();

*assertThat*(view, *is*(*equalTo*("helloview")));

} }

TIPS: -

1. Whenever your open new test class, then inside the class, you just type “test” and say CTRL + SPACE. Then it will give option to auto generate test case method format. Say enter then just give the method name. and add body according to your requirement.

**Setup for Integration Test cases writing and example: -**

What is Integration test: - Integration test is where we write test cases on DAO transactions. The purpose of integration testing is to verify functional, performance, and reliability [requirements](https://en.wikipedia.org/wiki/Requirement) placed on major design items. **Integration testing** (sometimes called **integration and testing**, abbreviated **I&T**) is the phase in [software testing](https://en.wikipedia.org/wiki/Software_testing) in which individual software modules are combined and tested as a group.

Steps to follow write integration test cases:

1. You need to add **@RunWith** annotation and pass SpringJunit4ClassRunner.class as parameter for it.

Ex: @RunWith(SpringJUnit4ClassRunner.**class**)

1. And then you have to add **@SpringApplicationConfiguration** annotation and here you need to pass your spring starting of application as a parameter.

Ex: @SpringApplicationConfiguration(GssApplication.**class**)

GssApplication class has main method which is the starting of spring boot application.

1. Then Start writing your test cases on your need. One example will be shown below.

import static org.hamcrest.Matchers.greaterThan;

import static org.hamcrest.Matchers.is;

Import static org.junit.Assert.assertThat;

import java.util.List;

import java.util.Map;

import org.junit.Test;

import org.junit.runner.RunWith;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.boot.test.SpringApplicationConfiguration;

import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;

import com.target.gss.GssApplication;

@RunWith(SpringJUnit4ClassRunner.class)

@SpringApplicationConfiguration(GssApplication.class)

public class GssLoadDropDownsDaoTest {

@Autowired

GssLoadDropDownsDao dropDownsDao;

@Test

public void testThatSelectedAttributeCountShouldBeGreaterThan300() throws Exception {

List<Map<String,Object>> result = dropDownsDao.poulateSelectedAttribute("Department");

assertThat(result.size(), is(greaterThan(300)));;

}

}