Junit for Android

The following code shows a JUnit test using the JUnit 5 version. This test assumes that the MyClass class exists and has a multiply(int, int) method.

```
import static org.junit.jupiter.api.Assertions.assertEquals;

import org.junit.jupiter.api.Test;

public class MyTests {

  @Test
  public void multiplicationOfZeroIntegersShouldReturnZero() {
    MyClass tester = new MyClass(); // MyClass is tested

    // assert statements
    assertEquals(0, tester.multiply(10, 0), "10 x 0 must be 0");
    assertEquals(0, tester.multiply(0, 10), "0 x 10 must be 0");
    assertEquals(0, tester.multiply(0, 0), "0 x 0 must be 0");
  }
}
```

Junit naming conventions

There are several potential naming conventions for JUnit tests. A widely-used solution for classes is to use the "Test" suffix at the end of test classes names.

As a general rule, a test name should explain what the test does. If that is done correctly, reading the actual implementation can be avoided.

One possible convention is to use the "should" in the test method name. For example, "ordersShouldBeCreated" or "menuShouldGetActive". This gives a hint what should happen if the test method is executed.

Run your test from command line

The org.junit.runner.JUnitCore class provides the runClasses() method. This method allows you to run one or several tests classes. As a return parameter you receive an object of the type org.junit.runner.Result. This object can be used to retrieve information about the tests.

The following class demonstrates how to run the MyClassTest. This class executes your test class and write potential failures to the console.

```
package de.vogella.junit.first;
import org.junit.runner.JUnitCore;
import org.junit.runner.Result;
import org.junit.runner.notification.Failure;
public class MyTestRunner {
  public static void main(String[] args) {
    Result result = JUnitCore.runClasses(MyClassTest.class);
    for (Failure failure : result.getFailures()) {
    System.out.println(failure.toString());
    }
  }
}
```

Robotium

ANDROID WITH ROBOTIUM

Robotium is a test framework created to make it easy to write powerful and robust automatic black-box test cases for Android applications so test developers don't need any further information about the Android app's structure or implemented classes. All they need is the name of the main class and the path that links to it. With the support of Robotium, test case developers can write function, system and acceptance test scenarios, spanning multiple Android activities. This blog post is meant to serve as a mini-tutorial on setting up Robotium as an automated acceptance testing framework for Android applications. The post will summarize all the info related to setting up & writing tests in Robotium.

Robotium officially supports Android 1.6 and up. Robotium has full support for Activities, Dialogs, Toasts, Menus and Context Menus.

Robotium provides the following benefits:

You can develop powerful test cases, with minimal knowledge of the application under test.

The framework handles multiple Android activities automatically.

Minimal time needed to write solid test cases.

Readability of test cases is greatly improved, compared to standard instrumentation tests.

Test cases are more robust due to the run-time binding to GUI components.

Blazing fast test case execution.

Integrates smoothly with Maven or Ant to run tests as part of continuous integration

With Robotium it is possible to run test cases on applications that are pre-installed http://code.google.com/p/robotium/wiki/RobotiumForPreInstalledApps

Robotium can be integrated into continuous integration and can get code coverage for Robotium tests.(http://code.google.com/p/robotium/wiki/QuestionsAndAnswers)

CREATING AND RUNNING TESTCASES:

STEP 1: ROBOTIUM.JAR

STEP 2: CREATE JAVA CLASS

STEP 3: RUN TEST CASE

Eclipse: After all test cases are created right click on the test project Run As >> Run As Android JUnit Test.

ADB:

Use adb to Install the application apk

>> adb install ApplicationToTest.apk

Use adb to install the test project apk:

>> Adb install ExampleTesting.apk