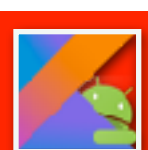




Basic Unit Testing with Android

@somkiat

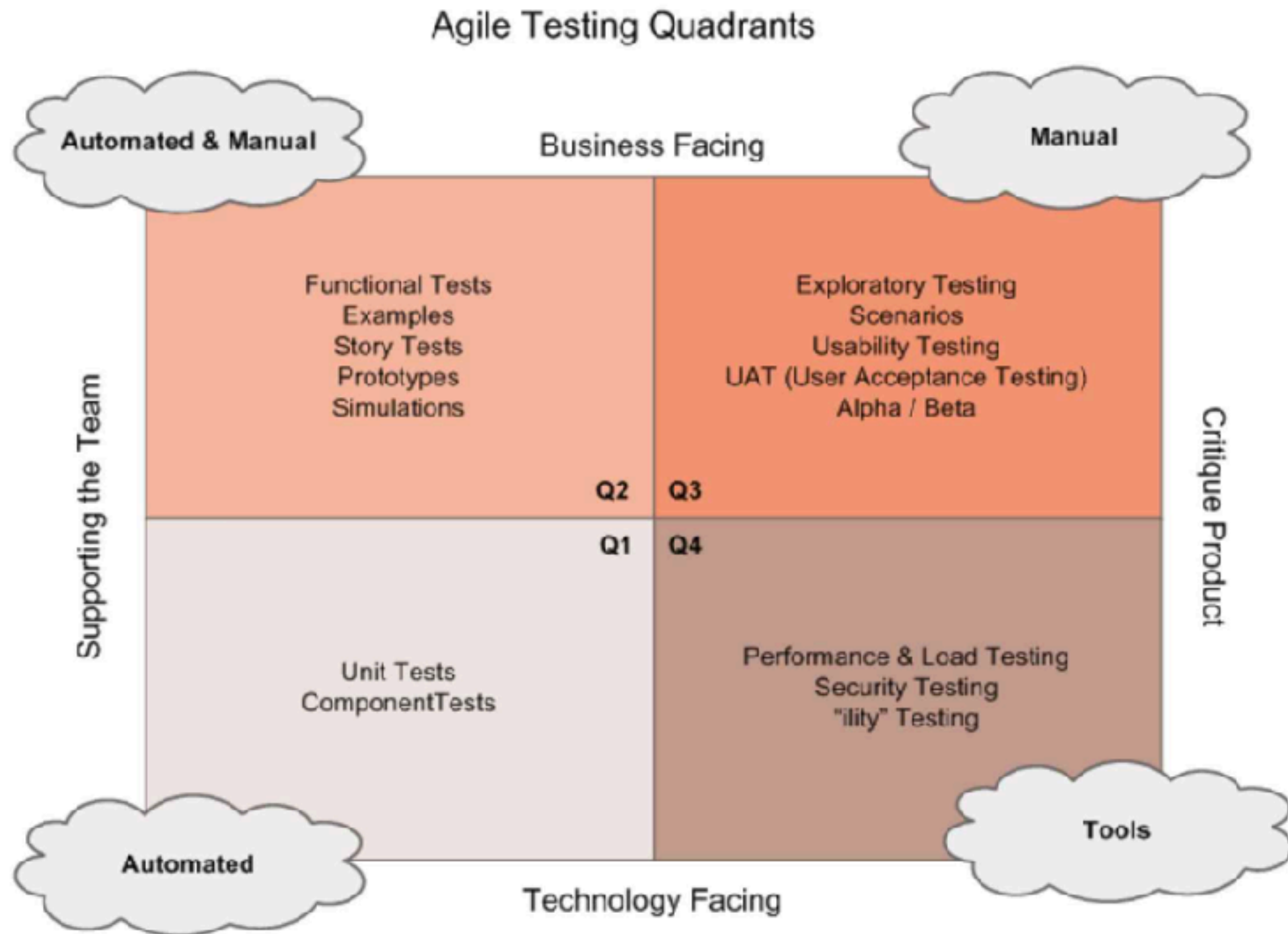


Topics

Mobile Testing
Unit Testing
Workshop
Homework



Agile Testing



<http://lisacrispin.com/2011/11/08/using-the-agile-testing-quadrants/>

Mobile Testing



Mobile Testing with Android



UI Testing

UI-less Testing

Unit Testing



Mobile Testing with Android



Espresso/UI Automator

Instrumentation test

JUnit + Mockito



Mobile Testing with Android



Espresso/UI Automator

Instrumentation test

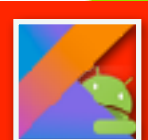
JUnit + Mockito



Unit Testing framework



<http://junit.org/junit4/>
<https://github.com/mockito/mockito>



Build tool



Let's start with Unit testing



Update Android Studio 3

Android Studio

The Official IDE for Android

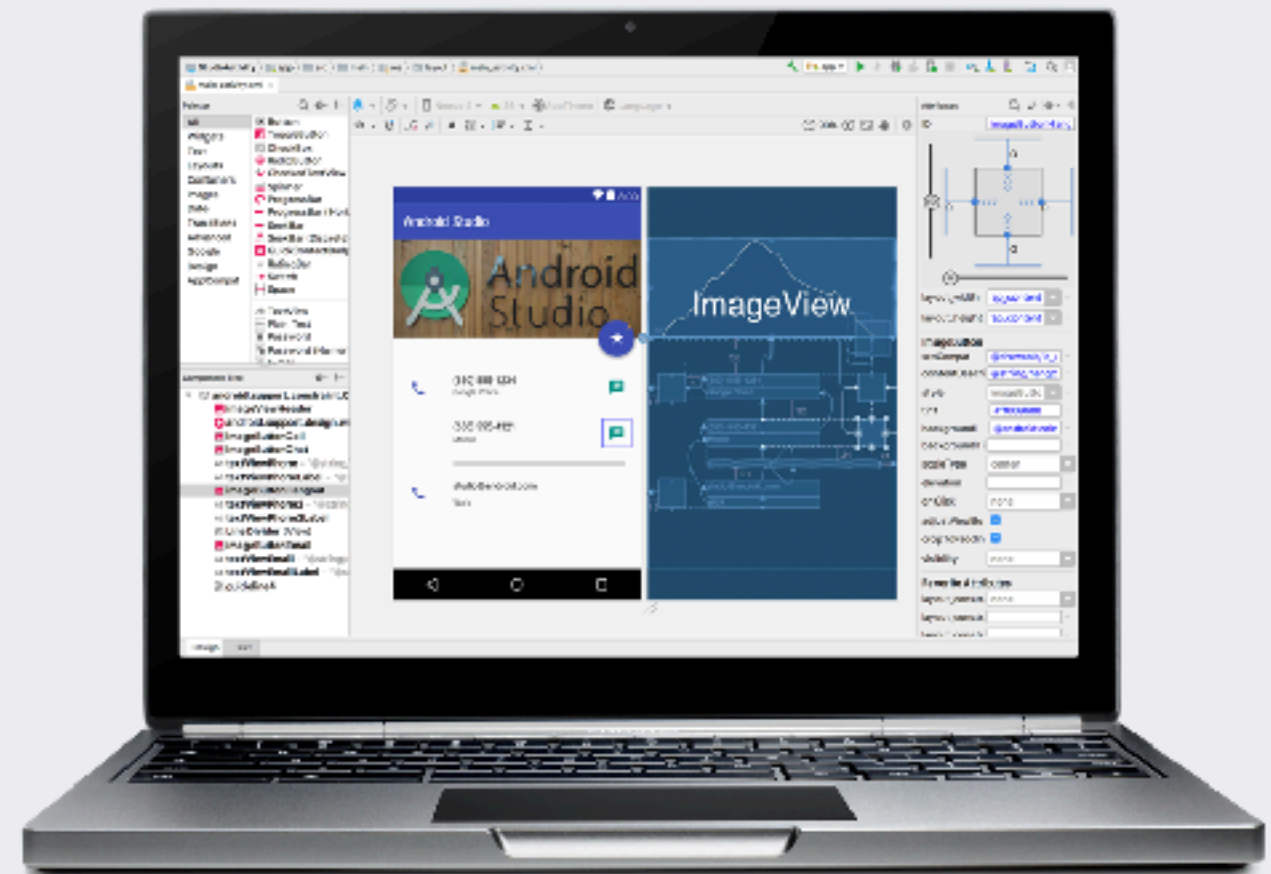
Android Studio provides the fastest tools for building apps on every type of Android device.

World-class code editing, debugging, performance tooling, a flexible build system, and an instant build/deploy system all allow you to focus on building unique and high quality apps.

DOWNLOAD ANDROID STUDIO
3.0 FOR MAC (731 MB)

➤ Read the docs

➤ See the release notes



<https://developer.android.com/studio/index.html>

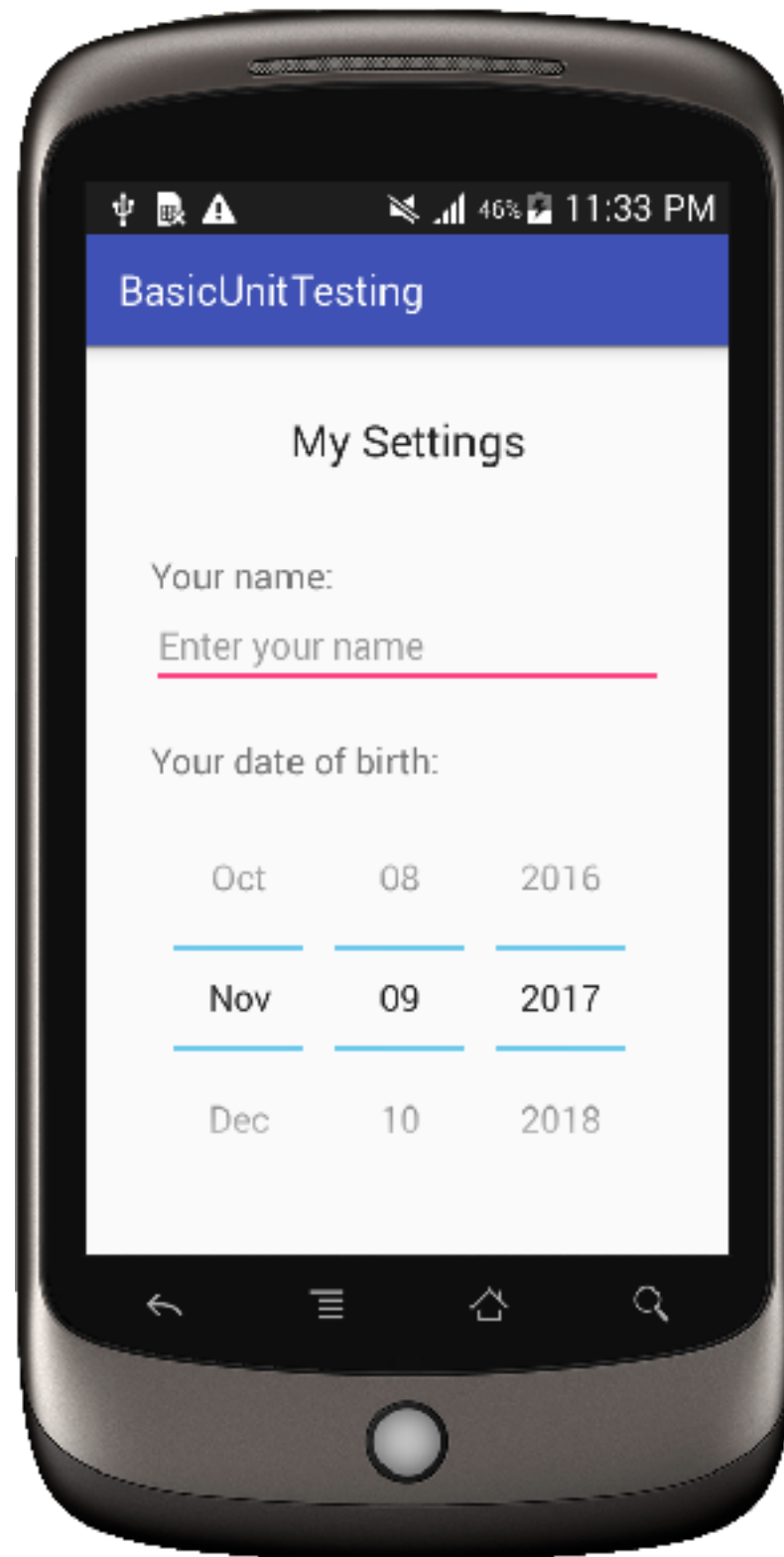


Import code from Github

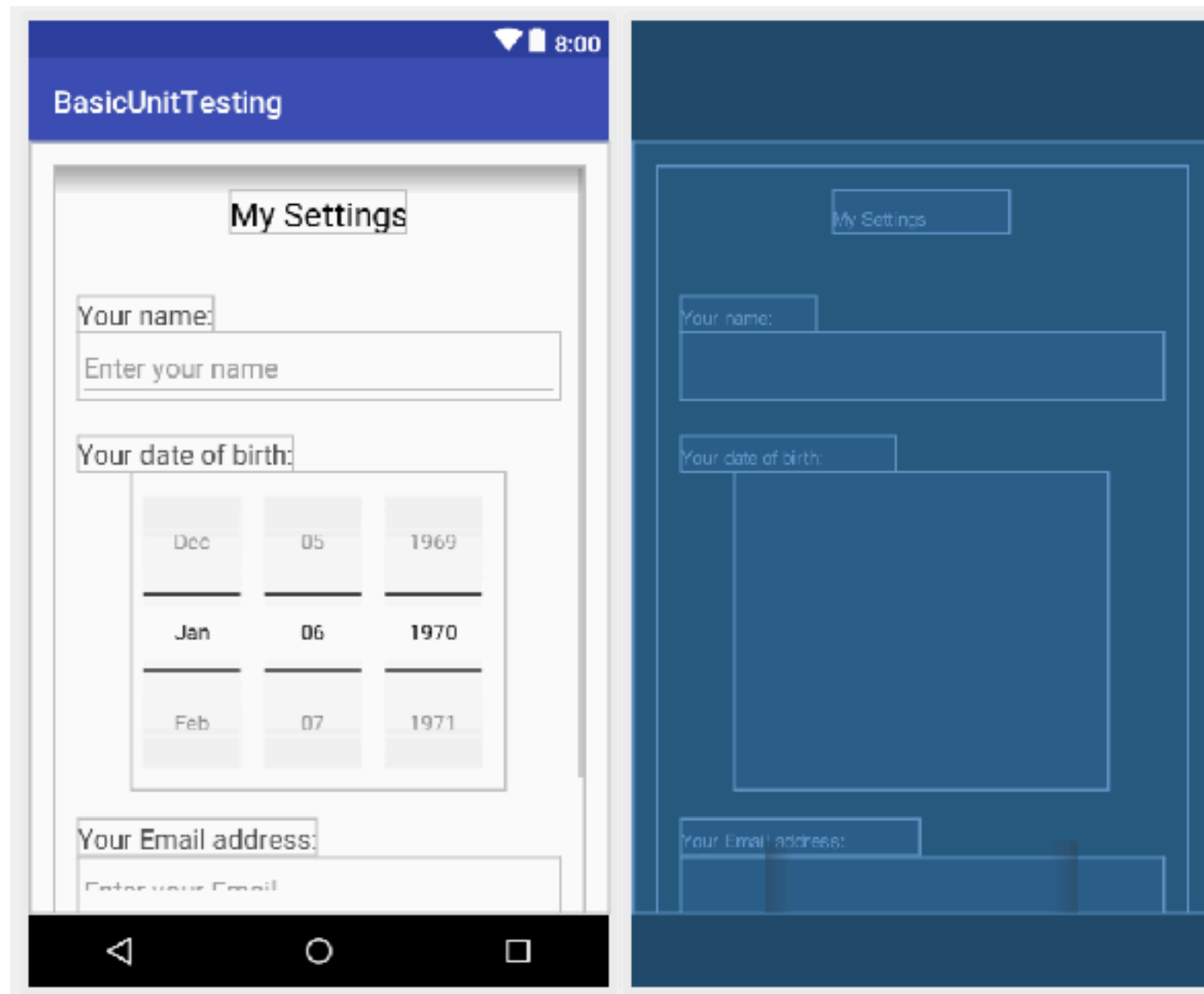
<https://github.com/up1/android-basic-testing>



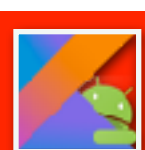
User Interface



User Interface



Let's start !!

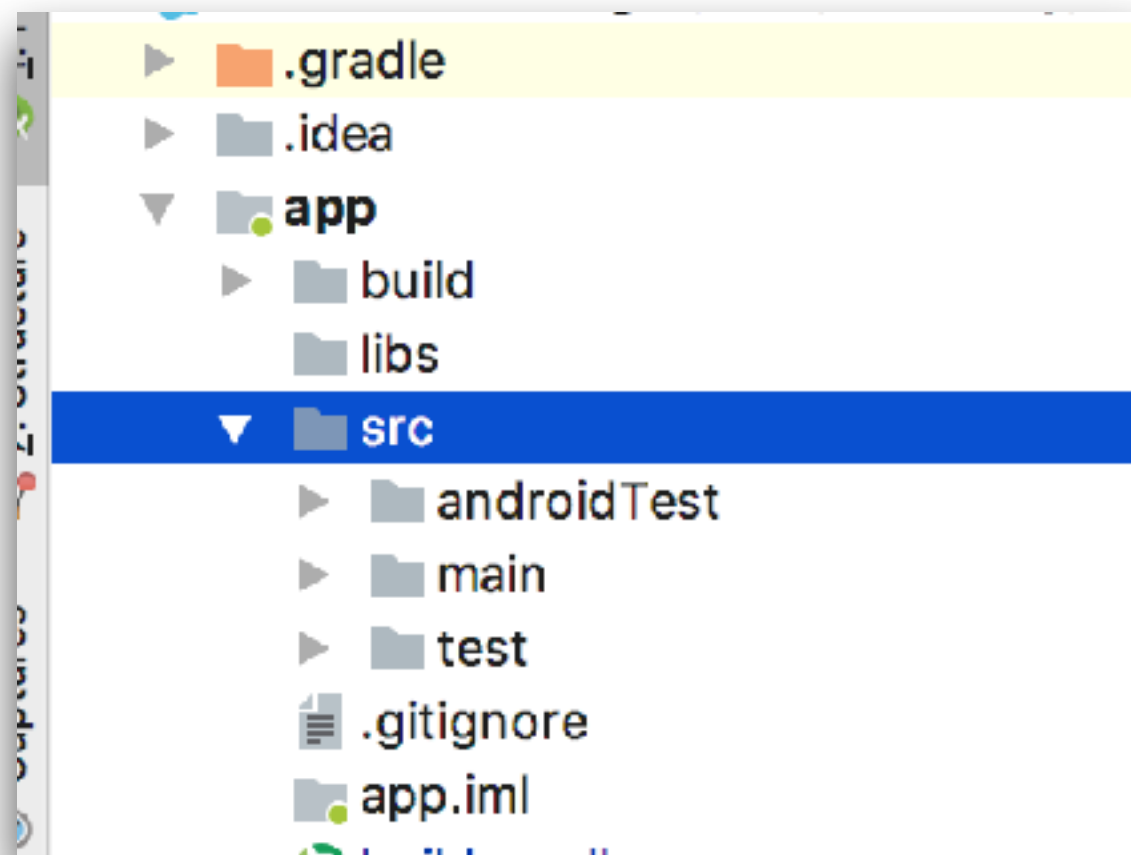


Hello Unit testing with JUnit



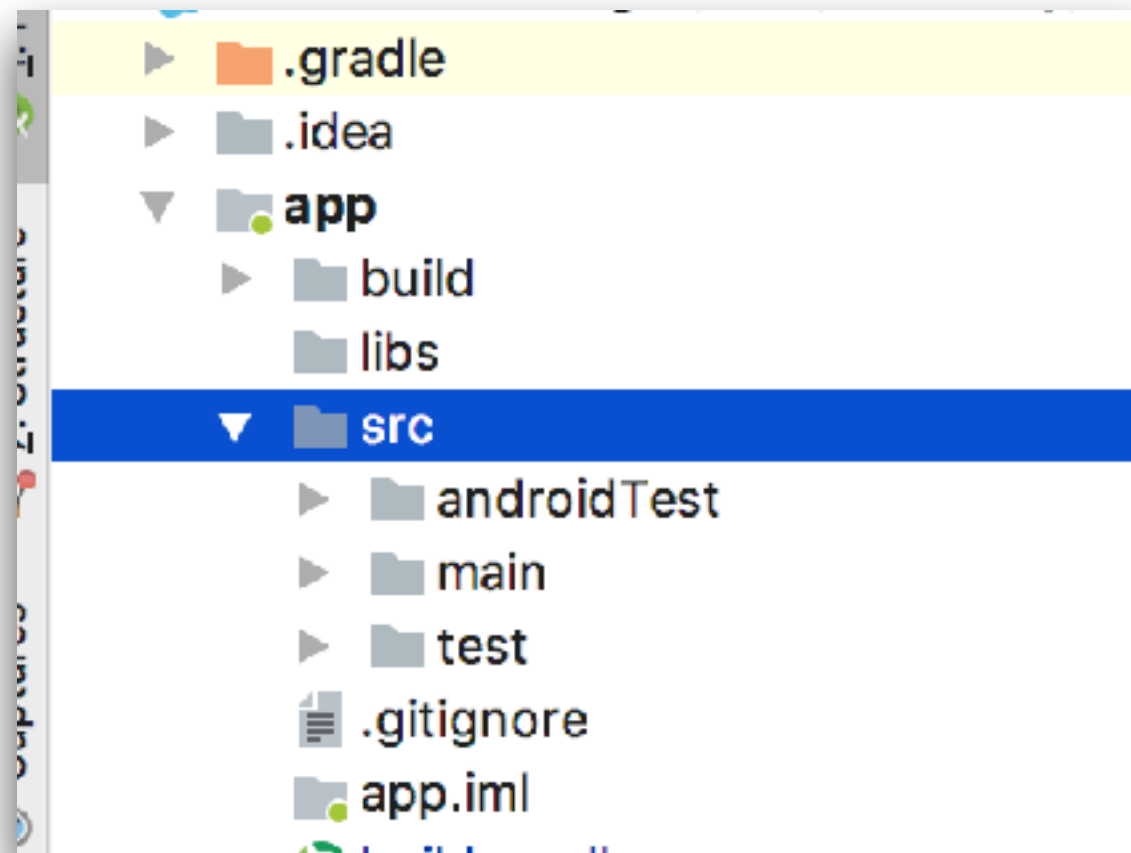
Test folder in Android project

/app/src/androidTesting
/app/src/test



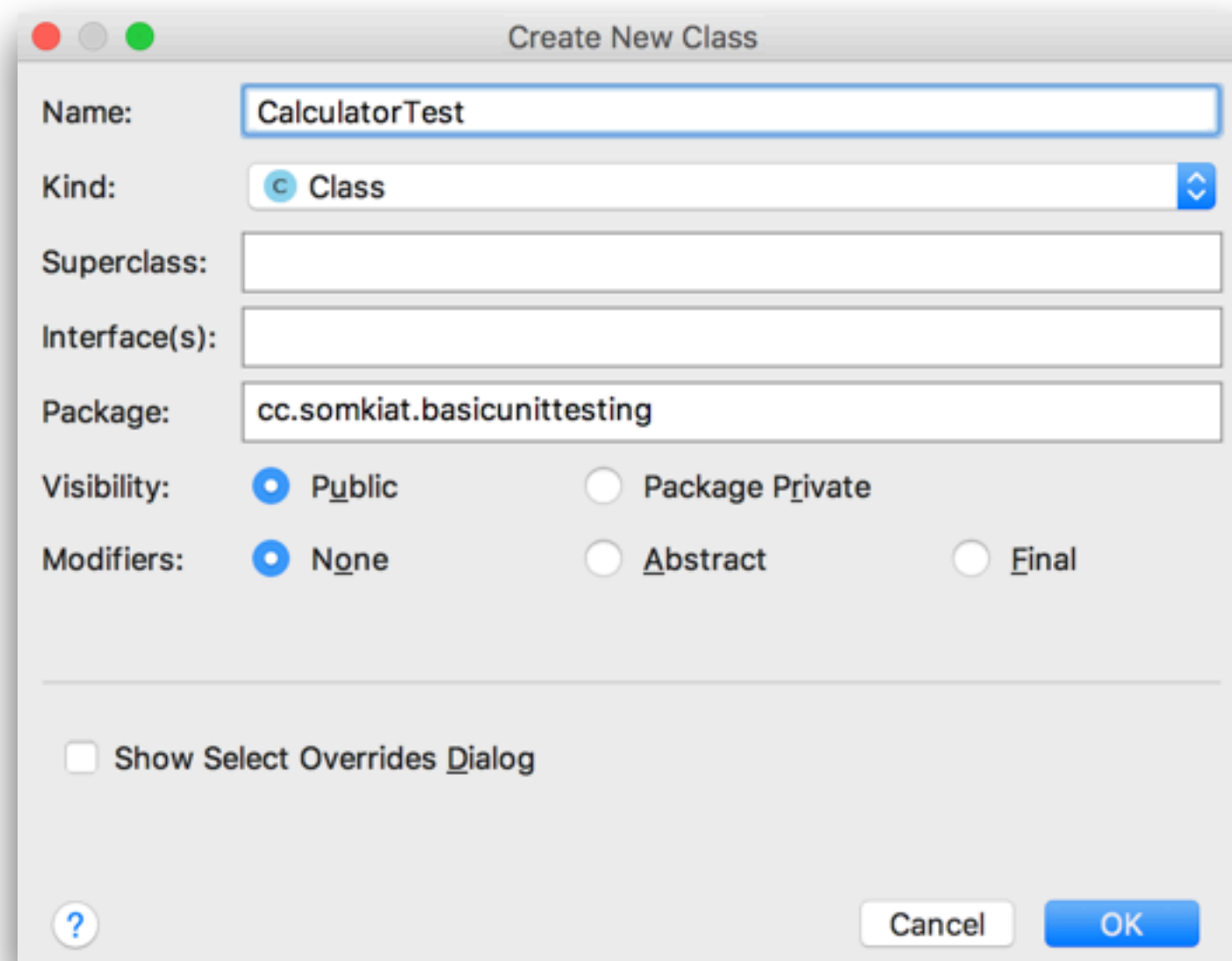
Unit testing

/app/src/androidTesting
/app/src/test



Create new Test class

/app/src/test/java/<package>/
CalculatorTest.java



Create New Class

Name:

Kind: Class

Superclass:

Interface(s):

Package:

Visibility: ☒ Public ☐ Package Private

Modifiers: ☒ None ☐ Abstract ☐ Final

☐ Show Select Overrides Dialog

? Cancel OK

Test case structure

```
public class CalculatorTest {  
  
    @Test  
    public void สวัสดีประเทศไทย() {  
        //Arrange  
        Hello hello = new Hello();  
        //Act  
        String actualResult = hello.say(name: "ประเทศไทย");  
  
        //Assert  
        assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);  
    }  
}
```



Test class name

```
public class CalculatorTest {
```

```
@Test
```

```
public void สวัสดีประเทศไทย() {
```

```
    //Arrange
```

```
    Hello hello = new Hello();
```

```
    //Act
```

```
    String actualResult = hello.say(name: "ประเทศไทย");
```

```
    //Assert
```

```
    assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);
```

```
}
```

```
}
```



Test case name

```
public class CalculatorTest {
```

```
@Test
```

```
public void สวัสดีประเทศไทย() {
```

```
    //Arrange
```

```
    Hello hello = new Hello();
```

```
    //Act
```

```
    String actualResult = hello.say(name: "ประเทศไทย");
```

```
    //Assert
```

```
    assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);
```

```
}
```

```
}
```



Arrange

```
public class CalculatorTest {  
  
    @Test  
    public void สวัสดีประเทศไทย() {  
        //Arrange  
        Hello hello = new Hello();  
        //Act  
        String actualResult = hello.say(name: "ประเทศไทย");  
  
        //Assert  
        assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);  
    }  
}
```



Act

```
public class CalculatorTest {  
  
    @Test  
    public void สวัสดีประเทศไทย() {  
        //Arrange  
        Hello hello = new Hello();  
        //Act  
        String actualResult = hello.say(name: "ประเทศไทย");  
  
        //Assert  
        assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);  
    }  
}
```

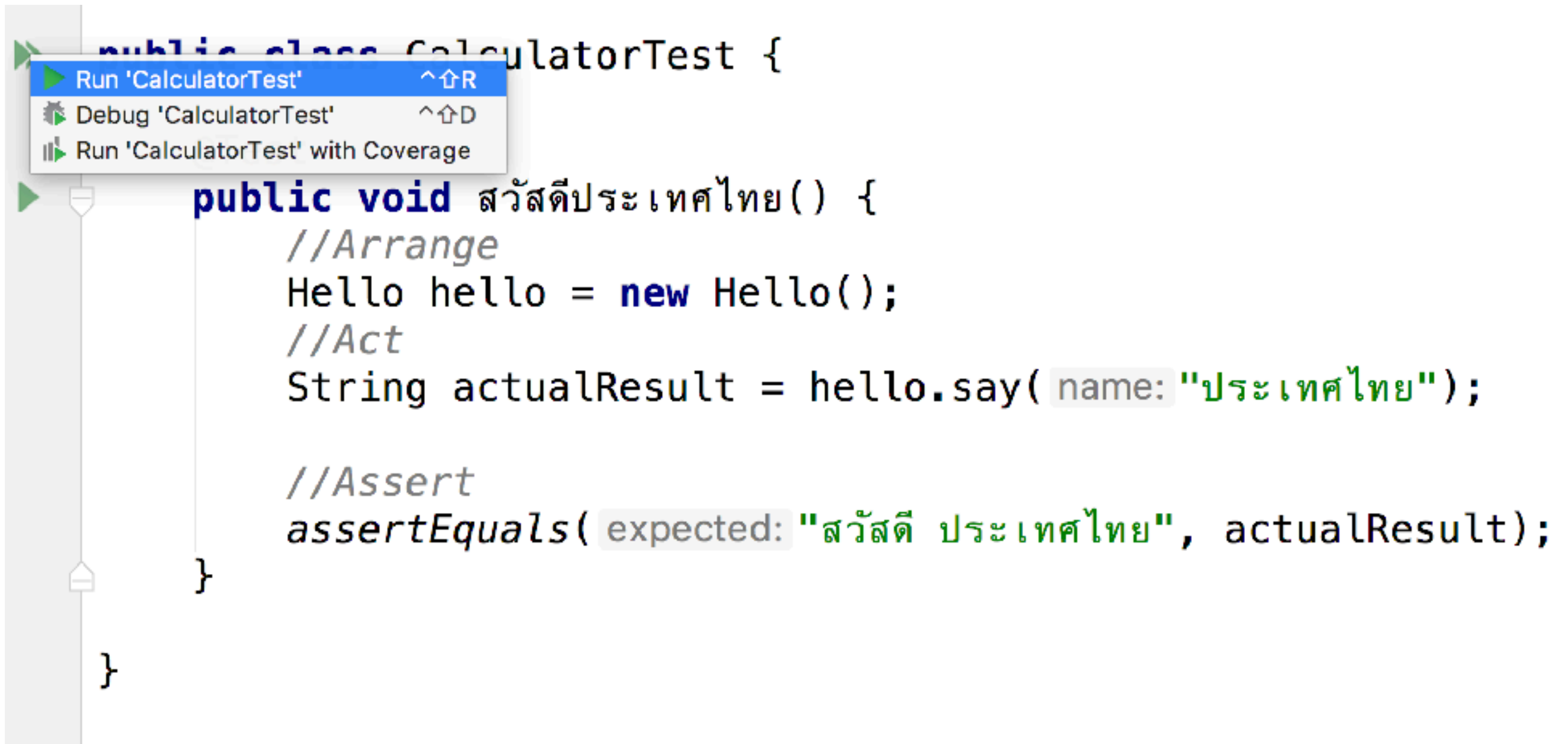


Assert

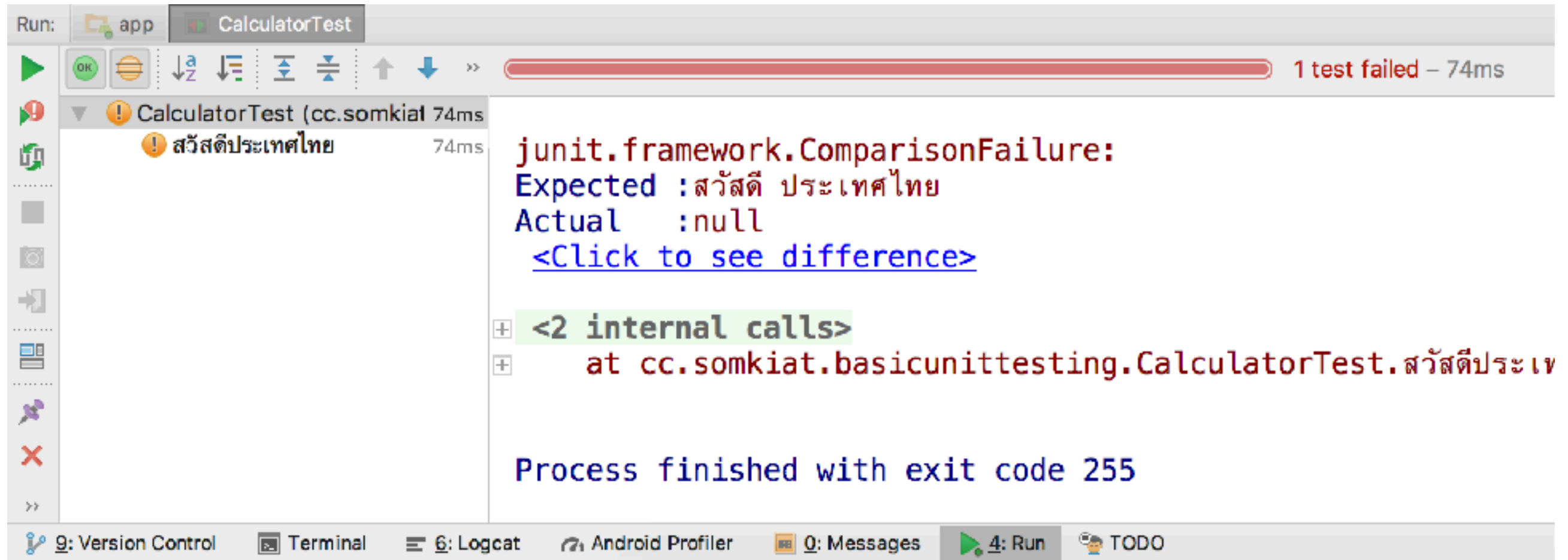
```
public class CalculatorTest {  
  
    @Test  
    public void สวัสดีประเทศไทย() {  
        //Arrange  
        Hello hello = new Hello();  
        //Act  
        String actualResult = hello.say(name: "ประเทศไทย");  
  
        //Assert  
        assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);  
    }  
}
```



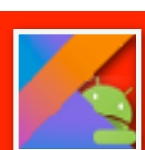
Run test !!



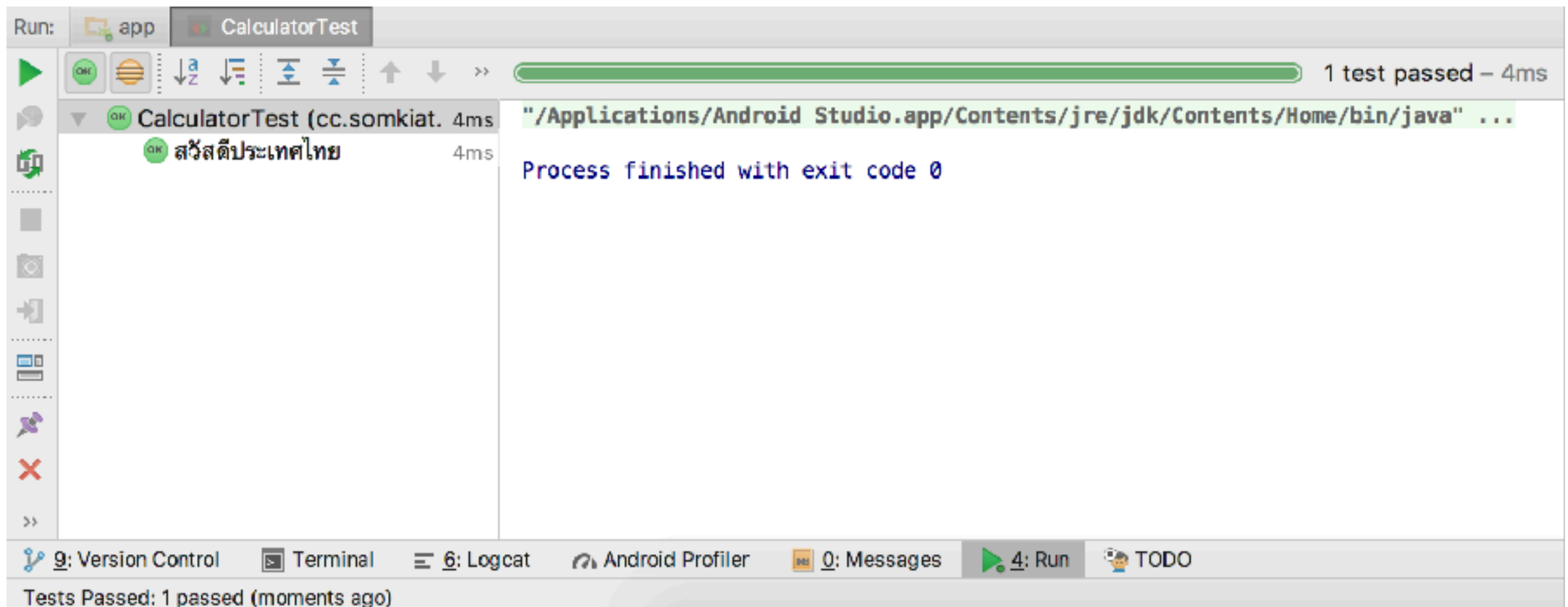
Failure !!



Do it by yourself



Pass



Run test in command line

\$gradlew tasks

\$gradlew testDebugUnitTest



Test result in html

/app/build/reports/tests/testDebugUnitTest

Test Summary

2
tests

0
failures

0
ignored

0.015s
duration

100%
successful

Packages

Classes

Package	Tests	Failures	Ignored	Duration	Success rate
cc.somkiat.basicunittesting	2	0	0	0.015s	100%



Test result in html

/app/build/reports/tests/testDebugUnitTest

Package cc.somkiat.basicunittesting

[all](#) > cc.somkiat.basicunittesting

2
tests

0
failures

0
ignored

0.015s
duration

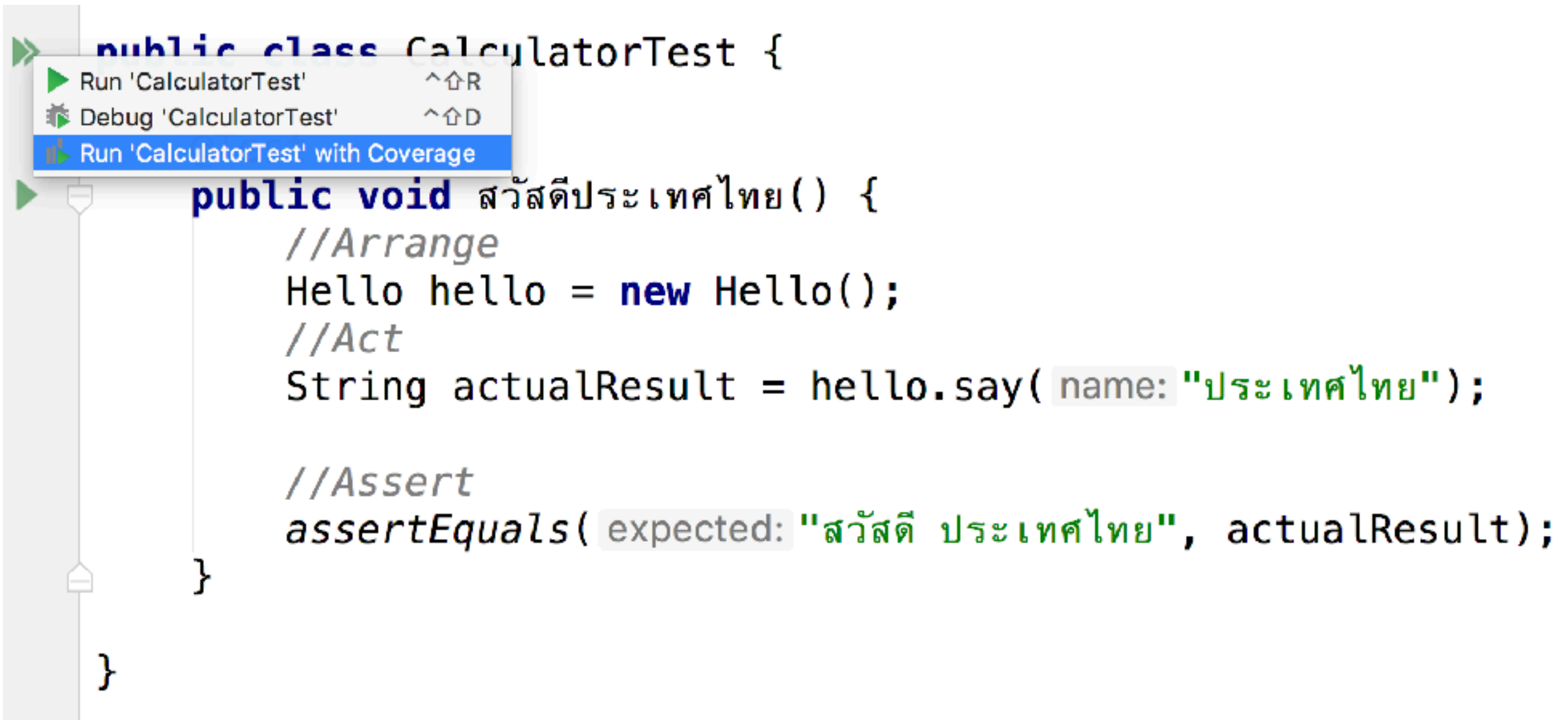
100%
successful

Classes

Class	Tests	Failures	Ignored	Duration	Success rate
CalculatorTest	1	0	0	0.015s	100%
ExampleUnitTest	1	0	0	0s	100%



Run test with coverage !!



The screenshot shows an IDE window with a Java file named `CalculatorTest.java`. A context menu is open over the first line of the code, `public class CalculatorTest {`. The menu contains three options: `Run 'CalculatorTest'` (with keyboard shortcut `^⇧R`), `Debug 'CalculatorTest'` (with keyboard shortcut `^⇧D`), and `Run 'CalculatorTest' with Coverage` (highlighted in blue). The code in the background is as follows:

```
public class CalculatorTest {  
    public void สวัสดีประเทศไทย() {  
        //Arrange  
        Hello hello = new Hello();  
        //Act  
        String actualResult = hello.say(name: "ประเทศไทย");  
  
        //Assert  
        assertEquals(expected: "สวัสดี ประเทศไทย", actualResult);  
    }  
}
```

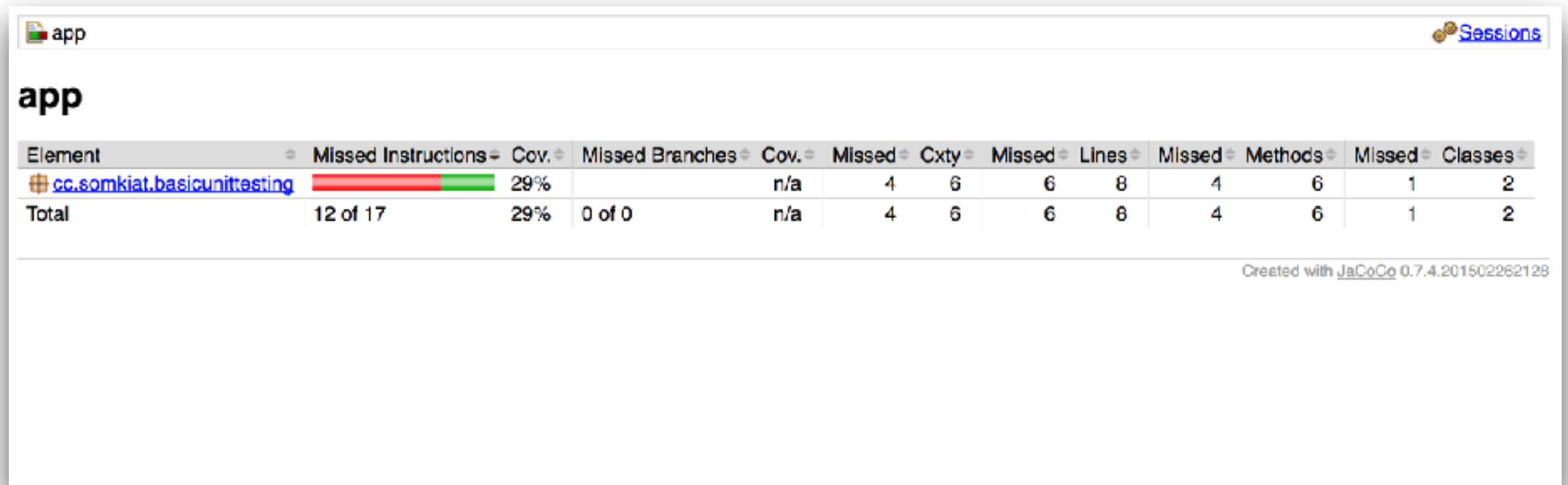
Run in command line

```
$gradlew fullCoverageReport
```



Coverage report

/app/build/reports/jacoco/fullCoverageReport/html



Goals

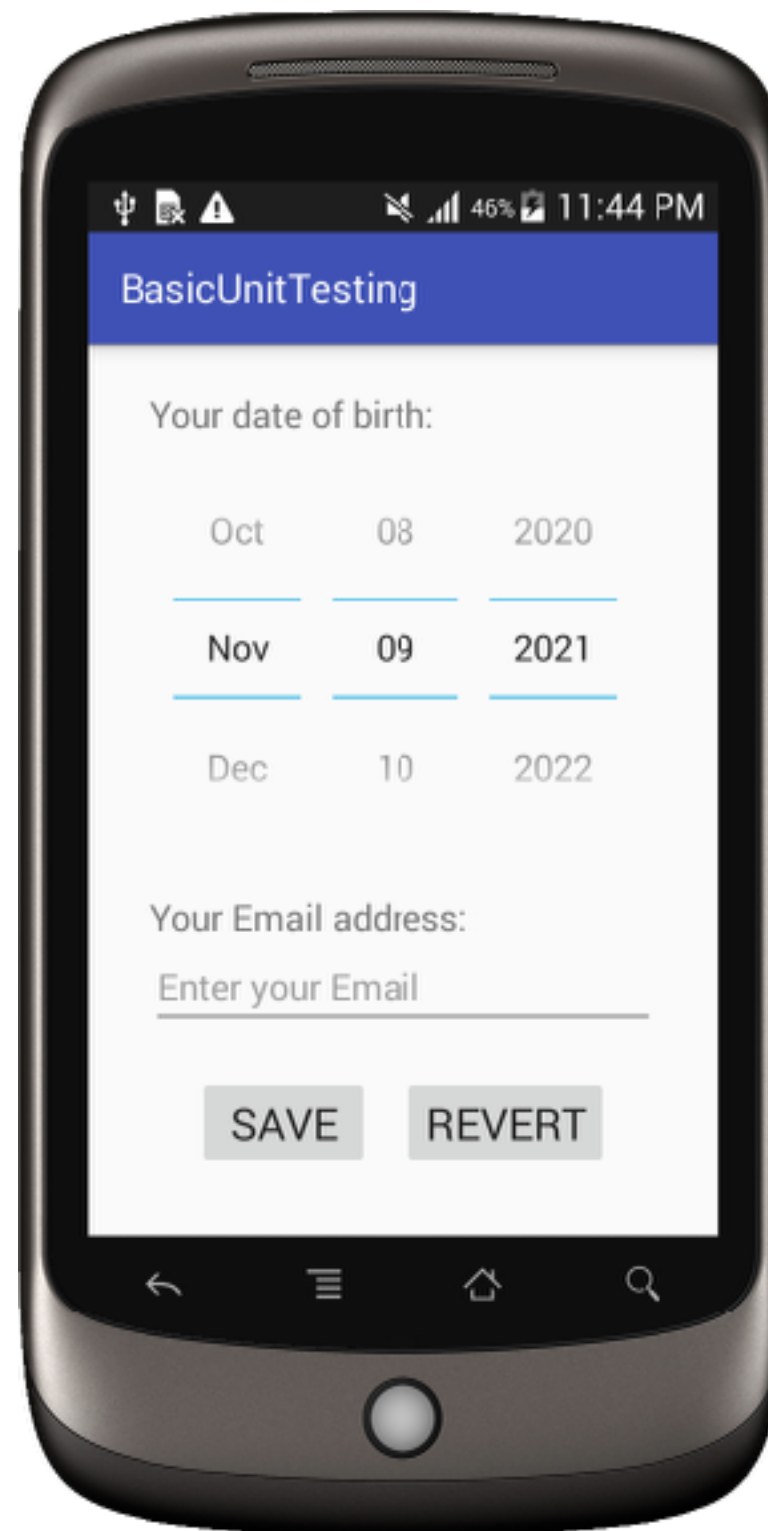
1. Unit testing all the things
2. Coverage 100%



Workshop



Workshop with Click Save Button



How to tests ?



Workshop

1. Validate name
2. Validate email



1. Validate name

Not null or Empty
Length between 2-20



2. Validate email

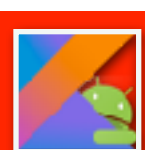
Not null or Empty name
Email pattern



How to write tests ?



Let's try !!



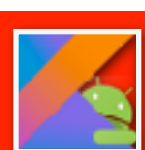
More



Monkey testing

```
$adb shell monkey -p your.package.name -v 500
```

<https://developer.android.com/studio/test/monkey.html>



How to tests ?



Mockito on next week



Homework



Homework



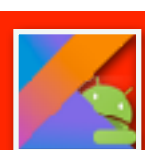
Homework

Coverage 100%

All tests pass

Monkey testing pass

Number of commit in git



Resources

<https://developer.android.com/studio/test/index.html>
<https://github.com/googlesamples/>



View

Responsible for rendering the display
Sending audio to speakers

In android, it's extend from **View class**

