

# Espresso workshop

[somkiat.cc](http://somkiat.cc)

ANDROID

บริษัท สยามชานาญกิจ จำกัด และเพื่อนพ้องน้องพี่



# ทดสอบไปทำไม ?

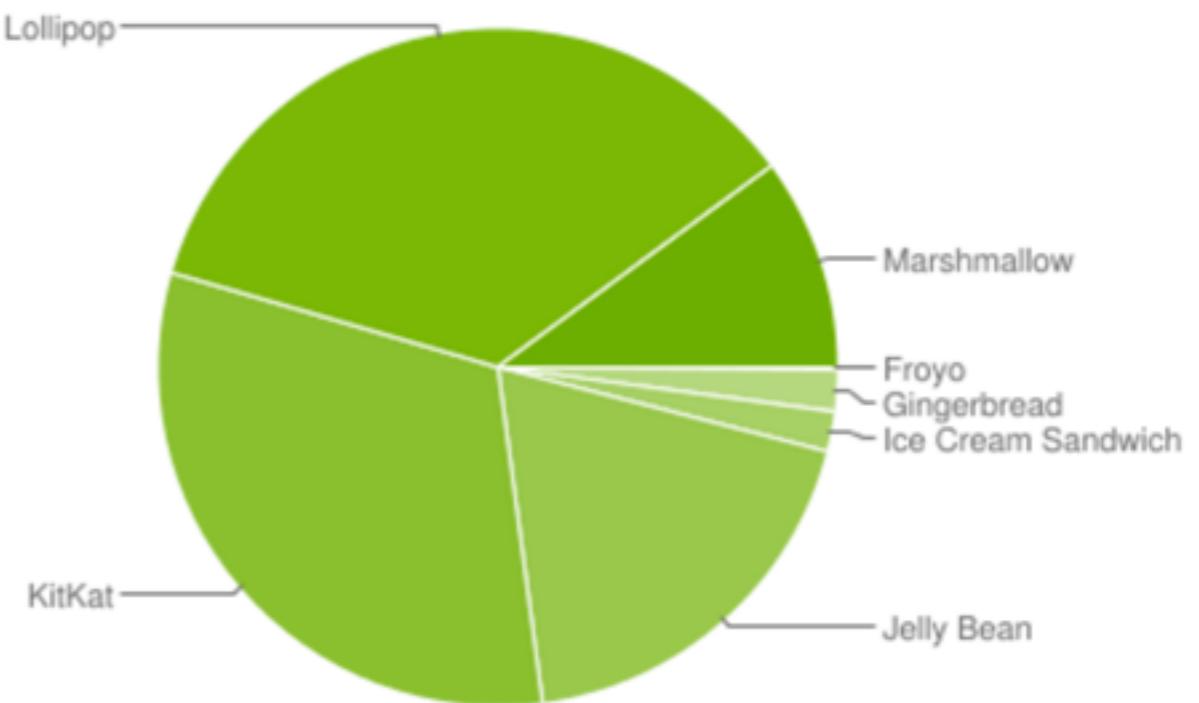




บริษัท สยามชนาญกิจ จำกัด และเพื่อนพ้องน้องพี่



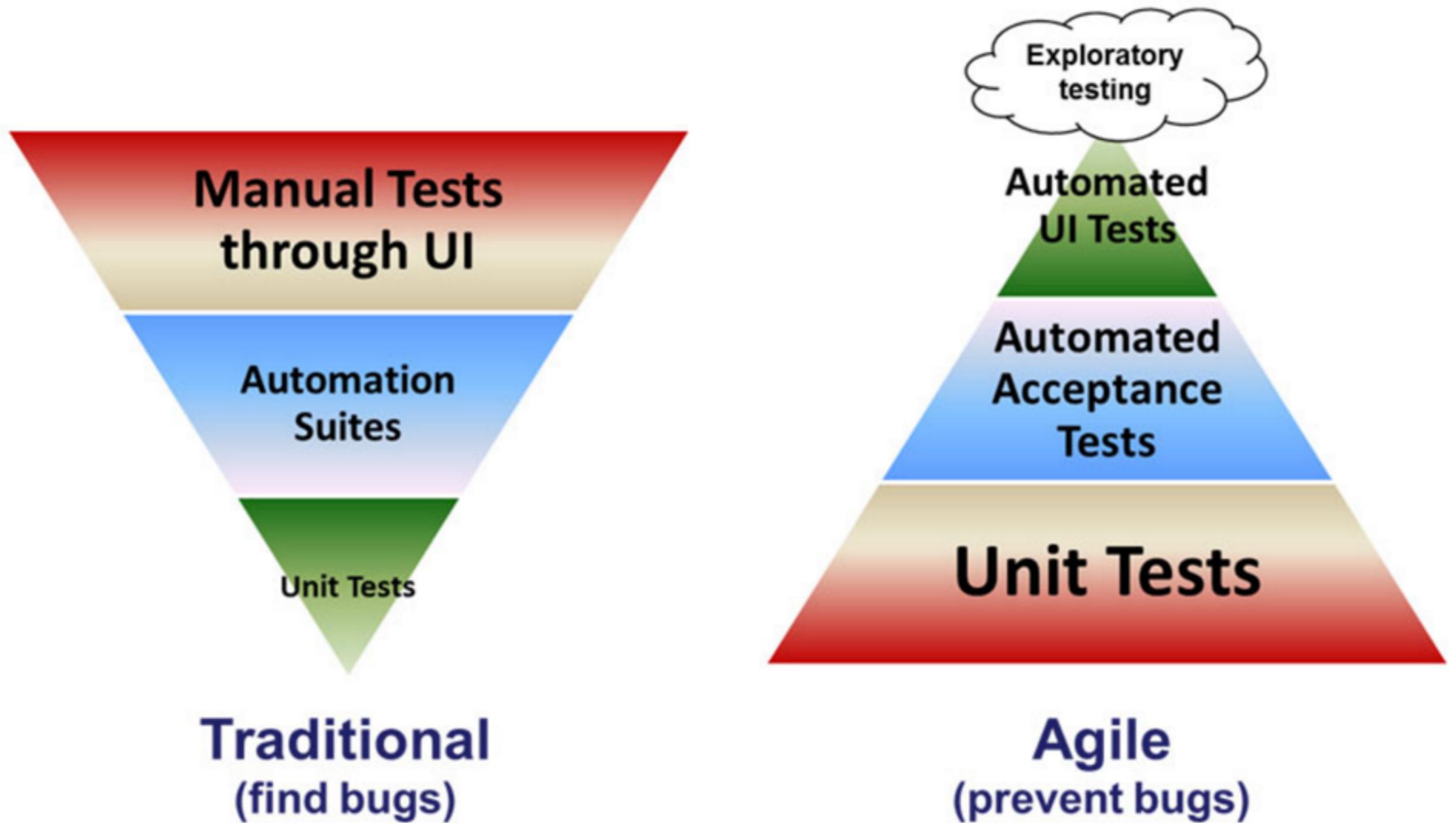
Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	2.0%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.9%
4.1.x	Jelly Bean	16	6.8%
4.2.x		17	9.4%
4.3		18	2.7%
4.4	KitKat	19	31.6%
5.0	Lollipop	21	15.4%
5.1		22	20.0%
6.0	Marshmallow	23	10.1%

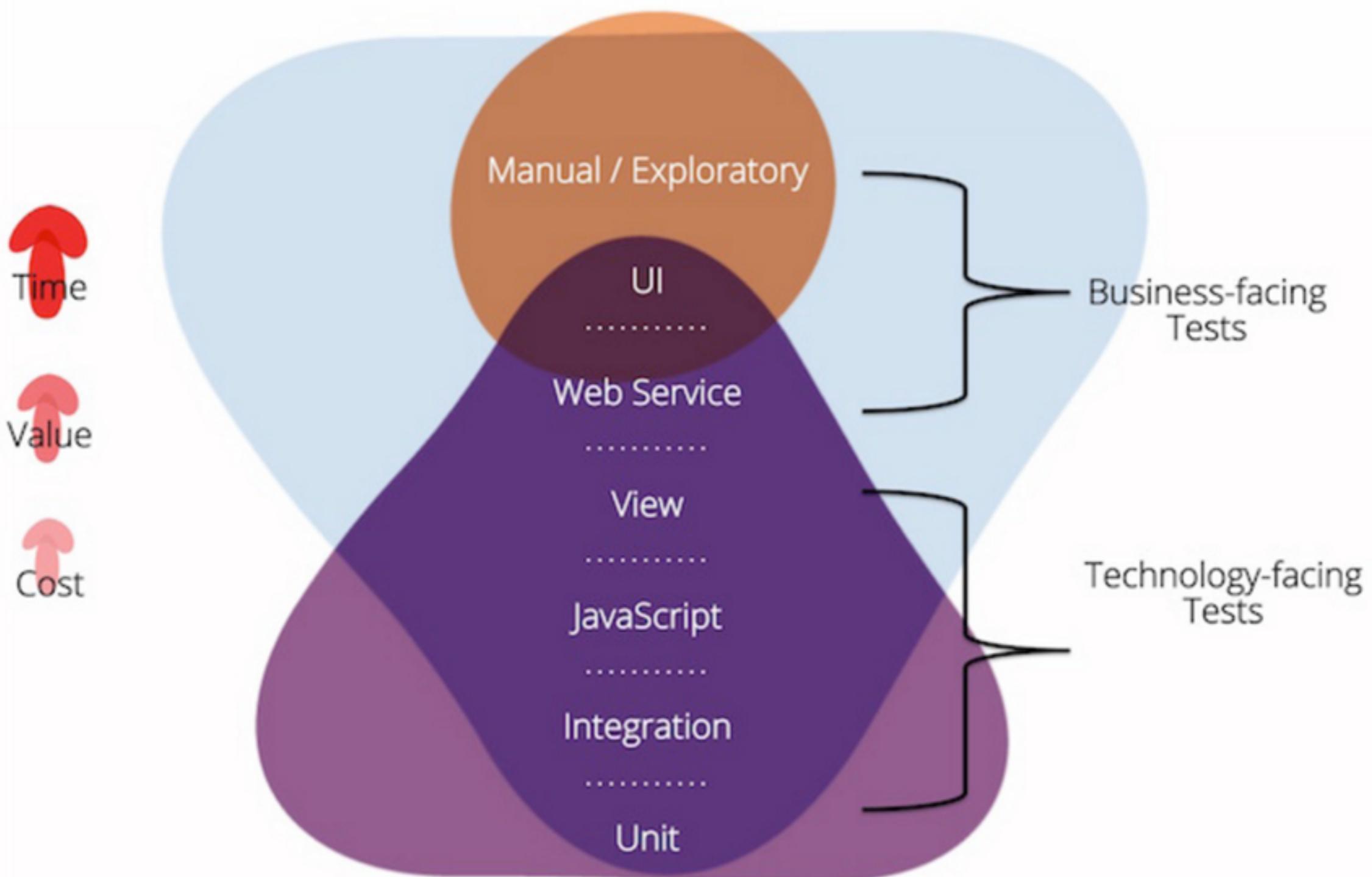


*Data collected during a 7-day period ending on June 6, 2016.*

*Any versions with less than 0.1% distribution are not shown.*





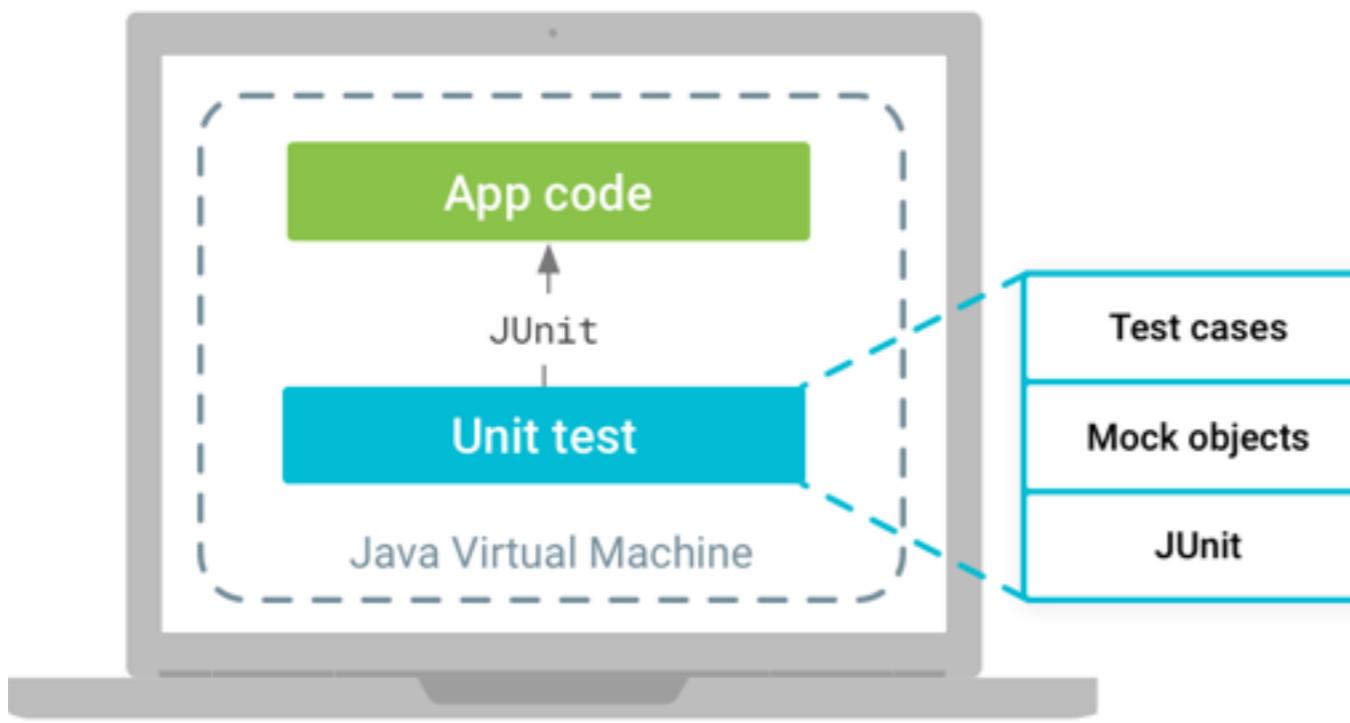


# การทดสอบสำหรับ Android

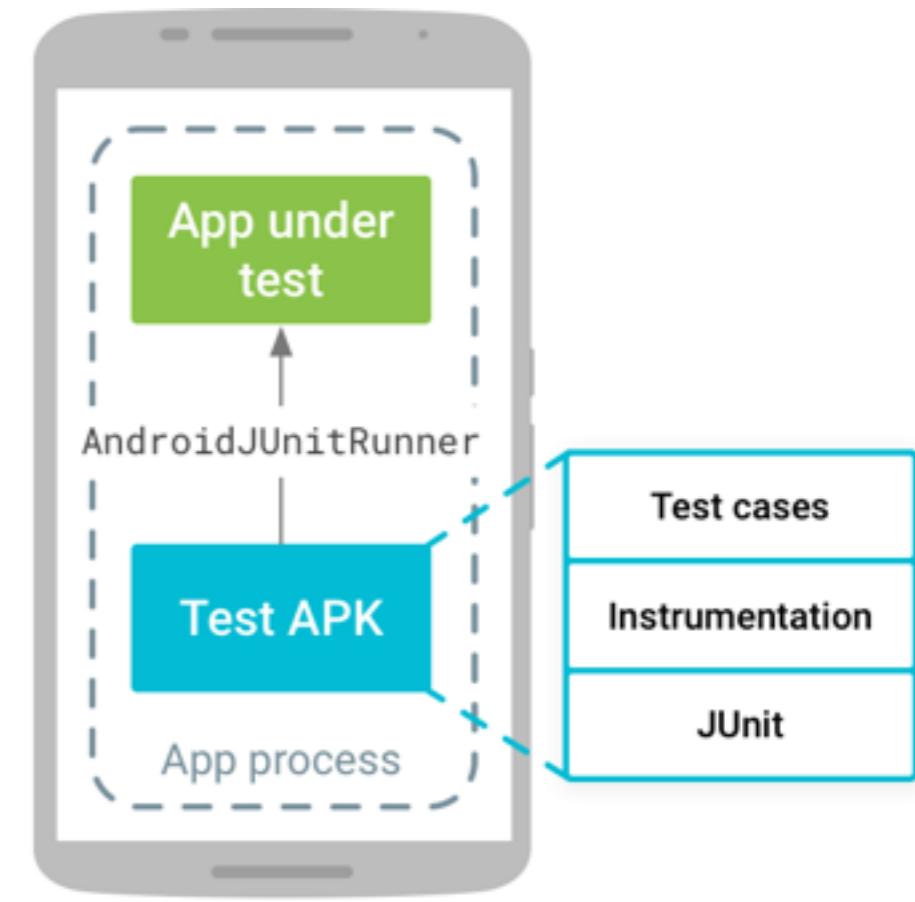
1. Unit testing
2. UI testing



# การทดสอบสำหรับ Android



Local unit test  
`src/test/java/`



Instrumented test  
`src/androidTest/java/`

<https://developer.android.com/training/testing/start/index.html>



# Unit testing

Unit  
Mockito  
PowerMock



# UI testing

Instrumentation test

Espresso

UI Automator

Calabash, Appium, Robotium, Robolectric

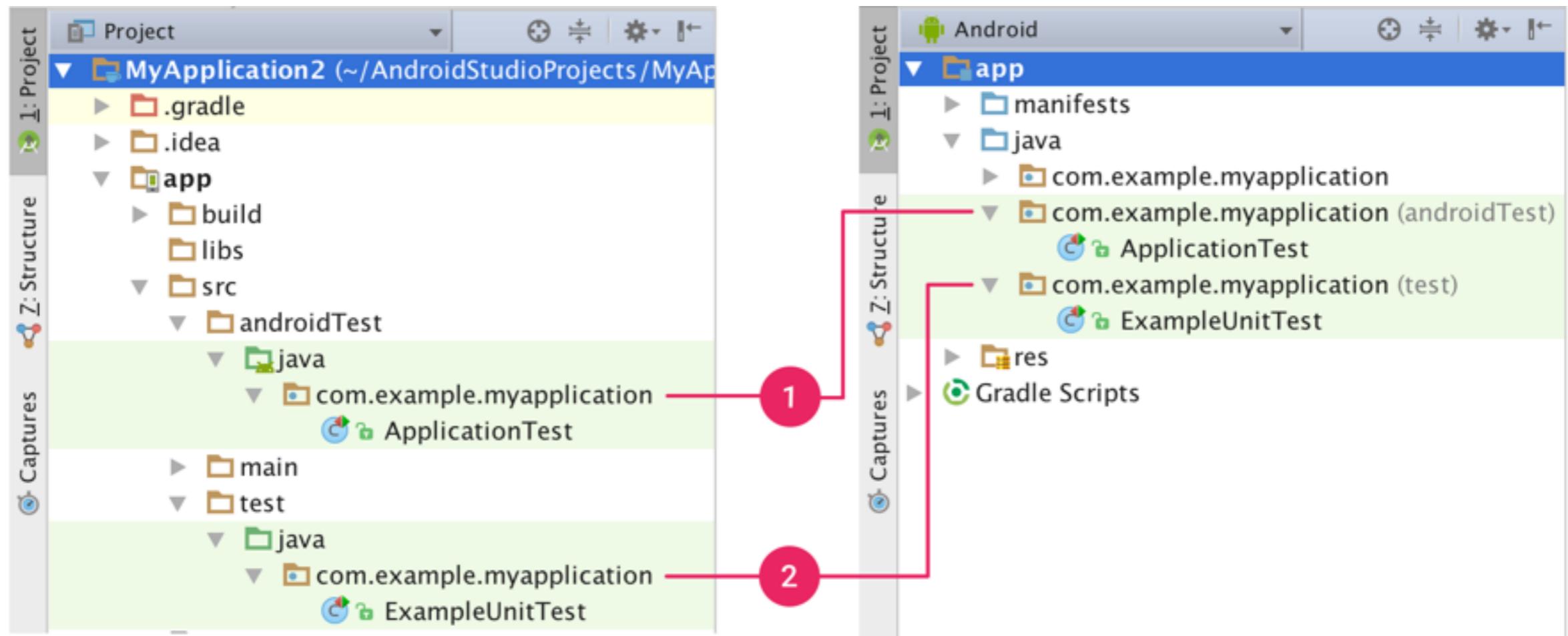




<https://google.github.io/android-testing-support-library/>



บริษัท สยามชนาญกิจ จำกัด และเพื่อนพ้องน้องพี่



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



# Build High-Quality App



บริษัท สยามชานาญกิจ จำกัด และเพื่อนพ้องน้องพี่

# Espresso คืออะไร ?

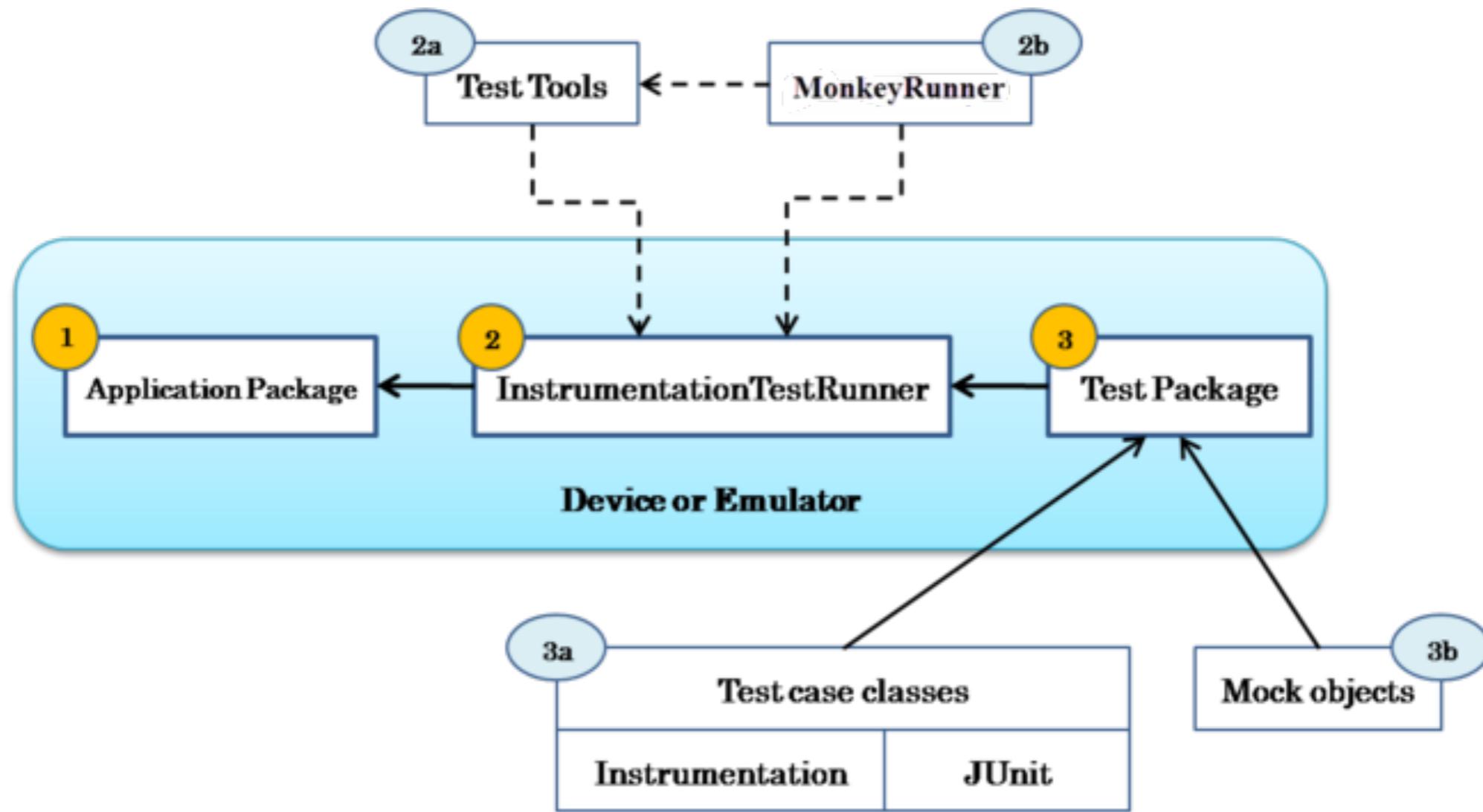


**Automated UI testing**

**Simulate user interaction**

**Synchronization of test actions with UI**





**Android**



**Instrumentation**



**AndroidJUnitRunner**

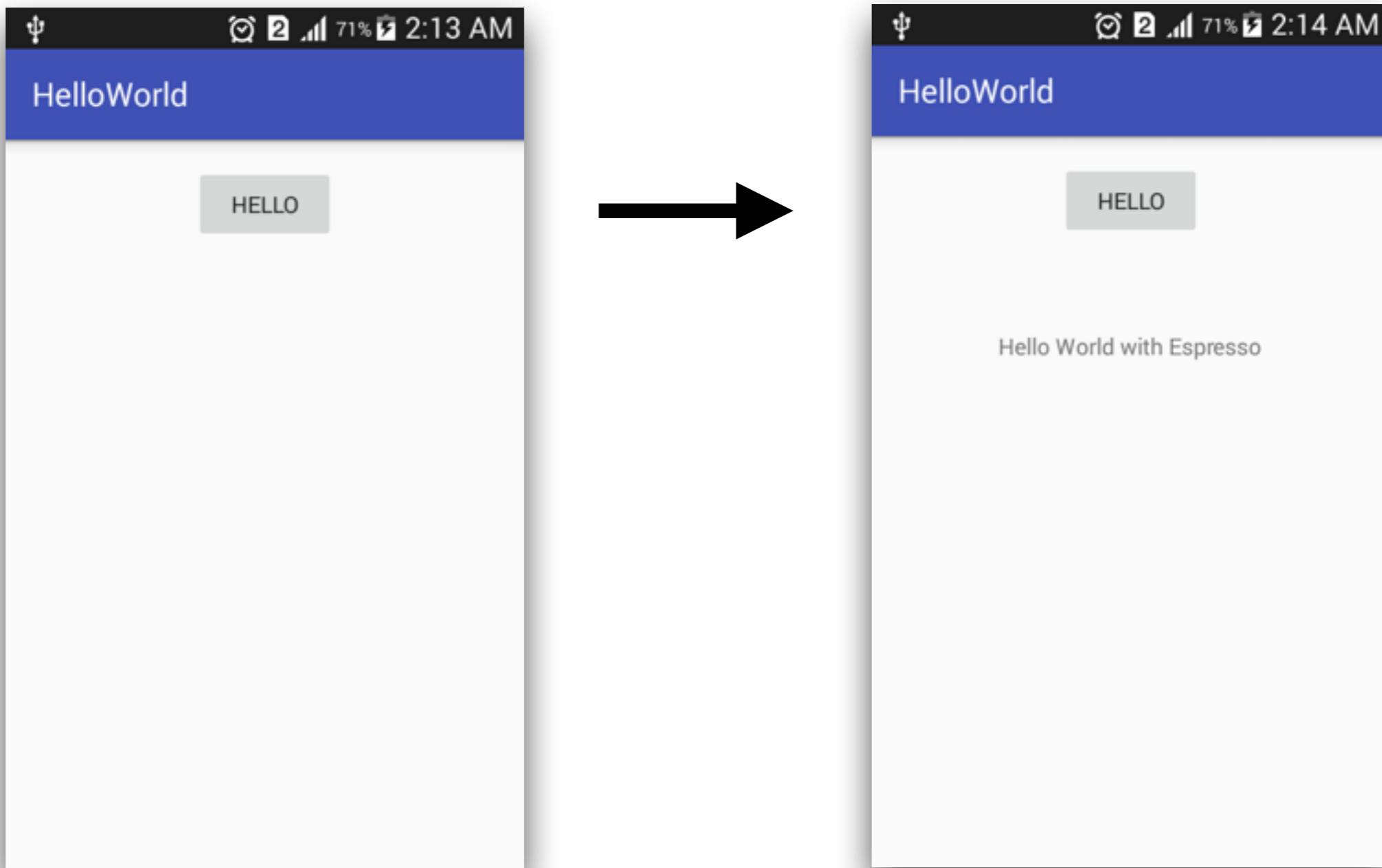


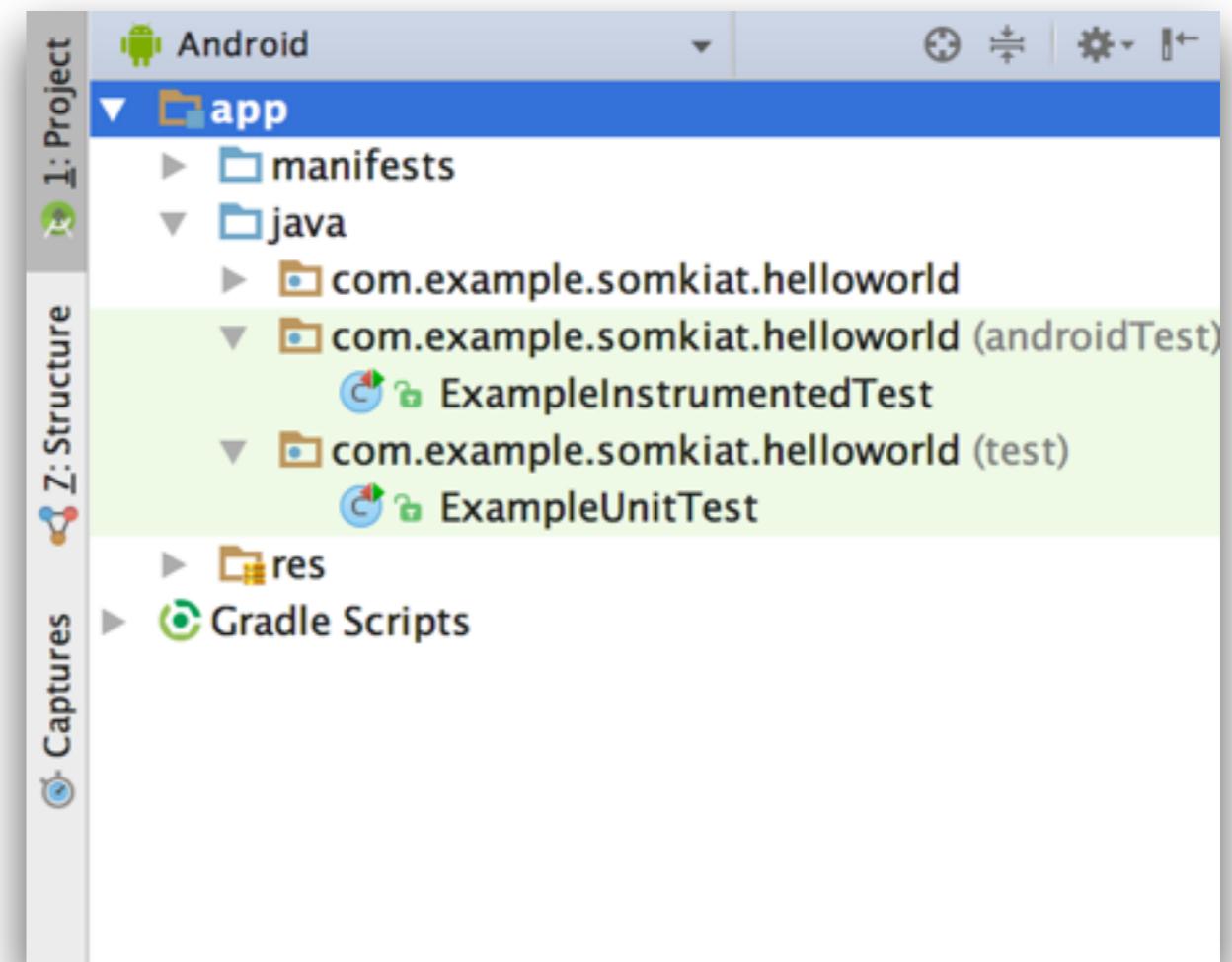
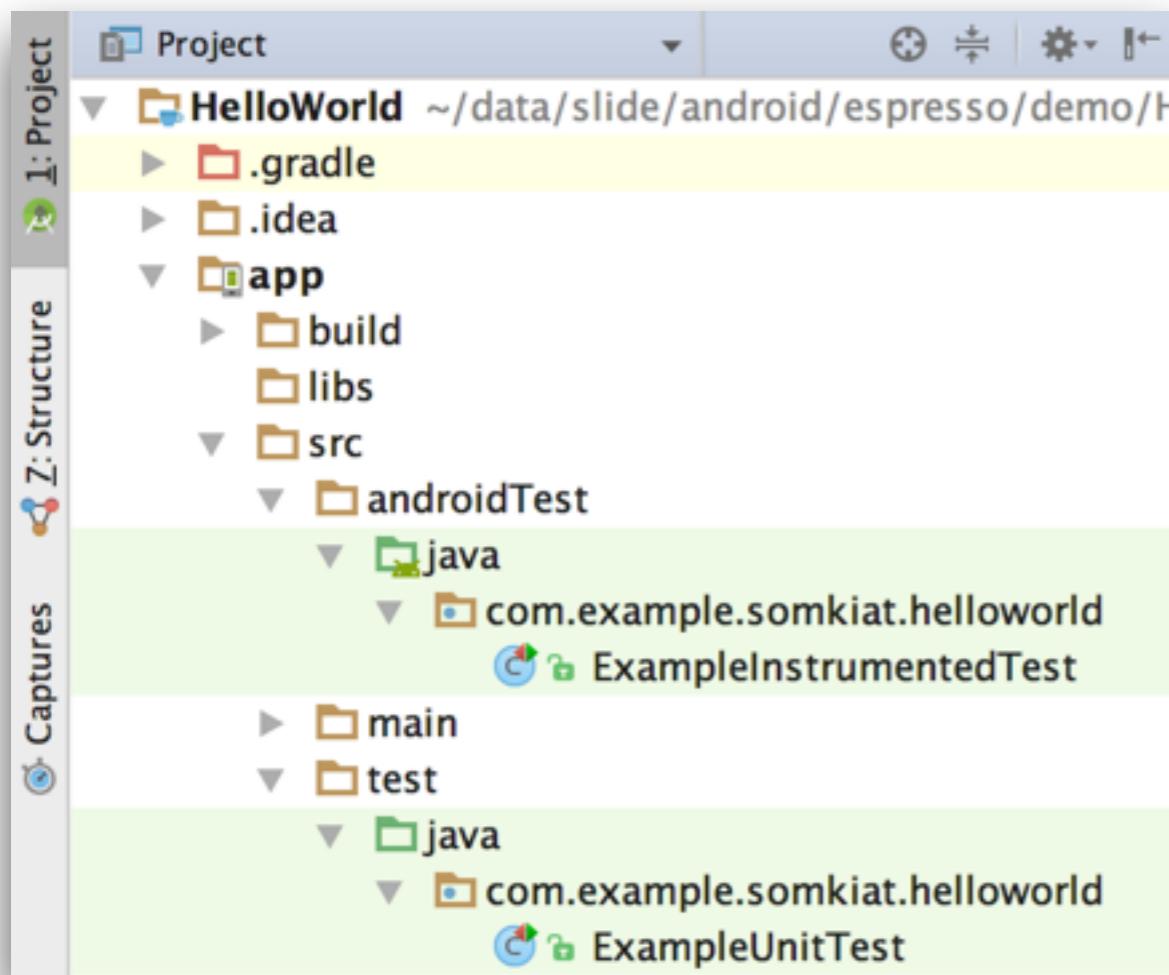
# Hello World



บริษัท สยามชนาญกิจ จำกัด และเพื่อนพ้องน้องพี่

# Hello World





# Configuration

```
android {  
    compileSdkVersion 23  
    buildToolsVersion "23.0.3"  
    defaultConfig {  
        applicationId "com.example.somkiat.helloworld"  
        minSdkVersion 15  
        targetSdkVersion 23  
        versionCode 1  
        versionName "1.0"  
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"  
    }  
}
```



# Configuration

```
android {  
    compileSdkVersion 23  
    buildToolsVersion "23.0.3"  
    defaultConfig {  
        applicationId "com.example.somkiat.helloworld"  
        minSdkVersion 15  
        targetSdkVersion 23  
        versionCode 1  
        versionName "1.0"  
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"  
    }  
}
```

Default in Android Studio 2.2



# Dependencies

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {  
        exclude group: 'com.android.support', module: 'support-annotations'  
    })  
    compile 'com.android.support:appcompat-v7:23.4.0'  
    compile 'com.android.support.constraint:constraint-layout:1.0.0-alpha3'  
    testCompile 'junit:junit:4.12'  
}
```



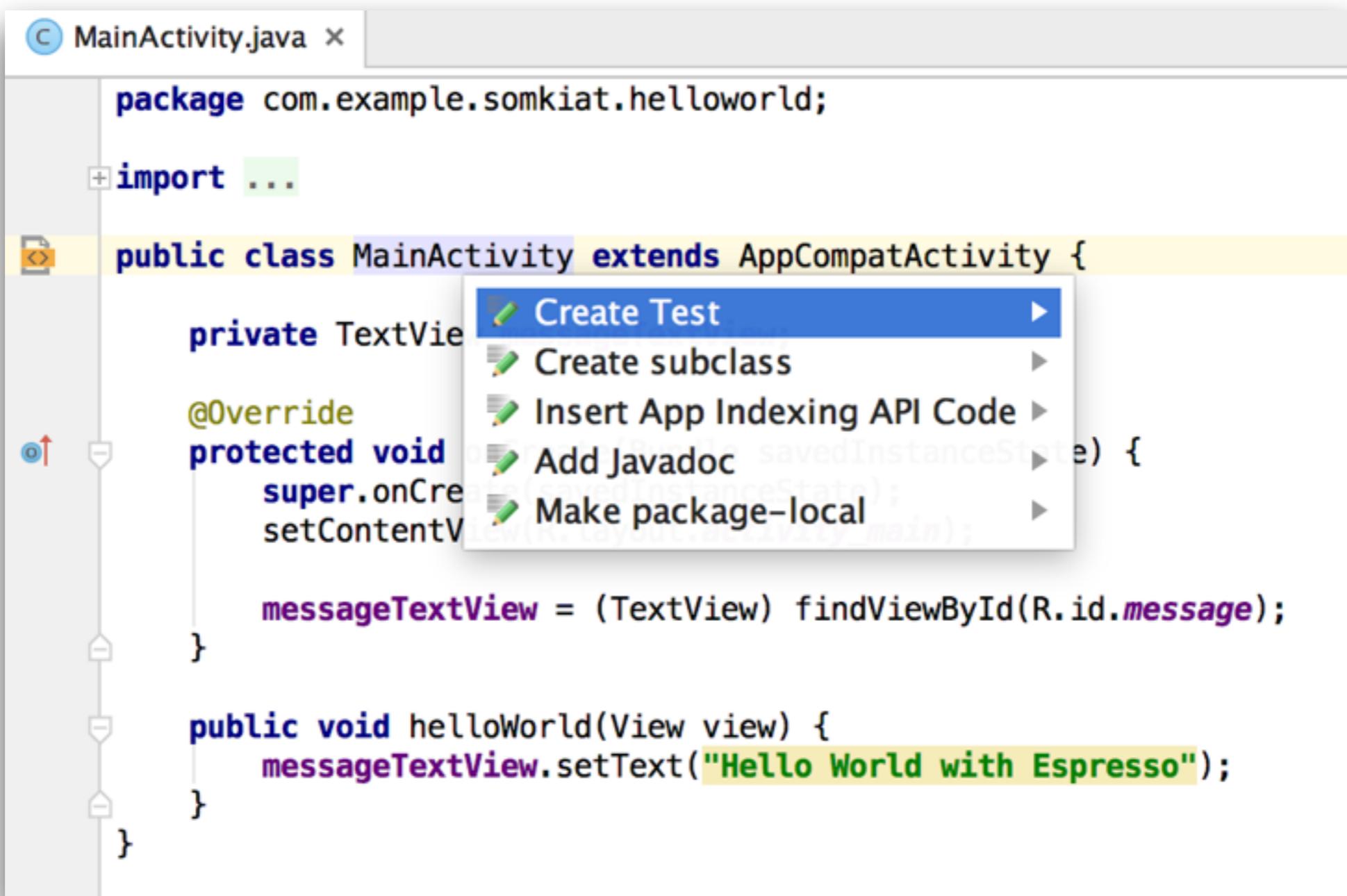
# Dependencies

```
dependencies {  
    compile fileTree(dir: 'libs', include: ['*.jar'])  
    androidTestCompile('com.android.support.test.espresso:espresso-core:2.2.2', {  
        exclude group: 'com.android.support', module: 'support-annotations'  
    })  
    compile 'com.android.support:appcompat-v7:23.4.0'  
    compile 'com.android.support.constraint:constraint-layout:1.0.0-alpha3'  
    testCompile 'junit:junit:4.12'  
}
```

Default in Android Studio 2.2



# Write first test



The screenshot shows the code editor for `MainActivity.java`. The class definition is highlighted. A context menu is open at the end of the class declaration line, with the 'Create Test' option selected. The code itself contains basic boilerplate for a UI application.

```
package com.example.somkiat.helloworld;

import ...

public class MainActivity extends AppCompatActivity {
    private TextView messageTextView;

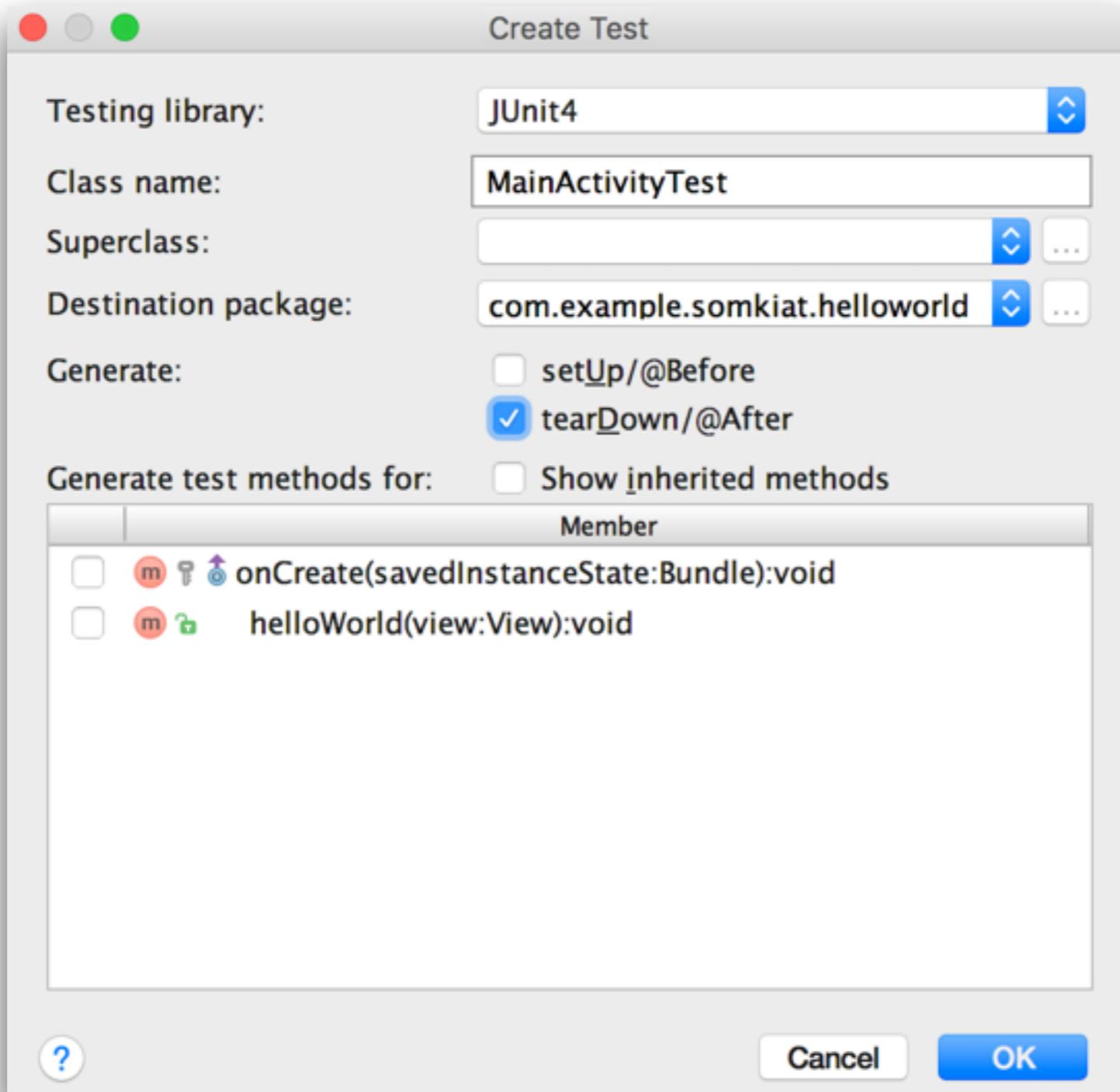
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        messageTextView = (TextView) findViewById(R.id.message);
    }

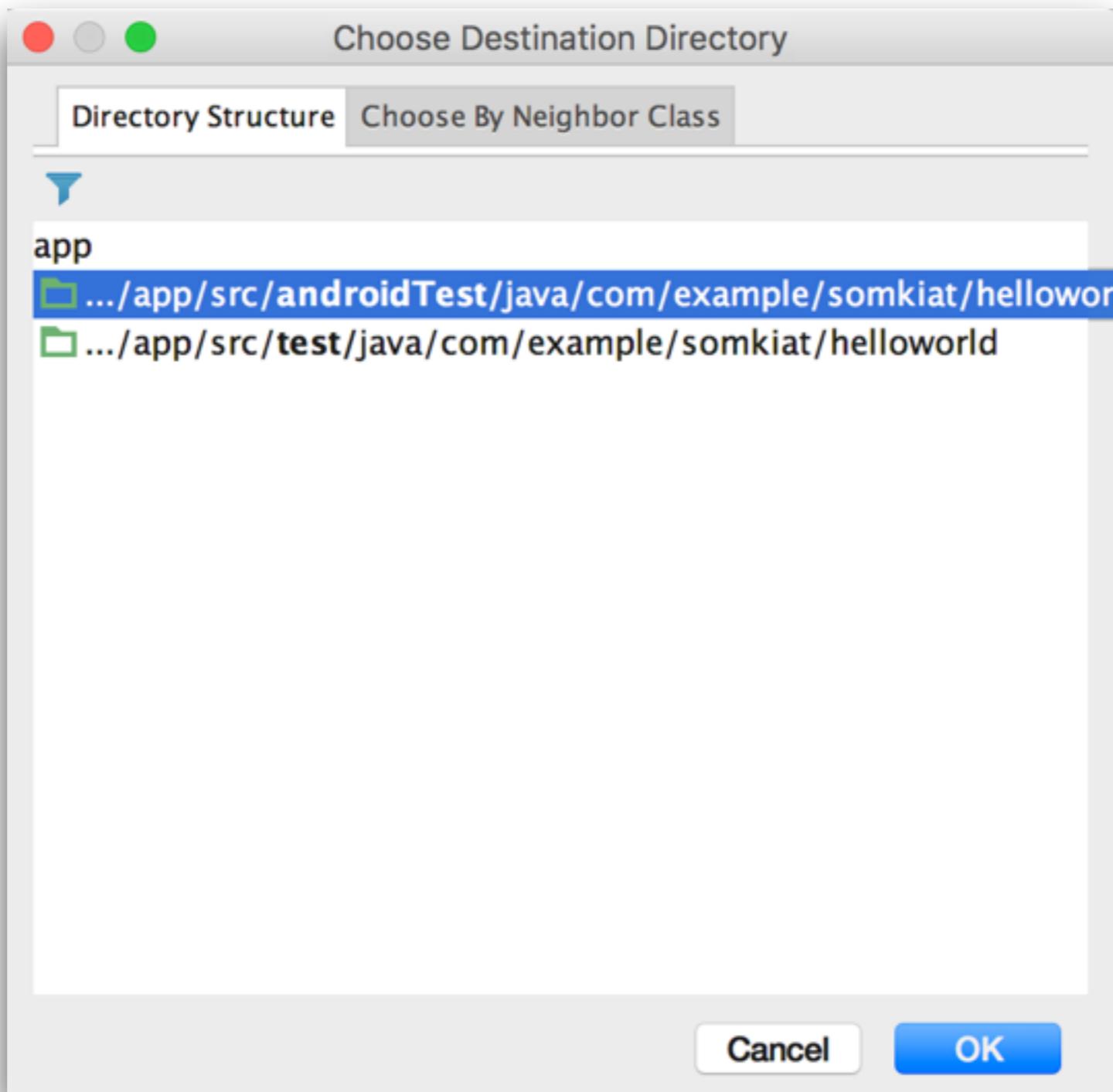
    public void helloWorld(View view) {
        messageTextView.setText("Hello World with Espresso");
    }
}
```



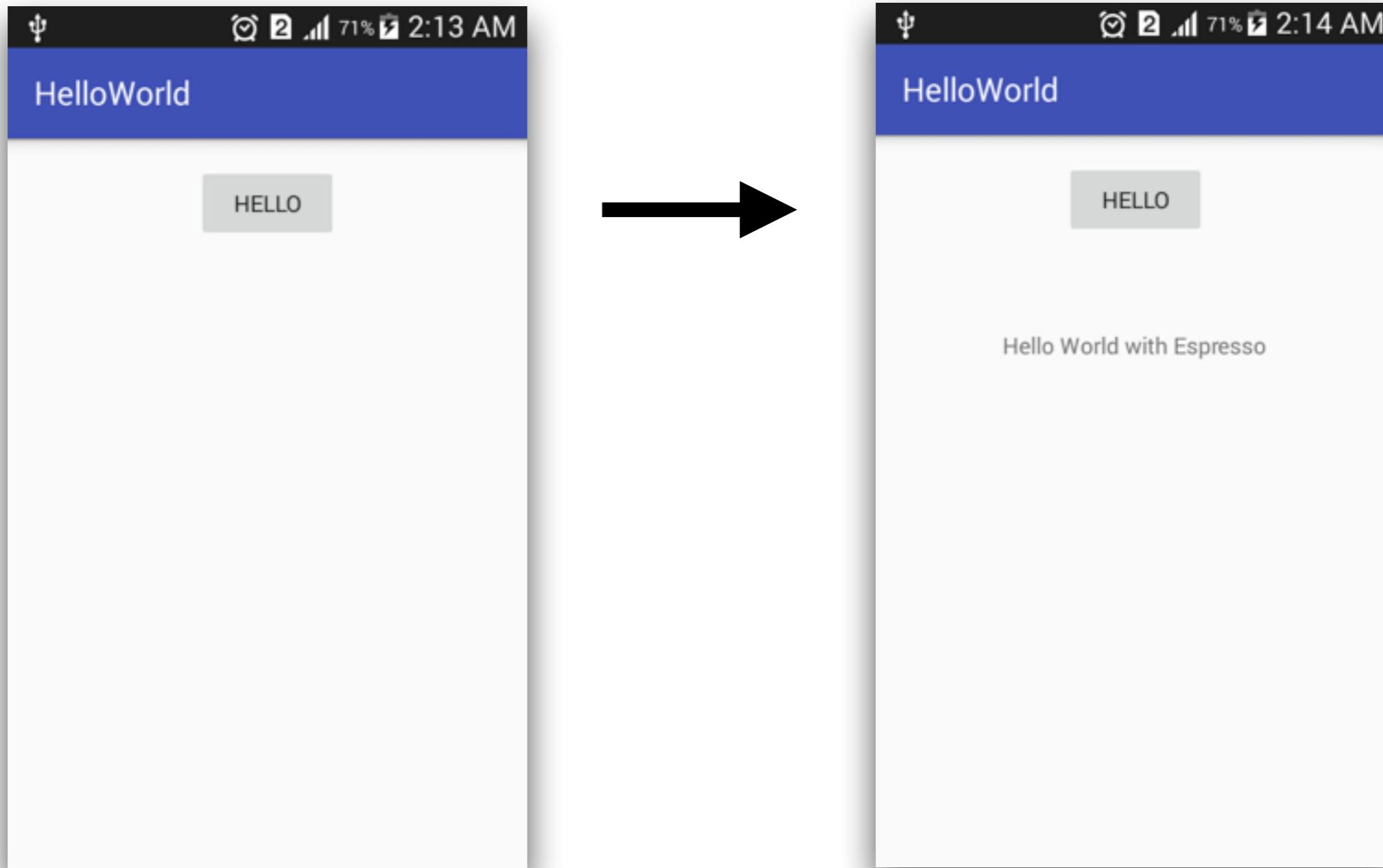
# Write first test



# Write first test



# Write first test



# Step 1 :: Find Hello Button

```
public class MainActivityTest {  
  
    @Rule  
    public ActivityTestRule<MainActivity> activityTestRule = new ActivityTestRule(MainActivity.class);  
  
    @Test  
    public void findHelloButtonInView() {  
        onView(withId(R.id.hello_button)).check(matches(isDisplayed()));  
    }  
}
```



# Step 2 :: Click Hello Button

```
public class MainActivityTest {  
  
    @Rule  
    public ActivityTestRule<MainActivity> activityTestRule = new ActivityTestRule(MainActivity.class);  
  
    @Test  
    public void findHelloButtonInView() {  
        onView(withId(R.id.hello_button)).check(matches(isDisplayed()));  
        onView(withId(R.id.hello_button)).perform(click());  
    }  
}
```



# Step 2 :: Click Hello Button

```
public class MainActivityTest {  
  
    @Rule  
    public ActivityTestRule<MainActivity> activityTestRule = new ActivityTestRule(MainActivity.class);  
  
    @Test  
    public void findHelloButtonInView() {  
        onView(withId(R.id.hello_button)).check(matches(isDisplayed()));  
        onView(withId(R.id.hello_button)).perform(click());  
    }  
}
```



# Step 3 :: Check result

```
public class MainActivityTest {  
  
    @Rule  
    public ActivityTestRule<MainActivity> activityTestRule = new ActivityTestRule(MainActivity.class);  
  
    @Test  
    public void findHelloButtonInView() {  
        onView(withId(R.id.hello_button)).check(matches(isDisplayed()));  
        onView(withId(R.id.hello_button)).perform(click());  
        onView(withId(R.id.message))  
            .check(matches(withText("Hello World with Espresso")));  
    }  
}
```



# Espresso APIs

View Matcher

View Action

View Assertion



# Espresso cheat sheet



onView(**ViewMatcher**)  
.perform(**ViewAction**)  
.check(**ViewAssertion**);

onData(**ObjectMatcher**)  
.DataOptions  
.perform(**ViewAction**)  
.check(**ViewAssertion**);

**View Matchers**

- USER PROPERTIES
  - isClickable(...)
  - isCheckable(...)
  - isLongClickable(...)
  - isFocusable(...)
  - isFocused()
  - isChecked()
  - isCheckable()
  - isClickableWithHint(...)
  - isSelected()
- HIERARCHY
  - isParent(**Matcher**)
  - isRoot(**Matcher**)
  - hasDescendant(**Matcher**)
  - isDescendantOf(**Matcher**)
  - hasSibling(**Matcher**)
  - isRoot()
- INPUT
  - supportsInputMethod(...)
  - hasEditText(...)
- OF PROPERTIES
  - isClickable()
  - isCheckable()
  - isFocusable()
  - isFocused()
  - isCheckable()
  - isChecked()
  - isClickableWithHint()
  - isSelected()
- CLASS
  - isAssignableFrom(...)
  - isSameName(...)
- ROOT MATCHERS
  - isClickable()
  - isCheckable()
  - isLongClickable()
  - isFocusable()
  - isFocusedView()
  - isLastForPop()
- OBJECT MATCHER
  - allOf(**Matchers**)
  - anyOf(**Matchers**)
  - not(...)
  - notAllOf(**Matchers**)
  - startsWith(CharSequence)
  - instanceOf(Class)
- SEE ALSO
  - PreferenceMatchers
  - CursorMatchers
  - LayoutMatchers

**Data Options**

- isExactlyViewMatcher()  
atPosition(Integer)  
onChildView(**Matcher**)

**View Actions**

- CLICK/PRESS
  - click()
  - doubleClick()
  - longClick()
  - press()
  - pressAndRelease()
  - pressKey([Int])
  - pressMetaKey()
  - clickOrKeyboard()
  - openLink()
- GESTURES
  - swipeLeft()
  - swipeUp()
  - swipeRight()
  - swipeDown()
  - swipe([Int])
- TEXT
  - clearText()
  - typeText(CharSequence)
  - typeTextUntilFocusedTime(String)
  - replaceText(CharSequence)

**View Assertions**

- MATCHERS
  - notMatch(**Matcher**)
  - doesNotStartWith(...)
  - selectsNoDescendants(**Matcher**)
- POSITION ASSERTIONS
  - isAfter(**Matcher**)
  - isBefore(**Matcher**)
  - isAfterAll(**Matchers**)
  - isBeforeAll(**Matchers**)
  - isBefore(**Matcher**)
  - isAfter(**Matcher**)
  - isBeforeOrIsGreaterOrEqual(**Matcher**)
  - isTopOrIsGreaterOrEqual(**Matcher**)
- LAYOUT ASSERTIONS
  - notHasSpannedCells(**Matcher**)
  - notHasAllCellsButtons()
  - notHasAny(**Matcher**)

intended(**IntentMatcher**);

intending(**IntentMatcher**)  
.respondWith(**ActivityResult**);

**Intent Matchers**

- INTENT
  - hasAction(...)
  - hasCategory(...)
  - hasExtra(...)
  - hasComponent(...)
  - hasExtra(...)
  - hasExtra(**Matcher**)
  - hasExtraWithKey(...)
  - hasType(...)
  - hasPackage(**String**)
  - hasFlag(int)
  - hasFlags(...)
  - isInternal()
- URI
  - hasScheme(...)
  - hasSchemeExact(...)
  - hasScheme(...)
  - hasSchemeString(...)
  - hasScheme(...)
  - hasSchemeSpecifiedPart(...)
- BUNDLE
  - hasEntry(...)
  - hasKey(...)
  - hasValue(...)
  - hasExtra(...)
  - hasExtra(...)
- COMPONENT NAME
  - hasClassName(...)
  - hasPackageName(...)
  - hasShortClassName(...)
  - hasMyPackageName()

v2.1.0, 4/21/2015



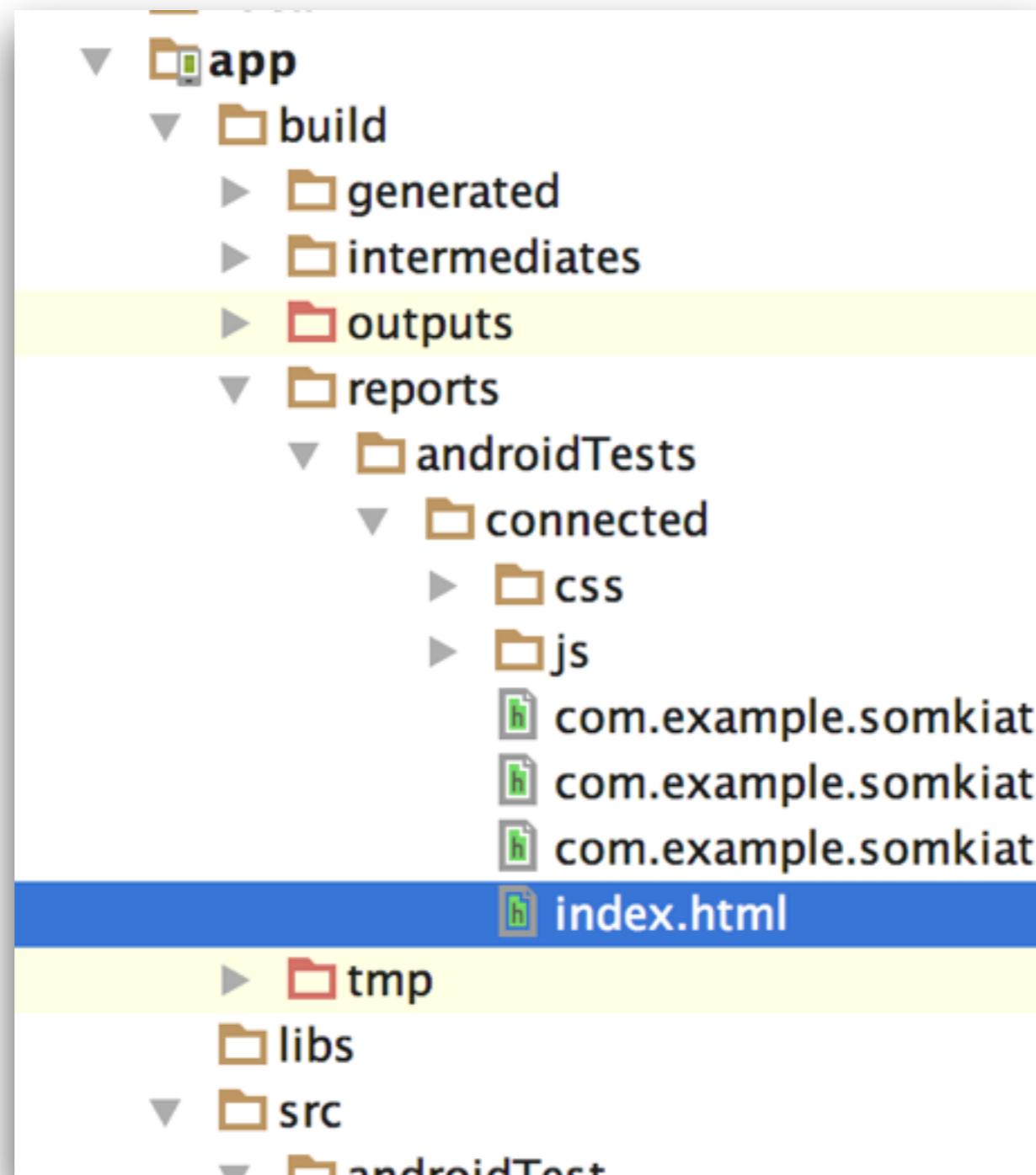
# Run test

`./gradlew clean connectedAndroidTest`

`./gradlew clean cAT`



# Test report



# Test report

## Test Summary

2 tests      0 failures      0.933s duration

100%  
successful

Packages

Classes

Package	Tests	Failures	Duration
<a href="#"><u>com.example.somkiat.helloworld</u></a>	2	0	0.933s



# Enable code coverage

```
buildTypes {  
    debug {  
        testCoverageEnabled = true  
    }  
}
```

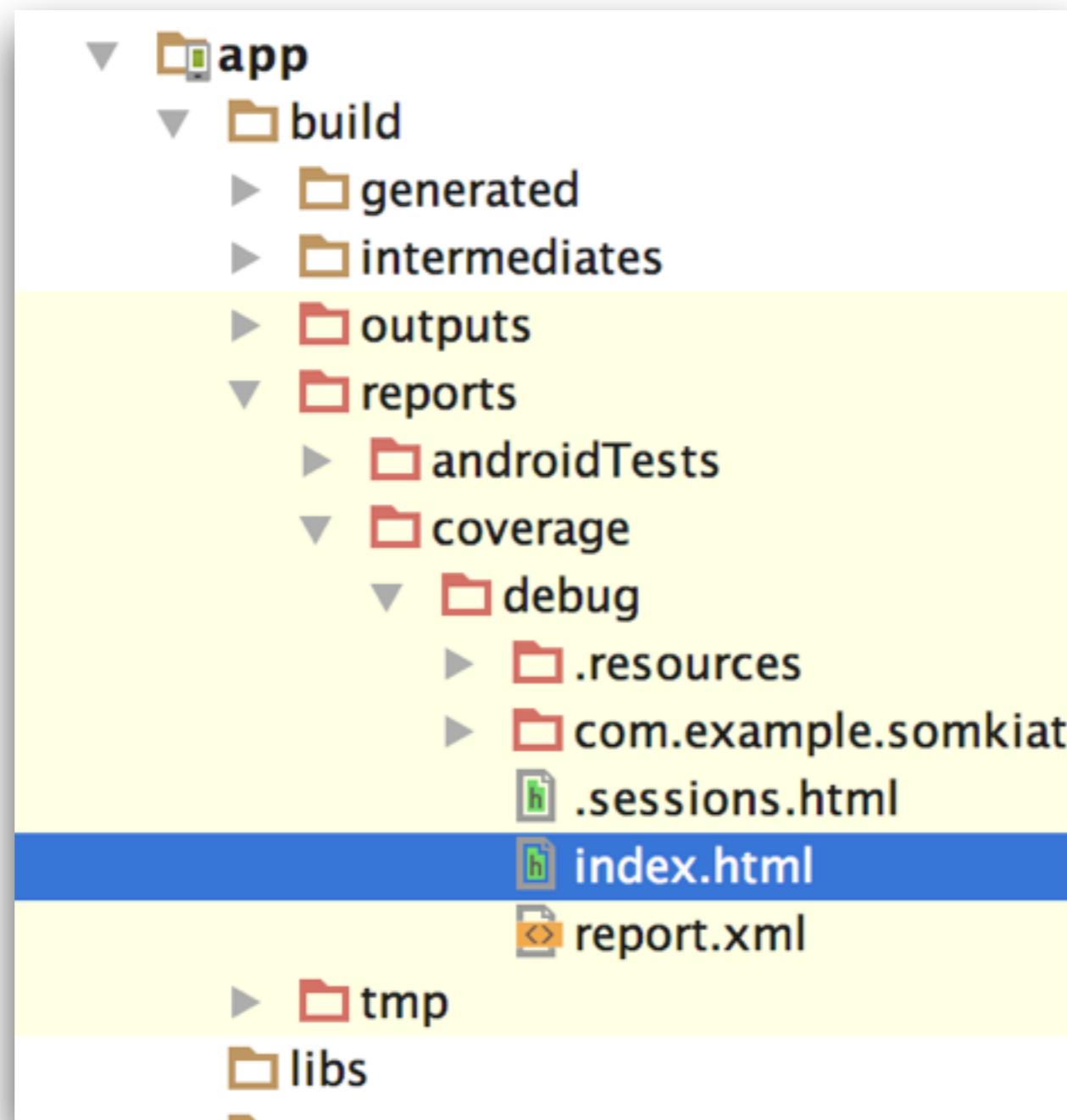


# Run with code coverage

```
./gradlew createDebugCoverageReport
```



# Code coverage report



# Code coverage report

 [debugAndroidTest](#) >  com.example.somkiat.helloworld

## com.example.somkiat.helloworld

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	M
 <a href="#">SecondActivity</a>		0%		n/a	2	2	
 <a href="#">MainActivity</a>		100%		n/a	0	3	
Total	10 of 31	68%	0 of 0	n/a	2	5	

Generated by the Android Gradle plugin 2.2.0-alpha4



# Working with second activity



# Check result

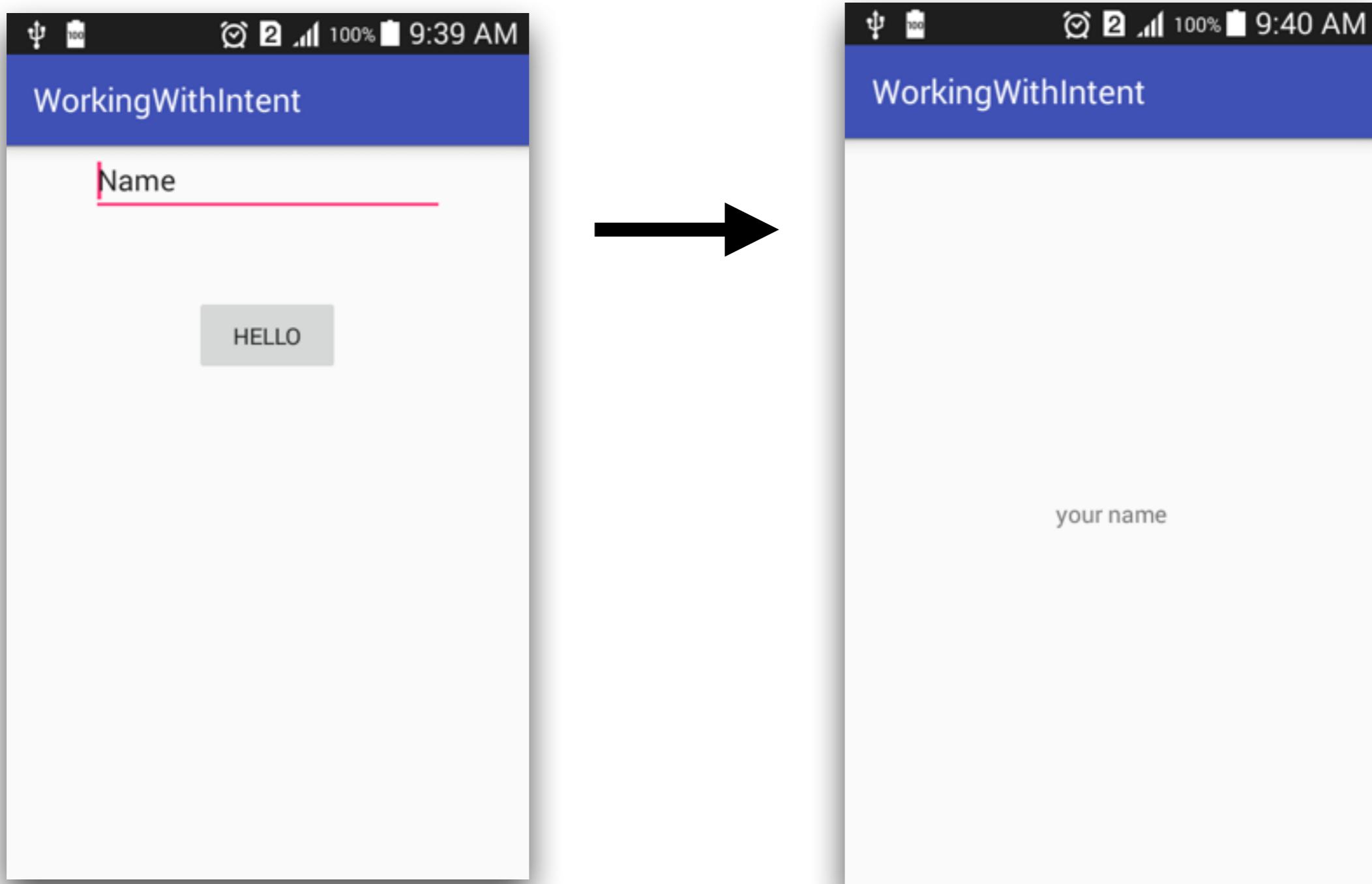
```
public class SecondActivityTest {  
  
    @Rule  
    public ActivityTestRule<SecondActivity> activityTestRule  
        = new ActivityTestRule(SecondActivity.class);  
  
    @Test  
    public void checkResultInView() {  
        onView(withId(R.id.result)).check(matches(isDisplayed()));  
        onView(withId(R.id.result)).check(matches(withText("Second activity")));  
    }  
}
```



# Working with intent



# Working with Intent



# Stub Intent

```
public class SecondActivityTest {  
  
    @Rule  
    public ActivityTestRule<SecondActivity> activityTestRule  
        = new ActivityTestRule(SecondActivity.class, false, false);  
  
    @Before  
    public void initialInput() {  
        Intent stubIntent = new Intent();  
        stubIntent.putExtra("result", "My message");  
        activityTestRule.launchActivity(stubIntent);  
    }  
  
    @Test  
    public void showResult() {  
        onView(withId(R.id.result)).check(matches(isDisplayed()));  
        onView(withId(R.id.result)).check(matches(withText("My message")));  
    }  
}
```



# Workshop

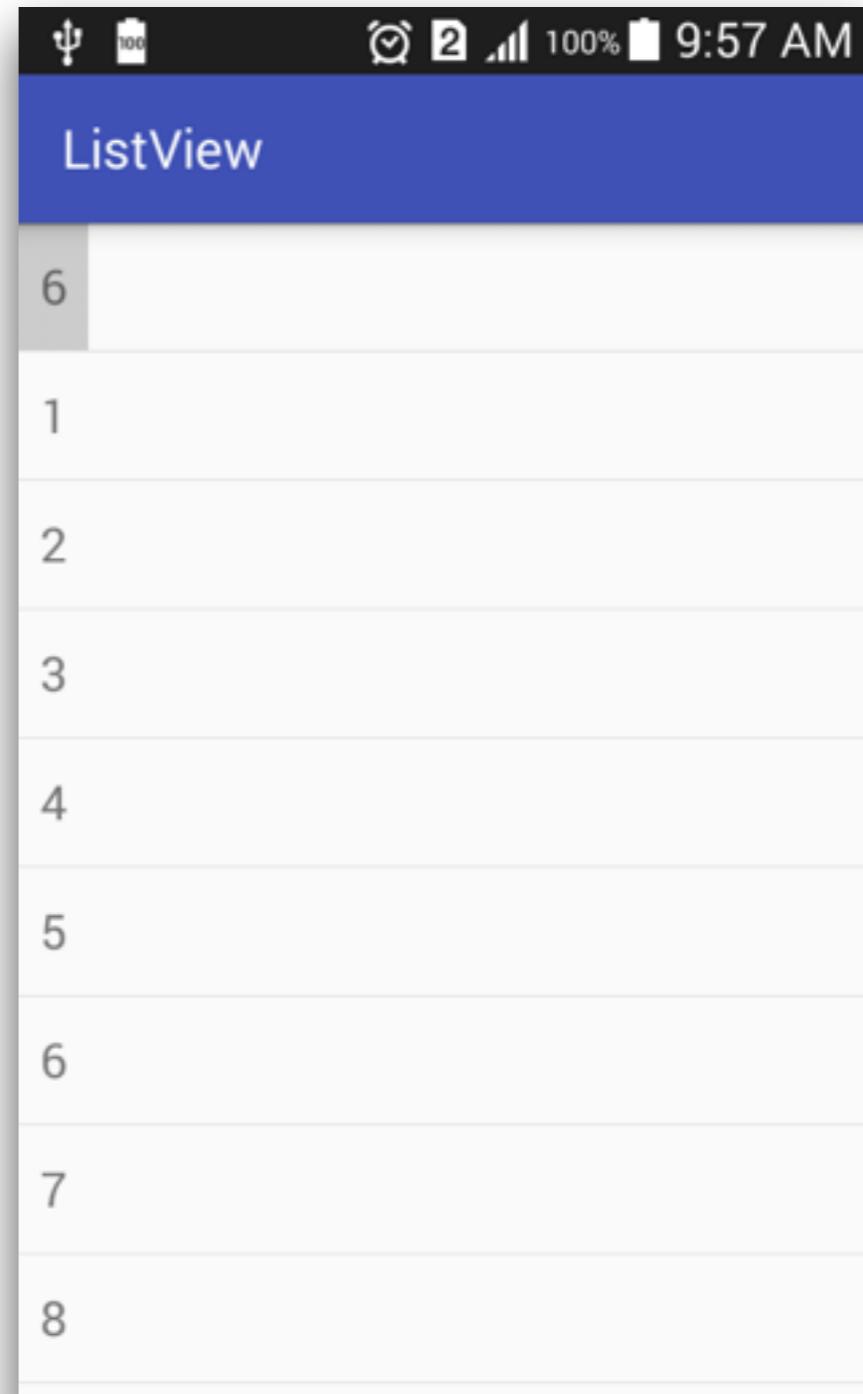
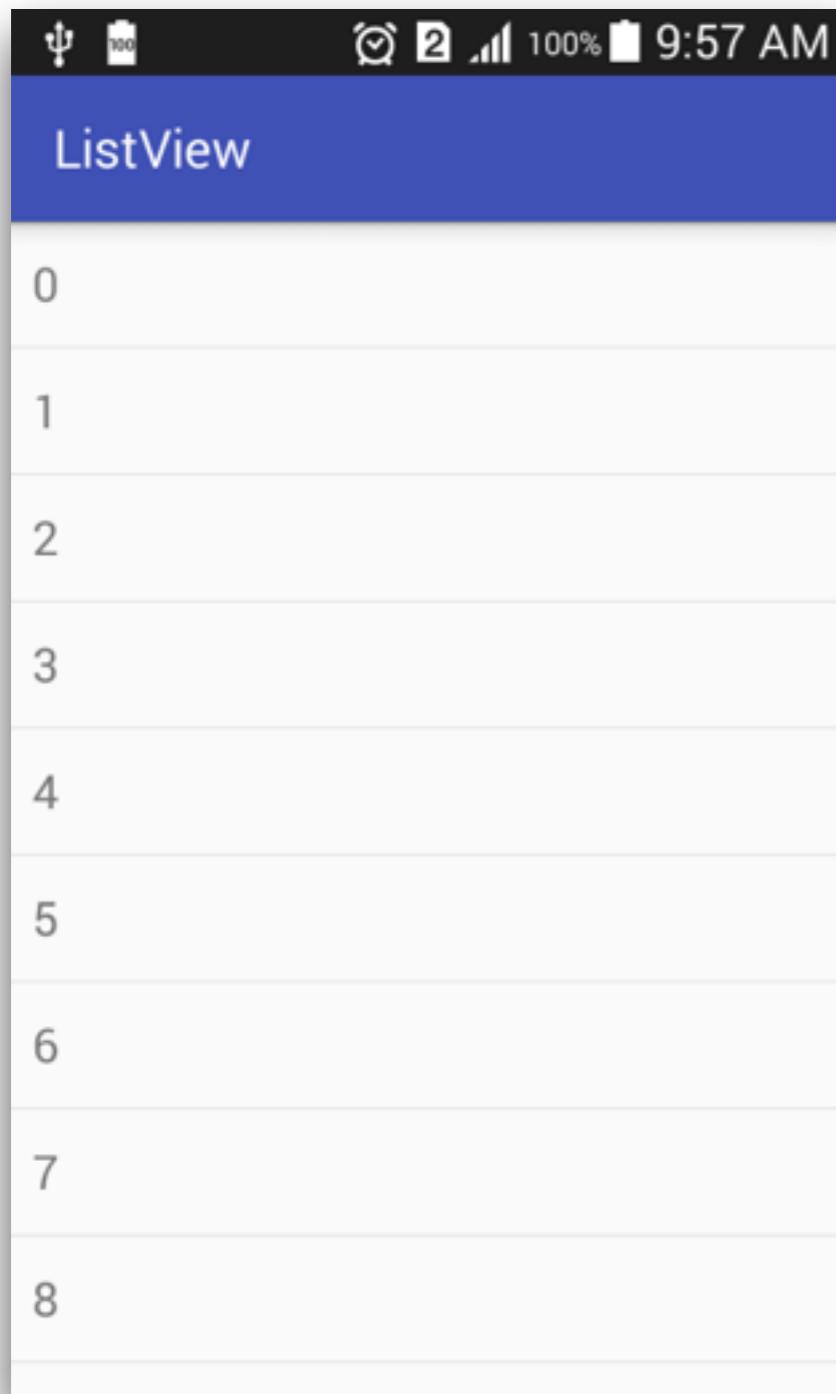


บริษัท สยามชานาญกิจ จำกัด และเพื่อนพ้องน้องพี่

# Working with Listview



# Working with ListView



# Resources

<https://google.github.io/android-testing-support-library/>

<https://github.com/googlesamples/android-testing>

<https://github.com/googlesamples/android-testing-templates>

<https://github.com/googlesamples/android-architecture>

