

# Basic of Android Testing





Somkiat
Home

Somkiat Puisungnoen

Update Info 1

View Activity Log 10+

...

Timeline

About

Friends 3,138

Photos

More ▾

When did you work at Opendream?

×

...

22 Pending Items

Intro

Software Craftsmanship

Software Practitioner at สยามชำนาญกิจ พ.ศ. 2556

Agile Practitioner and Technical at SPRINT3r

Post

Photo/Video

Live Video

Life Event

What's on your mind?

Public ▾

Post

Somkiat Puisungnoen

15 mins · Bangkok · 🌐 ▾

Java and Bigdata

...



Facebook interface for the page **somkiat.cc**. The top navigation bar includes the Facebook logo, the page name, a search bar, and icons for Home, Messages, Notifications (3), Insights, Publishing Tools, Settings, and Help.

The main content area displays a video of a man in a white Superman t-shirt with "SOMKIAT.CC" printed on it, posing against a white wall. A blue call-to-action button is overlaid on the video: "Help people take action on this Page. x". Below the video, there are buttons for "Liked", "Following", "Share", and a menu icon.

The left sidebar shows the page name **somkiat.cc**, the handle **@somkiat.cc**, and a menu with options: Home, Posts, Videos, and Photos.



**[https://github.com/up1/  
workshop-basic-android-testing](https://github.com/up1/workshop-basic-android-testing)**



# Agenda in 3 hours

Introduction of testing

Why we need to test ?

Types of tests

Testing pyramid concept

Android testing

Workshop (step-by-step)

Homework and assignment



# Testing for Android app



# Why we need to test ?

Help you to catch bugs

Develop features faster

Enforce modularity of your project





**But,  
It's take time to learning and  
practice !!**



# Goals

How to **THINK** when and where  
you should test



# What you need to know ?

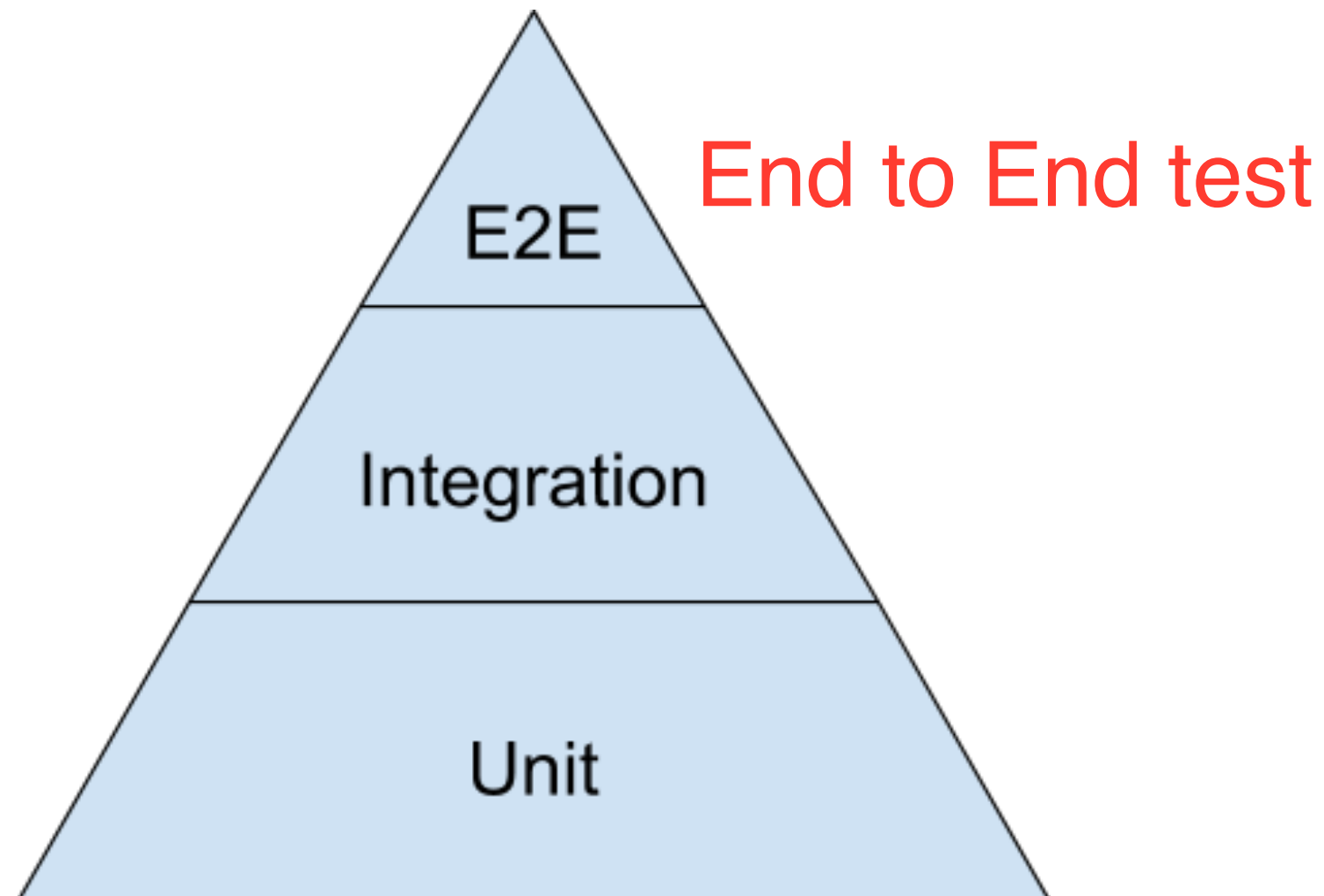
Android  
Android Studio  
**JUnit 4**  
**Espresso**



# Type of testing



# Testing Pyramid



# Android Testing

Android



# Android Testing

Android

Java



# Java run on JVM

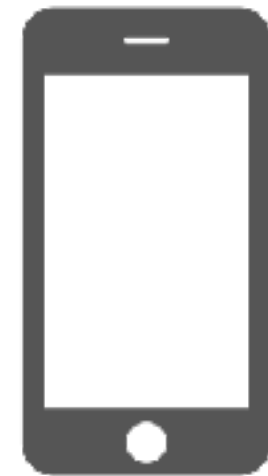


JVM (Java Virtual Machine)





# Run android need device



Build app -> Install to device -> Test



# Android Testing

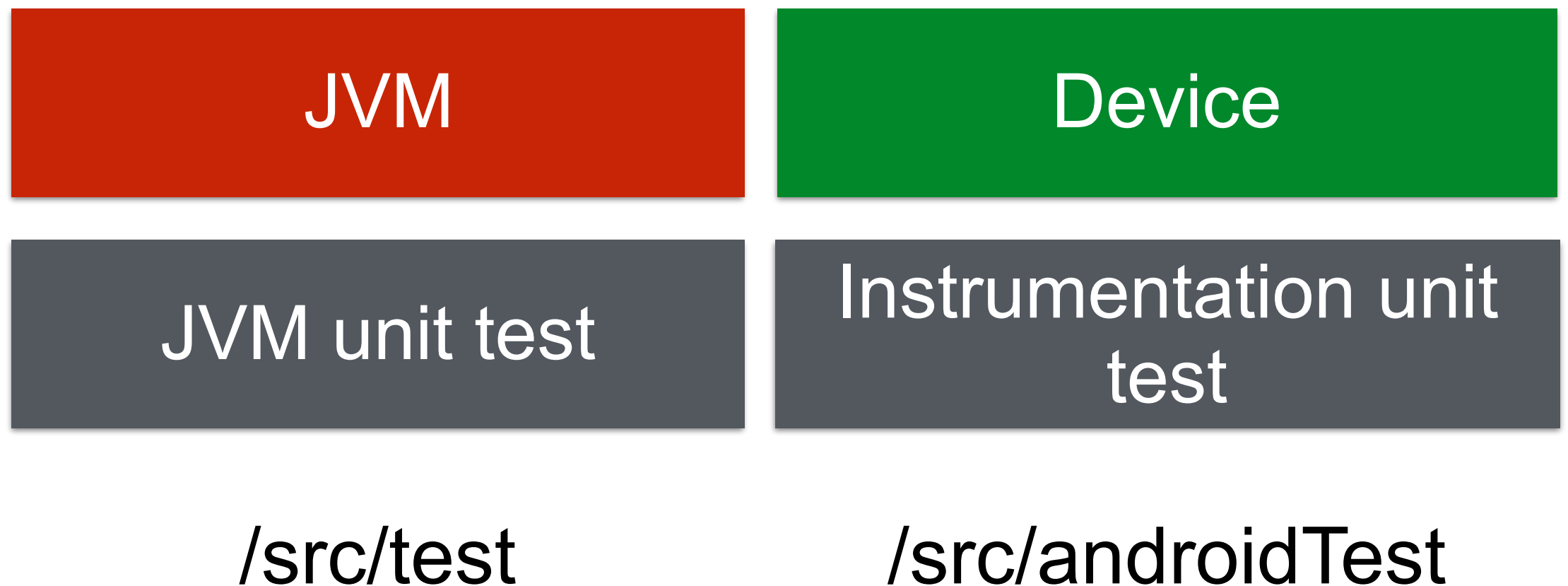


`/src/test`

*Business logic with pure java code*



# Android Testing



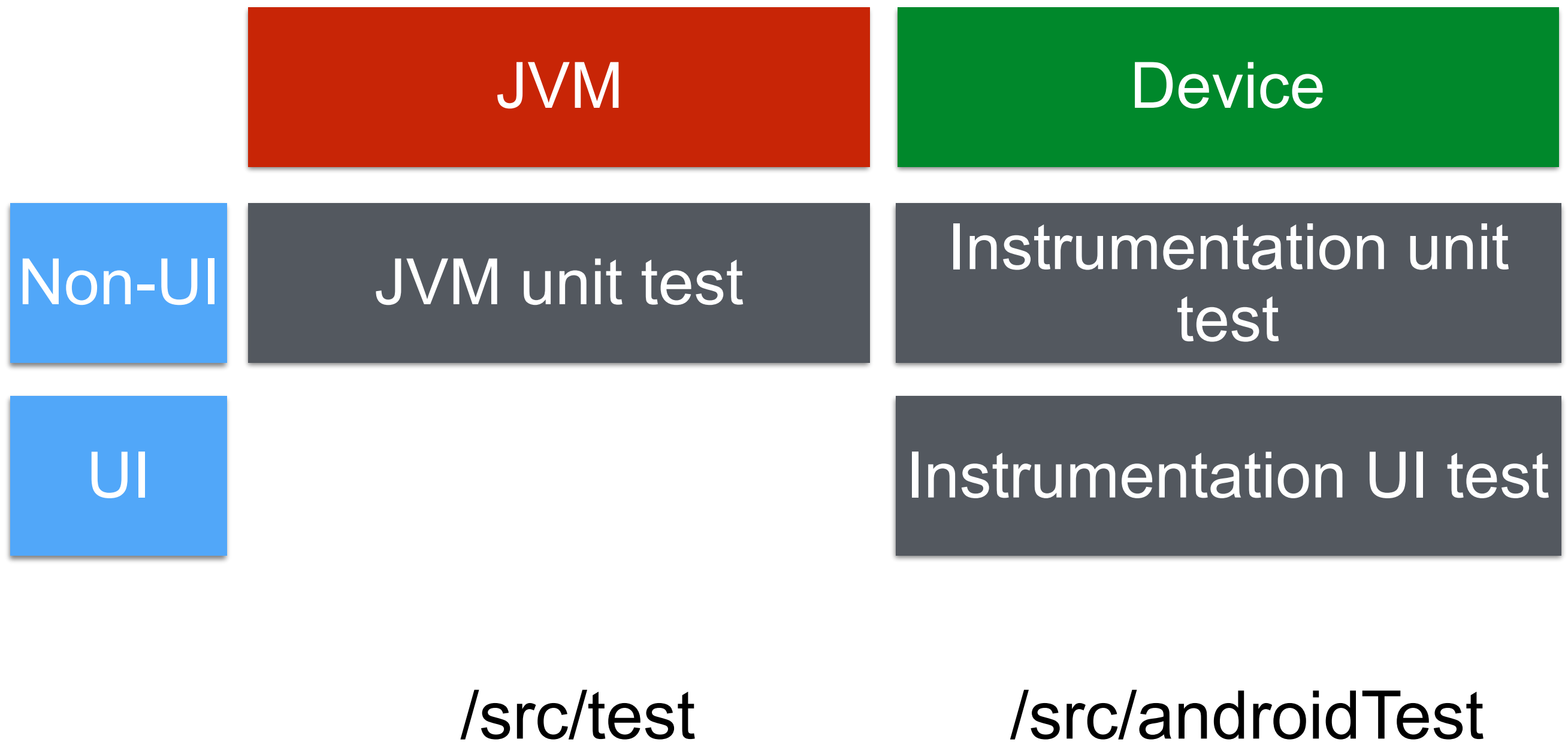
*Working with Android specific code, you need run on device such as AssetManager, SharedPreferences*



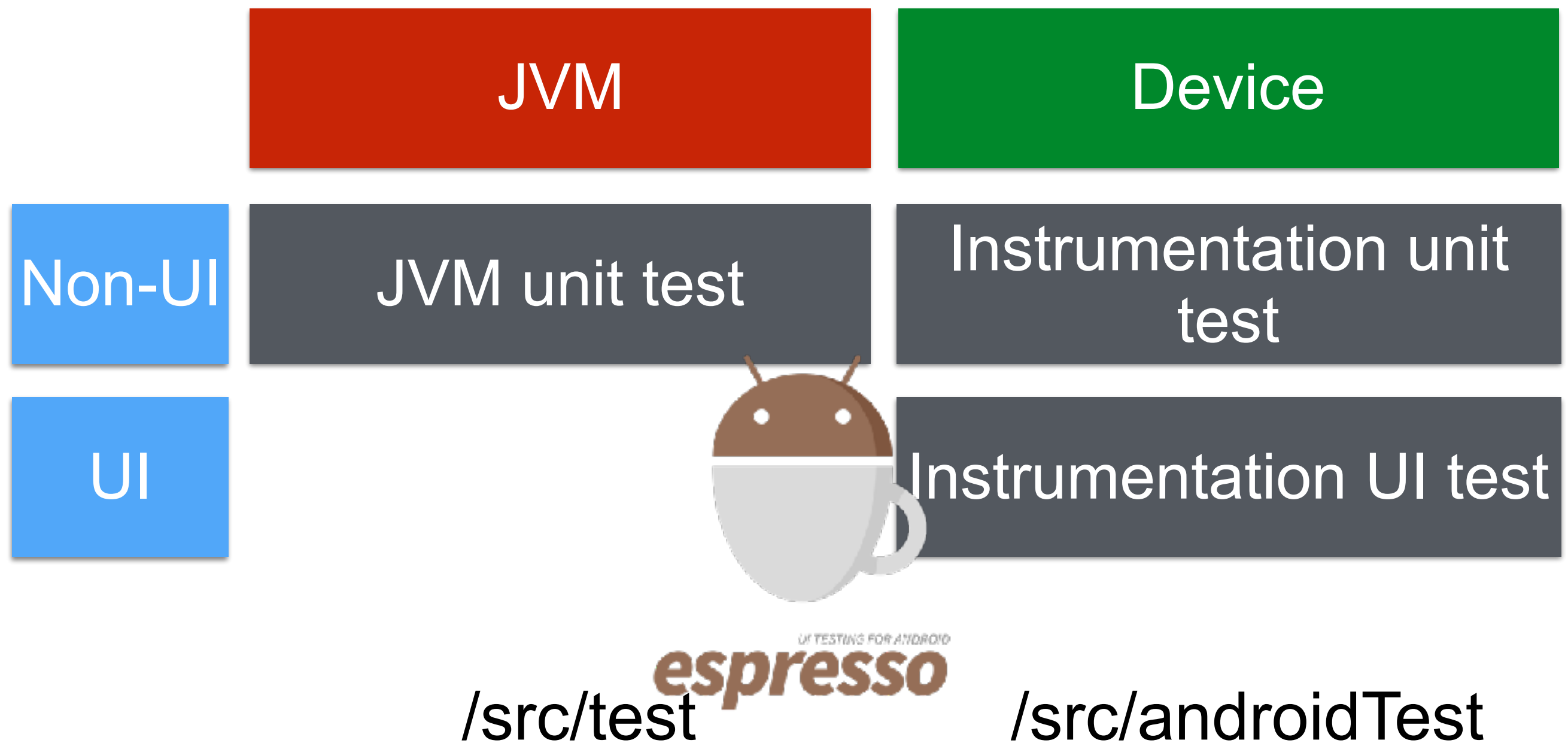
# UI vs Non-UI



# Android Testing



# Android Testing



# Resources

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

<https://developer.android.com/topic/libraries/testing-support-library/index.html#Espresso>



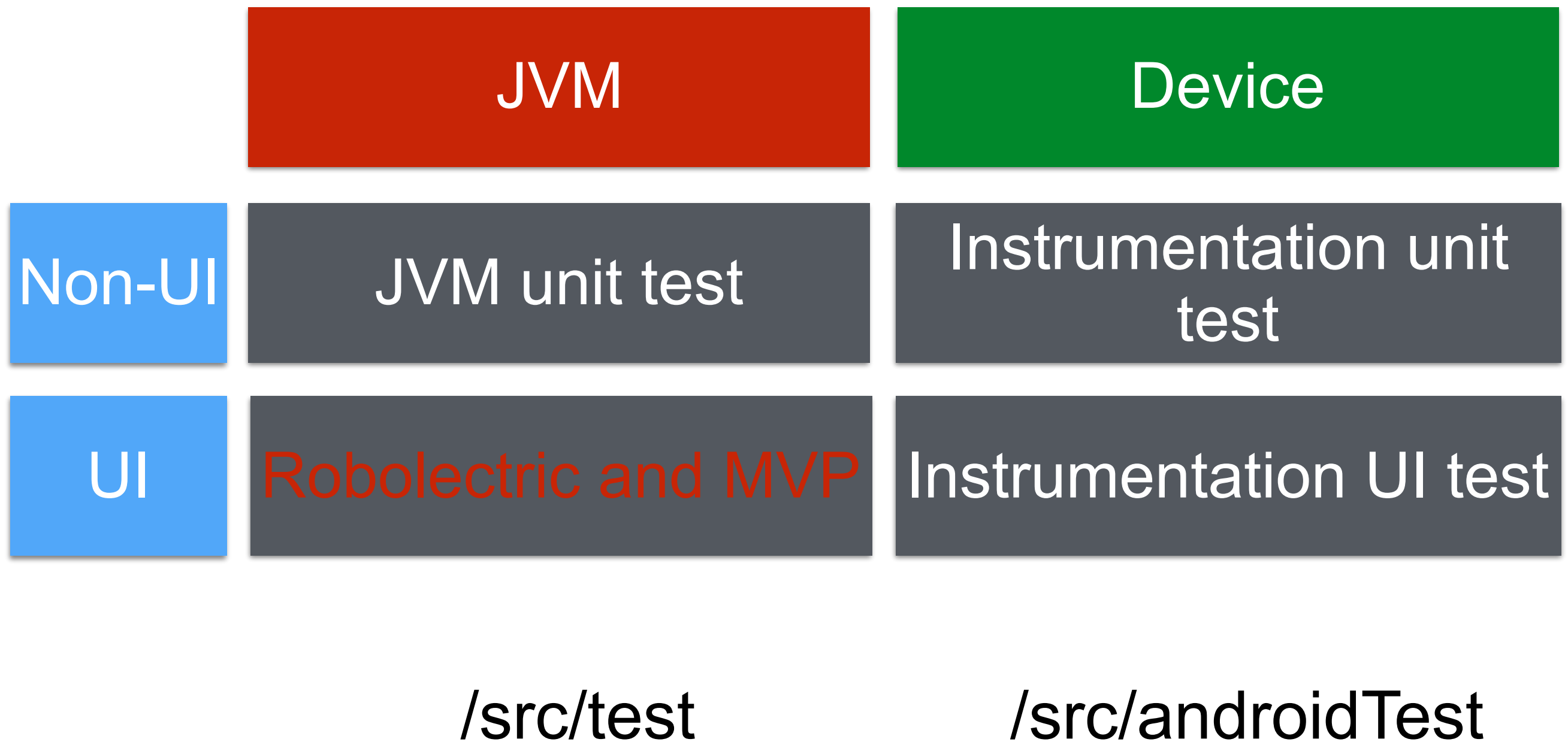
# Android Testing

	JVM	Device
Non-UI	JVM unit test	Instrumentation unit test
UI	???	Instrumentation UI test
	/src/test	/src/androidTest





# Android Testing



# Rule of thumb

Instrumentation tests are **slower** than JVM tests.

Try to separate the standard Java code from Android-dependent code.



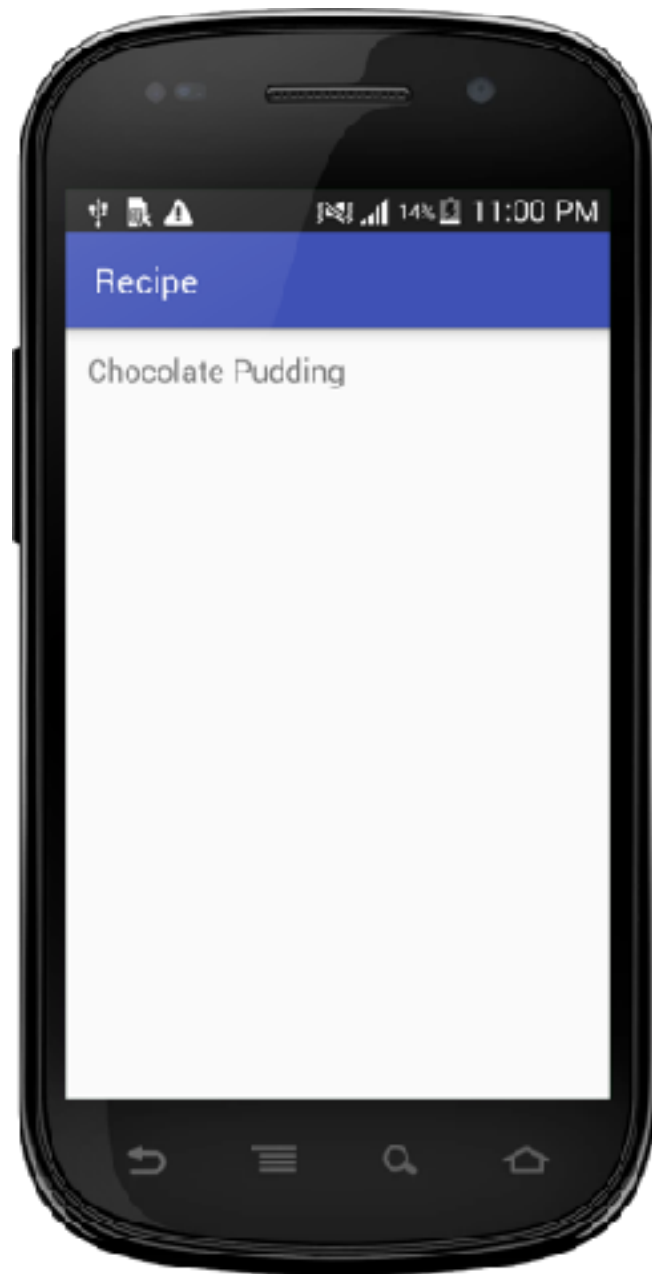
# Workshop with Testing

step-by-step to write tests



# Recipe application

List of recipes



Detail of recipe



# Prepare your project



# 1. Fork from Github repository

This screenshot shows the GitHub interface for a repository named 'up1 / workshop-basic-android-testing'. The 'Fork' button is highlighted with a red box. The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The commit history shows a single commit 'Started project of workshop' with files app, gradle/wrapper, .gitignore, build.gradle, and gradle.properties.

Repository: up1 / workshop-basic-android-testing

Unwatch 1 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Insights Settings

No description, website, or topics provided. [Add topics](#) [Edit](#)

1 commit 1 branch 0 releases 1 contributor

Branch: master New pull request Create new file Upload files Find file Clone or download

up1 Started project of workshop Latest commit 4c325d6 19 seconds ago

app	Started project of workshop	18 seconds ago
gradle/wrapper	Started project of workshop	18 seconds ago
.gitignore	Started project of workshop	18 seconds ago
build.gradle	Started project of workshop	18 seconds ago
gradle.properties	Started project of workshop	18 seconds ago

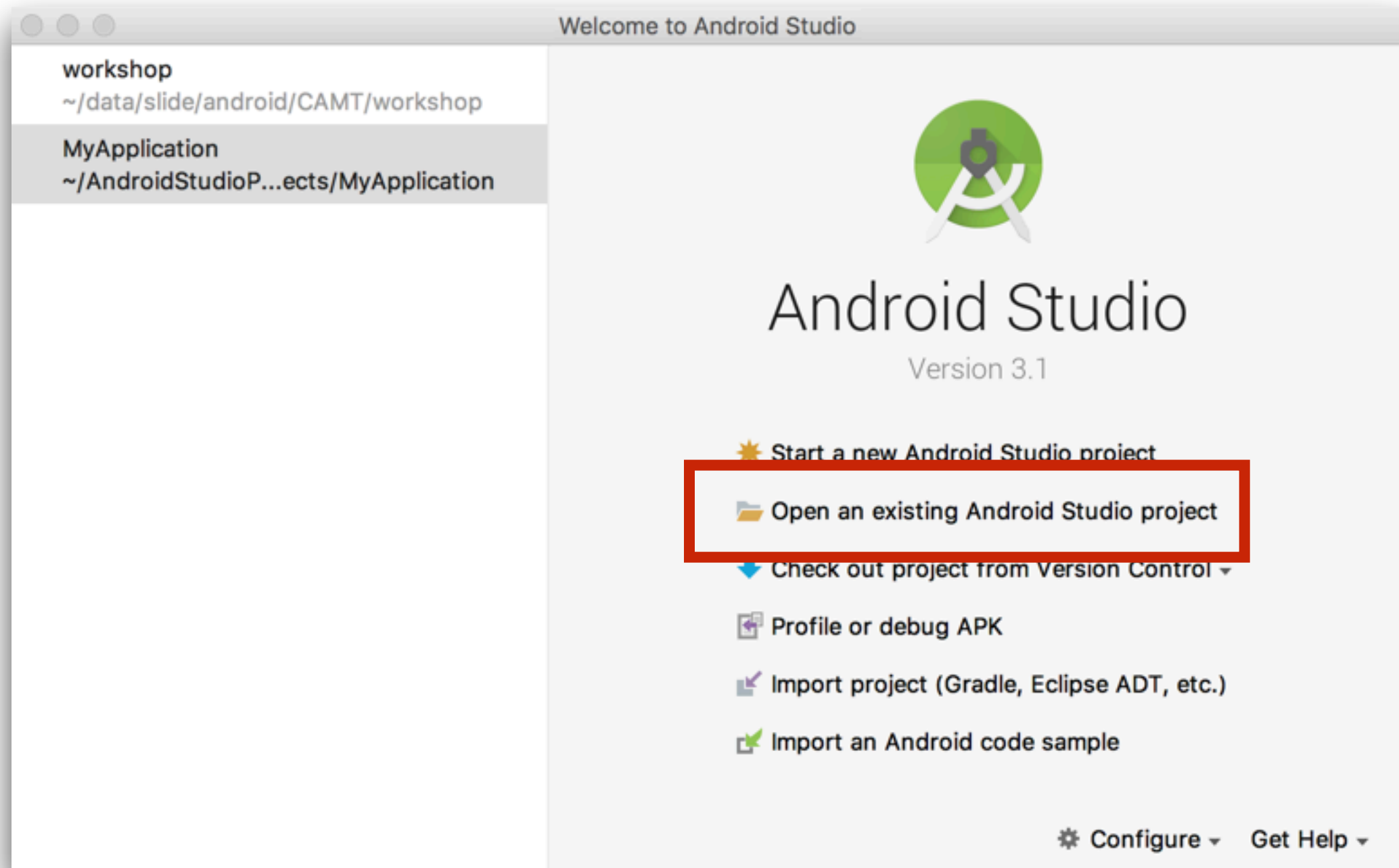


## 2. Clone from your repository

```
$git clone https://github.com/  
<username>/workshop-basic-android-  
testing
```

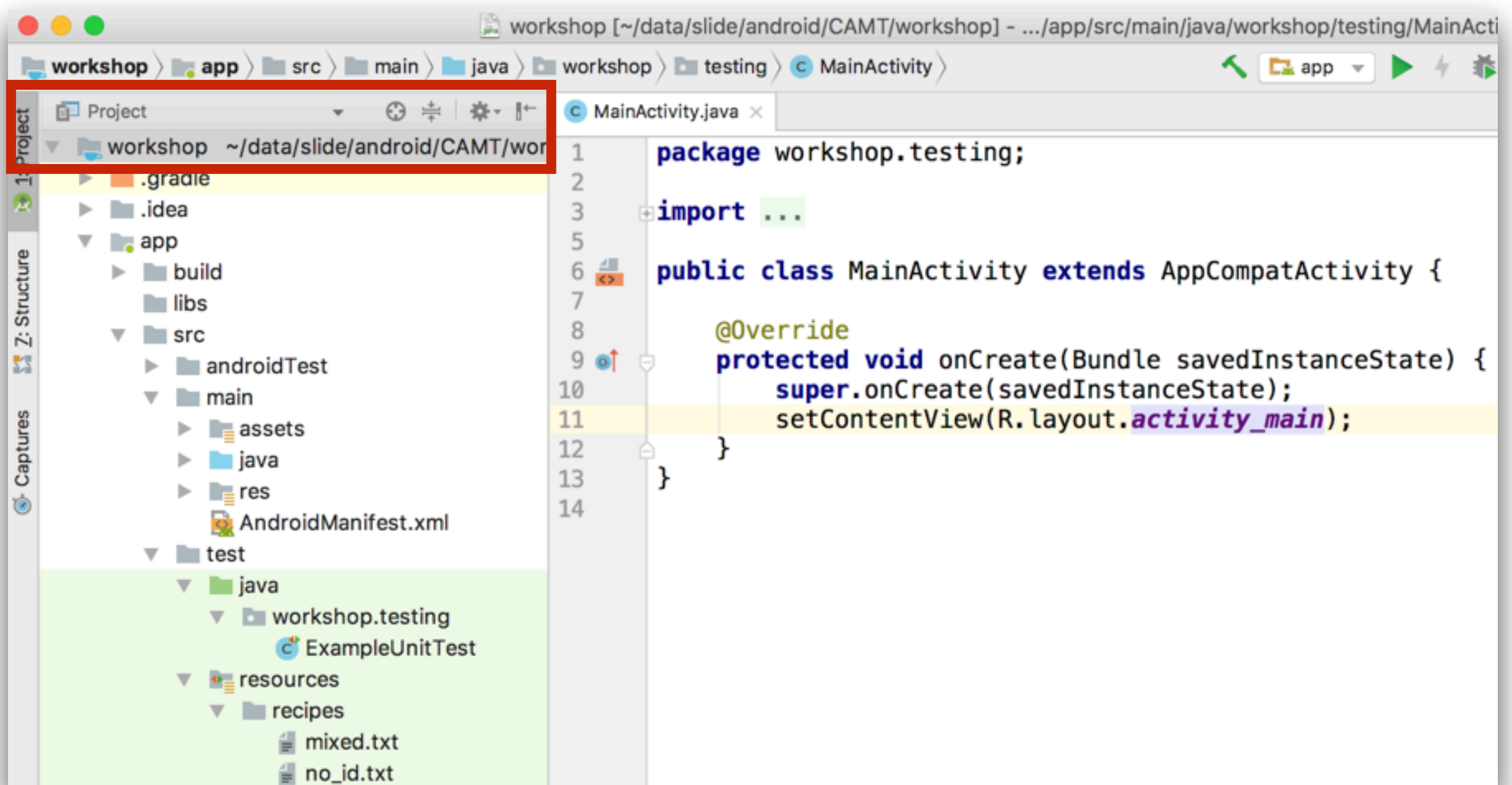


# 3. Import to Android Studio

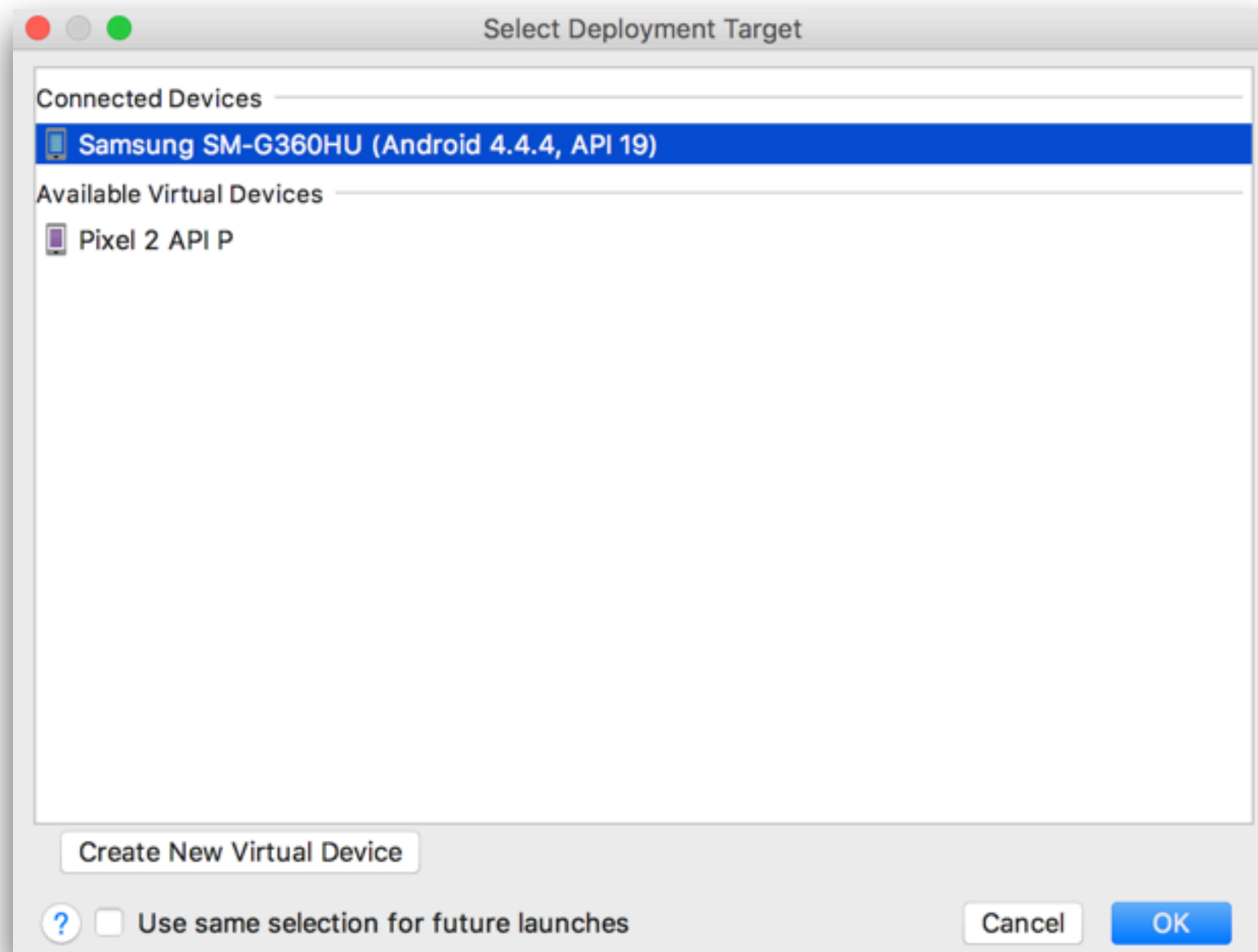




# 4. Switch to project view



# 5. Try to run on device/emulator



# Ready to start



# Rule of workshop

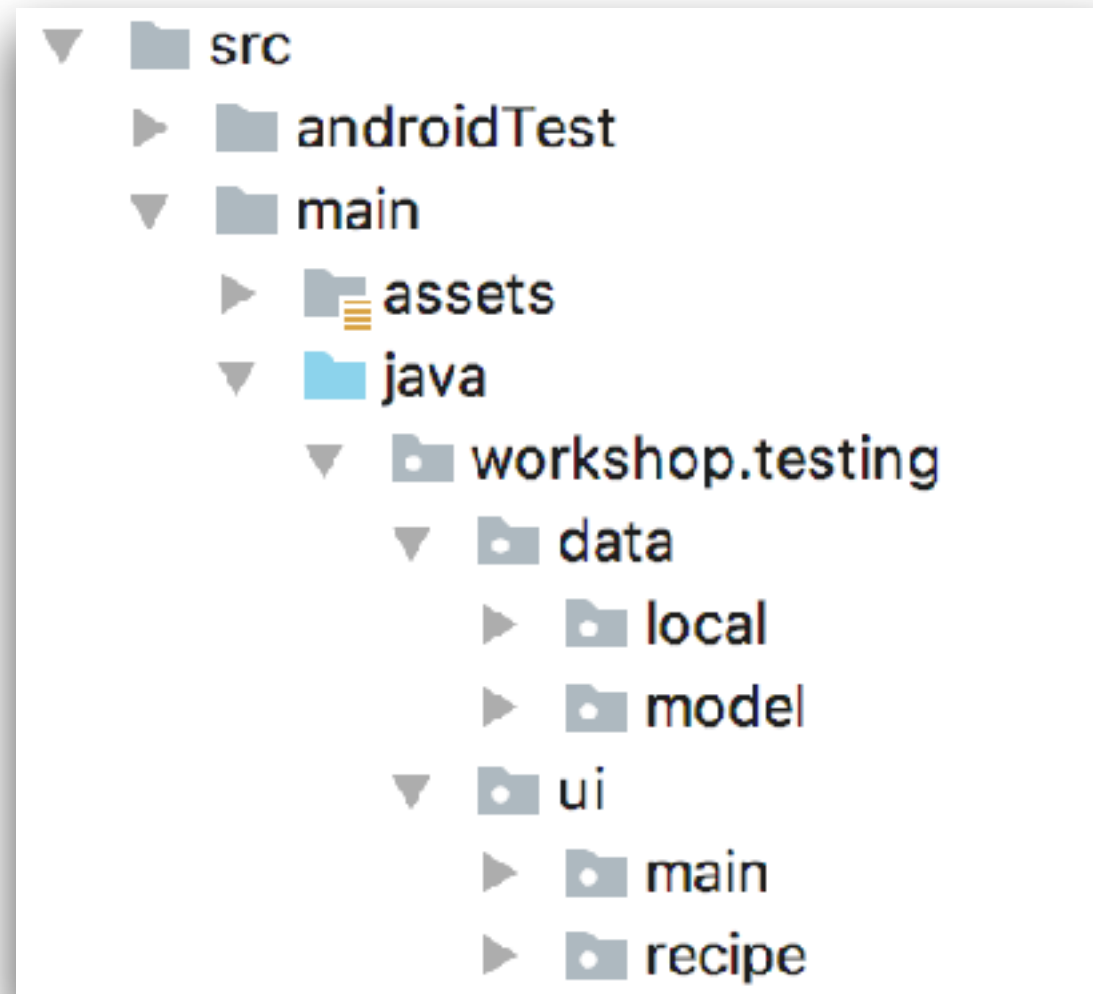
One test case per commit and push to  
Github repository



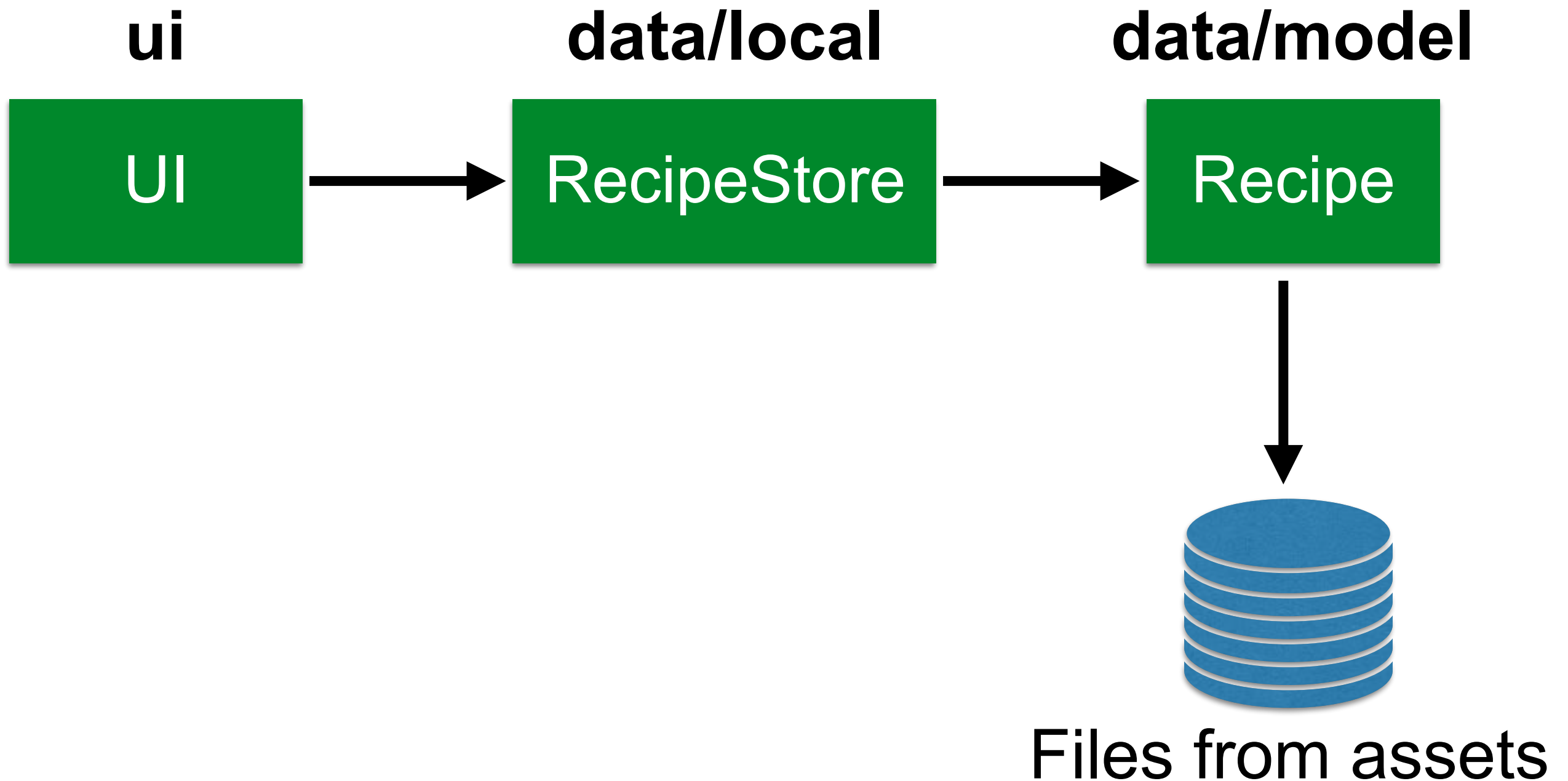
# Let' start



# Project structure



# Project structure



# Test structure

```
public class RecipeTest {  
  
    @Test  
    public void water() {  
        //Arrange  
        InputStream stream  
            = RecipeTest.class.getResourceAsStream( name: "/recipes/water.txt");  
  
        // Act  
        Recipe recipe = Recipe.readFromStream(stream);  
  
        // Assert  
        assertNotNull(recipe);  
        assertEquals( expected: "water", recipe.id);  
        assertEquals( expected: "Water", recipe.title);  
        assertEquals( expected: "Put glass under tap. Open tap. Close tap. Drink."  
            , recipe.description);  
    }  
}
```





# Test structure

Test class name is a group of tests

```
public class RecipeTest {
```

```
    @Test
```

```
    public void water() {
```

```
        //Arrange
```

```
        InputStream stream
```

```
            = RecipeTest.class.getResourceAsStream( name: "/recipes/water.txt");
```

```
        // Act
```

```
        Recipe recipe = Recipe.readFromStream(stream);
```

```
        // Assert
```

```
        assertNotNull(recipe);
```

```
        assertEquals( expected: "water", recipe.id);
```

```
        assertEquals( expected: "Water", recipe.title);
```

```
        assertEquals( expected: "Put glass under tap. Open tap. Close tap. Drink."
            , recipe.description);
```

```
    }
```

```
}
```



# Test structure

Test annotation of JUnit 4 use to define the method as a test case

```
public class RecipeTest {  
    @Test  
    public void water() {  
        // Arrange  
        InputStream stream  
            = RecipeTest.class.getResourceAsStream( name: "/recipes/water.txt");  
  
        // Act  
        Recipe recipe = Recipe.readFromStream(stream);  
  
        // Assert  
        assertNotNull(recipe);  
        assertEquals( expected: "water", recipe.id);  
        assertEquals( expected: "Water", recipe.title);  
        assertEquals( expected: "Put glass under tap. Open tap. Close tap. Drink."  
            , recipe.description);  
    }  
}
```



# Test structure

Arrange section to setup data and states of test case

```
public class RecipeTest {  
  
    @Test  
    public void water() {  
        //Arrange  
        InputStream stream  
            = RecipeTest.class.getResourceAsStream( name: "/recipes/water.txt");  
  
        // Act  
        Recipe recipe = Recipe.readFromStream(stream);  
  
        // Assert  
        assertNotNull(recipe);  
        assertEquals( expected: "water", recipe.id);  
        assertEquals( expected: "Water", recipe.title);  
        assertEquals( expected: "Put glass under tap. Open tap. Close tap. Drink."  
            , recipe.description);  
    }  
}
```



# Test structure

Act section to call the target method to check and verify behavior

```
public class RecipeTest {  
  
    @Test  
    public void water() {  
        //Arrange  
        InputStream stream  
            = RecipeTest.class.getResourceAsStream(name: "/recipes/water.txt");  
  
        // Act  
        Recipe recipe = Recipe.readFromStream(stream);  
  
        // Assert  
        assertNotNull(recipe);  
        assertEquals(expected: "water", recipe.id);  
        assertEquals(expected: "Water", recipe.title);  
        assertEquals(expected: "Put glass under tap. Open tap. Close tap. Drink."  
            , recipe.description);  
    }  
}
```



# Test structure

Assert section to check the result as we expected or not

```
public class RecipeTest {  
  
    @Test  
    public void water() {  
        //Arrange  
        InputStream stream  
            = RecipeTest.class.getResourceAsStream( name: "/recipes/water.txt");  
  
        // Act  
        Recipe recipe = Recipe.readFromStream(stream);  
  
        // Assert  
        assertNotNull(recipe);  
        assertEquals( expected: "water", recipe.id);  
        assertEquals( expected: "Water", recipe.title);  
        assertEquals( expected: "Put glass under tap. Open tap. Close tap. Drink."  
            , recipe.description);  
    }  
}
```



# Check code coverage



# Code coverage

A tool to measure how much of your code is covered by tests that break down into classes, methods and lines.



# Code coverage

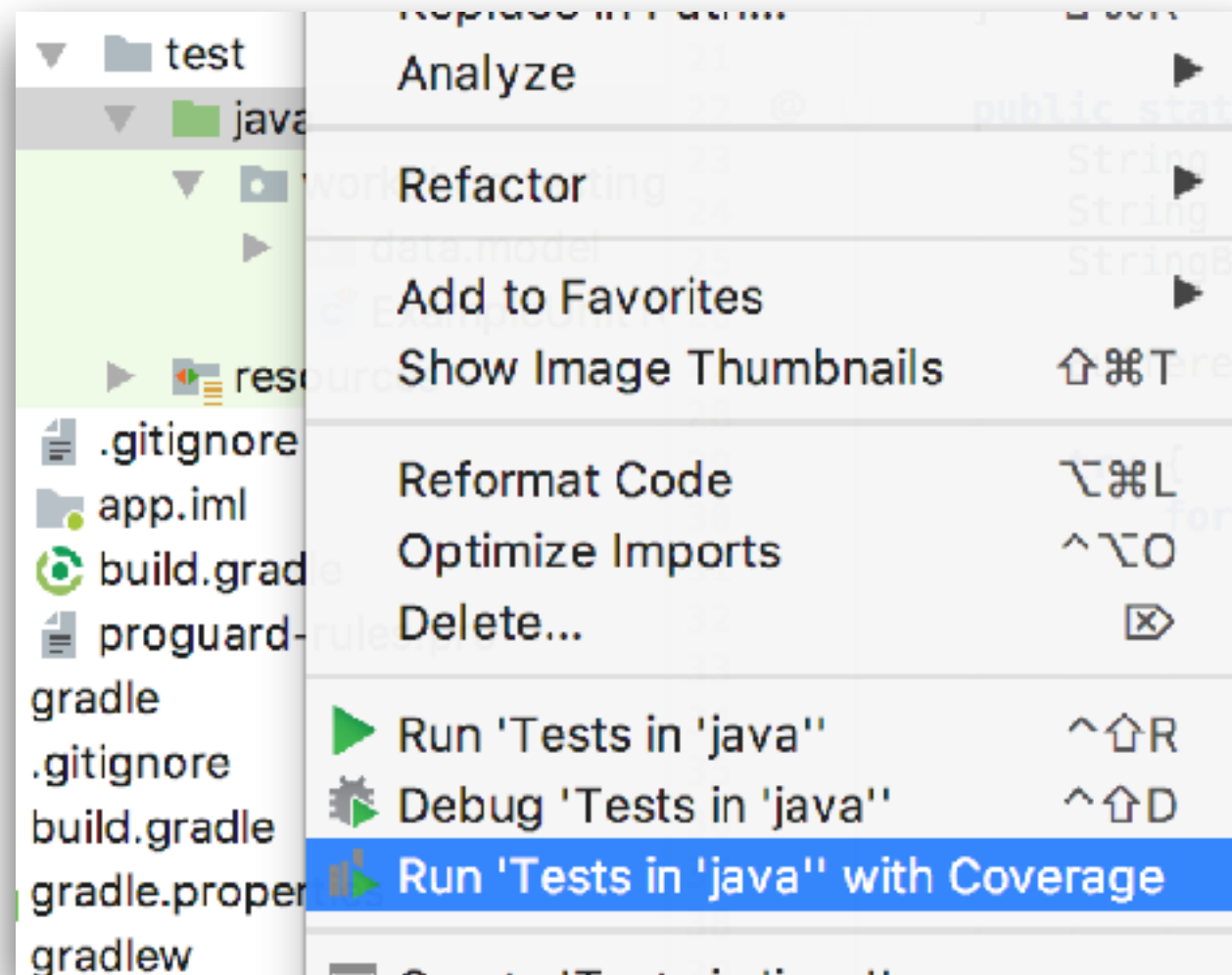
But 100% of code coverage does not mean that your code is 100% correct





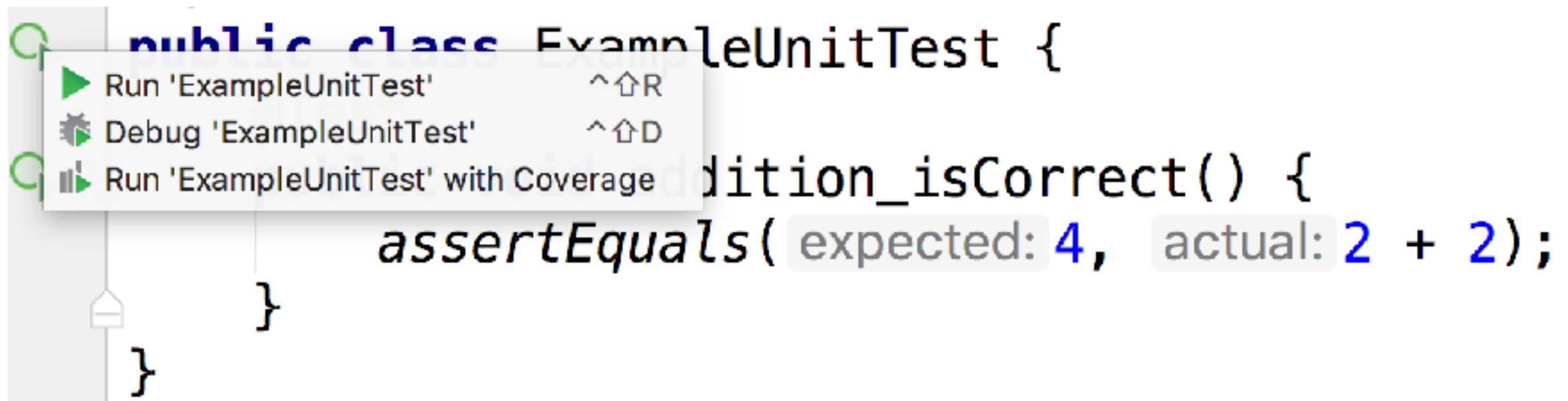
# Check code coverage

Right click at test or androidTest directory



# Check code coverage

Right click at test or androidTest directory



# Check code coverage

See the result

Coverage testing in app (1)

0% classes, 0% lines covered in package 'testing'

Element	Class, %	Method, %	Line, %
data	0% (0/2)	0% (0/4)	0% (0/37)
ui	0% (0/5)	0% (0/9)	0% (0/42)
BuildConfig	0% (0/1)	0% (0/1)	0% (0/1)
R	0% (0/14)	0% (0/1)	0% (0/42)



# Check code coverage

Export the result to HTML format

[ [all classes](#) ] [ workshop.testing ]

Coverage Summary for Package: workshop.testing

Package	Class, %	Method, %
workshop.testing	0% (0/ 15)	0% (0/ 17)

Class ▲	Class, %	Method, %
<a href="#">BuildConfig</a>	0% (0/ 1)	0% (0/ 2)
<a href="#">R</a>	0% (0/ 14)	0% (0/ 15)

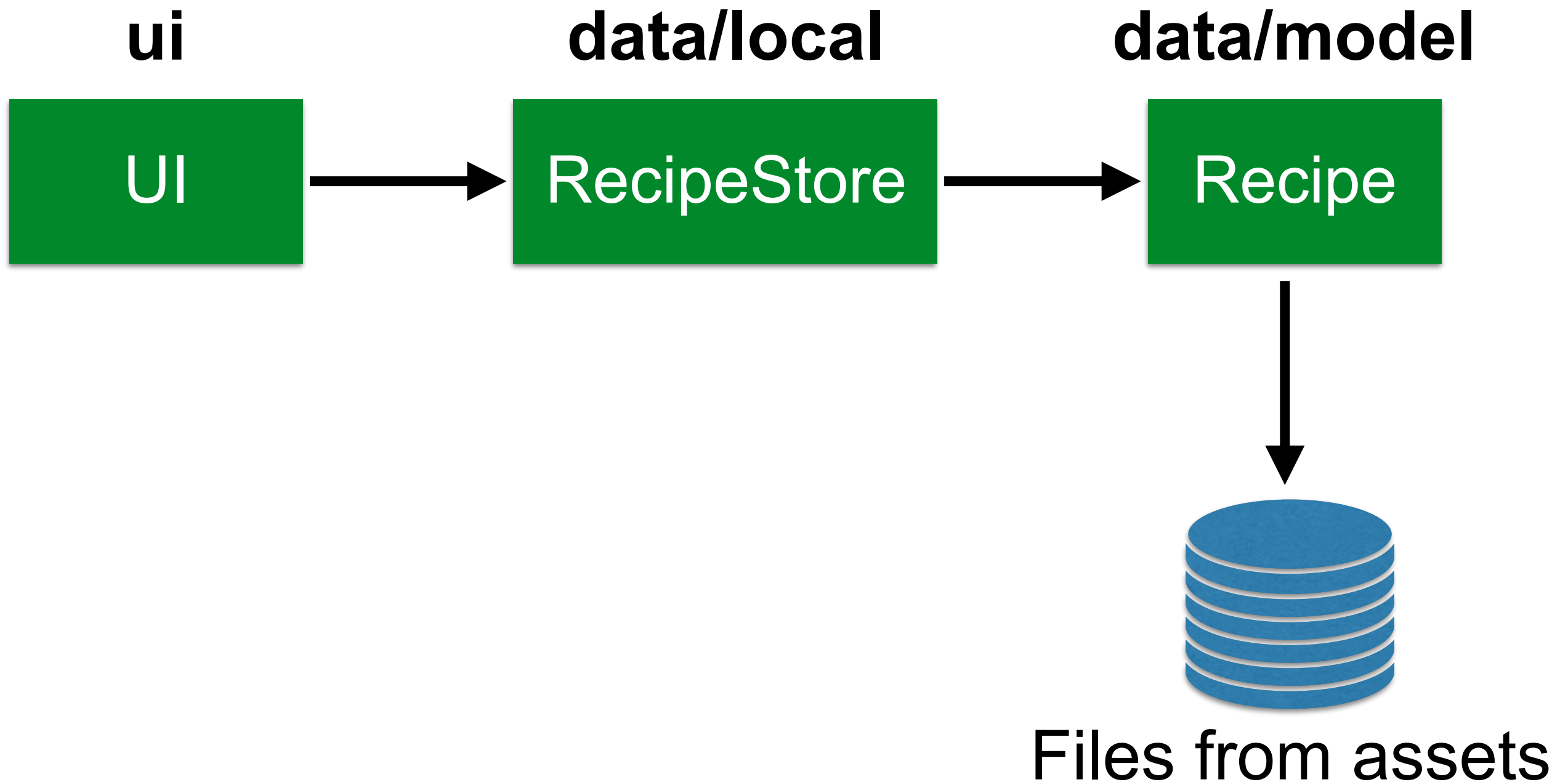
*0% is good point to improve !!*



# Let' coding with tests



# What and Where to test ?



# Steps to develop app with tests

Read data of recipe from file system

Show detail of recipe in Activity

more ...

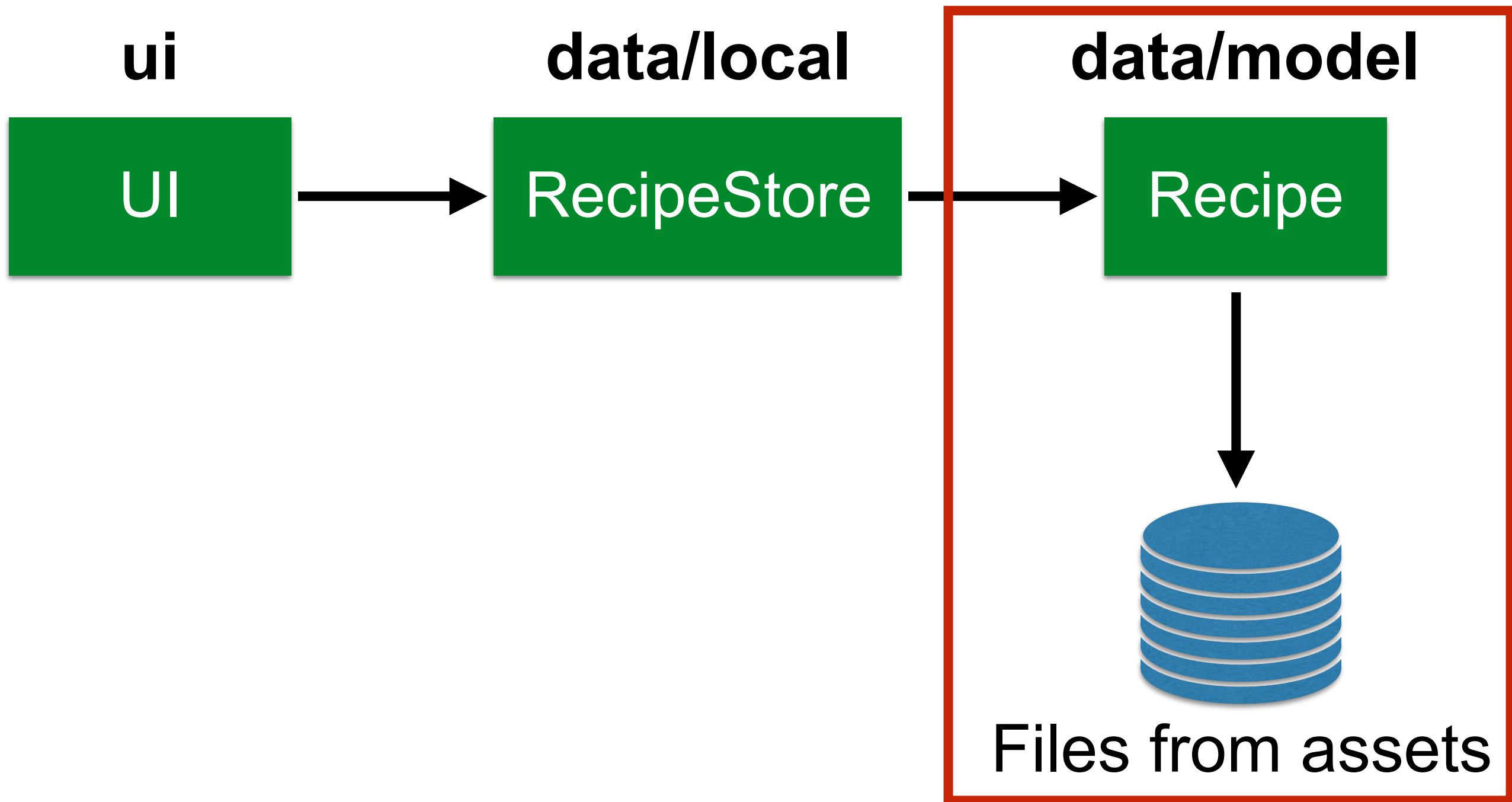


# 1. Read data of recipe from file system





# Read data from file

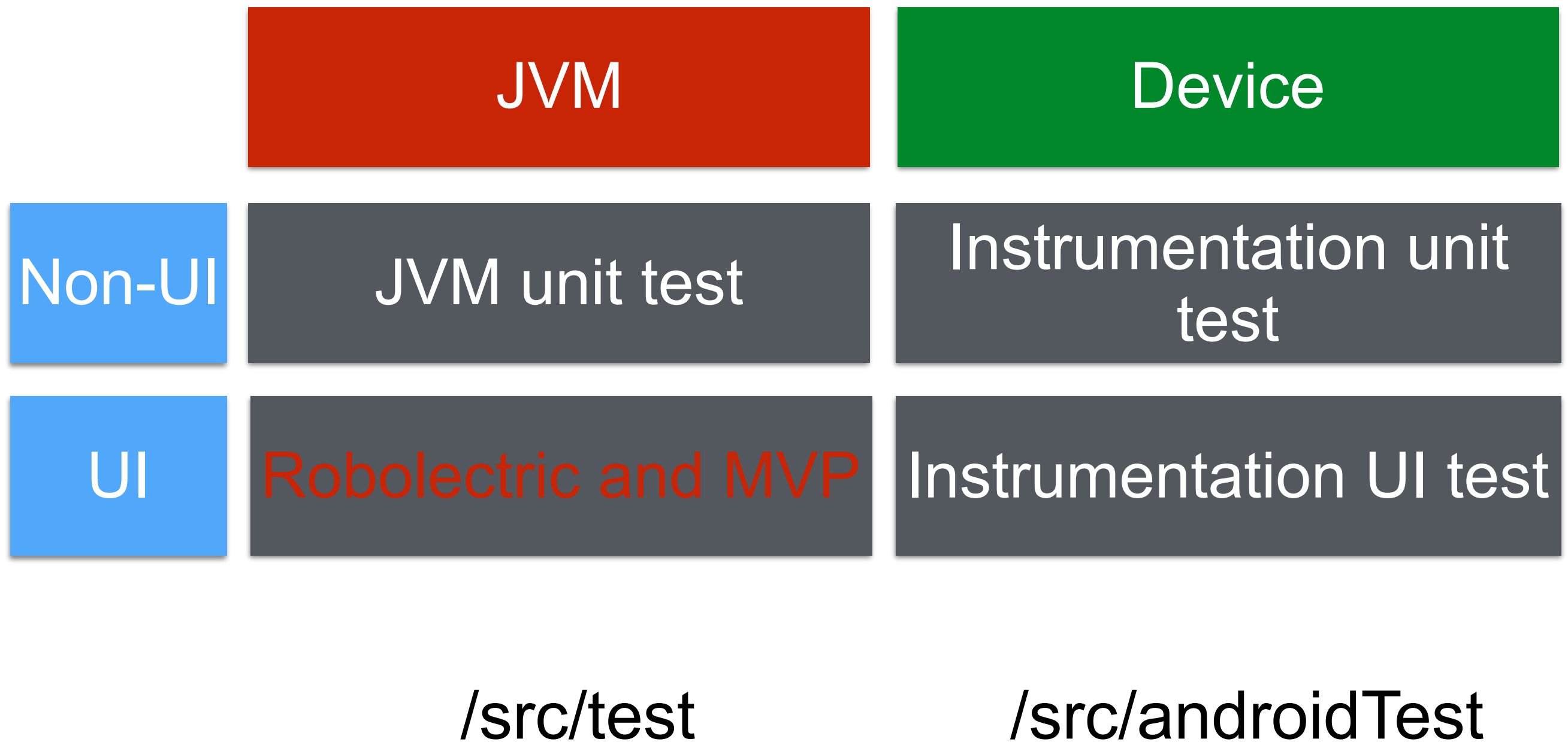


# Read data from file

What type of Android testing ?



# Android Testing



# Read data from file

Q: What type of Android testing ?

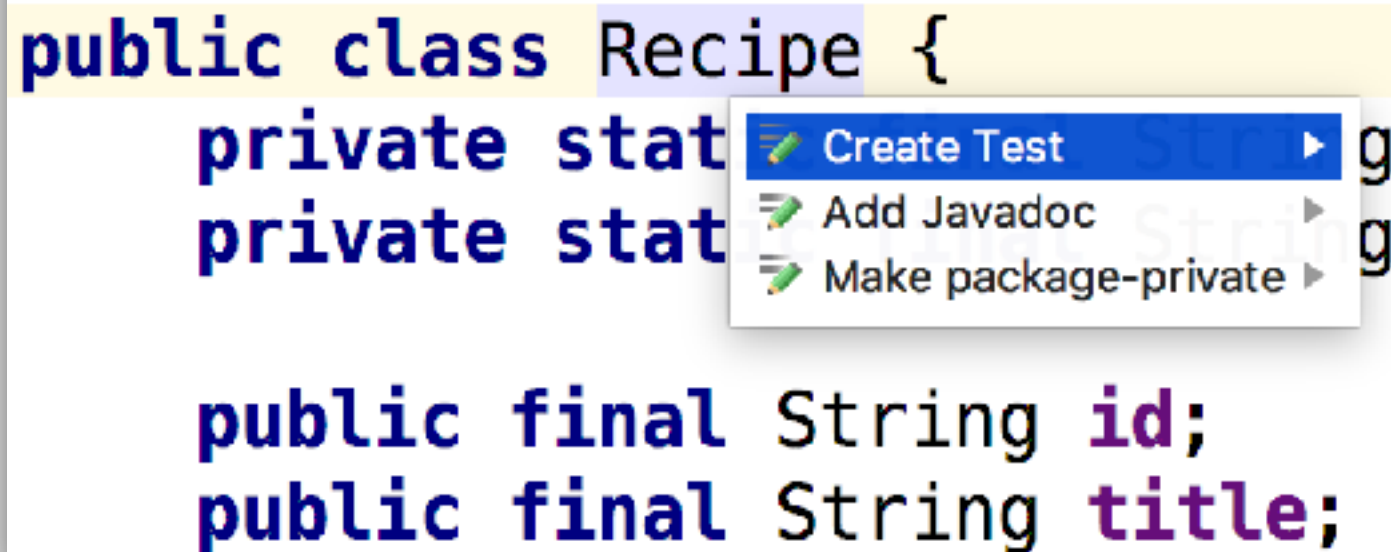
**A: JVM Unit test**



# Read data from file

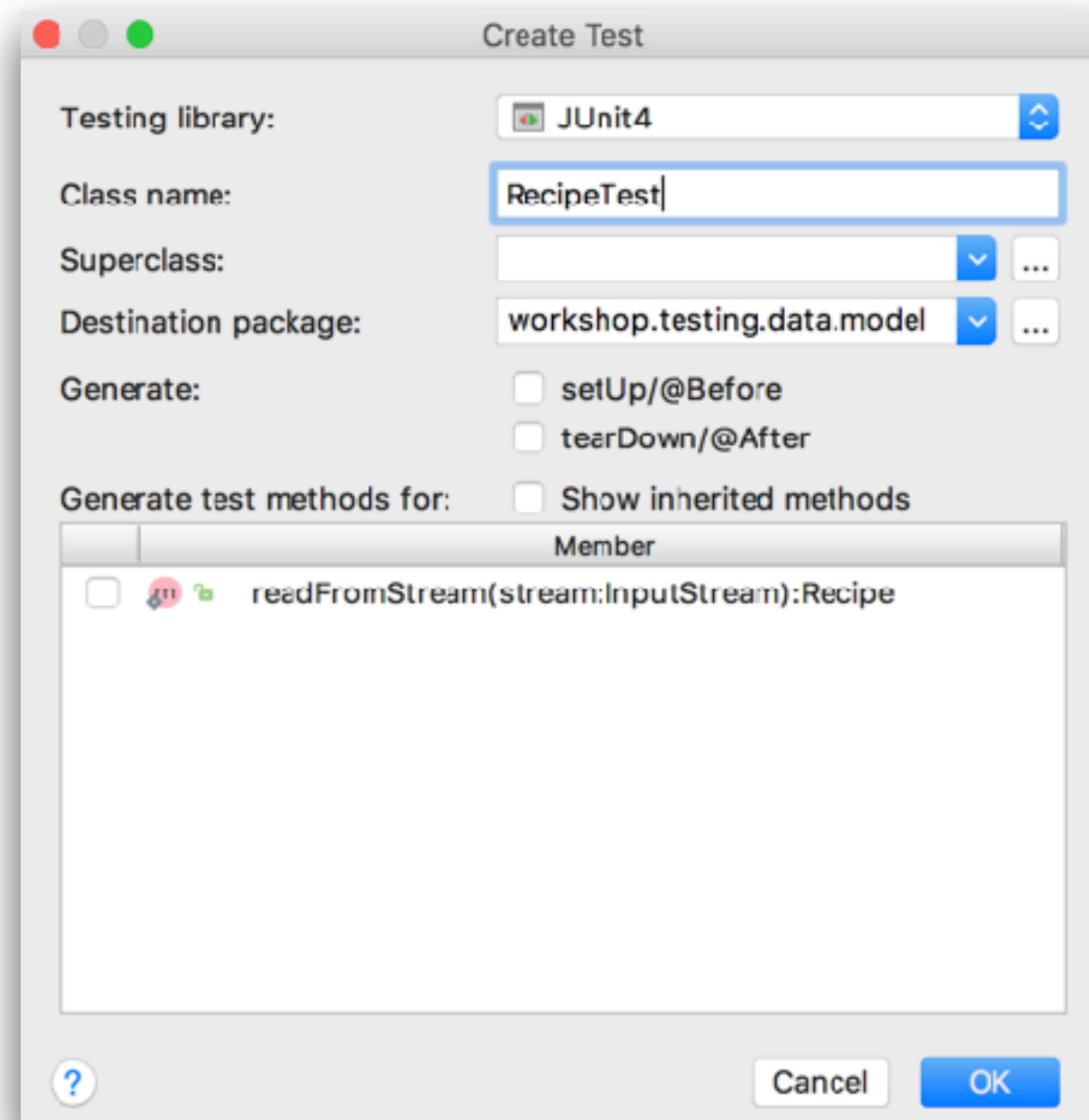
Create the new test class with Recipe  
(ALT + Enter)

```
public class Recipe {  
    private stat  
    private stat  
  
    public final String id;  
    public final String title;
```



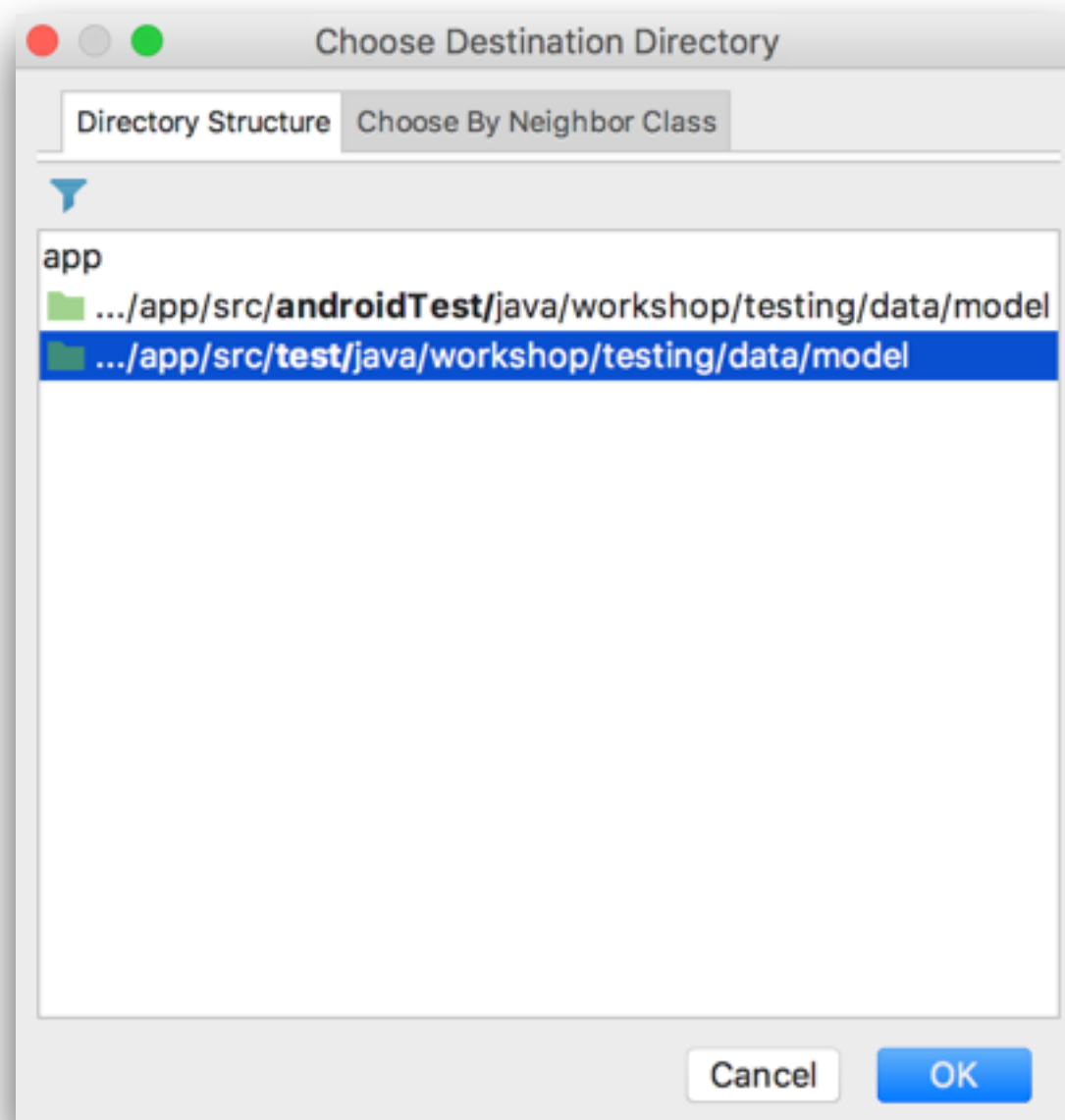
# Read data from file

## Choose JUnit 4



# Read data from file

Choose the destination to **test** directory



# Read data from file

First test case :: read data from water.txt

```
@Test
public void water() {
    InputStream stream
        = RecipeTest.class.getResourceAsStream(name: "/recipes/water.txt");

    Recipe recipe = Recipe.readFromStream(stream);

    assertNotNull(recipe);
    assertEquals(expected: "water", recipe.id);
    assertEquals(expected: "Water", recipe.title);
    assertEquals(expected: "Put glass under tap. Open tap. Close tap. Drink."
        , recipe.description);
}
```





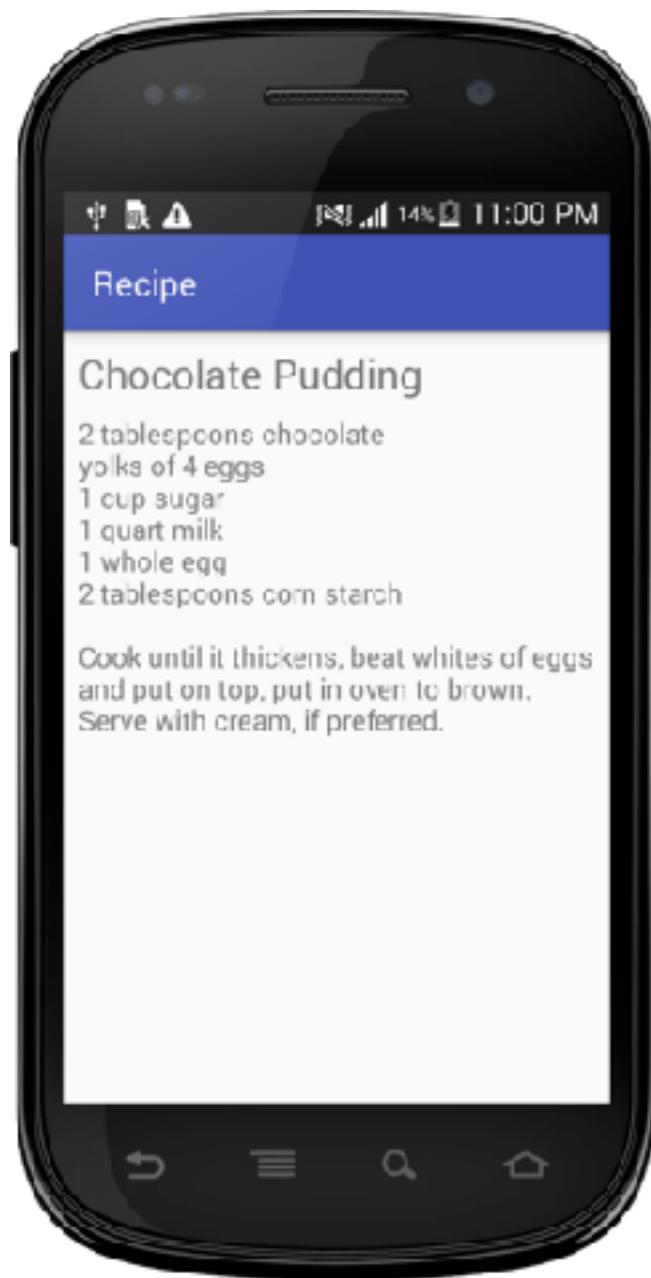
# Check code coverage



## **2. Show detail of recipe in Activity**



# Show detail of recipe

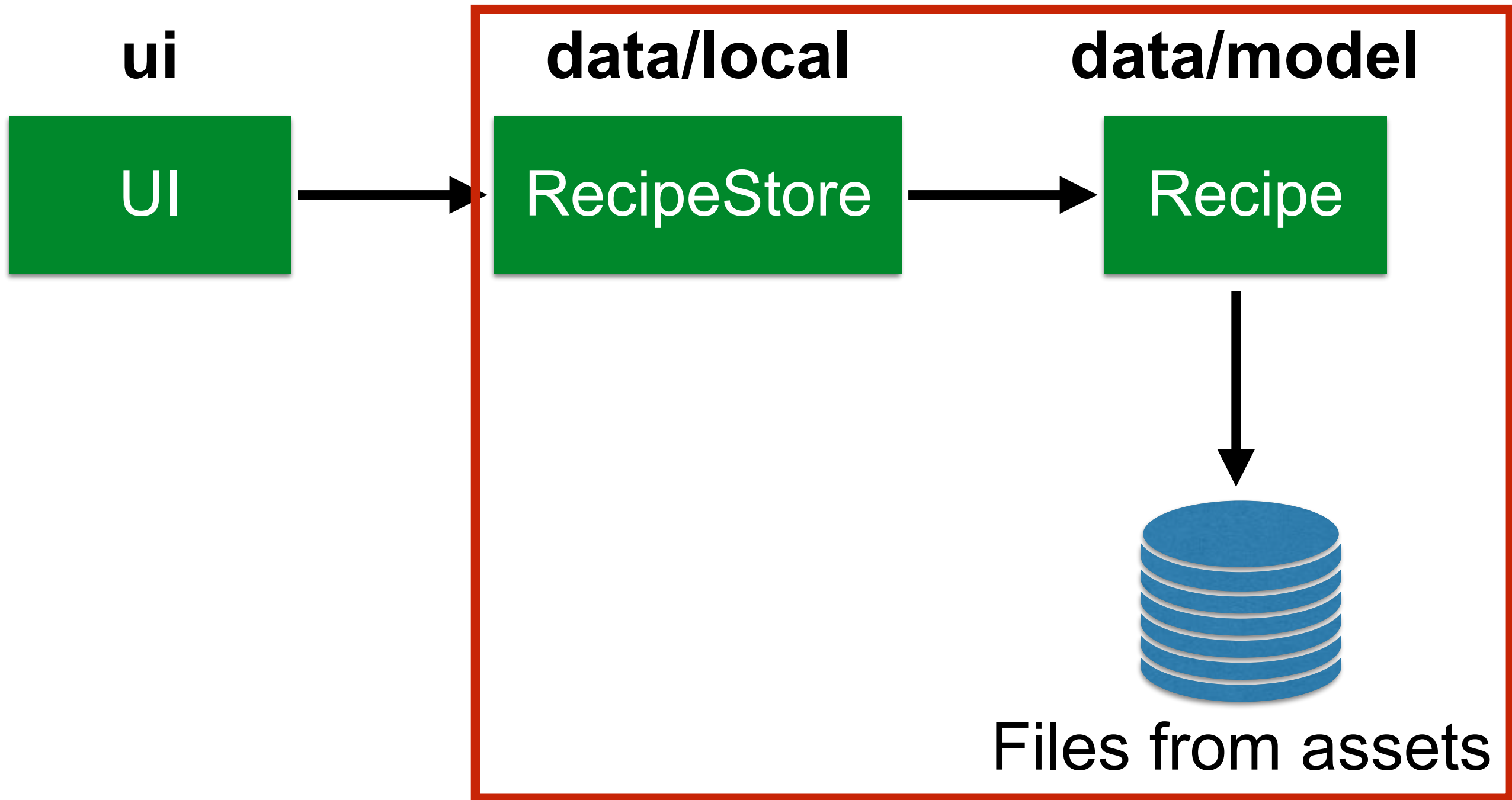


RecipeStore

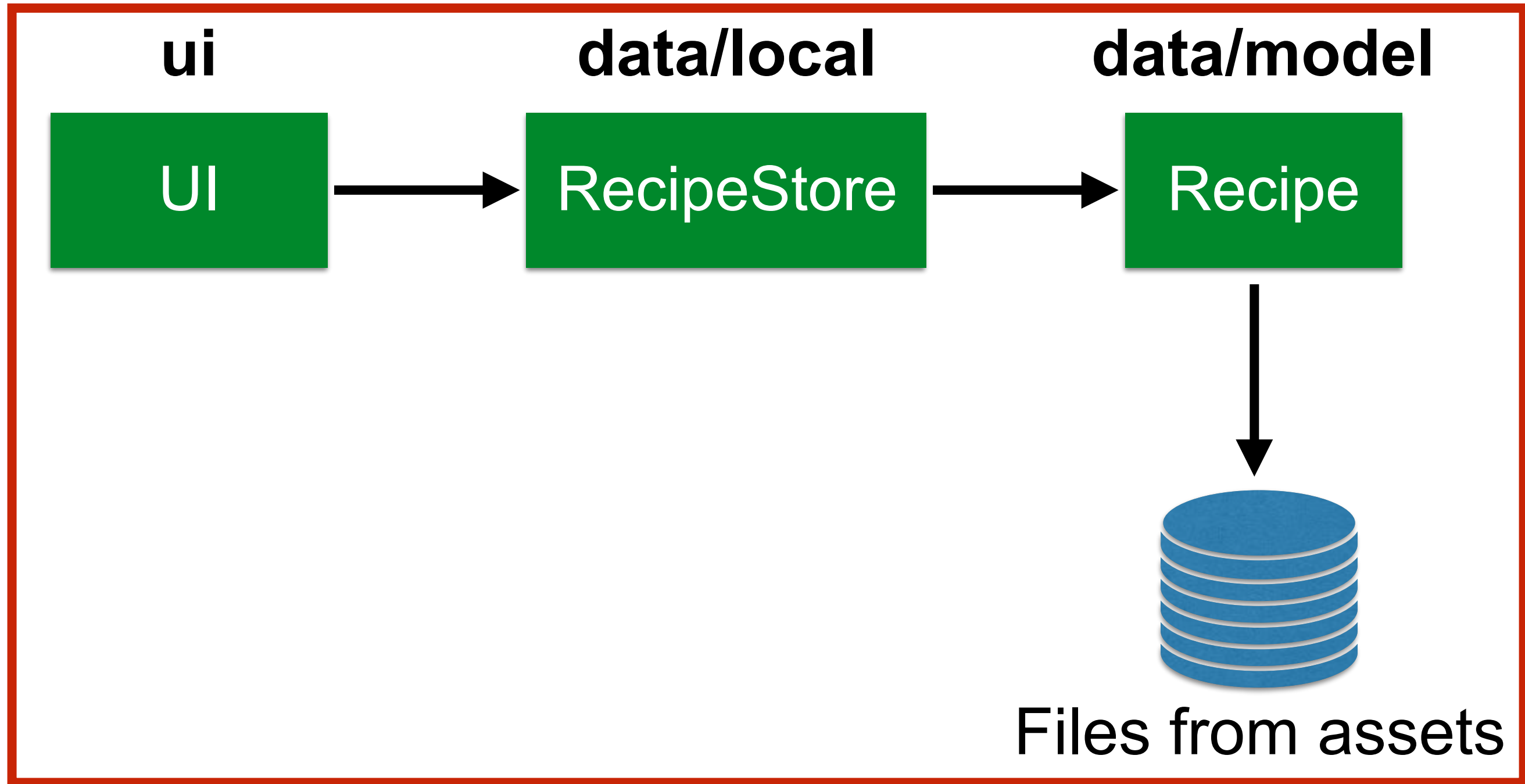
Android  
Framework



# Show detail of recipe (1)



# Show detail of recipe (2)

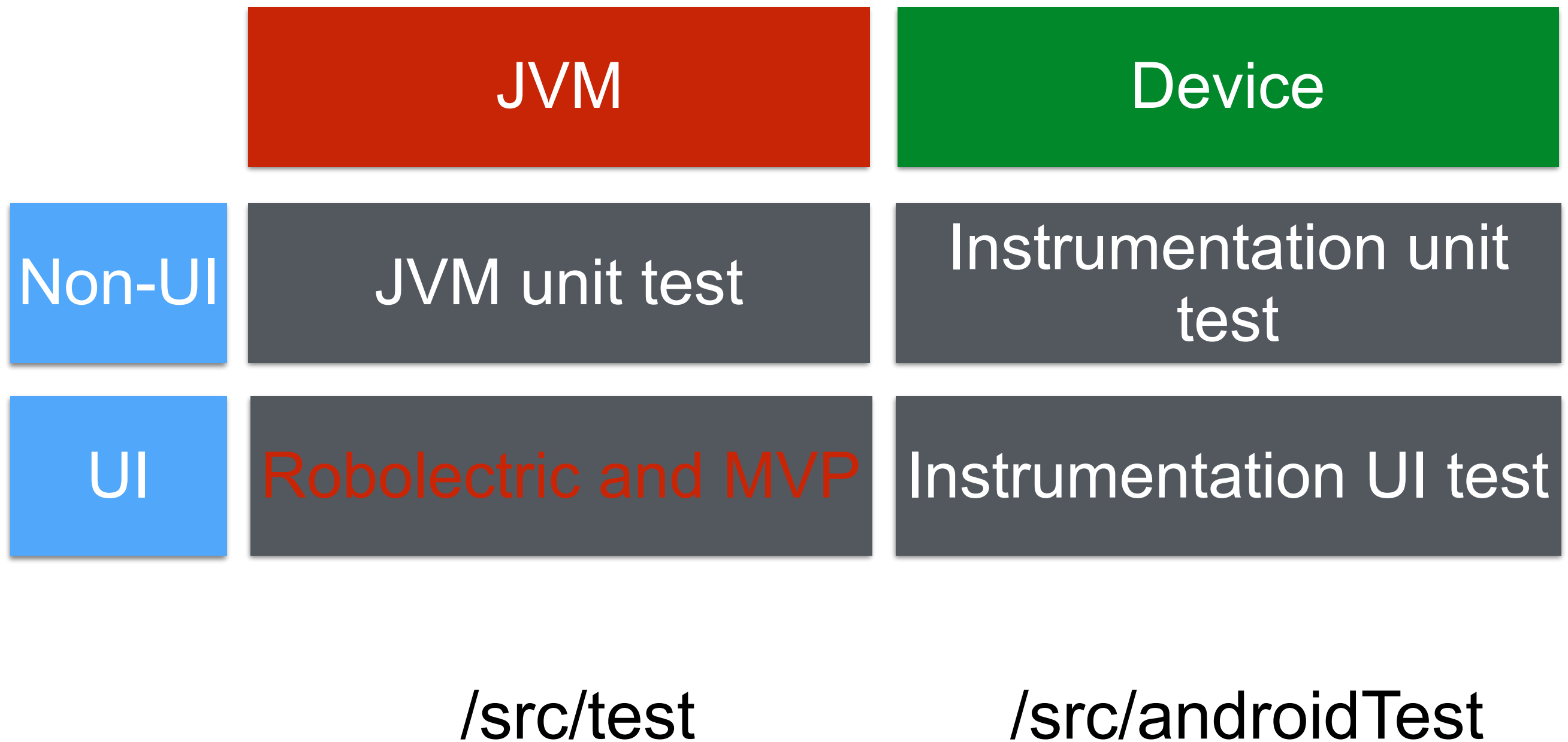


# Show detail of recipe

What type of Android testing ?



# Android Testing



# Show detail of recipe

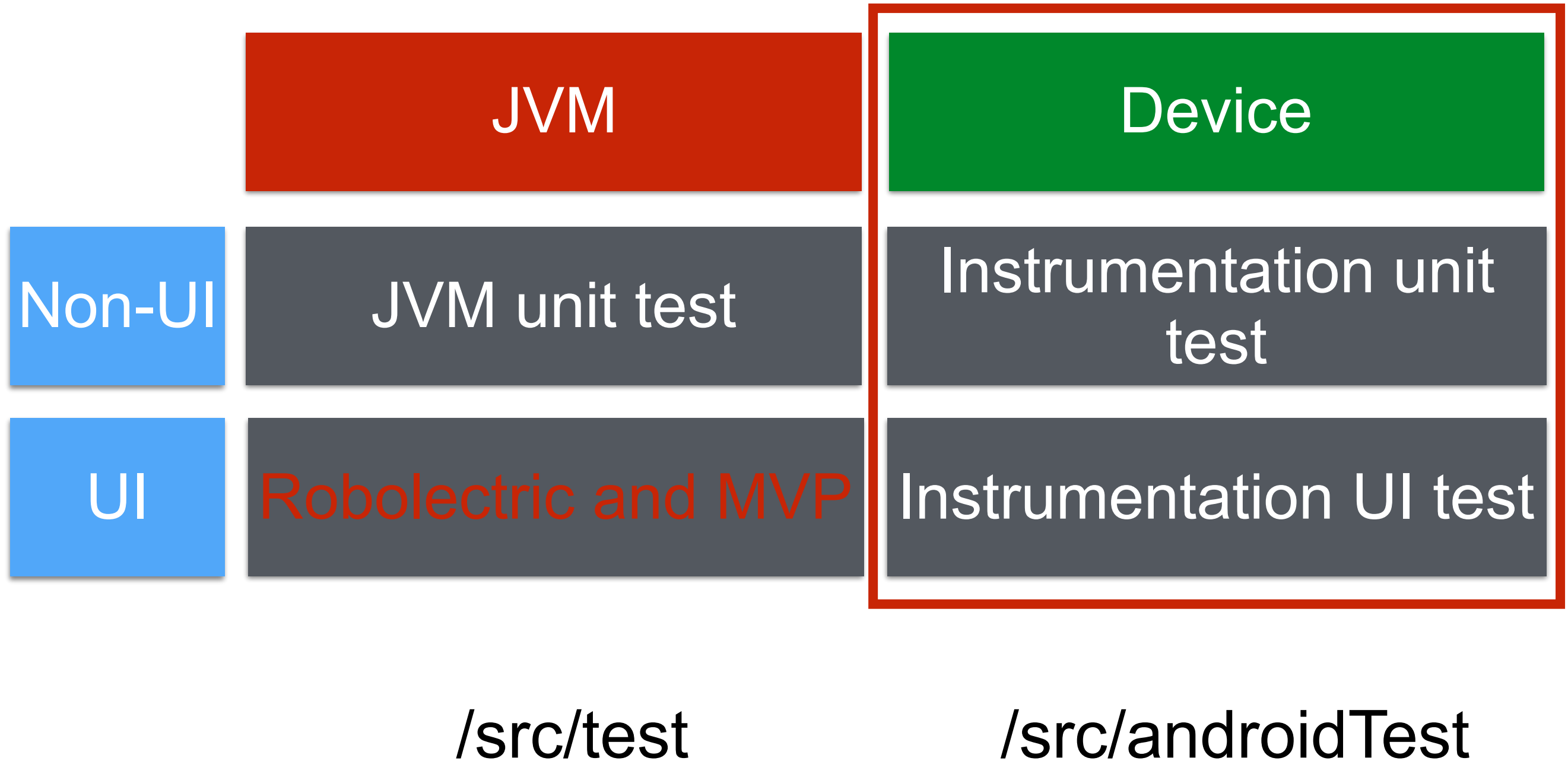
Q: What type of Android testing ?

**A: Separated tests in 2 types**





# Android Testing



# 1. Instrumentation Unit test

Q: What to test ?

**A: Check and verify behavior of  
RecipeStore**



# 1. Instrumentation Unit test

Q: What to test ?

**Number of recipe(s)**

**Get detail of recipe**



# 1. Instrumentation Unit test

Test case :: number of recipe(s)

```
@Test
public void number_of_recipe() {
    Context context = InstrumentationRegistry.getTargetContext();
    RecipeStore store = new RecipeStore(context, directory: "recipes");
    assertNotNull(store);
    assertNotNull(store.recipes);
    assertEquals( expected: 1, store.recipes.size());
}
```



# Check code coverage



# 2. Instrumentation UI test

Q: What to test ?

**A: Choose a recipe and show detail in Activity**



# 2. Instrumentation UI test

Test case :: show detail of recipe in Activity

```
@Rule
public ActivityTestRule<RecipeActivity> activityRule
    = new ActivityTestRule<>(
        RecipeActivity.class, initialTouchMode: true, launchActivity: false);

@Test
public void show_detail_of_chocolate_pudding() {
    Intent intent = new Intent();
    intent.putExtra(RecipeActivity.KEY_ID, value: "chocolate_pudding");
    activityRule.launchActivity(intent);

    onView(withId(R.id.title))
        .check(matches(withText("Chocolate Pudding")));
    onView(withId(R.id.description))
        .check(matches(withText("2 tablespoons chocolate\n" +
            "yolks of 4 eggs\n" +
            "1 cup sugar\n" +
            "1 quart milk\n" +
            "1 whole egg\n" +
            "2 tablespoons corn starch\n" +
            "\n" +
            "Cook until it thickens, beat whites of eggs and pu
        ));
}
```



# Step 1 :: Start activity

```
@Rule
public ActivityTestRule<RecipeActivity> activityRule
    = new ActivityTestRule<>(
        RecipeActivity.class, initialTouchMode: true, launchActivity: false);

@Test
public void show_detail_of_chocolate_pudding() {
    Intent intent = new Intent();
    intent.putExtra(RecipeActivity.KEY_ID, value: "chocolate_pudding");
    activityRule.launchActivity(intent);

    onView(withId(R.id.title))
        .check(matches(withText("Chocolate Pudding")));
    onView(withId(R.id.description))
        .check(matches(withText("2 tablespoons chocolate\n" +
            "yolks of 4 eggs\n" +
            "1 cup sugar\n" +
            "1 quart milk\n" +
            "1 whole egg\n" +
            "2 tablespoons corn starch\n" +
            "\n" +
            "Cook until it thickens, beat whites of eggs and pu
```





# Step 2 :: Create new test case

```
@Rule
public ActivityTestRule<RecipeActivity> activityRule
    = new ActivityTestRule<>(
        RecipeActivity.class, initialTouchMode: true, launchActivity: false);
```

```
@Test
public void show_detail_of_chocolate_pudding() {
    Intent intent = new Intent();
    intent.putExtra(RecipeActivity.KEY_ID, value: "chocolate_pudding");
    activityRule.launchActivity(intent);

    onView(withId(R.id.title))
        .check(matches(withText("Chocolate Pudding")));
    onView(withId(R.id.description))
        .check(matches(withText("2 tablespoons chocolate\n" +
            "yolks of 4 eggs\n" +
            "1 cup sugar\n" +
            "1 quart milk\n" +
            "1 whole egg\n" +
            "2 tablespoons corn starch\n" +
            "\n" +
            "Cook until it thickens, beat whites of eggs and pu
```



# Step 3 :: Pass data with intent

```
@Rule
public ActivityTestRule<RecipeActivity> activityRule
    = new ActivityTestRule<>(
        RecipeActivity.class, initialTouchMode: true, launchActivity: false);

@Test
public void show_detail_of_chocolate_pudding() {
    Intent intent = new Intent();
    intent.putExtra(RecipeActivity.KEY_ID, value: "chocolate_pudding");
    activityRule.launchActivity(intent);

    onView(withId(R.id.title))
        .check(matches(withText("Chocolate Pudding")));
    onView(withId(R.id.description))
        .check(matches(withText("2 tablespoons chocolate\n" +
            "yolks of 4 eggs\n" +
            "1 cup sugar\n" +
            "1 quart milk\n" +
            "1 whole egg\n" +
            "2 tablespoons corn starch\n" +
            "\n" +
            "Cook until it thickens, beat whites of eggs and pu
        ));
}
```



# Step 4 :: Verify data in activity

```
@Rule
public ActivityTestRule<RecipeActivity> activityRule
    = new ActivityTestRule<>(
        RecipeActivity.class, initialTouchMode: true, launchActivity: false);

@Test
public void show_detail_of_chocolate_pudding() {
    Intent intent = new Intent();
    intent.putExtra(RecipeActivity.KEY_ID, value: "chocolate_pudding");
    activityRule.launchActivity(intent);
```

```
onView(withId(R.id.title))
    .check(matches(withText("Chocolate Pudding")));
onView(withId(R.id.description))
    .check(matches(withText("2 tablespoons chocolate\n" +
        "yolks of 4 eggs\n" +
        "1 cup sugar\n" +
        "1 quart milk\n" +
        "1 whole egg\n" +
        "2 tablespoons corn starch\n" +
        "\n" +
        "Cook until it thickens, beat whites of eggs and pu
```



# Check code coverage



# Assignments/Homework



# Assignments/Homework

List of recipes from assets

Add more tests/more code coverage

Add more features such as Favorite

Push all changes to your Github repository



# Summary of this course



# Resources

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

<https://developer.android.com/topic/libraries/testing-support-library/index.html#Espresso>





# Test-Driven Development

