Understanding Automated UI Tests

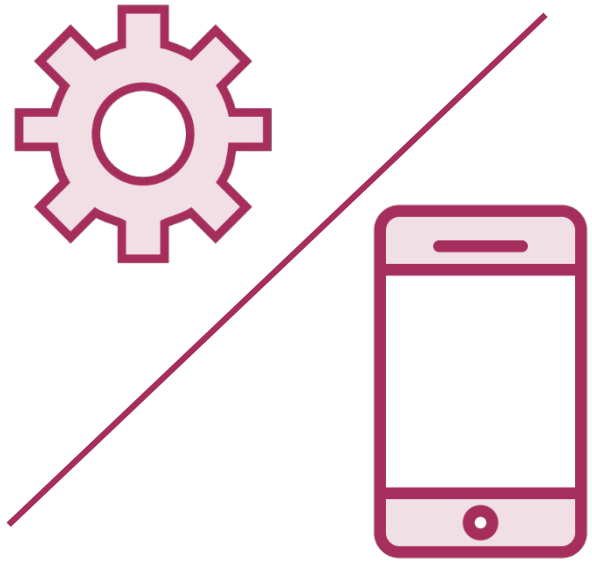
Jim Wilson
MOBILE SOLUTIONS DEVELOPER & ARCHITECT

@hedgehogjim blog.jwhh.com

What to Expect from This Module

- Instrumented testing overview
- Implementing instrumented tests
- Setting up UI tests
- Basic UI test interactions

Android Testing



Android apps

Logic-based behavior

Android-based behavior

Testing logic-based behavior

Local JVM tests

Testing Android-based behavior

Instrumented tests

Instrumented Tests

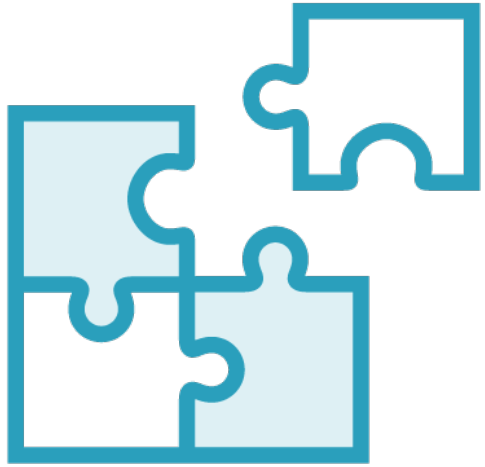
Full Android Environment

Run on emulator or physical device

Instrumented unit tests

Unit tests

Rely on Android features/capabilities

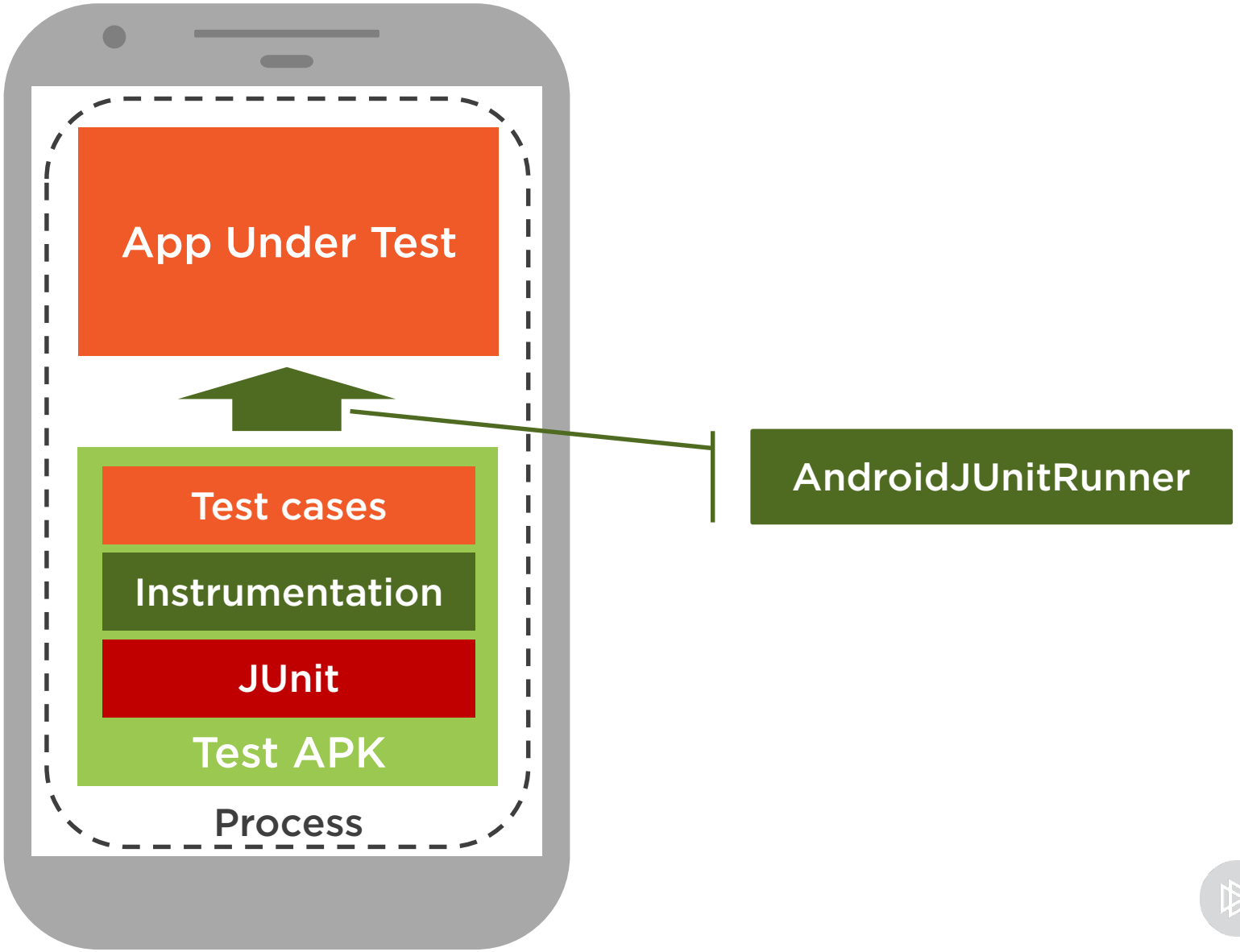


Automated UI tests

Integration test

App behavior in response to UI actions

Instrumented Tests



Implementing Instrumented Tests

Leverages JUnit

Grouped in classes

Same function annotations

Uses Assert class

Indicates expectations

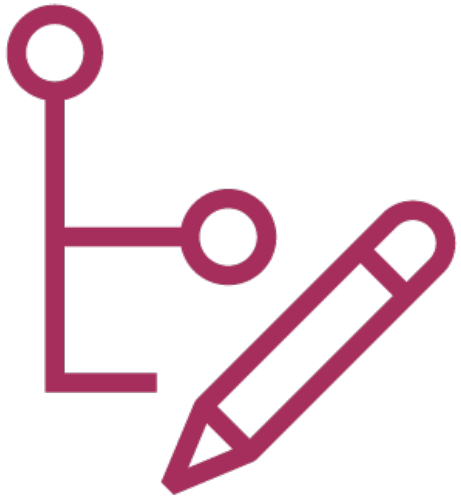
Fails test if not met

Managed with Android Studio

Can run or debug tests

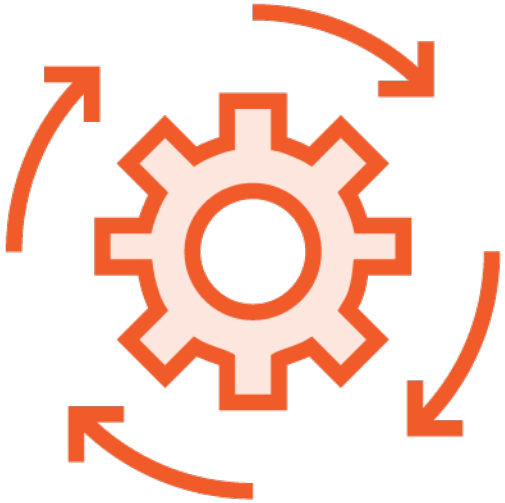
Displays test results

Implementing Instrumented Tests



Separate from JVM tests

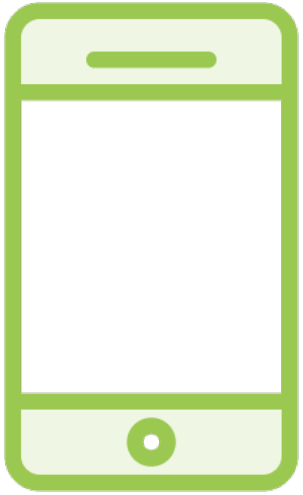
In androidTest source set



Relies on a test runner

Mark class with @RunWith

Pass AndroidJUnit4::class



Requires Android environment

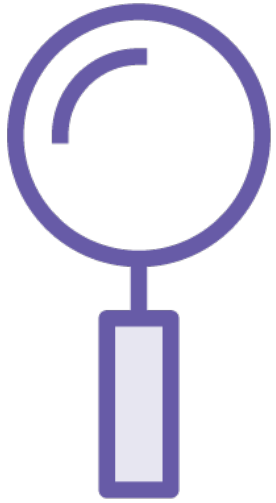
Run on emulator or physical device

Implementing Instrumented Tests

```
@RunWith(AndroidJUnit4::class)
class MyExampleTestClass {

  @Before
  fun testSetUp() { . . . }

  @Test
  fun myTestMethod() {

    // Android dependent test code

  }
}
```

Creating UI Test Interactions

UI tests require a series of view interactions



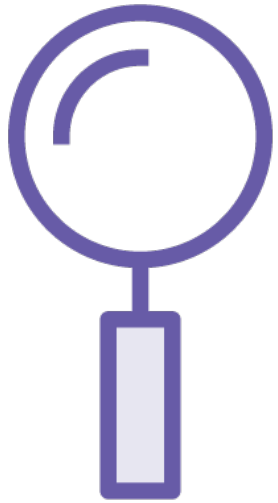
Need way to specify view of interest

Need way to specify action on the view

Espresso UI testing framework

Creating UI Test Interactions

Espresso.onView method



Accepts Matcher

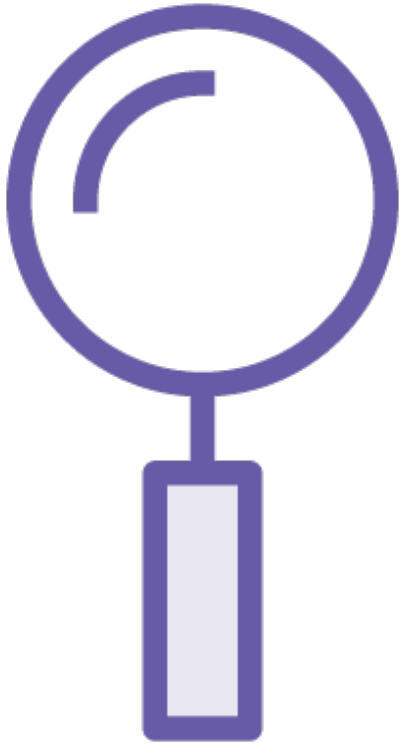
Specifies the view matching criteria

Returns ViewInteraction

Associated with matching view

Used to perform action on view

Specifying View of Interest



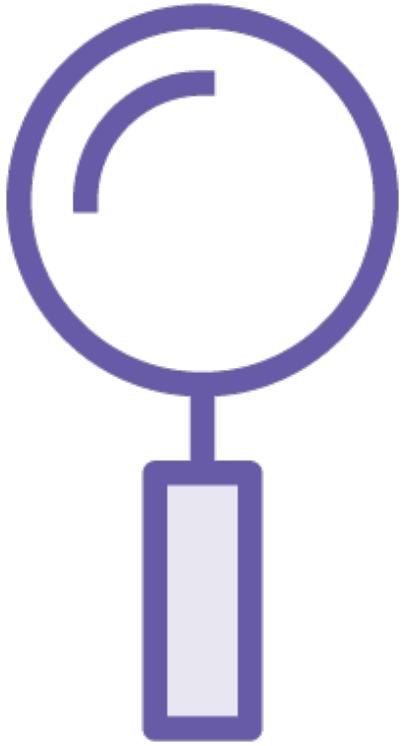
Hamcrest matchers

- Provides declarative matching
- General purpose Java-based framework
- http://hamcrest.org

ViewMatchers class

- Provides matchers for Android Views
- Methods return a Hamcrest matcher
- Easily combined with Hamcrest general purpose matchers

Example ViewMatchers Methods



withId

- Match views based on id property

withText

- Match views based on text property

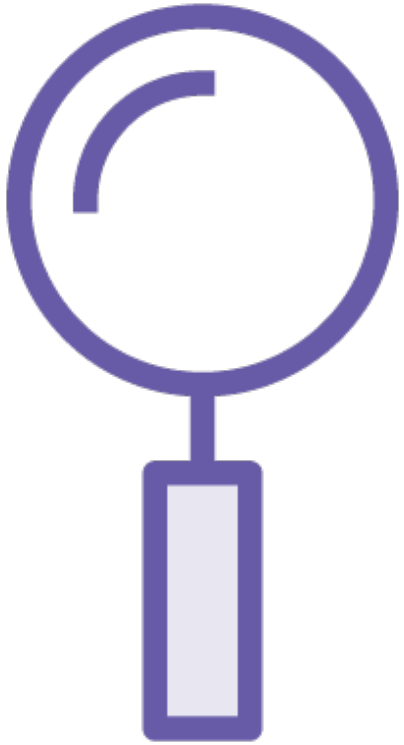
isDisplayed

- Match views currently on screen

isChecked

- Match currently checked checkable views (Switch, CheckBox, etc.)

Example Hamcrest Matchers



equalTo

- Match based on equals method

instanceOf

- Match based on class type

allOf

- Accepts multiple Matchers
- Match if all Matchers match

anyOf

- Accepts multiple Matchers
- Match if any Matchers match

Performing View Action

ViewInteraction.perform method

- Performs one or more specified actions
- Specific action passed as a parameter

ViewActions class

- Provides action methods
- Each method returns specified action

Example ViewActions Methods

click

- Click on the view

typeText

- Type text into view

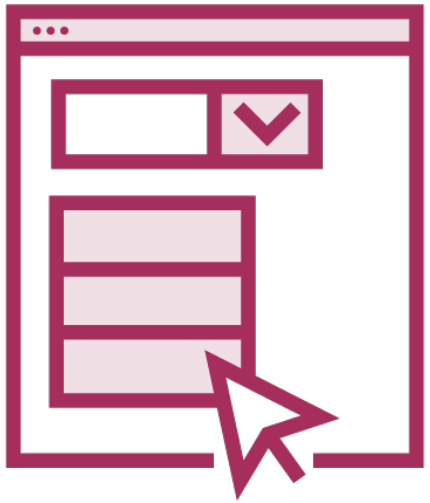
replaceText

- Replace view's text

closeSoftKeyboard

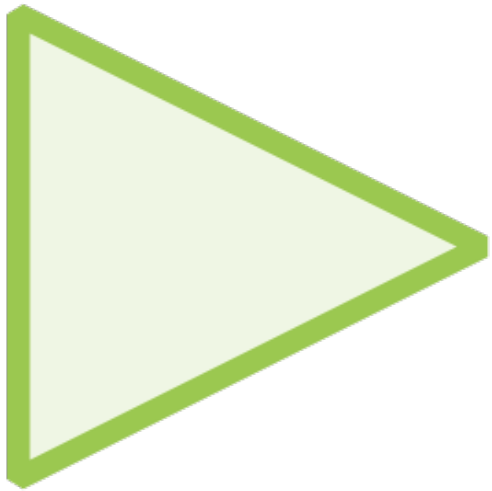
- Closes the soft keyboard

Starting the Target Activity



ActivityTestRule

Automates test activity lifetime



Starts activity before each test

Terminates activity after each test

Activity life includes @Before/@After methods

Summary

Instrumented tests

- Run on an emulator or device
- Have full Android environment

Types of instrumented tests

- Unit tests that rely on Android
- Automated UI tests

Summary

Instrumented tests use JUnit

- Test methods marked with @Test
- Support pre/post-processing methods
- Mark test class with @RunWith passing AndroidJUnit4::class.java

Automated UI tests

- A type of instrumented test
- Generally use ActivityTestRule

Summary

Espresso.onView method

- Locates view based on view criteria
- Returns a ViewInteraction reference

Criteria based on matchers

- Hamcrest matchers
- ViewMatchers class

Performing UI actions

- ViewInteraction.perform method
- Actions specified with ViewActions