Google Developers

What’s new in Android Testing

github.com/googlesamples/android-testing

Developer Platforms Engineer @Google

+Stephan Linzner
@onlythoughtwork

The next generation of Android Testing Tools

Unit Test Support
AndroidJUnitRunner
Espresso
Espresso-Intents

Unit Test Support

a.k.a JVM Tests
a.k.a Local Tests

Before unit test support

Running tests **on device/emulator** was slow

Build, deploy, make an espresso, run tests

Stub implementations in **android.jar**

```
Error java.lang.RuntimeException: Stub!
```

Framework limitations

Final classes

Unit test support for the Android Gradle Plugin

`com.android.tools.build:gradle:1.1+`

New source set **`test/`** for unit tests

Mockable android.jar

Mockito to stub dependencies into the Android framework

app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    testOptions {
        unitTests.returnDefaultValues = true // Caution!
    }
}

dependencies {
    // Unit testing dependencies
    testCompile 'junit:junit:4.12'
    testCompile 'org.mockito:mockito-core:1.10.19'
}
```

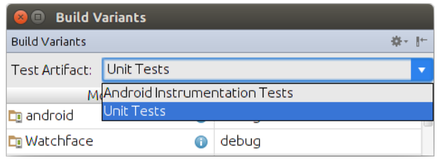
Unit test sample

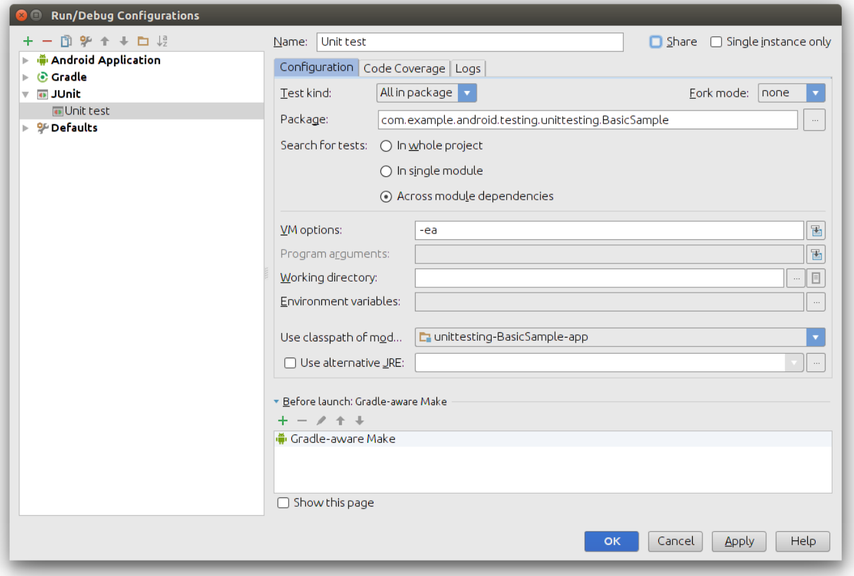
```
@RunWith(MockitoJUnitRunner.class)
@SmallTest
public class UnitTestSample {

    private static final String FAKE_STRING = "HELLO WORLD";

    @Mock
    Context mMockContext;

    @Test
    public void readStringFromContext_LocalizedString() {
        // Given a mocked Context injected into the object under test...
        when(mMockContext.getString(R.string.hello_word)).thenReturn(FAKE_STRING);
        ClassUnderTest myObjectUnderTest = new ClassUnderTest(mMockContext);

        // ...when the string is returned from the object under test...
        String result = myObjectUnderTest.getHelloWorldString();

        // ...then the result should be the expected one.
        assertThat(result, is(FAKE_STRING));
    }
}
```

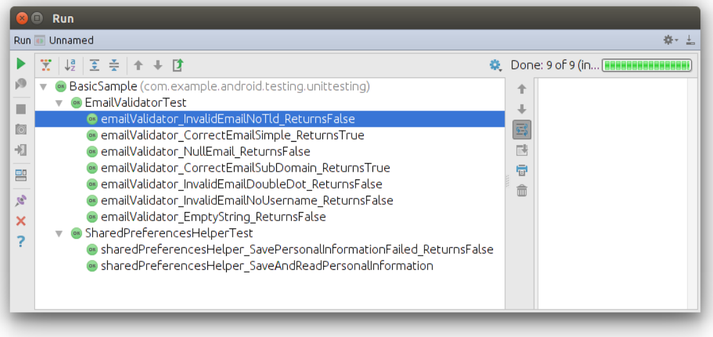
Use MockitoJUnitRunner for easier initialization of @Mock fields.

Dependency Injection

Android Studio







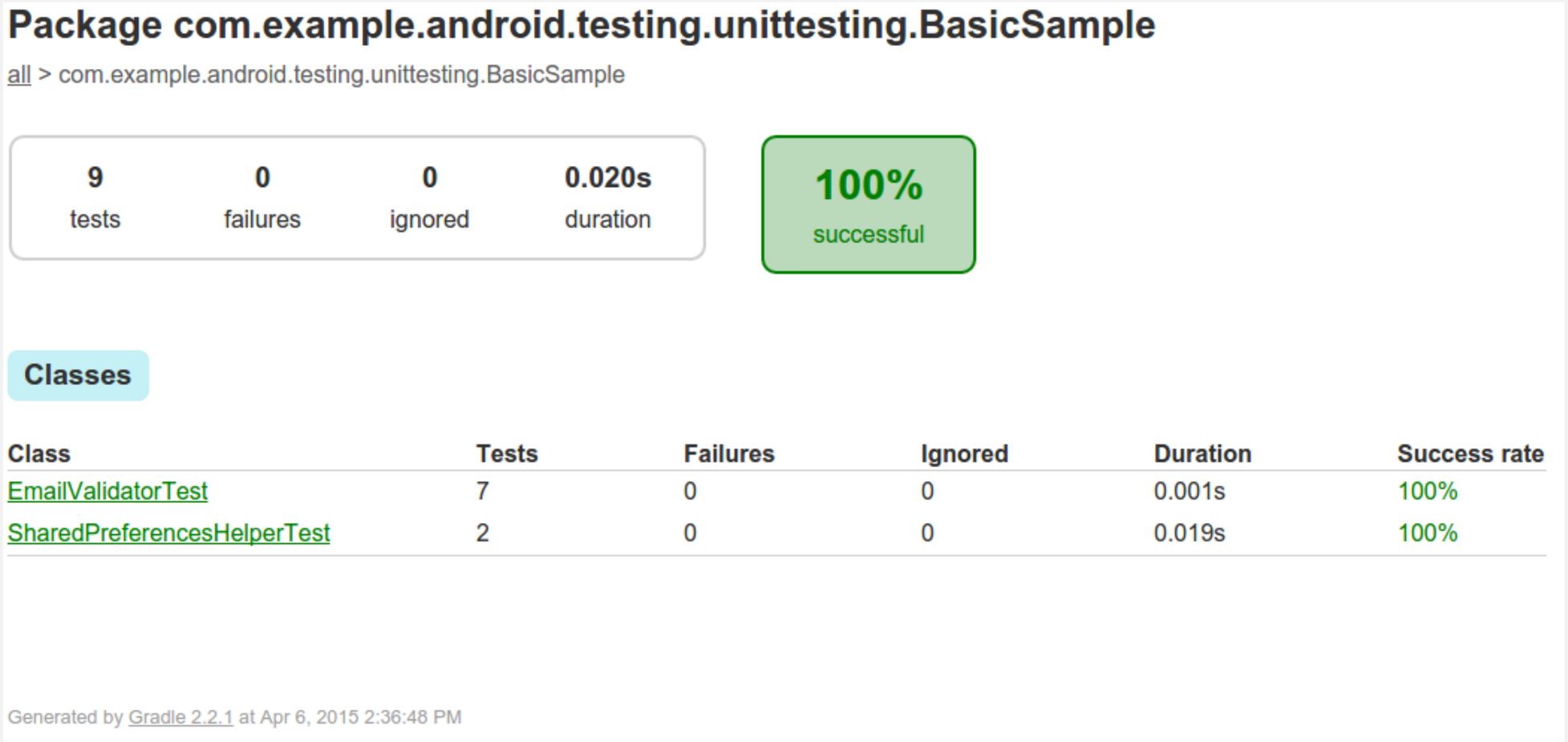
Command-line

```
$ ./gradlew test
...
:app:mockableAndroidJar
...
:app:assembleDebugUnitTest
:app:testDebug
:app:test

BUILD SUCCESSFUL

Total time: 3.142 secs
```

Reports

app/build/reports/**tests**/debug/index.**html**
app/build/**test-results**/debug/TEST-com.example.android.testing.unittesting.BasicSample.EmailValidatorTest.**xml**

New in 1.1+, XML Reports



Package com.example.android.testing.unittesting.BasicSample

all > com.example.android.testing.unittesting.BasicSample

9	0	0	0.020s
tests	failures	ignored	duration

100% successful

Classes

Class	Tests	Failures	Ignored	Duration	Success rate
EmailValidatorTest	7	0	0	0.001s	100%
SharedPreferencesHelperTest	2	0	0	0.019s	100%

Generated by Gradle 2.2.1 at Apr 6, 2015 2:36:48 PM

Limitations

Tight coupling with Android

When Mockito is not enough

Stubbing static methods

`TextUtils.*, Log.*, etc.`

Workarounds

Tight coupling with Android

Revisit your design decisions

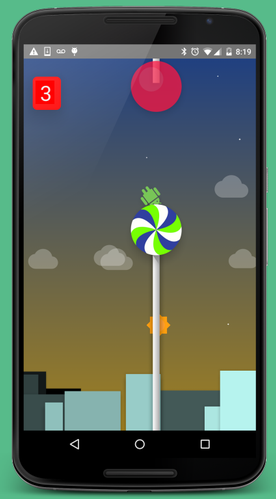
Run unit tests on device or emulator

Stubbing static methods

Wrapper

PowerMockito

```
unitTests.returnDefaultValues = true
```



Android Testing Support Library

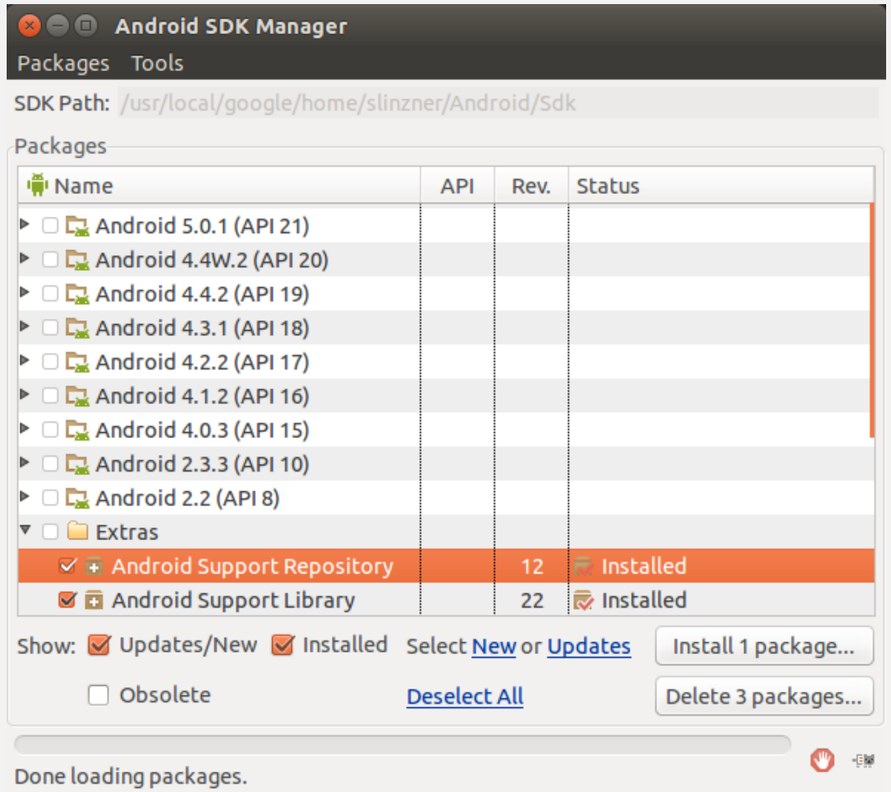
a.k.a Instrumentation Tests
a.k.a Device or Emulator Tests

Instrumentation Tests

Run on Device or Emulator
Run With AndroidJUnitRunner
Located androidTest/ source set

Android SDK Manager

Open Sdk Manager from Android Studio

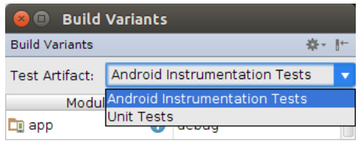


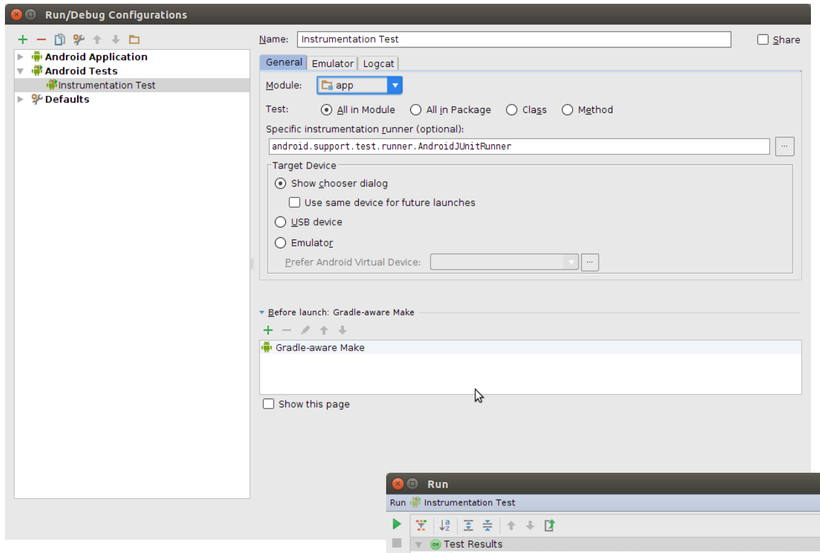
Download latest Support Repository

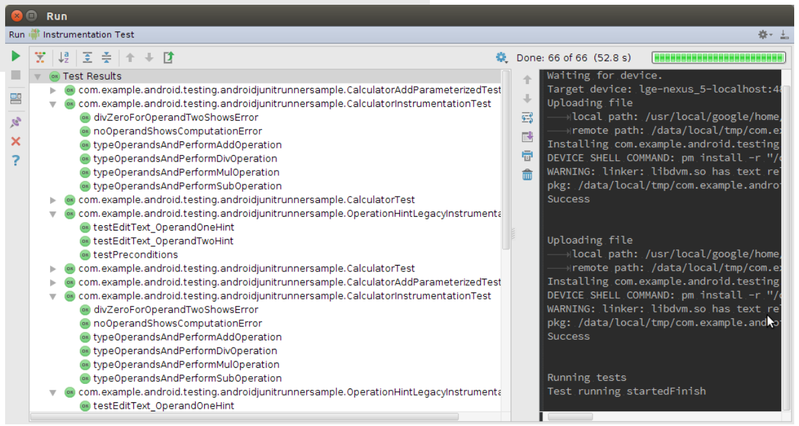
app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    // AndroidJUnit Runner dependencies
    androidTestCompile 'com.android.support.test:runner:0.2'
}
```

Android Studio







Command-line

```
$./gradlew connectedAndroidTest
...
:app:assembleDebug
...
:app:assembleDebugAndroidTest
:app:connectedAndroidTest

BUILD SUCCESSFUL

Total time: 59.152 secs
```

App Under Test

Android Test App

Reports

app/build/outputs/**reports**/androidTests/connected/index.**html**
app/build/outputs/**androidTest-results**/connected/TEST-Nexus-6-5.1-app-flavor.**xml**

New in 1.1+, XML Reports

Package com.example.android.testing.androidjunitrunnersample

all > com.example.android.testing.androidjunitrunnersample

22	0	16.216s
tests	failures	duration

100% successful

Classes

Class	Tests	Failures	Duration	Success rate
CalculatorAddParameterizedTest	7	0	0s	100%
CalculatorInstrumentationTest	6	0	12.795s	100%
CalculatorTest	6	0	0.027s	100%
OperationHintLegacyInstrumentationTest	3	0	3.394s	100%

Generated by Gradle 2.2.1 at Apr 6, 2015 1:46:31 PM

AndroidJUnitRunner

AndroidJUnitRunner

A new test runner for Android
JUnit3/JUnit4 Support
Instrumentation Registry
Test Filtering
Intent Monitoring/Stubbing
Activity/Application Lifecycle Monitoring

JUnit4

```
@RunWith(AndroidJUnit4.class)
@SmallTest
public class DroidconItalyTest {
  Droidcon mDroidcon;

  @Before
  public void initDroidcon() {
    mDroidcon = Droidcons.get(Edition.ITALY);
    mDroidcon.init();
  }

  @Test
  public void droidcon_IsAwesome_ReturnsTrue() {
    assertThat(mDroidcon.isAwesome(), is(true));
  }

  @After
  public void releaseDroidcon() {
    mDroidcon.release();
  }
}
```

JUnit4 test need to be annotated with AndroidJUnit4.class

Use @Before to setup your test fixture

Annotate all tests with @Test

Use @After to release any resource

Instrumentation Registry

```
@Before
public void accessAllTheThings() {
  mArgsBundle = InstrumentationRegistry.getArguments();
  mInstrumentation = InstrumentationRegistry.getInstrumentation();
  mTestAppContext = InstrumentationRegistry.getContext();
  mTargetContext = InstrumentationRegistry.getTargetContext();
}
```

Test Filters

```
@SdkSuppress(minSdkVersion=15)
@Test
public void featureWithMinSdk15() {
  ...
}
```

Suppress test to run on certain target Api levels

```
@RequiresDevice
@Test
public void SomeDeviceSpecificFeature() {
  ...
}
```

Filter tests that can only run on a (physical) device

JUnit4 Rules

app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    // AndroidJUnit Runner dependencies
    androidTestCompile 'com.android.support.test:runner:0.2'
    androidTestCompile 'com.android.support.test:rules:0.2'
}
```

```
@Deprecated
public class ActivityInstrumentationTestCase2
```

ActivityInstrumentationTestCase2 vs. ActivityTestRule

Before

```
@LargeTest
public class DroidconTest extends ActivityInstrumentationTestCase2<Droidcon> {

    public DroidconTest() {
        super(Droidcon.class);
    }

    @Override
    protected void setUp() throws Exception {
        super.setUp();
        getActivity();
    }

    public void testDroidcon_awesomness() {}
}
```

After

```
@RunWith(AndroidJUnit4.class)
@LargeTest
public class DroidconTest {

    @Rule
    public ActivityTestRule<Droidcon> mActivityRule = new ActivityTestRule<>(Droidcon.class);

    @Test
    public void droidcon_awesomness() {}
}
```

ActivityTestRule Sample

https://github.com/googlesamples/android-testing/tree/master/espresso/BasicSample

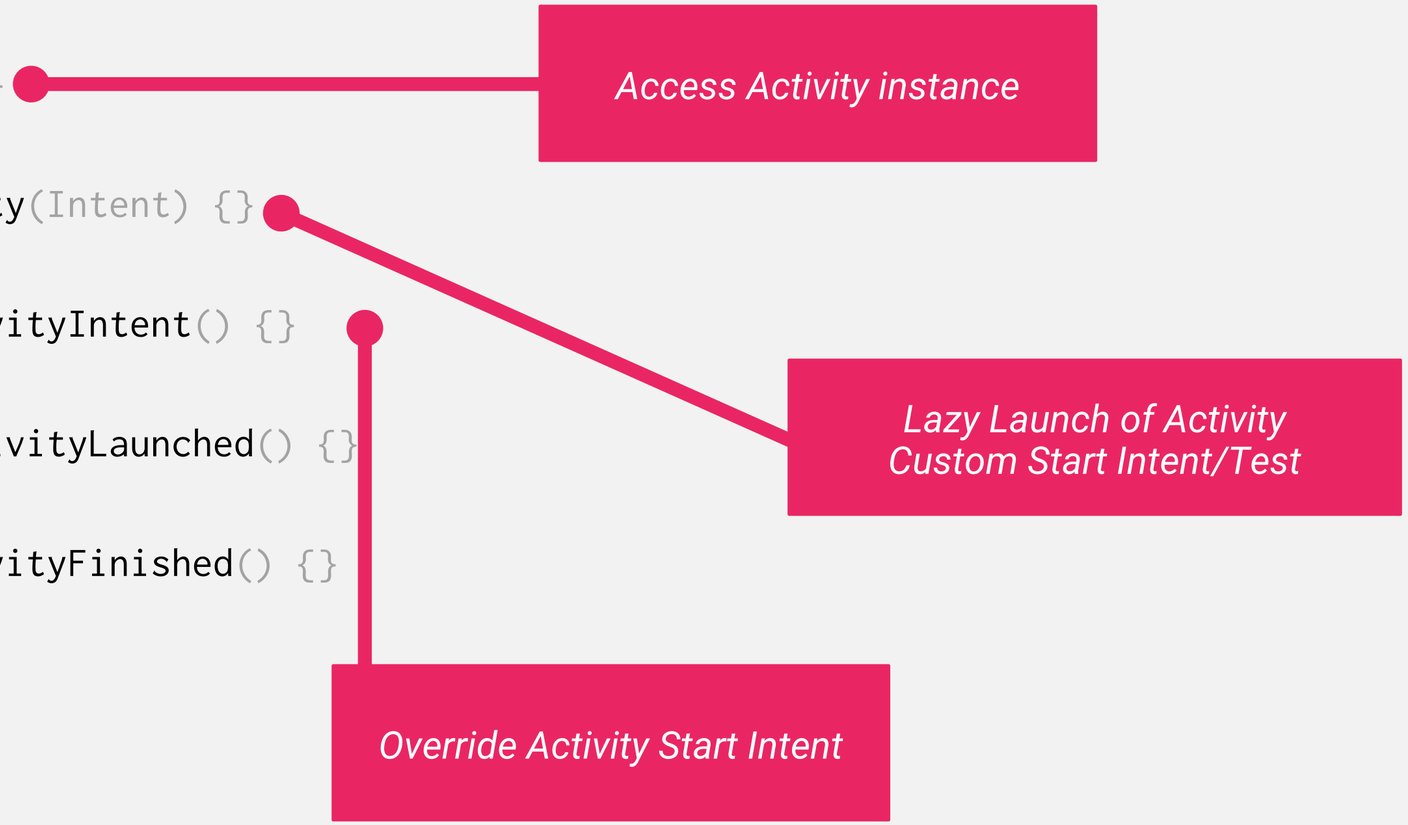
```
@RunWith(AndroidJUnit4.class)
@LargeTest
public class ChangeTextBehaviorTest {

    ...

    @Rule
    public ActivityTestRule<MainActivity> mActivityRule = new ActivityTestRule<>(MainActivity.class);

    @Test
    public void changeText_sameActivity() {
        // Type text and then press the button.
        onView(withId(R.id.editTextUserInput))
                .perform(typeText(STRING_TO_BE_TYPED), closeSoftKeyboard());
        onView(withId(R.id.changeTextBt)).perform(click());

        // Check that the text was changed.
        onView(withId(R.id.textToBeChanged)).check(matches(withText(STRING_TO_BE_TYPED)));
    }
}
```

Use @Rule annotation

Create an ActivityTestRule for your Activity

ActivityTestRule

```
public class ActivityTestRule<T extends Activity> extends UiThreadTestRule {

    public T getActivity() {}

    public void launchActivity(Intent) {}

    protected Intent getActivityIntent() {}

    protected void beforeActivityLaunched() {}

    protected void afterActivityFinished() {}

}
```



ServiceTestRule Sample

```
@RunWith(AndroidJUnit4.class)
@MediumTest
public class MyServiceTest {

    @Rule
    public final ServiceTestRule mServiceRule = new ServiceTestRule();

    @Test
    public void testWithStartedService() {
        mServiceRule.startService(
            new Intent(InstrumentationRegistry.getTargetContext(), MyService.class));
        // test code
    }

    @Test
    public void testWithBoundService() {
        IBinder binder = mServiceRule.bindService(
            new Intent(InstrumentationRegistry.getTargetContext(), MyService.class));
        MyService service = ((MyService.LocalBinder) binder).getService();
        assertTrue("True wasn't returned", service.doSomethingToReturnTrue());
    }
}
```

Create the ServiceTestRule

Use @Rule annotation

Start Service under Test

Bind to Service under Test

Espresso

A new approach to UI Testing

An API from Developers for Developers

What would a user do?

Find a view
Perform an action
Check some state

Find, Perform, Check

```
onView(Matcher)
        .perform(ViewAction)
        .check(ViewAssertion);
```

Espresso Cheat Sheet

https://code.google.com/p/android-test-kit/wiki/EspressoV2CheatSheet

Matchers

USER PROPERTIES

```
withId(…)
withText(…)
withTagKey(…)
withTagValue(…)
hasContentDescription(…)
withContentDescription(…)
withHint(…)
withSpinnerText(…)
hasLinks()
hasEllipsizedText()
hasMultilineText()
```

UI PROPERTIES

```
isDisplayed()
isCompletelyDisplayed()
isEnabled()
hasFocus()
isClickable()
isChecked()
isNotChecked()
withEffectiveVisibility(…)
isSelected()
```

COMMON HAMCREST MATCHERS

```
allOf(Matchers)
anyOf(Matchers)
is(...)
not(...)
endsWith(String)
startsWith(String)
```

HIERARCHY

```
withParent(Matcher)
withChild(Matcher)
hasDescendant(Matcher)
isDescendantOfA(Matcher)
hasSibling(Matcher)
isRoot()
```

INPUT

```
supportsInputMethods(…)
hasImeAction(…)
```

CLASS

```
isAssignableFrom(…)
withClassName(…)
```

ROOT MATCHERS

```
isFocusable()
isTouchable()
isDialog()
withDecorView(…)
isPlatformPopup()
```

SEE ALSO

Preference matchers
Cursor matchers

View Actions

CLICK/PRESS

```
click()
doubleClick()
longClick()
pressBack()
pressImeActionButton()
pressKey([int/EspressoKey])
pressMenuKey()
closeSoftKeyboard()
openLink(…)
```

GESTURES

```
scrollTo()
swipeLeft()
swipeRight()
swipeUp()
swipeDown()
```

TEXT

```
clearText()
typeText(String)
typeTextIntoFocusedView(String)
replaceText(String)
```

View Assertions

```
matches(Matcher)
doesNotExist()
selectedDescendantsMatch(…)
```

LAYOUT ASSERTIONS

```
noEllipsizedText(Matcher)
noMultilineButtons()
noOverlaps([Matcher])
```

POSITION ASSERTIONS

```
isLeftOf(Matcher)
isRightOf(Matcher)
isLeftAlignedWith(Matcher)
isRightAlignedWith(Matcher)
isAbove(Matcher)
isBelow(Matcher)
isBottomAlignedWith(Matcher)
isTopAlignedWith(Matcher)
```

app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    androidTestCompile 'com.android.support.test:runner:0.2'
    androidTestCompile 'com.android.support.test:rules:0.2'

    // Espresso dependencies
    androidTestCompile 'com.android.support.test.espresso:espresso-core:2.1'
}
```

Espresso BasicSample

https://github.com/googlesamples/android-testing/tree/master/espresso/BasicSample

```
@RunWith(AndroidJUnit4.class)
@LargeTest
public class ChangeTextBehaviorTest {
    ...
    @Rule
    public ActivityTestRule<MainActivity> mActivityRule = new ActivityTestRule<>(
            MainActivity.class);

    @Test
    public void changeText_sameActivity() {
        onView(withId(R.id.editTextUserInput))
                .perform(typeText(STRING_TO_BE_TYPED), closeSoftKeyboard());
        onView(withId(R.id.changeTextBt)).perform(click());

        onView(withId(R.id.textToBeChanged)).check(matches(withText(STRING_TO_BE_TYPED)));
    }
}
```

Start Activity Under Test

Find EditText view

Type text and then press the button.

Check that text was changed

Espresso APIs

onData() API for Adapter Views
Multi Window Support
Synchronization APIs

Espresso Contrib APIs

DrawerActions
RecyclerViewActions
[Time/Date]PickerActions

app/build.gradle

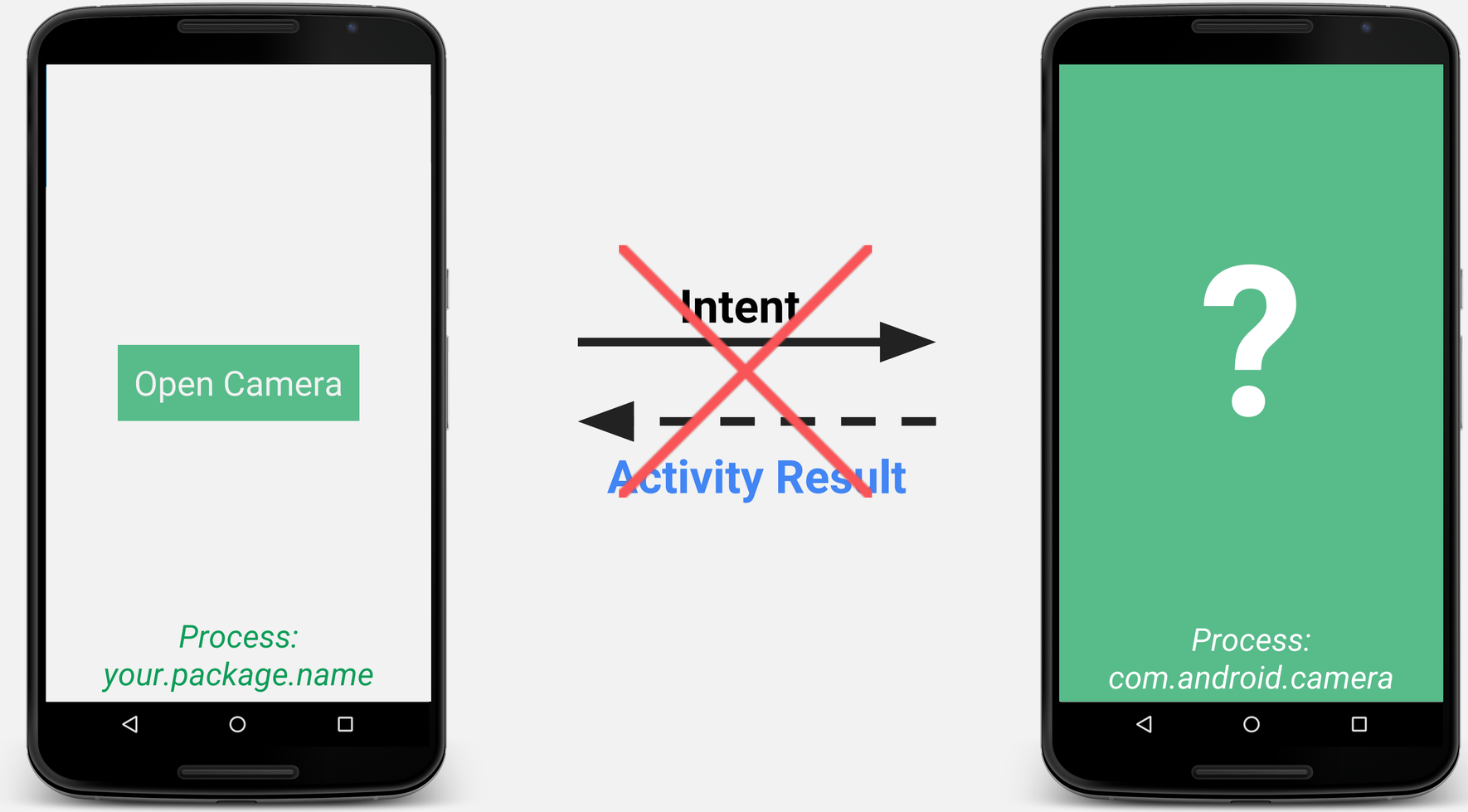
```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    androidTestCompile 'com.android.support.test:runner:0.2'
    androidTestCompile 'com.android.support.test:rules:0.2'

    // Espresso dependencies
    androidTestCompile 'com.android.support.test.espresso:espresso-core:2.1'
    androidTestCompile 'com.android.support.test.espresso:espresso-contrib:2.1'
}
```

Espresso-Intents

Espresso-Intents is like
Mockito but for Intents

Hermetic inter-app testing

Hermetic Testing



Intent Validation

```
intended(IntentMatcher);
```

Intent Stubbing

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

app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    androidTestCompile 'com.android.support.test:runner:0.2'
    androidTestCompile 'com.android.support.test:rules:0.2'

    // Espresso dependencies
    androidTestCompile 'com.android.support.test.espresso:espresso-core:2.1'
    androidTestCompile 'com.android.support.test.espresso:espresso-intents:2.1'
}
```

IntentsBasicSample

https://github.com/googlesamples/android-testing/tree/master/espresso/IntentsBasicSample

```
@RunWith(AndroidJUnit4.class)
@LargeTest
public class DialerActivityTest {
    ...
    @Rule
    public IntentsTestRule<DialerActivity> mRule = new IntentsTestRule<>(DialerActivity.class);

    @Test
    public void typeNumber_ValidInput_InitiatesCall() {
        intending(not(isInternal())).respondWith(new ActivityResult(Activity.RESULT_OK, null));

        onView(withId(R.id.edit_text_caller_number)).perform(typeText(VALID_PHONE_NUMBER),
                closeSoftKeyboard());
        onView(withId(R.id.button_call_number)).perform(click());

        intended(allOf(
                hasAction(Intent.ACTION_CALL),
                hasData(INTENT_DATA_PHONE_NUMBER),
                toPackage(PACKAGE_ANDROID_DIALER)));
    }
}
```

Create IntentsTestRule

Stub all external Intents

Type Number and press Call Button

Verify Intent was sent

UI Automator

UI Automator 2.0

Black box testing
Inter-app behavior testing
Context Access

app/build.gradle

```
apply plugin: 'com.android.application'

android {
    ...
    defaultConfig {
        …
        testInstrumentationRunner ‘android.support.test.runner.AndroidJUnitRunner’
    }

}
dependencies {
    androidTestCompile 'com.android.support.test:runner:0.2'

    // UiAutomator Dependencies
    androidTestCompile 'com.android.support.test.uiautomator:uiautomator-v18:2.0.0'

}
```

UIAutomator BasicSample

https://github.com/googlesamples/android-testing/tree/master/uiautomator/BasicSample

```
@Before
public void startMainActivityFromHomeScreen() {

    mDevice = UiDevice.getInstance(InstrumentationRegistry.getInstrumentation());

    mDevice.pressHome();

    final String launcherPackage = getLauncherPackageName();
    assertThat(launcherPackage, notNullValue());
    mDevice.wait(Until.hasObject(By.pkg(launcherPackage).depth(0)), LAUNCH_TIMEOUT);

    Context context = InstrumentationRegistry.getContext();
    final Intent intent = context.getPackageManager()
            .getLaunchIntentForPackage(BASIC_SAMPLE_PACKAGE);
    intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TASK);
    context.startActivity(intent);

    // Wait for the app to appear
    mDevice.wait(Until.hasObject(By.pkg(BASIC_SAMPLE_PACKAGE).depth(0)), LAUNCH_TIMEOUT);
}
```

Initialize UiDevice

Launch Basic Sample

UIAutomator BasicSample

https://github.com/googlesamples/android-testing/tree/master/uiautomator/BasicSample

```
@Test
public void testChangeText_sameActivity() {

    mDevice.findObject(By.res(BASIC_SAMPLE_PACKAGE, "editTextUserInput"))
            .setText(STRING_TO_BE_TYPED);
    mDevice.findObject(By.res(BASIC_SAMPLE_PACKAGE, "changeTextBt"))
            .click();

    UiObject2 changedText = mDevice
            .wait(Until.findObject(By.res(BASIC_SAMPLE_PACKAGE, "textToBeChanged")),
                    500 /* wait 500ms */);
    assertThat(changedText.getText(), is(equalTo(STRING_TO_BE_TYPED)));
 }
```

Type text and click Button

Verify text displayed in UI

Contribute

Contribute to Android Testing Support Library

`https://source.android.com/source/life-of-a-patch.html`

Initialize your build environment
https://source.android.com/source/initializing.html

Install Repo

https://source.android.com/source/downloading.html

Checkout android-support-test branch

repo init -u https://android.googlesource.com/platform/manifest -g all -b android-support-test

Sync the source

repo sync -j24

Browse the source

cd frameworks/testing

Build and test

// Just build debug build type

./gradlew assembleDebug

// Run tests

./gradlew connectedCheck

Thank you

https://plus.google.com/+AndroidDevelopers/posts/jHXFkebKjEb
https://plus.google.com/+JoseAlcerreca/posts/3aU1J6EDGKd
https://github.com/googlesamples/android-testing

Android Developers
Shared publicly - 20 Dec 2014

Espresso 2.0 is here!

"*A release is never too late, nor is it too early. It arrives precisely when it's meant to.*" That being said, we would like to apologize for

Read more (40 lines)

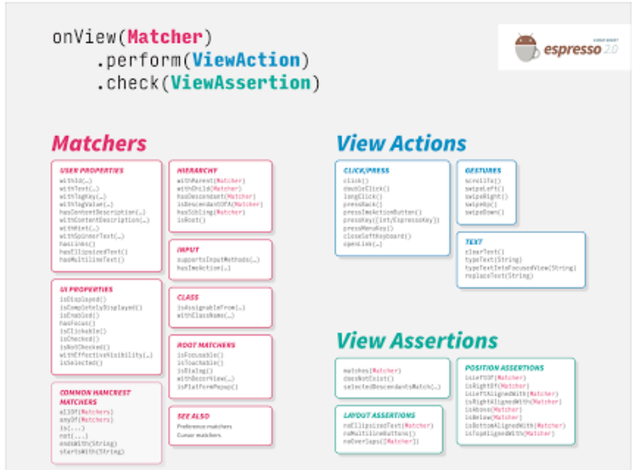
+1489 457

Jose Alcérreca
Shared publicly - 23 Dec 2014
#Android

I don't usually make cheat sheets, but when I do, I ask **+Roman Nurik** to design them.

Here's the **Espresso 2.0 cheat sheet**.

Read more (14 lines)



android-test-kit
code.google.com

+67 114

Unwatch 45 Star 607 Fork 63

Q&A

Android Testing Support Library
developer.android.com/tools/testing-support-library

Ui Testing Training (Espresso & UIAutomator)
developer.android.com/training/testing/ui-testing

Samples
github.com/googlesamples/android-testing

Espresso Cheat Sheet
https://code.google.com/p/android-test-kit/wiki/EspressoV2CheatSheet

#HappyTesting

Google Developers