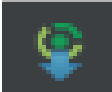# Espresso UI Test (SayHello Decl.)

- A simple Espresso UI Test for SayHelloActivity of SayHello Declarative tutorial.
- Uses AndroidJUnitRunner as a test runner.
- "Espresso":
  - Espresso is a testing framework for Android by Google to make it easy to write reliable user interface tests for a single application.
  - Espresso automatically synchronizes your test actions with the user interface of your application.
  - The framework also ensures that your activity is started before the tests run and that a test wait until all background activities have finished.
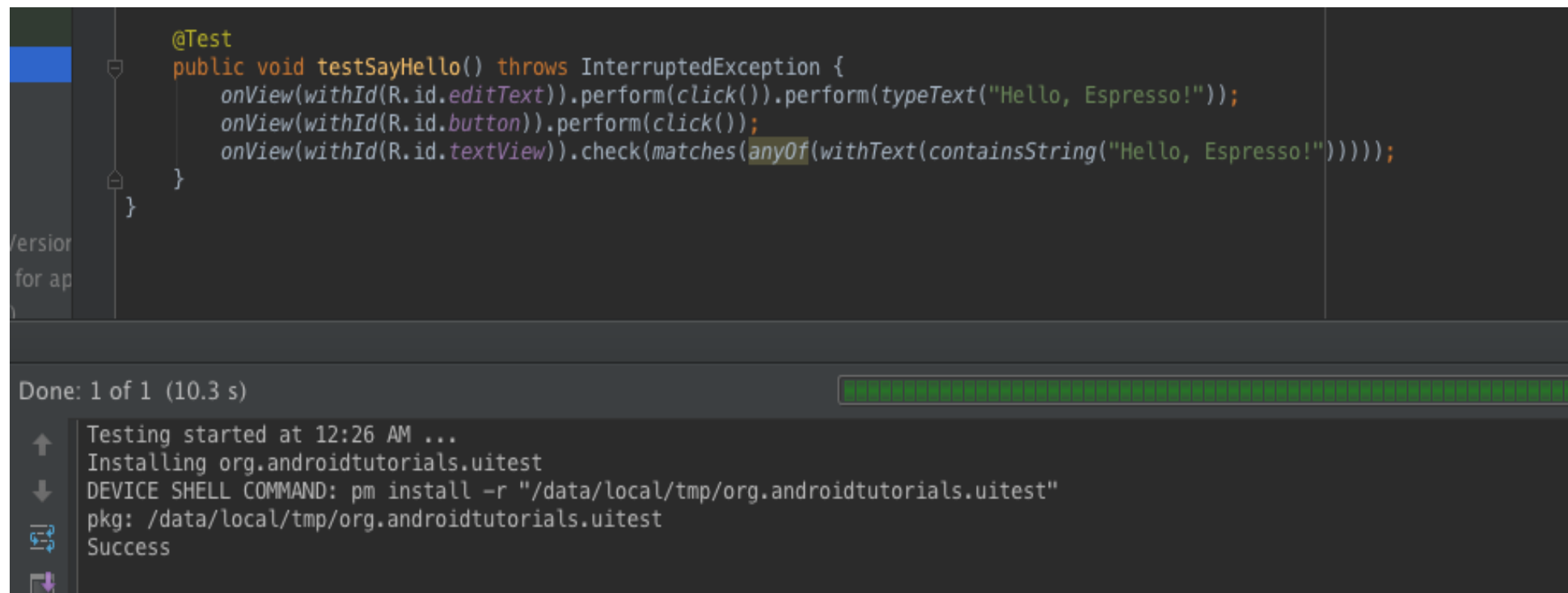
Espresso UI Test (SayHello Decl.)

- Android Espresso tests are written in app/src/androidTest directory. Android Studio automatically generates this directory once you create a project in it.

- 'Main' folder in app/src directory contains activities to be tested.

- Espresso relies on following 2 dependencies in order to work:
  - androidTestCompile 'com.android.support.test.espresso:espresso-core:2.2'
  - androidTestCompile 'com.android.support.test:testing-support-lib:0.1'

- These dependencies are added to the 'build.gradle' file of the app module and not the root directory's 'build.gradle' file.

- Always perform gradle sync once you make a change in build.gradle file

Espresso UI Test (SayHello Decl.)

- Espresso core dependency contains dagger(dependency injection tool) written by people at Square and this may cause an exception of "class not found"
- Exclusion for dagger can be added inside espresso core dependency via: exclude group: 'javax.inject' (refer build.gradle in the source code tab above)
- AndroidJUnitRunner needs a manual configuration in Studio:
  - Select 'Run' tab → Edit Configurations... → + →Select 'app' module and set Instrumentation runner as: android.support.test.runner.AndroidJunitRunner
- Let app's build.gradle also know about this by specifying runner in 'android { defaultConfig { } }' block (again refer app's build.gradle)

Espresso UI Test (SayHello Decl.)


```
@Test
public void testSayHello() throws InterruptedException {
    onView(withId(R.id.editText)).perform(click()).perform(typeText("Hello, Espresso!"));
    onView(withId(R.id.button)).perform(click());
    onView(withId(R.id.textView)).check(matches(anyOf(withText(containsString("Hello, Espresso!")))));
}
}
```

```
Done: 1 of 1 (10.3 s)
Testing started at 12:26 AM ...
Installing org.androidtutorials.uitest
DEVICE SHELL COMMAND: pm install -r "/data/local/tmp/org.androidtutorials.uitest"
pkg: /data/local/tmp/org.androidtutorials.uitest
Success
```

Screenshot: Android Studio showing the result of *SayHelloActivityTest*
*Green bar indicates that the test has passed.

References:

- [UI Testing Espresso](#)
- [Espresso Sheet](#)

Exercise:

Sample Directory Structure:

- Input a string 'Android' instead of 'Hello, Espresso!' and see if the test fails. (Analyze log cat and see how Espresso throws errors with view hierarchy display)
- Create and launch another activity via intent on button click from SayHelloActivity. Now, verify with Espresso that you have navigated to the newly created activity by validating text (say "Hello") of textView.