Creating an Android Robotium Bootstrap Robot

Appurify

Setup to creating your bootstrapped script:

- Have an android phone (Api 12+)
- Have your phone plugged in.
- Have the .apk you want to bootstrap
- Be working on a Mac (Linux or Windows may work but some steps may be slightly different)

First download the android SDK

- The SDK can be found here for download:

http://developer.android.com/sdk/index.html

- Unzip the file and you will see two folders
  - eclipse (This is the IDE we will work with)
  - SDK (The android SDK)

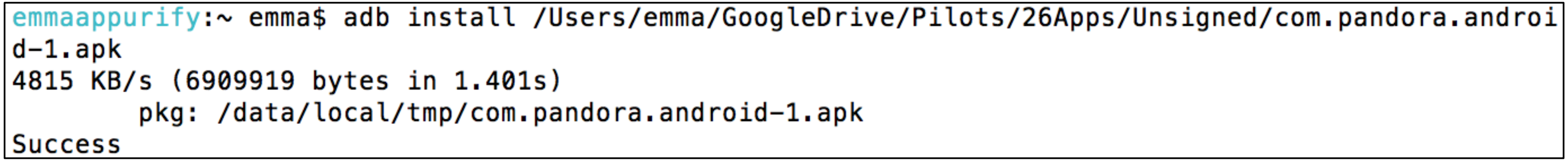
(additional info can be found here: http://robotium.googlecode.com/files/SetupAndroidEnvironment-V2_0.pdf)

Start Eclipse

- Double click on eclipse application in the eclipse folder
- Eclipse will prompt you to choose a workspace.
- You can name your workspace after the app you are working on so that all tests for that app are kept in one place.
  - You can switch and create other workspaces at any time.

Installing the APK

- The command to install is:
    - /path/to/android/SDK/**platform-tools/adb install yourapp.apk**
        - (path to SDK is the path to the sdk folder you downloaded)
- You should now see your apk on the phone.

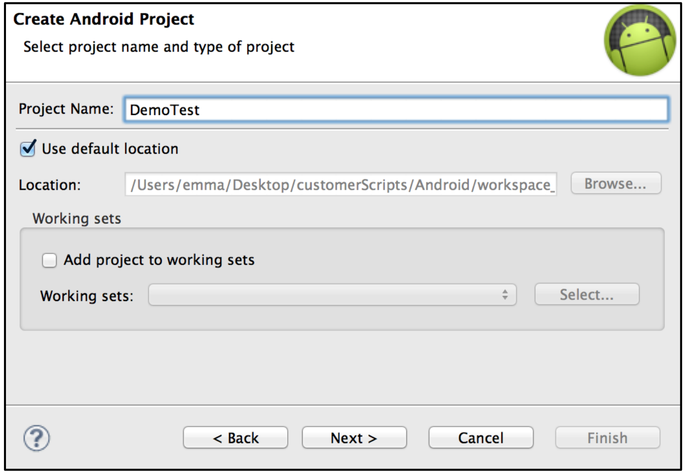


```
emmaappurify:~ emma$ adb install /Users/emma/GoogleDrive/Pilots/26Apps/Unsigned/com.pandora.androi
d-1.apk
4815 KB/s (6909919 bytes in 1.401s)
        pkg: /data/local/tmp/com.pandora.android-1.apk
Success
```

Setting up your eclipse Environment

We will now set up your eclipse environment to create your bootstrap Robotium code.

- Go to File -> New -> Other-> Android Test Project. Then click Next
  - pictured to the right ---->
- Enter project name. Then click Next.
  - For the Demo we call it “DemoTest”
  - pictured below.



Create Android Project

Select project name and type of project

Project Name: DemoTest

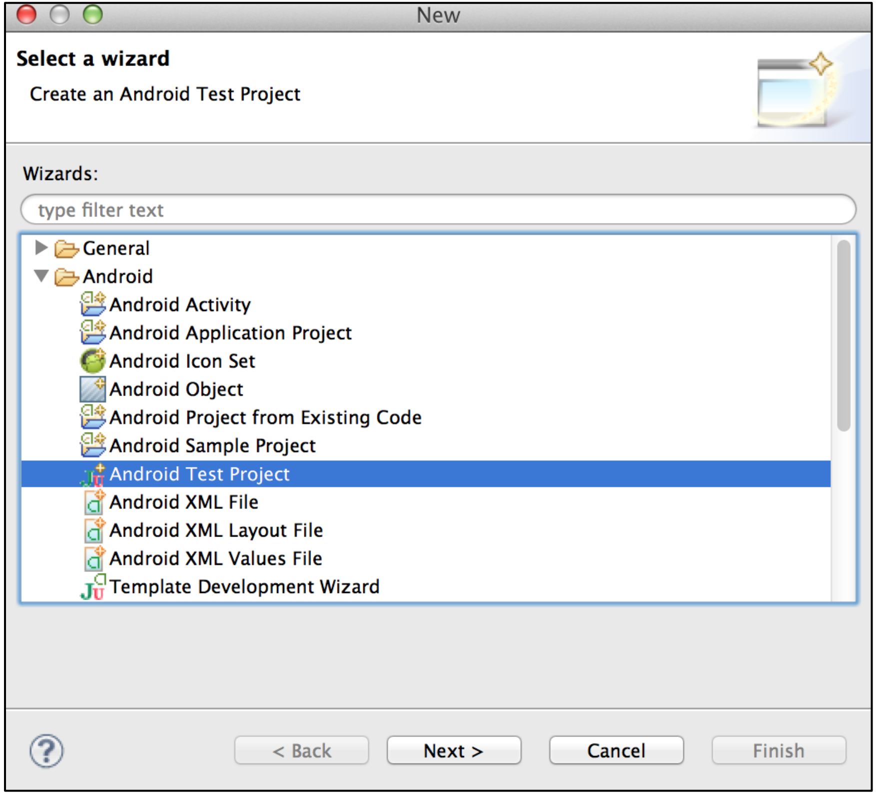
☑ Use default location

Location: /Users/emma/Desktop/customerScripts/Android/workspace_

Working sets

□ Add project to working sets

Working sets:



New

Select a wizard

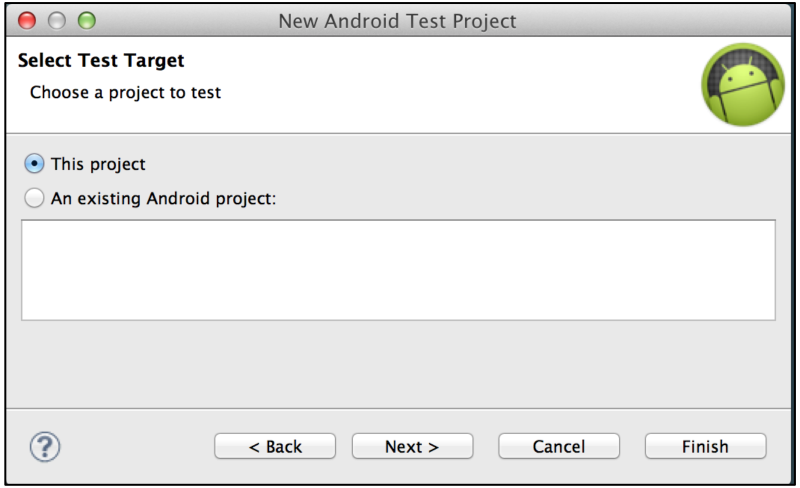
Create an Android Test Project

Wizards:

type filter text

Setting up your eclipse Environment

- Click "This project". Then click Finish.
- Note: You may encounter an Eclipse bug after clicking “Finish”, such as the one pictured below. If this occurs you will have to exit eclipse and repeat steps from File -> New -> Other-> Android Test Project again. I should not happen a second time.
    - Ignore this if it does not occur.



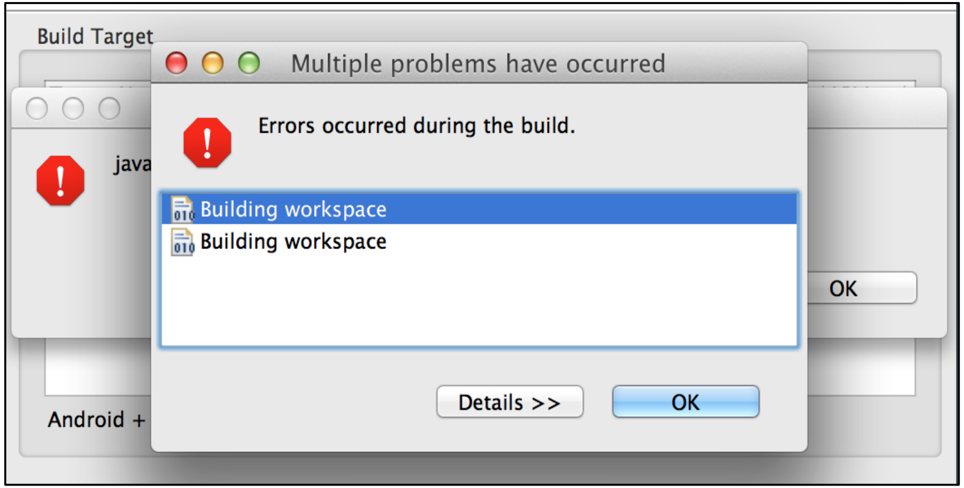
New Android Test Project

Select Test Target

Choose a project to test

● This project

○ An existing Android project:



Multiple problems have occurred

Errors occurred during the build.

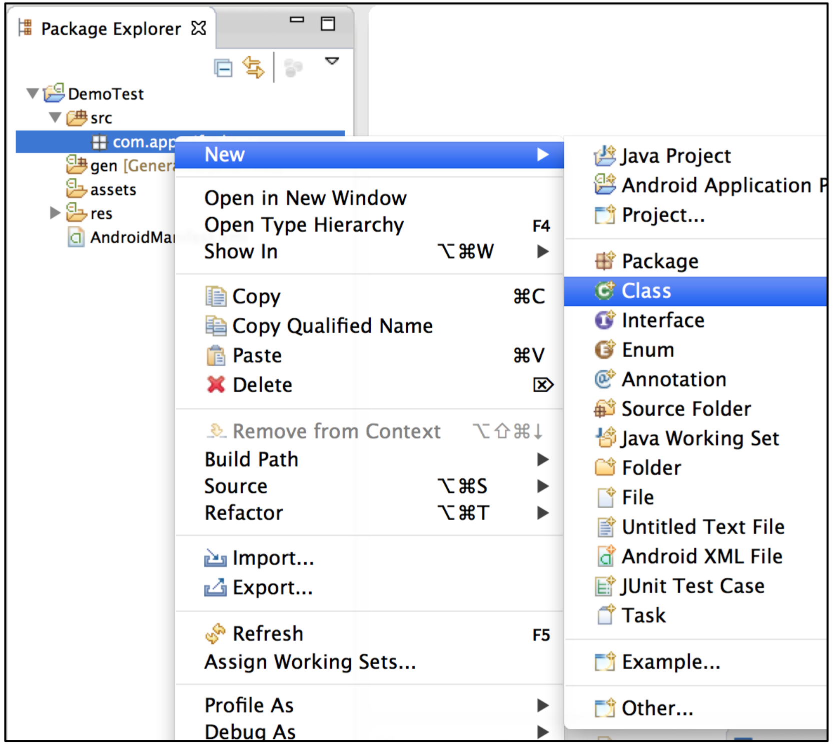
Building workspace

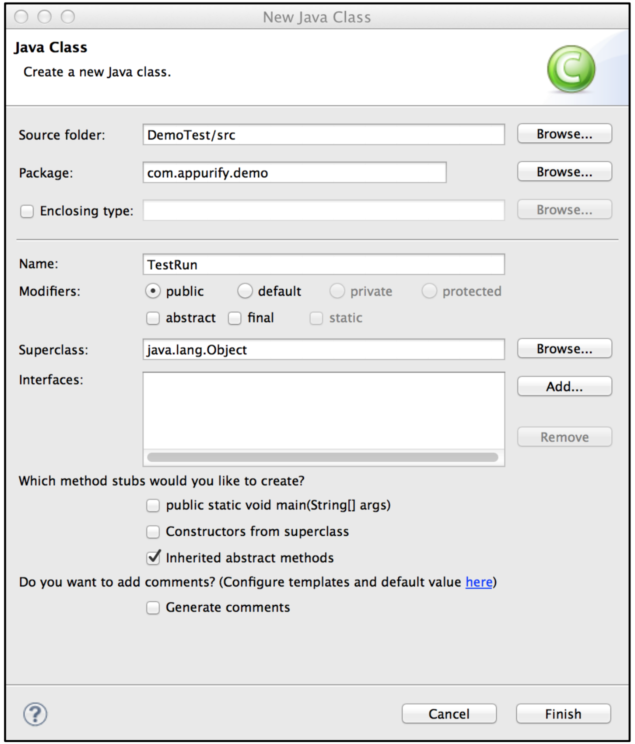
Building workspace

Setting up your eclipse Environment

You should now see a set up like the one pictured to the left in the Package Explorer.

- Next right click the package in the src folder and click new class (This is where the test code is written).
  - For this Demo I will call the class “TestRun”





New Java Class

Java Class

Create a new Java class.

Source folder: DemoTest/src [Browse...]

Package: com.appurify.demo [Browse...]

□ Enclosing type: [Browse...]

Name: TestRun

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object [Browse...]

Interfaces: [Add...] [Remove]

Which method stubs would you like to create?

□ public static void main(String[] args)

□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

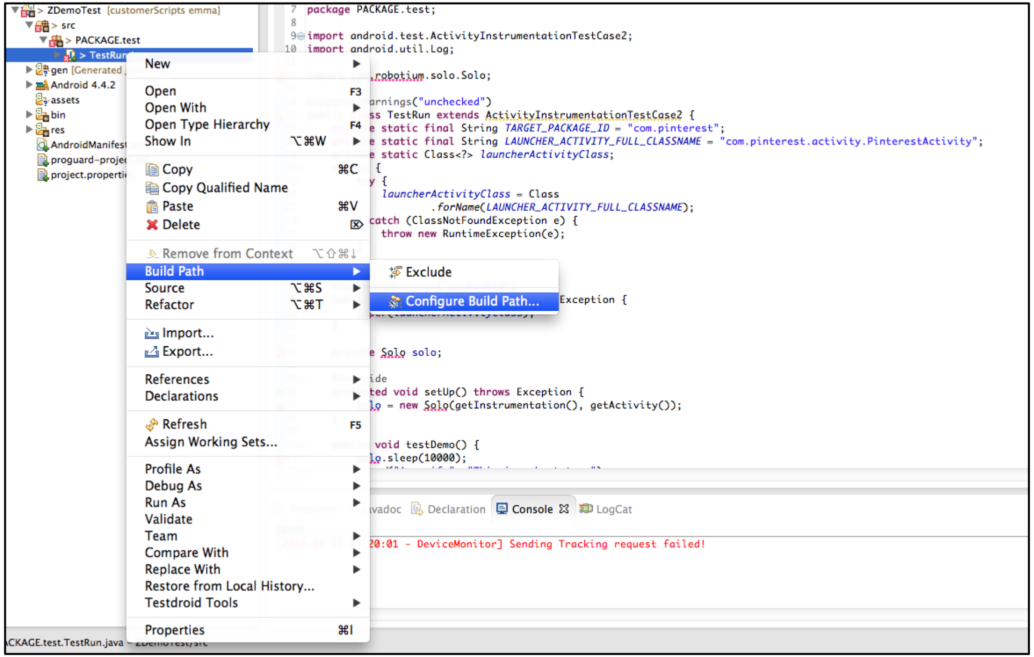
[Cancel] [Finish]

Skeleton Robotium Script

- This is a skeleton Robotium test written for the Pinterest Application pulled from the Google Play Store.
    - https://github.com/eemar7/AndroidAutomationTools/blob/master/Robotium/TestRun.java
    - As is, it will sleep for 10 seconds, log a message and sleep for 10 more seconds.
- This skeleton should be copy and pasted into your class file. It should replace everything except your top line which states you package name (your package name must stay at the top).
- This will cause a few errors on the page. We will fix those by adding the Robotium Library to the build path next.

Add Robotium-solo.jar to your test.

- Download the Robotium-solo.jar from here (where it says “Robotium jar”):
    - https://code.google.com/p/robotium/wiki/Getting_Started
- Next, right click on either your project name/package name/or class name and click "Build Path"-> "configure build path"
    - pictured below.

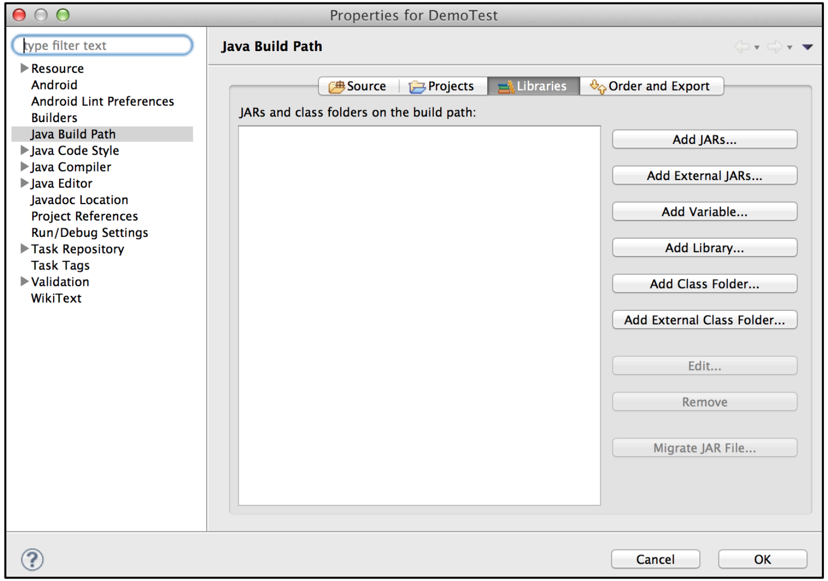


```
package PACKAGE.test;

import android.test.ActivityInstrumentationTestCase2;
import android.util.Log;

.robotium.solo.Solo;

arnings("unchecked")
ss TestRun extends ActivityInstrumentationTestCase2 {
e static final String TARGET_PACKAGE_ID = "com.pinterest";
e static final String LAUNCHER_ACTIVITY_FULL_CLASSNAME = "com.pinterest.activity.PinterestActivity";
e static Class<?> launcherActivityClass;
 {
y {
 launcherActivityClass = Class
         .forName(LAUNCHER_ACTIVITY_FULL_CLASSNAME);
catch (ClassNotFoundException e) {
 throw new RuntimeException(e);

Exception {

e Solo solo;

ide
ted void setUp() throws Exception {
lo = new Solo(getInstrumentation(), getActivity());

 void testDemo() {
lo.sleep(10000);
```

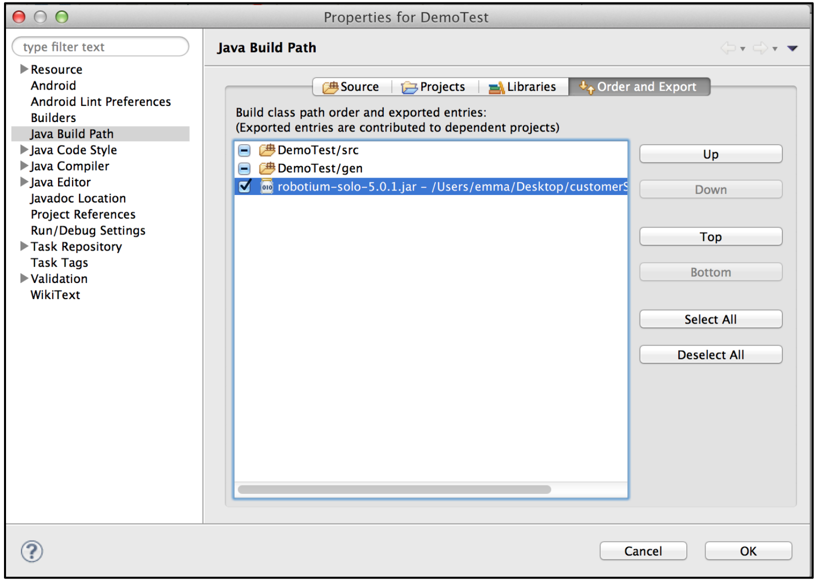
```
20:01 - DeviceMonitor] Sending Tracking request failed!
```

Add Robotium-solo.jar to your test.

- Next go to "Libraries" -> "Add external jar", and find the Robotium-solo.jar file that you downloaded.
- Then go to "Order and Export" and tick the box next to robotium-solo jar.
- Click ok.
- Once you click ok, the robotium-solo.jar should appear under **Referenced Libraries** in your project folder in **Package Explorer**



JARs and class folders on the build path:

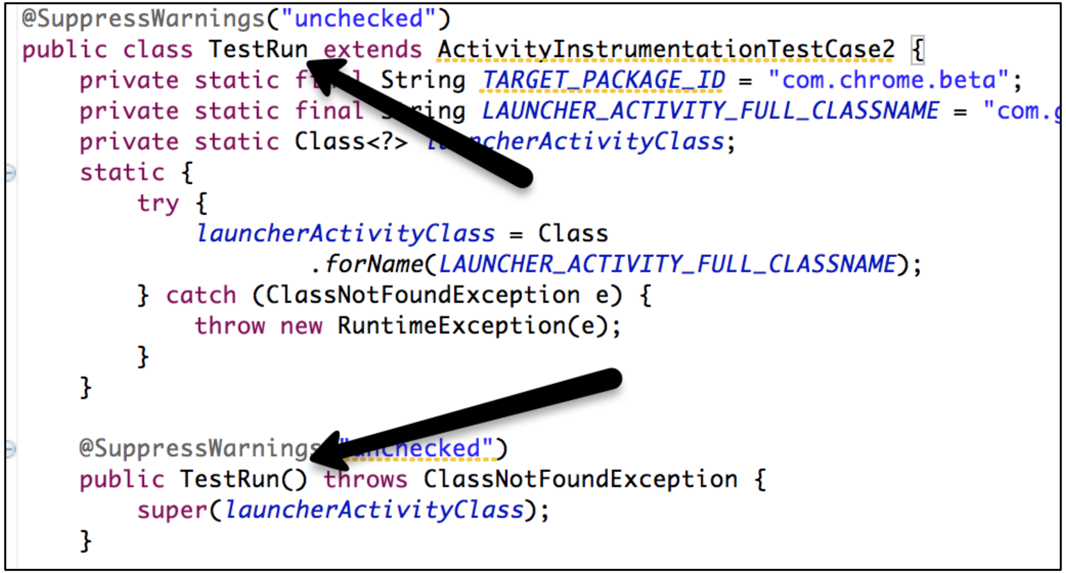


Build class path order and exported entries:
(Exported entries are contributed to dependent projects)

- DemoTest/src
- DemoTest/gen
- ☑ robotium-solo-5.0.1.jar – /Users/emma/Desktop/customerS

Fixing the errors in your class

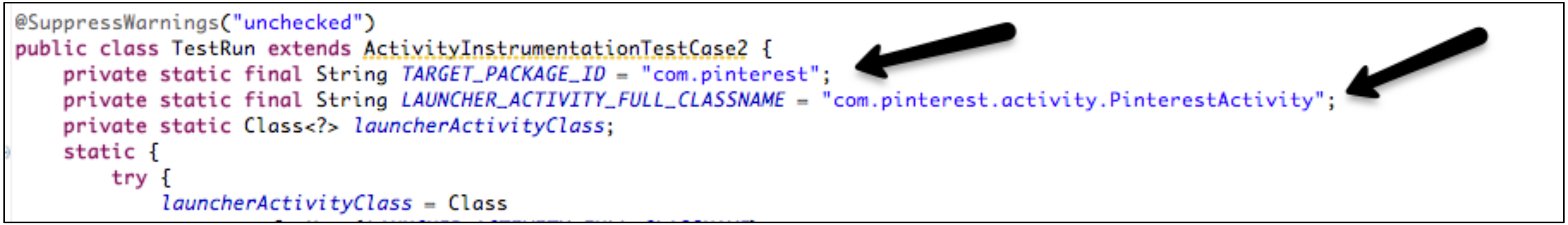
- Next, you may need to change the name of the skeleton class “TestRun” back to the name you chose for your class.
  - For our demo it was “TestRun” so we don’t need to make the change
- If you have anything else that is red, hover over it and eclipse will suggest a fix you can click on.
- All your errors should now be fixed.



```
@SuppressWarnings("unchecked")
public class TestRun extends ActivityInstrumentationTestCase2 {
    private static final String TARGET_PACKAGE_ID = "com.chrome.beta";
    private static final String LAUNCHER_ACTIVITY_FULL_CLASSNAME = "com.g
    private static Class<?> launcherActivityClass;
    static {
        try {
            launcherActivityClass = Class
                    .forName(LAUNCHER_ACTIVITY_FULL_CLASSNAME);
        } catch (ClassNotFoundException e) {
            throw new RuntimeException(e);
        }
    }

    @SuppressWarnings("unchecked")
    public TestRun() throws ClassNotFoundException {
        super(launcherActivityClass);
    }
```

Altering the code for your apk

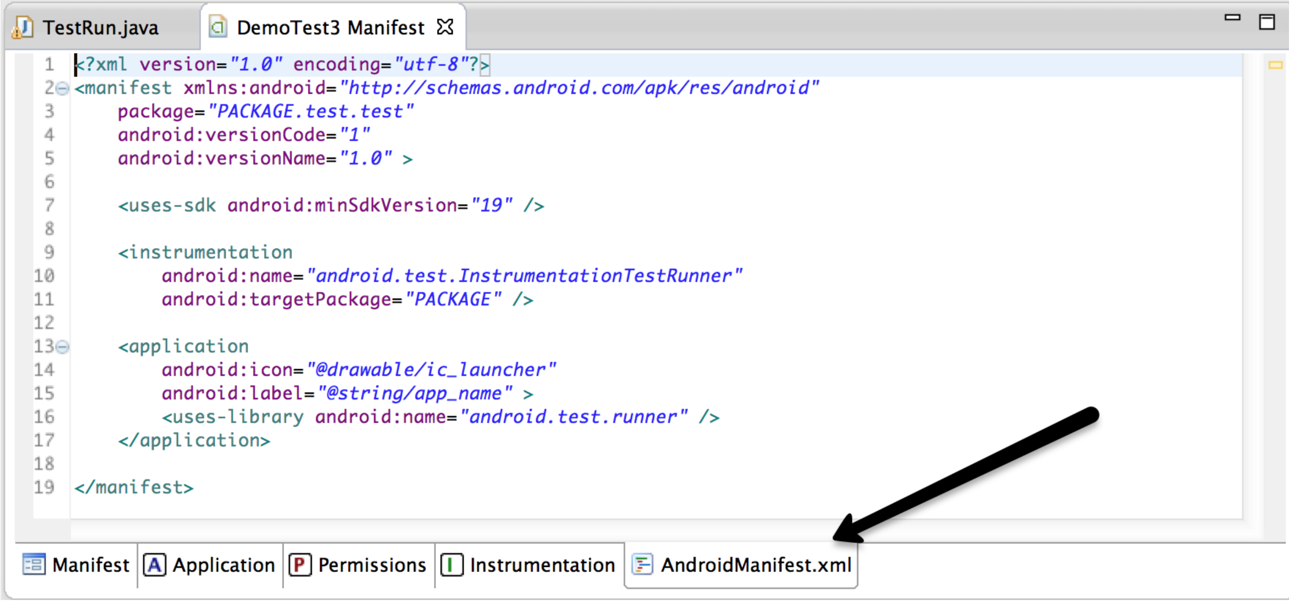
- Two things in the code must be changed
  - 1)TARGET_PACKAGE_ID = “replace with your package name”
  - 2)LAUNCHER_ACTIVITY_FULL_CLASSNAME = “replace with your main activity name”



```
@SuppressWarnings("unchecked")
public class TestRun extends ActivityInstrumentationTestCase2 {
    private static final String TARGET_PACKAGE_ID = "com.pinterest";
    private static final String LAUNCHER_ACTIVITY_FULL_CLASSNAME = "com.pinterest.activity.PinterestActivity";
    private static Class<?> launcherActivityClass;
    static {
        try {
            launcherActivityClass = Class
```

Altering the Android manifest

- Click on AndroidManifest.xml in your Package Explorer.
- Next, click on AndroidManifest.xml in the tabs along the bottom.



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="PACKAGE.test.test"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk android:minSdkVersion="19" />

    <instrumentation
        android:name="android.test.InstrumentationTestRunner"
        android:targetPackage="PACKAGE" />

    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name" >
        <uses-library android:name="android.test.runner" />
    </application>

</manifest>
```

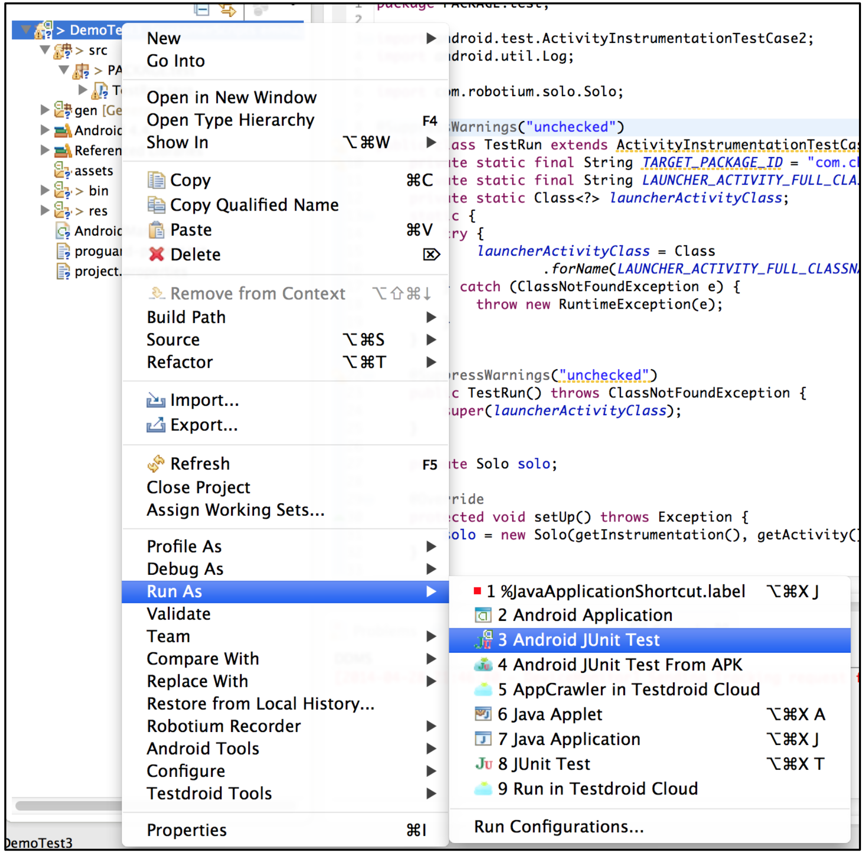
Altering the Android manifest

- We will be changing 2 things:
  - <uses-sdk android:minSdkVersion="**19**" />
  - <instrumentation android:name="android.test.InstrumentationTestRunner" android:targetPackage="**PACKAGE**" />
- Change **SdkVersion** to something lower like 12 (therefore it can work on a larger set of devices.
- Change **PACKAGE** to your .apk package name.

Code Explained

- All Robotium scripts need 3 functions in the code
  - 1) **setUp()** -- Sets up the test by creating a new solo object to bind to.
  - 2) A function beginning with “**test**”, in the Demo our function is called testDemo. -- This is where the main code is run.
  - 3) **tearDown()** -- ends the test and finishes the activity.
- Right now in our “testDemo” function we are just sleeping for 10 seconds, logging a message and then sleeping for 10 more seconds.
  - The Log.d(“Appurify”, “message”); line will print to LogCat which can be found by going to Window -> Show View -> Other -> Android -> LogCat.
    - You can then filter LogCat to show only messages with the “Appurify” tag.

Running the test



```
package PACKAGE.test;
idroid.test.ActivityInstrumentationTestCase2;
idroid.util.Log;

m.robotium.solo.Solo;

Warnings("unchecked")
ass TestRun extends ActivityInstrumentationTestCa
te static final String TARGET_PACKAGE_ID = "com.c
te static final String LAUNCHER_ACTIVITY_FULL_CLA
te static Class<?> launcherActivityClass;
c {
ry {
    launcherActivityClass = Class
            .forName(LAUNCHER_ACTIVITY_FULL_CLASSN
catch (ClassNotFoundException e) {
    throw new RuntimeException(e);


ressWarnings("unchecked")
c TestRun() throws ClassNotFoundException {
uper(launcherActivityClass);


te Solo solo;

ride
cted void setUp() throws Exception {
olo = new Solo(getInstrumentation(), getActivity(
```

- So far we have
  - Installed the .apk on the device
  - Created a class that will sleep for 10 seconds, log a message and sleep for 10 seconds
    - That class points to the .apk
  - Modified the Android Manifest to point to the .apk
- Now we are ready to run the test.
- You can now run your simple test by right clicking on the project folder/package name/or class name and clicking Run As -> Android JUnit Test

Running the test

If an emulator pops up (ignore otherwise):

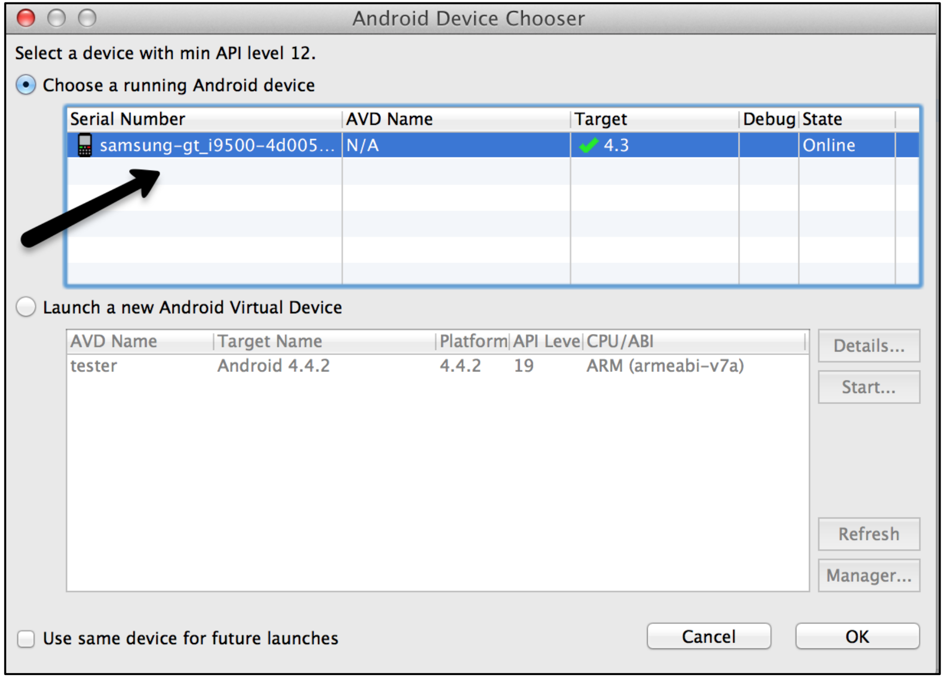
- Exit the emulator
- Unplug your device and plug back in.

If your console says adb not found (ignore otherwise):

- Exit out of eclipse
- In your terminal type
  - /path/to/sdk/platform-tools/adb kill-server
  - /path/to/sdk/platform-tools/adb start-server

If None of these occur you should be prompted to pick your device (as pictured to the right)

- Once you click "ok" the test will start on your device!



Android Device Chooser

Select a device with min API level 12.

● Choose a running Android device

Serial Number	AVD Name	Target	Debug	State
samsung-gt_i9500-4d005...	N/A	✓ 4.3		Online

○ Launch a new Android Virtual Device

AVD Name	Target Name	Platform	API Leve	CPU/ABI
tester	Android 4.4.2	4.4.2	19	ARM (armeabi-v7a)

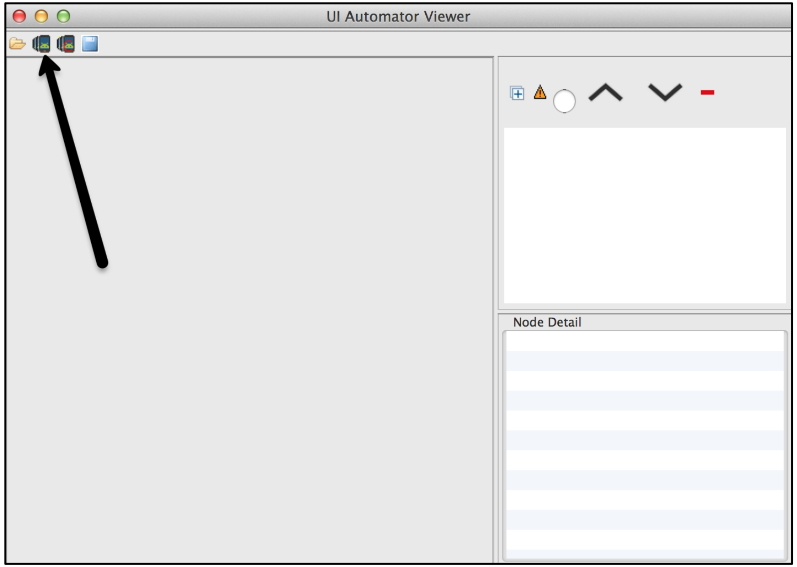
□ Use same device for future launches

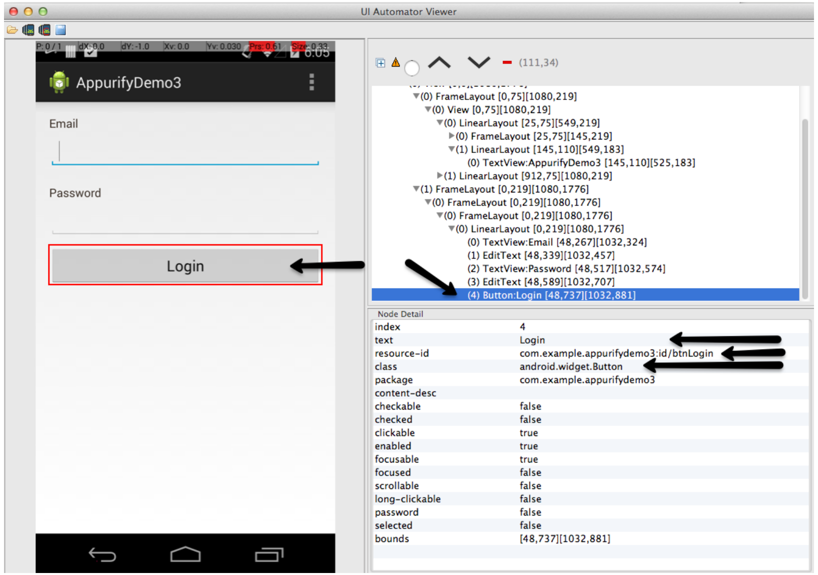
Extending Robotium script

- To create a bootstrap specific to your apk and preferences you will have to extend the script, in the “testDemo” portion of the class, to perform the clicks you desire by looking at the **Robotium Solo API**
- The Robotium Solo API can be found here:
  - http://robotium.googlecode.com/svn/doc/index-all.html
  - In our demo we used **solo.sleep(10000);** another command might be **solo.clickOnText (“hello”);** if you wanted to click on an element that contains text of “hello”
- There are also some useful tutorials found here:
  - http://www.vogella.com/tutorials/Robotium/article.html
- Next we will talk about using UIAutomatorViewer as a tool for creating scripts and show some specific examples of how to click on an Element.

UIAutomatorViewer

- To open UIAutomatorViewer use the command:
  - /path/to/android-sdk/**tools/uiautomatorviewer**
- This allows you to quickly observe the Element tree of the page you are on and figure out what to click and how to reference it in your Robotium code.
- Click the top left device button to take a snapshot and view the tree.





```
▼(0) FrameLayout [0,75][1080,219]
  ▼(0) View [0,75][1080,219]
    ▼(0) LinearLayout [25,75][549,219]
      ▶(0) FrameLayout [25,75][145,219]
      ▼(1) LinearLayout [145,110][549,183]
          (0) TextView:AppurifyDemo3 [145,110][525,183]
    ▶(1) LinearLayout [912,75][1080,219]
▼(1) FrameLayout [0,219][1080,1776]
  ▼(0) FrameLayout [0,219][1080,1776]
    ▼(0) FrameLayout [0,219][1080,1776]
      ▼(0) LinearLayout [0,219][1080,1776]
          (0) TextView:Email [48,267][1032,324]
          (1) EditText [48,339][1032,457]
          (2) TextView:Password [48,517][1032,574]
          (3) EditText [48,589][1032,707]
          (4) Button:Login [48,737][1032,881]
```

index	4
text	Login
resource-id	com.example.appurifydemo3:id/btnLogin
class	android.widget.Button
package	com.example.appurifydemo3
content-desc	
checkable	false
checked	false
clickable	true
enabled	true
focusable	true
focused	false
scrollable	false
long-clickable	false
password	false
selected	false
bounds	[48,737][1032,881]

Using uiautomatorviewer

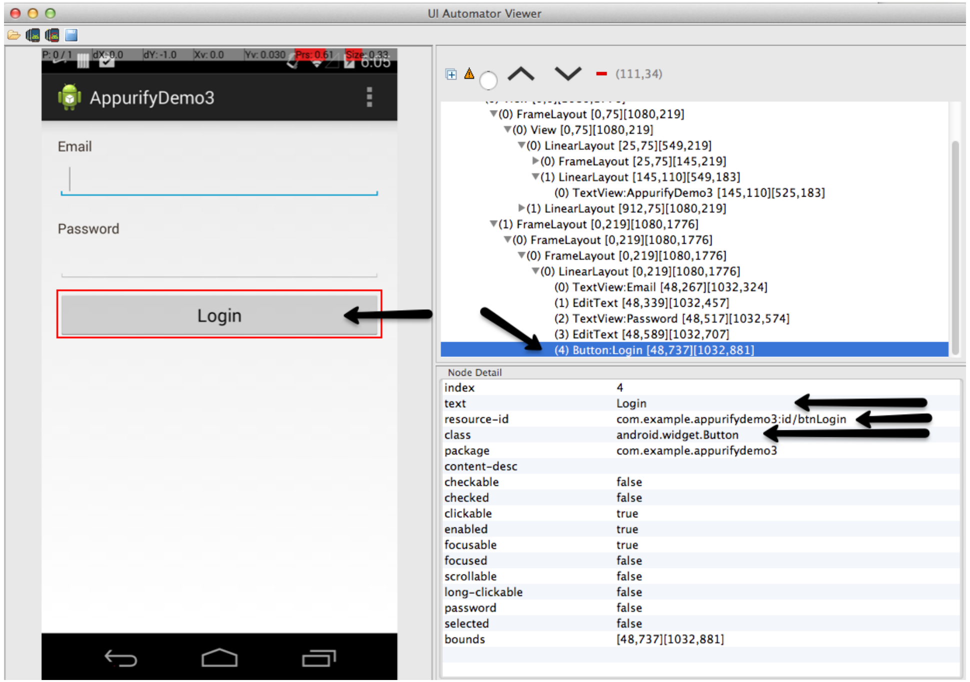
- To the left is an example App being viewed on UIAutomatorviewer.
- The Login Element is clicked on and shown on the Element tree.
- Below the Element tree you can see it gives important information about the Element you clicked on. (resource-id, text and class name)
  - ○ All three can be used to identify the Element when testing using Robotium.
- For instance if you want to click on this element you can use:

-solo.clickOnText(“Login”);

-solo.clickOnButton(0);

-solo.clickOnView(solo."com.example.appurifydemo3:id/btnLogin");

- You can extend the skeleton script by adding clicks like any of these to the testDemo portion of your script.



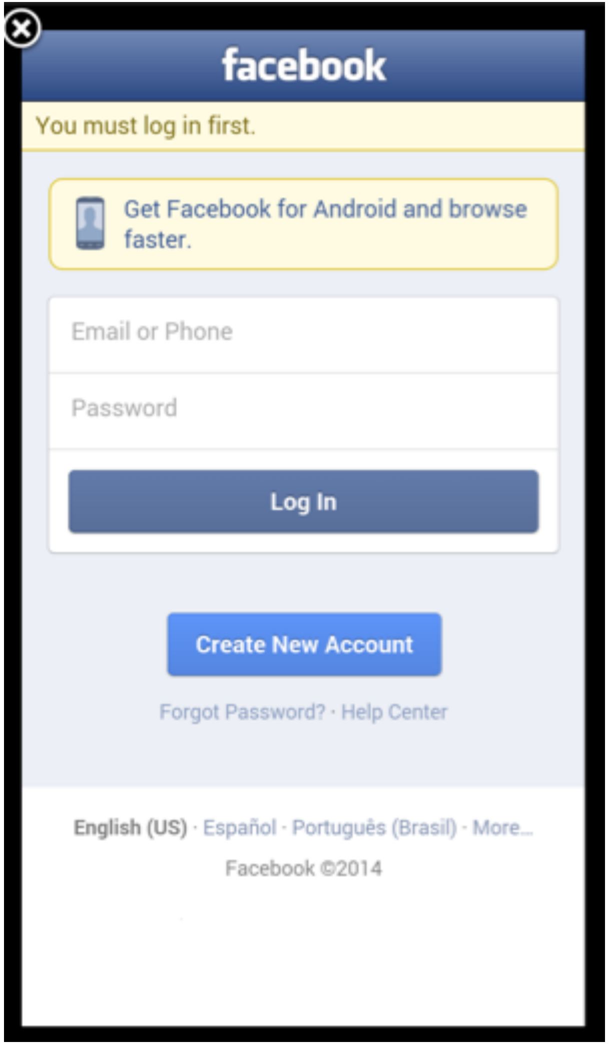
```
▼(0) FrameLayout [0,75][1080,219]
  ▼(0) View [0,75][1080,219]
    ▼(0) LinearLayout [25,75][549,219]
      ▶(0) FrameLayout [25,75][145,219]
      ▼(1) LinearLayout [145,110][549,183]
          (0) TextView:AppurifyDemo3 [145,110][525,183]
    ▶(1) LinearLayout [912,75][1080,219]
▼(1) FrameLayout [0,219][1080,1776]
  ▼(0) FrameLayout [0,219][1080,1776]
    ▼(0) FrameLayout [0,219][1080,1776]
      ▼(0) LinearLayout [0,219][1080,1776]
          (0) TextView:Email [48,267][1032,324]
          (1) EditText [48,339][1032,457]
          (2) TextView:Password [48,517][1032,574]
          (3) EditText [48,589][1032,707]
          (4) Button:Login [48,737][1032,881]
```

index	4
text	Login
resource-id	com.example.appurifydemo3:id/btnLogin
class	android.widget.Button
package	com.example.appurifydemo3
content-desc	
checkable	false
checked	false
clickable	true
enabled	true
focusable	true
focused	false
scrollable	false
long-clickable	false
password	false
selected	false
bounds	[48,737][1032,881]

Example login

- If you use Facebook Connect Login, the Element Tree will be HTML based and the same as in other apps.
- Here is how you would log in through Facebook Connect after clicking the
- Facebook Connect button

```
// Type email
assertTrue("Email field not visible", solo.waitForWebElement(By.name("email")));
solo.enterTextInWebElement(By.name("email"), "testing2@appurify.com");

// Type password
solo.enterTextInWebElement(By.name("pass"), "appurify123");

// Click login
solo.clickOnWebElement(By.name("login"));
```



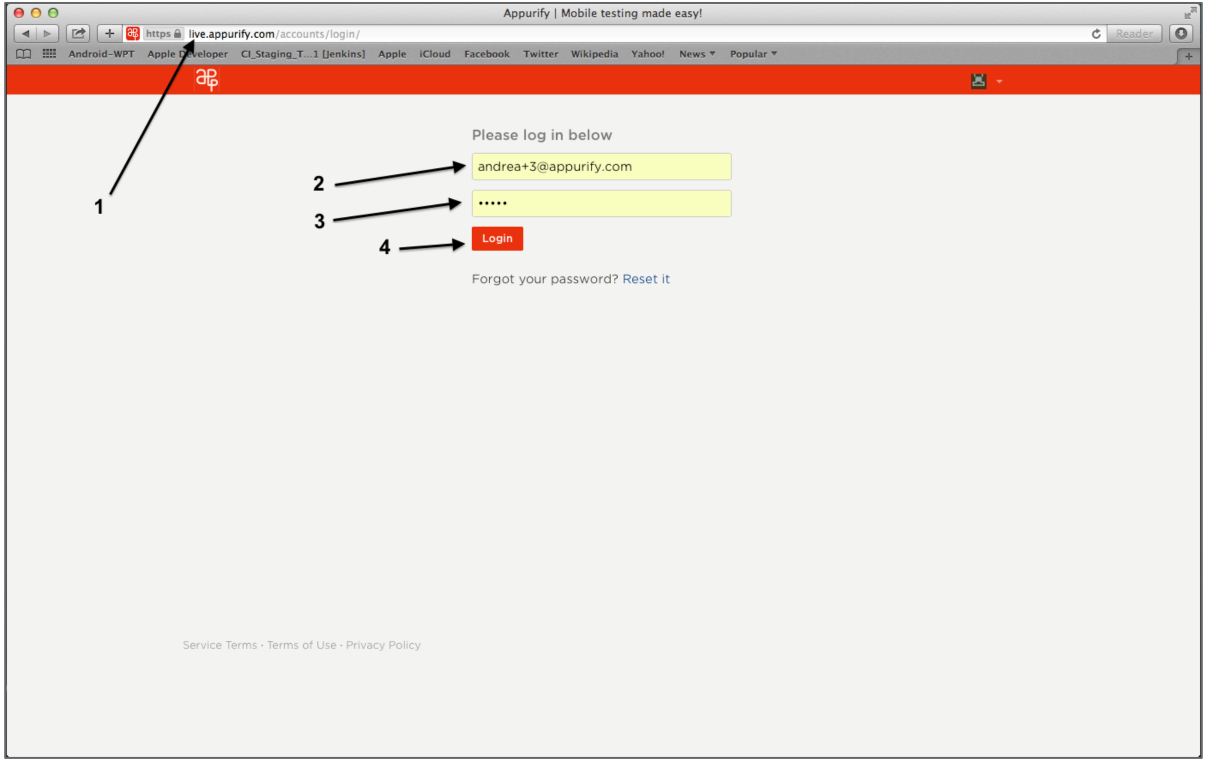
You must log in first.

Get Facebook for Android and browse faster.

facebook
Email or Phone
Password
Log In
Create New Account
Forgot Password? · Help Center
English (US) · Español · Português (Brasil) · More...
Facebook ©2014

Uploading Bootstrap to Platform

- Once you have your bootstrap running the way you want, you can now upload it to the platform and append the Appurify Robot.


Appurify | Mobile testing made easy!
https live.appurify.com/accounts/login/
Reader
Android-WPT
Apple Developer
CI_Staging_T...1 [Jenkins]
Apple
iCloud
Facebook
Twitter
Wikipedia
Yahoo!
News
Popular
1
2
3
4
Login
Service Terms · Terms of Use · Privacy Policy

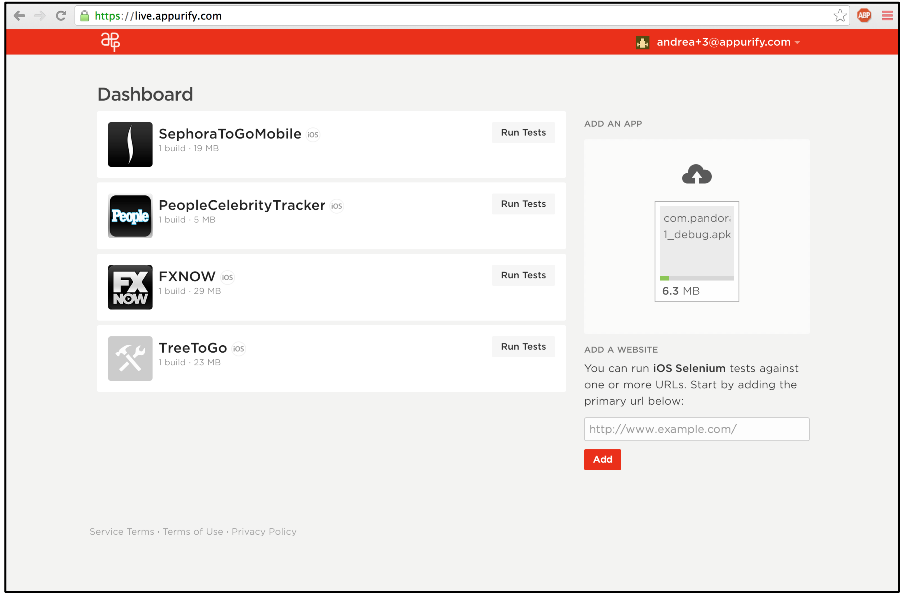
Please log in below

andrea+3@appurify.com

•••••

Forgot your password? Reset it

- Open browser and go to live.appurify.com
- Login using username and password (Demo account - andrea+3@appurify.com / test3



https://live.appurify.com

andrea+3@appurify.com

Dashboard

SephoraToGoMobile ios
1 build · 19 MB
Run Tests

PeopleCelebrityTracker ios
1 build · 5 MB
Run Tests

FXNOW ios
1 build · 29 MB
Run Tests

TreeToGo ios
1 build · 23 MB
Run Tests

ADD AN APP

com.pandora.android-1_debug.apk

6.3 MB

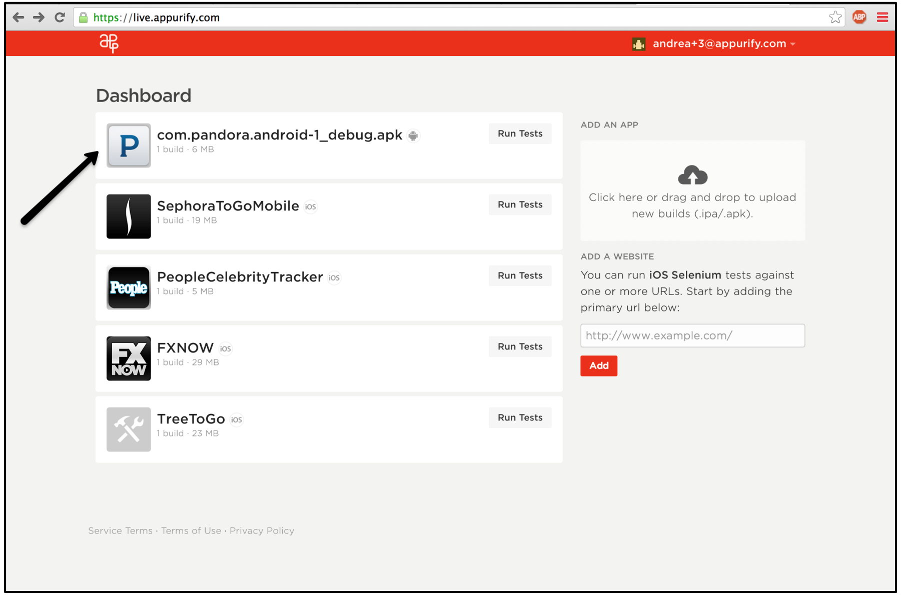
ADD A WEBSITE

You can run **iOS Selenium** tests against one or more URLs. Start by adding the primary url below:

http://www.example.com/

Add

Service Terms · Terms of Use · Privacy Policy



https://live.appurify.com

andrea+3@appurify.com

Dashboard

com.pandora.android-1_debug.apk
1 build · 6 MB
Run Tests

SephoraToGoMobile ios
1 build · 19 MB
Run Tests

PeopleCelebrityTracker ios
1 build · 5 MB
Run Tests

FXNOW ios
1 build · 29 MB
Run Tests

TreeToGo ios
1 build · 23 MB
Run Tests

ADD AN APP

Click here or drag and drop to upload new builds (.ipa/.apk).

ADD A WEBSITE

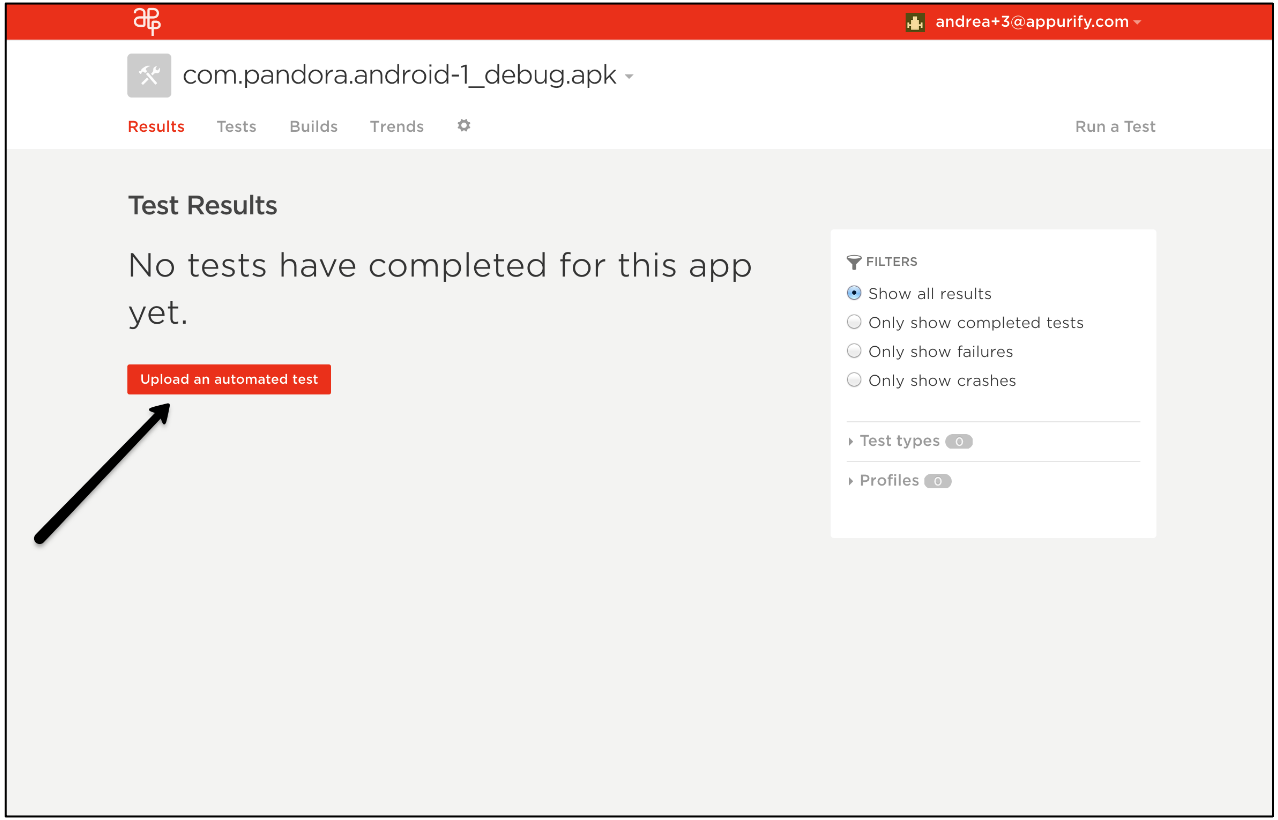
You can run **iOS Selenium** tests against one or more URLs. Start by adding the primary url below:

http://www.example.com/

Add

Service Terms · Terms of Use · Privacy Policy

- Upload Signed .apk to Dashboard (click and upload or drag and drop app into box).
- New App will appear on Dashboard.



com.pandora.android-1_debug.apk

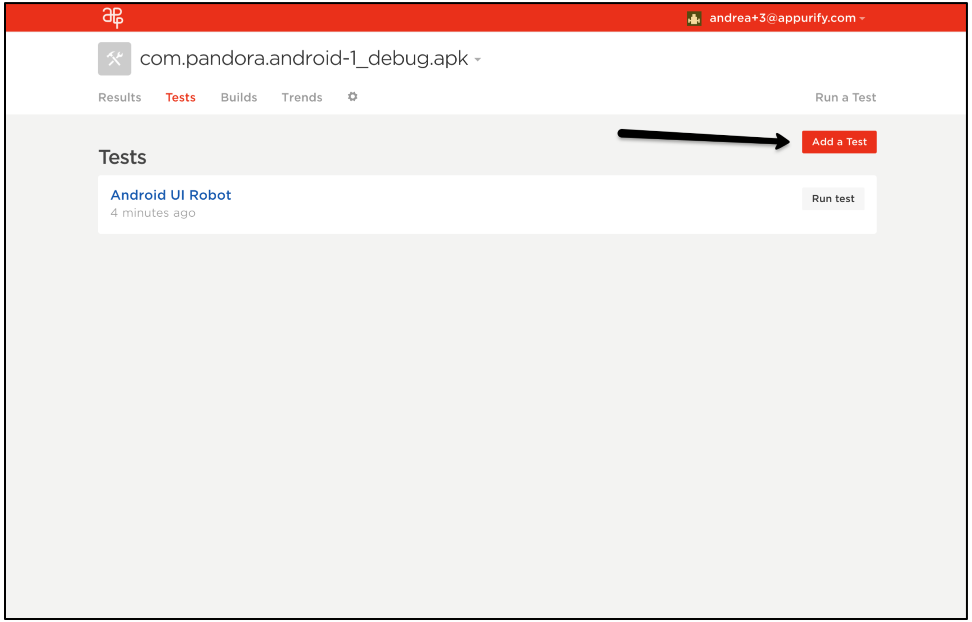
Test Results

No tests have completed for this app yet.

FILTERS

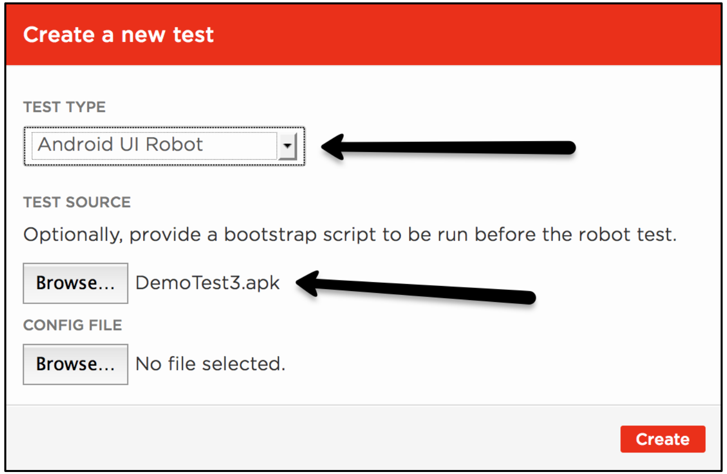
- ● Show all results
- ○ Only show completed tests
- ○ Only show failures
- ○ Only show crashes

- After uploading, it will automatically navigate to the Application Overview page.
- Click on “Upload an Automated Test” to upload the test you just created.



Tests

Android UI Robot
4 minutes ago



Create a new test

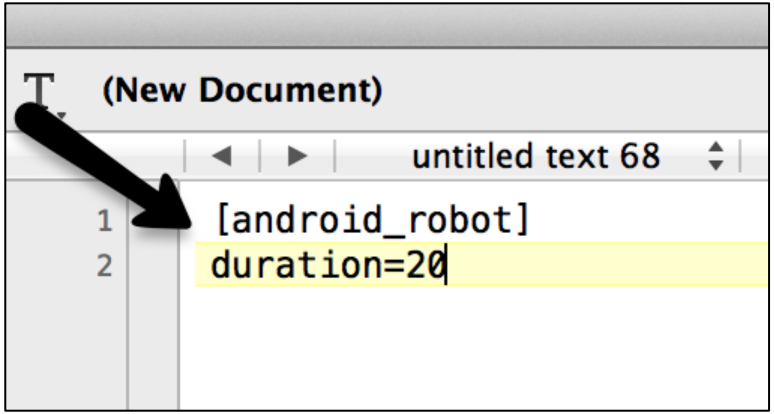
TEST TYPE
Android UI Robot

TEST SOURCE
Optionally, provide a bootstrap script to be run before the robot test.
Browse… DemoTest3.apk

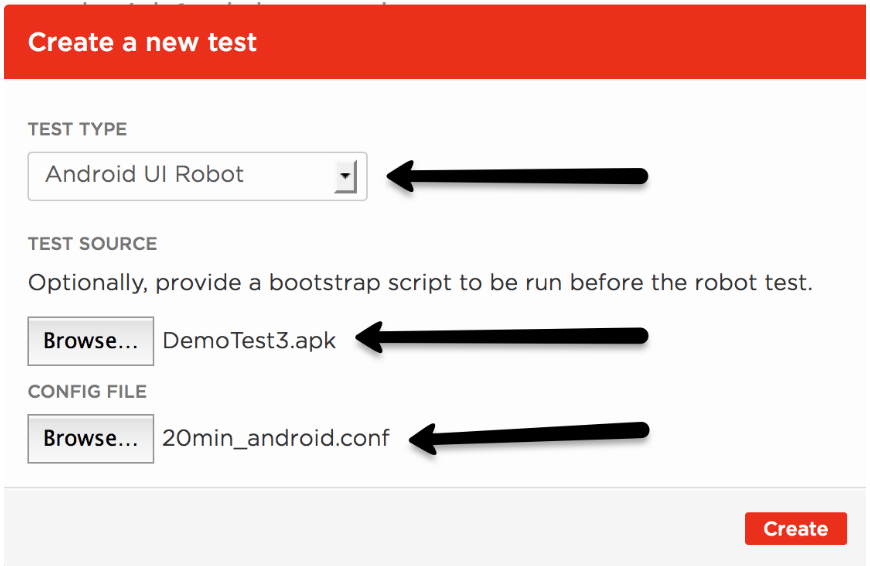
CONFIG FILE
Browse… No file selected.

To add a Standard Robot with bootstrap (Standard robot runs for 5 mins)

- Click “Add a test” and the “Create a New Test” box will appear.
- Choose test type. (For our test choose Android Robot)
- Browse for your Robotium script. (to find your .apk test script go to /path/to/**workspace/project/bin/project.apk**)
- You do not need to add a config file.
- Click Create



```
[android_robot]
duration=20
```


Create a new test

TEST TYPE

Android UI Robot

TEST SOURCE

Optionally, provide a bootstrap script to be run before the robot test.

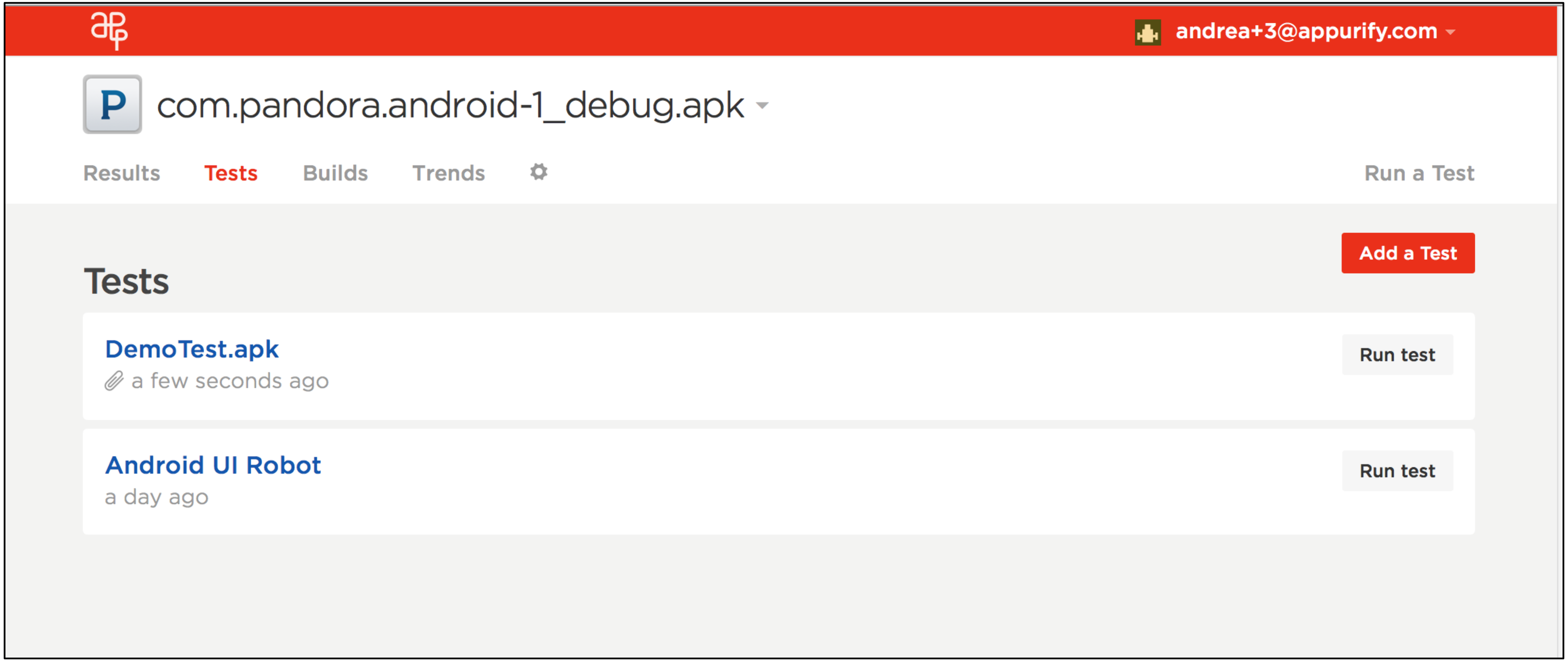
Browse... DemoTest3.apk

CONFIG FILE

Browse... 20min_android.conf

To add a Bootstrapped robot that runs for longer than 5 mins:

- Create a config file in a text editor. Set duration to # desired in mins under the tag [android_robot] and save it as “somename.conf”.
- On the platform, click “Add a test” and the “Create a New Test” box will appear.
- Choose test type. (For our test choose Android Robot).
- Browse for your Robotium script.
- Add the config file you just created.
- Click Create



andrea+3@appurify.com

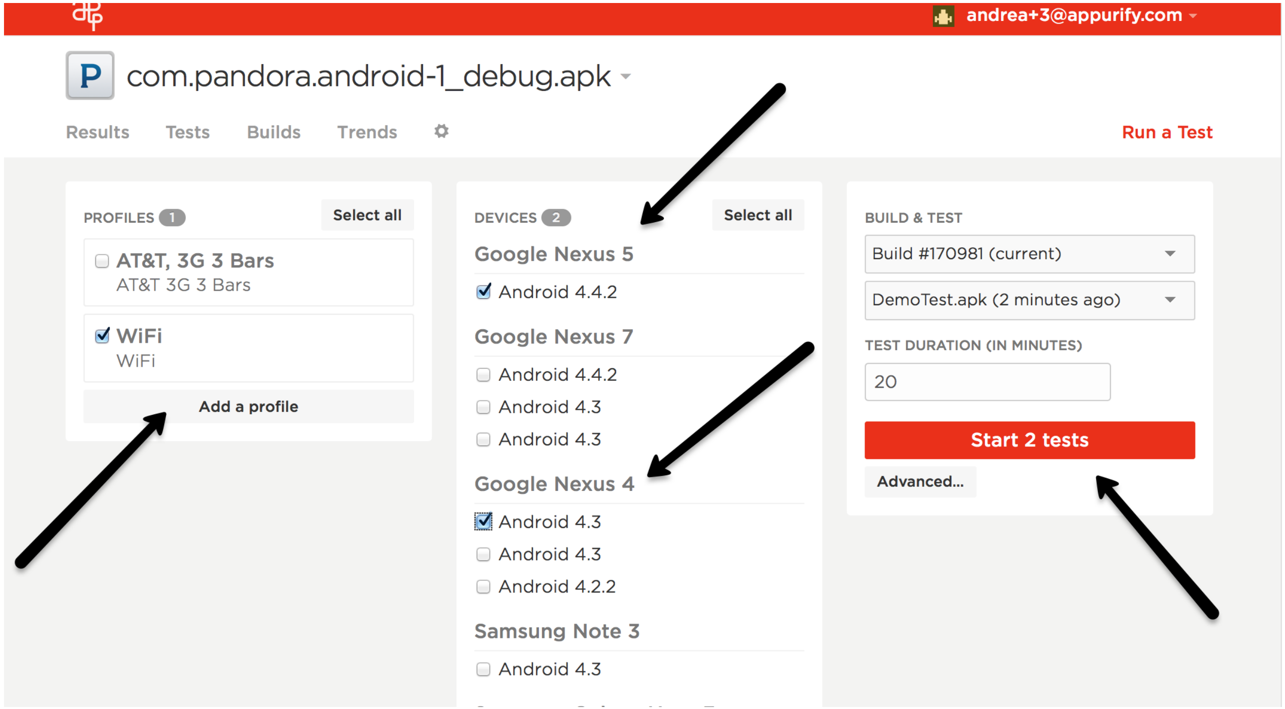
com.pandora.android-1_debug.apk

Tests

DemoTest.apk
a few seconds ago

Android UI Robot
a day ago

- Test should appear on test page
- Click “Run Test”



com.pandora.android-1_debug.apk

PROFILES 1

- ☐ AT&T, 3G 3 Bars — AT&T 3G 3 Bars
- ☑ WiFi — WiFi

DEVICES 2

Google Nexus 5
- ☑ Android 4.4.2

Google Nexus 7
- ☐ Android 4.4.2
- ☐ Android 4.3
- ☐ Android 4.3

Google Nexus 4
- ☑ Android 4.3
- ☐ Android 4.3
- ☐ Android 4.2.2

Samsung Note 3
- ☐ Android 4.3

BUILD & TEST

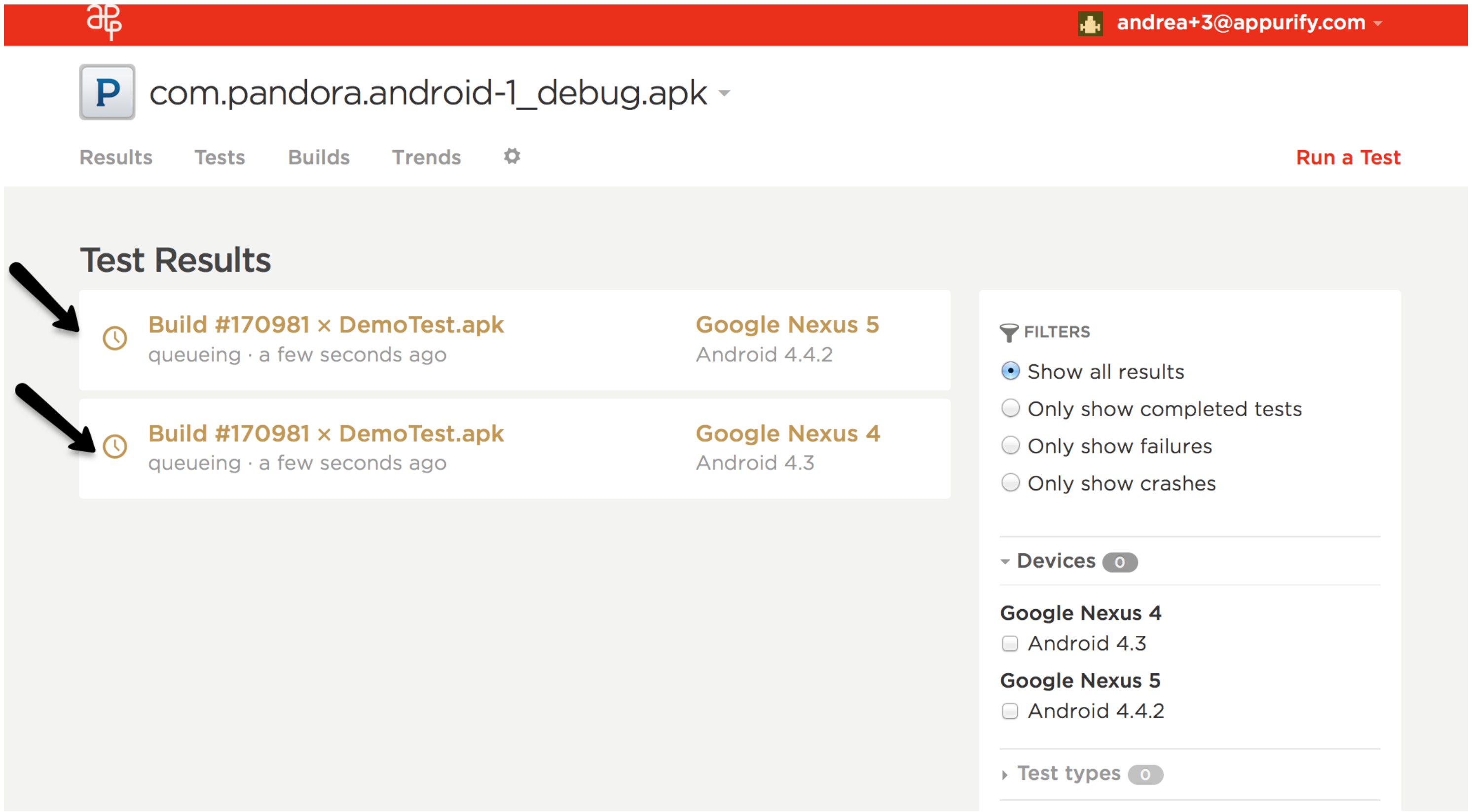
Build #170981 (current)

DemoTest.apk (2 minutes ago)

TEST DURATION (IN MINUTES)

20

- Select profile (Network to run devices on).
- Select Phones types to run your test on.
- Select “Start # tests”



com.pandora.android-1_debug.apk

Test Results

Build #170981 × DemoTest.apk — Google Nexus 5
queueing · a few seconds ago — Android 4.4.2

Build #170981 × DemoTest.apk — Google Nexus 4
queueing · a few seconds ago — Android 4.3

FILTERS

● Show all results
○ Only show completed tests
○ Only show failures
○ Only show crashes

Devices 0

Google Nexus 4
□ Android 4.3

Google Nexus 5
□ Android 4.4.2

Test types 0

- Tests will automatically appear in the results page with the status (queuing -> installing -> in progress -> completed)

Useful links and other details

- https://code.google.com/p/robotium/wiki/RobotiumTutorials
- http://www.vogella.com/tutorials/Robotium/article.html
- Video: https://www.youtube.com/watch?v=T_8euppCz3k
- Signing your tests: https://github.com/eemar7/AndroidAutomationTools/tree/master/Robotium/SigningAPKS

Running Robotium from command line

To run Robotium script from command line:

```
/path/to/adb -s 071f322f shell am instrument -w -e class appurify.testcases.Test -r 'appurify.testcases'/android.test.InstrumentationTestRunner
```

- 071f322f -- device UDID (can be found by running “/path/to/**adb devices**” with the phone plugged in
- appurify.testcases.Test -- the packagename.classname of your test
- ‘appurify.testcases' --- the packagename of your test