Selendroid - Mobile Test automation

Selendroid is a test automation framework which drives off the UI of Android native and hybrid applications (apps) and the mobile web. Tests are written using the Selenium 2 client API.Selendroid can be used on emulators and real devices and can be integrated as a node into the Selenium Grid for scaling and parallel testing.

**Getting started with Selendroid**

Selendroid is based on the Android instrumentation framework. Currently only the testing of one specific app is supported. Selendroid requires the following permission for your app: Usually it comes by default with in the androidmanifest.xml

<uses-permission android:name="android.\*\*permission.INTERNET"/>

## System Requirements

1. Selendroid can be used on Mac, Linux and Windows.
2. Java SDK (minimum 1.6) must be installed and *JAVA\_HOME* configured. IMPORTANT: If JAVA\_HOME is pointing to a Java runtime environment, selendroid will produce errors because tools like the *jarsigner* are not available!
3. Latest [**Android-Sdk**](http://developer.android.com/sdk/index.html) must be installed and *ANDROID\_HOME* set. [http://spring.io/guides/gs/android/]
4. If you run selendroid on a 64bit Linux machine, please install:
   1. sudo dpkg --add-architecture i386
   2. sudo apt-get update
   3. sudo apt-get install libc6:i386 libncurses5:i386 libstdc++6:i386
5. At least one Android virtual device must exist or an Android hardware device must be plugged in to the computer. For best practices about Android devices please read [this section](http://selendroid.io/setup.html#androidDevices).

## Getting an application under test

Selendroid can be used to test already built apps. Those Android apps (apk file) must exist on the machine, where the selendroid-standalone server will be started. The reason for this is that a customized selendroid-server for the app under test (AUT) will be created. Both apps (selendroid-server and AUT) must be signed with the same certificate in order to install the apks on the device.

## Launching Selendroid

Download [selendroid-standalone.jar](https://github.com/selendroid/selendroid/releases/download/0.17.0/selendroid-standalone-0.17.0-with-dependencies.jar) and cd to the folder where you downloaded the jar, and run [http://selendroid.io/setup.html#getAut]

java -jar selendroid-standalone-0.17.0-**with**-dependencies.jar -app selendroid-test-app-0.17.0.apk

Selendroid-standalone will start http server on port 4444 and will scan all Android virtual devices (avd) that the user has created (~/.android/avd/). You can check that the application(s) and the devices are recognized by opening a browser and navigating to: <http://localhost:4444/wd/hub/status>.

{status:0,value:{"os":{"name":"Android"},"build":{"browserName":"selendroid","version":"0.17.0"},"supportedApps":[{"appId":"io.selendroid.testapp:0.17.0","mainActivity":"io.selendroid.testapp.HomeScreenActivity","basePackage":"io.selendroid.testapp"}],"supportedDevices":[{"screenSize":"320x480","targetPlatform":"ANDROID17","emulator":true,"avdName":"latest"},{"screenSize":"320x480","targetPlatform":"ANDROID16","emulator":true,"avdName":"es"},{"screenSize":"320x480","targetPlatform":"ANDROID10","emulator":true,"avdName":"AVD\_for\_api10"}]}

**Sample tests**

The sample project is based on [Apache Maven](http://maven.apache.org/) and uses [JUnit 4](http://junit.org/) to run the tests.

**package** io.selendroid.demo.webui;

**import** io.selendroid.client.SelendroidDriver;

**import** io.selendroid.common.SelendroidCapabilities;

**import** java.util.concurrent.TimeUnit;

**import** org.junit.After;

**import** org.junit.Assert;

**import** org.junit.Before;

**import** org.junit.Test;

**import** org.openqa.selenium.By;

**import** org.openqa.selenium.WebDriver;

**import** org.openqa.selenium.WebElement;

**public** **class** EmployeeDirectoryTest {

**private** WebDriver driver = **null**;

        @Before

**public** **void** setup() **throws** Exception {

          driver = **new** SelendroidDriver(**new** SelendroidCapabilities("io.selendroid.directory:0.0.1"));

          driver.manage().timeouts().implicitlyWait(15, TimeUnit.*SECONDS*);

        }

        @Test

**public** **void** assertEmployeeCanBeDisplayedWithDirects() **throws** Exception {

          driver.get("and-activity://io.selendroid.directory.EmployeeDirectory");

          // Switch to the web view context

          driver.switchTo().window("WEBVIEW");

          String vpOfEngineering = "John Williams";

          driver.findElement(By.*tagName*("input")).sendKeys(vpOfEngineering);

          driver.findElement(By.*partialLinkText*(vpOfEngineering)).click();

          Assert.*assertEquals*(driver.getCurrentUrl(), "[file:///android\_asset/www/index.html#employees/4](file:///\\android_asset\www\index.html#employees/4)");

        /\*  // Verify Manager

          Assert.assertTrue(driver.findElements(By.tagName("li")).get(0).getText().

              endsWith("James King"));

          // Verify number of direct reports

          WebElement directs = driver.findElements(By.tagName("li")).get(1);

          Assert.assertTrue(directs.getText(). endsWith("3"));

          directs.click();

          Assert.assertEquals(driver.getCurrentUrl(),

              "[file:///android\_asset/www/index.html#employees/4/reports](file:///\\android_asset\www\index.html#employees/4/reports)");

          // Verify directs by name

          Assert.assertTrue(driver.findElements(By.tagName("li")).get(0).getText().

              startsWith("Paul Jones"));

          Assert.assertTrue(driver.findElements(By.tagName("li")).get(1).getText().

              startsWith("Paula Gates"));

          Assert.assertTrue(driver.findElements(By.tagName("li")).get(2).getText().

              startsWith("Steven Wells"));

          driver.navigate().back();\*/

          Assert.*assertEquals*(driver.getCurrentUrl(), "[file:///android\_asset/www/index.html#employees/4](file:///\\android_asset\www\index.html#employees/4)");

        }

        @After

**public** **void** teardown() {

          //driver.quit();

        }

      }