

Mobile Automation with Appium

1. Android SDK
2. Android Device (configuration)
3. Eclipse / Java
4. Appium

Setting up Android SDK

Step 1: Open <https://developer.android.com/studio/index.html>

Step 2: Scroll to “Get just the command line tools”

(* You can download Android Studio also instead of SDK Tools but for testing SDK tools is sufficient)

Step 3: Download SDK Tools for respective platform

Get just the command line tools

If you do not need Android Studio, you can download the basic Android command line tools and other SDK packages.

These tools are included in Android Studio.

Platform	SDK tools package	Size	SHA-256
Windows	sdk-tools-windows-3859397.zip	132 MB (138,449,982 bytes)	7f6037d
Mac	sdk-tools-darwin-3859397.zip	82 MB (86,182,133 bytes)	4a81754
Linux	sdk-tools-linux-3859397.zip	130 MB (136,964,098 bytes)	444e22c

Step 4: Extract download zip

Step 5: Open “cmd” and navigate to the tools folder in SDK.

Step 6: commands

1. Sdkmanager.bat --list
2. Sdkmanager.bat platforms;android-26
3. Sdkmanager.bat build-tools;26.0.3

4. sdkmanager.bat platform-tools

Step 7: set environment variable.

1. ANDROID_HOME = path to sdk folder
2. PATH
\$ANDROID_HOME/build-tools/25.0.2;\$ANDROID_HOME/platform-tools;\$ANDROID_HOME/tools;

* Note - Update path according your installation

Step 8. Close all CMD and start again. Execute adb device command. If you get error for adb then check installation.

Device Set Up configuration:

1. Enable developer option
2. Enable usb debugging
3. Connect device to computer via usb and check "adb devices" shows device connected.

Appium Set Up

1. Download and Install Node.js <https://nodejs.org/en/>
2. Open Node Js command prompt and execute below command
 - a. npm install appium -g
3. Verify appium installed with below command
 - a. appium -v

Contents

- Environment Setup and Tools - Done
- Locating Mobile Elements - Done
- Native App Automation on real android device - Done
- Appium Server - Done
- Automating Built In Apps & System controls - Done
- Hybrid App Automation - Done
- iOS Automation
- Mobile Web Automation

Day 2 -

Locating Android elements

Id, Xpath

IOS - Name, Xpath

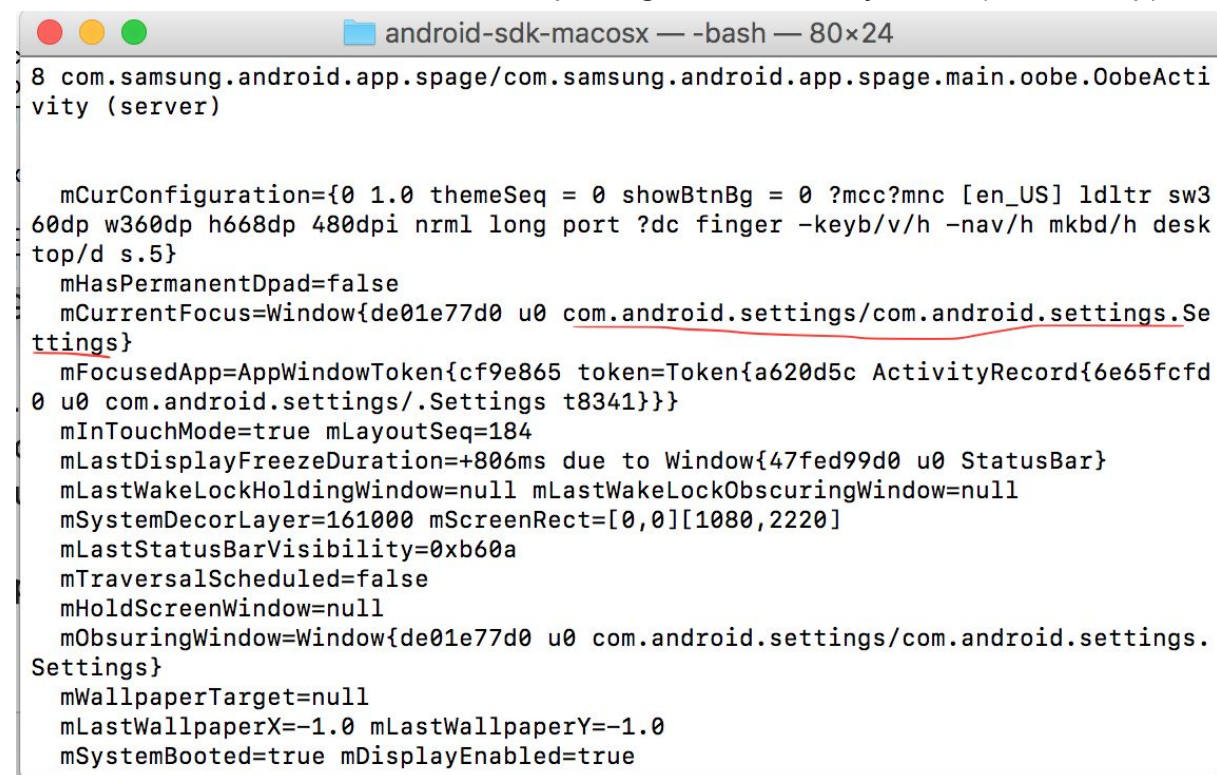
Automating already existing App using Appium.

Step 1: Get package name and activity from the Application.

- I. Start app. Launch screen
- li. Execute below 2 commands

```
adb shell
dumpsys window windows
```

Search for 'mCurrentFocus=' Get the package name /activity name (check snap)

A screenshot of an Android terminal window titled 'android-sdk-macosx — -bash — 80x24'. The terminal shows the output of the 'dumpsys window windows' command. The output includes various system properties and a 'mCurrentFocus' entry. The 'mCurrentFocus' entry is highlighted with a red underline and shows the package name 'com.android.settings' and the activity name 'com.android.settings.Settings'.

```
8 com.samsung.android.app.spage/com.samsung.android.app.spage.main.oobe.OobeActi
vity (server)

{
  mCurConfiguration={0 1.0 themeSeq = 0 showBtnBg = 0 ?mcc?mnc [en_US] ldltr sw3
60dp w360dp h668dp 480dpi nrml long port ?dc finger -keyb/v/h -nav/h mkbd/h desk
top/d s.5}
  mHasPermanentDpad=false
  mCurrentFocus=Window{de01e77d0 u0 com.android.settings/com.android.settings.Se
ttings}
  mFocusedApp=AppWindowToken{cf9e865 token=Token{a620d5c ActivityRecord{6e65fcfd
0 u0 com.android.settings/.Settings t8341}}}
  mInTouchMode=true mLayoutSeq=184
  mLastDisplayFreezeDuration=+806ms due to Window{47fed99d0 u0 StatusBar}
  mLastWakeLockHoldingWindow=null mLastWakeLockObscuringWindow=null
  mSystemDecorLayer=161000 mScreenRect=[0,0][1080,2220]
  mLastStatusBarVisibility=0xb60a
  mTraversalScheduled=false
  mHoldScreenWindow=null
  mObscuringWindow=Window{de01e77d0 u0 com.android.settings/com.android.settings.
Settings}
  mWallpaperTarget=null
  mLastWallpaperX=-1.0 mLastWallpaperY=-1.0
  mSystemBooted=true mDisplayEnabled=true
```

Package name: com.android.settings

Activity name: com.android.settings.settings

Step 2.

Set both to desired capabilities

```
DesiredCapabilities capabilities=new DesiredCapabilities();
capabilities.setCapability("deviceName", "Your device Name");
capabilities.setCapability("appPackage", "com.android.settings");
```

```
capabilities.setCapability("appActivity",  
"com.android.settings.Settings");
```

That's all you need...Now you can create driver instance and use it for automation.