



MAR 2021



REBON רְבּוֹנָה ROBOCON REMOTE EVENT

#ROBOCON20



Robot
Framework
Foundation



MAR 2021



Getting Started With The Robotframework-AppiumLibrary

:: RoboCon Workshop Presented By ::

Joshua Gorospe
Principal QA Engineer
Secureworks <https://www.secureworks.com/>

LinkedIn Profile <https://www.linkedin.com/in/joshua8481/>
Twitter Profile <https://twitter.com/8481jq>



ROBOCON 2021



MAR 2021



Before We Begin...

I want to thank the following people and groups.

- Pekka Klarck
- Ed Manlove
- Serhat Bolsu
- Antti Karjalainen
- René Rohner
- James Bach
- Jonathan Bach
- Michael Bolton
- Cem Kaner
- Kristian Karl
- Harry Robinson
- Mikko Korpela
- Bryan Oakley
- Shiva Prasad Adirala
- The Appium development team
- The Graphwalker development team
- The thought-leaders who inspired me to learn model-based testing
- The thought-leaders of Rapid Software Testing Methodology
- The entire Robot Framework community and its contributors





MAR 2021



FYI Windows Users

The Robotframework-AppiumLibrary examples in this workshop use a lot of Bash scripting. Here are some possible options that may help you, but I have not tested them myself...

- Windows Subsystem for Linux <https://docs.microsoft.com/en-us/windows/wsl/>
- Git for Windows <https://git-scm.com/download/win>
- CygWin <https://www.cygwin.com/>



Introduction And Workshop Outline

This will be a very hands on Workshop covering Robotframework-AppiumLibrary combined with various tools. I will be frequently moving back and forth between slides and running the examples from a terminal. From a high-level, the following Workshop areas will be covered in two parts.

- Part One
 - Technical requirements summary.
 - Appium installation advice and mobile device Emulator/Simulator setup guidance.
 - App automation workflows for IOS, and Android (including adb shell).
 - Browser automation workflows for Safari on IOS and Chrome on Android.
- Part Two
 - Appium Desktop walkthrough, and useful features for Robot Framework users.
 - Charles Proxy combined with Robotframework-AppiumLibrary.
 - PaBot running IOS + Android simultaneously, combined with Graphwalker, and Android Emulator CPU monitoring.



Part One: Technical Requirements Summary

The following are needed to run the Workshop Examples. Please follow instructions on each website to install them.

- Python 3, this workshop was created using version 3.9.2 -> <https://www.python.org/downloads/>
- NodeJS 14, this workshop was created using version 14.16.0 -> <https://nodejs.org/en/> or <https://github.com/nvm-sh/nvm>
- Java 11, this workshop was created using version 11.0.10 -> <https://www.java.com/en/>
- Appium Server -> <http://appium.io/docs/en/about-appium/getting-started/?lang=en#getting-started>
- Android Studio -> <https://developer.android.com/studio>
- XCode, only available for MacOS users -> <https://developer.apple.com/xcode/>
- Robot Framework and Robotframework-AppiumLibrary -> <https://robotframework.org>
- PaBot -> <https://github.com/mkorpela/pabot>
- Charles Proxy -> <https://www.charlesproxy.com/documentation/proxying>
- Graphwalker -> <https://github.com/GraphWalker/graphwalker-project>
- Appium Desktop -> <https://github.com/appium/appium-desktop>
- wget -> <https://www.gnu.org/software/wget/>
- cURL -> <https://curl.se/>
- jq <https://stedolan.github.io/jq/>
- git <https://git-scm.com/>





Part One: Technical Requirements Summary

Run the following commands to get the Workshop Examples ([requires git](#)).

- [git clone https://github.com/jq8481/Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021.git](https://github.com/jq8481/Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021.git)
- cd ./Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021
- You can also download a zip file from the GitHub page ->
<https://github.com/jq8481/Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021>

The screenshot shows a GitHub repository page. At the top left is a 'README.md' file icon. On the right, there are download options: 'Use Git or checkout with SVN using the web URL.', 'Open with GitHub Desktop', and a highlighted 'Download ZIP' button, which is enclosed in an orange rounded rectangle. The main content area features a large title: 'Getting started with the Robotframework - RoboCon 2021 Workshop'. Below the title is a section titled 'General Information'.

This [repo](#) contains all of the examples that will be covered in the "Getting started with the Robotframework-AppiumLibrary" online workshop. The core audience for this workshop are beginners but there's also content mentioned below that could be interesting to more advanced testers as well. A lot of the examples kept in this repo were heavily influenced by the [Robot-Framework-Lone-Tester-Strategies-RoboCon-2019](#) and [Tool-Strategies-Lone-Tester-Strategies-RoboCon-2019](#).



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Please make sure you have reviewed all of the items in the Technical Requirements Summary slides.

- The rest of this Part One section assumes you have installed all of the basics.
- Basic requirements for this Workshop are NodeJS 14, Python 3, and Java 11
 - I recommend NVM (<https://github.com/nvm-sh/nvm>) for NodeJS installation.
 - Installing anything with Homebrew (<https://brew.sh/>) may fail sometimes, for example this happened after I installed the MacOS Big Sur update.

```
[joshuas-MBP:~ jgorospe$ brew install node
==> Downloading https://homebrew.bintray.com/bottles/icu4c-68.2.big_sur.bottle.tar.gz

curl: (35) error:1400410B:SSL routines:CONNECT_CR_SRVR_HELLO:wrong version number
Error: Failed to download resource "icu4c"
Download failed: https://homebrew.bintray.com/bottles/icu4c-68.2.big_sur.bottle.tar.gz
joshuas-MBP:~ jgorospe$ ]
```

Based on the Appium "Getting Started" instructions kept here ->

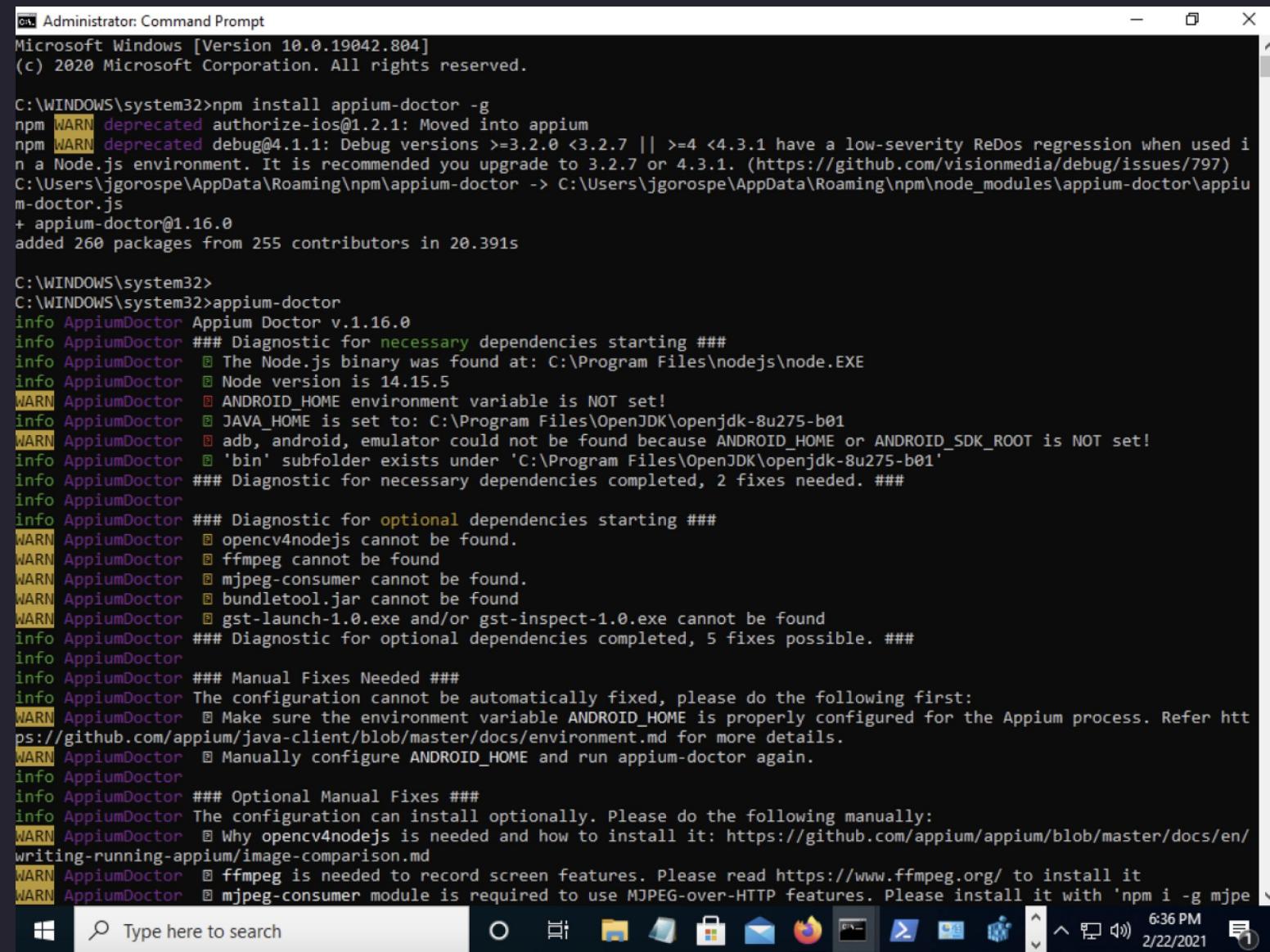
<http://appium.io/docs/en/about-appium/getting-started/index.html>, you should be able to run the following commands successfully without issues.

- Install Appium Server -> [npm install -g appium](#)
- Install appium-doctor -> [npm install -g appium-doctor](#)
- Run appium-doctor...



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

From a Windows operating system the appium-doctor output should look similar to this.



```
C:\WINDOWS\system32>npm install appium-doctor -g
npm WARN deprecated authorize-ios@1.2.1: Moved into appium
npm WARN deprecated debug@4.1.1: Debug versions >=3.2.0 <3.2.7 || >=4 <4.3.1 have a low-severity ReDos regression when used in a Node.js environment. It is recommended you upgrade to 3.2.7 or 4.3.1. (https://github.com/visionmedia/debug/issues/797)
C:\Users\jgorospe\AppData\Roaming\npm\appium-doctor -> C:\Users\jgorospe\AppData\Roaming\npm\node_modules\appium-doctor\appium-doctor.js
+ appium-doctor@1.16.0
added 260 packages from 255 contributors in 20.391s

C:\WINDOWS\system32>
C:\WINDOWS\system32>appium-doctor
info AppiumDoctor Appium Doctor v.1.16.0
info AppiumDoctor ### Diagnostic for necessary dependencies starting ###
info AppiumDoctor ② The Node.js binary was found at: C:\Program Files\nodejs\node.EXE
info AppiumDoctor ③ Node version is 14.15.5
WARN AppiumDoctor ④ ANDROID_HOME environment variable is NOT set!
info AppiumDoctor ④ JAVA_HOME is set to: C:\Program Files\OpenJDK\openjdk-8u275-b01
WARN AppiumDoctor ④ adb, android, emulator could not be found because ANDROID_HOME or ANDROID_SDK_ROOT is NOT set!
info AppiumDoctor ④ 'bin' subfolder exists under 'C:\Program Files\OpenJDK\openjdk-8u275-b01'
info AppiumDoctor ### Diagnostic for necessary dependencies completed, 2 fixes needed. ###
info AppiumDoctor
info AppiumDoctor ### Diagnostic for optional dependencies starting ###
WARN AppiumDoctor ④ opencv4nodejs cannot be found.
WARN AppiumDoctor ④ ffmpeg cannot be found
WARN AppiumDoctor ④ mjpeg-consumer cannot be found.
WARN AppiumDoctor ④ bundletool.jar cannot be found
WARN AppiumDoctor ④ gst-launch-1.0.exe and/or gst-inspect-1.0.exe cannot be found
info AppiumDoctor ### Diagnostic for optional dependencies completed, 5 fixes possible. ###
info AppiumDoctor
info AppiumDoctor ### Manual Fixes Needed ###
info AppiumDoctor The configuration cannot be automatically fixed, please do the following first:
WARN AppiumDoctor ④ Make sure the environment variable ANDROID_HOME is properly configured for the Appium process. Refer https://github.com/appium/java-client/blob/master/docs/environment.md for more details.
WARN AppiumDoctor ④ Manually configure ANDROID_HOME and run appium-doctor again.
info AppiumDoctor
info AppiumDoctor ### Optional Manual Fixes ###
info AppiumDoctor The configuration can install optionally. Please do the following manually:
WARN AppiumDoctor ④ Why opencv4nodejs is needed and how to install it: https://github.com/appium/appium/blob/master/docs/en/writing-running-appium/image-comparison.md
WARN AppiumDoctor ④ ffmpeg is needed to record screen features. Please read https://www.ffmpeg.org/ to install it
WARN AppiumDoctor ④ mjpeg-consumer module is required to use MJPEG-over-HTTP features. Please install it with 'npm i -g mjpeg'
```





Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

From a Linux (Ubuntu) operating system the appium-doctor output should look similar to this.

Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Pay attention to any red "X" output. Follow all of the instructions provided by appium-doctor.

```
jgorospe -- bash -- 128x32
info AppiumDoctor ✓ The Authorization DB is set up properly.
WARN AppiumDoctor ✖ Carthage was NOT found!
info AppiumDoctor ✓ HOME is set to: /users/jgorospe
info AppiumDoctor ✓ ANDROID_HOME is set to: /Users/jgorospe/Library/Android/sdk
info AppiumDoctor ✓ JAVA_HOME is set to: /Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home
info AppiumDoctor Checking adb, android, emulator
info AppiumDoctor     'adb' is in /Users/jgorospe/Library/Android/sdk/platform-tools/adb
info AppiumDoctor     'android' is in /Users/jgorospe/Library/Android/sdk/tools/android
info AppiumDoctor     'emulator' is in /Users/jgorospe/Library/Android/sdk/emulator/emulator
info AppiumDoctor ✓ adb, android, emulator exist: /Users/jgorospe/Library/Android/sdk
info AppiumDoctor ✓ 'bin' subfolder exists under '/Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home'
info AppiumDoctor ### Diagnostic for necessary dependencies completed, one fix needed. #####
info AppiumDoctor
info AppiumDoctor ### Diagnostic for optional dependencies starting #####
WARN AppiumDoctor ✖ opencv4nodejs cannot be found.
WARN AppiumDoctor ✖ ffmpeg cannot be found
WARN AppiumDoctor ✖ mjpeg-consumer cannot be found.
WARN AppiumDoctor ✖ set-simulator-location is not installed
WARN AppiumDoctor ✖ idb and idb_companion are not installed
WARN AppiumDoctor ✖ applesimutils cannot be found
info AppiumDoctor ✓ ios-deploy is installed at: /usr/local/bin/ios-deploy. Installed version is: 1.9.4
WARN AppiumDoctor ✖ bundletool.jar cannot be found
WARN AppiumDoctor ✖ gst-launch-1.0 and/or gst-inspect-1.0 cannot be found
info AppiumDoctor ### Diagnostic for optional dependencies completed, 8 fixes possible. #####
info AppiumDoctor
info AppiumDoctor ### Manual Fixes Needed #####
info AppiumDoctor The configuration cannot be automatically fixed, please do the following first:
WARN AppiumDoctor → [For lower than Appium 1.20.0] Please install Carthage. Visit https://github.com/Carthage/Carthage#installing-carthage for more information.
info AppiumDoctor
info AppiumDoctor ### Optional Manual Fixes #####
info AppiumDoctor The configuration can install optionally. Please do the following manually:
```



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

After following all of the instructions, all of the appium-doctor output should eventually show green check marks. This is a MacOS example.

```
joshuas-MBP:~ jgorospe$ appium-doctor
info AppiumDoctor Appium Doctor v.1.16.0
info AppiumDoctor ### Diagnostic for necessary dependencies starting ####
info AppiumDoctor ✓ The Node.js binary was found at: /Users/jgorospe/.nvm/versions/node/v14.16.0/bin/node
info AppiumDoctor ✓ Node version is 14.16.0
info AppiumDoctor ✓ Xcode is installed at: /Applications/Xcode.app/Contents/Developer
info AppiumDoctor ✓ Xcode Command Line Tools are installed in: /Applications/Xcode.app/Contents/Developer
info AppiumDoctor ✓ DevToolsSecurity is enabled.
info AppiumDoctor ✓ The Authorization DB is set up properly.
info AppiumDoctor ✓ Carthage was found at: /usr/local/bin/carthage. Installed version is: 0.37.0
info AppiumDoctor ✓ HOME is set to: /Users/jgorospe
info AppiumDoctor ✓ ANDROID_HOME is set to: /Users/jgorospe/Library/Android/sdk
info AppiumDoctor ✓ JAVA_HOME is set to: /Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home
info AppiumDoctor Checking adb, android, emulator
      'adb' is in /Users/jgorospe/Library/Android/sdk/platform-tools/adb
      'android' is in /Users/jgorospe/Library/Android/sdk/tools/android
      'emulator' is in /Users/jgorospe/Library/Android/sdk/emulator/emulator
info AppiumDoctor ✓ adb, android, emulator exist: /Users/jgorospe/Library/Android/sdk
info AppiumDoctor ✓ 'bin' subfolder exists under '/Library/Java/JavaVirtualMachines/adoptopenjdk-8.jdk/Contents/Home'
info AppiumDoctor ### Diagnostic for necessary dependencies completed, no fix needed. ####
info AppiumDoctor
info AppiumDoctor ### Diagnostic for optional dependencies starting ####
WARN AppiumDoctor ✘ opencv4nodejs cannot be found.
WARN AppiumDoctor ✘ ffmpeg cannot be found
WARN AppiumDoctor ✘ mjpeg-consumer cannot be found.
WARN AppiumDoctor ✘ set-simulator-location is not installed
WARN AppiumDoctor ✘ idb and idb_companion are not installed
WARN AppiumDoctor ✘ applesimutils cannot be found
info AppiumDoctor ✓ ios-deploy is installed at: /usr/local/bin/ios-deploy. Installed version is: 1.9.4
WARN AppiumDoctor ✘ bundletool.jar cannot be found
WARN AppiumDoctor ✘ gst-launch-1.0 and/or gst-inspect-1.0 cannot be found
info AppiumDoctor ### Diagnostic for optional dependencies completed, 8 fixes possible. ####
```

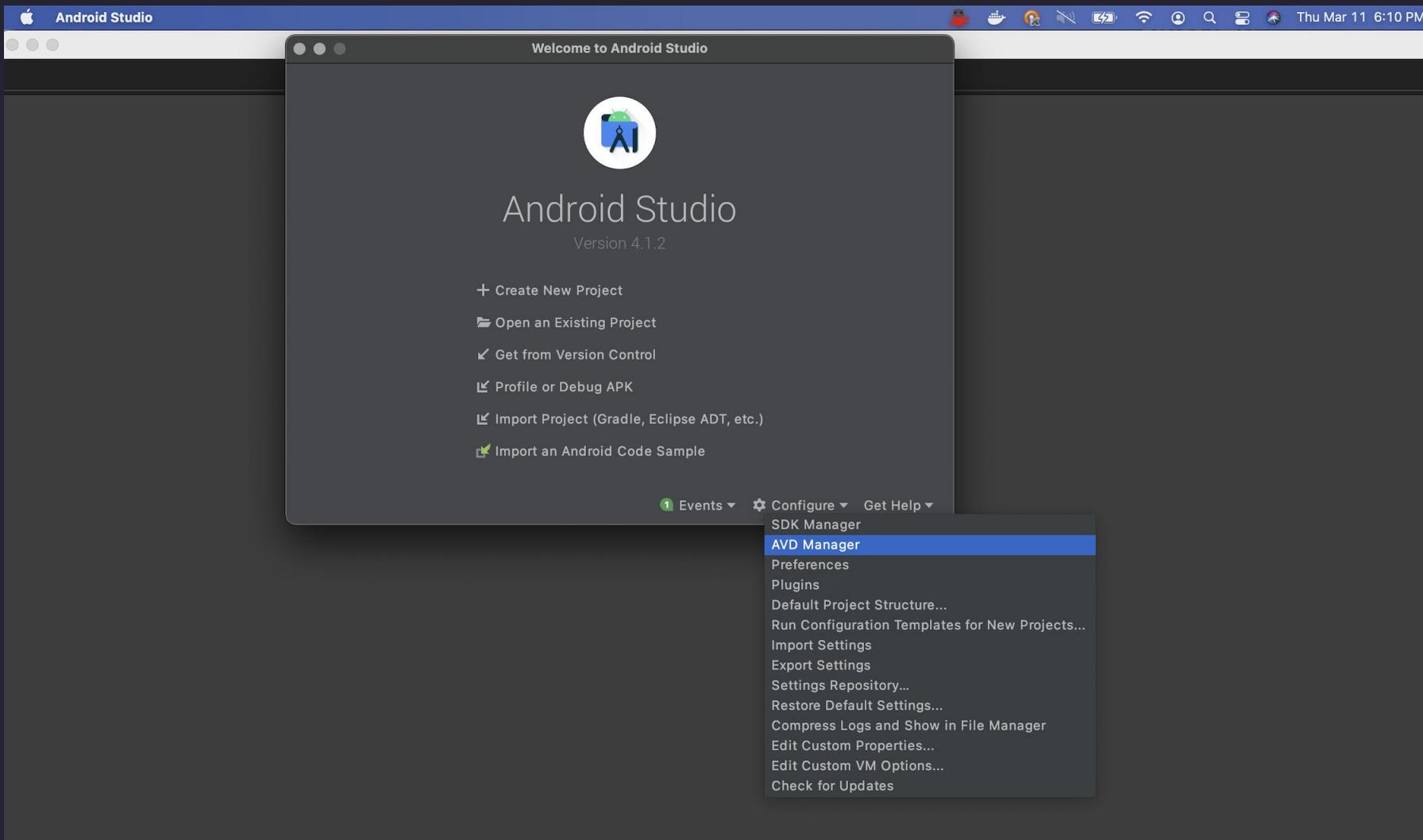




Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your Android Emulator.

- Start Android Studio.
- Click on the "Configure" button, and select AVD Manager.

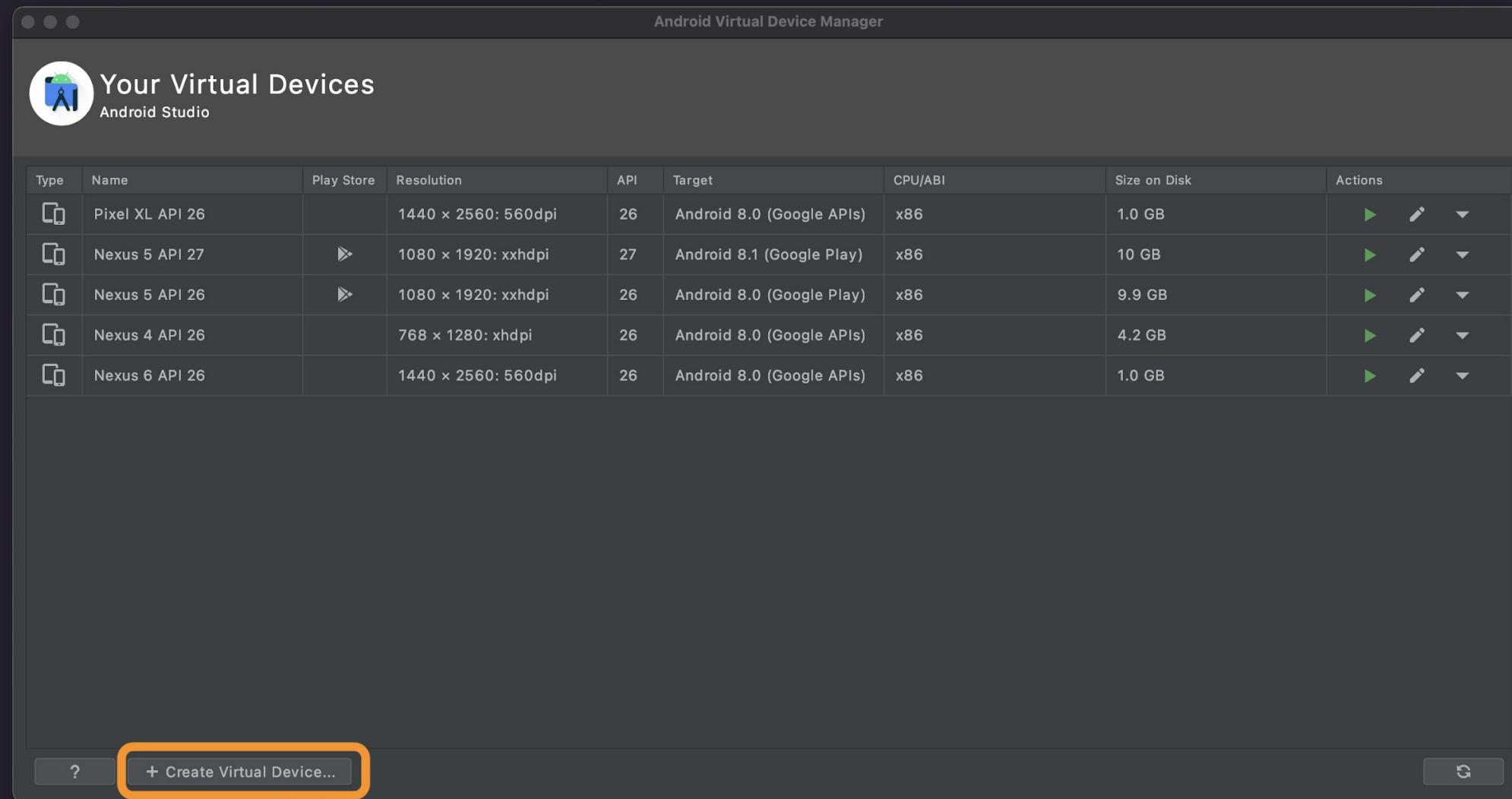




Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your Android Emulator.

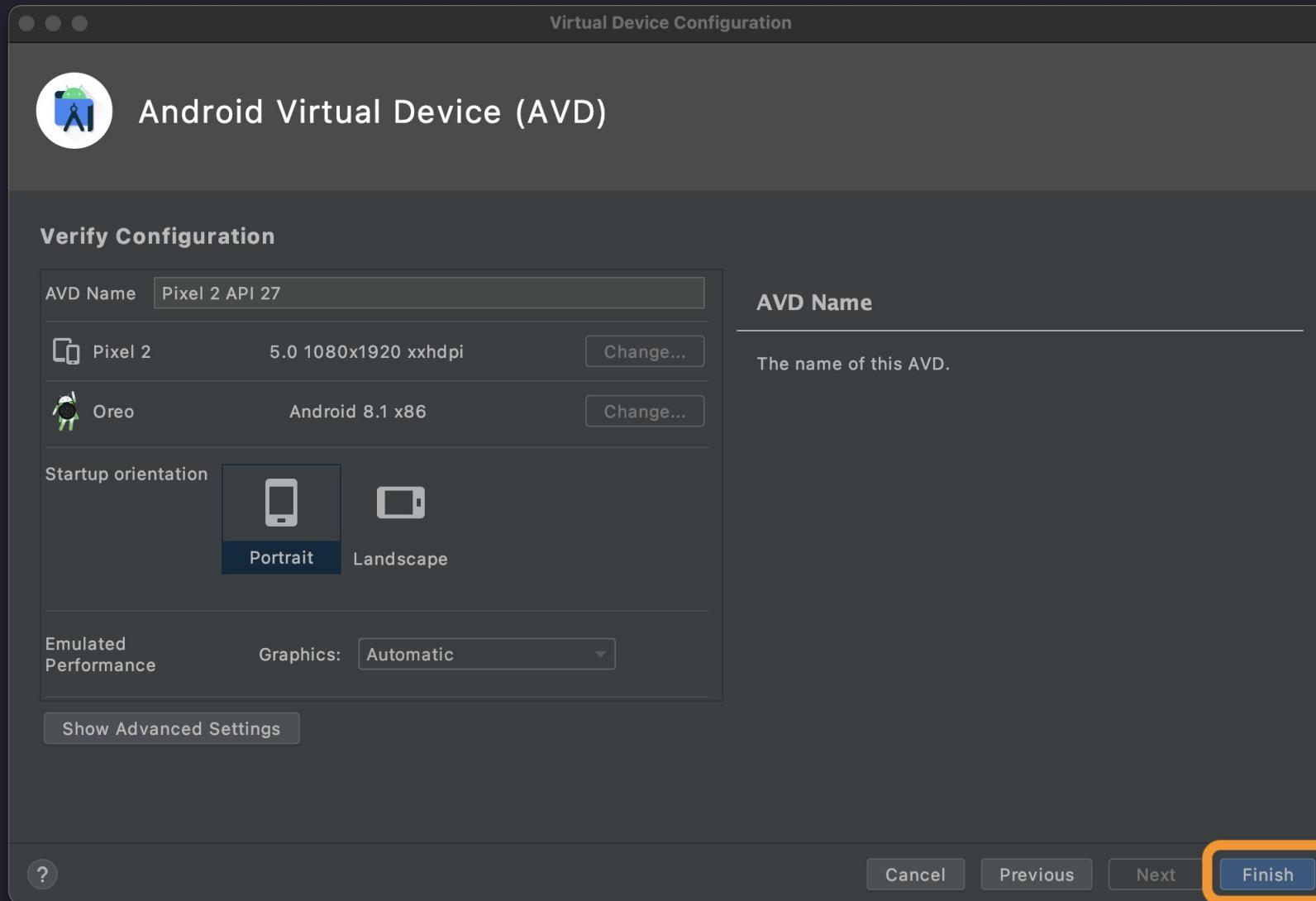
- From "AVD Manager", click on the "Create Virtual Device" button.
- The "Virtual Device Configuration" will appear, select all of the options you need for your device.



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your Android Emulator.

- Click the "Finish" button from "Virtual Device Configuration".
- Your new device should appear on the "AVD Manager" list.



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your Android Emulator.

- Go to the root of the Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021 repo.
- Open the start-specific-appium-example-workflows-for-workshop.sh file in an editor.
- Change the following (two places in the Bash script) to match the Android device that you have just set up...or...
- Workshop demonstrations are using a Nexus 5 API 27. You can also set up the same Android device.

####

```
#### Copy and paste the following line for every new Android emulator device you want to add to every test setup. The following d
## "$HOME"/Library/Android/sdk/emulator/emulator -avd REPLACE_THIS_WITH_THE_NAME_OF_YOUR_ANDROID_EMULATOR -netdelay none -netspee
#### Copy and paste the following line for every new Android emulator device you want to add to every test setup. The following u
## "$HOME"/Library/Android/sdk/emulator/emulator -avd REPLACE_THIS_WITH_THE_NAME_OF_YOUR_ANDROID_EMULATOR -netdelay none -netspee
####
"$HOME"/Library/Android/sdk/emulator/emulator -avd Nexus_5_API_27 netdelay none -netspeed full > $path/Workshop-Examples/Shared-
#"$HOME"/Library/Android/sdk/emulator/emulator -avd Nexus_6_API_26 -netdelay none -netspeed full > $path/Workshop-Examples/Shared
#"$HOME"/Library/Android/sdk/emulator/emulator -avd Pixel_XL_API_28 -netdelay none -netspeed full > $path/Workshop-Examples/Share
#####
sleep 5s &&
appium -p 4723 --webdriveragent-port 8109 >> $path/Workshop-Examples/Shared-Resources/appium_output_log.txt 2>&1 & echo $! > $pat
appium -p 4724 -bp 5724 --allow-insecure chromedriver_autodownload --relaxed-security >> $path/Workshop-Examples/Shared-Resources
echo
echo
APPIUM_SERVER_PID1=$(cat ./Workshop-Examples/Shared-Resources/appium_server_PID1.txt)
```



Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your IOS Simulator.

- Installing XCode on your Mac machine will automatically install IOS Simulators for you.
- To check your available IOS Simulators, run this command -> xcrun simctl list
- Workshop demonstrations are using an iPad Pro (12.9-inch) (4th generation). Check it in your list.

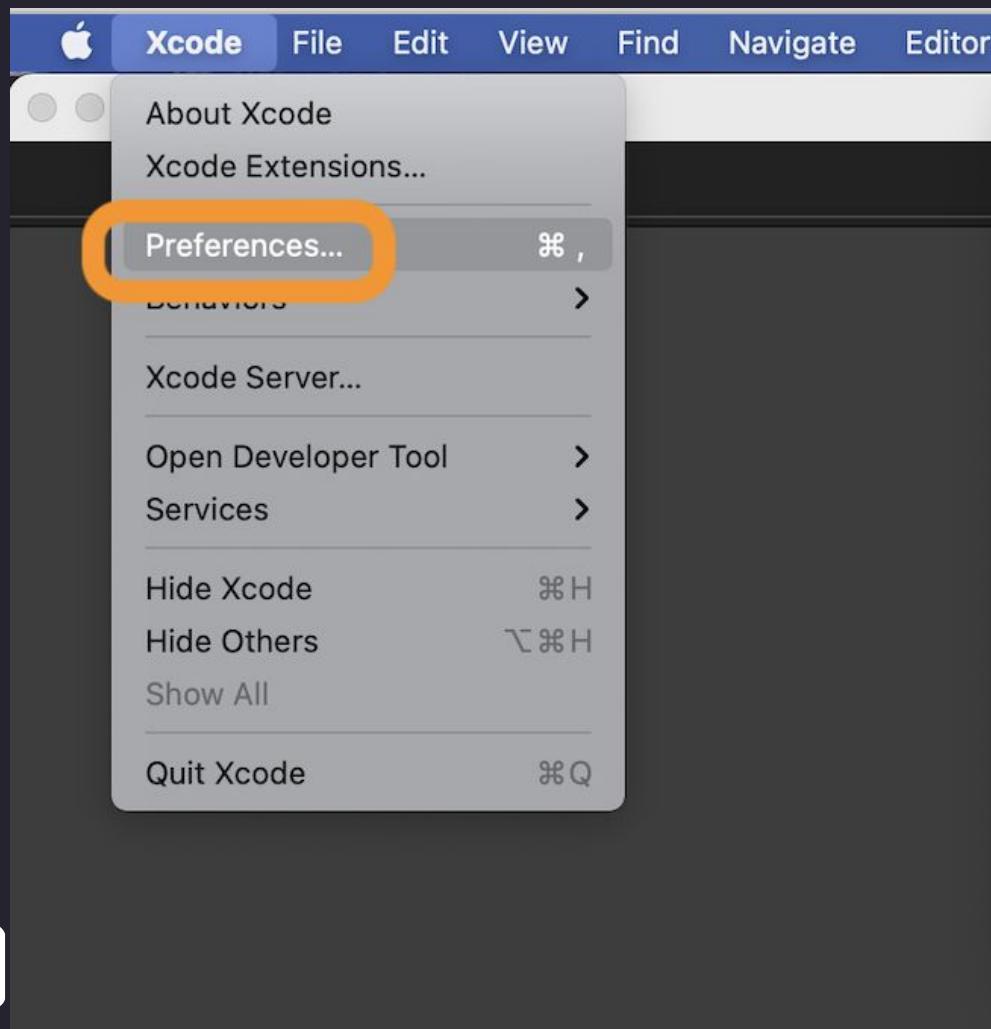


```
Joshua-MBP:Getting-Started-Robotframework-Appiumlibrary-RoboCon-2021 jgorospe$ xcrun simctl list
== Device Types ==
iPhone 4s (com.apple.CoreSimulator.SimDeviceType.iPhone-4s)
iPhone 5 (com.apple.CoreSimulator.SimDeviceType.iPhone-5)
iPhone 5s (com.apple.CoreSimulator.SimDeviceType.iPhone-5s)
iPhone 6 Plus (com.apple.CoreSimulator.SimDeviceType.iPhone-6-Plus)
iPhone 6 (com.apple.CoreSimulator.SimDeviceType.iPhone-6)
iPhone 6s (com.apple.CoreSimulator.SimDeviceType.iPhone-6s)
iPhone 6s Plus (com.apple.CoreSimulator.SimDeviceType.iPhone-6s-Plus)
iPhone SE (1st generation) (com.apple.CoreSimulator.SimDeviceType.iPhone-SE)
iPhone 7 (com.apple.CoreSimulator.SimDeviceType.iPhone-7)
iPhone 7 Plus (com.apple.CoreSimulator.SimDeviceType.iPhone-7-Plus)
iPhone 8 (com.apple.CoreSimulator.SimDeviceType.iPhone-8)
iPhone 8 Plus (com.apple.CoreSimulator.SimDeviceType.iPhone-8-Plus)
iPhone X (com.apple.CoreSimulator.SimDeviceType.iPhone-X)
iPhone Xs (com.apple.CoreSimulator.SimDeviceType.iPhone-XS)
iPhone Xs Max (com.apple.CoreSimulator.SimDeviceType.iPhone-XS-Max)
iPhone XR (com.apple.CoreSimulator.SimDeviceType.iPhone-XR)
iPhone 11 (com.apple.CoreSimulator.SimDeviceType.iPhone-11)
iPhone 11 Pro (com.apple.CoreSimulator.SimDeviceType.iPhone-11-Pro)
iPhone 11 Pro Max (com.apple.CoreSimulator.SimDeviceType.iPhone-11-Pro-Max)
iPhone SE (2nd generation) (com.apple.CoreSimulator.SimDeviceType.iPhone-SE--2nd-generation-)
iPhone 12 mini (com.apple.CoreSimulator.SimDeviceType.iPhone-12-mini)
iPhone 12 (com.apple.CoreSimulator.SimDeviceType.iPhone-12)
iPhone 12 Pro (com.apple.CoreSimulator.SimDeviceType.iPhone-12-Pro)
iPhone 12 Pro Max (com.apple.CoreSimulator.SimDeviceType.iPhone-12-Pro-Max)
iPod touch (7th generation) (com.apple.CoreSimulator.SimDeviceType.iPod-touch--7th-generation-)
iPad 2 (com.apple.CoreSimulator.SimDeviceType.iPad-2)
iPad Retina (com.apple.CoreSimulator.SimDeviceType.iPad-Retina)
iPad Air (com.apple.CoreSimulator.SimDeviceType.iPad-Air)
iPad mini 2 (com.apple.CoreSimulator.SimDeviceType.iPad-mini-2)
```

Part One: Appium Installation Advice, Android Emulator, IOS Simulator Setups

Steps for setting up your IOS Simulator.

- XCode can also automatically check for new IOS updates by going to "Preferences", then "Components"
- You can enable the automatic update checkbox, or you can click "Check and Install Now".
- I installed iOS 13.7 and 14.4, which has iPad Pro (12.9-inch) (4th generation) in the xcrun simctl list output (previous slide).



The screenshot shows the 'Components' tab of the Xcode Preferences window. The 'Components' tab is highlighted with a thick orange border. Below it, the 'Simulators' tab is active, showing a list of available iOS simulators with their sizes:

Simulator	Size
iOS 14.2 Simulator	4.91 GB
iOS 14.1 Simulator	4.88 GB
iOS 14.0 Simulator	4.71 GB
iOS 13.7 Simulator	(selected)
iOS 13.6 Simulator	3.37 GB
iOS 13.5 Simulator	3.36 GB
iOS 13.4 Simulator	3.36 GB
iOS 13.3 Simulator	3.27 GB
iOS 13.2 Simulator	3.27 GB
iOS 13.1 Simulator	3.24 GB
iOS 13.0 Simulator	3.26 GB
iOS 12.4 Simulator	2.57 GB
iOS 12.2 Simulator	2.54 GB

At the bottom of the window, there is a checkbox labeled 'Check for and install simulator updates automatically' which is checked, and a button labeled 'Check and Install Now'.



MAR 2021



Questions About Part One (installation, device setup, etc.)?





MAR 2021



Part One: App automation workflows for iOS, and Android + adb shell

From your terminal or command-line.

- Go to the root of the Getting-Started-Robotframework-AppiumLibrary-RoboCon-2021 repo.
- Run the following commands first before running any of the examples for Part One.
 - bash ./start-specific-appium-example-workflows-for-workshop.sh All-Appium-Tests-Teardown &&
 - bash ./start-specific-appium-example-workflows-for-workshop.sh Appium-No-Proxy-Test-Setup

After the teardown and setup scripts successfully completed, you should see the Android device that was set up earlier appear on the screen.

- The following will install and run simple Wikipedia app tests on an iOS Simulator.
 - bash ./start-specific-appium-example-workflows-for-workshop.sh Robot-Framework-IOS-App-Tests



Part One: App automation workflows for iOS, and Android + adb shell

The following will install and run simple Wikipedia app tests on an Android device.

- This example will also create a video recording.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_App_Recording_Screen_Capture](#)

The following will run various adb shell commands and native Calculator app tests on an Android device. The commands below come from this comprehensive and useful GitHub list ->

<https://gist.github.com/Pulimet/5013acf2cd5b28e55036c82c91bd56d8>

- This example will check the status of a specific Android app process that is active
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_PS_Command](#)
- This example will check the file system contents for the current directory on the Android device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_LS_Command](#)
- This example will check that a specific package is installed on the Android device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_Dumpsys_Grep_Command](#)
- This example will check detailed information on all installed packages.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_Dumpsys_Package_Command](#)
- This example will check a list of feature information about the Android device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_PM_List_Command](#)
- This example will check the current version of Android OS on the device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests Android_Adb_Build_Version_Command](#)



Part One: App automation workflows for iOS, and Android + adb shell

The following will run various adb shell commands and native Calculator app tests on an Android device. The commands below come from this comprehensive and useful GitHub list -> <https://gist.github.com/Pulimet/5013acf2cd5b28e55036c82c91bd56d8>

- This example will check netstat (network statistics) for TCP connections going through the device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Netstat Command](#)
- This example will display the contacts list.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Contacts List Command](#)
- This example will start and end a phone call on the device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Phone Call Command](#)
- This example will prepare a text message from the device.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Text Message Command](#)
- This example will run logcat to collect the device's system logs in a file.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Logcat Command](#)
- This example will run bugreport to collect the device's system logs, stack traces, and other helpful diagnostic data.
 - [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-Android-Apps-And-Adb-Tests](#) [Android Adb Bugreport Command](#)



Part One: Browser automation for iOS and Android

The following will run Safari and Chrome browser automation examples.

- [bash ./start-specific-appium-example-workflows-for-workshop.sh](#)
[Robot-Framework-IOS-And-Android-Mobile-Browsers-Test-Examples](#)





MAR 2021



Questions About Part One (examples etc.)?





MAR 2021



Part Two: Appium Desktop Walkthrough

There are useful Appium Desktop features that can assist you while you are building out your automation.

- Provides a convenient GUI for Appium Server and easy access to the following.
 - Capability sets
 - Connection options for mobile automation Cloud providers
- Appium Inspector -> gathers details about the UI elements of your app.
- Inspector Recording option generates Appium automation code in Javascript, Python, Ruby, Java, and Robot Framework



The screenshot shows the 'Custom Server' tab of the Appium Desktop interface. It displays 'Desired Capabilities' for a session named 'localhost'. The capabilities include: app (text), platformName (text, iOS), deviceType (text, simulator), deviceName (text, iPad Pro (12.9-inch) (4th generation)), automationName (text, XCUITest). The JSON representation on the right shows the same configuration with additional details like 'deviceOrientation': "Portrait", 'automationName': "XCUITest", and 'platformVersion': "14.4".

The screenshot shows the 'Select Cloud Providers' dialog box. It lists various cloud services: SAUCELABS, headspin, bitbar, Perfecto, TestingBot, ROBOTIC.MOBI, TestObject, BrowserStack, Kobiton, pCloudy, and experitest. The 'TestingBot' provider is selected. At the bottom right are 'Save As...' and 'Start Session' buttons.

The screenshot shows the Appium Inspector interface during a recording session. On the left is a screenshot of a Wikipedia Android app displaying news articles. On the right, the 'Recorder' panel shows the recorded script in Robot Framework. The script includes variables and actions corresponding to the UI elements. The 'Source' tab of the Inspector shows the detailed XML structure of the UI elements being interacted with.

Part Two: Charles Proxy Combined With Robotframework-AppiumLibrary

Charles Proxy is a popular, well documented, and very useful web debugging proxy. [It is not free software and requires a license key](#), but it's very affordable. It has a large list of builtin tools and features.

- <https://www.charlesproxy.com/documentation/proxying/>
- <https://www.charlesproxy.com/documentation/tools/>

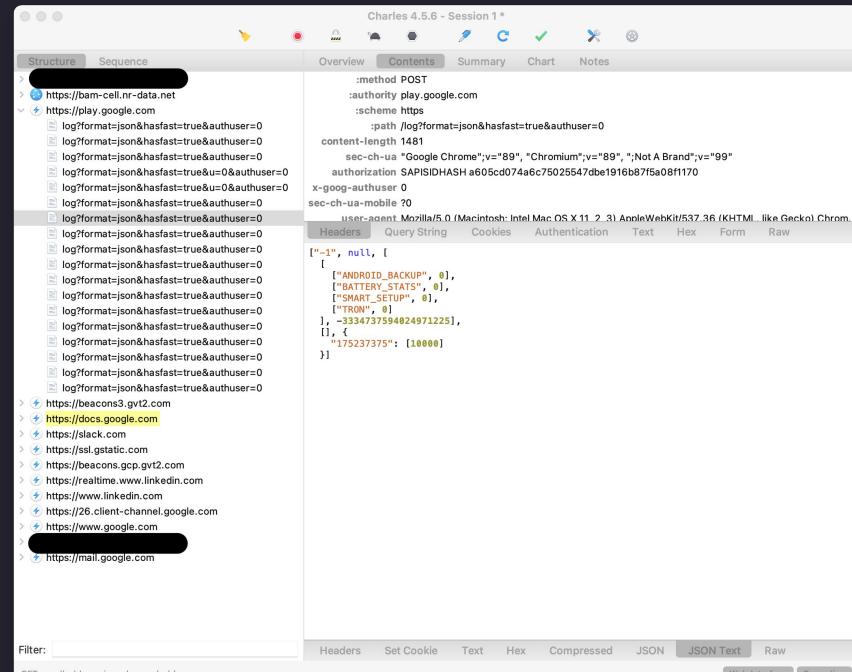
The key features that we will be exploring in this Workshop are the following.

- Headless automation of Charles Proxy from its command-line tools.
- [Capturing and recording all HTTP traffic going into any connected mobile device \(real or Emulator/Simulator\)](#).

The [following example commands requires a properly configured charles.config file](#), a generic file is provided in the Workshop repo.

More information can be found here by searching for "charles.config" -> <https://github.com/jq8481/Tool-Strategies-Lone-Testers-Test-Leadership-Congress-2019#ongoing-work>

- [bash ./start-specific-appium-example-workflows-for-workshop.sh All-Appium-Tests-TearDown &&](#)
- [bash ./start-specific-appium-example-workflows-for-workshop.sh Appium-Charles-Proxy-Test-Setup &&](#)
- [bash ./start-specific-appium-example-workflows-for-workshop.sh Robot-Framework-Charles-Proxy-IOS-And-Android-Mobile-Browsers-Test-Example](#)



Part Two: PaBot, Appium, And Graphwalker Combined With CPU Monitoring

The following examples will run an IOS and Android device in parallel using PaBot, a Robot Framework parallel test runner ->
<https://github.com/mkorpela/pabot>

- On the IOS device, PaBot + Robotframework-AppiumLibrary will run the Wikipedia app in a loop.
- On the Android device PaBot + Robotframework-AppiumLibrary will run two types of tests in parallel.
 - Android Calculator app will be automated with a Model Based Testing approach that uses Graphwalker to generate different sequences of keywords based on a graph.
 - While the Android Graphwalker test is running, PaBot will monitor CPU usage using adb shell and Linux top command.

This command will run the PaBot automation.

- `bash ./start-specific-appium-example-workflows-for-workshop.sh`
`Robot-Framework-Parallel-IOS-Android-Tests`

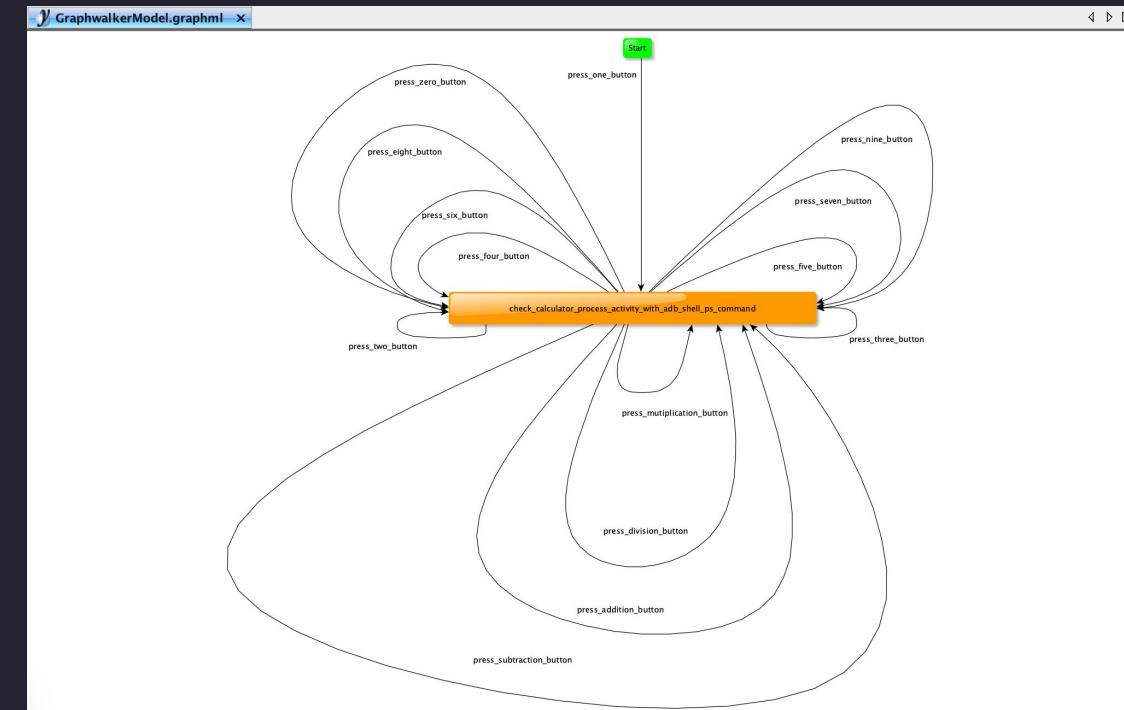
Side Note: This same PaBot Android + Grahpwalker + adb shell CPU monitoring parallel running test can also be performed on a real Android device.

- Take your real Android device and enable the "Developer Options" by following the Android developers website instructions found here -> <https://developer.android.com/studio/debug/dev-options>
- Enable the "USB debugging" option in the Developer Options menu
- Plug in your device and run the "adb device" command from a terminal, then copy the alpha-numeric device name from adb into your `DEVICE_NAME_ANDROID1` environment variable in your .env file

```

Joshuas-MBP:Getting-Started-Robotframework-Appiumlibrary-RoboCon-2021 --bash -- 94x32
Joshuas-MBP:Getting-Started-Robotframework-Appiumlibrary-RoboCon-2021 jgorospe$ adb devices
List of devices attached
ab36bd88          device

Joshuas-MBP:Getting-Started-Robotframework-Appiumlibrary-RoboCon-2021 jgorospe$ 
```



```

1 export PLATFORM_VERSION_IOS="14.4"
2 export PLATFORM_VERSION_ANDROID="8"
3 export PARALLEL_APPium_REMOTE_URL1="http://localhost:4723/wd/hub"
4 export PARALLEL_APPium_REMOTE_URL2="http://localhost:4724/wd/hub"
5 export PARALLEL_APPium_REMOTE_URL3="http://localhost:4725/wd/hub"
6 export PARALLEL_APPium_PORT1=4723
7 export PARALLEL_APPium_PORT2=4724
8 export PARALLEL_APPium_PORT3=4725
9 export PARALLEL_APPium_WDALOCALPORT1=8100
10 export PARALLEL_APPium_WDALOCALPORT2=8200
11 export PARALLEL_APPium_WDALOCALPORT3=8300
12 export DEVICE_NAME_ANDROID1="ab36bd88" *
13 export DEVICE_NAME_ANDROID2="Nexus_5_API_26"
14 export DEVICE_NAME_ANDROID3="Pixel_XL_API_28"
15 export DEVICE_NAME_IOS1="iPad Pro (12.9-inch) (4th generation)"
16 export DEVICE_NAME_IOS2="iPhone 12 Pro"
17 export DEVICE_NAME_IOS3="iPhone 8"
18 export APP_URL="http://nodegoat.herokuapp.com/login"
19 export ANDROID_NATIVE_DEFAULT_APP_ACTIVITY="com.android.calculator2.Calculator"
export ANDROID_NATIVE_DEFAULT_APP_PACKAGE="com.android.calculator2"
20 export ANDROID_APP_NAME="Wikipedia.apk"
21 export IOS_APP_NAME="Wikipedia.app"
22 
```





MAR 2021



Questions About Part Two (examples etc.)?



Workshop Complete...

- Feel free to reach out to me on LinkedIn or Twitter with any high-level questions.
- I'm also open to presenting this live workshop again (not pre-recorded) through your Meetup Group.
- For deeper-dives into this content, I'm also open to perform live/pre-recorded small-scale online half-day workshops services for your team members or organization.

That's
ROBOCON 2021



#ROBOCON20



Robot
Framework
Foundation