Mobile App Automation Testing using Appium

Date: 17 July 2020

Senthilmurugan S

Agenda

- Overview – Mobile Application Testing
- Tool Comparison – Mobile Application
- Appium -Overview
- Appium-Supports
- Appium Architecture
- Appium Block Diagram
- Setup and Configuration (Android)
- UIAutomator viewer
- Appium Demo

Overview-Mobile Application Testing

Testing Mobile Applications is

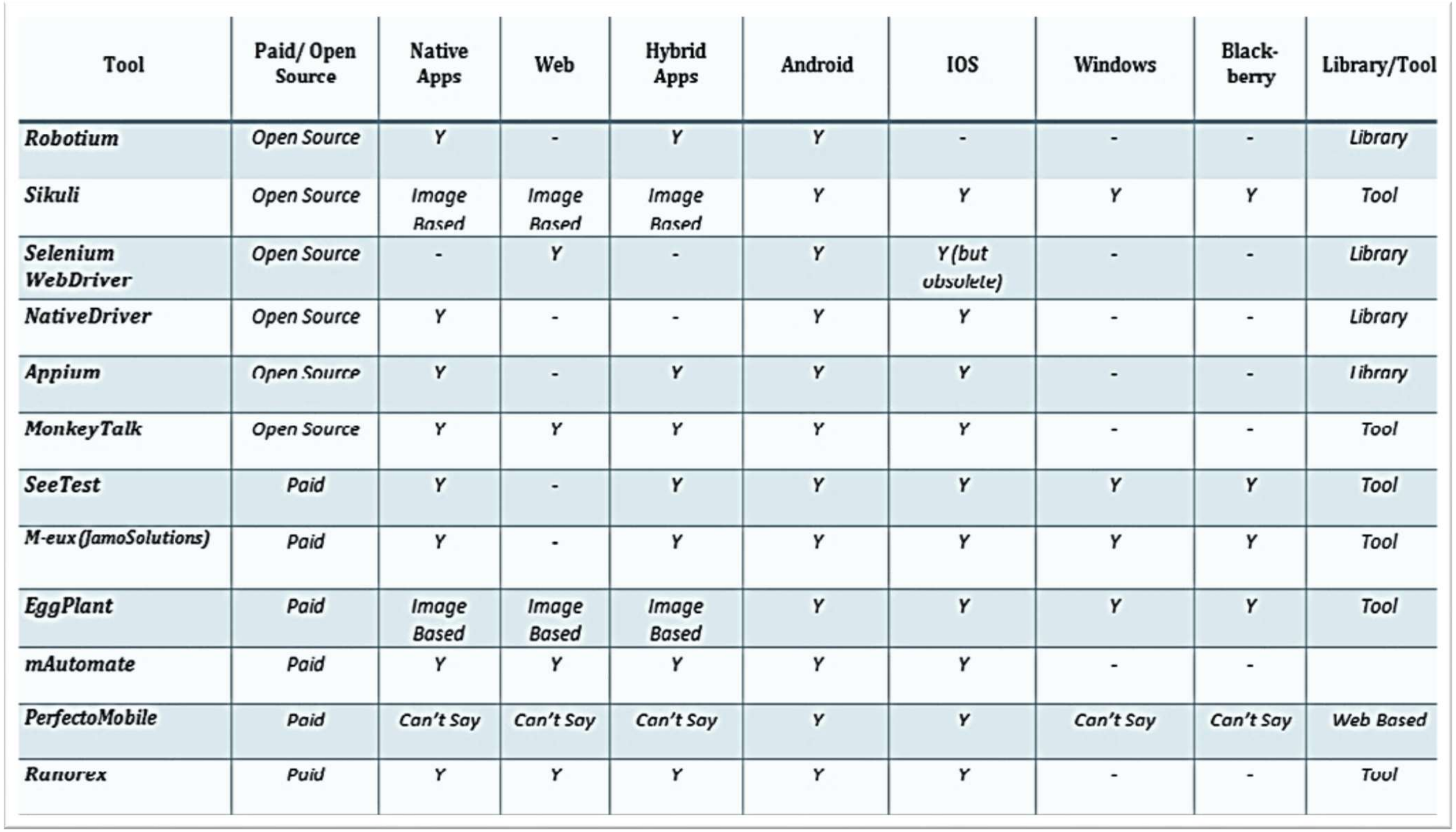
- More complex
- Time consuming
- Platform Variations
- Quality Concerns
- So, alike Manual Mobile Testing process,
- we should also adopt

Mobile Automation Testing

Why Automation for Mobile Application ?

- increased testing efficiency,
- increased testing coverage,
- faster time to market.

Tools Comparison - Mobile Application



Tool	Paid/ Open Source	Native Apps	Web	Hybrid Apps	Android	IOS	Windows	Black-berry	Library/Tool
Robotium	*Open Source*	*Y*	-	*Y*	*Y*	-	-	-	*Library*
Sikuli	*Open Source*	*Image Based*	*Image Based*	*Image Based*	*Y*	*Y*	*Y*	*Y*	*Tool*
Selenium WebDriver	*Open Source*	-	*Y*	-	*Y*	*Y (but obsolete)*	-	-	*Library*
NativeDriver	*Open Source*	*Y*	-	-	*Y*	*Y*	-	-	*Library*
Appium	*Open Source*	*Y*	-	*Y*	*Y*	*Y*	-	-	*Library*
MonkeyTalk	*Open Source*	*Y*	*Y*	*Y*	*Y*	*Y*	-	-	*Tool*
SeeTest	*Paid*	*Y*	-	*Y*	*Y*	*Y*	*Y*	*Y*	*Tool*
M-eux (JamoSolutions)	*Paid*	*Y*	-	*Y*	*Y*	*Y*	*Y*	*Y*	*Tool*
EggPlant	*Paid*	*Image Based*	*Image Based*	*Image Based*	*Y*	*Y*	*Y*	*Y*	*Tool*
mAutomate	*Paid*	*Y*	*Y*	*Y*	*Y*	*Y*	-	-	
PerfectoMobile	*Paid*	*Can't Say*	*Can't Say*	*Can't Say*	*Y*	*Y*	*Can't Say*	*Can't Say*	*Web Based*
Ranorex	*Paid*	*Y*	*Y*	*Y*	*Y*	*Y*	-	-	*Tool*

Overview

appium

Important : Cross-Platform Supporting

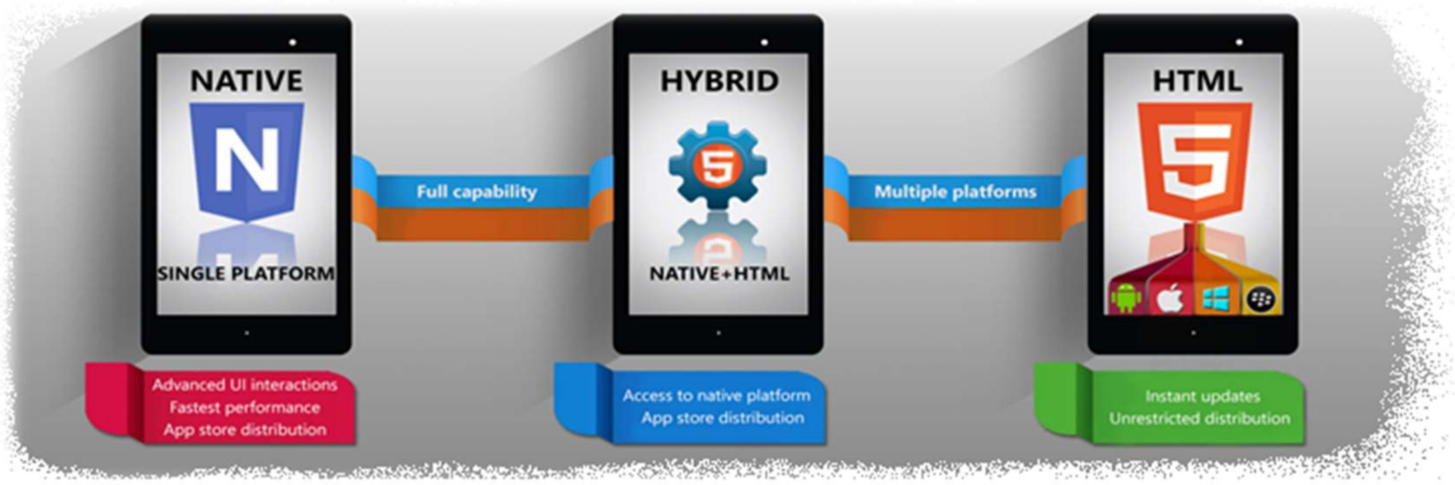
Android & IOS

What is Appium?

&

Why Appium?

Appium is an open-source test automation tool Allows testing for all types of Mobile Applications: Native Apps, Hybrid Apps and Mobile Web Apps



Appium Desktop

Appium Desktop is a new open source GUI application for Windows, Mac, and Linux which gives you the power of the Appium automation server in a more organized manner with a flexible UI.

Appium Desktop is a combination of two essential components of Appium:

Appium Server: Server instance for enabling testing (and test automation) of apps.

Appium Inspector: For inspecting and getting all the details of UI elements of your apps.

Enables users to work with Appium on desktop

Provides Inspector for better analysis of apps

Provides the ability to switch between web-view and native app view from inspector

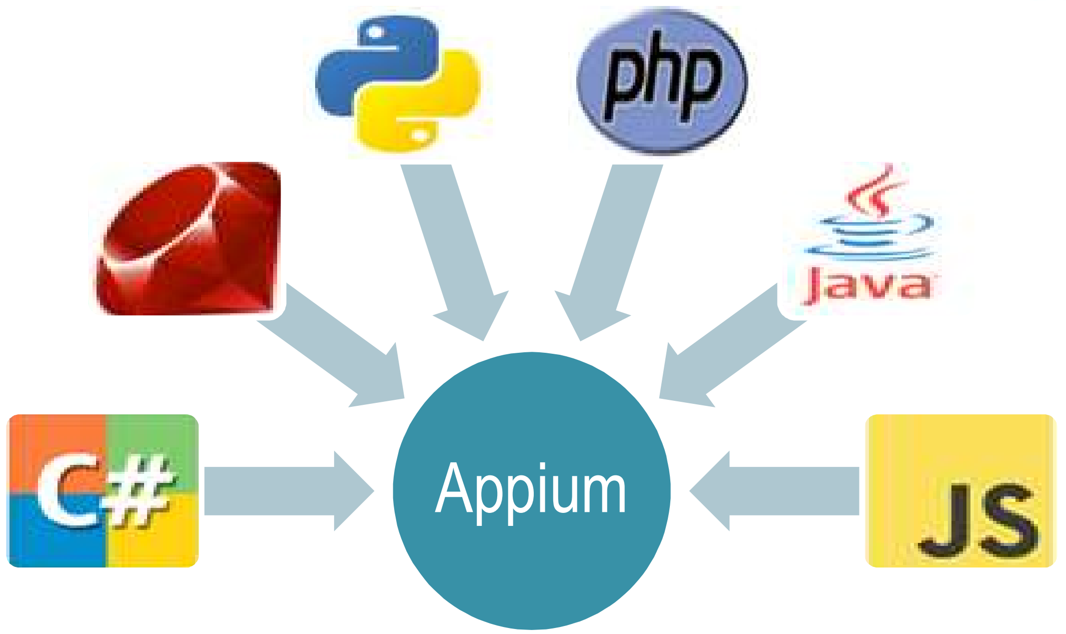
Enables more element access and context handling

Provides an action recorder and code generator

Allows using desired capabilities and presets configuration for convenient use

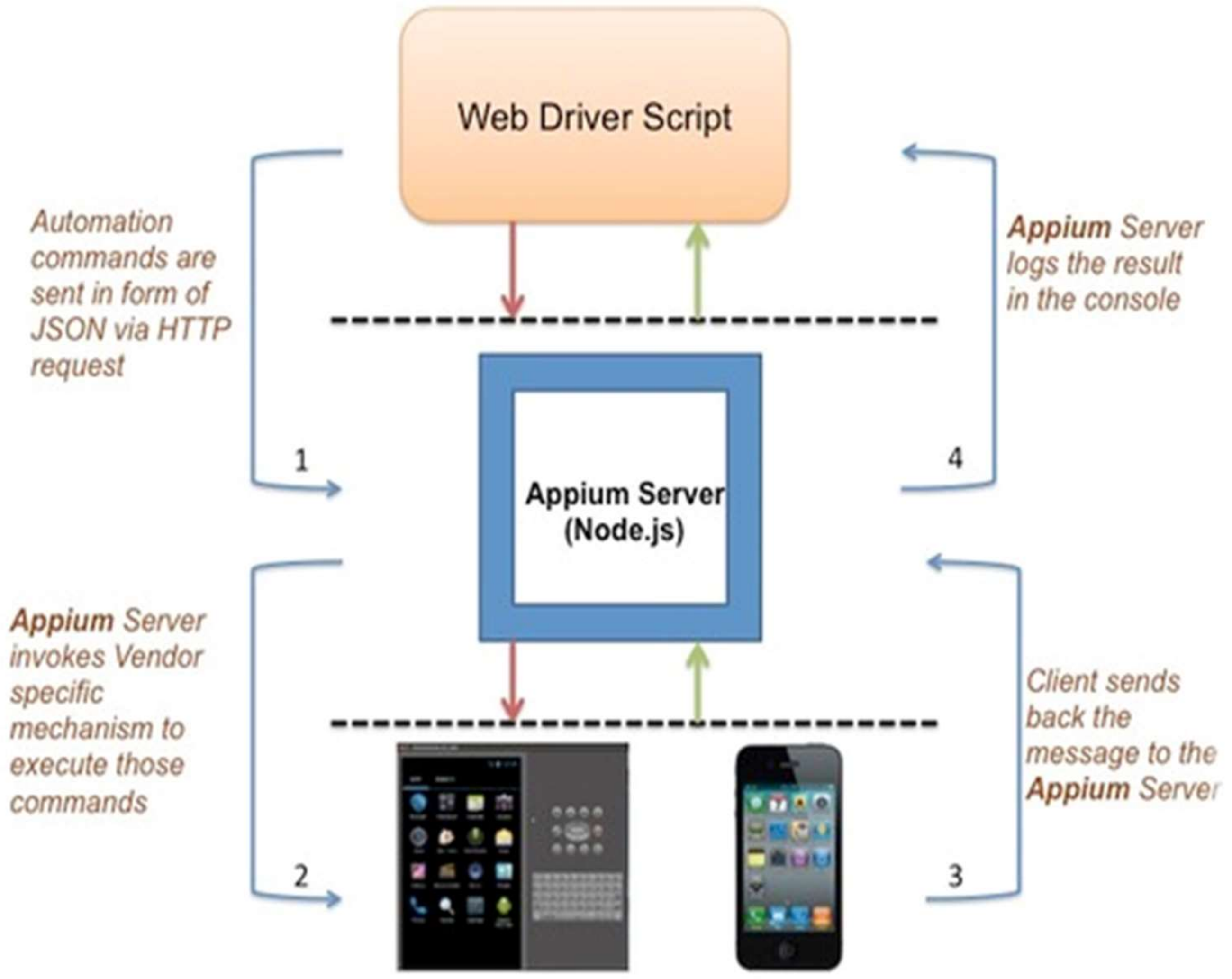
Provides enhancement in the test script build

Appium – Language Supports



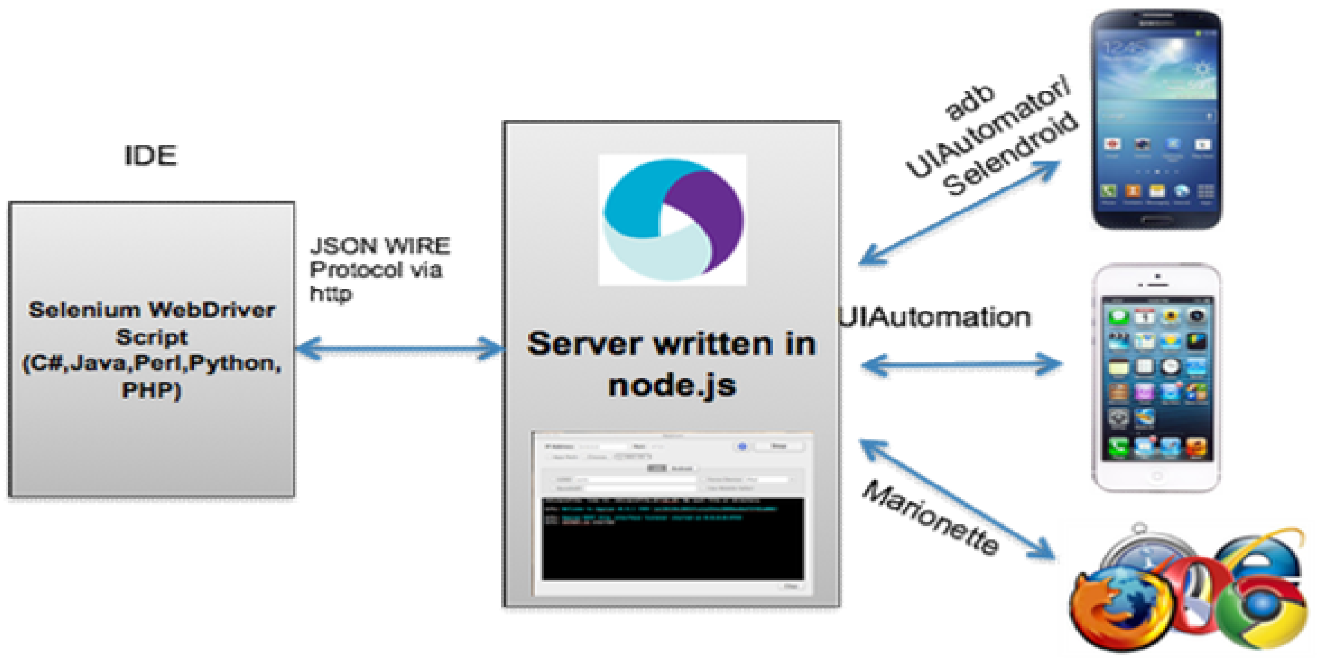
Appium allows you to run on emulators and simulators

Appium Architecture


Web Driver Script
Automation commands are sent in form of JSON via HTTP request
1
Appium Server logs the result in the console
4
Appium Server (Node.js)
Appium Server invokes Vendor specific mechanism to execute those commands
2
Client sends back the message to the Appium Server
3

Block Diagram

1. Receives connection from client
2. Listen command
3. Execute command
4. Respond back the command execution status


IDE
Selenium WebDriver Script (C#,Java,Perl,Python, PHP)
JSON WIRE Protocol via http
Server written in node.js
adb UIAutomator/ Selendroid
UIAutomation
Marionette

Pre-requisite to use Appium

Pre-requisite to use Appium :

- ANDROID SDK
- JDK (Java Development Kit)
- TestNG
- Eclipse
- Selenium Server JAR
- APPIUM For Windows
- APK App Info On Google Play
- Node.js (Not Required - Whenever Appium server is installed, it by default comes with "Node.exe" & NPM. It's included in Current version of Appium.)

- Environment Variables & Path Settings
  - JAVA_HOME
  - ANDROID_HOME

Setup and Configuration

- Download and Configure Android SDK Bundle
- API level and Version – Android SDK
- API level and Android version supported by Appium
- Getting correct API level for Android Phone
- Enabling Developer mode in Android
- To verify connection connect with DDMS /ADB
- Install Node.js
- Install Microsoft .NET framework 4.5
- Download and Install Appium (Windows)

From Source:

- Install Node.js
- Download Appium or Clone it using GitHub
- In cmd navigate to node_modules/appium/bin
- Run 'node Appium [server arguments]'

From GUI Interface:

- Do the configurations as needed from GUI
- Click launch button to launch Appium server

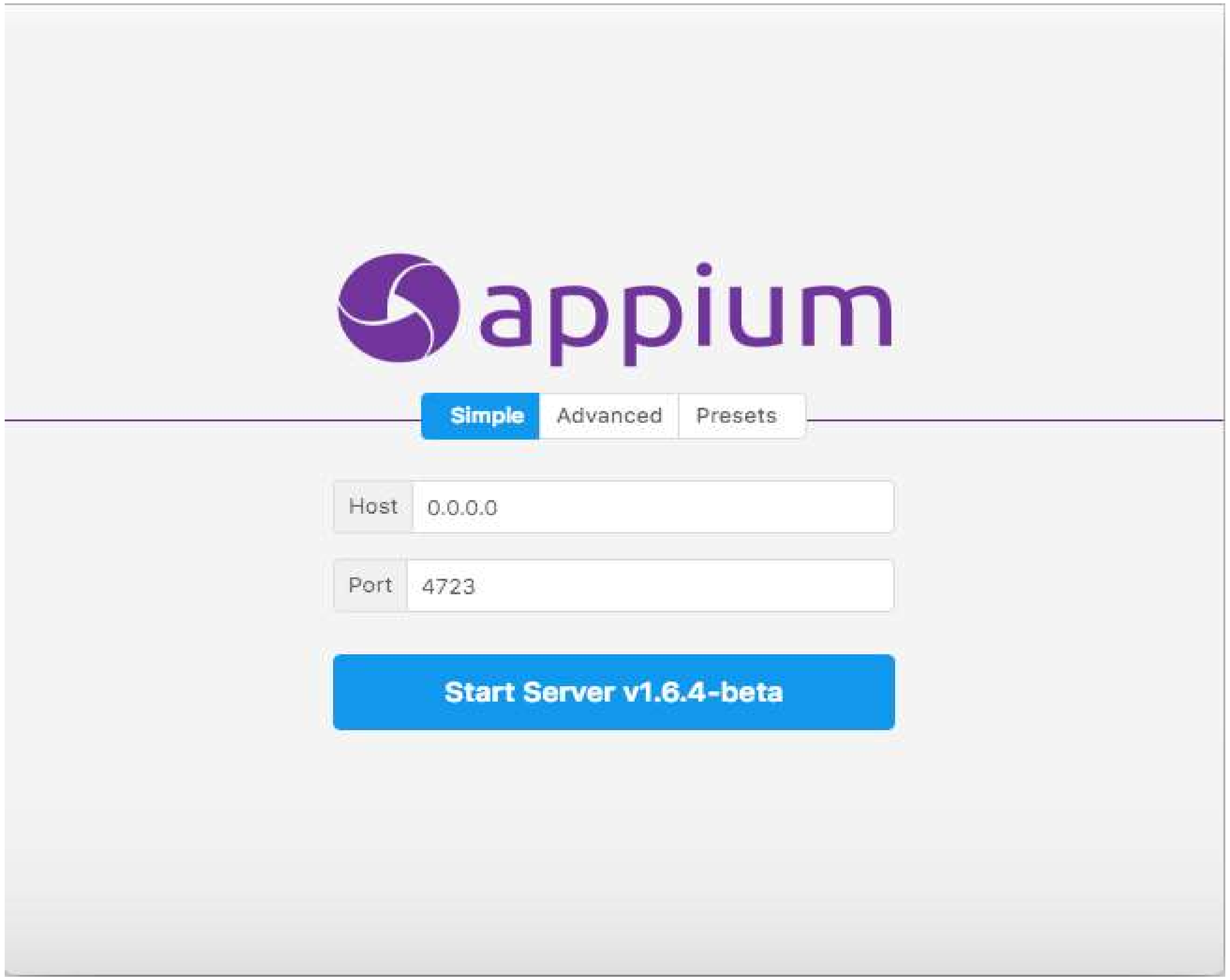
Appium Server Arguments

Usage : node appium [arguments](windows)

--app	: To specify the path (iOS: .app, android: apk)
-U , --udid	: Unique device identifier of the connected physical device
-a, --address	: IP Address to listen on
-p, --port	: port to listen on
--session-override	: Enables session override
--full-reset	: (Android) Reset app state by uninstalling app instead of clearing app data. On Android, this will also remove the app after the session is complete.
--no-reset	: Don't reset app state between sessions
-l, --pre-launch	: Pre-launch the application before allowing the first session

Complete List : https://github.com/appium/appium/blob/master/docs/en/server-args.md

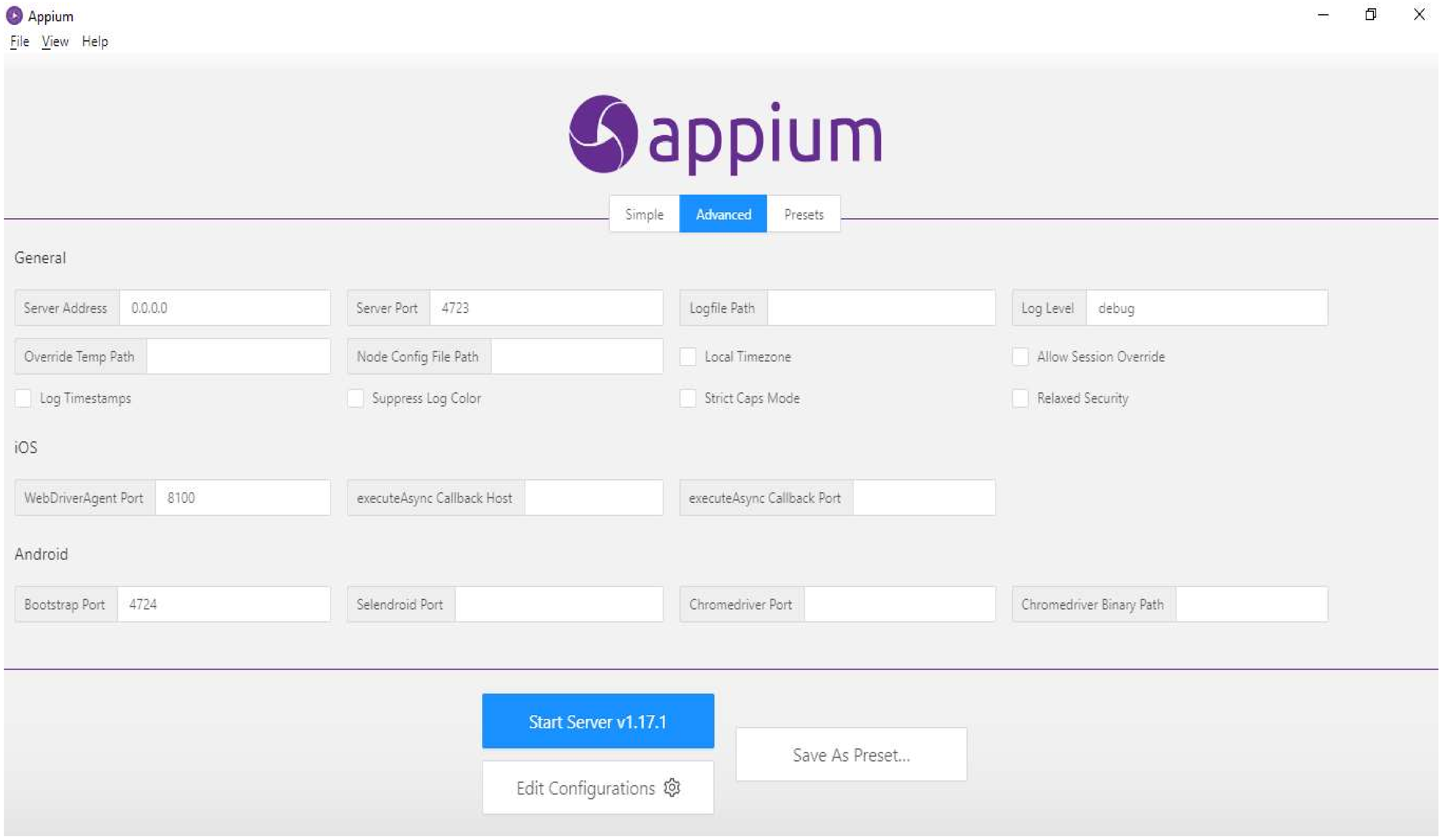
Appium Screenshot



Host 0.0.0.0

Port 4723

Appium Screenshot


Appium

File View Help

Simple | Advanced | Presets

General

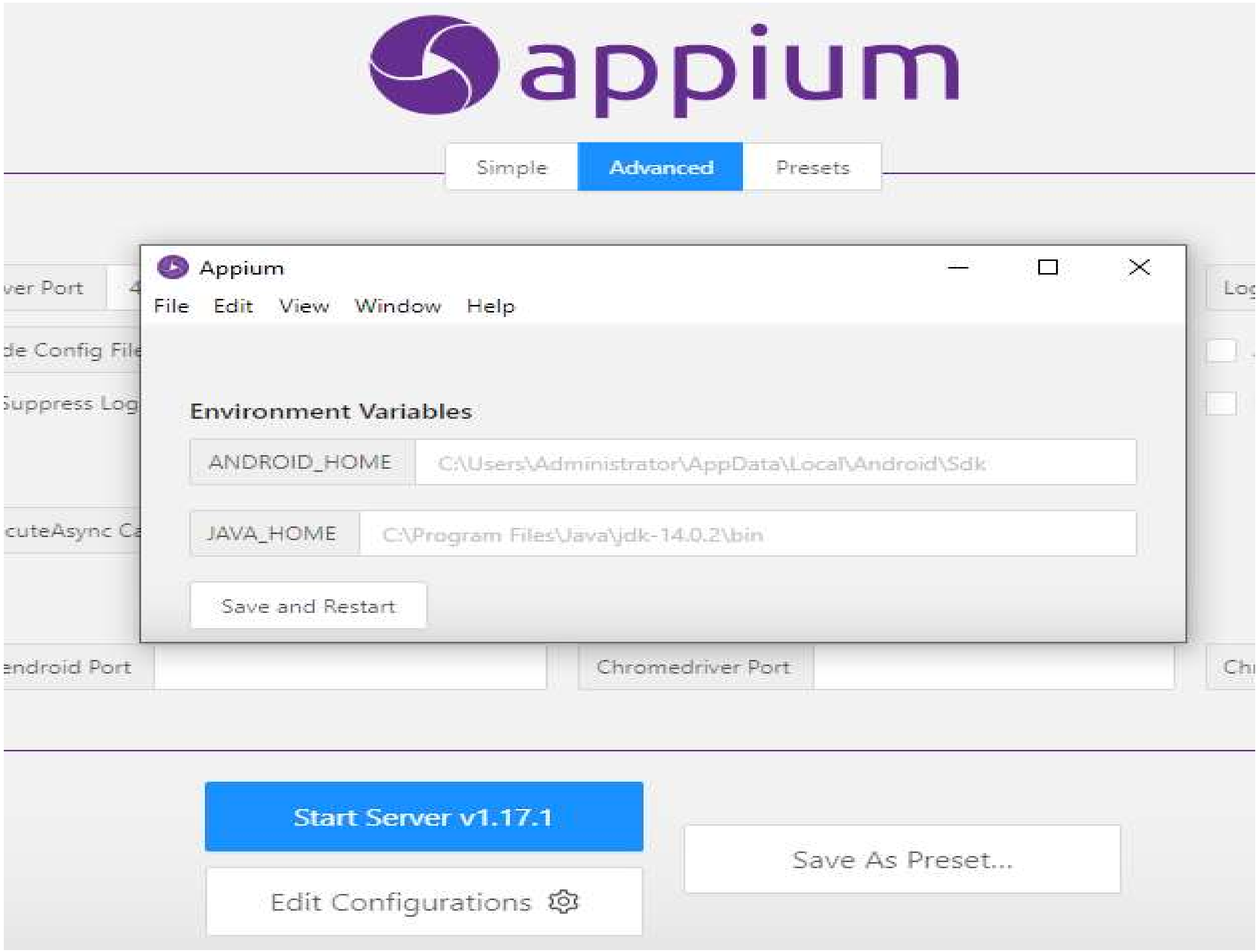
Server Address	0.0.0.0	Server Port	4723	Logfile Path		Log Level	debug
Override Temp Path		Node Config File Path		☐ Local Timezone		☐ Allow Session Override	
☐ Log Timestamps		☐ Suppress Log Color		☐ Strict Caps Mode		☐ Relaxed Security	

iOS

WebDriverAgent Port	8100	executeAsync Callback Host		executeAsync Callback Port	

Android

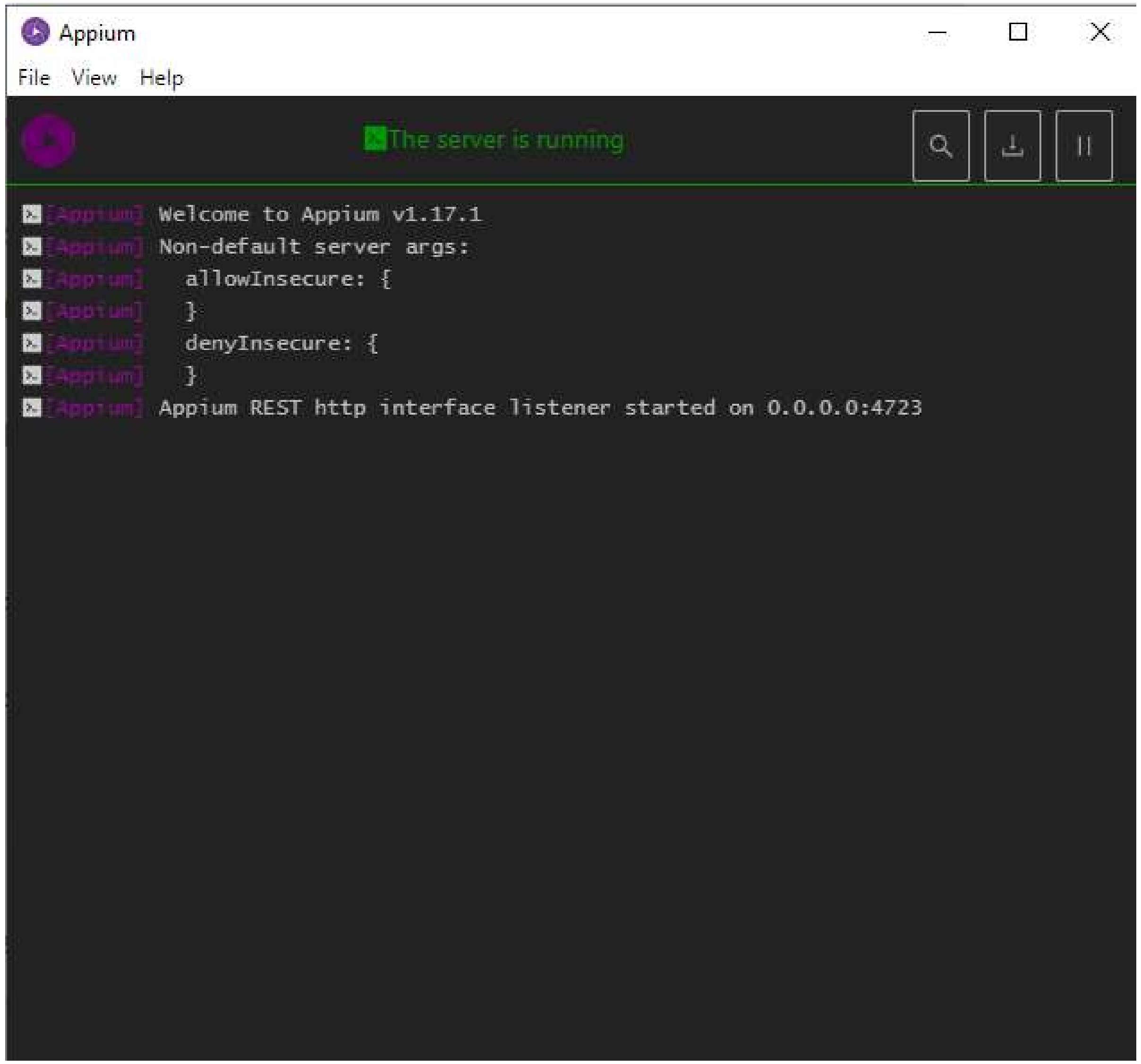
Bootstrap Port	4724	Selendroid Port		Chromedriver Port		Chromedriver Binary Path	



Environment Variables

ANDROID_HOME	C:\Users\Administrator\AppData\Local\Android\Sdk
JAVA_HOME	C:\Program Files\Java\jdk-14.0.2\bin

Appium Screenshot



```
[Appium] Welcome to Appium v1.17.1
[Appium] Non-default server args:
[Appium]   allowInsecure: {
[Appium]   }
[Appium]   denyInsecure: {
[Appium]   }
[Appium] Appium REST http interface listener started on 0.0.0.0:4723
```

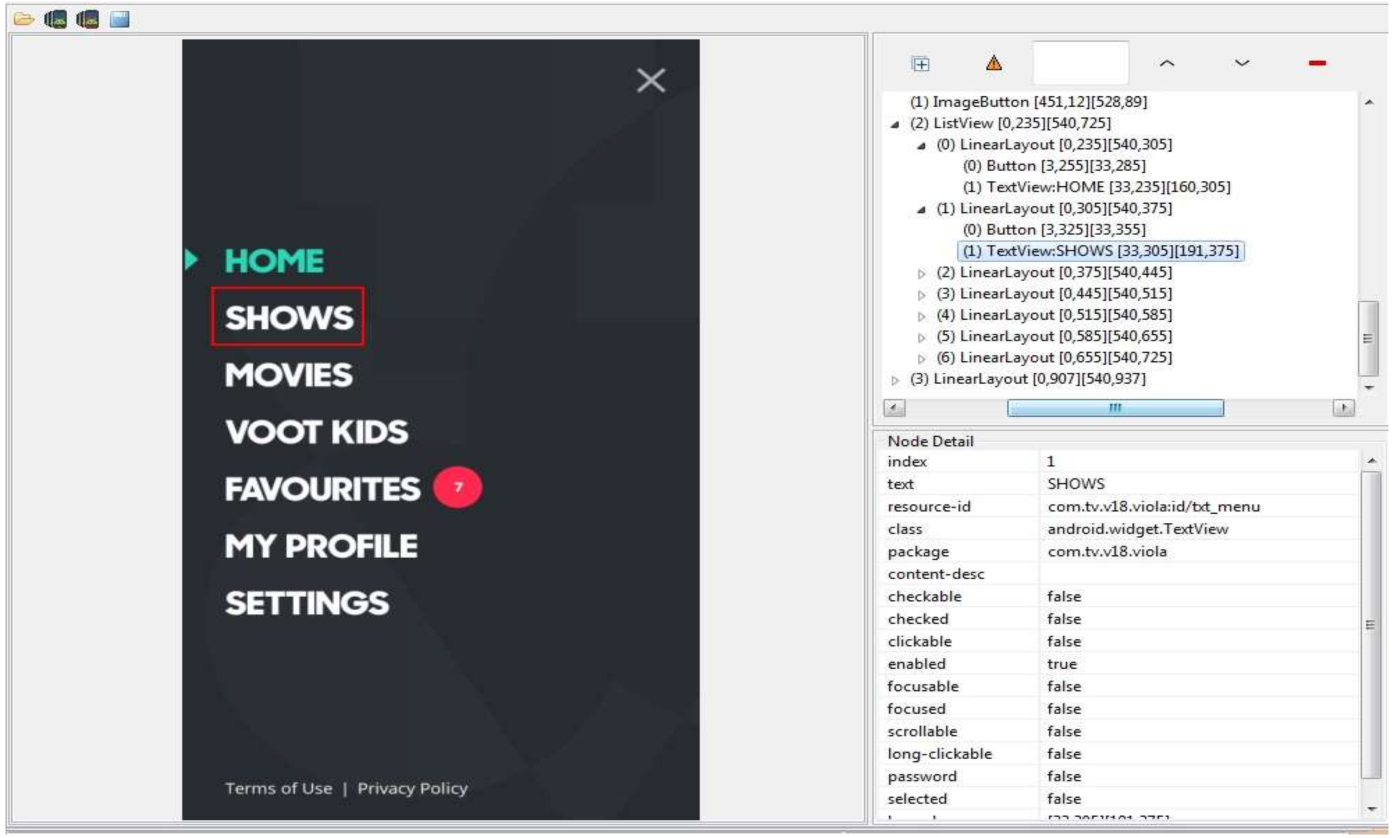
Automating Android Apps

appium

```
import os
import unittest
from appium import webdriver
from time import sleep

class ChessAndroidTests(unittest.TestCase):
    "Class to run tests against the Chess Free app"
    def setUp(self):
        "Setup for the test"
        desired_caps = {}
        desired_caps['platformName'] = 'Android'
        desired_caps['platformVersion'] = '8.0'
        desired_caps['deviceName'] = 'Pixel'
        # Returns abs path relative to this file and not cwd
        desired_caps['app'] = os.path.abspath(os.path.join(os.path.dirname(__file__),'apps/Chess Free.apk'))
        desired_caps['appPackage'] = 'uk.co.aifactory.chessfree'
        desired_caps['appActivity'] = '.ChessFreeActivity'
        self.driver = webdriver.Remote('http://localhost:4723/wd/hub', desired_caps)

    def tearDown(self):
        "Tear down the test"
        self.driver.quit()

    def test_single_player_mode(self):
        "Test the Chess app launches correctly and click on Play button"
        element = self.driver.find_element_by_id("uk.co.aifactory.chessfree:id/ButtonPlay")
        element.click()
        sleep(5)

#---START OF SCRIPT
if __name__ == '__main__':
    suite = unittest.TestLoader().loadTestsFromTestCase(ChessAndroidTests)
    unittest.TextTestRunner(verbosity=2).run(suite)
```

UIAutomator viewer

The UI Automator viewer tool provides a convenient visual interface to Inspect the layout hierarchy and view the properties of UI components that are visible on the foreground of the device

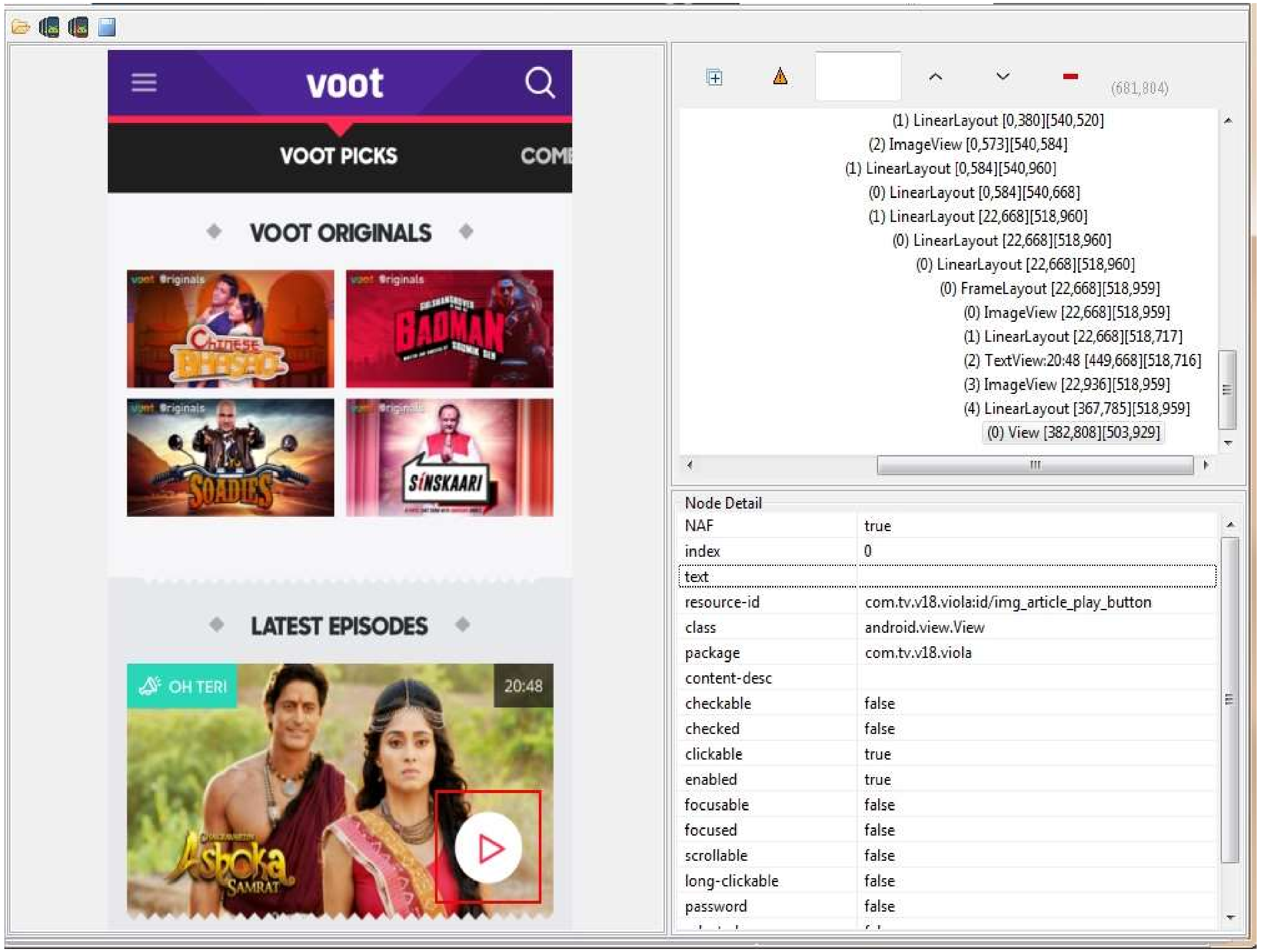


```
(1) ImageButton [451,12][528,89]
(2) ListView [0,235][540,725]
    (0) LinearLayout [0,235][540,305]
        (0) Button [3,255][33,285]
        (1) TextView:HOME [33,235][160,305]
    (1) LinearLayout [0,305][540,375]
        (0) Button [3,325][33,355]
        (1) TextView:SHOWS [33,305][191,375]
    (2) LinearLayout [0,375][540,445]
    (3) LinearLayout [0,445][540,515]
    (4) LinearLayout [0,515][540,585]
    (5) LinearLayout [0,585][540,655]
    (6) LinearLayout [0,655][540,725]
(3) LinearLayout [0,907][540,937]
```

Node Detail

index	1
text	SHOWS
resource-id	com.tv.v18.viola:id/txt_menu
class	android.widget.TextView
package	com.tv.v18.viola
content-desc	
checkable	false
checked	false
clickable	false
enabled	true
focusable	false
focused	false
scrollable	false
long-clickable	false
password	false
selected	false

UIAutomator viewer

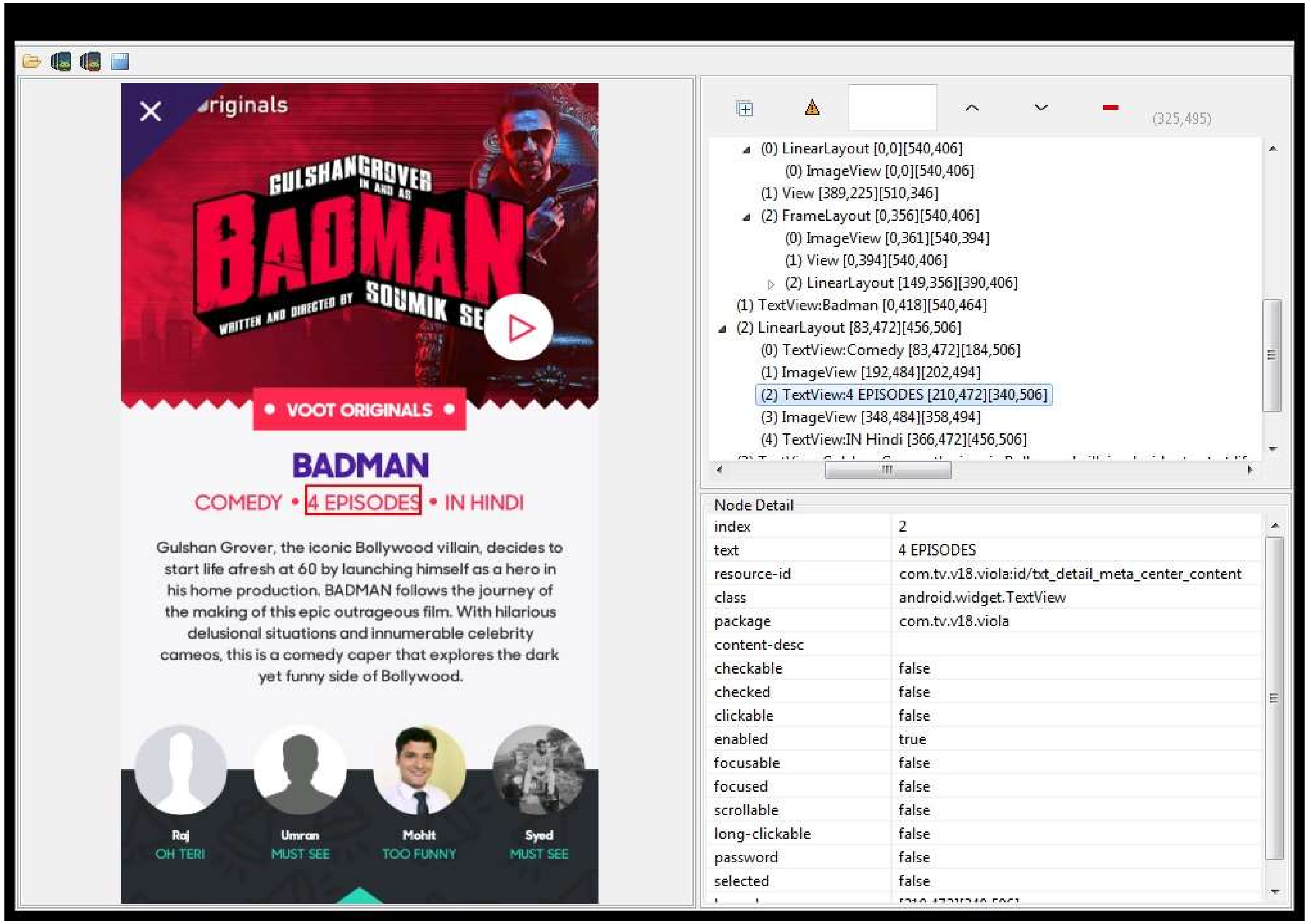


```
        (1) LinearLayout [0,380][540,520]
      (2) ImageView [0,573][540,584]
    (1) LinearLayout [0,584][540,960]
      (0) LinearLayout [0,584][540,668]
      (1) LinearLayout [22,668][518,960]
        (0) LinearLayout [22,668][518,960]
          (0) LinearLayout [22,668][518,960]
            (0) FrameLayout [22,668][518,959]
              (0) ImageView [22,668][518,959]
              (1) LinearLayout [22,668][518,717]
              (2) TextView:20:48 [449,668][518,716]
              (3) ImageView [22,936][518,959]
              (4) LinearLayout [367,785][518,959]
                (0) View [382,808][503,929]
```

Node Detail

NAF	true
index	0
text	
resource-id	com.tv.v18.viola:id/img_article_play_button
class	android.view.View
package	com.tv.v18.viola
content-desc	
checkable	false
checked	false
clickable	true
enabled	true
focusable	false
focused	false
scrollable	false
long-clickable	false
password	false

UIAutomator viewer



Gulshan Grover, the iconic Bollywood villain, decides to start life afresh at 60 by launching himself as a hero in his home production. BADMAN follows the journey of the making of this epic outrageous film. With hilarious delusional situations and innumerable celebrity cameos, this is a comedy caper that explores the dark yet funny side of Bollywood.

```
    (0) LinearLayout [0,0][540,406]
        (0) ImageView [0,0][540,406]
    (1) View [389,225][510,346]
    (2) FrameLayout [0,356][540,406]
        (0) ImageView [0,361][540,394]
        (1) View [0,394][540,406]
        (2) LinearLayout [149,356][390,406]
(1) TextView:Badman [0,418][540,464]
(2) LinearLayout [83,472][456,506]
    (0) TextView:Comedy [83,472][184,506]
    (1) ImageView [192,484][202,494]
    (2) TextView:4 EPISODES [210,472][340,506]
    (3) ImageView [348,484][358,494]
    (4) TextView:IN Hindi [366,472][456,506]
```

Node Detail

index	2
text	4 EPISODES
resource-id	com.tv.v18.viola:id/txt_detail_meta_center_content
class	android.widget.TextView
package	com.tv.v18.viola
content-desc	
checkable	false
checked	false
clickable	false
enabled	true
focusable	false
focused	false
scrollable	false
long-clickable	false
password	false
selected	false

Appium Locators

- Finding elements by ID
    - Finding elements by ID (resource-id)
- Finding elements by name
    - Finding elements by name (text)
- Finding elements by className
    - Finding elements by className (class)
- Finding elements by AccessibilityId
    - Finding elements by AccessibilityId
- Finding elements by Xpath

Demo

Appium Demo

Thank you

TATA ELXSI

ITPB Road Whitefield

Bangalore 560 048 India
Tel +91 80 2297 9123
Fax +91 80 2841 1474
e-mail info@tataelxsi.com

www.tataelxsi.com