

Mobile Testing with Robot Framework

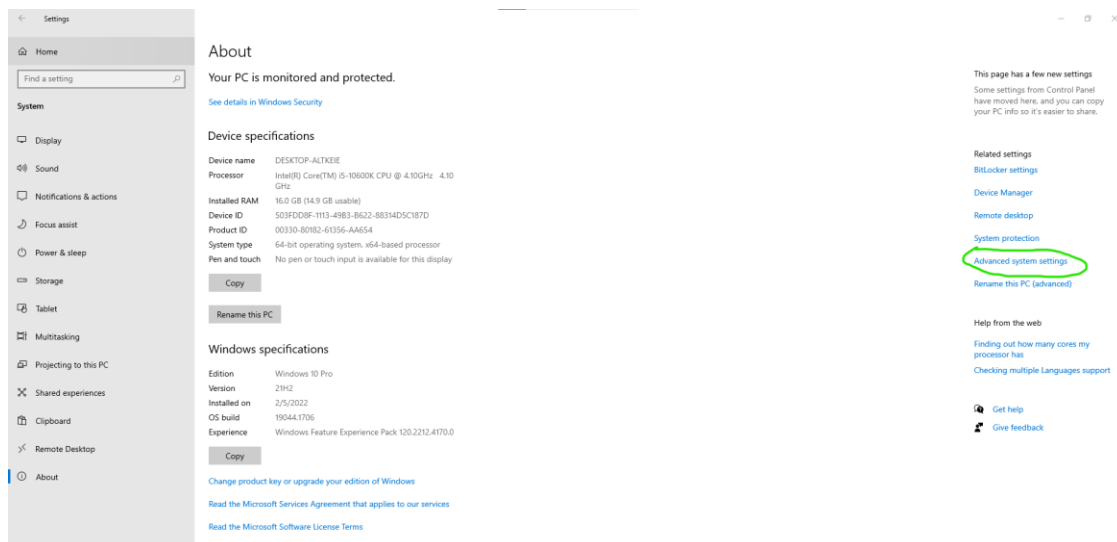
Part 1: Installation

Note for JDK and Android Studio installation, can refer to

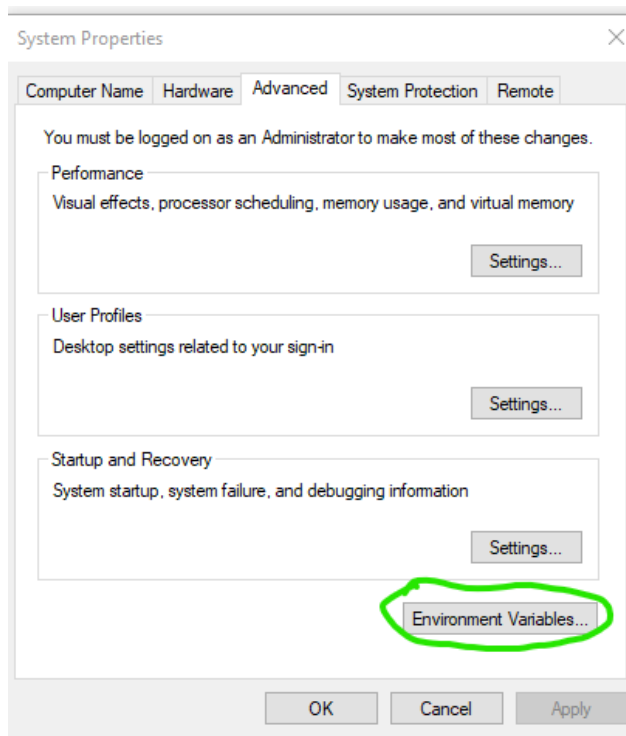
<https://www.youtube.com/watch?v=W5hcHbzTjOc&list=PL4GZKvvcjS3vAPWLqWbKZogkL5cD71yrT&index=3>

A. Install Java SE Development Kit (JDK)

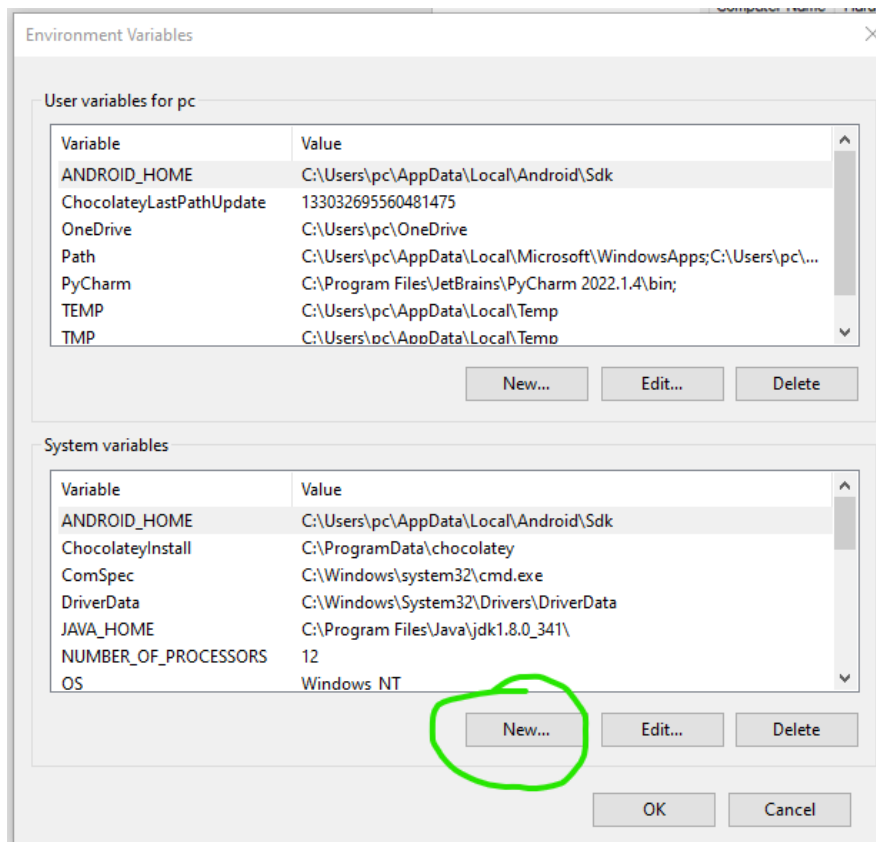
1. Download Java SE Development Kit 8u202 from:
<https://www.oracle.com/java/technologies/javase/javase8-archive-downloads.html>.
Where you want to download, website will ask you to login to Oracle or sign up if you don't have Oracle account.
2. Run JDK installation file. Just follow instructions. You need to remember the folder location where JDK is installed.
3. After that, go to:
Start -> Setting (Gear icon above Power icon) -> System -> About (On left side and below) -> Advanced System Settings (as shown in figure below)



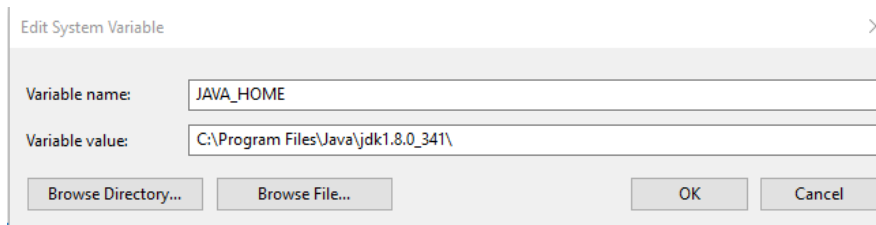
4. Advanced System Settings window be opened. Click 'Environment Variable ...'



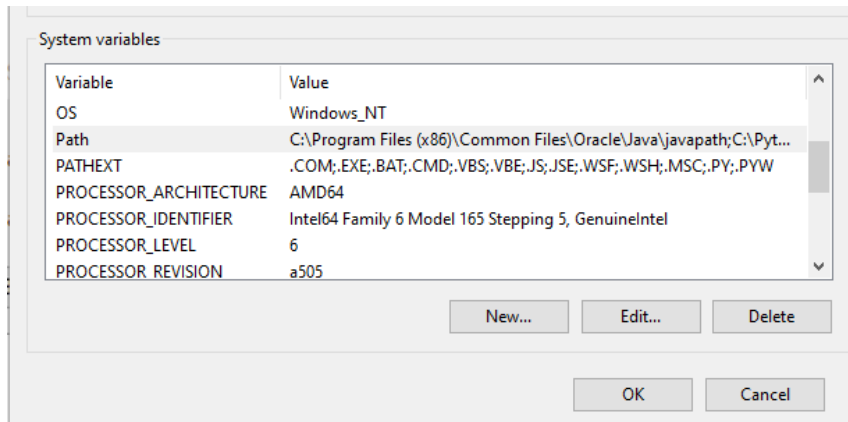
5. Click 'New' under System Variable



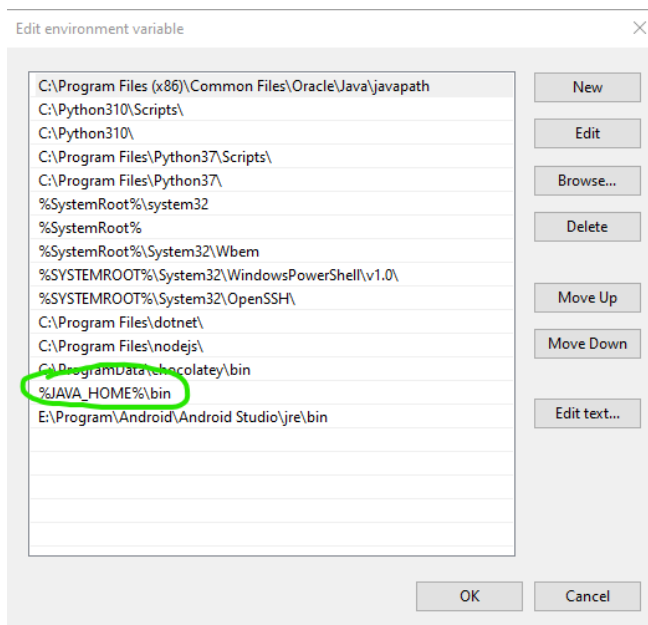
6. Write **JAVA_HOME** on 'Variable name:' and write location of JDK on 'Variable value:'. Then click OK.



7. Click 'Path'



8. Click 'New' and write %JAVA_HOME%\bin. Then click Ok

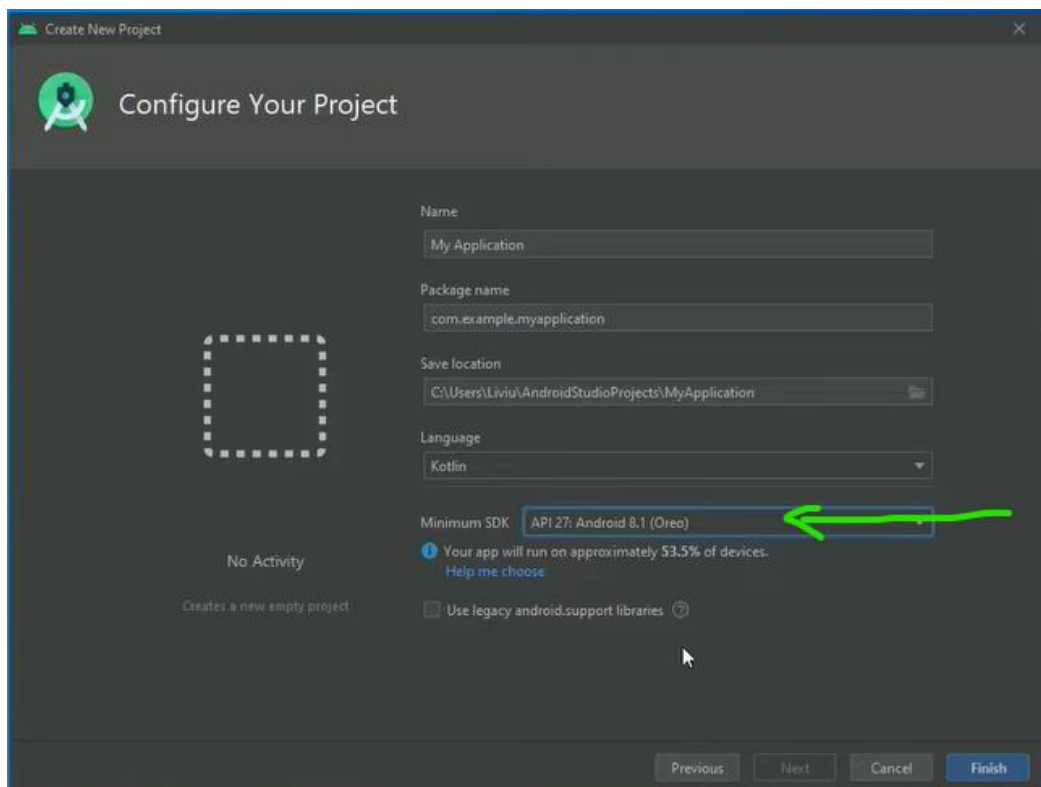


9. Don't forget to click OK on Environment Variable window and then click OK on Advanced System Setting window.

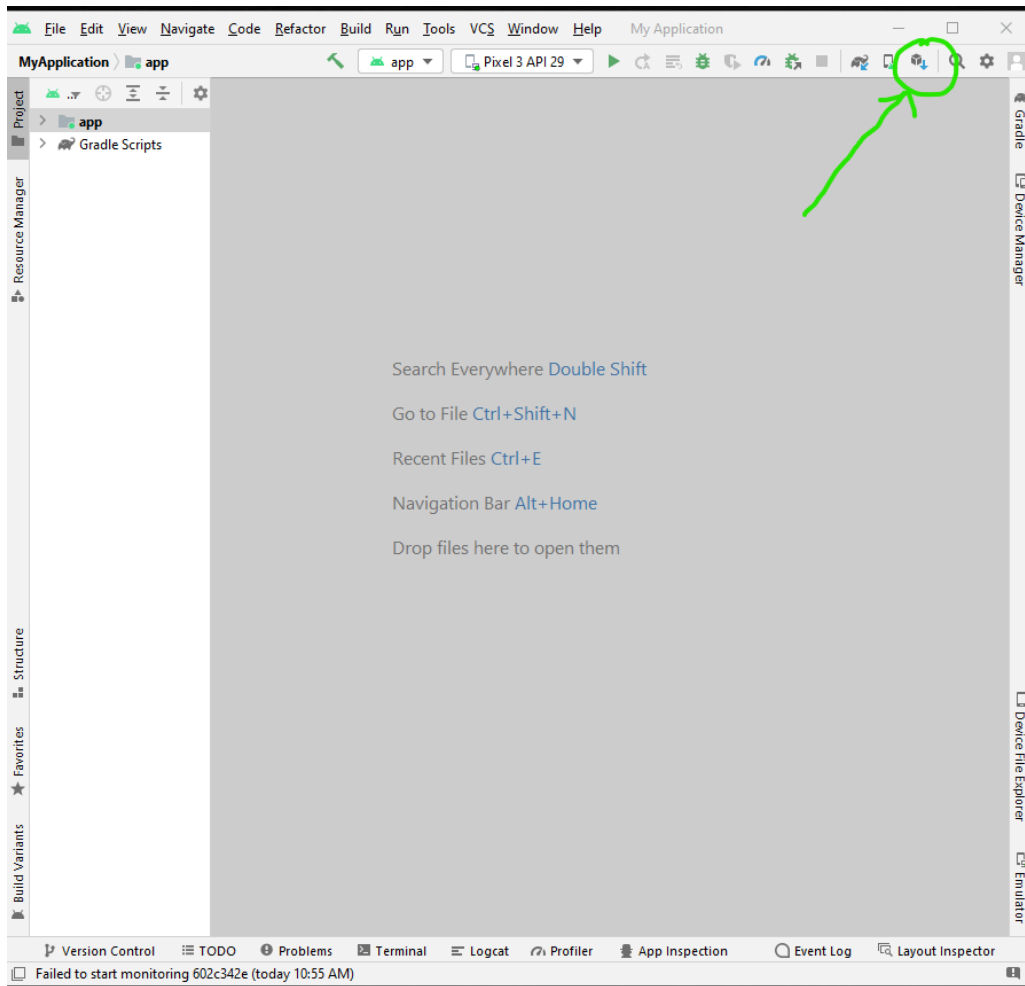
B. Android Studio

Note: Install Android Studio is recommended because it can install Android SDK (important for mobile testing) and install Android smartphone emulator.

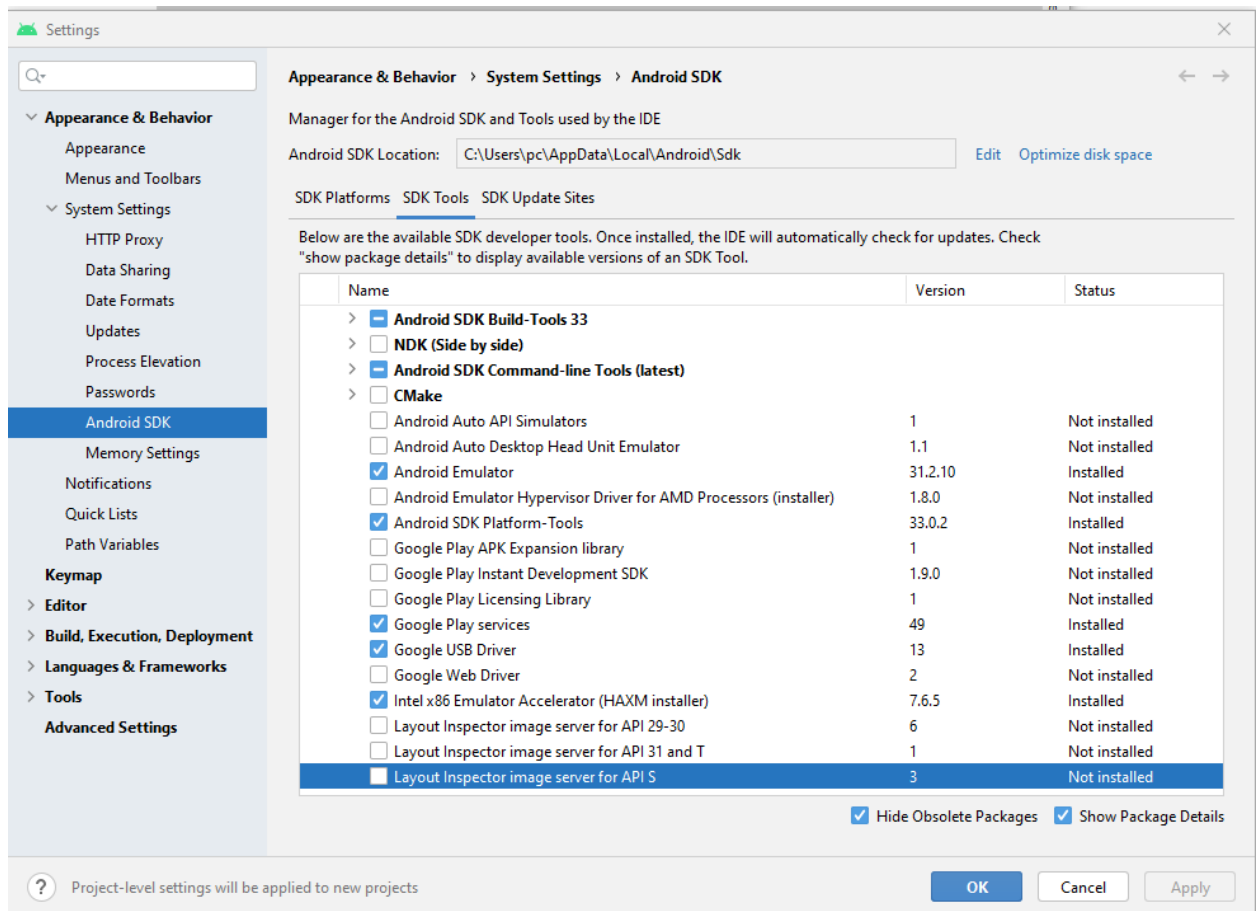
1. Go to <https://developer.android.com/studio> to download
2. Run this installation and just follow the instructions and use Standard.
3. After finish installation, open Android Studio.
4. On front menu, click 'Start a new Android Studio Project'
5. On creating menu, go to Minimum SDK and choose API 27: Android 8.1 (Oreo). Then click finish.



6. After project is created, click SDK Manager (top and right) as shown in figure below:

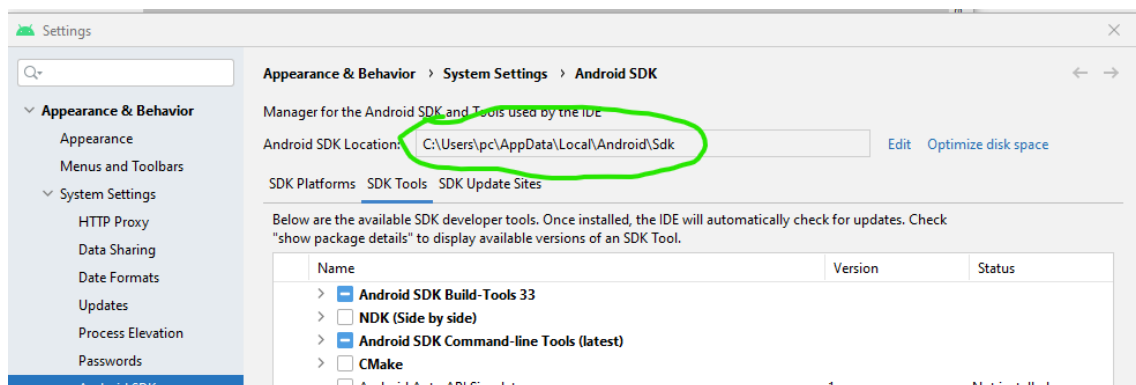


- Go to Android SDK section (on left site). Go to SDK Tools and tick tools as shown in figure below.



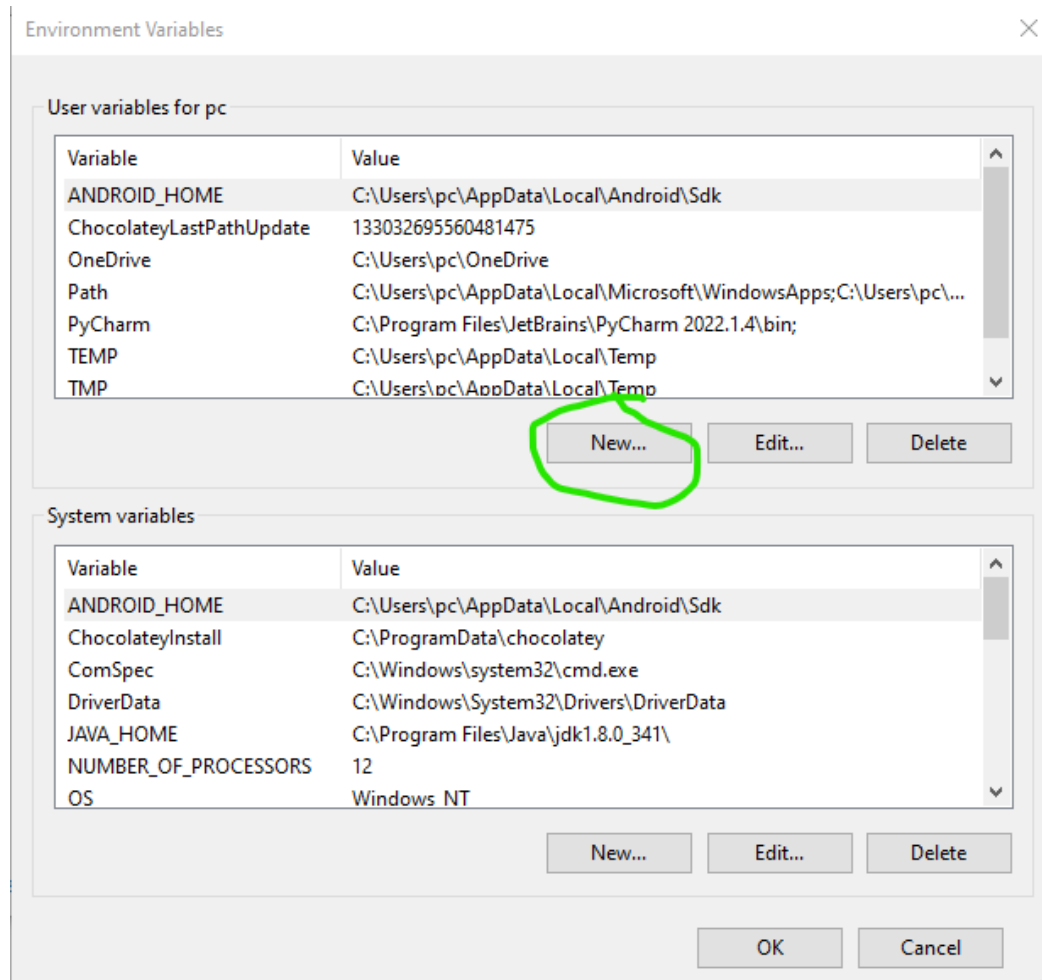
- Then click OK. It will download and install tools that are no installed yet.

- Take note Android SDK location as shown:

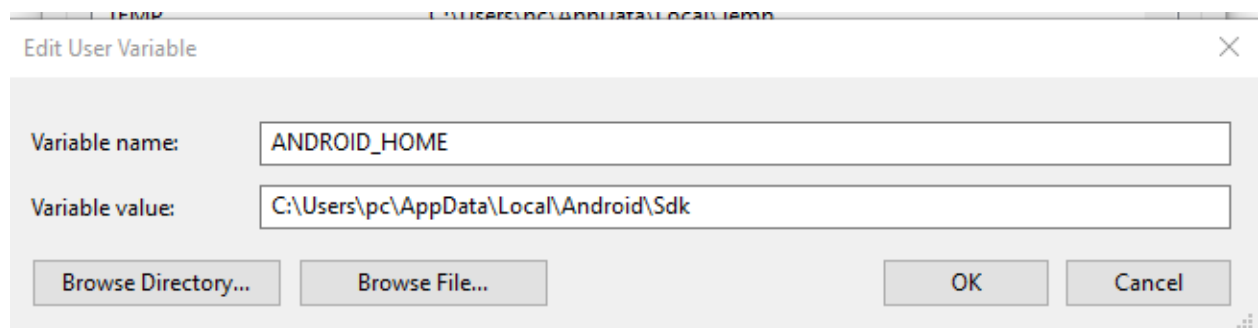


10. Then open Advanced System Settings by following Step 3 and Step 4 from A. Java JDK Installation part

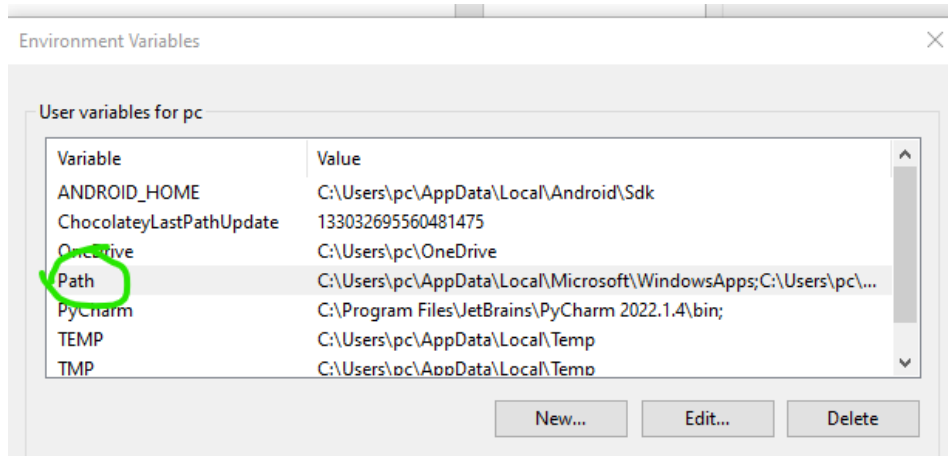
11. Click 'New' under 'User variable for pc'



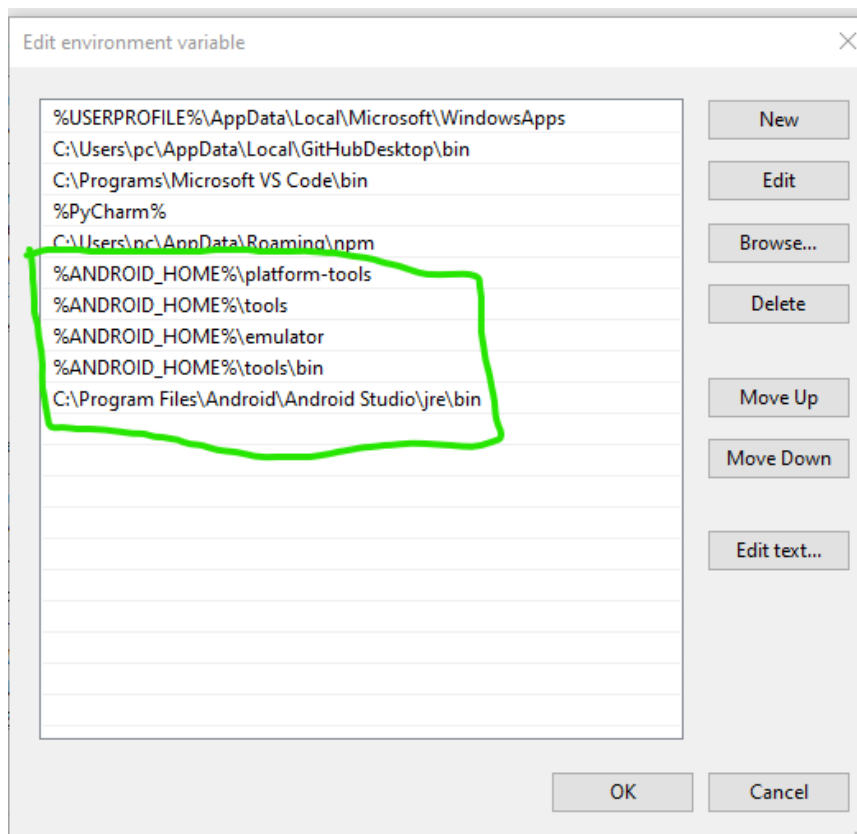
12. Write **ANDROID_HOME** on 'Variable name:' and write location of Android SK on 'Variable value:'. Then click OK



13. Then click Path under 'User variable for pc'



14. Just add variables as shown in figure below



15. Don't forget to click OK on Environment Variable window and then click OK on Advanced System Setting window.

C. Node.js Installation

First, install node.js (<https://nodejs.org/en/download/>). Please watch the instructions in the next video, which explains how to download and install node.js using the Microsoft Installer (.msi).

<https://www.youtube.com/watch?v=AuCuHvgOeBY&t=2s>

Note: When Node.js install, the latest Python be installed automatically.

D. Appium Installation

Install Appium (<https://github.com/appium/appium-desktop/releases/tag/v1.22.3-4>). Please watch the next video for instructions on installing Appium Desktop.

<https://www.youtube.com/watch?v=1ot8cZoUk6o>

E. Vysor and ADB Driver

Next, you will need to set-up your mobile device to work with Vysor. Please watch the next video for instructions on how to install and use Vysor.

Download link: <https://www.vysor.io/>

Tutorial video link: <https://www.youtube.com/watch?v=ECZ5e-OuKqQ&t=1s>

For your phone to be viewable in Vysor, you will need to install ADB Drivers (Android SDK Platform) on your computer and enable USB Debugging on your phone.

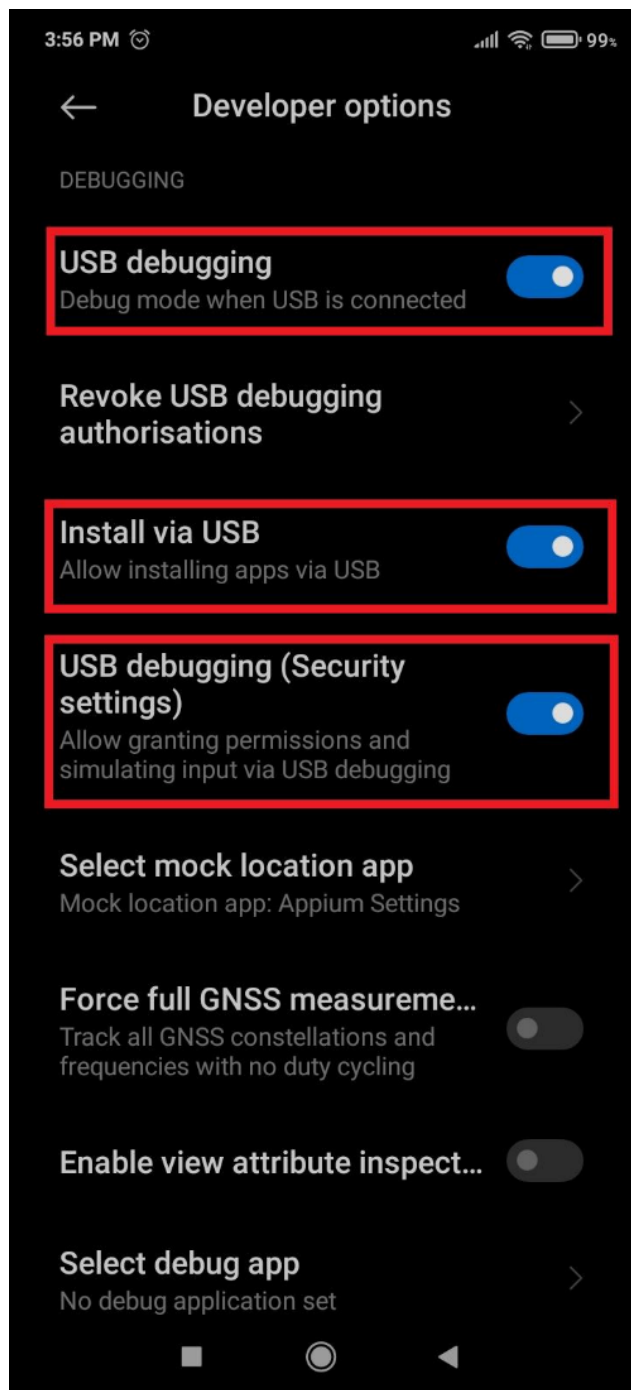
Download link <https://adb.clockworkmod.com/>

Once you have installed Vysor and started it, you will see a screen like the image below. On this screen, you will see two links (highlighted in red, in the image) that will take you to a video showing how to enable USB Debugging and to the webpage where you can download the ADB Drivers Windows Installer, respectively.



Please note, in addition to the above, when enabling USB Debugging on your phone, and you are in the Developer Options on your mobile user interface (see image below), make sure you enable:

- USB debugging – Debug mode when USB is connected
- Install via USB – Allow installing apps via USB
- USB debugging (Security Setting) – Allow granting permissions and simulating input via USB Debugging



See this [link](#) for additional information on this step: Enabling USB Debugging on an Android Device.

F. Python Library Installation

1. Open command prompt and run it as administration
2. Type these two commands:

```
pip install robotframework
```

```
pip install --upgrade robotframework-appiumlibrary
```

G. PyCharm (Python IDE)

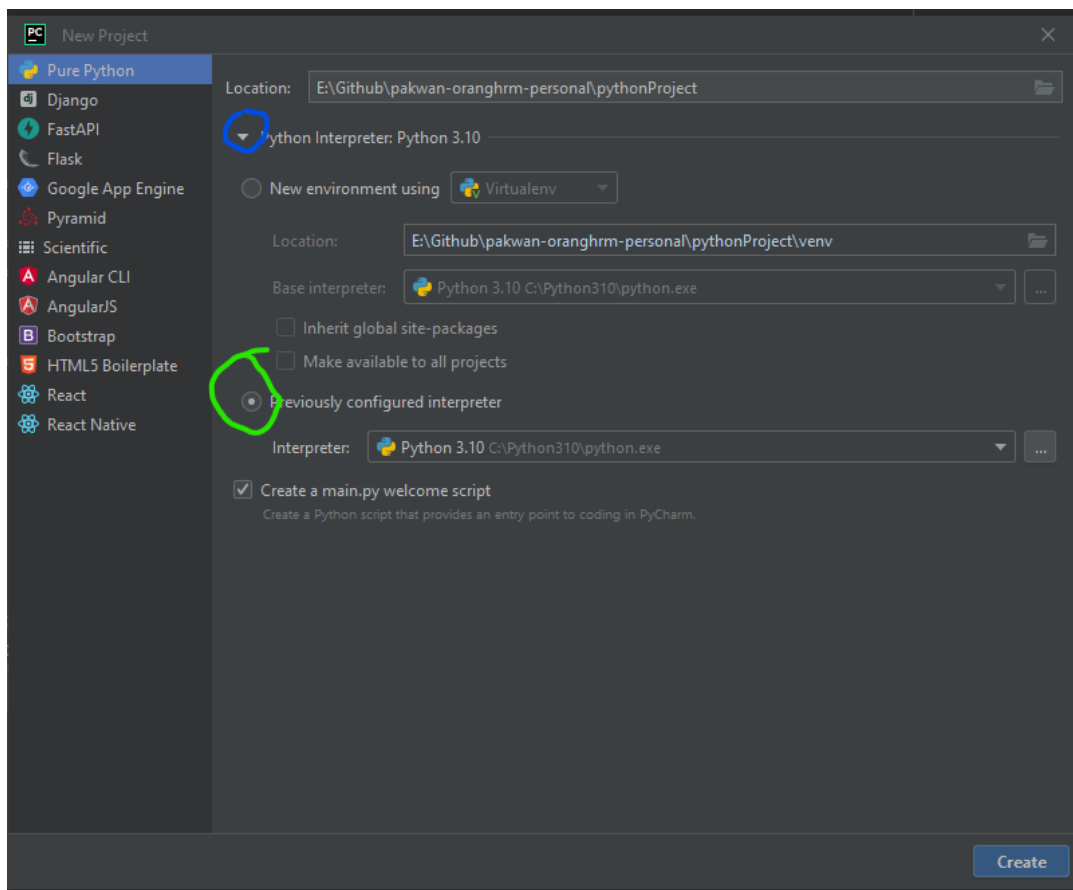
Download from <https://www.jetbrains.com/pycharm/download/#section=windows> and choose Community version

Part 2: Setup Project

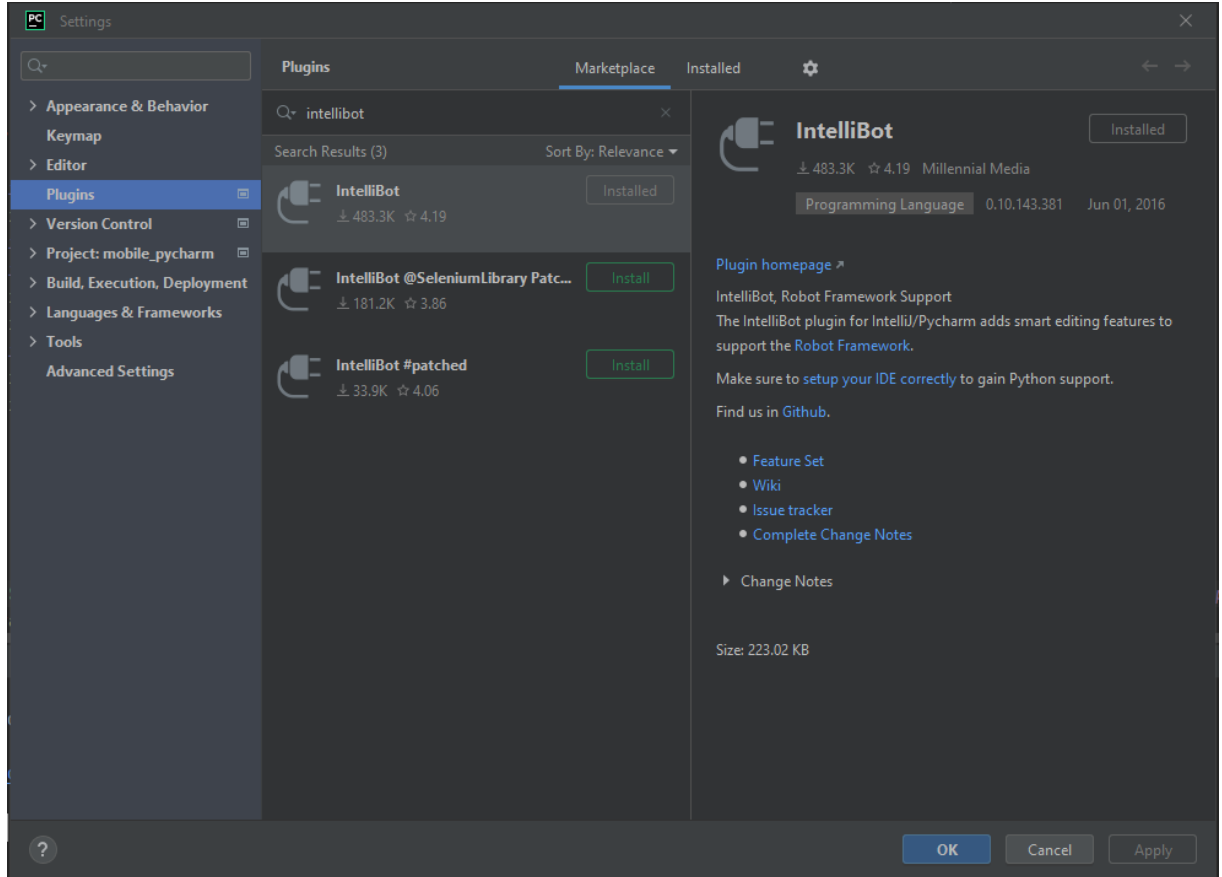
A. Create Project in PyCharm

Note: Check version of python by typing **python --version** on command prompt and check the location for python folder

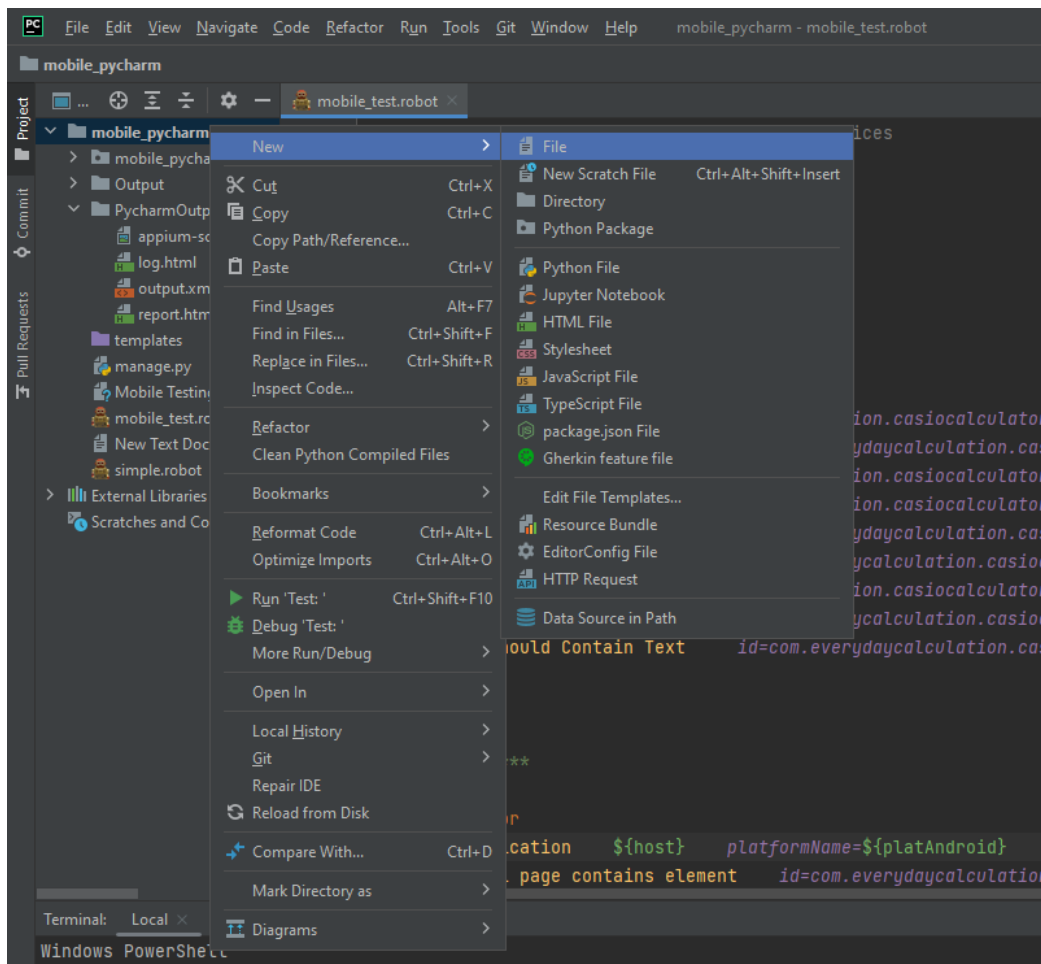
1. Click File->New Project
2. Click on arrow icon (blue circle). Next, click 'Previously configuration interpreter' (green circle). Then choose location of python.exe



3. New project is created. Next, go to File->Setting
4. On left side, choose Plugin. Then on search bar, type **IntelliBot** (plugin for Robot Framework). Click Install. After finish, PyCharm will automatically restart.



5. PyCharm is opened back. On folder directory on left side, right click on main project folder and choose New->File



6. Type **AnyName.robot** and press Enter button.

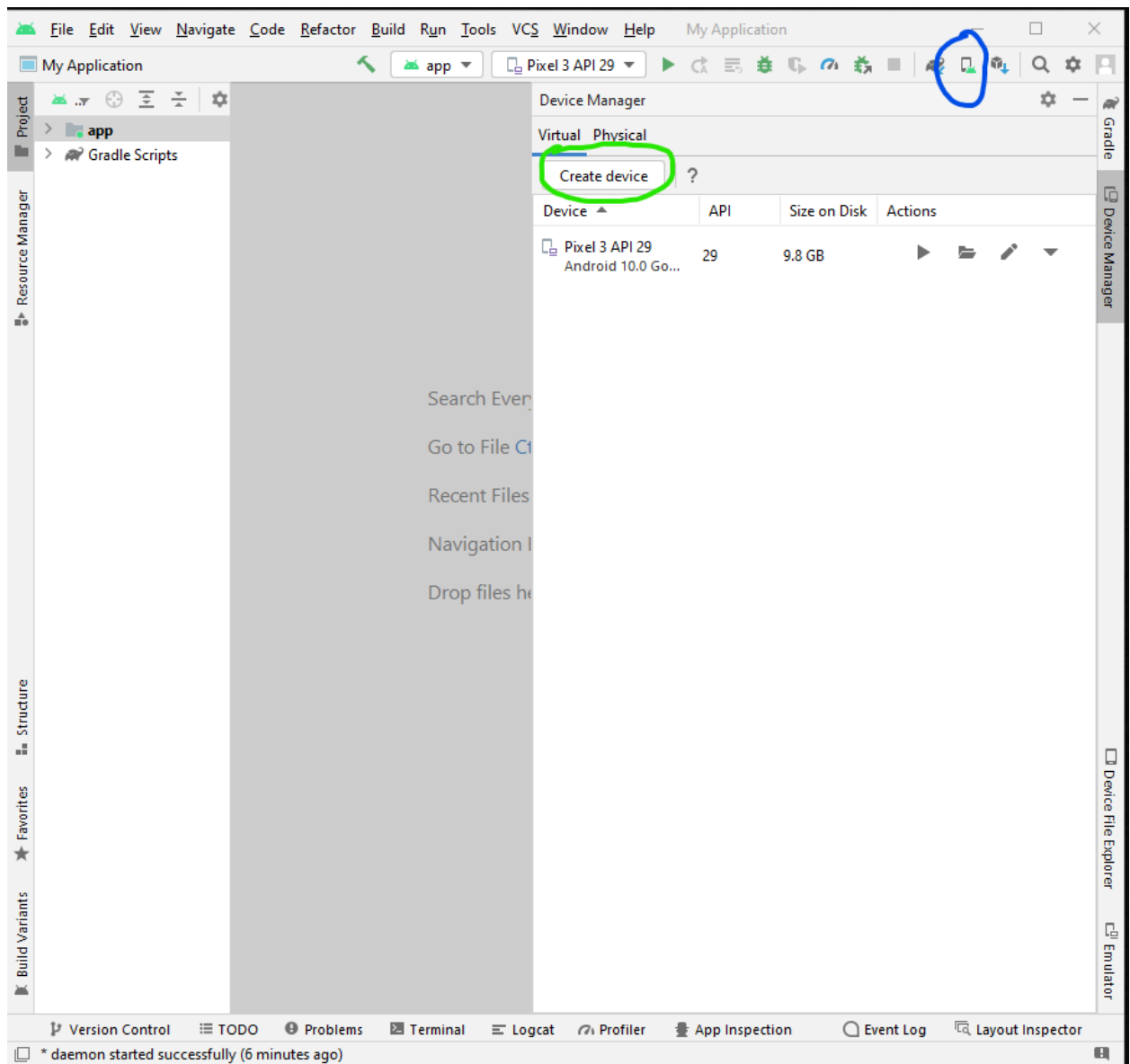
B. Setup on smartphone.

On smartphone (and emulator), install **SimpleCalc** app as sample app from Google Play Store. And you also need to install **APK Info** app from Play Store because this app can provide information about sample app's information that useful for testing.

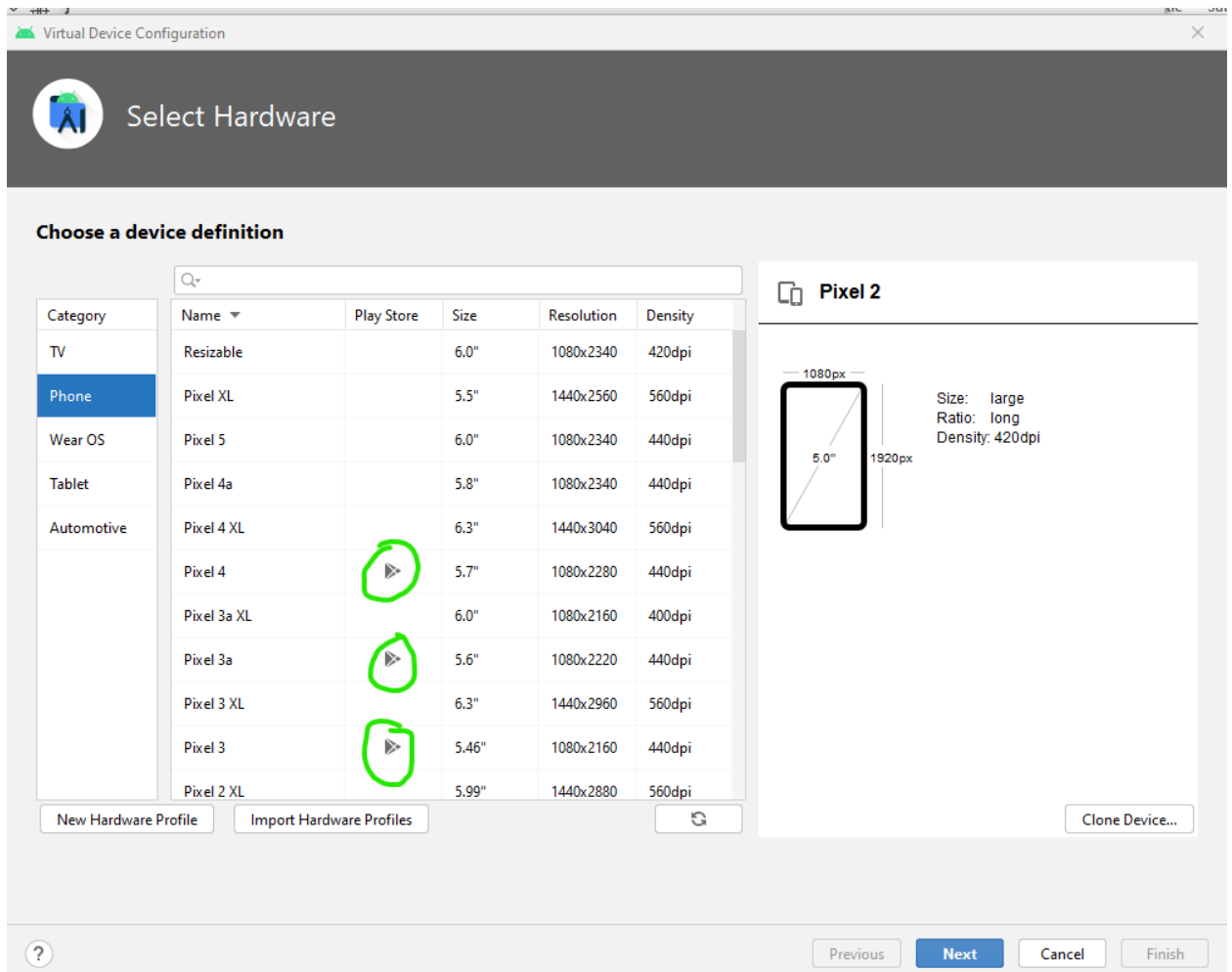
C. Smartphone Emulator (Optional)

This part is good if your smartphone cannot connect to your computer. Even so, if you can connect your smartphone but you want to use emulator, you need disconnect your computer from computer to avoid Appium confuse to connect between emulator and real smartphone.


1. Open Android Studio.
2. Click the button (blue circle). And then click 'Create Device' (green circle)



3. Select Hardware window is opened. You can choose any model as long as it has Play Store as shown in figure below:



4. Click Next. It will navigate to ‘Select System Image’. Choose **Q**. Then click ‘Next’. It will download and install Q.


 System Image

Select a system image

Recommended x86 Images Other Images

Release Name	API Level ▼	ABI	Target
<i>TiramisuPrivacySandbox</i>	<i>TiramisuPrivacySan</i>	<i>x86_64</i>	<i>Android API TiramisuPrivacy:</i>
API 33 Download	33	x86_64	Android API 33 (Google Play)
Sv2 Download	32	x86_64	Android API 32 (Google Play)
S Download	31	x86_64	Android 12.0 (Google Play)
R Download	30	x86	Android 11.0 (Google Play)
Q	29	x86	Android 10.0 (Google Play)
Pie Download	28	x86	Android 9.0 (Google Play)
Oreo Download	27	x86	Android 8.1 (Google Play)
Oreo Download	26	x86	Android 8.0 (Google Play)
Nougat Download	25	x86	Android 7.1.1 (Google Play)
Nougat Download	24	x86	Android 7.0 (Google Play)

Q



API Level
29

Android
10.0

Google Inc.

System Image
x86


We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
[See the API level distribution chart](#)

?


Previous **Next** Cancel Finish

5. It will navigate to Verify Configuration. You can let unchanged or change its configuration based on your preference. Finally, click Finish

 Android Virtual Device (AVD)


Verify Configuration

AVD Name

 Pixel 3a

5.6 1080x2220 xxhdpi


Change...


 Q

Android 10.0 x86

Change...

Startup orientation

 Portrait

 Landscape

Emulated Performance

Graphics:


Device Frame

☒ Enable Device Frame

Show Advanced Settings

AVD Name

The name of this AVD.

 ?

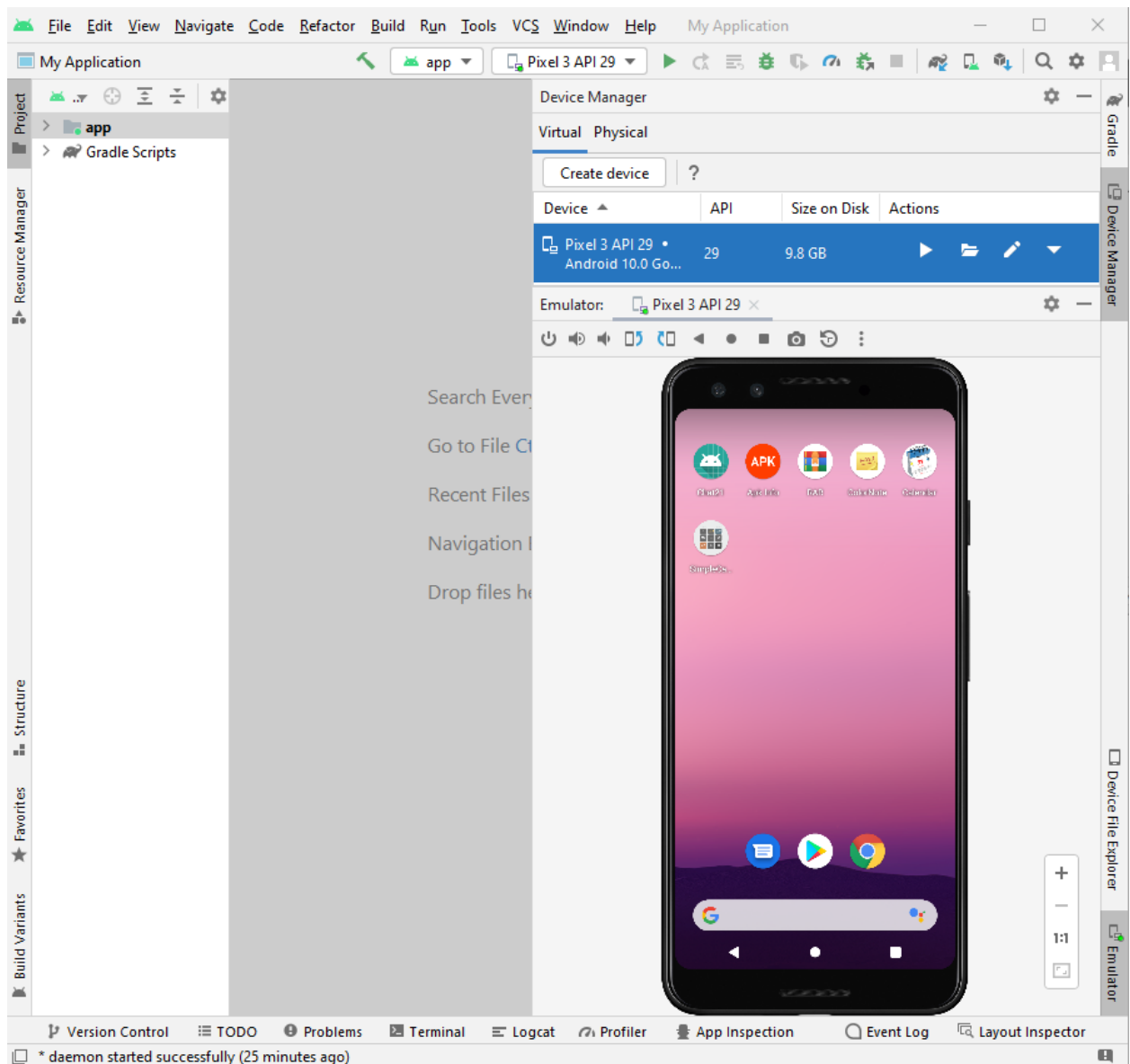
Previous

Next

Cancel

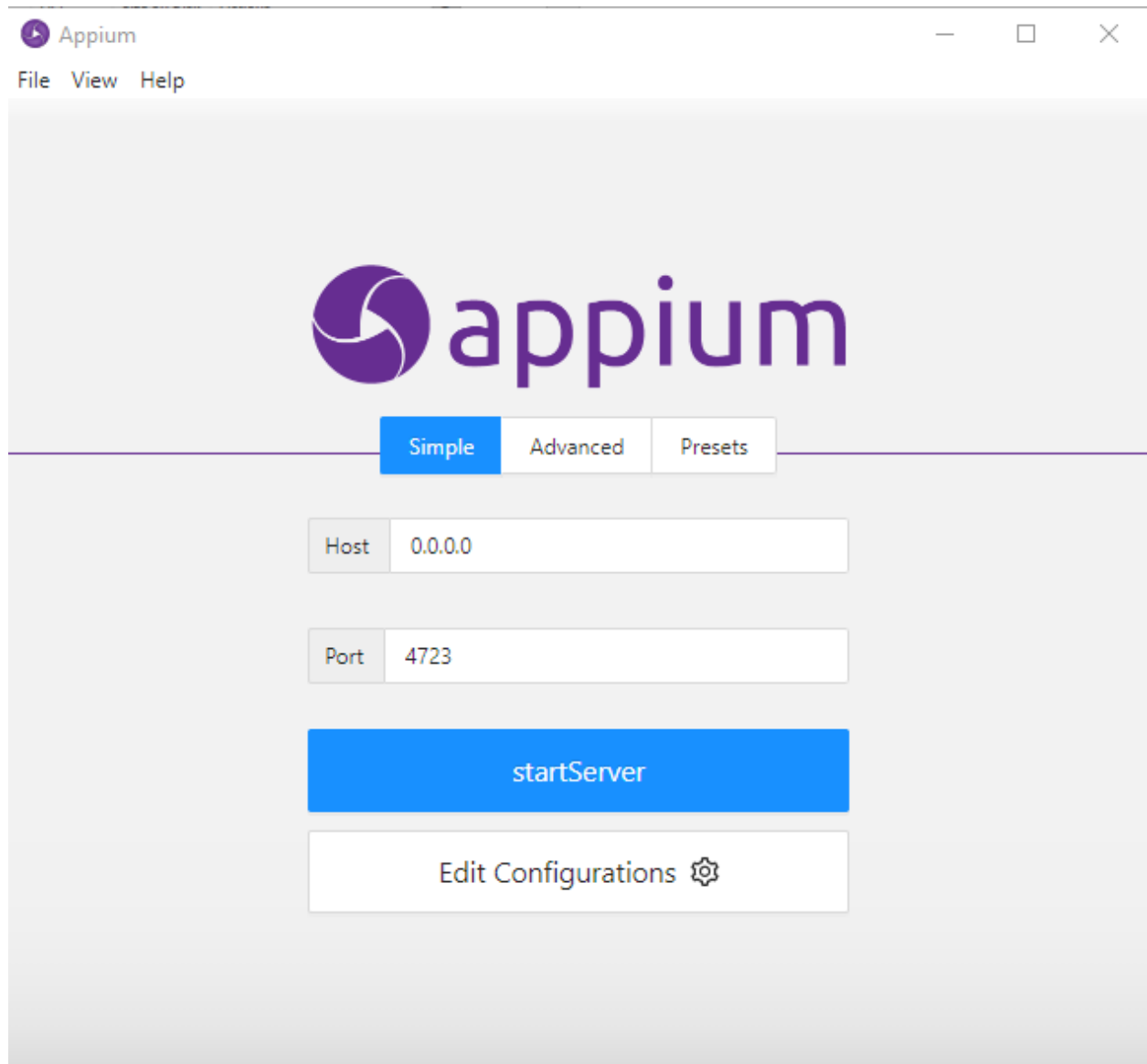
Finish

- Those windows will be closed. Now, emulator be opened. You just login Google Account on this emulator and you can install requirement apps for this mobile testing



D. Appium Desktop

When you want to start mobile testing, make sure Appium is open. Appium window be like figure below:



You can remain Port number or you can change it. Make sure you take note Port number. Those numbers will be used on Robot Framework. To start Appium, just click startServer.

Part 3: Open Application Testing

1. Open PyCharm. Then open project and AnyName.robot

2. Start write

```
*** Settings ***
```

3. Import Appium Library by typing

```
Library    Appium
```

4. After that, type

```
*** Variable ***
```

5. There are three variable that we need to set:

a. Host: Put Port number on variable. For example, our port is 4723. So, we need type like this:

```
${host}    http://localhost:4723/wd/hub
```

b. Platform: For this testing, we use Android. Just tpe:

```
${platAndroid}    Android
```

c. Devices: To get your device/emulator id number, open command prompt and type **adb devices**. It will list devices with their id. Go back to script in PyCharm and then type:

```
${device1}    602c342e
```

6. In figure below shows how Settings and Variable is written:

```
*** Settings ***
Library           AppiumLibrary

*** Variables ***
${host}           http://localhost:4723/wd/hub
${platAndroid}    Android
#for code of device, open cmd and type: adb devices
${redmi9}         602c342e
${emulator}       emulator-5554
```

7. Now we go to Test Case. We create test case Open as shown in figure below:

```
28
29 *** Test Cases ***
30
31 Open_calculator
32
```

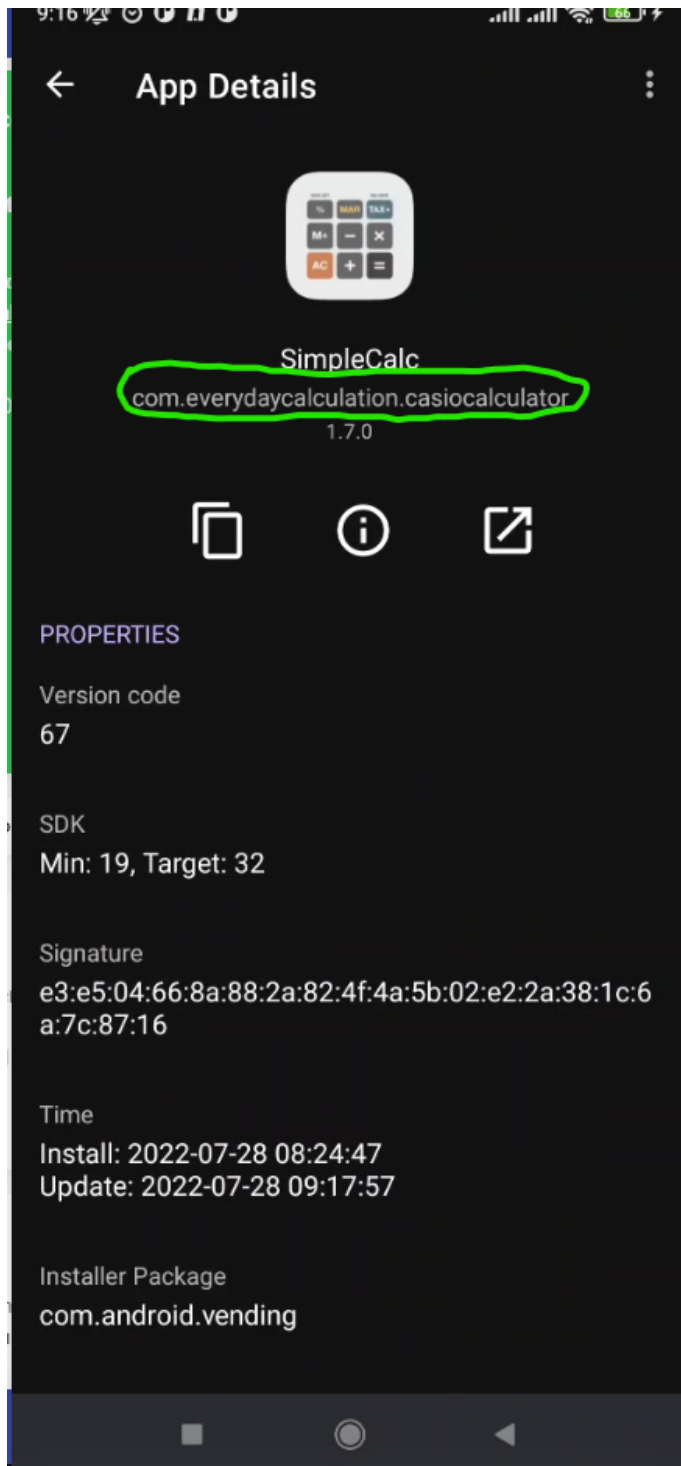
8. To start it, type script as shown in below:

```
Open_calculator
    Open Application    ${host}    platformName=${platAndroid}    deviceName=${emulator}

    appPackage=com.everydaycalculation.casiocalculator    appActivity=com.everydaycalculation.casiocalculator.Basic
```

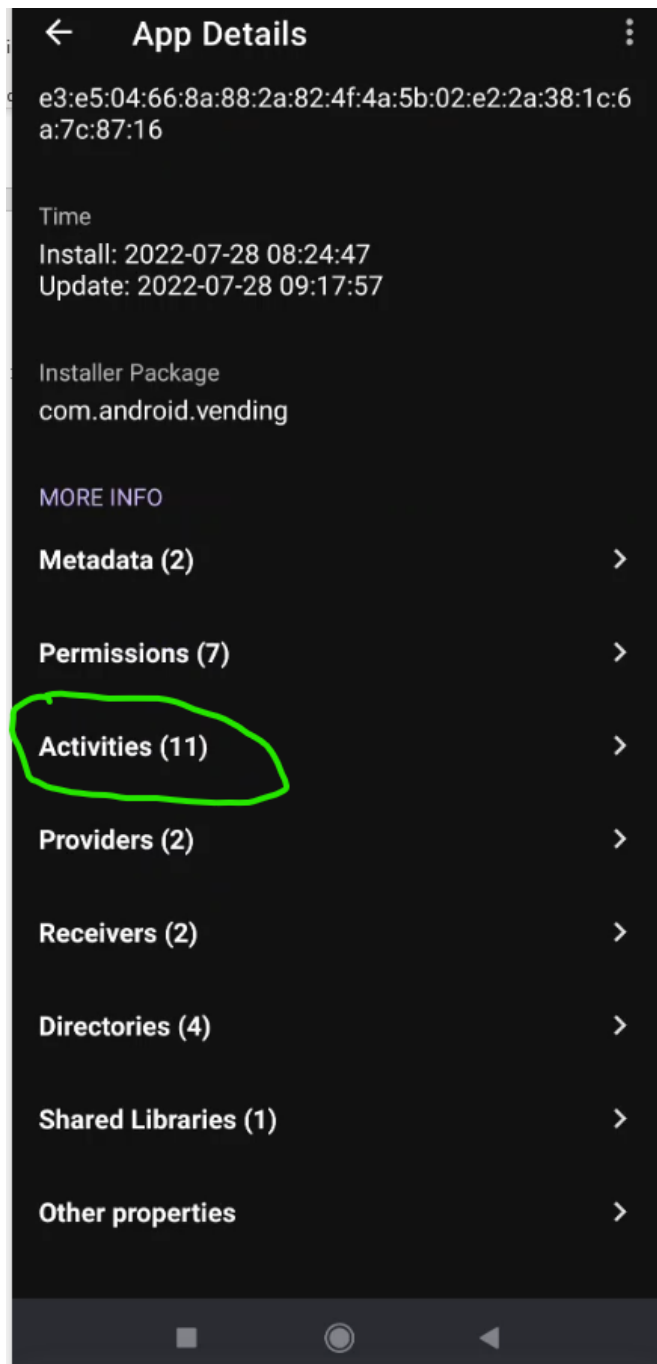
9. There two variables which are appPackage and appActivity. The appPackage refers ID for app that we want to test and appActivity refers to ID of open application activity. To get them, we need open App Info from smartphone/emulator.
10. Open Apk Info and search that app. For this testing, we search SimpleCalc.

11. When we found it, click this app. It will show information as shown in figure below:

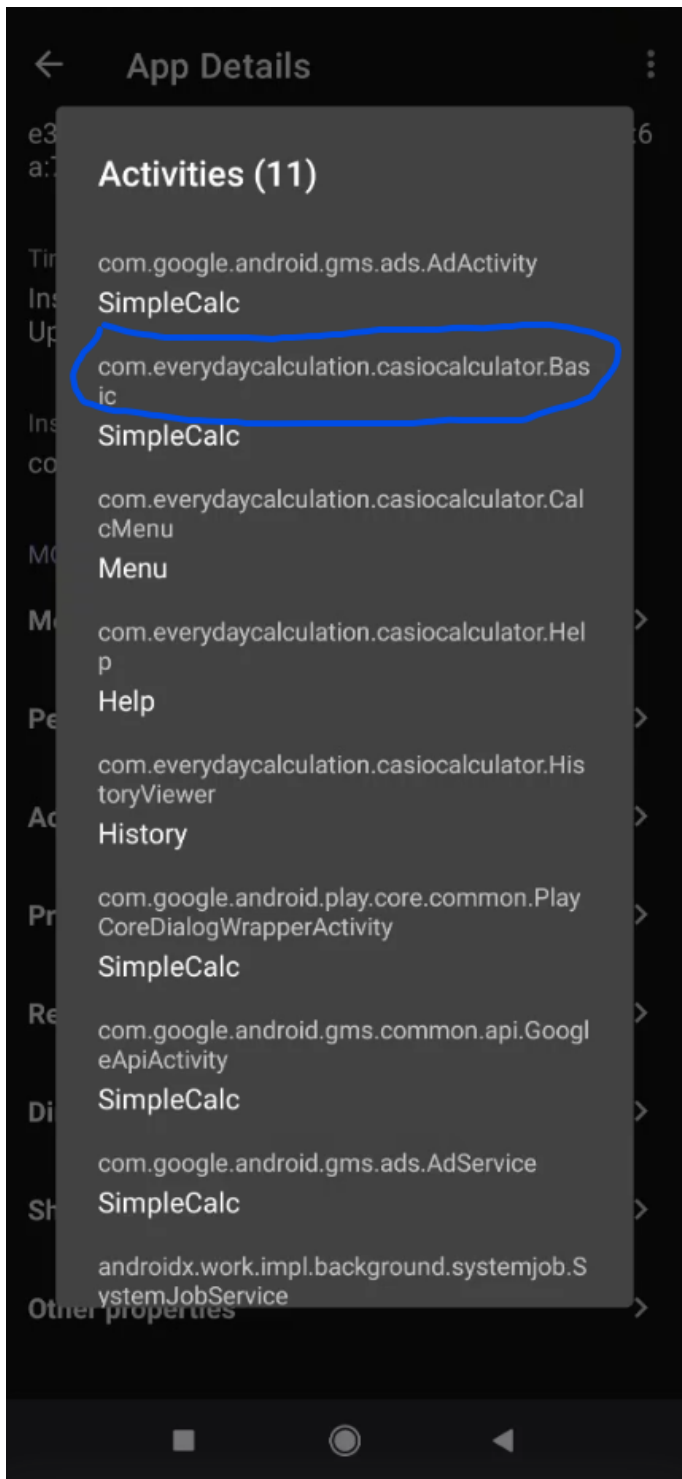


12. The green circle is the appPackage ID. Take note this ID by copying it.

13. To get appActivity, scroll down and you will find Activities as shown in below. Then click it.



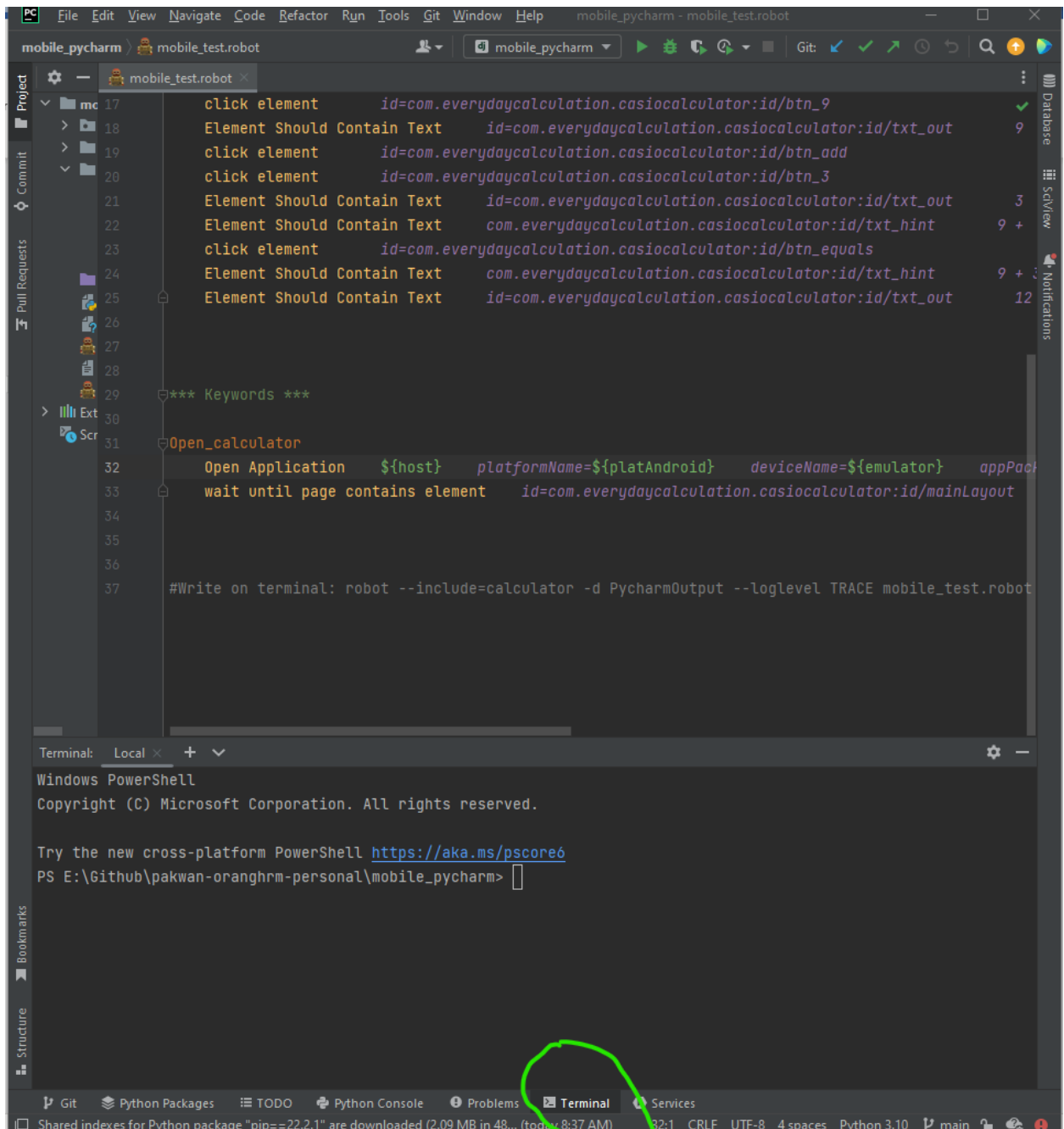
14. When click it, it will shows a lot of activities as shown in below:



15. After that., you need to find activities that related to open application or menu. Usually, we can find it at the top or second of the list (blue circle). When found it, copy activity ID.

16. Go back to PyCharm. Then paste those two ids into script.

17. Now, we execute testing script. To do so, click terminal in the bottom of PyCharm (green circle) and it will open as shown in figure below:



18. On terminal, type

`robot -d PycharmOutputfolder --loglevel TRACE AnyName.robot`

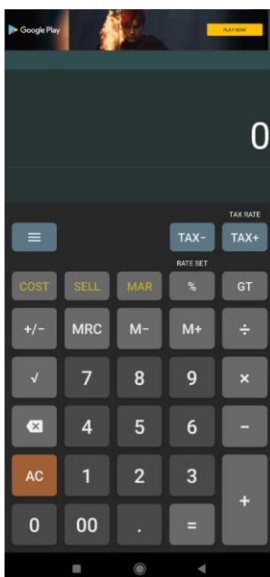
19. Then, press Enter. It will run test. After finish, we get read testing report from project folder in PycharmOutputFolder.

Part 4: Input and output validation testing.

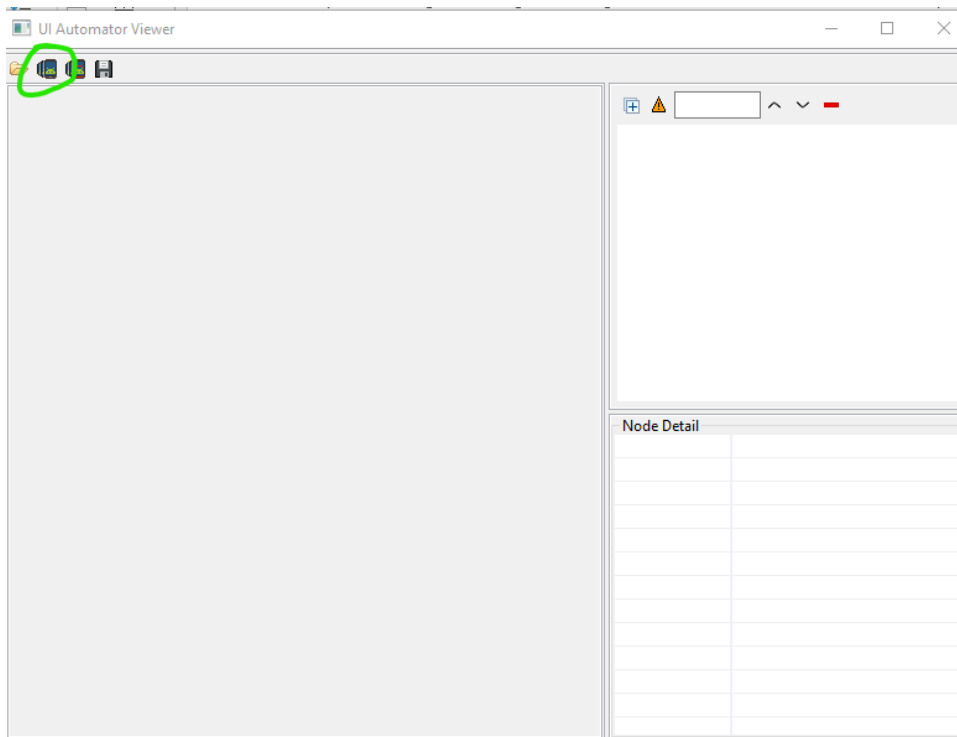
1. For open test case, you can use it as user keywords as shown in below:

```
*** Keywords ***
Open_calculator
    Open Application    ${host}    platformName=${platAndroid}    deviceName=${emulator}    appPackage=com.everydaycalcul
```

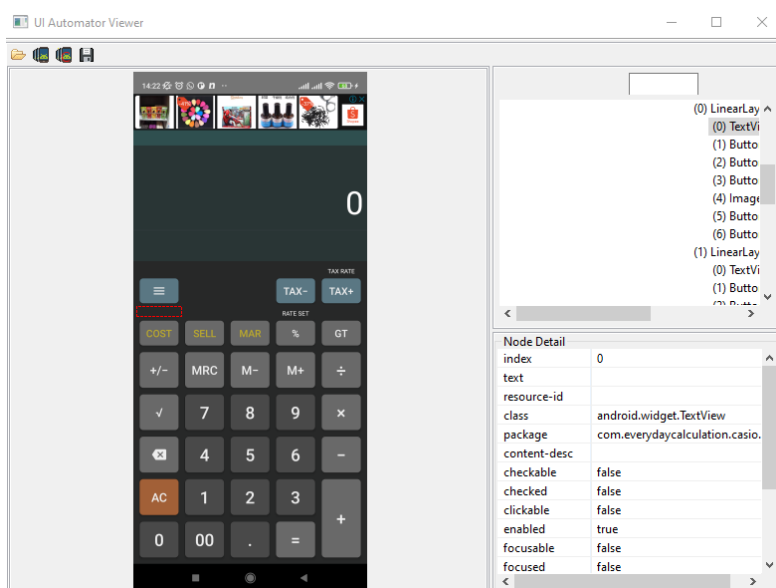
2. Create new test case. For this part, we create additional operation test cases.
3. There are two Appium keywords that will be used
 - a. Click element {element path}
 - b. Element should Contains Text {element path} {Text that should be}
4. If we look on webpage testing, we can get element path from webpage by clicking right mouse button and choosing 'Inspect'. For mobile testing, we use UI Automator Viewer from Android SDK.
5. Firstly, open app (more specifically, open page inside app) that you want to test and extract element path. For this testing, we open SimpCalc main page as shown in below:



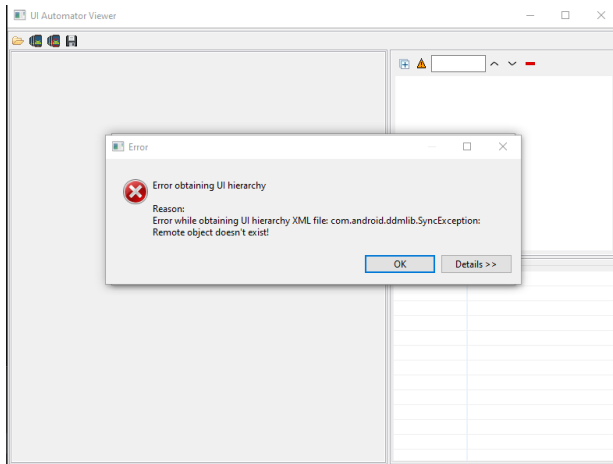
6. After that, open command prompt and type **uiautomatorviewer**.
7. When you press Enter, UI Automator Viewer is opened as shown in below:



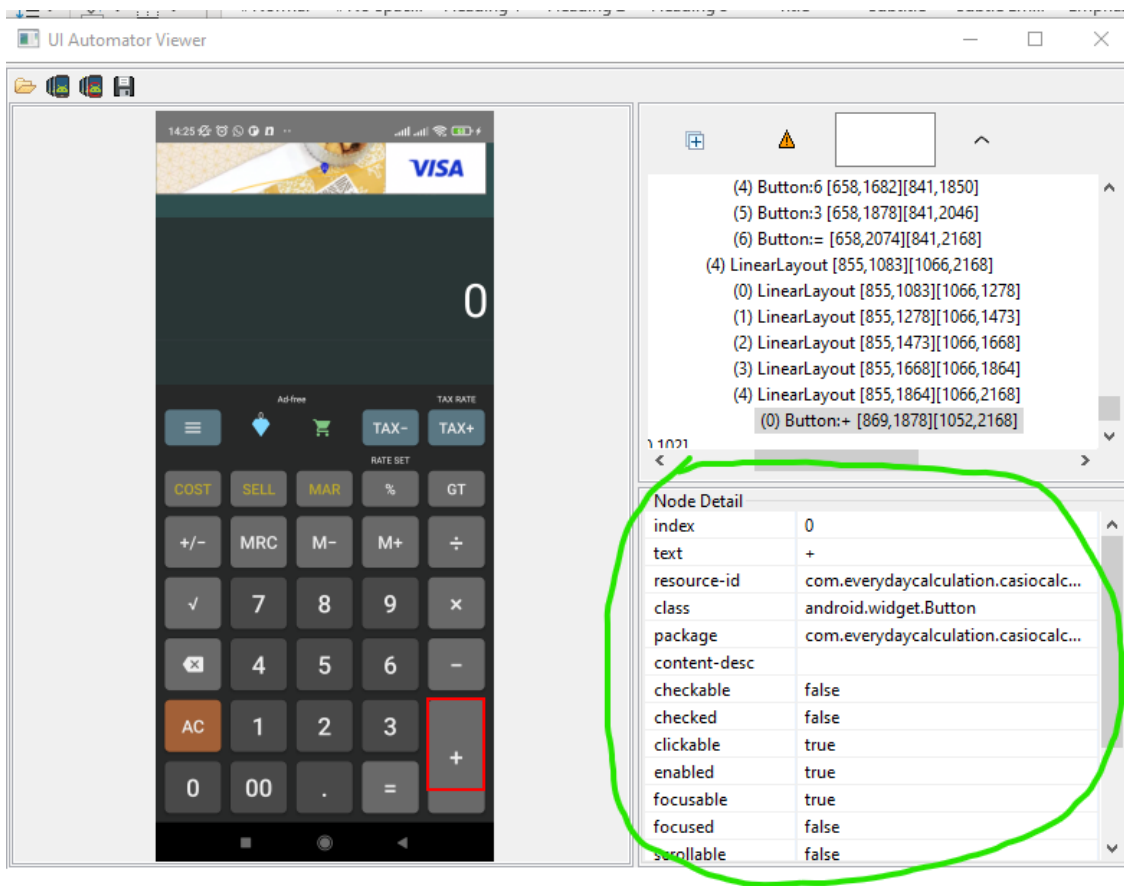
8. To screenshot apps, click Device Screen Shot (green circle). It screenshot the app from device as shown in below:



Note: If you get error message when click screenshot (as shown in figure below), it means you cannot do that. To solve it, you type **adb reconnect** on command prompt and open UI Automator Viewer again.



9. On this page, click element that you want to use. For example, we click on + button, Ui Automator Viewer show all information about that element on left side is shown in below



10. The information that we need to look up is resource-id.
11. For this test cases, we want to test $9 + 3$ equal to 12. So, we need to get id of:
- a. Plus button
 - b. Equal button
 - c. Number 9 button
 - d. Number 3 button
 - e. Output screen
 - f. Hint screen (below output screen)
12. Before we start write script, we plan the steps of test case. For this test case, its steps are:
- i. Click number 9 button
 - ii. Number 9 is shown on output screen
 - iii. Click + button
 - iv. Click number 3
 - v. Number 3 is shown on output screen
 - vi. '9 +' is shown on hint screen
 - vii. Click = button
 - viii. Number 12 is shown on output screen
 - ix. '9 + 3' is shown on hint screen
13. Now, write the script as shown in figure below:

```
*** Test Cases ***
Test_additional
    [Tags]    calculator
    Open_calculator
    click element    id=com.everydaycalculation.casiocalculator:id/btn_9
    Element Should Contain Text    id=com.everydaycalculation.casiocalculator:id/txt_out    9
    click element    id=com.everydaycalculation.casiocalculator:id/btn_add
    click element    id=com.everydaycalculation.casiocalculator:id/btn_3
    Element Should Contain Text    id=com.everydaycalculation.casiocalculator:id/txt_out    3
    Element Should Contain Text    com.everydaycalculation.casiocalculator:id/txt_hint    9 +
    click element    id=com.everydaycalculation.casiocalculator:id/btn_equals
    Element Should Contain Text    com.everydaycalculation.casiocalculator:id/txt_hint    9 + 3
    Element Should Contain Text    id=com.everydaycalculation.casiocalculator:id/txt_out    12
```

14. Finally, run the test on terminal.