What is Appium?

Appium is an open-source test automation framework for mobile applications. It allows you to automate testing of:

- Native mobile apps (written for iOS or Android)
- Hybrid apps (combining web and native components)
- Mobile web apps (accessed through mobile browsers)

History

Appium was originally developed by Dan Cuellar in 2011

Under the name “IOS Auto” written in the C# programming language

Key Features

1. **Cross-platform:** Write tests for both Android and iOS using the same API
2. **Language flexibility:** Supports multiple programming languages (Java, Python, JavaScript, Ruby, C#, PHP)

3. **No app modification**: Tests your app as-is without requiring SDK or recompilation
4. **Open-source:** Free to use with a large community support
5. **WebDriver protocol:** Uses the standard Selenium WebDriver JSON wire protocol

Architecture

Appium follows a client-server architecture:

1. Appium Server: Written in Node.js, handles connections from clients
2. Appium Clients: Libraries in various languages that send commands to the server
3. Mobile JSON Wire Protocol: Extends Selenium's protocol for mobile-specific commands
4. Platform-specific drivers:
    - XCUITest for iOS
    - UIAutomator2/Espresso for Android

How Appium Works

1. Your test script sends a request to the Appium server
2. Appium interprets the command and converts it to the platform-specific automation framework
3. The automation framework executes the command on the device/emulator
4. Results are sent back through the same chain

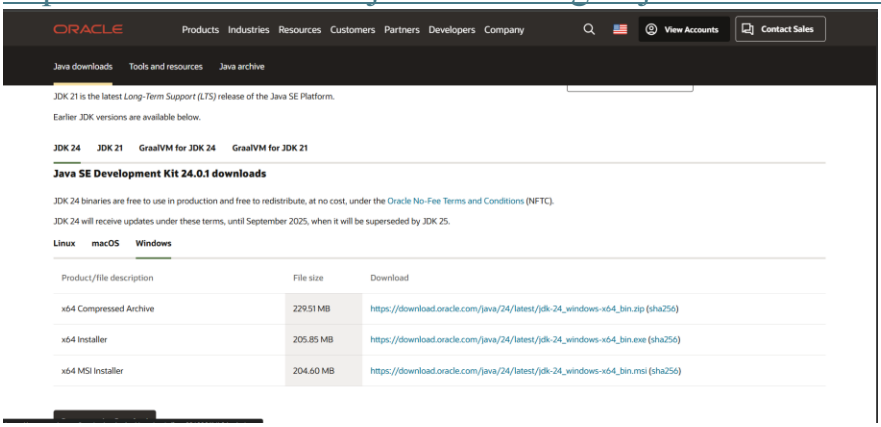
Step-by-Step Setup Guide (For Windows)

Step1 :Install Prerequisites

1.Java Development Kit (JDK)

Download JDK from

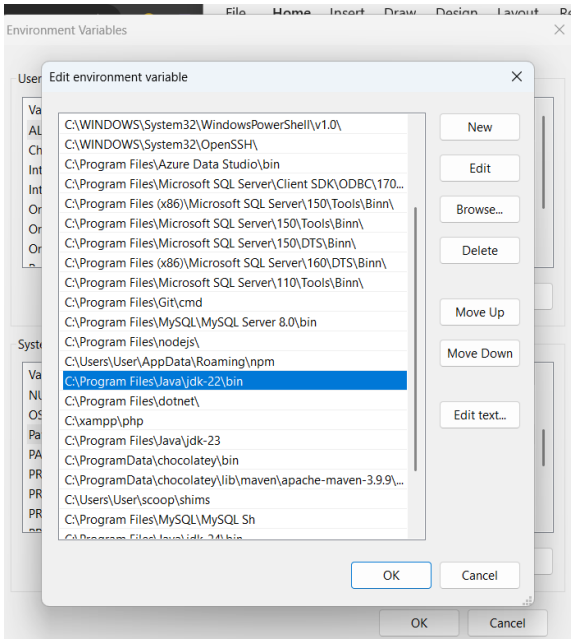
https://www.oracle.com/java/technologies/javase-downloads.html



Install it.

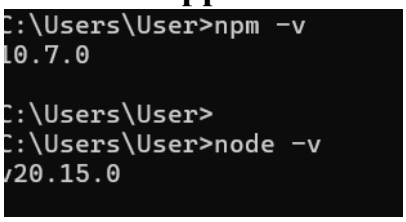
Set environment variables:

- **JAVA_HOME = path to your JDK (e.g., C:\Program Files\Java\jdk-20)**
- **Add %JAVA_HOME%\bin to Path**



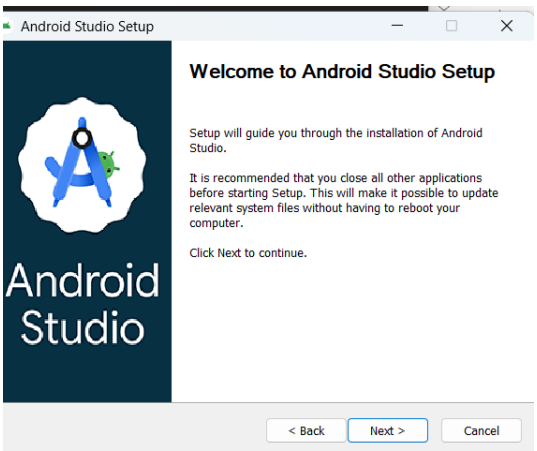
2. Node.js

- **Download from: https://nodejs.org/**
- **Install it. Appium runs on Node.js.**



3. Android Studio (for emulator + Android SDK)

- **Download from: https://developer.android.com/studio**
- **Install it and:**
  Create a virtual device (emulator)
  Install required SDK tools (API 30+ is good)



Step 2:Install Appium

1.Install Appium Server

Run this in Command Prompt:

npm install -g appium

```
C:\Users\User>npm install -g appium
npm warn deprecated inflight@1.0.6: This module is not sup
```

2. Install Appium Doctor (to check setup)

npm install -g appium-doctor

```
C:\Users\User>npm install -g appium-doctor
npm warn deprecated inflight@1.0.6: This module is
```

3. Check environment with:

appium-doctor

```
C:\Users\User>appium-doctor
WARN AppiumDoctor [Deprecated] Please use appium-doctor installed with "npm install @appium/doctor --location=global"
info AppiumDoctor Appium Doctor v.1.16.2
info AppiumDoctor ### Diagnostic for necessary dependencies starting ###
info AppiumDoctor  ✔The Node.js binary was found at: C:\Program Files\nodejs\node.EXE
info AppiumDoctor  ✔Node version is 20.15.0
WARN AppiumDoctor  ✖ANDROID_HOME environment variable is NOT set!
info AppiumDoctor  ✔JAVA_HOME is set to: C:\Program Files\Java\jdk-21
WARN AppiumDoctor  ✖adb, android, emulator, apkanalyzer.bat could not be found because ANDROID_HOME or ANDROID_SDK_ROOT
 is NOT set!
info AppiumDoctor  ✔'bin' subfolder exists under 'C:\Program Files\Java\jdk-21'
info AppiumDoctor ### Diagnostic for necessary dependencies completed, 2 fixes 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  ✖bundletool.jar cannot be found
WARN AppiumDoctor  ✖gst-launch-1.0.exe and/or gst-inspect-1.0.exe cannot be found
```

Step 3: Install Appium Desktop (GUI) *(Optional but helpful)*

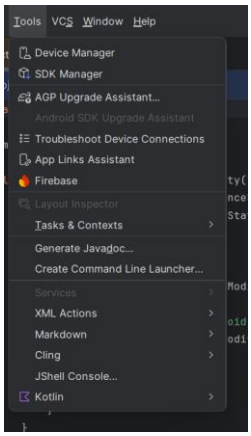
- Download from: https://github.com/appium/appium-desktop/releases
- Install and open
- You can inspect elements in your mobile app using this GUI

Step 4: Set Up Real or Virtual Device

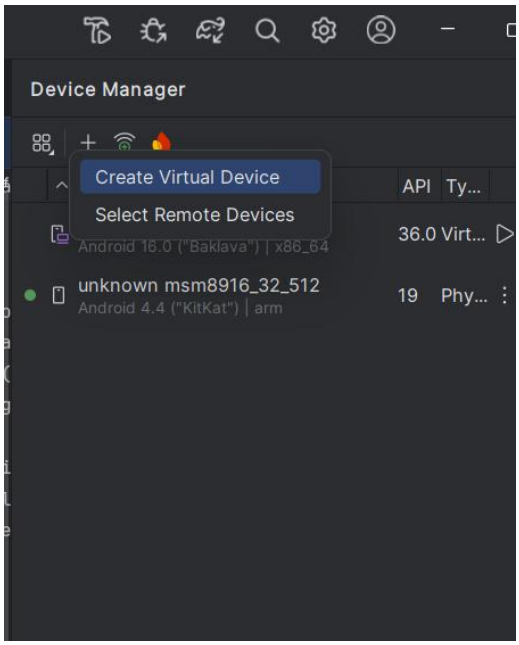
Option1 :Virtual Device

Create a Virtual Device (Emulator)

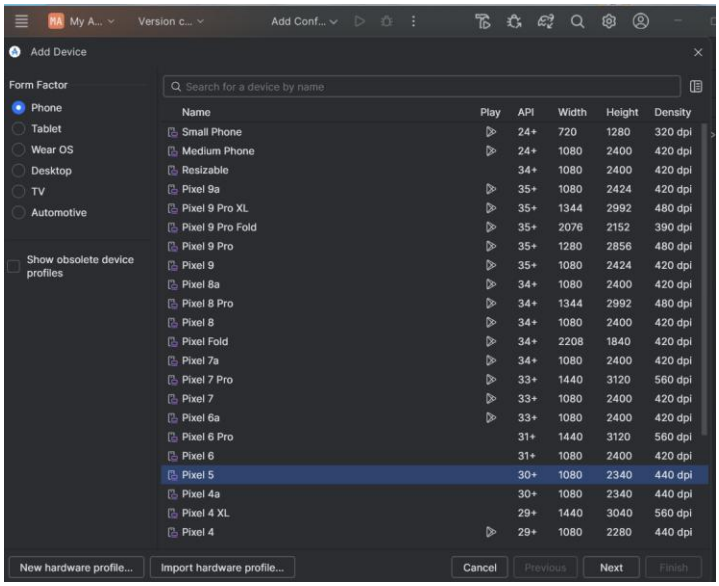
1. In Android Studio, go to **Tools > Device Manager**



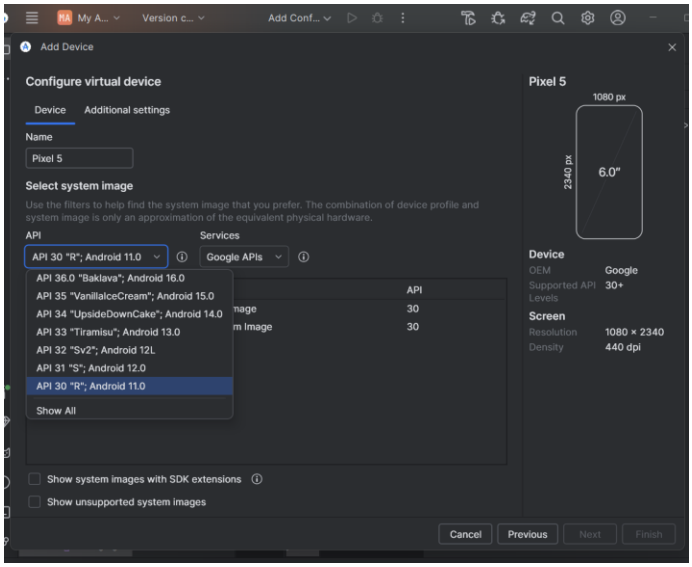
2. Click **Create Device**



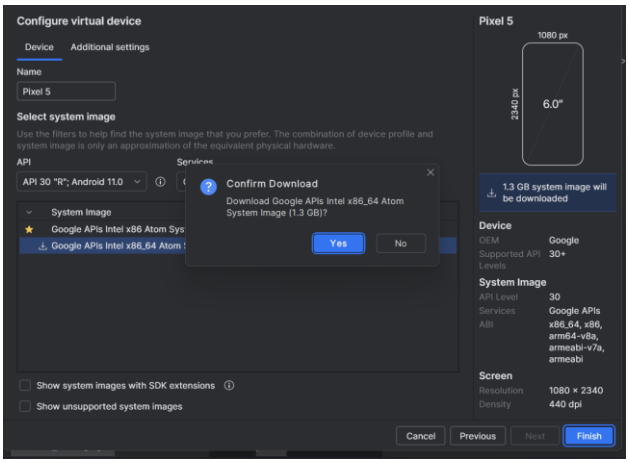
3. Choose a device like **Pixel 5**, then click **Next**



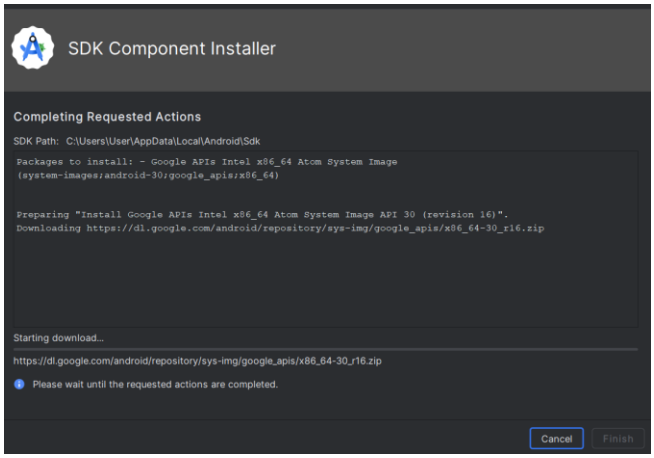
4. Select a system image:
   Recommended: **R (API 30)** or **S (API 31)**



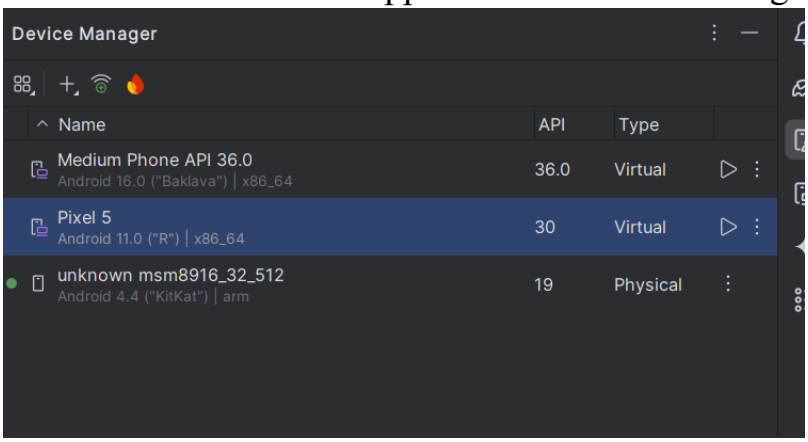
5.Click **Download** if it's not downloaded yet



5. Once downloaded, click **Next > Finish**



Your emulator will now appear in the Device Manager



Launch Emulator

1.In **Device Manager**, click the ▶ **(play)** button next to your virtual device

2. Wait a moment — the emulator will start up like a phone.

Option 2:Real device

How to Enable USB Debugging on an Android Device

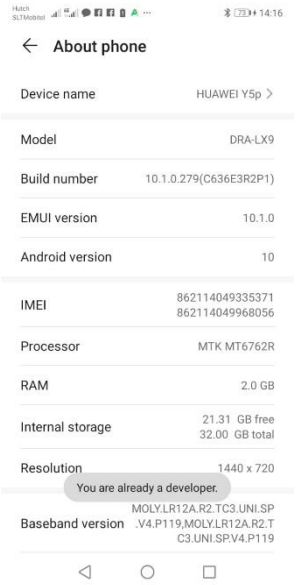
Step 1: Enable Developer Options

1.On your Android phone, go to:
Settings > About phone

2. Find **Build number**

3. Tap Build number 7 times quickly

You'll see a message:"You are now a developer!"



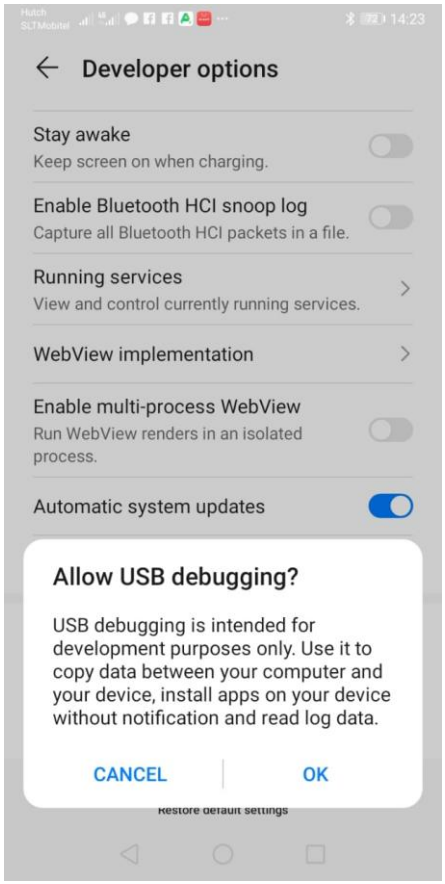
Step 2: Turn on USB Debugging

1.Go to:
Settings > System > Developer options (Sometimes under "Additional settings" or "More settings" depending on your device)

2. Scroll down and find:**USB Debugging**

3. Toggle it ON

4.Confirm if it asks for permission.



Step 3: Connect Your Device to PC

1.Use a USB cable to connect your phone to your computer.

2. On your phone, you might see a popup:

   Choose **"Allow USB debugging"**

   Tap "Always allow from this computer" if asked

Step 4: Verify Connection

1.Open Command Prompt or Terminal

2. Type: adb devices

3. Output should show:

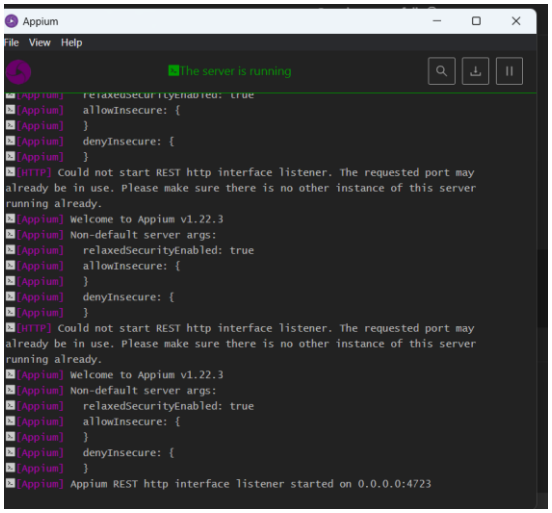
```
C:\Users\User>adb devices
List of devices attached
4383303e        device
9FT9K20713905084        device


C:\Users\User>
```

Start Appium Server

Option 1: Command Line

Appium

```
C:\Users\User>Appium
[Appium] Welcome to Appium v2.19.0
[Appium] The autodetected Appium home path: C:\Users\User\.appium
[Appium] Appium REST http interface listener started on http://0.0.0.0:4723
[Appium] You can provide the following URLs in your client code to connect to this server:
        http://192.168.100.85:4723/
        http://192.168.243.1:4723/
        http://192.168.204.1:4723/
        http://10.0.1.1:4723/
        http://127.0.0.1:4723/ (only accessible from the same host)
[Appium] No drivers have been installed in C:\Users\User\.appium. Use the "appium driver" command to install the one(s)
you want to use.
[Appium] No plugins have been installed. Use the "appium plugin" command to install the one(s) you want to use.
```



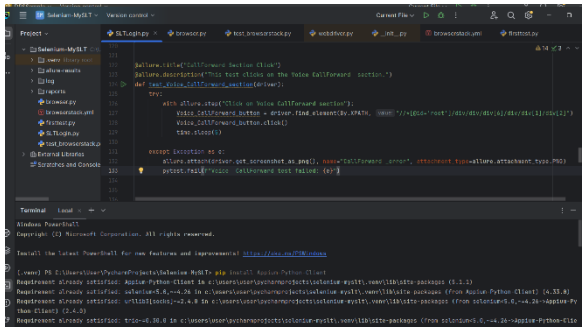
Option 2: Appium Desktop App

1. Open Appium Desktop
2. Click “Start Server”

Install Appium Client for Your Language

Let’s continue with Python (can do Java later if needed):

pip install Appium-Python-Client



Prepare your Test Script (test_appium.py)

1. **Confirm Android SDK Environment Variables**
   Before the script, make sure ANDROID_HOME or ANDROID_SDK_ROOT is set in your terminal:

```
C:\Users\User>echo %ANDROID_HOME%
C:\Users\User\AppData\Local\Android\Sdk

C:\Users\User>echo %ANDROID_SDK_ROOT%
C:\Users\User\AppData\Local\Android\Sdk
```

2. **Start Appium Server with Base Path /wd/hub**
   Open a new terminal (PowerShell or CMD) and run: appium --base-path /wd/hub

```
C:\Users\User>appium --base-path /wd/hub
[Appium] Welcome to Appium v2.19.0
[Appium] Non-default server args:
[Appium] {
  basePath: '/wd/hub'
}
[Appium] The autodetected Appium home path: C:\Users\User\.appium
[Appium] Attempting to load driver uiautomator2...
[Appium] Requiring driver at C:\Users\User\.appium\node_modules\appium-uiautomator2-driver\build\index.js
[Appium] AndroidUiautomator2Driver has been successfully loaded in 1.421s
[Appium] Appium REST http interface listener started on http://0.0.0.0:4723/wd/hub
[Appium] You can provide the following URLs in your client code to connect to this server:
        http://192.168.100.85:4723/wd/hub
        http://192.168.243.1:4723/wd/hub
        http://192.168.204.1:4723/wd/hub
        http://10.0.1.1:4723/wd/hub
        http://127.0.0.1:4723/wd/hub (only accessible from the same host)
[Appium] Available drivers:
[Appium]   - uiautomator2@4.2.4 (automationName 'UiAutomator2')
[Appium] No plugins have been installed. Use the "appium plugin" command to install the one(s) you want to use.
```

3. **Create your Python Test Script**
   test_appium.py
   In your project folder, create or update test_appium.py as:

```python
from appium import webdriver
from appium.options.android import UiAutomator2Options

# Setup Appium options
options = UiAutomator2Options()
options.platform_name = "Android"
options.device_name = "9FT9K20713905084"  # Your connected device ID
options.automation_name = "UiAutomator2"
options.app = r"C:\Users\User\Downloads\myapp.apk"  # Path to your APK

# Connect to Appium server with /wd/hub base path
driver = webdriver.Remote(command_executor: "http://localhost:4723/wd/hub", options=options)

# Implicit wait example
driver.implicitly_wait(10)

print("App launched successfully!")

# Quit the driver session after test
driver.quit()
```

4. Verify Your Device Is Connected and Ready
   Run this command in terminal: adb devices

```
C:\Users\User>adb devices
List of devices attached
4383303e        device
9FT9K20713905084        device
```

5. Run Your Test Script
   Activate your Python virtual environment and run: python test_appium.py

```
    at doListen (node:net:2116:7)
    at processTicksAndRejections (node:internal/process/task_queues:83:21)
(.venv) PS C:\Users\User\PycharmProjects\Selenium-MySLT> python test_appium.py
```