Appium Testing

Mobile Testing Automation

SoftUni Team

Technical Trainers

Software University

http://softuni.bg

You Have Questions?

sli.do

#QA-FrontEnd

Table of Contents

1. **Mobile Testing**
2. **Appium** Introduction
3. Appium for **Android App** Automation
   - Basic Appium Setup
   - Setup for Android Mobile Testing
   - Android Emulator
4. Appium **Inspector**
5. Testing Demo

Mobile Testing

Introduction

Mobile Testing Intro

- Evaluates mobile applications to ensure they meet required standards of quality, performance, and user experience across various devices and operating systems
  - **Functionality**: Verifies the app works as intended
  - **Usability**: Ensures the app is user-friendly
  - **Performance**: Tests the app's speed, responsiveness, and stability
  - **Compatibility**: Checks the app's behavior on different devices, screen sizes, and OS versions

Types of Mobile Apps

- **Native Apps**
  - Developed for specific platforms (iOS using Swift, Android using Kotlin)
  - High performance, requires separate development for each platform
- **Hybrid Apps**
  - Built with web technologies (HTML, CSS, JavaScript) in a native container
  - Cross-platform, may sacrifice some performance
- **Web Apps**
  - Accessed through mobile browsers
  - No installation, limited device feature access

Native Vs. Cross-Platform Mobile Testing Tools

- **Cross-Platform**
  - Support all mobile platforms, including iOS, Android, and Windows such as Appium
  - Support many programming languages
- **Native**
  - Developed, released, and support one single mobile platform
    - Espresso for Android; can use only Java or Kotlin
    - XCUITest for iOS; Swift or Objective-C

Appium

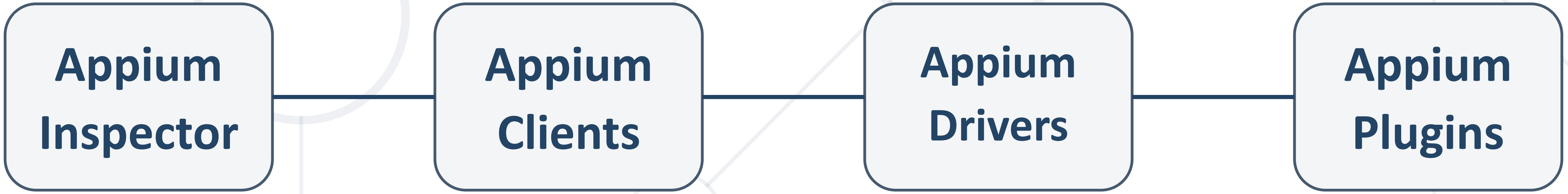
Introduction

What is Appium?

- An **open-source tool** for **testing** native, hybrid, and mobile web apps
- Known as the "**standard**" for mobile app test automation
- Created by Dan Cuellar in 2012 as an open-source framework for mobile automation
- Inspired by Selenium WebDriver, aims to offer a **single solution for automating mobile apps** across different platforms and languages
- Initially focused on iOS, later added support for Android
- **Simple** to use, active user community, keeps up with new mobile technologies

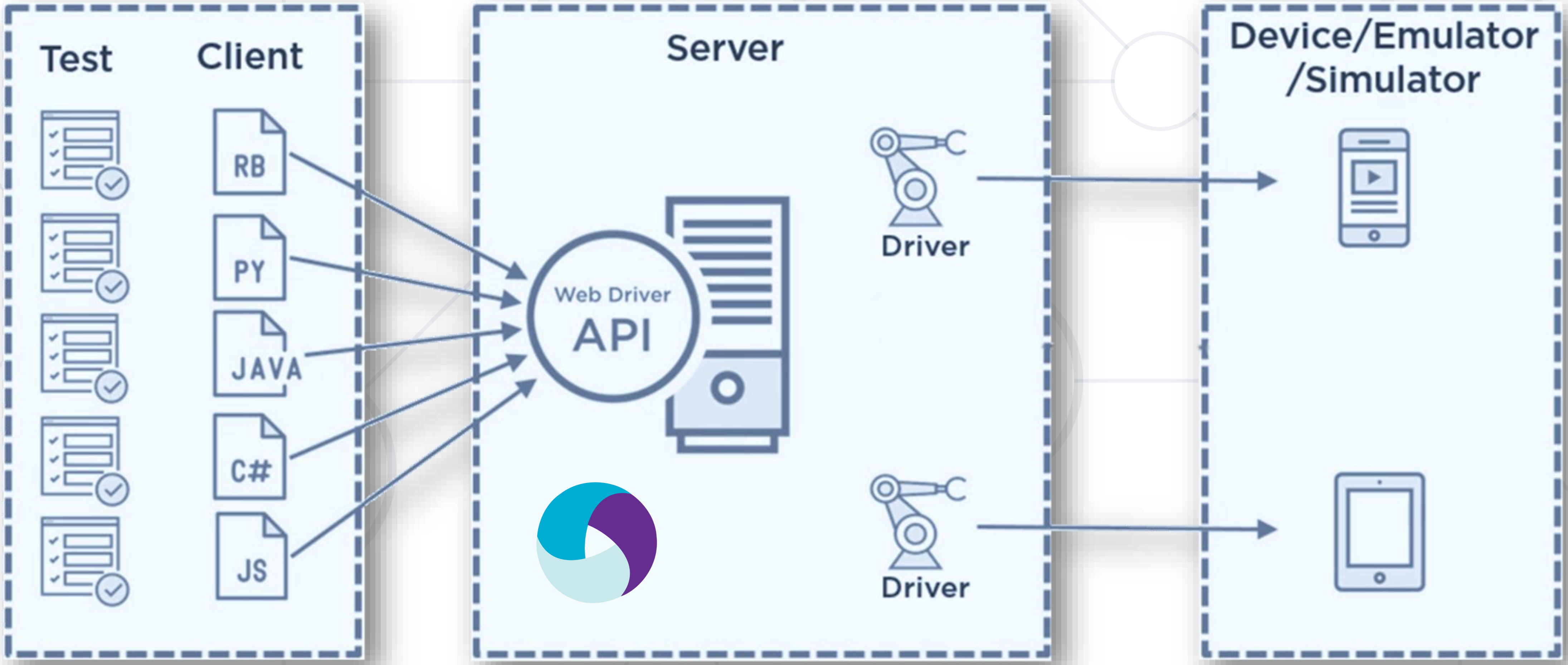
Appium Ecosystem

- **Inspector**: A desktop based application, used to inspect and identify different locators; also execute different Appium commands
- **Clients**: Appium provides client libraries, also known as "client bindings", which are available in multiple programming languages
- **Drivers**: Node.js based drivers
- **Plugins**: Extend and modify the functionalities of Appium



Appium Architecture

- Appium utilizes a **client-server architecture**



Appium Architecture

- **Appium Server**
  - Handles communication between test scripts and mobile devices
  - Receives commands from test scripts
  - Translates them into corresponding actions
  - Forwards them to mobile devices through the WebDriver interface
- **Appium Options**
  - Defines options for device platform, version, app information, and other parameters
  - Sends these capabilities to the Appium server
  - Establishes a connection with the appropriate mobile device or emulator

Appium Architecture

- WebDriver
  - Interacts with the Appium server for mobile web browsers or hybrid applications
  - Sends commands to perform actions on the mobile application
  - Provides a unified API for interacting with mobile applications
- **W3C Protocol (World Wide Web Consortium (W3C) WebDriver Protocol)**
  - Communicates between the test script and the Appium server
  - Uses RESTful HTTP endpoints and corresponding commands
  - Controls mobile devices and applications

Appium Architecture

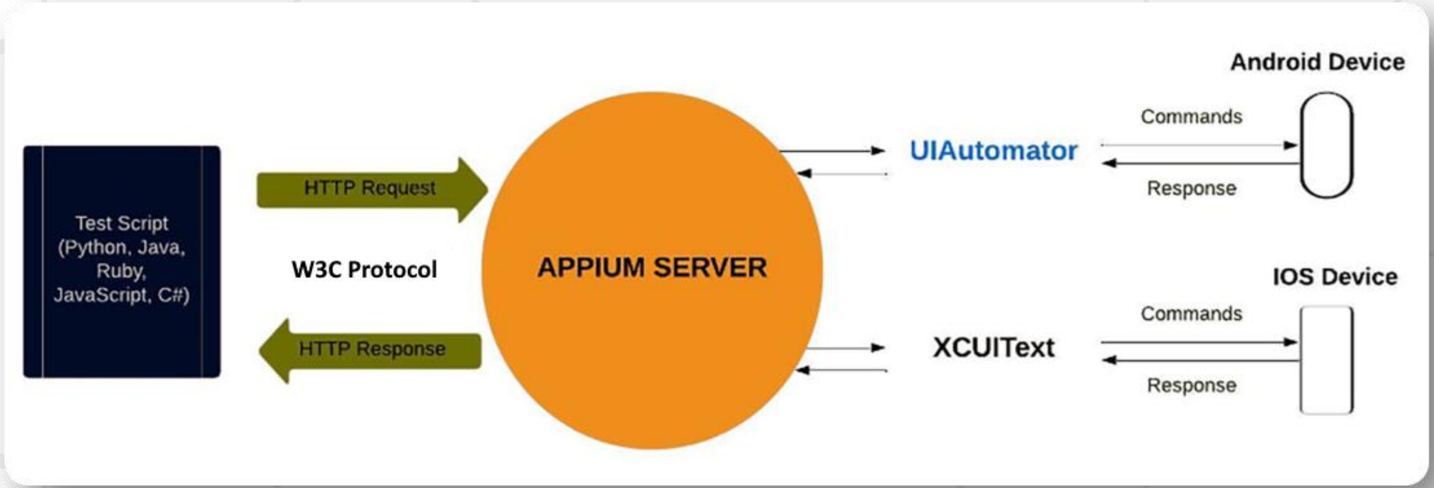
- Mobile Device
  - Uses platform-specific automation frameworks to interact with mobile devices
  - **ADB (Android Debug Bridge)**: Executes commands on Android devices
  - **Instruments Library:** Used by Appium to control iOS applications
- Mobile Application
  - The app under test is installed on the mobile device or emulator
  - Uses automation frameworks to access the application's elements

Low-Level Drivers

- **UiAutomator2Driver**: Automates Android applications using the UiAutomator2 framework
- **EspressoDriver**: Automates Android applications using the Espresso testing framework
- **XCUITestDriver**: Default driver for automating iOS applications; Utilizes Apple's XCUITest framework
- **XCTestDriver**: Alternative for automating iOS applications using the XCTest framework; Suitable for older devices or versions of iOS that do not support XCUITest

Execution Flow

- **Receive Connection**
  - Client initiates a session
  - The server creates a session ID
- **Execute Commands**
  - Server processes and sends commands to devices
  - Commands are translated and sent to devices
- **Return Responses**
  - Server returns execution results
  - Devices execute commands and send back responses



Basic Appium Setup on Windows

Appium Server, Appium-Doctor, Appium Driver

Basic Setup on Windows

- Check if **Node.js** is installed:

```
node -v and npm -v
```

- If not, download & install Node.js
  - https://nodejs.org/en/download/
- Install **Appium Server** via NPM

```
npm install -g appium@latest
```

- Verify that Appium Server is **installed**

```
appium --version or appium -v
```

Basic Setup on Windows

- Install required **Appium drivers** as per testing needs

```
// Android
appium driver install uiautomator2
// iOS
appium driver install xcuitest
```

- Check installed drivers

```
appium driver list
```

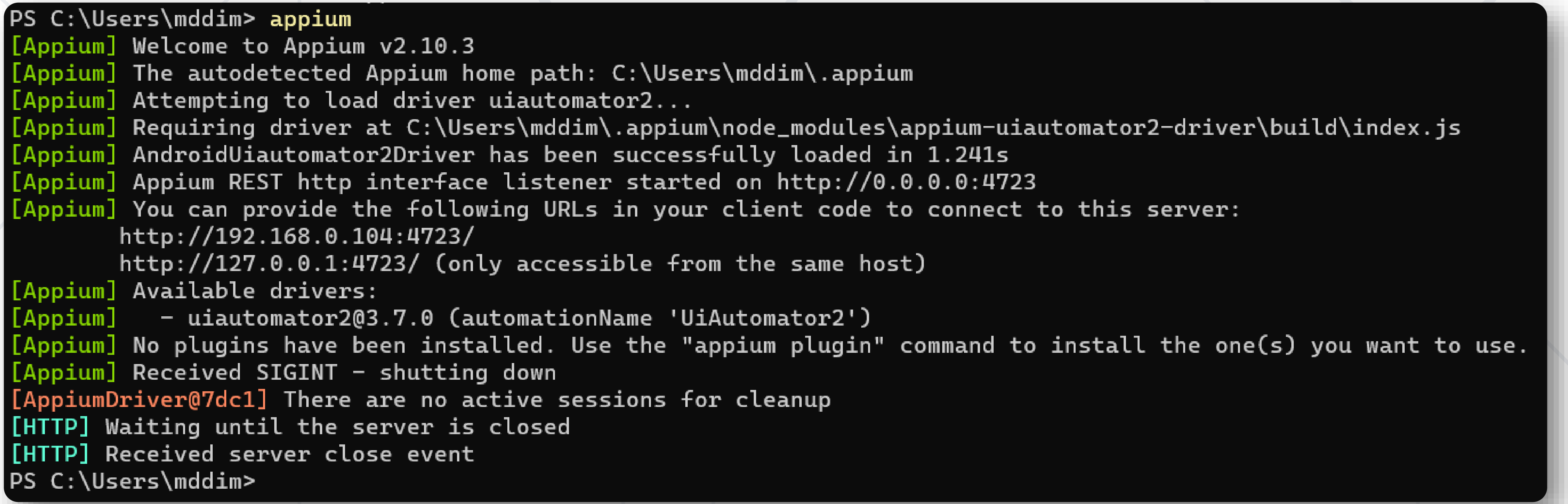
- Check available driver updates

```
appium driver list --updates
```

Basic Setup on Windows

- Run command "**appium**" to start the server and get information on our installed Appium (**ctrl + c** to quit)

```
appium
```



```
PS C:\Users\mddim> appium
[Appium] Welcome to Appium v2.10.3
[Appium] The autodetected Appium home path: C:\Users\mddim\.appium
[Appium] Attempting to load driver uiautomator2...
[Appium] Requiring driver at C:\Users\mddim\.appium\node_modules\appium-uiautomator2-driver\build\index.js
[Appium] AndroidUiautomator2Driver has been successfully loaded in 1.241s
[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://192.168.0.104:4723/
        http://127.0.0.1:4723/ (only accessible from the same host)
[Appium] Available drivers:
[Appium]   - uiautomator2@3.7.0 (automationName 'UiAutomator2')
[Appium] No plugins have been installed. Use the "appium plugin" command to install the one(s) you want to use.
[Appium] Received SIGINT - shutting down
[AppiumDriver@7dc1] There are no active sessions for cleanup
[HTTP] Waiting until the server is closed
[HTTP] Received server close event
PS C:\Users\mddim>
```

Setup for Android Mobile Testing

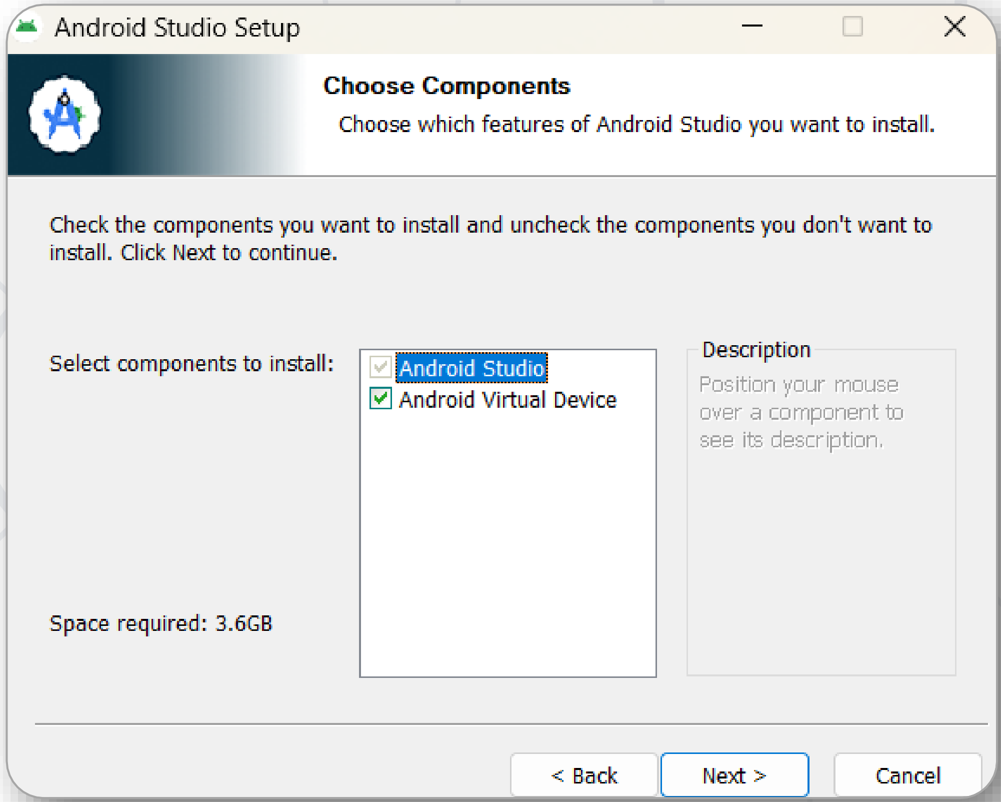
Dependencies

Necessary Dependencies for Appium Testing

- **Android SDK** - Libraries and tools for Android apps
- **Android SDK Tools** - Utilities for debugging and deploying
- **Java JDK** - Includes the Java Runtime Environment (JRE) and development tools
- **Environment Variables**
  - **ANDROID_HOME**: Path to the Android SDK directory
  - **JAVA_HOME**: Path to the Java Development Kit directory
  - **Path Variables**: Ensure SDK and Java tools are accessible from the command line

Setup Android SDK and SDK Tools

- **Download and install Android Studio**
  - https://developer.android.com/studio
  - Install on the system


Android Studio Setup

Choose Components

Choose which features of Android Studio you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- [✓] Android Studio
- [✓] Android Virtual Device

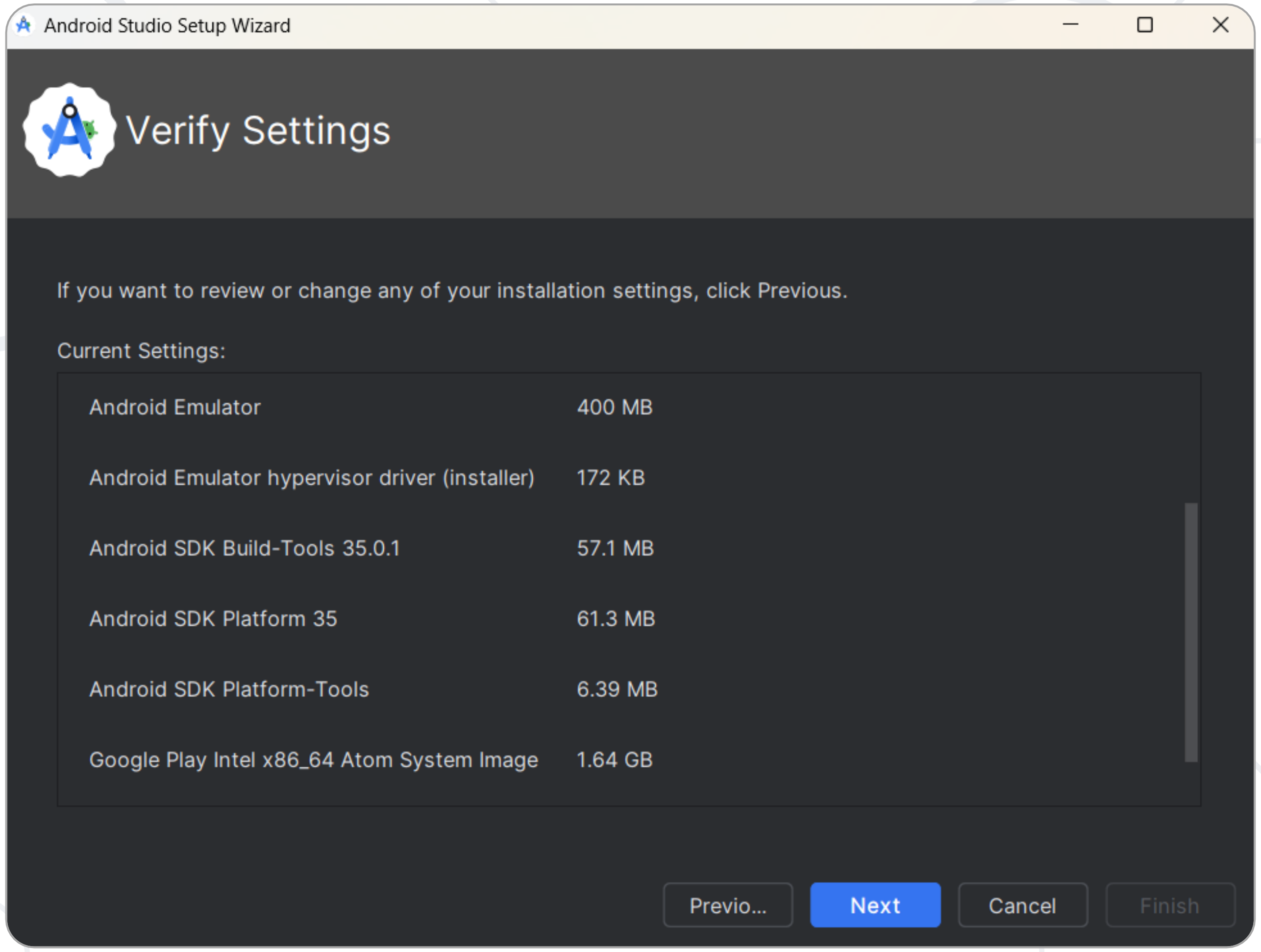
Description

Position your mouse over a component to see its description.

Space required: 3.6GB

Setup Android SDK and SDK Tools

- **Start Android Studio** after the installation
- **Android Setup Wizard** will open, which will guide you through the process of installing **Android SDK**



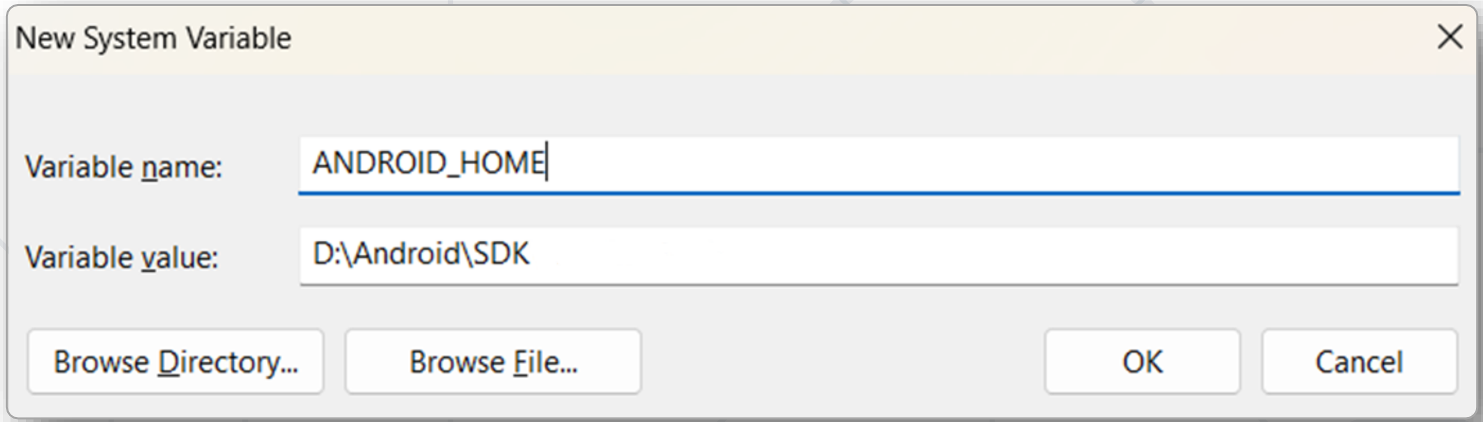
If you want to review or change any of your installation settings, click Previous.

Current Settings:

Android Emulator	400 MB
Android Emulator hypervisor driver (installer)	172 KB
Android SDK Build-Tools 35.0.1	57.1 MB
Android SDK Platform 35	61.3 MB
Android SDK Platform-Tools	6.39 MB
Google Play Intel x86_64 Atom System Image	1.64 GB

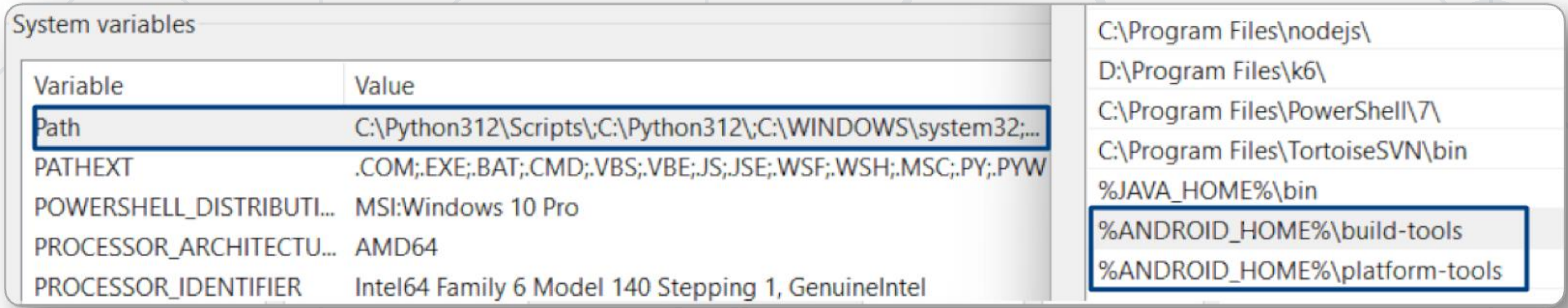
ANDROID_HOME and Path

- Set Environment Variables **ANDROID_HOME** and **Path**



Variable name: ANDROID_HOME

Variable value: D:\Android\SDK



System variables

Variable	Value
Path	C:\Python312\Scripts\;C:\Python312\;C:\WINDOWS\system32;...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
POWERSHELL_DISTRIBUTI...	MSI:Windows 10 Pro
PROCESSOR_ARCHITECTU...	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 140 Stepping 1, GenuineIntel

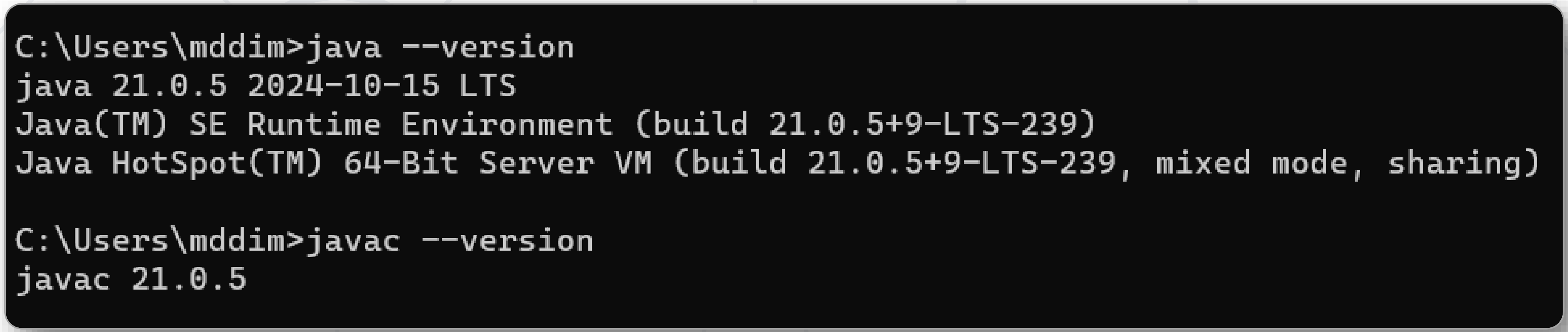
```
C:\Program Files\nodejs\
D:\Program Files\k6\
C:\Program Files\PowerShell\7\
C:\Program Files\TortoiseSVN\bin
%JAVA_HOME%\bin
%ANDROID_HOME%\build-tools
%ANDROID_HOME%\platform-tools
```

Java JDK

- Check if **Java JDK** is present:

```
java --version and javac --version
```

- If not, download & install Java JDK
  - https://www.oracle.com/java/technologies/downloads/#java21
- Verify the installation:

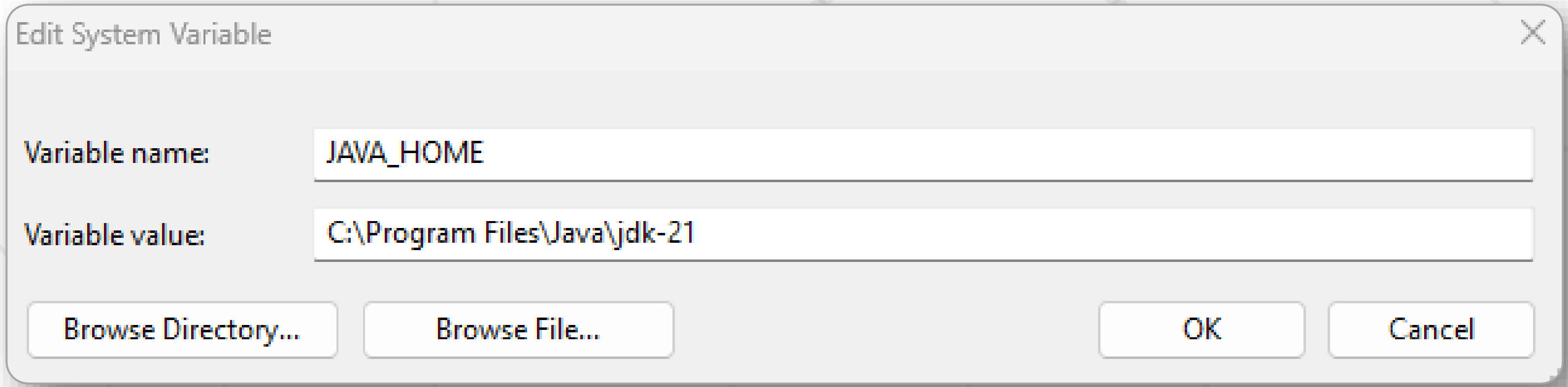


```
C:\Users\mddim>java --version
java 21.0.5 2024-10-15 LTS
Java(TM) SE Runtime Environment (build 21.0.5+9-LTS-239)
Java HotSpot(TM) 64-Bit Server VM (build 21.0.5+9-LTS-239, mixed mode, sharing)

C:\Users\mddim>javac --version
javac 21.0.5
```

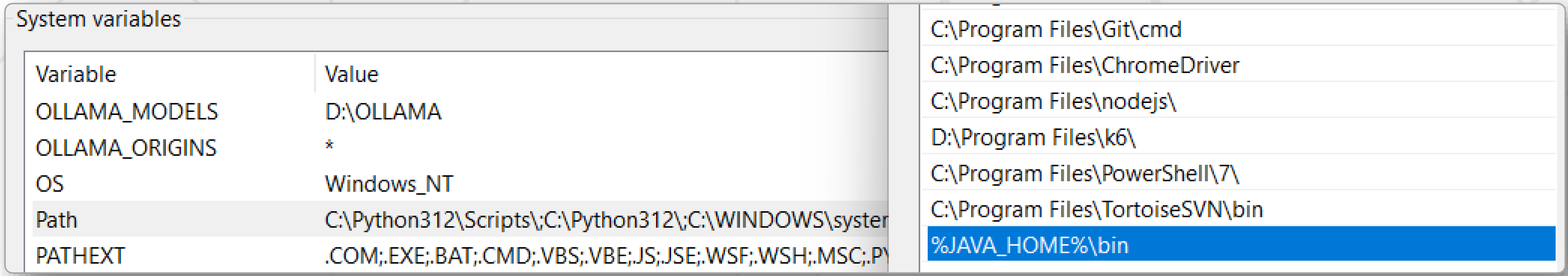
JAVA_HOME and Path

- Setup Environment variables **JAVA_HOME** and **Path**



Edit System Variable

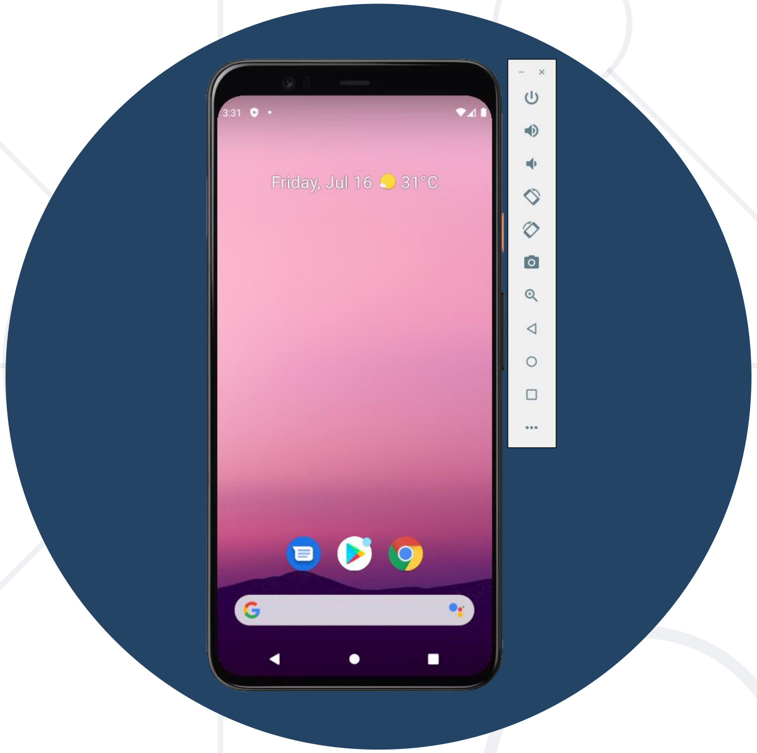
Variable name:	JAVA_HOME
Variable value:	C:\Program Files\Java\jdk-21



System variables

Variable	Value
OLLAMA_MODELS	D:\OLLAMA
OLLAMA_ORIGINS	*
OS	Windows_NT
Path	C:\Python312\Scripts\;C:\Python312\;C:\WINDOWS\syster
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.P\

```
C:\Program Files\Git\cmd
C:\Program Files\ChromeDriver
C:\Program Files\nodejs\
D:\Program Files\k6\
C:\Program Files\PowerShell\7\
C:\Program Files\TortoiseSVN\bin
%JAVA_HOME%\bin
```



Android Virtual Device (ADV)

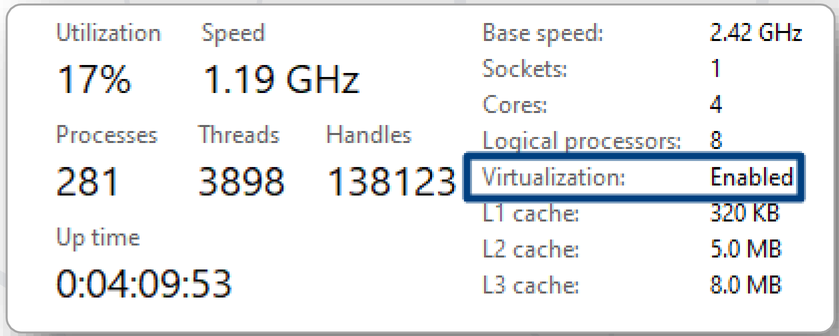
Running Android OS and Apps on Your Laptop

VT-x Virtualization

- To run a virtual device, your laptop should enable the **VT-x virtualization**, also known as Intel® Virtualization Technology
- Verify VT-x is enabled on Windows:
  - Open Task Manager
  - Performance tab
  - Look for the Virtualization section → Enabled
- Learn more at:
  - *https://www.thewindowsclub.com/disable-hardware-virtualization-in-windows-10*

VT-x Virtualization

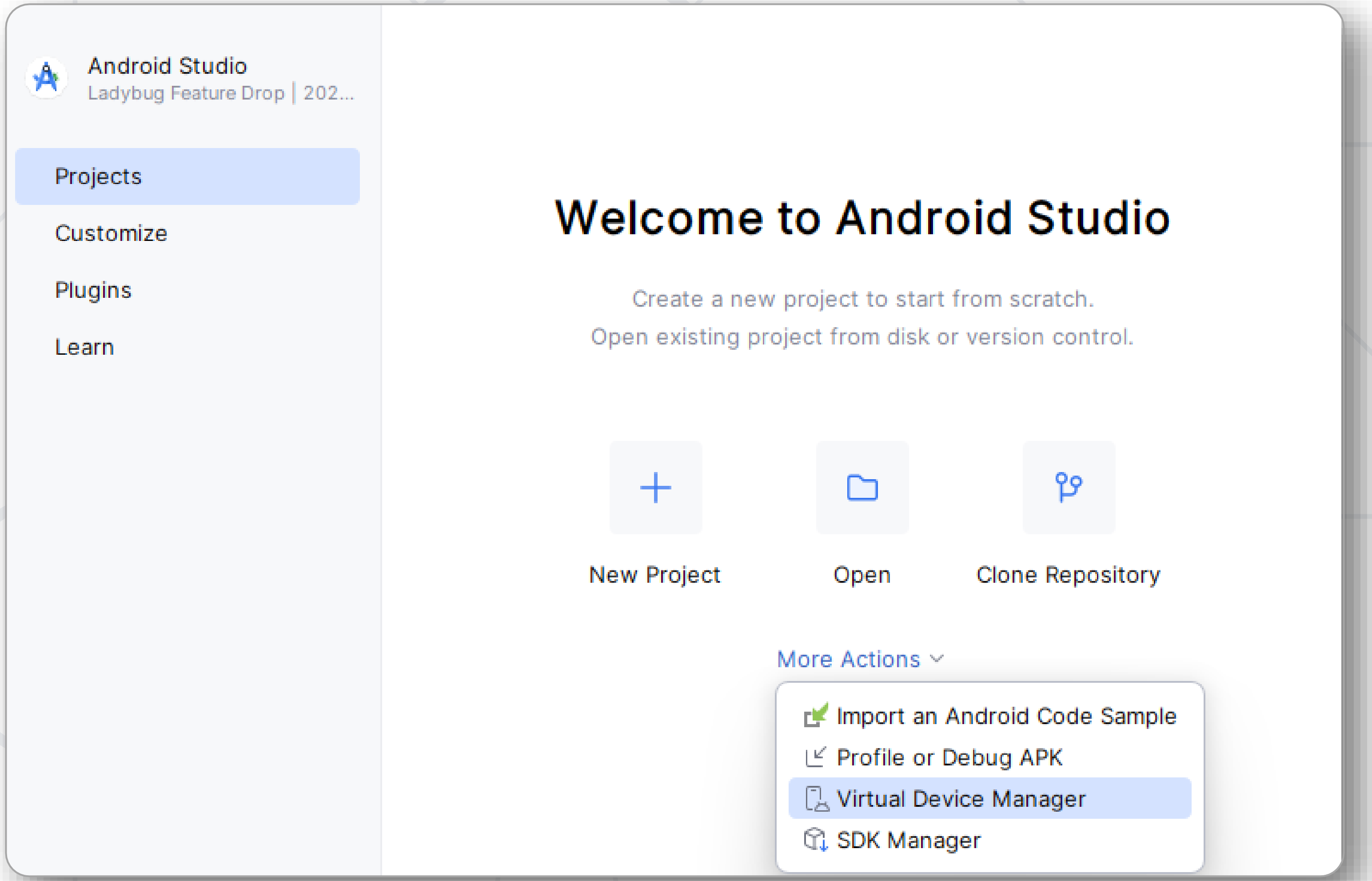
- To run a virtual device, your laptop should enable the **VT-x virtualization**, also known as Intel® Virtualization Technology
- Verify VT-x is enabled on Windows:
  - Open Task Manager
  - Performance tab
  - Look for the Virtualization section → Enabled
- Learn more at:
  - [*https://www.thewindowsclub.com/disable-hardware-virtualization-in-windows-10*](https://www.thewindowsclub.com/disable-hardware-virtualization-in-windows-10)



```
Utilization   Speed                 Base speed:          2.42 GHz
17%           1.19 GHz              Sockets:             1
                                    Cores:               4
Processes     Threads   Handles     Logical processors:  8
281           3898      138123      Virtualization:      Enabled
                                    L1 cache:            320 KB
Up time                             L2 cache:            5.0 MB
0:04:09:53                          L3 cache:            8.0 MB
```

Virtual Device Manager

- Run Android Studio → **Virtual Device Manager**

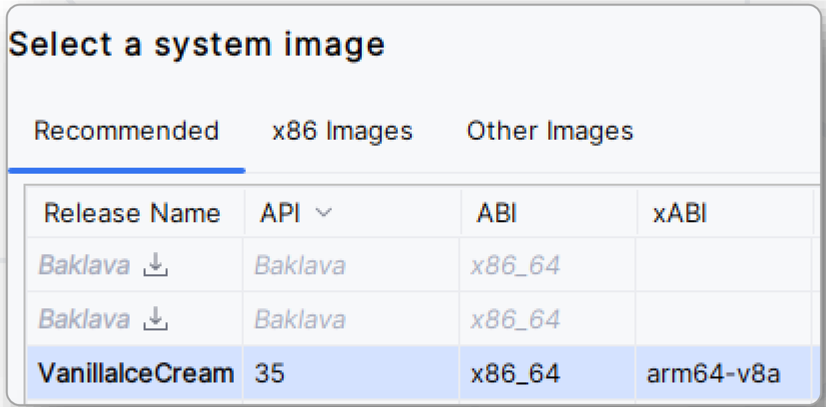


Welcome to Android Studio

Create a new project to start from scratch.
Open existing project from disk or version control.

Create Virtual Device

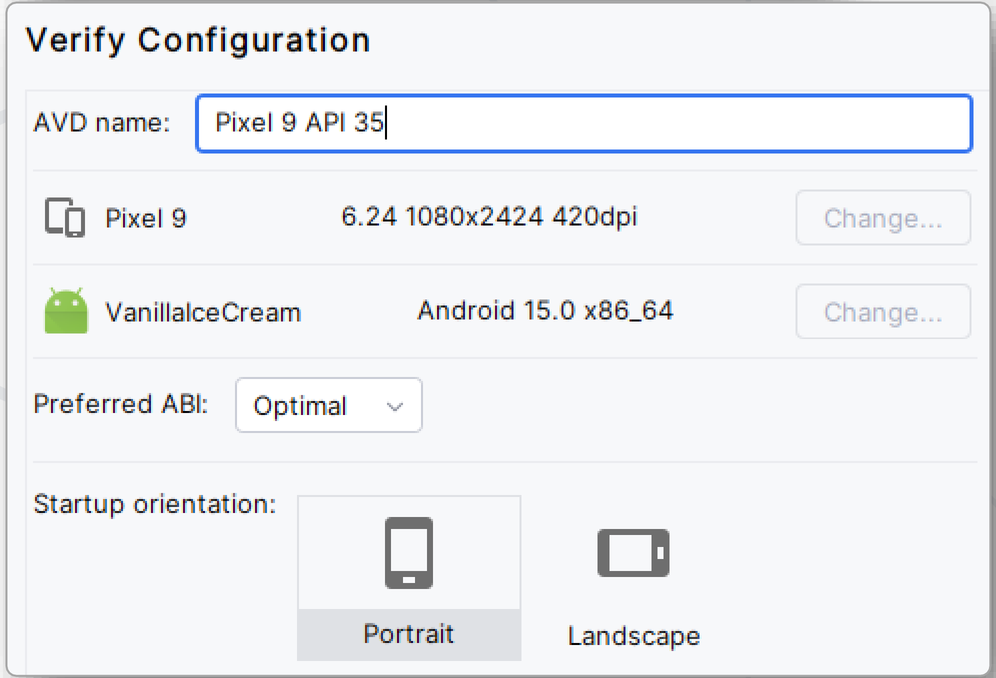
- **Virtual Device Configuration**
  - Select Hardware
  - Install System Image
  - Select System Image
  - Verify Configuration
  - Give Name to your AVD (optional)



Select a system image

Recommended | x86 Images | Other Images

Release Name	API	ABI	xABI
Baklava	Baklava	x86_64	
Baklava	Baklava	x86_64	
VanillaIceCream	35	x86_64	arm64-v8a



Verify Configuration

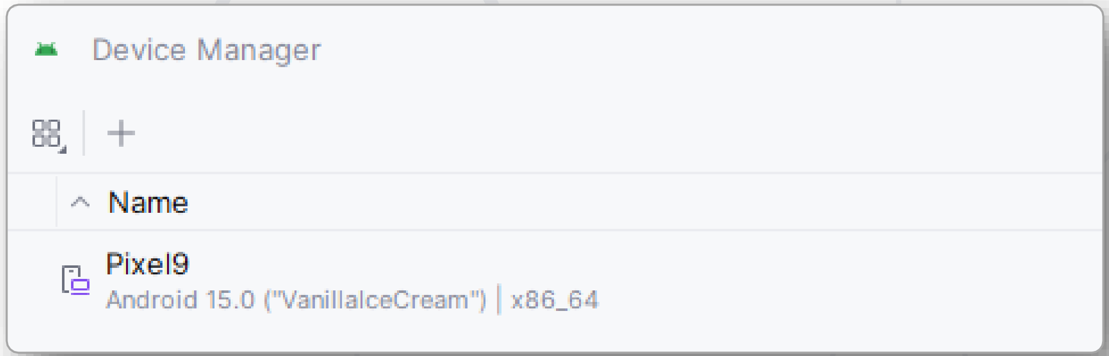
AVD name: Pixel 9 API 35

Pixel 9 — 6.24 1080x2424 420dpi — Change...

VanillaIceCream — Android 15.0 x86_64 — Change...

Preferred ABI: Optional

Startup orientation: Portrait | Landscape



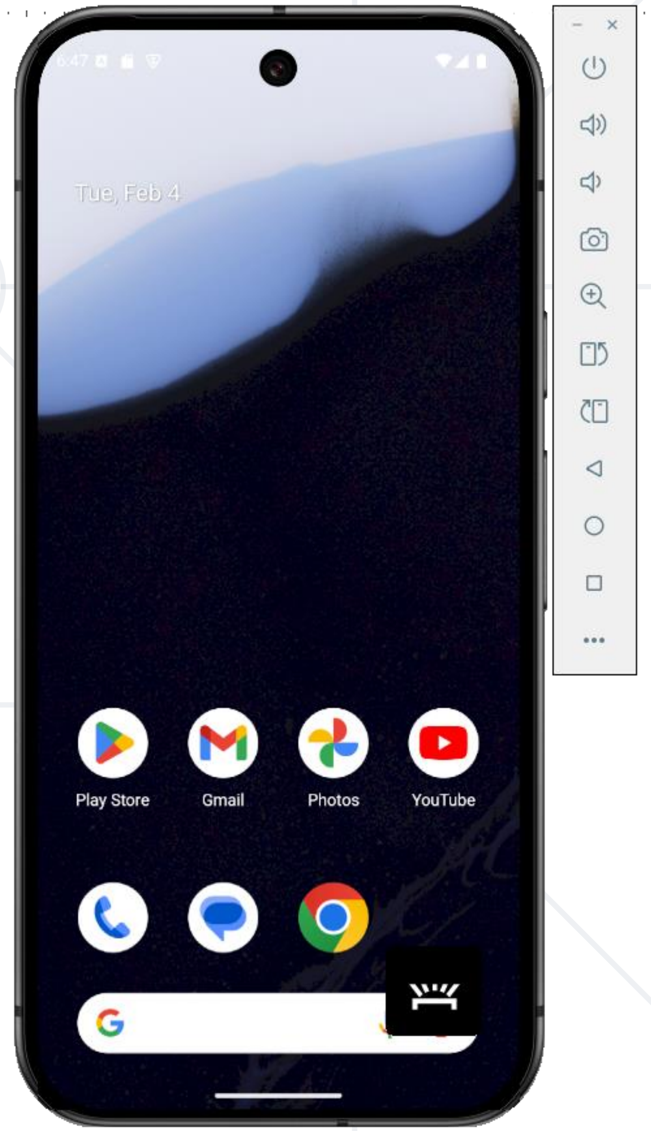
Device Manager

Name

Pixel9
Android 15.0 ("VanillaIceCream") | x86_64

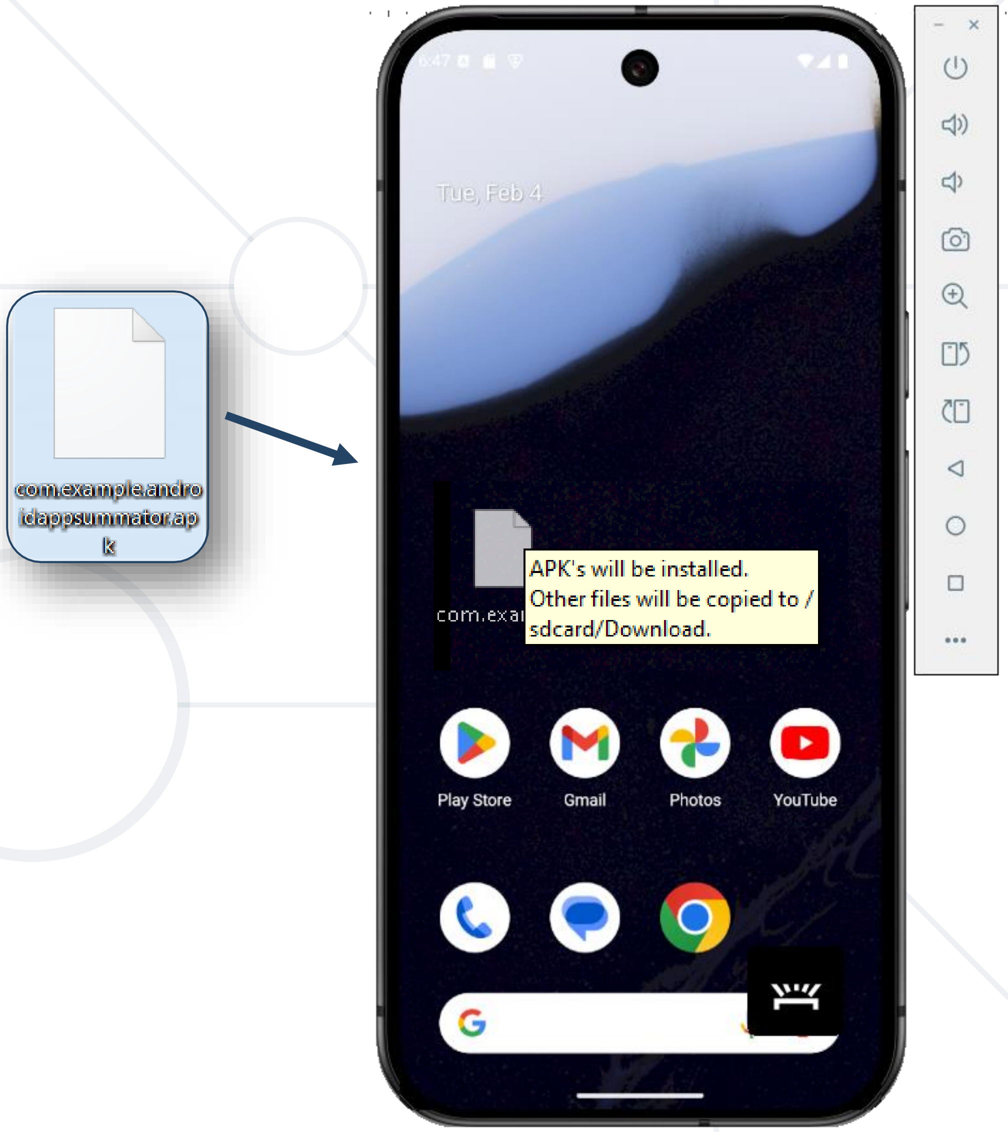
Start Virtual Device

- The virtual device in Android Studio **emulates a real smartphone**, allowing exploration and interact with Android features
- **Navigate** through menus, launch apps, and test different functionalities



The Android App for Testing

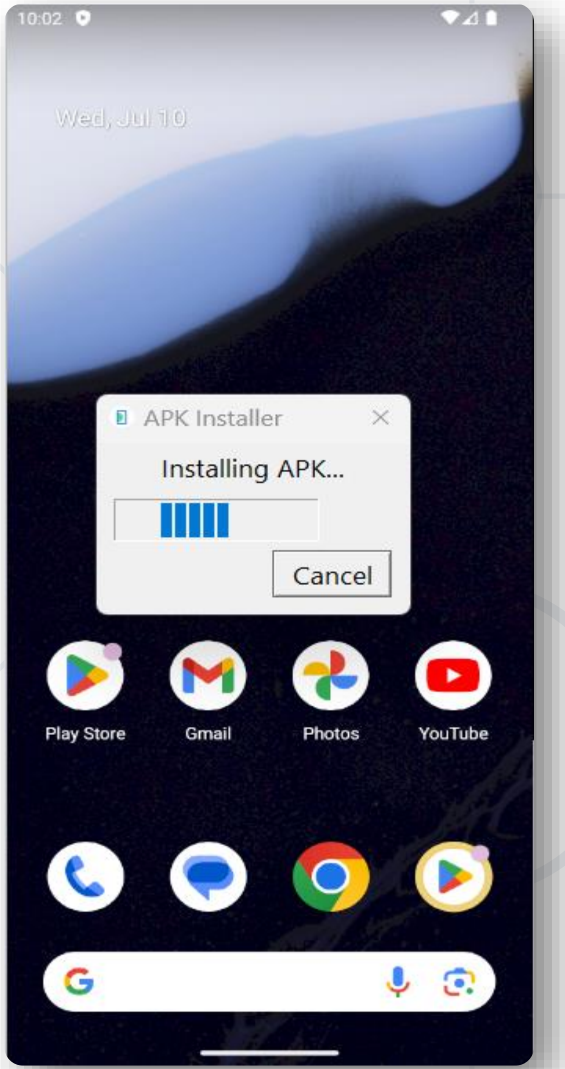
- A simple Android app for testing:
  - https://github.com/nakov/AndroidApp-Summator
- Download the **.apk** file (Android app package)
  - https://github.com/nakov/AndroidApp-Summator/releases
- **Install** and **run** it in the Emulator (use drag & drop)

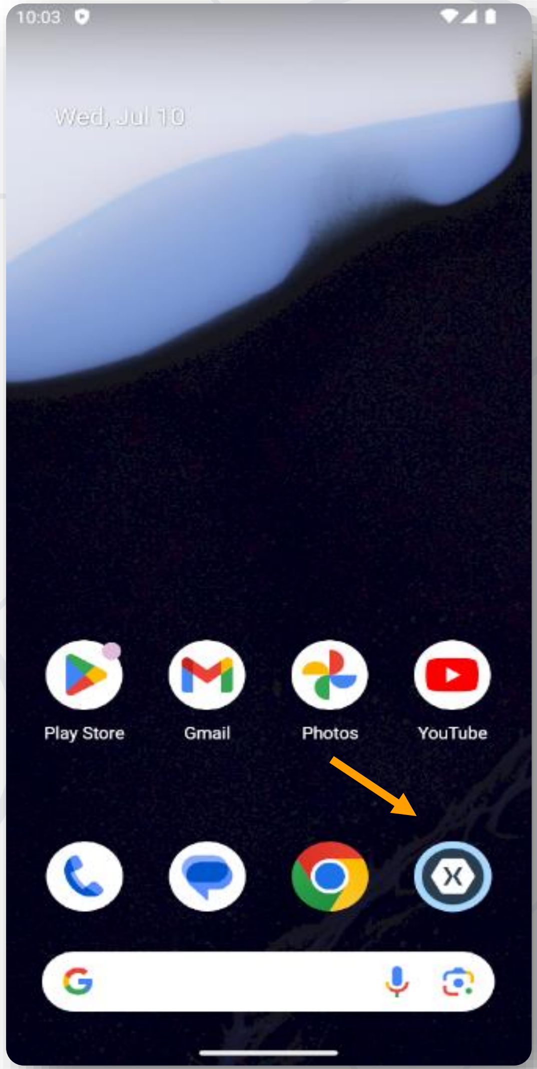


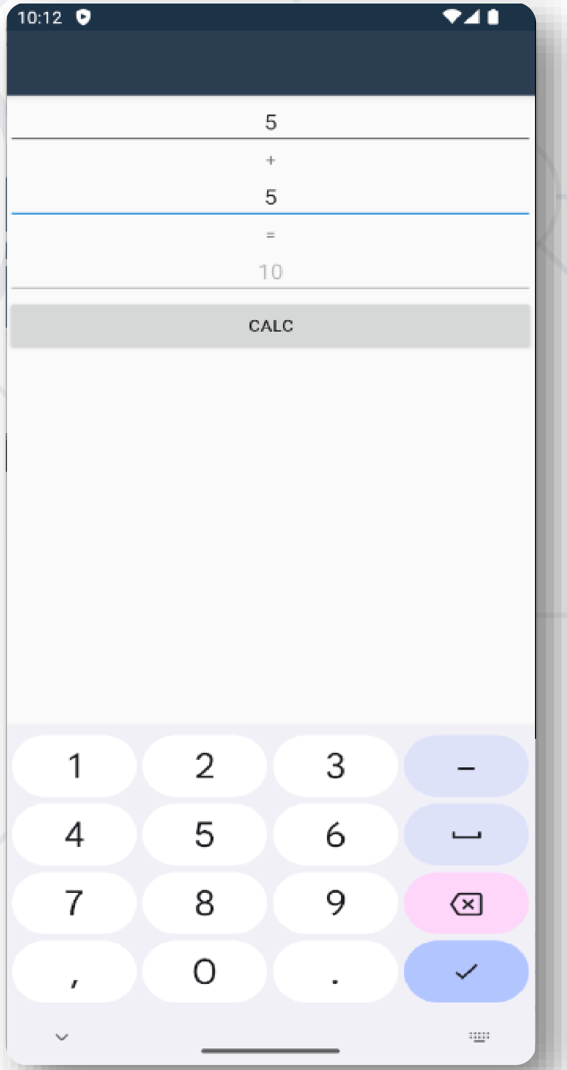
APK's will be installed.
Other files will be copied to /sdcard/Download.

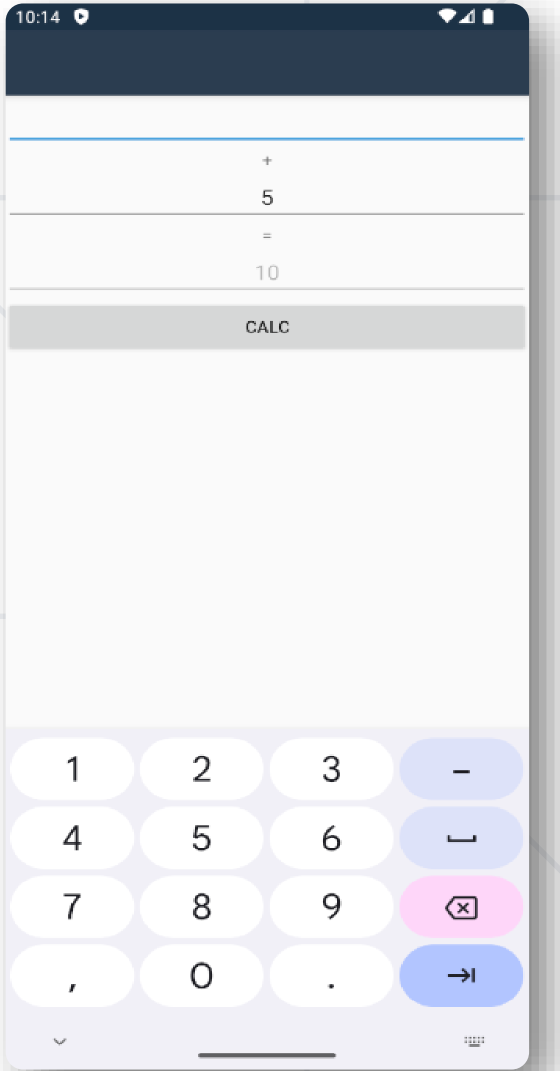
Run the Sample Android App

- **Run** the sample "**Summator**" Android app to ensure it works



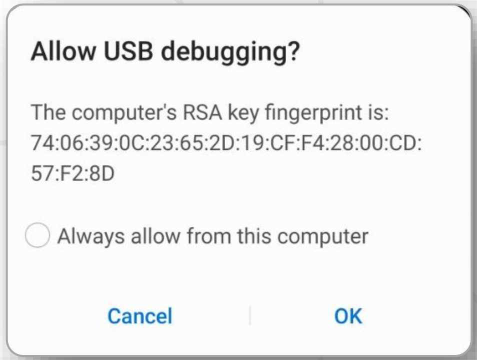






Using Physical Device

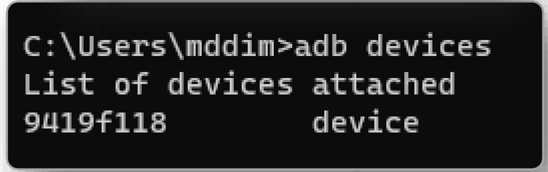
- On your Android device, go to **Settings**
- Scroll down and find **About Phone** (or similar)
- Tap on **Build Number** or **Software version** multiple times until you see a message indicating that **Developer Mode** is enabled
- Go back to the main Settings screen and find Developer Options
- Enable **USB Debugging**
- Connect Your Android Device to Your Computer
- If prompted allow connection
- Verify the device is recognized by your system CMD → **`adb devices`**



Allow USB debugging?

The computer's RSA key fingerprint is:
74:06:39:0C:23:65:2D:19:CF:F4:28:00:CD:57:F2:8D

◯ Always allow from this computer



```
C:\Users\mddim>adb devices
List of devices attached
9419f118        device
```

Appium Inspector

Inspect, Interact, and Debug Mobile UI Elements

Appium Inspector Overview

- A **graphical tool** designed to **inspect** and **interact** with mobile apps
- Allows users to inspect **UI elements** of mobile apps for both Android and iOS platforms
- Enables interaction with UI elements to **test** their **behavior**
- **Displays attributes** and **properties** of UI elements (e.g., resource ID, text, class)
- Assists in **generating XPath expressions** for locating elements
- Provides a **visual representation** of the app's **UI hierarchy**
- Manages **Appium sessions** to connect and disconnect from devices or emulators

Download Appium Inspector

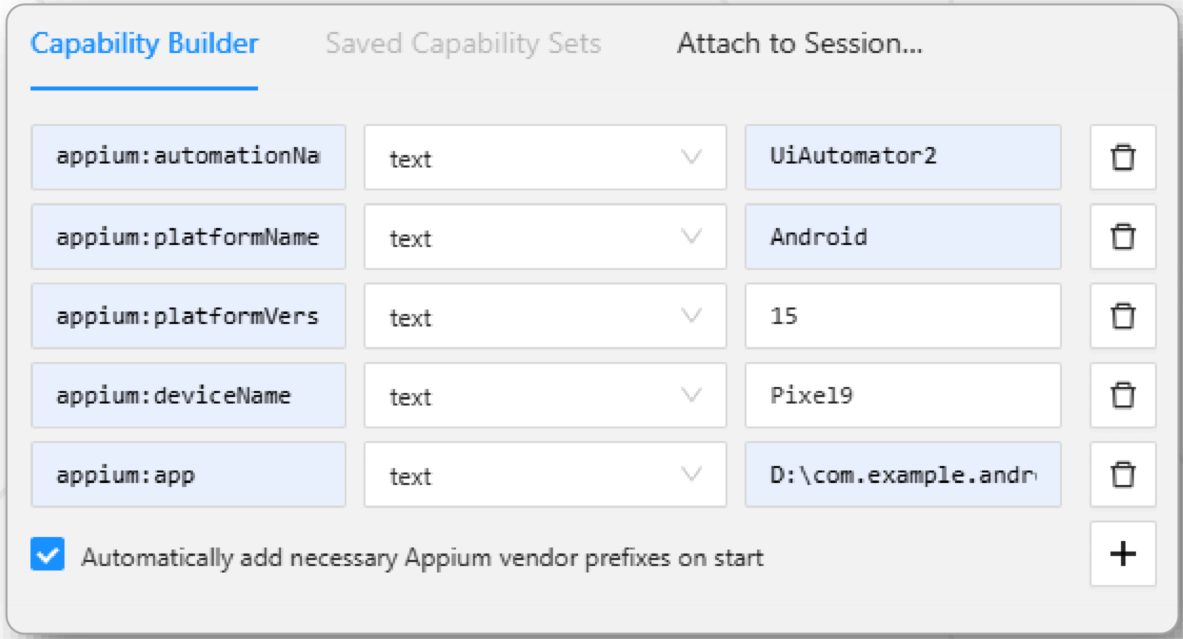
- Install Appium Inspector here: https://github.com/appium/appium-inspector/releases
- Or use the web version here: https://inspector.appiumpro.com/
- You also can check the documentation https://appium.github.io/appium-inspector/latest/

Configure Appium Inspector

- **Start** Appium Server:
  - Appium Inspector Desktop App → CMD → **`appium`**
  - Appium Inspector Web → CMD → **`appium --allow-cors`** (allows Cross-Origin Resource Sharing)
- Provide the **host and port of Appium server** in Appium Inspector
- Add the **Desired Capabilities** of the mobile **device** or emulator **connected** to the system

Configure Appium Inspector

- **automationName** - driver for Android or iOS
- **platformName** - get it using command **appium driver list**
- **platformVersion** - get it using command
  adb shell getprop ro.build.version.release
- **appium:deviceName** –
  get it using command
  adb devices
- **appium:app** – path to the .apk file for testing on your computer

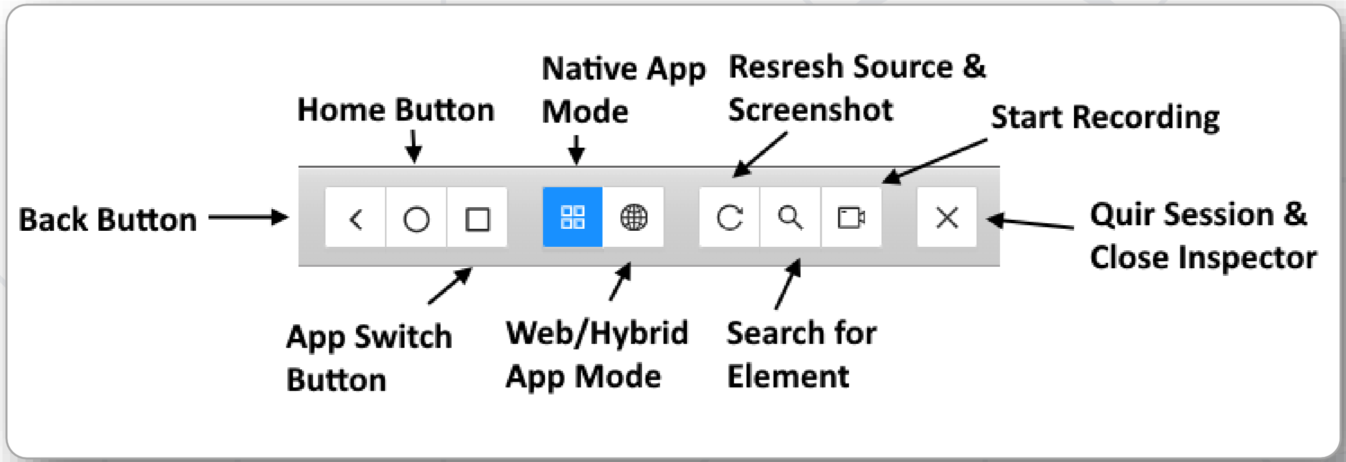


appium:automationNa	text	UiAutomator2
appium:platformName	text	Android
appium:platformVers	text	15
appium:deviceName	text	Pixel9
appium:app	text	D:\com.example.andr

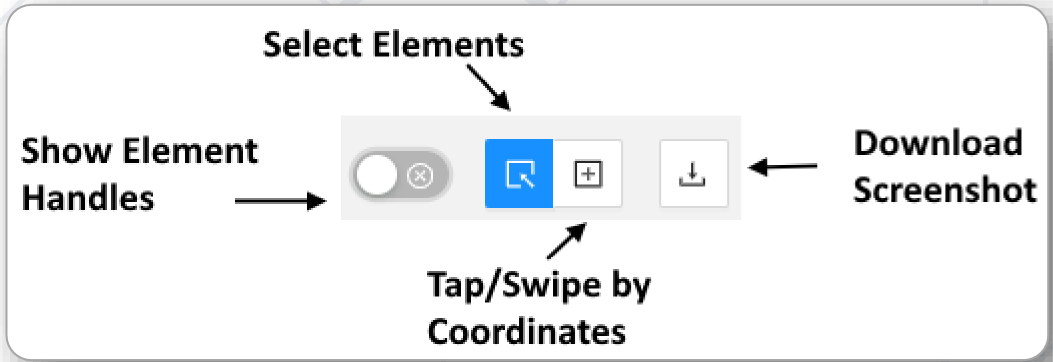
✓ Automatically add necessary Appium vendor prefixes on start

Appium Inspector Menu

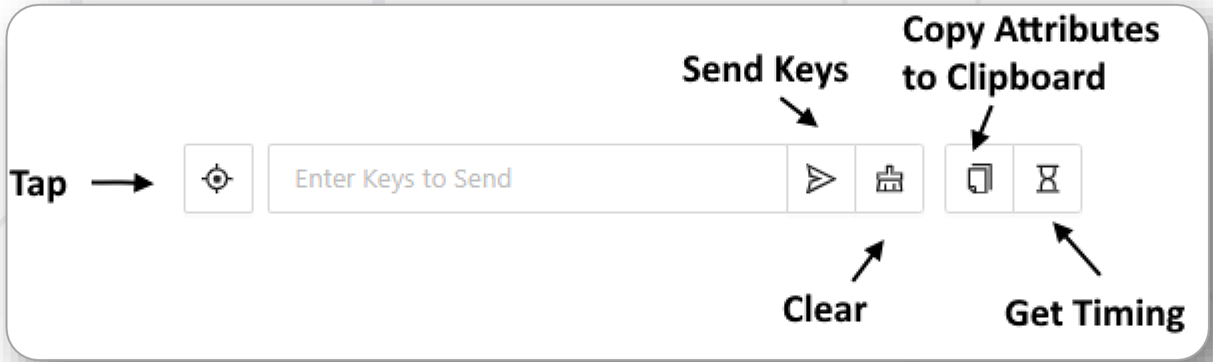
- Main Toolbar



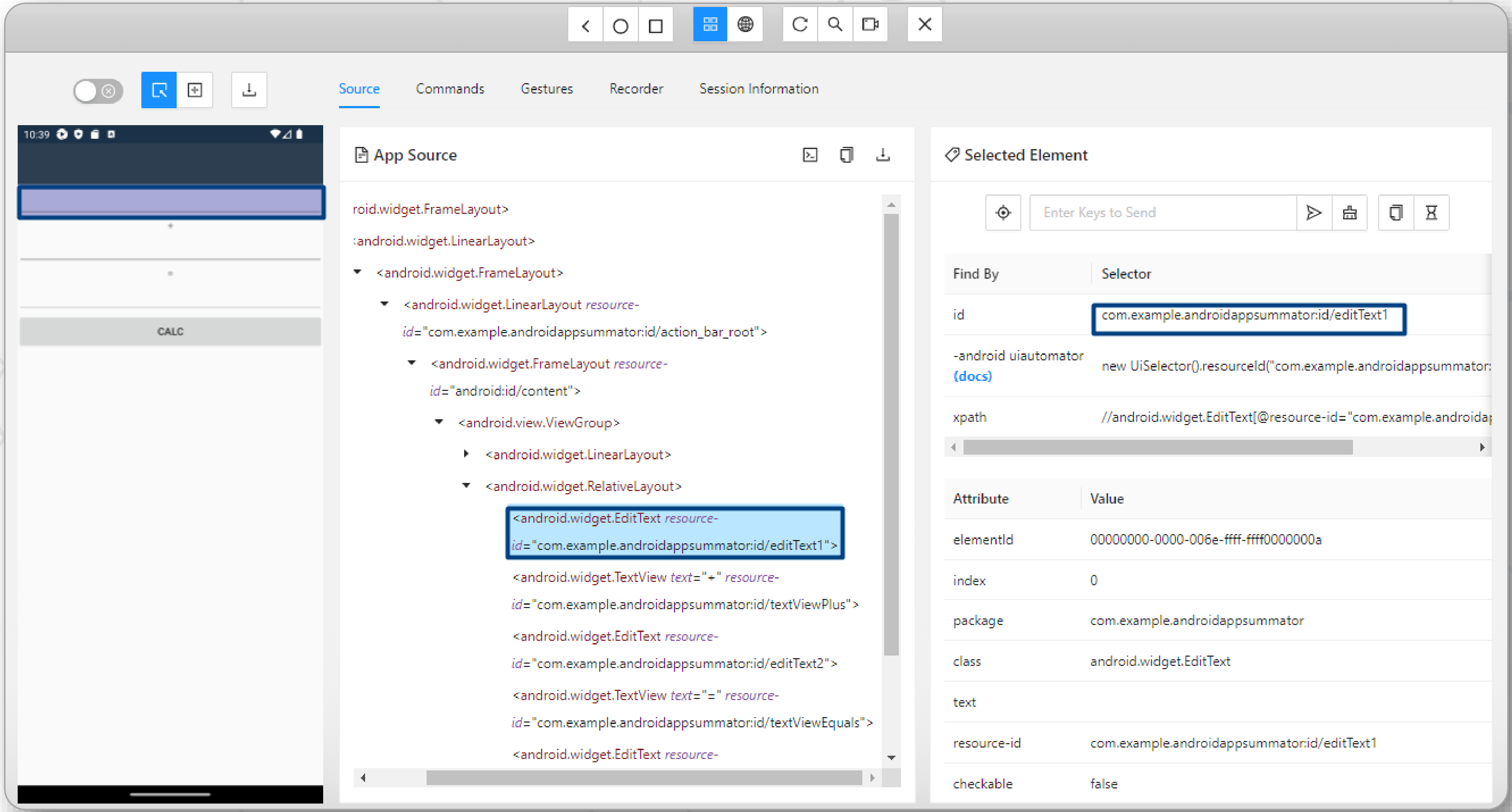
- Screenshot Panel



- Selected Element Panel



Select and View Element Attributes



Source · Commands · Gestures · Recorder · Session Information

App Source

```
roid.widget.FrameLayout>
:android.widget.LinearLayout>
▾ <android.widget.FrameLayout>
  ▾ <android.widget.LinearLayout resource-id="com.example.androidappsummator:id/action_bar_root">
    ▾ <android.widget.FrameLayout resource-id="android:id/content">
      ▾ <android.view.ViewGroup>
        ▸ <android.widget.LinearLayout>
        ▾ <android.widget.RelativeLayout>
            <android.widget.EditText resource-id="com.example.androidappsummator:id/editText1">
            <android.widget.TextView text="+" resource-id="com.example.androidappsummator:id/textViewPlus">
            <android.widget.EditText resource-id="com.example.androidappsummator:id/editText2">
            <android.widget.TextView text="=" resource-id="com.example.androidappsummator:id/textViewEquals">
            <android.widget.EditText resource-
```

Selected Element

Find By	Selector
id	com.example.androidappsummator:id/editText1
-android uiautomator (docs)	new UiSelector().resourceId("com.example.androidappsummator:
xpath	//android.widget.EditText[@resource-id="com.example.androida

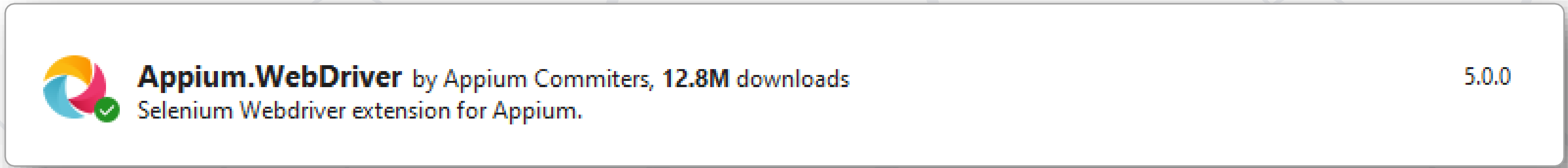
Attribute	Value
elementId	00000000-0000-006e-ffff-ffff0000000a
index	0
package	com.example.androidappsummator
class	android.widget.EditText
text	
resource-id	com.example.androidappsummator:id/editText1
checkable	false

Appium Tests for Android

Demo

Set up Appium for Android with C#

- Create a new **NUnit** project with C# in VS
- Install **Appium.WebDriver** from NuGet



Appium.WebDriver by Appium Commiters, **12.8M** downloads
Selenium Webdriver extension for Appium.
5.0.0

- Start Appium Server
- Start Emulator / Physical Device
- Make sure that the app for testing is installed
- Locate elements needed via Appium Inspector
- Write and run tests in Visual Studio

Summary

- Mobile Testing - Evaluates functionality, usability, performance, and compatibility of mobile apps
- Appium - Open-source tool for testing native, hybrid, and mobile web apps
- Appium Setup and Configuration
- Using Android Emulator
- Appium Inspector - Interacting with mobile app UI elements
- Writing and Running Tests

Software University

Questions?

SoftUni

Software University

SoftUni Creative

SoftUni Digital

SoftUni Foundation

SoftUni Kids

СофтУни БУДИТЕЛ

Finance Academy

Diamond Partners

Software University

SmartIT

Coca-Cola HBC Bulgaria

SUPER HOSTING .BG

createX

Postbank

top EMPLOYER България Bulgaria 2024 FOR A BETTER WORLD OF WORK

DRAFT KINGS

THE CROWN IS YOURS

VIVACOM

INDEAVR
Serving the high achievers

AMBITIONED

PHAR VISION

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg, about.softuni.bg
- Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity

License