1. Introduction and Purpose

- All mobile device testing automation should be a part of the QA process
- QA should validate all required mobile app changes before it pushed to production
- QA should be able to run automation script on Emulator (required for support application) and check release changes and regression on any test cases and open issues (from customer support)

2. Hardware & Software Requirements

2.1) First, you should have access to a **VDI** where you will work on the tests. You may find instructions on how to get one here: https://associated-bank.atlassian.net/wiki/spaces/EQ/pages/982876229/VDI+Setup+Guide#VDI-Setup-%2F-Submit-Requests

2.2) Once you are able to log into the VDI as your user, you should begin installing **the necessary software** for mobile automation:

- ***Java***: https://www.oracle.com/java/technologies/downloads/#java11-windows
The code can be written in any languages supported by Webdriver: Java, C3, JavaScript, Python, Ruby. We will use Appium on Java.

- ***Android Studio*** - https://developer.android.com/studio (download Android Studio Electric Eel)
The easiest way to get started with your android test automation setup is to download Android Studio. Through Android Studio, it will be easier for us to manage the SDKs, Android Virtual Devices, and other android components necessary for android test automation.

- ***Node.js*** - https://nodejs.org/dist/v18.15.0/node-v18.15.0-x64.msi (download Windows Installer (.msi) for 64-bit)
Node.js is a software required for executing the different node modules like Appium.

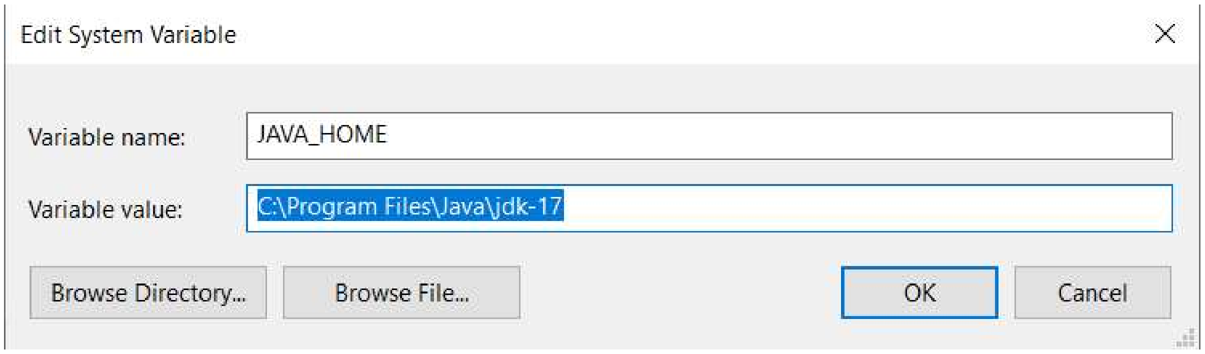
- Download **IDE (Eclipse) -** the development environment where tests are written: https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/2023-03/R/eclipse-java-2023-03-R-win32-x86_64.zip

- Download **Appium Inspector** https://github.com/appium/appium-inspector/releases/download/v2023.3.1/Appium-Inspector-windows-2023.3.1.exe
It is a handy tool for QA engineers, powered by a (separately installed) Appium server, who want to automate mobile applications.

- **GIT** - Version control software. This is how we store/update our code.
Note: *You will also need a GitHub account to work with the repository. Create one if you do not have one and file a ticket in Cherwell to gain access to the company GitHub project.*

3. Initial Local Environment Setup

3.1) Once you have Java in your system, you must set **system variables.**
Go to C drive, Program Files folder, Java folder, JDK folder – this is the Java home directory.
Copy this path (i.e., C/Program Files/Java/JDK), then go to Control Panel > Advanced System Settings > Environment Variables > click New on System Variables section.
Set variable name as **JAVA_HOME**, and in variable value field put the path to JDK folder.
It will look like that:


Edit System Variable

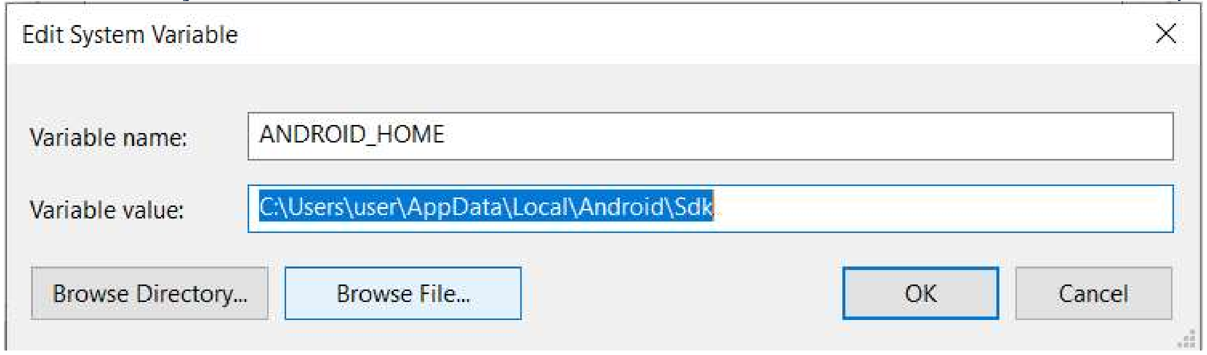
Variable name: JAVA_HOME

Variable value: C:\Program Files\Java\jdk-17

3.2) To work with android apps we need Android SDK package.

Note: *Once you downloaded Android Studio, open it, and make sure: do not import any settings; select standard installation; grab and save SDK folder path (you will use it when set system variables).*

Go back to System variables and create **ANDROID_HOME** variable with SDK folder path value.

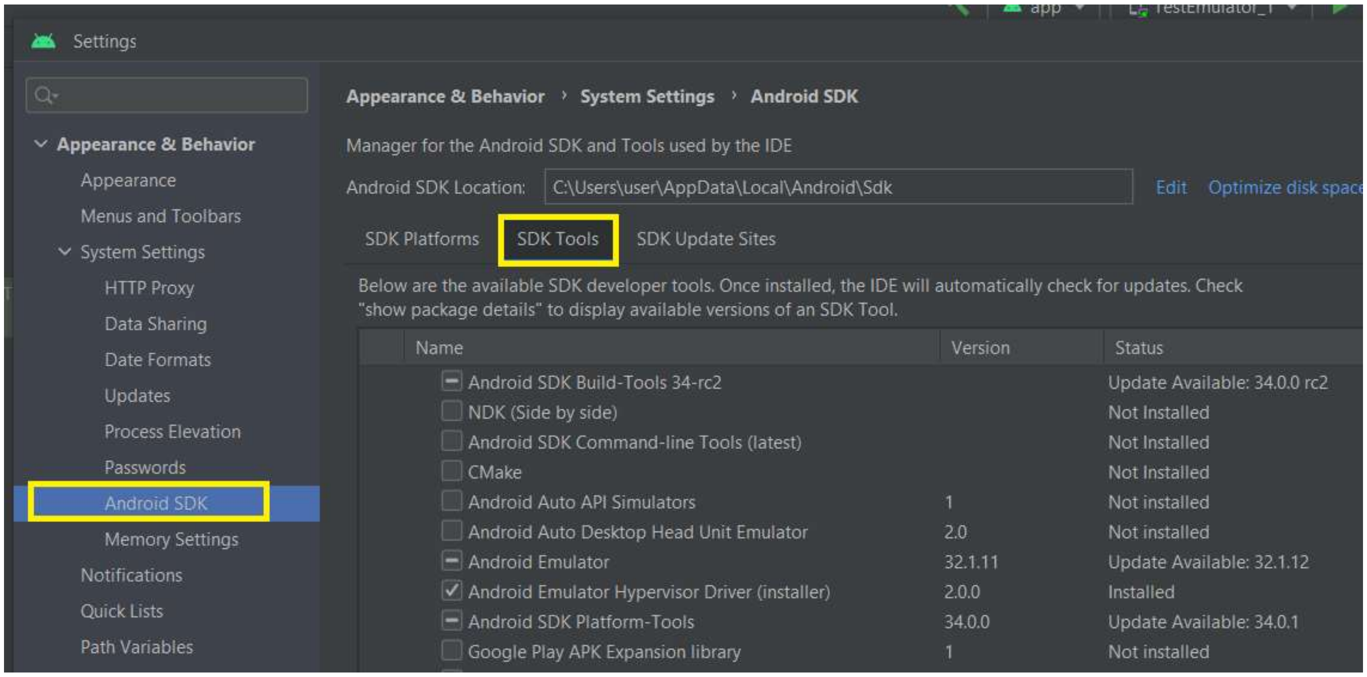

Edit System Variable

Variable name: ANDROID_HOME

Variable value: C:\Users\user\AppData\Local\Android\Sdk

In SDK folder there should be the **Tool** folder. To make this folder visible, do this:
- launch Android Studio > select Basic or Empty Activity, then click Next and Finish. That is how you will get the homepage and will be able to change settings. Wait until the build got finished.

- Open **Tools** menu option on top > click SDK Manager. On opened Settings window go to Android SDK > **SDK Tools** tab:


Settings

Appearance & Behavior
- Appearance
- Menus and Toolbars
- System Settings
  - HTTP Proxy
  - Data Sharing
  - Date Formats
  - Updates
  - Process Elevation
  - Passwords
  - Android SDK
  - Memory Settings
- Notifications
- Quick Lists
- Path Variables

Appearance & Behavior › System Settings › Android SDK

Manager for the Android SDK and Tools used by the IDE

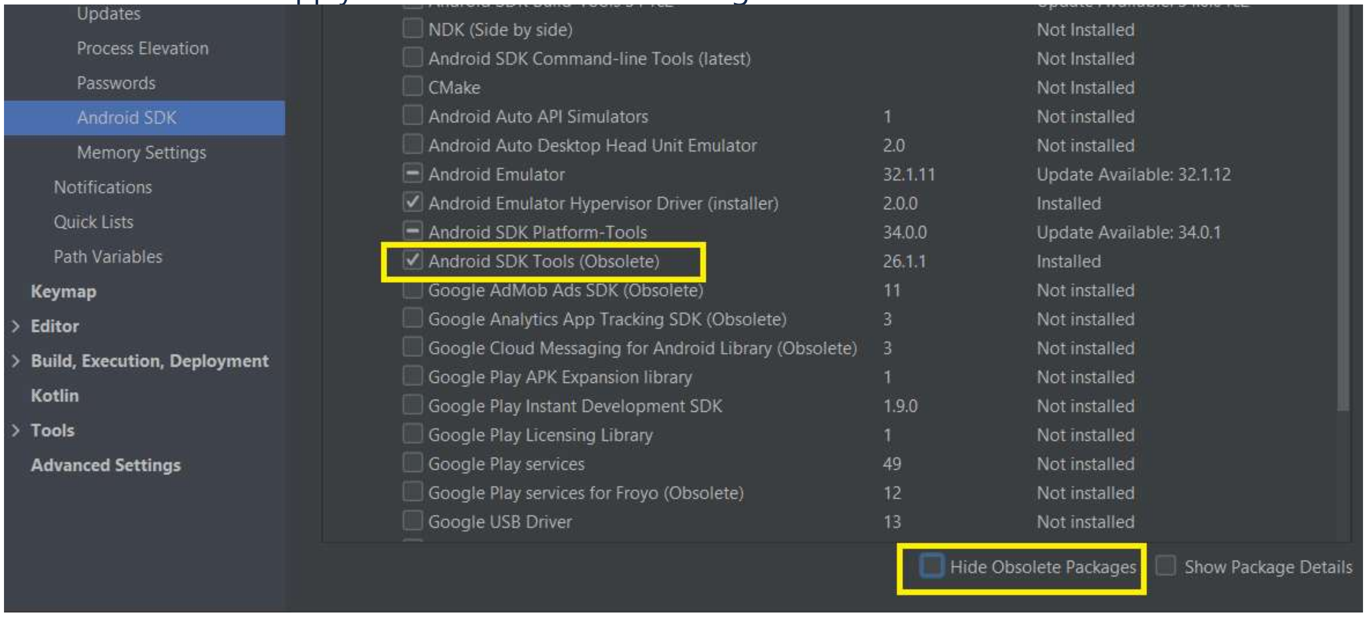
Android SDK Location: C:\Users\user\AppData\Local\Android\Sdk Edit Optimize disk space

SDK Platforms | SDK Tools | SDK Update Sites

Below are the available SDK developer tools. Once installed, the IDE will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.

Name	Version	Status
▣ Android SDK Build-Tools 34-rc2		Update Available: 34.0.0 rc2
☐ NDK (Side by side)		Not Installed
☐ Android SDK Command-line Tools (latest)		Not Installed
☐ CMake		Not Installed
☐ Android Auto API Simulators	1	Not installed
☐ Android Auto Desktop Head Unit Emulator	2.0	Not installed
▣ Android Emulator	32.1.11	Update Available: 32.1.12
✓ Android Emulator Hypervisor Driver (installer)	2.0.0	Installed
▣ Android SDK Platform-Tools	34.0.0	Update Available: 34.0.1
☐ Google Play APK Expansion library	1	Not installed

Uncheck **Hide Obsolete Packages** checkbox on the bottom and check **Android Obsolete Tools** checkbox and click Apply. Wait until the installation got finished.

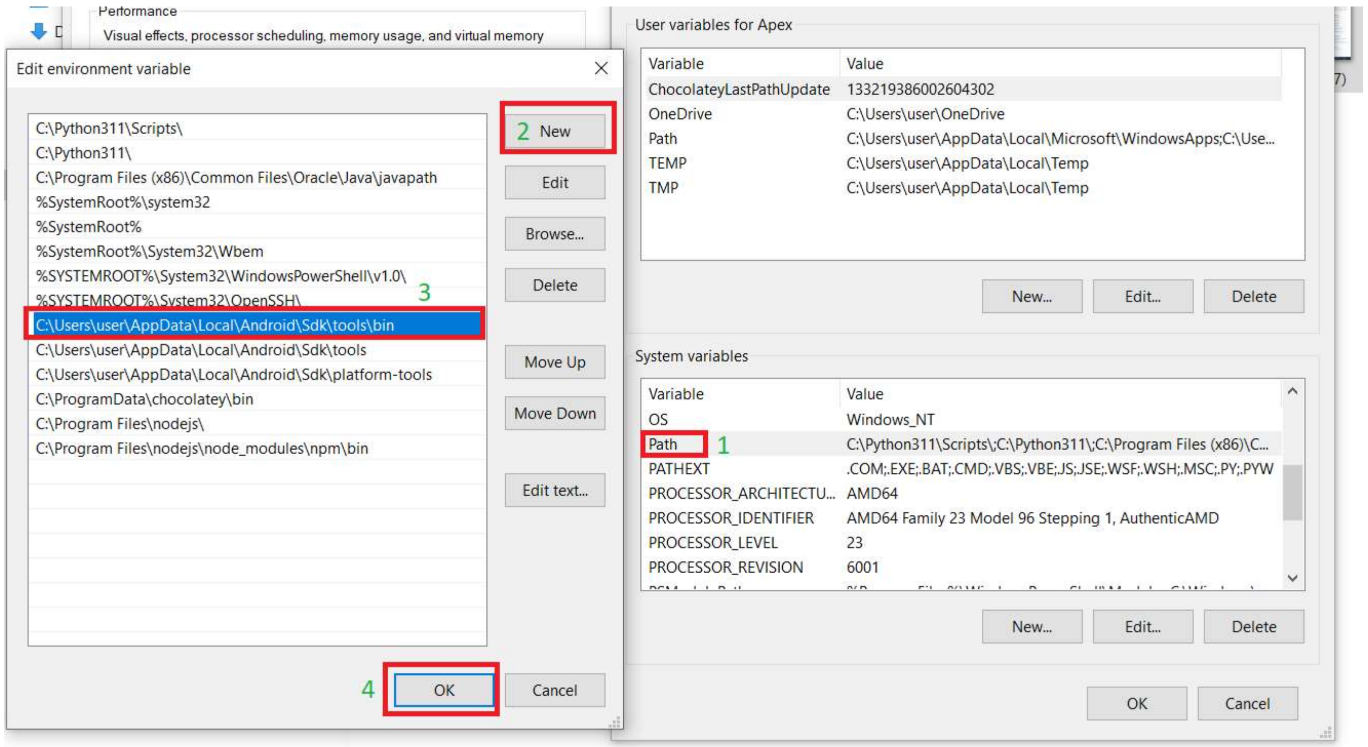

- Updates
- Process Elevation
- Passwords
- Android SDK
- Memory Settings
- Notifications
- Quick Lists
- Path Variables
- Keymap
- Editor
- Build, Execution, Deployment
- Kotlin
- Tools
- Advanced Settings

Name	Version	Status
☐ NDK (Side by side)		Not Installed
☐ Android SDK Command-line Tools (latest)		Not Installed
☐ CMake		Not Installed
☐ Android Auto API Simulators	1	Not installed
☐ Android Auto Desktop Head Unit Emulator	2.0	Not installed
▣ Android Emulator	32.1.11	Update Available: 32.1.12
✓ Android Emulator Hypervisor Driver (installer)	2.0.0	Installed
▣ Android SDK Platform-Tools	34.0.0	Update Available: 34.0.1
✓ Android SDK Tools (Obsolete)	26.1.1	Installed
☐ Google AdMob Ads SDK (Obsolete)	11	Not installed
☐ Google Analytics App Tracking SDK (Obsolete)	3	Not installed
☐ Google Cloud Messaging for Android Library (Obsolete)	3	Not installed
☐ Google Play APK Expansion library	1	Not installed
☐ Google Play Instant Development SDK	1.9.0	Not installed
☐ Google Play Licensing Library	1	Not installed
☐ Google Play services	49	Not installed
☐ Google Play services for Froyo (Obsolete)	12	Not installed
☐ Google USB Driver	13	Not installed

☐ Hide Obsolete Packages ☐ Show Package Details

Now the Tools folder should be presented in SDK folder.

- Go back to SDK folder, open **Tools** folder and grab the path to **Bin** folder location. Copy that path, come back to System Variables section in the Environment Variables modal, double click on the **path** > click New > and paste the full path to the SDK\Tools\Bin folder. Click OK.

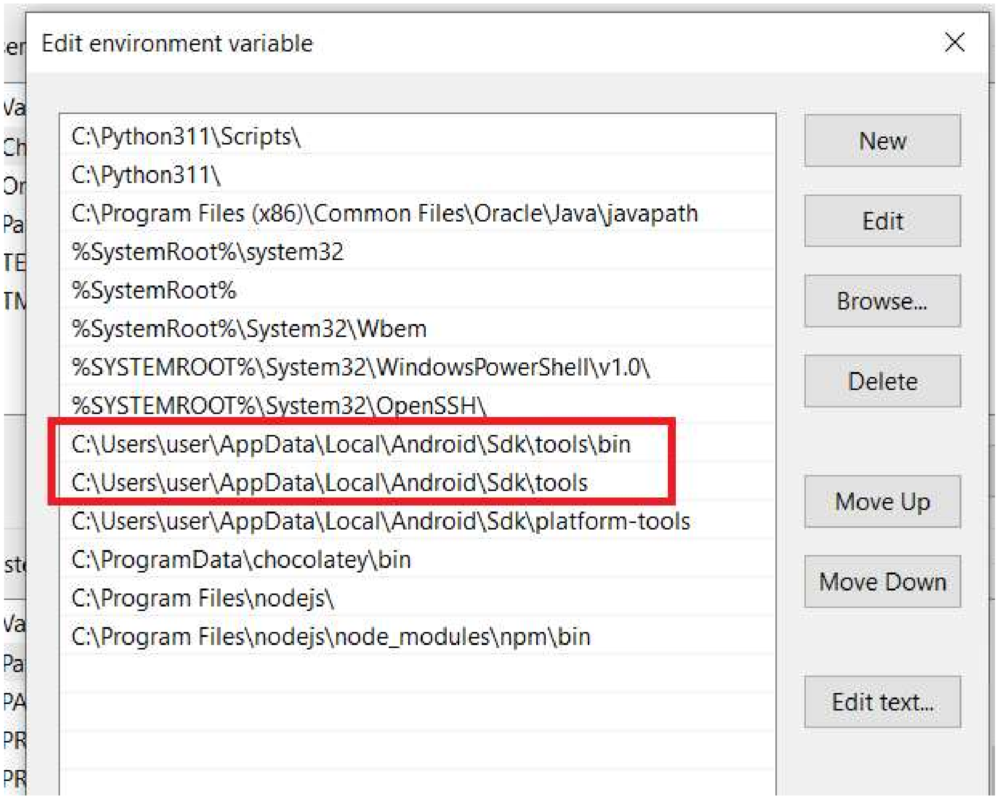


```
C:\Python311\Scripts\
C:\Python311\
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Users\user\AppData\Local\Android\Sdk\tools\bin
C:\Users\user\AppData\Local\Android\Sdk\tools
C:\Users\user\AppData\Local\Android\Sdk\platform-tools
C:\ProgramData\chocolatey\bin
C:\Program Files\nodejs\
C:\Program Files\nodejs\node_modules\npm\bin
```

Variable	Value
ChocolateyLastPathUpdate	133219386002604302
OneDrive	C:\Users\user\OneDrive
Path	C:\Users\user\AppData\Local\Microsoft\WindowsApps;C:\Use...
TEMP	C:\Users\user\AppData\Local\Temp
TMP	C:\Users\user\AppData\Local\Temp

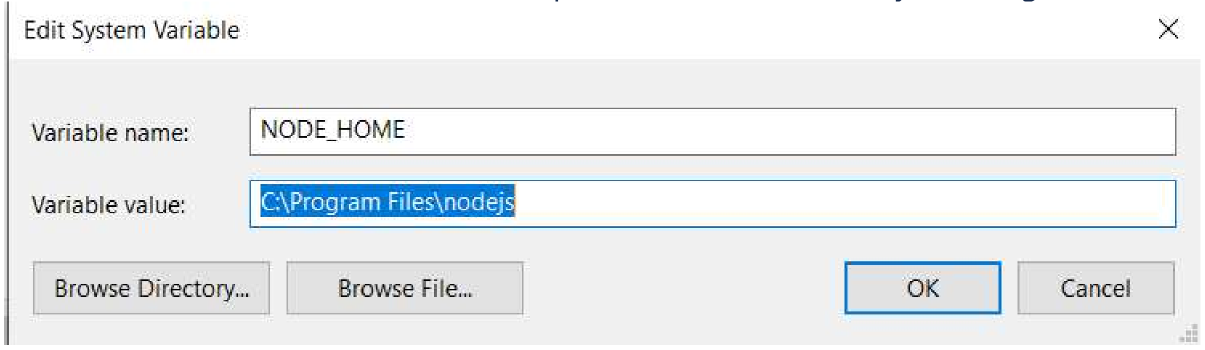
Variable	Value
OS	Windows_NT
Path	C:\Python311\Scripts\;C:\Python311\;C:\Program Files (x86)\C...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTU...	AMD64
PROCESSOR_IDENTIFIER	AMD64 Family 23 Model 96 Stepping 1, AuthenticAMD
PROCESSOR_LEVEL	23
PROCESSOR_REVISION	6001

Important: make there another one variable for **Tools** folder itself. So that you will have 2 paths: for SDK\Tools\Bin and SDK\Tools.



```
C:\Python311\Scripts\
C:\Python311\
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Users\user\AppData\Local\Android\Sdk\tools\bin
C:\Users\user\AppData\Local\Android\Sdk\tools
C:\Users\user\AppData\Local\Android\Sdk\platform-tools
C:\ProgramData\chocolatey\bin
C:\Program Files\nodejs\
C:\Program Files\nodejs\node_modules\npm\bin
```

3.3) Once Node is successfully installed, create new system variable for **NODE_HOME** (similarly as for Java and Android SDK). Take the path to installed directory (C:\Program Files\Nodejs).

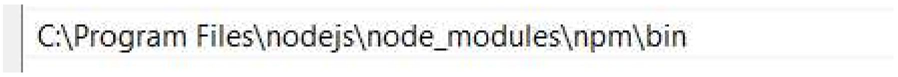


Variable name: NODE_HOME

Variable value: C:\Program Files\nodejs

Node is a software. In Node there is a command line installer called npm. With npm we can download any Node module (like Appium).

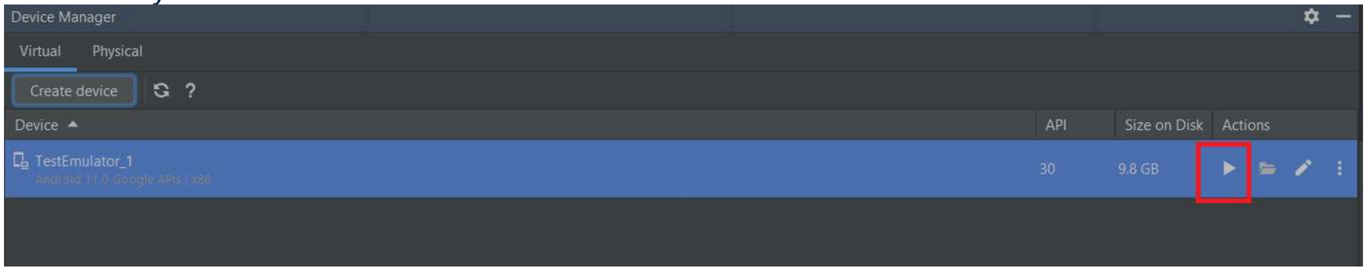
That's why you also need to create path to **npm\bin** folder.
Go to C:\Program Files\Nodejs > node_modules > npm > bin. Copy the path and paste as New Path:



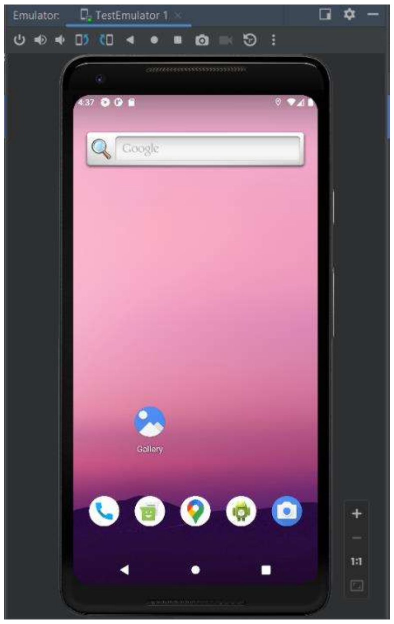
C:\Program Files\nodejs\node_modules\npm\bin

3.4) Configure **Emulator.**
- Launch Android Studio > Basic Activity > Next > Finish. Wait until the build got finished.
- Go to Tools menu option > select Device Manager. Click Create Device button.
- On Select Hardware screen select required device > Next.
- On System Image screen select required Android OS version > Next.
- On AVD screen enter the name of the device and set advanced settings if needed > Finish.

Note: *if your virtual device will work slowly, in this case you can increase the RAM in Advanced Settings.*
- Launch your device by clicking Play icon to make sure it is ready to use. It should be successfully launched and loaded.



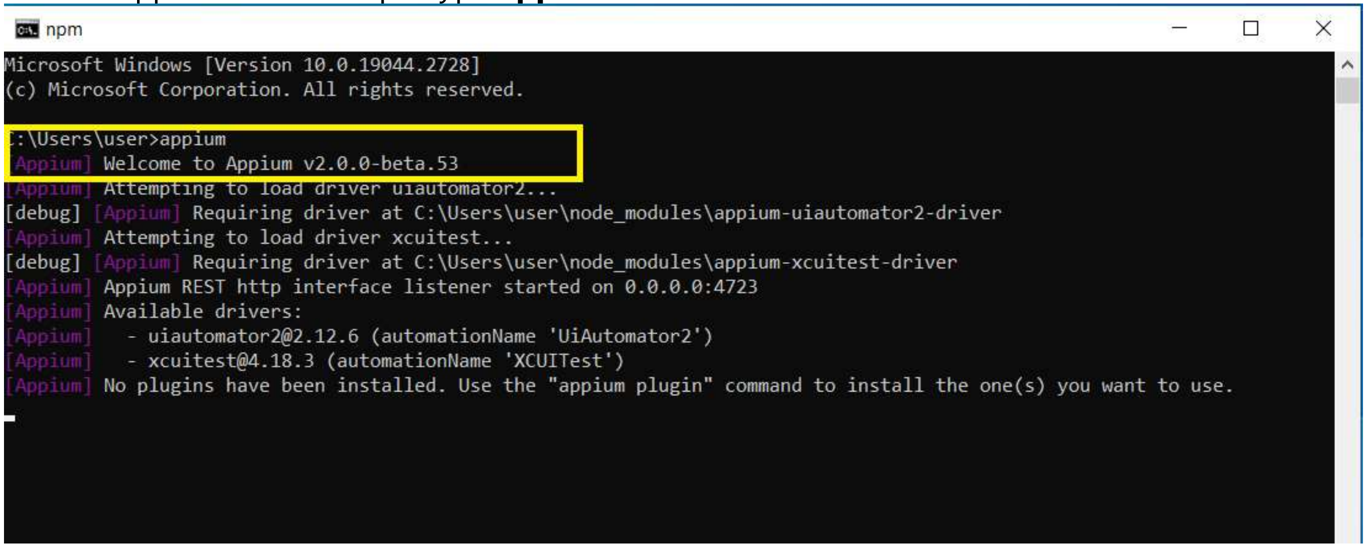
Device	API	Size on Disk	Actions
TestEmulator_1 Android 11.0 Google APIs | x86	30	9.8 GB	



3.5) Install and start the **Appium Server** via npm.
Appium is an open-source test automation framework for use with native, hybrid and mobile web apps. It drives iOS, Android, and Windows apps using the WebDriver protocol.
Npm stands for Node Package Manager. Any Node related packages you can directly get from Node repository.

To install Appium globally, open command line as administrator and type:
npm install -g appium@next
Wait until it will be successfully installed.

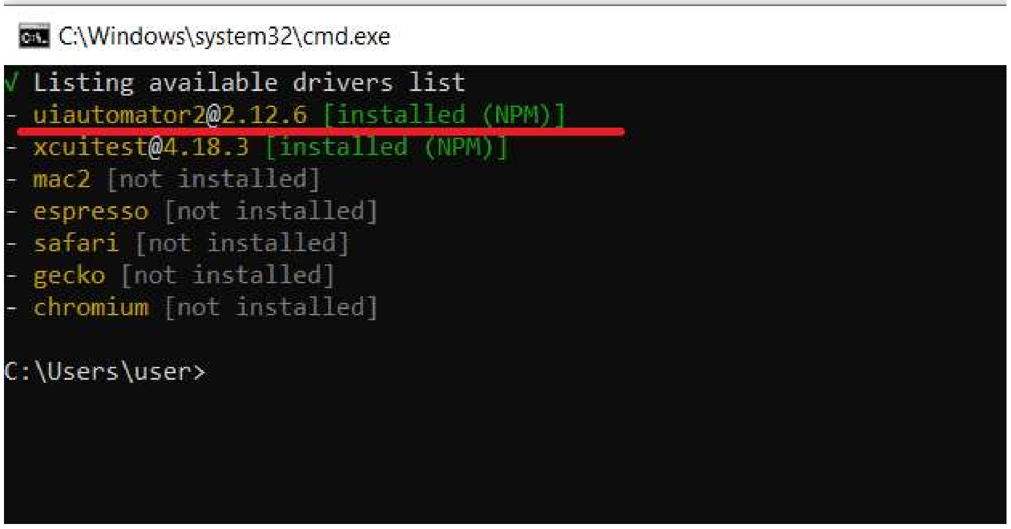
To start appium server simple type **appium** and hit enter.



```
Microsoft Windows [Version 10.0.19044.2728]
(c) Microsoft Corporation. All rights reserved.

:\Users\user>appium
[Appium] Welcome to Appium v2.0.0-beta.53
[Appium] Attempting to load driver uiautomator2...
[debug] [Appium] Requiring driver at C:\Users\user\node_modules\appium-uiautomator2-driver
[Appium] Attempting to load driver xcuitest...
[debug] [Appium] Requiring driver at C:\Users\user\node_modules\appium-xcuitest-driver
[Appium] Appium REST http interface listener started on 0.0.0.0:4723
[Appium] Available drivers:
[Appium]   - uiautomator2@2.12.6 (automationName 'UiAutomator2')
[Appium]   - xcuitest@4.18.3 (automationName 'XCUITest')
[Appium] No plugins have been installed. Use the "appium plugin" command to install the one(s) you want to use.
```

3.6) Install **Appium Drivers** for Android - UIAutomator2.
UI Automator is a UI testing framework introduced by Google to facilitate automation on an Android device or emulator.
Appium leverages this UIAutomator with its own wrapper and came up with UIAutomator2 Driver to automate the Android apps. For iOS apps there is XCUITest driver.

Go to command line and type **appium driver install uiautomator2**. Wait until it will be successfully installed and then type **appium driver list** to make sure that this driver is installed.



```
C:\Windows\system32\cmd.exe
√ Listing available drivers list
- uiautomator2@2.12.6 [installed (NPM)]
- xcuitest@4.18.3 [installed (NPM)]
- mac2 [not installed]
- espresso [not installed]
- safari [not installed]
- gecko [not installed]
- chromium [not installed]

C:\Users\user>
```

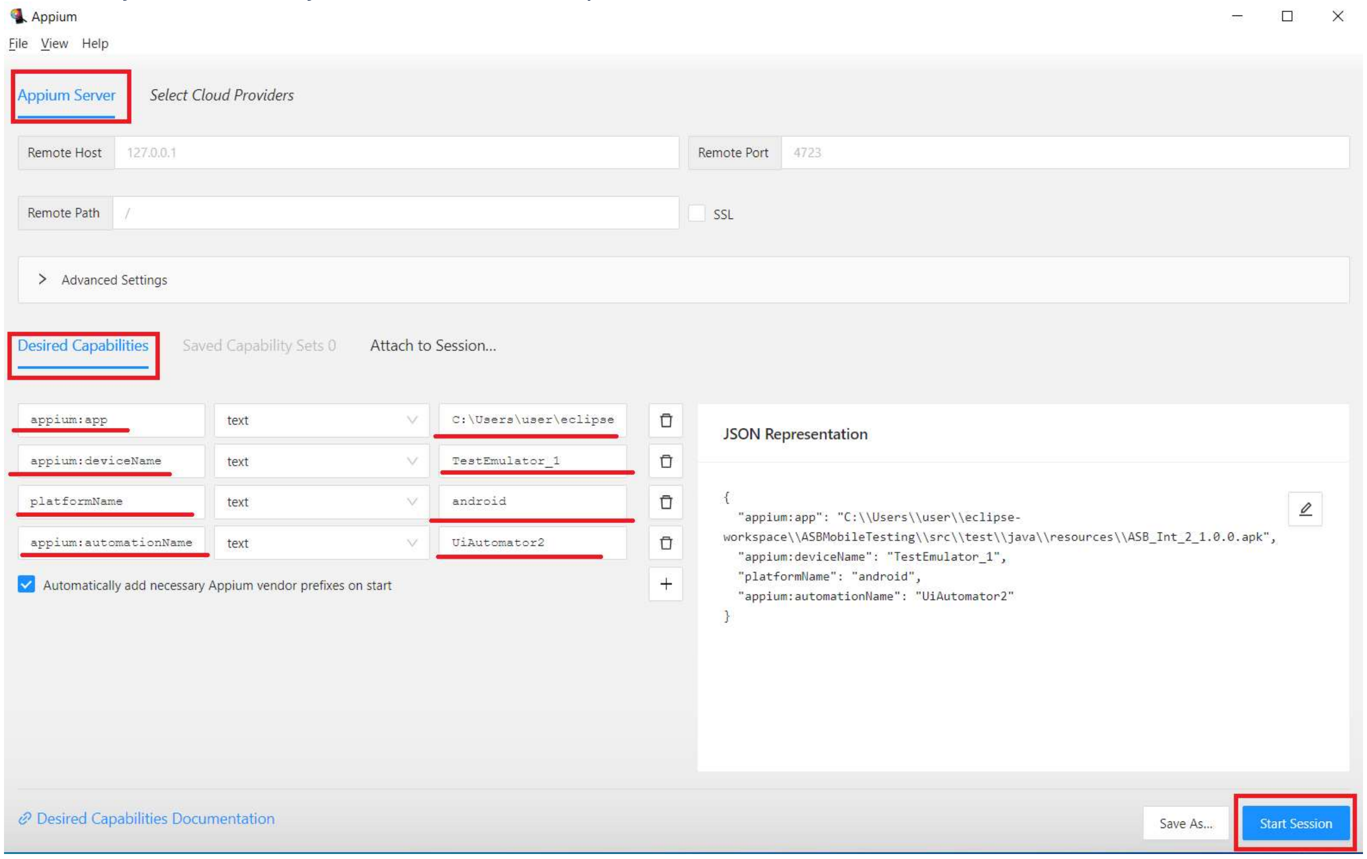
3.7) Install **Eclipse** editor.

3.8) Set up **capabilities for Appium Inspector.**
Appium Inspector allows you to define locators for each element on the mobile screen. Appium supports the following locators – Xpath, id, accessibilityId, className, anroidUIAutomator.

- Install Appium Inspector.
- Set up Desired Capabilities.

Desired Capabilities are keys and values encoded in a JSON object, sent by Appium clients to the server when a new automation session is requested. They tell the Appium drivers all kinds of important things about how you want your test to work. Each Appium client builds capabilities in a way specific to the client's language, but at the end of the day, they are sent over to Appium as JSON objects.

You need to type capabilities such as **platform name, device name, driver name and path to the .apk file**. You can Save this set of capabilities for the future use, so you don't need to type all this every time. Once you’ve done with capabilities click **Start Session**.



Remote Host 127.0.0.1 | Remote Port 4723

Remote Path / | SSL

appium:app	text	C:\Users\user\eclipse
appium:deviceName	text	TestEmulator_1
platformName	text	android
appium:automationName	text	UiAutomator2

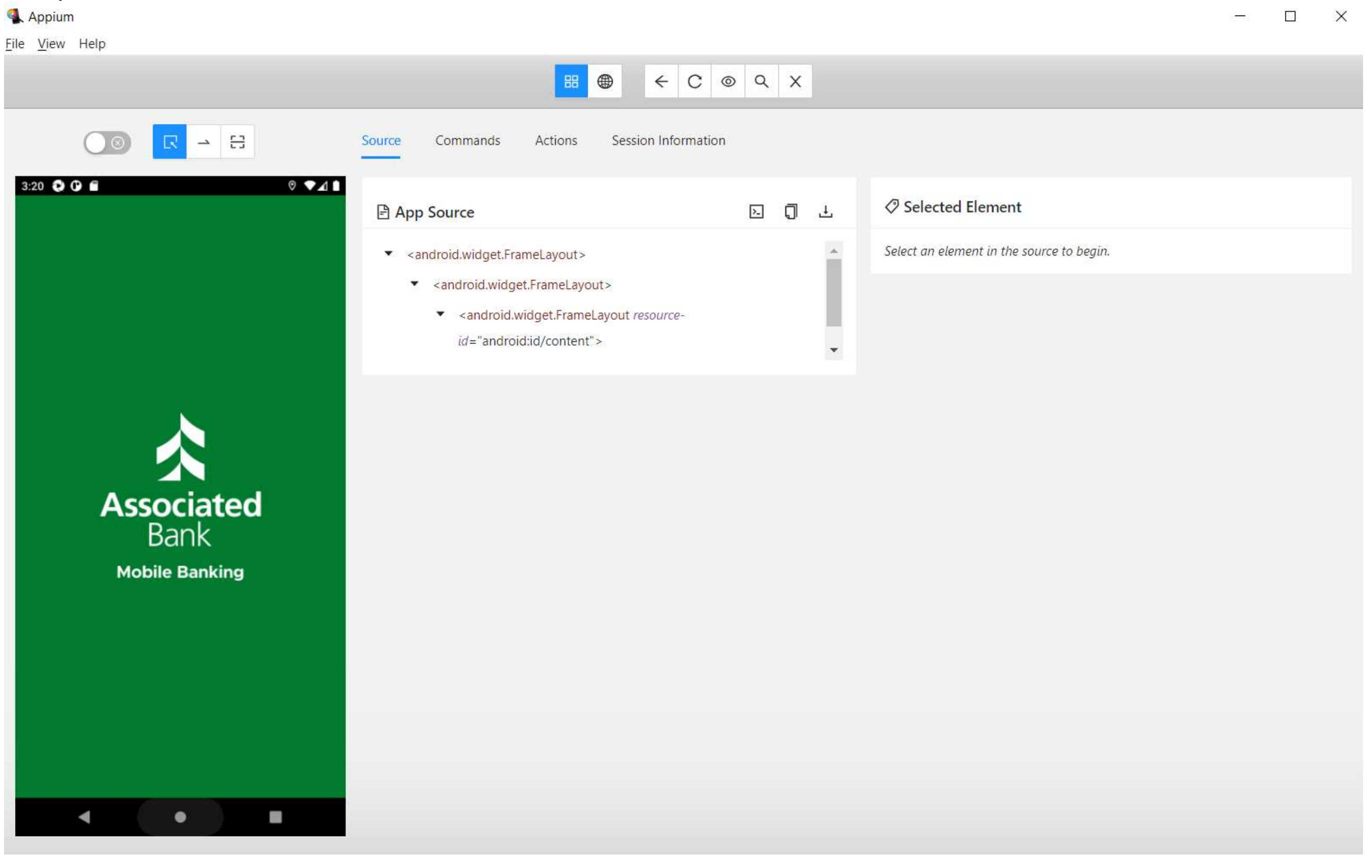
✓ Automatically add necessary Appium vendor prefixes on start

```
{
  "appium:app": "C:\\Users\\user\\eclipse-
workspace\\ASBMobileTesting\\src\\test\\java\\resources\\ASB_Int_2_1.0.0.apk",
  "appium:deviceName": "TestEmulator_1",
  "platformName": "android",
  "appium:automationName": "UiAutomator2"
}
```

Once you’ve done with capabilities click **Start Session**.

Note: *Make sure that you started Appium Server and Emulator before staring Appium inspector session!*

Output:



```
<android.widget.FrameLayout>
  <android.widget.FrameLayout>
    <android.widget.FrameLayout resource-id="android:id/content">
```

Select an element in the source to begin.

4. GIT Setup & Use

GitHub

Set up your local and remote repositories.
You can check if git is installed by running the following command from command line:

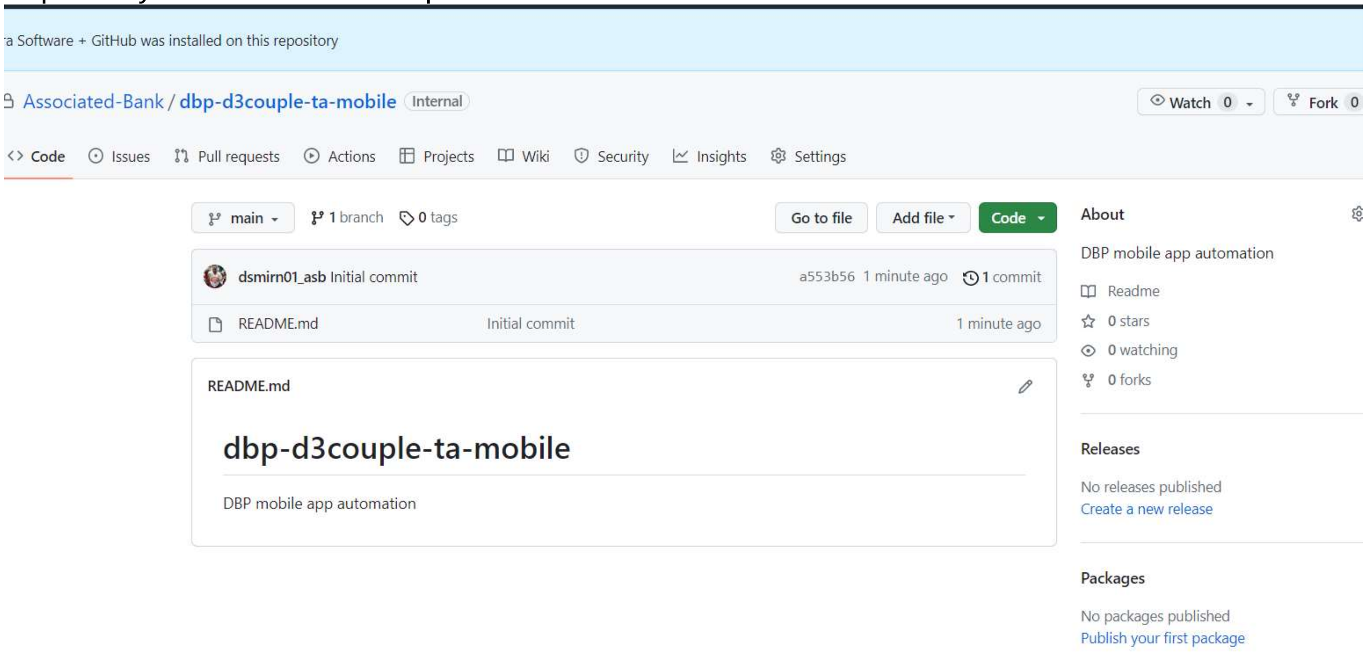
```
git --version
```

If there is an output with the git version, that means it's installed. Otherwise, you will need to **open a ticket in Cherwell to add GIT to your ABC account**, which you can then **install using the software center** on your machine.

Once git is installed and functional, you need to gain access to the company's github repos. This is done via Cherwell. You need to open a ticket and ask them to add permissions for your github account.

NOTE*: You need an existing github account for them to associate. If you do not have one, create it before you file the Cherwell ticket!*

Repository for mobile test scripts:



a Software + GitHub was installed on this repository

Associated-Bank / dbp-d3couple-ta-mobile Internal

dsmirn01_asb Initial commit		a553b56 1 minute ago	1 commit
README.md	Initial commit	1 minute ago	

README.md

dbp-d3couple-ta-mobile

DBP mobile app automation

About

DBP mobile app automation

Readme
0 stars
0 watching
0 forks

Releases

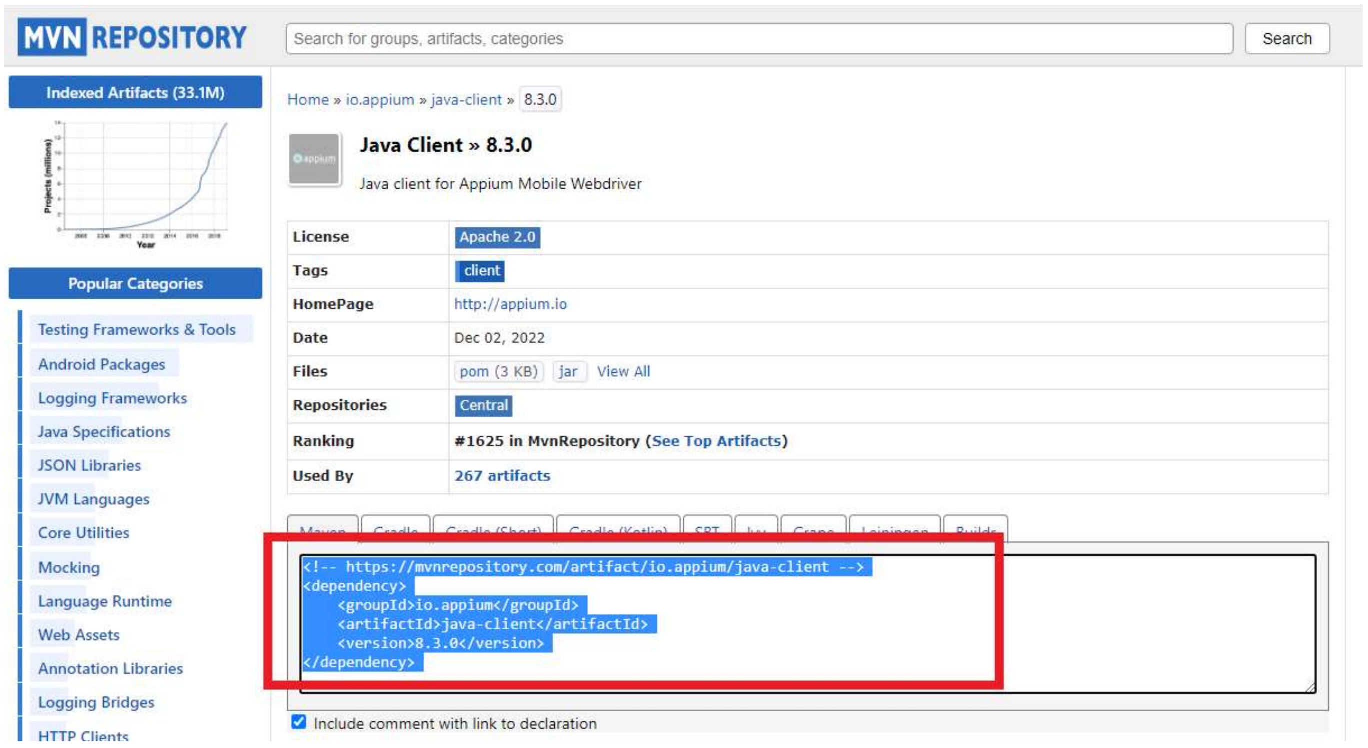
No releases published
Create a new release

Packages

No packages published
Publish your first package

5. Project Setup.

5.1) Create Maven project in Eclipse.
Maven is a Java-based management and automation tool. Maven is pre-defined and declared in an XML format called POM (Page Object Model) and referred to as 'pom.xml'.
Go to Maven repository and find Appium Maven Dependencies - https://mvnrepository.com/artifact/io.appium/java-client/8.3.0



Home » io.appium » java-client » 8.3.0

Java Client » 8.3.0

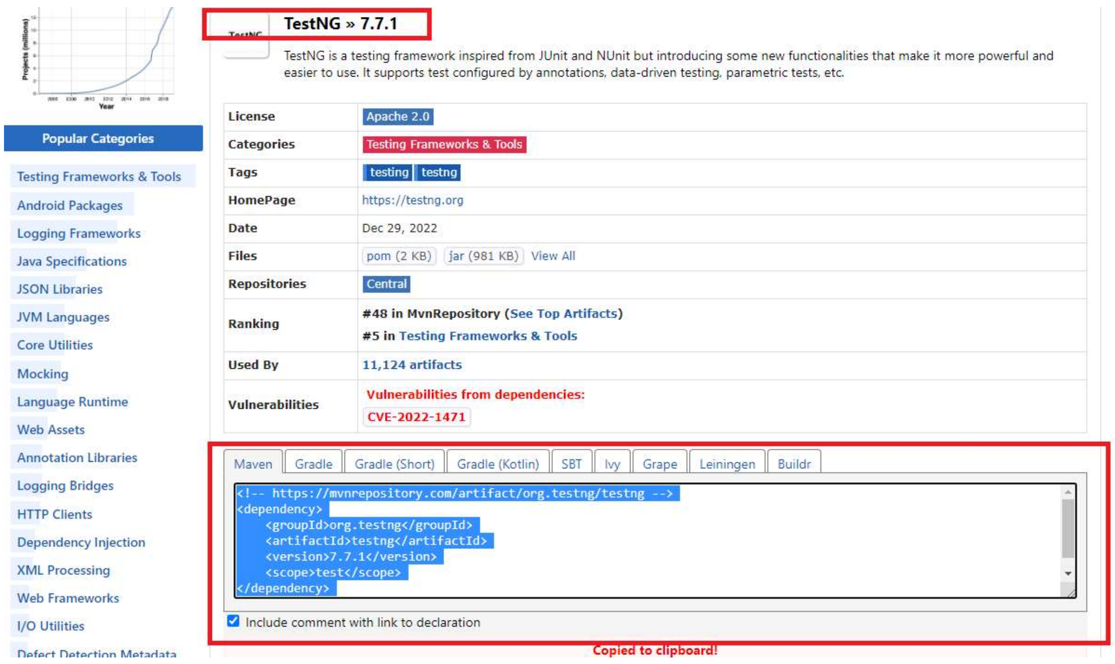
Java client for Appium Mobile Webdriver

License	Apache 2.0
Tags	client
HomePage	http://appium.io
Date	Dec 02, 2022
Files	pom (3 KB) jar View All
Repositories	Central
Ranking	#1625 in MvnRepository (See Top Artifacts)
Used By	267 artifacts

```
<!-- https://mvnrepository.com/artifact/io.appium/java-client -->
<dependency>
    <groupId>io.appium</groupId>
    <artifactId>java-client</artifactId>
    <version>8.3.0</version>
</dependency>
```

☑ Include comment with link to declaration

Also, we will need TestNG dependency:
https://mvnrepository.com/artifact/org.testng/testng/7.7.1



TestNG » 7.7.1

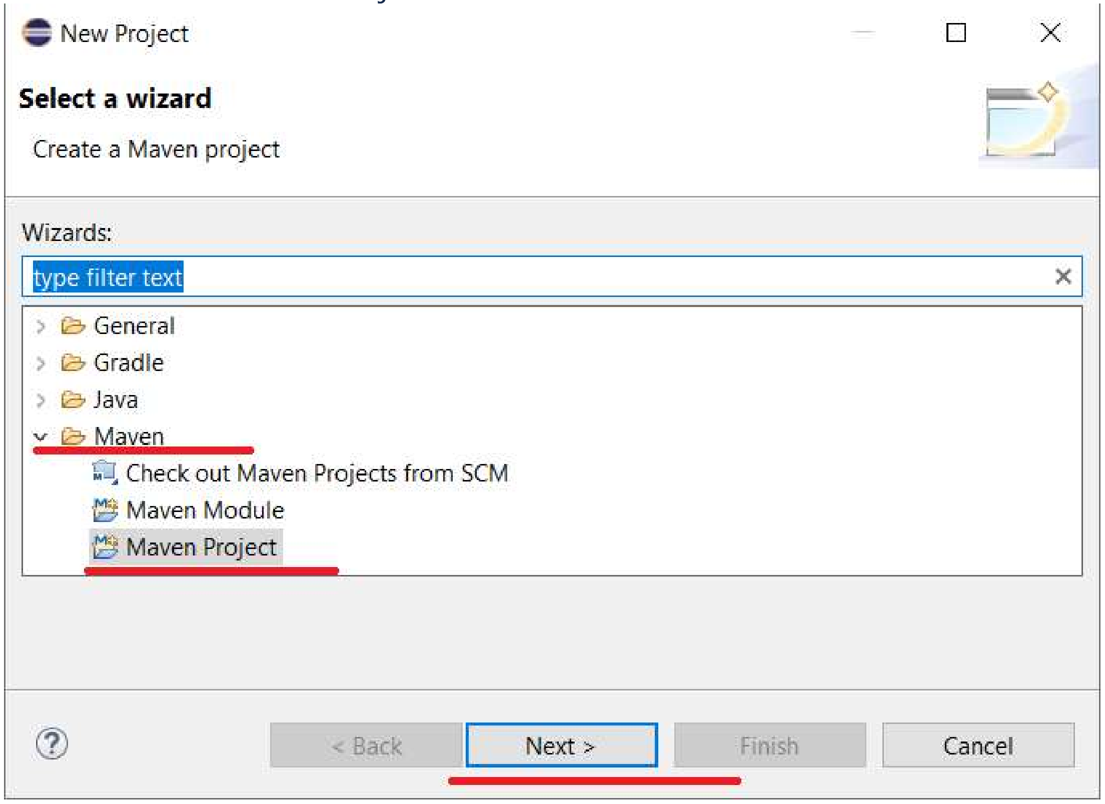
TestNG is a testing framework inspired from JUnit and NUnit but introducing some new functionalities that make it more powerful and easier to use. It supports test configured by annotations, data-driven testing, parametric tests, etc.

License	Apache 2.0
Categories	Testing Frameworks & Tools
Tags	testing testng
HomePage	https://testng.org
Date	Dec 29, 2022
Files	pom (2 KB) jar (981 KB) View All
Repositories	Central
Ranking	#48 in MvnRepository (See Top Artifacts) #5 in Testing Frameworks & Tools
Used By	11,124 artifacts
Vulnerabilities	Vulnerabilities from dependencies: CVE-2022-1471

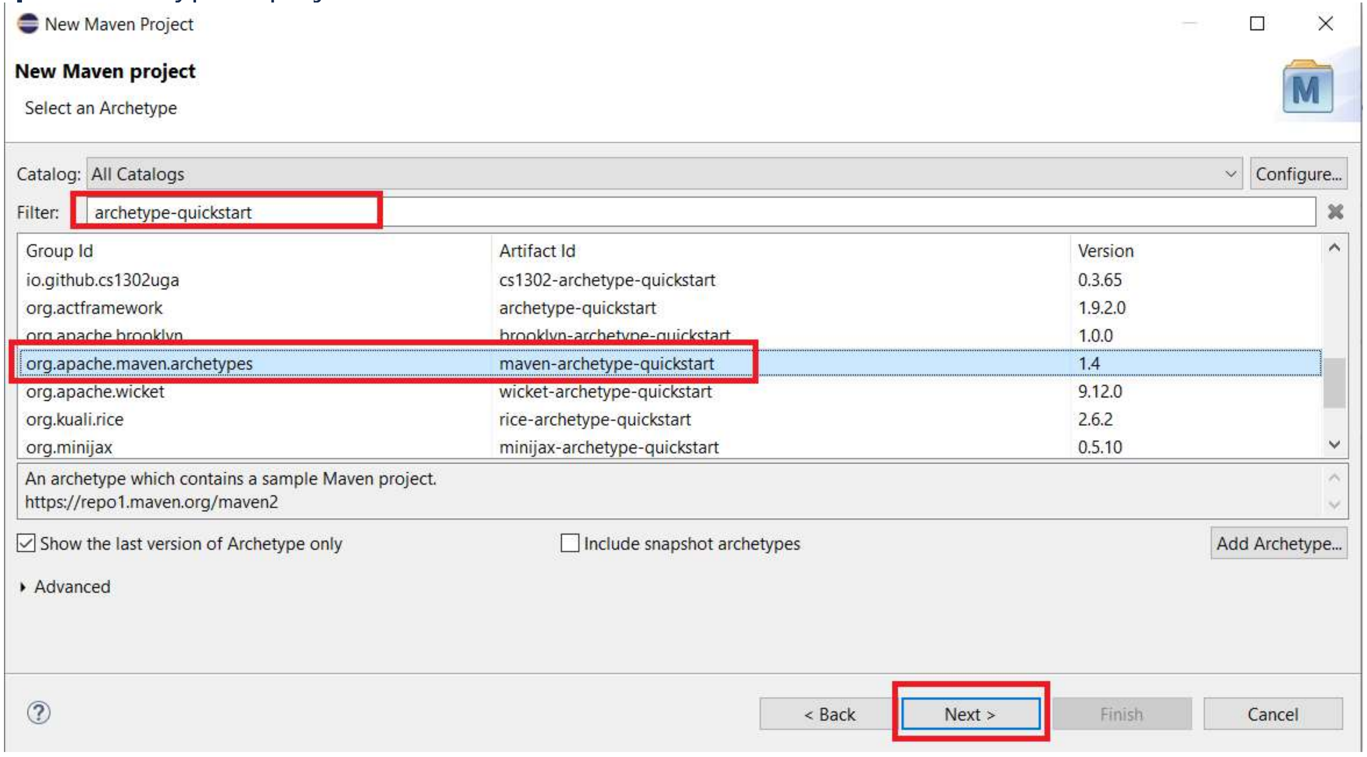
```
<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.7.1</version>
    <scope>test</scope>
</dependency>
```

☑ Include comment with link to declaration

- Open Eclipse editor and click on Create New Project.
Select Maven > Maven Project. Click Next.



Skip the next page by clicking Next.
Select type of the project: Type in Filter field 'archetype-quickstart' and select **maven-archetype-quickstart** type of project from the list. Then click Next.


New Maven Project

New Maven project

Select an Archetype

Catalog: All Catalogs

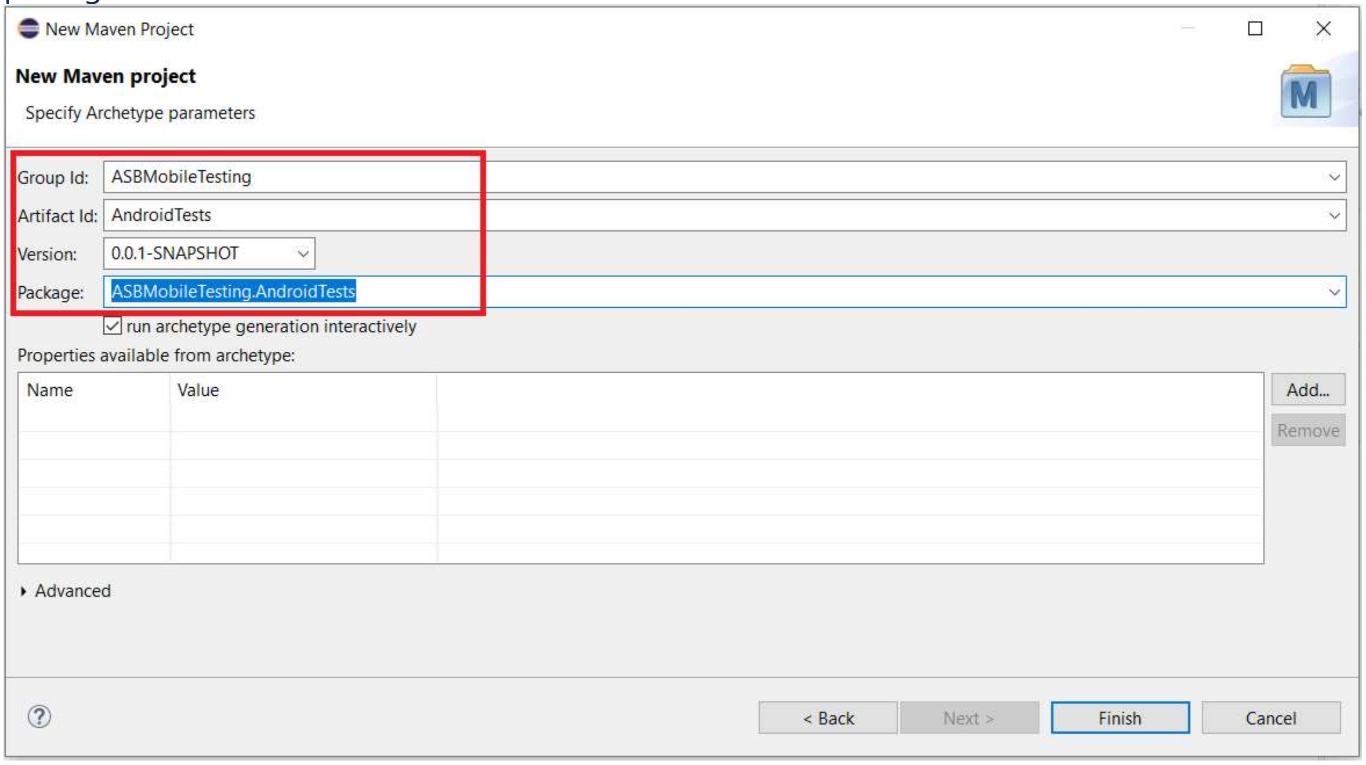
Filter: archetype-quickstart

Group Id	Artifact Id	Version
io.github.cs1302uga	cs1302-archetype-quickstart	0.3.65
org.actframework	archetype-quickstart	1.9.2.0
org.apache.brooklyn	brooklyn-archetype-quickstart	1.0.0
org.apache.maven.archetypes	maven-archetype-quickstart	1.4
org.apache.wicket	wicket-archetype-quickstart	9.12.0
org.kuali.rice	rice-archetype-quickstart	2.6.2
org.minijax	minijax-archetype-quickstart	0.5.10

An archetype which contains a sample Maven project.
https://repo1.maven.org/maven2

☑ Show the last version of Archetype only
☐ Include snapshot archetypes

Fill out the Group ID and Artifact ID fields with the name of the project (for example ASBMobileTesting and AndroidTests). In that case **ASBMobileTesting. AndroidTests** will be the package name. Click Finish.


New Maven Project

New Maven project

Specify Archetype parameters

Group Id: ASBMobileTesting

Artifact Id: AndroidTests

Version: 0.0.1-SNAPSHOT

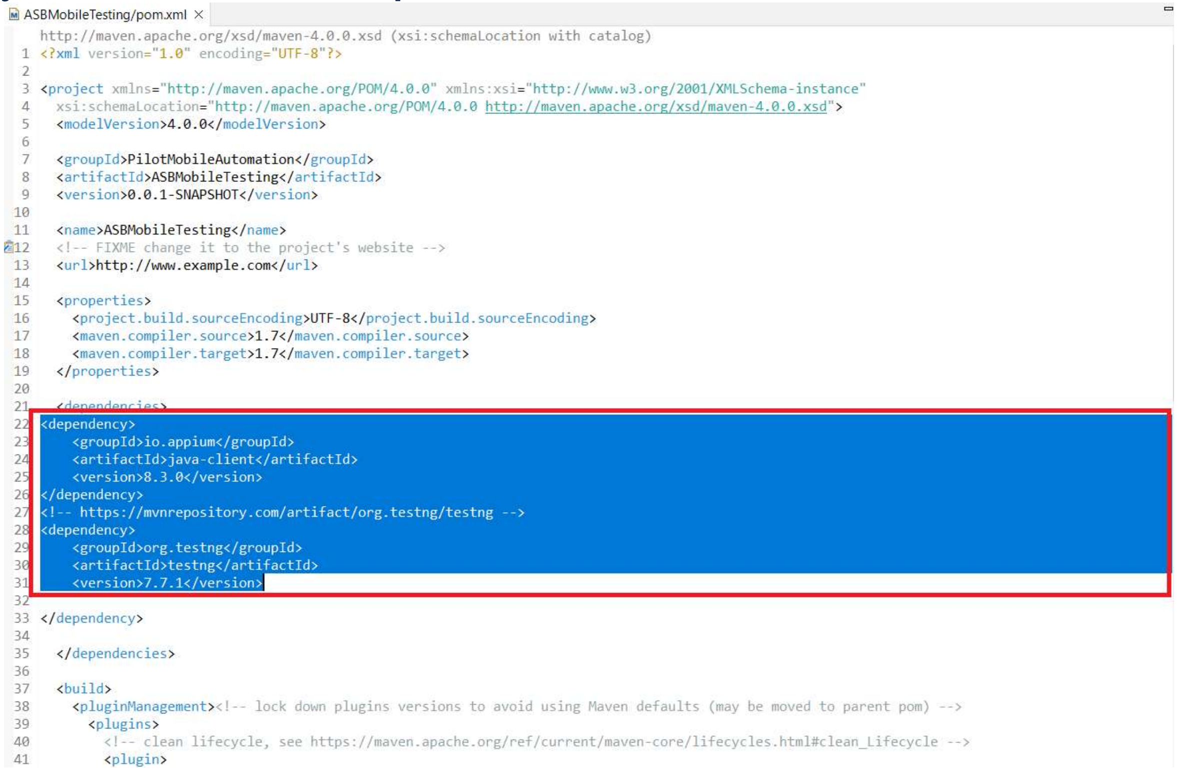
Package: ASBMobileTesting.AndroidTests

☑ run archetype generation interactively

Properties available from archetype:

Name	Value

- Set up dependencies. Open POM.xml file, *remove junit set of dependencies* and paste **appium java-client and testNG dependencies** and that we found above.

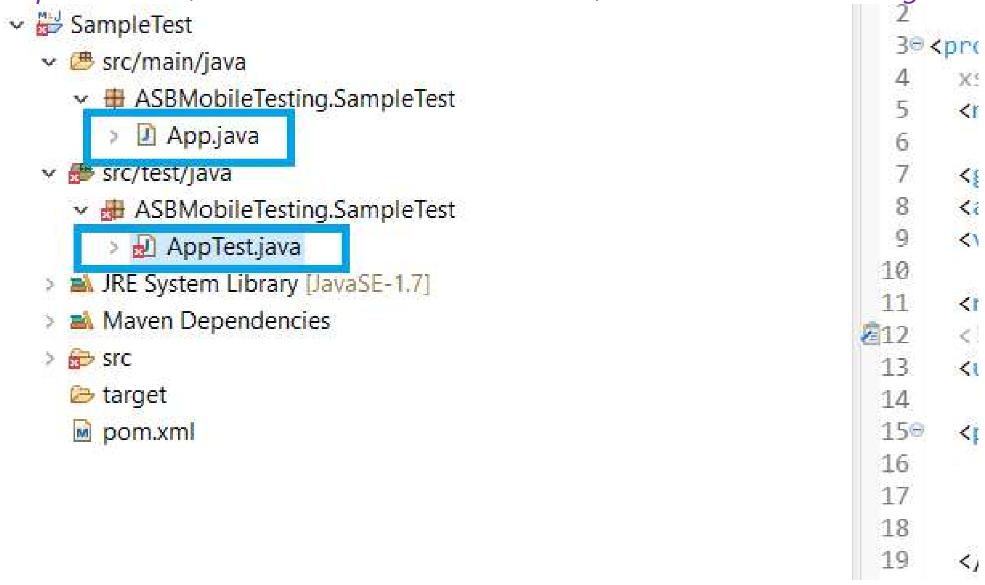


ASBMobileTesting/pom.xml

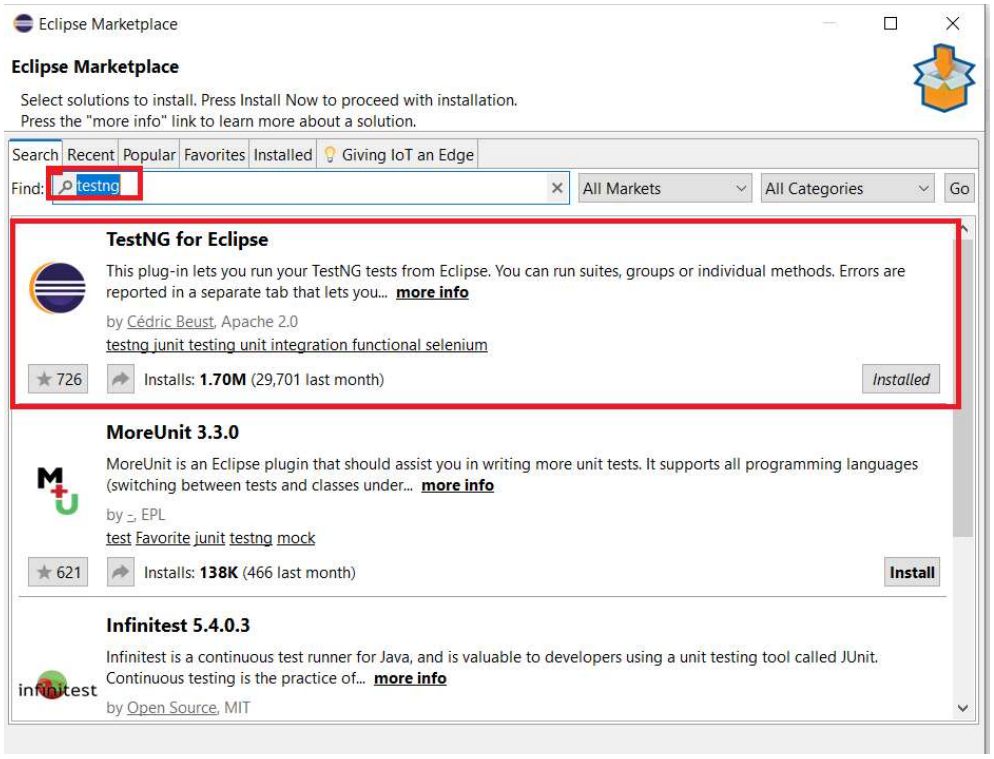
```
http://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation with catalog)
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>PilotMobileAutomation</groupId>
  <artifactId>ASBMobileTesting</artifactId>
  <version>0.0.1-SNAPSHOT</version>

  <name>ASBMobileTesting</name>
  <!-- FIXME change it to the project's website -->
  <url>http://www.example.com</url>

  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.source>1.7</maven.compiler.source>
    <maven.compiler.target>1.7</maven.compiler.target>
  </properties>

  <dependencies>
<dependency>
    <groupId>io.appium</groupId>
    <artifactId>java-client</artifactId>
    <version>8.3.0</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.testng/testng -->
<dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.7.1</version>

</dependency>

  </dependencies>

  <build>
    <pluginManagement><!-- lock down plugins versions to avoid using Maven defaults (may be moved to parent pom) -->
      <plugins>
        <!-- clean lifecycle, see https://maven.apache.org/ref/current/maven-core/lifecycles.html#clean_Lifecycle -->
        <plugin>
```

Save the project.

***Note:** there will be an error presented after saving the project. It's because there are sample classes App.java and AppTest.java that are created by default, and it refers to junit. Since we removed junit dependencies, remove these classes as well, and the error will be gone.*



-let’s install TestNG plugin for Eclipse, that is required to run your test cases in TestNG framework. Go to Help menu option > Eclipse Marketplace. Type ‘testNG’ in the search field and install TestNG plugin for Eclipse. **Restart Eclipse**.


Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search | Recent | Popular | Favorites | Installed | Giving IoT an Edge

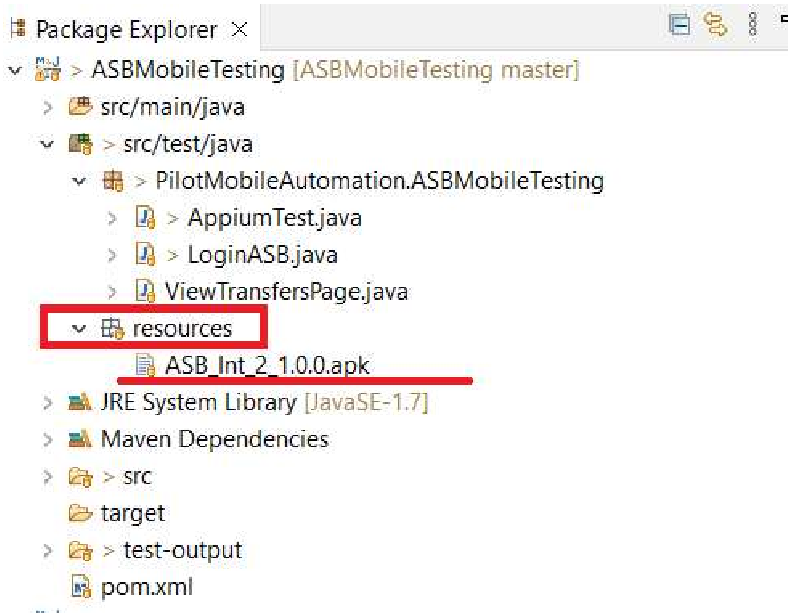
Find: testng | All Markets | All Categories | Go

TestNG for Eclipse
This plug-in lets you run your TestNG tests from Eclipse. You can run suites, groups or individual methods. Errors are reported in a separate tab that lets you... **more info**
by Cédric Beust, Apache 2.0
testng junit testing unit integration functional selenium
726 | Installs: **1.70M** (29,701 last month) | Installed

MoreUnit 3.3.0
MoreUnit is an Eclipse plugin that should assist you in writing more unit tests. It supports all programming languages (switching between tests and classes under... **more info**
by -, EPL
test Favorite junit testng mock
621 | Installs: **138K** (466 last month) | **Install**

Infinitest 5.4.0.3
Infinitest is a continuous test runner for Java, and is valuable to developers using a unit testing tool called JUnit. Continuous testing is the practice of... **more info**
by Open Source, MIT

-Create package, set the name ‘resources’, and paste the .apk file into this package. Simple copy the local path to the app you’re going to automate, right click on package name > Paste.


```
Package Explorer
ASBMobileTesting [ASBMobileTesting master]
  src/main/java
  src/test/java
    PilotMobileAutomation.ASBMobileTesting
      AppiumTest.java
      LoginASB.java
      ViewTransfersPage.java
    resources
      ASB_Int_2_1.0.0.apk
  JRE System Library [JavaSE-1.7]
  Maven Dependencies
  src
  target
  test-output
  pom.xml
```

-Create first test automation script where you can indicate for Appium what kind of driver you are using; set up desired capabilities, such as an emulator device name and path to the application; IP address and port number from where Appium server will start.

Note: *once you created this class, set* ***@Test*** *annotation (instead of @BeforeClass) and run it (as TestNG test) for the first time. After first run the app will be installed on emulator and available for testing.*

```
public AndroidDriver driver;
public AppiumDriverLocalService service;

@BeforeClass
public void ConfigureAppium() throws MalformedURLException

{

service = new AppiumServiceBuilder().withAppiumJS(new
File("C:\\Users\\user\\AppData\\Roaming\\npm\\node_modules\\appium\\build\\lib\\main.js"))
.withIPAddress("127.0.0.1").usingDriverExecutable (new File ("C:\\Program
Files\\nodejs\\node.exe")).usingPort(4723).build();
service.start();

UiAutomator2Options options = new UiAutomator2Options();
options.setDeviceName("TestEmulator_1");
options.setApp("C:\\Users\\user\\git\\ASBMobileTesting\\ASBMobileTesting\\src\\test\\java\\r
esources\\ASB_Int_2_1.0.0.apk");
driver = new AndroidDriver(new URL("http://127.0.0.1:4723"), options);
driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

}

@AfterClass
public void tearDown()
{

driver.quit();
service.stop();

}

}
```

Note: *this is the first base test script that you can use to start and stop Appium Server programmatically and launch the app on the emulator before each test. There can be added more details about platform version, OS version, etc., but the code presented above contains enough information for understand from what point the process of testing a mobile application begins.*

Now you should be ready to work on the automated tests.