Table of Contents

- Prerequisites
- Download the Sample Appium Project
- Install Eclipse Plugins
- Edit Script Capabilities
- Start the Appium Server
- Launch the Emulator
- Test the Script

1 Prerequisites

Before you begin to setup your IDE, you need to install and configure the following software on your local machine in order for the environment to work properly.

Java/JDK

Install Java/JDK

>> Click here for instructions

- **Unix/Linux Instructions**
  1. Download the latest jdk from Oracle
  2. Install it following the install wizard prompts
  3. Open a terminal, and open up your `.bash_profile`

```
$  cd  ~
$ vim ~/.bash_profile
```

  4. Set JAVA_HOME variable to `/usr/libexec/java_home`

```
#Java Home
export  JAVA_HOME=$( /usr/libexec/java_home )

#export path
export  PATH= "$PATH:/usr/local/bin"
export  PATH= "$PATH:$JAVA_HOME/bin"
```

Why /usr/libexec/java_home?

The reason we choose /usr/libexec/java_home rather than a direct path is because we want `java_home` to return the Java version, specified in Java Preferences, for the current user. For example if there were multiple versions of jdk installed on the current system:

```
##return top Java version
$  /usr/libexec/java_home
/Library/Java/JavaVirtualMachines/1 .7.0.jdk /Contents/Home

## I want Java version 1.6
$  /usr/libexec/java_home  - v  1.6
/System/Library/Java/JavaVirtualMachines/1 .6.0.jdk /Contents/Home
```

- **Windows Instructions**
    1. Download the latest jdk from Oracle
    2. Install it following the install wizard prompts
    3. Open your Advanced System Settings
    4. Click on Advanced, select Environment Variables
    5. In System variables, add a new `JAVA_HOME` variable and point it to the JDK installed folder.
    6. In System variables, find `PATH`, clicks edit and append this `%JAVA_HOME%\bin` to the end.
    7. Open a windows terminal (Winkey + R then type cmd) and test by entering the following:

```
C:\Users\home>java -version
java version "1.8.0_60"
Java(TM) SE Runtime Environment (build 1.8.0_60-b27)
Java HotSpot(TM) 64-Bit Server VM (build 25.60-b23, mixed mode)

C:\Users\home>javac -version
javac 1.8.0_60

C:\Users\home>echo %JAVA_HOME%
C:\Program Files\Java\jdk1.8.0_60
```

Apache Ant

Install Ant

>> Click here for instructions

- **Unix/Linux Instructions**
  1. Visit Apache Ant Website and download the tar.gz file.
  2. Copy the file to the desired directory

```
$  cp  ~ /Downloads/apache-ant- <version>. tar .gz  /Users/home/
```

  3. Extract the file

```
$  tar  -xvf apache-ant-<version>. tar .gz
```

  4. Set the command `ant` as an environment variable so you can "ant" build anywhere. Start by opening your `.bash_profile`

```
$ vim ~/.bash_profile
```

     Export `$ANT_HOME/bin,` append the `PATH` variable, and then restart your terminal.

```
#Java Home
export  JAVA_HOME=$( /usr/libexec/java_home )

#Apache Ant
export  ANT_HOME= /Users/home/apache-ant- <version>

#Export to PATH
export  PATH= "$PATH:/usr/local/bin"
export  PATH= "$PATH:$JAVA_HOME/bin:$ANT_HOME/bin"
```

  5. In a new terminal, confirm Ant was installed correctly (do not worry about "build failed" message as you haven't created a build.xml file).

```
$ ant - v

Apache Ant(TM) version 1.9.7 compiled on April 9 2016
Trying the default build  file : build.xml
Buildfile: build.xml does not exist!
Build failed
```

- **Windows Instructions**
    1. Visit Apache Ant Website and download the Ant binary zip file, for example : `apache-ant-1.9.4-bin.zip`
    2. Unzip it to the folder you want to store Apache Ant.
    3. Add ANT_HOME as a windows environment variable
    4. Update the PATH variable by appending %ANT_HOME%\bin so that you can run ant everywhere.
    5. Verify that above steps were completed by typing `ant -v` in a new Windows command prompt (Winkey + R then type cmd).

```
C:\Users\home>ant -v
Apache Ant(TM) version 1.9.4 compiled on April 29 2014
Trying the default build file: build.xml
Buildfile: build.xml does not exist!
Build failed
```

Eclipse

Install Eclipse

>> Click here for instructions

- **Unix/Linux Instructions**
    1. Download the latest version of Eclipse.
    2. Follow the installation prompts.
    3. Create/Select a workspace to store your projects
    4. (Optional) Configure Eclipse theme by going to **Window > Preferences > General > Appearance**

- **Windows Instructions**
  - Download the latest version of Eclipse.
  - Follow the installation prompts.
  - Create/Select a workspace to store your projects
  - (Optional) Configure Eclipse theme by going to **Window > Preferences > General > Appearance**

Appium

Install Appium

>> Click here for instructions

- **Linux/Unix Instructions**
  1. Install appium through the command line.

     ```
     $ apt-get  install  appium
     ```

  2. Update your repository to fix any dependency errors

     ```
     apt-get update
     ```

- **Mac OSX Instructions**
  1. Download and install the necessary client libraries (we use Java in the course).
  2. Download the appium server client.
  3. Follow the installation wizard prompts
- **Windows Instructions**
  1. Download and install the necessary client libraries (we use Java in the course).
  2. Download the appium server client
  3. Follow the installation wizard prompts

Android SDK

Install Android SDK

>> Click here for instructions

- Mac OSX/Linux/Unix Instructions
  1. Navigate to the website and download the stand alone sdk command line tools:
     a. Mac sdk command line tools.
     b. Linux sdk command line tools.
  2. Extract the file into the folder of your choice

```
$  tar  -xvf <android-sdk-package- file >
```

  3. Set the environment variables for `$ANDROID_HOME`, and also append the `$PATH` variable in your `.bash_profile` script.

```
#Java Home
export  JAVA_HOME=$( /usr/libexec/java_home )

#Apache Ant
export  ANT_HOME= /Users/home/apache-ant- <version>

#Android Home
export  ANDROID_HOME= /Users/home/Library/Android/sdk

#Export to PATH
export  PATH= "$PATH:/usr/local/bin"
export  PATH= "$PATH:$JAVA_HOME/bin:$ANT_HOME/bin:$ANDROID_HOME/bin"
```

  4. To test, open a terminal and type the following commands to launch the SDK manager

```
$  cd  $ANDROID_HOME
$  cd  tools

$ . /android
```

- **Windows Instructions**
    1. Navigate to the website and download the android installer.
    2. Follow the installation prompts
    3. Navigate to Advanced Settings
    4. Set ANDROID_HOME environment variable

```
set ANDROID_HOME=C:\<installation location>\android-sdk-windows
set PATH=%PATH%;%ANDROID_HOME%\tools;%ANDROID_HOME%\platform-tools
```

    5. To test and see if it has successfully installed open a command terminal (Winkey + R then type cmd) and type `android`. The SDK Manager will launch.

Maven

Install Maven

>> Click here for instructions

- **Unix/Linux Instructions**
  1. Download and extract the tar.gz file

```
$  tar  -xvf apache-maven-<version>-bin.zip
```

  2. Set the M2_HOME environment variable and update the PATH variable in your .bash_profile

```
$ vim ~/.bash_profile
```

```
#Java Home
export  JAVA_HOME=$( /usr/libexec/java_home )

#Apache Ant
export  ANT_HOME= /Users/home/apache-ant- <version>

#Android Home
export  ANDROID_HOME= /Users/home/Library/Android/sdk

#Maven Home
export  M2_HOME= /Users/home/apache-maven- <version>

#Export to PATH
export  PATH= "$PATH:/usr/local/bin"
export  PATH= "$PATH:$JAVA_HOME/bin:$ANT_HOME/bin:$ANDROID_HOME/bin:$M2_HOME/bin"
```

  3. Check to see if maven installed correctly

```
$ mvn -version

Apache Maven 3.1.1 (0728685237757ffbf44136acec0402957f723d9a; 2013-09-17 23:22:
22+0800)
Maven home:  /Users/home/apache-maven-3 .1.1
Java version: 1.7.0_05, vendor: Oracle Corporation
Java home:  /Library/Java/JavaVirtualMachines/1 .7.0.jdk /Contents/Home/jre
Default locale: en_US, platform encoding: UTF-8
OS name:  "mac os x" , version:  "10.9" , arch:  "x86_64" , family:  "mac"
```

- **Windows Instructions**
  1. Download and install the maven binary distrubtion: http://mirrors.ocf.berkeley.edu/apache/maven/maven-3/3.3.9/binaries/apache-maven-3.3.9-bin.zip
  2. Ensure JAVA_HOME variable was properly set
  3. Add the unpacked distribution's bin directory to your user PATH environment variable by opening up the system properties (WinKey + Pause),
     a. selecting the "Advanced" tab, and the "Environment Variables" button,
     b. then adding or selecting the *PATH* variable in the user variables with the value C:\Program Files\apache-maven-3.3.9\bin.
  4. Open a new command prompt (Winkey + R then type cmd) and run mvn -v to verify the installation.

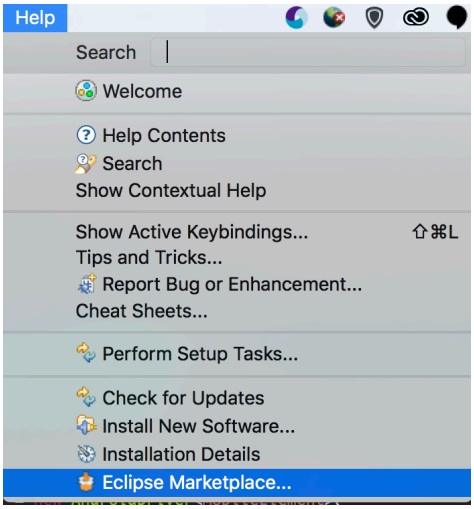
2 Download the Sample Appium Project

1. The first thing we need to do is download the sample project so that we can import it into Eclipse. This will include the Eclipse project, .apk file, maven, and testNG libraries.
2. Point Eclipse to the correct workspace (either right where you downloaded it, or move it into your workspace folder in your Eclipse Directory).
3. Finally, open the project in eclipse to verify if the project imported correctly.

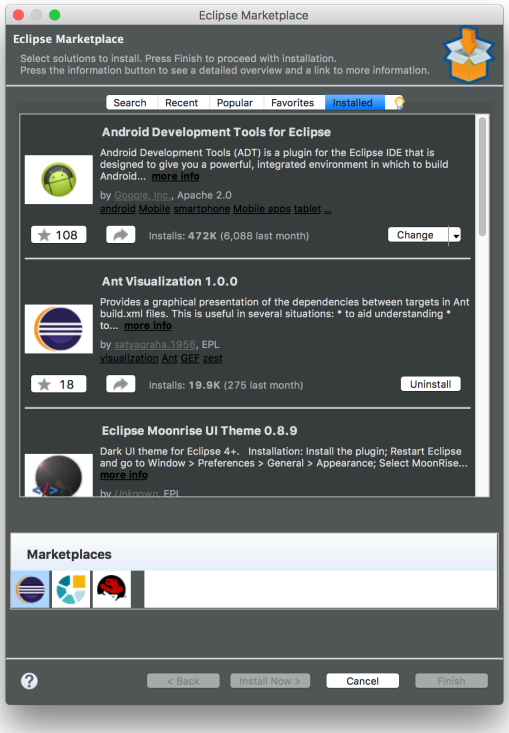
3 Install Eclipse Plugins

There are various plugins that you will need to install/configure within Eclipse.

1. Go to Help > Eclipse Marketplace

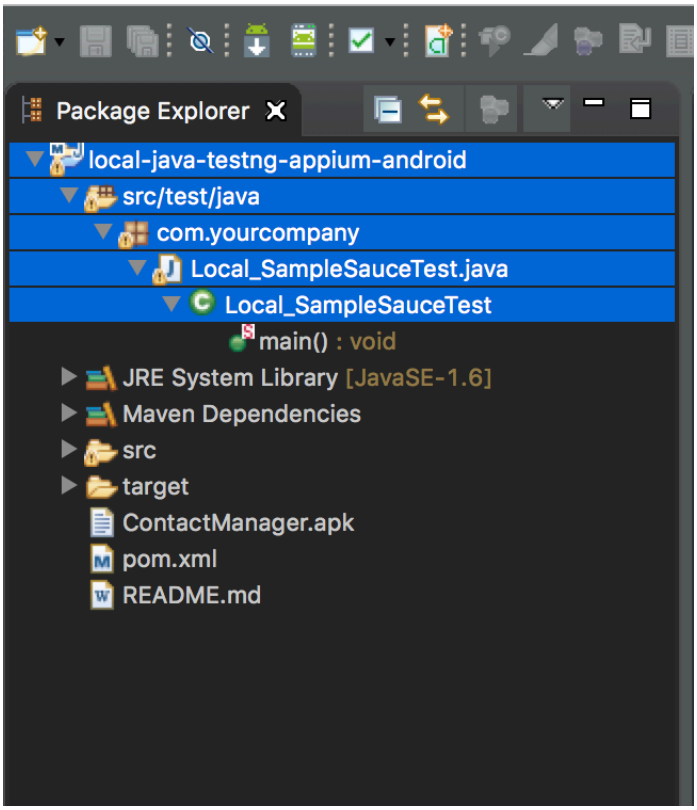


2. In the search bar, search and download/install the following plugins:
    a. **>> Android Development Tools for Eclipse:**
    Android Development Tools (ADT) is a plugin for the Eclipse IDE that is designed to give you a powerful, integrated environment in which to build Android applications.
    ADT extends the capabilities of Eclipse to let you quickly set up new Android projects, create an application UI, add packages based on the Android Framework API, debug your applications using the Android SDK tools, and even export signed (or unsigned) .apk files in order to distribute your application.
    Developing in Eclipse with ADT is highly recommended and is the fastest way to get started. With the guided project setup it provides, as well as tools integration, custom XML editors, and debug output pane, ADT gives you an incredible boost in developing Android applications.
    b. **>> Ant Visualization:**
    Provides a graphical presentation of the dependencies between targets in Ant build.xml files. This is useful in several situations:
    * to aid understanding
    * to highlight excessive complexity
    * to provide images for use in documentation
    c. **>> Maven Integration for Eclipse (latest version):**
    m2e provides comprehensive Maven integration for Eclipse. You can use m2e to manage both simple and multi-module Maven projects, execute Maven builds via the Eclipse interface, and interact with Maven repositories. m2e makes development easier by integrating data from a project's Object Model with Eclipse IDE features. With m2e, you can use Maven within Eclipse in a natural and intuitive interface.
    d. **>> TestNG for Eclipse:**
    A test engine for testing selenium/appium scripts through Eclipse

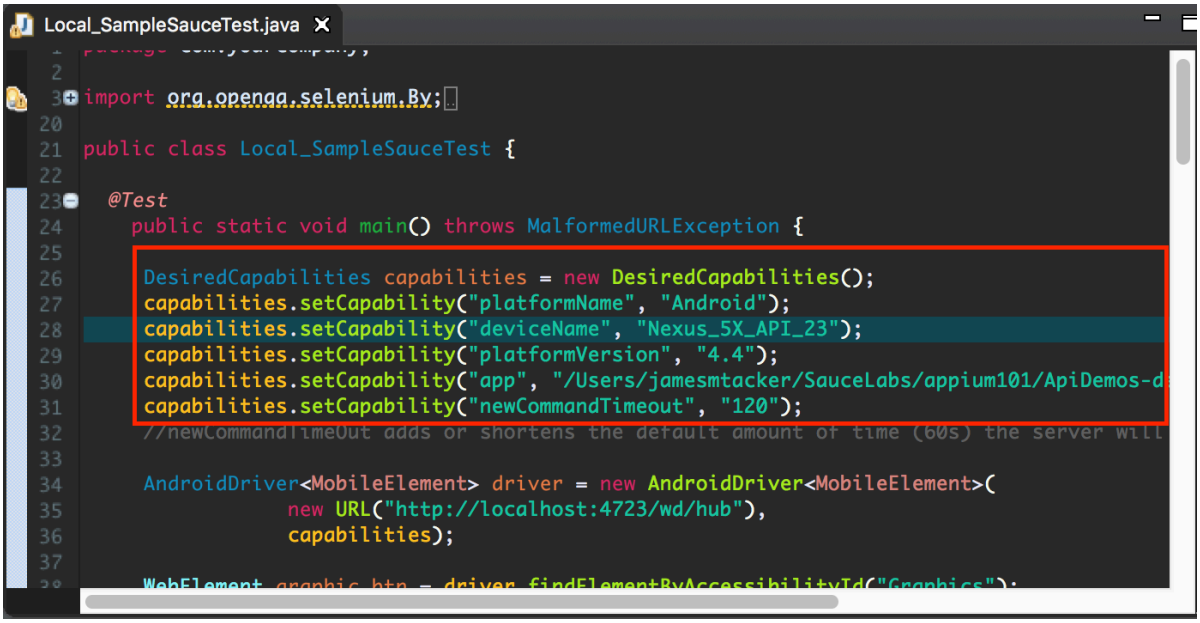


4 Edit Script Capabilities

1. Open the sample project in the Eclipse GUI by choosing
    a. **`local-java-testing-appium-android>src/test/java>com.yourcompany.com>Local_SampleSauceTest.java`**

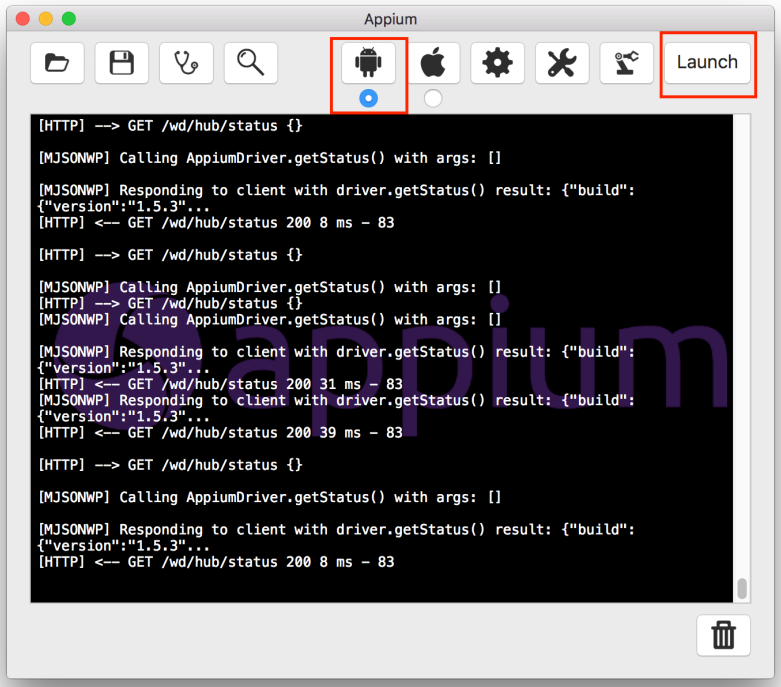


2. With the script open, edit the the following details so that the WebDriver capabilities reflect the Appium Server, and emulator that you want to use (for this example we're using Android SDK platform 4.4 and a Nexus_5X tablet with API 23). You also need to edit the "app" location (**`ApiDemos-debug.apk`**)—this is usually in the same location as your Eclipse project folder.



5 Start the Appium Server

1. Open Appium
2. Click the Android Icon
3. Select Launch



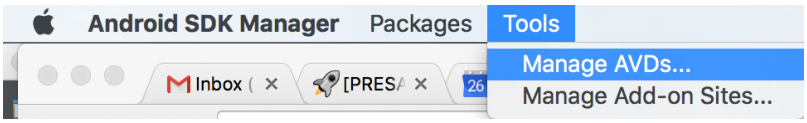
6 Launch the Emulator

1. Open a terminal
2. Navigate to your home directory and launch the SDK manager

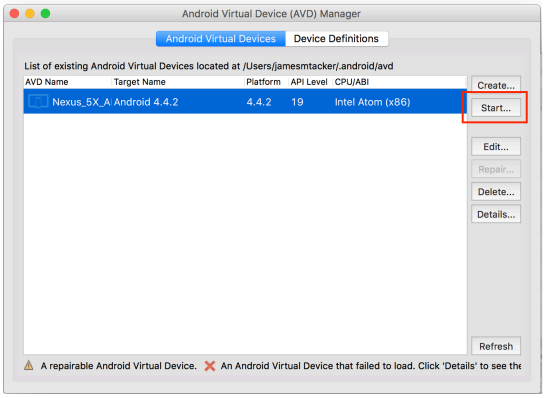
```
$ cd $ANDROID_HOME
$ cd tools

$ ./android
```

3. Select Tools



4. Select the AVD (Android Virtual Device) you wish to test, then select **Start**.



7 Test the Script

1. Back in Eclipse, right click your project and select **Run As > Maven Test**.