Table of Contents

[1. Install mobile device driver 1](#_Toc415214929)

[2. Install Android SDK 2](#_Toc415214930)

[3. Install Appium 4](#_Toc415214931)

[4. Create AVD 5](#_Toc415214932)

[5. Profile settings for Appium and Android 5](#_Toc415214933)

[**I.** **Profile** **Settings for Appium**  5](#_Toc415214934)

[**II.** **Profile settings for Android Emulator**  7](#_Toc415214935)

[**III.** **Profile settings for Android Device**  9](#_Toc415214936)

[6. Wireless connection using ABD 16](#_Toc415214937)

[**I.** **Connect using Wi-Fi network**  16](#_Toc415214938)

# Install mobile device driver

Installing Mobile Device Driver (for Samsung) on windows : -

1. Download **Samsung-Usb-Driver-v1.5.49.0.exe** from <http://androidxda.com/download-samsung-usb-drivers>
2. Enable USB debugging on the mobile and then connect the mobile to the system and it should install the driver software and open a new window for the mobile connected
3. Ensure that the connected device is listed in Device Manager

Note: download the driver based on the mobile model used

Also install Intel HAXM ([haxm-windows\_r04.zip (1.0.8)](https://software.intel.com/en-us/android/articles/intel-hardware-accelerated-execution-manager-end-user-license-agreement) from <https://software.intel.com/en-us/android/articles/intel-hardware-accelerated-execution-manager> ) to work with Android AVD

# Install Android SDK

**Pre-requisites :**

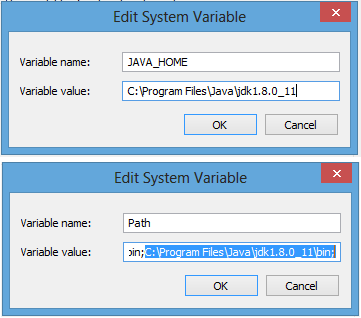
1. Install JDK in your machine. (java version "1.8.0\_11")

**Follow the below steps :**

1. Download **Android Studio** from <http://developer.android.com/sdk/installing/studio.html> and run the application - “android-studio-bundle-135.1245622-windows.exe”

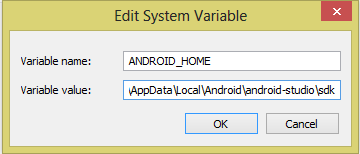
(As the studio plugins consumes more memory, it is safe to install the software in the drive having more memory space)

1. After installation set the path for Android Studio
   1. create an environment variable ‘**JAVA\_HOME** ‘ with value ‘**C:\Program Files\Java\jdk1.8.0\_11**’
   2. ensure that the **path** variable also has the value ‘**C:\Program Files\Java\jdk1.8.0\_11\bin**’

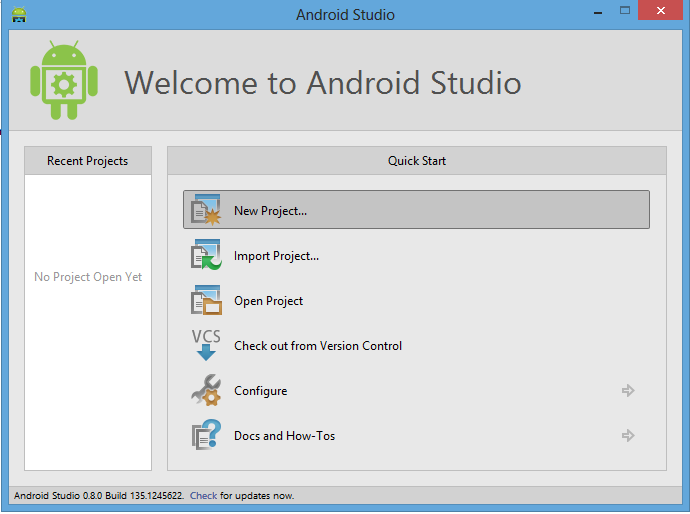


* 1. create an environment variable ‘**ANDROID\_HOME**’with value

‘/path/to/Android/android-studio/sdk’ and ensure that the **ANDROID\_HOME** is added to **path** variable



1. Now launch the Android Studio application



# Install Appium

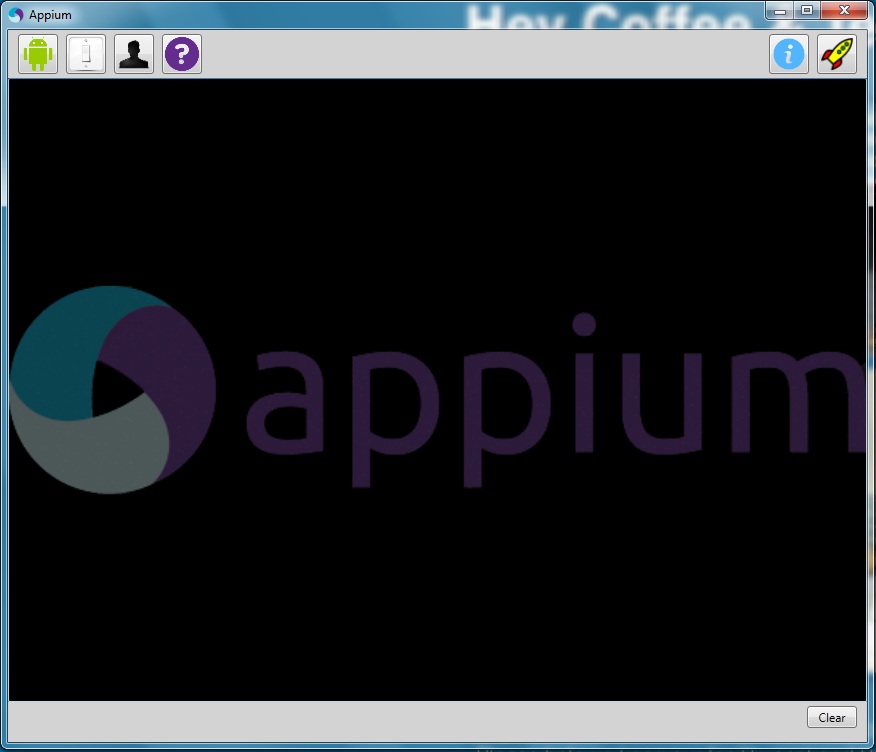
**Pre-requisite: -**

1. Download .net framework 4.5 from <http://www.microsoft.com/en-in/download/> (under menu Products/Microsoft .Net Framework)
2. Install the downloaded set-up file

Note : .net framework 4.5 is bundled with Windows 8 operating systems and hence no need to install .net 4.5 separately

**Installing appium on windows : -**

1. Download the latest Appium zip file (AppiumForWindows-1.1.0.0.zip) from <https://bitbucket.org/appium/appium.app/downloads/>
2. Unzip the file under “C:\Appium” directory
3. Now, run the Appium application (having the Appium logo as icon) inside the unzipped file and it should show a window like this



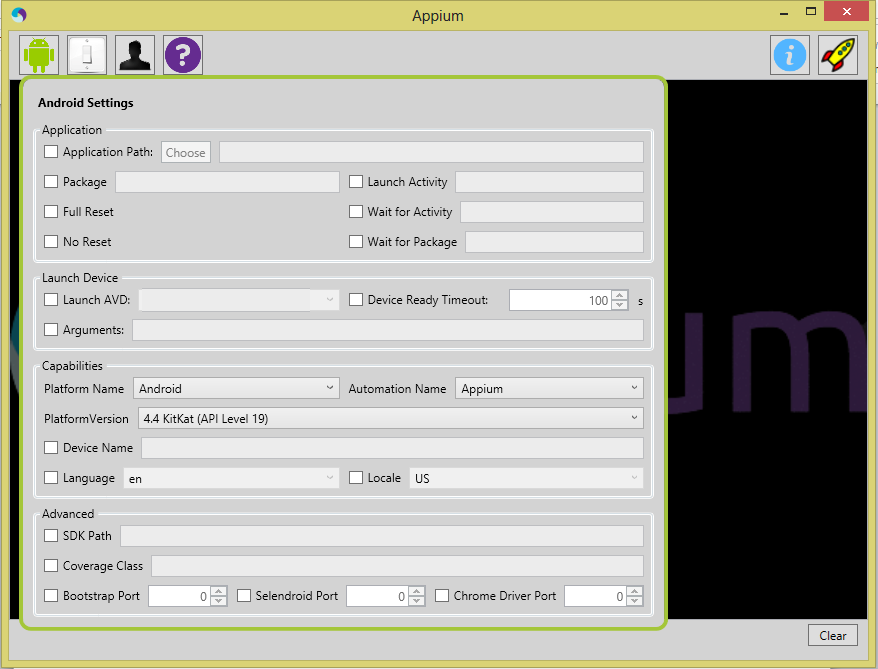
# Create AVD

**Follow the below steps : -**

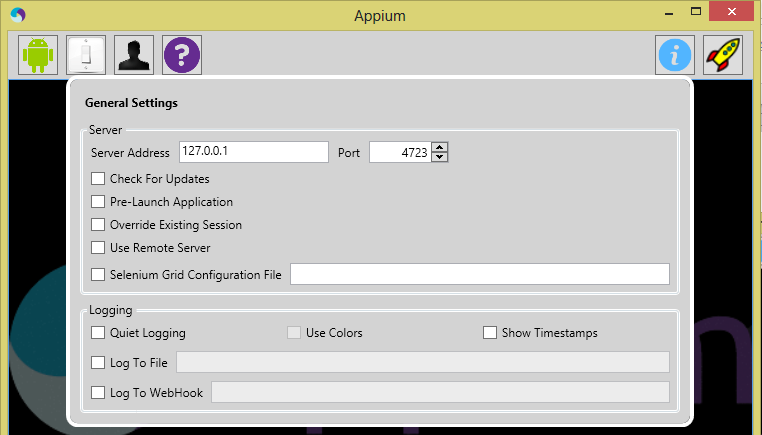
1. Launch Android Studio
2. Go to Tools >> Android >> SDK Manager and select the required Android version(s) and install them
3. To create a new AVD go to Tools >> Android >> AVD Manager , click “Create” button to create a new AVD profile, fill the required fields
4. Select the created AVD profile and click “Start” button to launch the AVD
5. The AVD can also be launched from command line :
   1. Go to “..\android-studio\sdk\platform-tools” in command prompt and type “emulator –avd emulator\_name”
   2. This will launch the AVD and type “adb devices” to list the emulators launched
   3. If the current emulator is not listed, then type “adb kill-server” and “adb start-server” then type “adb devices”, it should list the currently running emulators
6. To install an app in emulator then go to “..\android-studio\sdk\platform-tools” in command prompt and type “adb install path/to/apk\_file.apk”

# Profile settings for Appium and Android

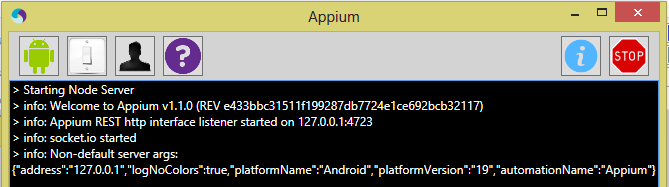
1. **Profile Settings for Appium :**
2. Launch Appium
3. Click “Android Settings” button (with image of Android logo)
4. Under Capabilities section set Platform name to “Android”, Automation name to “Appium” and Platform version to the required Android version



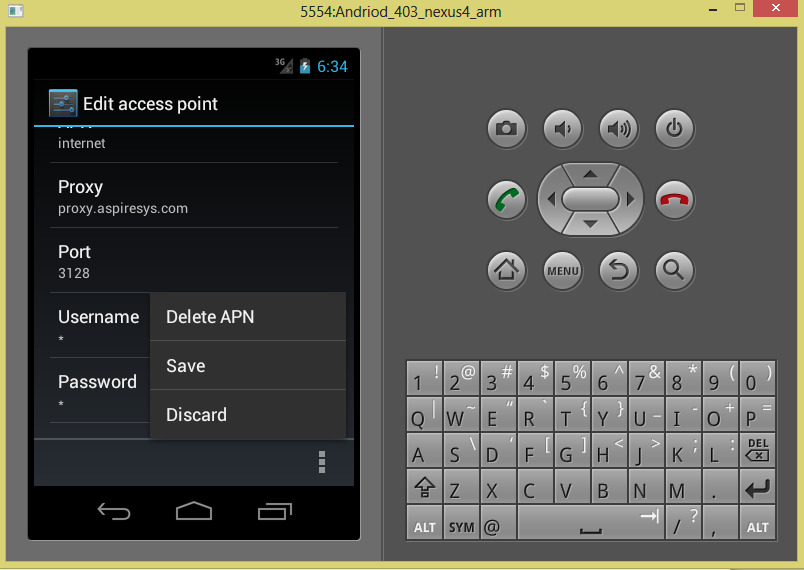
1. Click “General settings” button (with image of Switch like)
2. Under Server section set Server Address to “127.0.0.1” and Port to “4723”



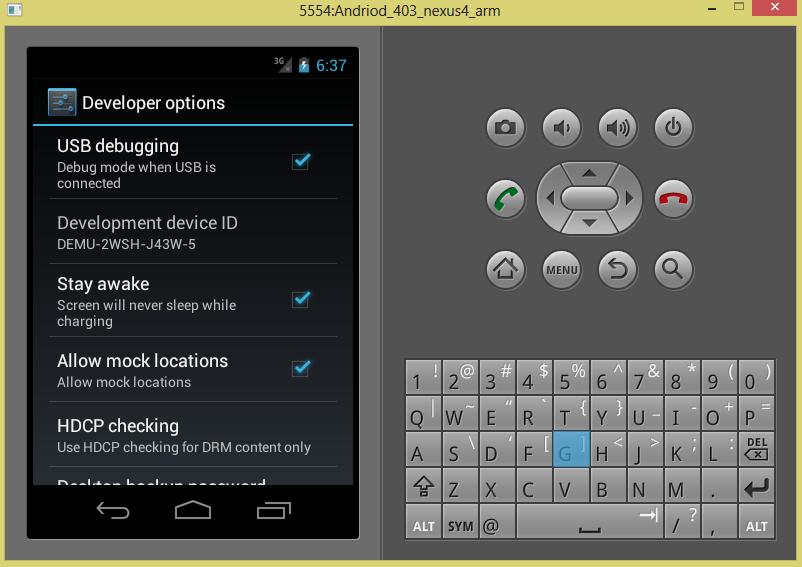
1. That’s all. Click the launch button (with image of rocket) to start the appium server and again click it to stop it.
2. This should be the appium server log



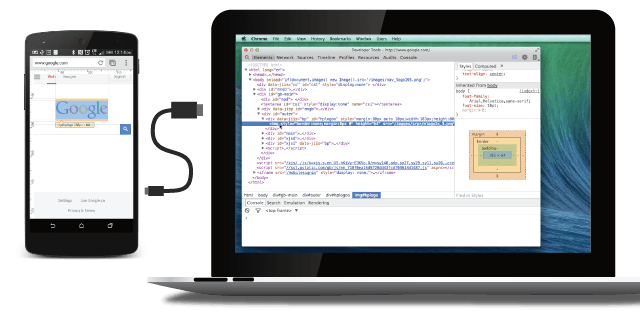
1. **Profile settings for Android Emulator :**
2. Launch the emulator
3. Go to Settings >> More >> Mobiles networks >> Access Point Names >> click the Option displayed (Telkila) and give only the proxy details and Save it



1. Go to Settings >> Developer options and enable “USB debugging”



1. **Profile settings for Android Device :**



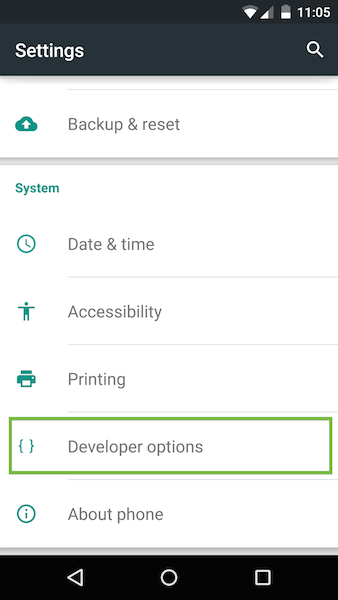
* Chrome 32 or later installed on your development machine.
* A USB cable to connect your Android device.
* **For browser debugging**: Android 4.0+ and [**Chrome for Android**](https://play.google.com/store/apps/details?id=com.android.chrome&hl=en).
* **For app debugging**: Android 4.4+ and a WebView [**configured for debugging**](https://developer.chrome.com/devtools/docs/remote-debugging#debugging-webviews).

**Note**: Remote debugging requires your version of desktop Chrome to be newer than the version of Chrome for Android on your device. For best results, use [**Chrome Canary**](https://www.google.com/intl/en/chrome/browser/canary.html) (Mac/Windows) or the Chrome [**Dev channel**](http://www.chromium.org/getting-involved/dev-channel)release (Linux) on desktop

Follow these instructions to set up your Android device for remote debugging.

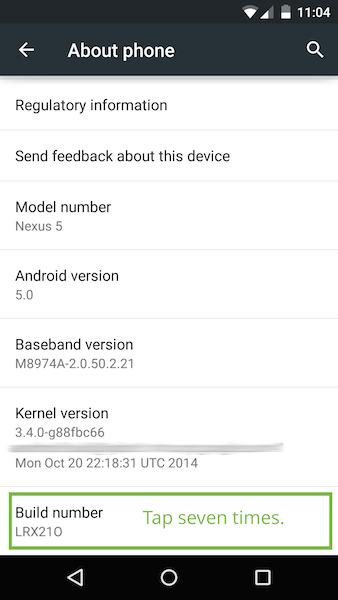
**Enable USB debugging**

On your Android device, select **Settings > Developer options**.

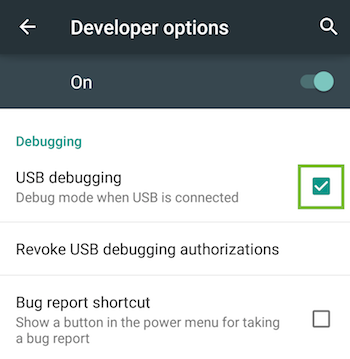


**Developer options** on the **Settings** page.

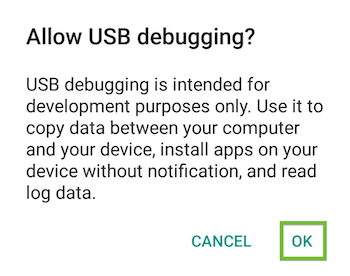
Note: On Android 4.2 and later, the developer options are hidden by default. To enable the developer options, select Settings > About phoneand tap Build number seven times.



In **Developer options**, select the **USB debugging** checkbox:



An alert prompts you to allow USB debugging. Tap **OK**.

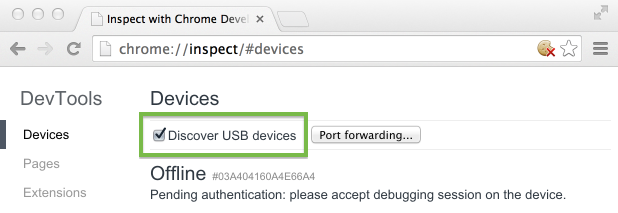


**Connecting device to PC**

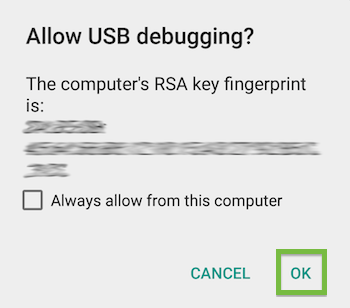
Connect the Android device to your development machine using a USB cable.

Note : If you are developing on **Windows**, install the appropriate USB driver for your device. See [**OEM USB Drivers**](http://developer.android.com/tools/extras/oem-usb.html) on the Android Developers' site.

After setting up remote debugging on Android, discover your device in Chrome.On your desktop Chrome browser, navigate to **chrome://inspect**. Confirm that **Discover USB devices** is checked:

**Tip**: You can also get to **chrome://inspect** by selecting **Chrome menu > More tools > Inspect Devices**.

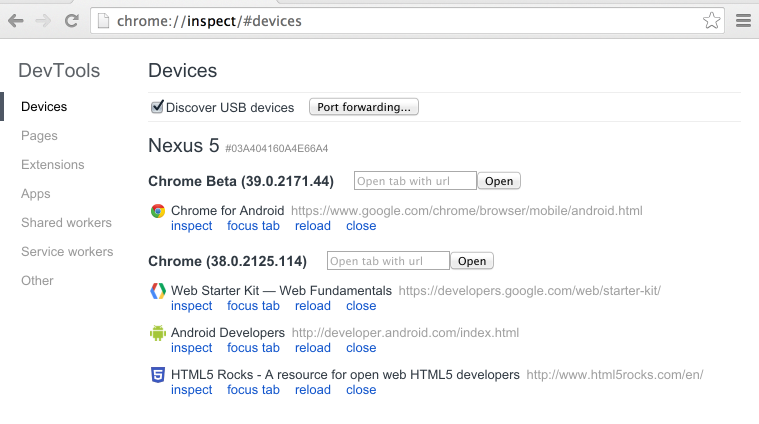
On your device, an alert prompts you to allow USB debugging from your computer. Tap **OK**.



**Tip**: To skip this alert in the future, check **Always allow from this computer**.

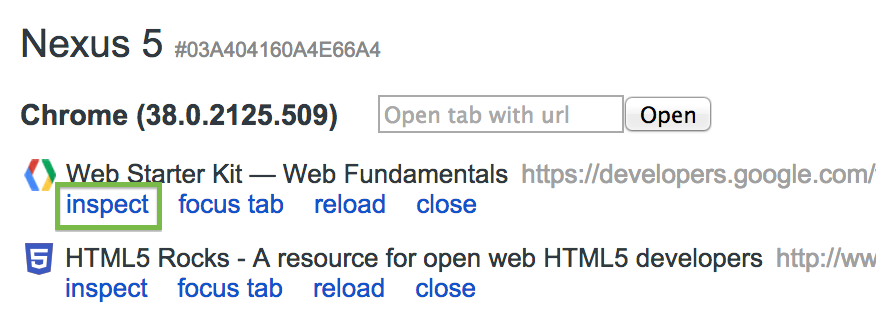
The message **USB debugging connected** displays in the device's notification drawer.

On your computer, the **chrome://inspect** page displays every connected device, along with its open tabs and debug-enabled WebViews.

Viewing connected devices from the **chrome://inspect** page.

From the **chrome://inspect page**, you can launch DevTools and debug your remote browser tabs.

To start debugging, click **inspect** below the browser tab you want to debug.



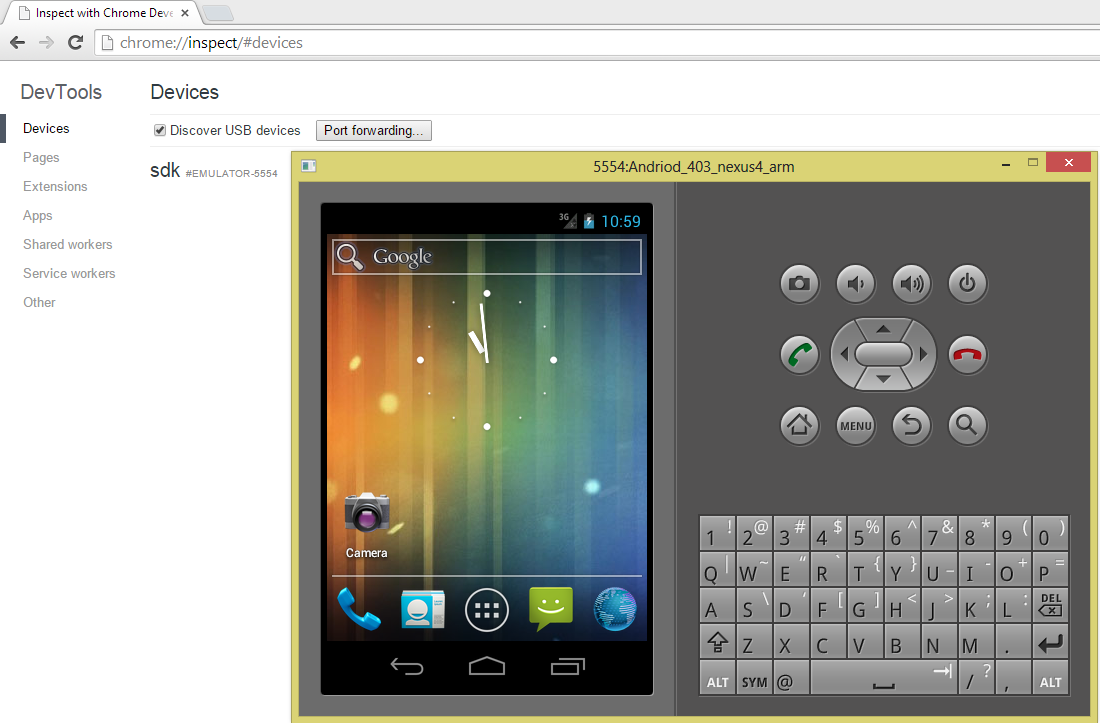
A new instance of Chrome DevTools launches on your computer. From this instance, you can interact with the selected browser tab on your device in real time.



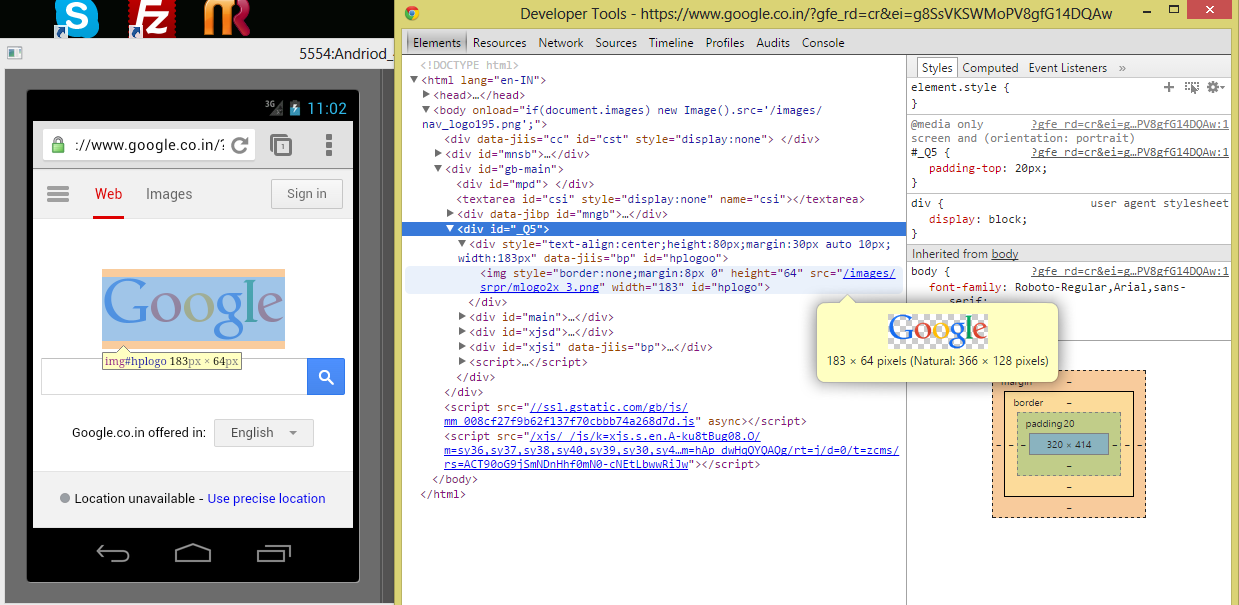
Debug a web page on your Android phone from your laptop using Chrome DevTools.

For example, you can use DevTools to inspect web page elements on your device:

* When you mouse over an element in the **Elements** panel, DevTools highlights the element on your device.
* You can also click the **Inspect Element** inspect element icon icon in DevTools and tap your device screen. DevTools highlights the tapped element in the **Elements** panel.
* You can do the same with Android emulator. Launch the emulator and go to **chrome://inspect/#devices** in your chrome browser and it will list the emulator launched



* Now you can launch the browser in emulator and **inspect** the web page

****

# Wireless connection using ABD

* + 1. **Connect using Wi-Fi network :**

1. Connect Android device and adb host computer to a common Wi-Fi network accessible to both
2. Connect the device with USB cable to PC
3. Open command prompt type

***adb devices***

it should list the device connected to PC

1. Now to set ABD to the TCP/IP port for remote connection, type in prompt :

***adb tcpip 5555***

1. Disconnect the device from PC
2. Now connect your device to ADB by typing this command :

***adb connect* xxx.xxx.xxx.xxx**

(IP address of the device, which will present in Settings>About phone)

1. Now in command prompt you should see the result like :

**\* daemon not running. starting it now on port 5037 \***

**\* daemon started successfully \***

**connected to xxx.xxx.xxx.xxx:5555**

1. To disconnect device from TCP/IP port:

***adb disconnect* xxx.xxx.xxx.xxx**