Tutorial and Demos on Android

CSC 4320/6320 - Operating Systems

Spring 2018

Outline

- Introduction of Android App Development.
- How to install Android Studio?
- Environment setup in Android Studio.
- Demo 1: Develop a Hello World App.
- Demo 2: A simple Process Manager App.

ANDROID APP DEVELOPMENT

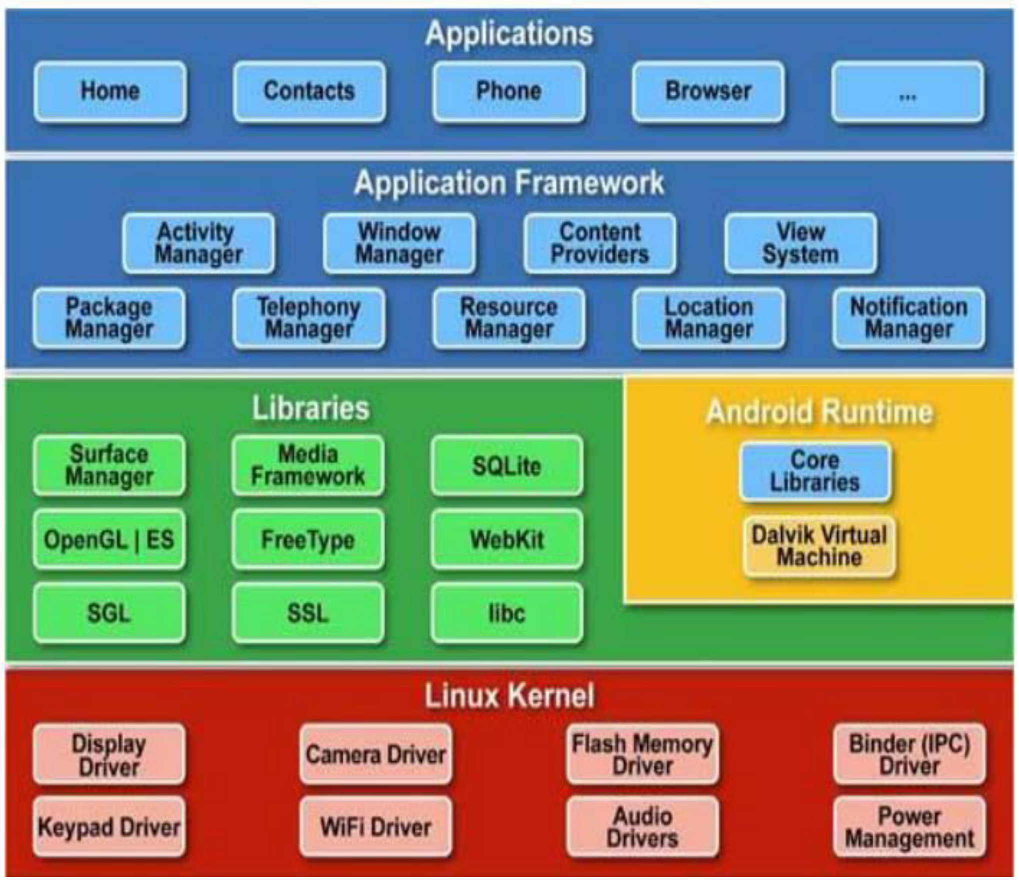
Introduction

- Android is an open source and Linux-based operating system for mobile devices such as smartphones and tablet computers.
- Developed by the Open Handset Alliance, led by Google, and other companies.
- Apps are developed in the **Java** language using the Android Software Development Kit.
- First beta version of the Android Software Development Kit (SDK) was released by Google in 2007.
- First commercial version, Android 1.0, was released in September 2008.
- Current Android version - Android 8.0 Oreo.

Android vs iOS

- Pros
  - Easy to program : Java based
  - Can be developed on any OS : Mac, Windows, Linux
- Cons
  - Too many different types of devices to support
  - Quality control

Android Architecture



Android Architecture

- 4 Layered Architecture
  - Linux Kernel

    Process management, memory management, device management
  - Libraries

    SQLite database, web browser engine WebKit
  - Android Runtime

    Dalvik Virtual Machine, kind of JVM optimized for Android
  - Application Framework

    Provide high level services to applications like Telephony, Location
  - Applications

    Android applications are written in this layer

Basic Components of Android Application

- Activities

Dictate the UI and handle the user interaction to the smartphone screen

- Services

Handle background processing associated with an application.

- Broadcast Receivers

Handle communication between Android OS and applications.

- Content Providers

Handle data and database management issues.

Managing the Activity Lifecycle

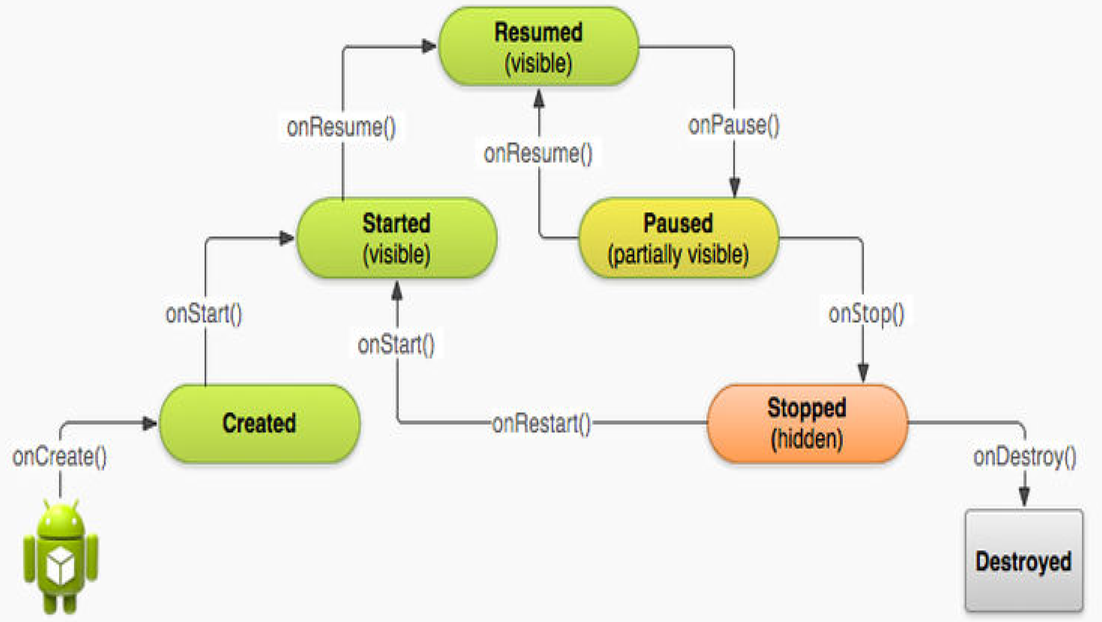


Figure 1. A simplified illustration of the Activity lifecycle, expressed as a step pyramid. This shows how, for every callback used to take the activity a step toward the Resumed state at the top, there's a callback method that takes the activity a step down. The activity can also return to the resumed state from the Paused and Stopped state.

Lifecycle methods

- onCreate(), onStart(), onResume(), onPause(), onStop(), onDestroy()
- Implementation of lifecycle methods is important because it ensures that application behaves well in following scenarios:
  - Does not crash if the user receives a phone call or switches to another app while using your app.
  - Does not consume valuable system resources when the user is not actively using it.
  - Does not lose the user's progress if they leave your app and return to it at a later time.
  - Does not crash or lose the user's progress when the screen rotates between landscape and portrait orientation.

Development Environment Setup

- Android Studio
  - Step 1: Set up Java SDK
  - Step 2: Android Studio

Testing the Android App

- Run on Emulator – Android Virtual Device

AVD Manager is used to configure a virtual device that models a specific device

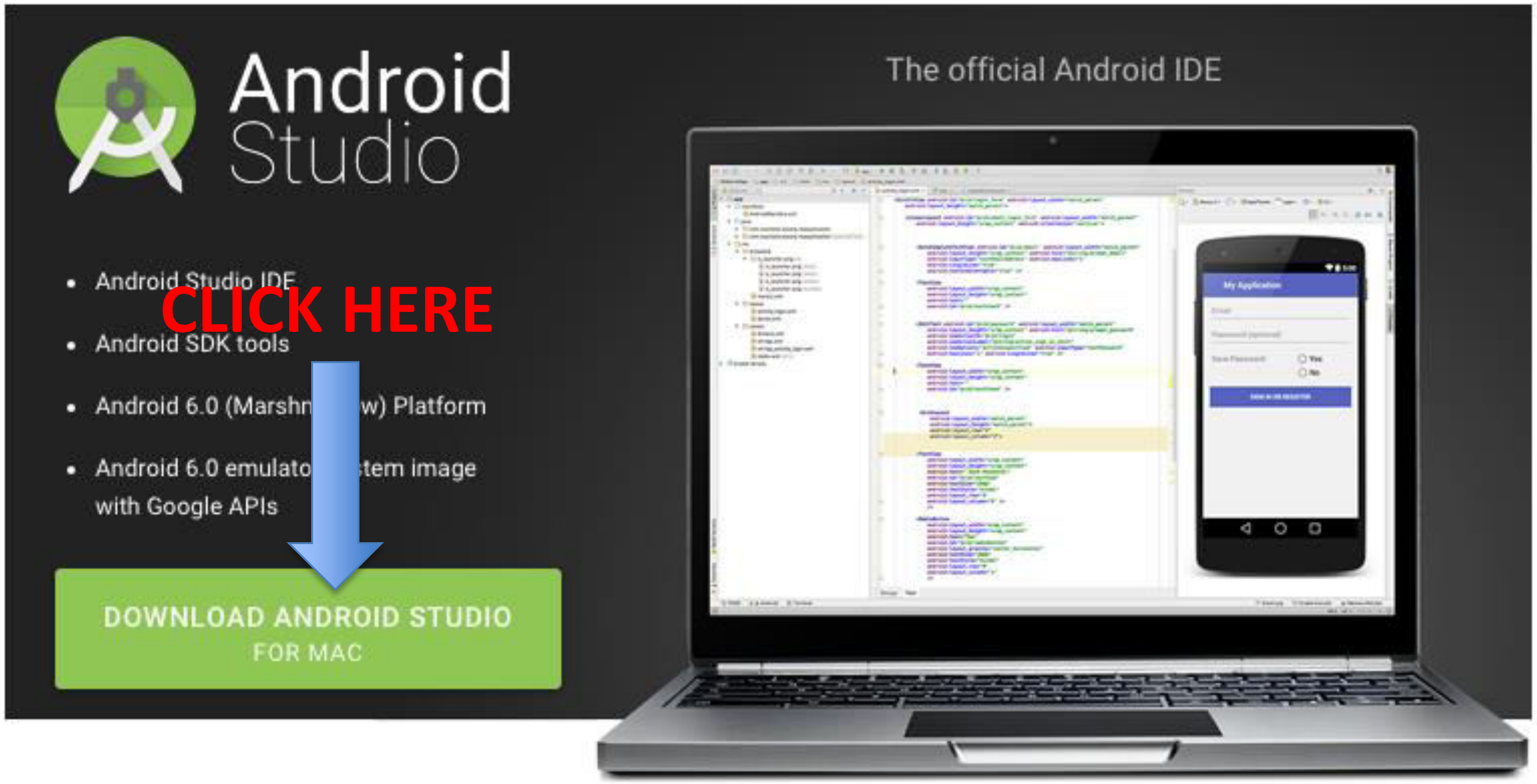
- Run on real Android device

Developer mode should enabled on the mobile device

HOW TO INSTALL ANDROID STUDIO?

How To Install Android Studio?

- Download Android Studio from website below and install it.
  - http://developer.android.com/sdk/index.html

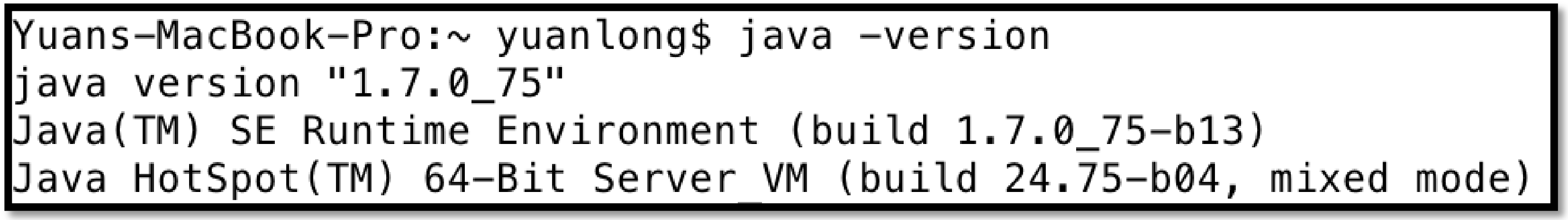


- Android Studio IDE
- Android SDK tools
- Android 6.0 (Marshmallow) Platform
- Android 6.0 emulator system image with Google APIs

DOWNLOAD ANDROID STUDIO
FOR MAC

How To Install Android Studio?

- Make sure that JDK 1.7 or higher has been installed.
  - How to check?
    - In your terminal or cmd, type ***java –version***



```
Yuans-MacBook-Pro:~ yuanlong$ java -version
java version "1.7.0_75"
Java(TM) SE Runtime Environment (build 1.7.0_75-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.75-b04, mixed mode)
```

  - Android studio should include wizard to examine the system requirement

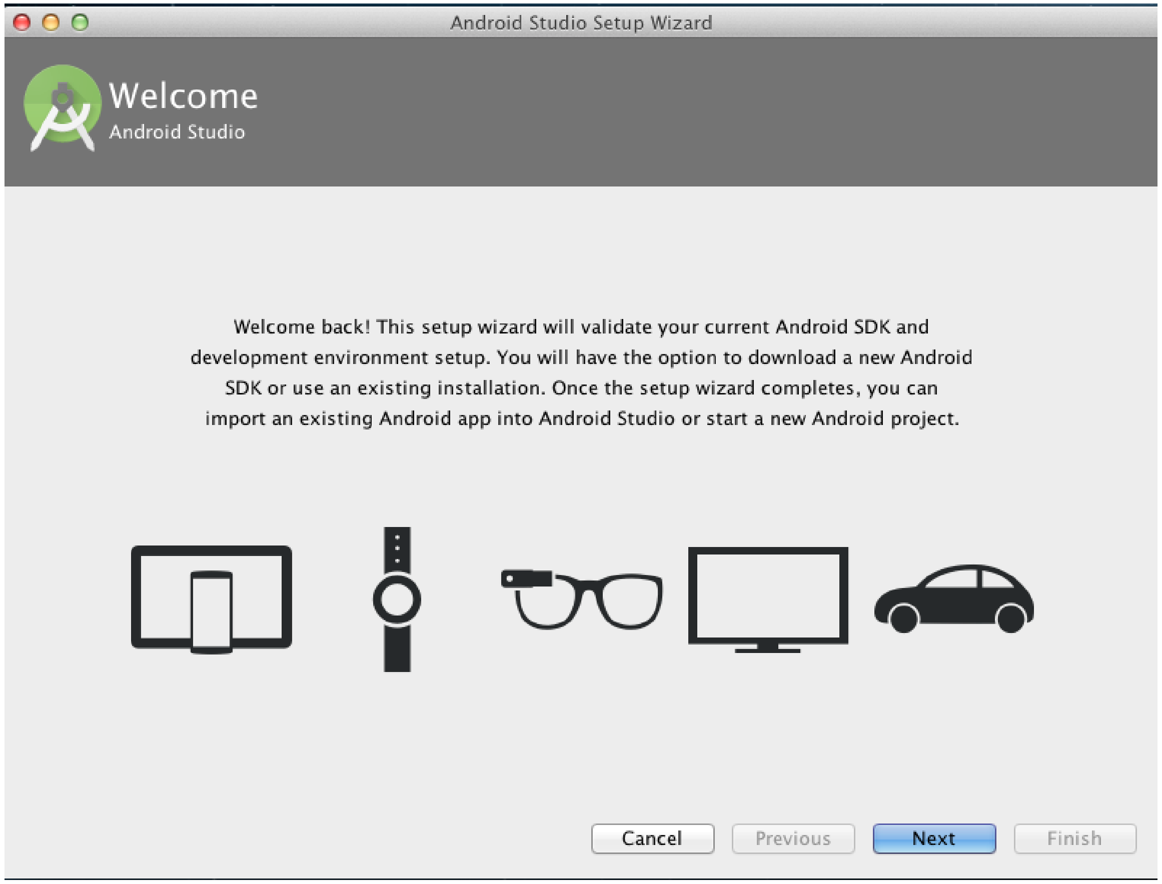
ENVIRONMENT SETUP IN ANDROID STUDIO

Developer Workflow

- **Environment Setup**.
  - Install Android SDK (bundled with Android Studio).
  - Create Android Virtual Device or connect an Android device.
- Project Setup and Development.
- Build
  - Generate .apk file using Gradle.
- Debug
  - Device log messages(logcat)
- Test
  - In emulator or your Android device.

Environment Setup

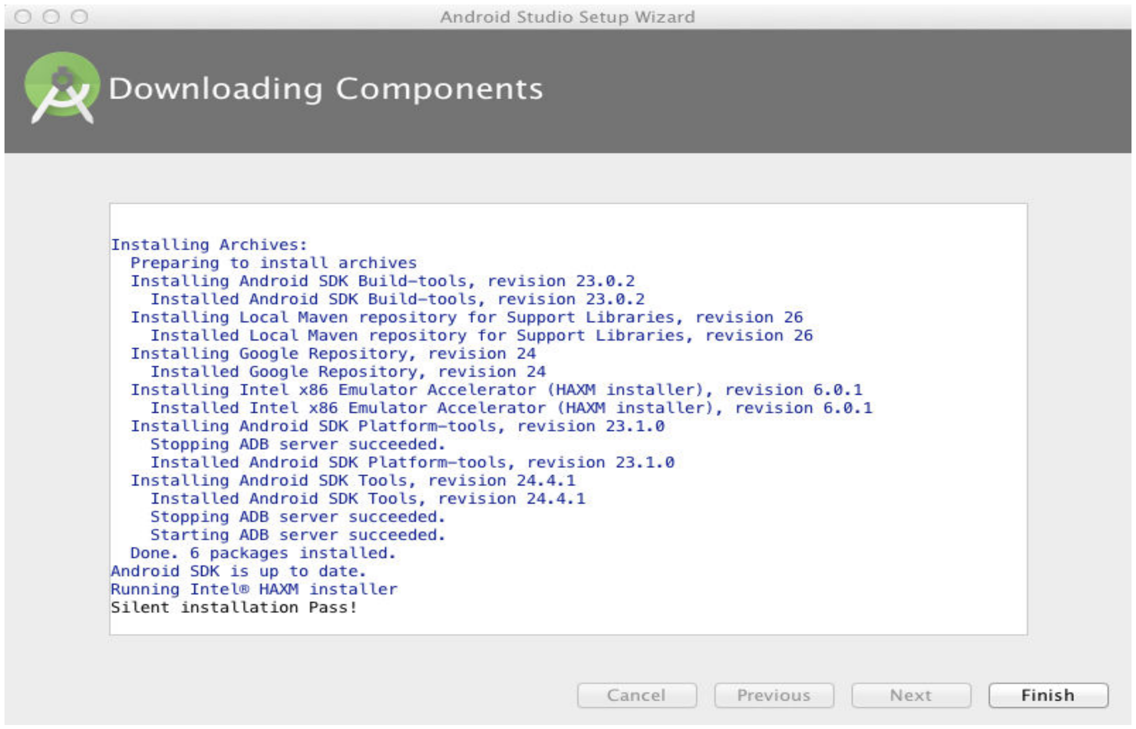
- Set up environment by following the "Setup Wizard".



Welcome back! This setup wizard will validate your current Android SDK and development environment setup. You will have the option to download a new Android SDK or use an existing installation. Once the setup wizard completes, you can import an existing Android app into Android Studio or start a new Android project.

Environment Setup

- Setup Wizard guides you to download required SDK and components.
- Click "finish" when downloading is done.



```
Installing Archives:
  Preparing to install archives
  Installing Android SDK Build-tools, revision 23.0.2
    Installed Android SDK Build-tools, revision 23.0.2
  Installing Local Maven repository for Support Libraries, revision 26
    Installed Local Maven repository for Support Libraries, revision 26
  Installing Google Repository, revision 24
    Installed Google Repository, revision 24
  Installing Intel x86 Emulator Accelerator (HAXM installer), revision 6.0.1
    Installed Intel x86 Emulator Accelerator (HAXM installer), revision 6.0.1
  Installing Android SDK Platform-tools, revision 23.1.0
    Stopping ADB server succeeded.
    Installed Android SDK Platform-tools, revision 23.1.0
  Installing Android SDK Tools, revision 24.4.1
    Installed Android SDK Tools, revision 24.4.1
    Stopping ADB server succeeded.
    Starting ADB server succeeded.
  Done. 6 packages installed.
Android SDK is up to date.
Running Intel® HAXM installer
Silent installation Pass!
```

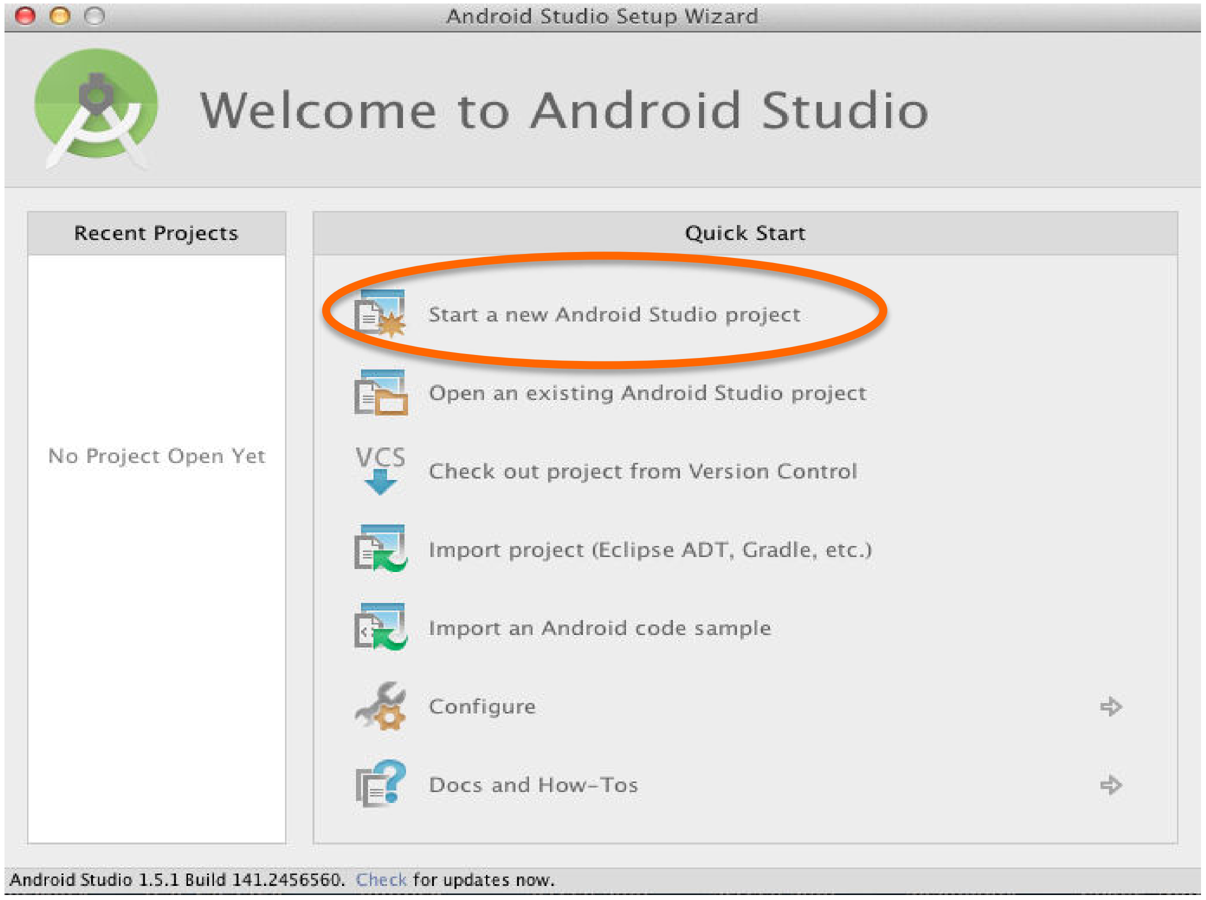
DEMO 1 : DEVELOP A HELLO WORD APP

Developer Workflow

- Environment Setup.
  - Install Android SDK (bundled with Android Studio).
  - Create Android Virtual Device or connect an Android device.
- **Project Setup and Development.**
- Build
  - Generate .apk file using Gradle.
- Debug
  - Device log messages(logcat)
- Test
  - In emulator or your Android device.

Create A New Project

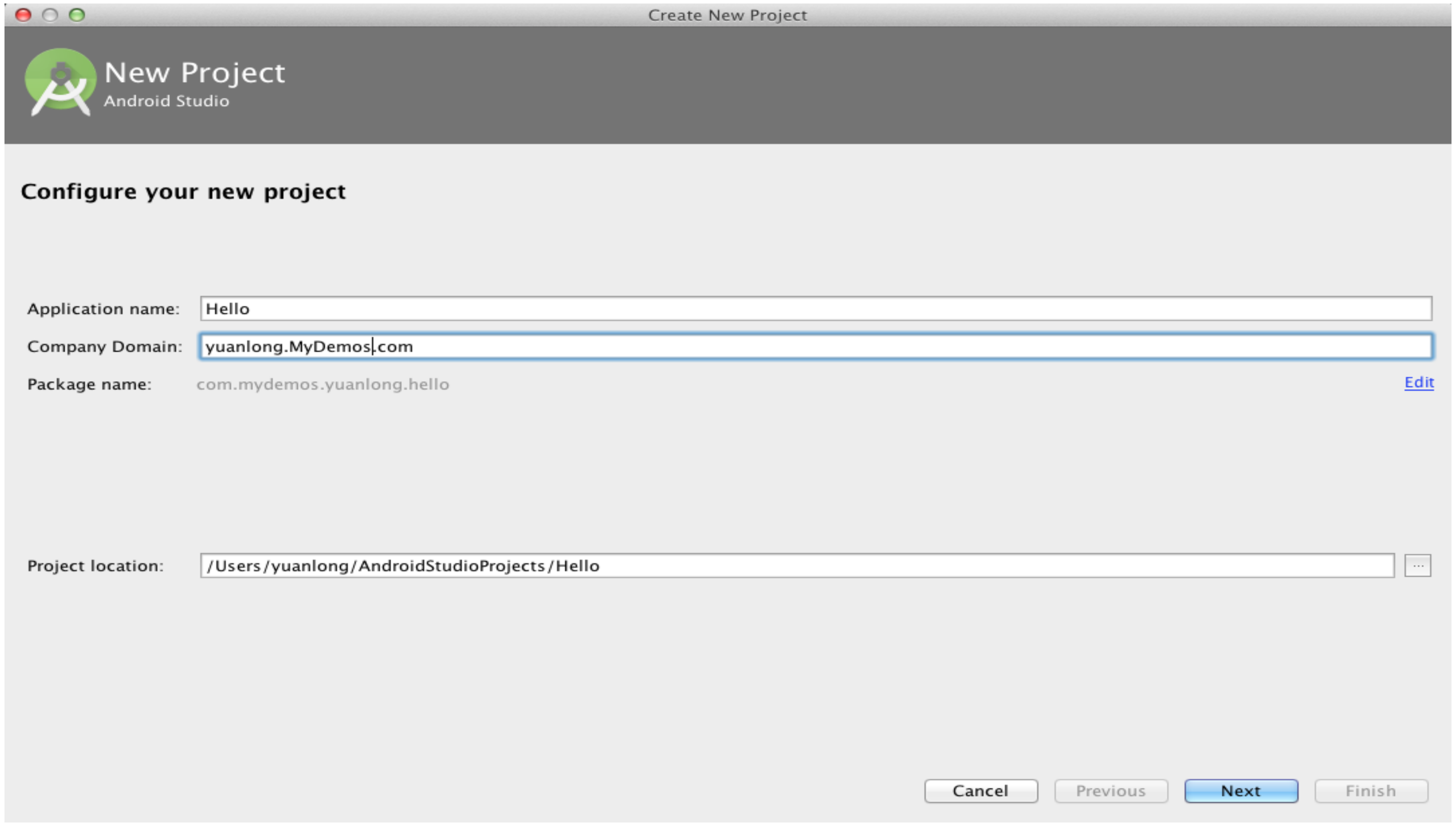
- Start a new Android Studio Project.


Android Studio Setup Wizard
Welcome to Android Studio
Recent Projects
No Project Open Yet
Quick Start
Start a new Android Studio project
Open an existing Android Studio project
VCS
Check out project from Version Control
Import project (Eclipse ADT, Gradle, etc.)
Import an Android code sample
Configure
Docs and How-Tos

Android Studio 1.5.1 Build 141.2456560. Check for updates now.

Create A New Project

- Define the name and company domain for your App. Then click “Next”.



Configure your new project

Application name: Hello

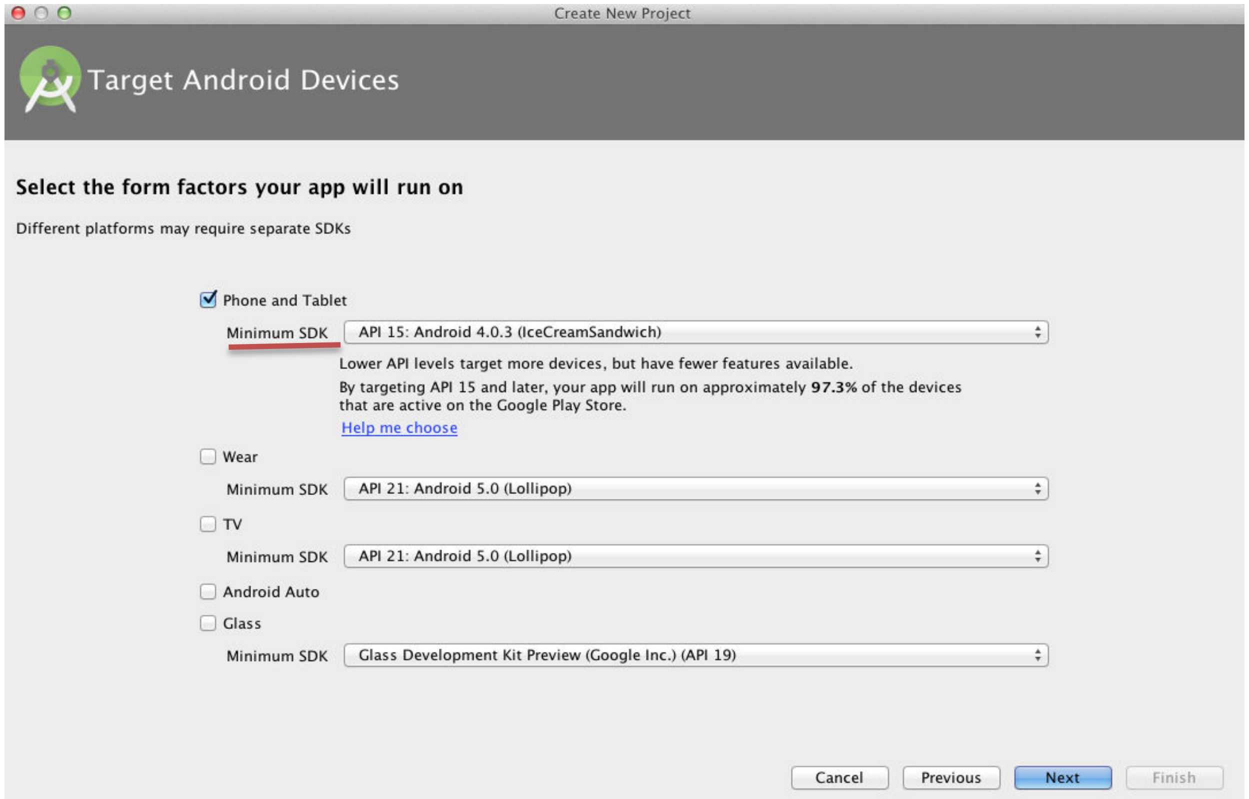
Company Domain: yuanlong.MyDemos.com

Package name: com.mydemos.yuanlong.hello

Project location: /Users/yuanlong/AndroidStudioProjects/Hello

Create A New Project

- Set the Minimum SDK. Then click "Next".
  - Configure the minimum API Level required for the app to run. You can keep the default setting.



Select the form factors your app will run on

Different platforms may require separate SDKs

[✓] Phone and Tablet

Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)

Lower API levels target more devices, but have fewer features available.

By targeting API 15 and later, your app will run on approximately **97.3%** of the devices that are active on the Google Play Store.

Help me choose

[ ] Wear

Minimum SDK: API 21: Android 5.0 (Lollipop)

[ ] TV

Minimum SDK: API 21: Android 5.0 (Lollipop)

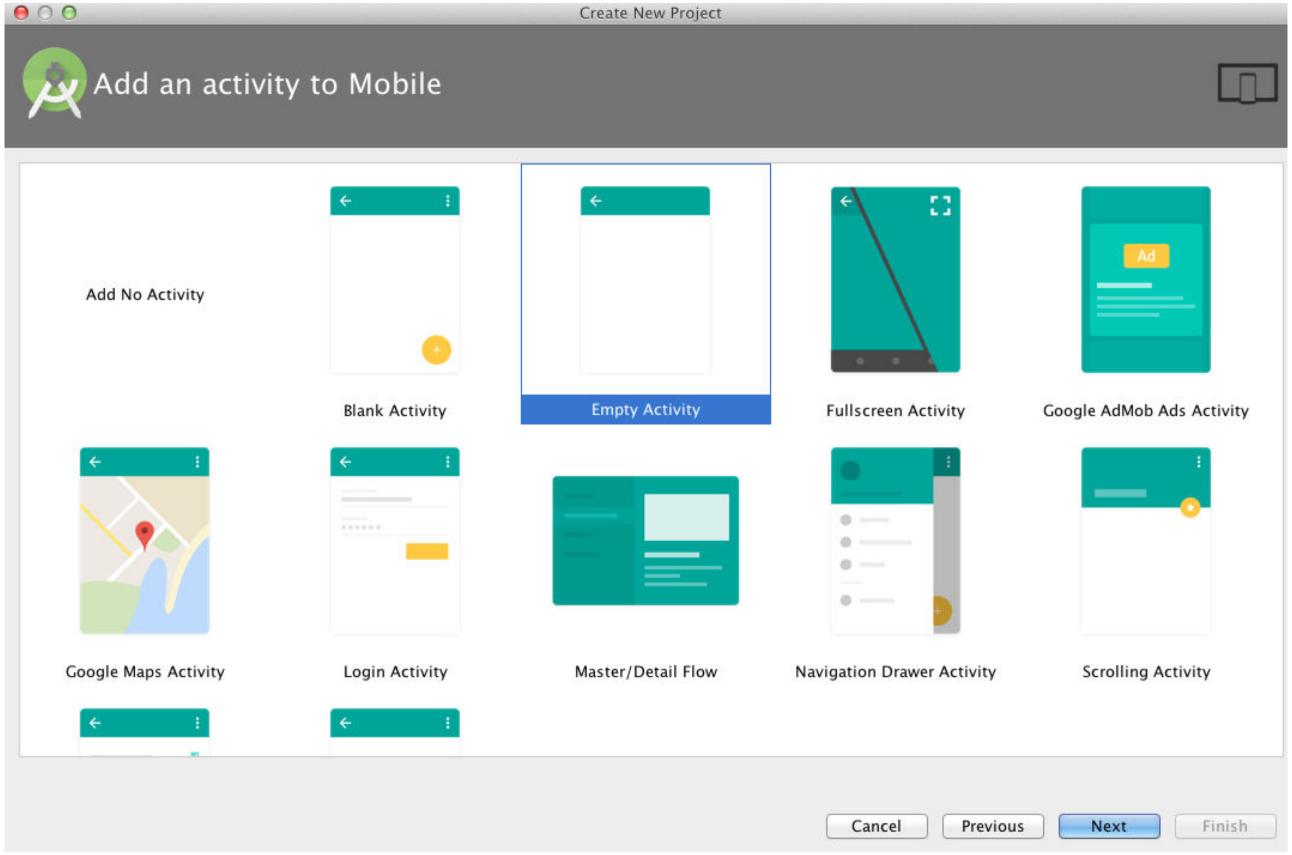
[ ] Android Auto

[ ] Glass

Minimum SDK: Glass Development Kit Preview (Google Inc.) (API 19)

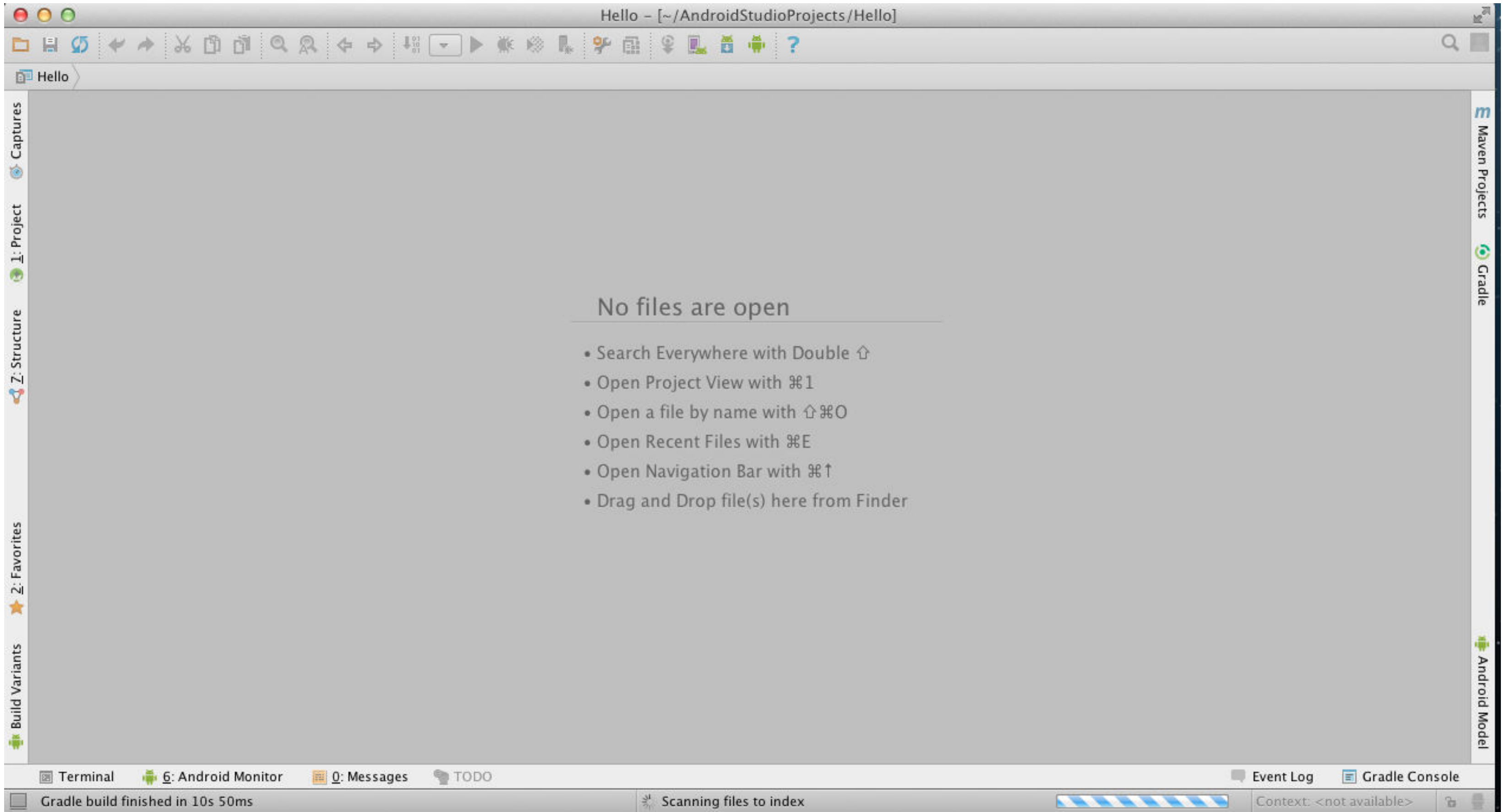
Create A New Project

- Add an "Empty Activity" to Mobile.



Create A New Project

- After that, your project will be successfully created.



Hello – [~/AndroidStudioProjects/Hello]

No files are open

- Search Everywhere with Double ⇧
- Open Project View with ⌘1
- Open a file by name with ⇧⌘O
- Open Recent Files with ⌘E
- Open Navigation Bar with ⌘↑
- Drag and Drop file(s) here from Finder

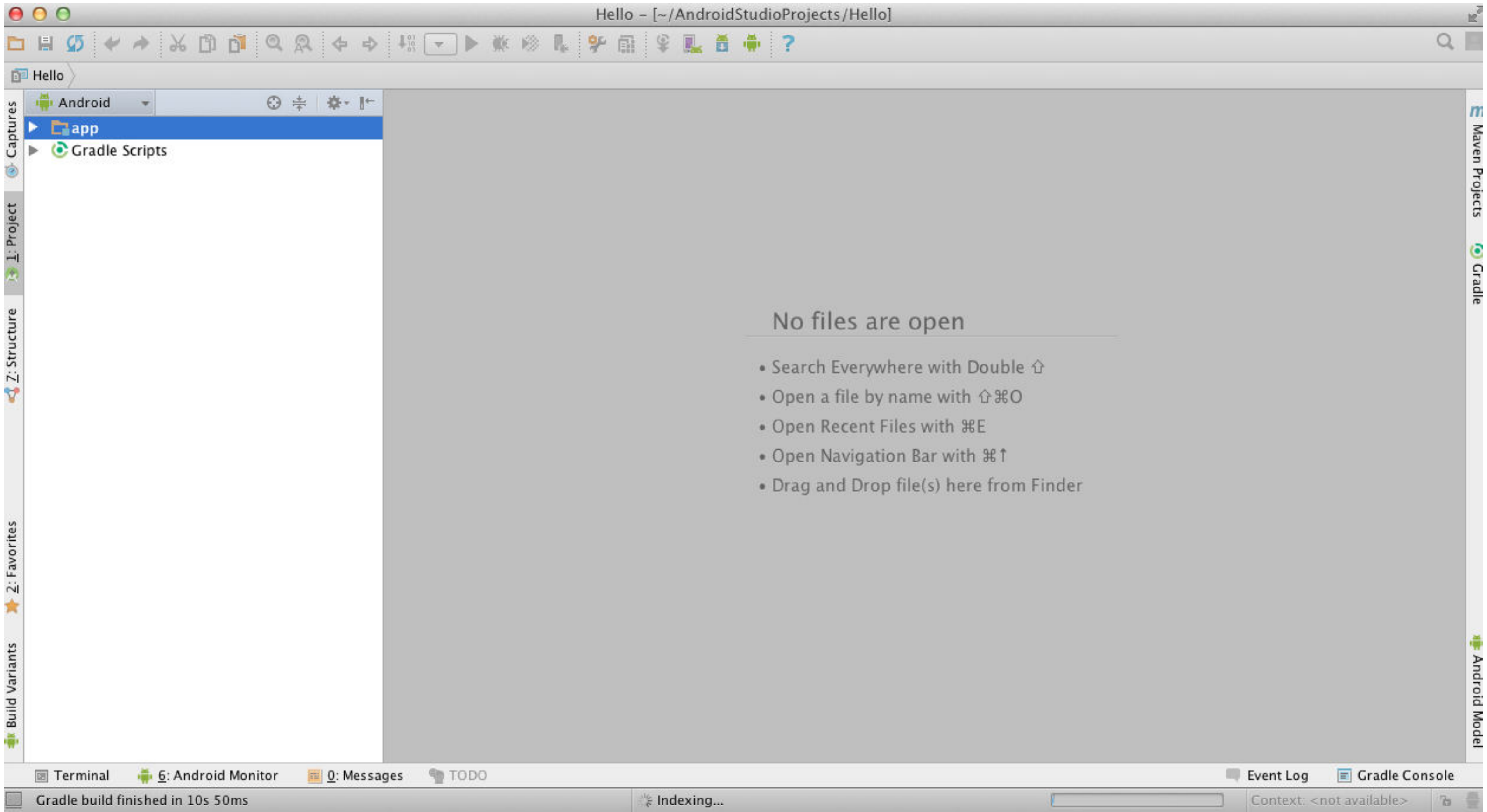
Gradle build finished in 10s 50ms

Scanning files to index

Context: <not available>

Create A New Project

- Open the “Project” view to start the development.



No files are open

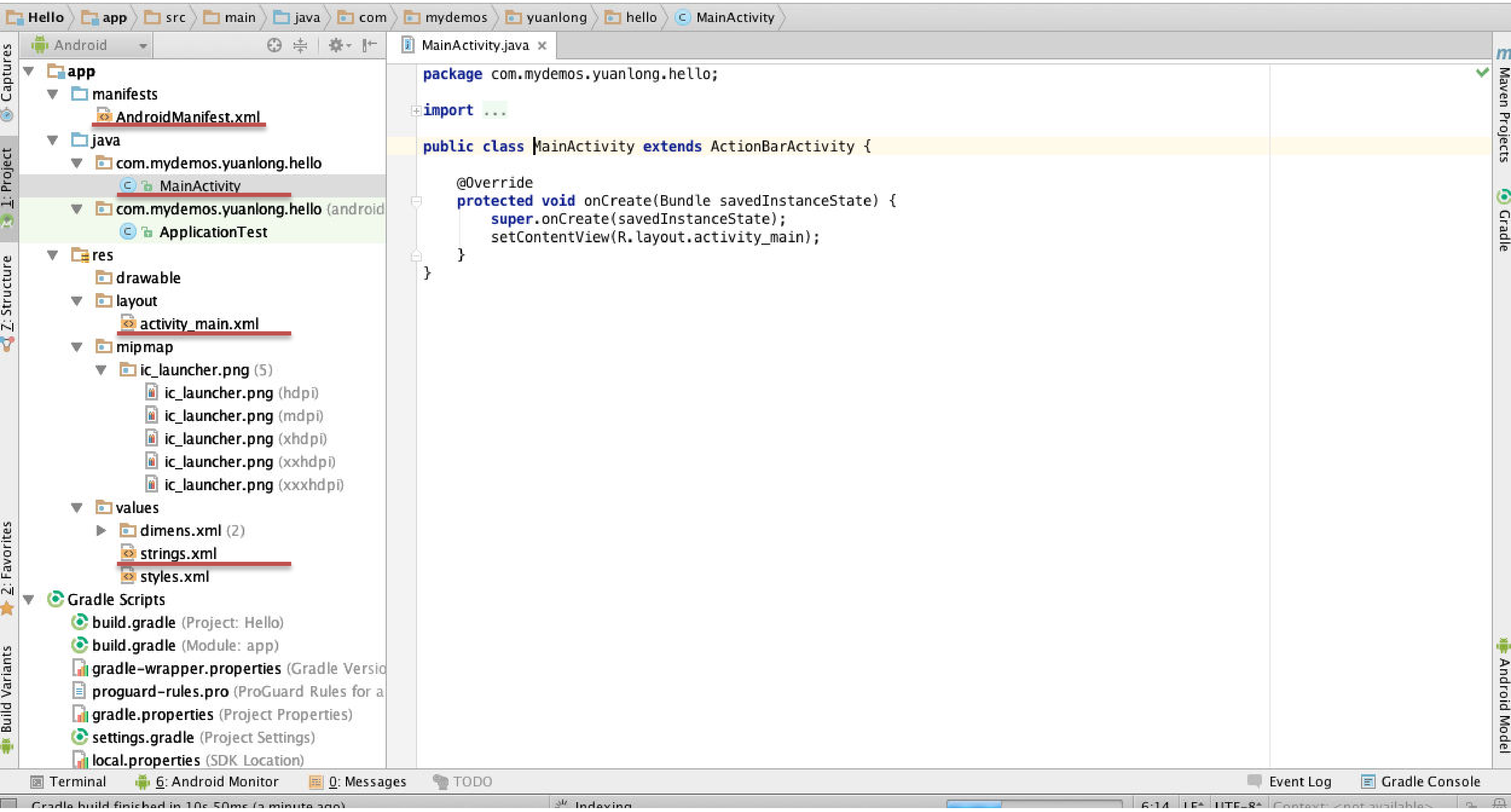
- Search Everywhere with Double ⇧
- Open a file by name with ⇧⌘O
- Open Recent Files with ⌘E
- Open Navigation Bar with ⌘↑
- Drag and Drop file(s) here from Finder

Gradle build finished in 10s 50ms

Indexing...

Context: <not available>

File Structure



```
app
  manifests
    AndroidManifest.xml
  java
    com.mydemos.yuanlong.hello
      MainActivity
    com.mydemos.yuanlong.hello (android
      ApplicationTest
  res
    drawable
    layout
      activity_main.xml
    mipmap
      ic_launcher.png (5)
        ic_launcher.png (hdpi)
        ic_launcher.png (mdpi)
        ic_launcher.png (xhdpi)
        ic_launcher.png (xxhdpi)
        ic_launcher.png (xxxhdpi)
    values
      dimens.xml (2)
      strings.xml
      styles.xml
Gradle Scripts
  build.gradle (Project: Hello)
  build.gradle (Module: app)
  gradle-wrapper.properties (Gradle Versio
  proguard-rules.pro (ProGuard Rules for a
  gradle.properties (Project Properties)
  settings.gradle (Project Settings)
  local.properties (SDK Location)
```

```java
package com.mydemos.yuanlong.hello;

import ...

public class MainActivity extends ActionBarActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

File Structure

- Important Files
  - activity_main.xml

    Default file for XML layout file for the activity. Provides both text view and preview of the screen UI.
  - MainActivity.java

    Java code for activity class that gets executed when application is run.
  - AndroidManifest.xml

    Presents essential information about your app to the Android system like components, java package, permissions.
  - Strings.xml

    Contains all the text that your application uses.

File structure

- java

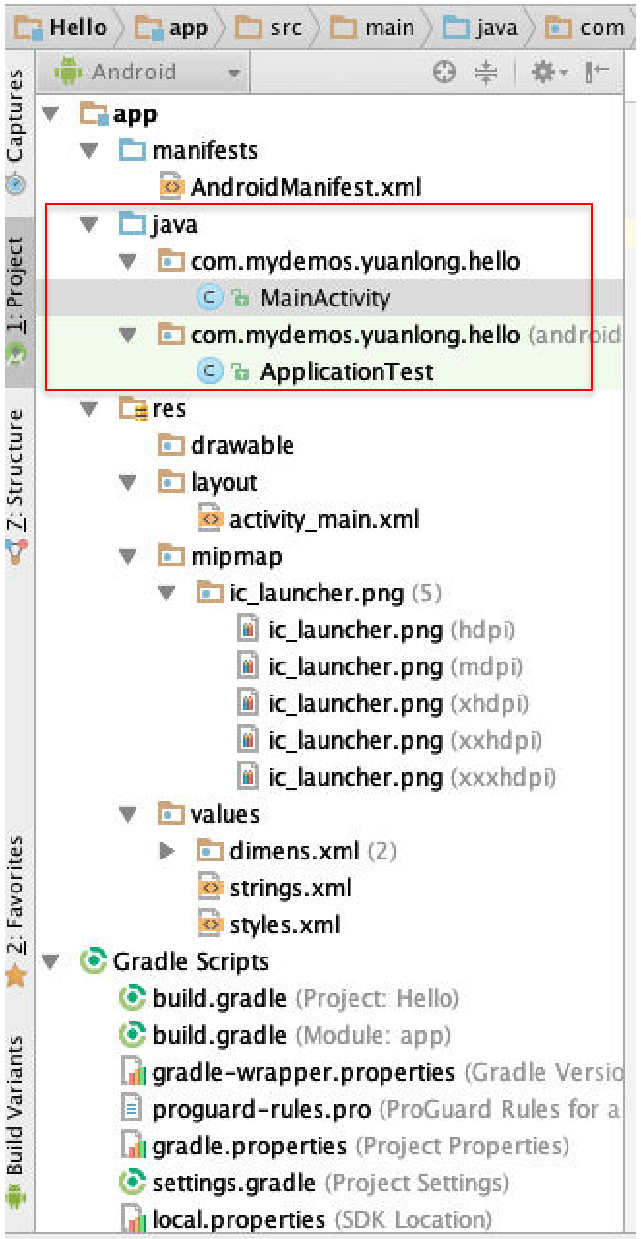
Contains the java class and application logic.

  - MainActivity.java

  Java code for activity class that gets executed when application is run.

  - ApplicationTest.java

  Java code for testing your App.

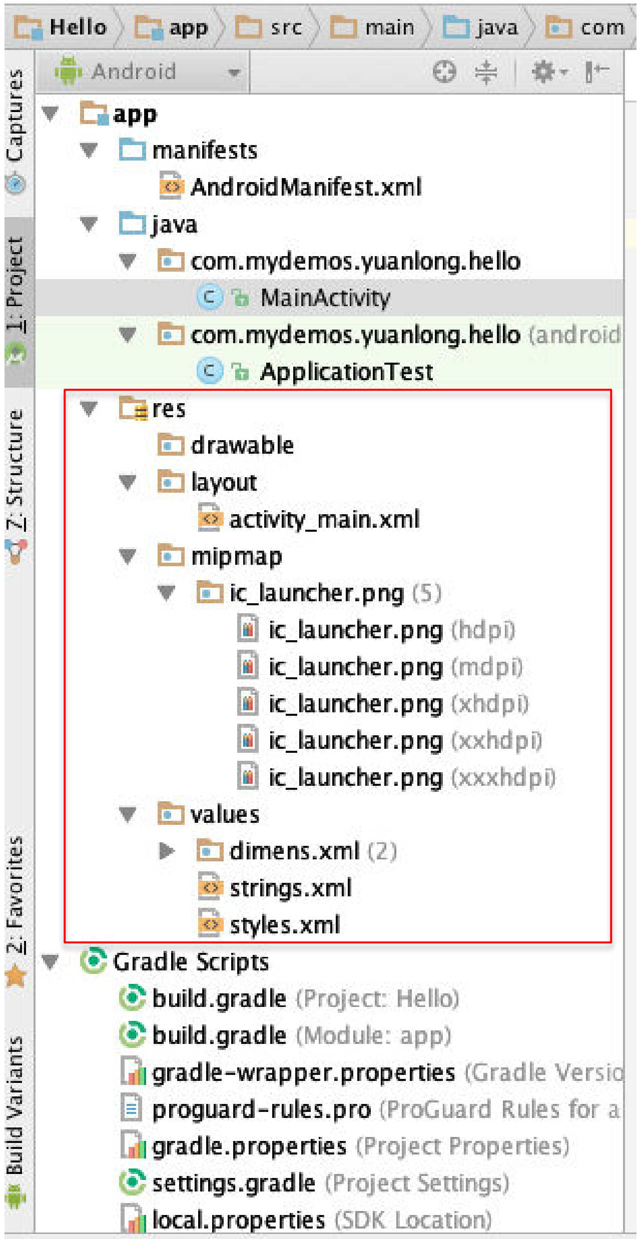


```
Hello > app > src > main > java > com
Android
▼ app
  ▼ manifests
      AndroidManifest.xml
  ▼ java
    ▼ com.mydemos.yuanlong.hello
        MainActivity
    ▼ com.mydemos.yuanlong.hello (android
        ApplicationTest
  ▼ res
      drawable
    ▼ layout
        activity_main.xml
    ▼ mipmap
      ▼ ic_launcher.png (5)
          ic_launcher.png (hdpi)
          ic_launcher.png (mdpi)
          ic_launcher.png (xhdpi)
          ic_launcher.png (xxhdpi)
          ic_launcher.png (xxxhdpi)
    ▼ values
      ▶ dimens.xml (2)
        strings.xml
        styles.xml
▼ Gradle Scripts
    build.gradle (Project: Hello)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Versio
    proguard-rules.pro (ProGuard Rules for a
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

File structure

- res

Contains resources, e.g. images(drawable folder), layout, icons (mipmap folder) strings(values folder).



```
app
  manifests
    AndroidManifest.xml
  java
    com.mydemos.yuanlong.hello
      MainActivity
    com.mydemos.yuanlong.hello (android
      ApplicationTest
  res
    drawable
    layout
      activity_main.xml
    mipmap
      ic_launcher.png (5)
        ic_launcher.png (hdpi)
        ic_launcher.png (mdpi)
        ic_launcher.png (xhdpi)
        ic_launcher.png (xxhdpi)
        ic_launcher.png (xxxhdpi)
    values
      dimens.xml (2)
      strings.xml
      styles.xml
Gradle Scripts
  build.gradle (Project: Hello)
  build.gradle (Module: app)
  gradle-wrapper.properties (Gradle Versio
  proguard-rules.pro (ProGuard Rules for a
  gradle.properties (Project Properties)
  settings.gradle (Project Settings)
  local.properties (SDK Location)
```

Hello World

- Make sure “Hello World” is in a TextView
  - Open activity_main.java
  -
```
<TextView
        …
    android:text="Hello World!"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    …
                              />
```

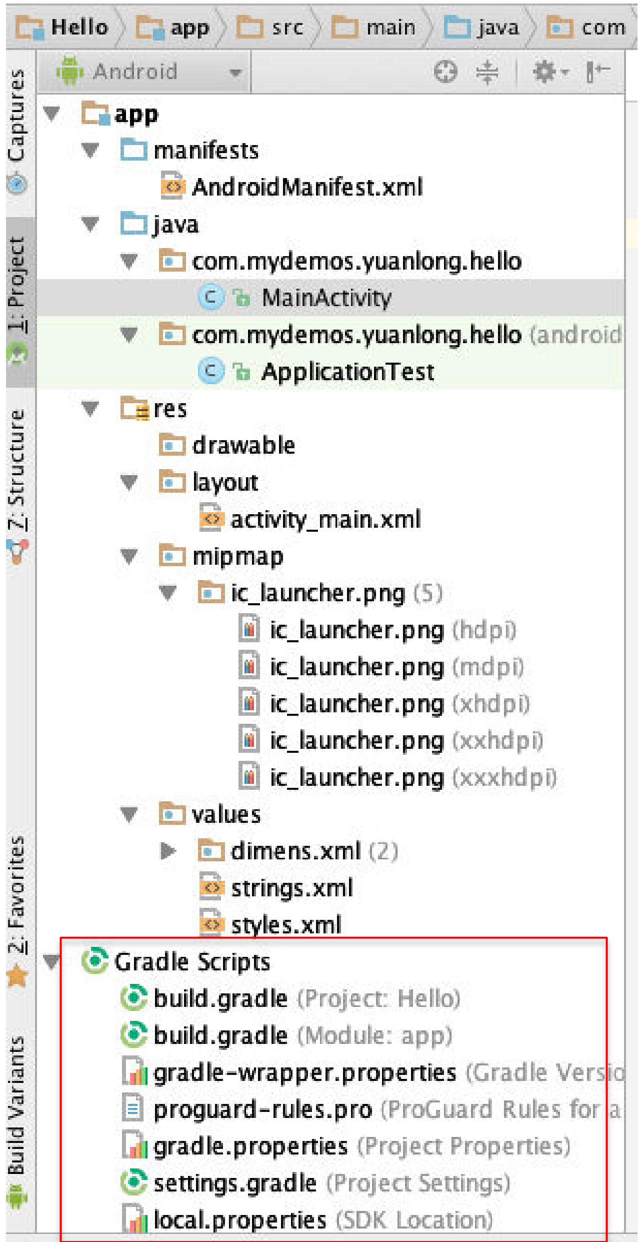
File structure

- Gradle

An advanced build toolkit for android.

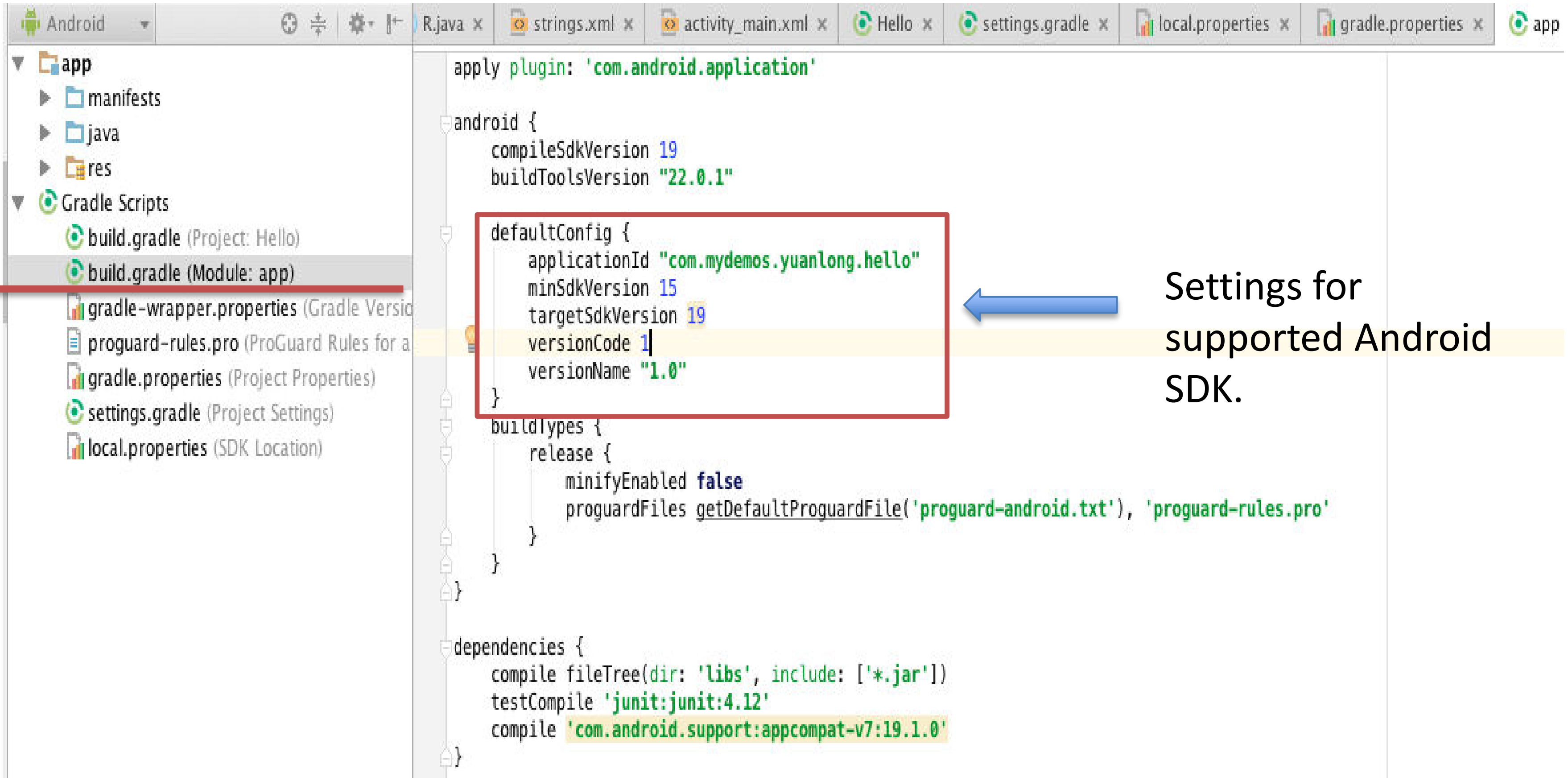
  - build.gradle

  Plain text files to configure the build.



```
Hello > app > src > main > java > com
Android
▼ app
    ▼ manifests
        AndroidManifest.xml
    ▼ java
        ▼ com.mydemos.yuanlong.hello
            MainActivity
        ▼ com.mydemos.yuanlong.hello (android
            ApplicationTest
    ▼ res
        drawable
        ▼ layout
            activity_main.xml
        ▼ mipmap
            ▼ ic_launcher.png (5)
                ic_launcher.png (hdpi)
                ic_launcher.png (mdpi)
                ic_launcher.png (xhdpi)
                ic_launcher.png (xxhdpi)
                ic_launcher.png (xxxhdpi)
        ▼ values
            ▶ dimens.xml (2)
            strings.xml
            styles.xml
▼ Gradle Scripts
    build.gradle (Project: Hello)
    build.gradle (Module: app)
    gradle-wrapper.properties (Gradle Versio
    proguard-rules.pro (ProGuard Rules for a
    gradle.properties (Project Properties)
    settings.gradle (Project Settings)
    local.properties (SDK Location)
```

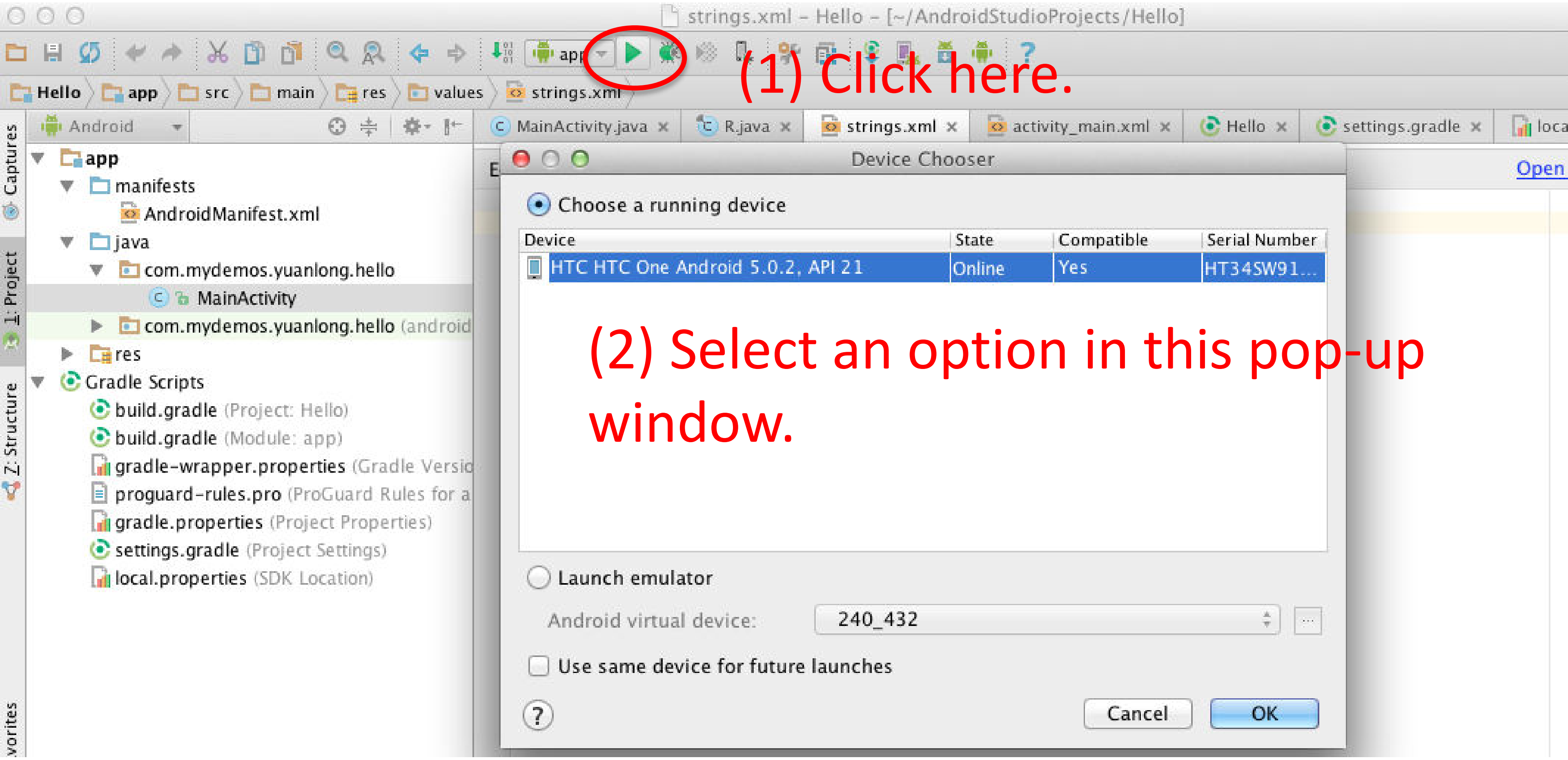
File Structure


Android
R.java
strings.xml
activity_main.xml
Hello
settings.gradle
local.properties
gradle.properties
app
app
manifests
java
res
Gradle Scripts
build.gradle (Project: Hello)
build.gradle (Module: app)
gradle-wrapper.properties (Gradle Versio
proguard-rules.pro (ProGuard Rules for a
gradle.properties (Project Properties)
settings.gradle (Project Settings)
local.properties (SDK Location)

```
apply plugin: 'com.android.application'

android {
    compileSdkVersion 19
    buildToolsVersion "22.0.1"

    defaultConfig {
        applicationId "com.mydemos.yuanlong.hello"
        minSdkVersion 15
        targetSdkVersion 19
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    testCompile 'junit:junit:4.12'
    compile 'com.android.support:appcompat-v7:19.1.0'
}
```

Settings for supported Android SDK.

Test App


(1) Click here.

(2) Select an option in this pop-up window.

strings.xml – Hello – [~/AndroidStudioProjects/Hello]

Hello > app > src > main > res > values > strings.xml

MainActivity.java | R.java | strings.xml | activity_main.xml | Hello | settings.gradle

- app
  - manifests
    - AndroidManifest.xml
  - java
    - com.mydemos.yuanlong.hello
      - MainActivity
    - com.mydemos.yuanlong.hello (android
  - res
- Gradle Scripts
  - build.gradle (Project: Hello)
  - build.gradle (Module: app)
  - gradle-wrapper.properties (Gradle Versio
  - proguard-rules.pro (ProGuard Rules for a
  - gradle.properties (Project Properties)
  - settings.gradle (Project Settings)
  - local.properties (SDK Location)

Device Chooser

● Choose a running device

Device	State	Compatible	Serial Number
HTC HTC One Android 5.0.2, API 21	Online	Yes	HT34SW91...

○ Launch emulator

Android virtual device: 240_432

☐ Use same device for future launches

Test App

- Option 1: Test your App in an Android device.
- Option 2: Test your App in an Emulator.

Test App

- Option 1: Test your App in an Android device.

  Step 1: In your Android device,

  – Enable USB debugging mode. (Note: the steps may depend on what device you are using)
    - E.g. Go to Settings -> About -> Software information -> More . And then tap “Build number” 7 times to become developer.

  – Connect your Android device to computer via USB.

  – Allow USB debugging in your Android Device.

Test App

- Option 1: Test your App in an Android device.

  Step 2: Go back to “Device Chooser” in Android Studio.

  – Make sure that the state of device now is “online”.

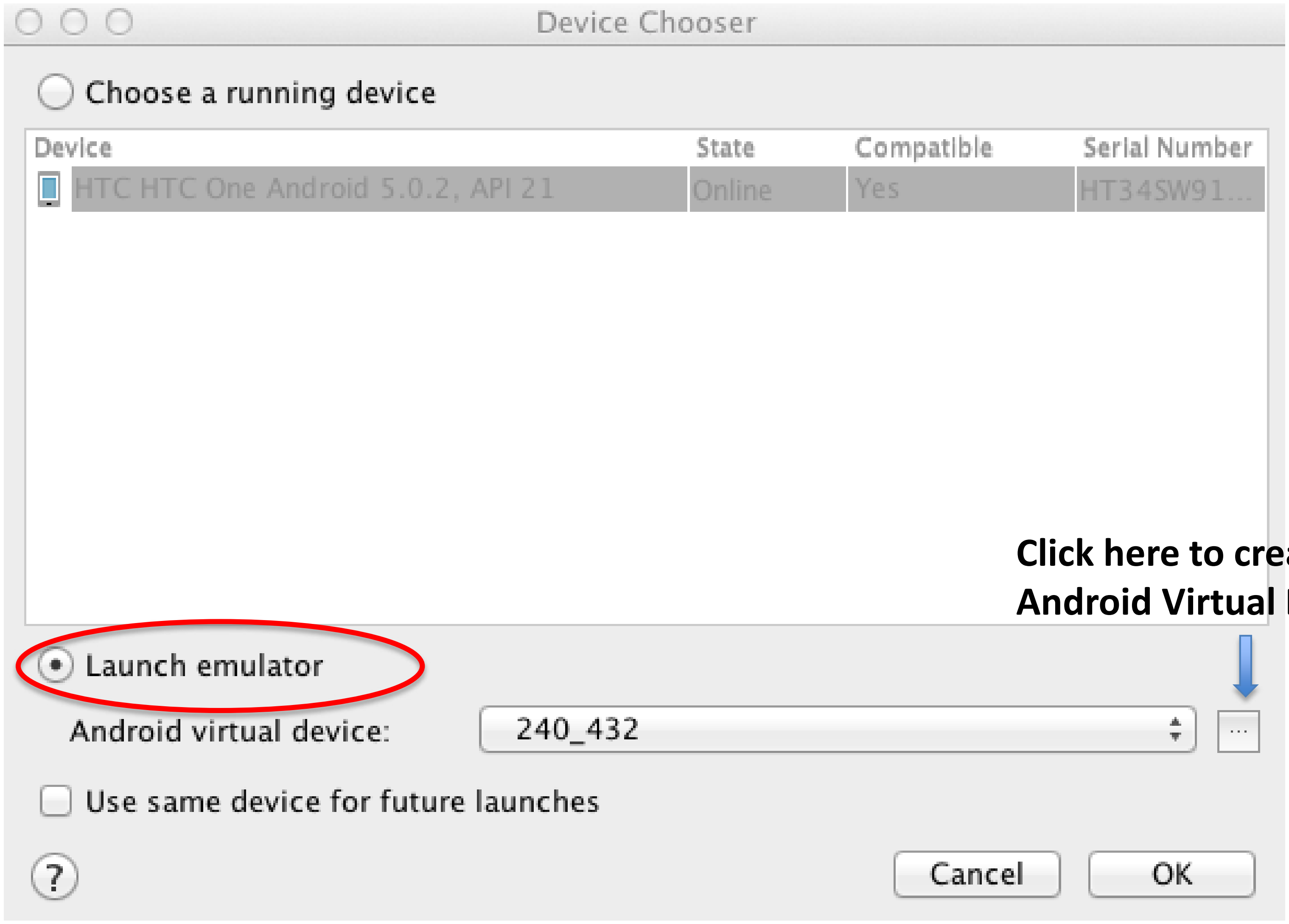
  – Choose a running device.

  – Click “Ok”.

  Step 3: Find your Hello App in your Android Studio.

Test App

- Option 2: Test your App in an Emulator.
  - Select “Launch emulator” in “Device Chooser”.
  - Create an Android Virtual Device(AVD) in virtual device manager.
  - Select an AVD and install your App in the emulator.



Device Chooser

Choose a running device

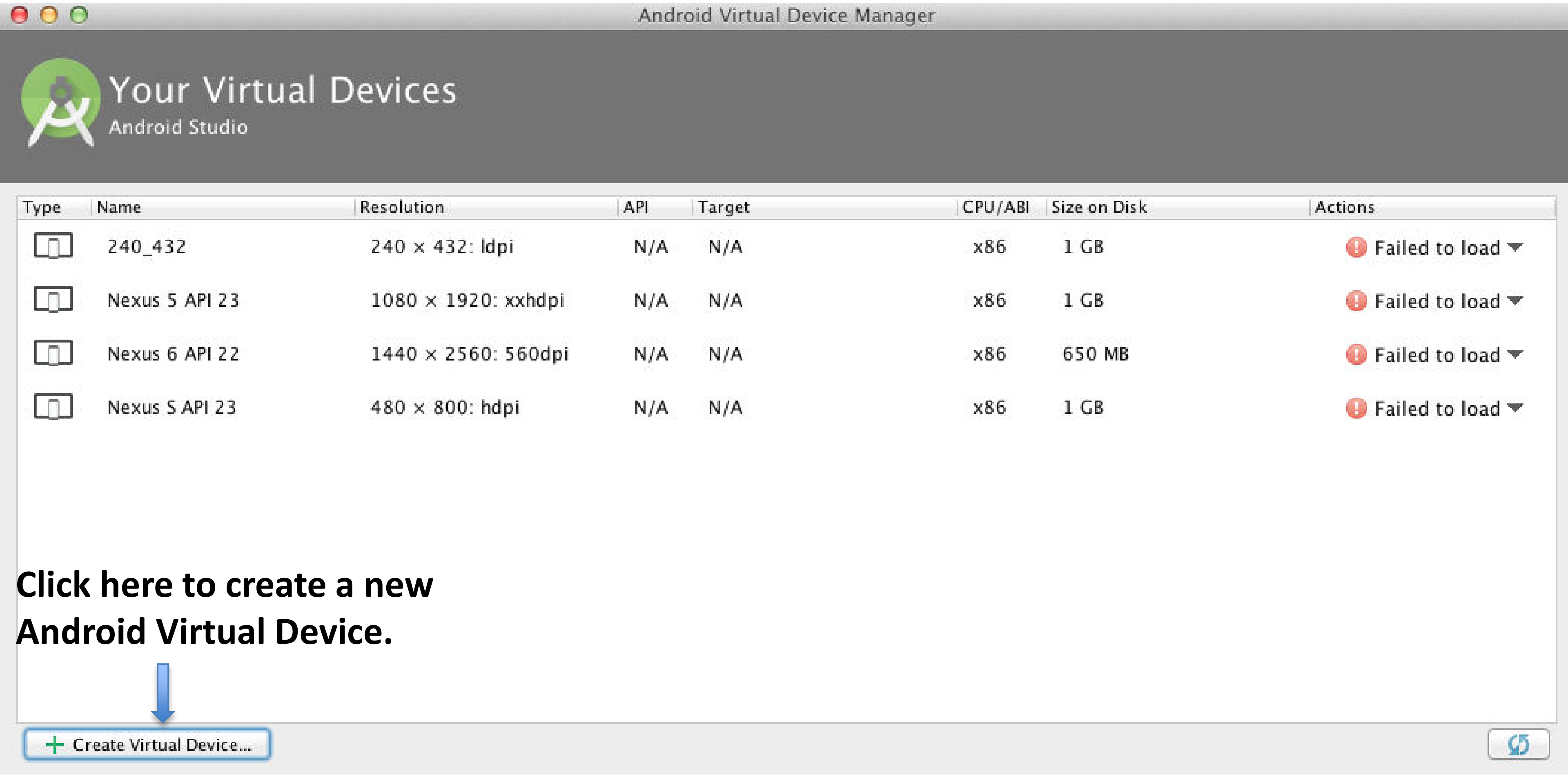
Device	State	Compatible	Serial Number
HTC HTC One Android 5.0.2, API 21	Online	Yes	HT34SW91...

(●) Launch emulator

Android virtual device: 240_432

☐ Use same device for future launches

Click here to create a new Android Virtual Device.



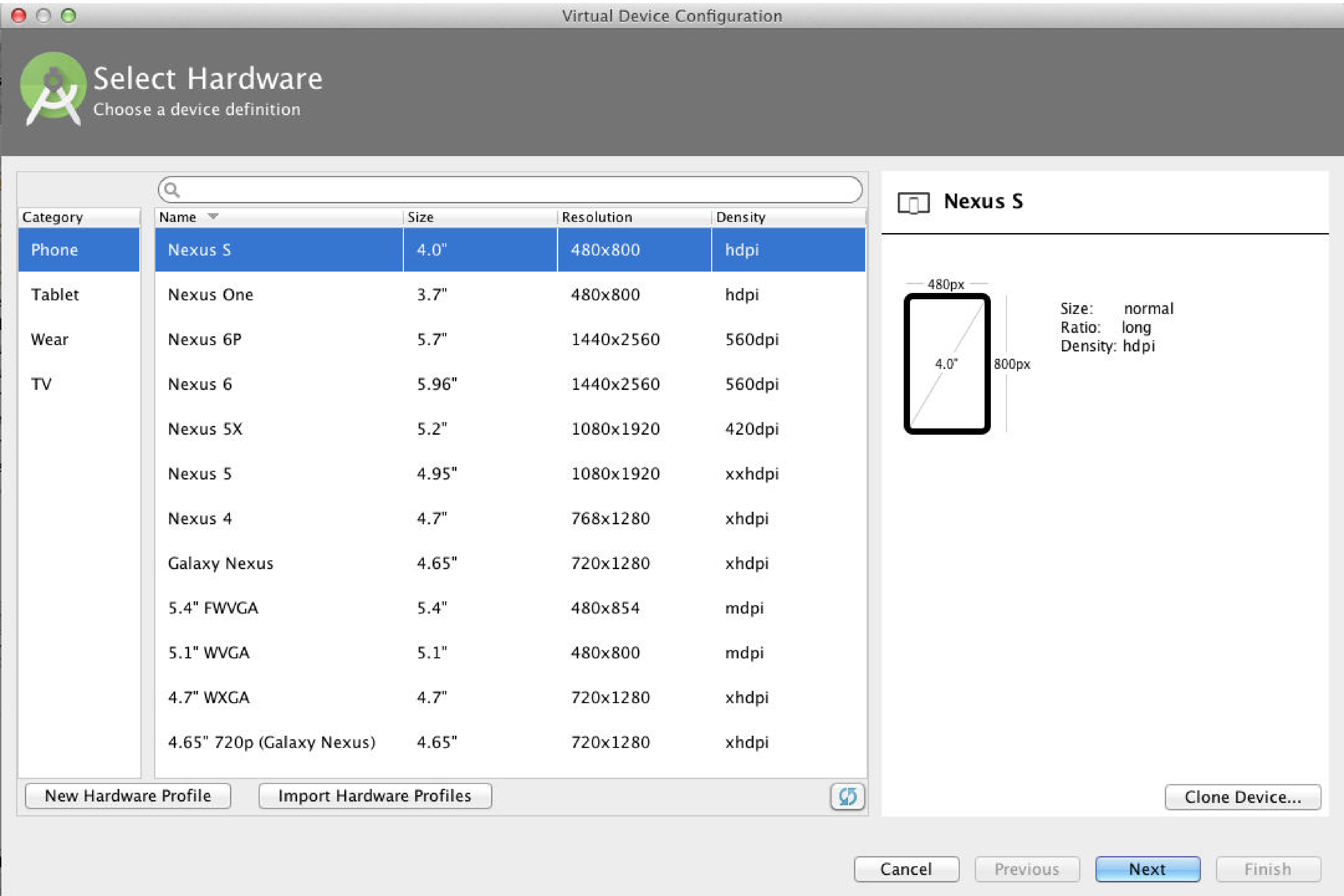
Android Virtual Device Manager

Your Virtual Devices

Android Studio

Type	Name	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	240_432	240 × 432: ldpi	N/A	N/A	x86	1 GB	Failed to load
	Nexus 5 API 23	1080 × 1920: xxhdpi	N/A	N/A	x86	1 GB	Failed to load
	Nexus 6 API 22	1440 × 2560: 560dpi	N/A	N/A	x86	650 MB	Failed to load
	Nexus S API 23	480 × 800: hdpi	N/A	N/A	x86	1 GB	Failed to load

Click here to create a new Android Virtual Device.



Virtual Device Configuration

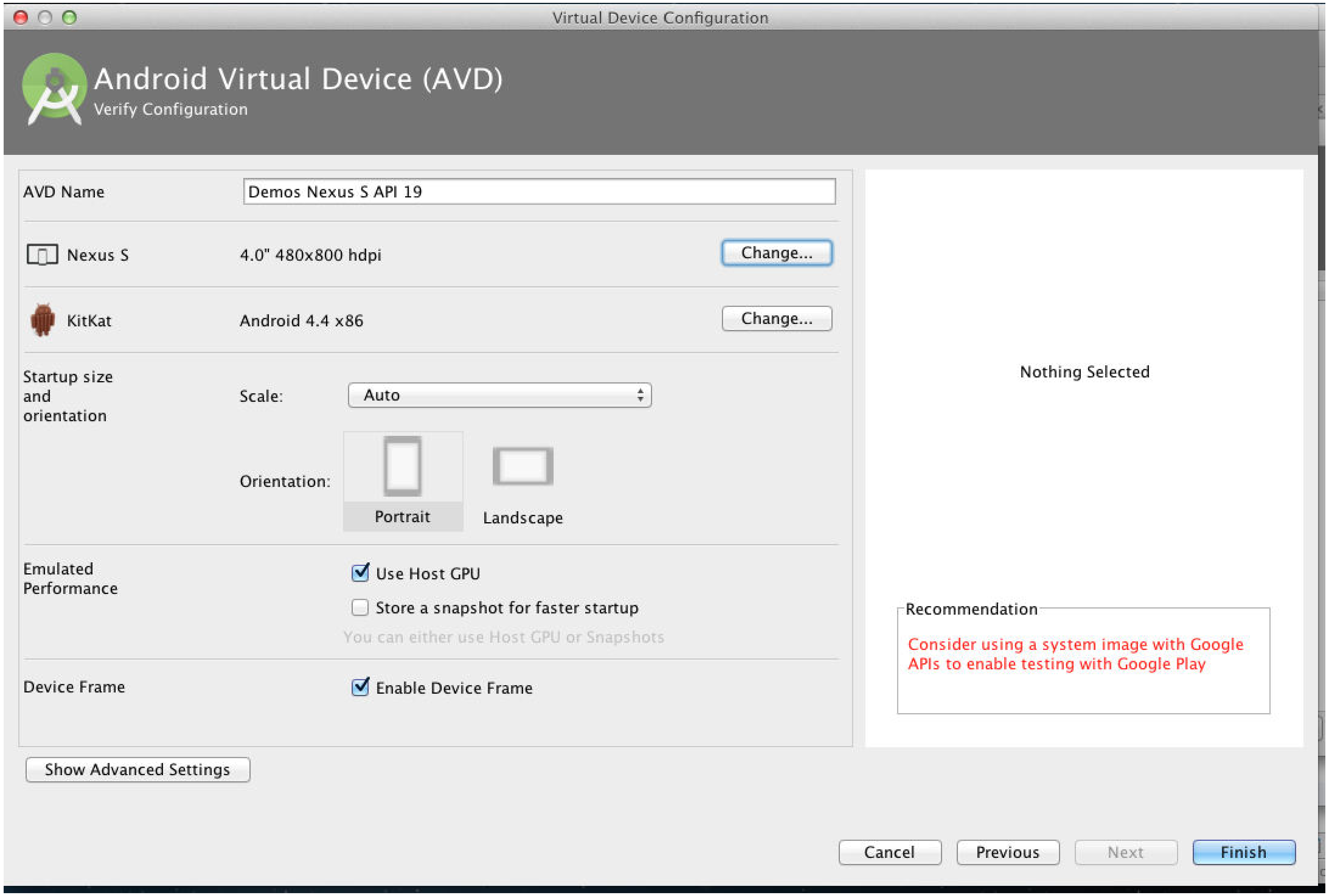
Select Hardware

Choose a device definition

Category	Name	Size	Resolution	Density
Phone	Nexus S	4.0"	480x800	hdpi
Tablet	Nexus One	3.7"	480x800	hdpi
Wear	Nexus 6P	5.7"	1440x2560	560dpi
TV	Nexus 6	5.96"	1440x2560	560dpi
	Nexus 5X	5.2"	1080x1920	420dpi
	Nexus 5	4.95"	1080x1920	xxhdpi
	Nexus 4	4.7"	768x1280	xhdpi
	Galaxy Nexus	4.65"	720x1280	xhdpi
	5.4" FWVGA	5.4"	480x854	mdpi
	5.1" WVGA	5.1"	480x800	mdpi
	4.7" WXGA	4.7"	720x1280	xhdpi
	4.65" 720p (Galaxy Nexus)	4.65"	720x1280	xhdpi

Nexus S

Size: normal
Ratio: long
Density: hdpi



Virtual Device Configuration

Android Virtual Device (AVD)

Verify Configuration

AVD Name: Demos Nexus S API 19

Nexus S — 4.0" 480x800 hdpi — Change...

KitKat — Android 4.4 x86 — Change...

Startup size and orientation — Scale: Auto

Orientation: [✓ Portrait] Landscape

Emulated Performance
- [✓] Use Host GPU
- [ ] Store a snapshot for faster startup

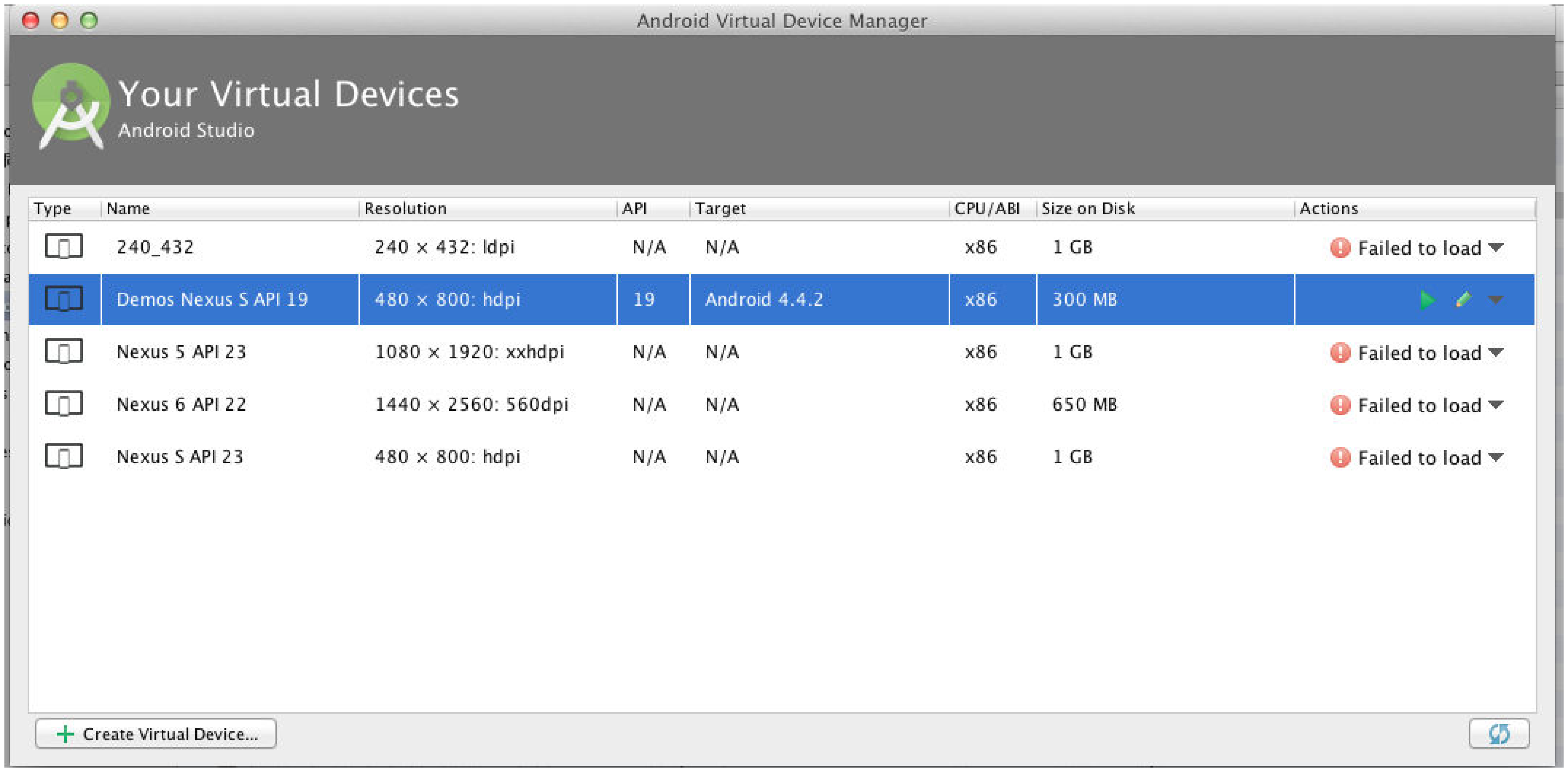
You can either use Host GPU or Snapshots

Device Frame
- [✓] Enable Device Frame

Nothing Selected

Recommendation

Consider using a system image with Google APIs to enable testing with Google Play



Android Virtual Device Manager

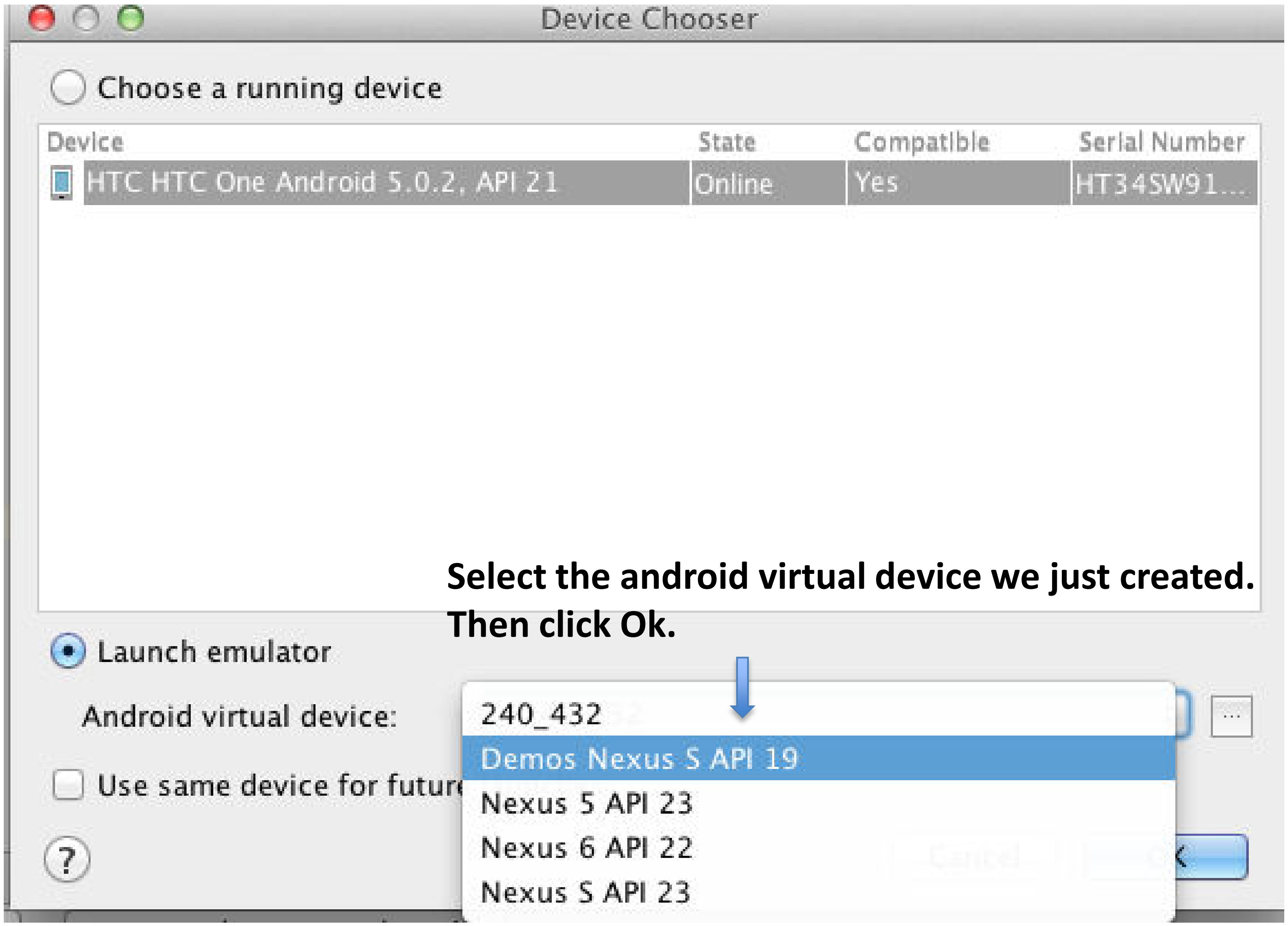
Your Virtual Devices

Android Studio

Type	Name	Resolution	API	Target	CPU/ABI	Size on Disk	Actions
	240_432	240 × 432: ldpi	N/A	N/A	x86	1 GB	Failed to load
	Demos Nexus S API 19	480 × 800: hdpi	19	Android 4.4.2	x86	300 MB	
	Nexus 5 API 23	1080 × 1920: xxhdpi	N/A	N/A	x86	1 GB	Failed to load
	Nexus 6 API 22	1440 × 2560: 560dpi	N/A	N/A	x86	650 MB	Failed to load
	Nexus S API 23	480 × 800: hdpi	N/A	N/A	x86	1 GB	Failed to load

Once the configuration is finished, a new Android Virtual Device named "Demos Nexus S API 19 " will be created in the list. After that, please close this window.

Note: if it failed to load, please restart your Android Studio.


Device Chooser

Choose a running device

Device	State	Compatible	Serial Number
HTC HTC One Android 5.0.2, API 21	Online	Yes	HT34SW91...

● Launch emulator

Android virtual device: 240_432

Demos Nexus S API 19

Nexus 5 API 23

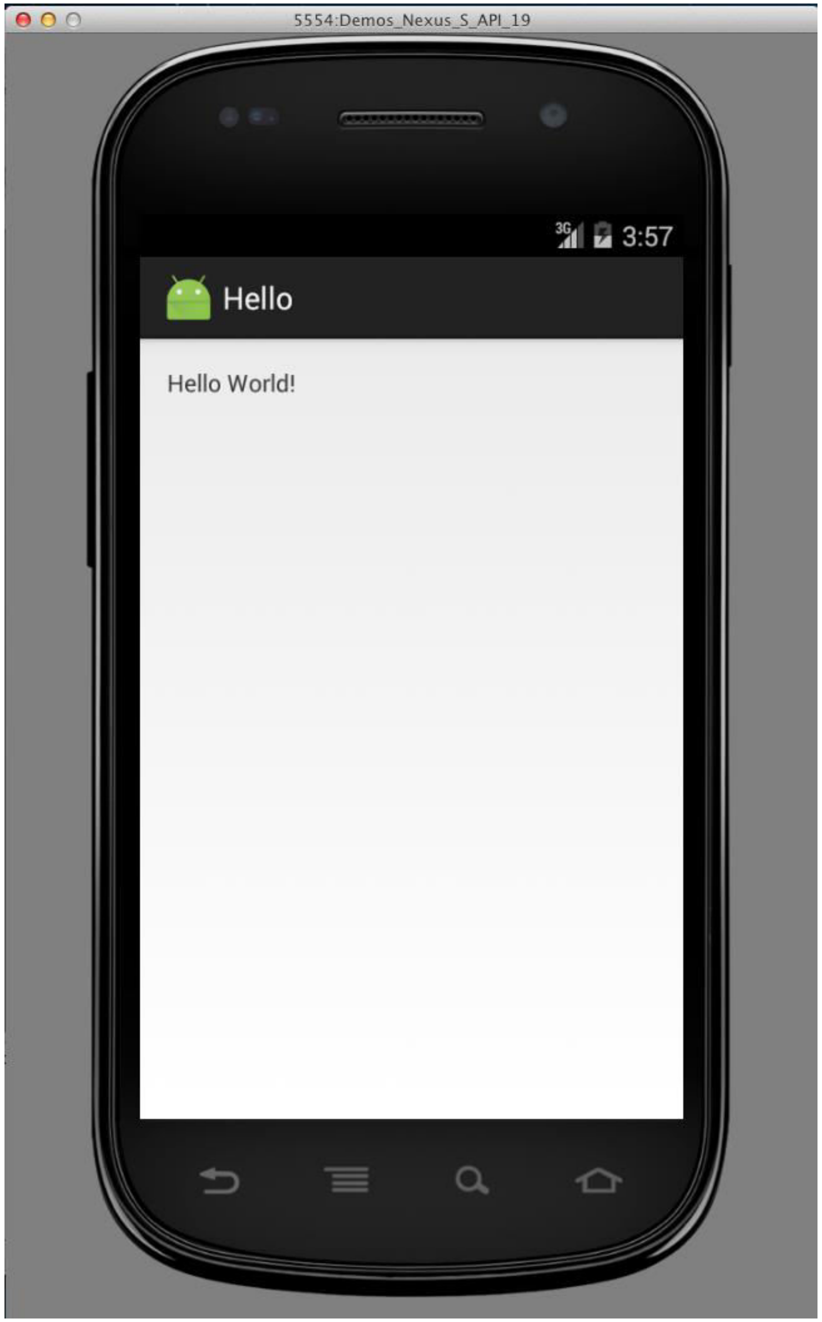
Nexus 6 API 22

Nexus S API 23

Use same device for futur

Select the android virtual device we just created. Then click Ok.

An emulator will be generated. Android App "Hello" will run on this emulator and print out string "Hello World!"



Hello World!

Try More Features

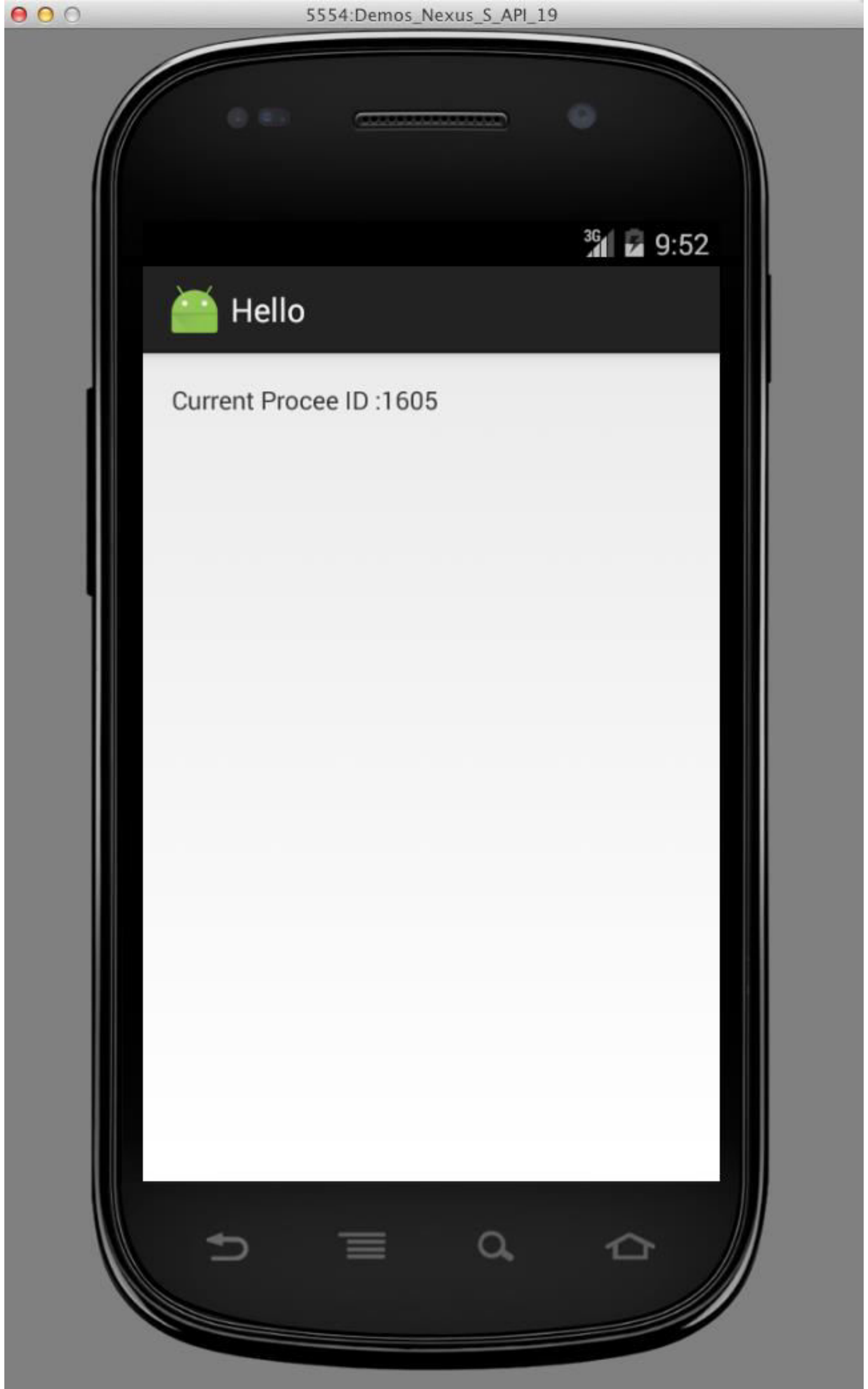
- Display current process ID.
  - Open activity_main.java
  - Add an id for the component TextView.

```
<TextView
        android:id="@+id/textViewHello"
        android:text="Hello World!"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```

  - Open MainActivity.java
    - Add statement “ **import** android.widget.TextView; “
    - Type the following statements at the end of onCreate().

```
        int id= android.os.Process.myPid();
TextView text = (TextView) findViewById(R.id.textViewHello);
text.setText("Current Procee ID :"+Integer.toString(id));
```

Notę: R is a class automatically generated for resources.



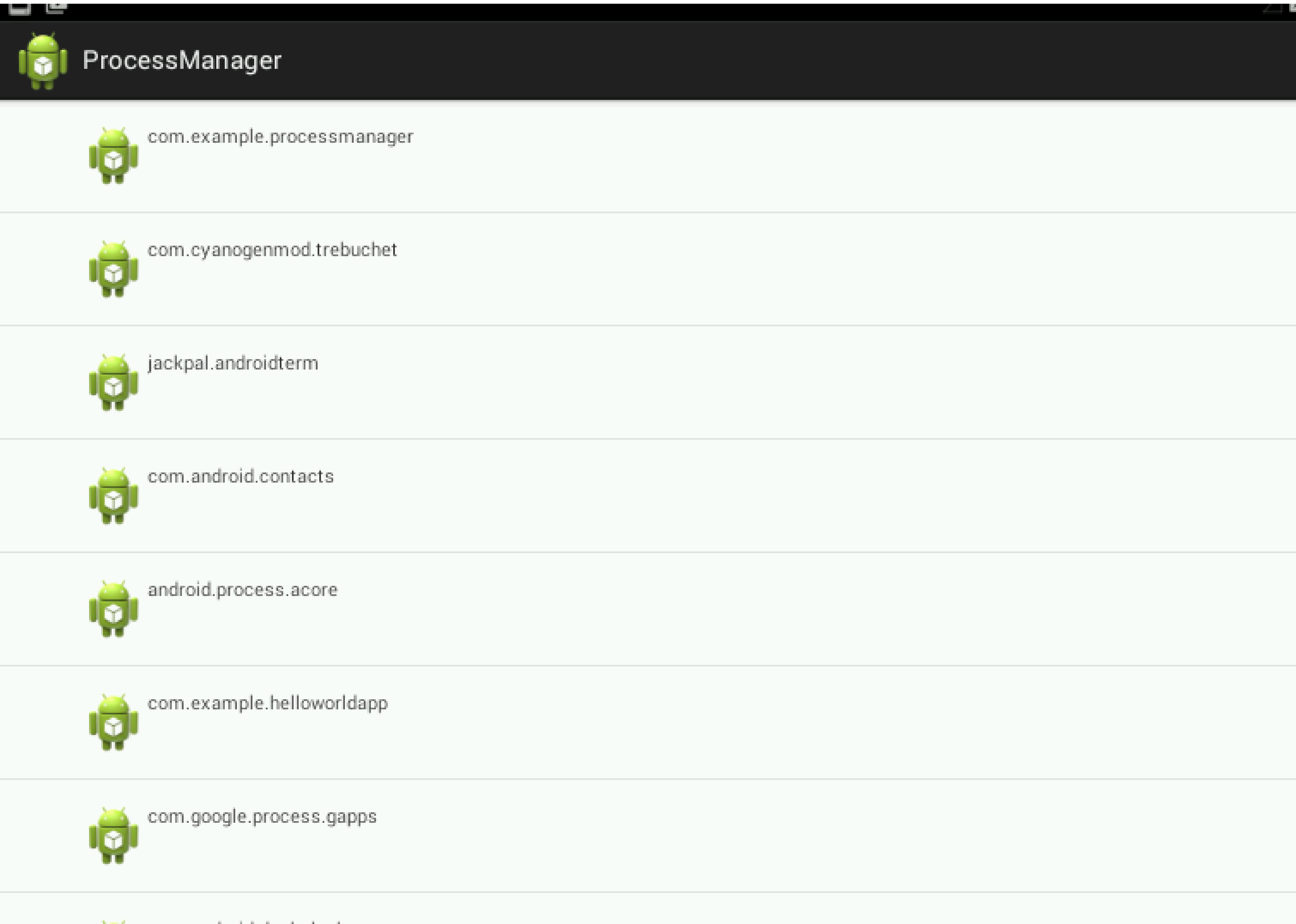
```
Current Procee ID :1605
```

DEMO 2: A SIMPLE PROCESS MANAGER APP.

Demo 2

- A simple Process Manager
  - List the processes in Android System.
  - Displaying the traffic statistics.

See more at http://www.itcuties.com/android/how-to-get-running-process-list-and-traffic-statistics/#sthash.HAPRV4By.dpuf



ProcessManager

com.example.processmanager

com.cyanogenmod.trebuchet

jackpal.androidterm

com.android.contacts

android.process.acore

com.example.helloworldapp

com.google.process.gapps

Demo 2

- Layout design (res/layout/activity_main.xml)
  - Linear layout
    - Image
    - Text

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/linearLayout1"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:layout_alignParentTop="true"
        android:orientation="horizontal" >
      <ImageView
        android:id="@+id/detailsIco"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:src="@drawable/ic_launcher" />
      <TextView
          android:id="@+id/appNameText"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:text="APP NAME GOES HERE" />

 </LinearLayout>
```

Note: You need to copy ic_launcher.png from folder "mipmap" to folder "drawable".

Demo 2

- Building a list for processes
  - Create a file ListAdapter.java under folder “java”.
  - Import following packages.

```
import java.util.List;

import android.app.ActivityManager.RunningAppProcessInfo;
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
```

Demo 2

- Building a list for processes(continues)
  - Let ListAdapter extends ArrayAdapter<RunningAppProcessInfo>
  - Create a constructor for ListApdapter.

```
public class ListAdapter extends ArrayAdapter<RunningAppProcessInfo> {
    // List context
    private final Context context;
    // List values
    private final List<RunningAppProcessInfo> values;

    public ListAdapter(Context context, List<RunningAppProcessInfo> values) {
        super(context, R.layout.activity_main, values);
        this.context = context;
        this.values = values;
    }
}
```

Demo 2

- Building a list for processes(continues)
  - Override method getView()

```
@Override
public View getView(int position, View convertView, ViewGroup parent) {

      LayoutInflater inflater = (LayoutInflater)
context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);

      View rowView = inflater.inflate(R.layout.activity_main, parent, false);

      TextView appName = (TextView) rowView.findViewById(R.id.appNameText);
      appName.setText(values.get(position).processName);

      return rowView;
 }
```

Demo 2

- Display process list and show traffic statistics
  - Modify MainActivity.java
    - Extends class ListActivity
    - Modify OnCreate() so that once app is open all the processes are listed.
    - Override OnListItemClick() so that once an item in the list is clicked the traffic statistics will be displayed.

Demo 2

- Display process list and show traffic statistics
  - Import Packages

```
import java.util.List;

import android.app.ActivityManager;
import android.app.ActivityManager.RunningAppProcessInfo;
import android.app.ListActivity;
import android.net.TrafficStats;
import android.os.Bundle;
import android.view.View;
import android.widget.ListView;
import android.widget.Toast;
```

Demo 2

- Display process list and show traffic statistics
  - Definition of MainActivity.java

```
public class MainActivity extends ListActivity {



}
```

Demo 2

- Display process list and show traffic statistics
  - Override OnCreate()

```
@Override
  public void onCreate(Bundle savedInstanceState) {
     super.onCreate(savedInstanceState);

     // Get running processes
     ActivityManager manager = (ActivityManager) getSystemService(ACTIVITY_SERVICE);
     List<RunningAppProcessInfo> runningProcesses = manager.getRunningAppProcesses();
     if (runningProcesses != null && runningProcesses.size() > 0) {
        // Set data to the list adapter
        setListAdapter(new ListAdapter(this, runningProcesses));
     } else {
        // In case there are no processes running (not a chance :))
        Toast.makeText(getApplicationContext(), "No application is running", Toast.LENGTH_LONG).show();
     }

  }
```

Demo 2

- Display process list and show traffic statistics
  - Override OnListItemClick()

```
@Override
   protected void onListItemClick(ListView l, View v, int position, long id) {
      long send       = 0;
      long recived    = 0;
      // Get UID of the selected process
      int uid = ((RunningAppProcessInfo)getListAdapter().getItem(position)).uid;

      // Get traffic data
      recived = TrafficStats.getUidRxBytes(uid);
      send = TrafficStats.getUidTxBytes(uid);

      // Display data
      Toast.makeText(getApplicationContext(), "UID " + uid + " details...\n send: " + send/1000 + "kB" + " \n
recived: " + recived/1000 + "kB", Toast.LENGTH_LONG).show();
   }
```

Demo 2

- Test your App in your Android Device or Emulator.

Useful Resources

- 1. **Android tutorial for beginners.**
- https://www.raywenderlich.com/78574/android-tutorial-for-beginners-part-1
- 2.**Android APIs.**
- https://developer.android.com/reference/classes.html
- 3. **How to enable USB debugging mode on Android.**
- https://www.kingoapp.com/root-tutorials/how-to-enable-usb-debugging-mode-on-android.htm
- 4. **Android Developer Guide.**
- https://developer.android.com/guide/index.html
- 5.**“Android Tablet Application Development For Dummies”** By Gerhard Franken