1 Introduction to Android, Android application design

Topics Covered

- Open Handset Alliance
- The Android Platform
- Android SDK
- Building a Simple Android Application
- Anatomy of an Android applications
- Android terminologies
- Application Context, Activities, Services, Intents
- Receiving and Broadcasting Intents
- Android Manifest File and its common settings
- Using Intent Filter, Permissions
- Managing Application resources in a hierarchy
- Working with different types of resources

Introduction

- Now a days every one use mobile phones. Stylish phones have many features like GPS, accelerometers and touch screens are enticing platform upon which to create innovative mobile applications.
- It will be designed to attract customers.
- Android developers are free to write applications that take full advantage of increasingly powerful mobile hardware.
- Android has opened many ways for mobile phone development for developers to develop apps for mobile phones.
- Experienced developers can expand their time to develop application into the android platform and get many advantages.
- They create innovative products with unique features.

What is android?

- Android is an operating system based on the Linux kernel.
- It is mainly designed for touch screen mobile devices such as smart phones and tablet computers.
- It is developed by Google.
- In other words we can also say that Android is a Linux-based operating system for smart phones and tablets.
- It includes touch screen user interface, widgets, camera, network and all other features that enable a cell phone to be called a smart phone.
- It is a one type of platform that supports various applications, available through the Android play store.
- The android platform allows end users to develop, install and use their own application on the top of the Android framework.

Versions of android

- Alpha:
  - Android 1.0
- Beta:
  - Android 1.1
- Cupcake:
  - Android 1.5
- Donut:
  - Android 1.6
- Éclair:
  - Android 2.0
  - Android 2.1
- Froyo: (short for "frozen yogurt")
  - Android 2.2
- Gingerbread:
  - Android 2.3
- Honeycomb:
  - Android 3.0
  - Android 3.1
  - Android 3.2
- Ice Cream Sandwich:
  - Android 4.0
- Jelly Bean:
  - Android 4.1
  - Android 4.2
  - Android 4.3
- Kit Kat:
  - Android 4.4 (confirmed)
- Lollipop:
  - Android 5.0 (confirmed)
- Marshmallow:
  - Android 6.0 (confirmed)
- Nogut:
  - Android 7.0 (confirmed)
- Oreo:
  - Android 8.0 (confirmed)
- Pie:
  - Android 9.0 (confirmed)
- Android 10:
  - Android 10.0 (confirmed)
- Android 11:
  - Android 11.0 (confirmed)

The Open Handset Alliance

- The open handset alliance is a group of mobile and technology leaders who share this vision for changing the mobile experience for customers.
- The OHA was establish on 6 th November 2007 led by Google with 47 members including mobile handset makers, application developers, some mobile carriers and chip makers.
- The overall cost of handsets will be lower and mobile operators will have complete flexibility to customize and differentiate their product lines. Innovation in handsets and in services.
- There are many companies are in OHA like all parts of the mobile ecosystem are available in the Alliance. Members include mobile operators, handset manufacturers, semiconductor companies, software companies, and commercialization companies.

Open Handset Alliance Members

- Operators
  - T-Mobile, Telecom Italia, Telephonic, NTT Do Como, KDDI, Sprint, China Mobile
- Handset Manufacturers
  - HTC, LG, Motorola, Samsung
- Semiconductor Companies
  - Audience, Broadcom, Intel, Marvell, NVIDIA, Qualcomm, SiRF, Synaptic, Texas Instruments
- Software Companies [Applications]
  - Ascender, eBay, Esmertec, Google, Living Image, NMS Communications, Packet Video, Sky Pop, SONi VOX
- Commercialization Companies
  - Aplix, TAT, Wind River, Noser

The Android Platform

- The android platform was launched in 2007 by the OHA.
- An alliance is a group of famous companies that includes Google, HTC, Motorola and others.
- Most of the applications that run on the Android Platform are written in Java, but there is no Java Virtual Machine.
- Java classes are first compiled into the Dalvik Executable and run on the Dalvik Virtual Machine.
- To create an application for the platform, a developer requires the Android SDK, which includes tools and APIs.
- Following are the different type of platform.

Android 1.0(API level 1)		
Version	**Release Date**	**Features**
1.0	23 september 2008	• Android market application download, web browser to show, zoom and HTML support • Camera support • Folders allowing the group of applications.

		• Different types of google search. • Wi-fi and Bluetooth support • Other applications are included like alarm clock, calculator.

Android 1.1(API level 2)		
Version	**Release Date**	**Features**
1.1	9 february 2009	• It gives the details when user searches for business. • It has the ability to save attachments in messages. • It support marquee.

Android 1.5 Cupcake(API level 3)		
Version	**Release Date**	**Features**
1.5	30 April 2009	• It support virtual keyboards with text prediction. • It support video recording and playback. • Copy and paste features in web browser. • Animated screen transition, auto rotation option. • Ability to upload videos to you tube.

Android 1.6 Donut(API level 4)		
Version	**Release Date**	**Features**
1.6	15 September 2009	• Voice and text entry search • Easier searching • Gallery, camera are fully integrated. • Speed improvements in searching

Android 2.0 Eclair(API level 5)		
Version	**Release Date**	**Features**
2.0	26 october 2009	• Voice and text entry search, bookmark history, contacts and the web. • It allow any android application to “speak” a string of text. • Ability for users to select multiple photos for deletion. • Speed improvements in searching and camera applications.

Android 2.0.1 Eclair(API level 6)		
Version	**Release Date**	**Features**
2.0.1	3 december 2009	• Minor API changes, bug fixes and framework behavior changes.

Android 2.1 Eclair(API level 7)		
Version	**Release Date**	**Features**
2.1	12 January 2010	• Minor improvements to the API and bug fixes.

Android 2.2-2.2.3 Froyo(API level 8)		
Version	**Release Date**	**Features**
2.2	20 May 2010	• Speed, memory and performance optimization • USB tethering and Wi-Fi hot spot functionality. • Option to disable data access over mobile network. • Support for numeric and alphanumeric password. • Adobe flash support. • Gallery allows users to view picture stack using a zoom gesture.
2.2.1	18 january 2011	• Bug fixes, security updates and performance improvements.
2.2.2	22 january 2011	• Minor bug fixes including SMS routing issues
2.2.3	21 november 2011	• Two security patches

Android 2.3-2.3.2 Gingerbread(API level 9)		
Version	**Release Date**	**Features**
2.3	6 December 2010	• Updated user interface design with increased simplicity and speed. • Support for extra-large screen sizes and resolution. • Enhanced copy/paste functionality, allowing users to select a word by press-hold, copy and paste. • support for multiple cameras on the device, including a front facing camera.
2.3.1,2.3.2	December 2010,January 2011	• improvements and bug fixes

Android 2.3.3-2.3.7 Gingerbread(API level 10)		
Version	**Release Date**	**Features**
2.3.3	9 february 2011	• several improvements and API fixes
2.3.4	28 April 2011	• support for voice or video chat using Google Talk
2.3.5	25 July 2011	• improved network performance • fixed Bluetooth bug on Samsung Galaxy S • Improved Gmail application • Improved battery efficiency

2.3.6	2 september 2011	• Fixed a voice search bug
2.3.7	21 september 2011	• Google wallet support

Android 3.0 Honeycomb(API level 11)		
Version	**Release Date**	**Features**
3.0	22 february 2011	• Optimized tablet support with a new user interface. • Added system bar, featuring quick access to notifications, status and soft navigation buttons. • More impressive copy/paste interface. • Multiple browser tabs. • Ability to encrypt all user data.

Android 3.1 Honeycomb(API level 12)		
Version	**Release Date**	**Features**
3.1	10 May 2011	• Connectivity for USB accessories. • Resizable home screen widgets • Support for joysticks and gamepads. • High performance
Android 3.2 Honeycomb(API level 13)		
Version	**Release Date**	**Features**
3.2	15 July 2011	• Improved hardware support. • Increased ability to applications to access files on the SD card.
3.2.1	20 september 2011	• Bug fixes and minor security, stability and Wi-Fi improvements • Improved Adobe flash support in browser.
3.2.2	30 August 2011	• Bug fixes and other minor improvements
3.2.3		• Bug fixes and other minor improvements
3.2.4	December 2011	• Support for 3G and 4G tablets.

Android 4.0-4.0.2 Ice Cream Sandwich(API level 14)		
Version	**Release Date**	**Features**
4.0	19 october 2011	• Separation of widgets in a new tab, listed in a similar manner to application. • Pinch to zoom functionality in the calendar. • Improved error correction on the keyboard. • Wi-Fi direct
4.0.1	21 October 2011	• Fixed minor bugs for the Samsung galaxy.
4.0.2	28 November 2011	• Fixed minor bugs for the version Samsung.

Android 4.0.3-4.0.4 Ice Cream Sandwich(API level 15)		
Version	**Release Date**	**Features**
4.0.3	16 December 2011	• Numerous bug fixes and optimization. • Improvements to graphics, database, spell-

		checking and bluetooth functionality. • New camera application.
4.0.4	29 March 2012	• Better camera performance. • Smoother screen rotation.

Android 4.1 Jelly Bean(API level 16)		
Version	**Release Date**	**Features**
4.1	9 July 2012	• Smoother user interface. • Bi-directional text and other language support. • Bluetooth data transfer for Android Beam. • Audio chaining
4.1.1	23 July 2012	• Fixed a bug on the Nexus 7
4.1.2	9 October 2012	• Lock/home screen rotation support for Nexus. • Bug fixes and performance enhancements.

Android 4.2 Jelly Bean(API level 17)		
Version	**Release Date**	**Features**
4.2	13 November 2012	• Lock screen improvements. • Multiple user account. • Support for wireless display. • Audio chaining • Group messaging
4.2.1	27 November 2012	• Adding some event to the particular date. • Added Bluetooth gamepads.
4.2.2	11 February 2013	• New sounds for wireless charging and low battery. • Bug fixes and performance enhancements.

Android 4.3 Jelly Bean(API level 18)		
Version	**Release Date**	**Features**
4.3	24 July 2013	• Bluetooth low energy support. • Bluetooth audio/video remote control Profile.
4.3.1	3 October 2013	• Bug fixes for the Nexus.

Android 4.4 KitKat(API level 19)		
Version	**Release Date**	**Features**
4.4	31 October 2013	• Wireless printing capability. • sensor batching, step detector and counter API's
4.4.1	5 december 2013	• improvements to auto focus.
4.4.2	9 December 2013	• Furthur security

Android 5.0		
Version	**Release Date**	**Features**

5.0	25 june 2014	• The Lollipop version supported all screen sizesin both phones and tablets, TV, and Android Wear Watch. It also added direct notifications feature from lock screen so that users could view notification on lock screen and respond to the messages directly from home screen.

Android 6.0		
Version	**Release Date**	**Features**
6.0	28 may 2015	• It included features such as a new vertically scrolling app drawer along with Google Now on top,making Google Now accessible by just tapping and holding home key whether in app or website. It added fingerprint biometric unlocking of smartphone, USB Type-C support and Android Pay, etc. It also added permission feature so that users could decide what they want to share with this app.

Android 7.0		
Version	**Release Date**	**Features**
7.0	22 August 2016	• This version included multitasking enabling people to use split-screen mode that allowed use of two apps at the same screen and quick switching between apps.

Android 8.0		
Version	**Release Date**	**Features**
8.0	21 August 2017	• This version allowed short notifications based on importance and ability of snooze notification. It also included visual changes such as settings menu, along with native support for picture-in-picture mode, auto fill APIs for management to fill data and password and it replaced all blob-shaped emojis with Gradient Outlines.

Android 9.0		
Version	**Release Date**	**Features**
9.0	6 August 2018	• It does not have three button setup at the bottom buttons only single pill-shaped button and gestures for controlling things like swiping left-right to switch between recently opened apps, etc. This version is designed to

		extend the battery, including prediction of app that can be used in the device by machine learning. It has a feature named Shush that automatically puts your device at Do Not Disturb mode when you turn your phone screen-down on a flat surface.

Android SDK

- It is Android Software Development Kit.
- It is a collection of tools which is used to develop android application.
- These include a debugger, libraries, a handset emulator, documentation, sample code, and tutorials.
- The currently supported IDE for Android is Eclipse.
- It use the Android Development Tools(ADT) Plugin.
- Netbeans IDE is also supports Android development via plugin.
- Developer may use any text editor to edit java and XML files, then use command line tools(JDK) to create, debug and build Android application.

Following are the steps for download and install android SDK

▶ **System Requirement :**

→ **Operating Systems :**

» Windows XP (32-bit), Vista (32- or 64-bit), or Windows 7 (32- or 64-bit)

» Mac OS X 10.5.8 or later (x86 only)

» Linux (tested on Ubuntu Linux, Lucid Lynx)

→ GNU C Library (glibc) 2.7 or later is required.

→ On Ubuntu Linux, version 8.04 or later is required.

→ 64-bit distributions must be capable of running 32-bit applications.

→ **Hardware :**

» Dual-core cpu

» Min. 2 GB of ram

» 40 GB of HDD.

» And other standard hardwares

▶ **Steps to download :**

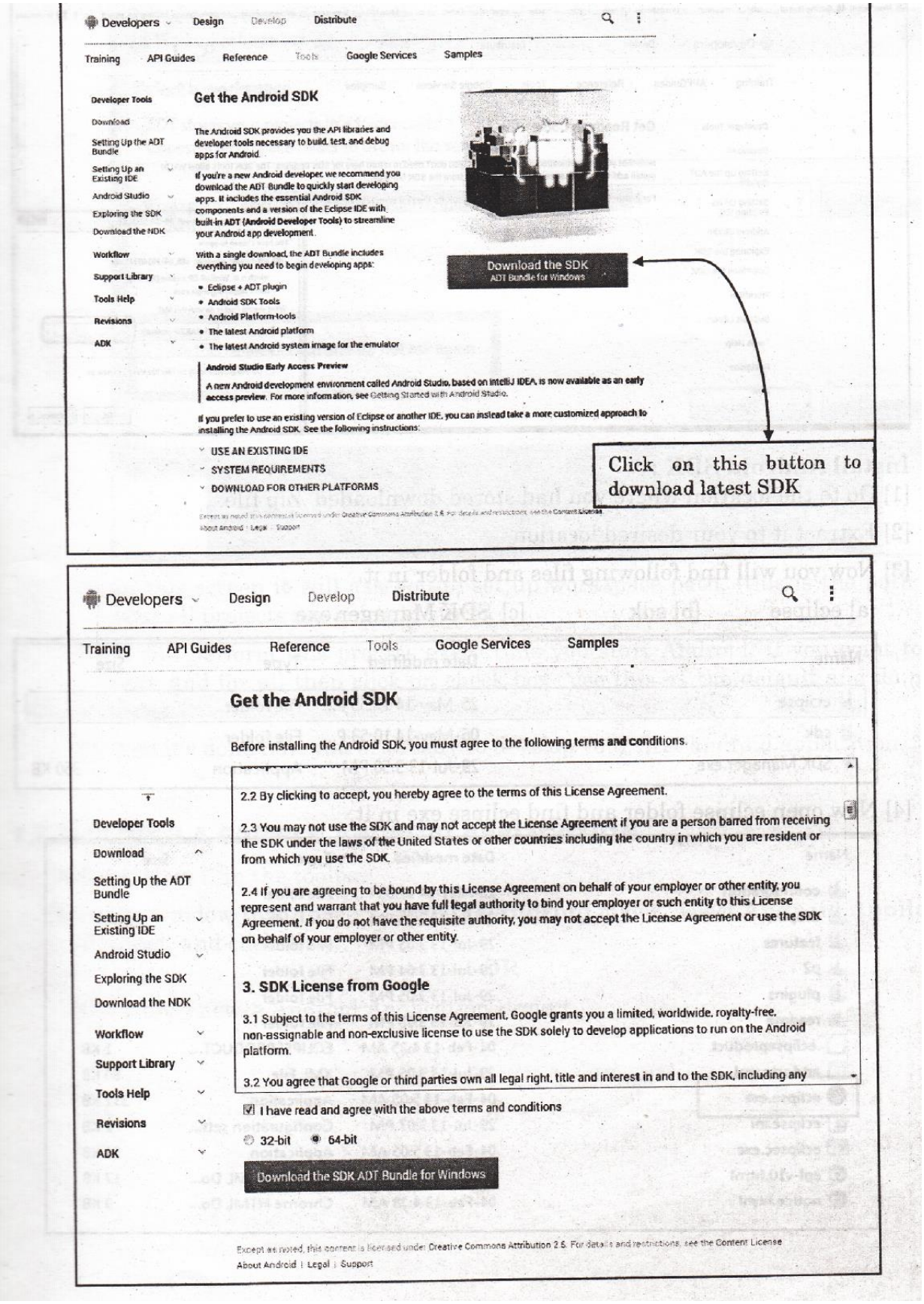
[1] Open your web browser and type following url in it.
https://developer.android.com/sdk/index.html

[2] You will find following screen.

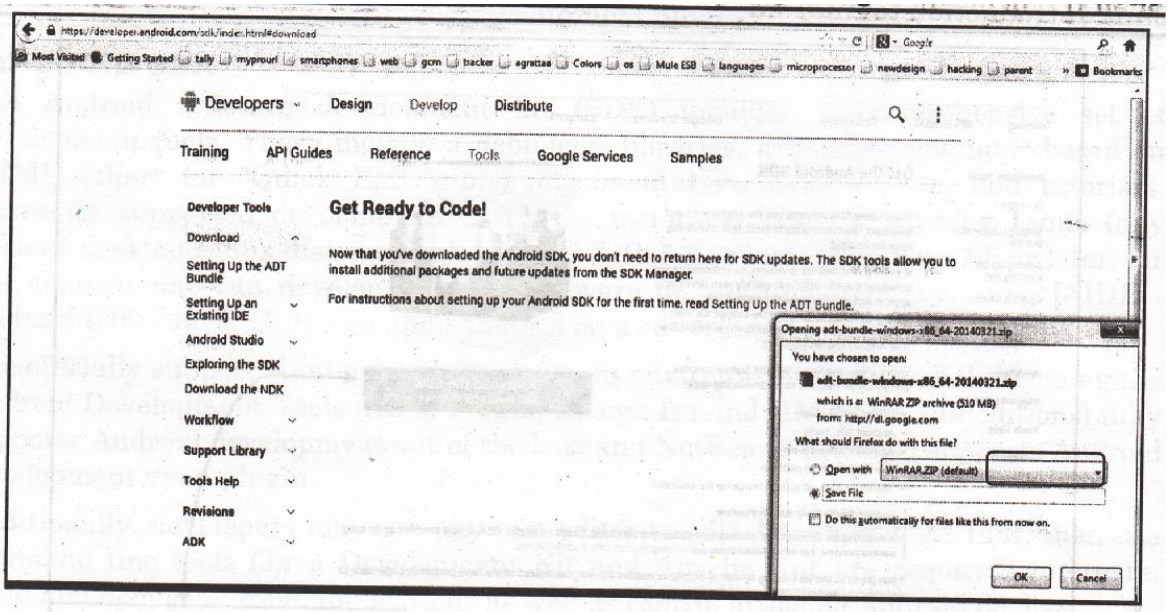
[3] Next Screen is for license agreement. In that agree to their rules and then provide your OS bit (i.e. 32 bit or 64 bit).

[4] Now click on "download SDK" button.

[5] Now you will find the 3rd screen. You can notice that its size is 510MB so you need proper net speed to download it.


Developers
Design
Develop
Distribute
Training
API Guides
Reference
Tools
Google Services
Samples
Developer Tools
Download
Setting Up the ADT Bundle
Setting Up an Existing IDE
Android Studio
Exploring the SDK
Download the NDK
Workflow
Support Library
Tools Help
Revisions
ADK
Get the Android SDK
The Android SDK provides you the API libraries and developer tools necessary to build, test, and debug apps for Android.
If you're a new Android developer, we recommend you download the ADT Bundle to quickly start developing apps. It includes the essential Android SDK components and a version of the Eclipse IDE with built-in ADT (Android Developer Tools) to streamline your Android app development.
With a single download, the ADT Bundle includes everything you need to begin developing apps:
Eclipse + ADT plugin
Android SDK Tools
Android Platform-tools
The latest Android platform
The latest Android system image for the emulator
Android Studio Early Access Preview
A new Android development environment called Android Studio, based on IntelliJ IDEA, is now available as an early access preview. For more information, see Getting Started with Android Studio.
If you prefer to use an existing version of Eclipse or another IDE, you can instead take a more customized approach to installing the Android SDK. See the following instructions:
USE AN EXISTING IDE
SYSTEM REQUIREMENTS
DOWNLOAD FOR OTHER PLATFORMS
Download the SDK
ADT Bundle for Windows
Click on this button to download latest SDK
About Android | Legal | Support

Developers
Design
Develop
Distribute
Training
API Guides
Reference
Tools
Google Services
Samples
Get the Android SDK
Before installing the Android SDK, you must agree to the following terms and conditions.
2.2 By clicking to accept, you hereby agree to the terms of this License Agreement.
2.3 You may not use the SDK and may not accept the License Agreement if you are a person barred from receiving the SDK under the laws of the United States or other countries including the country in which you are resident or from which you use the SDK.
2.4 If you are agreeing to be bound by this License Agreement on behalf of your employer or other entity, you represent and warrant that you have full legal authority to bind your employer or such entity to this License Agreement. If you do not have the requisite authority, you may not accept the License Agreement or use the SDK on behalf of your employer or other entity.
3. SDK License from Google
3.1 Subject to the terms of this License Agreement, Google grants you a limited, worldwide, royalty-free, non-assignable and non-exclusive license to use the SDK solely to develop applications to run on the Android platform.
3.2 You agree that Google or third parties own all legal right, title and interest in and to the SDK, including any
I have read and agree with the above terms and conditions
32-bit
64-bit
Download the SDK ADT Bundle for Windows
Except as noted, this content is licensed under Creative Commons Attribution 2.5. For details and restrictions, see the Content License
About Android | Legal | Support



▶ Install Android SDK :

[1] Go to the location where you had stored downloaded .Zip file.

[2] Extract it to your desired location.

[3] Now you will find following files and folder in it.

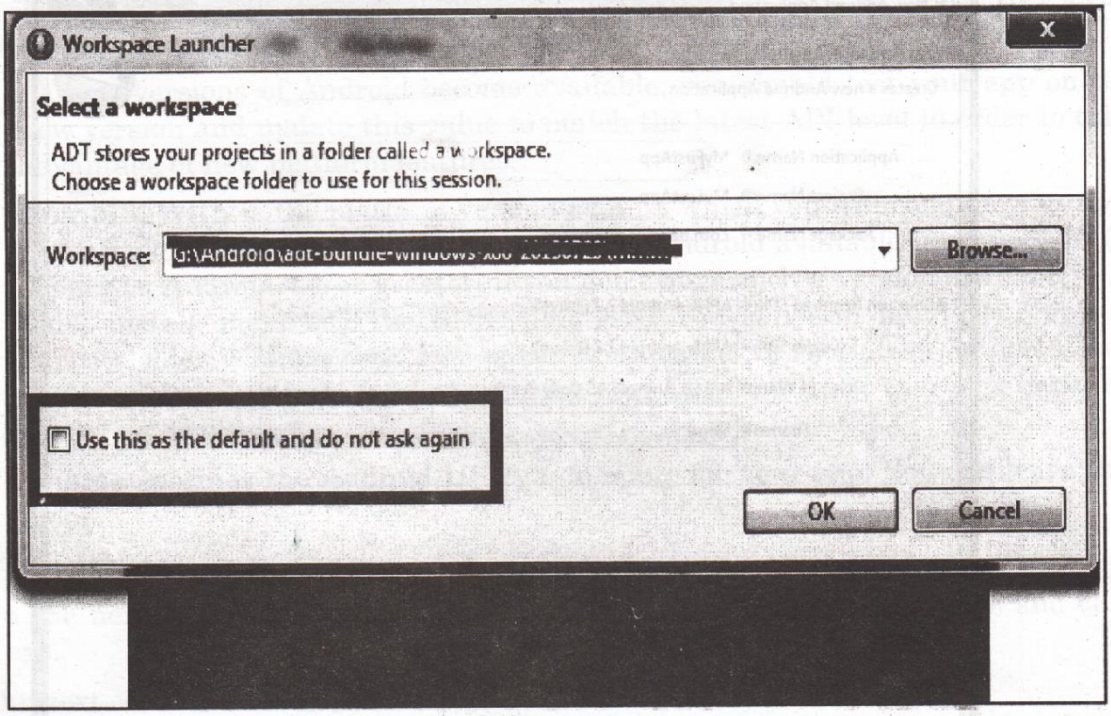
[a] eclipse [b] sdk [c] SDK Manager.exe

Name	Date modified	Type	Size
eclipse	25-May-14 12:55 P...	File folder	
sdk	06-May-14 10:53 P...	File folder	
SDK Manager.exe	29-Jul-13 3:50 PM	Application	350 KB

[4] Now open eclipse folder and find eclipse.exe in it.

Name	Date modified	Type	Size
configuration	25-May-14 12:55 P...	File folder	
dropins	29-Jul-13 3:05 PM	File folder	
features	29-Jul-13 3:05 PM	File folder	
p2	29-Jul-13 3:04 PM	File folder	
plugins	29-Jul-13 3:05 PM	File folder	
readme	29-Jul-13 3:05 PM	File folder	
.eclipseproduct	04-Feb-13 4:25 AM	ECLIPSEPRODUCT...	1 KB
artifacts.xml	29-Jul-13 3:05 PM	XML File	80 KB
eclipse.exe	04-Feb-13 5:05 AM	Application	312 KB
eclipse.ini	29-Jul-13 3:07 PM	Configuration sett...	1 KB
eclipsec.exe	04-Feb-13 5:05 AM	Application	24 KB
epl-v10.html	04-Feb-13 4:28 AM	Chrome HTML Do...	17 KB
notice.html	04-Feb-13 4:28 AM	Chrome HTML Do...	9 KB

[5] Double click on it and it will be started. And you will find following screen.



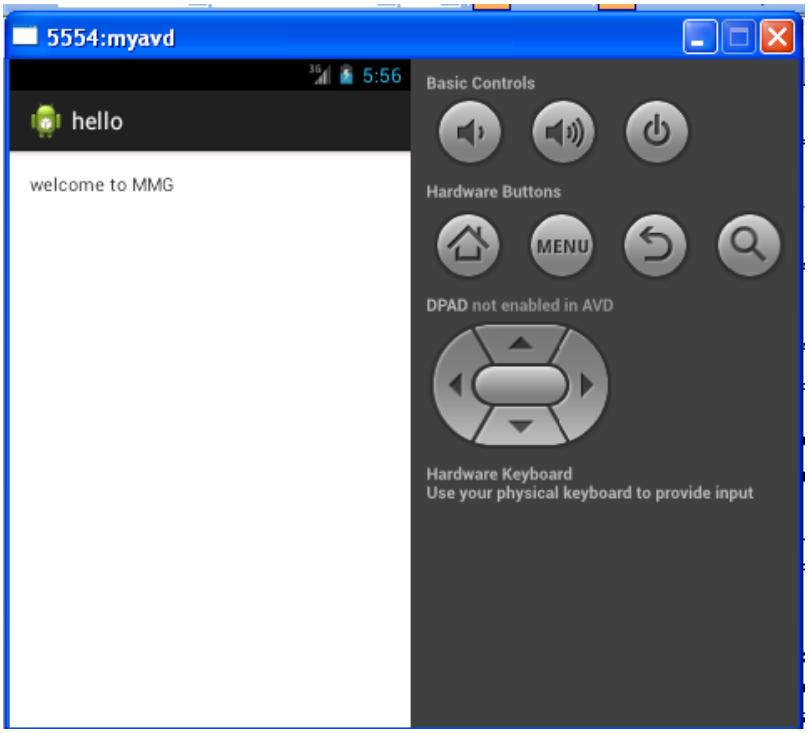
[6] In this screen it will ask you to set up workspace path. (this is the path where your all projects are going to store).

[7] It will perform this process every time you start Android. If you want to set it once and for all then click on check box "use this as the default and do not ask again".

[8] And it's done now you are ready to develop your first android application.

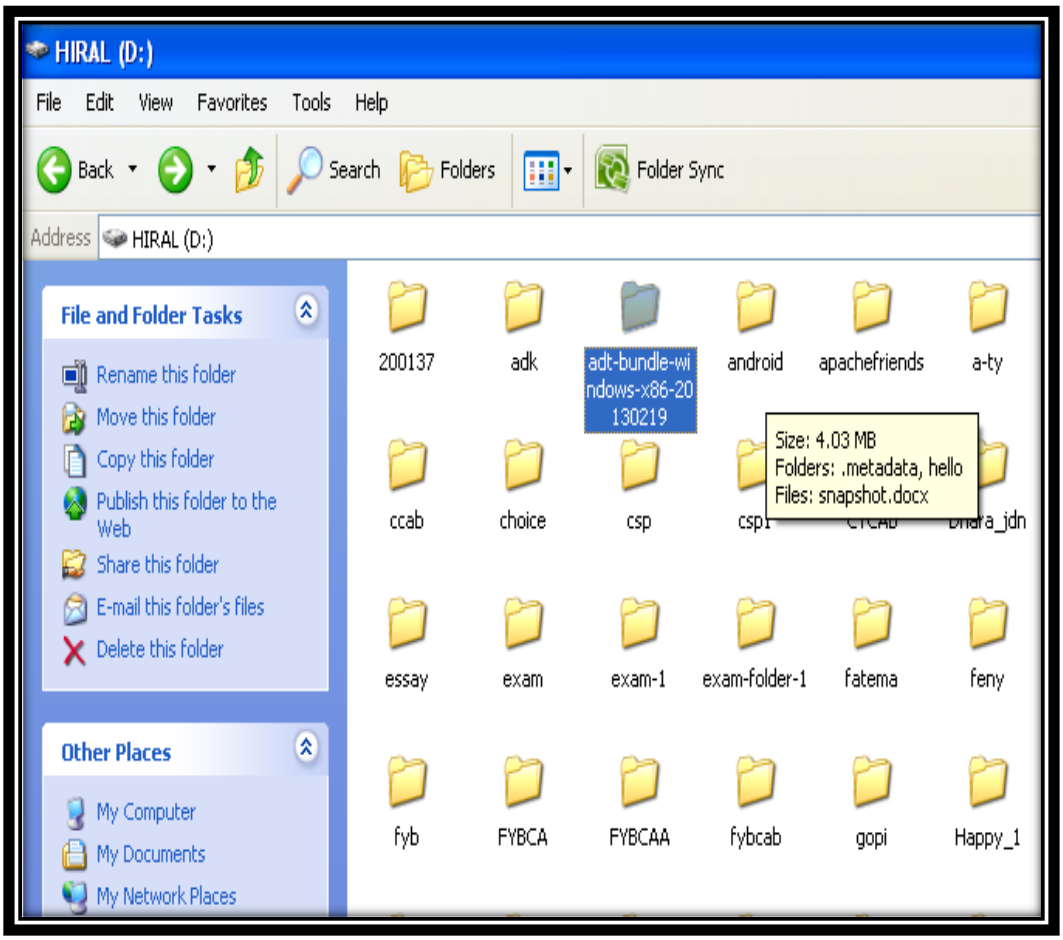
Building a Simple Android Application

1) Open the folder **adt-bundle-windows-x86-20130219** for create program.
2) Open eclipse folder.
3) Select eclipse program.
4) Select the path when you want to save your program by clicking browse button and click on OK button.
5) Select file->new->Android Application Project and give the application name hello and click on next button.
6) Then click on next button.
7) Click on next button.
8) Click on next 2 times.
9) Give the activity name Prg1 and remind that the first character is always capital.
10) Layout name's first character is always small.
11) And click on finish button.
12) Now two files are displayed one is graphical file and other is the xml file.
13) Now change the setting through the xml file.
14) Create the AVD by clicking on AVD manager
15) Click on new give the name select the device and click on ok.
16) Select your avd click on start and click on launch.
17) Now run the program by right clicking on the project name and select run as Android Application project.

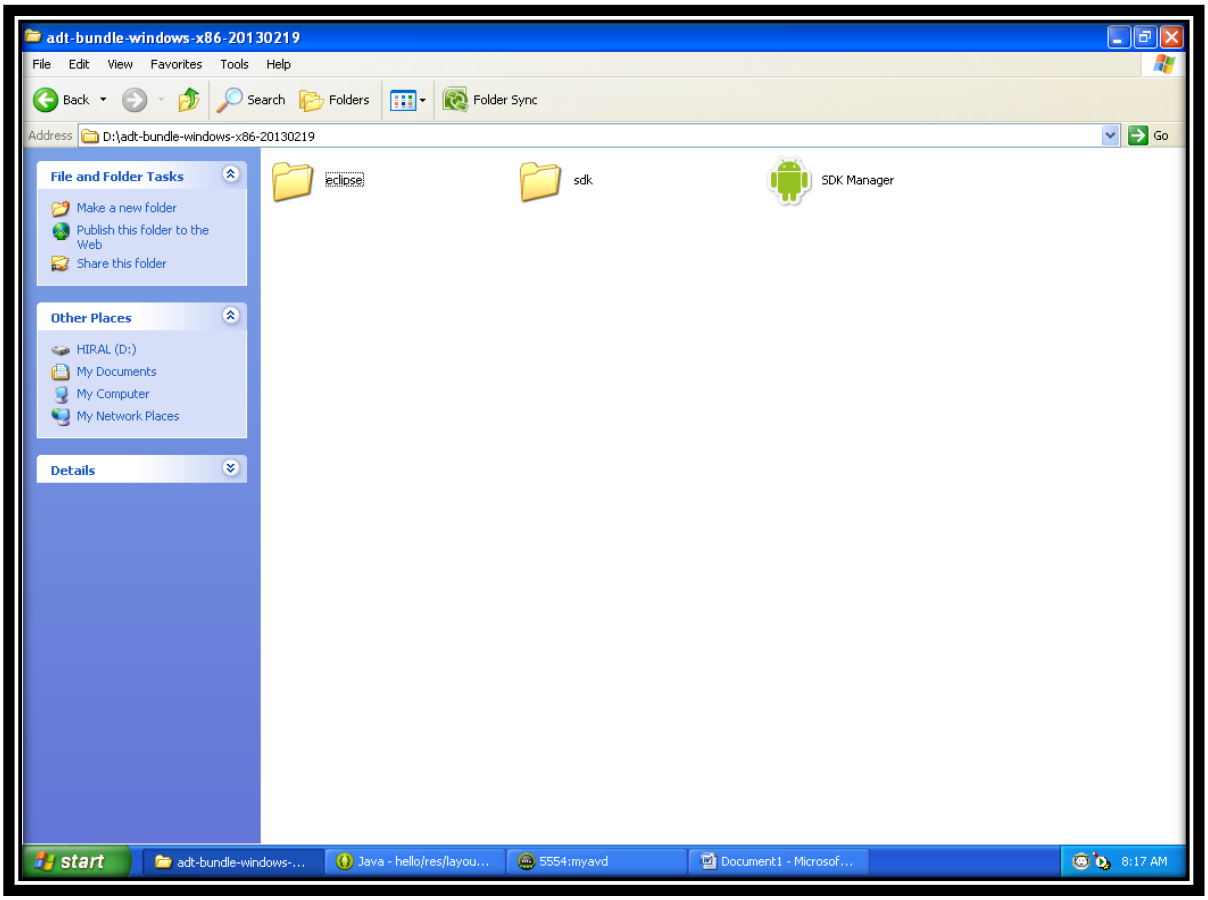


How to start Android?

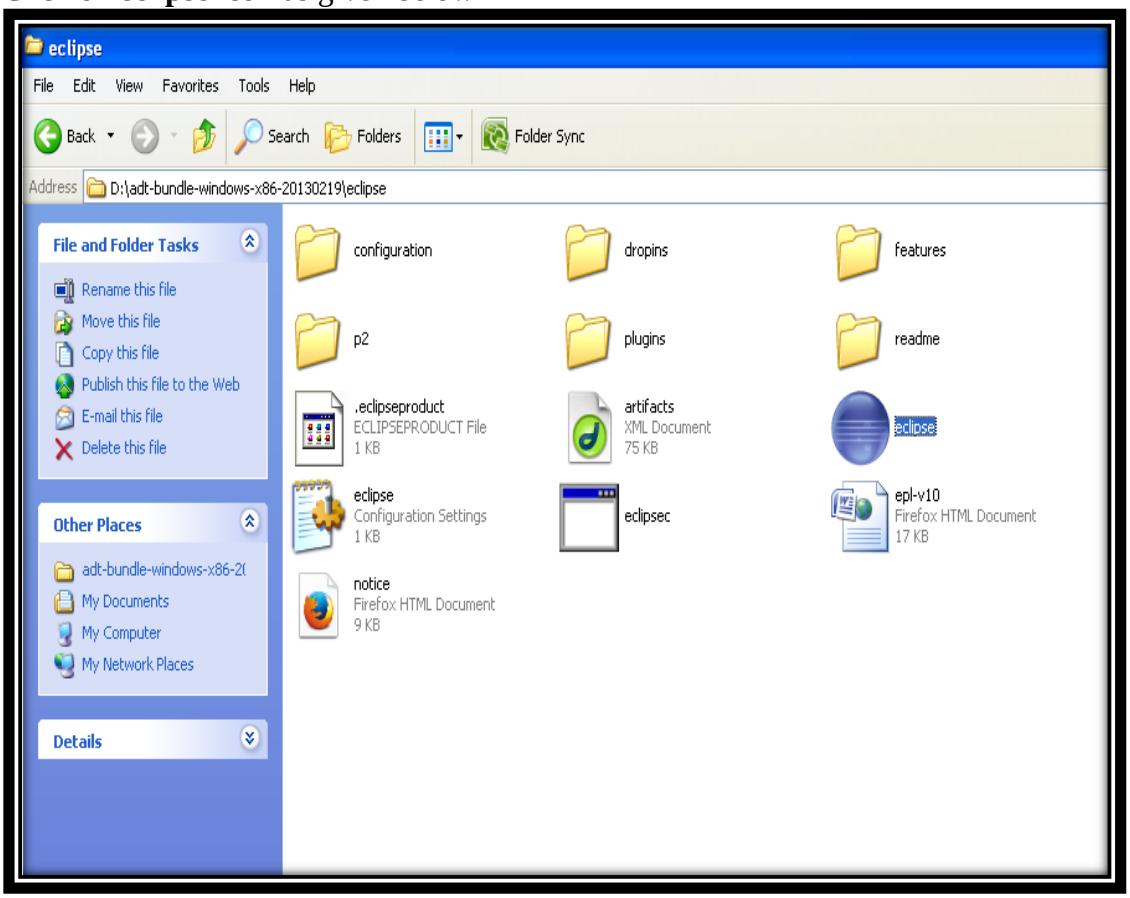
- **Double click on adt-bundle-windows folder as given below.**



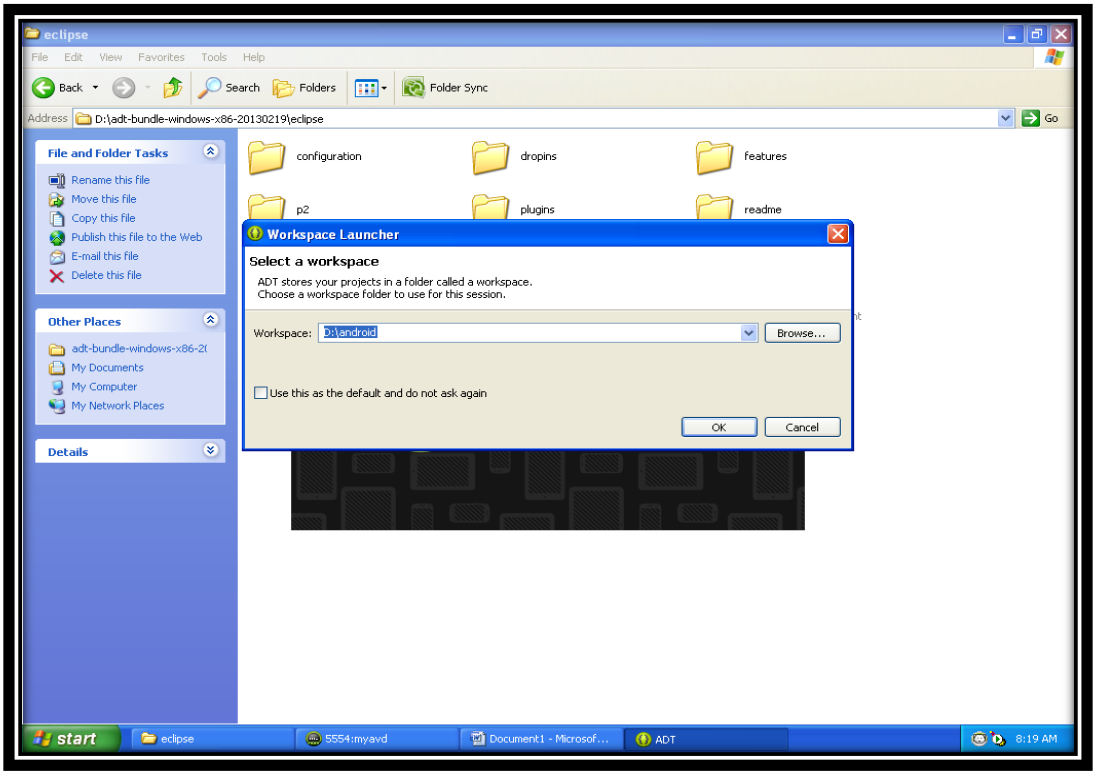
- Select the **eclipse** folder



Click on **eclipse icon** as given below



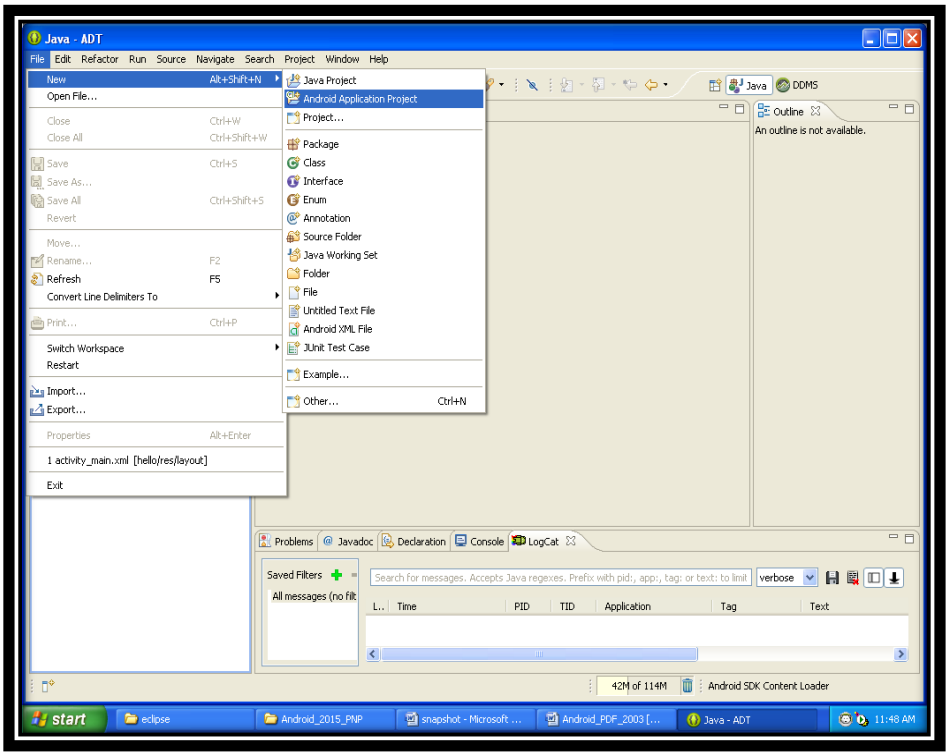
- Choose your **workspace** where you can want **to save your project** by clicking on **browse** button



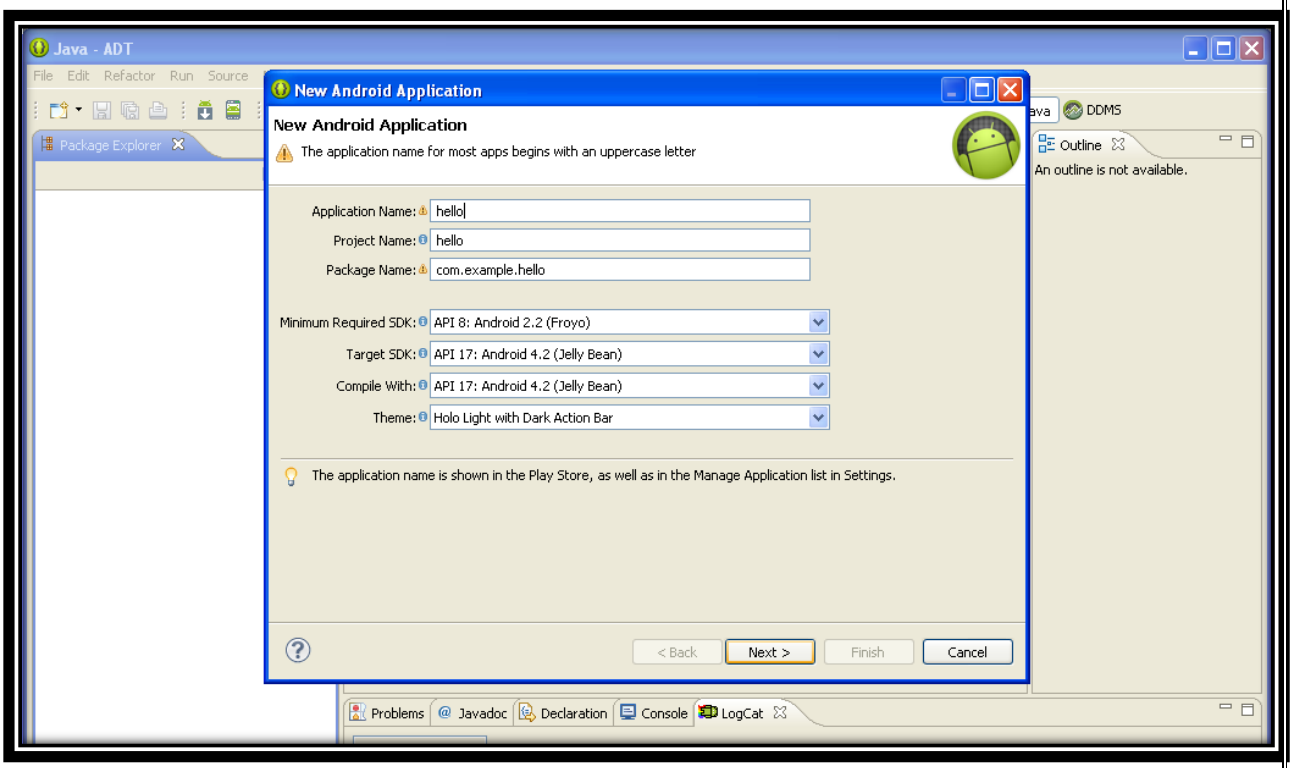
- Then click on ok button

How to create an Application?

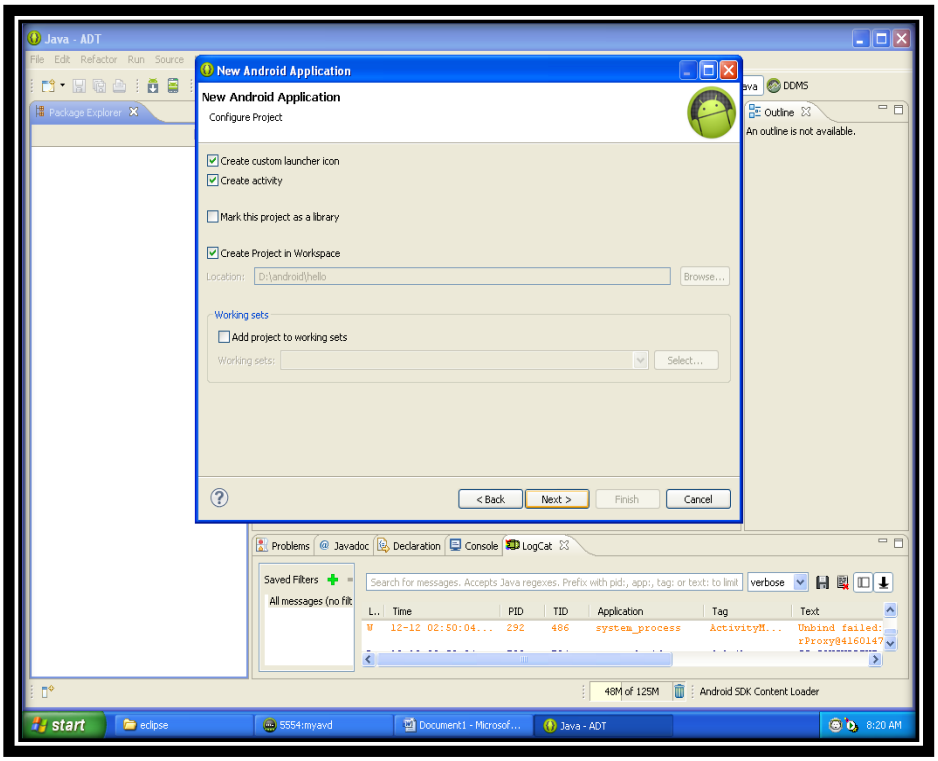
- To create android application follow these steps
- **File->Android Application Project**



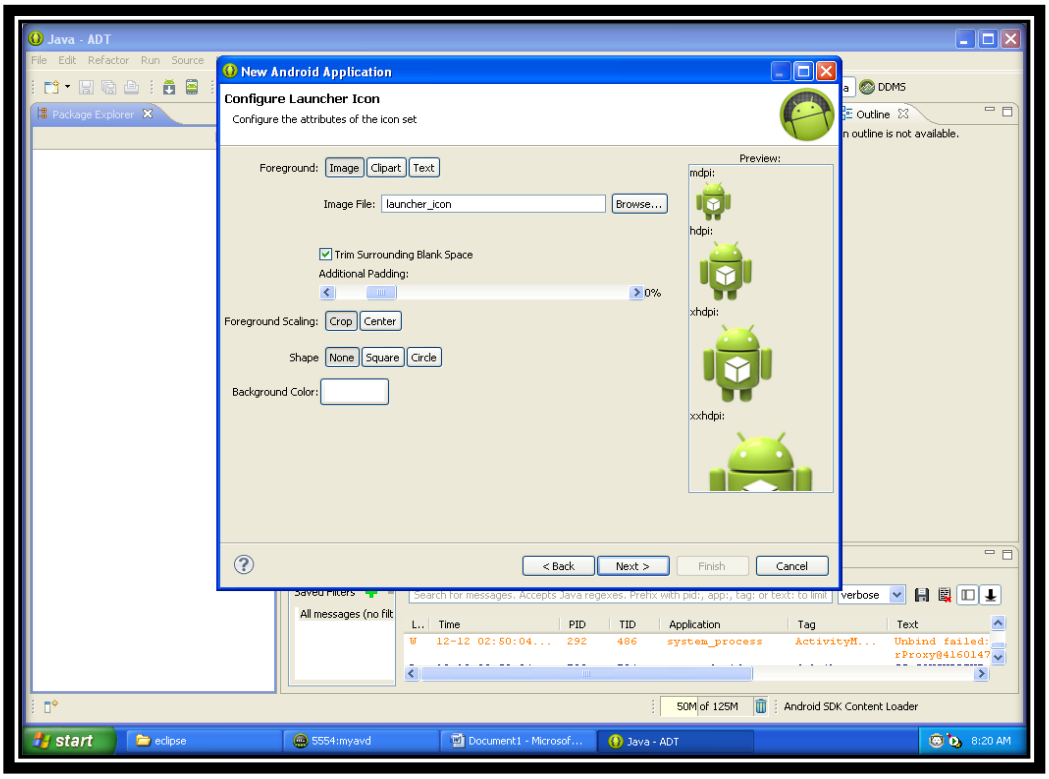
- Write **your application name**, choose your **version** (in which version you want to develop your application) then click on **Next** button



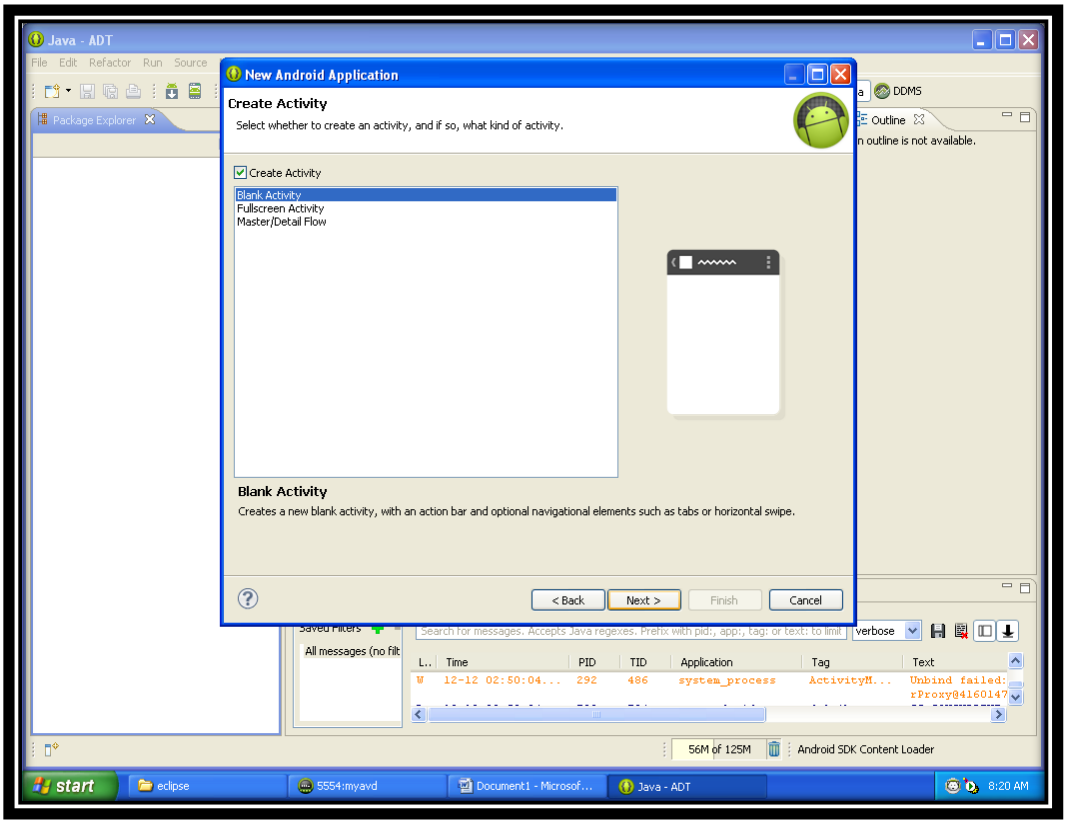
- Click on **Next** button



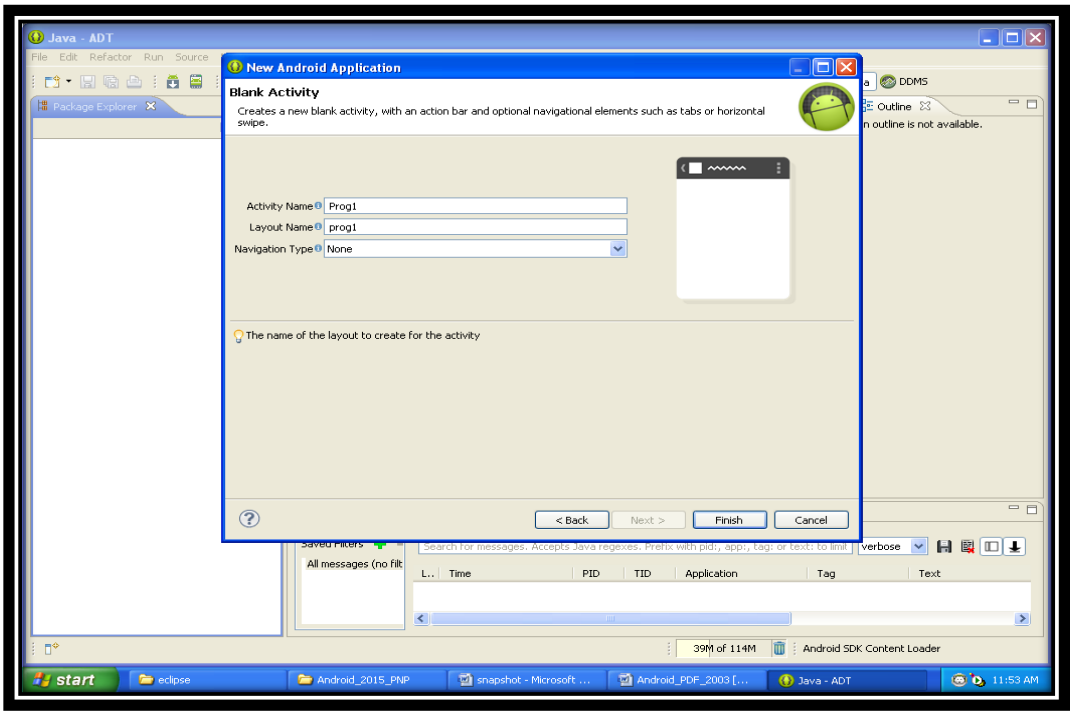
- Click on **Next** button



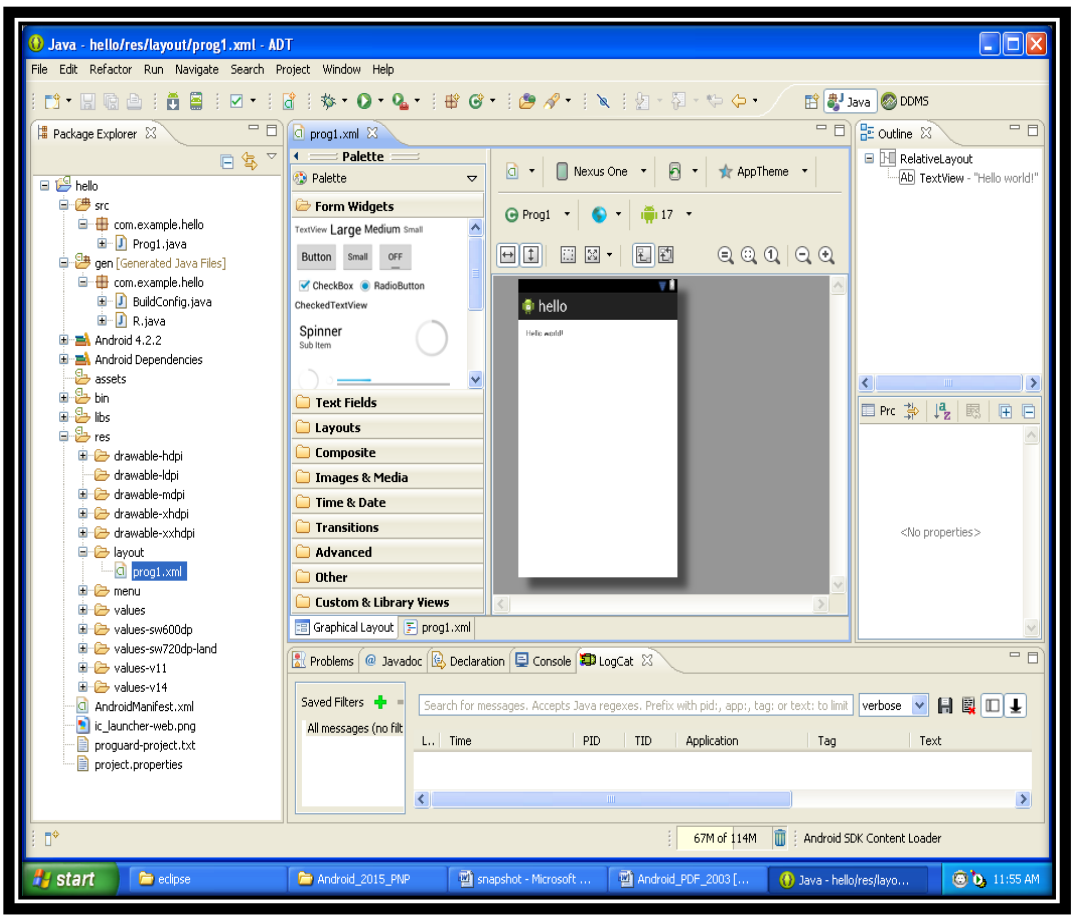
click on **Next** button



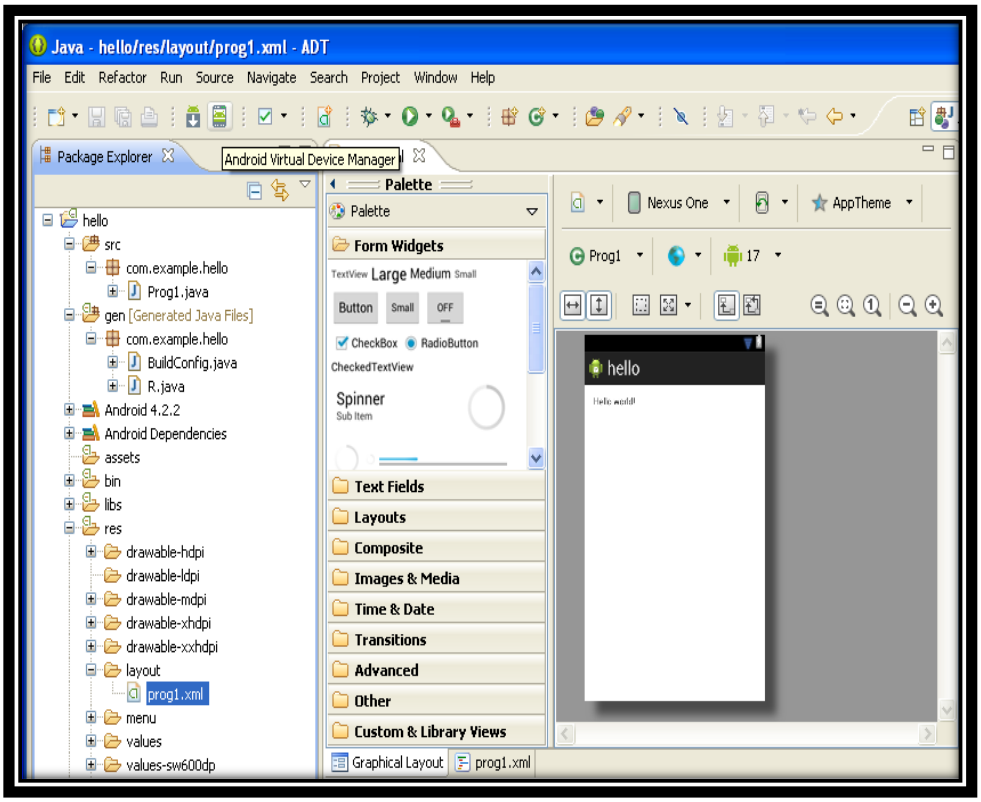
- Wirte name of your **activity file (java file**) and **layout file(xml file).**
- Then click on **finish** button.



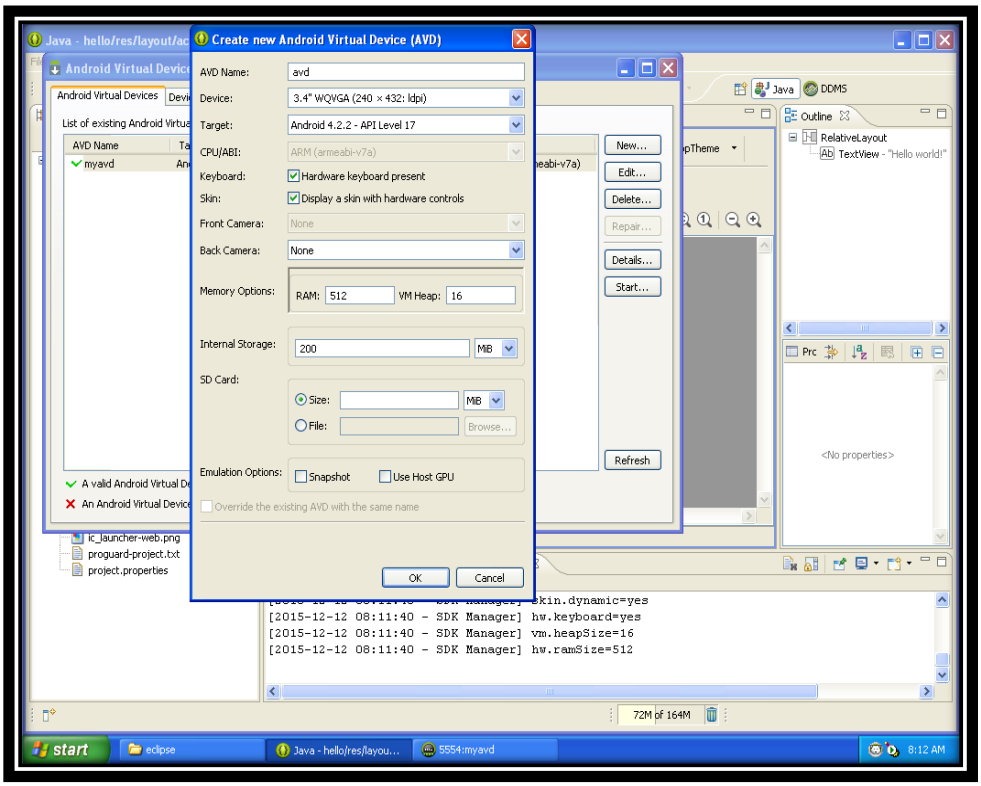
- **Structure of your application will look like as given below**



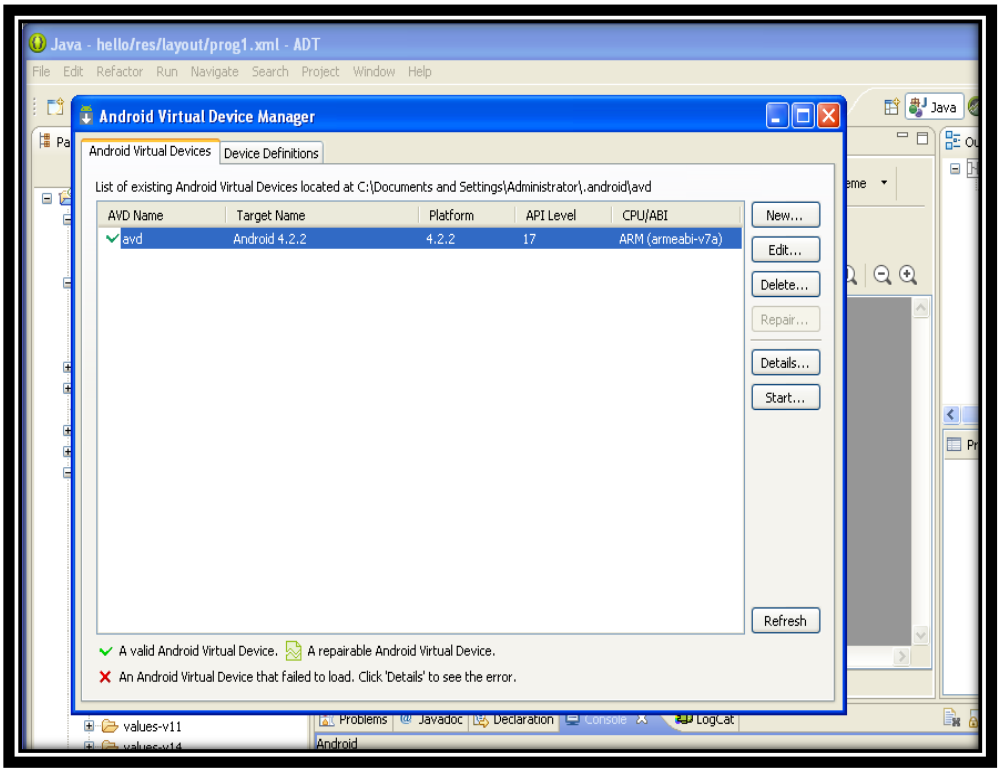
- Then create **AVD(Android virtual device)** to **run your Application**
- Click on the **Android Virtual Device Manager button** to create an AVD



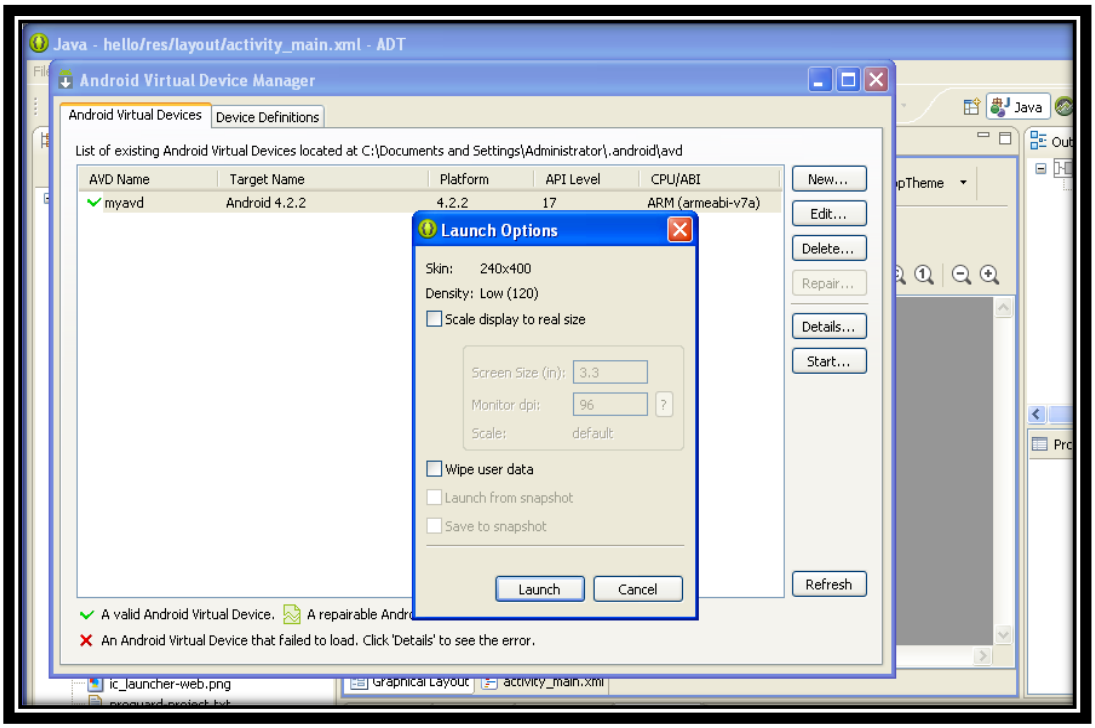
- Give **AVD name** and choose **your device** then click on **OK** button



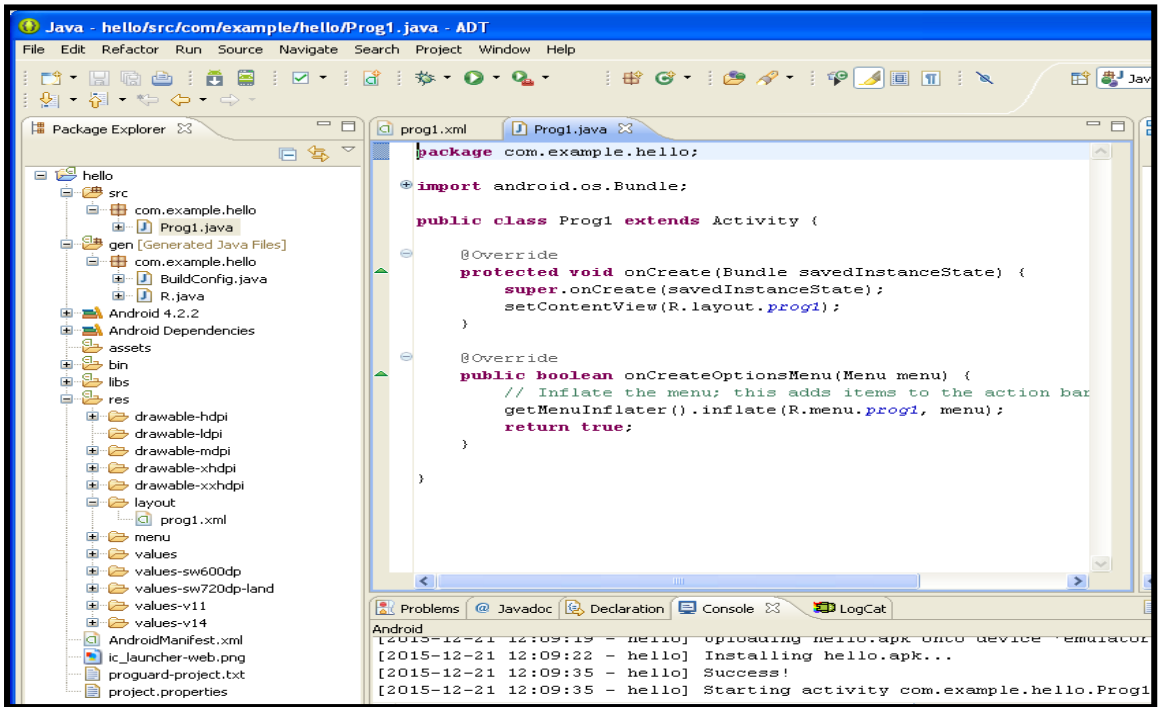
- Then after **select your AVD** and click on **Start** button



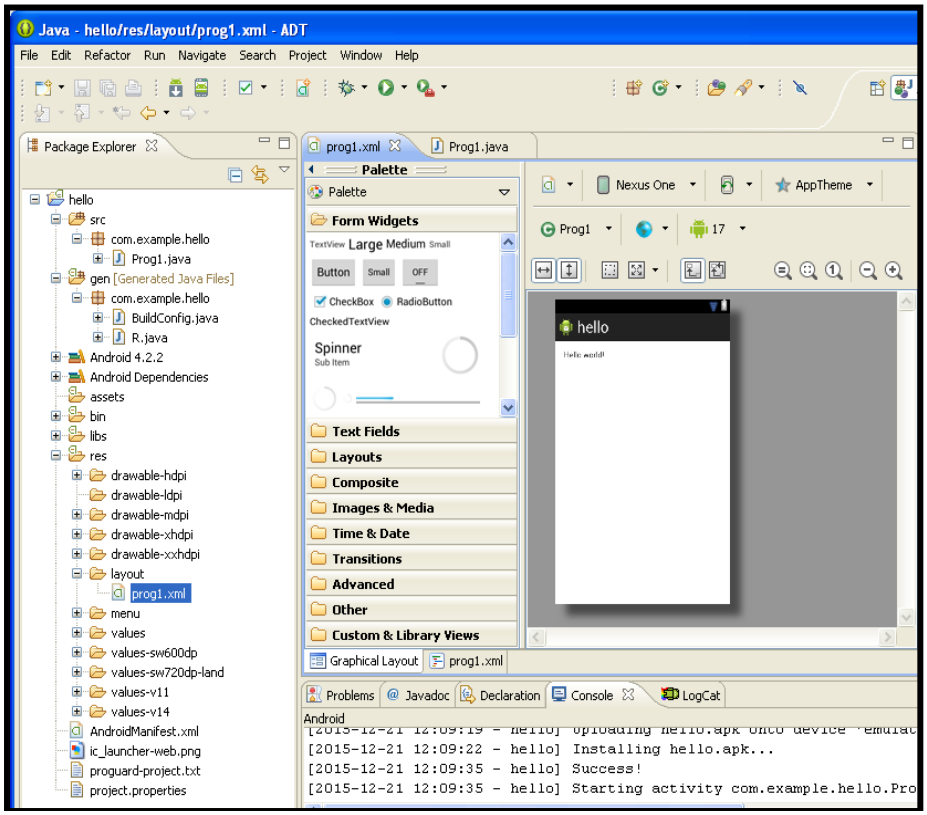
Click on **Launch** button



- **For coding open your java file from src > [app name]Prog1.java**

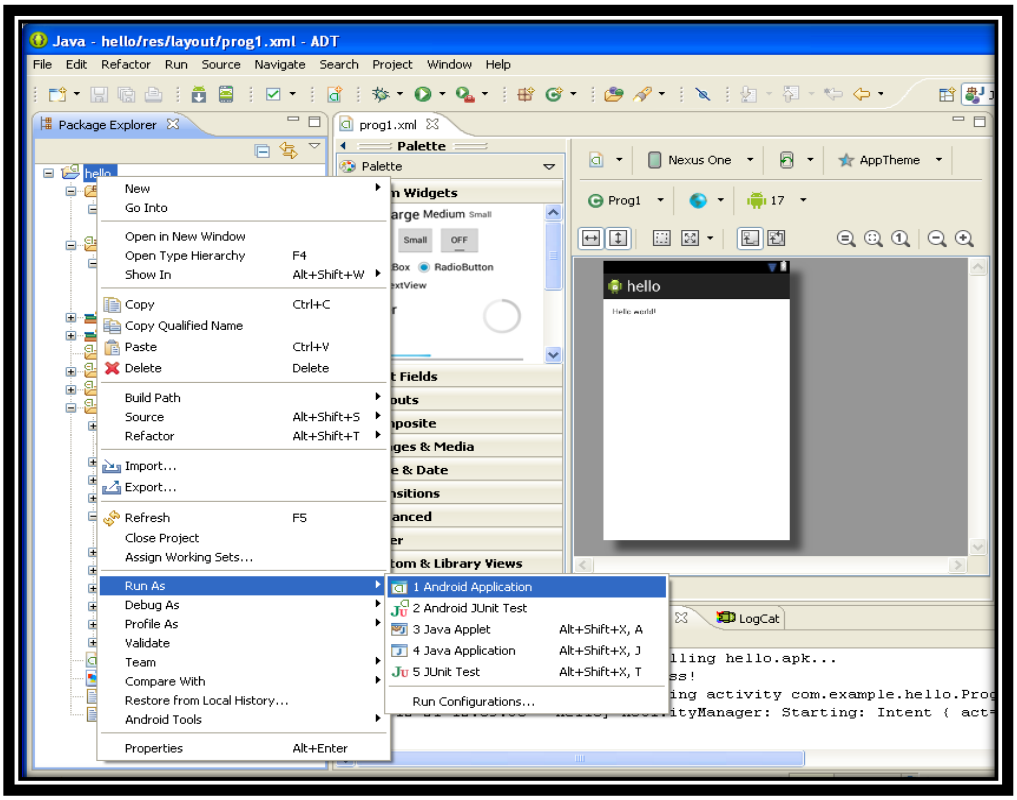


- **For Designing open your xml file from res->layout > [app name]prog1.xml**

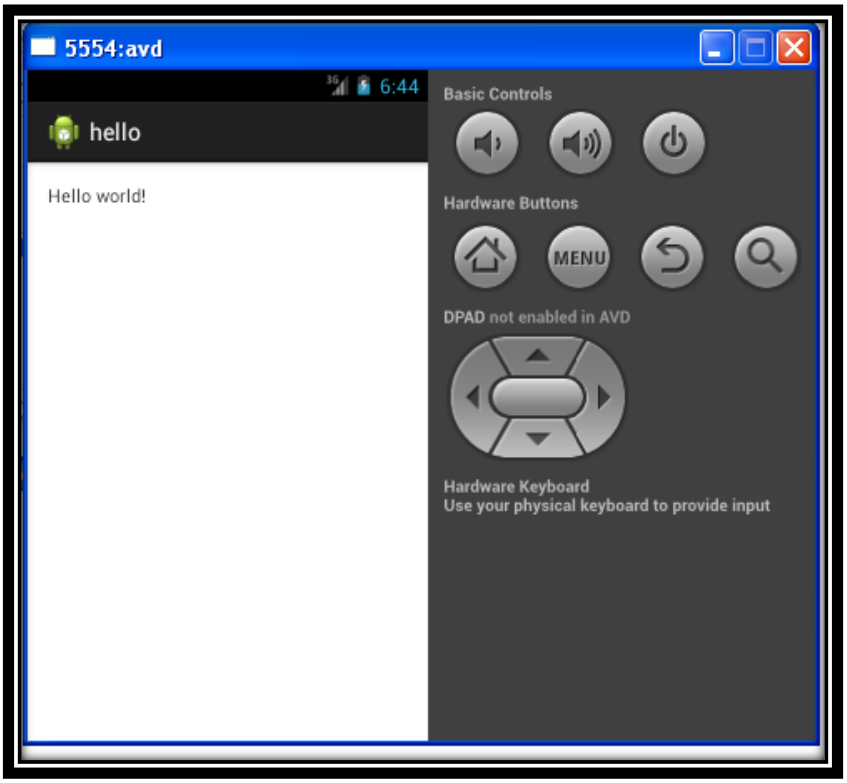


How To Run your Application?

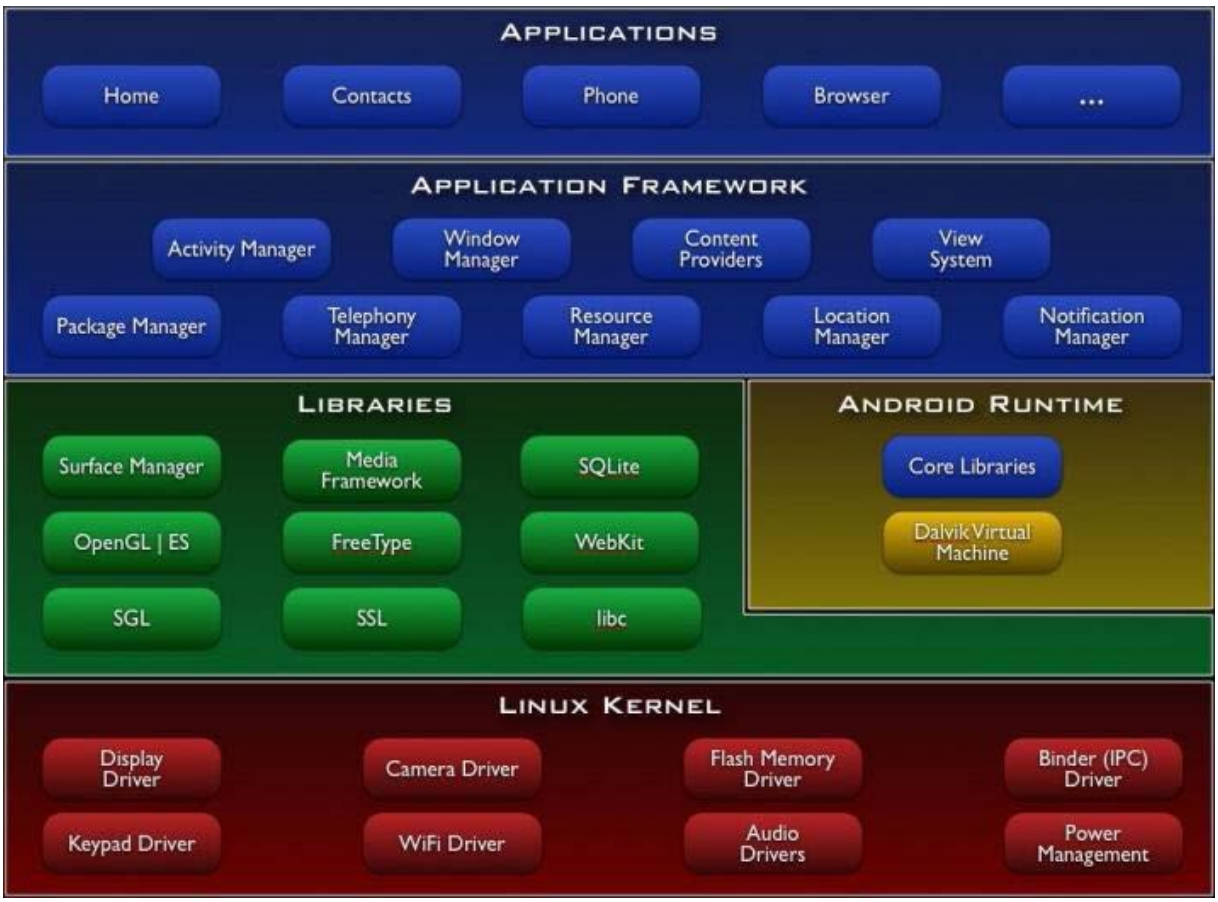
- To **Run** your **Android application**
- Right click on your project **choose Run As -> Android Application option**.



Output will look like this.



Android Architecture



Linux kernal

- It is the bottom layer.
- It provides basic functionality like process management, memory management, device management like camera, keypad, display etc.

Libraries

- It is the top on the linux kernel.
- It is set of libraries including open-source Web browser engine webkit.
- It contains SQLite database which is a useful repository for storage data, libraries to play and record audio and video, SSL libraries are use for Internet security.

Android Runtime

- It is the third section of the architecture.
- It provides the key components called Dalvik Virtual Machine which is a one kind of java virtual machine specially designed for android.

Application Framework

- It provides many higher level services to applications in the form of Java classes.
- Application developers are allowed to make use of these services in their application.

Applications

- It is at the top.
- You will write your application to be installed on this layer only.
- Examples are contacts, browser games etc.

Anatomy (structure) of an android application

- **Src:** Java source code files will be available here.
- **Gen:** The gen directory in an Android project contain auto generated files. You can see R.java inside this folder which is a generated class. R.java is automatically created by the Eclipse IDE.
- **Res:** Android supports resources like images and certain XML configuration files ; these can be keeping separate from the source code. All these resources should be placed inside the res folder. This res folder will be having sub-folders to keep the resources based on its type.
- **/res /values :** Used to define strings, colors, dimensions, styles and static arrays of strings or integers. By convention each type is store in a separate file, e.g. string are defined in the res/values/strings.xml file.
- **/res/values-v11 :** is the values of the API version 11,and /res/values –v14 is the values of the API version 14
    - **/res/values :**layout for normal screen size or default
    - **/res/values –small :** layout for small screen size
    - **/res/values –large :**layout for large screen size
    - **/res/values –xlarge :** layout extra large screen size
    - **/res/values –xlarge-land :**layout for extra- large in landscape orientation
    - **/res/values –sw600dp :** layout for tablets or layout for 7” tablets (600dp wide and bigger)
    - **/res/values – sw720dp :** layout for 10” tablets (720dp wide and bigger)
    - **/res/values – w600dp :** layout for multi-pane (any screen with 600dp available width or more)
    - **/res/layout :** this folder contains the layouts to be used in the application.
    - **/res/menu :** This folder contains menu resources to be used in the application (Option Menu ,Context Menu or submenu)
    - **/res/drawable :**Drawable folders are resource directories in an application that provides different bitmap drawables for medium ,high , and extra high density screen.
        - **/res/drawable-ldpi :** bitmap for lower density
        - **/res/drawable-mdpi :** bitmap for medium density
        - **/res/drawable-hdpi :** bitmap for high density
        - **/res/drawable-xhdpi :** bitmap for extra high density
        - **/res/drawable-xxhdpi :** bitmap for X extra high density
- **libs :** External library files will be placed in this folder. If you want to any external library in your project place the library jar inside this folder and it will be added to the classpath automatically.
- **assets :** This folder contains raw hierarchy of files and directories.
- **bin:** Bin folder is the area used by the compiler to prepare the files to be finally packaged to the application ‘s APK file. This includes
    - Compiling your Java code into class files
    - Putting your resources (including images) into a structure to be zipped into the APK

This is the output directory of the build. This is where you can find the final **.apk** file and other complied resources.

- **AndroidManifest.xml :** All the android applications will have an AndroidManifest.xml file in the root directory. This file will contain essential information about the application to the Android system.
- **ic_launcher-web.png :** This is an icon to be used in Google play. Applications on Google Play require a high fidelity version of the application icon.
- **proguard-project.txt :** Everything in the proguard-project.txt file will be in commented out state. Because in general most people don't have any project specific needs, just to run ProGuard tool with standard settings.
- **Project.properties** : project.properties is the main project's properties file containing information such as the build platform target and the library dependencies has been renamed from default.properties in older SDK versions. This file is integral to the project.

Android Terminologies

1) .apk file

- It is an Android application package file.
- Each application is compiled and packaged in a single file that includes all the application code.
- The application package file can have any name but must use the .apk extension.

2) .dex file

- It is a compiled android application code file.
- They are compiled in the Dalvik Executable files, which are in turn zipped into a single .apk file on the devices.

3) Action

- It is a string value assign to intent.
- Action string can be defined by android developer.

4) activity

- A single screen that support java code is derived from the activity class.
- Activity is visibly represented by full screen window.

5) adb

- It is Android Debug Bridge.
- It is a command line debugging application included with the SDK.

6) Application

- An android application consist of one or more activities, services, listeners and intent receivers.
- These files are packaged in a single file called application package file.

7) Canvas

- It is the simplest way to draw 2D objects on the screen.

8)Content Provider

- It handles content query strings of a specific format.

9) Dalvik

- It is Android platform's virtual machine.

- The Dalvik VM is an interpreter only virtual machine that executes files on the Dalvik Executable(.dex) format.

10) DDMS

- It is Dalvik Debug Monitor Service.
- It is a GUI debugging application included with the SDK.
- It provides screen capture, log dump, and process examination capabilities.

11)Dialog

- A window that acts as a lightweight form.
- A dialog can have button controls only and is intended to perform a simple action and return a value.

12) Drawable

- It is typically loaded into another UI element, for example as a background image.

13)Intent

- It includes several criteria fields that you can supply, to determine what application receives the intent and what the receiver does when handling the Intent.

14)Intent Filter

- It is an application that declares in its manifest file, to tell the system what types of intents each of its components.

15)Broadcast Receiver

- It is an application class that listens for intents that are broadcast, rather than being sent to a single target application.

16)Layout Resource

- An XML file that describes the layout of an activity screen.

17)Manifest file

- An XML file that each application must define, to describe the application's package name, version, components and describes the various activities.

18)Nine-patch/9-patch image

- It is a resizable bitmap resource that can be used for background or other image on the device.

19)OpenGL ES

- Android provides OpenGL ES Libraries that you can use for fast, complex 3D images.
- It is harder to use than a Canvas object, but better for 3D objects.

20)Resources

- It is a programmatic application components that are external to the compiled application code, but which can be loaded from application application code, using a well-known reference format.

21)Service

- It is an object of a class that runs in background to perform various actions such as playing music or monitoring network activity.

22)Surface

- It represents block of memory that gets composited to the screen.
- It holds canvas object for drawing and provides various helper methods to draw layers and resize the surface.

23)SurfaceView

- It wraps a surface for drawing and exposes methods to specify its size and format dynamically..

24)Theme

- It is a set of properties(text, size,background color and so on) are collected together to define various default display setting.

25) URIs in Android

- Android uses URI strings as the basis for requesting data in a content provider(such as a retrieve a list of contacts) and for requesting actions in an Intent(such as opening a web page in a browser).

26)View

- It draw a rectangular area on the screen and handles click, keystrock and other interaction events.
- It is a base class for most layout like textbox,windows,buttons etc.

27)Viewgroup

- It is responsible for deciding where child views are positioned and how large they can be.

28)Widget

- It is a group of elements and other UI components, such as a textbox or popup menu.

29)Window

- It is android application, and derived from the abstract class window that specifies the elements of a generic window, such as title bar,menus etc.

Application Context, activities, services, intents

Context

- It is the most commonly used element in android application.

Application

- This object is created whenever one of your android component is started.
- It is started in a new process with a unique ID under a single user.
- This object provides following lifecycle methods.
- **onCreate()**
  - It is called before the first components of the application starts.
- **onLowMemory()**
  - It is called when android system requests cleans up the memory.
- **onTerminate()**
  - It is used only for testing.
- **onConfigurationChanged()**
  - It is called when some configuration are changes.

If an android system needs to terminate processes it follows the following priority system.

Process Status	Description	Priority
Foreground	It is an application in which the user is interacting with an activity. If a service is executing one of its lifecycle methods or a broadcast receiver which runs onReceive() method.	1
Visible	User is not activity, but the activity is still visible or the application has a service which is used by	2

	visible activity.	
Service	Application with a running service.	3
Background	Application with only stopped activities and without a service or executing receiver.	4
Empty	Application without any active components	5

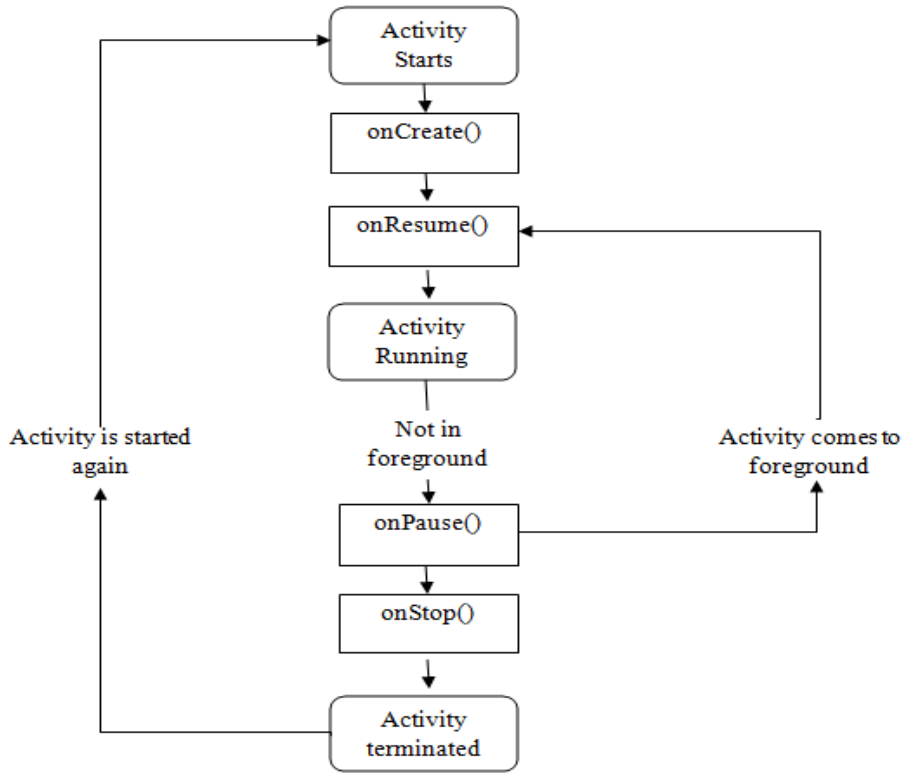
Activity Lifecycle

It has different types of following state.

State	Description
Running	Activity is visible and interacts with the user.
Paused	Activity is still visible but partially hidden, instance is running but might be killed by the system.
Stopped	Activity is not visible.
Killed	Activity has been terminated by the system.

Some important methods of activity lifecycle are:

Method	Purpose
onCreate()	It is called when the activity is created.
onResume()	It is called when if the activity gets visible again and the user starts interacting with the activity again. It is used to initialize fields, register listeners.
onPause()	It is called once another activity gets into the foreground.
onStop()	It is called once the activity is no longer visible.



Services :

- A service is a component that runs in the background to perform long-running operations without needing to interact with the user.
- For example, a service might play music in the background while the user is in a different application, or it might fetch data over the network without blocking user interaction with an activity.
- A service can essentially take two states:

State	Description
Started	A service is **started** when an application components, such as an activity, starts it by calling startService (). Once started, a service can run in the background indefinitely, even if the component that started it is destroyed.
Bound	A service is **bound** when an application component binds to it by calling bindService (). A bound service offers a client-server interface that allows components to interact with the service, send requests, get results and even do so across processes with inter process communication (IPC).

- A service has lifecycle callback methods that you can implement to monitor changes in the service's state and you can perform work at the appropriate stage.
- The following diagram on the left shows the lifecycle when the service is created with startService () and the diagram on the right shows the lifecycle when the service is created with bindService ().
- To create a service, you create a Java class that extends the Service base class or one of its existing subclasses.
- The Service base class defines various callback methods and most important are given below.
- You don't need to implement all the callbacks methods.
- However, it's important that you understand each one and implement those that ensure your app behaves the way users expect.

Callback	Description
onStartCommand ()	The system calls this method when another component, such as an activity, requests that the service be started, by calling startService (). If you implement this method, it is your responsibility to stop the service when its work is done, by calling stopSelf () or stopService () methods.
onBind ()	The system calls this method when another component wants to bind with the service by calling bindService (). If you implement this method, you must provide an interface that clients use to communicate with the service, by returning an IBinder object. You must always implement this method, but if you don't want to allow binding, then you should return null.
onUnbind ()	The system calls this method when all clients have disconnected from a particular interface published by the service.
onRebind ()	The system calls this method when new clients have connected

	to the service, after it had previously been notified that all had disconnected in its onUnbind (Intent).
onCreate ()	The system calls this method when the service is first created using onStartCommand () or onBind(). This call is required to perform one-time setup.
onDestroy ()	The system calls this method when the service is no longer used and is being destroyed. Your service should implement this to clean up any resources such as threads, registered listeners, receivers, etc.

Intents and Intent Filters

- Intent is a messaging object you can use to request an action from another app component.
- Following are three fundamental use-cases:

To start an activity:

- An Activity represents a single screen in an app. You can start a new instance of an Activity by passing intent to startActivity().
- The Intent describes the activity to start and carries any necessary data.

To start a service:

- A service is a component that performs operations in the background without a user interface.
- You can start a service to perform one-time operation (such as download a file) by passing an Intent for startService().
- The Intent describes the service to start and carries any necessary data.

To deliver a broadcast:

- A broadcast is a message that any app can receive.
- The system delivers various broadcasts for system events, such as when the system boots up or the device starts charging.

➢ Intent Types :

There are two types of intents:
Explicit intent specify the component to start by name (the fully –qualified class name).You'll typically use an explicit intent to start a component in your own app, because you know the class name of the activity or service you want to start.

Implicit intents do not name a specific component, but instead declare a general action to perform, which allows a component from another app to handle it.

➢ Create an intent

- An intent object carries information that the android system uses to determine which component to start.
- The primary information contained in intent is the following.

Component name:-

- The name of the component to start.

- This is optional, but it's the critical piece of information that makes intent explicit, meaning that the intent should be delievered only to the app component defined by the component name.
- Without a component name, the intent is implicit.
- If you need to start a specific component in your app, you should specify the name.

Action:-

- A string that specifies the generic action to perform.
- The action largely determines how the rest of the intent is structured-particularly what is contained in the data and extras.
- You can use your intents within your app.
- Following are some action for starting an activity.

ACTION_VIEW:

- Use this action in an intent with startActivity () when you have some information that an activity can show to the user, such as a view in a gallery app, or an address to view in a map app.

ACTION_SEND:

- Also known as the "share" intent, you should use this in an intent with startActivity () when you have some data that the user can share through another app, such as an email app or social sharing app.
- You can specify the action for an intent with setAction () or with an Intent constructor.
- If you define your own action, be sure to include your app's package name as a prefix. For example:
- Static final String ACTION_TIMETRAVEL = "com.bagdaic action.TIMETRAVEL";

Data:

- The URI (an Uri object) that references the data to be acted on and/or the MIME type of that data .
- The type of data supplied is generally dictated by the intent's action.
- For example,if the action is ACTION_EDIT, the data should contain URI of the document to edit.
- When you creating an intent that is necessary to specify the type of its data.
- To set data call setData() method.
- To set the MIME type call setType() method.

Category

- It is a one type of string that contain additional information about the kind of component that should handle the component.
- Some common categories are as follow:

CATEGORY_BROWSABLE

It allows to be started by a web browser to display data.

CATEGORY_LAUNCHER

It is an initial activity of a task and is listed in the system's application launcher.

➢ Using a Pending Intent:

- A Pending Intent object is a wrapper around an Intent object.

- The primary purpose of a PendingIntent is to grant permission to other application to use the contained Intent as if it were executed from your app's own process.
- Major use cases for a pending intent include:
    - Declare intent to be executed when the user performs an action with your Notification.
    - Declare intent to be executed when the user performs an action with your App Widget.
    - Declare intent to be executed at a specified time in the future.

➢ Intent Resolution:

- When the system receives an implicit intent to start an activity, it searches for the best activity for the intent by comparing the intent to intent filters based on three aspects:
    - The intent action
    - The intent data (both URI and data type)
    - The intent category

- The following sections describe how intents are matched to the appropriate components (s) in terms of how the intent filter is declared in an app's manifest file.
    1. **Action test:**
       To specify accepted intent actions, an intent filter can declare zero or more <action> elements.

    2. **Category test:**
       To specify accepted intent categories, an intent filter can declare zero or more <category> elements.

    3. **Data test:**
       To specify accepted intent categories, an intent filter can declare zero or more <data> elements.

➢ Intent matching:

- Intents are matched against intent filters not only to discover a target component to activate, but also to discover something about the set of components on the device.
- For example, the Home app populates the app launcher by finding all the activities with intent filters that specify the ACTION_MAIN action and CATEGORY_LAUNCHER category.

Android Manifest file and its common settings

- It is a resource file which contains all the information about the application.
- It is a one type of xml file.
- Following are elements of Androidmanifest file.
    - <action>,<activity>,<activity-alias>,<application>,<category>,<data>,<grant-uri-permission>,<instrumentation>,<intent-filter>,<manifest>,<meta-

data>,<permission>,<permission-group>,<permission-tree>,<provider>,<receiver>,<service>,<support-screens>,<uses-configuration>,<uses-feature>,<uses-library>,<uses-permission>,<uses-sdk>

Elements for Application Properties:

- **uses_permission –** used to specify permissions that are requested for the purpose of security.
- **permission –** used to set permissions to provide access control for some specific component of the application.
- **permission-tree** – does the same as above for a set of components.
- **instrumentation** – enables to know interaction between Android system and application.
- **uses-sdk** – specifies the platform compatibility of the application.
- **uses-configuration** – specifies set of hardware and software requirement of the application.
- **uses-feature –** specifies single hardware and software requirement and their related entity.
- **support-screens, compatible-screens –** both these tags deals with screen configuration mode and size of the screen and etc.
- **support-gl-texture –** specifies texture based on which the application is filtered.

Elements for Application Components These should be enclosed in <application>container.

- **activity –** has the set of attribute based on user interface.
- **activity-alias –** specifies target activities.
- **service –** has the operation provided by any library or API, running in background that is not visible.
- **receiver –** that makes to receive message broadcasted by the same application or by outside entity.
- **provider –** provides some structure to access application data.
- **uses-library –** it specifies set of library files need to run the application.

Working with different types of resources

➢ Animation Resources

- Define pre-determined animations.
  Tween animations are saved in res/anim/ and accessed from the R.anim class.
  Frame animations are saved in res/drawable/ and accessed from the R.drawable class.

➢ Color State List Resource

- Define a color resources that changes based on the View state.
  Saved in res/color/ and accessed from the R.color class.

➢ Drawable Resources

- Define various graphics with bitmaps or XML.
  Saved in res/drawable/ and accessed from the R.drawable class.

➢ Layout Resource

- Define the layout for your application UI.
  Saved in res/layout/ and accessed from the R.layout class.

➢ Menu Resource

- Define the contents of your application menus.
  Saved in res/menu/ and accessed from the R.menu class.

➢ String Resources

- Define strings, string arrays, and plurals (and include string formatting and styling).
  Saved in res/values/ and accessed from the R.string, R.array, and R.plurals classes.

➢ Style Resource

- Define the look and format for UI elements.
  Saved in res/values/ and accessed from the R.style class.

➢ More Resource Types

- Define values such as booleans, integers, dimensions, colors, and other arrays.
  Saved in `res/values/` but each accessed from unique `R` sub-classes (such as `R.bool`, `R.integer`, `R.dimen`, etc.).

Managing Application resources

Directory	Resource Type
animation	XML files that define property animation.
anim/	XML files that define tween animation.
color/	XML files that define a state list of colors.
drawable/	Bitmap files or XML files that are compiled into the following drawable resource subtypes: • Bitmap files • Nine-Patches (re-sizable bitmaps) • States lists • Shapes • Animation drawables • Other drawables
layout/	XML files that define a user interface layout.
menu/	XML files that define application menus, such as an Option Menu, Context Menu, or Sub Menu.
raw/	Arbitrary files to save in their raw form. To open these resources with a raw InputStream.
values/	XML files that contain simple values, such as strings, integers and colors.
xml/	Arbitrary XML files that can be read at runtime by calling Resources.getXML (). Various XML configuration files must be saved here, such as a searchable

	configuration

➢ Configuration qualifier names

Configuration	Qualifier Values	Description
Layout Direction	ldrtl ldltr	The layout direction of your application. ldrtl means "layout-direction-right-to-left". Ldltr means "layout-direction-left-to-right" and is the default implicit value.
smallestWidth	sw<N>dp **Examples:** sw320dp sw600dp sw720dp etc.	The fundamental size of a screen, as indicated by the shortest dimension of the available screen area. Specifically, the device's smallestWidth is the shortest of the screen's available height and width. The device's smallestWidth does not change when the screen's orientation changes.
Available width	w<N>dp **Examples:** w720dp w1024dp etc.	Specifies a minimum available screen height, in "dp" units at which the resource should be used- defined by the <N> value. This configuration value will change when the orientation changes between landscape and portrait to match the current actual height.
Screen size	small normal large xlarge	**small:** Screens that are of similar size to a low-density QVGA screen. The minimum layout size for a small screen is approximately 320*426 dp units. **normal:** Screens that are of similar size to a medium-density HVGA screen. The minimum layout size for a normal screen is approximately 320*470 dp units. **large:** Screens that are of similar size to a medium-density VGA screen. The minimum layout size for a large screen is approximately 480*640 dp units. **xlarge:** Screens that are considerably larger than the traditional medium-density HVGA screen. The minimum layout size for a xlarge screen is approximately 720*960 dp units.
Screen aspect	Long notlong	**long:** Used for long screens. **notlong:**used for Notlong screens.
Screen Orientation	port land	**port:** Device is in portrait orientation (vertical). **land:** Device is in landscape orientation (horizonatal).

UI mode	car desk television appliance	**car:** Device is displaying in a car dock. **desk:** Device is displaying in a desk dock. **television:** Device is displaying on a television. **appliance:** Device is serving as an appliance, with no display.
Night mode	night notnight	**night:** Night time **notnight:** Day time
Scrren pixel density (dpi)	ldpi mdpi hdpi xhdpi nodpi tvdpi	**ldpi:** Low-density screens; approximately 120dpi. **mdpi:** Meduim-density screens; approximately 160dpi. **hdpi:** High-density screens; approximately 240dpi. **xhdpi:** Extra high-density screens; approximately 320dpi. **nodpi:** This can be used for bitmap resources that you do not want to be scaled to match the device density. **tvdpi:** Screens somewhere between mdpi and hdpi; approximately 213dpi.
Touchscreen type	notouch finger	**notouch:** Device does not have a touchscreen. **finger:** Device has a touchscreen that is intended to be used through direction interaction of the user's finger.
Keyboard availability	keysexposed keyshidden keyssoft	**keysexposed:**Device has a keyboard available. **keyshidden:** Device has a hardware keyboard available but it is hidden and the device does not have a software keyboard enabled. **keyssoft:** Device has a software keyboard enabled, whether it's visible or not.
Primary text input method	nokeys qwerty 12key	**nokeys:** Device has no hardware keys for text input. **qwerty:**Device has a hardware qwerty keyboard, whether it's visible to the user or not. **12key:**Device has a hardware 12-key keyboard, whether it's visible to the user or not.
Navigation key avaliability	navexposed navhidden	**navexposed:** Navigation keys are available to the user. **navhidden:** Navigation keys are not available to the user.
Primary non-touch tion method	nonav dpad trackball wheel	**nonav:** Device has no navigation facility other than using the touchscreen. **dpad:** Device has a directional-pad (d-pad) for navigation. **trackball:** Device has a trackball for navigation. **wheel:** Device has a directional wheel (s) for navigation.
Platform Version (API	**Example:**	The API level supported by the device.

	v3 v4 v7 etc	