Module - I

Introduction To Android (An)

* Android is a s/w package & linux based os for mob devices such as tablet comp & Smart Phns.
* Developed by Google & later the OHA. (Open Handset Alliance).
* Java lang is mainly used to write the android code evn though othr langs can be used.
* open source (source code is visible to public & can be edit)
* Anyone can customize the (An) system.
* There are a lot of mob appli~ that can be chosen by the consumer.
* It provides many interesting features like weathr details, opening screen, etc.
* categories of (An) appli~ :

- Entertainment → Net flix, Amazon prime
- Communi → whtsapp, telegram
- Productivity → google drive, proof kule
- personalization → wallpapers, Lauches
- Music & Audio → MX player, Youtube music
- social media & video → FB, Instragram.

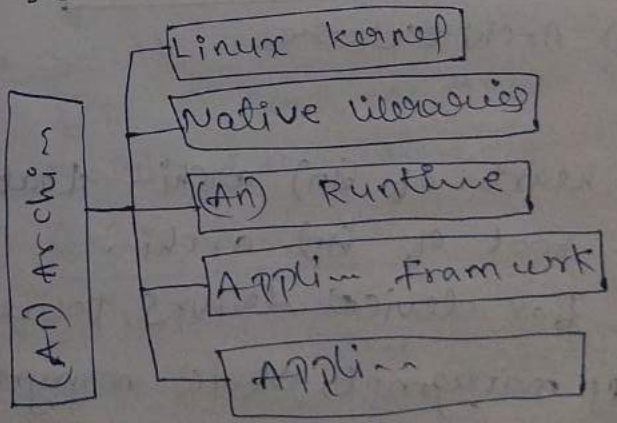
⇒ Android History :

* Initially Andy Rubin founded (An) incorporation in 2003.
* In 2005, google acquired (An) incorporation.
* The key employees of (An) incorporation are Andy Rubin, Rich Miner, Chris white & Nick Sears.
* In 2007, google announces the development of android OS.
* In 2008, HTC launched the 1st (An) mob.
* (An) is the nick name of Andy Rubin given by coworkers bcz of his love to robots

version	codname	Features
1.5	cupcake	• Soft keyboard • Advanced media recording (img, video) • widget, live folders
1.6	Donut	• Advanced search capability. • Text to Speech
2.1	Eclair	• Improved google maps
2.2	Froyo	• USB tethering • wifi hotspot
2.3	Gingerbread	installation of apps frm SD card.
4.0	icecream sandwitch	HD screen, unlock screen in diffrnt ways
4.4	kitkat	more sensors, GPS support.
10	—	spt for foldable phn.
11	—	new permission control.

⇒ (An) Architecture / (An) s/w stack :

It is categorized into 5 parts —



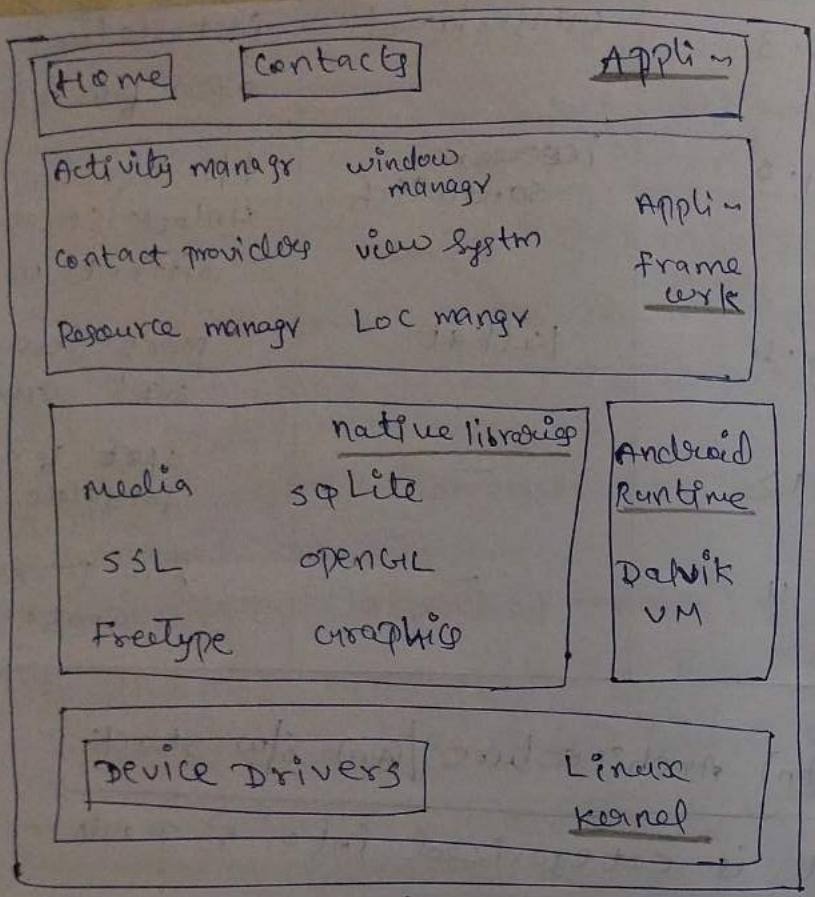


fig: (An) Architecture

1) Linux kernel:

It is the heart of (An) archi~ that exists at the root of (An) archi~

Responsible for device drivers, power managmnt, mfy managmnt, device managmnt & resource access.

2) Native libraries:

on the top of linux kernel, there are native libraries like webkit (responsible for browsr sprt), SQLite (for DB), Freetype (for font sprt), Media (for playing & recording audio & video)

3) (An) Runtime:

* Here, there are core libraries & DVM (Dalvik virtual machine) which is responsible to run (An) appli~
* DVM is like JVM bt it is optimized for mob devices
* It consumes less mfy & provides fast performance.
* Dalvik VM takes the generated Java cls files & combines them into 1/more Dalvik Executable (.dex) files.

4) (An) Framework:

* Includes (An) API's like UI, telephony, resources, loc, content providers, & pkge managrs.
* provides a lot of clses & interfaces for An) appli~ development.

5) Appli ~ :

* on the top of (An) framwork, there are appli.
* All appli like home, contact, Settings, games, browsers are using (An) framework that uses (An) runtime & libraries.
* (An) runtime & native libraries are ~~use~~ using Linux kernel.

⇒ (An) Java packages :

* android.app → implemnts appli model for (An).
* android.app.admin → provides device administran features at the Systm level
* android.accounts → provides clses to manage accounts like google, Fb.
* andriod.database → contain clses to explore data returned through a contnt provider.
* android.animation → to apply animations for diff obj; animation clses & methods are included.
* android.content → manages access to central ~~respira~~ repository of data.
* android.text → provides txt processing clses.

→ (An) Studio =

It is the official integrated develpmnt envrmnt (IDE) for (An) app development.
Based on intelliJ IDAE, a Java IDE for s/w.

⇒ Components of (An) / fundamtal compnts :

There are some necessary building blocks that an (An) appli consist of.

1) Anctivity :

- represnts a single screen with a user interface & generally contains 1 more views
- implemnted as a subcls of cls activity
- UI of our appli is built around 1 1 more extensions of activity cls.
- public cls MainActivity extends Activity { - - - } → general formula.
- user signup → eg.

2) service :

- background process that can run for a long time.
- 2 types → local & remote
  local service is accessed frm within the appli whereas remote service is

- accessed remotely frm othr appln running on same device
  - eg → music
  - public class ServiceName extends Service{..}

3) **view** :
  - UI element like btn, label, txtfield, etc
  - Anything that you see is a view.
  - ~~public class ViewName extends View{..}~~

4) **Intent** :
  - used to invoke components
  - used to start the service, launch an activity, display a web pg.
  - powerful intr-appln msg-passing framework.

5) **Contnt providers** :
  - used to manage & persist the appln data also typically interacts with SQL DB.
  - should be a sub-cls of cls contnt providers

    public cls c-p-Name extends c-p-{-
    public void onCreate() {}}.

6) **Fragments**:
  - are like parts of activity
  - An activity can display 1/more fragmnts on the screen at same time.

7) **AndroidManifest.xml** :
  - contains info about activities, contnt providers, permissions, etc
  - It is like web.xml file in Java EE

=> **(An) virtual device (AVD)** :

* It is an emulator configuration that allows developrs to test the appln by simulating the real device capabilities.
* we can configure the AVD by specifying the hardware & s/w options.
* AVD managr enables an easy way of creating & managing the AVD with its graphical intrface.
* Stps to create an AVD frm AVD managr graphical intrface —
- Go to window → AVD managr & select virtual devices.
- click on New to create a virtual device
- give it some name & select target (An) platform from the ~~...~~ dropdown list
- click ~~..~~ 'Create AVD' & we are done.

([illegible])

→ Str of (An) Applin :

(An) applin in eclipse (is any development tool hue a pre-defined Str with code & resource organized into a hon of folders.

Some components –

1) src – Stands for source code, contains Java source files.
2) gen – Stands for Generated Java library, This library is for (An) internal use only
3) assets → used to Store raw asset files
4) Android 4.2.2 – (An) framework library is Stored here.
5) libs – contains private libraries.
6) res – Stands for resource files.
7) AndroidManifest.xml – (An) definition file.
8) project.properties.
9) MainLayout.xml, etc.
10) Proguard-project.txt.

⇒ Applin lifecycle :

* It is controlled by 7 methods of android.app.Activity cls. (An) activity is the subcls of Context Themewrapper cls.
* The phases that an applin goes through frm Start to end → applin lifecycle
* It is a collection of activities & an activity corelates to a screen.
* Each (An) applin runs inside its own instance of a VM.
* Unlike a common windows / Linux (P), an (An) applin does not completely control the completion of its lifecycle.
* To manage limited system resources, the (An) System can terminate running applin
* If the (An) system needs to terminate (P)'s, it follows the follo Priority system-
  Foreground → visible → service →
  Background → Empty.
* All the (P)s in the empty list are added to a Least Recently used (LRU) list.

I States of an Activity :

1) Active state :

* whn an activity is in active State, it means it is active & running.
* It is visible to the user & the user is able to interact with it.
* Process State is foreground.

2) paused state:

* means that the user can still see the Activity in the background like behind a transparnt window (i.e) it is partially visilele.
* user cannot interact with the Activity until he/she is done with crnt view.
* (A) state is background.

3) Stopped state:

* whn a new Activity is started on top of the crnt (/whn a user hits the home key, the activity is brought to stopped state.
* The activity in this state is visilele bt it is not destroyed.
* (A) state is background.

4) Destroyed state:

* whn a user hits a (An) runtime decidy to reclaim the mly allocated to an activity. (i.e) in the paused/stoped state it goes into destroyed state
* The activity is out of the mly & it is invisilele to the user.

II Lifecycle Methods:

1) onCreate(): whn ever an (A) starts running the 1st method to gt executed is onCreate(). This method is executed only once during the lifetime.
After onCreate() method, the onStart() method is executed.

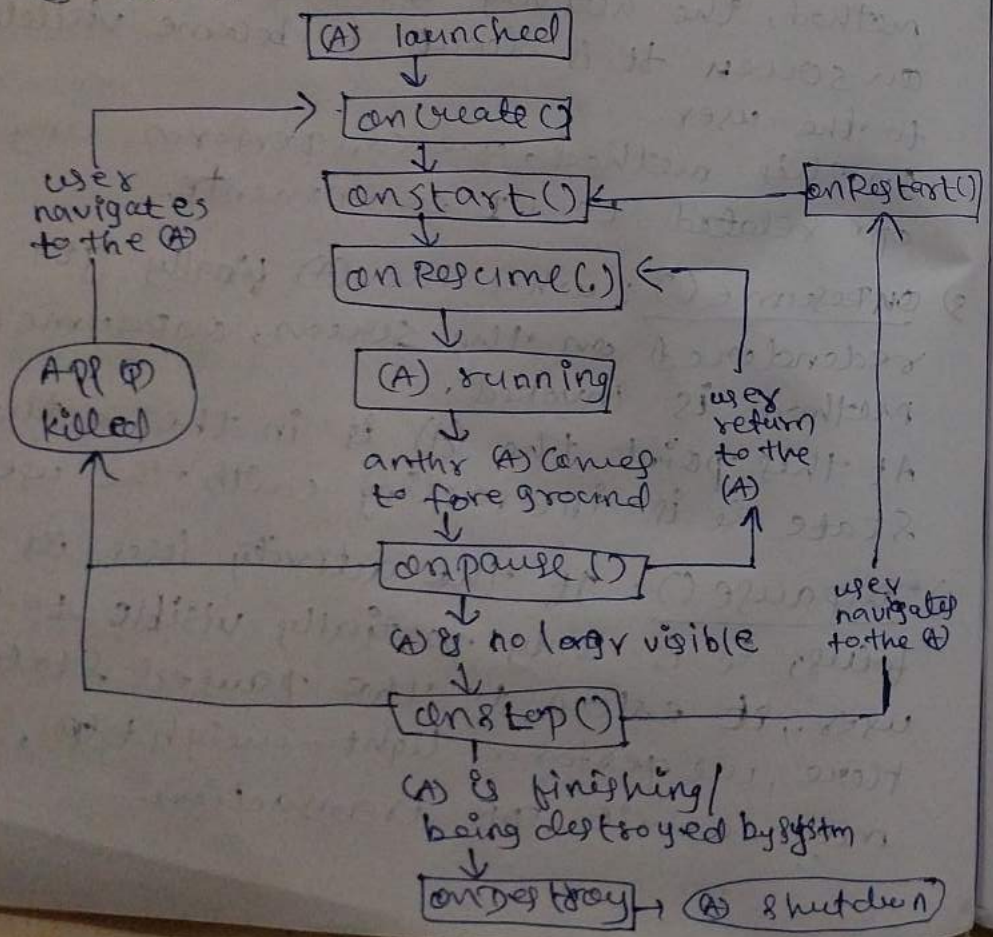
2) onStart(): During the execution of onStart() method, the activity is not yet rendered on screen bt is about to become visilele to the user.
In this method, we can perform any opr related to UI components.

3) onResume(): whn the (A) finally gts redendered on the screen, onResume() method is invoked.
At this point the (A) is in the active state & is interacting with the user.

4) onPause(): If the activity loses its focus, & is only partially visible to the user., it enters to the paused state.
Here, we perform light-weight (p), may commit DB transaction.

5) onStop() : From the active state, if we hit the home key, the (A) goes to the background & the homescreen of the device is made visible. During this evnt, the (A) enters to stopped state.

6) onDestroy() : whn an (A) is destroyed by a user / (An) system, onDestroy() is called.



→ Developing end user Appln :

1) (An) Emulator :

* (An) SDK includes a virtual mob device emulator that runs on the comp.
* Emulator lets you prototype, develop & test (An) appln without using a physical device.
* used to run, debug & tst applns

2) (An) UI :

* (An) uses a UI framework that resembles othr desktop-based, full featured UI framework, bt it is more modern & more asynchronous in ntre.
* UI is declarative & indepndly themed.
* prgming in the (An) UI involves declaring the interface in XML files.
* screen / windows in (An) are → activities