ANDROID O.S & DEVELOPMENT

Dr. Bhasha Anjaria
Assistant Professor
PIT, CSE Department

Lesson Plan

Subject/Course	**Mobile App Development**
Lesson Title	**Android Operating System and Environment Development**

Lesson Objectives
Introduction, Android Architecture
Android Development Tools
Directory Structure of Android Application
Android Manifest file

Outline

- Introduction
- Android Architecture
- Versions
- Features
- OHA
- Dalvik VM
- Android SDK
- Android Development Tools
- Android Virtual Devices
- Development Environment
- Directory Structure of Android Application
- Android Manifest file

Introduction

- Android is a Linux based mobile operating system.
- Initially Android was available for mobile devices only.
- Later on android is also available for tablets,ware, big screen(TV), automobile etc.
- Android was unveiled during 2007 along with the founding of the Open Handset Alliance

Introduction

- Android applications are usually developed in the Java language using the Android Software Development Kit.
- Once developed, Android applications can be packaged easily and sold out either through a store such as Google Play, SlideME, Opera Mobile Store, Mobango, F-droid and the Amazon Appstore.
- Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast.
- Every day more than 1 million new Android devices are activated worldwide.

Android Architecture

- Android OS is a software stack of different layers, where each layer is a group of several program components.
- Android has the following layers:

Applications

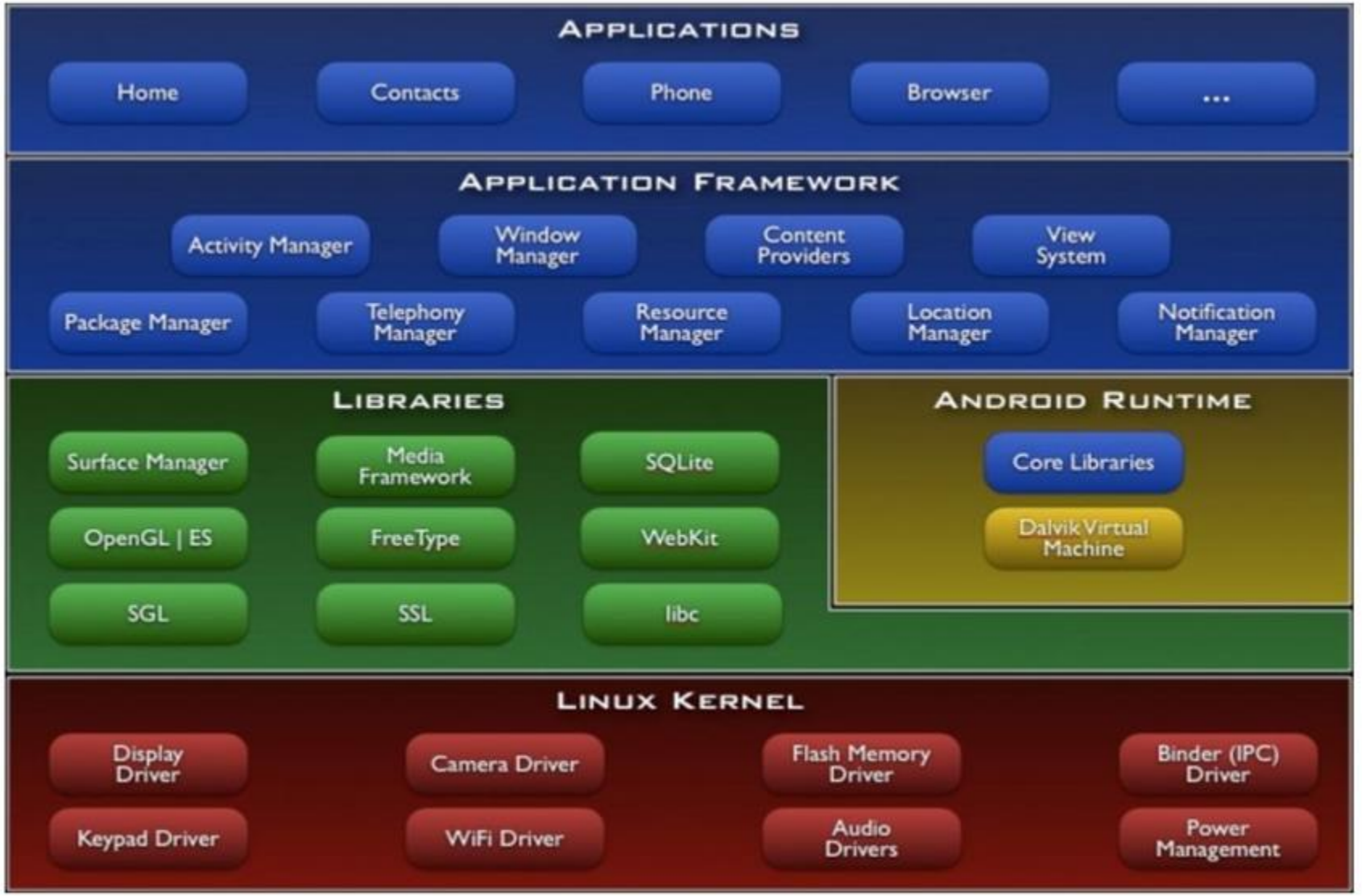
Application Framework

Libraries

Android Runtime

Linux Kernel

Android Architecture



Android Architecture

- **Applications :**
- Written using Java Language
  - Email Client
  - SMS Program
  - Maps
  - Browser
  - Calendar
  - Contacts
- Supports Parallel running
- No compulsory applications

Android Architecture

- **Application Framework :**
- The Application Framework layer 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 applications.

Android Architecture

- **Application Framework :**
- **The Android framework includes the following key services –**
- **Activity Manager – Controls all aspects of the application lifecycle and activity stack.**
- **Content Providers – Allows applications to publish and share data with other applications.**
- **Resource Manager – Provides access to non-code embedded resources such as strings, color settings and user interface layouts.**
- **Notifications Manager – Allows applications to display alerts and notifications to the user.**
- **View System – An extensible set of views used to create application user interfaces.**

Android Architecture

- **Library:**
- Useful to develop any third-party application.
- Native libraries are written in a language that compiles to native code for the platform it run.

Android Architecture

- **Library:**

1. SQLite - Responsible for Database Operation
2. Free Type - Font Support
3. Media F/W - Audio,Video Format
4. Open GL(Open Google Library) - 2D,3D Graphics Support
5. SSL - Encrypted Communication between Client and Server
6. SGL (Scalable Graphics Libraries)- For Basic Graphics Support

Android Architecture

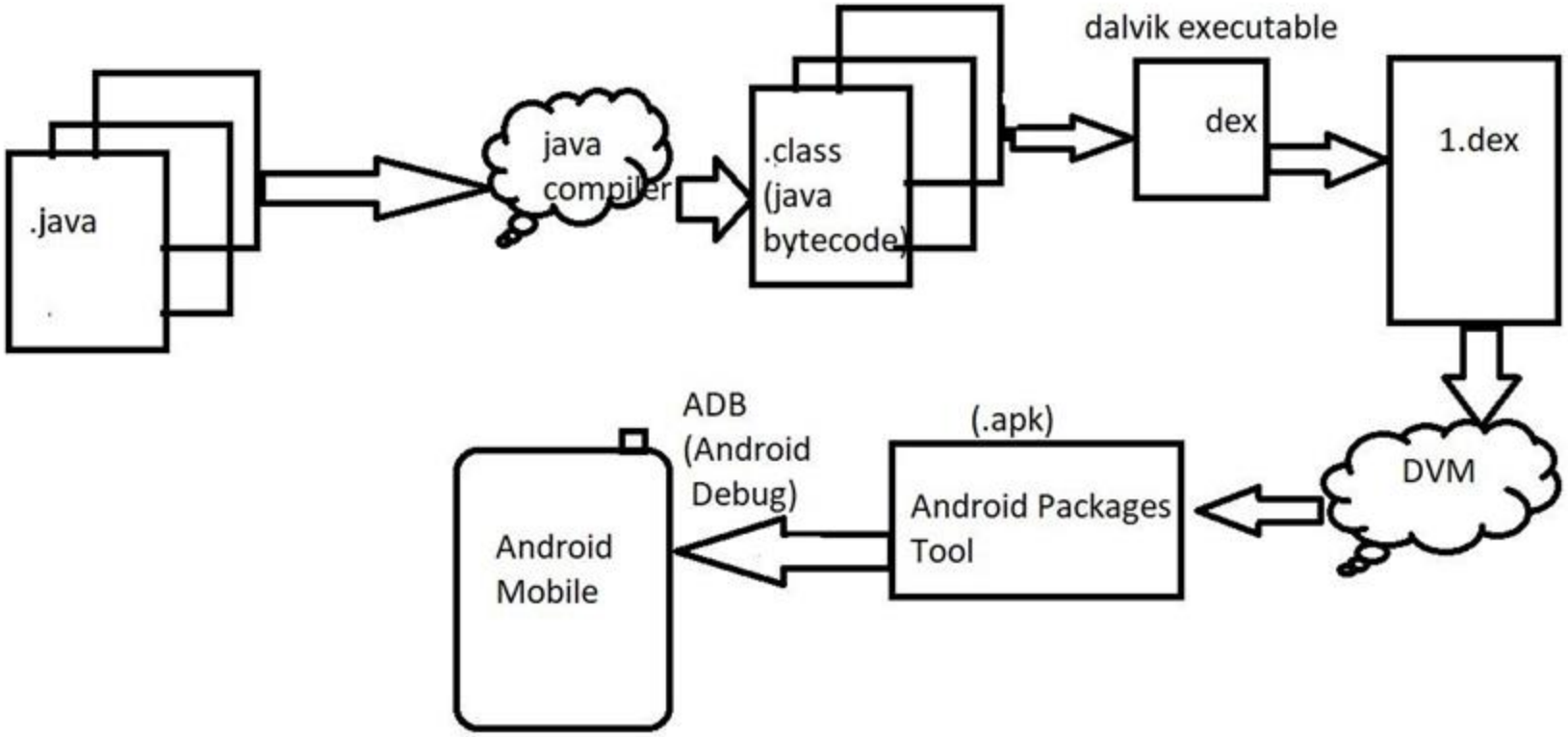
- **Android Runtime:**

- Android Runtime consists of Core Libraries and Dalvik Virtual Machine.
- Core Libraries are written in C/C++ languages. This libraries are helpful for runtime environment. Some of the core libraries are Data Structure, File Access, Network Access, Utilities and Graphics.

Android Architecture

- **Linux Kernal:**
- Linux Kernel is a Root layer of android Architecture which is responsible for device drivers(display, camera, Bluetooth, flash, web driver, usb, keypad,wifi etc.)
- power management, memory management and resource management.

Android Architecture (How Android works)



Versions

Version	Nickname	Release Date
1.0	Android	September 23,2008
1.1	Beta Android	February 9,2009
1.5	Cupcake	April 27,2009
1.6	Donut	September 15,2009
Android 1.x mobiles only		
Android 2.0/2.1	Éclair	October 26,2009
2.2 – 2.2.3	Froyo	May 20,2009
2.3 – 2.3.7	Gingerbred	December 6,2010

- Android 2.x is also designed for mobiles but from 2.x android started supporting API.
- - By using Google API android application can interact with Google products such as Gmail, YouTube, google maps,
- navigation, google search engine, google Clouds

Versions

Version	Nickname	Release Date
Android 3.0 – 3.2.6	Honey Comb	February 22,2011
Android 3.x is specially designed for tablets. Started supporting Fragments		
Android 4.0 – 4.0.4	Ice-cream Sandwich	October 18,2011
4.1 – 4.3.1	Jellybean	July 9,2012
4.4 – 4.4.4	KitKat	October 31,2013
4.4.w supports for wearable devices like wrist-watch. From 4.x android started supporting mobiles & tablets application. It means single app can run in mobiles & tablets		

Versions

Version	Nickname	Release Date
Android 3.0 – 3.2.6	Honey Comb	February 22,2011
Android 3.x is specially designed for tablets. Started supporting Fragments		
Android 4.0 – 4.0.4	Ice-cream Sandwich	October 18,2011
4.1 – 4.3.1	Jellybean	July 9,2012
4.4 – 4.4.4	KitKat	October 31,2013
4.4.w supports for wearable devices like wrist-watch. From 4.x android started supporting mobiles & tablets application. It means single app can run in mobiles & tablets		

Versions

Version	Nickname	Release Date
5.0 – 5.1.1	Lollypop	November 12,2014
Android 5.x designed for Big Screens ie. TV		
Android 6.0 – 6.0.1	Marshmallow	October 5,2015
Android 6.0 designed for Automobiles(speed of car, km)		
Android 7.0 – 7.1.2	Nougat	August 22,2016
8.0 – 8.1	Oreo	August 21,2017
9.0	Pie	August 6,2018
10.0		September 3,2019
11.0		September 8,2020

Versions


Cupcake
1.5
Donut
1.6
Eclair
2.0/2.1
Froyo
2.2
Gingerbread
2.3
Honeycomb
3.0/3.1
Ice Cream Sandwich
4.0
Jelly Bean
4.1/4.2/4.3
KitKat
4.4
Lollipop
5.0
Marshmallow
6.0
Nougat
7.0
Oreo
8.0
Pie
9.0
android
10
android

Android 1.0 to 1.1(September 23, 2008)

Features:

- Google Maps.
- Camera.
- Gmail, Contacts, and Google Synchronization.
- Web Browser.
- Wireless supports – Wi-Fi and Bluetooth.

Beta 1.1(February 9, 2009)

- released on February 9, 2009.
- Features:
  - Add Save attachment in the message.
  - Provides reviews and details when the user search business on maps.

Android version 1.5: Cupcake April 30, 2009

Features:

- New upload service on YouTube and Picasa like Uploading Videos and Photos.
- Supporting in MPEG-4, Video recording.
- Improving Web Browser-Copy and Paste facility.

Android version 1.6: Donut September 15, 2009

Features:

- The main enhancement was a Power Control widget for managing Wi-Fi, Bluetooth, GPS, etc.
- Provided Gallery and Camera features with quick toggling features.
- Improve the speed in system apps.
- Introduction of the Quick Search Box.

Android versions 2.0 to 2.1: Éclair December 3, 2009

Features:

- Update UI.
- Support Live Wallpaper.
- Support Bluetooth 2.1.
- Improve Google map.
- Minor API Changes.

Android version 2.2: Froyo May 20, 2010

Features:

- Support for Animated GIF.
- Wi-Fi Support Hotspot functionality.
- Speed improvements.
- Upload file support in the browser.
- Support numeric and alphanumeric passwords.

Android version 2.3: Gingerbread December 6 2010

Features:

- Improve Copy-Paste Facility.
- Updated UI design.
- VP8 and WebM video format support.
- Social Networking Supports.
- Easy use of the keyboard.
- Multiple camera support (usually known as a selfie camera nowadays).

Android 3.0/3.1/3.2 – Honeycomb

Features:

- Gmail App improvements.
- Updated 3D UI.
- Supports multiprocessors and hardware acceleration for graphics.
- Media Sync from SD Card.
- Google eBooks.
- Google Talk Video Chat.
- Support Adobe Flash in Browser.
- High-performance Wi-Fi Connections and Lock.
- Chinese handwriting.

Android version 4.0: Ice Cream Sandwich October 19, 2011

Features:

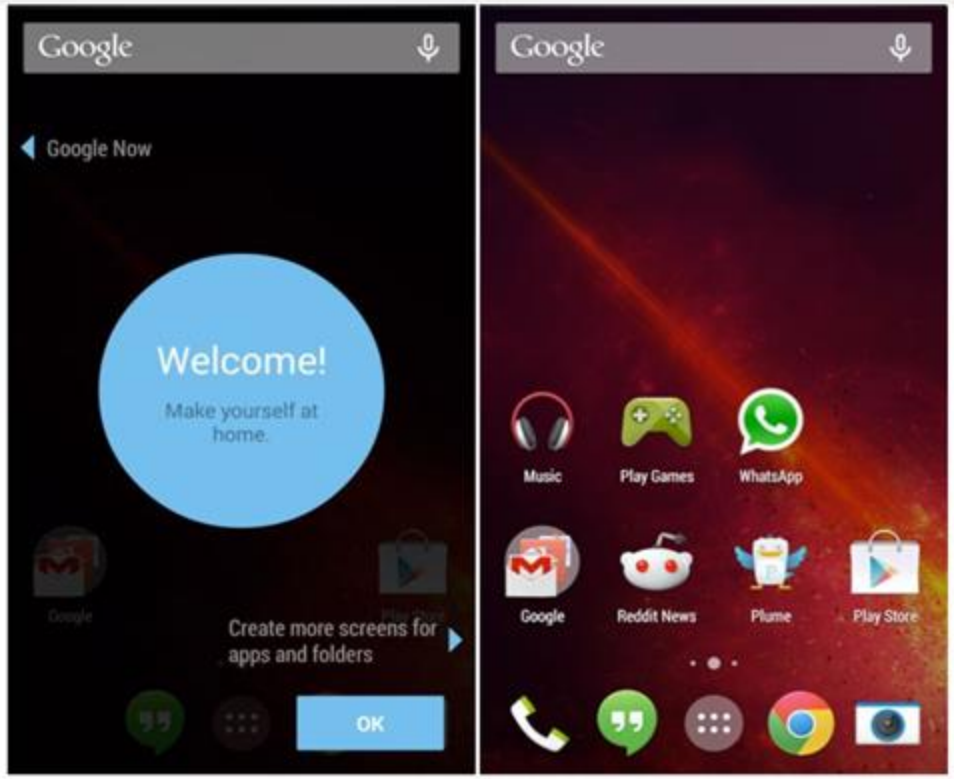
- Improved text input and spelling check.
- Wi-Fi direct (Sharing information using NFC).
- Photo Decor facility.
- Improved keyboard correction.
- Unlocking with face-fixing.
- Improved video recording resolution.
- Camera performance.
- Up to 16 tabs in the web browser.

Android versions 4.1 to 4.3: Jelly Bean

Features:

- Voice search.
- Panorama.
- Daydream as a screensaver.
- Power control.
- Improve camera application.
- Security enhancement.
- Voice typing.
- Multiple user accounts on tablets only.
- 4k resolution support.
- Supporting Bluetooth Low Energy.
- Bi-directional text and other language support.
- Support USB audio.
- Set the volume of incoming calls and show a message alert.
- Native emoji support.

Android version 4.4: KitKat September 3, 2013.



Features:

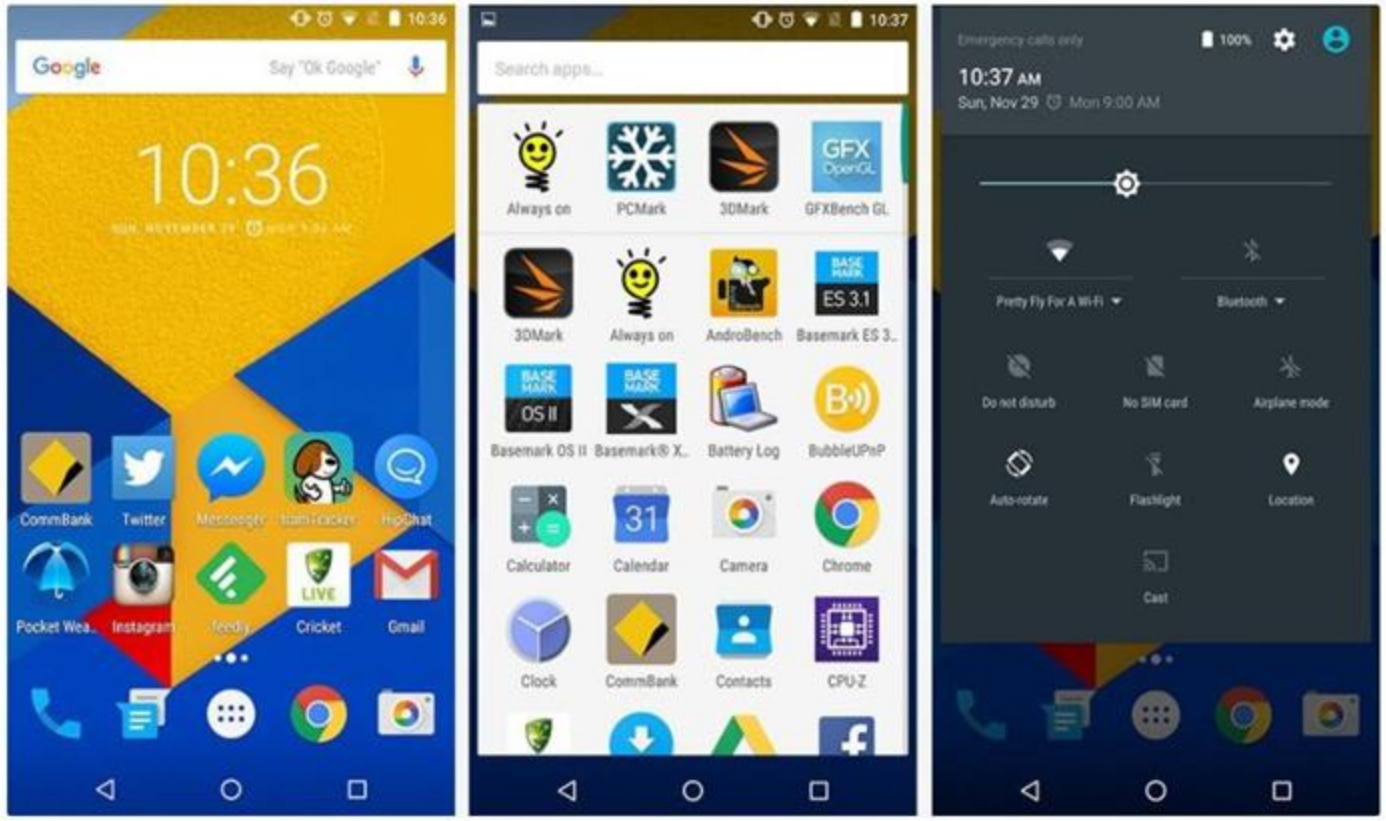
- Screen Recording.
- KitKat adds a feature in 'Google now'. Its name is 'OK Google'. "OK, Google" allows access to Google to the users without touching your mobile phone.
- GPS Support.
- Offline music support.
- UI updates for google map navigation and alarm.
- Introduction of 'Emoji' to the google keyboard.

Android versions 5.0 and 5.1: Lollipop November 12, 2014

Features:

- Support ART(Android Runtime).
- Improvement in UI.
- New material design.
- Notifications on the Lock screen.
- Revamped navigation bar.
- Multiple sim card support.
- High definition voice call.

Android version 6.0: Marshmallow



Features:

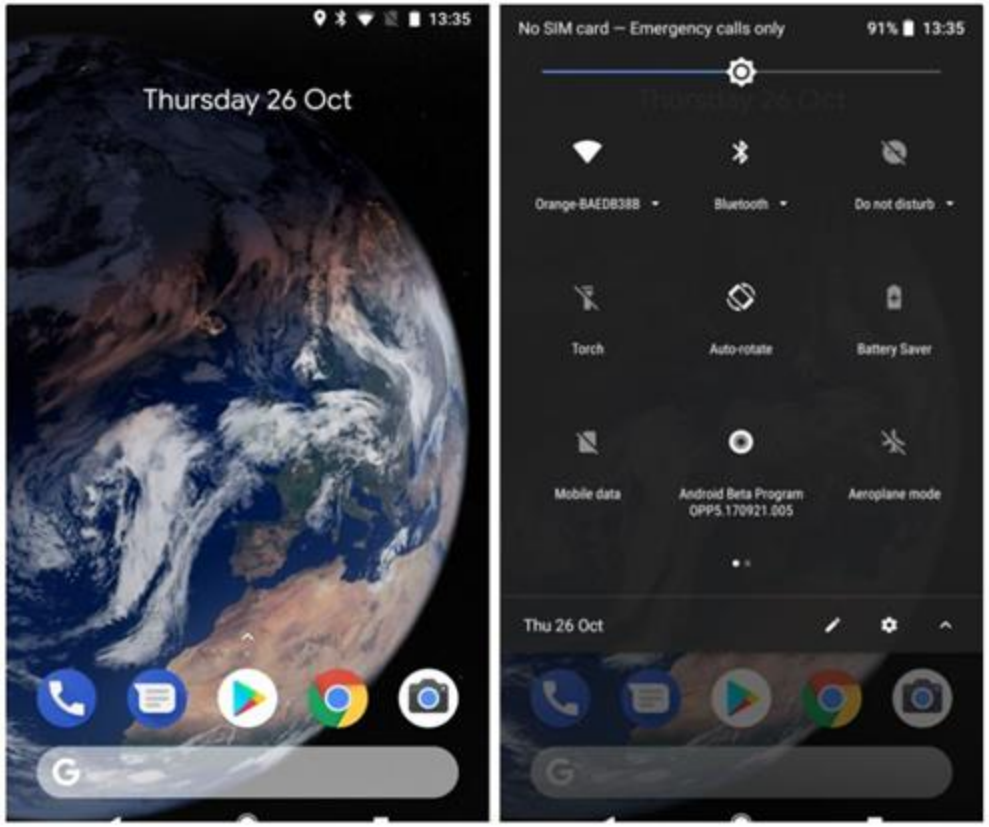
- Fingerprint authentication to unlock the screen.
- USB Type C support.
- Multi-window experiments (user can use two different apps in one screen).
- Save battery-'Sleep Mode'.
- App permission model-OPT(send a request for permission).

Android versions 7.0 and 7.1: Nougat March 2016

Features:

- Provide multitasking.
- Inline reply to messages and notifications so you won't have to open up your Messenger application for quick replies.
- Providing multi-window mode.
- Improvements in storage manager.
- Display touch improvement.

Android version 8.0 and 8.1: Oreo



Features:

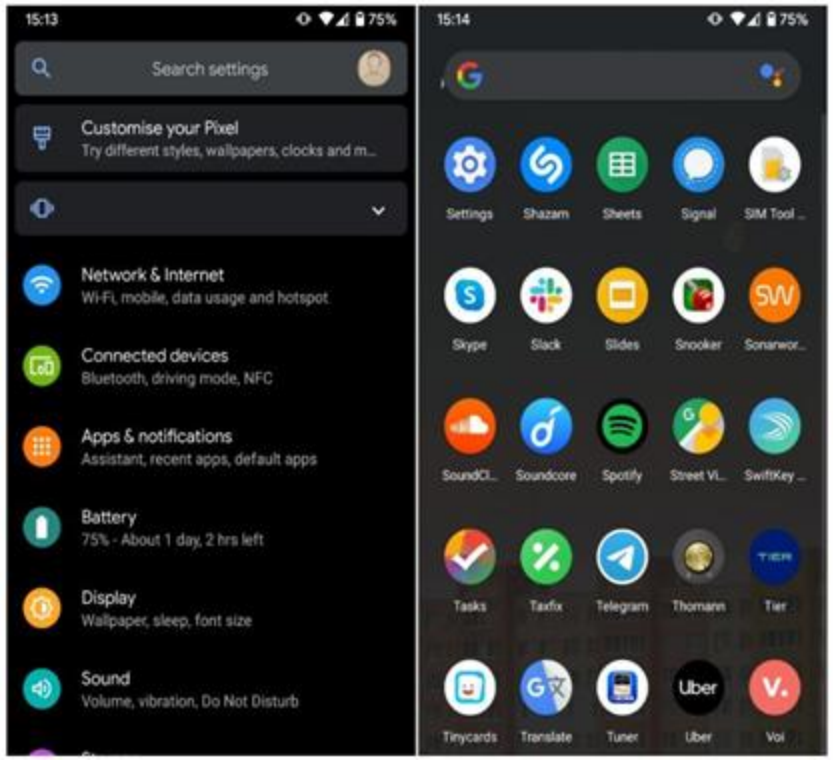
- Support PIP(Picture-in-Picture).
- Multi-display support.
- Google Play support.
- Adaptive icons.
- Revamped notification section(Users can set which notifications you want to show).

Android version 9: Pie August 2018.

Features:

- New Gesture Navigation.
- Artificial intelligence (AI) Compatible.
- Adaptive Battery and Brightness.
- App Actions.
- New Screenshot Shortcuts.
- Easier Screen Rotation.
- Volume and Sound Improvement.
- Selectable Dark Mode.
- Slices.
- Improved Security Features.
- Digital Wellbeing.
- New Accessibility Menu.
- Easier Text Selection.
- More Notification Information.

Android version 10: Android Q September 3, 2019



Features:

- Support for the upcoming foldable smartphones with flexible displays which is an upcoming rush.
- System-wide dark mode.
- Navigation control over gesture.
- Smart reply for all messaging apps.
- Support for Live caption.
- Better notification control.

Android version 11 (Developer Preview) preview on February 19 of year 2021.

- **Features (announced):**
  - New Support for 5G.
  - Privacy and Security; A new privacy choice for apps is the “Only This Time” option when you’re allowing the app access to your location, microphone, or camera.
  - Support new screen types (pinhole and waterfall).
  - Low Latency Options; adds low latency support in new MediaCodec APIs and HDMI which is very useful for use on external displays and TVs.

Features

- 1. Open Source: the original source code of android is available for modifications.
- 2. Android OS is device independent.
- 3. Android OS supports NFC(Near Field Communication)– Transfer any amount of data by simply touching the two devices.– Maintain less than 1 cm distance in between two devices.
- 4. Android OS supports IPC(Inter Process Communication)
- 5. By default Android OS is available with SQLite

Features

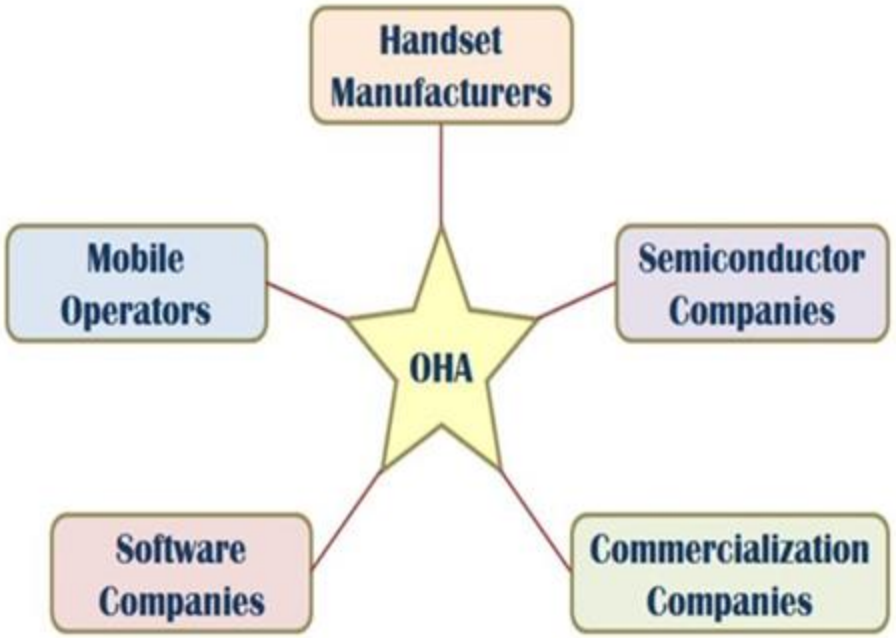
- 6. Android OS supports for SSL(Secure Socket Layer)– TLS(Transport Layer Security)– In SSL client & server interaction done in encrypted mode.
- 7. Android OS will support Text to Speech and Speech to Text conversion.
- 8. Android device can be controlled through voice commands.
- 9. Android supports all types of images, audio, video,different types of languages, different types of fonts.
- 10. Android OS supports basic graphics and 2-D,3-D animation.

Features

- 11. Supports Video calling.
- 12. External storage– Most Android devices include microSD slot and can read microSD cards formatted with FAT32,Ext3 or Ext4 file system

OHA

- Open Handset Alliance(OHA) was formed in November 2007.
- The OHA is the group that is in charge of the Android Smartphone Operating System. It was created by Google.
- The OHA is a business alliance that consists of 47 companies for developing open standard platform for mobile devices.



Dalvik VM

- Runtime environment for running android application.
- JVM is used to run high-end applications while DVM is used for small-end applications.
- DVM was first written by "Dan Bornstein“
- Unlike JVM, the DVM does not run .class files but it runs .dex files.
- .dex files are built from .class file at the time of compilation and provide higher efficiency in row resource environments.

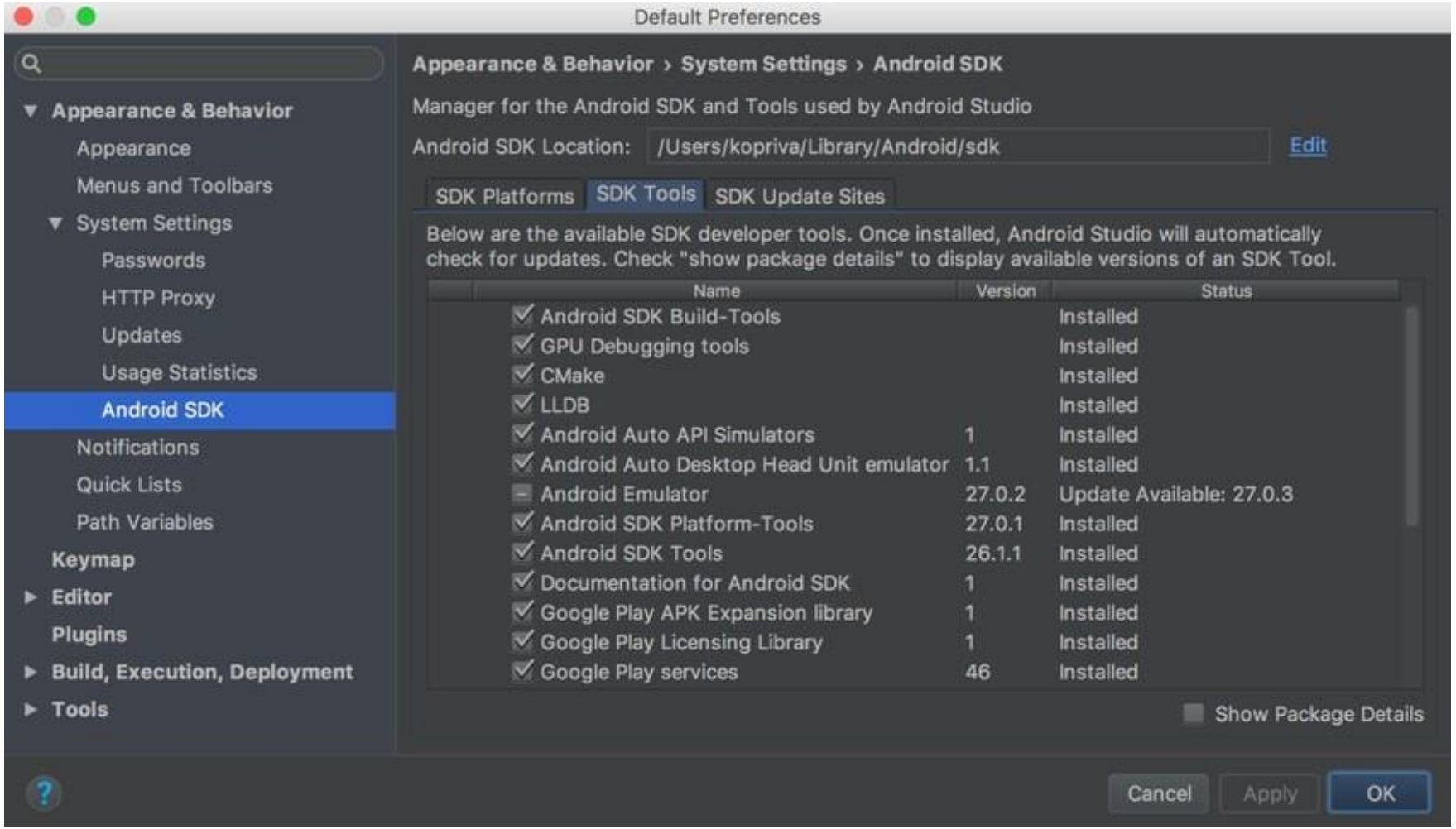
Android SDK

- Android SDK is a software development kit developed by Google for the Android platform.
- Android SDK comes bundled with Android Studio, Google's official integrated development environment (IDE) for the Android operating system

Android SDK

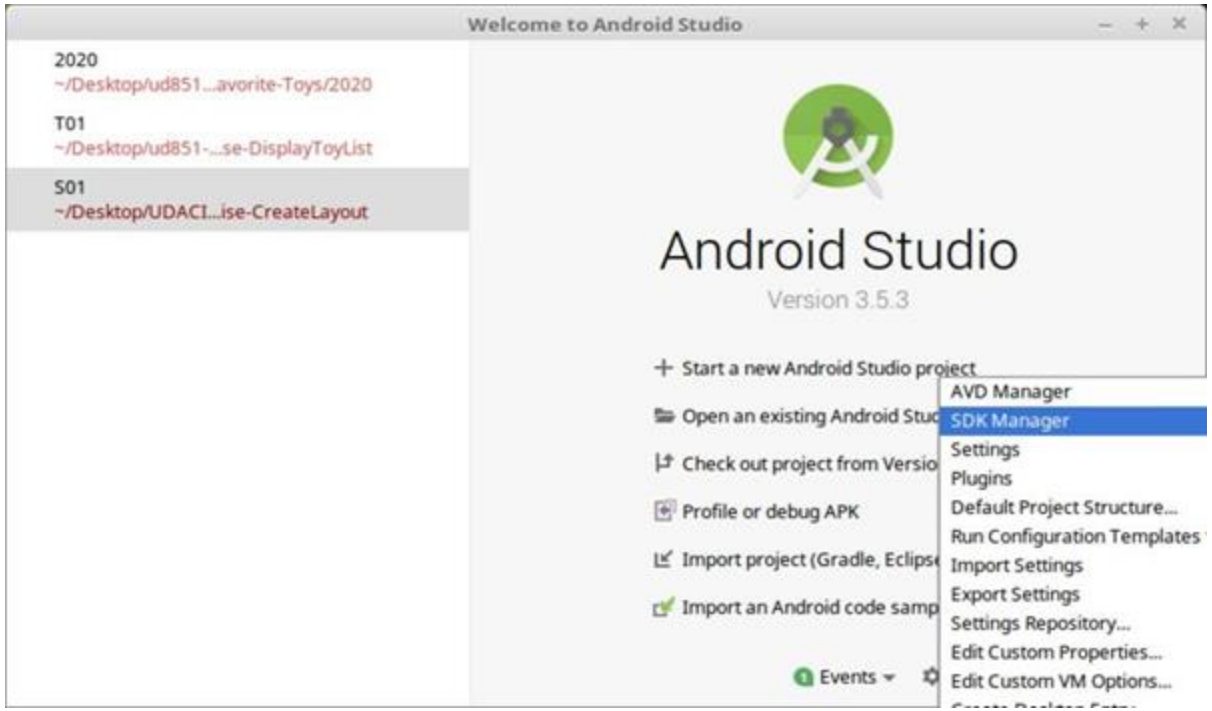
- The Android SDK is a collection of software development tools and libraries required to develop Android applications.
- Every time Google releases a new version of Android or an update, a corresponding SDK is also released which developers must download and install.
- The Android SDK comprises all the tools necessary to code programs from scratch and even test them. These tools provide a smooth flow of the development process from developing and debugging, through to packaging.
- The Android SDK is compatible with Windows, macOS, and Linux, so you can develop on any of those platforms.

Android SDK


Default Preferences
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
Passwords
HTTP Proxy
Updates
Usage Statistics
Android SDK
Notifications
Quick Lists
Path Variables
Keymap
Editor
Plugins
Build, Execution, Deployment
Tools
Appearance & Behavior › System Settings › Android SDK
Manager for the Android SDK and Tools used by Android Studio
Android SDK Location: /Users/kopriva/Library/Android/sdk
Edit
SDK Platforms
SDK Tools
SDK Update Sites
Below are the available SDK developer tools. Once installed, Android Studio will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.
Name
Version
Status
Android SDK Build-Tools Installed
GPU Debugging tools Installed
CMake Installed
LLDB Installed
Android Auto API Simulators 1 Installed
Android Auto Desktop Head Unit emulator 1.1 Installed
Android Emulator 27.0.2 Update Available: 27.0.3
Android SDK Platform-Tools 27.0.1 Installed
Android SDK Tools 26.1.1 Installed
Documentation for Android SDK 1 Installed
Google Play APK Expansion library 1 Installed
Google Play Licensing Library 1 Installed
Google Play services 46 Installed
Show Package Details
Cancel
Apply
OK

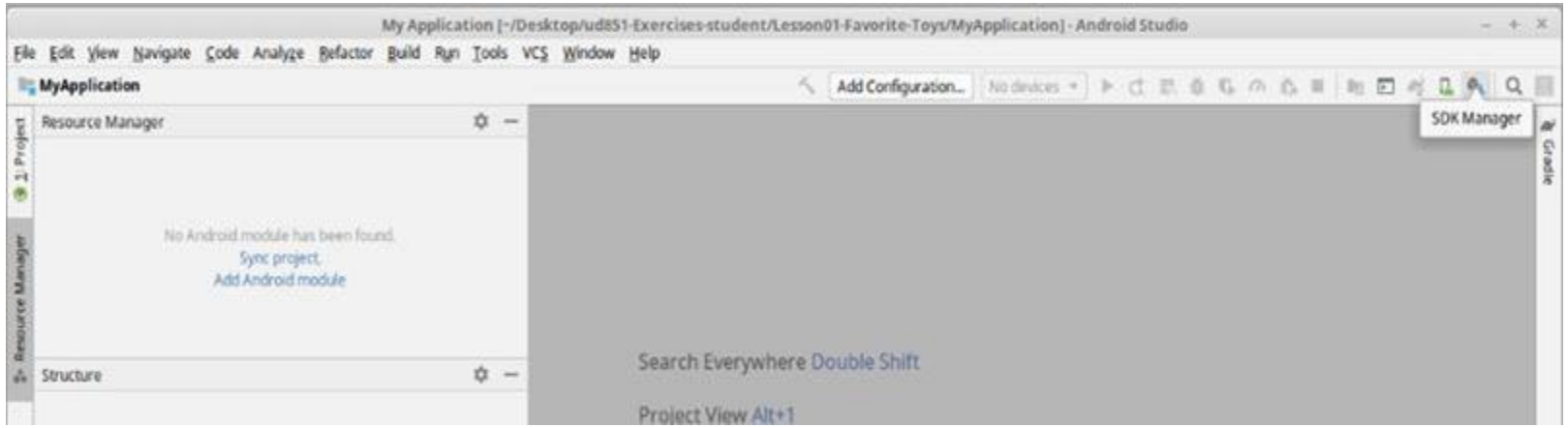
Android SDK

- To install the Android SDK from within Android Studio, first start Android Studio.
- From the Android Studio start page, select **Configure > SDK Manager.**



Android SDK

- If you already have Android Studio open, the SDK Manager icon is found on the top right corner, as shown below.



Android SDK

- The Android SDK consists of an emulator, development tools, sample projects with source code, Google API, and the required libraries to build Android applications

Android SDK

- Provided in the android.jar file that made up of several important packages.

Top-Level Package Name	Description
android.*	Android application fundamentals
dalvik.*	Dalvik Virtual machine support fundamentals
java.*	Core java classes and generic utilities for networking, security, math and so on
javax.*	Encryption support
junit.*	Unit-Testing support
org.apache.http.*	HTTP Protocol support
org.json	JavaScript Object Notation Support

Android SDK

Few Popular Third-Party Android APIs:

- Com.google.android.gms.ads.*= Google Mobile Ads SDK Package.
- Com.google.android.gms.analytics.* = Google Analytics SDK for Android Package.
- Com.google.android.gms.gcm = Google Cloud Messaging for Android.
- Com.google.android.gms.appindexing = Google App Indexing Package.
- Com.google.android.gms.appinvite = Google App Invites Package
- Com.google.android.gms.games = Google Play Games Services Package
- Com.google.android.gms.fitness = Google Fit Package

Android SDK

- Android SDK provides many tools to design, develop, debug and deploy your applications.
  - Android Studio
  - Android SDK and AVD Managers
  - Android Emulator

Android Virtual Devices

- An **Android Virtual Device (AVD)** is a configuration that defines the characteristics of an Android device you want to simulate in the Android Emulator. This includes the device's hardware profile, system image, storage area, skin, and other properties

Android Virtual Devices

- **Key Components of AVD**
- Hardware Profile: Defines the characteristics of a device as shipped from the factory. The Device Manager in Android Studio comes pre-loaded with certain hardware profiles, such as Pixel devices, and you can define or customize the hardware profiles as needed.
- System Image: A system image labeled with Google APIs includes access to Google Play services. The API level of the target device is crucial because your app doesn't run on a system image with an API level lower than the one required by your app[1].
- Storage Area: The AVD has a dedicated storage area on your development machine. It stores the device user data, such as installed apps and settings, as well as an emulated SD card[1].
- Skin: An emulator skin specifies the appearance of a device. The Device Manager provides some predefined skins, and you can also define your own or use skins provided by third parties

Android Virtual Devices

- **Creating and Managing AVDs**
- To create a new AVD, you can use the Device Manager in Android Studio. Here are the steps:
- Open the Device Manager.
- Click "Create Device."
- Select a hardware profile and click "Next."
- Select the system image for a particular API level and click "Next."
- Change the AVD properties as needed and click "Finish"

Android Virtual Devices

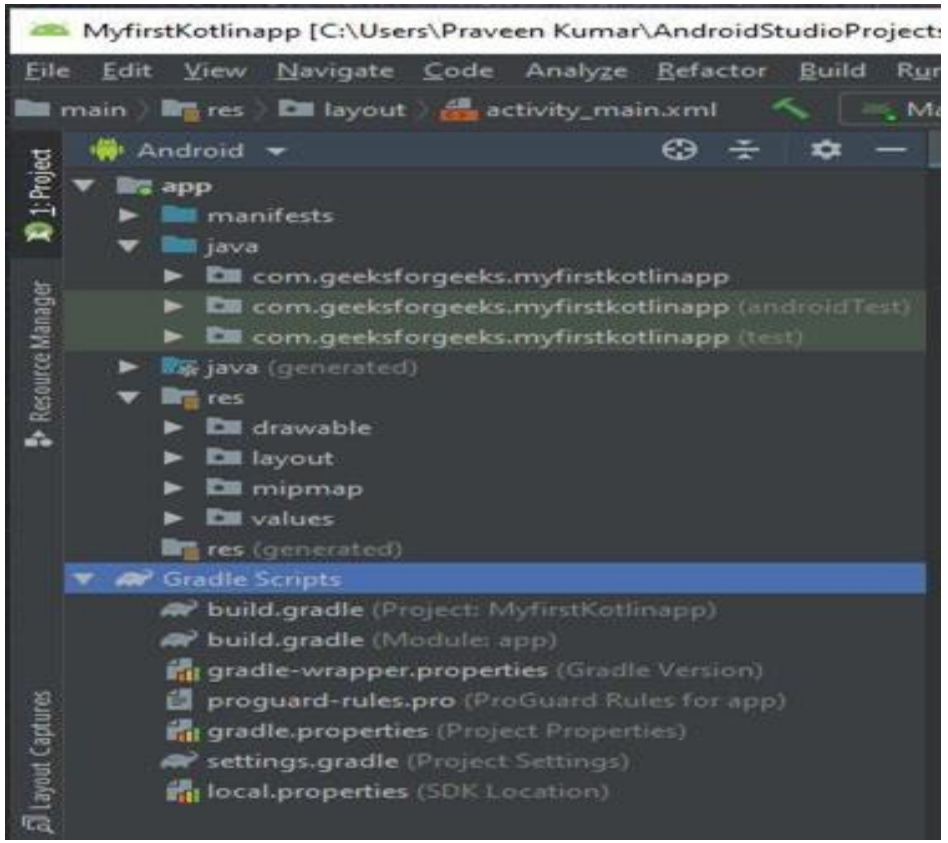
- **Advantages of Using AVD**
- **Testing and Debugging**: The Android emulator provides almost all the functionality of a real device, making it easier to test and debug applications without needing physical hardware.
- **Predefined Configurations**: The emulator comes with predefined configurations for several Android phones, Wear OS, tablet, and Android TV devices.
- **Faster Data Transfer**: Transferring data to the emulator is faster than to a real device connected through USB.

Android Virtual Devices

- **Limitations of AVD**
- **Performance**: Emulators can be slower compared to actual physical devices.
- **Accuracy**: Testing on an emulator is not as accurate as using a real device, especially for network and hardware-related activities.
- In summary, an AVD is a powerful tool for Android developers, allowing them to simulate various Android devices and test their applications in a controlled environment

Directory Structure of Android Application

- Android Studio is the official IDE (Integrated Development Environment) developed by the JetBrains community which is freely provided by Google for android app development. After completing the setup of Android Architecture we can create an android application in the studio. We need to create a new project for each sample application and we should understand the folder structure. It looks like this:



Directory Structure of Android Application

The android project contains different types of app modules, source code files, and resource files. We will explore all the folders and files in the android app.

Manifests Folder

Java Folder

res (Resources) Folder

- Drawable Folder
- Layout Folder
- Mipmap Folder
- Values Folder

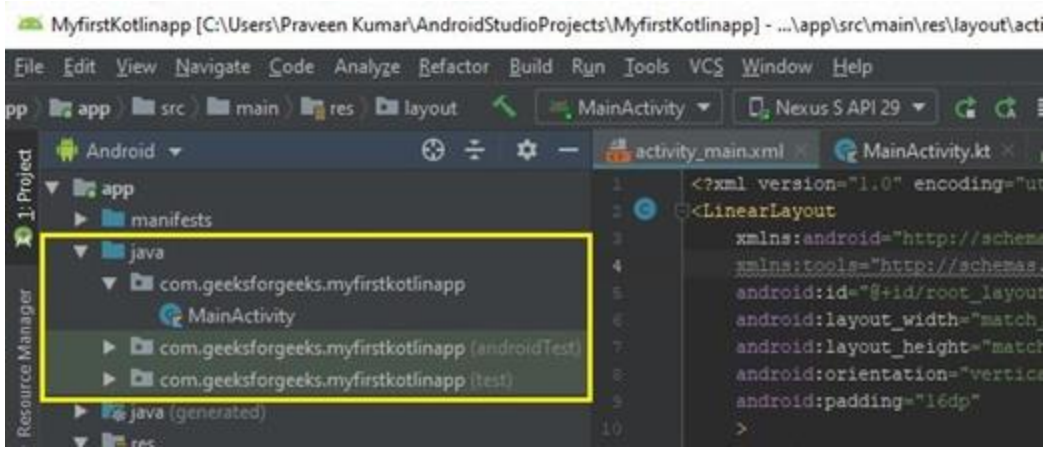
Gradle Scripts

Directory Structure of Android Application

- **Manifests Folder**
- Manifests folder contains **AndroidManifest.xml** for creating our android application. This file contains information about our application such as the Android version, metadata, states package for Kotlin file, and other application components. It acts as an intermediator between android OS and our application.

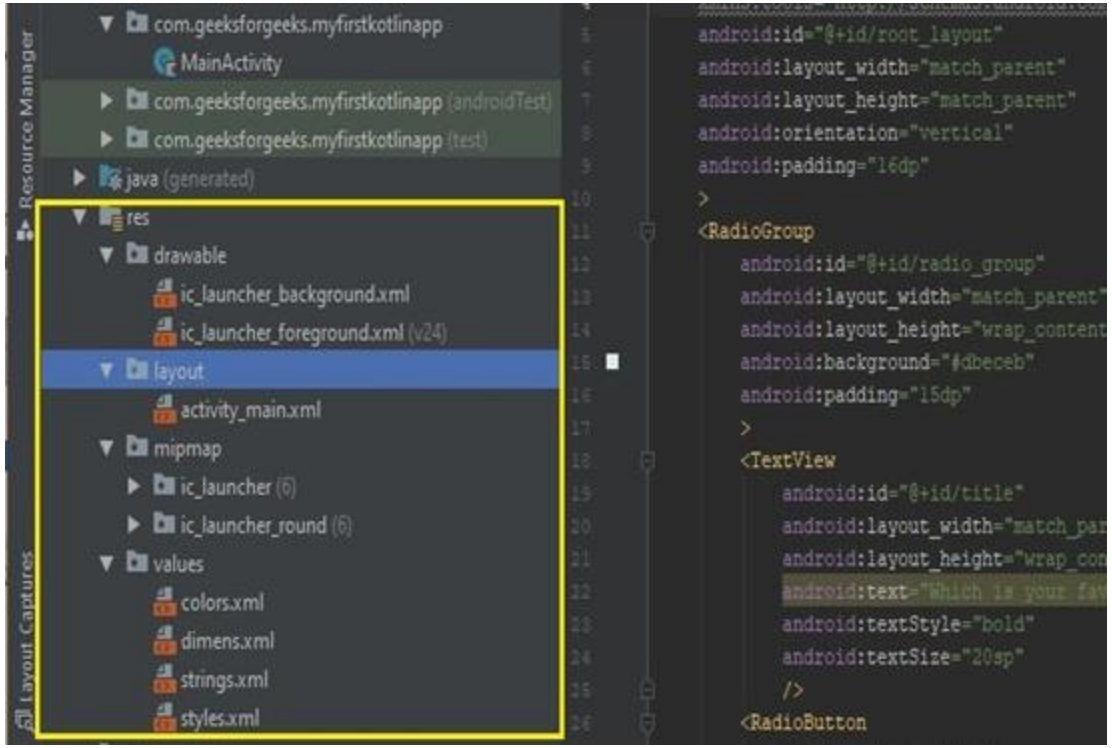
Directory Structure of Android Application

- **Java folder**
- The Java folder contains all the java and Kotlin source code (.java) files that we create during the app development, including other Test files. If we create any new project using Kotlin, by default the class file MainActivity.kt file will create automatically under the package name "com.geeksforgeeks.myfirstkotlinapp" as shown below.



Directory Structure of Android Application

- **Resource (res) folder**
- The resource folder is the most important folder because it contains all the non-code sources like images, XML layouts, and UI strings for our android application.



Directory Structure of Android Application

- **res/drawable folder**

  It contains the different types of images used for the development of the application. We need to add all the images in a drawable folder for the application development.

- **res/layout folder**

  The layout folder contains all XML layout files which we used to define the user interface of our application.

Directory Structure of Android Application

- **res/mipmap folder**

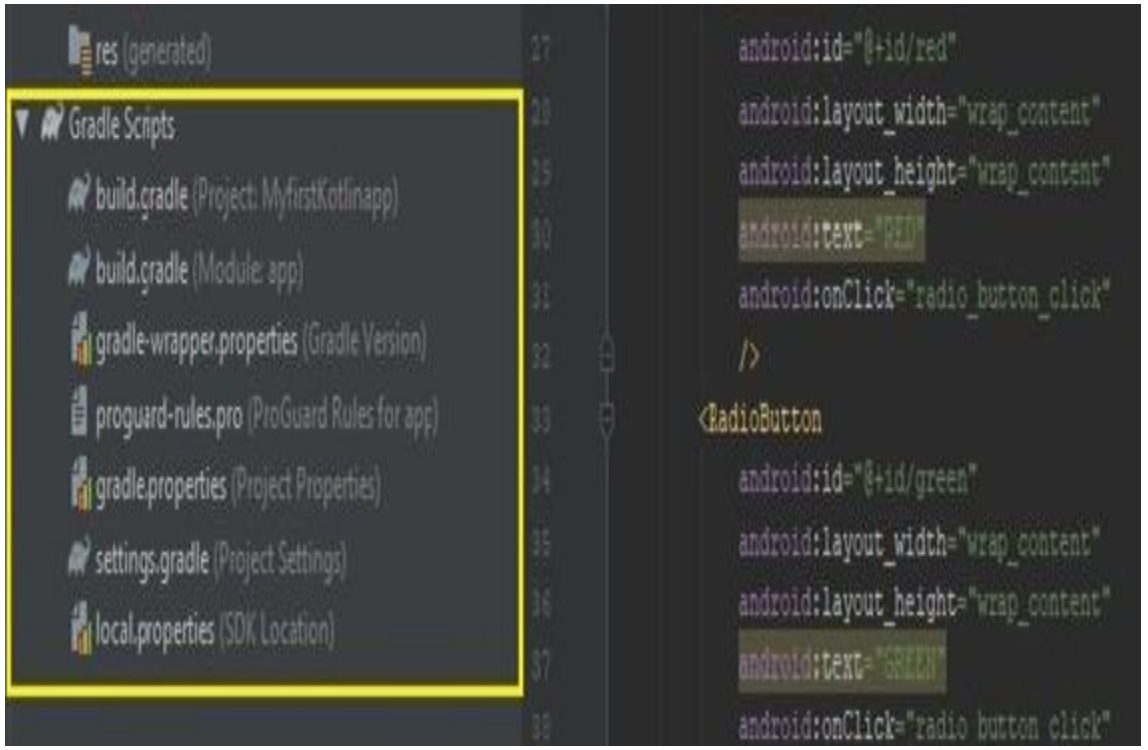
  This folder contains launcher.xml files to define icons that are used to show on the home screen. It contains different density types of icons depending upon the size of the device such as hdpi, mdpi, xhdpi.

- **res/values folder**

  Values folder contains a number of XML files like strings, dimensions, colors, and style definitions.

Directory Structure of Android Application

- **Gradle Scripts folder**
- Gradle means automated build system and it contains a number of files that are used to define a build configuration that can be applied to all modules in our application. In build.gradle (Project) there are buildscripts and in build.gradle (Module) plugins and implementations are used to build configurations that can be applied to all our application modules.



Android Manifest file

- Every project in Android includes a Manifest XML file, which is **AndroidManifest.xml**, located in the root directory of its project hierarchy. The manifest file is an important part of our app because it defines the structure and metadata of our application, its components, and its requirements.
- This file includes nodes for each of the **Activities**, **Services**, **Content Providers**, and **Broadcast Receivers** that make the application, and using **Intent Filters** and Permissions determines how they coordinate with each other and other applications.

Android Manifest file

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.geeksforgeeks"
    android:versionCode="1"
    android:versionName="1.0"
    android:installLocation="preferExternal">

      <uses-sdk
        android:minSdkVersion="18"
        android:targetSdkVersion="27" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

Android Manifest file

- **1. manifest**
- The main component of the AndroidManifest.xml file is known as manifest. Additionally, the packaging field describes the activity class package name. It must contain an <application> element with the xmlns:android and package attribute specified.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.geeksforgeeks">

    <!-- manifest nodes -->

    <application>

    </application>

</manifest>
```

Android Manifest file

- **2. uses-sdk**
- It is used to define a minimum and maximum SDK version by means of an API Level integer that must be available on a device so that our application functions properly, and the target SDK for which it has been designed using a combination of minSdkVersion, maxSdkVersion, and targetSdkVersion attributes, respectively. It is contained within the <manifest> element.

```
<uses-sdk
    android:minSdkVersion="18"
    android:targetSdkVersion="27" />
```

Android Manifest file

- **3. uses-permission**
- It outlines a system permission that must be granted by the user for the app to function properly and is contained within the <manifest> element. When an application is installed (on Android 5.1 and lower devices or Android 6.0 and higher), the user must grant the application permissions.

```
<uses-permission
    android:name="android.permission.CAMERA"
    android:maxSdkVersion="18" />
```

Android Manifest file

- **4. application**
- A manifest can contain only one application node. It uses attributes to specify the metadata for your application (including its title, icon, and theme). During development, we should include a debuggable attribute set to true to enable debugging, then be sure to disable it for your release builds. The application node also acts as a container for the Activity, Service, Content Provider, and Broadcast Receiver nodes that specify the application components. The name of our custom application class can be specified using the android:name attribute.

```
<application
    android:name=".GeeksForGeeks"
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@drawable/gfgIcon"
    android:label="@string/app_name"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@android:style/Theme.Light"
    android:debuggable="true"
    tools:targetApi="31">

    <!-- application nodes -->

</application>
```

Android Manifest file

- **5. uses-library**
- It defines a shared library against which the application must be linked. This element instructs the system to add the library's code to the package's class loader. It is contained within the <application> element.

```
<uses-library
    android:name="android.test.runner"
    android:required="true" />
```

Android Manifest file

- **6. activity**
- The Activity sub-element of an application refers to an activity that needs to be specified in the AndroidManifest.xml file. It has various characteristics, like label, name, theme, launchMode, and others. In the manifest file, all elements must be represented by <activity>. Any activity that is not declared there won't run and won't be visible to the system. It is contained within the <application> element.

```
<activity
    android:name=".MainActivity"
    android:exported="true">
</activity>
```

Android Manifest file

- **7. intent-filter**
- It is the sub-element of activity that specifies the type of intent to which the activity, service, or broadcast receiver can send a response. It allows the component to receive intents of a certain type while filtering out those that are not useful for the component. The intent filter must contain at least one <action> element.

```
<intent-filter>
    <action android:name="android.intent.action.MAIN" />

    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

Android Manifest file

- **8. action**
- It adds an action for the intent-filter. It is contained within the <intent-filter> element.

```
<action android:name="android.intent.action.MAIN" />
```

- **9. category**
- It adds a category name to an intent-filter. It is contained within the <intent-filter> element.

```
<category android:name="android.intent.category.LAUNCHER" />
```

Android Manifest file

- **11. uses-features**
- It is used to specify which hardware features your application requires. This will prevent our application from being installed on a device that does not include a required piece of hardware such as NFC hardware, as follows:

```
<uses-feature android:name="android.hardware.nfc" />
```

Android Manifest file

- **12. permission**
- It is used to create permissions to restrict access to shared application components. We can also use the existing platform permissions for this purpose or define your own permissions in the manifest.

```
<permission
    android: name="com.paad.DETONATE_DEVICE"
    android:protectionLevel="dangerous"
    android:label="Self Destruct"
    android:description="@string/detonate_description">
</permission>
```

SIX PHRASE
a Verondo Enterprise

THANK YOU