MOBILE APPLICATIONS DEVELOPMENT

C. FIRZA AFREEN

BOOK RIVERS
WE CREATE READERS

Website: www.bookrivers.com

Email: publish@bookrivers.com

Mobile: +91-9695375469

Place: Lucknow

ISBN: 978-93-90548-23-1

MRP: 999/-

Dedicated to

my parents for their love,

support and encouragement.

ACKNOWLEDGEMENT

I sincerely thank Almighty Allah for guiding and blessing me with strength and determination to complete this book successfully. I thank my beloved parents, family members and friends for their immense love and trust in me.

INDEX

UNIT-I

INTRODUCTION TO MOBILE APPLICATIONS

What is Mobile Application Development?

Mobile Application Development is the process of developing mobile applications by using softwarewhich will run on different mobile devices.

All types of mobile application will utilize a network connection. The developers will createinstallable software, which includes backend services such as accessing of data with an ApplicationProgramming Interface, not only that all the mobile application developers will test the developedapplication in different mobile devices to make sure that it is functioning properly.

Three major development approaches:

1. Native Applications
2. Hybrid Applications
3. Web Applications

The above given three application has its own pros and cons. Based on the requirement the developerwill decide what type of application should be developed. When selecting the right developmentapproach the developer must consider the following factors, the desired user experience, thecomputing resources which are needed to develop applications and the features which the app shouldinclude in it, the time factor, cost, budget and most importantly the target devices.

When the developer follows the right development approach, it will become possible for thedeveloper to build a successful android application.

NATIVE, HYBRID AND WEB APPLICATIONS

Native Applications

This is one type of mobile application which will run on any one specific platform that is OperatingSystem, for example,either on Android, iOS, Windows Phone. Programming languages like Java orKotlin will be used widely to develop this type of applications. Integrated Development Environment(IDE) will be used to develop native applications. The following languages are usually used in orderto built Native Applications.

1. For iOS the programming language which is used is Swift or Objective C.

2. For Android the suitable programming languages are Java or Kotlin.

3. For Windows applications, mostly C# or VB.NET will be used.

Native Apps – Advantages and Disadvantages

Advantages

1. Native applications are very fast, easy to deploy, easy to install on a specific platform.

2. The performance of native applications is best.

3. These applications are very interactive, it will run smoother, accepts requests provided by the user and the app will response according to the user's request.

Disadvantages

1.Compared to hybrid and web applications this types of applications are more expensive to develop.

2. The time which takes to develop this type of application is more as it has to be written in different languages for different operating systems.

3. Cost of maintenance for this type of application is high and publishing updates are little difficult.

Hybrid Applications

These are applications developed to be used across multiple platforms. Hybrid mobile applications are built in a similar manner as websites. Hybrid applications are not platform specific. Once app is developed and deployed it will run in any platform like iOS, Android, Windows etc. Hybrid apps developing method is mostly similar to the one which was used to develop website. The following technologies will be used in order to develop hybrid applications.

1. HTML

2. CSS

2. JavaScript

Hybrid Apps – Advantages and Disadvantages

Advantages

1. Adaptable to multiple platforms, as the same code can be re-used for Android, iOS, and Windows.

2.As mentioned when once this type of applications are developed it will run in any platform that is operating system, which means the developer can reuse the same code to deploy applications in different platforms.

3. It will take less time to develop this type of applications, as the app has to be developed only once, which will run in any mobile OS.

4. Application development cost is low compared to native applications.

Disadvantages

1.Eventhough it is taking less time to develop this type of applications, and application development cost too is low, but still the disadvantage which is present here is app performance will be slower compared to Native Apps.

2. If the app has to be developed which includes most animations and sound effects then, trying to develop such types of app in hybrid form is not recommendable, because hybrid app supports only less animations and sound effects.

Web Applications

One great thing about this web app is, it is not necessary that the user has to install this app from AppStore/PlayStore. This type of app uses modern web technologies to deliver an app like experience to its users. The users can access this app by using web URL (Uniform Resource Locator). The below given technologies are mostly used in order to develop Web Apps.

Web Apps are usually built by using the following technologies

1. HTML (HyperText Markup Language)

2. CSS (Cascading Style Sheet)

3. JavaScript

Web Apps – Advantages and Disadvantages

Advantags

1. Web Apps are easy to build.

2. Easy to maintain.

3. This type of apps works offline and also works properly if the network quality is low.

4. Once app is developed this app will run on all platforms like Android, iOS, Windows phone etc.

Disadvantages

1. To run this type of applications browser is mandatory.

2. Web Apps interact with the user less compared to Native Apps.

MOBILE OPERATING SYSTEMS

What is Mobile Operating System?

Mobile Operating System is a platform on top of which all types of mobile applications will run on mobile devices. It is the responsibility of the mobile operating system to take care of the proper functioning of mobile devices. The mobile operating system also decides which third party applications that is Mobile Apps, will run on the device.

9 Popular Mobile Operating Systems

1. Android OS

2. iPhone OS

3. Blackberry OS

4. Bada

5. MeeGo OS

6. Palm OS

7. Symbian OS

8. Web OS

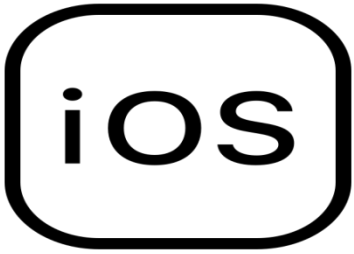
9. Windows Mobile (Windows Phone)

Android OS

Android is the most widely used operating system and it is designed mainly for touch screen mobile devices, tabs. Android is open source software, which means it will become possible for any user to view the code and to modify it according to their application requirements. Different versions of Android were available in the market like Cupcake, Donut, Eclair, Froyo, Kitkat etc.

iPhone OS / iOS (Apple)

iPhone Operating System was mainly developed to run on its iPhone devices. iOS which is alsoknown as the mobile operating system is supported only on different devices manufactured byApple like iPhone, iPad2 and iPod.



As the company does not license the Operating System for third-party hardware, this OS will not run in any devices manufactured by different company except Apple.

From Apple's Macintosh Operating System X, Apple iOS has been derived. This is the competitor of Android, and it's the second top-selling smart phones.

Blackberry OS

This Operating System has been especially developed for BlackBerry devices. This OS will run only on the devices manufactured by BlackBerry company like Curve, Perl, Storm Series etc.

The BlackBerry Operating System is a proprietary mobile operating system, which is Closed Source and Non-Free Software. Programming language like Java is used widely to write all the applications that will run in BlackBerry

devices. This OS is similar to Apple iOS because it can't run on different brands of phones.

Bada (Samsung Electronics)

Bada is a proprietary Samsung mobile Operating System that was first launched in market in the year2010.

The Samsung Wave was the first smart phone to use this mobile OS. It is also one of the Closed Source Software.

MeeGo OS (Nokia & Intel)

MeeGo is one type of Mobile Operating System which has been designed to work on a number ofdevices including smart phones, tabs, and various devices which uses Intel.

MeeGo is an Open Source Mobile operating System.

Palm OS

Palm OS is the mobile operating system developed by Palm Inc. in 1996. The Palm OS is a proprietary mobile operating system.

Palm OS was designed for ease of use with a touch screen-based graphical user interface. Palm OS is the operating system and it is designed mainly for touch screen mobile devices, tabs.

Symbian OS (Nokia)

Symbian was Nokia's OS developed and sold by Symbian Ltd.

The developers of this OS concentrated on the phone's functionality. For example, fast texting, taking pictures, voice control, quick dialling and it also includes some features of Personal Digital Assistants.

webOS

webOS is a mobile operating system. It is a proprietary Mobile OS, it was acquired by HP and referred as webOS

HP company uses webOS in a number of devices including several Smart Phones and Touch Pads. Initially, this OS was developed by Palm and later sold to LG Electronics.

Windows Mobile (Windows Phone)

This is the third most widely used operating system in the world. It is a series of Smart Phones OS developed by Microsoft. It includes many Software Features and this OS strongly competes with Android and iOS. This OS supports 25 different languages.

Windows Phone Development Tools

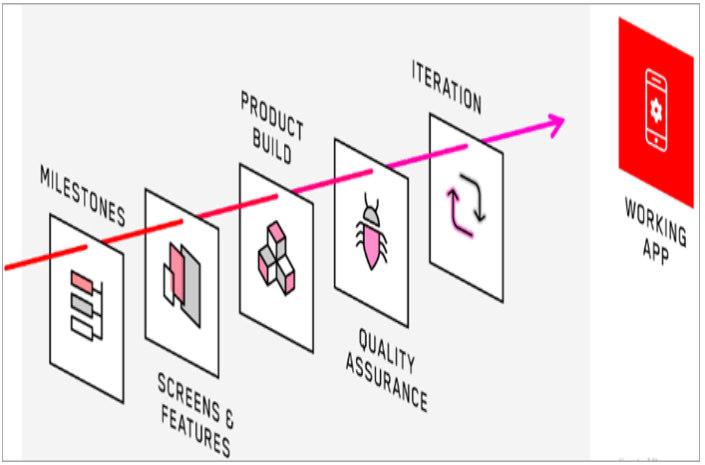
The following tools are used to develop Windows Phone:-

1. Microsoft Silverlight
2. Microsoft Visual Studio
3. Third Party IDE Eg: Microsoft xna, which is a set of freeware tools.

List of the top mobile application development software

1. AppyPie
2. AppSheet
3. Bizness Apps
4. Appery
5. iBuildApp
6. Shoutem
7. Rollbar
8. JIRA
9. AppInstitute
10. Caspio

Mobile Application Development Process



MOBILE APPLICATIONS

What is Mobile Applications?

A mobile application is a program or software application which will run on mobile devices.

Mobile Applications are also referred to as Mobile App. Mobile apps are specifically designed torun on mobile devices like smart phones, tabs etc.

The owner of the Mobile Operating System operates the application distribution platforms likethe App Store, Google Play Store. Some apps are free and for some apps users have to pay aprice to work with the app. Apps are generally downloaded by the user from Applicationdistributed platforms.

Types of Mobile Applications

The following is the three different types of Mobile Applications.Native App

1. Hybrid App

2. Web-based apps.

Native Apps

A native app will run specifically on one platform. It can be installed through an application store (such as Google Play Store or Apple's App Store).

Example:- Whatsapp

Benefits of Native Apps

1. Once installed Native Apps live on the device and it is accessed through icons on the device home screen.

2. When installed this applications will take full advantage of the entire device features present in the device– for example, they can use the camera, GPS, the list of contacts etc.

3. When a new content has been developed for the app, the developer will use the push-notifications which alert the users when every time if developer updates the app.

4. Native Apps maintain User Interface design of each operating system, this type of app offers best user experience. One very simple example is a Native App can have a left-aligned header in Android and a center-aligned header in iOS.

5. Redistribution of Native Apps is easy, as it is found in app store.

Limitations of Native Apps

1. High cost for building the app: Native apps which are developed for one OS will not run on another OS. Developers need to build a different app for different OS. Because of this

reason, it is time consuming; the developers have to maintain different versions of the Apps.

2. Even though the developer might publish native Apps, it is necessary that they have to keep the mobile website well maintained, as mobile brings more traffic. Because of this maintenance is higher.

Hybrid Apps

Hybrid as the name indicates it is the combination of Native and Web App. Hybrid App is a wayto expose the contents of existing websites in App format.

Example:- Instagram, Wikipedia.

Benefits of Hybrid App

1. Hybrid App development takes less time as once developed it can be deployed in different Operating Systems, that is hybrid apps supports cross-platforms.

1. Many Versions of apps will not be present, so maintenance is easy.

2. It can take advantage of a few features available in the device, like it access device Cameras, GPS, Phone Contacts etc.

3. Users can download this type of apps from App Store or Play Store which makes the distribution easy.

4. Embedded browsers will be present inside the apps.

Limitations of Hybrid Apps

1. Hybrid Apps are slower than Native Apps.

2.Graphics like animations and sound effects are less supported by this type of apps compared to Native Apps.

Web Apps

Web apps are actually websites that open in your smart phone with the help of browser, web apps users can't install, and in order to access this app the user should go to browser and type the accurate address to access this app. Web Apps are the most widely used apps by the users. This apps need browser in the device to run.

Example :- w3schools

Benefits of Web Apps

1. First benefit of this app is, it is easy to access.

2. Developers will do the updates easily− Just update in one location and all the users automatically have access to the latest version of the site, it is not necessary for the developer to send the push notifications to the users as alert for updates.

3. Installation of app is not required, as compared to native or hybrid app.

Limitations of Web Apps

1. It will not become possible for the mobile website to use some features. For example, access to the file system and local resources isn't available in websites.

2. It will not become possible for the users to access websites when they are offline.

3. Users will not have the app's icon on their home screen as a constant reminder. The website needs to be opened in a web browser only.

4. Developers use App Store and Google Play for distribution of native and hybrid apps, but in the case of web app it is not possible. So redistribution is not that easy.

Other Categories of Mobile App includes:

1. Gaming Apps
2. Business Apps
3. Educational Apps
4. Entertainment Apps
5. Utility Apps

Gaming Apps

The most popular category among types of apps, as more than 24% of all mobile applications isavailable in the App Store.

Examples of Gaming Apps

1. Clash of Clans (Android& IOS)
2. Candy Crush Saga (Android & IOS)
3. Angry Birds Go (Android & IOS)
4. Temple Run (Android & IOS)
5. Solitaire (Android & IOS)

Business Apps

This app is also referred to as the productivity apps, they holds the second place. Modern-day smart phones are capable of performing many complex tasks on the run.

Billing, buying, booking, sending emails, tracking working progress etc all these things can be carried out by the user easily through Business Apps.

Examples of Business Apps

1. Adobe Acrobat Reader (Android & IOS)

2. Voxer Walkie Talkie Messenger (Android & IOS)

3. Indeed Job Search (Android & IOS)

4. Facebook Pages Manager (Android & IOS)

Educational Apps

Kids can learn while playing educational game apps. Students may learn out of the class andadjust individual learning pace. Moreover, according to recent reviews many educational appsare useful for students.

Examples of Educational Apps

1. Duolingo – Learn Languages for Free (Android & IOS)

2. Photomath – Camera Calculator (Android & IOS)

3. Quizlet: Flashcard & Language App to Study & Learn (Android & IOS)

4. Lumosity – Brain Training (Android & IOS)

5. TED (Android & IOS)

Entertainment Apps

These are the apps that have a tendency to invoke a kind of dependency, they keep us engaged,logged in, always checking for updates.

Examples of Entertainment Apps

1. Netflix (Android & IOS)

2. Dubsmash (Android & IOS)

3. Talking Tom Cat (Android & IOS)

4. Amazon Prime Video (Android & IOS)

Utility Apps

Utility software is used on a daily basis by literally every one of us. For example taxi apps. However, most popular types of applications here are scanners, trackers, healthcare, first aid manuals, etc.

Other Examples of Utility Apps

1. Flashlight (Android & IOS)

2. QR Reader (IOS)

3. Speed test (Android & IOS)

MOBILE DATABASES

What is Mobile Database?

SQLite is the inbuilt database present in the mobile phones. The Mobile database, SQLite is differentfrom normal database and mobile database can be transferred easily. The mobile database are notconnected with main database, still it will become possible for mobile database to share the data. Inmobile database any form of data can be stored either it may be text, audio, video etc.

The mobile database includes the following components

1. In main database all the data will be stored normally and it is linked directly to the mobile database.

2. By using mobile database users can view the data present in it even while if they are on the move. The mobile database also shares information with the main database.

3. The different devices like mobile phone, laptop etc are the one which can be able to access the mobile database to retrieve the data present inside it.

4. Between main database and mobile database the communication link is the one which transfers the information.

Advantages of Mobile Databases

1. The main advantage of mobile database is data can be accessed from anywhere. It provides a wireless database access.

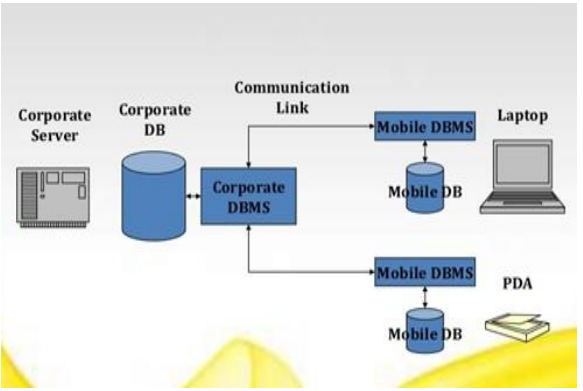
2. The database systems are synchronized using mobile databases and multiple users can access the data with seamless delivery process.

3. Because the mobile database are inbuilt it requires very little support and maintenance.

Disadvantages of Mobile Databases

1. Storing data in a mobile database is not much secured compared to data stored in conventional database.

2. Sometimes loss of data in database will take place. Moreover, the mobile database depends on the battery; database may frequently lose power because of limited battery.



Features of Mobile Database

It provides all the services which relational database does; still it requires only little memory. This database will analyze and manipulate the data.

Users can choose to download data and work on them, confine real time and synchronize later, personalized for Mobile Applications

Three parties involved in Mobile Databases

Fixed Hosts

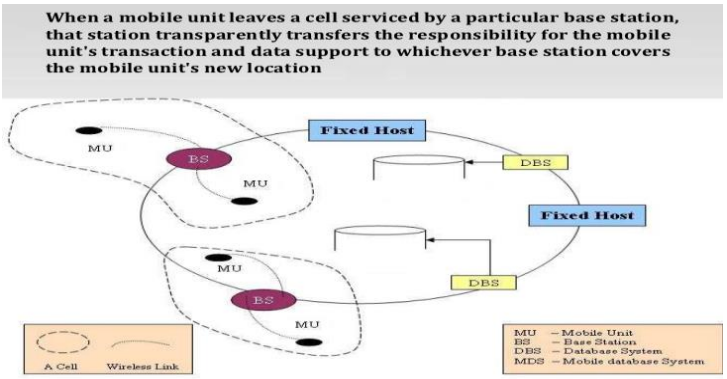
The work of Fixed Hosts is to perform the transaction and data management functions. The fixedhost will perform these functions with the help of database servers.

Mobile Units

They are the portable systems, which move around a geographical region that includes the cellular network.

Base Stations

The work of Base Stations is to pass communications with the mobile units to and from the fixed hosts.



Mobile Database Examples:

Example 1: SQL Anywhere

Developed by:Watcom

Initial Name: Watcom SQL

Operating System: Cross-Platform

Type: RDBMS

Example 2: Scimore DB

Developed by-Scimore UAB

Stable Release-4.0.2581/Jan 2

Operating System-Windows

Type:RDBMS

License:Proprietary

Website:www.scimore.com

Example 3: SQLite

Developed By:D.Richard Hipp

Initital Release: August 2000

Stable Release-3.7.10 Jan 16, 2012

Written in-C

Operating System: Cross-Platform

Size-275 KB

Type: RDBMS

License: Public Domain

Website: www.sqlite.org

ANDROID BASICS

1. What is Android?
2. Why Android?
3. History of Android
4. Features of Android
5. Categories of Android Applications
6. OSS and OHA
7. Android Versions

What is Android?

Android is an open source and Linux based operating system for mobile devices such as tablet computers and smart phones. It is developed by Google and later the OHA (Open Handset Alliance). Java language is mainly used to write the android code. The goal of android project is to create a successful real-world product that improves the mobile experience for end users.

Why Android?

1. Open Source
2. Larger Developer and Community Reach
3. Increased Marketing
4. Inter App Integration
5. Reduces cost of development
6. Higher Success Ratio
7. Rich Integrated development environment

HISTORY OF ANDROID

In October 2003, Andy Rubin has founded Android in Palo Alto, California. In 17th August 2005, Google has acquired Android. Rich Miner, Chris White and Nick Sears are the key employees of Android. Android is the nick name of Andy Rubin given by his co-workers. On 2007, Google announced the development of Android Operating System. In 2008, HTC company launched the first android mobile.

FEATURES OF ANDROID

The following are the features of Android:-

1.Beautiful UI

Android Operating System provides a beautiful user interface with the help of which easily users can interact with the device.

2.Connectivity

Android Devices supports different connectivities like GSM/LTE,Bluetooth, WiFi etc.

3.Storage

SQLite is the inbuilt database present in the mobile devices with the help of which users can save any type of data in their mobile devices.

4.Media Support

Android supports different media as listed, H.263, H.264, MPEG-4 SP, AMR, AMR-WB, AAC, HE-AAC, AAC 5.1, MP3, MIDI.

5.Messaging

Android supports both SMS and Multimedia Messaging System.

6.Multi-touch

Android device supports multi-touch which was first introduced only on the handset such as the HTC.

7.Multi-tasking

Users can run more than one application at the same time simultaneously, they can move from one app to another.

8.Resizeable Widgets

Widgets are resizable so users can expand or shrink them to save space.

9.Wi-Fi Direct

A technology that lets app discover and pair directly with another device, over a high-bandwidth.

Categories of Android Applications

1. Music
2. News
3. Sports
4. Multimedia
5. Lifestyle
6. Travel
7. Weather
8. Books
9. Social Media
10. Utilities
11. Finance
12. Business

OPEN SOURCE SOFTWARE (OSS)

Android is an open source software, which has been mainly designed for mobile devices, and this open source project is led by Google. Since Android is an open source, developers can refer the codings and easily they can create a customize applications and not only that it offers various different information needed to build a customised applications.

Android provides a stable environment for millions of users; it also ensures that the devices meet the compatibility requirements. Goal is to avoid any central point failure, for example, because of open source many developers will develop many different types of apps. Android makes sure that one industry player that is developer does not restrict or control the work of any other developer.

Public documentation is available to everyone. Android is a full quality production operating system, and it provides customizable source code that can be ported to any device.

OPEN HANDSET ALLIANCE (OHA)

The OHA stands for Open Handset Alliance, it's a business association. The main work of OHA is todevelop open mobile device standards. The companies such as HTC, Dell, Intel, Motorola etc comesunder this OHA, and also OHA has about 80 member companies.

The famous product which was developed by OHA is the Android Operating System, and it is theworld's most popular smart phone platform. The members of the OHA include the mobile operators,handset manufacturers, software development firms, and commercialization companies.

ANDROID VERSIONS – BRIEF OVERVIEW

The following lists shows the Platform Version, API Level and VERSION_CODE

1. Android 6.0-23-Marshmallow
2. Android 5.1-22-Lollipop_MR1
3. Android 5.0-21-Lollipop
4. Android 4.4W-20-Kitkat_Watch
5. Android 4.4-19-Kitkat
6. Android 4.3-18-Jelly_Bean_MR2
7. Android 4.2,4.2.2-17-Jelly_Bean_MR1
8. Android 4.1,4.1.1-16-Jelly_Bean
9. Android 4.0.3,4.0.4-15-Ice_Cream_Sandwich_MR1
10. Android 4.0,4.0.1,4.0.2-14-Ice_Cream_Sandwich
11. Android 3.2-13-Honeycomb_MR2
12. Android 3.1.x-12-Honeycomb_MR1
13. Android 3.0.x-11-Honeycomb
14. Android 2.3.3, 2.3.4-10-Gingerbread_MR1
15. Android 2.3, 2.3.1, 2.3.2-9-Gingerbread
16. Android 2.2.x-8-Froyo
17. Android 2.1.x-7-Eclair_MR1
18. Android 2.0.1-6-Eclair_0_1
19. Android 2.0-5-Eclair
20. Android 1.6-4-Donut
21. Android 1.5-3-Cupcake
22. Android 1.1-2-Base_1_1
23. Android 1.0_1_Base

ANDROID VERSIONS AND COMPATIBILITY

Android Versions – A living history from 1.0 to 10

Android version 1.0 to 1.1

The software in this version includes early suite of Google apps which includes Gmail, GoogleMaps, Calendar and YouTube, all of which were incorporated into the operating system.

The Android 1.0 Home Screen

Android version 1.5: Cupcake

The first on-screen keyboard was introduced in this version. Various refinements the developers has done in this version.

Cupcake 1.5 Screen

Android version 1.6: Donut

The main feature which was included in this version Android 1.6, is the ability of the Operating System to operate on a variety of different screen sizes and decree.

Donut 1.6 Screen

Android versions 2.0 to 2.1: Eclair

The name given to Android Version 2.0 is Eclair, which was just emerged six weeks after version 1.6that is Donut, then 2.1 update of the version has emerged in the market after 2 months, which is alsocalled as Eclair. Converter app like speech-to-text function has been introduced in this version andalso live wallpapers too has been introduced in this version.

Eclair 2.1 Screen

Android Version 2.2: Froyo

The next version of Android which came into market after the advent of 2.1 is 2.2 and named it as Froyo. The developer has brought changes in the Interface Design in this version. For example, including the addition of the new-standard dock at the bottom of the home screen, front facing features etc.

Froyo 2.2 Screen

Android version 3.0 to 3.2: Honey Comb

Even in this version the developers concentrated on the User Interface design and created much refined interface. The name given to this version is the Honey Comb. Under the guidance of chief Matias this modifications has been done to the basic interface. It had a space-like "holographic" design that traded the platform's trademark green for blue.

HoneyComb 3.2 Screen

Android Version 4.0: Ice Cream Sandwich

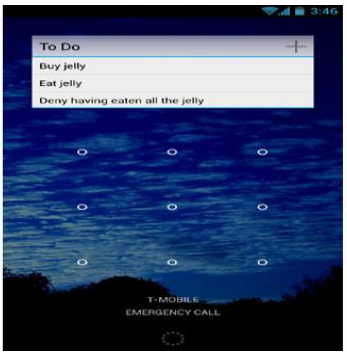
The version 4.0 named Ice Cream Sandwich was released in the year 2011, from this version only the modern design era has started for Android. Ice Cream Sandwich dropped much of Honeycomb's "holographic" design appearance, but kept its use of blue as a system-wide highlight.

Ice Cream Sandwich 4.0 Screen

Android versions 4.1 to 4.3: Jelly Bean

The name given to this version is the Jelly Bean, the main feature which is present in this versionis the interactive notifications, more featured voice search, and an advanced system for displaying search results.

Android Jelly Bean Screen



Android version 4.4: KitKat

KitKat release marked the end of Android's dark era, the blacks of Gingerbread version and the blues of Honeycomb version was removed from the operating system.

Lighter backgrounds and more neutral highlights took their places. The “OK, Google”, feature was first supported in this version.

Kitkat 4.4 Screen

Android versions 5.0 and 5.1: Lollipop

Android 5.0 Lollipop version was first introduced in the year 2014, in this version Google completelyredesigned Android. The card-based concept that has been spread throughout Android became animportant User Interface design pattern.

Lollipop 5.1 Screen

Android version 6.0: Marshmallow

The On Tap that is the search-screen feature was introduced in Marshmallow which was one of the most attractive elements present in this version. Android 6.0 did introduce some important feature like stuff with lasting impact, including app permissions.

Marshmallow 6.0 Screen

Android versions 7.0 and 7.1: Nougat

Google Assistant was the important feature which was launched in this version.

Google Assistant is an artificial intelligence–powered virtual assistant developed by Google Company that is primarily available on mobile and smart home devices.

The following functionalities were offered by Google Assistants such as voice commands, voice searching, and voice-activated device control. Split screen mode was introduced in this version by Google in the year 2016.

Nougat 7.1 Screen

Android version 8.0 and 8.1: Oreo

The 2017 release also included some important elements that furthered Google's goal of aligning Android and Chrome OS and improving the experience of using Android apps on Chrome. Android Oreo version added a variety of nice things to the platform, for example, a native picture-in-picture mode.

Oreo 8.1 Screen

Android Version 9: Pie

In this version for privacy and security feature the developers has given much importance. A twist to Android's Battery Saver mode and a variety of enhancements to privacy and security features. In the year August 2018, Google released this version. Large Home and small Back buttons to the user interface are the most visible updates present in this version.

Pie 9 Screen

Features of Android 10

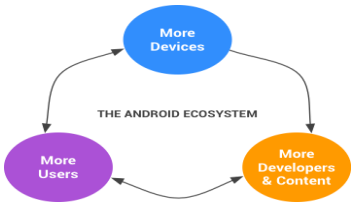
1.The Smart Reply feature present in this version helps simplify communication by suggesting responses and recommend actions.To allow the user to hear more clearly the Sound Amplifier present lets user fine-tune the audio settings of the phone.

2. Dark Theme uses true black to save battery power and give your eyes a rest. Some users may prefer it to Android's normal look.

3. Compared to version 9, enhanced security and privacy settings are present in this version. The user's data will be more secure. By changing settings, users can decide and control whether to share the data and the location information.

ANDROID COMPATIBILITY

As mentioned earlier, android is an open source, any hardware device manufacturer can build a device that runs the Android operating system. We can say that a device is "Android compatible" only if the Android runs the apps properly which has been specifically designed for Android Environment. Android compatibility program is the one which contains the exact details of the Android execution environment. In order to consider the device as Android compatible each device must pass through the Compatibility Test Suite (CTS).



The three key components of the Android compatibility program:

1. The first is the Android Open Source Project, source code.

2. The CDD that is the **Compatibility Definition Document (CDD)**, which represents the aspects of compatibility.

3. TheCTS that is the **Compatibility Test Suite (CTS)**, which represents the "method" of compatibility.

In order to build an Android compatible mobile device, the following three steps should be followed:

1. The first step is to get the Android software source code. It will become easy for the developer to get it, because an Android is an open source. Then the source code for the Android platform must be ported to your hardware.

2. Conform to the Android Compatibility Definition Document. The CDD document defines what the exact software is and hardware requirements which are needed to make a device Android compatible.

3. It is necessary to pass the Compatibility Test Suite (CTS). Use the CTS as an ongoing support to assess compatibility during the development process.

ANDROID DEVICES

Android device is nothing but the devices which contains the Android as an Operating System in it. Android is a set of software which includes operating system, core applications and middleware, mainly designed for mobile devices. We can say that Android device may be smart

phones, tabs etc, which requires Android as an Operating System.

Open Handset Alliance that is the OHA initially developed Android, which is lead by Google. Acer,HTC, Samsung, LG, Sony Ericson and Motorola are the Android device manufacturers. PopularAndroid devices include smart phones, tabs.

Since android is an open source, android developers and programmers can find most of theinformation at the Android developers website, which acts as the guide for them in order to build a successful Android Project. This site offers Software development kit too.

Android has become one of the most famous and widely used operating system, and it has left behinddifferent operating systems like Windows, Symbian, Bada etc.

Various companies which manufacture mobile devices choose Android as an Operating System fortheir device, due to its popularity.

The following are the reasons behind the success of an Android:

First, is the cutting edge technology offered by Google. User friendly platform, tech-savvy and casual users both can work with this platform easily.

Android, can be used not only in smart phones but also in tabs, smart devices etc. As Android is an open source, any users can do the modifications to the Android Software Development Kit easily. An android device supports huge volume of applications.

3 FUNDAMENTAL THINGS (PREREQUISITES) TO LEARN ANDROID

1.Mastering the following programming Language

Java and XML (Extended Markup Language)

2. Acquaintance with the Right Development Tools and environment

Users should know which environment is best for developing Android Applications out of available different environments like Android Studio IDE, Eclipse, Apache Maven.

3.Understanding of the Application Components Activities

Different events are represented as an Activity in a single screen with a user interface (for example, an email app may have one activity showing a list of new emails, another activity which is composing emails, and reading emails etc). Another example is if we take any app we can perform different operations on it, each operation is represented as an activity here.

ServicesWithout this component, it will not become possible for any application to run successfully, this services component provides background support to all the apps, another important point here is this component does not provide any user interface.

SETTING UP SOFTWARE – IDE

The following are the steps which should be followed to setup an Android Studio – Integrated Development Environment.

Step 1 - System Requirements

Microsoft Windows 10/8/7/Vista/2003 (32 or 64-bit) Mac OS X10.8.5 or higher, up to 10.9

Step 2 - Setup Android Studio Installation

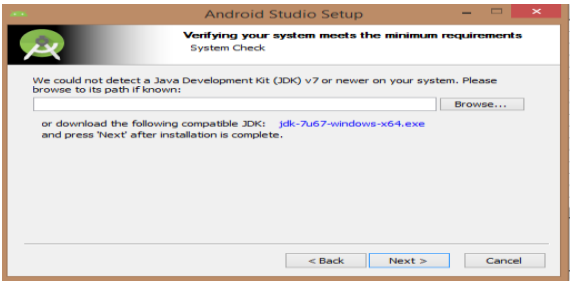
Android Studio.exe file is needed, which can be downloaded for Android site, Before installing Android Studio, Machine requires Java Development Kit that is JDK to be installed.

Step 3:

Verify that the machine has Java Development Kit, then launch the *Android Studio.exe*. The following window will be opened click on Next.

Step 4:

The next step is to locate the path of the JDK which is present in our device. After mentioning JDK path click on next.

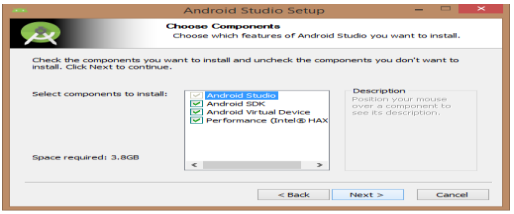


Step 5:

Once JDK path has been specified properly the initiation of JDK to Android SDK will take place, click OK and then click on Next.

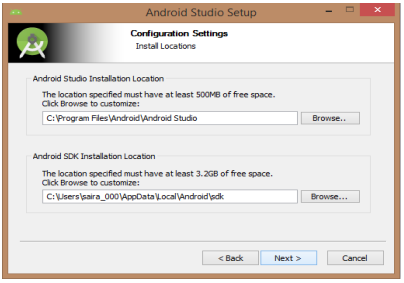
Step 6:

In the Android Studio Setup dialog box the necessary components like Android SDK, Android Virtual Machine and performance (Intel chip) should be selected because all these components are required to develop an Android Application, after checking click on Next.



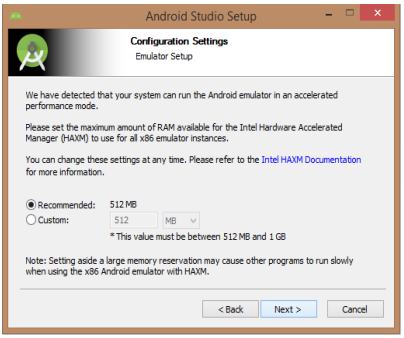
Step 7:

The system will take default location to store Android Studio and SDK, if needed by clicking on Browse button the user can select the space where they want to launch the application. Below the image has taken default location of windows 8.1 x64 bit architecture, Click on Next.



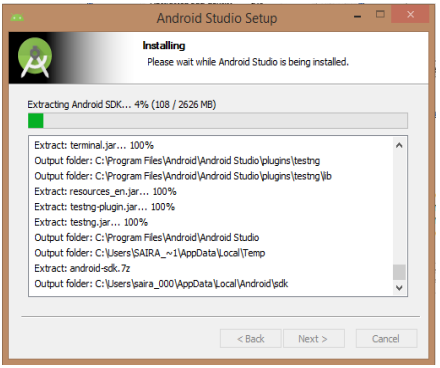
Step 8:

The Android Programs will run in Android Emulator only, it is necessary to specify the RAM space for emulator, by default it will take 512 MB, users can modify the size too if needed, then click on Next.



Step 9:

This is the final step, in this step the system will extract SDK packages into your machine, it will take a little time to finish the task and will take approximately 2626MB of Hard disk space, Click on Next.

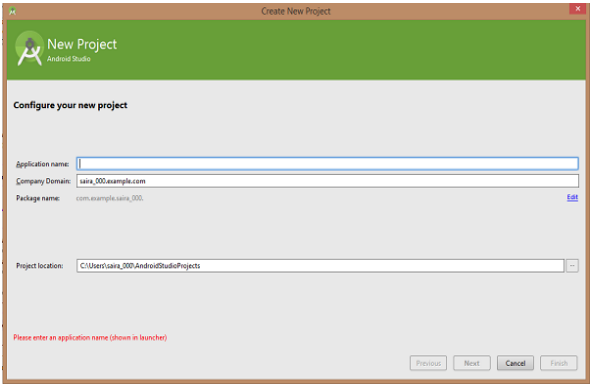


Step 10:

After completing all above steps perfectly, you will get finish button. After clicking on it, the following shown screen will be displayed.

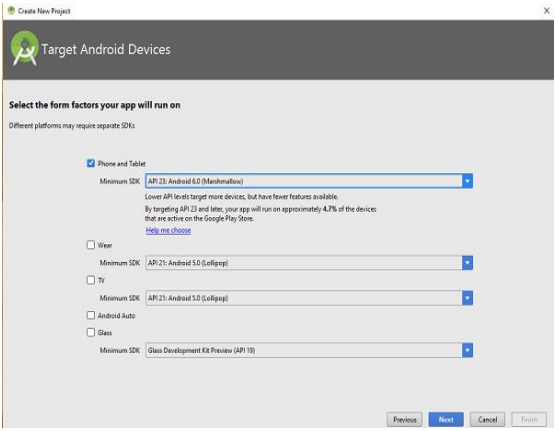
Step 11:

From the above shown dialog box click on Start a New Android Studio Project. The create new project dialog box will be displayed, enter Application name, package information and location where you want your project to be saved.



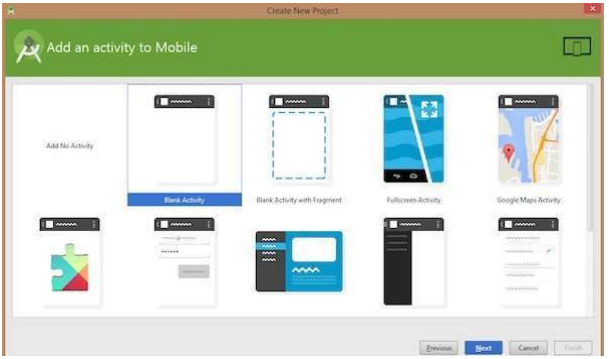
Step 12:

After entering the details like Application Name and its location etc, the following screen will be displayed, from which the user has to select the Minimum SDK Version, remaining details can be left as default, click on Next.



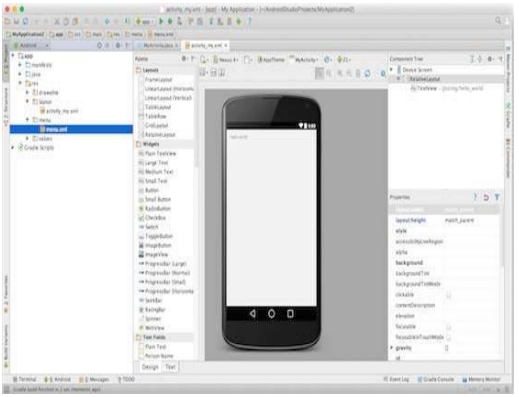
Step 13:

This is the next level, here the user should select the layout corresponding to their application, the blank activity will be selected by default, if needed the user can select the different layout.



Step 14:

Finally Android Studios Integrated Development Environment will be opened, by using which the developer can start designing the projects.



XML

XML stands for Extensible Mark-up Language. XML is mainly used for sharing data on the internet. It is very popular. The three types of XML Parsers provided by Android are DOM (Document Object Model), SAX (Simple API for XML) and XMLPullParser. XMLPullParser is the most widely used one out of three, the reason is it is efficient and easy to use.

Simple XML Example

<?xml version="1.0"?>

<currentlocation>

<city uniqueid="2643743" cityname="London">

<coord lon="-0.12574" lat="51.50853"/>

<country>GB</country>

</city>

<temp value="289.54" min="289.15" max="290.15"

unit="kelvin"/>

</currentlocation>

XML – Elements

The following are the various XML elements and its description.

1.Prolog

Prolog is nothing but the information about the file. The first line always contains the Prolog of the file.

2.Events

The different events present in the XML file are the Document Start, Document End, Tag Start, Tag End and Text etc, as it's the basic thing that the XML files will have, text and tags which are considered as the elements of XML.

3.Text

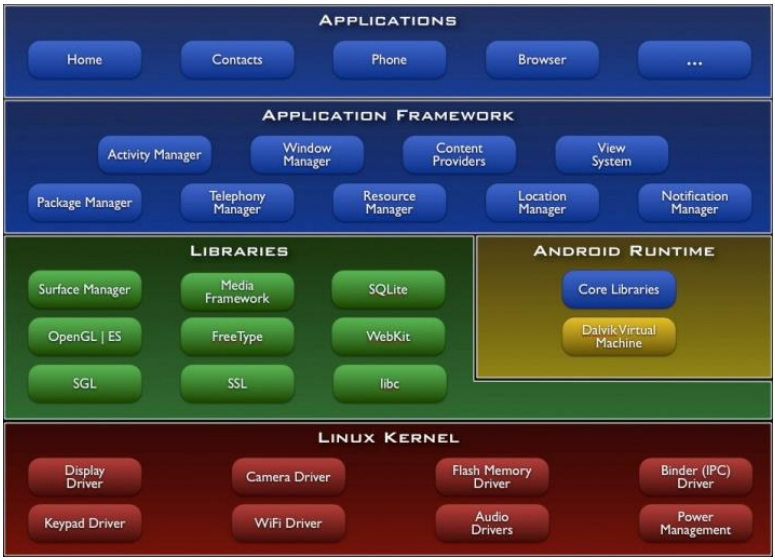
Apart from simple tags and events, the XML file also contains text, which is the important element of the XML. Whichever text is present in the tag that will be displayed as the output.

4.Attributes

Attributes are the additional properties of a tag. For example from the above shown simple XML program, the different attributes which is present are the temp value, min, max, unit.

ANDROID ARCHITECTURE

Android operating system is a heap of software components which is divided into five sections and four layers.



Android Architecture

The following is the 5 important sections present in the Android Operating System

1. Linux Kernel
2. Libraries
3. Android Runtime

4. Application Framework

5. Applications

LINUX KERNEL

Linux is the bottom most layer present in the Android Operating System, it contains nearly 115 patches which is nothing but a program that updates file according to the instruction, this is the main work done by the Linux Kernel.

The Linux kernel provides a generalization between the device hardware and it has all the important hardware drivers for camera, keypad, display, USB, Power Management etc.

Not only this, the Linux Kernel handles all the things related to networking and a vast array of device drivers.

LIBRARIES

The next layer which is present on top of the Linux Kernel is the set of libraries, which contains important elements like SQLite, SSL, SGL etc. As the name indicates, this layer libraries are mainly used to store and retrieve the data, share the data among different applications, to record and play audio and video etc. SSL (Secure Socket Layer) libraries are responsible for Internet security.

Android Libraries

The following are the summary of important libraries available for the Android Developer.

android.app– This is very important library it provides access to the application model and is the foundation of all Android applications.

android.database– This library mainly includes SQLite Database Management and also used to access data published by content providers.

android.opengl– This library is mainly used if the user wants to work with animation and sound effects, for this the SGL present in the Library provides support. **android.os** – If two process wants to communicate with one another, or if one application wants to communicate with OS, then this library will be very useful, not only that, android.os library provides access to all the system services.

android.text– Mainly used to display text on a device, this library is used to make and control text on a device display.

android.view– This is the basic building blocks of any application. Inside view only all the necessary components needed for app will be placed.

android.widget – In order to build an Android Application the different components present inside widget such as Labels, TextBox, List Views, Layout Managers, Check Boxes, Spinners etc will be used, the widget package also contains lots of other useful components too.

android.webkit– As the name indicates it is a set of classes proposed to allow web-browsing capabilities to be built into applications.

ANDROID RUNTIME

Android Runtime is present in the second layer from the bottom in Android Architecture and it is the third section present in the architecture. The important component present in this section is the Dalvik Virtual Machine, it's a kind of

Java Virtual Machine which is specially designed for Android.

DALVIK VIRTUAL MACHINE

The three important works which the DVM does is controlling memory, battery life and performance of the mobile. The DVM was developed by Dan Bornstein. The Dalvik Virtual Machine uses two core features of Linux Kernel, they are memory management and multi-threading. When user opens one application the DVM creates its own process, with its instance and it makes the application to run in its own process. When the Android Application developers creates an application using Java programming language, this DVM provides full support as it has the core libraries for it.

Advantages of Dalvik Virtual Machine

1. In Dalvik Virtual Machine, Application Package that is APK is executable.
2. With the help of DVM the application executes faster.
3. From Android version 2.2, the Software Development Kit SDK, has its own JIT Compiler, different compilers like load-and-go, debugging and optimizing compilers are present. This DVM mainly makes use of JIT Compiler.
4. The main purpose of designing DVM is that a device can run multiple instances of the Virtual Machine effectively.

Disadvantages of Dalvik Virtual Machine

1. It is not possible to use DVM for other Operating Systems like Windows, iOS etc, it will work only in Android Operating System.

2. In order to implement this DVM effectively, the developer has to write more instructions.

3. Even though the DVM is present in device, sometimes app installation takes more time.

4. The Dalvik VM consumes more internal storage space of your device.

APPLICATION FRAMEWORK

This is the third layer present in Android Operating System and this layer provides many important services to the Application Developer in the form of Java class. Application developers can make use of these services to develop their applications.

The Android framework provides the following key services:

1. Activity Manager
2. Content Providers
3. Resource Manager
4. Notifications Manager
5. View System

Activity Manager

The Android Activity has many life cycles, and it is the responsibility of the Activity Manager totake care of life cycles of all the activities.

Content Providers

If the application wants to store or share the data with other application then this framework Content Provider will be very useful.

Resource Manager

This framework allows access to non-code embedded resources, which are the strings, colour settings and user interface layouts.

Notifications Manager

This Notification Manager allows applications to display and gives notification to the user.

View System

This application framework provides an extensible set of views used to create application user interfaces.

APPLICATIONS

All the Android Application will be present at the top layer. The developer develops application, and then it will be installed on this layer only. Examples of some Android Applications are Contacts, Messages, and Browser etc.

The Application components are the important building blocks of an Android Application, there are 4 application components which is present in Android. AndroidManifest.xml file describes each application component, its purpose and how it interacts with the application.

Four main components that can be used with an Android application :-

Activities

Activities describe, how and what are the different activities which the user performs by interacting with the components present in an Application.

Services

The application developed by the developer will not work all alone, it needs some supportive files, and the services provide the background processing support for all application.

Broadcast Receivers

The main work of broadcast receiver is to handle the communication between the Operating System and applications.

Content Providers

In order to handle data and database management issues this component is very useful.

ANDROID EMULATOR

1. What is Android Emulator?
2. Functionalities present in the emulator
3. Emulator Commands.

What is Android Emulator?

The Android Software Development Kit includes virtual device which is also known as Emulator, the programs which you run to check for its proper functioning will run on this emulator only. Without physical device, in emulator itself all the applications can be tested, each time in emulator the user can set the minimum SDK version to know whether the application developed is compatible with the device or not.

The different functionalities which are performed by the emulator are as follows:

1. Creating AVD
2. Changing Orientation
3. Sending SMS through Emulator

4. Emulator making calls

5. Emulator Transferring files

Creating AVD

In order to run the application in an emulator, the first step is to create AVD. Once if the user has installed Android Studio or Eclipse, then the AVD Manager can be launched from both of these applications, it depends on the user whether they want to use Android Studio or Eclipse for the application development. In order to launch AVD in Android Studio, the developer should follow the given steps.

Select Tools -> AVD Manager. (OR)

Click AVD Manager in the toolbar.

The following Android Virtual Device Manager dialog box will be displayed. By clicking on the“Create Virtual Device” button, the developer can create a virtual device that is the emulator to run the application.

Changing Orientation

By default, when virtual device is launched then it will be in vertical form. By pressing Ctrl+F11 Key, the orientation of the virtual device can be changed. Only two types of orientation are supported, either it may be horizontal or vertical.

Example

First when the emulator is launched, then the view will be like how it's shown below.

Once if it is launched, press Ctrl+F11 key to change its orientation from vertical to horizontal if needed.

Example Android Program for Sending SMS through Virtual Device

Normally by using the default messaging app present in our device we will send message, but it is also possible in Android to develop our own messaging app.

The three important files which are used to develop Android Application is MainActivity.java, activitymain.xml and AndroidManifest.xml. The following coding shows how

to send a message using messaging app developed by the user by writing simple coding in the respective files.

CODING:-

MainActivity.java

```
package com.example.smspermissionpgm;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.app.PendingIntent;
import android.content.Intent;
import android.telephony.SmsManager;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
public class MainActivity extends ActionBarActivity
{
EditText mobileno,message;
Button sendsms;
@Override
protected void onCreate(Bundle savedInstanceState)
{
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
mobileno=(EditText)findViewById(R.id.editText1);
message=(EditText)findViewById(R.id.editText2);
sendsms=(Button)findViewById(R.id.button1);
sendsms.setOnClickListener(new OnClickListener()
{
@Override
public void onClick(View arg0)
 {
String no=mobileno.getText().toString();
String msg=message.getText().toString();
//Getting intent and PendingIntent instance
Intent intent=new
Intent(getApplicationContext(),MainActivity.class);
PendingIntent
pi=PendingIntent.getActivity(getApplicationContext(), 0,
intent,0);
//Get the SmsManager instance and call the sendTextMessage
method to send message
SmsManager sms=SmsManager.getDefault();
sms.sendTextMessage(no, null, msg, pi,null);
Toast.makeText(getApplicationContext(), "Message Sent
successfully!",
Toast.LENGTH_LONG).show();
```

```
}
});
}
@Override
public boolean onCreateOptionsMenu(Menu menu)
 {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
@Override
public boolean onOptionsItemSelected(MenuItem item)
 {
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId();
if (id == R.id.action_settings)
{
return true;
}
return super.onOptionsItemSelected(item);
}
```

}

activity_main.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
 android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.smspermissionpgm.MainActivit
y" >
<EditText
android:id="@+id/editText1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentRight="true"
android:layout_alignParentTop="true"
android:layout_marginRight="20dp"
android:ems="10" />
<EditText
    android:id="@+id/editText2"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText1"
    android:layout_below="@+id/editText1"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine" />
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/editText1"
    android:layout_alignBottom="@+id/editText1"
    android:layout_toLeftOf="@+id/editText1"
    android:text="Mobile No:" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/editText2"
    android:layout_alignBottom="@+id/editText2"
    android:layout_alignLeft="@+id/textView1"
    android:text="Message:" />
<Button
```

```
    android:id="@+id/button1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText2"
    android:layout_below="@+id/editText2"
    android:layout_marginLeft="34dp"
    android:layout_marginTop="48dp"
    android:text="Send SMS" />

</RelativeLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.smspermissionpgm"
android:versionCode="1"
android:versionName="1.0">
<uses-permission
android:name="android.permission.SEND_SMS"/>
<uses-permission
android:name="android.permission.RECEIVE_SMS"/>
<uses-sdk
android:minSdkVersion="8"
android:targetSdkVersion="18" />
```

<application

android:allowBackup="true"

android:icon="@drawable/ic_launcher"

android:label="@string/app_name"

android:theme="@style/AppTheme">

<activity

android:name=".MainActivity"

android:label="@string/app_name">

<intent-filter>

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

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

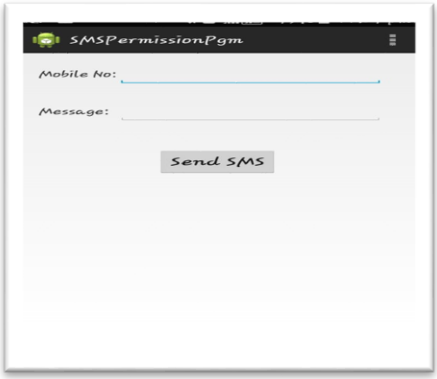
</intent-filter>

</activity>

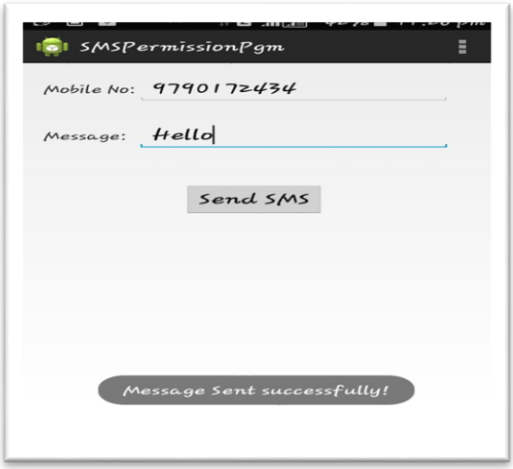
</application>

</manifest>

OUTPUT


SMSPermissionPgm
Mobile No:
Message:
Send SMS


SMSPermissionPgm
Mobile No: 9790172434
Message: Hello
Send SMS



Emulator Making Call

It is also possible in Android to develop our own Phone Call app, and make a call through emulator. In order to accomplish this, the following coding should be written in respective files.

The three important files which are used to develop Android Application is MainActivity.java, activitymain.xml and AndroidManifest.xml. The following coding shows how to make a call through the emulator, by writing simple coding in the respective files, and also by doing certain settings in the device.

CODING:

MainActivity.java

```
package com.example.callpermission;

import android.net.Uri;

import android.os.Bundle;

import android.support.v7.app.ActionBarActivity;
```

```
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
publicclass MainActivity extends ActionBarActivity
{
EditText edittext1;
Button button1;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
edittext1=(EditText)findViewById(R.id.editText1);
button1=(Button)findViewById(R.id.button1);
button1.setOnClickListener(new OnClickListener()
{
@Override
publicvoid onClick(View arg0)
 {
```

```
String number=edittext1.getText().toString();
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:"+number));
startActivity(callIntent);
}
});
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId();
if (id == R.id.action_settings)
{
```

returntrue;

}

returnsuper.onOptionsItemSelected(item);

}

}

activity_main.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"
android:layout_width="match_parent"

android:layout_height="match_parent">

<TextView

android:id="@+id/fstTxt"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:layout_marginTop="150dp"

android:text="Mobile No"/>

<EditText

android:id="@+id/editText1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:ems="10">

</EditText>

<Button

android:id="@+id/button1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:text="Call" />

</LinearLayout>

AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>

<manifest
xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.callpermission"

android:versionCode="1"

android:versionName="1.0">

<uses-permission
android:name="android.permission.CALL_PHONE" />

<uses-sdk

android:minSdkVersion="8"

android:targetSdkVersion="18" />

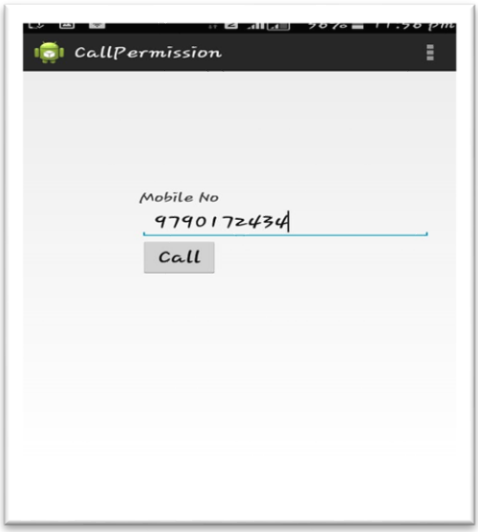
<application

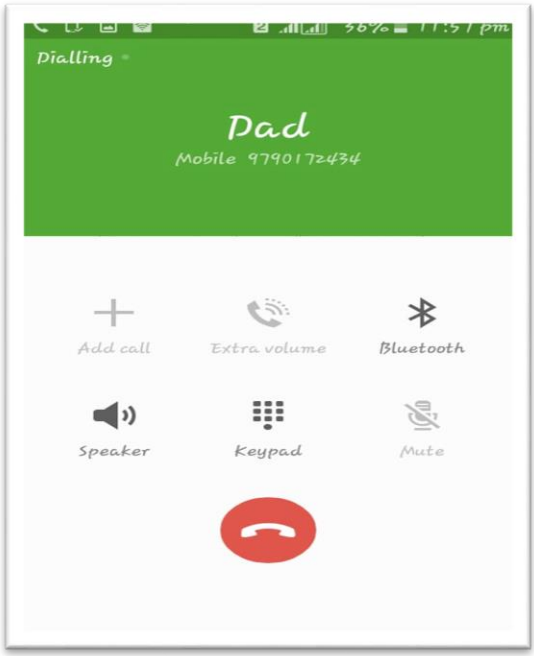
android:allowBackup="true"

android:icon="@drawable/ic_launcher"

```
android:label="@string/app_name"
android:theme="@style/AppTheme">
        <activity
        android:name=".MainActivity"
        android:label="@string/app_name">
<intent-filter>
    <action android:name="android.intent.action.MAIN" />
    <category
android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
</manifest>
```

OUTPUT:


CallPermission
Mobile No
9790172434
Call


Dialling
Dad
Mobile 9790172434
Add call
Extra volume
Bluetooth
Speaker
Keypad
Mute

UNIT- II

ANDROID DEVELOPMENT

JAVA FOR ANDROID

Java is the Programming language used to develop Android Application. Different Integrated Development Environment, which is used to develop Android Applications are:

1. Android Studio
2. Eclipse

Reasons why Java has been selected as Programming Language for Android?

Java is the **official language for Android App Development** and consequently, it is the most used language as well. Android App can be developed by using languages such as Java and Kotlin, still developers prefers java as a programming language for Android App development because of its features such as portability, security etc.Java has much online community for support in case of any problems.

Why Java for Android?

Android is an open source software and Linux-based operating system, mainly designed for Mobile Devices. To manage and control the Android Device the code which is used by the developers is Java. Android Software Development Kit is used mainly to develop Android applications.In order to build Android, the prerequisite which

is needed by the developer is to learn the basics of Java Programming.

The preference of Java

James Gosling developed Java Programming Language in Sun Microsystems in late 1990s. Java has become much popular among different programming languages because of its features. Java is more popular because of its **"Write once and run anywhere"** feature. This is one of the major factors for the success of Java.

In Java different editions are there, out of that the **Java Mobile Edition** was used for creating applications that run on mobile devices. The most important feature of Java is the security, we can say that the programs written in Java are secure, because it runs in a secure environment. Bytecode is the intermediate code which is generated by the system when the Java program is compiled. The JVM that is Java Virtual Machine is used to execute the bytecode.

Java for building Mobile Applications

The **mobile edition of Java** is called Java ME. Most Smart phones and tabs support Java ME, which is based on Java Standard Edition. In order to build and execute applications, the Java Platform Mobile Edition (Java ME) provides much support like **flexibility, protected environment etc.**

The applications that are built using Java ME are **portable and protected.** All the problems like executing applications on low memory, display problem and power problem, are addressed properly if Android Application is developed using Java Mobile Edition. By using different programming languages and different IDE's Android

Application can be built, but the most recommended approach is to use the **Java programming language and the Android SDK.**

JDK (Java Development Kit)

Java Key Features for Mobile Apps

1. Once developed application using Java language, it can be ported in all executable platforms.

2. Java supports all Object Oriented Programming Concepts

3. Lots of Libraries are present which supports the program developers

4. Huge global community with Android support

5. Java programming is easy to learn, understand and execute.

ANDROID STUDIO

What is Android Studio?

Android Studio is the official **Integrated Development Environment (IDE)** for Android app development. The **best environment** to develop Android Applications because it provides all the necessary options to

develop Android Applications in a single unified environment.

Characteristics of Android Studio

1. Visual Layout Editor
2. Fast Emulator
3. Intelligence Code Editor
4. Helps to Build Up App for all devices
5. Support Java and KOTLIN
6. Maven Repository

Visual Layout Editor

By using Layout Editor, the layouts can be built quickly, either by manually typing the code or simply by dragging and dropping the components. The changes to this component can be made easily according to the requirements like adding few more components, or resizing existing components. Within a moment the changes done can be reflected in the Visual Layout Editor.\

Fast Emulator

Emulator is nothing but the virtual mobile device which is used to test the mobile applications. The advantage of using emulator is before deploying the application in the actual device; the developer can change the minimum SDK version and can check in which different mobile devices the developed mobile application is compatible.

When tested applications in Android Virtual device the developer will get the real time applications. It allows you to **test your applications faster** and on different configuration devices like tabs, android phone etc. It helps you to make your

application **development life cycle shorter and more efficient.**

Intelligence Code Editor

Only minimum coding the developer has to write, Android Studio provides semi-automated code, which means only part of the coding the developer has to write. Quick Code Editor is present in Android Studio. It's not necessary that the developer has to wait until the completion of full coding test the application. The intelligence code editor point out the errors easily so that the developer rectifies it easily and concentrate on to develop next part of coding.

When developer starts typing the code the possible methods and properties applicable for that element will be displayed automatically by the software.

Helps to build App for all devices

With the help of Android Studio the developer can develop app for any version or for any screen size of Android device. It also can arouse the various types of features which hardware has like GPS location tracker.

Supports Java and KOTLIN

Android Studio supports both Java and Kotlin, still Java is considered as the official language for Android Application development. Java is a **language without** having any new **restrictions and has various features in it.** The great feature of Java is **it will run in older as well as the newer versions.**

Maven Repository

Android Studio holds Maven repository, within Software Development Kit. Various jar files like Project Jars and Plug-ins are stored in Maven Repository.

Android Studio - IDE

The following are the steps which should be followed to setup an Android Studio – Integrated Development Environment.

Step 1 - System Requirements

Microsoft Windows 10/8/7/Vista/2003 (32 or 64-bit) Mac OS X10.8.5 or higher, up to 10.9

Step 2 - Setup Android Studio Installation

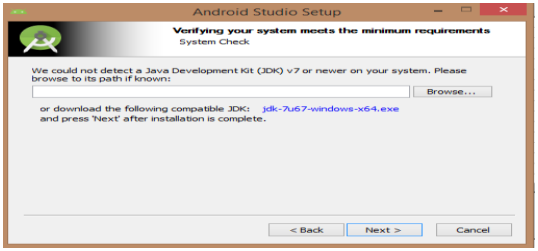
Android Studio.exe file is needed, which can be downloaded for Android site, Before installing Android Studio, Machine requires Java Development Kit that is JDK to be installed.

Step 3:

Verify that the machine has Java Development Kit, then launch the *Android Studio.exe*. The following window will be opened click on Next.

Step 4:

The next step is to locate the path of the JDK which is present in our device. After mentioning JDK path click on next.

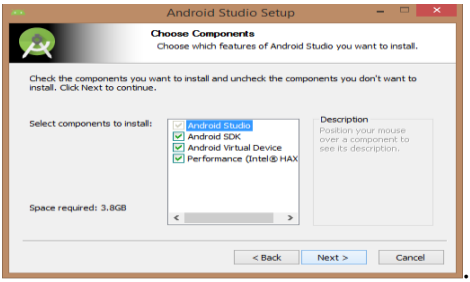


Step 5:

Once JDK path has been specified properly the initiation of JDK to Android SDK will take place, click OK and then click on Next.

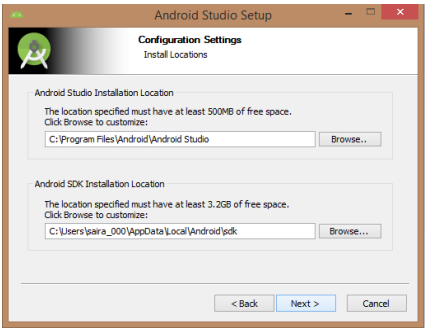
Step 6:

In the Android Studio Setup dialog box the necessary components like Android SDK, Android Virtual Machine and performance (Intel chip) should be selected because all these components are required to develop an Android Application, after checking click on Next.



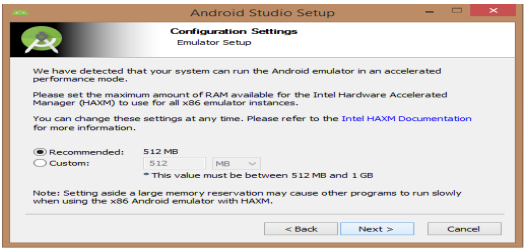
Step 7:

The system will take default location to store Android Studio and SDK, if needed by clicking on Browse button the user can select the space where they want to launch the application. Click on Next.



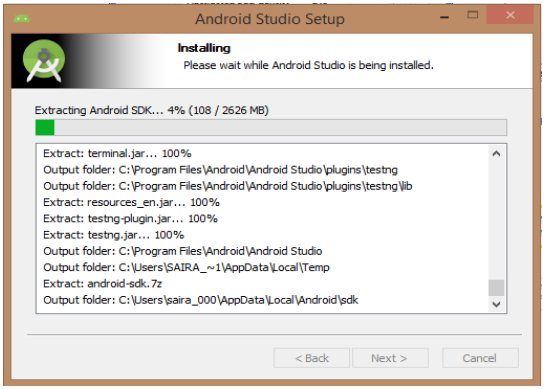
Step 8:

The Android Programs will run in Android Emulator only, it is necessary to specify the RAM space for emulator, by default it will take 512 MB, users can modify the size too if needed, then click on Next.



Step 9:

This is the final step, in this step the system would extract SDK packages into your machine, it will take a little time to finish the task and will take approximately 2626MB of Hard disk space, Click on Next.

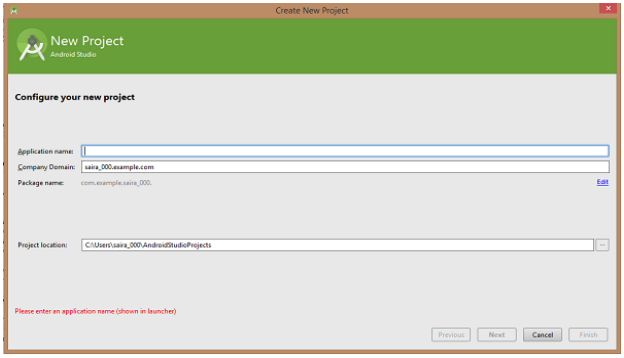


Step 10:

After completing all above steps perfectly, you will get finish button. After clicking on it, the following shown screen will be displayed.

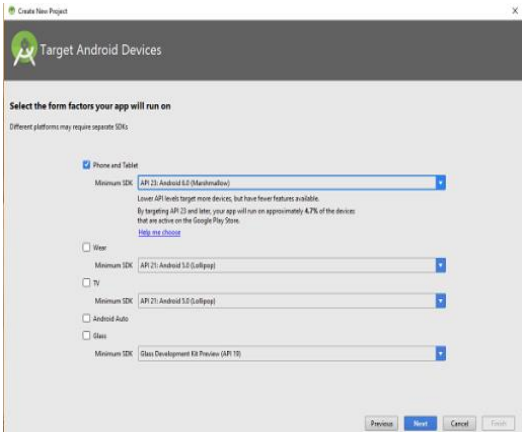
Step 11:

From the above shown dialog box click on Start a New Android Studio Project. Create new project dialog box will be displayed, enter Application name, package information and location where you want your project to be saved.



Step 12:

After entering the details like Application Name and its location etc, the following screen will be displayed, from which the user has to select the Minimum SDK Version, remaining details can be left as default, click on Next.

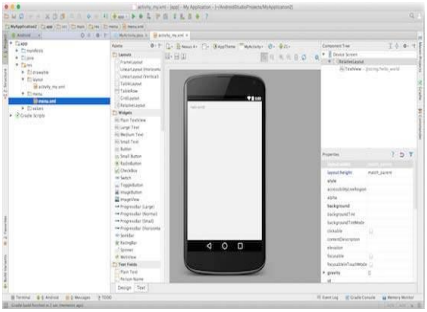


Step 13:

This is the next level, here the user should select the layout corresponding to their application, the blank activity will be selected by default, if needed the user can select the different layout.

Step 14:

Finally Android Studio's Integrated Development Environment will be opened, by using which the developer can start designing the projects.



ECLIPSE

1. What is Eclipse?

2. Few reasons why eclipse will be used for Android App Development.

1. Downloading and Installing the JDK

2. Downloading and Installing Eclipse

3. What is SDK?

4. Downloading and Installing android plugin for Eclipse

5. Configuring the Android plugin for Eclipse

What is Eclipse?

Eclipse is an integrated development environment **(IDE)**. It contains extensible plug-in system and a workspace for customizing the environment. Once after installing plug-ins in eclipse it can be used to write program in different programming languages. **Eclipse** software development kit **(SDK) is free and open-source software,** developers can easily download the software from Google.

Few reasons why Eclipse will be used for Android App Development

1. The Open Handset Alliance has released an **Android plugin for Eclipse,** by using that developers can build Android specific projects, and use the Android Virtual device to compile and test the app.

2. Eclipse is **very easy to use**, learning is easy.

3. This makes Eclipse a very **attractive IDE** for solid Android Application development using Java programming language.

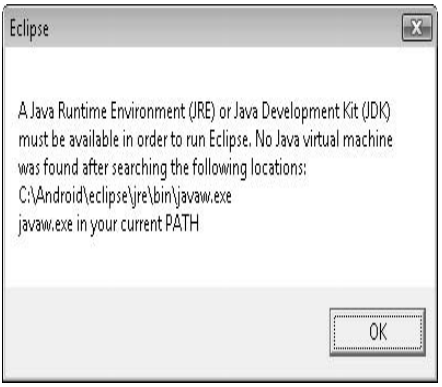
4. The help provided by the Android plugin for Eclipse saves your precious development time, else if Plug-ins are not present then for each and everything the developer has to write the coding. Moreover, without plug-ins the applications will not work properly.

Downloading and Installing the JRE

In order to work with Eclipse, before installing it make sure that the Java Runtime Environment (JRE) is present in your machine. Because Eclipse as an application was written in Java, it **requires the JRE to run.** If the JRE is not installed

or is not detected, the following error dialog box will be shown if the user tries to open the eclipse environment.

Error window if the JRE is not present



Downloading and Installing JDK

Navigate to the Eclipse Downloads page and install the Eclipse of any version as per your application requirement.

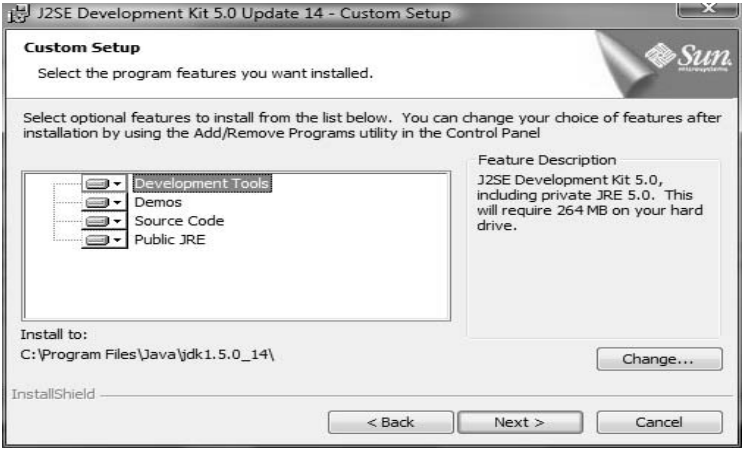
From the SDN that is Sun Developer Network, Downloads page, navigate to the download for the proper JDK. Select and initiate the download

If you are downloading for a Microsoft Windows environment, you will see the notification whether to run the application directly or just to save the application, click Run to begin the installation of the JDK.

If you want **to retain a copy of the JDK package, click Save rather than Run.** However, if you choose to save the JDK, note the location of the SDK**.** Once software has been downloaded go to the download location and execute the package manually so that the software starts its installation.

During the installation process, it is necessary to read the Given License Agreement. After agreeing to the standard License Agreement and clicking Next, you will be able to select your custom setup options.

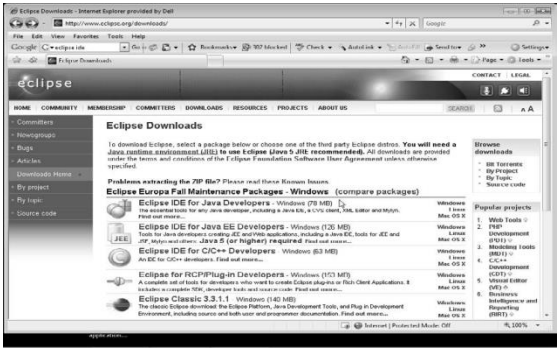
The following illustration shows the Custom Setup screen for the Java JDK.



By default everything will be selected, to keep the process simple, leave the selected default things as such, click on Next, so that the installation continues without any interruption. Once the installation steps have been completed, the final page will be displayed, just click on Finish to complete the installation successfully.

Downloading and Installing Eclipse

Navigate to the Eclipse Downloads page

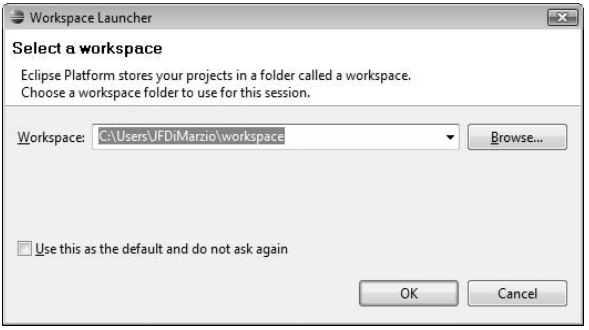


Once the eclipse has been downloaded, the next step is to install it, go to the location where Eclipse has been downloaded in your system.The Eclipse software size is very small, it consumes only limited amount of Storage compared to Android Studio.

Once the eclipse software has been installed by the developer by following the necessary steps, each time when the developer opens the Eclipse the system asks to create new Workspace, it depends on the developer whether to open eclipse using same workspace or with different workspace.

Always it is advisable to launch Eclipse from the same workspace. The following image shows the Eclipse title screen that appears upon start up.

Once the Eclipse installation commences, you will be prompted to create a **default workspace, or folder.** The default path for the workspace is your User directory, as shown in the illustration that follows. **Click on Browse and Navigate to it, to select a new location.** Enter Workspace and click on **"Use this as the default and do not ask again"** Checkbox, finally click **"OK"** button.



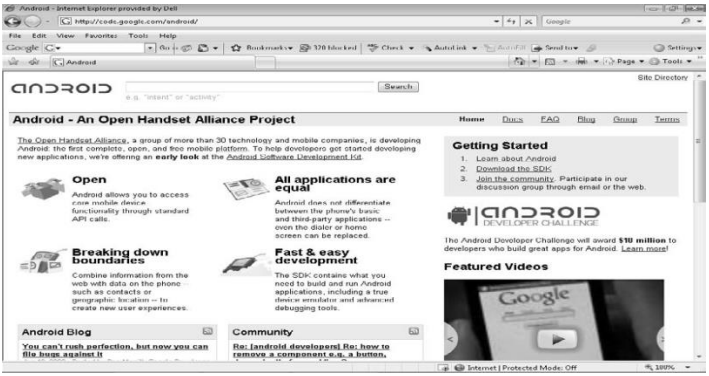
Eclipse installation gets complete here. Next step is to download and install the Android SDK, download and install the Android plugin for Eclipse. Configure eclipse settings.

What is SDK?

SDK stands for **Software Development Kit.** The Android SDK is no different from any other SDK. The Android SDK contains all the different Java code important libraries which are needed to build Android Applications, and the application developed using Android SDK will work only on Android Platform.

The supportive files which is present under Android Software Development Kit is the help files, documentation for assistance, virtual devices and debugging tools to find out and rectify the errors.

The following figure shows the home page for Google Android Development



From the development home page, click **Download theSDK** link under Getting Started. After you agree to the terms of the Android SDK License Agreement, you will see the Download the Android SDK page.

The Android SDK is downloaded in a **79MB** (for Windows) package and it will download fairly quickly. **There is no "setup" or installation process** to speak of for the Android SDK. It is necessary to follow a set of steps to add

the SDK with Eclipse development environment. The first is to get the Android Plug-in for Eclipse.

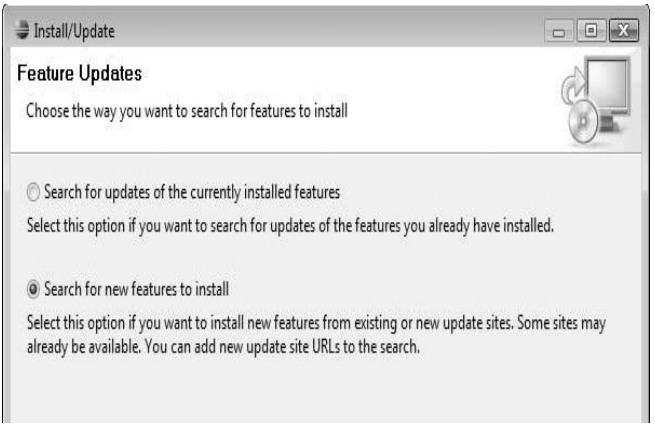
Downloading and Installing the Android Plugin for Eclipse

Follow the steps given below in order to achieve the same.

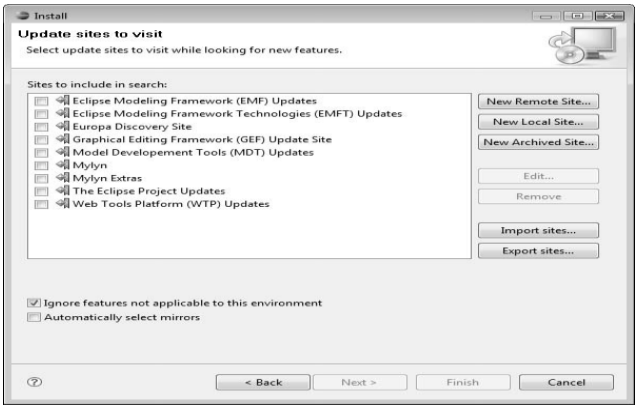
1. Open the Eclipse application. You will **download the Android plugin** for Eclipse from within the Eclipse IDE.

2. Choose Help -> Software Updates -> Find and Install.

3. In the Install/Update window, which allows you to begin the process of downloading and installing any of the plugin that are available to you for Eclipse, click the Search for New Features to Install radio button and then click Next.

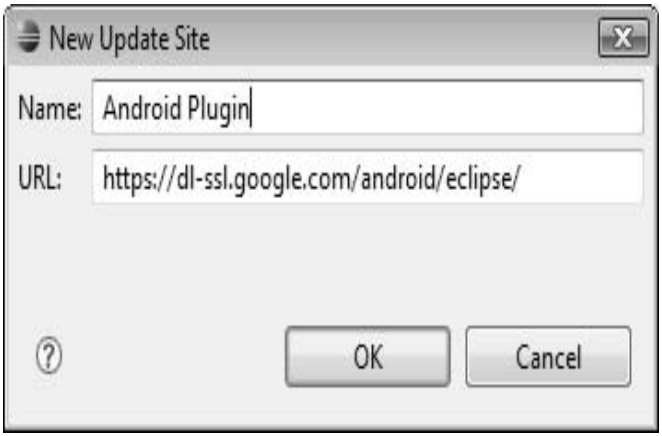


4. To download the Android plugin, you must tell Eclipse where to look for it, so click the New Remote Site button.

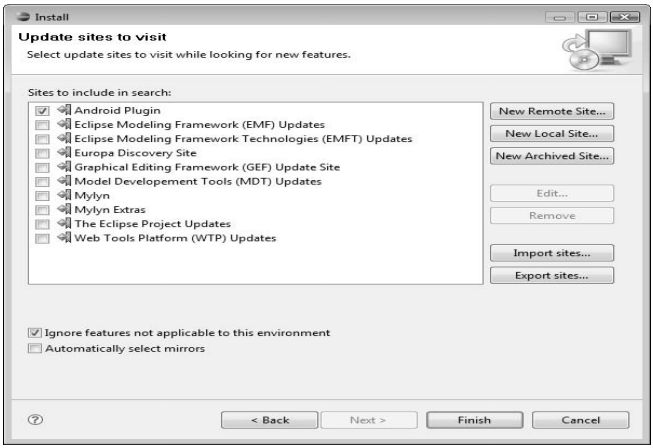


5. In the New Update Site dialog box, enter two pieces of information to continue: **a name for your new site, and its associated URL.**

The name is only for display purposes and does not affect the downloading of the plugin. In the Name field, enter **Android Plugin.** In the URL field**, enter the URL from** which Eclipse will obtain information about the plugin that are available: **Click OK.**

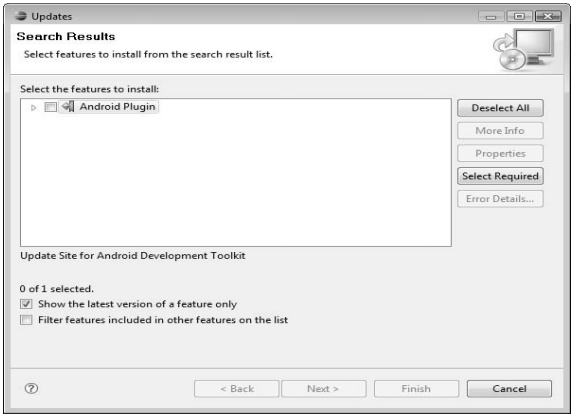


6. A new site named Android Plugin will now be in your list of available sites.

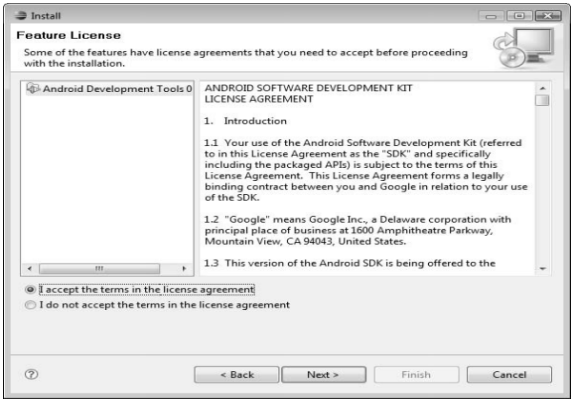


7. Check the check box next to Android Plugin and then click Finish. Eclipse searches the URL associated with the Android Plugin site for any available plugins.

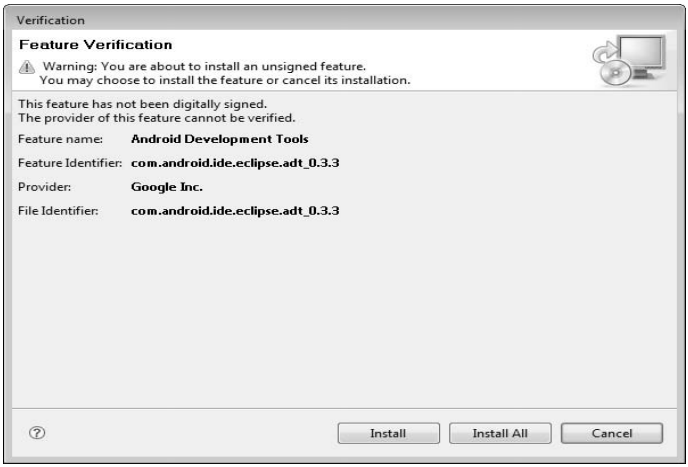
8. On the Search Results page of the Updates window, select the Android Plugin and then click Finish.



9. Accept the licensing agreement for the Android Development Tools andclick Next.



1. On the final plugin installation page, Feature Verification, click Install All to complete the installation of the Android plugin.



Configuring the Android Plugin for Eclipse

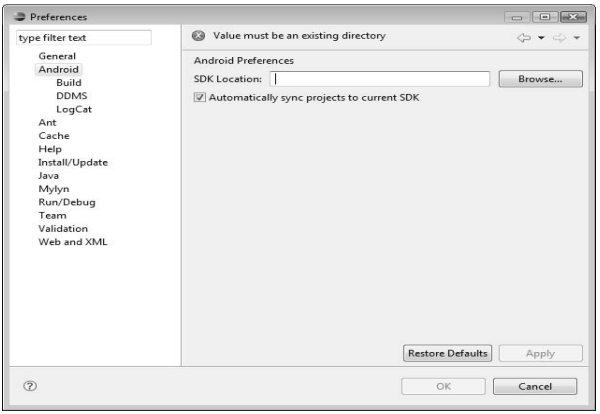
After installing the Android plugin for Eclipse, **restart the application.** Restarting Eclipse will ensure that the program has a chance to reinitialize with the plugin installed. The Android plugin for Eclipse is configured from the **Preferences window** of Eclipse.

Proceed as follows:

1. From the main Eclipse window, choose **Window->Preferences.**

2. In the Preferences window, select Android in the menu on the left. On the right side of the window, click Browse, find the location of the Android SDK on your hard drive, and enter it in the SDK Location field. Eclipse needs this information to be able to access all the tools that are supplied with Android, such as the emulator.

3. Check the Automatically Sync Projects to Current SDK check box and then click Apply.



The JDK, Android SDK, Eclipse, and the Android plugin for Eclipse are now fully configured and ready for development.

VIRTUALIZATION

What is Virtualization (Emulations)?

Types of Emulations

Limitations of Virtualization

Virtualization

Android devices have the ability to emulate (Virtualize). Android is a Mobile Operating System based on Linux Kernel. Android is an open source software designed primarily for touch screen mobiles.

An **emulator is a software** that enables one mobile device to behave like an another mobile device, not only that, with the help of emulator the developer can test the application before deploying that in any physical device.

Types of Emulations

1. Emulation of other Operating Systems
2. Terminal emulation of internal operating system
3. With enhancements via semi-emulation

Emulation of other Operating Systems

In order to emulate other operating systems the compatibility layer is mandatory, the compatibility layer uses either technologies or Application Programming Interface to run the Operating System inside the app container.

Terminal emulation of Internal Operating System

Terminal emulation of the Android device itself is done via emulation.

With enhancements via semi-emulation

Semi-emulation means adding additional packages. Some emulators allow the users to add additional packages. This is known as semi-emulation. With the semi-emulating, some predefined ported packages can be used and installed. This type of semi-emulators does not make use of File System

and they reside and run in the apps own data containers and directories.

Limitations of Virtualization

The limitation which is present in virtualization is because of the compatibility layer. **Lots of limitations are seen in emulation** based apps, as the emulation is making use of compatibility layer, if the compatibility layer doesn't work accurately, then it is not possible to achieve the desired result.

The compatibility layer must work properly and it should provide accurate information then only all the libraries and packages will work exactly as expected like a real operating system. Virtualization requires the compatibility layer or any predefined software it uses, **to have access to many system related and device related information.**

APPLICATION PROGRAMMING INTERFACE (API)

What is API's?

The best API's and libraries for Android Developers

Types of API

Types of API Protocol

What is API's?

API stands for Application Programming Interface. In order to access the tool or database, API is mandatory, API is nothing but a set of instructions. Since android in an open source, a software company develops and releases API in the market, so that the application developers can make use of it and develops the new software products.

The SDK always holds the API inside it. **APK stands for Android Application Package.** After the creation of android application the developers will convert that into an APK format then only it will become possible for them to distribute the app, and then the users install that from Play Store into Android Operating System.

The best APIs and libraries for Android developers

1. Cloud Storage API from CloudRail.
2. **Retrofit** from Square.
3. GSON from Google.
4. **EventBus** from Green Robot.
5. **Android Pay** from Google.
6. In-app Billing from Google Play.

Types of API's

1. Open API's aka Public API's
2. Partner API's
3. Internal API's aka Private API's

Open API's aka Public API's

With minimum restriction the Open API's are available to all developers. The developers can make use of this API to develop their applications. Registration is required to access Public API's. They focus on external users, to access data or services.

Partner API's

Partner API's are not available publicly. In order to access Partner APIsspecific privilege is needed. APIs exposed by/to the strategic business partners.

Internal APIs, aka **private APIs**

Internal API's which is also known as Private API's as the name indicates are highly confidential. It will not become possible for all the developers to access it. They are hidden from external users and only exposed to internal systems.

Internal APIs the developers will use only inside the company to build the product. It is not meant for consumption outside the company but fairly for use across different internal development teams for better efficiency and reuse of services.

Types of API Protocols

1. REST (Representational State Transfer)
2. SOAP (Simple Object Access Protocol)
3. RPC (Remote Procedure Calls)

REST (Representational State Transfer)

REST is one type of protocol mainly used for web services. REST (short for Representational State Transfer). REST APIs are very important for modern web applications.Examples of web services where the REST protocol will be implemented are the Uber, Amazon etc.The REST Protocol must adhere to the following rules.

Uniform Interface—In this the client and server communication will take place, via HTTP (HyperText Transfer Protocol) using URIs (Uniform Resource Identifiers), the client device sends a request and the server responds for that request.

Client-Server—Another important thing is even though the client and server is connected and the server responds to client request. The changes made in the client should not affect the server and vice versa.

Cache—The client should possess the ability to cache the responses from the server as this improves the user experience by making them faster and more efficient. If the application communication should take place over the network then the REST has been chosen as the preferred standard. Compared to Simple Object Access Protocol, REST is simple, and it fully leverages all the standards that power the World Wide Web. REST, it **allows for a loosely coupled layered architecture**, which makes possible to update and maintain REST protocol easily.

SOAP (simple object access protocol)

SOAP (simple object access protocol) is a well-established protocol similar to REST. It is a type of Web API. It has been started from the late 1990s. SOAP was the first to regiment the way applications should use network connections to manage services. But SOAP comes with strict rules, very resource demanding.

Most developers prefer REST Protocol compared to SOAP, because both the protocols are used to work with applications that communicate over network, but the SOAP has more restrictions compared to REST.

RPC (Remote Procedure Call)

RPC (Remote Procedure Call)protocols are the simplest and the oldest protocol used by API's. Even though both the protocols are used for applications which communicate over network. Some differences exist between REST and SOAP. The main difference is REST is loosely coupled so updation can be done very easily but RPC APIs **are very tightly coupled,** so this makes it difficult to maintain or update them.

Since RPC is tightly coupled, when one part of software is edited chances are there that it may affect some other part. While making changes, developer must go through various RPC documents to know clearly that if certain modifications are done then its fine or it will affect any part, then updation should be done accordingly.

ANDROID TOOLS

In order to develop Android Application the main thing which is needed is Android SDK, we can say that Android development **starts with the Android SDK** (Software Development Kit). By using different programming languages like Java, Kotlin etc and by using different IDEs like Android Studio and Eclipse, Android Applications will be developed, but **the SDK is a constant.**

In order to give assurance that the process goes smoothly for developing an app **SDK** is mandatory, not only that the SDK also provides a selection of different tools necessary to build an Android Application.

The Android SDK can be wrecked down into several components. These include:

1. SDK tools
2. Android Emulator
3. Build Tools
4. Platform Tools

SDK Tools

SDK tools are generally **platform independent** which means no matter either as a developer you may use Android Studio or Eclipse to develop an Android Application, SDK is mandatory because it provide lots of supportive tools

necessary to build an Android Application. All the supportive tools get installed automatically, when Android SDK is installed.

The following is the list of SDK tools:

1.Android

This is one of the very important tool of SDK, Android tool lets you manage Projects, Android Virtual Devices etc.

2. DDMS

DDMS stands for Distributed Debug Monitor Server. This tool is mainly used to debug Android Applications.

3. Draw 9-Patch

By using this tool the developer can easily create a graphics in their applications, using a **WYSIWYG** editor that is **What You See Is What You Get.**

1. Emulator

Emulator is nothing but a virtual device, by using emulator the developer can test their applications without using a physical device.

5. mksdcard

This tool is used to create an external sdcard storage which can be used with the emulator.

6. Proguard

This code is mainly used to remove the dead code, the dead code is nothing but the unused code, this tool removes such code.

7. SQLite

SQLite is the inbuilt database present in all the mobile devices. This SQLite tool is used to access the data present in SQLite.

8. ADB

ADB stands for Android Debug Bridge, if the android application is tested by using virtual device or if the developer is deploying the app in actual physical device, then this ADB is very useful. The ADB is also used to communicate with Emulator Instance.

Android Emulator

Without the availability of the actual device, if the developer wants to test the Android App, then the emulator is used, which is nothing but a virtual device. While selecting the Emulator for testing and deploying the app, the SDKs Android Virtual Device Manager will provide different options like version of the device, screen size, RAM Size of the device etc which is also known as the device specifications. When the developer wants to test the Android application, if needed each time they can select the Virtual device with different specifications.

Build Tools

This is one important tool which is used to build Android apps. This includes the **zipalign tool** for instance, which optimizes the app to use minimal memory when running. Not only that, it also makes the apps use less memory while running but also alerts the user if certain apps consume more memory.

Platform Tools

The Platform tools are more specifically suited to the version of Android that you want to target. Always it is best to install the latest platform tools. It is necessary that the developer needs to update the Platform Tools constantly.

The tools should be backwards compatible, which means that you the software must still be able to support older versions of Android.

DEBUGGING WITH DDMS

DDMS stands for Distributed Debug Monitor Server, it is the component which provides many services on the device. The different services provided by the DDMS is the message formation, capturing screenshot, finding internal threads and file systems etc. If the developer tests the application by using the Virtual device then the DDMS will provide support, and also when the developer tries to deploy the application in the physical device then too the DDMS will provide support.

If physical device and virtual device both are selected simultaneously by the developer to tests their application, then DDMS by default will select the Virtual device.

Running DDMS

In order to run DDMS from Android Studio, select the following menu and sub-menu, **Tools->Android->Android device Monitor**.

How DDMS Works?

The DDMS will acts as an intermediator between the Integrated Development Environment and the device which is used to run the application. Different applications present in Android use different process and each app use different

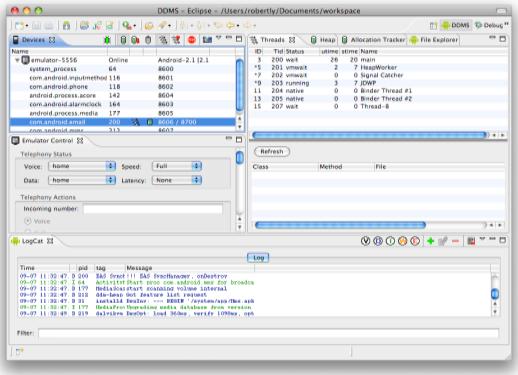
virtual machine, and each process listens for a debugger on a different port.

DDMS connects to ADB that is Android Debug Bridge, and it starts a main service that is device monitoring and it will notify DDMS when a device has connected or disconnected.

Debugging with DDMS

Whatever IDE the developer may use to develop Android Application, either it may be Eclipse or Android Studio, the DDMS is integrated in both IDEs already.The DDMS provides a number of different features, the different services provided by DDMS are interacting with emulators and physical device, debugging applications.

The developer can select the specific process to run on an emulator or on a physical device, and then to start debugging click on the Debug Button or even the developer can directly click on the Run button to run the application. Specific process can be selected either to run an emulator or to run on a physical device. The DDMS will work only with IDEs not as Standalone Application.



The five features of DDMS are as follows:

1. Task management
2. File management
3. Emulator interaction
4. Logging
5. Screen captures

Using LogCat

LogCat is integrated into DDMS, and outputs the messages. The LogCat feature of DDMS are as follows:

1. Verbose
2. Debug
3. Info
4. Warn
5. Error

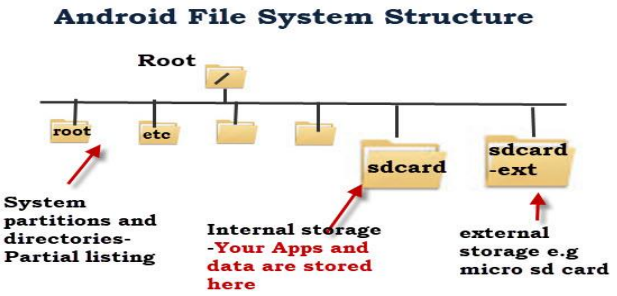
ANDROID FILE SYSTEM

The Android Operating System is the most popularly used platform for smart phones. Apart from the basic feature which is present in Android device such as Calls, SMS, Browsing etc, it is also necessary for the user to know the internal structure of the Android file system.

Just like in Windows files data are organized in the form of files and directories, Android system too uses several 6 different partitions to organize files and folders. Each partition has its own functions and features.

File System

In different Android devices like Android Phones, Tabs there are mainly 6 different partitions are present. Partition**s** **differs from Model to Model.** But logically minimum 6 partitions will be found in any Android devices.



boot – This is very important partition, this partition holds Kernel, RAM disk etc which is required by the phone to boot when powered on.

system –Once the booting process done, the system partition gets activated and houses all the operating system files such as Android User Interface and pre-installed applications for proper functioning.

recovery – The recovery partition allows the user to backup and restore other partitions.

data– Data as the name indicates this partition saves all the users data such as data ranging from contacts, messages, apps, music and different files etc.

cache– The frequently used data and applications components will be present here. The users frequency of using data and apps changes constantly, according to that the cache partition will be updated automatically.

misc– The following system settings information such as USB configuration, carrier ID and other hardware settings information will be present in this part.

View of File System Layout

Android's file system layout is not identical to your Personal Computer. In the following three different manners, the Android File System layout will be divided.

1. Device Storage
2. Portable SD Card
3. Device Root

Device Storage

As the name indicates clearly this is device storage space where the user will store and retrieve the data. There is no restriction, the user can delete, modify or add contents which they want to store in this part.

When we install apps from App Store/Play Store, that apps too will take certain space to store the supportive files, while downloading apps certain apps requests from the user the credentials in order to download the app, the credentials such as user id and password information the app will not store in this space.

Portable SD Card

Most of the Android devices will have SD Card slots. It is also possible to transfer the contents of SD Card to system or to any other device, load files into it and then plug it into your device.

Device Root

Android device also has a special file system where its operating system files, installed applications and sensitive application data are stored. It will not become possible for the user to modify these files. File manager apps cannot update or delete this file system for security reasons.

WORKING WITH EMULATOR AND SMART DEVICES

What is Android Emulator?

If the developer wants to test the application before deploying it in a physical device, they can make use of Android Emulator. The developer can create virtual device, by mentioning the version, RAM size, screen size etc.

If the developer wants they can create different virtual devices to test different applications or the developer can use the same virtual device to test any number of applications. The main advantage of using emulator is without using physical device the developer can test applications with emulator.

Working with Android Emulator

First point to be considered here is, if developer has created one Android Application and if they want to test it, practically it is not possible for them to test it in many physical devices, but by setting the configuration of physical

devices in virtual device the developer can test their app in any number of virtual devices which simulates the physical device. The capability which the physical device provides all those capabilities the virtual device too will provide.

The functionalities which physical device does can be simulated into virtual devices, functionalitiessuch as text messaging, rotations, phone call, performing browsing, accessing Google Play Store etc. Testing apps on the virtual device is in some ways faster and easier than doing so on a physical device. For example, Data transfer to the emulator can be done faster than transferring data to a physical device.

The same configuration virtual devices cannot work for Android Phone, Tablet and Android TV. Emulator comes with different predefined configuration for Phones, Tabs etc out of which, the developer have to select the needed configuration and type of device for testing their apps.

The Android Emulator has additional requirements beyond the basic system requirements for Android Studio, which are described below:

SDK Tools 26.1.1 or higher 64-bit processor

Windows: CPU with unrestricted guest support

The different functionalities which are performed by theemulator are as follows:

1. Creating AVD
2. Changing Orientation
3. Sending SMS through Emulator
4. Emulator making calls
5. Emulator Transferring files

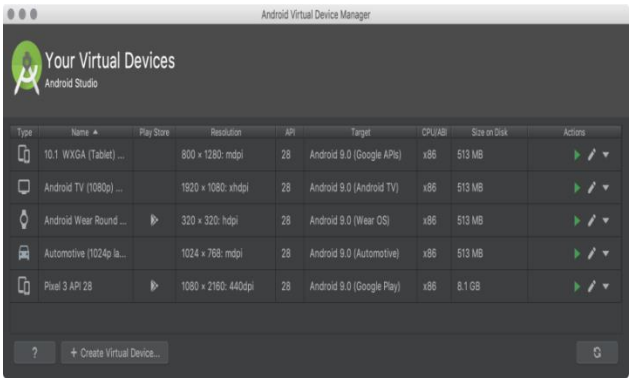
Creating AVD

In order to run the application in an emulator, the first step is to create AVD. Once if the user has installed Android Studio or Eclipse, then the AVD Manager can be launched from both of these applications, it depends on the user whether they want to use Android Studio or Eclipse for the application development. In order to launch AVD in Android Studio, the developer should follow the given steps.

Select Tools -> AVD Manager. (OR)

Click AVD Manager in the toolbar.

The following Android Virtual Device Manager dialog box will be displayed. By clicking on the "Create Virtual Device" button, the developer can create a virtual device that is the emulator to run the application.



Changing Orientation

By default, when virtual device is launched then it will be in vertical form. By pressing Ctrl+F11 Key, the orientation of the virtual device can be changed. Only two types of orientation are supported, either it may be horizontal or vertical.

Example

First when the emulator is launched, then the view will be like how its shown below.

Once it is launched, press Ctrl+F11 key to change its orientation from vertical to horizontal if needed.

Example Android Program for Sending SMS through Virtual Device

Normally by using the default messaging app present in our device we will send message, but it is also possible in Android to develop our own messaging app.

The three important files which is used to develop Android Application is MainActivity.java, activitymain.xml and AndroidManifest.xml. The following coding shows how

to send a message using messaging app developed by the user by writing simple coding in the respective files.

CODING:

MainActivity.java

```
package com.example.smspermissionpgm;
import android.os.Bundle;
import android.support.v7.app.ActionBarActivity;
import android.app.PendingIntent;
import android.content.Intent;
import android.telephony.SmsManager;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
public class MainActivity extends ActionBarActivity
{
EditText mobileno,message;
Button sendsms;
@Override
protected void onCreate(Bundle savedInstanceState)
{
```

```
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
mobileno=(EditText)findViewById(R.id.editText1);
message=(EditText)findViewById(R.id.editText2);
sendsms=(Button)findViewById(R.id.button1);
sendsms.setOnClickListener(new OnClickListener()
{
@Override
public void onClick(View arg0)
 {
String no=mobileno.getText().toString();
String msg=message.getText().toString();
//Getting intent and PendingIntent instance
Intent intent=new Intent(getApplicationContext(),MainActivity.class);
PendingIntent pi=PendingIntent.getActivity(getApplicationContext(), 0, intent,0);
//Get the SmsManager instance and call the sendTextMessage method to send message
SmsManager sms=SmsManager.getDefault();
sms.sendTextMessage(no, null, msg, pi,null);
Toast.makeText(getApplicationContext(), "Message Sent successfully!",
Toast.LENGTH_LONG).show();
```

```
}
});
}
@Override
public boolean onCreateOptionsMenu(Menu menu)
 {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
@Override
public boolean onOptionsItemSelected(MenuItem item)
 {
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId();
if (id == R.id.action_settings)
{
return true;
}
return super.onOptionsItemSelected(item);
}
```

}

activity_main.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.smspermissionpgm.MainActivit
y" >
<EditText
android:id="@+id/editText1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentRight="true"
 android:layout_alignParentTop="true"
android:layout_marginRight="20dp"
android:ems="10" />
<EditText
android:id="@+id/editText2"
android:layout_width="wrap_content"
```

```
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText1"
android:layout_below="@+id/editText1"
android:layout_marginTop="26dp"
android:ems="10"
android:inputType="textMultiLine" />
<TextView
android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/editText1"
    android:layout_alignBottom="@+id/editText1"
    android:layout_toLeftOf="@+id/editText1"
    android:text="Mobile No:" />
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/editText2"
    android:layout_alignBottom="@+id/editText2"
    android:layout_alignLeft="@+id/textView1"
    android:text="Message:" />
<Button
    android:id="@+id/button1"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/editText2"
    android:layout_below="@+id/editText2"
    android:layout_marginLeft="34dp"
    android:layout_marginTop="48dp"
    android:text="Send SMS" />
</RelativeLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.smspermissionpgm"
android:versionCode="1"
android:versionName="1.0">
<uses-permission
android:name="android.permission.SEND_SMS"/>
<uses-permission
android:name="android.permission.RECEIVE_SMS"/>
<uses-sdk
android:minSdkVersion="8"
android:targetSdkVersion="18" />
<application
android:allowBackup="true"
```

android:label="@string/app_name"

android:theme="@style/AppTheme">

<activity

android:name=".MainActivity"

android:label="@string/app_name">

<intent-filter>

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

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

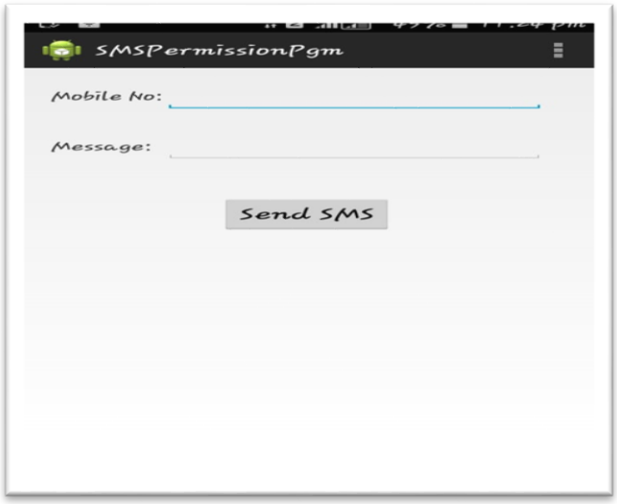
</intent-filter>

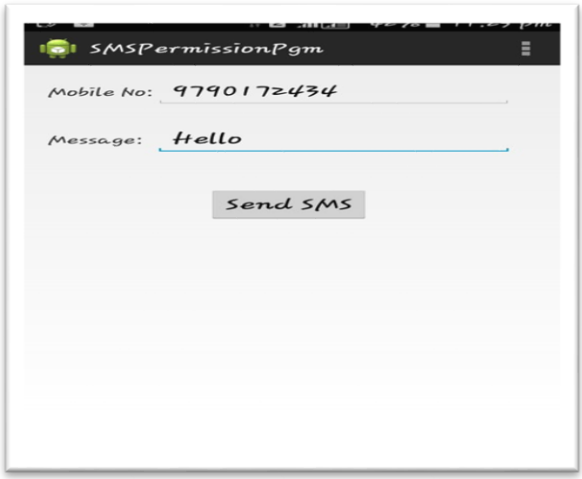
</activity>

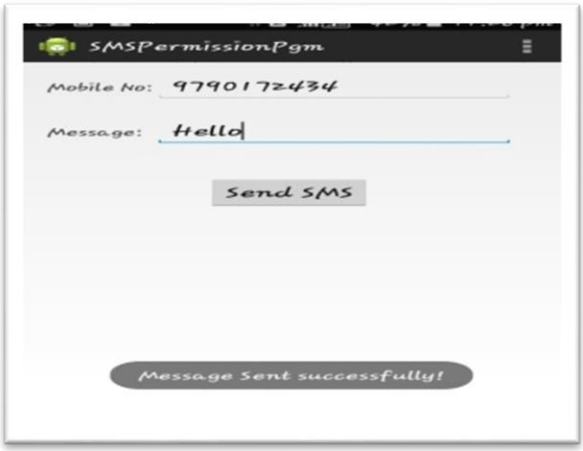
</application>

</manifest>

OUTPUT







Emulator Making Call

It is also possible in Android to develop our own Phone Call app, and make a call through emulator. In order to accomplish this, the following coding shoud be written in respective files.

The three important files which are used to develop Android Application is MainActivity.java, activitymain.xml

and AndroidManifest.xml. The following coding shows how to make a call through the emulator, by writing simple coding in the respective files, and also by doing certain settings in the device.

CODING:

MainActivity.java

```
package com.example.callpermission;

import android.net.Uri;

import android.os.Bundle;

import android.support.v7.app.ActionBarActivity;

import android.app.Activity;

import android.content.Intent;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

import android.widget.EditText;

publicclass MainActivity extends ActionBarActivity

{

EditText edittext1;

Button button1;

@Override

protectedvoid onCreate(Bundle savedInstanceState)
```

```
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
edittext1=(EditText)findViewById(R.id.editText1);
button1=(Button)findViewById(R.id.button1);
button1.setOnClickListener(new OnClickListener()
{

@Override
publicvoid onClick(View arg0)
 {
String number=edittext1.getText().toString();
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:"+number));
startActivity(callIntent);
}
});
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
```

returntrue;

}

@Override

publicboolean onOptionsItemSelected(MenuItem item)

{

// Handle action bar item clicks here. The action bar will

// automatically handle clicks on the Home/Up button, so long

// as you specify a parent activity in AndroidManifest.xml.

int id = item.getItemId();

if (id == R.id.action_settings)

{

returntrue;

}

returnsuper.onOptionsItemSelected(item);

}

}

activity_main.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"
android:layout_width="match_parent"

android:layout_height="match_parent">

```
<TextView
android:id="@+id/fstTxt"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:layout_marginTop="150dp"
android:text="Mobile No"/>
<EditText
android:id="@+id/editText1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:ems="10">
</EditText>
<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:text="Call" />
</LinearLayout>
```

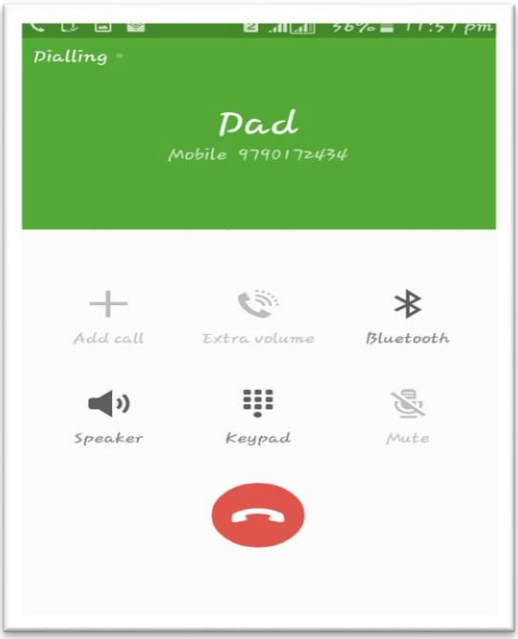
AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.example.callpermission"
android:versionCode="1"
android:versionName="1.0">
<uses-permission
android:name="android.permission.CALL_PHONE" />
<uses-sdk
android:minSdkVersion="8"
android:targetSdkVersion="18" />
<application
android:allowBackup="true"
android:icon="@drawable/ic_launcher"
android:label="@string/app_name"
android:theme="@style/AppTheme">
<activity
android:name=".MainActivity"
android:label="@string/app_name">
<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category
android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</activity>
</application>
```

</manifest>

OUTPUT:



Emulator Commands

The following are the different emulator commands and its description:

1.Home – When in Virtual device if this button is clicked, then the virtual device will take you to the main screen.

2. F3 – This key brings out the call log.

3. F4 – This key is used to end a call.

4. F5 – This key is used to get the Search screen.

5. F7 – This key will acts like a power button.

6. Ctrl+F5 – This shortcut key is used to increase the ring volume up

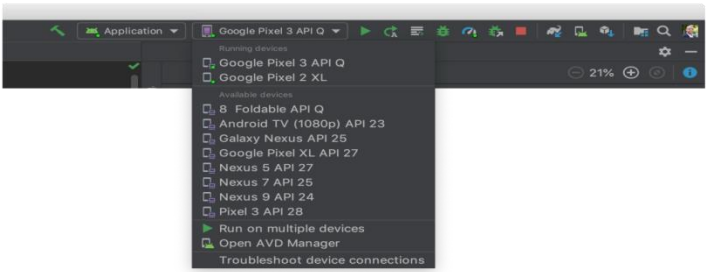
7. Ctrl+F6 – This shortcut key is used to decrease the ring volume down

Steps to be followed to run an app in the Android Emulator:

1. The first step is to create a AVD that is an Android Virtual Device in the Android Studio.

2. In order to do that, in the toolbar select the AVD that you want to run your application from the available options of virtual device which will be displayed in the drop down list form.

3. After selecting the exact virtual device, select app that you want to run and then Click on **Run.**



ANATOMY OF ANDROID APPLICATION (A BASIC ANDROID APP)

Android Applications will be present at the top layer. The applications created by the user will be deployed in this layer only. Examples of different Android applications are the **Contacts Books, Browser, Games etc.**

4 Application Components

The following is the four Application components which is used to build an Android Application. They are

1. Activities
2. Services
3. Broadcast Receivers
4. Content Providers

Activities

Activities describe, how and what are the different activities which the user performs by interacting with the components present in an Application.

Services

The application developed by the developer will not work all alone, it needs some supportive files, and the services provide the background processing associated with an application.

Broadcast Receivers

The main work of broadcast receiver is to handle the communication between the Operating System and applications.

Content Providers

In order to handle data and database management issues this component is very useful.

Activity

An activity represents a single screen with a user interface, in-short Activity performs actions on the screen.For example, any application which you install as a user from App Store/Play Store performs different activities. Specifically if we take mail then we might have one activity that shows a list

of new emails, activity to compose an email, and another activity for reading emails etc.

An activity is implemented as a subclass of Activity class as follows

public class MainActivity extends Activity

{

Program Statements

}

Services

A service is an Application component that runs in the background to perform long-running operations.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.

In short the services component makes sure that the apps with which the user may working, it may be any number of apps, it will make sure that all apps should run concurrently without any interruption and also one app should not affect working of another app.The services component provides the necessary service to all the apps.

A service is implemented as a subclass of Service class as follows

public class ServiceEg extends Service

{

Program Statements

}

Broadcast Receivers

The work of the Broadcast Receivers is to simply respond to broadcast messages, it may be from other applications or it may be from the Operating System itself.

When one application downloads data it will be useful for other application also, in such case one application will initiate broadcast to other applications to let them know that some data has been downloaded to the device and it is available for them too to use.

Content Provider

The main work of Content Provider to is to provide data from one application to another application on request. The ContentResolver Class handles such data transfers. The data may get store either at File System, or Database the content resolver will take care of all the data.

A content provider is implemented as a subclass of ContentProvider class and must implement a standard set of APIs that enable other applications to perform transactions.

```
public class MyContentProviderExample extends ContentProvider
{
public void onCreate()
{
}
}
```

Apart from the four above mentioned components, the Android Application has some additional components. They are:

1. Fragments

The fragment as the name indicates it represents a portion of User Interface in an Activity.

2. Views

The different User Interface elements which comes under views are the Buttons, TextFields, Labels, CheckBoxes etc with which the user interacts to work with an application.

3. Layouts

The main work of the layout is to control the screen format and the appearance of the views. Different types of layouts are present in Android, which the user can use to build an Application.

4.Intents

Android has different components, the main work of Intent is to send messages from one component to another component.

5.Manifest

The AndroidManifest.xml file contains the configuration file for different applications. The Manifest file will be present, with all the applications.

Details of files and Folders present in Android Application

1.Java

By default this file is named as MainActivity.java file. This file contains the important coding which is needed to run your Android Project. It is possible to rename this file, another possibility is to create any number of java files for your Android Application.

2.res/drawable-hdpi

This directory contains the drawable objects that are designed for high-quality screens.

3.res/layout

This is the directory for files, this directory defines your apps user interface.

4.res/values

This is one directory which contains various XML files, the details which are present in the XML file and the collection of resources, strings and colour definitions.

5.Build.gradle

This file will be generated automatically. The different details which is present inside this file are the compileSdkVersion, buildToolsVersion, applicationId, minSdkVersion, targetSdkVersion, versionName and versionCode etc.

ANDROID ACTIVITIES

What is Activity?

Activity Life Cycle

Life Cycle Methods

Creating Activity

What is Activity?

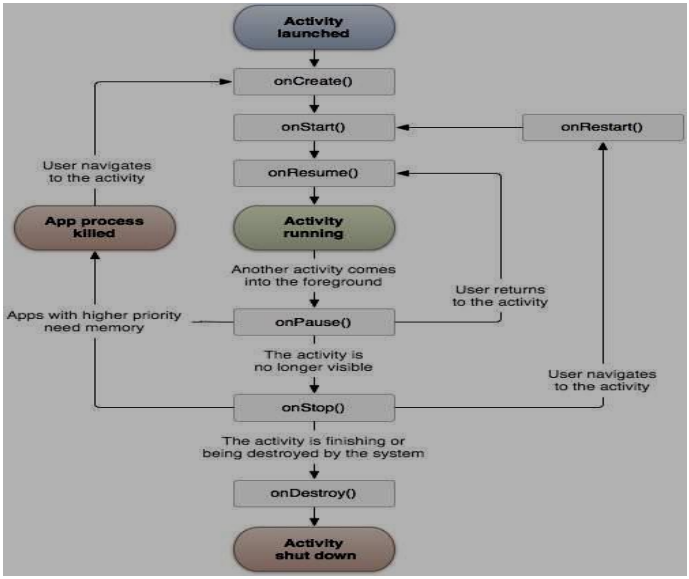
An activity is the single screen which is present in Android. All the User Interface components or widgets are placed according to program requirement in a single screen with the help of Activity. In programming languages like C,C++ and Java the program starts with main() function. In

Android the main program starts within an Activity, by calling the onCreate() method.

Activity Life Cycle

The Activity has different Life Cycle, the transition will take from one activity to another activity. The Activity class provides seven methods (callbacks)

1. onCreate()
2. onStart()
3. onResume()
4. onPause()
5. onStop()
6. onDestroy()
7. onRestart()



Activity Life Cycle Diagram

Activity Lifecycle Methods and its description

1. onCreate()

When the Activity is first created then this method is called. Basic application startup activity is performed by this method, for the entire life time of the activity only once this application startup activity will take place.

2.onStart()

When the activity becomes visible to the user, then this particular method will be called by the Activity class. This method makes the activity visible to the user. When this method is called the activity comes foreground and it will become possible for the user to interact with the app.

In this method only the app will initialize the code, the code is mainly used to maintain User Interface and the different activities associated with it. It will not take much time for the system to complete onStart() method.

3.onResume()

When the user starts working with the application, then this method is called by the system. This is the state in which the app interacts with the user. The app stays in this state for a long time, until some distraction the app gets from the user, for example while the user starts another app, or if the user gets phone call, or if the device screen turns off.

4.onPause()

When the activity is paused then it will not receive any user input and also it will not execute any code. If the user is leaving the activity then this method is called, it doesn't mean that the activity has been destroyed, it indicates that the user has started working with another activity. When the activity is

in paused state, then it can be assumed that it will resume back shortly.

5.onStop()

This method is called when the Activity is not visible to the user. The possibility when the system will call this method means when the newly launched activity covers the entire screen, or if the process has completed its execution and about to terminate then the system will call this method.

6.onDestroy()

The system calls this method before it destroys the Activity. The possibility when the system will call this method means when the Activity has finished or if the system temporarily destroys the activity due to change in the configuration.

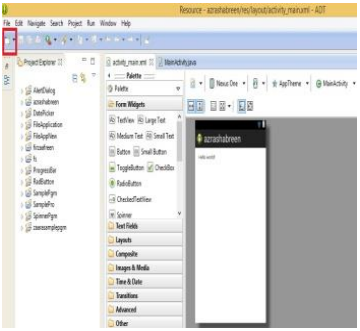
7.onRestart()

After stopping if the Activity starts again then this Restart method is called.

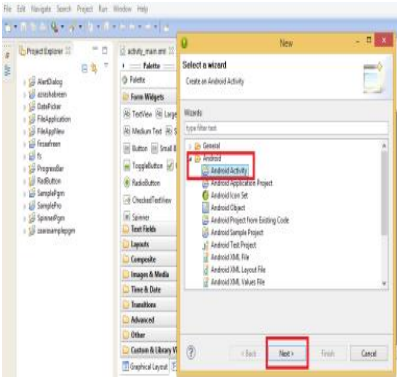
Creating Activity

To create a new Activity using Eclipse

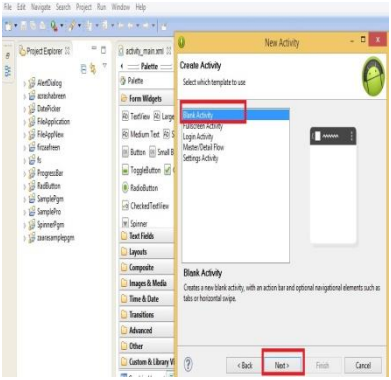
Step 1: Click New in the Toolbar



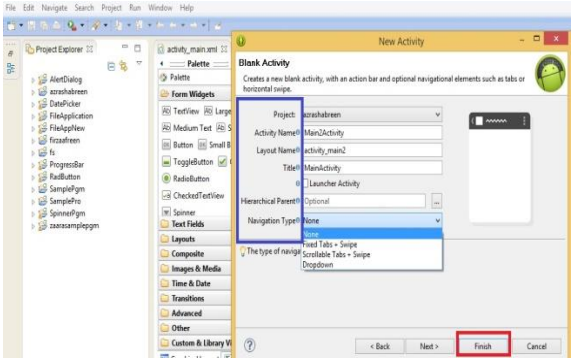
Step 2: In the window that appears open the Android Folder and select Android Activity and click Next



Step 3: Select Blank Activity and click Next



Step 4: Fill in the Activity Details and Click on Finish



INTENT AND INTENT FILTERS

What is Intent?

Difference between Intent and Intent Filter

Types of Intent

Building an Intent

Primary information's present in Intent

Important system generated events present in Intents class

What is Intent?

A messaging object is otherwise known as Intent. In order to request for an action this messaging object is used. Intent objects can be used to communicate with the components, either the components may be present in same or different applications.

In order to work with an Intent, the required header file is, android.content.Intent, as the intents are the object of this type. The intent should be used to start an Activity, via the startActivity() method.

Android Intent can be passed between components such as Activities, Services, Broadcast Receivers and Content Providers. Even if the component is not the part of the application, still it will become possible to trigger that with the help of Intent object. For example selecting photo from one application and returning back to another application to work with that selected photo.

Difference between Intent and Intent Filter

An intent is an object which holds two information, one is app activity and the other is it's URI (Uniform Resource Identifier). Activity will be started using startActivity() method, whereas the IntentFilter can fetch the information about the activity.

Android intents are mainly used to:

1. Start the service
2. Launch an activity
3. Display a web page
4. Display a list of contacts
5. Broadcast a message
6. Dial a phone call etc.

Types of Intents

1. Explicit Intents
2. Implicit Intents

Explicit Intent

Explicit Intent is mainly used to specify the component. Explicit intent provides the information about the external class to be invoked.In more simple words, explicit

intent call another activity in android.From one activity to another activity the information will be passed using an explicit intent.

Intent a = new Intent(getApplicationContext(), ActivitySecond.class);

startActivity(a);

Implicit Intent

Implicit Intent is opposite to an explicit intent. Here, the implicit intent is not used to specify the component. Implicit Intent provides information of available components which is provided by the system to be invoked. The following coding line can be written to invoke an implicit intent:

Intent impint=new Intent(Intent.ACTION_VIEW);

impint.setData(Uri.parse("http://www.google.com"));

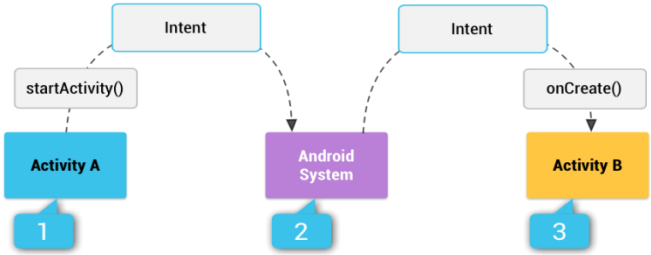
startActivity(impint);

How the system delivers the implicit intent to start another Activity:

[1] *Activity A* describes an action to be performed and creates an Intent for it and this is passed to startActivity() method.

[2]The main work of Android System is to search all apps for an Intent Filter, through which it will try to find out an exact Intent.

[3]When an exact matching Intent has been found, the system starts the Activity B by calling onCreate() method.



Building Intent

The different information which the Intent object holds is mainly used to determine which component to start such as the component name, it's category, not only that it also holds the information about the receipt component name and what action to be performed on that component etc.

The basic information present in an Intent is the following:

1. Component Name
2. Action
3. Category
4. Extras

Component Name

Gives the information about which component to start such as the name of the component. If the component name is not present then the intent is implicit, in such case the system will decide which component should receive an action based on the other information.

Action

As the name indicates it specifies string which tells the generic action to be performed.

Category

This category holds a string which contains the additional information about the component which the intent is going to handle. There are different numbers of category descriptions that can be placed in intent, but most intent does not require a category.

Here are some common categories:

CATEGORY_BROWSABLE

As the name indicates, here the use of web browser will be done. The intended activity allows itself to be started by a web browser to display data.

CATEGORY_LAUNCHER

This activity is the initial activity of a task; it may be any type of task and is listed in the system's application launcher.

Extras

This extras contains the Key-value pairs which contains additional information required to complete the requested action.

Some actions use URIs that is Uniform Resource Identifier and some actions use particular extras. putExtra() method is used to add extra data.

System generated Intents and it's description

1. android.intent.action.BATTERY_CHANGED

This intent holds different information about battery like it's charging state and level etc.

2. android.intent.action.BATTERY_LOW

This intent indicates the low battery condition of the device.

3. android.intent.action.BATTERY_OKAY

This intent indicates the device battery is ok now after low.

4.android.intent.action.BOOT_COMPLETED

After the device completes booting, this intent is broadcasted once.

5.android.intent.action.CALL

This intent performs a call to someone specified by the data.

6.android.intent.action.CALL_BUTTON

This intent performs an action, if the user uses dialler or other appropriate user interface to make a call by pressing the "call" button.

7.android.intent.action.DATE_CHANGED

This intent perform action that is changing of date and indicates that the date has changed.

8.android.intent.action.TIIME_CHANGED

This intent perform action that is changing of time and indicates that the time has changed.

9.android.intent.action.REBOOT

After the device completes rebooting, this intent is broadcasted.

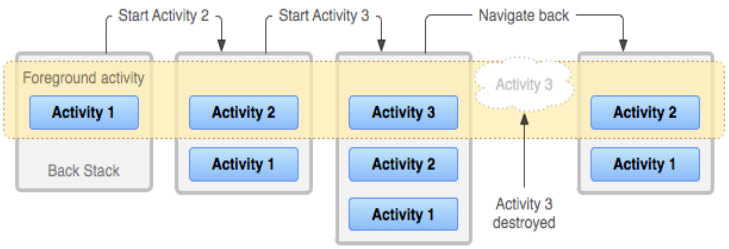
ACTIVITY STACK

An activity is nothing but a collection of task with which the user interacts in order to perform certain job. The user will perform different tasks, then it is arranged in the stack, in the order in which in which it is opened.

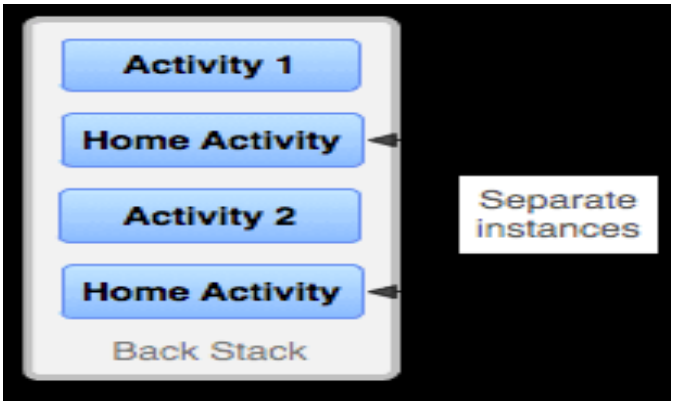
For example, with email app the user performs different activity. One activity is showing lists of new messages. When user selects a message, a new activity will be opened, so that the user can view the selected message. New activity will be added to the stack. If the user wants to come out of that app, then the new activity which is opened recently is popped off from the stack.

When the new activity is started, then it is pushed on the top of the stack and take focus. The old activity which is present in the stack already will be stopped. When certain activity is stopped, the system retains the current state of its user interface.When the user presses the back button, the current activity which is present at the top of the stack will be popped and the previous activity resumes back its state. The basic concept of stack remains same here, if number of activities resides in the stack, then it is not possible to rearrange it, the activity which is present on the top will be popped out from the stack.

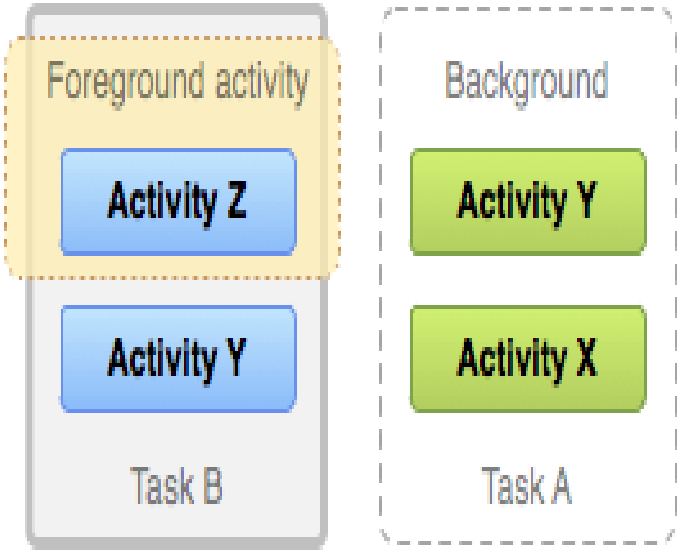
Stack - Activity Stack - Diagram



Creation of separate instances for same activity is possible as shown in the below given figure.



As discussed already the user interacts with the activity which is present at foreground. Below given figure shows that, Task B receives user interaction in the foreground, while Task A is in the background, waiting to be resumed.



UNIT- III

ANDROID SERVICES

Simple Services –Types of Service

Binding and Querying the service

Two states of service

Various methods of Service Base Class

Difference between Start Service and Bind Service

Executing Services

What is Android Service (Simple Service)?

In order to perform long running operations service will always run in a background. Service component will not interact with the user and it works in the background even if the application is destroyed.

In order to monitor changes in service state different service methods can be implemented. A service will have different method. In each service state the user can perform specific operations.

Types of Services

Foreground

As the name indicates if foreground services is performed by the system then it will become possible for the user to note that, since it is running in the foreground. Even if the user doesn't interact with the app the foreground services will run. The foreground service will display a Notification. For example, Audio app will use the foreground service.

Background

It’s very simple and opposite of the foreground service. The background service performs an operation which isn’t directly noticed by the user. For example, If the app tries to compact a storage space for an app, then it is an example for a background service.

Binding and Querying a Service

bindService() method is used by an application component to bound a service. In order to work with the client-server scenario, a bound service method provides support, as if services are bound, one component present in one application will interact with the component of another application.

Bound Service method will work only when another application component is bounded to it. It is also possible to bind multiple components to the service, the service gets destroyed when all the binded components is destroyed.

Two methods plays an important role, one is onStartCommand() which allows the component to start and another is onBind() method which allows the components to get binded. If the developer wants to create a new service then it is necessary to extend the service class, so that all the methods which the existing service class holds the developer can use that in a new class too.

Not only extending a service class, but also overriding some callbacks methods is also possible, the advantage of overriding methods is it provides support for binding services.

A service can fundamentally take two states :-

S.No	Service and Description
1	**Started** Activity is one type of application component. Activity is started by calling startService() method. A service will keep on running in the background for an indefinite period once it is started.
2	**Bound** Bound Service method will work only when another application component is bounded to it, the method which is used to perform this operation is bindService(). A bound service offers a client-server interface that allows components to interact with the service, send requests, get results etc.

Different methods present in Service Base Class

In order to create a service, first the developer will create Java class that extends the Service bass class. The Service Base class has various callbacks methods and the most important once are given below:

1. onStartCommand()

2. onBind()

3. onUnbind()

4. onRebind()

5. onCreate()

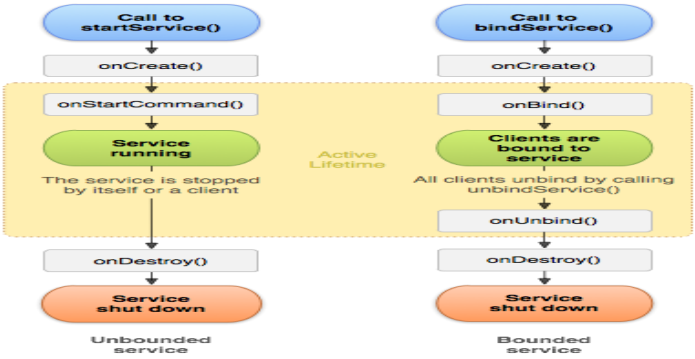
6. onDestroy()

S.No	Callback and Description
1	**onStartCommand()** The Application component such as an Activity, if requests that the service needs to be started then the system calls this startService() method. The service which is started can be stopped by calling either stopService() or stopSelf() methods.
2	**onBind()** The bindService() method is called when one component wants to bind with another component, in more simple words, if one service wants to bind with another service then this method is called. If you don't want to allow two services to be bind then it is going to return null.
3	**onUnbind()** From a particular interface if all the clients have disconnected then the system calls this method.
4	**onRebind()** When one client has been connected to the service, after the completion of that service the particular client will be Unbinded by the system by calling its onUnbind() method. Again if the same client wants to connect back to the service then the system will use onRebind() method.
5	**onCreate()** When the service is first created then the system

	calls this onStart() or onBind() command.This call is required to perform one-time set-up.
6	**onDestroy()** When the service is no longer necessary then this particular method is called to destroy that service. This method is mainly used to clean up resources which that particular service has acquired.

Difference between Start Service and Bind Service

As mentioned already in the above given table the services can be created using startService() method or bindService(). The diagram present in left shows how the service will be created using startService() method and the diagram present on the right hand side shows how the service is created by using bindService().

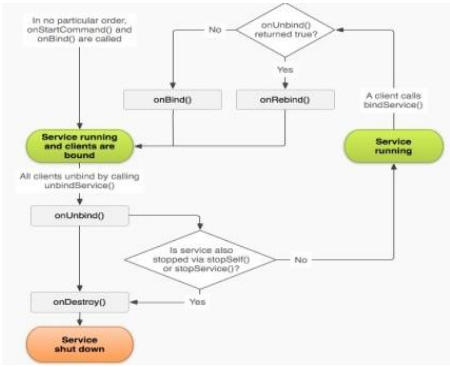


Executing a Service

A service will always run in a background, it is not possible for the user to interact with the service directly. If the user works with some application then the service is going to provide background support for the application. It will not

become possible for the user to run application without service.

The below given diagram shows the life cycle of a service that is started and the diagram also shows how the binding will take place.



BROADCAST RECEIVERS

1.What is Broadcast Receivers?

2.Creating a Broadcast Receiver

3.Managing a Broadcast Receiver

a.Registering Broadcast Receiver

b.By Using Intent

c.Broadcasting Custom Intent

4.Receiver Intent

a.What is Receiver Intent?

b.Attributes of Receiver Intent

c.System Generated Intent

5.Ordered broadcasts

What is Broadcast Receivers?

The main task done by the broadcast receiver is simply to respond to the broadcast message, the broadcast receiver will respond to the messages either it may be from application or from the system.These messages are sometimes called as events or intents.

For example, Applications too can generate broadcast for the purpose to let know other applications that it has downloaded certain data and it is available for them to use it.

The applications receive intents that are broadcasted by the system with the help of broadcast receivers.An application listens for specific broadcast intents by registering a broadcast receiver.

By overriding the onReceive() method or by extending the Android BroadcastReceiver class, the BroadcastReceiver service is initiated and implemented.

The broadcast receiver, registration can be done in two ways, either within the code or in the manifest file.

Two important steps to make Broadcast Receiver works for the system broadcasted intents :-

1. Creating the Broadcast Receiver.

2. Registering Broadcast Receiver.

Creating a Broadcast Receiver

A broadcast receiver can be created in two ways. First is, for BroadcastReceiver class a broadcast receiver is implemented as a subclass, second is by overriding the onReceive() method where each message is received as a Intent object parameter.

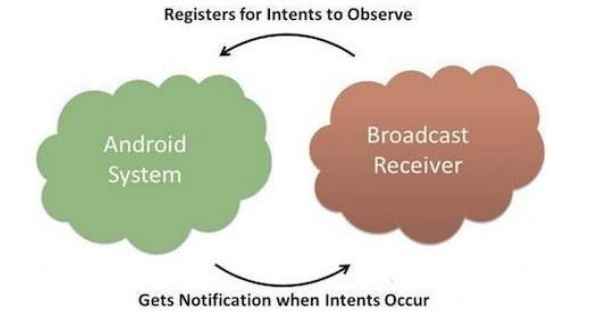
```
public class EgSubBroadcastReceiver extends
 BroadcastReceiver
{
@Override public void onReceive(Context context, Intent intent)
{
Toast.makeText(context,
"ActionDetected.",Toast.LENGTH_SHORT).show();
}
}
```

Managing Receivers

1. Registering the broadcast receivers
2. By Using Intent
3. Broadcasting Custom Intent

Registering Broadcast Receiver

If it's necessary that certain applications should listens for specific broadcast intent, then it is necessary that broadcast receiver should be registered in AndroidManifest.xml file.



Broadcast-Receiver

What is Intent?

An Android Intent describes what action to be performed.

The different methods present in an Intent and its description is given below.

S.No	Method and Description
1	**Context.startActivity()** In order to launch a new activity the intent object is passed to this method, and also to get an existing activity, this particular method will be used.
2	**Context.startService()** To initiate a service or to deliver a new instructions to an ongoing service, this method will be used.
3	**Context.sendBroadcast()** In order to deliver messages to the intended and all interest broadcast receivers, this method will be used.

Broadcasting Custom Intents

By using sendBroadcast() method inside activity class, as a developer we can make our application itself to generate and send customized intents. One more method is there which is sendStickyBroadcast(Intent) method, the purpose of this method is also to generate and send customized intents only.

The difference between sendBroadcast() and sendStickyBroadcast(Intent) is, if the latter is used then the Intent will be sticky, meaning that the Intent stays around

even after the broadcast completion. It's always preferable to use the sendBroadcast() method.

Receiver Intent

1. What is Receiver Intent?
2. Attributes of Receiver Intent
3. System Generated Intent

What is Receiver Intent?

An Android Intent describes what action to be performed. Receiver intent refers to which application is receiving an Intent (action).

<receiver android:directBootAware=["true" | "false"]

android:enabled=["true" | "false"]

android:exported=["true" | "false"]

android:icon="drawable resource"

android:name="string"
android:process="string">
</receiver>

The following is the different attributes of Receiver Intent

android:enabled

This attribute can hold only two values either True or False. If the broadcast receiver can be instantiated by the system then the attribute will be true, else it will be false. All the enabled attributes present in the <application> element will be applicable to all the application components. The values for both <application> and <receiver> cannot be

instantiated. For both the attribute value will be true if the broadcast receiver is enabled else the value will be false.

android:exported

If the broadcast receiver receives messages from sources outside its application then the value for this attribute will be true else it is false.

android:icon

This is an icon representing the broadcast receiver. This attribute is mainly used to set a reference to a drawable resource.

android:name

This attribute is used to hold the name of the class that implements the broadcast receiver, a subclass of BroadcastReceiver. This should be a fully qualified class name (such as, "com.example.project.SampleProgram")

android:process

The different components present in the application will run in the default process created for that application. The process and package will have the same name.

System generated Intents and it's description

1. android.intent.action.BATTERY_CHANGED

This intent holds different information about battery like its charging state and level etc.

2. android.intent.action.BATTERY_LOW

This intent indicates the low battery condition of the device.

3. android.intent.action.BATTERY_OKAY

This intent indicates the device battery is ok now after low.

4.android.intent.action.BOOT_COMPLETED

After the device completes booting, this intent is broadcasted once.

5.android.intent.action.CALL

This intent performs a call to someone specified in the data.

6.android.intent.action.CALL_BUTTON

This intent performs an action, if the user uses dialler or other appropriate user interface to make a call by pressing the "call" button.

7.android.intent.action.DATE_CHANGED

This intent performs action that is changing of date and indicates that the date has changed.

8.android.intent.action.TIIME_CHANGED

This intent performs action that is changing of time and indicates that the time has changed.

9.android.intent.action.REBOOT

After the device completes rebooting, this intent is broadcasted.

Ordered Broadcasts

In order the broadcasts are sent to each receiver, to achieve this, the system uses priority. The android:priority attribute is used to control the priority, which will be sent to the manifest file for the intent-filter element. The receiver

with highest priority is able to receive the broadcast first and it also possess the capability to abort the lower priority broadcast.

Important Point about Priorities

If the Receivers are having same priority, then it will be executed in the order in which it is created. Priorities will be set by using the non-negative numbers only. Negative numbers too are acceptable while setting priority but sometimes it will end up in an unexpected errors. So it is always best to use positive numbers to set priority.

CONTENT PROVIDERS

1.What is Content Provider?

a. Overview diagram of how Content Providers manage access to data storage

b. Swapping out a SQLite database for alternative storage

c. Content URIs

2.Creating and Using Content Providers

a. Methods used with ContentProvider class

b.Relationship between content provider and other components

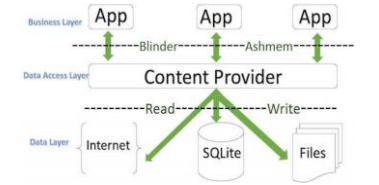
3.Content Resolver

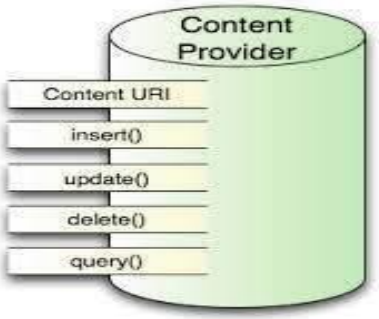
a.Interaction between ContentProvider, Content Resolver, other classes, and storage

What is Content Provider?

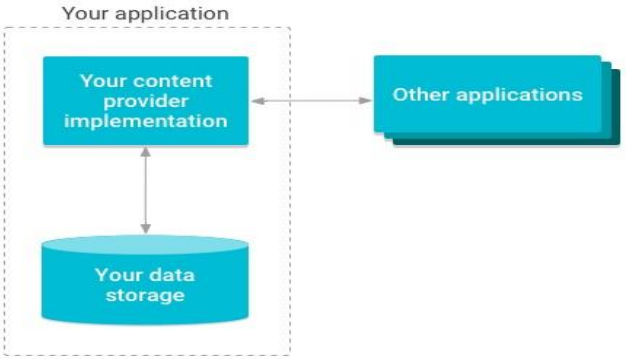
The work done by Content Provider is very important. Mainly, this content provider helps the application to access the data stored by the app itself, stored by an another app, and

to share data with other apps. The content provider sums up all the data and also provides a mechanism for data security. Implementation of content provider provides many advantages. It is also possible to configure a content provider to allow other applications to securely access and modify your app data.





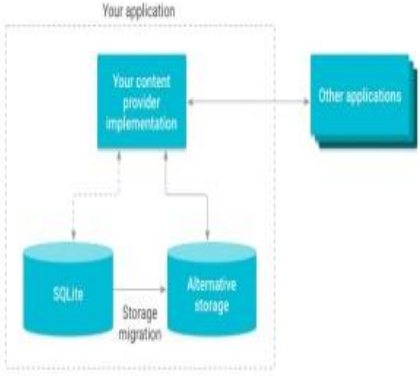
Overview diagram of how Content Providers manage access to data storage



The above diagram shows clearly that in application data has been stored and for the same application the content provider implementation too has been done, the other applications, if they want to access data then this content provider will acts as an intermediator to share data between two applications securely.

If the developer wants to share the app data or not, implementation of content provider is advisable, the reason is first it provides security to your data. This allows you to make modifications to your application data storage implementation without affecting other existing applications that rely on access to your data. In this situation only your content provider is affected and not the applications that access it.

Swapping out a SQLite database for alternative storage



As discussed earlier, the Content Provider provides security and with the help of which only the data will be shared between different applications. The above given diagram shows that from SQLite which is the inbuilt database in a mobile, the data migration is possible, here, the data migration is nothing but completely moving data from one database to another database.

In normal database the basic operations which we perform like querying database, editing it, inserting data, deleting data, updating it, all these are possible with SQLite Database too. ContentProvider is the main class which holds the sub class and a set of Application Programming Interfaces. A content provider is implemented as a subclass of **ContentProvider** class and must implement a standard set of APIs that enable other applications to perform transactions.

The ContentProvider shares data from one application to an another application on request. Such requests will be handled by the ContentResolver class.

Content Provider uses different methods to store the data, it can be stored in database, in files or even over a network.

Content URIs

The format of URI that is Uniform Resource Identifier is shown below. In order to retrieve data from the Content Provider it is necessary to specify the query string in the form given below.

<prefix>://<authority>/<data_type>/<id>

S.No	Meaning
1	**prefix** This is always set to content ://
2	**authority** The name of the content provider is specified by this authority. For example, Contacts, Browser etc. The contents, browser etc are the content providers present in the system. The third party content provider needs fully qualified name for example, com.java.statusprovider

3	**data_type** If the content provider is used to provide data, then what type of data it is going to provide that information will be present in data_type. For example, if we want to retrieve all the contacts from the Contacts content provider, then the data path would be people and URI would look like this content://contacts/people
4	**id** This specifies the exact information to be retrieved from the content provider. For example, if the user wants to fetch the contact number 10 from the Contacts, then Uniform Resource Identifier will look like this *content://contacts/people/5*.

Creating and Using Content Provider

A content provider manages access of data. The content provider can be implemented as a one or more class in an Android application, along with that its element in the manifest file. One of your class implements a subclass ContentProvider, which is the interface between your provider and other applications.

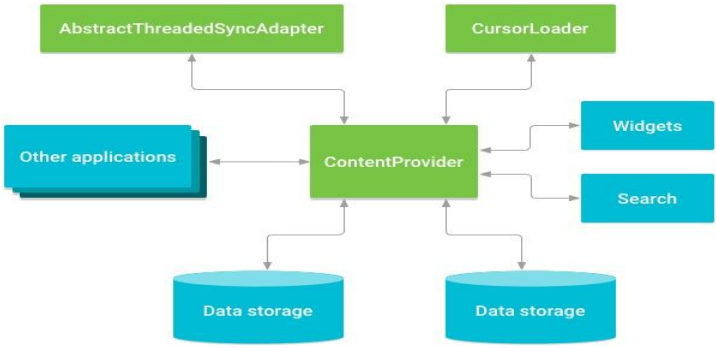
```
public class SampleApplication extends ContentProvider
{
//Code goes here
}
```

Content Providers are mainly designed in order to make data available to other applications on request. The activity present in your application allows the developer to query and make changes to the data managed by your provider.

Methods used with ContentProvider class

Method	Purpose
onCreate()	The system calls this method when the Content Provider has been started.
query()	This method receives a request from the client. This method processes the request and the result is returned to the client. But, the result is returned as a Cursor object.
insert()	In order to add new record to the Content Provider this method is used.
delete()	This method deletes an existing record from the content provider.
update()	This method is mainly used to modify the content present in the content provider.

Relationship between content provider and other components



The above given diagram shows the relationship between the ContentProvider and other components.

1. Sharing application data with other applications on request.

2. Sending data to different widgets.

3. Returning custom search suggestions for the application framework.

4.With the implementation of Abstract Threaded Sync Adapter, application data synchronization will take place.

5. Using Cursor Loader, loading data in UI

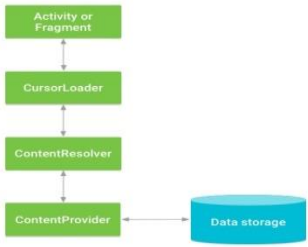
CONTENT RESOLVER

From the content provider if the application needs to access the data, the ContentResolver object is used, in application context to communicate with the provider. ContentProvider and ContentResolver will not communicate with each other directly. The object of ContentResolver sends a message to the object of ContentProvider to access the data.

The ContentProvider object receives the request for data, performs the action to provide the requested data and once found the requested data is returned.

The different ContentResolver methods provide the basic Create, Retrieve, Update, and Delete functions.

Interaction between ContentProvider, Content Resolver, other classes, and storage



The different components shown in the above diagram are the ContentProvider, ContentResolver, CursorLoader and an Activity or Fragment. A CursorLoader is used to run a synchronous query in the background.

The Activity or Fragment in User Interface calls a CursorLoader, which in turn gets the ContentProvider using the ContentResolver.

WORKING WITH DATABASES : SQLite

What is SQLite?

Difference Between SQLite and SQL Database?

Database – Package

Database – Creation

Database –Insertion

Database – Fetching

Example program to work with SQLite database

What is SQLite?

SQLite is an open source, inbuilt database, this database can store any type of data. Like different features present in relational database, it is also possible to implement the same in SQLite database.

Normally in order to access data from other database, JDBC and OBDC drivers is needed, but while working with SQLite database, in order to retrieve the data from database, drivers are not required.

Difference between SQLite and SQL Database

1. SQL is query language whereas SQLite is embeddable relational database management system.

2. SQLite Database is the light version of SQL RDBMS.

3. Since SQLite is a light version of SQL, it is mostly used in portable devices like Android, iOS etc.

Database – Package

The package which is needed to work with SQLite database is**android.database.sqlite**

Database - Creation

OpenOrCreateDatabase, this method is used to create a database, with database name and parameter, in parameter the information which is going to be present is the mode of the database.

Its syntax is given below:

SQLiteDatabase samplefirstpgm =

openOrCreateDatabase("Database

Name",MODE_PRIVATE,null);

Database Method and it's description

1. openDatabase(String path,

SQLiteDatabase.CursorFactory factory, int flags,

DatabaseErrorHandler errorHandler)

This method is used to open existing database with the appropriate mode. The common flags mode can be OPEN_READWRITE, OPEN_READONLY,

OPEN_WRITEONLY etc.

2.openDatabase(String path,

SQLiteDatabase.CursorFactory factory, int flags)

It is similar to the above mentioned method, the only small different is, the current method does not have database error handler.

3. openOrCreateDatabase(String path,

SQLiteDatabase.CursorFactory factory)

This method not only opens the existing database but also used to create a new database if it doesn't exists.

4.openOrCreateDatabase(File file,

SQLiteDatabase.CursorFactory factory)

This method not only opens the existing database but also used to create a new database if it doesn't exist. Compared to previous method which is also used for the same purpose, only the argument differs. The file parameter which is present in this method, gives the file path.

Database - Insertion

In SQLiteDatabase class execSQL method is present which is used to insert data into table.

Syntax

samplefirstpgm.execSQL("CREATE TABLE IF NOT EXISTS student(Firstname VARCHAR,Lastname VARCHAR);");

Example

samplefirstpgm.execSQL("INSERT INTO student VALUES('zaara','asma');");

Method to insert data and description:its

execSQL(String sql, Object[] bindArgs)

This method not only inserts the data but also used to modify the existing data.

Database - Fetching

Once the database has been created and the data is inserted, it is possible to fetch the data from database, either complete data or based on condition.The rawQuery method is called to fetch the data it will return the resultset. We can move the cursor forward and retrieve the data.

Example for fetching data from database

```
Cursor resultSet = samplefirstpgm.rawQuery("Select * from student",null);

resultSet.moveToFirst();

String Firstname = resultSet.getString(0);

String Lastname= resultSet.getString(1);
```

Functions of Cursor class:

Sr.No	Method and Description
1	**getColumnCount()** This method is used to get the total number of columns present in the table.
2	**getColumnIndex(String columnName)** This method is used to retrieve the column index, it takes one argument which contains the name of the column, and in that argument whatever column name the user passes for that this method returns the index value.
3	**getColumnName(int columnIndex)** This method is used to retrieve the column name, it takes one argument which contains the index of the column, and in that argument

	whatever column index the user passes for that this method returns the column name.
4	**getColumnNames()** This method is used to return all the column names present in the table.
5	**getCount()** This method is used to return the total number of rows present in the table.
6	**getPosition()** This method is used to return the current position of cursor in the table.
7	**isClosed()** This method returns only two values either true or false. If the cursor is closed then this method returns true otherwise this method returns false.

Sample Program to insert Username, Password and City in SQLite Database

CODING:

activity_main.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
```

```
android:paddingRight="@dimen/activity_horizontal_margin"
   android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.database_demo_sql.MainActivit
y">
<TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentTop="true"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="36dp"
      android:text="Storing Data in SQLite Database"
android:textAppearance="?android:attr/textAppearanceLarge"
/>
<Button
      android:id="@+id/btninsert"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_below="@+id/textView1"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="70dp"
      android:text="PRESS BUTTON TO ENTER DATA IN
DATABASE" />
</RelativeLayout>
```

MainActivity.java

```
package com.example.database_demo_sql;
import android.support.v7.app.ActionBarActivity;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
publicclass MainActivity extends ActionBarActivity implements OnClickListener
{
Button btninsert,btnview,btndelete,btnedit;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
btninsert=(Button)findViewById(R.id.btninsert);
btninsert.setOnClickListener(this);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
```

```
{
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
int id = item.getItemId();
if (id == R.id.action_settings)
{
returntrue;
}
returnsuper.onOptionsItemSelected(item);
}
@Override
publicvoid onClick(View arg0)
{
// TODO Auto-generated method stub
if(arg0.getId()==R.id.btninsert)
{
Intent i=new Intent(MainActivity.this,InsertActivity.class);
startActivity(i);
}
}
```

}

activity_insert.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.database_demo_sql.InsertActivit
y">
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentLeft="true"
    android:layout_alignParentTop="true"
    android:layout_marginLeft="101dp"
    android:layout_marginTop="51dp"
    android:text="Insert Data" />
<EditText
    android:id="@+id/insertuser"
```

```
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignRight="@+id/textView1"
    android:layout_below="@+id/textView1"
    android:layout_marginTop="42dp"
    android:hint="Username"
    android:ems="10">
    <requestFocus />
    </EditText>
<EditText
    android:id="@+id/insertpwd"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignLeft="@+id/insertuser"
    android:layout_below="@+id/insertuser"
    android:layout_marginTop="33dp"
    android:ems="10"
    android:hint="Password"
    android:inputType="textPassword" />
<EditText
      android:id="@+id/insertcity"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
android:layout_alignLeft="@+id/insertpwd"
```

```
        android:layout_below="@+id/insertpwd"
        android:layout_marginTop="44dp"
  android:hint="City"
        android:ems="10" />
<Button
        android:id="@+id/insbtn"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
android:layout_alignRight="@+id/insertuser"
        android:layout_below="@+id/insertcity"
        android:layout_marginTop="34dp"
  android:text="INSERT" />
</RelativeLayout>
```

InsertActivity.java

```
package com.example.database_demo_sql;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
```

```
publicclass InsertActivity extends ActionBarActivity implements OnClickListener
{
EditText edtuser,edtpwd,edtcity;
Button btn;
DBHelper dbh;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_insert);
edtuser=(EditText)findViewById(R.id.insertuser);
edtpwd=(EditText)findViewById(R.id.insertpwd);
edtcity=(EditText)findViewById(R.id.insertcity);
btn=(Button)findViewById(R.id.insbtn);
btn.setOnClickListener(this);
dbh=new DBHelper(InsertActivity.this);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
 {
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.insert, menu);
```

```
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
 {
int id = item.getItemId();
if (id == R.id.action_settings)
 {
returntrue;
}
returnsuper.onOptionsItemSelected(item);
}
@Override
publicvoid onClick(View arg0)
 {
// TODO Auto-generated method stub
String user=edtuser.getText().toString();
String pwd=edtpwd.getText().toString();
String city=edtcity.getText().toString();
Student s=new Student(user,pwd,city);
dbh.InsertData(s);
Toast.makeText(InsertActivity.this,"Record
Inserted",500).show();
finish();
```

```
}
}
```

Student.java

```
package com.example.database_demo_sql;
publicclass Student
{
int _id;
String user;
String pwd;
String city;
public Student()
{
super();
}
public Student(int _id, String user, String pwd, String city)
 {
super();
this._id = _id;
this.user = user;
this.pwd = pwd;
this.city = city;
}
public Student(String user, String pwd, String city)
 {
```

```
super();
this.user = user;
this.pwd = pwd;
this.city = city;
}
public Student(String user, String city)
{
super();
this.user = user;
this.city = city;
}
@Override
public String toString()
{
// TODO Auto-generated method stub
return _id+" "+user+" "+pwd+" "+city;
}
public String getUser()
{
// TODO Auto-generated method stub
returnnull;
}
public String getPwd()
 {
```

```
// TODO Auto-generated method stub
returnnull;
}
public String getCity()
{
// TODO Auto-generated method stub
returnnull;
}
public Integer get_id()
{
// TODO Auto-generated method stub
returnnull;
}
}
```

OUTPUT:


InsertActivity
Insert Data
Username
Password
City
INSERT


InsertActivity
Insert Data
Zaara
•••••
Vellore
INSERT

DATA ANALYSIS

What is Data Analysis?

Data analysis is concerned with the nature and use of data. In data analysis first step is to find out the data elements needed for placing, second step is logically dividing the data and third step is creating meaningful relationships between data to produce a resulting group.

Once the data has been analyzed it provides quick overview of complete data which in turn provides a quick solution for any problem. Data analysis provides useful insights of data design, provides a quick solution. It is a processing of changing data and designing data to find out useful information.

The main purpose is to extract the useful information from the database. With the help of techniques which is used to analyze data, it will become possible for the user to understand the structure and meaning of data.

Data Analysis = establishing the nature of data.

Functional Analysis = establishing the use of data.

Data analysis in SQLite

When for large amount of data if data analysis should be done then first data can be imported from CSV files, then the functionality will be applied on that data in order to generate numerous data reports.

Complex analysis can be done using simple scripts written in Python (which come with SQLite built-in) or in R or other languages. The advantage of SQLite database is, it is open source, inbuilt and light weight version of Relational Database Management System so it takes less memory space, and in this SQLite after analysis data will be present in single file that can be stored in USB or even can be saved as a backup copy in mail.

Types of Data Analysis

1. Text Analysis
2. Statistical Analysis
    i. Descriptive Analysis
    ii. Inferential Analysis
3. Diagnostic Analysis
4. Predictive Analysis
5. Prescriptive Analysis

Text Analysis

Text Analysis which is also referred to as Data Mining, is a process of extracting meaningful information from high volume of data. Text analysis also transforms the data into meaningful information.It also offers a different way to fetch and test data, then finally the data interpretation will take place.

Statistical Analysis

Statistical Analysis includes collecting, analyzing data, converting it from one form to another form, presenting and modelling data. Statistical analysis can be done only on set of data or sample data not on entire database.

There are two categories of this type of Analysis - Descriptive Analysis and Inferential Analysis.

Descriptive Analysis

Descriptive analysis will analyze sample data as well as it will analyze complete data and provides overview of it.

Inferential Analysis

Analyses sample from complete data. In this type of Analysis, each time when different sample is selected for analysis then different conclusions will come as a result.

Diagnostic Analysis

If any new problem occurs, then this analysis is done to find out the similar pattern. This Analysis is useful to identify behavior patterns of data. Because chances are there it may have similar perceptions for the new problem.

Predictive Analysis

As the name indicates it is a predictive analysis. This analysis is mainly done to find out the future outcomes based on current or past data. In predictive analysis, the accuracy of result depends on the detailed information present in the database.

Prescriptive Analysis

Prescriptive Analysis is used to determine from all previously discussed analysis which analysis is best to take in order to get the accurate result. Most data uses Prescriptive Analysis because predictive and descriptive Analysis is not enough to improve data performance.

UNIT – IV

ANDROID USER INTERFACE

Android Layouts

What is Android View?

What is Android ViewGroup?

Difference between View and ViewGroup?

Android Layout Types:-

1. Linear Layout
2. Table Layout
3. Frame Layout
4. Grid Layout

ANDROID LAYOUT

What is Android Layout?

A layout is used to define the structure for a user interface in your app. A View usually draws something the user can see and interact with.

Opposite of the View is the ViewGroup, where the ViewGroup is an invisible container, which is used, in order to define the layout structure for View and ViewGroup objects.

What is Android View?

For all User Interface components in Android, the View is a base class. One Simple example of user interface is EditText, which is used to get the input from the user during

runtime in Android Apps, EditText is a subclass of View Class.

Following are some of common View subclasses that will be used in android applications.

1. TextView
2. EditText
3. Button
4. CheckBox
5. RadioButton
6. ImageButton
7. Progress Bar
8. Spinner

What is Android ViewGroup?

The ViewGroup is a base class for layout and layout parameters and subclass of View. To hold Views or ViewGroups and to define the layout properties, the ViewGroup provides an invisible container.

Following are the commonly used ViewGroup subclasses in android applications.

1.Linear Layout
2. Relative Layout
3. Table Layout
4. Frame Layout
5. Grid View

Android Layout Types

The Android provides different layouts; in order to change the View, Look and Design of Android Application the developer will select any one view from the available ones.

1. Linear Layout

Linear Layout is a ViewGroup which is used to align the components in Android Application either horizontally or vertically.

2. Relative Layout

If the developer wants to place the components by mentioning the relevant position in the User Interface, then this layout will be used.

3. Table Layout

If the developer wants to place the components in the form of rows and columns then this table layout will be used.

4. Frame Layout

Frame Layout is like a placeholder which is used to display a single view.

5. Grid Layout

Grid Layout is used to display the component in a 2Dimensional View, it is somewhat similar to the Table View, but the main difference is any number of rows any numbers of columns are allowed in table view, but in Grid View the rows and columns number should be same.

ANDROID LAYOUT ATTRIBUTES

The following are the attributes which is applicable for all types of layout.

1.android:id

This attribute is used to uniquely identify the view.

2.android:layout_width

This attribute is used to set the width of the layout.

3.android:layout_height

This attribute is used to set the height of the layout.

4.android:layout_marginTop

This attribute is used to set the extra space on the top side of the layout.

5.android:layout_marginBottom

This attribute is used to set the extra space on the bottom side of the layout.

6.android:layout_marginLeft

This attribute is used to set the extra space on the left side of the layout.

7.android:layout_marginRight

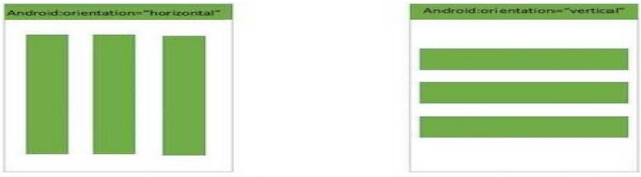
This attribute is used to set the extra space on the right side of the layout.

8.android:layout_gravity

This attribute is used to specify how child views are to be positioned.

LINEAR LAYOUT

Linear Layout is a ViewGroup which is used to align the components in Android Application either horizontally or vertically.



Linear Layout

Linear Layout Attributes

1.android:id

This attribute is used to uniquely identify the layout**.**

2.android:orientation

This is the main attribute of linear layout and is used to specify the orientation, either it may be horizontal or vertical.

3.android:baselineAligned

This attribute can hold only two values either True or False, and this attribute prevents the layout from aligning baselines.

4.android:gravity

This attribute is used to specify how the object should be placed, either left, right, top, bottom, center etc.

Example of Linear Layout with Vertical Orientation

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"

android:layout_width="match_parent"

android:layout_height="match_parent"

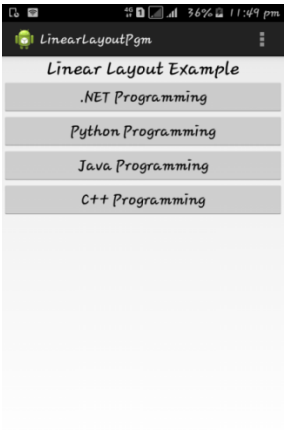
android:orientation="vertical">

<TextView

android:layout_width="wrap_content"

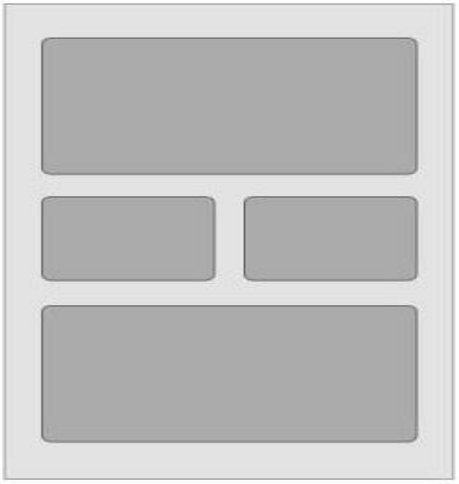
```
android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceLarge"
android:text="Linear Layout Example"
android:id="@+id/textView"
android:layout_gravity="center_horizontal" />
<Button
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="C++ Programming" />
<Button
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Java Programming"/>
<Button
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text=".NET Programming"/>
<Button
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:text="Python Programming"/>
</LinearLayout>
```

OUTPUT – Linear Layout



RELATIVE LAYOUT

If the developer wants to place the components by mentioning the relevant position in the User Interface, then this layout will be used.



Relative Layout

Relative Layout Attributes

1.android:id

This attribute is used to uniquely identify the layout**.**

2.android:gravity

This attribute is used to specify how the object should be placed either left, right, top, bottom, center etc.

3.android:ignoreGravity

This attribute is used to set which view should not be affected by Gravity.

Example of Relative Layout

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent">
<TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
android:textAppearance="?android:attr/textAppearanceLarge"
      android:text="SIGN IN PAGE"
      android:id="@+id/textView3"
      android:layout_alignParentTop="true"
      android:layout_centerHorizontal="true" />
```

```
<TextView
    android:id="@+id/userName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
android:layout_marginLeft="@dimen/activity_horizontal_ma
rgin"
    android:layout_marginTop="110dp"
    android:text="UserName:"/>
<TextView
    android:id="@+id/password"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/userName"
android:layout_margin="@dimen/activity_horizontal_margin
"
    android:text="Password:"/>
<EditText
    android:id="@+id/edt_userName"
    android:layout_width="fill_parent"
    android:layout_height="40dp"
android:layout_marginLeft="@dimen/activity_horizontal_ma
rgin"
    android:layout_marginTop="100dp"
    android:layout_toRightOf="@+id/userName"
    android:hint="User Name" />
```

```
<EditText
    android:layout_width="fill_parent"
    android:layout_height="40dp"
    android:layout_below="@+id/edt_userName"
    android:layout_centerVertical="true"
    android:layout_toRightOf="@+id/password"
    android:hint="Password" />
<Button
    android:id="@+id/btnLogin"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/password"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="20dp"
    android:text="LOGIN"
    android:textStyle="bold" />
</RelativeLayout>
```

Output of Relative Layout

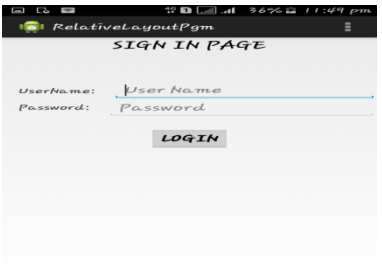


TABLE LAYOUT

If the developer wants to place the components in the form of rows and columns then this table layout will be used.**<TableRow>**element is used to build a row in the table. Each row can have zero or more cells, each cell can hold one view object. Table Layout will not display border.

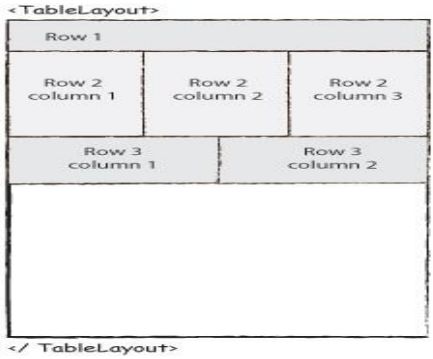


Table Layout Attributes

1.android:id

This attribute is used to uniquely identify the layout**.**

2.android:shrinkColumns

This attribute is used to shrink the table columns. The column indices must be separated by commas.

3.android:stretchColumns

This attribute is used to stretch the table columns. The column indices must be separated by commas.

Example of Table Layout

<?xml version="1.0" encoding="utf-8"?>

```
<TableLayout
xmlns:android="http://schemas.android.com/apk/res/android"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:layout_marginTop="100dp"
   android:paddingLeft="10dp"
   android:paddingRight="10dp">
<TableRow>
<TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="StudentID" />
<TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Student Name" />
<TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="Department" />
</TableRow>
```

```
<TableRow>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="01" />
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="Firza" />
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="B.C.A" />
</TableRow>
<TableRow>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="02" />
```

```
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="Aamin" />
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="B.C.A" />
</TableRow>
<TableRow>
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="3" />
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_weight="1"
android:text="Farhath" />
<TextView
android:layout_width="wrap_content"
```

android:layout_height="wrap_content"

android:layout_weight="1"

android:text="M.C.A" />

</TableRow>

</TableLayout>

Output of Table Layout

FRAME LAYOUT

Frame Layout is like a placeholder which is used to display a single view.

Attributes of Frame Layout

1.android:id

This attribute is used to uniquely identify the layout**.**

2.android:foregroundGravity

Possible values of Gravity are left, right, top, bottom, center etc, and this attribute defines the gravity to apply for the foreground drawable.

3.android:measureAllChildren

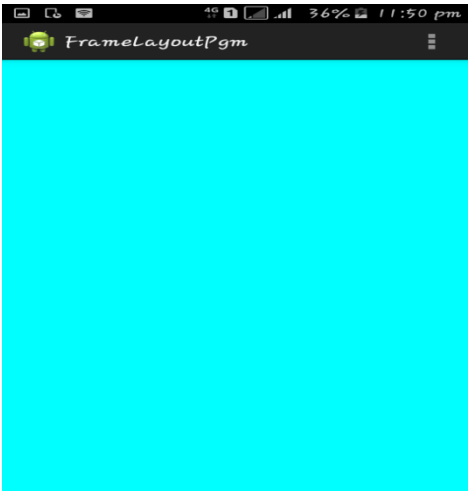
Default value for this attribute is false. This attribute is used to measure all children or just those in the VISIBLE or INVISIBLE state.

Frame Layout Program:

```
<FrameLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:id="@+id/framelayout"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_gravity="center"
android:foregroundGravity="fill"
android:foreground="#0ff">
<LinearLayout
android:orientation="vertical"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerInParent="true">
<TextView
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:gravity="center_horizontal"
android:text="Hello World"/>
</LinearLayout>
```

</FrameLayout>

Output of Frame Layout:



GRID LAYOUT

Grid Layout is used to display the component in a 2Dimensional View, it is somewhat similar to the Table View, but the main difference is for any number of rows any numbers of columns are allowed in table view, but in Grid View the rows and columns number should be same.

Attributes of GridView

1.android:id

This attribute is used to uniquely identify the layout**.**

2.android:gravity

This attribute is used to specify how the object should be placed, either left, right, top, bottom, center etc.

3.android:horizontalSpacing

This attribute is used to define the default horizontal spacing between columns.

4.android:verticalSpacing

This attribute is used to define the default vertical spacing between rows.

5.android:numColumns

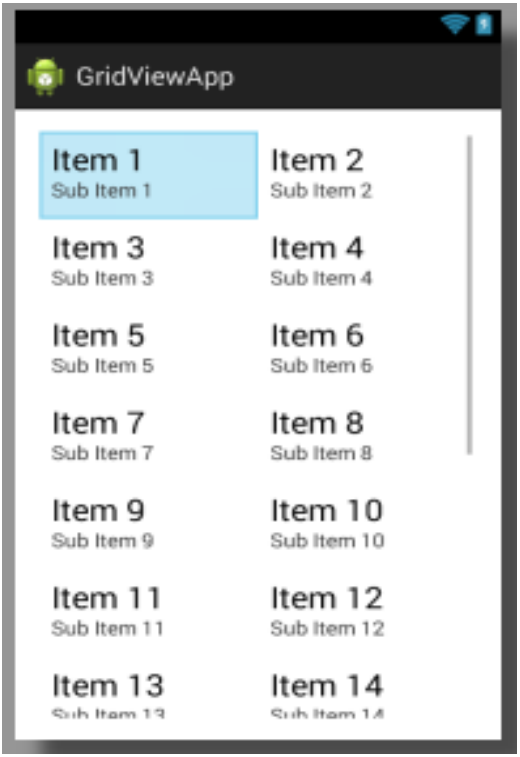
This attribute is used to specify how many Columns to be display.

Grid View Coding:

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.gridviewapp.MainActivity" >
<GridView
android:id="@+id/simpleGridView"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:numColumns="2"/>
```

</RelativeLayout>

Grid Layout will always be displayed according to the num of Columns specified in the coding:



MENUS

What is Android Menu?

Defining a menu in XML

Types of Android Menu:-

1. Options Menu
2. Context Menu
3. Popup Menu

What is Android Menu?

In many types of Android application, Menus will be used as a User Interface component. In order to give user friendly interface, the developer will implement in the menus in an app.

Menu Elements and its Description

1. <menu>

This element is used to define a menu in XML file, root element will be there and it will hold one or more elements.

2. <item>

This element is used to create a menu items. This element may contain a nested <menu> in order to create a submenu.

Defining a Menu in XML

Standard XML format Android provides to define menu items.The Menu and its item definition will be done by the developer in XML menu resource and not in Activity file.

Using a menu resource is a good practice for a few reasons:

First reason is easy to visualize what are the different menus, submenu and items are available. Second, the application content will be kept separately from the menu, Third reason is for different platform versions, screen sizes and other configuration Android allows the developer to create an alternative menu configuration.

Types of Android Menus

1. Option Menu
2. Context Menu
3. Pop-up Menu

OPTIONS MENU

In Android application for an Activity, **Options Menu** is a primary collection of menu items, which will have a global impact on the app such as Settings, Search, and Bookmark etc. Android options menu can be displayed in two different ways. One is simple options and the other is options with an image.

The menu will be inflated by calling the inflate() method present in the MenuInflator class. To perform event handling on menu items, it is necessary to override the onOptionsItemSelected() method of Activity class.

OPTIONS MENU SAMPLE PROGRAM:

main_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android"
>
<item
android:id="@+id/about_us_id"
android:title="About"
android:showAsAction="always"/>
<item
android:id="@+id/settings_id"
android:title="Settings"
android:showAsAction="always"/>
<item
android:id="@+id/Share_id"
```

```
android:title="Share"
android:showAsAction="always"/>
<item
android:id="@+id/Help_id"
android:title="Help"
android:showAsAction="always"/>
</menu>
```

MainActivity.java

```
package com.example.menudemo;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
```

```
getMenuInflater().inflate(R.menu.main_menu, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
Toast.makeText(this, "Selected Item ID: "+item.getItemId(),
0).show();
switch (item.getItemId())
{
case R.id.about_us_id:
returntrue;
case R.id.settings_id:
returntrue;
case R.id.Share_id:
returntrue;
case R.id.Help_id:
returntrue;
default:
returnsuper.onOptionsItemSelected(item);
}
}
}
```

OUTPUT:


5554:AVD_for_3_7in_WVGA_Nexus_One
11:19
ABOUT
SETTINGS
SHARE
HELP
Options Menu Program to get ID of Selected Item
Selected Item ID: 2131230721


5554:AVD_for_3_7in_WVGA_Nexus_One
11:18
ABOUT
SETTINGS
SHARE
HELP
Options Menu Program to get ID of Selected Item
Selected Item ID: 2131230722

CONTEXT MENU

This is one type of Android Menu which will be displayed when the user long click on an element. This menu will not have a global impact on the app; this menu is useful to implement actions that affect the selected content.

When the user right clicks on some icons in the system then the list of options will be displayed, the Context Menu is similar to like that only. To display the items present in the menu when the user right clicks then the developer should override the onCreateContextMenu() in activity or fragment.

SAMPLE CODING:

Coding to be written in activitymain.xml file

<Button
android:id="@+id/showContents"
android:width="wrap_content"
android:height="wrap_content"
android:text="LONG PRESS THE BUTTON"

android:marginTop="500dp“

android:marginLeft="200dp"/>

Coding to be written in MainActivity.java file

public void onCreateContextMenu(ContextMenu menu, View v)

{
menu.setHeaderTitle("Context Menu");

item.add("Java");

item.add("C#")

item.add(".NET");

item.add("Python");

}

@Override

public boolean onContextItemSelected(MenuItem item)

{

Toast.makeText(this, "The Selected Item is" +item.getTitle(), Toast.LENGTH_LONG).show()

return true;

}

POPUP MENU

Popup menu will display a list of items. The popup menu will be displayed either above or below the view. If there is a space below the view then the popup menu will appear below else it will appear above.It is necessary to create a folder named menu inside a project resource directory to create popup menu, and also it is important to add a new XML file to build the menu. The next step is to open a newly created XML file and write the necessary coding.

SAMPLE CODING:

<**menu** xmlns:android="http://schemas.android.com/apk/res/android" >

<**item** android:id="@+id/javalanguage"

android:title="Java" />

<**item** android:id="@+id/C#language"

android:title="C#" />

<**item** android:id="@+id/pythonlanguage"

android:title="Python" />

<item android:id="@+id/CProgramming"
android:title="C" **/>**

<item android:id="@+id/Androidprogramming"

android:title="Android" **/>**

</menu>

Commonly used <item> attributes in android applications for OPTIONS, CONTEXT and POPUP MENU and it's description

1.android:id

This attribute is used to uniquely identify an element in the application.

2.android:icon

As the name indicates this attribute is used to set the item's icon.

3.android:title

This attribute is used to set the item's title.

4.android:showsAsAction

This attribute specifies how the item will appear as an action item in the app.

LISTS AND NOTIFICATIONS

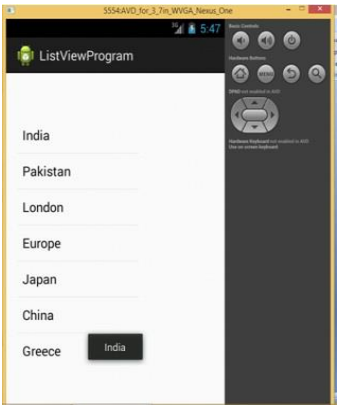
Lists and Notifications

Creation and Display

Android Lists

Android ListView is used to display the group of items in a Scrollable list, to work with ListView, it is necessary to import android.widget.ListView class. Android ListView

groups several items and it will display that with a vertical scrollable list.



List View Attributes

1.android:id

This attribute is used to uniquely identify the control.

2.android:divider

This attribute is used to draw a line between two items.

3.android:entries

This attribute is used to specify the reference to an array resource, which displays the view.

4.android:footerDividersEnabled

The default value for this attribute is true. When the value is set to false, the ListView will not draw the divider before each footer.

5 android:headerDividersEnabled.

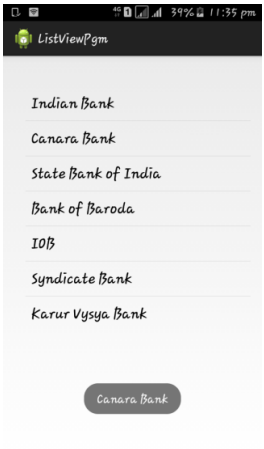
The default value for this attribute is true. When the value is set to false, the ListView will not draw the divider after each header.

Sample Program:

```
package com.example.listviewpgm;
import android.app.Activity;
import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;
publicclass MainActivity extends Activity
{
 ListView list;
staticfinal String[] country = new String[]
{"Indian Bank", "Canara Bank", "State Bank of India", "Bank of Baroda", "IOB","Syndicate Bank","Karur Vysya Bank"};
 @Override
publicvoid onCreate(Bundle savedInstanceState)
 {
super.onCreate(savedInstanceState);
  setContentView(R.layout.activity_main);
  list = (ListView) findViewById(R.id.listView1);
  ArrayAdapter<Object> adapter = new
```

```
ArrayAdapter<Object>(this,android.R.layout.simple_list_item
_1, country);
 list.setAdapter(adapter);
 list.setOnItemClickListener(new OnItemClickListener()
 {
publicvoid onItemClick(AdapterView<?> parent, View v, int
position, long id) {
Toast.makeText(getApplicationContext(),((TextView)
v).getText(),   Toast.LENGTH_SHORT).show();
```

OUTPUT:

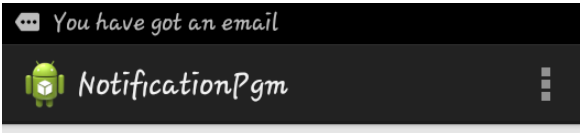


NOTIFICATIONS IN ANDROID – CREATION AND DISPLAY

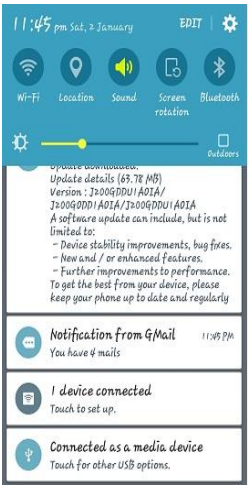
Notification is mainly used to alert the users that some events is occurring in the app. Notifications will be displayed outside the app's user Interface and it alerts the users without interrupting the current tasks which they are performing. In android, different ways the developer uses to create

notifications, in different forms like a flash, making sounds, or displaying small icons etc.

First, when system wants to display certain notification, then first it will display an icon in notification bar like as shown below.



To see the details of our android app notification, it is necessary to open the notification drawer as shown below.



Create a Notification in Android

To create a notification, it is necessary to specify the User Interface and the required actions. To create a notification, we need to specify the UI content and required actions. In Builder object the following three properties needs to be set to display the icon, title and the detailed text of notification.

setSmallIcon() - This method is used to set the small icon be to be displayed as a notification.

setContentTitle() - This method is used to set the title of the notification.

setContentText()-This method is used to set the detailed text to be displayed in the notification.

The above-mentioned properties are necessary to display a notification and it is also possible to set a different properties like setStyle, setSound, setLights, setLargeIcon etc in the notification based on the requirements.

Define the Android Notification Actions

PendingIntent object is used to define an action inside notification. If action has been assigned to notification, then it is possible for the user to perform action when they are clicking on the notification.

Once the notification creation has been done, it is necessary to pass a notification to the system by using NotificationManager.Notify() method, assigning ID for notification is also important, because by specifying the ID only updation to the notification will take place.

Android Notification Example

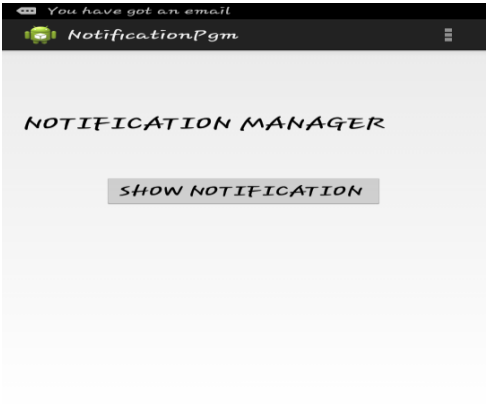
Following is the example of implementing Notifications in the android application.

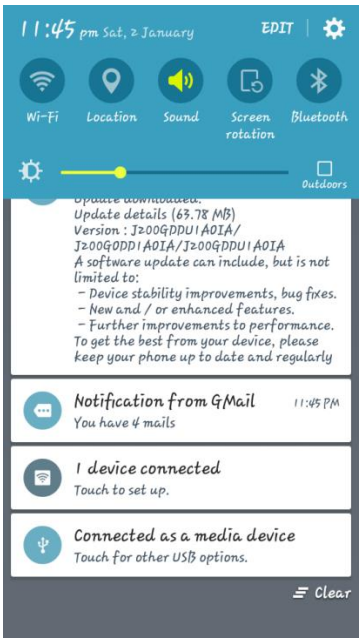
```
package com.example.notificationpgm;

import android.os.Bundle;

import android.app.Activity;

import android.app.Notification;
```

```
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Button bt=(Button)findViewById(R.id.button1);
bt.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
NotificationManager
nm=(NotificationManager)getSystemService(NOTIFICATIO
N_SERVICE);
Notification notification=new
```

```
Notification(android.R.drawable.stat_notify_more,"You have got an email",System.currentTimeMillis());

Context context=MainActivity.this;

Intent intent=new Intent(context,MainActivity.class);

PendingIntent pending=PendingIntent.getActivity(getApplicationContext(),0,intent,0);

Notification.setLatestEventInfo(context,"Notification from

GMail","You have 4 mails",pending);

nm.notify(0,notification);

}

});

}

@Override

publicboolean onCreateOptionsMenu(Menu menu)

{

getMenuInflater().inflate(R.menu.main, menu);

returntrue;

}

}
```

OUTPUT





INPUT CONTROLS

1.Buttons

2. Text Fields

3. Check boxes

4. Alert dialogs
5. Spinners
6. Rating bar
7. Progress bar

BUTTONS

Android Button

When the user clicks or tap on the button then this control will perform an action. Button is one of the most used User Interface control to develop applications. Button in android contains a text or an icon or both, so that the user can identify for what particular purpose that button is present in app. Different types of buttons is present in Android which will be used based on requirement, they are ImageButton, ToggleButton and RadioButton.

In Android, for button control coding will be written in two places. One in XML file to create a button, and another part of coding in MainActivity.java file which contains coding for what action to be performed when the user clicks on the button. Button control can be created in two ways, either simply by dragging and dropping the component from tool box or by writing code in XML file.

Following is the pictorial representation of usingButtons in android applications. Button with text, Button with image

Android Button Control Attributes

1.android:id

This attribute is used to set the unique id for each component.

2.android:gravity

This attribute is used to set where to place the control either left, right, bottom, top, center etc.

3.android:text

This attribute is used to set the text.

4.android:textColor

This attribute is used to change the colour of the text.

5.android:textSize

This attribute is used to increase or decrease the text size.

6.android:tetxStyle

This attribute is used to set the style for the text.

7.android:background

This attribute is used to set the background colour for the button control.

8.android:padding

This attribute is used to set the padding from left, right, top, bottom.

9.android:drawableBottom

When this attribute is used then the text will appear first in the button, the icon will appear below the given text.

10.android:drawableRight

When this attribute is used then the text will appear first in the button, the icon will appear right to the text.

11.android:drawableLeft

When this attribute is used then the icon will appear first in the button, then the given text.

Android Button Control Example

activitymain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
android:orientation="vertical"
android:layout_width="match_parent"
android:layout_height="match_parent">
android:id="@+id/fstTxt"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:layout_marginTop="150dp"
android:text="First Number" />
<EditText
android:id="@+id/firstNum"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
```

```
android:ems="10" />
<TextView
android:id="@+id/secTxt"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Second Number"
android:layout_marginLeft="100dp" />
<EditText
android:id="@+id/secondNum"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:ems="10" />
<Button
android:id="@+id/addBtn"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="100dp"
android:text="Add" />
</LinearLayout>
```

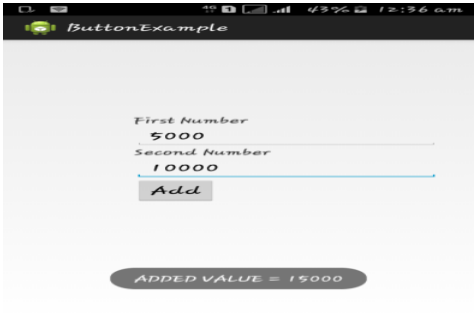
MainAcivity.java

```
package com.example.buttonexample;
import android.annotation.SuppressLint;
import android.app.Activity;
```

```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
@SuppressLint("NewApi")
publicclass MainActivity extends Activity
 {
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
final EditText firstNum =
(EditText)findViewById(R.id.firstNum);
final EditText secNum =
 (EditText)findViewById(R.id.secondNum);
Button btnAdd = (Button)findViewById(R.id.addBtn);
btnAdd.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
if(firstNum.getText().toString().isEmpty() ||
```

```
secNum.getText().toString().isEmpty())
{
Toast.makeText(getApplicationContext(), "Please fill all the fields", Toast.LENGTH_SHORT).show();
}
else
{
int num1 = Integer.parseInt(firstNum.getText().toString());
int num2 = Integer.parseInt(secNum.getText().toString());
Toast.makeText(getApplicationContext(), "ADDED VALUE = " + (num1 + num2),
Toast.LENGTH_SHORT).show();
}
}
});
}
}
```

OUTPUT



TEXT FIELDS

EditText is the user interface control, by using which the user can edit or modify the text. Specifying inputType attribute is necessary while using this control.

For example, if the EditText control should accept plain text then the inutType for this attribute must be “text” else if the EditText is going to get the password from the user then the inputType for this attribute must be “textPassword”.

EditText control is an extended version of TextView control, and it has additional features and allows the users to enter input values. In Android the developer can create the EditText control in two ways, either simply by dragging and dropping the control from toolbox or by writing the coding in XML file.

Android EditText Attributes

1.android:id

This attribute is used to set the unique id for each component.

2.android:gravity

This attribute is used to set where to place the control either left, right, bottom, top, center etc.

3.android:text

This attribute is used to set the text.

4.android:textColor

This attribute is used to change the colour of the text.

5.android:textSize

This attribute is used to increase or decrease the text size.

6.android:tetxStyle

This attribute is used to set the style for the text.

7.android:background

This attribute is used to set the background colour for the button control.

8.android:hint

When the text is empty then this attribute is used to display the hint.

9.android:width

This attribute is used to set the width of the EditText control

10.android:height

This attribute is used to set the height of the EditText control

11.android:textAllCaps

This attribute is used to set the text in all Caps.

12.android:typeface

This attribute is used to specify the Typeface (Normal, Sans serif etc)

13.android:editable

This attribute takes only two values either true or false, when set to true it will allow the user to edit or modify the text, when set to false this control will not allow the user to modify the text.

Android Text Fields Example

Activitymain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/activity_main"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.edittextexample.MainActivity">
<EditText
android:id="@+id/editText1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentStart="true"
android:layout_alignParentTop="true"
android:layout_marginLeft="50dp"
android:layout_marginStart="50dp"
android:layout_marginTop="24dp"
```

```
android:ems="10"
android:hint="Name"
android:inputType="textPersonName"
android:selectAllOnFocus="true" />
<EditText
android:id="@+id/editText2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText1"
android:layout_alignStart="@+id/editText1"
android:layout_below="@+id/editText1"
android:layout_marginTop="19dp"
android:ems="10"
android:hint="Password_0_9"
android:inputType="numberPassword" />
<EditText
android:id="@+id/editText3"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText2"
android:layout_alignStart="@+id/editText2"
android:layout_below="@+id/editText2"
android:layout_marginTop="12dp"
android:ems="10"
```

```
android:hint="e_mail"
android:inputType="textEmailAddress" />
<EditText
android:id="@+id/editText4"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText3"
android:layout_alignStart="@+id/editText3"
android:layout_below="@+id/editText3"
android:layout_marginTop="18dp"
android:ems="10"
android:hint="Date"
android:inputType="date" />
<EditText
android:id="@+id/editText5"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText4"
android:layout_alignStart="@+id/editText4"
android:layout_below="@+id/editText4"
android:layout_marginTop="18dp"
android:ems="10"
android:hint="Contact_number"
android:inputType="phone" />
```

<Button

android:id="@+id/button"

style="@android:style/Widget.Button"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:layout_alignParentLeft="true"

android:layout_alignParentStart="true"

android:layout_below="@+id/editText5"

android:layout_marginTop="62dp"

android:text="Submit"

android:textSize="16sp"

android:textStyle="normal|bold" />

</RelativeLayout>

MainActivity.java

package com.example.edittextprogram;

import android.app.Activity;

import android.os.Bundle;

import android.view.View;

import android.widget.Button;

import android.widget.EditText;

import android.widget.Toast;

publicclass MainActivity extends Activity

{

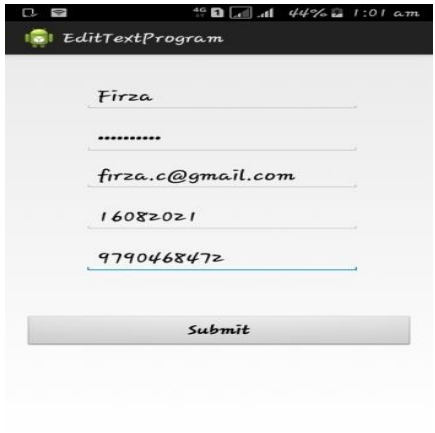
Button submit;

```
EditText name, password, email, contact, date;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
name = (EditText) findViewById(R.id.editText1);
password = (EditText) findViewById(R.id.editText2);
email = (EditText) findViewById(R.id.editText3);
date = (EditText) findViewById(R.id.editText4);
contact = (EditText) findViewById(R.id.editText5);
submit = (Button) findViewById(R.id.button);
submit.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
Toast.makeText(getApplicationContext(), "Name -  " +
name.getText().toString() + " \n" + "Password -  " +
password.getText().toString() + " \n" + "E-Mail -  " +
email.getText().toString() + " \n" + "Date -  " +
date.getText().toString()
+ " \n" + "Contact -  " + contact.getText().toString(),
Toast.LENGTH_SHORT).show();
}
```

});

}

OUTPUT:



CHECKBOXES

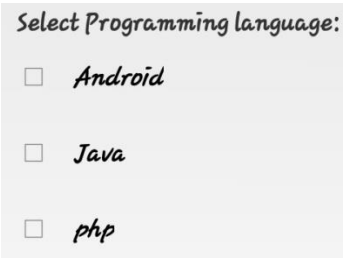
In Android, CheckBox is a two-state button which can be either checked (ON) or unchecked (OFF), it will allow the user to always switch between two states either ON or OFF. In order to allow the users to select more than one option from the set of values CheckBoxes will be used. While creating

Android Applications, the developer can use more than one checkboxes.

By default the CheckBox will always be in an Unchecked state. android:checked attribute is used to set the whether the checkbox should be checked or unchecked, if the android:checked attribute is set to true, then the check box will be in checked state else it will be in unchecked state.

The developer can use the CheckBox by using these two ways. Either they can simply drag and drop the checkbox from toolbox in their application, or the developer will write the coding in XML file to create a checkbox.

Following is the pictorial representation of using **CheckBox** control in android applications.



Android CheckBox Control Attributes

1.android:id

This attribute is used to set the unique id for each component.

2.android:gravity

This attribute is used to set where to place the control either left, right, bottom, top, center etc.

3.android:text

This attribute is used to set the text.

4.android:textColor

This attribute is used to change the colour of the text.

5.android:textSize

This attribute is used to increase or decrease the text size.

6.android:tetxStyle

This attribute is used to set the style for the text.

7.android:background

This attribute is used to set the background colour for the CheckBox control.

8.android:padding

This attribute is used to set the padding from left, right, top, bottom.

9.android:checked

This attribute is used to specify the current state of the check box.

10.android:onClick

This attribute holds the name of the method to invoke when the checkbox is checked.

11.android:visibility

This attribute is used to control the visibility of the checkbox.

Checkboxes Example Program:

activitymain.xml

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

```
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity">
<TextView
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Select Programming language:"
android:textSize="20sp"
android:textStyle="bold" />
<LinearLayout
android:id="@+id/linearLayout"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:layout_marginTop="30dp"
android:orientation="vertical">
<CheckBox
android:id="@+id/androidCheckBox"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:layout_centerHorizontal="true"
android:checked="false"
android:padding="20dp"
android:text="Android"
android:textSize="20sp"
android:textStyle="bold|italic" />
<CheckBox
android:id="@+id/javaCheckBox"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:checked="false"
android:padding="20dp"
android:text="Java"
android:textSize="20sp"
android:textStyle="bold|italic" />
<CheckBox
android:id="@+id/phpCheckBox"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:checked="false"
android:padding="20dp"
android:text="php"
```

```
android:textSize="20sp"
android:textStyle="bold|italic" />
<CheckBox
android:id="@+id/pythonCheckBox"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:checked="false"
android:padding="20dp"
android:text="Python"
android:textSize="20sp"
android:textStyle="bold|italic" />
<CheckBox
android:id="@+id/unityCheckBox"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:checked="false"
android:padding="20dp"
android:text="Unity"
android:textSize="20sp"
android:textStyle="bold|italic" />
</LinearLayout>
```

</RelativeLayout>

MainActivity.java

```
package com.example.checkboxprogram;

import android.app.Activity;

import android.graphics.Color;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.widget.Button;

import android.widget.CheckBox;

import android.widget.Toast;

publicclass MainActivity extends Activity implements
View.OnClickListener

{

CheckBox android, java, python, php, unity3D;

@Override

protectedvoid onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

android = (CheckBox)

findViewById(R.id.androidCheckBox);
android.setOnClickListener(this);
```

```
java = (CheckBox) findViewById(R.id.javaCheckBox);
java.setOnClickListener(this);
python = (CheckBox) findViewById(R.id.pythonCheckBox);
python.setOnClickListener(this);
php = (CheckBox) findViewById(R.id.phpCheckBox);
php.setOnClickListener(this);
unity3D = (CheckBox) findViewById(R.id.unityCheckBox);
unity3D.setOnClickListener(this);
}
@Override
publicvoid onClick(View view)
{
if (android.isChecked())
{
Toast.makeText(getApplicationContext(), "Android",
Toast.LENGTH_LONG).show();
}
if (java.isChecked())
{
Toast.makeText(getApplicationContext(), "Java",
Toast.LENGTH_LONG).show();
}
if (php.isChecked())
{
```

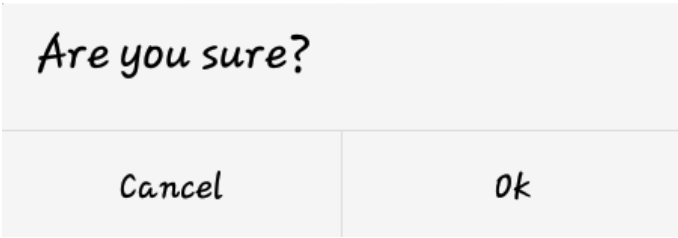
```
Toast.makeText(getApplicationContext(), "PHP",
Toast.LENGTH_LONG).show();
}
if (python.isChecked())
{
Toast.makeText(getApplicationContext(), "Python",
Toast.LENGTH_LONG).show();  }
if (unity3D.isChecked())
{
Toast.makeText(getApplicationContext(), "Unity 3D",
Toast.LENGTH_LONG).show();
}
}
}
```

OUTPUT:

ALERT DIALOGS

Android Dialog is a small window which will prompt messages to the user to make a decision. Generally, this dialogs are used with events it will prompt messages based on which the user selects what action to perform further in the application.In AlertDialog messages and buttons will be there, when prompted the user will read the message, and selects an action by clicking on the desired button.

Following is the pictorial representation of using dialogs in android applications.



The following is the different types of Dialog available in Android:

1.AlertDialog

This dialog is used to display message to the user with title, content and upto three buttons.

2.DatePickerDialog

This is a predefined dialog box and it allows the user to pick date.

3.TimePickerDialog

This is a predefined dialog box and it allows the user to pick time.

The three regions present in an Alert Dialog and it's description are as follows:

1.Title

This is optional, it is used to show the title of the AlertDialog.

2.Content Area

This area is used to display message, based on requirements.

3.Action Buttons

Maximum 3 buttons are allowed for AlertDialog they are Positive, Negative and Neutral.

Android AlertDialog Methods

1.setTitle()

This method is used to set the title for the AlertDialog and is the optional one.

2.setIcon()

This method is used to set the small icon to the alert dialog, this icon will usually appear before the title.

3.setMessage()

This method is very important and is used to set the message to be displayed in AlertDialog.

4.setCancelable()

After setting this method, if the user clicks anywhere in the application outside the alert dialog, then the alertdialog will be cancelled.

5.setPositiveButton()

As the name indicates this method is used to set the positive button in an alert dialog, and in Android it is possible to implement click event of a positive button.

6.setNegativeButton()

This method is used to set the Negative button in an alert dialog, and in Android it is possible to implement click event of a negative button.

7.setNeutralButton()

This method is used to set the neutral button in an alert dialog, and in Android it is possible to implement click event of a neutral button.

Android AlertDialog Example

activitymain.xml

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout_width="match_parent"

android:layout_height="match_parent"

android:paddingBottom="@dimen/activity_vertical_margin"

android:paddingLeft="@dimen/activity_horizontal_margin"

android:paddingRight="@dimen/activity_horizontal_margin"

android:paddingTop="@dimen/activity_vertical_margin"

tools:context="com.example.alertdialog.MainActivity">

<Button

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:id="@+id/button"
android:text="Close app"/>
</RelativeLayout>
```

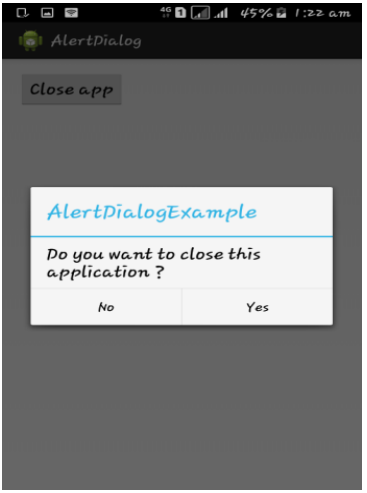
MainActivity.java

```
package com.example.alertdialog;
import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.app.Activity;
import android.app.AlertDialog;
import android.widget.Toast;
public class MainActivity extends Activity
{
Button closeButton;
AlertDialog.Builder builder;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
closeButton = (Button) findViewById(R.id.button);
```

```
builder = new AlertDialog.Builder(this);
closeButton.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
{
builder.setMessage(R.string.dialog_message)
.setTitle(R.string.dialog_title);
builder.setMessage("Do you want to close this application ?")
.setCancelable(false)
.setPositiveButton("Yes", new
DialogInterface.OnClickListener()
{
public void onClick(DialogInterface dialog, int id)
{
finish();
Toast.makeText(getApplicationContext(),"You choose yes
action for alertbox",
Toast.LENGTH_LONG).show();
}
})
.setNegativeButton("No", new
DialogInterface.OnClickListener()
{
```

```
public void onClick(DialogInterface dialog, int id)
{
dialog.cancel();
Toast.makeText(getApplicationContext(),"You choose no action for alertbox",
Toast.LENGTH_SHORT).show();
}
});
AlertDialog alert = builder.create();
alert.setTitle("AlertDialogExample");
alert.show();
}
});
}  }
```

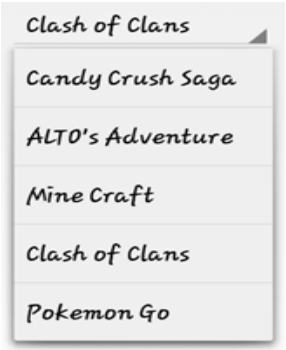
AlertDialog Output



SPINNERS

Android Spinner is like a combo box of Abstract Window Toolkit, It is possible in Android to display multiple options at the same time by using Spinner, out of which only one option the user can select at a time. In more simple words, it is like a drop down menu. Android spinner is associated with Adapter View. Because of this it is necessary to use any one of the Adapter class with spinner. Out of all the displayed values, the user can select any one value simply by clicking on it.

Following is the pictorial representation of spinner



Android Spinner Example

activitymain.xml

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:layout_width="match_parent"

android:layout_height="match_parent"

android:paddingBottom="@dimen/activity_vertical_margin"

android:paddingLeft="@dimen/activity_horizontal_margin"

android:paddingRight="@dimen/activity_horizontal_margin"

android:paddingTop="@dimen/activity_vertical_margin"

tools:context="com.example.spinnereg.MainActivity" >

<TextView

android:id="@+id/textView1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignLeft="@+id/spinner1"

android:layout_alignParentTop="true"

android:layout_marginTop="24dp"

android:text="Select your favourite game"

android:textAppearance="?android:attr/textAppearanceLarge"
/>

<Spinner

android:id="@+id/spinner1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/textView1"

android:layout_centerHorizontal="true"

android:layout_marginTop="46dp" />

</RelativeLayout>

MainActivity.java

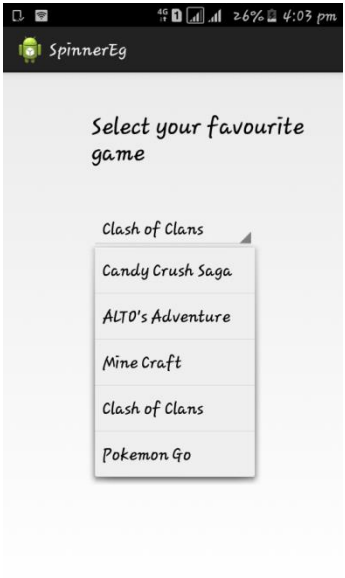
package com.example.spinnereg;

import android.app.Activity;

```
import android.os.Bundle;

import android.view.View;

import android.widget.AdapterView;

import android.widget.ArrayAdapter;

import android.widget.Spinner;

import android.widget.Toast;

publicclass MainActivity extends Activity implements
AdapterView.OnItemSelectedListener

{

String[] games={"Candy Crush Saga","ALTO's
Adventure","Mine Craft","Clash of Clans","Pokemon Go"};

@Override

protectedvoid onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);

Spinner spin = (Spinner) findViewById(R.id.spinner1);

spin.setOnItemSelectedListener(this);

ArrayAdapter aa = new

ArrayAdapter(this,android.R.layout.simple_spinner_item,gam
es);

aa.setDropDownViewResource(android.R.layout.simple_spin
ner_dropdown_item);

spin.setAdapter(aa);

}
```

```
@Override
publicvoid onItemSelected(AdapterView<?> arg0, View arg1,
int position,long id)
{
Toast.makeText(getApplicationContext(), games[position],
Toast.LENGTH_LONG).show();
}
@Override
publicvoid onNothingSelected(AdapterView<?> arg0)
{
}
}
```

OUTPUT



RATING BAR

Rating Bar is one of the User Interface control available in Android, in order to get the rating from the user. This control shows a rating in stars and it allows users to set the rating value simply by touching the stars. The Android rating bar will always return a value in the floating point number, such as 1.0,2.0,3.0 etc.

The attribute which is mainly used for rating bar is the android:numStars, by using this attribute the developer can set the number of stars to be displayed in the rating bar. This rating bar is maximum used in movies or products site to get the rating from the user for movies and products.

The header will needs to be imported to work with rating bar is **android.widget.RatingBar.**

Following is the pictorial representation of Rating Bar

Following are some of the commonly used attributes related to RatingBar control in android applications.

1.android:id

This attribute is used to set the unique id for each component.

2.android:numStars

This attribute is used to define a number of stars to display.

3.android:rating

This attribute is used to set the default rating value for the rating bar.

4.android:background

This attribute is used to set the background colour for a rating bar.

5.android:padding

This attribute is used to set the padding for left, right, top and bottom value for a rating bar.

Android Rating Bar Control Example

activitymain.xml

<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

```
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity">
<RatingBar
android:id="@+id/simpleRatingBar"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_marginTop="60dp"
android:paddingLeft="5dp"
android:paddingRight="5dp" />
<Button
android:id="@+id/submitButton"
android:layout_width="200dp"
android:layout_height="wrap_content"
android:layout_centerInParent="true"
android:padding="10dp"
android:text="Submit"
android:textSize="20sp"
android:textStyle="bold" />
```

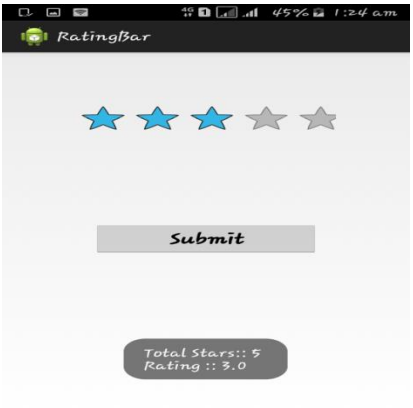
</RelativeLayout>

MainActivity.java

```
package com.example.ratingbar;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.RatingBar;
import android.widget.Button;
import android.widget.Toast;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
final RatingBar simpleRatingBar
= (RatingBar) findViewById(R.id.simpleRatingBar);
Button submitButton = (Button)
findViewById(R.id.submitButton);
```

```
submitButton.setOnClickListener(new
View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
String totalStars = "Total Stars:: " +
simpleRatingBar.getNumStars();
String rating = "Rating :: " + simpleRatingBar.getRating();
Toast.makeText(getApplicationContext(), totalStars + "\n" +
rating, Toast.LENGTH_LONG).show();
}
});
}
}
```

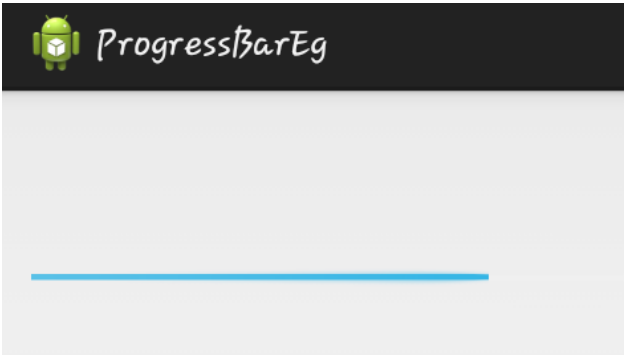
OUTPUT



PROGRESS BAR

In Android, this user interface control is used to indicate the progress of an operation. For example, downloading a file and uploading file etc.

Following is the pictorial representation of Progress Bar



Android ProgressBar with Determinate Mode

Determinate progress mode is mainly used to show the quantity of progress occurred. For example, to show the exact quantity of file uploaded or downloaded, to show the number of rows present in the database etc.

The determinate progress bar mainly uses the attribute **android:progress** in order to show the progress. Widget_ProgressBar_Horizontal or progressBarStyleHorizontalis used to set a determinate progress bar. When the progress value reaches 100, then the progress bar is full. **android:max** attribute is used to set the default value.

Android ProgressBar with Indeterminate Mode

This indeterminate progress bar is used when the user don't know how much time it will take to complete an operation. The actual progress will not be shown in the

indeterminate progress bar, only the cyclic animation will be shown.

Android ProgressBar Control Attributes

1.android:id

This attribute is used to set the unique id for each control.

2.android:max

This attribute is used to set the maximum value which the progress bar can take.

3.android:progress

This attribute is used to specify the default progress value.

4.android:background

This attribute is used to set the background colour for the progress bar.

5.android:indeterminate

In order to enable indeterminate progress mode this attribute is enabled.

6.android:padding

This attribute is used to set the padding from left, right, top and bottom.

7.android:minHeight

This attribute is used to set the height of the progress bar.

8.android:minWidth

This attribute is used to set the width of the progress bar.

ProgressBar Example Program

activitymain.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity">

<ProgressBar
android:id="@+id/simpleProgressBar"
style="@style/Widget.AppCompat.ProgressBar.Horizontal"
android:layout_width="fill_parent"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_marginTop="70dp"/>

<Button
android:id="@+id/startButton"
android:layout_width="200dp"
```

```
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_marginTop="120dp"
android:padding="10dp"
android:text="Start"
android:textSize="30sp"
android:textStyle="bold" />

</RelativeLayout>
```

MainActivity.java

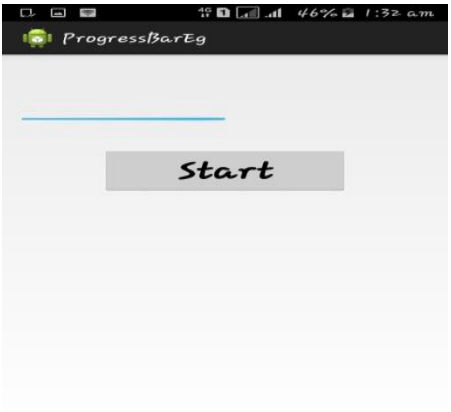
```
package com.example.progressbareg;
import android.app.Activity;
import android.app.Dialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.provider.Settings;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.widget.ProgressBar;
import android.widget.Button;
```

```
import android.widget.Toast;
public class MainActivity extends Activity
{
int progress = 0;
ProgressBar simpleProgressBar;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
simpleProgressBar = (ProgressBar)
findViewById(R.id.simpleProgressBar);
Button startButton = (Button)
findViewById(R.id.startButton);
startButton.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
{
setProgressValue(progress);
}
});
}
private void setProgressValue(final int progress)
```

```
{
simpleProgressBar.setProgress(progress);
Thread thread = new Thread(new Runnable()
{
@Override
public void run()
{
try
{
Thread.sleep(100);
}
catch (InterruptedException e)
{
e.printStackTrace();
}
setProgressValue(progress + 10);
}
});
thread.start();
}
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.menu_main, menu);
```

```
return true;
}
@Override
public boolean onOptionsItemSelected(MenuItem item)
{
int id = item.getItemId();
if (id == R.id.action_settings)
{
return true;
}
return super.onOptionsItemSelected(item);
}    }
```

OUTPUT



UNIT- V

PUBLISHING AND INTERNATIONALIZING MOBILE APPLICATIONS

LIVE MOBILE APPLICATIONS DEVELOPMENT

1.Games

2. Clock

3. Calendar

4. Converter

5. Phone Book

GAMES

Android Games

1. Flag Quiz Game App

2. Cannon Game App

Flag Quiz Game App

The Flag Quiz Game App is an interesting game which the developer has designed using Android, this game is mainly used to test the user's ability to correctly identify country flags. Like multiple choice questions, this app presents the user with a flag image and three possible answers. This app displays the user's progress throughout the quiz, showing the quiz number in a TextView.

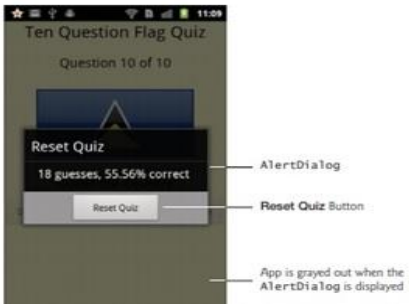
User Making a Correct Selection

The user chooses the country flag image by touching the corresponding button. If user selects the correct country

name then the same will be displayed in Green Colour with exclamation point. After few seconds, the app loads the next flag and displays a new set of options.

Completing the QUIZ

As soon as the user completes the quiz, a popup AlertDialog will be displayed over the app and shows the user's total number of correct guesses and the percentage obtained. In order to start a new quiz the user should touch the Reset Quiz button.

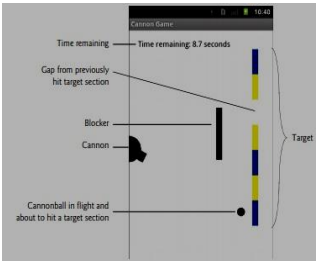


Cannon Game App

This is also one game which has been developed by using android. This game challenges the player to destroy a seven-piece target before a ten-second time limit expires. A

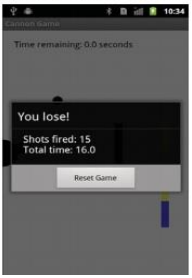
cannon that you control, a cannon ball, the target and a blocker are the four visual components of this game.

The player aims at the cannon by touching the screen – the cannon then aims at the touched point. From cannon, a ball will be fired when the player double clicks on the screen. Once the game has been finished, the app displays an Alert Dialog which indicates whether you won or lost, and also shows the number of shots fired and the total time taken.



a) `AlertDialog` displayed after user destroys all seven target sections

b) `AlertDialog` displayed when game ends before user destroys all seven target sections



Technologies Overview

1. Attaching a Custom View to the Layout
2. Using the resource folder raw
3. Activity LifeCycle Methods onPause and onDestroy
4. Overriding Activity Method on TouchEvent
5. Adding Sounds with SoundPool and AudioManager
6. Drawing Graphics using Paint and Canvas

1.Attaching a Custom View to the Layout

It is possible in Android to add Custom View, simply by extending the View Class.

2.Using the resource folder raw

Media files such as sound files used in the Game are placed in the folder res/raw.

3.Activity LifeCycle Methods onPause and onDestroy

Methods onPause is called when the other Activity receives the focus, which sends the current activity to the background. When an activity is ShutDown its onDestroy method is called. This method is used to release the app's resources.

4.Overriding Activity Method on TouchEvent

User interacts with the app by touching the device screen. In order to make sure that when the user single taps or double clicks the on the screen then the onTouchEvent method must be overridden, android.view package is needed to perform all these operations.

5. Adding Sounds with SoundPool and AudioManager

It is not possible in Android to develop any game without sound effect, Sound effect of an app will be managed with a SoundPool, to load play and unload sound, to work with this it is necessary to import android.media package. sounds.setVolumeControlStream method is used to control the volume.

6.Drawing Graphics using Paint and Canvas

For designing games for sure it is necessary to work with graphics. It is necessary to import android graphics

package to draw text, lines and circles. Each drawing method present in a class canvas uses an object to specify draw characteristics which includes colour, line thickness, font size and more.

CLOCK

In android, TextClock, AnalogClock and DigitalClock these three different clocks are available, it is a user interface control which is used to show the current time. These three clocks provide a time in two different formats, one is in 12-hour format and another is in 24-hour format. It depends on the developer, in which time format the developer needs the time to be displayed and they have to write the coding according to that.

Following is the pictorial representation of android Analog Clock

Create Android TextClock in XML Layout File

In android, it is possible to create Analog Clock in XML layout file using **<AnalogClock>** element with different attributes like as shown below.

<AnalogClock
android:id="@+id/analogClock"

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:format12Hour="hh:mm:ss " />

Android Clock Attributes

1.android:id

This attrbute is used to uniquely identify the control.

2.android:format12Hour

Whent this attribute is used then the clock will display time in 12 Hour format.

3.android:format24Hour

Whent this attribute is used then the clock will display time in 24 Hour format.

4.android:gravity

This attribute is used to specify how the alignment should be done for the component either left, right, top, bottom, center etc.

5.android:textColor

This attribute is used to change the colour of the text.

6.android:textSize

This attribute is used to increase or decrease the text size.

7.android:tetxStyle

This attribute is used to set the style for the text.

8.android:background

This attribute is used to set the background colour for the clock control.

9.android:padding

This attribute is used to set the padding from left, right, top, bottom.

Android Clock Example 1 – Display of Analog Clock

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.clockpgm.MainActivity" >

<AnalogClock
android:id="@+id/analogClock1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="133dp" />

<TextView
```

android:id="@+id/textView1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignParentTop="true"

android:layout_centerHorizontal="true"

android:layout_marginTop="61dp"

android:text="Android Analog Clock"

android:textAppearance="?android:attr/textAppearanceLarge"
/>

</RelativeLayout>

OUTPUT:

Example Program 2 –Talking Analog Clock

MainActivity.java

package com.example.talkingclock;

import android.os.Bundle;

import android.app.Activity;

```
import java.util.Calendar;
import android.speech.tts.TextToSpeech;
import android.speech.tts.TextToSpeech.OnInitListener;
import android.view.View;
import android.widget.AnalogClock;
import android.widget.TextView;
publicclass MainActivity extends Activity implements OnInitListener
{
TextToSpeech Talktome;
AnalogClock Clock2;
TextView ReadText;
privateint mHour, mMinute;
publicvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Talktome = new TextToSpeech(this, this);
Clock2 = (AnalogClock)findViewById(R.id.AnalogClock);
Clock2.setOnClickListener(MyAnalogClockOnClickListener)
;
ReadText = (TextView)findViewById(R.id.Text2Read);
}
publicvoid onInit(int status)
```

```
{
// TODO Auto-generated method stub
}
 @Override
protectedvoid onDestroy()
{
 // TODO Auto-generated method stub
super.onDestroy();
Talktome.shutdown();
 }
Private Analog Clock. On Click Listener My Analog Clock On Click Listener
  = new AnalogClock.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
final Calendar c = Calendar.getInstance();
mHour = c.get(Calendar.HOUR_OF_DAY);
mMinute = c.get(Calendar.MINUTE);
String myTime= "The Time is  is "
+ String.valueOf(mHour)
+ " Hour "
+String.valueOf(mMinute)
```

+ " Minute";

ReadText.setText(myTime);

Talktome.speak(myTime, TextToSpeech.QUEUE_FLUSH, null);

}

};

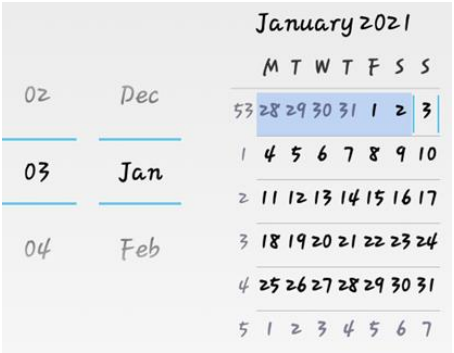
}

OUTPUT:

CALENDAR (DATE PICKER)

In Android, Calendar is a control which allows the user to select the date by day, month and year from application user interface. If this control has been used in app, it will ensure that the users will select a valid date.

Following is the pictorial representation of using Calendar in android applications.



Create Android DatePicker in XML Layout File

In android, we can create a Calendar control in XML layout file using **<DatePicker>** element with different attributes like as shown below

<DatePicker android:id="@+id/DatePickerControl1"
android:layout_width="wrap_content"
android:layout_height="wrap_content" />

In android, the DatePicker supports two types of modes, those are **Calendar** and **Spinner**.

Android DatePicker with Calendar Mode

We can define android DatePicker to show only a calendar view by using

android:datePickerMode attribute. Following is the example of showing the DatePicker in **Calendar** mode.

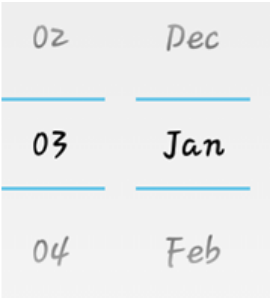
<DatePicker
android:id="@+id/datePicker1"

android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:datePickerMode="calendar"/>

Following is the example of showing the DatePicker in CalendarMode.

Android DatePicker with Spinner Mode

It is possible in Android to show the date picker in Spinner mode, in order to do that the simple step which the developer has to do is to set the datePickerMode to Spinner in XML coding file.



Following is the example of showing the DatePicker in SpinnerMode.

```
<DatePicker
    android:id="@+id/datePicker1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:datePickerMode="spinner"/>
```

To get only spinner mode date selection, then we need to set **android:calendarViewShown="false"** attribute in DatePicker control like as shown below.

```
<DatePicker
   android:id="@+id/datePickerCtrl1"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:datePickerMode="spinner"
   android:calendarViewShown="false"/>
```

Android DatePicker Attributes

1.android:id

This attribute is used to uniquely identify the attribute.

2.android:datePickerMode

This attribute is used to specify in which mode the date picker control should be displayed, either in spinner or calendar mode.

3.android:background

This attribute is used to set the background colour for the datepicker control.

4.android:padding

This attribute is used to set the padding of datepicker control from left, right, top, bottom etc.

EXAMPLE PROGRAM ANDROID CALENDAR

activitymain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent">
<DatePicker
android:id="@+id/datePicker1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_centerHorizontal="true"
android:layout_marginTop="20dp" />
<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_below="@+id/datePicker1"
android:layout_marginLeft="100dp"
android:text="Get Date" />
<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_below="@+id/button1"
android:layout_marginLeft="100dp"
android:layout_marginTop="10dp"
android:textStyle="bold"
android:textSize="18dp"/>
```

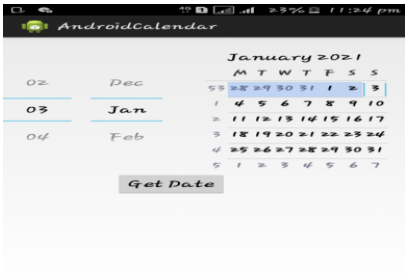
</RelativeLayout>

MainActivity.java

```
package com.example.androidcalendar;
import android.app.Activity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
DatePicker picker;
Button btnGet;
TextView tvw;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
tvw=(TextView)findViewById(R.id.textView1);
picker=(DatePicker)findViewById(R.id.datePicker1);
btnGet=(Button)findViewById(R.id.button1);
btnGet.setOnClickListener(new View.OnClickListener()
{
```

```
@Override
publicvoid onClick(View v)
{
tvw.setText("Selected Date: "+
picker.getDayOfMonth()+"/"+ (picker.getMonth() +
1)+"/"+picker.getYear());
}
});
}
}
```

OUTPUT:



CONVERTER

It is possible in Android, to convert text into speech with the help of TextToSpeech class. After completion of conversion, the user can playback the sound file, and it is also possible to download the sound file.

TexttoSpeech class plays an important role if the developer wants to do the conversion. Even this class supports different types of languages. It is necessary to select the speaking language based on the requirements in the android application.

TextToSpeech.OnInitListener, this **OnInitListener** method is used to indicate the completion of initialization. Based on requirements, it is possible to set the type of language to speak, audio pitch rate, audio speed etc.

Methods of TextToSpeech Class

1.addSpeech(String text,String filename)

This method takes two arguments and this method is used to create a mapping between a string of text and a sound file.

2.getLanguage()

This method is used to get the different languages which the app supports.

3.isSpeaking()

This method is used to check whether the TextToSpeech engine is busy in speaking or it is free.

4.setPitch(float pitch)

This method takes one argument and is used to set the pitch of the sound.

5.setSpeechRate(floatSpeechRate)

This method takes one argument and is used to set the speech rate.

6.shutdown()

This method is used to release the resource used by the TextToSpeech engine.

7.stop()

This method is used to stop the speech.

Example Program – This method is used to convert metre into centimetre

activitymain.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:id="@+id/activity_main"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="codedost.unitconverter.MainActivity">
<EditText
android:layout_width="match_parent"
```

android:layout_height="wrap_content"

android:inputType="numberDecimal"

android:hint="Enter the length in metres"

android:ems="10"

android:layout_alignParentTop="true"

android:layout_centerHorizontal="true"

android:layout_marginTop="84dp"

android:id="@+id/editText" />

<Button

android:text="Convert to centimetre"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignParentBottom="true"

android:layout_centerHorizontal="true"

android:layout_marginBottom="105dp"

android:id="@+id/button" />

<TextView

android:text="Centimetre"

android:layout_width="match_parent"

android:layout_height="wrap_content"

android:textSize="26sp"

android:layout_below="@+id/editText"

android:gravity="center"

android:layout_centerHorizontal="true"

```
android:layout_marginTop="82dp"
android:id="@+id/textView" />
</RelativeLayout>
```

MainActivity.java

```
package com.example.converterpgm;
import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class MainActivity extends Activity
{
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Button button;
final EditText met;
final TextView mtv;
button= (Button) findViewById(R.id.button);
met=(EditText) findViewById(R.id.editText);
```

```
mtv=(TextView) findViewById(R.id.textView);
button.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
{
Double convert=
Double.parseDouble(met.getText().toString());
mtv.setText(String.valueOf(convert*100));
mtv.setTextColor(Color.BLUE);
}
});
}
}
```

OUTPUT

PHONEBOOK

A phone book contains a list of names, addresses, contact number and other details of the people in alphabetical order. The simple operation which the user will do using phone app is retrieving a specific contact out of all the available contacts and making a call to that contact.

Android phone book is used to retrieve a contact from list of contact whose data matches the search string. The user will search either by giving the part of a search string, or giving a part of numbers.

Match contact names

This method is used to retrieve an exact data by matching the search string. The Content Provider allows multiple instances of the same name, this technique can return list of matches.

Match a specific type of data

This method is used to retrieve a list of contacts by matching the search string to a particular type of data such as e-mail address, dob etc, this technique retrieves a list of all contacts whose email address or dob matches the search string.

Match any type of data

This method is used to retrieve the data by matching any string not only number, contact number, e-mail address but also street address, dob and so forth. This technique allows the device to accept any type of data and displays contact which matches the search string.

Request permission to read the provider

Through the Android App which you have developed from that if you want to read all your mobile contacts, and also you want to make a call through that app, then it is necessary to write the coding in <manifest> file apart from the coding which will be present in MainActivity.java and activitymain.xml

The following coding App developer must write in android manifest file.

<uses-permission android:name="android.permission.READ_CONTACTS" />

In order to display the search results in a ListView, it is necessary to include a list view component while designing an app.

From android app it is possible to make a phone call from android applications by calling inbuilt call app using

Intent action (ACTION_CALL). Intent is a messaging object which is used to request an action from another app component such as **activities, services, broadcast receivers, and content providers.**

To make a phone call using Intent object in android application, it is necessary to write the following coding.

```
Intent callIntent = new Intent(Intent.ACTION_CALL);

callIntent.setData(Uri.parse(txtPhone.getText().toString()));

startActivity(callIntent);
```

Android Phone Call Example

CODING:

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"
android:layout_width="match_parent"

android:layout_height="match_parent">

<TextView

android:id="@+id/fstTxt"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:layout_marginTop="150dp"

android:text="Mobile No"/>
```

<EditText

android:id="@+id/editText1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:ems="10">

</EditText>

<Button

android:id="@+id/button1"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_marginLeft="100dp"

android:text="Call" />

</LinearLayout>

AndroidManifest.xml

<?xml version="1.0" encoding="utf-8"?>

<manifest
xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.callpermission"

android:versionCode="1"

android:versionName="1.0">

<uses-permission
android:name="android.permission.CALL_PHONE" />

<uses-sdk

android:minSdkVersion="8"

android:targetSdkVersion="18" />

<application

android:allowBackup="true"

android:icon="@drawable/ic_launcher"

android:label="@string/app_name"

android:theme="@style/AppTheme">

<activity

android:name=".MainActivity"

android:label="@string/app_name">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

MainActivity.java

package com.example.callpermission;

import android.net.Uri;

import android.os.Bundle;

import android.support.v7.app.ActionBarActivity;

```
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
publicclass MainActivity extends ActionBarActivity
{
EditText edittext1;
Button button1;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
edittext1=(EditText)findViewById(R.id.editText1);
button1=(Button)findViewById(R.id.button1);
button1.setOnClickListener(new OnClickListener()
{
@Override
publicvoid onClick(View arg0)
{
```

```
String number=edittext1.getText().toString();
Intent callIntent = new Intent(Intent.ACTION_CALL);
callIntent.setData(Uri.parse("tel:"+number));
startActivity(callIntent);
}
});
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
// Handle action bar item clicks here. The action bar will
// automatically handle clicks on the Home/Up button, so long
// as you specify a parent activity in AndroidManifest.xml.
int id = item.getItemId();
if (id == R.id.action_settings)
{
```

returntrue;

}

returnsuper.onOptionsItemSelected(item);

}

}

OUTPUT:

APP DEPLOYMENT AND TESTING

Publishing is the process of making your android applications available to users, to publish an Android applications it is necessary to perform two main tasks.

1. Preparing applications for release
2. Releasing the application to users

1. Preparing applications to release

While preparing an application for release the developer will prepare a release version of applications, which the users can download on their android powered devices.

2.Releasing the application to users

While releasing the app, the developer will publicize so that the app will reach maximum users, sell the app, distribute the release version of application to users.

App Deployment Checklist

1. Understand the developer program policies
2. Prepare your developer account
3. Plan for simultaneous releases
4. Test against the quality guidelines
5. Target a recent API Level
6. Run internal tests
7. Plan your app's Play Store listing
8. Upload your Android App Bundle to the closed or an open test track
9. Define your app's device compatibility
10. Set up your apps price and countries of distribution

1.Understand the Developer Program Policies.

Before deploying your app in Playstore it is necessary to go through the policies clearly, to make sure that the app which you are deploying is not violating any policies.

2.Prepare your developer account

Developer account creation is necessary, sign up for developer account to check your account details are accurate, if you want to sell the app which you have created then you have to create the Merchant account.

3.Plan for simultaneous releases

It is possible for the developer to deploy the app in multiple platform at the same time, the advantage is the app downloads will increase, the promotion activities and the number of installations will increase. The only thing which the developer has to keep in mind is if the developer has developed native applications then it will not become possible for them to deploy that applications same time in a multiple platform.

4.Test against the quality guidelines

App should satisfy the quality guidelines, in order to satisfy this test the developer should see that the app is providing basic User Interface design, features and functions expected by Android users. Testing template will differ for Android smart phones, TV and tabs.

5.Target a recent API level

Google play requires that the app which the developer develops, should be compatible at least with the version 9.0, the reason behind this is if the app is compatible with this version, then it's going to be compatible will all the lower versions.

6.Run internal tests

The developer should test the app properly before publishing it in the App Store/Play Store. Minimum recommendation is at least giving app to100 users and getting feedback from them and working according to that is very mandatory, before actually bringing the app to public platform.

7.Plan your apps Play Store Listing

If the user wants to download the app they will look for description of that app, graphics like screenshots about the app interface, videos etc, which as a developer you will load in the Play Store with your application, so that the downloader gets some clear idea about your app.

8.Upload your Android App Bundle to the closed or an open test track

Open and Close testing track is very necessary to discover issues with the app, it gives the developer an

opportunity to fix the issues and increase the quality of the app by fixing if any issues exists.

9.Define your apps device compatibility

It is necessary to give details that your app is compatible with which Android versions and screen size.

10.Set up your apps price and countries of distribution

Mention your app is free or paid for users, and also mention in which different countries the app should be published.

Releasing your app to users

1. Releasing through an app marketplace
2. Releasing your apps on Google Play
3. Releasing through a website

Releasing through an app Marketplace

If the developer wants to make their app reach to the maximum number of people then deploying that through App Store/Play Store is the best option. Google play is the best market place for Android Apps.

Releasing your apps on Google Play

Google Play is the famous publishing platform which helps the developer to publicize, sell and distribute applications to users around the world. Once the developer chooses this platform then this Google Play provides access to a set of developer tools that lets the developer analyze their application related activities.

Releasing through a Website

Not only in App Store/Play Store, the developer can deploy the apps on their own website, including a private or enterprise server. In order to do that, it is necessary for the developer to create application in normal way, this is the first step. Then the APK file application package can be hosted on the website and provide a download link to the users.

Normally the user will download the app from Play Store/App Store, since in this method the APK link is given in website, the user must go to that website and click on the link the device automatically starts downloading and installing the app.

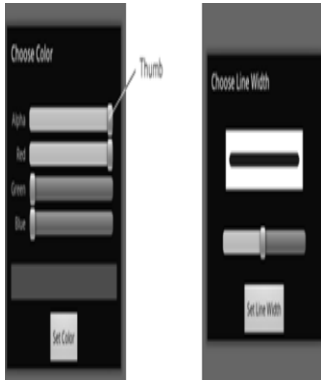
DOODLZ APP

There are lots of Doodlz app available in the play store. This app turns the user's mobile screen into virtual canvas. This app provides different options to users like setting drawing colour, line width etc. The Choose Colour option that is, ARGB that is Alpha, Red, Green and Blue colour provides precision. In order to control the thickness of the line, Choose Line Width option is available, a seekbar is used here to control the thickness of the line.

Objectives

The screen must detect if the user touches the screen, moves a finger across the screen, and removes the finger from the screen and multiple screen touches so that the user of an app can draw with multiple fingers at once. Paint object is used to specify the colour and thickness of a line, using toast to briefly display a message on the screen.

Choose Colour and Line Width dialog of the Doodlz App



Options available in Doodlz App

Technologies Overview

1.Using SensorManager to Listen for Accelerometer Events

2. Creating Custom Dialogs

3. Custom Colours

4. Drawing Lines and Paths

5. Processing Touch Events

6. Saving the Drawing to Device's Gallery

7. Using Toasts to display a Message for a Short Time

1.Using SensorManager to Listen for Accelerometer Events

Not only the app should recognize the finger on the screen, but also feature should be provided where if the user shakes the app then device should erase the current drawing.

2.Creating Custom Dialogs

Here, the custom dialogs is nothing but the AlertDialogs which is used to display some message to the user with buttons.

3.Custom Colors

The user can set a custom drawing Colour in the app by changing the Alpha, Red, Green and Blue values. The user can take this four colours directly and also they can get a new colour combination by setting values. The range value is from 0 to 255. As usual the 0 represents completely transparent colour and 255 represents opaque colour.

4.Drawing Lines and Paths

To draw different shapes like lines, paths etc, the developer should import android.graphics package. It is also important to associate a canvas with a bitmap, which can the then be displayed on the screen.

5.Processing Touch Events

The user can touch the screen with more than one finger, and drag the fingers to draw different objects. OnTouchEvent method must be overridden in order to process the touch events. This method is used to receive the MotionEvent that contains the type of touch event that has happen and the pointer which generated the event, in order to

perform all these operations the header file which needs to be imported is android.view

6.Saving the Drawing to the Device's Gallery

The app must provide a Save Image button and that should allows the user to save the created Doodlz in the device gallery. A ContentResolver class is used to perform this operation, it reads the data from the app and stores that in the device gallery. In order to work with ContentResolver class then importing of the package android.content is mandatory.

7.Using Toasts to Display a Message for a Short Time

Toast is used to display the message. The package android.widget is mandatory to work with the toast. A toast will display a message for a short time on the app screen and then it disappears. Toast are mostly used to display small error messages, information messages etc. The toast will indicate whether the image has been stored properly in the gallery or not.

TIP CALCULATOR APP

Android Tip Calculator is different from normal calculator, by using normal calculator we can perform basic mathematical operations like addition, subtraction, multiplication, division, square root etc. By using scientific calculator few more additional operations like finding scientific, trigonometric, logarithmic operations.

This **TIP CALCULATOR** is completely different, where when bill amount is entered and tip percentage is entered then this calculator calculates and returns the tip value, this is the exact purpose of TIP CALCULATOR.

Sample Tip Calculator Program

activitymain.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity">
<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentRight="true"
android:layout_alignParentTop="true"
android:layout_marginTop="47dp"
android:text="Enter Your Bill amount"
android:textAppearance="?android:attr/textAppearanceMediu
m"/>
```

```
<TextView
android:id="@+id/res"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentBottom="true"
android:layout_alignParentLeft="true"
android:layout_alignRight="@+id/button1"
android:layout_marginBottom="14dp"
android:text="Result : "
android:textAppearance="?android:attr/textAppearanceLarge"
/>
<Button
android:id="@+id/button1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_above="@+id/res"
android:layout_alignParentLeft="true"
android:layout_alignRight="@+id/textView2"
android:layout_marginBottom="34dp"
android:background="@android:color/black"
android:text="Calculate"
android:textColor="@android:color/white" />
<TextView
android:id="@+id/textView2"
```

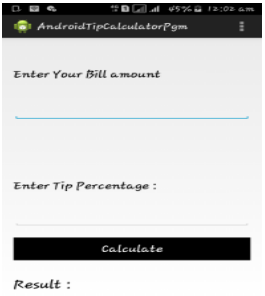
```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_above="@+id/button1"
android:layout_alignParentLeft="true"
android:layout_alignParentRight="true"
android:layout_marginBottom="96dp"
android:text="Enter Tip Percentage :"
android:textAppearance="?android:attr/textAppearanceMediu
m"/>
<EditText
android:id="@+id/bill_amt"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentRight="true"
android:layout_below="@+id/textView1"
android:layout_marginTop="41dp"
android:ems="10">
<requestFocus />
</EditText>
<EditText
android:id="@+id/bill_per"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:layout_above="@+id/button1"
android:layout_alignParentLeft="true"
android:layout_alignParentRight="true"
android:layout_marginBottom="22dp"
android:ems="10" />
</RelativeLayout>
```

ActivityMain.java

```
package com.example.androidtipcalculator;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class MainActivity extends Activity
{
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
final EditText amt = (EditText) findViewById(R.id.bill_amt);
final EditText tip = (EditText) findViewById(R.id.bill_per);
```

```
final TextView result = (TextView) findViewById(R.id.res);
Button calc = (Button) findViewById(R.id.button1);
calc.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
{
// TODO Auto-generated method stub
double amount = Double.parseDouble(amt.toString());
double tip_per = Double.parseDouble(tip.toString());
double tip_cal = (amount * tip_per) / 100;
result.setText("Result : " + Double.toString(tip_cal));
}
});
}
@Override
public boolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is
present.
getMenuInflater().inflate(R.menu.main, menu);
return true;
}
```



}

WEATHER VIEWER APP

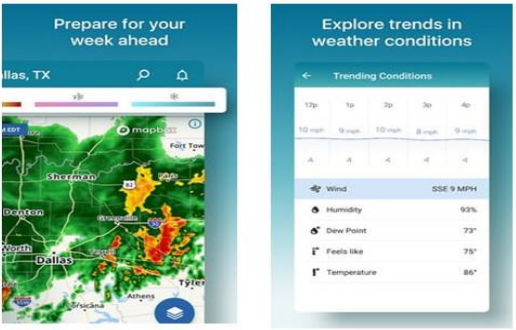
Weather Viewer is a free **weather application,** which when installed it will become possible for the user to see the weather forecast for upto 10 days. The best Weather apps for Android which is available in App Store freely are

1. The Weather Channel
2. AccuWeather
3. Weather Kitty
4. Flowx

Weather Channel

The Weather Channel is free app. This app once installed from PlayStore, automatically changes based on your location and provides the current weather plus hourly weather for the upcoming two days. Through this app it will become possible for the user to watch videos recorded from the Weather Television Channel's. Not only that this app also notify the user about the severe weather alerts.Dynamic home screen option is available to the Android users that changes based on time, location and weather conditions.

UI Screen of Weather Channel



AccuWeather

AccuWeather app is one app through which the user will get the information about weather conditions. This app is very famous for its simplicity and comprehensiveness. Upto two hours in advance this app gives the weather conditions, and also the minute-to-minute update, the name given to this feature is MinuteCast. Hourly forecasts are available for upto three days in advance.

The other feature present in this app is, this apps gives the information about the sunrise and sunset, as well as the current weather news and videos. It is also possible for the casual user of that app to send the weather condition related videos to this app.

UI Screen of AccuWeather

Weather Kitty

This app gives 10-days weather forecast in advance and for the next 24 hours the hourly update, not only that, this app also throws in the forecast to include more measurements like humidity, current moon phase, the day's high and low temperatures and feels, wind speed, wind chill, sunrise, sunset, barometric pressure, dew point and projected rain.

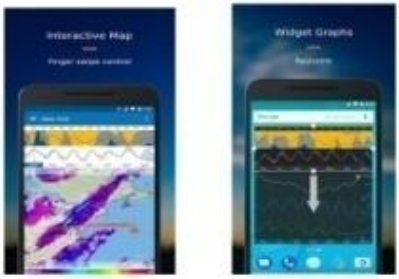
UI Screen of Weather Kitty

FlowX

The FlowX is one the unique app which provides accurate data about weather conditions for whole week's weather ahead in one go. Not much complicated things the user should do, simple swipe gestures let the user browse the whole week's weather forecasts. Through this app it is also possible to get the details of wind speed and direction.

There's a travel mode so you can **plan trips in advance**, as well as widgets with graphs for your home screen. A premium subscription gives you access to a longer 10-day forecast. The **paid plan** also offers additional data to consume, plus **the ability to edit weather graphs.** It is possible for the user to get the Pro mode either by paying a one-time price or subscribing per month.

UI Screen of FlowX



Software Used to Generate Weather Viewer App in Android

1. Android Studio
2. JSON (JavaScript Object Notation)
3. Eclipse ADT Bundle
4. OpenWeatherMap Application Programming Interface (API) key.

VIDEO LECTURE LINKS

UNIT- I LINKS

Introduction

Part 1

https://youtu.be/XogNh6yW6kk

Part 2

https://youtu.be/-WRsfK0JowY

Mobile operating system

Part 1

https://youtu.be/Gw9qObWYWCw

Part 2

https://youtu.be/mnzJsk3p3sE

Mobile Applications types

Part 1

https://youtu.be/leCLZyqrxV4

Part 2

https://youtu.be/4QZxNjMuUlc

Mobile Databases

https://www.loom.com/share/4e02738eddba417a8c2983aafcdcbcf5

Android Basics

Part 1

https://www.loom.com/share/fafdc7e5e8ea4e15b6190f9291ef7a3c

Part 2

https://www.loom.com/share/a861d81ad3ba4e98ba8bd87b2fa8939f

Android versions and compatibility

https://www.loom.com/share/87c55d1c9aab4800891a2a05ab1f50f4

Prerequisites to learn Android, How to install android etc

https://www.loom.com/share/89d238c127564722923daeffac340a48

Android architecture

Part 1

https://www.loom.com/share/921bdc80eb2e40828c6dd934d5dd9af6

Part 2

https://youtu.be/JzQt8Kp_nVc

Part 3

https://youtu.be/SkagItRM6PE

Part 4

https://youtu.be/iWZQGFa2EuI

Android emulator and XML

Part 1

https://youtu.be/fGWIQnC869A

Part 2

https://youtu.be/Vi40KaIAieU

Part 3

https://youtu.be/1XHEF1iGZuE

UNIT- II LINKS

Java for building android app

https://www.loom.com/share/708b2ecfd5384343be5814eacb78005d

Android studio and it's features

https://www.loom.com/share/a9a4da672acb49e7b8f1a1712e7077bb

Android studio

Part 1

https://youtu.be/LXip5ru-kvE

Part 2

https://youtu.be/cjXcoSuURMw

Eclipse

Part 1

https://youtu.be/Hnfyxk6nn6E

Part 2

https://youtu.be/kGUizudXvFk

Part 3

https://youtu.be/fuF5DEfNuRQ

Part 4

https://www.loom.com/share/545c5ef289cd494e89e593fb4c1ebca4

Part 5

https://www.loom.com/share/7f479c5ad108429e92ff48e598916a45

Virtualization and API

Part 1

https://youtu.be/Oih2-lIenv0

Part 2

https://youtu.be/Wa1EtITLU2w

Part 3

https://youtu.be/d91wJDNlueg

Android tools

Part 1

https://youtu.be/cuUpTTkjqow

Part 2

https://youtu.be/U3zdzflQW2A

Debugging with DDMS

Part 1

https://youtu.be/nhzFIR-35h8

Part 2

https://youtu.be/KwyVZuaxloM

Android file system

Part 1

https://youtu.be/EeCDXMYu6Rs

Part 2

https://youtu.be/kGeeYlRfk34

Android emulator

Part 1

https://youtu.be/IDE_-hC7UC8

Part 2

https://youtu.be/ssYPcJdXjRY

Anatomy of Android application

Part 1

https://youtu.be/JnfKARwnKnc

Part 2

https://youtu.be/Rpc7ikQbLu4

A basic android application deployment

Part 1

https://www.loom.com/share/fe03ecf1fe4c4d5f8ecec95f83619151

Part 2

https://www.loom.com/share/411c303bb44d4d6a95a2f73ed5868e9f

Activity life cycle methods and activity creation

Part 1

https://youtu.be/pfYDEQ1DhqY

Part 2

https://youtu.be/HabsaF7Or7k

Intent and intent filter

Part 1

https://youtu.be/oZZpodjhNQ4

Part 2

https://youtu.be/0OC8ujVciIk

Part 3

https://youtu.be/nqNMIKZdxXs

Activity Stack

https://youtu.be/bZaVCtdiYns

UNIT- III LINKS

Simple Services

Part 1

https://www.loom.com/share/7bbab72e8b5841fca7d24e6065d60531

Part 2

https://www.loom.com/share/bf09d2df7d2e44a09c228dbbc99fa109

Broadcast Receivers

(Creating and Managing Receiver, Receiver Intents, Ordered Broadcasts)

Part 1

https://www.loom.com/share/fbec4d821a014707bb12d7b3e280d2b5

Part 2

https://www.loom.com/share/b39727f1c28b40d785468a98ff887461

Part 3

https://www.loom.com/share/17e3fffb98c24754b1005227322b3bbb

Content Provider (Creating and Using Content Provider, Content Resolver)

Part 1

https://www.loom.com/share/b8c1b9f4565f414dab23dc0779ab1037

Part 2

https://www.loom.com/share/2b21fed3a5a142eeba87a8effdcb0298

Part 3

https://www.loom.com/share/a1a00e294b4d4aca8420d20b5e0c1ec7

Working with Database: SQLite

Part 1

https://youtu.be/27KMB39VNmI

Part 2

https://youtu.be/PLF0tb6zBGA

Part 3

https://youtu.be/TL9gRsWNEuw

Coding for sqlite using android (Building Sample Applications)

Part 1

https://www.loom.com/share/f94cb5ecaba84b0a94b2b4b6c0c40ed0

Part 2

https://www.loom.com/share/8bbbf445c07e4907ab11dca6f674f2e5

Data Analysis

Part 1

https://youtu.be/oJe0HZ2e7Os

Part 2

https://youtu.be/K53ySLVkpso

UNIT- IV LINKS

Android layout

Part 1

https://www.loom.com/share/1a6271f4b2cc418aa34930a661e5cd55

Part 2

https://www.loom.com/share/044ec2e60d22400d967cbc6cd8d95d4c

Part 3

https://www.loom.com/share/ec301a36b70249f68c668186a8540b4f

Part 4

https://www.loom.com/share/98d067c8b772416c977b8150ed36e9be

Part 5

https://www.loom.com/share/900531ab7cd54e199143d7dba4fd96e2

Android Menu - Android Options Menu

Part1

https://youtu.be/Uggouc0U7Yo

Part 2

https://youtu.be/X-0TV6G0R2k

Part 3

https://youtu.be/iIuXO9qkH1M

Context Menu

https://www.loom.com/share/146b8a2f60c345a099a00b299c71eb54

Pop Up Menu and Lists

https://www.loom.com/share/2a301afe5848434ebfd913e0c997685d

Notifications – Creation and Display

Part 1

https://www.loom.com/share/c04b81e8d17c4463b004c85e239f3241

Part 2

https://www.loom.com/share/c28dc0b197994f91874595b6b066eb13

Input Controls

Buttons and TextFields

Part 1

https://youtu.be/lAH4F-m5lF8

Part 2

https://youtu.be/5NkKk4u-DfE

Part 3

https://youtu.be/zsBsMt-vC5U

Part 4

https://youtu.be/X9TpvirmmSY

Checkboxes

Part 1

https://youtu.be/DoyBupBWhjQ

Part 2

https://youtu.be/mZJlVKU7jZQ

Part 3

https://youtu.be/e80IAswGw2E

Alert Dialogs

Part 1

https://youtu.be/3tXu9EcXBRA

Part 2

https://youtu.be/C_0LL9UqB9E

Android Spinners

Part 1

https://youtu.be/H-YDKRdq_q0

Part 2

https://youtu.be/nv0GsjP6rek

Part 3

https://youtu.be/7dtL7tMwLtk

Android Rating Bar

Part 1

https://youtu.be/NHxTt3Ahv1w

Part 2

https://youtu.be/jaaF3R4WgNw

Android Progress Bar

Part 1

https://youtu.be/z1wZve41lRI

Part 2

https://youtu.be/8yUlXEhXYFc

UNIT- V LINKS

Live Mobile Applications Development

Android Games

Part 1

https://youtu.be/uX7-PyxsWr0

Part 2

https://youtu.be/TZpDr9za7Mo

Part 3

https://youtu.be/7P-YgPeY4hw

Android Clock

Part 1

https://youtu.be/xGhxnTtiCDM

Part 2

https://youtu.be/iXIh0mKeMpU

Part 3

https://youtu.be/rJuHt8x5wx4

Android Calendar

Part 1

https://youtu.be/MTs61AUKxn0

Part 2

https://youtu.be/g1Mvz9LiXWo

Part 3

https://youtu.be/nRNucyPjpL0

Android Converter

Part 1

https://youtu.be/QjbeI4tTKr8

Part 2

https://youtu.be/On1uAdKF-Ek

Part 3

https://youtu.be/TRO_KHOCSSk

Android Phone Book

Part 1

https://youtu.be/x8e9zVy4QKU

Part 2

https://youtu.be/wq82gNV6vBI

Part 3

https://youtu.be/0Ck4VxAw73Y

Part 4

https://youtu.be/SY4Lg1kFKa8

App deployment and testing

Part 1

https://youtu.be/-3a15FMiyxs

Part 2

https://youtu.be/vzL1tm32Kr4

Part 3

https://youtu.be/OzozorlfMV4

Part 4

https://youtu.be/uBQXEw2w1hQ

Doodlz App

Part 1

https://youtu.be/4_CTVY2cmk0

Part 2

https://youtu.be/yZ1diEJXnbo

Part 3

https://youtu.be/KmjYc81C17I

Tip Calculator App

Part 1

https://youtu.be/Q_VWz_7BVpA

Part 2

https://youtu.be/WLwEv4oIQ-8

Part 3

https://youtu.be/zTY_MOq3ICc

Weather Viewer App

Part 1

https://youtu.be/LnTVKxU9Oa8

Part 2

https://youtu.be/GQbbpf0IXzA

Part 3

https://youtu.be/04sU26pc9_Q

ANDROID SAMPLE PROGRAMS WITH OUTPUT

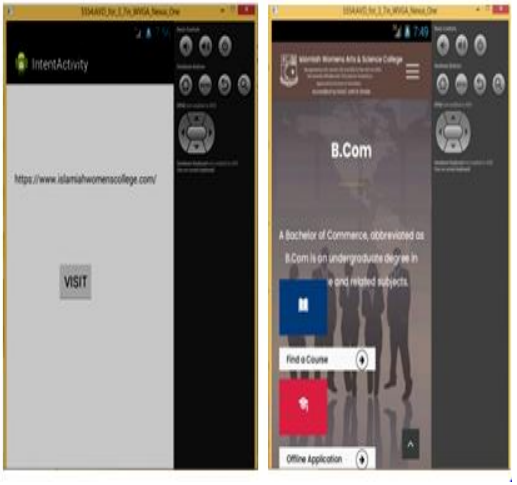
01 - INTENT AND ACTIVITY

CODING:

```
package com.example.intentactivity;
import android.app.Activity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
Button but;
TextView tv;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
but = (Button)findViewById((Integer) R.id.button);
tv =  (TextView)findViewById(R.id.textView);
```

```
but.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View view)
{
String url=tv.getText().toString();
Intent intent=new Intent(Intent.ACTION_VIEW,
Uri.parse(url));
startActivity(intent);
}
});
}
}
```

OUTPUT:



02 – APPLICATION USING CONTROLS

CODING:

```
package text.button;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
TextView txt1,txt2,txt3,txt4,txt5;
Button btn;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
txt1=(TextView)findViewById(R.id.textView6);
txt2=(TextView)findViewById(R.id.textView7);
txt3=(TextView)findViewById(R.id.textView8);
txt4=(TextView)findViewById(R.id.textView9);
txt5=(TextView)findViewById(R.id.textView10);
```

```
btn=(Button)findViewById(R.id.mybutton);
btn.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
txt1.setText("V906785");
txt2.setText("A987867");
txt3.setText("SBI947564");
txt4.setText("9876453423990912");
txt5.setText("9876789876567867");
}
});
}
}
```

OUTPUT:

03 - ALERT DIALOGS

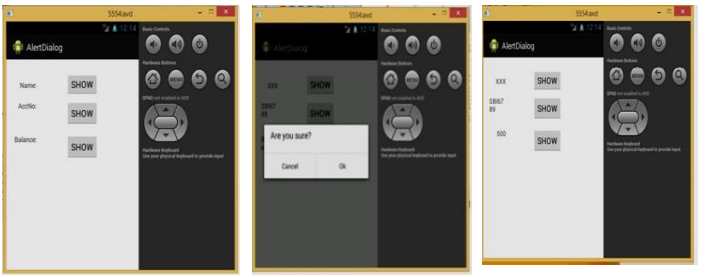
CODING:

```
package alert.dialog;
import android.os.Bundle;
import android.app.Activity;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
TextView txtName,txtAcc,txtBal;
Button btnName,btnAcc,btnBal;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
txtName=(TextView) findViewById(R.id.txtName);
txtAcc=(TextView) findViewById(R.id.txtAcc);
txtBal=(TextView) findViewById(R.id.txtBal);
btnName=(Button) findViewById(R.id.btnName);
btnAcc=(Button) findViewById(R.id.btnAcc);
```

```
btnBal=(Button) findViewById(R.id.btnBal);
btnName.setOnClickListener(new  View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
txtName.setText("XXX");
}
});
btnAcc.setOnClickListener(new  View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
txtAcc.setText("SBI6789");
}
});
btnBal.setOnClickListener(new  View.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
AlertDialog.Builder builder=new
AlertDialog.Builder(MainActivity.this);
builder.setMessage("Are you
```

```
sure?").setPositiveButton("Ok",new
DialogInterface.OnClickListener()
{
@Override
publicvoid onClick(DialogInterface dialog, int which)
{
txtBal.setText("500");
}
})
.setNegativeButton("Cancel",null);
AlertDialog alert=builder.create();
alert.show();
}
});
}
}
```

OUTPUT:



04 – LIST VIEW

CODING:

```
package com.example.listviewprogram;
import android.app.Activity;
import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;
publicclass MainActivity extends Activity
{
ListView list;
staticfinal String[] country = new String[]
{"India", "Pakistan", "London", "Europe", "Japan","China",
"Greece"};
@Override
publicvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
```

list = (ListView) findViewById(R.id.*ListView1*);

ArrayAdapter<Object> adapter = new

ArrayAdapter<Object>(this,android.R.layout.*simple_list_item_1*, *country*);

list.setAdapter(adapter);

list.setOnItemClickListener(new OnItemClickListener()

{

publicvoid onItemClick(AdapterView<?> parent, View v, int position, long id) {

Toast.*makeText*(getApplicationContext(),((TextView) v).getText(), Toast.*LENGTH_SHORT*).show();

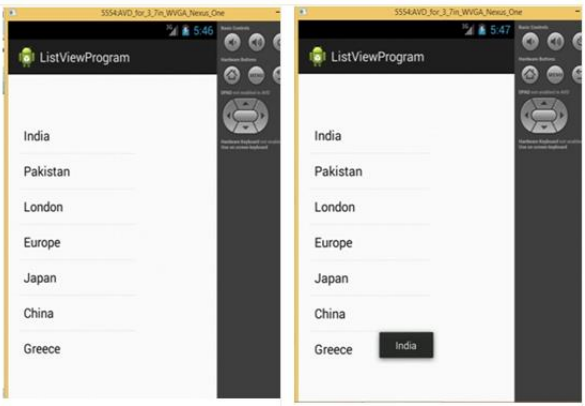
}

});

}

}

OUTPUT:



05 - OPTIONS MENUCODING:

main_menu.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu
xmlns:android="http://schemas.android.com/apk/res/android"
>
<item
android:id="@+id/about_us_id"
android:title="About"
android:showAsAction="always"/>
<item
android:id="@+id/settings_id"
android:title="Settings"
android:showAsAction="always"/>
<item
android:id="@+id/Share_id"
android:title="Share"
android:showAsAction="always"/>
<item
android:id="@+id/Help_id"
android:title="Help"
android:showAsAction="always"/>
</menu>
```

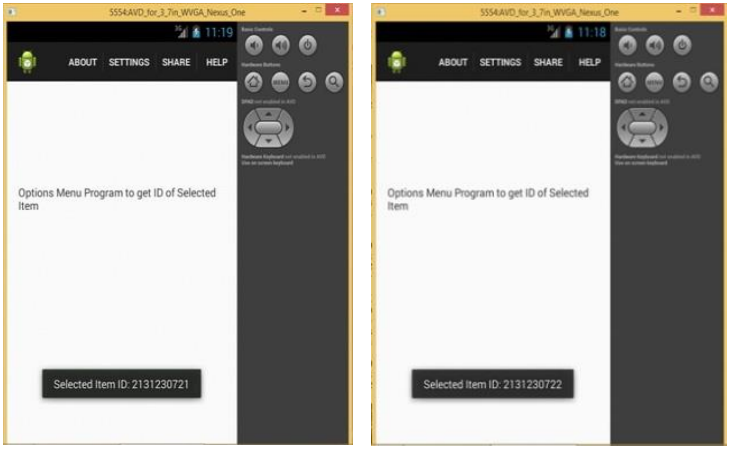
MainActivity.java

```
package com.example.menudemo;
import android.os.Bundle;
```

```
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main_menu, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
Toast.makeText(this, "Selected Item ID: "+item.getItemId(),
0).show();
switch (item.getItemId())
{
case R.id.about_us_id:
```

```
returntrue;
case R.id.settings_id:
returntrue;
case R.id.Share_id:
returntrue;
case R.id.Help_id:
returntrue;
default:
returnsuper.onOptionsItemSelected(item);
}
}
}
```

OUTPUT:



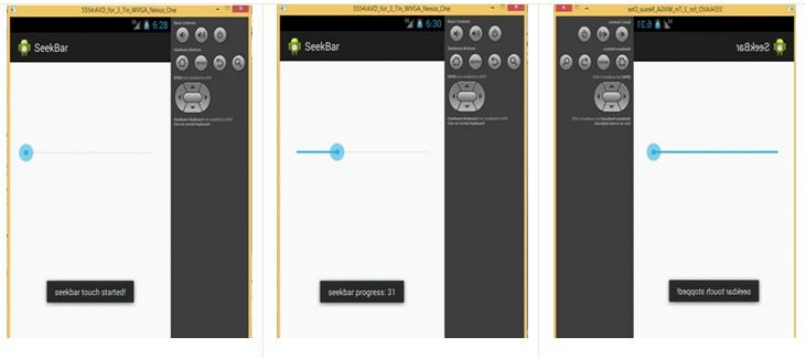
06 – SEEK BARS

CODING:

```
package com.example.seekbar;
import android.app.Activity;
import android.os.Bundle;
import android.widget.SeekBar;
import android.widget.Toast;
publicclass MainActivity extends Activity
{
SeekBar seekBar;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
seekBar=(SeekBar)findViewById(R.id.seekBar1);
seekBar.setOnSeekBarChangeListener(new
SeekBar.OnSeekBarChangeListener()
{
@Override
publicvoid onProgressChanged(SeekBar seekBar, int
progress,boolean fromUser)
{
Toast.makeText(getApplicationContext(),"seekbar progress:
"+progress,
```

```
Toast.LENGTH_SHORT).show();
}
@Override
publicvoid onStartTrackingTouch(SeekBar seekBar)
{
Toast.makeText(getApplicationContext(),"seekbar touch
started!", Toast.LENGTH_SHORT).show();
}
@Override
publicvoid onStopTrackingTouch(SeekBar seekBar)
{
Toast.makeText(getApplicationContext(),"seekbar touch
stopped!", Toast.LENGTH_SHORT).show();
}
});
}
}
```

OUTPUT:



07 - SHARED PREFERENCES

CODING:

MainActivity.java

```
package com.example.sharedpreferencepgm;
import android.os.Bundle;
import android.app.Activity;
import android.content.SharedPreferences;
import android.widget.EditText;
public class MainActivity extends Activity
{
EditText et1,et2;
@Override
protected void onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
et1=(EditText)findViewById(R.id.editText1);
et2=(EditText)findViewById(R.id.editText2);
SharedPreferences
settings=getSharedPreferences("Asmahusna",0);
et1.setText(settings.getString("name",""));
et2.setText(settings.getString("email",""));
}
```

```
@Override
protected void onStop()
{
super.onStop();
SharedPreferences
settings=getSharedPreferences("Asmahusna",0);
SharedPreferences.Editor editor=settings.edit();
editor.putString("name",et1.getText().toString());
editor.putString("email",et2.getText().toString());
editor.commit();
}
}
```

activitymain.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context=".MainActivity">
```

```
<RelativeLayout
android:id="@+id/relativeLayout1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentRight="true"
android:layout_centerVertical="true"
android:layout_marginRight="111dp">
</RelativeLayout>

<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentTop="true"
android:layout_marginLeft="26dp"
android:layout_marginTop="94dp"
android:text="Name"
android:textAppearance="?android:attr/textAppearanceLarge"
/>

<EditText
android:id="@+id/editText1"
```

```
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/textView1"
android:layout_below="@+id/textView1"
android:layout_marginTop="20dp"
android:ems="10"
android:inputType="textPersonName" />
<EditText
android:id="@+id/editText2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/textView2"
android:layout_below="@+id/textView2"
android:layout_marginTop="37dp"
android:ems="10"
android:inputType="textEmailAddress">
<requestFocus />
</EditText>
<TextView
android:id="@+id/textView2"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/editText1"
android:layout_below="@+id/relativeLayout1"
```

android:text="E-Mail Address"

android:textAppearance="?android:attr/textAppearanceLarge"
/>

</RelativeLayout>

OUTPUT:

08 – STATUS BAR NOTIFICATIONS

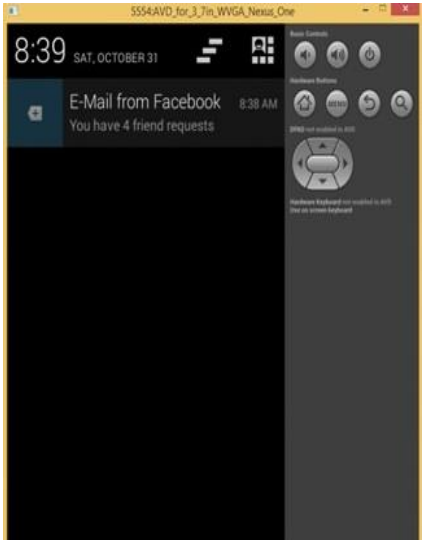
CODING:

```
package com.example.notification;
import android.os.Bundle;
import android.app.Activity;
import android.app.Notification;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
Button bt=(Button)findViewById(R.id.button1);
bt.setOnClickListener(new View.OnClickListener()
{
@Override
```

```
publicvoid onClick(View v)
{
NotificationManager
nm=(NotificationManager)getSystemService(NOTIFICATIO
N_SERVICE);
Notification notification=new
Notification(android.R.drawable.stat_notify_more,"You have
got an email",System.currentTimeMillis());
Context context=MainActivity.this;
Intent intent=new Intent(context,MainActivity.class);
PendingIntent
pending=PendingIntent.getActivity(getApplicationContext(),0
,intent,0);
notification.setLatestEventInfo(context,"E-Mail from
Facebook","You have 4 friend requests",pending);
nm.notify(0,notification);
}
});
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
}
```

OUTPUT:


5554:AVD_for_3_7in_WVGA_Nexus_One
You have got an email
Notification
NOTIFICATION


5554:AVD_for_3_7in_WVGA_Nexus_One
8:39
SAT, OCTOBER 31
E-Mail from Facebook
8:38 AM
You have 4 friend requests

09 - TAB WIDGET TALKING CLOCK

CODING:

MainActivity.java

```
package com.example.talkingclock;

import android.os.Bundle;

import android.app.Activity;

import java.util.Calendar;

import android.speech.tts.TextToSpeech;

import android.speech.tts.TextToSpeech.OnInitListener;

import android.view.View;

import android.widget.AnalogClock;

import android.widget.TextView;

publicclass MainActivity extends Activity implements
OnInitListener

{

TextToSpeech Talktome;

AnalogClock Clock2;

TextView ReadText;

privateint mHour, mMinute;

publicvoid onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity_main);
```

```
Talktome = new TextToSpeech(this, this);
Clock2 = (AnalogClock)findViewById(R.id.AnalogClock);
Clock2.setOnClickListener(MyAnalogClockOnClickListener)
;
ReadText = (TextView)findViewById(R.id.Text2Read);
}
publicvoid onInit(int status)
{
// TODO Auto-generated method stub
}
@Override
protectedvoid onDestroy()
{
// TODO Auto-generated method stub
super.onDestroy();
Talktome.shutdown();
}
private AnalogClock.OnClickListener
MyAnalogClockOnClickListener
= new AnalogClock.OnClickListener()
{
@Override
publicvoid onClick(View v)
{
```

```
final Calendar c = Calendar.getInstance();
mHour = c.get(Calendar.HOUR_OF_DAY);
mMinute = c.get(Calendar.MINUTE);
String myTime= "The Time is  is "
+ String.valueOf(mHour)
+ " Hour "
+String.valueOf(mMinute)
+ " Minute";
ReadText.setText(myTime);
Talktome.speak(myTime, TextToSpeech.QUEUE_FLUSH,
null);
}
};
}
```

OUTPUT:

10 - TWEEN ANIMATION

CODING:

anim.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set
xmlns:android="http://schemas.android.com/apk/res/android"
>
<scale
android:duration="4000"
android:fromXScale=".25"
android:fromYScale=".25"
android:pivotX="50%"
android:pivotY="50%"
android:toXScale="2"
android:toYScale="2"/>
</set>
```

MainActivity.java

```
package com.example.tweenanimation;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.View;
import android.view.View.OnClickListener;
```

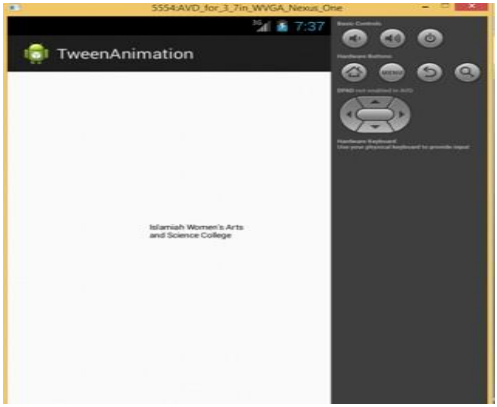
```
import android.view.animation.Animation;
import android.view.animation.AnimationUtils;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
final TextView tv=(TextView)
findViewById(R.id.textView1);
tv.setOnClickListener(new OnClickListener()
{
@Override
publicvoid onClick(View arg0)
{
// TODO Auto-generated method stub
Animation
anim=AnimationUtils.loadAnimation(getApplicationContext(
),R.anim.anim);
tv.startAnimation(anim);
}
});
```

```
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
}
```

OUTPUT:

Original Text Size

When clicked on text, text starts resizing itself from small to big


5554:AVD_for_3_7in_WVGA_Nexus_One
7:37
TweenAnimation
Islamiah Women's Arts
and Science College


5554:AVD_for_3_7in_WVGA_Nexus_One
8:17
TweenAnimation
amiah Women's
d Science Colle

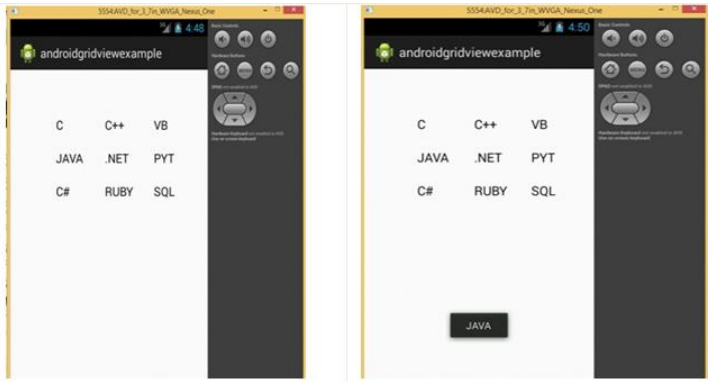
11 – GRID VIEW

CODING:

```
package com.example.androidgridviewexample;
import android.app.Activity;
import android.os.Bundle;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.GridView;
import android.widget.TextView;
import android.widget.Toast;
import android.view.View;
import android.widget.AdapterView.OnItemClickListener;
publicclass MainActivity extends Activity
{
GridView grid;
staticfinal String[] letters = new String[]
{ "C", "C++", "VB", "JAVA", ".NET","PYT", "C#",
"RUBY", "SQL"};
@Override
publicvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
```

```
grid = (GridView) findViewById(R.id.gridView);

ArrayAdapter<Object> adapter = new
ArrayAdapter<Object>(this,
android.R.layout.simple_list_item_1, letters);

grid.setAdapter(adapter);

grid.setOnItemClickListener(new OnItemClickListener()

{

publicvoid onItemClick(AdapterView<?> parent, View v, int
position, long id)

{

Toast.makeText(getApplicationContext(),((TextView)
v).getText(), Toast.LENGTH_SHORT).show();

}

});

}

}
```

OUTPUT:



12 – INTERNAL STORAGE - FILES

CODING:

MainActivity.java

```
package com.example.programtwelvefinal;
import android.app.Activity;
import android.content.Context;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
publicclass MainActivity extends Activity
{
EditText editText;
Button save,display;
TextView textView;
String filename = "myfile";
@Override
```

```
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
editText=(EditText)findViewById(R.id.input);
save=(Button)findViewById(R.id.b1);
display=(Button)findViewById(R.id.b2);
textView=(TextView)findViewById(R.id.textDisplay);
save.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View view)
{
String msg=editText.getText().toString().trim();
if (!msg.equals(""))
{
try
{
FileOutputStream fileOutputStream =
openFileOutput(filename,   Context.MODE_PRIVATE);
fileOutputStream.write(msg.getBytes());
fileOutputStream.close();
Toast.makeText(getApplicationContext(), "Data saved
successfuly",
```

```
Toast.LENGTH_SHORT).show();
}
catch (Exception e)
{
e.printStackTrace();
}
}
else
{
Toast.makeText(getApplicationContext(), "Data is empty",
Toast.LENGTH_SHORT).show();
}
}
});
display.setOnClickListener(new View.OnClickListener()
{
@Override
publicvoid onClick(View view)
{
try
{
FileInputStream fileInputStream=openFileInput(filename);
InputStreamReader inputStreamReader=new
InputStreamReader(fileInputStream);
```

```
BufferedReader bufferedReader=new
BufferedReader(inputStreamReader);
String msg;
StringBuilder stringBuilder=new StringBuilder();
while ((msg=bufferedReader.readLine()) != null)
{
stringBuilder.append(msg);
}
textView.setText(stringBuilder.toString());
}
catch (Exception e)
{
e.printStackTrace();
}
}
});
}
}
```

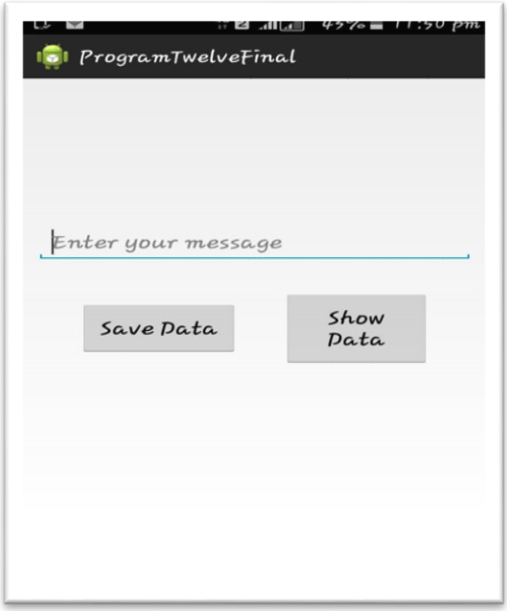
activitymain.xml

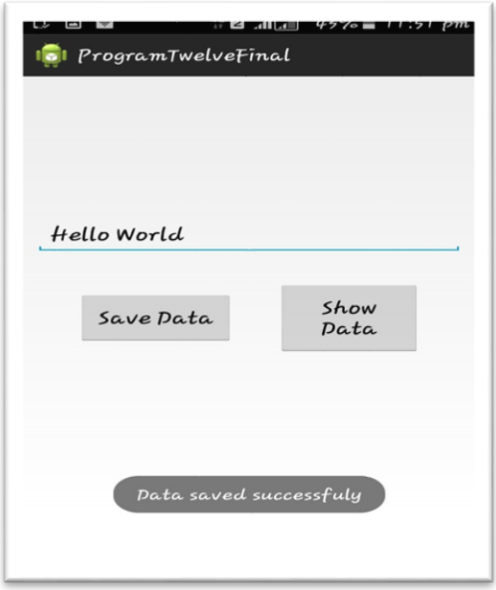
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
```

```
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".InternalStorageExample"
android:padding="16dp"
android:gravity="center"
android:orientation="vertical">

<EditText
android:id="@+id/input"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:hint="Enter your message"/>

<LinearLayout
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:orientation="horizontal"
android:padding="16dp"
android:gravity="center">

<Button
android:id="@+id/b1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:layout_margin="16dp"
android:padding="16dp"
android:text="Save Data"
android:textAllCaps="false" />

<Button
android:id="@+id/b2"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_margin="16dp"
android:padding="16dp"
android:text="Show Data"
android:textAllCaps="false" />

</LinearLayout>

<TextView
android:id="@+id/textDisplay"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:textSize="20dp"
android:padding="16dp"
android:gravity="center"/>
</LinearLayout>
```

OUTPUT:


ProgramTwelveFinal
Enter your message
Save Data
Show Data


ProgramTwelveFinal
Hello World
Save Data
Show Data
Data saved successfuly


43% 11:51 pm
ProgramTwelveFinal
Hello World
Save Data
Show Data
Hello World

13 - SQLite DATABASE

CODING:

activity_main.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.database_demo_sql.MainActivit
y">
<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentTop="true"
android:layout_centerHorizontal="true"
android:layout_marginTop="36dp"
android:text="Storing Data in SQLite Database"
android:textAppearance="?android:attr/textAppearanceLarge"
/>
```

<Button

android:id="@+id/btninsert"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_below="@+id/textView1"

android:layout_centerHorizontal="true"

android:layout_marginTop="70dp"

android:text="PRESS BUTTON TO ENTER DATA IN DATABASE" />

</RelativeLayout>

MainActivity.java

```
package com.example.database_demo_sql;

import android.support.v7.app.ActionBarActivity;

import android.content.Intent;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.Button;

publicclass MainActivity extends ActionBarActivity

implements OnClickListener

{
```

```
Button btninsert,btnview,btndelete,btnedit;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
btninsert=(Button)findViewById(R.id.btninsert);
btninsert.setOnClickListener(this);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
int id = item.getItemId();
if (id == R.id.action_settings)
{
returntrue;
}
returnsuper.onOptionsItemSelected(item);
```

```
}
@Override
publicvoid onClick(View arg0)
{
// TODO Auto-generated method stub
if(arg0.getId()==R.id.btninsert)
{
Intent i=new Intent(MainActivity.this,InsertActivity.class);
startActivity(i);
}
}
}
```

activity_insert.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.database_demo_sql.InsertActivit
y">
```

```
<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentTop="true"
android:layout_marginLeft="101dp"
android:layout_marginTop="51dp"
android:text="Insert Data" />
<EditText
android:id="@+id/insertuser"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignRight="@+id/textView1"
android:layout_below="@+id/textView1"
android:layout_marginTop="42dp"
android:hint="Username"
android:ems="10">
<requestFocus />
</EditText>
<EditText
android:id="@+id/insertpwd"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
```

```
android:layout_alignLeft="@+id/insertuser"
android:layout_below="@+id/insertuser"
android:layout_marginTop="33dp"
android:ems="10"
android:hint="Password"
android:inputType="textPassword" />
<EditText
android:id="@+id/insertcity"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignLeft="@+id/insertpwd"
android:layout_below="@+id/insertpwd"
android:layout_marginTop="44dp"
android:hint="City"
android:ems="10" />
<Button
android:id="@+id/insbtn"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignRight="@+id/insertuser"
android:layout_below="@+id/insertcity"
android:layout_marginTop="34dp"
android:text="INSERT" />
```

</RelativeLayout>

InsertActivity.java

```
package com.example.database_demo_sql;
import android.support.v7.app.ActionBarActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
publicclass InsertActivity extends ActionBarActivity
implements OnClickListener
{
EditText edtuser,edtpwd,edtcity;
Button btn;
DBHelper dbh;
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_insert);
edtuser=(EditText)findViewById(R.id.insertuser);
```

```
edtpwd=(EditText)findViewById(R.id.insertpwd);
edtcity=(EditText)findViewById(R.id.insertcity);
btn=(Button)findViewById(R.id.insbtn);
btn.setOnClickListener(this);
dbh=new DBHelper(InsertActivity.this);
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
// Inflate the menu; this adds items to the action bar if it is present.
getMenuInflater().inflate(R.menu.insert, menu);
returntrue;
}
@Override
publicboolean onOptionsItemSelected(MenuItem item)
{
int id = item.getItemId();
if (id == R.id.action_settings)
{
returntrue;
}
returnsuper.onOptionsItemSelected(item);
}
```

```
@Override
publicvoid onClick(View arg0)
{
// TODO Auto-generated method stub
String user=edtuser.getText().toString();
String pwd=edtpwd.getText().toString();
String city=edtcity.getText().toString();
Student s=new Student(user,pwd,city);
dbh.InsertData(s);
Toast.makeText(InsertActivity.this,"Record
Inserted",500).show();
finish();
}
}
```

Student.java

```
package com.example.database_demo_sql;
publicclass Student
{
int _id;
String user;
String pwd;
String city;
public Student()
{
```

```
super();
}
public Student(int _id, String user, String pwd, String city)
{
super();
this._id = _id;
this.user = user;
this.pwd = pwd;
this.city = city;
}
public Student(String user, String pwd, String city)
{
super();
this.user = user;
this.pwd = pwd;
this.city = city;
}
public Student(String user, String city)
{
super();
this.user = user;
this.city = city;
}
@Override
```

```
public String toString()
{
// TODO Auto-generated method stub
return _id+" "+user+" "+pwd+" "+city;
}
public String getUser()
{
// TODO Auto-generated method stub
returnnull;
}
public String getPwd()
{
// TODO Auto-generated method stub
returnnull;
}
public String getCity()
{
// TODO Auto-generated method stub
returnnull;
}
public Integer get_id()
{
// TODO Auto-generated method stub
returnnull;
```

}

}

OUTPUT:


Database_DEmo_sql
Storing Data in SQLite Database
PRESS BUTTON TO ENTER DATA IN DATABASE


InsertActivity
Insert Data
Username
Password
City
INSERT


InsertActivity
Insert Data
Zaara
.....
Vellore
INSERT


Database_DEmo_sql
Storing Data in SQLite Database
PRESS BUTTON TO ENTER DATA IN DATABASE
Record Inserted

14 - GOOGLE MAP

PROCEDURE:

Step 1: Start Android Developer Tools

Step 2: Select File->New->Android Application Project

Step 3: New Android Application dialog box will appear. Enter Application

Name and Click Next

Step 4: Configure project if needed and Click Next

Step 5: Configure the attributes of the icon set if needed and Click Next

Step 6: Select Blank Activity and Click Next

Step 7: Activity Name and Layout Name will be displayed Click Finish.

Step 8: Android app design area will be displayed. Place necessary controlsin the design area.

Step 9: Open MainActivity.java file and open activitymail.xml, write the respective codings given for each file.

Step 10: Save the application.

To deploy an application in your android mobile follow the steps:

Step 11: Open Google Chrome and install Samsung USB Driver on yourlaptop by using the following link

https://developer.samsung.com/mobile/android-usb-driver.html

Step 12: In your Android Mobile go to **Settings** and click on **About**

Device, Build number will be displayed, click on it 7 times to enable the **Developer Options.**

Step 13: Once Developer Options is enabled, click on it and options like

On, Stay Awake, USB Debugging and Allow Mock Locations

options should be enabled.

Step 14: In your Android Mobile go to **Lock Screen and Security** and enable **Unknown sources** option.

Step 15: Open Eclipse, select the file name which you want to deploy and click on Run Button.

CODING:

MainActivity.java

```
package com.example.googlemapapp;

import android.app.Activity;

import android.content.Intent;

import android.net.Uri;

import android.os.Bundle;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

publicclass MainActivity extends Activity

{
```

```
@Override
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
}
publicvoid process(View view)
{
Intent intent=null,chooser=null;
if(view.getId()==R.id.LaunchMap)
{
intent=new Intent(android.content.Intent.ACTION_VIEW);
intent.setData(Uri.parse("geo: 12.685841,78.6089975"));
chooser=Intent.createChooser(intent,"Launch Maps");
startActivity(chooser);
}
}
@Override
publicboolean onCreateOptionsMenu(Menu menu)
{
getMenuInflater().inflate(R.menu.main, menu);
returntrue;
}
@Override
```

```
publicboolean onOptionsItemSelected(MenuItem item)
{
int id = item.getItemId();
if (id == R.id.action_settings)
{
returntrue;
}
returnsuper.onOptionsItemSelected(item);
}
}
```

activitymain.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.gmfs.MainActivity">
```

<Button

android:id="@+id/LaunchMap"

android:layout_width="wrap_content"

android:layout_height="wrap_content"

android:layout_alignParentTop="true"

android:layout_centerHorizontal="true"

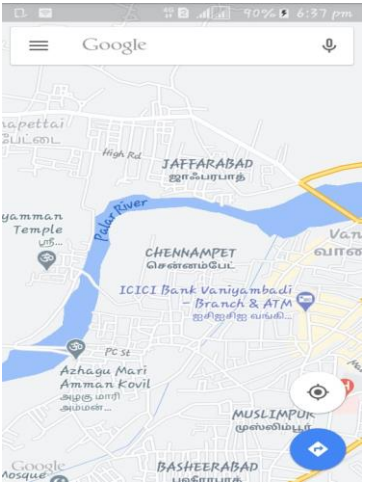
android:layout_marginTop="142dp"

android:onClick="process"

android:text="Launch Map" />

</RelativeLayout>

OUTPUT:



15 - PERMISSIONS (READING PHONE STATE)

CODING:

MainActivity.java

```
package com.example.samplenewpgm;
import android.os.Bundle;
import android.app.Activity;
import android.content.Context;
import android.telephony.TelephonyManager;
import android.widget.TextView;
publicclass MainActivity extends Activity
{
TextView textView1;
@Override
```

```
protectedvoid onCreate(Bundle savedInstanceState)
{
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);
textView1=(TextView)findViewById(R.id.textView1);
//Get the instance of TelephonyManager
TelephonyManager
tm=(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
//Calling the methods of TelephonyManager the returns the information
String IMEINumber=tm.getDeviceId();
String subscriberID=tm.getDeviceId();
String SIMSerialNumber=tm.getSimSerialNumber();
String networkCountryISO=tm.getNetworkCountryIso();
String SIMCountryISO=tm.getSimCountryIso();
String softwareVersion=tm.getDeviceSoftwareVersion();
String voiceMailNumber=tm.getVoiceMailNumber();
//Get the phone type
String strphoneType="";
int phoneType=tm.getPhoneType();
switch (phoneType)
{
case (TelephonyManager.PHONE_TYPE_CDMA):
strphoneType="CDMA";
```

```
break;
case (TelephonyManager.PHONE_TYPE_GSM):
strphoneType="GSM";
break;
case (TelephonyManager.PHONE_TYPE_NONE):
strphoneType="NONE";
break;
}
//getting information if phone is in roaming
boolean isRoaming=tm.isNetworkRoaming();
String info="Phone Details:\n";
info+="\n IMEI Number:"+IMEINumber;
info+="\n SubscriberID:"+subscriberID;
info+="\n Sim Serial Number:"+SIMSerialNumber;
info+="\n Network Country ISO:"+networkCountryISO;
info+="\n SIM Country ISO:"+SIMCountryISO;
info+="\n Software Version:"+softwareVersion;
info+="\n Voice Mail Number:"+voiceMailNumber;
info+="\n Phone Network Type:"+strphoneType;
info+="\n In Roaming? :"+isRoaming;
textView1.setText(info);//displaying the information in the
textView
}
}
```

activity_main.xml

```
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:paddingBottom="@dimen/activity_vertical_margin"
android:paddingLeft="@dimen/activity_horizontal_margin"
android:paddingRight="@dimen/activity_horizontal_margin"
android:paddingTop="@dimen/activity_vertical_margin"
tools:context="com.example.samplenewpgm.MainActivity" >
<TextView
android:id="@+id/textView1"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_alignParentLeft="true"
android:layout_alignParentTop="true"
android:layout_marginLeft="38dp"
android:layout_marginTop="30dp"
android:text="Phone Details:" />
</RelativeLayout>
```

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

<manifest
xmlns:android="http://schemas.android.com/apk/res/android"

package="com.example.samplenewpgm"

android:versionCode="1"

android:versionName="1.0">

<uses-permission
android:name="android.permission.READ_PHONE_STATE"/>

<uses-sdk

android:minSdkVersion="8"

android:targetSdkVersion="18" />

<application

android:allowBackup="true"

android:icon="@drawable/ic_launcher"

android:label="@string/app_name"

android:theme="@style/AppTheme">

<activity

android:name=".MainActivity"

android:label="@string/app_name">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

OUTPUT:

***