1		MAD Architecture
2		Creating Android Application
3		Interactivity Tools.
4		Interaction with Database
5		Web Services & Web View

① MAD Architecture

Date : 4/03/024

Mobile Application technologies

Talking about mobile applications, the first thing that comes to mind are the apps like Whatsapp, Instagram, Swiggy, etc that we use in our everyday life.

Mobile apps are majorly developed for 3 operating System.

① Android
② IOS
③ Windows

There are three different ways to develop mobile apps :-

① Native app development.
② Progressive web application.
③ Cross-platform application.

- Native app development :-

⇒ Native apps runs only in the OS that it is specifically designed for it. These apps cannot be used on different devices using a different OS.

⇒ Android apps are coded using Java or Kotlin language and in Android Studio IDE.

⇒ IOS apps are coded using Swift language and in XCode IDE.

Advantages :-

① The performances of these apps are very high, these apps are very fast compared to any other apps.

② We All the features and APIs are easily accessible.

~~③~~ Disadvantages :-

① The development speed is too slow as we have to code it again for different OS.

~~② Native apps~~

- Progressive web Application :-

⇒ Progressive web apps are essentially a website which runs locally on our device.

⇒ The technologies used are React, Angular JS, Iconic etc.

⇒ These technologies normally used for web development purpose.

~~③ When~~

⇒ A website can be called as PWA, if it has following features :-

* If the website is functional in offline mode.
* If it can alert users by sending push notifications.
* Can be installed on the users home screen.
* Has a responsive design that adapts to different screen sizes.

Advantages :-

① The development speed is fast because it uses same code base.

② The web dev team can be asked to develop the moblile apps.

Disadvantages :-

① PWAs don't have access to all the features and so the user experience is not good.

② IOS doesn't support all the features of PWA.

- **Cross-Platform application :-**

⇒ These are frameworks that allows developing applications which have access to native features of IOS and android with the same codebase.

⇒ Technologies used are Flutter (Google) which uses language Dart, React Native (Facebook) which uses language JavaScript, Xamarin (Microsoft) which uses language .NET, C#.

Advantages :-

① Development Speed is very high as they use the same code base for ~~both~~ Android and IOS's.

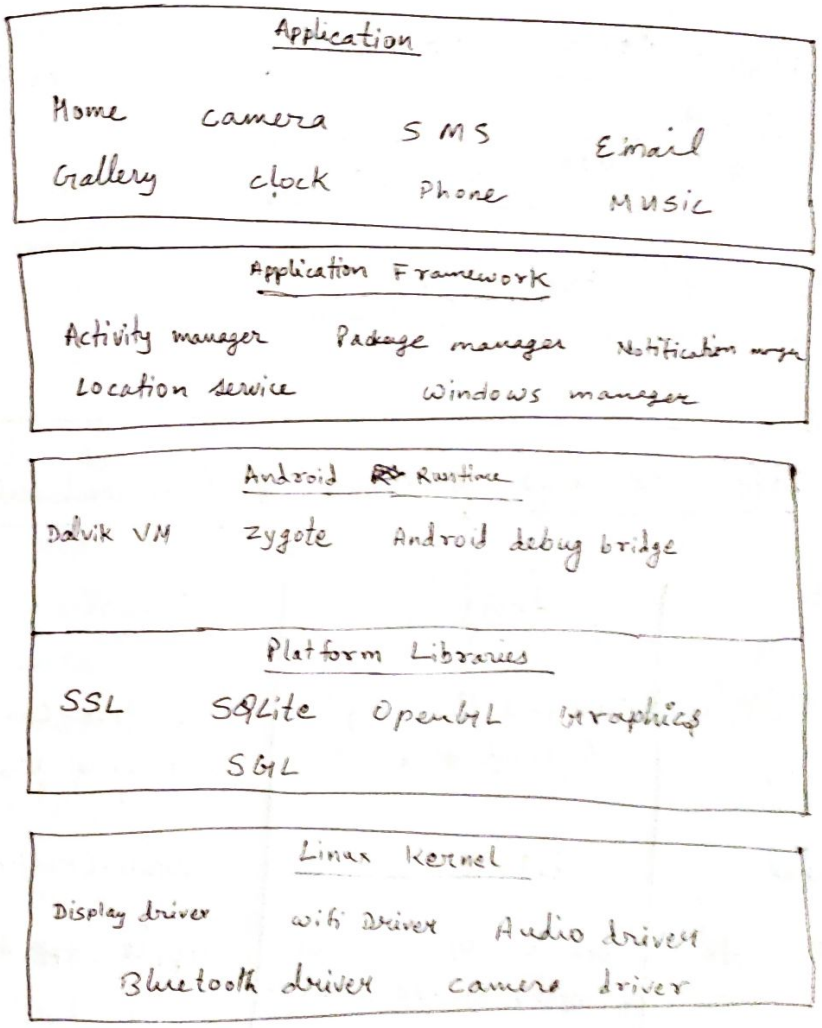
② The maintenance cost is very low, because the errors and updates are countered only once.

Disadvantages :-

① These apps does not have access to Native only features.

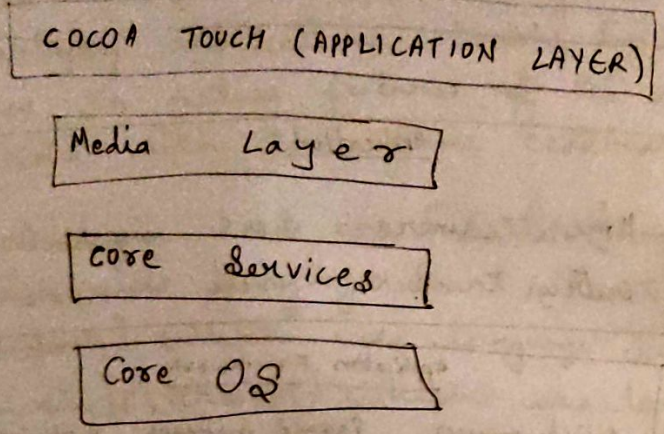
Date : 5/03/024

Mobile app development architecture



Android architecture

IOS Architecture



Difference b/w Android & IOS architecture

Aspect	Android	IOS
① App development	Java or kotlin using Android studio	Objective-C or Swift using Xcode
② Kernel	Linux	Darwin (modified UNIX)
③ App distribtn	Google play store or Third-party app stores	Apple App store
④ UI	Material Design guidelines	Human Interface Guidelines
⑤ customization	more options	Less customization
⑥ Security	more security vulnerability.	more secure due to closed ecosystem
⑦ updates	often delayed or dependent on manufacturers and carriers.	updates are available to all devices simultaneously.
⑧ Integration with ecosystem	Integrates well with Google services	Integrates easily with other apple devices
⑨ Hard ware fragmentation	wide range of devices from various manufacturers.	limited to apple devices.

IOS Advantages

→ more secure than other OS.
→ excellent UI and fluid responsive.
→ suits best for professionals and business.
→ generate less heat than Android

Disadvantages

→ very costly
→ less user friendly as compared to Android
→ not flexible as it supports IOS devices.
→ battery performance is poor.

Hybrid Architecture

Hybrid Apps.

⇒ It is a software application that combines elements of both Native apps and web apps.

⇒ Hybrid applications are written in HTML, CSS, JS ~~(React)~~ and they needs to be installed like Native apps.

⇒ It can run on both online & offline.

⇒ If the hybrid software doesnot depend on the data from dB, then it can be used offline.

⇒ Eg - Ola, Uber, Twitter, etc.

Advantages

⇒ It can work in online as well as offline mode.

⇒ It is cheaper to develop unlike Native apps where it cost double to develop two versions for two platforms.

① Disadvantages

⇒ As Hybrid apps works on various platform, the UI may differ from platform to platform.

⇒ The hybrid apps need to be tested accross various platforms to ensure proper functioning of the app

8/03/024

Chapter - 2

Creating Android Applicant (apps)

Project folder structure :

- v app
  - > manifests
  - v java
    - > com. ~~example~~ example. helloworld
    - > com. example. helloworld (androidTest)
    - > com. example. helloworld (test)
  - v res
    - drawable
    - > layout
    - > mipmap
    - > values
- v Gradle Scripts

  build related folders.

① Manifests ⇒ contains "AndroidManifest.xml" file and contains info about Android version, Access permission, metadata etc. ~~and info~~.

⇒ This file acts as an intermediate b/w android OS and our application.

② Java folder ⇒ contains all java source code (.java file)

and also JUnit test code. Whenever we create any new project, by default the class file MainActivity.java gets created automatically under the package com.example.helloWorld.

③ res folder (Resources)

⇒ contains non-code resources, like images, fonts, XML layouts, sounds, etc.

Drawable folder:

⇒ contains different types of images as reqd by app. other than launcher icons.

Layout folder:

⇒ This folder will contains all XML files which we used to define the user interface of our application.

Mipmap folder:

⇒ This folder will contain app/launcher icons that are used to show on the home screen.

values folder:

⇒ This folder will contain various XML files, such as strings, colors, style definitions and a static array of strings or integers.

④ Gradle Scripts

Gradle means automated build system. by using this folder we can define a build configuration that applies to all modules in the apps.

Default (activity_main.xml)

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.example.helloworld.MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
```

```
app:layout_constraintRight_toRightOf="parent"
app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

Default MainActivity.java

```
package com.example.helloworld;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState)
    {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

Default (AndroidManifest.xml)

An app contains multiple activities and they need to be defined in the AndroidManifest file.

If we didn't mention MAIN action or LAUNCHER category for the main activity, the app icon will not be visible in the home screen's list of apps.

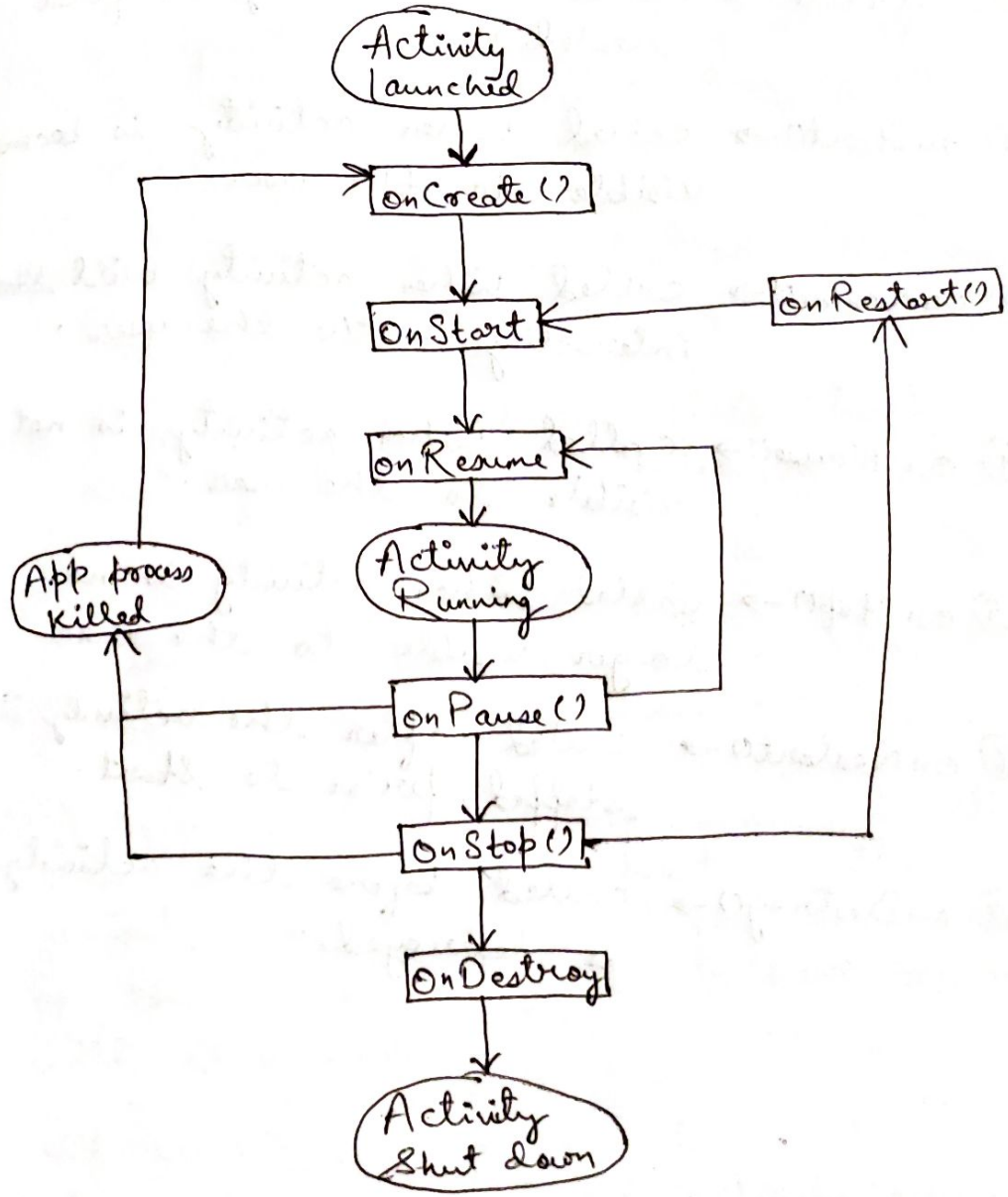
code :

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.helloworld">
  - <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
    <activity android:name=".MainActivity">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.
                LAUNCHER" />
        </intent-filter>
    </activity>
</application>
</manifest>
```

Activity and Activity Life Cycle

10/03/024

⇒ An Activity is the single screen in android.
⇒ It is like window or frame.
⇒ By the help of Activity, we can place all our UI components or widgets in a single screen.
⇒ In C, C++ or Java, main() is the starting point of any program.
Similarly in Android, the system starts its program within an Activity starting with a call on onCreate() callback method.
⇒ There is a sequence of callback methods that runs during the execution of an application. This is known as Lifecycle of Activity.



① onCreate() → called when activity is first created.

② onStart() → called when activity is becomes visible to the user.

③ onResume() → called when activity will start interacting with the user.

④ onPause() → called when activity is not visible to the user.

⑤ onStop() → called when activity is no longer visible to the user.

⑥ onRestart() → called after the activity is stopped, prior to start.

⑦ onDestroy() → called before the activity is destroyed.

Fragment & its lifecycle

⇒ Fragment is the part of the activity which represents the portion of user interface (UI) on the screen.

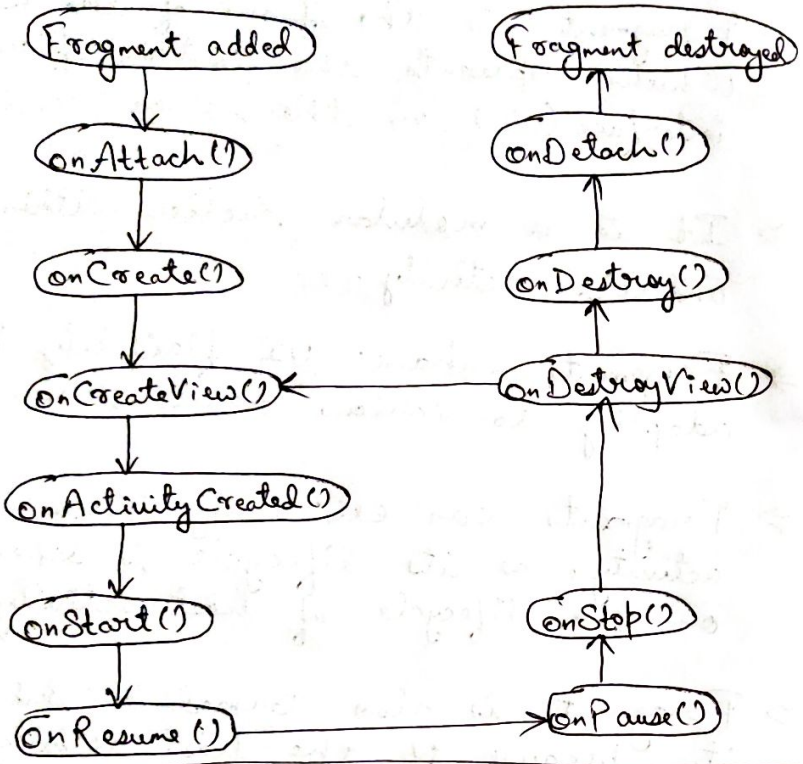
⇒ It is a modular section within an android activity.

⇒ Fragments enhance UI flexibility by adapting to various device screen sizes.

⇒ Fragments can exist only inside an activity, as its lifecycle is dependent on the lifecycle of host activity.

⇒ Fragment is also termed as sub-activity, because if the host activity is paused, then all the methods and operations of the fragment related to that activity will stop functioning.

⇒ <fragment> tag is used to insert a fragment in an android activity layout.



① onAttach() → This method will be called first, even before onCreate(). It attaches the fragment to an activity.

② onCreateView() → The System calls this method when its the time for a fragment to draw its UI for the first time. The method returns null if the fragment does not provide a UI.

③ onActivityCreated() → This will be called after onCreate() and onCreateView(), to indicate that the activities onCreate() has completed.

④ onStart() → This methods is called once the fragments gets visible.

⑤ onResume() → This method is called to make the visible fragment interactive.

⑥ onPause() → The System calls this method as the first indication that the user is leaving the fragment.

⑦ onStop() → Fragment going to be stopped by calling onStop().

⑧ onDestroyView() → It is called before onDestroy(). If there are things that are needed to be cleaned up especially related to UI, then ~~this~~ ~~logic~~ that logic can be put up in the onDestroyView().

⑨ onDestroy() → called to do final cleanup of the fragment's state.

⑩ onDetach() → called to notify that the fragment has been disassociated from its hosting activity.

11/03/024

Views & View Groups

~~Views~~:

View: It is a component which Android provides us to design the layouts of the app. We can visualize a view as a rectangular area which is going to contain some elements ~~on~~ inside it.

⇒ A view is a super class for all the UI components.

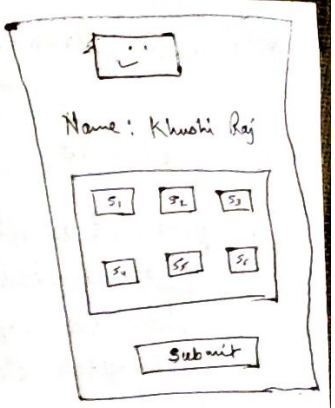
Eg:- TextView ⇒ to add some text in an app.
EditText ⇒ takes input from users.
ImageView ⇒ add image in the app.
ProgressBar ⇒ to show progress of something. Eg- loading screen
Button → triggers ~~some~~ action ~~on a the click on it~~ upon clicked.

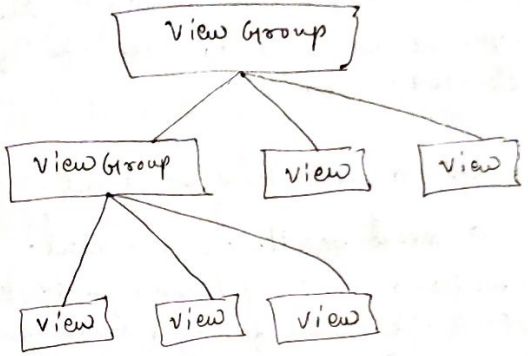
View Group: → A group of views is called a view group.

→ Top level view group is parent and under it all the view and other view groups are its children.

→ Eg- Under a LinearLayout, we can add two buttons and one EditText. Here, Linear Layout is Parent view & the ~~the~~ others remaining are children.

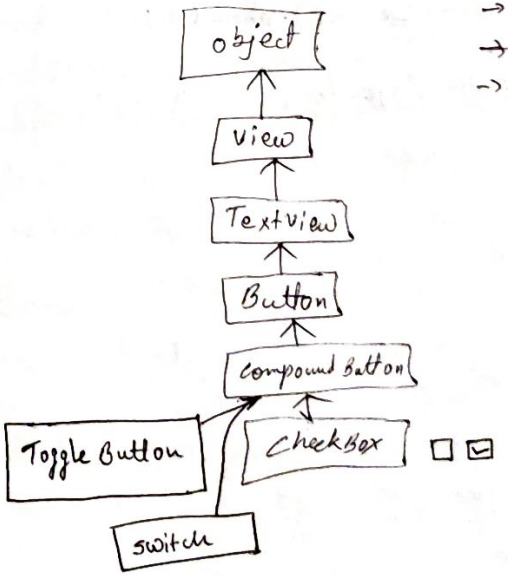
Eg- of View Groups :-
- Linear Layout
- Relative Layout
- Constraint Layout
- Radio Group
- TableV, GridV, ListV, scroll Layout





① Button :

types ⇒ ImageButton, ToggleButton, RadioButton, CheckBox (compound Button),



→ TextButton
→ ~~outlined Button~~
→ ~~Icon~~ IconButton

* RadioButton → RadioGroup. [Aurangabad]

(2) TextFields :- Edit Text,

(3) Spinner (drop down list)

⇒ is associated with AdapterView so to fill the data in spinner we need to use one of the Adapter classes. (देख लेना)

(4) Listview :→ used to show items in a list.

→ no ~~scroll~~ scrollview is needed.

→ list items are inserted using an Adapter that pulls the content from a source such as arraylist, array or database.

(5) Toast ~~[Snack Bar in]~~ :-

It contains a message to be displayed quickly and disappears after some time.

Eg - Toast. makeText(context, "Hello, Toast me", Toast. LENGTH_SHORT). show ( );

(6) Difference b/w Layouts :-

~~Abracadabra is baar 2 number nahi aana chahiye~~

UNIT - 3 } Interactivity Tools

* Intent

→ An intent is a messaging object that is used to request an action from another app component.

or

→ An intent is a mechanism for communication and coordination between different parts of an app, allowing them to work together to achieve the desired functionality.

→ For example, if we want to open a new Screen (activity), when a button is clicked, we can create an intent that specifies which activity to open and then send that intent. The android System will then find the appropriate component to handle that intent and perform the requested action.

→ Some of the basic information that an intents contains are action (action to be performed by particular activity) data, category, component, name, extras (in the form of key value pairs)

Three fundamentals use cases for intents

① Starting an activity :-

- an activity represents a single screen in an app.
- we can start a new instance of an activity by passing an intent to "startActivity()"

② Starting a Service :-

- A Service performs operations in the background without a UI.
- we can start a Service using "startService()" for one time operation (e.g., downloading a file).

③ Delivering a broadcast :-

- A broadcast is a message that any app can receive.
- System events (e.g., device boot, charging) trigger various broadcasts.
- we can use "sendBroadcast()" or "sendOrderedBroadcast()" to deliver a broadcast to other apps.

* Intent types :-

⇒ There are two types of intents :

(i) Explicit Intents :- It is used to communicate between the components of single app.

for e.g.:- if you want to launch an activity, by clicking some button on the present activity then you can specify the fully qualified address of the desired activity to launch that activity.

~~This intent is~~ Due to fully qualified address this intent is applicable to a single application.

Example : Shift from one activity to another activity.

(ii) Implicit Intent :- It is used to communicate with different apps within a device.

for eg.,, → Sharing content with another app.

→ Start camera on button click.

→ open google map when clicked on given location.

* Intent - filters

⇒ It is powerful features that allow components (like activities, services, broadcast receiver) to declare the type of intents they can respond to

⇒ Intent - filter are declared in the Android Manifest.xml file

⇒ An activity or an action can be called by using explicit or implicit intent. But the question that arises here is, how does the android System come to know that a particular activity or action is to be called?
This is done by reading the information that is present in the intent.
The android System reads the information present in the intent and based on that information the android System decides which activity is to be launched.

⇒ Most of the intent-filter are described by

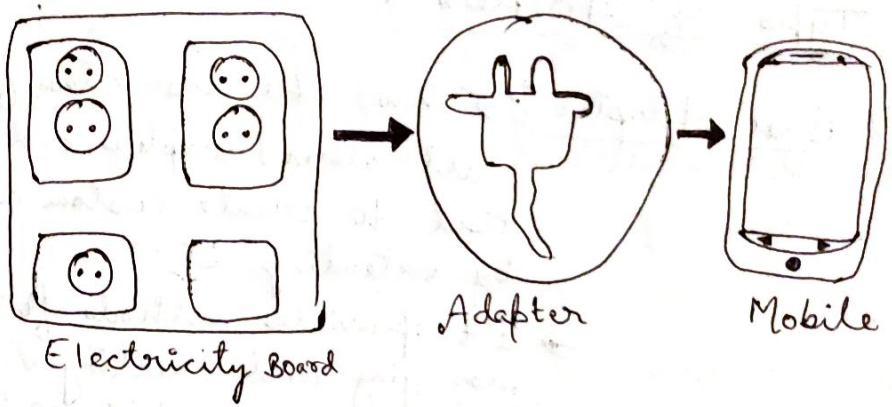
① <action>
② <category>
③ <data>

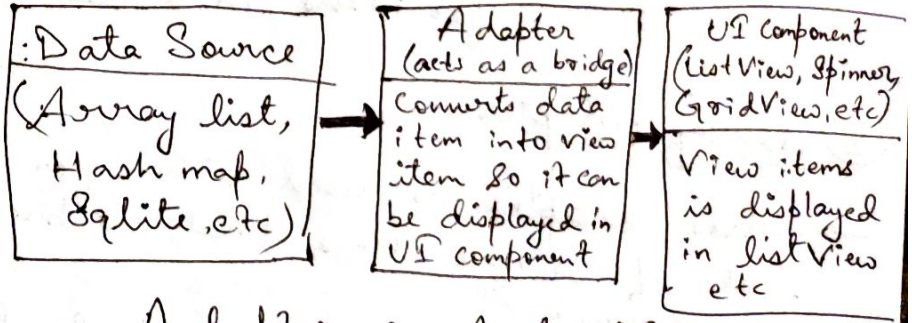
Eg - here

Examples

```
<!-- LAUNCHER INTENT FILTER-->
<intent-filter>
  <action android:name="android.intent.
     action.MAIN" />
  <category android:name="android.intent.
     category.LAUNCHER" />
</intent-filter>
```

* Adapters





Adapter in Android

In android, ~~Add~~ Adapter is a bridge between UI component and data source that helps to fill data in UI component. It holds the data and sends the data to an adapter views then view can take the data from the adapter views and shows the data on different views like ListView, GridView, Spinner, etc.

Types of Adapters :-

① Base Adapter :- → This the base class for all other adapters and is used to create custom adapter by extending it.
→ It provides methods for managing and creating views for items in a List or Grid.

② Array Adapter :-

→ This adapter is used when you have a list of single items, that is backed by an array.
→ It is a simple way to bind data to views in a ListView or Spinner.

③ Custom Array Adapter :-

→ This is used when you need to display a custom list, where each item in the list is complex views that may contain multiple sub-views or require custom logic to bind data to the views.

④ Simple Adapter :-

→ This adapter is used to map static data to views defined in your XML file.
→ It is useful when you have a fixed set of data that you want to display in a ListView or GridView.

⑤ Custom Simple Adapter :-

→ This is used when you need to display a customised list and need to access the child items of the list or grid.
→ It allows you to define custom logic for binding data to views in your list.

Program here → Page no. 80

Dialogs

Alert dialog :-

① It can be used to display the dialog message with OK and Cancel button.

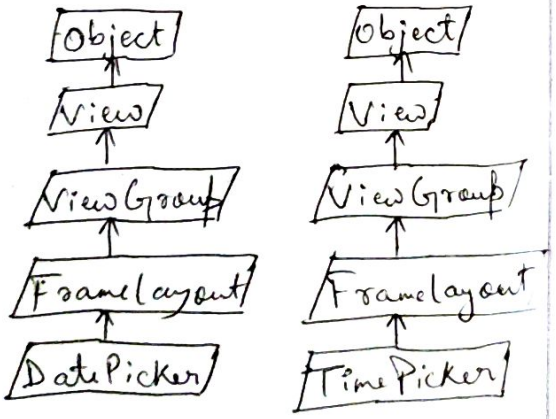
② It can be used to interrupt and ask the user about his/her choice to continue or discontinue.

③ Android alert dialog is composed of three regions :
Title, contentArea and ActionButton

④ ~~Ato~~ Android AlertDialog is the sub-class of Dialog class.

DatePicker :-

TimePicker :-



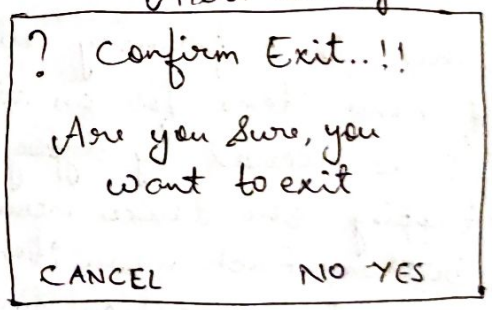
CustomDialog :-

① The customDialog uses DIALOG to create customAlerts in Android Studio. Dialog displays a small window that is ~~top~~-up which draws the user attention over the activity before they continue moving forward.

② The dialog appears over the current window and displays the content defined on it.

AlertDialog Vs CustomAlertDialog

Alert Dialog



Custom Alert Dialog

Both dialogs prompt a small window to make decisions. The alert dialog makes use of the define components or methods like SetIcon, SetTitle, SetMessage, etc. but with CustomAlertDialog we can have the dialog customised and can define the layout of dialog as required.

---

* Menus

Menu helps us provide a user-friendly interface that handles a lot of action.

⇒ Types of menus :-

(i) Option menu :- This is a primary collection of menu items for an activity. It is accessed by tapping the devices menu button. Each menu item represents an action such as Search, Settings, etc.

(ii) Context menu :- This menu is also known as long press menu because it is opened by long pressing on a view element. It provides options like editing, deleting an item in a list within the selected view.

(iii) Popup menu :-

It displays the list of items in a popup window that is anchored to the view. The popup will appear below the view if there is space or above the view in case there is no space.

It will be closed automatically when we touch outside of the popup.

popup menu doesn't support icons.

---

* Notifications

→ A notification is a message that you display to the user outside of the application's normal UI.

→ When we tell the system to issue a notification, it first appears as an icon. To see all details, we need to open notification drawer.

→ Notification drawer can be seen anytime or by scrolling down the notification drawer.

→ Few methods that are related to Notification creation in Android are -

1. setSmallIcon() → set icon of notification
2. setContentTitle() → set title of notification
3. setContentText() → set text message
4. setAutoCancel() → set cancelable property of notification
5. setPriority() → sets priority of notification

2023

Q Create an Android app using SQLite to create a table Student (Sid, Sname, Scourse, Smarks) and update a record of Student.

2022 Q Write an Android application using SQLite to create table employee (eid, ename, edept, esal) and insert a record in table and display appropriate message on toast to user.

2022 Q Write a short note on Firebase database

2022 Q What is cursor? Demonstrate a SQLite Database application to insert a record in table.

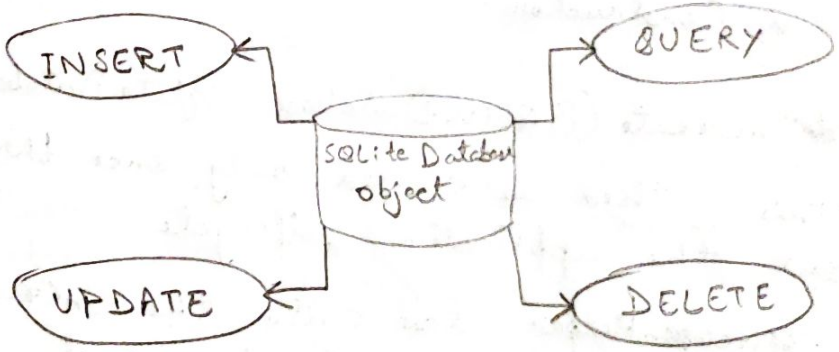
Ch-4 Interaction with Database

4.1 Introduction to Database (SQLite & Firebase)

→ SQLite is an open source relational database that is used to perform database operations on Android devices such as storing, manipulating or retrieving persistent data from the database.

→ It is embedded in Android by default. So, there is no need to perform any database setup or administration tasks.

→ SQLiteOpenHelper class provides the functionality to use the SQLite database.



⇒ To use SQLite Database, android provides an API.

① SQLiteOpenHelper :- (i) It is used to create the database and version management.

(ii) For performing any database operation, you have to provide the implementation of following two methods of SQLiteOpenHelper class :-

→ onCreate().
→ ~~onUpdate~~ onUpgrade()

(iii) The SQLiteOpenHelper is responsible for opening the database if it exists, creating database if it doesn't exist, and upgrading if required.

(iv) SQLiteOpenHelper only requires the ~~def~~ DATABASE-NAME to create a database.

(v) After extending SQLiteOpenHelper we need to implement its methods - OnCreate(), OnUpgrade() and Constructor.

② onCreate (SQLiteDatabase sqLiteDatabase)

(i) This method is called only once throughout the application lifecycle.

(ii) SQLiteOpenHelper class calls the OnCreate() after creating the database and instantiating the SQLite database Object.

(iii) Database name is passed in the constructor call.

③ onUpgrade (SQLiteDatabase db, int oldVersion, int newVersion)

(i) It is called whenever there is an update in the existing version.

(ii) In this method, we can write queries to perform whatever action is required.

(iii) In most examples, we will see that existing tables are dropped and again the OnCreate() is called to create tables again (but it is not mandatory).

② SQLiteDatabase class
→ insert()
→ delete()
→ update()
→ query()
→ execSQL(query)

③ Cursor class (used to retrieve data from db)
→ moveToFirst()
→ moveToNext()

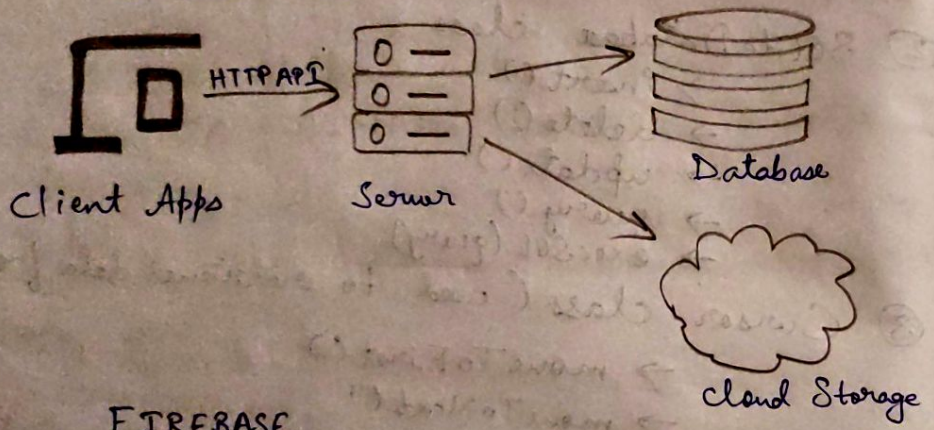
④ User defined Helper class which will extends SQLiteOpenHelper class
= getWritableDatabase()
= getReadableDatabase()

Firebase :-

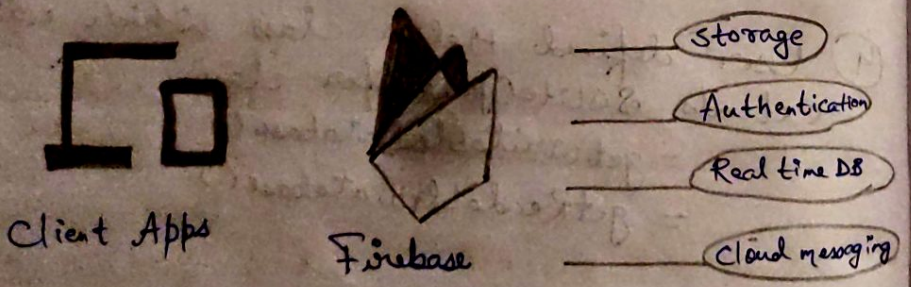
⇒ It is a Backend-as-a-Service, and it is a real-time database which is designed for mobile applications.

⇒ Firebase (a NoSql JSON database) is a real-time database that allows storing a list of objects in the form of a tree. We can synchronize data between different devices.

TRADITIONAL



FIREBASE



⇒ Firebase is supported by Google.

⇒ It has various functionalities as given in the diagram

⇒ It has three main services :-

(i) Real-time database
(ii) User authentication
(iii) Hosting

Cursors :

→ It is an object that allows to retrieve and traverse data from a result set returned by a dB query.

→ They provide methods for moving to the next or previous row, accessing data in the current row, and they are used with **Cursor** class.

→ Eg.

Cursor

→ Cursor is an object which is used to retrieve data from dB.

→ The basic purpose is to point to a single row of the result fetched by "query()" or "rawQuery()"

→ We load the row pointed by the cursor object.

→ By using cursor we can save a lot of RAM & memory.

→ Cursor is the interface which represents a 2D table of any dB.

→ When we try to retrieve some data using SELECT statement, then the dB will first create a cursor object and returns its reference to us.

→ To user cursors ~~Android.~~ android.database.cursor must be imported.

→ Methods provided by cursor class - moveToFirst(), moveToLast(), moveToNext(), close(), etc.

→ Eg -

```
Cursor cursor = db.rawQuery("SELECT * FROM
    your_table WHERE any_condition", null);
while (cursor.moveToNext()){
}   // Do something

cursor.close();
```

ContentValues (Python में dictionary बोलते है)

→ It is a Key-value pair data structure used to insert or update values into a dB.

→ It is widely used with "insert()" and "update" methods of the SQLiteDatabase class to add or update rows in a dB table.

→ Eg - using 'ContentValues' to insert a new row

```
ContentValues values = new ContentValues();
values.put("column_name1", "value1");
values.put("column_name2", "value2");

SQLiteDatabase db = dbHelper.getWritableDatabase();
db.insert("table_name", null, values);
```

Ch-5 Webservices & Webview

Q Explain Webservices with suitable examples.

Ans(i) A webservice is like a mini-program that another programs can access over the internet. They allows systems written in different programming languages and running on different platforms to exchange data and perform various operations.

(ii) Imagine we are building a mobile app to order food. We do not want to build the entire system for finding restaurants, placing orders, and handling payments from scratch, because that would be a lot of work. This is where Web Services comes in

* Breakdown of Web Service :

⇒ Standardized way to communicate :-
Web Services use XML, HTTP, JSON, etc to communicate with each other.

⇒ Perform Specific functionality :-
A web service performs a specific task.
Ex:- a payment processing web service handles secure transactions, ~~whether~~ ~~web~~ weather web service provides weather data.

⇒ Accessible over the Internet :-
These services are available on the internet and allowing programs to access them from anywhere.

* Real life applications of web Services

⇒ Weather Service
⇒ Payment Gateway
⇒ Social Media Integration
⇒ Maps & Locations Services
⇒ Stock Market API, Email API, Messaging API, etc.

* Types of Web Services in Android

(i) XML-RPC :-
It is popularly known as Remote Procedure Calls (RPC). Every call is encoded using XML and HTTP is used for its transmission.

(ii) UDDI :-
UDDI Stands for Universal Descriptive Discovery and Integration. It is a XML based standard that is used to describe, publish and discover new web Services.

(iii) SOAP :-

It stands for Simple Object Access Protocol and is a XML based web service protocol for exchanging data or documents over HTTP or SMTP. It allows separate processes on ~~diffort~~ different platforms to communicate with one another.

(iv) REST :-

REST Stands for REpresentational State Transfer. It is an architectural pattern that allows multiple web service-based systems to interact and communicate efficiently.
RESTful systems are distinguished by their statelessness and separation of client and server concerns.

* Advantages of Android Web Services

* Various application communication.
* Reusability.
* Availability.
* They communicate across various apps using high quality industry standard protocol.
* They employ SOAP ~~over~~ over HTTP to enable web services via low-cost internet connection.

* Limitations of Android Web Services

→ They use the HTTP protocol which is unreliable and unsafe.

→ Network dependency :-
Web services required internet connection to function, so applications that use web services ~~does~~ may not work properly in offline mode.

SOAP	REST
1. This is a function-based protocol.	1. This is Database based protocol.
2. It only uses XML.	2. This permit HTTP, plain text, XML, JSON.
3. It can't be cached.	3. It can be cached.
4. It has a strict communication.	4. It has an easy communication.
5. Build-in ACID Compliance	5. Lack of ACID Compliance
6. Used in banking applications where security is prior.	6. It is advanced and Simple.

Parsing JSON and XML

JavaScript Object Notation (JSON) :-

(i) It is not a programming language, it is a data interchanged format

(ii) It is mostly used to get and post data in web services.

(iii) It is the mostly used format in today's application development.
- → It is a file in which data is written in text format.
- → It is ~~read~~ easy and simple to understand.

Example of JSON format :-

```
{
  "name" : "Rahul";
  "college" : "St. Xavier's College";
  "yoj" : 2020;
  "projects" : ["p1", "p2", "p3"];
}
```

JSON Parser

(i) The process of extracting necessary information from JSON file is called JSON parsing.

(ii) When we parse a JSON file, we are essentially converting the JSON data into a format that can be easily used by our program.

(iii) Android provides four types of classes to manipulate JSON data. (JSON PARSERS)
- → JSONArray
- → JSONObject
- → JSONTokenizer
- → JSONStringer

Q Write a short note on JSON parsing with example.

Ans JSON file

```
{
  "sys":
  {
    "country": "IND",
    "sunrise": 1381107633,
    "sunrise": 1381149604,
  },
  "weather":
    {"temp": 304.15,
     "pressure": 1009,
}}
```

PTO

For parsing a ~~JSON~~ JSON object, we will create an object of class JSONObject and specify a String containing JSON data to it. Its syntax is —

```
String jsonString;
JSONObject reader = new JSONObject(jsonString);
```

The last step is to parse the JSON. A JSON file consist of different objects with different key-value pair etc. So JSONObject has a seperate function for paring each of the component of JSON file. Its syntax is as follows :-

```
JSONObject sys = reader.getJSONObject("sys");
country = sys.getString("country");

JSONObject weather = reader.getJSONObject("weather");
temp = weather.getString("temp");
```

- getJSONObject → returns JSONObject
- getString → returns String value of given key.

Some other methods provided by JSONObject class for better parsing of JSON files—

→ getBoolean("Key")
→ getInt("Key")
→ length() :- returns no. of value mappings in an object.

Q What is Android XML parsing explain with example?

Ans In Android, XML parsing is the process of extracting data from an XML file. Android provides three types of XML parsers which are DOM, SAX and XMLPullParser. Among all of them android recommend XML PullParser because it is efficient and easy to use.

```
<?xml version = "1.0"?>
<current>
<city id="2643743" name="London">
 <coord lon="-0.12574" lat="51.50853"/>
 <country> IND </country>
 <Sun rise="2013-10-08T06:13:56" set="2013-10-
       08T17:21:45"/>
</city>

<temperature value="289.54" min="289.15"
    max="290.15" unit="Kelvin"/>
<humidity value="77" unit="%"/>
<pressure value="1025" unit="hPa"/>
</current>
```

we will create XMLPullParser object, but in order to create that we will first create XMLPullParserFactory object and then call its newPullParser() method to create XMLPullParser. Its Syntax is given below -

```
private XMLPullParserFactory xmlFactoryObject =
XmlPullParserFactory.newInstance();
private XmlPullParser myparser = xmlFactoryObject.
                                 newPullParser();
```

Next step is specifying the file for XMLPullParser that contains XML. Its Syntax is given below -

```
myparser.setInput(stream, null);
```

The last step is to parse the XML. An XML file consist of events, name, text, etc. So, XMLPullParser has a separate function for each of the component of XML file. Its Syntax is given below -

```
int event = myParser.getEventType();
while (event != XmlPullParser.END_DOCUMENT)
{
  String name = myParser.getName();
  switch (event){
    case XmlPullParser.START_TAG:
    break;

    case XmlPullParser.END_TAG:
  if (name.equals("temperature")){
    temperature = myParser.getAttributeValue
(null, "value");
  }
  break;
  }
  event = myParser.next();
}
```

Intro to Webview

Q: Demonstrate a webview to display the web page in an Android app.

Ans: To display a webpage in an Android app using webview, following steps must be followed

1) Add Internet permission in AndroidManifest.xml

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

2) Add a WebView element in layout file -

```
<WebView
    android:id="@+id/webView"
    android:layout_width="match_parent"
    android:layout_height="match_parent" />
```

3) Load the ~~webView in~~ webpage in the MainActivity.java file

```
public class MainActivity extends AppCompatActivity {
    private WebView webView;
    @Override
    protected void onCreate(Bundle savedInstanceState){
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        webView = findViewById(R.id.webView);
        webView.getSettings().setJavaScriptEnabled(true);
        webView.loadUrl("https://www.google.com");
    }
}
```

INTRO

In Android dev, a webView is a component that allows us to display web content within the app. It is essentially a mini browser embedded within the app. This can be useful for various purposes, such as displaying webpages, loading html content and even rendering web based applications.