1

What is an Operating System?

An **Operating** **System** or **OS** is a software program that enables the Hardware to communicate and operate with the Software.

Operating System is a interface between User and hardware.

Operating System provides a software platform on top of which other programs, called application programs, can run.

What is Open Source?

1. The source code of a computer program is made available free of charge to the general public.

2. Users can view the code that comprises the software and make any kind of changes to it they want.

3. Anyone can take the source code and distribute their own program from it.

2

**History of Android.**

**Android Inc? (Company)**

* Initially, Andy Rubin founded Android Incorporation in Palo Alto, California, United States in October, 2003.
* In 17th August 2005, Google (Lawee Manager) acquired android Incorporation. Since then, it is in the subsidiary of Google Incorporation.

**Android Inc. to Google Android**

* The Idea of Making a Open Source Smartphone Operating System that would rival Symbian, Apple iOS and Microsoft Windows Mobile was the key reason Google got attracted to.
* Back in 2005, the company quietly (and cheaply) made what we think was its best acquisition to date. It purchased the startup company Android for $50 Million.

**OHA (Open Handset Alliance) (2005-2008)**

* The Open Handset Alliance (OHA) is a consortium of 84 firms to develop open standards for mobile devices.

3

* The OHA was established on 5 November 2007, led by Google with 34 members, including mobile handset makers, application developers, some mobile carriers and chip makers.

**Key Dates**

* October 2003 – Android Inc. was founded.
* July 2005 – Google acquired Android Inc.
* November 5th, 2007 – OHA (Open Handset Alliance) was formed.
* October 22nd, 2008 – First android Smartphone HTC Dream was launched in US, 2009 – i-series Galaxy first was launched in India.
* September 8th, 2020 – Android OS’s Latest version Android 11 released.

4

Android Version/Flavors?

* The first commercial version, Android 1.0 was released on September 23, 2008.
* Android code names are confectionery (Sweets & Chocolates)-themed and have been in alphabetical order since 2009’s Android 1.5 cupcake.

C - Cupcake J – Jelly Bean

D - Donut K - Kitkat

E - Eclair L - Lollipop

F - Froyo M – Marshmallow

G - Gingerbread N - Nougat

H - Honeycomb O - Orea

I - Ice Cream Sandwich P - Pie

Cupcake (1.5) (April 27, 2009)

* It included all the basic Google programs (Search, mail, Contacts, calendar,etc.) alongs with Synchronization.

5

* Support for widgets, such as a search box into the app on the home screen, virtual keyboards, MPEG-4 video and YouTube/Picasa uploading.

**Donut (1.6) (September 15, 2009)**

* Search expanded to included bookmarks and history.
* More camera integration and features.

**Eclair (2.0-2.1) (April 27,2009)**

* Search expanded to include SMS and MMS messages, and HTML5 support in the browser.
* More camera features, including flash, zoom and white balance.

**Froyo (2.2-2.2.3) (May 20,2010)**

* Support for Bluetooth hands free
* Push Notifications
* Wi-Fi hotspot functionality
* And greater screen resolution.

6

**Gingerbread (2.3-2.3.7) (Decemeber 6,2010)**

* Enhanced copy/paste
* Front-facing camera
* AAC audio and
* Near field Communication (NFC).

**Honeycomb (3.0-3.2.6) (February 22,2011) for tablet**

* Introduced on the Motorola Xoom.
* Honeycomb was a tablet-only version that took advantage of larger screens.

**Ice Cream Sandwich (4.0-4.0.4) (October 18, 2011)**

* Introduced on Samsung’s Galaxy Nexus, Ice Cream Sandwich (ICS) combined Gingerbread ad Honeycomb version into one.
* It also introduced NFC data transfer.

**Jelly bean (4.1 – 4.3.1) (July 9, 2012)**

* It included an improved voice search, which is the counterpart to Apple’s Siri.
* Swiping gestures and predictive text were added to the keyboard.

7

**Kit-Kat (4.4 – 4.4.4) (October 31, 2013)**

* Support for older phones with less than 1 GB of Ram Memory.
* “OK Google” was added to activate a voice search.

**Lollipop (5.0-5.1.1) (November 12, 2014)**

* A redesigned user interface, knows as Material Design.
* Enterprise features included separating personal and business apps.

**Marshmallow (6.0 – 6.0.1) (October 5, 2015)**

* Android Runtime Permission included.
* Standardized fingerprint support.
* Automatic data backup to the Google cloud.
* The Google Now voice search could be accessed from the lock screen.

**Naugat (7.0 – 7.1.2) (August 22, 2016)**

* Emergency info could be displayed on the lock screen.

8

* The Settings menu showed more information without going to a second level.

**Orea (8.0 – 8.1) (August 21, 2017)**

* It improved battery life and performance by limiting what apps can do in the background.
* Users have more control over notification.

**Pie (9.0) (August 6, 2018)**

* It improves battery life by offering adaptive features such as learning how you use apps and change brightness.
* Pie also predicts what you might do next. Slices display relevant parts of apps, and gestures can be used in addition to buttons.

**Android 10 (September 3, 2019)**

**Android 11 (September 8, 2020)**

9

**What is IDE?**

* Enables programmers to consolidate the different aspects of writing a computer program.
* Increases Efficiency – Faster coding with less effort.

**Main Featured of IDE**

* Editing Source Code
* Syntax Highlighting
* AutoComplete (Suggestion)
* Building Executables
* Debugging (Manual Testing)

**Language?**

* A way of communication.
* A programming language is a formal language comprising a set of instructions that produce various kinds of output.
* Most programming languages consist of instructions for computers.

10

**Static and Dynamic Apps**

**Static**

* Static applications are not reliant on Internet connection to an online server or database.
* Apps built in this way are downloaded once, usually periodically updated.
* Are able to function offline only with the device on which they are installed.

**Dynamic**

* Dynamic applications are in some way reliant on an online server or database.
* When connected, these apps retrieve data updates from a main server
* So that any iterative changes to development, design functionality are rolled out across all devices simultaneously.

11

**Front End Language**

* It’s all about what a user able to see.
* It’s may be buttons, image, layout, input field etc.
* To design these components in android we need XML and Android programming (Java/Kotlin).

**Back End Language**

* Backend of anything is not visible to general user or administrator.
* It can be a piece of code or a program running on the server machine to serve the user need.
* Database is also a part of backend i.e., SQLite.

**What is Apk**

* It is a Package file format used by Android Operating system for distribution and installation of Mobile Apps.
* It means Android Application Package.
* Similar to executable file .exe in Windows Operating System.
* It consists of > Application code (.dex files)
* Manifest file
* Assets , Resource files

**12**

**Using Log**

* It is used to display system messages, such as when a garbage collection occurs, and messages that you added to your app with the Log class.
* It displays messages in real time and keeps a history so you can view older messages.

**Using Logcat**

* Same as Log, just difference is that Logcat always check the LOGS property that it is true or false before logging any message.
* Which can be set to true in the debug build type and to false in the release type.

**Using Break Points**

* When you are at the point where you can’t fix a bug just by looking at your code, it’s time to use breakpoints.
* Breakpoints allow you to pause the execution of your app at a particular line of code.

Ex - if(id == R.id.nav\_home){

loadFrag(new AFragment(),false);

13

}else if(id == R.id.nav\_search){

loadFrag(new BFragment(),false);

}

**Using Toast**

* It is used to display information for the short period of time. A toast contains message to be displayed quickly and disappears after sometime.

**XML?**

* It is used for storing and transferring data.
* It is a markup language much like HTML.
* Unlike HTML,XML is case-sensitive, requires each tag is closed properly, and preserves whitespace.
* XML Tags need not to be predefined like HTML(therefore known as extensible).
* A markup language is language that annotates text so that the computer can manipulate that text.
* Most markup languages are human-readable because the annotations are written in a way to distinguish them from the text itself.

14

* It was designed to be self-descriptive.

**XML in Android**

* We create XML layouts in Android, and later alter them using Java logic.
* Resources are the additional files and static content an application needs, such as animations, color schemes, layouts, menu layouts.
* Each layout file must contain one (and only one!) root element.

**View**

* It represents a rectangular area of the screen, and is responsible for displaying information or content and event handling.
* Text, images, and buttons are all Views in Android.

**View Group**

* It is essentially an ‘invisible container’ that holds multiple Views or View Group together and defines their layout properties.
* Common View Groups

15

* A List View displays a list of scrollable items.
* A Grid View displays items in a two-dimensional scrollable grid.
* A Table Layout groups views into rows and columns.

**Root View**

* It is Root Element of XML Layout file.
* Common View Groups
* A Linear Layout aligns its contents into a single direction, whether vertical or horizontal.
* A Relative Layout displays its child content in positions relative to the parent.
* A Frame Layout is a placeholder on a screen that display only a single view(Fragments)

16

**Java**

* It was developed by Sun Microsystems Inc in the year 1991, later acquired by Oracle Corporation.
* It was developed by James Gosling and Patrick Naughton.
* It is a simple programming Language.
* Java makes writing, compiling and debugging programming easy.
* It helps to create reusable code and modular programs.
* Java is a class-based, object-oriented programming language.
* A general-purpose programming language made for developers to write once run anywhere that is compiled Java code can run on all platforms that support Java.
* Java applications are compiled to byte code that can run on any Java Virtual Machine. The syntax of Java is similar to c/c++.
* It is used for:
* Mobile Applications(specially Android apps)
* Desktop Applications

17

* Web applications
* Web servers and application servers
* Games
* Database connection
* And much, much more!

**Java Terminology**

* **Java Virtual Machine(JVM):**
* Writing a program is done by java programmer like you and me.
* The compilation is done by JAVAC compiler which is primary Java compiler included in the Java development kit (JDK).It takes Java program as input and generates bytecode as output.
* It is saved as .class file by the compiler.
* In Running phase of a program, JVM executes the bytecode generated by the compiler.
* **Java Development Kit(JDK):**

18

* It is a complete java development kit that includes everything including compiler, Java Runtime Environment (JRE), Java debuggers, Java docs etc.
* For the program to execute in java, we need to install JDK in our computer in order to create, compile and run the java program.
* **Java Runtime Environment (JRE):**
* JDK includes JRE.
* JRE installation on our computers allows the java program to run.
* For running the java program, a computer needs JRE.

**What is Packages?**

* Packages are divided into two categories:
* Built-in Packages (Packages from the Java API)
* User-defined Packages (Create your own packages)

**Syntax**

Import package.name.Class; //Import a Single class

Import package.name.\*; // Import the whole package

19

**What is Class ?**

* A Class is an entity that determines how an object will behave and what the Object will contain.
* In other words, it is a blueprint or a set of instruction to build a specific type of object.
* Contains variables(fields), methods & Object.

**xlmns : (namespace URI)**

**Uniform Resource Indicator (URI):**

* In computing, it is a string of characters used to identify a name of a resource.
* Such identification enables interaction with representations of the resource over a network, typing the World Wide Web, using specific protocols.

**Xlms:prefix = “URI”**

* Ex: <https://schemas.android.com/apk/res/android:id> is the URI here
* The most common URI is the Uniform Resource Locator (URL) which identifies an Internet domain address.

20

Another, not so common type of URI is the Uniform Resource Name (URN).

**XML Namespace**

* XML namespaces are used for providing uniquely named elements and attributes in an XML document.
* Xmlns: android describes the android namespace.
* Also suppose we write our own textview widget with different features compared to android textview, android namespace helps to distinguish between our custom textview widget and android textview widget.

**Android Architecture (LAYER)**

* **LINUX KERNAL:**
* Hardware Drivers (USB Display, Bluetooth etc)
* Power Management
* Process Management
* Memory Management
* It is like heart of android operating system and present at the bottom of android architecture.
* It contains essential hardware drivers like display, audio, camera, Bluetooth, wifi etc.

21

* It provides functionalities such as power management, process management and memory management.
* It provides a level of abstraction between device hardware and upper layers of Android software stack.
* **LIBRARIES:**
* Graphics (OpenGL,SGL, FreeType)
* Media
* SSL & WebKit
* IBC
* SQlite
* Surface Manager
* Above Linux kernel there are native libraries such as SQLite,WebKit,OpenGL,SSL etc.
* Some of these libraries are briefly explained below.
* **SQLite –** It provides various classes used for database management.
* **WebKit –** It is a web browser engine used to display internet content.
* **SSL -** It provides internet security.

22

* **OpenGL –** It is a Java interface to the OpenGL ES 3D graphics rendering API.
* **ANDROID RUN TIME:**
* Android Libraries
* Dalvik Virtual Machine
* It is third section which is also present on the second layer from bottom of Android architecture.
* Android runtime provides core libraries and Dalvik virtual Machine (DVM).
* These core libraries enable Android developers to write Android applications using standard Java programming language.
* DVM is a major component of Android OS. It is same as like Java Virtual Machine (JVM).
* DVM is responsible for running Android applications. It consumes less memory and provides very fast performance.

23

* **APPLICATION FRAMEWORK**
* Location-Based Services
* Content Providers
* Window Manager
* Activity Manager
* Package Manager
* Telephony
* Bluetooth/NFC/Wi-Fi Direct
* Notification
* Views
* Resource Manager
* It is fourth section and third layer present on the top of native libraries and android runtime.
* It provides various API’s like activity manager, resource manager, content providers, telephony manager etc.
* These API’s are used by Android application developers.
* **APPLICATION LAYER**
* Native Apps (Contacts, Map, Browser etc).
* Third-Party Apps.
* Developer Apps

24

* Applications layer is present at the top. Various applications created by developers like games, contacts,browser etc are installed on this layer.

**Android Components**

* **Activities**
* An activity represents a single screen with a user interface, in-short Activity performs actions on the screen.
* Public class MainActivity extends Activity{

}

* **Services**
* A service is a component that runs in the background to perform long – running operations.
* public class MyService extends Service{

}

* **Broadcast Receivers**
* Broadcast Receivers simply respond to broadcast messages from other applications or from the system.

**25**

* Public class MyReceiver extends BroadcastReceiver{

Public void onReceive (context, intent){

}

}

* **Content Providers**
* A content provider component supplies data from one application to others on request. Such requests are handled by the methods of the ContentResolver class.
* Public class MyContentProvider extends ContentProvider{

Public void onCreate(){

}

}

**Main Components of Lifecycle**

Method Description

**onCreate** called when activity is first created.

**onStart** called when activity is becoming visible to the user.

**onResume** called when activity will start interacting with the user.

**onPause** called when activity is not visible to the user.

**onStop** called when activity is no longer visible to the user.

**onRestart** called after your activity is stopped, prior to start.

26

**onDestroy** called before the activity is destroyed.

**What is Animation?**

* Animation is the process of creating motion and shape change.
* These are used when the UI changes state,such as when new content loads.
* Animations also add a polished look to your which gives it a higher quality look and feel.

**Type of Animations**

* **Scale** Animation used to make a smaller or larger view either on x axis or y axis.
* **Rotate** Animation used to rotate a view around a pivot point a certain number of degrees.
* **Translate** Animation used to move a view along the x or y axis.
* **Alpha** Animation Transparency of a view can be changed.

27

Attribute Description

Android:duration=”1000” The duration in which animation is completed

Android:fillAfter=”true” defines whether the view should be visible or not at the end of the animation.

Android:interpolator=”@android:anim/linear\_interpolator”

Refers to the rate of change in animation.

Anroid:startOffset=”2000” refers to the waiting time before an animation start.

Android:repeatMode =”restart” Mode of Repetition two types of Modes: Restart(default) and reverse.

Android:repeatCount = “infinite” defines the number of repetitions on animation.

**What is Lottie?**

* A Lottie is a JSON-based animation file format that enables designers to ship animations on any platform as easily as shipping static assets.
* To use Lottie animations in our project we need to integrate Lottie Project(Library).

28

**What is Library?**

* It is structurally the same as an Android app module. It can include everything needed to build an app.
* Instead of compiling into an APK, it compiles into an Android Archive (AAR) file that is used as a dependency for an Android app module.

<com.airbnb.lottie.LottieAnimationView

android:id=”@+id/laView”

android:Layout\_width = “wrap\_content”

android:Layout\_height = ”wrap\_content”

android:Layout\_centerInParent = “true”

app:Lottie\_repeatCount = “10”

app:Lottie\_repeatMode = “reverse”/>

**List?**

* A list is any information displayed or organized in a logical or linear formation.
* It is a view which groups several items and display them in vertical scrollable list.

29

**Styles in Android?**

* A style resource defines the format and look for a UI.
* A style can be applied to an Individual View(from within a layout file) or to an entire Activity or application (from within the manifest file).

**Themes in Android**

* A theme is nothing but an Android style applied to an entire Activity or application, rather than an individual View.
* When a style is applied as a theme, every View in the Activity or application will apply each style property that it supports.

**What is Card View?**

* Display of data in similarly styled containers.
* These cards have a default elevation above their containing view group, so the system draws shadows below them.

30

**Attributes of Card View**

**Attributes**  **Description**

**cardBackgroundColor** Defines the background color of the card.

**cardElevation** Defines the elevation

**cardCornerRadius** Defines the corner radius card.

**cardUseCompatPadding** Adds space around the card view to prevent card’s shadow

**Recycler View?**

* In Android 5.0 Lollipop, Android introduced Recycler View widget.
* It is flexible and efficient version of ListView.
* It is an container for rendering larger data set of views that can be recycled and scrolled very efficiently.

**Why use RecyclerView?**

* RecyclerView makes it easy to efficiently display large sets of data.

31

* You supply the data and define how each item looks and the RecyclerView library dynamically creates the elements when they’s are needed.
* As the name implies, RecyclerView recycles those individual elements.
* When an item scrolls off the screen, RecyclerView does not destroy its View. Instead, RecyclerView reuses the view for new items that have scrolled onscreen.
* This reuse vastly improves performance, improving your app’s responsiveness and reducing power consumption.

**Toolbar**

* It is similar to an ActionBar(now called as App Bars).
* Toolbar is a Viewgroup that can be placed at anywhere in the Layout.We can easily replace an ActionBar with Toolbar.
* It provides more feature than ActionBar. A Toolbar may contain a combination of element from start to end.

**Drawable**

* A Drawable resource is a general concept for a graphic which can be drawn.

32

* The Simplest case is a graphical file (bitmap), which would be represented in Android via a BitmapDrawable class.
* Every Drawable is stored as individual files in one of the res/drawable folders.

**What is Toast?**

* It is used to display information for the short period of time. A toast contains message to be displayed quickly and disappears after sometime.

**Custom Layout**

* Create a customized layout for your toast notification.
* To create a custom layout, define a View layout , in XML or in your application code, and pass the root View object to the setView(View) method.

**Changing the Gravity**

* Change the Gravity of your toast notification.
* Use toast.setGravity(Gravity.TOP|Gravity.LEFT,0,0); method.

33

**What is Log?**

* It is used to display system messages,such as when a garbage collection occurs,and messages that you added to your app with the Log class.
* It displays messages in real time and keeps a history so you can view older messages.

**Basic of Log**

* Log Message Syntax

Log.d(tag,message);

* Method, listed in order from the highest to lowest priority:

Log.e(String,String) (error)

Log.w(String,String) (warning)

Log.i(String,String) (information)

Log.d(String,String) (debug)

Log.v(String,String) (verbose)

* Logcat Message Format

Date time PID-TID/package priority/tag:message

2020-12-21 16:08:57.938 2177-4533/system\_process D/TAG:MESSAGE

34

**What is Alert Dialog Box?**

* They’re used to help users answer questions, make selections,confirm actions,and read warning or error messages.
* It is a windows that partially obscures the Actually that launched it.

**AlertDialog vs AlertDialog.Builder**

Object Return Type

New AlertDialog.Builder (MainActivity.this).create AlertDialog class Object(Method Call) for Single Button

**Syntax**: AlertDialog alertDialog = new AlertDialog.Builder(MainActivity.this).create;

New AlertDialog.Builder(MainActivity.this) AlertDialog.Builder class Object (Constructor Call) for 2 or 3 Buttons

**Syntax**: AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);

**Using Custom Layout**

* Create a customized layout for your toast notification.
* To Create a custom layout,define a View layout, in XML or in your application code, and pass the root View object to the setContentView(View) method.

35

Notification**?**

* It is a message you can display to the user outside of your application’s normal UI.
* When you tell the system to issue a notification, it first appears as an icon in the notification area.
* To see the details of the notification, the user the notification drawer.
* It provides short, timely information about the action happened in the application, even it is not running.
* It provide the user with reminders, communication from other people, or other timely information from your app.
* It displays the icon, title and some amount of content text.

**Implicit Calls? (caller,message,camera etc)**

* Implicit means Internal calls
* These are use without a class name, where Android will help determine an appropriate Activity to handle the intent.

36

**Implicit vs Explicit**

* An action to be performed is declared by implicit intent not the Component. Then android operating system will filter out component which will response to the action.
* The targeted component is specified by explicit intent. So only the specified target component invoked.

**What is Fragment?**

* Represents a reusable portion of your app’s UI.
* A fragment defines and manages its own layout, has its own lifecycle, and can handle its own input events.

**Methods of Shared Preference class**

Method Description

1. onAttach(Activity) It is called only once when it is attached with activity.
2. onCreate(Bundle) It is used to initialize the fragment.
3. onCreateView(LayoutInflater,ViewGroup,Bundle)

creates and returns view hierarchy.

1. onActivityCreated(Bundle) It is invoked after the completion of onCreate() method.

**37**

1. onViewStateRestored(Bundle)

It provides information to the fragment that all the saved state of fragment view hierarchy has been restored.

1. OnStart() makes the fragment visible.
2. OnResume() makes the fragment interactive.
3. onPause() is called when fragment is no longer interactive.
4. onStop() is called when fragment is no longer visible.

10. onDestroyView() allows the fragment to clean up resources.

11. onDestroy() allows the fragment to do final clean up of fragment state.

12. onDetach() It is called immediately prior to the fragment no longer being associated with its activity.

**What is Dynamic**

* (Of a process or system) characterized by constant change, activity or progress.
* Similarly a Dynamic App’s Content, design or functionality changes w.r.t Time.
* For eg: Whatsapp, Facebook,..etc.

**38**

**What is API (REST)**

* REST determines how the API looks like. It stands for “Representational State Transfer”.
* Each URL is called a request while the data sent back to you is called a response.
* An API is an Application programming Interface.
* It is a set of rules that allow programs to talk to each other.
* The developers creates the API on the server an allows the client to talk to it.

**What is JSON**

* JSON stands for JavaScript Object Notation.
* JSON is a lightweight format for storing and transporting data.
* JSON is often used when data is sent from a server to a web page.
* JSON is “self-describing” and easy to understand.

**JSON Rules**

* Data is in name/value pairs
* Data is separated by commas

**39**

* Curly braces hold objects {}
* Square brackets hold arrays []

**JSON Data**

* JSON data is written as name/value pairs, just like JavaScript object properties.
* A name/value pair consists of a field name (in double quotes), followed by a colon, followed by a value.

“firstName”:”John”

**JSON Objects**

* JSON objects are written inside curly braces.
* Just like in JavaScript, Objects can contain multiple name/value pairs:

{“firstName”:”John”,”lastName”:”Harper”}

**JSON Arrays**

* JSON arrays are written inside square brackets.
* Just like in JavaScript, an array can contain objects:

“Students”:[

{“firstName”:”John”,”lastName”:”Harper”},

{“firstName”:”Anna”,”lastName”:”Jones”},

{“firstName”:”Bret”,”lastName”:”Lee”} ]

**40**

**Web View**

* It’s easier to build a WebView in your Android app that shows a web page with all the user data,
* Rather than performing a network request, then parsing the data and rendering it in an Android layout.

**Shared Preference**

* One of the ways to store data in Android.
* It saves and retrieves data in the form of key & value pair.

**Method and Description**

1. apply() – It is an abstract method. It will commit your changes back from editor to the sharedPreference object you are calling.

2. clear() – It will remove all values from the editor.

3. remove(String key) – It will remove the value whose key has been passed as a parameter.

4. putString(String key, String value) – It will save a float value in a preference editor.

50

5. putInt(String key, int value) – It will save a Integer value in a preference editor.

6. putLong(String key,long value) – It will save a long value in a preference editor.

7. putFloat(String key, float value) – It will save a float value in a preference editor.

**What is Database**

* It is an organized collection of structured information, or data, typically stored electronically in a computer system.
* A database is usually controlled by a database management system (DBMS).
* Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.
* Most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient.
* The data can then be easily accessed, managed, modified, updated, controlled, and organized.

**51**

* Most databases use structured query language (SQL) for writing and querying data.

**SQLite**

* It is an open-source relational database.
* It is lighter version of SQL.
* Android comes with built-in SQLite database implementation.
* In order to access this database, you don’t need to any kind of connections for it like JDBC,ODBC etc.

**Main Components**

* Table Structure (Rows & Columns)
* Types of Data (text,double,int,float,blob,Boolean …)
* Main Constraints of Data in Table (Primary Key, NotNull, Foreign Key, Unique...)
* Queries (Select, Insert,Update and Delete)

**Room Database**

* Abstraction over SQLite
* Less Boilerplate
* Compile Time Verification of SQL Queries.

**52**

* Entities (Table)
* DAO (Data Access Objects)
* Database
* Type Convertors
* Migrations
* Entities(Table)

ID NAME PHONE

1 John 9999

2 Sean 2222

3 Jack 7777

* DAO
* Data Access Objects
* Interface containing methods to access database – CRUD Operations
* We can define multiple DAOs

53



**What is Sensor?**

* A sensor is a component used in mobile devices.
* It detects the changes in the environment (Such as changes in brightness, magnetic fields, temperature, and gravity)
* And movement (Such as the device being moved, flipped , or picked up),
* And convert them into electronic signals that can be processed by the device.
* A mobile device sensor consists of precision components, which are sensitive to sources of external interference and physical factors.

**54**

* Avoid dropping your device or using it in environments with strong magnetic fields, abnormal humidity, extreme temperatures, or other unfavorable factors to avoid damaging its sensor.

**Types of Sensors**

* Ambient light Sensor
* Proximity sensor
* Gravity sensor (accelerometer)
* Gyroscope
* Barometer
* Fingerprint sensor

1. **Ambient light Sensor**

* Automatically adjusts the screen brightness of your device according to the amount of ambient light, making it more comfortable to look at the screen.
* The sensor not only supports automatic white balance (AWB) when taking photos, but also works with the proximity sensor to prevent misoperations when for example, the device is in your pocket.

**55**

**How it works:**

* The sensor generates strong or weak currents according to the amount of ambient light it senses,
* And the device increases or decreases the screen brightness accordingly.
* Please note that using a non-official protective case or unmatching protective film may block the ambient light sensor and effect its functions.

2. **Proximity Sensor**

* Detects the presence of nearby objects.
* Equipped with a proximity sensor, your device automatically turns off the screen when it detects that it is close to your ear. (10cm range)
* This helps prevent possible misoperations.

**How it works:**

* It is generally located at the top of the screen and near the receiver.
* It detects the distance between an object and the device by calculating changes in the infrared light signals it receives.

**56**

* The working range of a proximity sensor is generally 10 cm.

3. **Gravity sensor (accelerometer)**

* Allows your device to automatically switch between landscape and portrait modes.
* Count your daily steps,
* Indentify viewing orientation.
* Use compass apps, and recognize motion gestures ( such as picking up and flipping your device).

**How it works:**

* The sensor measures changes in the distance between the capacitance plates caused by motion on three axes (X,Y and Z).
* And determines the instantaneous acceleration and deceleration forces accordingly.
* For example, you can use device’s gravity sensor control a moving ball or steer a race car in games, shake to switch songs, flip to mute ringtones, and so on.

**57**

**What is Service?**

* A Service is a component that runs in the background.
* To Perform long- runnign operations without needing to interact with the user.
* And it works even if application is destroyed.
* A service can essentially take two states.
* **Started**

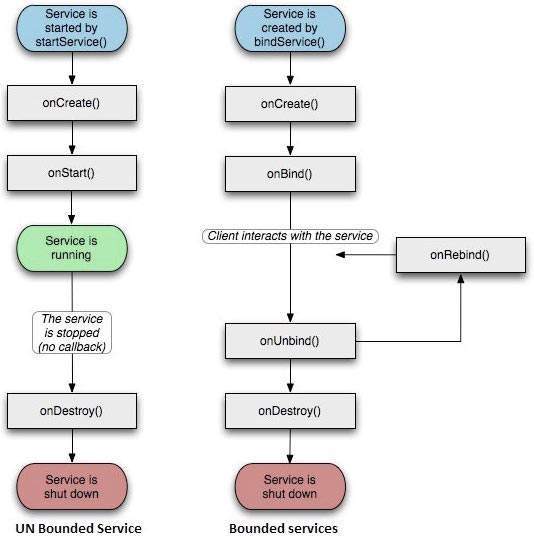
1. A service is started when an application component,such as an activity , starts it by calling startService().
2. Once started, a service can run in the background indefinitely, even if the component that started it is destroyed.

* **Bounded**

1. A Service is bound when an application component binds to it by calling bindService().
2. A bound service offers a client-server interface that allows components to interact with the service,
3. Send requests, get results, and even do so access process with interprocess communication(IPC).

**58**

**LIFE CYCLE OF SERVICE**



**Restart options for Service**

* **Service.START\_STICKY**
* Service is restarted if it gets terminated.
* Intent data passed to the onStartCommand method is null.
* Used for services which manages their own state and do not depend on the Intent data.
* **Service.START\_NOT\_STICKY**
* Service is not restarted.
* Used for services which are periodically triggered anyway.

**59**

* The service is only restarted if the runtime has pending startService() calls since the service termination.
* **Service.START\_REDELIVER\_INTENT**
* Similar to Service.START\_STICKY but the original intent is re-delivered to the onStartCommand method.

**What is Alarm Manager?**

* Android Alarm Manager allows you to access system alarm.
* By the help of this, we can schedule our Application Task to run at a specific time in the future. It works whether your phone is running or not.
* It holds a CPU wake lock that provides guarantee not to sleep the phone until broadcast is handled.

**What is Broadcast Receiver?**

* A Broadcast receiver (receiver) is an Android component which allows you to register for system or application events.
* All registered receivers for an event are notified by the Android runtime once this event happens.

**60**

* For example applications can register for the ACTION\_BOOT\_COMPLETED system event which is fired the Android system has completed the boot process.

**Types of Alarm ?**

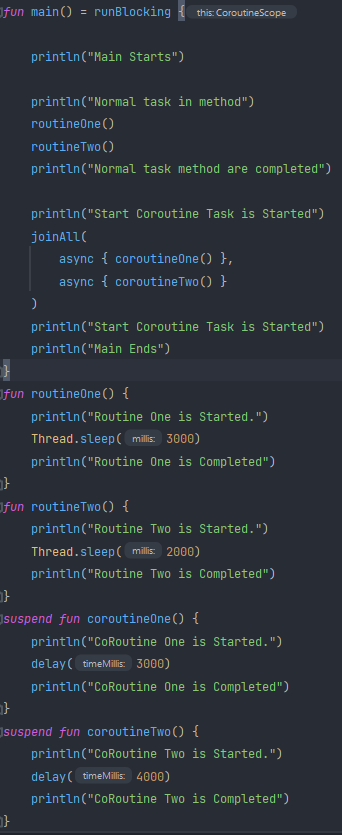
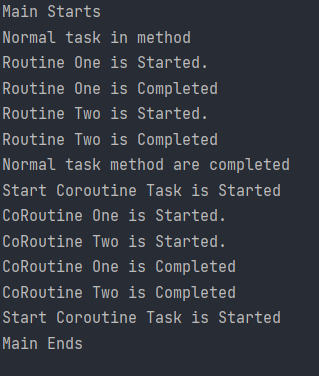
* ELAPSED\_REALTIME – Fires the pending intent after the specified length of time since device boot. If the device is a sleep, it fires when the device is next awake.
* ELAPSED\_REALTIME\_WAKEUP –Fires the pending intent after the specified length of time since device boot. It wakes up the device if it is a sleep.
* RTC – Fires the pending intent at a specified time. If the device is a sleep. It will not be delivered until the next time the device wakes up.
* RTC\_WAKEUP – Fires the pending intent at a specified time, waking up the device if a sleep.

61

Coroutines Kotlin

What is Routine?

* Routines(Function) are non cooperative
* funA() calls funB()
* Controls remains funB() until it complete its execution.

71

* A Coroutine can be thought of as an instance of **suspendable Computation.**
* i.e one that can suspend at some points and later resume execution .
* Coroutines calling each other (and passing data back and fourth) can form the machinery for cooperative multitasking.
* Not new in Kotlin.
* Based on establish concepts from other languages.
* Added version 1.3.
* A Coroutine is a concurrency design pattern that use can use to simplify code that executes asynchronously.
* Make a Asynchronous code very easy to write and manage.

**Concurrent or Asynchronous Programming**

* CPU executes one instruction at a time.
* Instruction are executed in a sequence.
* Single block of sequential code is called thread.
* Modern CPUs/Android device have multiple cores.
* Meaning CPU can do multiple things at a time.
* Multiple Threads running at same time.

72

* In Android,one single thread is responsible for showing/updating UI, called Main Thread.
* Main Thread can not be used for long running operating eg. Database operations,network operations etc.
* That’s why we need concurrent/Asynchronous programming in android.

**Problems with Threads**

* Creating native threads is very expensive performance wise.
* Context Switching is very expensive operation.(change the currently running thread)
* On JVM, Each threads take 1MB (very costly).
* Threads are memory management is very complicated.

**Kotlin Coroutines**

* Threads are managed by the OS.
* Coroutines are managed by programmer or programming language.
* “Green threads,fibers,lightweight threads” are different names for coroutine like constructs.
* Link all know here -> <https://developer.android.com/kotlin/coroutines>

**73**

* They are concurrently.
* Can wait for each other.
* Communicate with each other.
* Can form parent-child relationship.
* Provides Structures Concurrency.
* Coroutines can change threads very easily.
* Can be suspended (paused) and resumed (Cooperative routines)
* While suspended (paused), they don’t block the thread.
* Make **Asynchronous** code look like **synchronous**.
* Save us from call-back hell.
* Coroutines are recommended for async programming on android.
* Major android libraries have built in coroutines suppor eg. ROOM,retrofit.
* **But Coroutines are not threads(very important)**
* Coroutines run inside a thread.
* Many Coroutines (even a million, depends your machine) can run on a single thread.
* Because they are lightweight ,almost free
* Coroutine use lower RAM (1 Coroutine takes few bytes)

**74**

* Because when you execute 1,000,000 coroutines, it doesn’t have to create 1,000,000 threads.
* Coroutine can help to you optimize the threads usage.
* Makes the execution more efficient.
* No need to care about the thread anymore.

**Coroutine Builders**

* Coroutine Builders are functions that create & launch a coroutine.
* Called like normal functions because they are not suspending functions.
* Coroutins Builder are
* launch{}
* async{}
* runBlocking{}
* **launch{}**
* Extension function on **Coroutine Scope**.
* Create and launches a new coroutine.
* Doesn’t block the thread.
* Returns a job object.
* Unable to access result outside of coroutine (Fire and Forget).
* Normally used when we don’t need the coroutine result outside the coroutine.

**75**

* **Async{}**
* Extension function on **Coroutine Scope.**
* Create a launches a new Coroutine.
* Doesn’t block the thread .
* Returns a Deferred<T> object (Future/Promise in other frameworks).
* Can wait & access the coroutine result outside the coroutine.
* Async can be used to perform an asynchronous task which return a value.
* Use for parallel/concurrent coroutine execution.
* **runBlocking{}**
* Top level function.
* Create and launches a coroutine.
* But Blocks the thread.
* Mainly used for learning and testing purpose.
* Used for testing Suspend Function.
* Can be used for Calling suspend function as blocking functions.