SHRI DHARMASTHALA MANJUNATHESHWARA COLLEGE

(Autonomous) UJIRE D. K. -574240

Topic:- Write the Questions & Answers of Module – 1 & Module - 2

Subject:- Android & Mobile Application Development

Submitted To:- Rakshitha Ma'am
Assistant Professor
Dept. of B.Voc
Software & App Development

Submitted By:- Abhishek U S
211301
IInd Year B.Voc
Software & App Development

Date:-29/05/2023

Place:- Ujire

UNIT 1

1. Explain application priority and process states.

The order in which processes are killed to reclaim resources is determined by the priority of the hosted applications. An application's priority is equal to its highest-priority component.

If two applications have the same priority, the process that has been at a lower priority longest will be killed first. Process priority is also affected by interprocess dependencies; if an application has a dependency on a Service or Content Provider supplied by a second application, the secondary application will have at least as high a priority as the application it supports.

Android Process States

- Foreground Process These processes are assigned the highest level of priority. ...
- Visible Process ...
- Service Process ...
- Background Process ...
- Empty Process

2. With diagram explain the Android architecture.

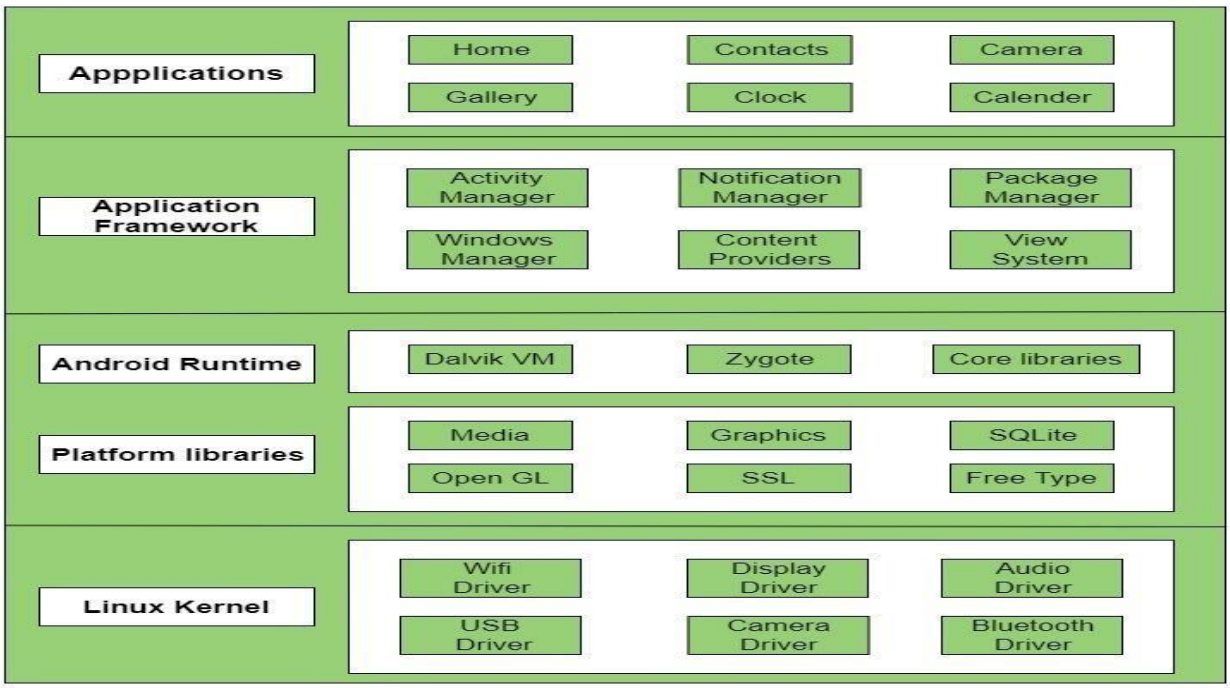
Android architecture contains different number of components to support any android device needs. Android software contains an open-source Linux Kernel having collection of number of C/C++ libraries which are exposed through an application framework services.

Among all the components Linux Kernel provides main functionality of operating system functions to smartphones and Dalvik Virtual Machine (DVM) provide platform for running an android application.

The main components of android architecture are following:-

- Applications
- Application Framework
- Android Runtime
- Platform Libraries
- Linux Kernel

Pictorial representation of android architecture with several main components and their sub components –



3. Describe the lifecycle of an Activity.

Android Activity Lifecycle is controlled by 7 methods of android.app.Activity class. The android Activity is the subclass of ContextThemeWrapper class.

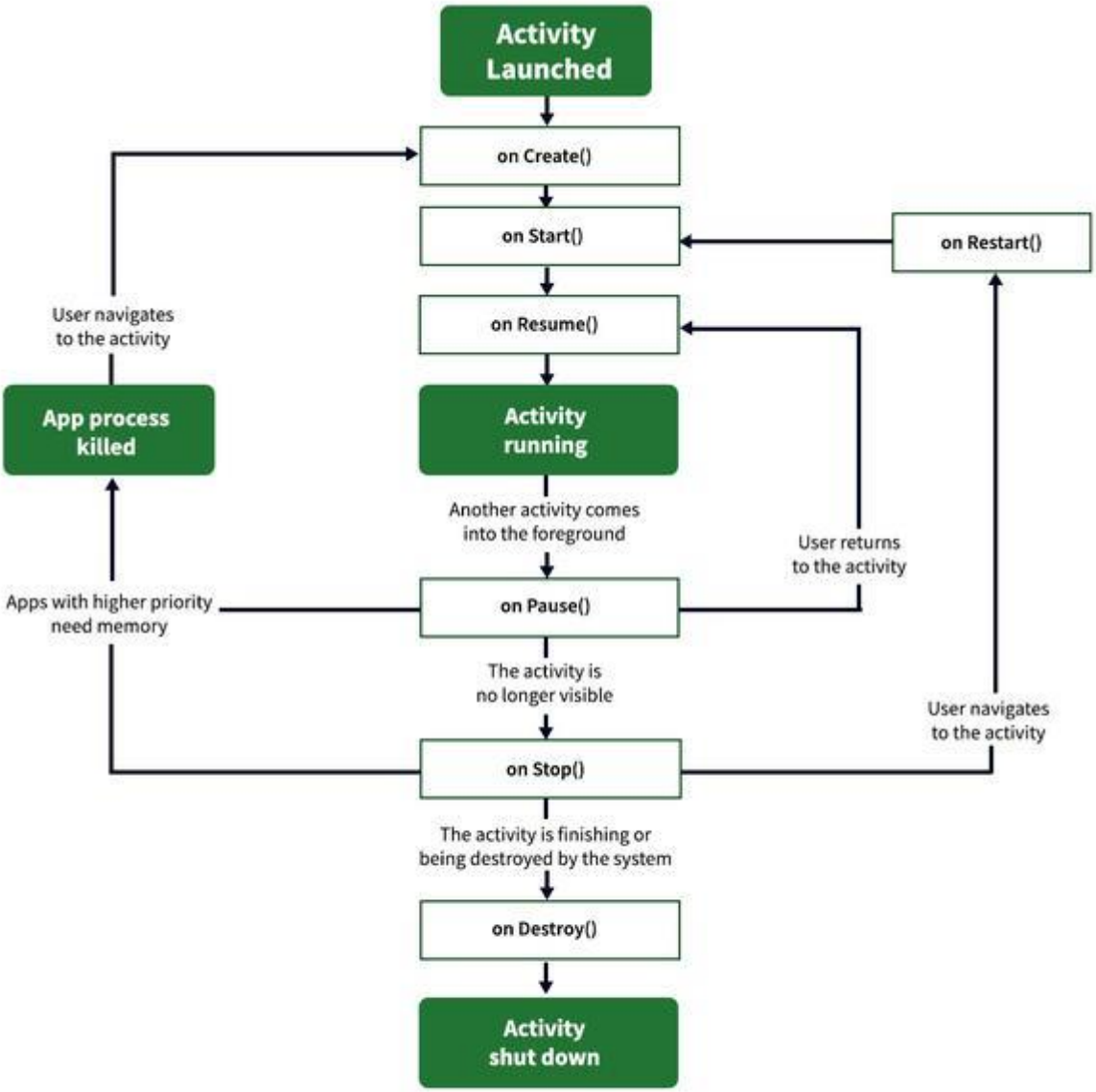
An activity is the single screen in android. It is like window or frame of Java.

By the help of activity, you can place all your UI components or widgets in a single screen.

The 7 lifecycle method of Activity describes how activity will behave at different states.

Android Activity Lifecycle methods

Let's see the 7 lifecycle methods of android activity.



Activity Lifecycle in Android

4. Briefly explain the Anatomy of Android Application.

Android applications are created by bringing together one or more components known as Activities. An activity is a single, standalone module of application functionality that usually correlates directly to a single user interface screen and its corresponding functionality. An appointments application might, for example, have an activity screen that displays appointments set up for the current day. The application might

also utilize a second activity consisting of a screen where new appointments may be entered by the user.

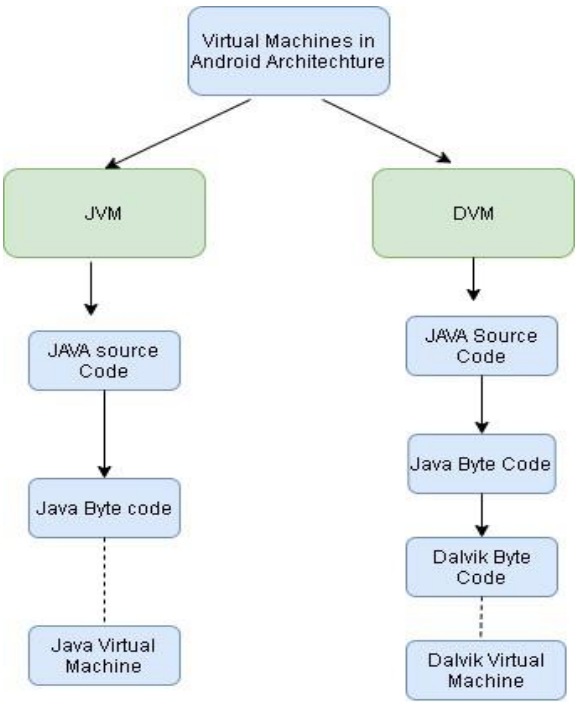
Activities are intended as fully reusable and interchangeable building blocks that can be shared amongst different applications. An existing email application, for example, might contain an activity specifically for composing and sending an email message. A developer might be writing an application that also has a requirement to send an email message. Rather than develop an email composition activity specifically for the new application, the developer can simply use the activity from the existing email application.

5. **What is DVM.**

Dalvik Virtual Machine is a Register-Based virtual machine. It was designed and written by Dan Bornstein with contributions of other Google engineers as part of the Android mobile phone platform. The Dalvik virtual machine was named after Bornstein after the fishing village "Dalvík" in Eyjafjörður, Iceland, where some of his ancestors used to live.

Working of DVM

The Java Compiler(javac) converts the Java Source Code into Java Byte-Code(.class). Then DEX Compiler converts this (.class) file into in Dalvik Byte Code i.e. ".dex" file.



6. **What the differences between DVM and JVM.**

Difference between DVM and JVM

	DVM (Dalvik Virtual Machine)	JVM (Java Virtual Machine)
1	It is Register based which is designed to run on low memory.	It is Stack based.
2	DVM uses its own byte code and runs the ".Dex" file. From Android 2.2 SDK Dalvik has got a Just in Time compiler	JVM uses java byte code and runs ".class" file having JIT (Just In Time).
3	DVM has been designed so that a device can run multiple instances of the VM efficiently. Applications are given their own instance.	A single instance of JVM is shared with multiple applications.
4	DVM supports the Android operating system only.	JVM supports multiple operating systems.
5	For DVM very few Re-tools are available	For JVM many Re-tools are available.
6	There is a constant pool for every application.	It has a constant pool for every class.
7	Here the executable is APK.	Here the executable is JAR.

7. **Explain the steps to create an activity.**

We create New Activity in Android Studio to create XML file for designing UI and java file coding. Below are the steps to create new Activity in Android Studio:

How to Create New Activity in Android Studio:

Step 1: Firstly, click on app > res > layout > Right Click on layout. After that Select New > Activity and choose your Activity as per requirement. Here we choose Blank Activity as shown in figure below.

Create Activity in android studio

Step 2: After that Customize the Activity in Android Studio. Enter the "Activity Name" and "Package name" in the Text box and Click on Finish button.

Customize the Activity

Step 3: After that your new Activity in Layout will be created. Your XML Code is in Text and your Design Output is in Design.

8. What are steps to be followed to create an emulator.

To create an Android emulator on your system, follow these steps:

1. Start the Android SDK Manager (select Start | All Programs | Embarcadero RAD Studio | Android Tools).
2. In the Android SDK Manager, select Tools | Manage AVDs.
3. In the Android Virtual Device Manager, click the New button to create a new virtual device.
4. In the Create new Android Virtual Device (AVD) dialog box, select an Android device to emulate, and enter the details describing the Android device you want to emulate. In order to run a FireMonkey mobile application, your Android emulator must meet the following requirements:

In Target, select an Android SDK with an API level that is 17 or higher. The drop-down list contains your installed versions of the Android SDK.

Under Emulation Options, check Use Host GPU.

In Device, select the Android device to emulate.

Tip: Emulating an older Android device like the Nexus S might be faster than emulating a newer, larger device like the Nexus 10.

5. Click OK twice to create your new Android emulator.
6. You can now view your emulator in the Android Virtual Device Manager.

UNIT 2

1. Explain android UI testing.

Android UI testing can test various aspects of the user interface on Android devices. It helps ensure that your Android app provides a consistent and user-friendly experience across all devices. It includes testing the following key aspects:

- **The layout of UI elements**: The placement of different UI elements such as icons, buttons, images, headings, text fields, selection fields, checkboxes, etc. is thoroughly checked if they are placed as expected.
- **Responsiveness of the UI**: It checks how the application appears on different screen resolutions to test the adaptability of the UI. By testing the Android Application using a real device cloud you can test your app on various popular Android devices in real-time, under real user conditions.
- **Overall look and feel of the UI**: This includes testing the font, alignment, and display type, of the Android application. It basically checks how user-friendly the UI is from the users' perspective with usability tests.

2. What are handlers?

A Handler allows you to send and process Message and Runnable objects associated with a thread's MessageQueue. Each Handler instance is associated with a single thread and that thread's message queue. When you create a new Handler it is bound to a Looper. It will deliver messages and runnables to that Looper's message queue and execute them on that Looper's thread.

There are two main uses for a Handler: (1) to schedule messages and runnables to be executed at some point in the future; and (2) to enqueue an action to be performed on a different thread than your own.

Scheduling messages is accomplished with the post(Runnable), postAtTime(java.lang.Runnable, long), postDelayed(Runnable, Object, long), sendEmptyMessage(int), sendMessage(Message),

sendMessageAtTime(Message, long), and sendMessageDelayed(Message, long) methods. The post versions allow you to enqueue Runnable objects to be called by the message queue when they are received; the sendMessage versions allow you to enqueue a Message object containing a bundle of data that will be processed by the Handler's handleMessage(Message) method (requiring that you implement a subclass of Handler).

3. **Difference between thread and services.**

	Service	**Thread**	**IntentService**	**AsyncTask**
When to use ?	Task with no UI, but shouldn't be too long. Use threads within service for long tasks.	- Long task in general. - For tasks in parallel use Multiple threads (traditional mechanisms)	- Long task **usually** with no communication to main thread. **(Update)**- If communication is required, can use main thread handler or broadcast intents - When callbacks are needed (Intent triggered tasks).	- Small task having to communicate with main thread. - For tasks in parallel use multiple instances OR Executor
Trigger	Call to method onStartService()	Thread start() method	Intent	Call to method execute()
Triggered From (thread)	Any thread	Any Thread	Main Thread (Intent is received on main thread and then worker thread is spawed)	Main Thread
Runs On (thread)	Main Thread	Its own thread	Separate worker thread	Worker thread. However, Main thread methods may be invoked in between to publish progress.
Limitations / Drawbacks	May block main thread	- Manual thread management - Code may become difficult to read	- Cannot run tasks in parallel. - Multiple intents are queued on the same worker thread.	- one instance can only be executed once (hence cannot run in a loop) - Must be created and executed from the Main thread

4. Explain Intent.

Android Intent is the *message* that is passed between components such as activities, content providers, broadcast receivers, services etc.

It is generally used with startActivity() method to invoke activity, broadcast receivers etc.

The **dictionary meaning** of intent is *intention or purpose*. So, it can be described as the intention to do action.

The LabeledIntent is the subclass of android.content.Intent class.

Android intents are mainly used to:

○ Start the service ○

Launch an activity ○

Display a web page ○

Display a list of

contacts ○ Broadcast a

message ○ Dial a

phone call etc.

5. Explain any 7 attributes of layout.

1. **android:id** : This is the ID which uniquely identifies the view
2. **android:layout_width** : This is the width of the layout
3. **android:layout_height** : This is the height of the layout
4. **android:layout_margin** : This is the extra space outside of the view. For example if you give `android:marginLeft=20dp`, then the view will be arranged after 20dp from left
5. **android:layout_padding** : This is similar to **android:layout_margin** except that it specifies the extra space **inside** the view
6. **android:layout_gravity** : This specifies how child Views are positioned

7. **android:layout_weight** : This specifies how much of the extra space in the layout should be allocated to the view
8. **android:layout_x** : This specifies the x-coordinate of the layout
9. **android:layout_y** : This specifies the y-coordinate of the layout

6. What are different kind of log files present in android.

- **Event Log**: a high-level log that records information about network traffic and usage, such as login attempts, failed password attempts, and application events.
- **Server Log**: a text document containing a record of activities related to a specific server in a specific period of time.
- **System Log (syslog)**: a record of operating system events. It includes startup messages, system changes, unexpected shutdowns, errors and warnings, and other important processes. Windows, Linux, and macOS all generate syslogs.
- **Authorization Logs and Access Logs**: include a list of people or bots accessing certain applications or files.
- **Change Logs**: include a chronological list of changes made to an application or file.
- **Availability Logs**: track system performance, uptime, and availability.
- **Resource Logs**: provide information about connectivity issues and capacity limits.
- **Threat Logs**: contain information about system, file, or application traffic that matches a predefined security profile within a firewall.

7. What is Android View Group

An activity or fragment contains Views and ViewGroups. A view is a widget that has an appearance on screen. Examples of views are buttons, labels, and text boxes. A view derives from the base class `android.view.View`.

One or more views can be grouped together into a ViewGroup. A ViewGroup (which is itself a special type of view) provides the layout in

which you can order the appearance and sequence of views. Examples of ViewGroups include `LinearLayout` and `FrameLayout`. A ViewGroup derives from the base class `android.view.ViewGroup`.

Android supports the following ViewGroups:

- `LinearLayout`
- `AbsoluteLayout`
- `TableLayout`
- `RelativeLayout`
- `FrameLayout`
- `ScrollView`

8. What are intent filters.

Android OS uses filters to pinpoint the set of Activities, Services, and Broadcast receivers that can handle the Intent with help of specified set of action, categories, data scheme associated with an Intent. You will use **<intent-filter>** element in the manifest file to list down actions, categories and data types associated with any activity, service, or broadcast receiver.

Following is an example of a part of **AndroidManifest.xml** file to specify an activity **com.example.My Application.CustomActivity** which can be invoked by either of the two mentioned actions, one category, and one data –

```
<activity android:name=".CustomActivity"
android:label="@string/app_name">    <intent-filter>
    <action android:name="android.intent.action.VIEW" />
    <action android:name="com.example.My Application.LAUNCH" />
    <category android:name="android.intent.category.DEFAULT" />
    <data android:scheme="http" />
  </intent-filter>

</activity>
```

9. Explain Android Debug Bridge (ADB)

Android Debug Bridge (adb) is a versatile command-line tool that lets you communicate with a device. The adb command facilitates a variety of device actions, such as installing and debugging apps. adb provides access to a Unix shell that you can use to run a variety of commands on a device. It is a client-server program that includes three components:

- **A client**, which sends commands. The client runs on your development machine.
  You can invoke a client from a command-line terminal by issuing an adb command.
- **A daemon (adbd)**, which runs commands on a device. The daemon runs as a background process on each device.
- **A server**, which manages communication between the client and the daemon. The server runs as a background process on your development machine.