https://www.interviewbit.com/android-interview-questions/

Android Interview Questions For Freshers

1. What is Android?

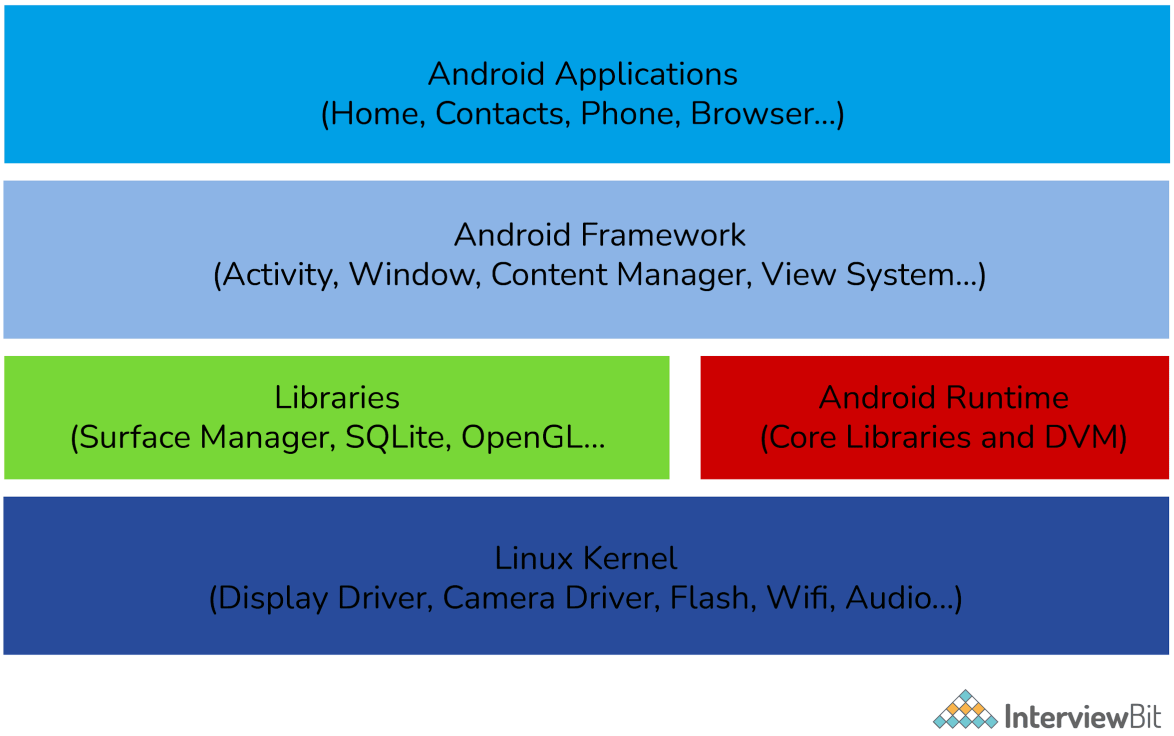
Android is an open-sourced operating system that is used on mobile devices, such as mobiles and tablets. The Android application executes within its own process and its own instance of Dalvik Virtual Machine(DVM) or Android RunTime(ART).

2. What are the features of Android architecture?

Android architecture refers to the various layers in the Android stack. It consists of operating systems, middleware, and applications. Each layer in the Android architecture gives different services to the layer just above it.

The five layers present in the Android stack are:

- Linux Kernel - It is responsible for device drivers, device management, memory management, power management, and resource access.
- Libraries - There are a set of libraries having open-source Web browser engine WebKit, well-known library libc, libraries to play and record audio and video, SQLite database for sharing of application data and storage, SSL libraries for internet security, etc.
- Android Runtime - There are core libraries along with DVM (Dalvik Virtual Machine) or ART(Android RunTime) as runtime which is helpful for running an Android application. DVM is optimized for mobile devices. DVM provides fast performance and consumes less memory. Replacing DVM, ART(Android RunTime) virtual machine was introduced to execute android apps from Android lollipop 5.0 version (API level 21).
- Android Framework - It consists of Android APIs like UI (User Interface), resources, content providers (data), locations, telephony, and package managers. It provides interfaces and classes for the development of Android applications.
- Android Applications - Applications like home, games, contacts, settings, browsers, etc. uses the Android framework that will make use of Android runtime and libraries.



Android Architecture

3. List the languages used to build Android.

The most popular programming languages that can be used to develop applications in Android are:

- Java: It has always been a starting point for new developers and used by the majority of people who work with Android development. Eclipse, NetBeans, and IntelliJ IDE are the most popular IDE's(Integrated Development Environment) used for developing an Android application using java.
- Kotlin: Kotlin is a relatively new, modern, safe, and object-oriented cross-platform programming language used in developing an Android application. IDE's used with kotlin are Android studio, Eclipse IDE, etc.
- C#: Developers can build native iOS and Android mobile applications by using the C# language. Visual Studio is the best tool for developing an Android application using C#.

- Python: It is a dynamic and object-oriented programming language. It is very popular in machine learning. Pydroid 3, Dcoder, spck code editor is some of the code editors for Python.
- Other languages which can be used in Android development are C++, HTML 5. C4droid, CppDroid, AIDE, etc. are IDE's for C++. Acode, spck code editor, etc. are examples of IDE's used with HTML.

4. What is an activity?

Activity in java is a single screen that represents GUI(Graphical User Interface) with which users can interact in order to do something like dial the phone, view email, etc.

For example, the Facebook start page where you enter your email/phone number and password to log in acts as an activity.

5. What is a service in Android?

Service is an application component that facilitates an application to run in the background in order to perform long-running operations without user interaction. A service can run continuously in the background even if the application is closed or even after the user switches to another application.

6. Differentiate Activities from Services.

Activities can be terminated or closed anytime the user wishes. On the other hand, services are designed to run in the background, and they can act independently.

Most of the services run continuously, irrespective of whether there are certain or no activities being executed.

Activities	Services
They are designed to run in the foreground.	These are mainly designed to run in the background. Foreground services are also available.
Used when the user interface is necessary.	Used when the user interface is not necessary.
They are dependent.	They act independently.

7. What is Google Android SDK? Which are the tools placed in Android SDK?

The Google Android SDK is a toolset used by developers to write applications on Android-enabled devices.

The tools placed in Android SDK are given below:

- Android Emulator - Android Emulator is a software application that simulates Android devices on your computer so that you can test the application on a variety of devices and Android API levels without having each physical device.
- DDMS(Dalvik Debug Monitoring Services) - It is a debugging tool from the Android software development kit (SDK) which provides services like message formation, call spoofing, capturing screenshots, etc.
- ADB(Android Debug Bridge) - It is a command-line tool used to allow and control communication with the emulator instance.
- AAPT(Android Asset Packaging Tool) - It is a build tool that gives the ability to developers to view, create, and update ZIP-compatible archives (zip, jar, and apk).

8. What is the use of Bundle in Android?

Bundles are used to pass the required data between various Android activities. These are like HashMap that can take trivial data types. Below code shows how to transfer a piece of data by using bundle:

```
Bundle b=new Bundle();
b.putString("Email","abc@xyz.com");
i.putExtras(b); // where i is intent
```

9. What is an Adapter in Android?

An adapter in Android acts as a bridge between an AdapterView and the underlying data for that view. The adapter holds the data and sends the data to the adapter view, the view can take the data from the adapter view and shows the data on different views like a spinner, list view, grid view, etc.

10. What is AAPT?

AAPT stands for Android Asset Packaging Tool. It is a build tool that gives the ability to developers to view, create, and update ZIP-compatible archives (zip, jar, and apk). It parses, indexes, and compiles the resources into a binary format that is optimized for the platform of Android.

11. What is portable Wi-Fi hotspot?

Portable Wi-Fi Hotspot permits you to share your mobile internet connection with other wireless devices. For example, using your Android phone as a Wi-Fi hotspot, you can use your laptop to connect to the internet using that access point.

12. What is Android Debug Bridge(ADB)?

Android Debug Bridge is a command-line tool used to allow and control communication with an emulator instance. It gives the power for developers to execute remote shell commands to run applications on an emulator.

13. What is DDMS?

DDMS(Dalvik Debug Monitor Server) is a debugging tool in the Android platform. It gives the following list of debugging features:

- Port forwarding services.
- Thread and heap information.
- Logcat.
- Screen capture on the device.
- Network traffic tracking.
- Incoming call and SMS spoofing.
- Location data spoofing.

14. What is AIDL? Which data types are supported by AIDL?

AIDL(Android Interface Definition Language) is a tool that handles the interface requirements between a client and a service for interprocess communication(IPC) to communicate at the same level.

The process involves dividing an object into primitives that are understood by the Android operating system. Data Types supported by AIDL is as follows:

- String
- List
- Map
- CharSequence
- Java data types (int, long, char, and boolean)

https://anywhere.epam.com/en/blog/25-essential-android-developer-interview-questions-with-answers

Android coding interview questions

A coding task is a mandatory part of the interview for every software engineering level. Usually, it is an algorithmic or practical task that shouldn't take longer than 10-15 minutes to solve. Technical interviewers try to avoid asking questions like "What will be printed?" since they are looking for engineers, not compilers.

In most cases, you won't need to follow all the syntactic aspects of your chosen language since there will be no IDE during the interview; if allowed, you can even resort to pseudo-code.

Android Intermediate Questions

1. What is the life cycle of Android activity?

- OnCreate(): It is called when activity is created. Using this, the views are created and data is collected from bundles.
- OnStart(): It is called if the activity is becoming visible to the user. It may be succeeded by onResume() if the activity comes to the foreground, or onStop() if it becomes hidden.
- OnResume(): It is called when the activity will start an interaction with the user.
- OnPause(): This is called when the activity is moving to the background but hasn't been killed yet.
- OnStop(): This is called when an activity is no longer visible to the user.
- OnDestroy(): This is called when the activity is finished or destroyed.

- OnRestart(): This is called after the activity has been stopped, prior to it being started again.

2. Explain Sensors in Android.

Android-based devices have a collection of built-in sensors in them, which measure certain parameters like motion, orientation, and many more through their high accuracy. The sensors can be both hardware and software based on nature. There are three prominent categories of sensors in Android devices. They are:

- Position Sensor: It is used for measuring the physical position of the Android device. This has orientation sensors and magnetometers.
- Motion Sensors: These sensors consist of gravity, rotational activity, and acceleration sensors which measure the rotation of the device or the acceleration, etc.
- Environmental Sensor: It includes sensors that measure temperature, humidity, pressure, and other environmental factors.

3. Explain the dialog boxes supported on Android.

Android supports four dialog boxes. They are:

- AlertDialog:
    - The AlertDialog supports 0-3 buttons, along with a list of selectable items such as checkboxes and radio buttons.
    - It is used when you want to ask the user about taking a decision between yes or no in response to any particular action taken by the user, by remaining in the same activity and without changing the screen.
- DatePickerDialog:
    - It is used for selecting the date by the user.
- TimePickerDialog:
    - Used for selecting the time by the user.
- ProgressDialog:
    - It is an extension of the AlertDialog and is used to display a progress bar. It also supports the addition of buttons.

- This class was deprecated in API level 26 because it prevents the user from interacting with the application. Instead of this class, we can use a progress indicator such as ProgressBar, which can be embedded in the user interface of your application.

4. What is AndroidManifest.xml file and why do you need this?

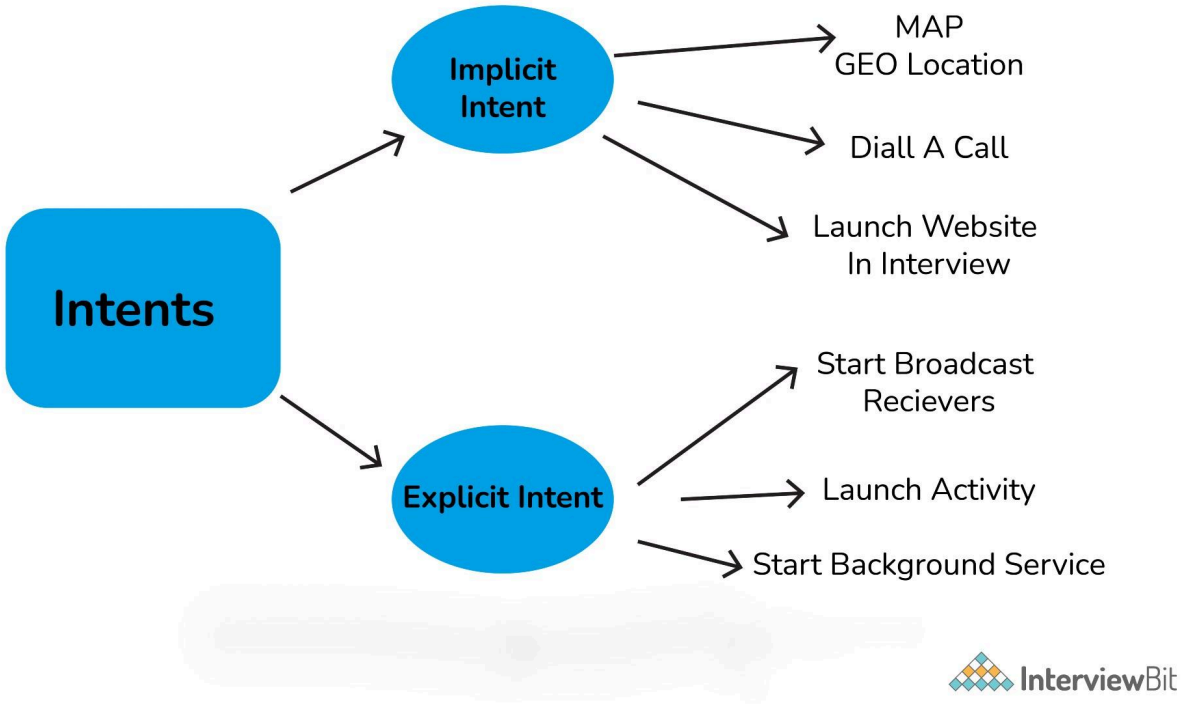
- The AndroidManifest.xml file contains information regarding the application that the Android system must know before the codes can be executed.
- This file is essential in every Android application.
- It is declared in the root directory.
- This file performs several tasks such as:
  - Providing a unique name to the java package.
  - Describing various components of the application such as activity, services, and many more.
  - Defining the classes which will implement these components.

5. What is an intent?

An intent is a messaging object that is used to request an action from other components of an application. It can also be used to launch an activity, send SMS, send an email, display a web page, etc.

It shows notification messages to the user from within an Android-enabled device. It alerts the user of a particular state that occurred. There are two types of intents in Android:

- Implicit Intent- Used to invoke the system components.
- Explicit Intent- Used to invoke the activity class.



Types Of Intents

6. Mention the difference between class, file and activity in Android?

The difference between them is as follows:

- Class is a compiled form of a .java file that Android uses to produce an executable .apk file.
- A file is a block of arbitrary information or resources used for storing information. It can be of any file type.
- Activity is a single screen that represents GUI(Graphical User Interface) with which users can interact in order to do something like dial the phone, view email, etc.

7. What is a Toast? Write its syntax.

Toast is a message that pops up on the screen. It is used to display the message regarding the status of the operation initiated by the user and covers only the expanse of space required for the message while the user's recent activity remains visible and interactive.

Toast notification automatically fades in and out and it does not accept interaction events.

Syntax:

```
Toast.makeText(ProjectActivity.this, "Your message here", Toast.LENGTH_LONG).show();
```

8. What is context?

The context in Android is the context of the current state of the application or object. The context comes with services like giving access to databases and preferences, resolving resources, and more.

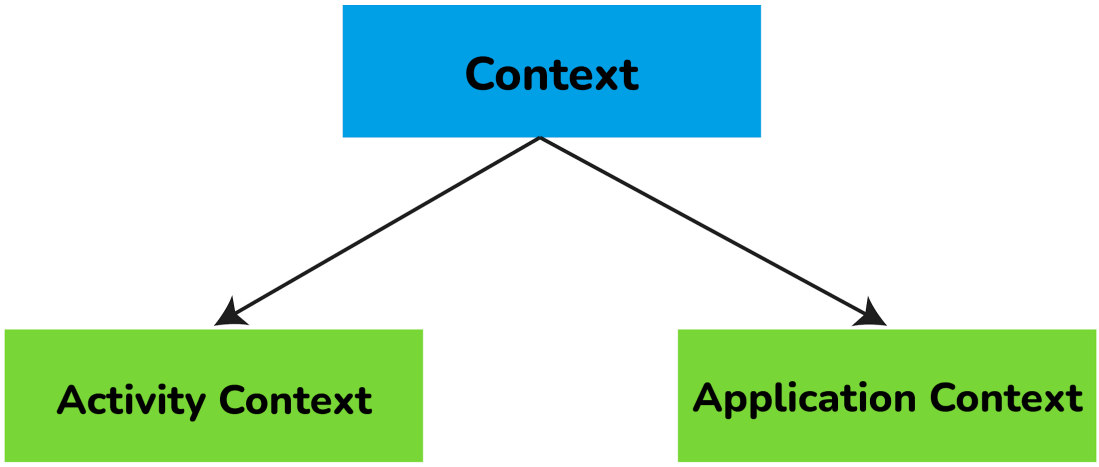
There are two types of context. They are:

Activity context

- This activity context is attached to the lifecycle of an activity.
- The activity context can be used when you are passing the context in the scope of an activity or you need the context whose lifecycle is attached to the context of the activity.

Application context:

- This application context is attached to the lifecycle of an application.
- The application context should be used where you need a context whose lifecycle is separate from the current context or when you are passing a context beyond the scope of activity.



InterviewBit

Types of Context

9. Explain the difference between Implicit and Explicit Intent.

The difference between the implicit and explicit Intents are given below:

Explicit Intent:

An Explicit Intent is where you inform the system about which activity should handle this intent. Here target component is defined directly in the intent.

For example,

```
Intent i = new Intent(this, Activitytwo.class); #ActivityTwo is the target component
i.putExtra("Value1","This is ActivityTwo");
i.putExtra("Value2","This Value two for ActivityTwo");
startactivity(i);
```

Implicit Intent:

An Implicit Intent permits you to declare the action you want to carry out. Further, the Android system will check which components are registered to handle that specific action based on intent data. Here target component is not defined in the intent.

For example,

```
Intent i = new Intent(ACTION_VIEW,Uri.parse("http://www.interview bit.com"));
startActivity(i);
```

10. What is ANR in Android? What are the measures you can take to avoid ANR?

ANR(Application is Not Responding) is a dialog box that appears when the application is not responding. This ANR dialogue is displayed whenever the main thread within an application has been unresponsive for a long time under the following conditions:

- When there is no response to an input event even after 5 seconds.
- When a broadcast receiver has not completed its execution within 10 seconds.

Following measures can be taken to avoid ANR:

- An application should perform lengthy database or networking operations in separate threads to avoid ANR.
- For background task-intensive applications, you can lessen pressure from the UI thread by using the IntentService.

11. What are the troubleshooting techniques you can follow if an application is crashing frequently?

If an Android application is crashing frequently, you can follow the below-given techniques:

Compatibility Check:

It is not possible to test an application for all kinds of devices and operating systems. There might be a possibility that an application is not compatible with your OS.

Memory Management:

- Some apps run perfectly on one mobile device but might crash on other devices. This is where processing power, memory management, and CPU speed are considered.
- As there is a limited amount of memory space on mobile devices, you can free up memory space for the application to function properly.
- If an application is frequently crashing, you can delete the application's data, which will clear its cache memory and allow some free space on your device and might boost the app's performance.

12. Explain different launch modes in Android.

The different launch modes in Android are given below:

Standard:

- This launch mode generates an activity's new instance in the task from which it originated.
- It is possible to create several instances for the same activity.
- For Example, suppose our current stack is A -> B -> C. Now, if we launch activity B again with the "standard" launch mode, then the new stack will be A -> B -> C -> B.

SingleTop:

- This launch mode is similar to the Standard launch mode except if there exists an activity's previous instance on the top of the stack, then a new instance will not be created.
- But the intent will be sent to the activity's existing instance.
- For example, suppose our current stack is A -> B -> C. Now, if we launch the activity B again with "singleTop" launch mode,then the new stack will be A -> B -> C -> B.
- Consider another example, where the current stack is A -> B -> C. Now, if we launch activity C again with the "singleTop" launch mode, then the stack will remain the same i.e., A -> B -> C. The intent will be passed to the onNewIntent() method.

SingleTask:

- This launch mode will create a new task and push a new instance to the task as the root.
- For example, suppose our current stack is A -> B -> C -> D. Now, if we launch activity B again with the “singleTask” launch mode, then the new stack will be A -> B. Here, a callback has been received on the old instance and C and D activities are destroyed.

SingleInstance:

- This launch mode is similar to the SingleTask launch mode. But the system doesn’t support launching any new activities in the same task.
- In a situation where the new activity is launched, it is launched in a separate task.
- For example, Suppose our current stack is A -> B -> C. Now, if we launch the activity D with the “singleInstance” launch mode, then there will be two stacks:
    - A -> B -> C
    - D, If you call activity E, then it will be added to the first stack.
    - A -> B -> C -> E
    - D

Again if you Call the activity D, then it will call the same activity from the 2nd stack and pass the intent to onNewIntent().

13. What are containers?

Containers carry objects and widgets together, based on which specific items are required and in what particular arrangement is needed. Containers may hold labels, buttons, fields, or even child containers, etc. For example, if you want a form with fields on the left and labels on the right, you will need a container. If you want the OK and Cancel buttons to be below the rest of the form, next to one another, and flush to the right side of the screen, you will need a container. If you have several widgets, you will need a container to have a root element to place the widgets inside.

Android provides a collection of view classes that serve as containers for views. These container classes are called layouts, which are defined in the form of XML files that cannot be changed by

our code during execution. The layout managers provided by Android SDK are LinearLayout, RelativeLayout, FrameLayout, AbsoluteLayout, GridLayout, and TableLayout.

14. What is the role of Dalvik in Android development?

Dalvik serves as a virtual machine, and it is responsible for running every Android application. Because of Dalvik, a device will have the ability to execute multiple instances of virtual machines efficiently through better memory management.

15. What is the latest version of Android? List all the versions of Android.

The latest version is Android 11.0 released in September 2020. Most of the Android version has been named after either sweet or desserts. The following table represents the different versions, Android name, API level, and the year of release. Here API level is the integer value that identifies the API framework revision given by the version of the android platform.

Version	Android Name	API Level	Year of Release
Android 1.0	No Codename	1	2008
1.1	No Codename	2	2009
1.5	Cupcake	3	2009
1.6	Donut	4	2009
2.0 - 2.1	Eclair	5 - 7	2009
2.2 – 2.2.3	Froyo	8	2010
2.3 – 2.3.7	Gingerbread	9 - 10	2010
3.0 – 3.2.6	Honeycomb	11 - 13	2011

4.0 – 4.0.4	Ice Cream Sandwich	14 - 15	2011
4.1 – 4.3.1	Jelly Bean	16 - 18	2012
4.4 – 4.4.4	Kitkat	19 - 20	2013
5.0 – 5.1.1	Lollipop	21 - 22	2014
6.0 – 6.0.1	Marshmallow	23	2015
7.0 – 7.1.2	Nougat	24 - 25	2016
8.0 – 8.1	Oreo	26 - 27	2017
9.0	Pie	28	2018
10.0	Android 10	29	2019
11.0	Android 11	30	2020

Android 1.0 (API 1): There were numerous loopholes in this Android version including the necessity of physical keyboards or hardware buttons. Important features are:

- Browser, Google maps, and calendar.
- Camera and scroll down the notification bar.
- Wireless supports Wi-Fi and BlueTooth.
- Contacts, Gmail integration, and Google synchronization.

Android 1.1 (API 2): This version was released with API changes and resolved issues found in 1.0. Important features of this version are:

- Display details as well as reviews for locations.
- Gives detailed information by clicking on the business.
- Add save an attachment in the message.

Android 1.5 Cupcake (API 3): It brought the third-party app widgets that were the most distinguishing and valuable feature. Important features of this version are:

- Video recordings, copy and paste facility.
- Supports MPEG4 and 3GP formats.
- Search function and on-screen keyboard.
- Uploading images and videos.

Android 1.6 Donut (API 4): It had valuable changes including the ability to operate on various screen resolutions and sizes. Donut provides the text and voice entry search including bookmark history and capacity to select multiple images for deletion. Important features of this version are:

- Camera quick toggling features and gallery.
- WVGA screen resolution speed.
- Power control widget for handling Wi-Fi, GPS, BlueTooth, etc.
- Technology support for CDMA/EVDO, VPNs, 802.1x, and a text-to-speech engine.
- Searching applications and speed improvements for cameras.
- Supports quick search Box.

Android 2.0-2.1 Eclair (API 5-API 7): It was released with multiple email account synchronizations and contacts. Many new features were added including flash support, color effect, scene mode, white balance, macro focus, and digital zoom. Android 2.1 version brought bug fixes and stability improvements. The most valuable feature introduced was real-time traffic information and voice-guided turn by turn navigation. Important features are:

- Updated UI.
- Minor API changes.
- Support Bluetooth 2.1.
- Improved Google map.
- Supports live and animated Wallpapers.
- Has the ability to add contact's photo and select to call, message or email.

Android 2.2-2.2.3 Froyo (API 8): It was released with memory, speed, and performance improvements. It was introduced with the enhanced Bluetooth functionality and also compatibility with docks, portable Wi-Fi hotspot for 3G connection sharing. Features are:

- Upload file support in the browser.

- Supports alphanumeric and numeric passwords to enhance security.
- Support for animated GIF and multiple keyboard languages.
- Increased compatibility with headsets and car kits.

Android 2.3-2.3.7 Gingerbread (API 9-API 10): The main enhanced feature was the introduction of gaming API with improved graphical intense gaming. Features are:

- Updated UI.
- Support for VP8 and WebM video format.
- Improvement in copy and paste facility.
- Easier use of keyboard with faster and intuitive typing.
- Supports video calling and social networking.

Android 3.0-3.2.6 Honeycomb (API 11-API 13): It was a tablet-only release to launch the Motorola Xoom. It was suitable for those mobiles having a larger view than current smartphones. Features will include:

- Updated 3D UI and encrypted storage.
- Gmail, camera, contacts, and gallery improvements.
- Supports passwords with complex characters.
- Supports multiprocessors and recent applications for easy visual multitasking.
- Media sync from SD card.
- Talk video chat and google eBooks.
- Support adobe flashes in the browser.
- More sensor support.
- High-performance Wi-Fi connections.
- Action bar for application control and system bar for global status and notifications
- Chinese handwriting and redesigned keyboard.

Android 4.0-4.0.4 Ice Cream Sandwich (API 14-API 15): It is released with many enhanced features to enter the era of modern design. The snapshot was introduced to take screenshots by holding the power and volume button. Ice cream sandwich widgets are more robust and resizable compared to all older versions. Supported features are:

- Unlocking with face-fixing.
- Card-like appearance for app-switching.
- Better camera performance and improved video recording with high resolution.
- Spelling check feature, Wi-Fi direct.
- On-screen buttons and better camera performance.
- Ability to open up to 16 tabs in the web browser.

Android 4.1-4.3.1 Jelly Bean (API 16-API 18): It is released with Google digital assistant technology accessible from the home screen. The spectacular predictive intelligence utility gives expandable and interactive notifications. Users can enjoy multi-user support(Only for tablets). Supported features are:

- Power control, support USB audio.
- The improved camera app, security improvements.
- Voice search and typing, panorama.
- Expandable notifications, daydream as a screensaver.
- Google displays relevant content based on search history.
- Native emoji support, new gestures, and accessibility features
- 4k resolution support, supporting Bluetooth with low energy.
- Bi-directional text and different language support.
- Set or adjust the volume of incoming calls and show a message alert as well.

Android 4.4-4.4.4 KitKat (API 19-API 20): It is released with more focus on better user experience. Supported features are:

- GPS Support, smarter caller ID.
- Offline music support, screen recording.
- Contact prioritization.
- Cartoonish ideograms and emojis to the Google keyboard.
- UI updates for Google map navigations and alarm.
- ‘OK, Google’ feature allows access to Google to the users without touching your smartphones.

Android 5.0-5.1.1 Lollipop (API 21-API 22): It was released with a redesigned user interface and built with “material design”. It is having many amazing features including support for better notification management. Supported features are:

- Support ART, better and improved UI.
- Better device protection, built-in battery saver feature.
- Notification can be flicked away from the lock screen.
- The revamped navigation bar supports multiple SIM cards.
- The high definition of a voice call.

Android 6.0-6.0.1 Marshmallow (API 23): Google used “Macadamia Nut Cookie” to describe Android version 6.0 before the official announcement of Marshmallow. Supported features are:

- Support for fingerprint readers, type-C USB support.
- Multi-window experience, clear permission system.
- ’Sleep Mode’ for saving the life of the battery.
- Custom Google tabs and improved copy-pasting.

Android 7.0-7.1.2 Nougat (API 24-API 25): It was released with a native split-screen mode, data saver functionality, and a “bundled-by-app” system to organize notifications. Supported features are:

- Multitasking and split-screen mode.
- Storage manager enhancements, quick setting toggles.
- Display touch enhancements, better setting application.
- Inline reply to message and notification without opening an application.

Android 8.0-8.1 Oreo (API 26-API 27): It is having notification snoozing options, native picture-in-picture mode, and better control over how applications can alert you by notifications. Supported features are:

- Password autofill, auto-enable Wi-Fi.
- Downloadable fonts, multi-display support.

- Support picture-in-picture.
- Adaptive icons and smart text selection.
- Notification channels and snooze notifications.
- Google play support and new emoji styling.

Android 9.0 Pie (API 28): It is having plenty of amazing features according to the user interests and requirements. Supported features are:

- HDR, HD audio, multiple Bluetooth connections.
- Much more information about notification and easier text selection.
- Sound amplifier with select to speak option.
- Artificial Intelligence(AI) compatibility.
- Adaptive battery and brightness with the background restrictions.
- Supports for multi-camera with external camera compatibility.
- New shortcut key for screenshot and accessibility menu.
- Improved security features for protection.
- Android backups, privacy enhancements.
- Easier screen rotation, edge-to-edge screens support.
- Volume and sound enhancements, selectable dark mode.

Android 10.0 (API 29): It is released with enhanced features and functionalities with higher API levels. Supported features are:

- Support for foldable smartphones with a flexible display.
- Dark mode for the comfortability of eyes.
- Navigation control over gesture quicker and intuitive ever.
- Smart reply suggestions for all messaging applications.
- Live caption for media playing on a smartphone device.
- Undo removal of the app.
- Better notification control with many options.
- Sound amplifier with much more clear sound.

Android 11.0 (API 30): Android developers are continuously working to provide more advanced applications as per the user requirements. Important features included are:

- Native screen recording.
- Auto revokes app permission.
- Mute notifications during the video.
- Increase in touch sensitivity.
- Notification history.

Android Interview Questions For Experienced

1. What are broadcast receivers? How is it implemented?

A broadcast receiver is a mechanism used for listening to system-level events like listening for incoming calls, SMS, etc. by the host application. It is implemented as a subclass of BroadcastReceiver class and each message is broadcasted as an intent object.

```
public class MyReceiver extends BroadcastReceiver
{
   public void onReceive(context,intent){}
}
```

2. Explain in detail about the important file and folders used when you create a new Android application.

App:

It describes the basic characteristics of the application and defines each of its components.

java:

- This contains the .java source files and .kt(source code written in Kotlin) source files of your project. By default, it includes a MainActivity.java or MainActivity.kt source file.
- You create all the activities which have .java and .kt extensions under this file and also it includes all the code behind the application.

res:

- It is used to store the values for the resources that are used in various Android projects to include features of color, styles, dimensions, etc.
- It is a directory for files like styles.xml, strings.xml, colors.xml, dimens.xml, etc.

Scripts:

This is an auto-generated file that consists of compileSdkVersion, buildToolsVersion, minSdkVersion, targetSdkVersion, applicationId, versionCode, and versionName. For example, build.gradle is a script file placed in the root project directory, defines build configurations that will be applied to all modules in your project.

3. What is the difference between Serializable and Parcelable? Which is the best approach in Android?

While developing applications usually it needs to transfer data from one activity to another. This data needs to be added into a corresponding intent object. Some additional actions are required to make the data suitable for transfer. For doing that the object should be either serializable or parcelable.

Serializable:

- Serializable is a standard Java interface. In this approach, you simply mark a class Serializable by implementing the interface and java will automatically serialize it.
- Reflection is used during the process and many additional objects are created. This leads to plenty of garbage collection and poor performance.

Parcelable:

- Parcelable is an Android-specific interface. In this approach, you implement the serialization yourself.
- Reflection is not used during this process and hence no garbage is created.

- Parcelable is far more efficient than Serializable since it gets around some problems with the default Java serialization scheme. Also, it is faster because it is optimized for usage on the development of Android, and shows better results.

4. What database is used in Android? How it is different from client-server database management systems?

SQLite is the open-source relational database used in Android. The SQLite engine is serverless, transactional, and also self-contained. Instead of the client-server relationship of most database management systems, the SQLite engine is integrally linked with the application. The library can be called dynamically and it can make use of simple function calls that reduce latency in database access.

5. What are the differences between Service and Thread?

The main difference between Service and Thread is given below:

Service	**Thread**
Service is an application component that facilitates an application to run in the background in order to perform long-running operations without user interaction.	A Thread is a concurrent unit of execution.

It exposes few functionalities to other applications by calling Context.bindService().	Google has brought in handlers and loopers into threads.
When an application is killed, service is not killed.	When an application is killed, the thread is killed.

6. What is the content provider? How it is implemented?

Content provider is one of the primary building blocks of Android applications, which manages access to a central repository of data. It acts as a standard interface that connects data in one process with code running in another process. So it can be used to share the data between different applications.

They are responsible for encapsulating the data and providing mechanisms for defining data security. It is implemented as a subclass of ContentProviderclass and must implement a set of APIs that will enable other applications to perform transactions.

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

7. What is the significance of the .dex file?

Android programs are compiled into a .dex file (Dalvik Executable file) by DVM, which are then zipped into a .apk file on the device. .dex files are created by translating compiled applications written in java. .dex is a format that is optimized for effective storage and memory-mappable executions.

8. What is the difference between compileSdkVersion and targetSdkVersion?

compileSdkVersion:

- The compileSdkVersion is the version of API the application is compiled against. You can use Android API features involved in that version of the API (as well as all previous versions).
- For example, if you try and use API 15 features but set compileSdkVersion to 14, you will get a compilation error. If you set compileSdkVersion to 15 you can still run the app on an API 14 device as long as your app's execution paths do not attempt to invoke any APIs specific to API 15.

targetSdkVersion:

- The targetSdkVersion indicates that you have tested your app on (presumably up to and including) the version you specify. This is like a certification or sign-off you are giving the Android OS as a hint to how it should handle your application in terms of OS features.
- For example, setting the targetSdkVersion value to "11" or higher permits the system to apply a new default theme (Holo) to the application when running on Android 3.0 or higher. It also disables screen compatibility mode when running on larger screens (because support for API level 11 implicitly supports larger screens).

9. Explain about java classes related to the use of sensors on Android.

Android sensor API provides many classes and interface for the use of sensors on Android. The important classes and interfaces of sensor API are given below:

- Sensor class: This class helps you to create an instance of a specific sensor. It provides methods that let you determine a sensor's capabilities.
- SensorManager class: This class is used to create an instance of the sensor service. It provides methods to access and list sensors, to register and unregister sensor listeners, etc.
- SensorEvent class: This Java class is used to create a sensor event object. It provides information about the sensor event including raw sensor data, the accuracy of data, type of sensor, timestamp of event, etc.
- SensorEventListener interface: This interface is used to create two callback methods that receive sensor event notifications when sensor value changes or when sensor accuracy changes. Those two methods are void onAccuracyChanged(Sensor sensor, int accuracy) which is called when sensor accuracy is changed and
  void onSensorChanged(SensorEvent event) which is called when sensor values are changed.

10. What is JobScheduler?

The JobSchedular API is used for scheduling different types of jobs against the framework that will be executed in your app's own process. This allows your application to perform the given task while being considerate of the device's battery at the cost of timing control.

The JobScheduler supports batch scheduling of jobs. The Android system can combine jobs for reducing battery consumption. JobManager automatically handles the network unreliability so it makes handling uploads easier.

Here is some example of a situation where you would use this job scheduler:

- Tasks that should be done when the device is connected to a power supply.
- Tasks that require a Wi-Fi connection or network access.
- Tasks should run on a regular basis as a batch where the timing is not critical

1. What are the commonly used languages in Android?

This is one of the more simple interview questions for Android developers, so you should be ready with a list of known languages such as Java, Kotlin, C#, Python, and Groovy. For extra points, you can list some Integrated Development Environments (IDEs) like Eclipse, NetBeans, IntelliJ, and Android Studio.

3. What are the main components of the Android application framework?

The Android application framework provides the underlying libraries that allow developers to use device resources for application creation. For an interview question, be ready to explain some classes such as activity, service, broadcast receiver, and content provider.

4. What are the benefits and disadvantages of Kotlin?

Java is established and widespread, but Kotlin continues to grow in popularity. If asked, use this query to showcase your knowledge and abilities with Kotlin, explaining how it can eliminate NullPointerException (NPE) from code, has cross-platform app development functions, and often has better readability.

5. How are permissions managed in Android?

Android permissions control how users access sensitive data held within the applications. As an Android developer, it is your job to limit information access while maintaining functionality. List some best practices for Android permissions, such as

setting default handlers or selecting third-party SDKs that also have limited permissions.

6. What are the launch modes available for the activity in Android?

Launch modes help you launch activities with specific sets of instructions and attendant navigation requirements. There are four launch modes that you should reference if asked this interview question:

- Standard
- SingleTop
- SingleTask
- SingleInstance

Each launch mode has its own role to play in allowing navigation across any possible configuration, so compare and contrast the benefits and drawbacks to achieve full marks.

7. What is the difference between fragment and activity?

The code for an activity is far more in-depth than for a fragment. Be sure to mention to your interviewer that due to the extended work, activities should only be used for swapping entire screens, while fragments are used for everything else. If possible, list some use cases in which you almost always use a fragment rather than an activity (e.g. when you use components or data that must persist across varied activities).

8. What are the examples of design patterns used in Android applications?

Design patterns are reusable templates that solve problems (think repetitive code). When presented with this question in an interview, be sure to outline the common Android pattern categories (Creational, Structural, and Behavioral) and give some use cases. Earn some bonus points by explaining how you resolved a problem from a previous project with a design pattern.

9. What types of Android app architectural patterns are there?

Architecture patterns are applied to structure a project and make it modular. There are three main Android architectural patterns that developers use and that you should mention:

- MVC (Model — View — Controller)
- MVP (Model — View — Presenter)
- MVVM (Model — View — ViewModel)

Since each architectural pattern involves a wealth of information, consider this a more broad coding question. Offer detail when possible, but use this question to show your extended grasp of Android programming.

10. What additional libraries do you use in Android app development?

Libraries let you take advantage of the work of other developers for better performance as you create apps. There are several popular libraries that you can talk

about (Glide, Retrofit, Picasso, Dagger, Koin), but take the opportunity to explain the library you prefer or have used in the past with great success.

11. What is PendingIntent in Android?

A PendingIntent passes a future intent to a different application that will perform execution. PendingIntents are a crucial part of the Android framework because they can specify actions to another app. It is good if your answer to this Android coding interview question touches upon implementation details and the new changes within Android 12 regarding mutability.

12. What is a content provider in Android?

A content provider controls access to an app's data. Other applications will use a provider client object, creating an interface to handle communication. As a programming question, be ready to explain how you would access data from a content provider in addition to supplying information on creating a content provider within your own app. Since accessing data can block a main thread, note how you would perform data queries on a separate thread.

13. What is meant by ANR?

As an intermediate interview question, use this opportunity to explain what an Application is Not Responding (ANR) is as well as the steps you could take to avoid an ANR. Mention both trigger conditions (a response to an input, a broadcast receiver does not execute in a reasonable time) and common patterns used for diagnosing ANRs (long calculations occurring on the main thread, synchronous binder calls with another process are taking a long time to return).

14. What kind of services are available on Android?

A service is an application component that does not need a UI to work (think of a music service playing music or downloading files). Services are mechanisms for background work. In your interview, make sure you discuss the three types of services:

> ForegroundService: Operations a user can recognize
> BackgroundService: Operations unnoticed by the user
> BoundService: Bound services that provide a client-server interface for component interactions

The more details you can provide the better, so including information about threads vs services, intent services, or how a service starts, can all help your case.

15. Explain flavors in Android

With this question, the interviewer wants to know if you can differentiate between product flavors and build types, as they are often mixed up. Also, convey how developer customization to apps across different codebases is tedious and causes errors, so using product flavors is ideal since you can create app variants with a single code base. Mention these details in addition to explaining Gradle, as it creates build type variants.

16. How do you handle a long-running process in an Android app?

Processes are defined as long-running by Android if they take longer than 10 minutes to complete (such as large uploads and downloads or user-specified tasks). In such

cases, you must create a long-running worker with WorkManager (assuming you can't chunk workloads).

17. How do you protect an Android app from reverse engineering?

Android application development interview questions will focus on the latest trends and topics. That will include discussions about security and data privacy. With the popularity and open-sourced nature of Android, you need to offer direct methods for protecting user data from reverse engineering. A business always wants to mitigate the negative impacts of security hacks or breaches, so your answers here are important.

Make sure you are well aware of prevention tactics and tools such as tamper detection, ProGuard Assistance, and SafetyNet. List other best practices such as using C++ for important code, securing user credentials, server-side database encryption, and altering Android's shared library loading process.

18. What is Android Jetpack and what are its libraries?

Android Jetpack is a suite of tools and libraries that can help you build high-quality apps. While the software offers many benefits regarding build complexity and writing boilerplate code, it also addresses several challenges such as limiting memory leaks or managing configuration changes. Jetpack offers four categories of components (Foundation, Architecture, Behavior, and UI) and a host of libraries. In the interview, list a few and explain how they helped you in the past.

19. What are the new updates and features in Android 13?

Android 13 is the latest release of the Android mobile operating system, expected to reach platform stability in June/August of 2022. Android 13 beta is already out and offers some quality-of-life upgrades and privacy and security improvements. As an Android developer, it is your job to stay near the edge of continual innovation, so having an awareness of upcoming upgrades is a crucial opportunity to demonstrate your expertise.

20. How do you publish an Android app to the Play Store?

Android job interview questions are not always the hardest, technical-based queries. But this deceptively simple question can include plenty of varied factors about configuring your app and signing release versions. Be prepared to explain some common tasks involved with app publishing.

21. What is needed to run code as coroutine?

This is a more technical Android interview question, and it pertains specifically to Kotlin coroutines. Kotlin creates fluid server-side or mobile applications by providing coroutines at the language level, leading to asynchronous programming. Display your understanding of Kotlin syntax, threads, and sequential code in Android architecture components to impress your interviewer.

22. How do you test an Android application?

As the interview progresses, you will receive more senior-level Android engineer interview questions. Advanced Android interview questions are an opportunity to showcase your advanced knowledge, and also your leadership skills and creativity. In this case, lay out some testing fundamentals to start (functional, performance, accessibility, and compatibility test types), and then shift into a discussion of your preferred testing approaches to demonstrate your problem-solving capabilities.

23. What’s the purpose of RxJava?

Different paradigms help solve different problems, and RxJava helps simplify asynchronous programming. As a top Android interview question, here is your chance to explain your understanding of programming libraries and functional reactive programming in Android. Explain how it can take complex implementation of asynchronous behaviors and simplify mobile programming.

24. What are the benefits of MVVM?

As a recommended architecture approach by Google, MVVM has become a standard in Android development. This question allows you to show an understanding of problems that MVVM solves, knowledge of other styles such as MVC/MVP, the pros and cons of each, and how to choose between them when starting a new project.

25. How do you provide dependencies between different parts of the app?

Persistent avoidance of "dependency hell" is one of the essential parts of clean code. Knowledge of inversion of the control pattern and its different implementations (ServiceLocator, Dependency injection, etc.) can help to provide robust and testable solutions. Be ready to provide possible solutions based on Android SDK and third-party libraries.