1. **Tell all the Android application components. 4 components**

Android application components include Activities, Services, Broadcast Receivers, and Content Providers. Activities represent the UI and handle user interactions. Services perform background tasks independently of UI. Broadcast Receivers respond to system-wide broadcast announcements. Content Providers manage access to a structured set of data.

1. **What is Context? How is it used?**

Context in Android refers to the current state of the application and provides access to resources, databases, preferences, etc. It is used to get information about the application environment and to create and access application components. Context is commonly used for accessing resources, launching activities, creating views, and accessing system services.

1. **What is the project structure of an Android Application?**

The project structure of an Android application typically includes folders like app, manifest, java, res, and gradle scripts. The 'app' folder contains the source code and resources for the application. The 'manifest' folder holds the AndroidManifest.xml file defining the application's components and permissions. The 'java' folder contains the Java source code files. The 'res' folder contains resources such as layouts, drawables, and strings. Gradle scripts manage the build process.

1. **What is AndroidManifest.xml?**

AndroidManifest.xml is a crucial file in an Android application that provides essential information about the app to the Android system. It declares the application's components like activities, services, broadcast receivers, and content providers. It also declares permissions required by the app, the app's hardware and software requirements, and other crucial metadata.

**5. What is the Application class?**

The Application class in Android is a base class for maintaining global application state. It's instantiated before any other class when the process for the application is created. It's typically used for tasks like initializing global variables, setting up configurations, and managing application-wide resources.

**6. Activity and Fragment**

Activities represent single screens with a user interface. They interact with users and handle user events like clicking buttons or typing text. Fragments are modular sections of an activity with their own lifecycle and UI. They are reusable components that can be combined within activities to build a multi-pane UI on large screens or for modular design.

**7. ListView vs RecyclerView**

RecyclerView was introduced as a more efficient and flexible alternative to ListView in Android. Unlike ListView, which loads all items at once, RecyclerView only displays the items currently visible on the screen, improving performance with large datasets. Its modular design separates concerns, making code more maintainable. Additionally, RecyclerView supports complex item animations, enhancing the user experience.

**8. What is Intent? Explicit & implicit - with example. What is Bundle**

An Intent is a messaging object that can be used to request an action from another app component. Explicit Intents specify the component to start by name. Implicit Intents specify the action to perform, and the system matches the intent with the appropriate component. Bundles are key-value pairs that can be passed along with intents to carry additional data.

**Example:**

1. \*\*Explicit Intent\*\*: In a messaging app, tapping on a "New Message" button to open a specific "Compose Message" screen.

2. \*\*Implicit Intent\*\*: In the same messaging app, tapping on a "Share" button to share a message with another app, like an email client or social media platform, without specifying which one.

**9. What is Broadcast Receiver?**

A Broadcast Receiver is an Android component that responds to system-wide broadcast announcements. It can be used to listen for and react to events like the completion of a long-running operation or the device entering airplane mode. Broadcast Receivers can be registered either statically in the manifest file or dynamically at runtime.

**10. What is a Service? (Types of service)**

A Service is an Android component that runs in the background to perform long-running operations independently of the UI. There are two types of services: Started services, which are explicitly started and stopped by other components, and Bound services, which provide an interface for other components to interact with by binding to it.

**11. What is ANR? How can the ANR be prevented?**

ANR (Application Not Responding) occurs when the main thread of an Android application is blocked for too long, typically more than 5 seconds. It can be prevented by moving long-running tasks to background threads, optimizing UI operations, using services for background tasks, and handling exceptions properly.

**12. What is SharedPreference in Android?**

SharedPreferences is a simple key-value pair storage mechanism in Android used for storing primitive data types persistently. It allows apps to save and retrieve small amounts of data like user preferences, settings, and application state that needs to persist across app launches.

**13. What is Contraint Layout in Android? How is it better than other layout(Linear/Relative) in android**

ConstraintLayout is a flexible layout manager in Android that allows you to create complex layouts with a flat view hierarchy. It's better than other layouts like LinearLayout or RelativeLayout because it reduces nesting of view groups, resulting in improved layout performance and easier maintenance of complex UI designs.

**14. How do you support different types of resolutions?**

Supporting different resolutions in Android involves providing alternative resources for different screen densities and sizes. This includes creating multiple versions of drawable, layout, and dimension resources targeted for different screen densities (ldpi, mdpi, hdpi, xhdpi, xxhdpi, xxxhdpi) and screen sizes (small, normal, large, xlarge).

**15. What is Android Data Binding?**

Android Data Binding is a feature that allows you to bind UI components in your layouts to data sources in your app's architecture. It simplifies the development process by reducing boilerplate code for updating UI components with data. Data Binding also helps in keeping UI components in sync with the underlying data source automatically.