Tadnya Softech Pvt. Ltd.

Android Interview Guide

We at Tadnya Softech Pvt Ltd. are looking for passionate Android developers who can work on native android applications. Following is list of technology components which we expect developer to have knowledge of. It is not compulsory to have expertise on all of the following components but it would be better to have an idea of following technology components.

We have divided components in different sections and have mentioned description and reference links related that. This will help you to be prepared on those.

1. **Android concepts**
    1. All components
       All android app components like activities, services, broadcast receivers and content providers.
    2. Architecture
       Deciding architecture of an android app based on scope of application, using intents navigation in best way possible.
    3. Styles and themes
       Following material design in complete app, using styles and themes extensively instead hard coding values in layout files
    4. Multi-layout support
       Supporting multiple layouts efficiently. All resolution devices and tablets should have same experience of App as of Mobile phones.
    5. Multi-language support
       Supporting multiple language with proper fonts.
    6. Build flavors
       Having different flavors for same app helps in keeping the code same for different environments like dev and production. Similarly providing modular feature support can be provided in single app.
    7. Debug, Release modes
       Different from flavours debug and release modes are help to differ app in development phase and release phase.
    8. Proguard code obfuscation
       One of most important part of any android app development is applying Proguard at end of app development process to avoid reverse engineering of app and reducing app size by shrinking code.

9. FCM
   FCM is used for sending out push notifications, having should knowledge of types of notifications will give idea about how to communicate to app from server.
10. Firebase Job Scheduler
    A scheduler pattern used to make background syncing enabled apps, it is developed by Firebase team as a library. Reference link: Github/JobDispatcher, Job Scheduling fsdf

2. Libraries expected

1. Retrofit
   A library for making network calls from app, it is used to make api calls in most efficient way. This is a must use library for any app making network calls for fetching data. Reference links : Retrofit, Vogella Retrofit, Android Hive Retrofit. More references can be found easily.
2. Okhttp
   A library used for making low level network calls, retrofit is internally using Okhttp itself. Reference links : OKhttp.
3. Jackson
   A json parsing library used to parse response in json format from an api call. This library parses response into POJO class directly which helps in receiving direct objects from API request. Reference links : Jackson Converter, Reference
4. RxJava
   RxJava is a Java VM implementation of ReactiveX (Reactive Extensions): a library for composing asynchronous and event-based programs by using observable sequences. Using Rxjava for making asynchronous calls. Reference links : RxJava, Retrofit+Rxjava.
5. Realm
   Realm is library for database in android, it is freshly developed database which is easy and faster than traditional SQLite databases. This is must have library in any android app currently. Reference links : Realm.io, Android Hive
6. Fabric Suite
   Fabric suit is set of tools provided by twitter which consist of analytics, crash reporting, OTP verification. Using this suite helps in reducing developers work for writing everything from scratch and also helps in getting insights about app. Reference links : Fabric.io
7. Butterknife
   A view binding library which helps developers to avoid writing boiler plate code for initialising views. Reference links : Butterknife
8. Piccasso
   Image loading library in android which helps in loading images in imageviews from network. Reference links : Piccasso

3. **Android architecture**
    1. MVP architecture
       Having architecture pattern while developing android app helps in keeping code modular and structured manner so that maintenance of code is easier. Reference links : Ref 1, Ref 2, Ref 3.
    2. Creating modules and libraries
       Creating everything under same app makes it difficult to keep different layers separately. Making libraries and modules for each part makes it easier for development and maintenance.
4. **Version Control System**
    1. Git commands
       We extensively use git for maintaining our code in repositories. Having sound knowledge of git flow is must. Reference links : Ref 1, Ref 2, Git Branching Model
    2. Git pull requests
       Creating pull request is the way to have a code review process before merging code into actual company repository. Reference links: Ref 1.
5. **Coding standards**
    1. Java coding standards
       We need to follow all the java coding standards extensively to have a cleaner code.
    2. Layout file coding standards
       Following standards for writing layout files is also necessary it increases readability of code.
    3. Style file coding standards
       Writing styles and dimensions in proper format is necessary so that uniformity is maintained across the project.
       Overall standards reference links : Ref 1, Ref 2, Ref 3, Ref 4.