INTRODUCTION TO ANDROID AND ARCHITECTURE

Prof. Devansh Parikh

TOPIC COVERED

- Android Debug bridge,
- Android Permission model,
- Android Manifest File,
- Android Project Framework

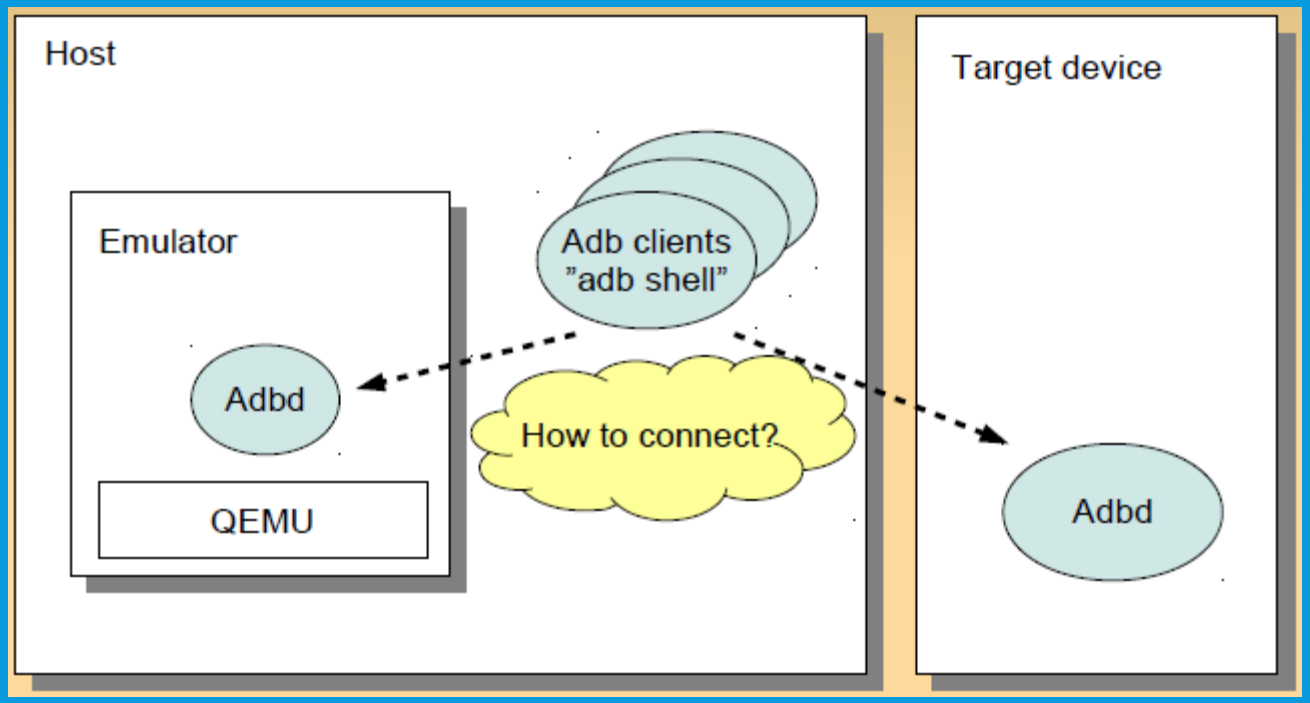
ANDROID DEBUG BRIDGE

- Android Debug Bridge (adb) is a versatile command-line tool that lets you communicate with a device. The ADB is typical, used to communicate with a smartphone, tablet, smartwatch, set-top box, or any other device that can run the Android operating system.
- We can do things on an Android device that may not be suitable for everyday use, like, install apps outside of the Play Store, debug apps, access hidden features, and bring up a UNIX shell, etc.
- For security reasons, Developer Options need to be unlocked and you need to have USB Debugging Mode enabled as well. Not only that, but you also need to authorize USB Debugging access to the specific PC that you're connected to with a USB cable.

ANDROID DEBUG BRIDGE

- 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, 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.

HOW ANDROID DEBUG BRIDGE(ADB) WORKS ?



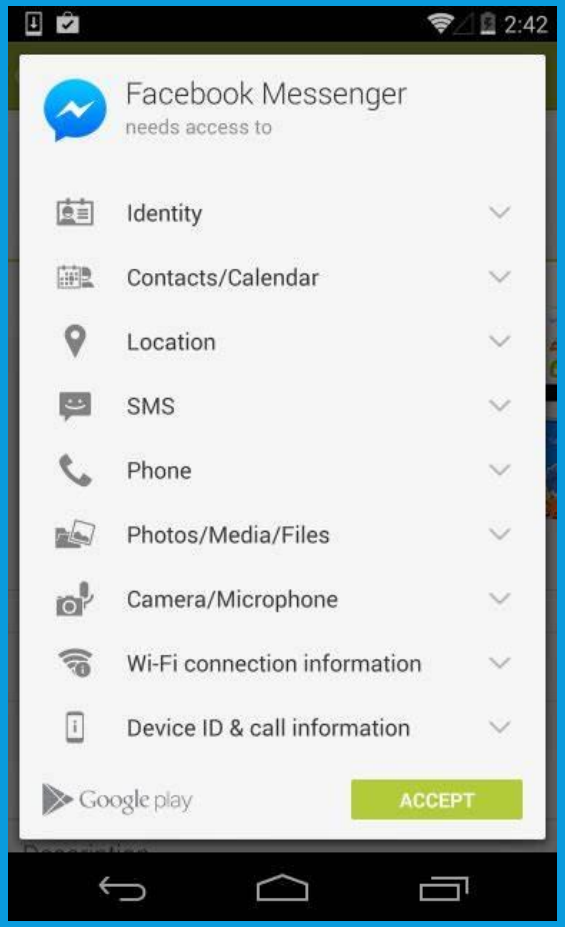
ANDROID PERMISSION MODEL

- App permissions help support user privacy by protecting access to the following:
- Restricted data, such as system state and users' contact information
- Restricted actions, such as connecting to a paired device and recording audio.
- Starting from Android 6.0 (API 23), users are not asked for permissions at the time of installation rather developers need to request the permissions at the run time. Only the permissions that are defined in the manifest file can be requested at run time.

TYPES OF PERMISSIONS

1. Install-Time Permissions:

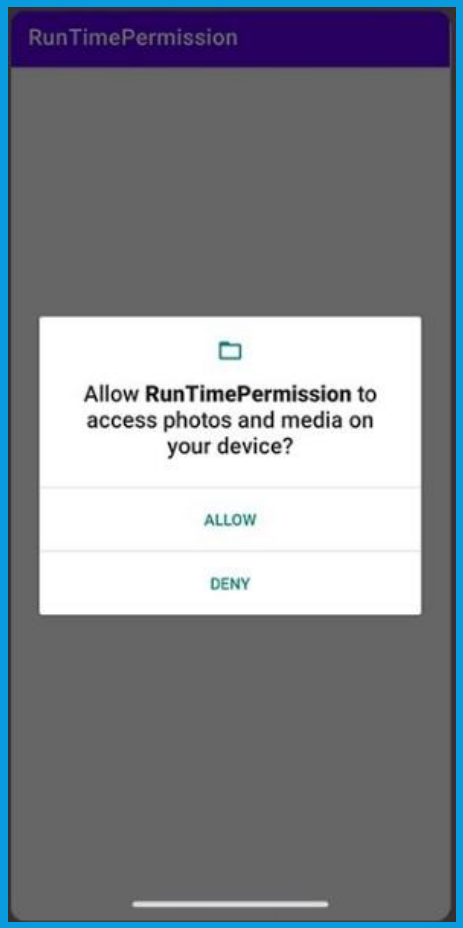
- If the Android 5.1.1 (API 22) or lower, the permission is requested at the installation time at the Google Play Store.
- If the user Accepts the permissions, the app is installed. Else the app installation is canceled.



TYPES OF PERMISSIONS

2. Run-Time Permissions:

- If the Android 6 (API 23) or higher, the permission is requested at the run time during the running of the app.
- If the user Accepts the permissions, then that feature of the app can be used. Else to use the feature, the app requests permission again.
- So, now the permissions are requested at runtime. In this article, we will discuss how to request permissions in an Android Application at run time.



ANDROID MANIFEST FILE

- The AndroidManifest.xml file contains information of your package, including components of the application such as activities, services, broadcast receivers, content providers etc.
- It performs some other tasks also:
  - It is responsible to protect the application to access any protected parts by providing the permissions.
  - It also declares the android api that the application is going to use.
  - It lists the instrumentation classes. The instrumentation classes provides profiling and other informations. These informations are removed just before the application is published etc.
  - This is the required xml file for all the android application and located inside the root directory.

ELEMENTS OF THE ANDROIDMANIFEST.XML FILE

- <manifest>
  - manifest is the root element of the AndroidManifest.xml file. It has package attribute that describes the package name of the activity class.
- <application>
  - application is the subelement of the manifest. It includes the namespace declaration. This element contains several subelements that declares the application component such as activity etc.

ELEMENTS OF THE ANDROIDMANIFEST.XML FILE

- The commonly used attributes are of this element are icon, label, theme etc.
  - android:icon represents the icon for all the android application components.
  - android:label works as the default label for all the application components.
  - android:theme represents a common theme for all the android activities.

ELEMENTS OF THE ANDROIDMANIFEST.XML FILE

- <activity>
  - activity is the subelement of application and represents an activity that must be defined in the AndroidManifest.xml file. It has many attributes such as label, name, theme, launchMode etc.
  - android:label represents a label i.e. displayed on the screen.
  - android:name represents a name for the activity class. It is required attribute.

ELEMENTS OF THE ANDROIDMANIFEST.XML FILE

- <intent-filter>
  - intent-filter is the sub-element of activity that describes the type of intent to which activity, service or broadcast receiver can respond to.
- <action>
  - It adds an action for the intent-filter. The intent-filter must have at least one action element.
- <category>
  - It adds a category name to an intent-filter.

EXAMPLE

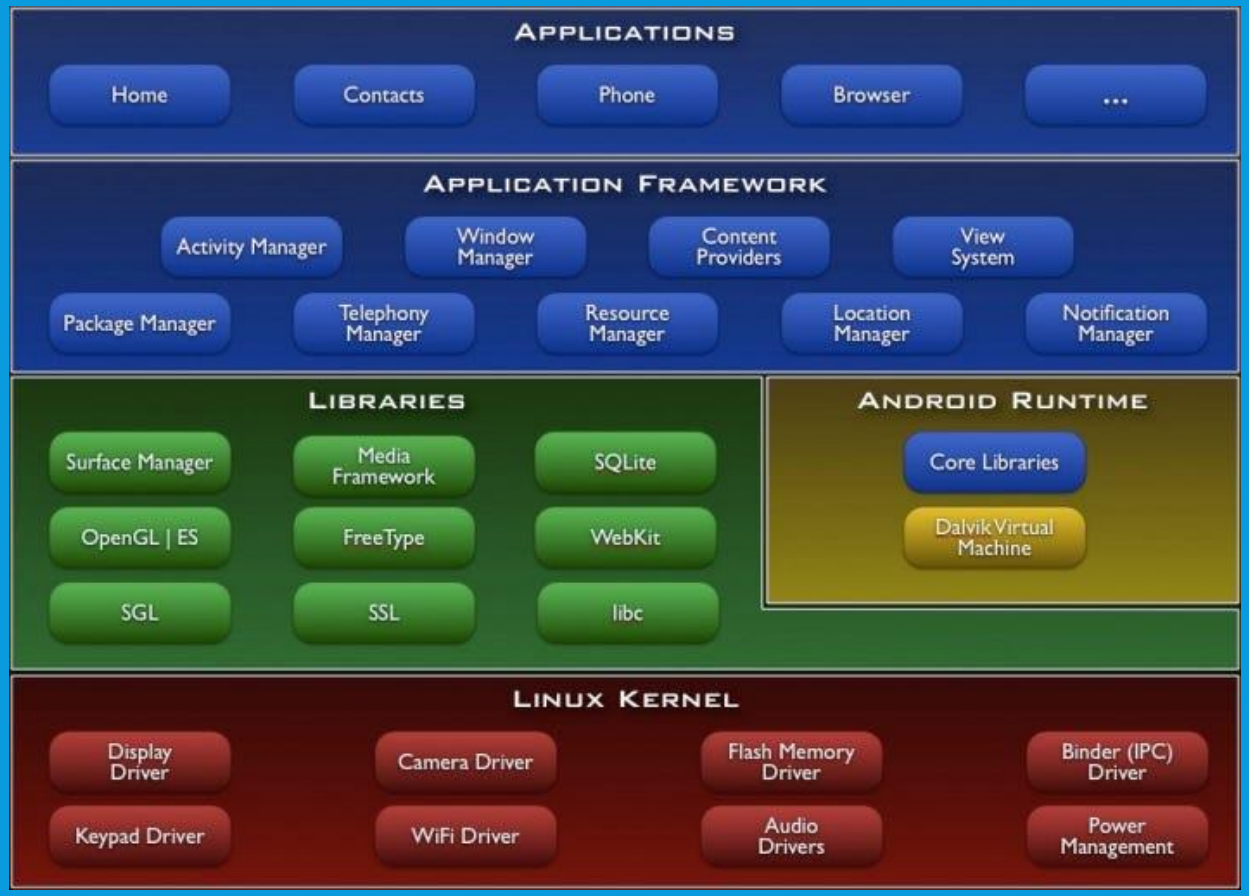
activity_main.xml | MainActivity.java | AndroidManifest.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.MANAGE_EXTERNAL_STORAGE"/>
    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="My Application"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MyApplication"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

ANDROID ARCHITECTURE (DIAGRAM)



ANDROID ARCHITECTURE OVERVIEW

1) Linux kernel

- It is the heart of android architecture that exists at the root of android architecture. Linux kernel is responsible for device drivers, power management, memory management, device management and resource access.

ANDROID ARCHITECTURE OVERVIEW

2) Native Libraries

- On the top of Linux kernel, there are Native libraries such as WebKit, OpenGL, FreeType, SQLite, Media, C runtime library (libc), etc.
- the WebKit library is responsible for browser support, SQLite is for database, FreeType for font support, Media for playing and recording audio and video formats.

ANDROID ARCHITECTURE OVERVIEW

3) Android Runtime

- In android runtime, there are core libraries and DVM (Dalvik Virtual Machine) which is responsible to run android application. DVM is like JVM but it is optimized for mobile devices. It consumes less memory and provides fast performance.

ANDROID ARCHITECTURE OVERVIEW

4) Android Framework

- On the top of Native libraries and android runtime, there is android framework. Android framework includes Android API's such as UI (User Interface), telephony, resources, locations, Content Providers (data) and package managers. It provides a lot of classes and interfaces for android application development.

ANDROID ARCHITECTURE OVERVIEW

5) Applications

- On the top of android framework, there are applications. All applications such as home, contact, settings, games, browsers are using android framework that uses android runtime and libraries. Android runtime and native libraries are using linux kernal.