

# Lab 02

# Android basic

Software Studio

DataLab, CS, NTHU

2022 spring

# Outline

- Introduction to Android
- Folders in Android app
  - Activity
  - User Interface
  - View & View Group
- Some rare cases in test

# Introduction to Android

- Android is an *Operating System for mobile devices* developed by Google, which is built upon Linux kernel.



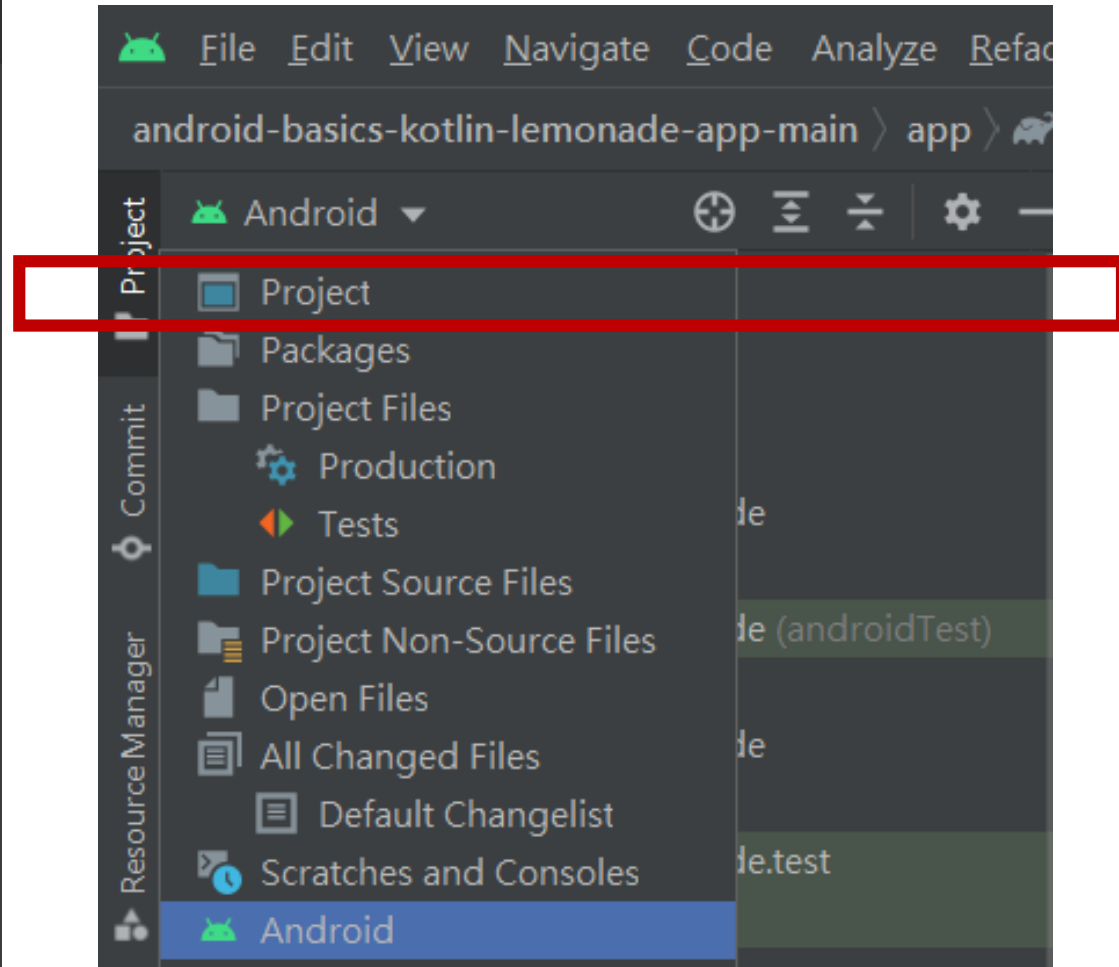
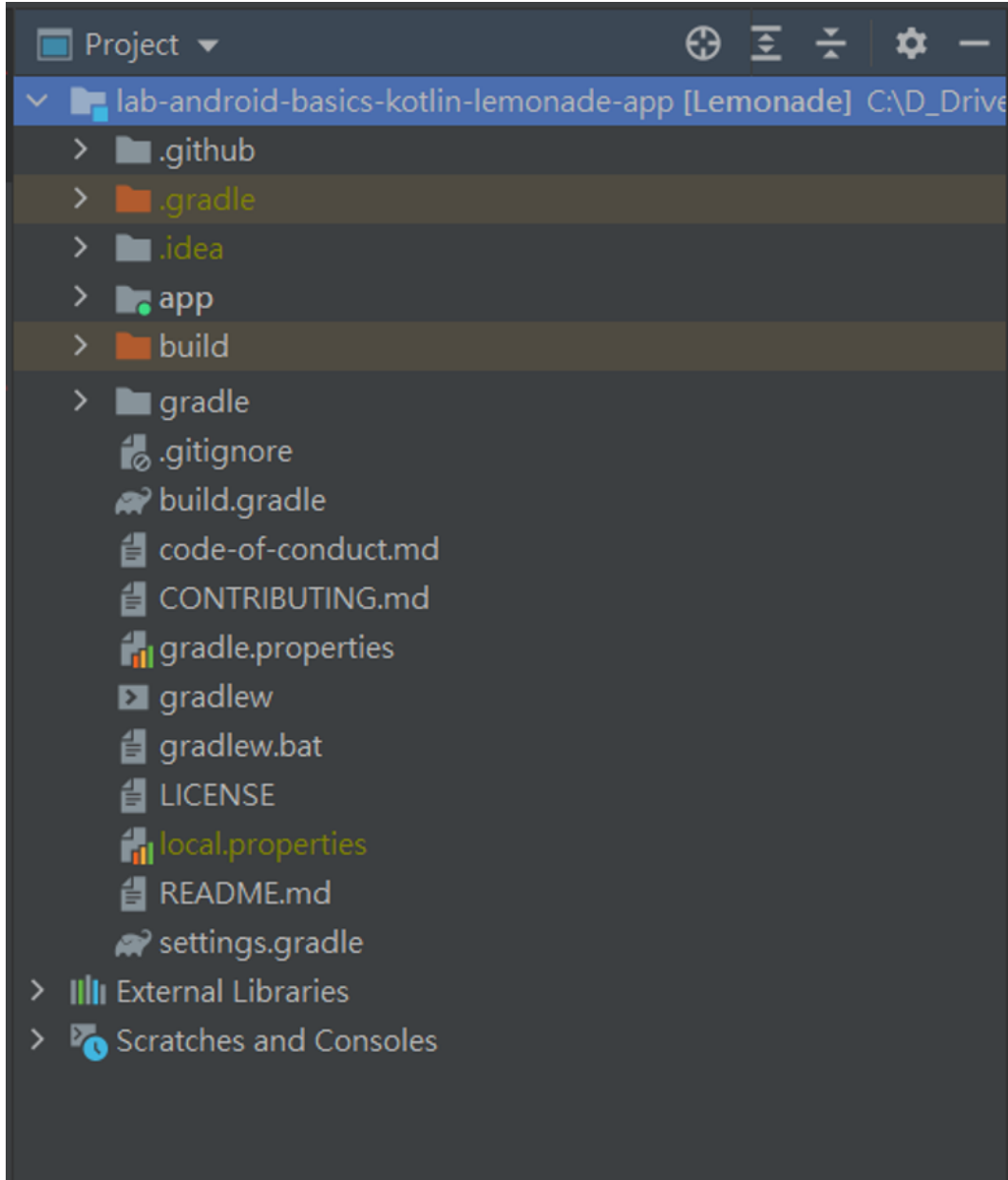
# Introduction to Android



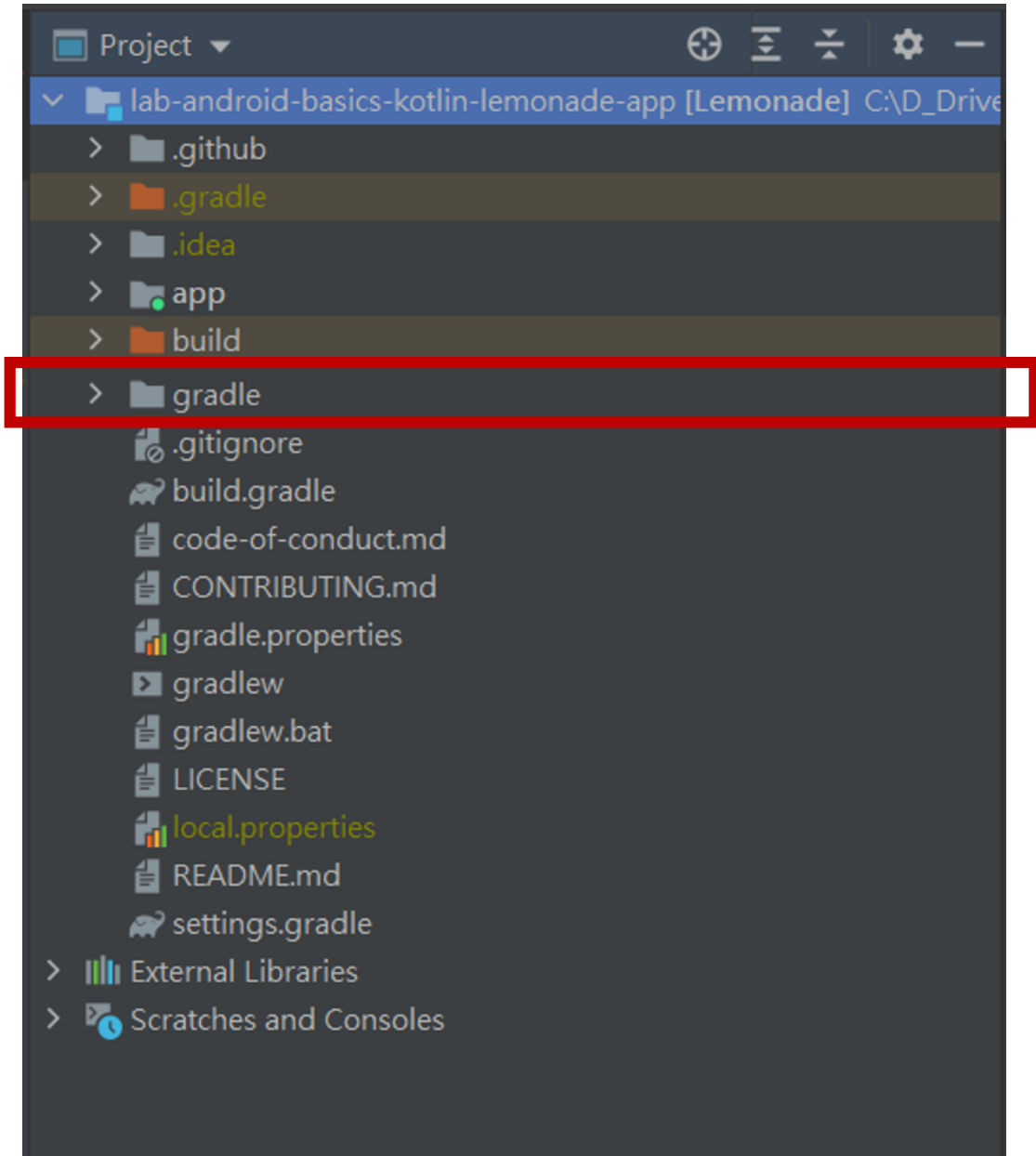
## Java vs Kotlin

Which Programming Language Is Better For Android Developers?

# Folders In Android App



# Folders In Android App



## 1. `gradle` folder

- What is gradle?
- Android Studio uses **Gradle**, an **advanced build toolkit**, to automate and manage the build process, while allowing you to define flexible custom build configurations.

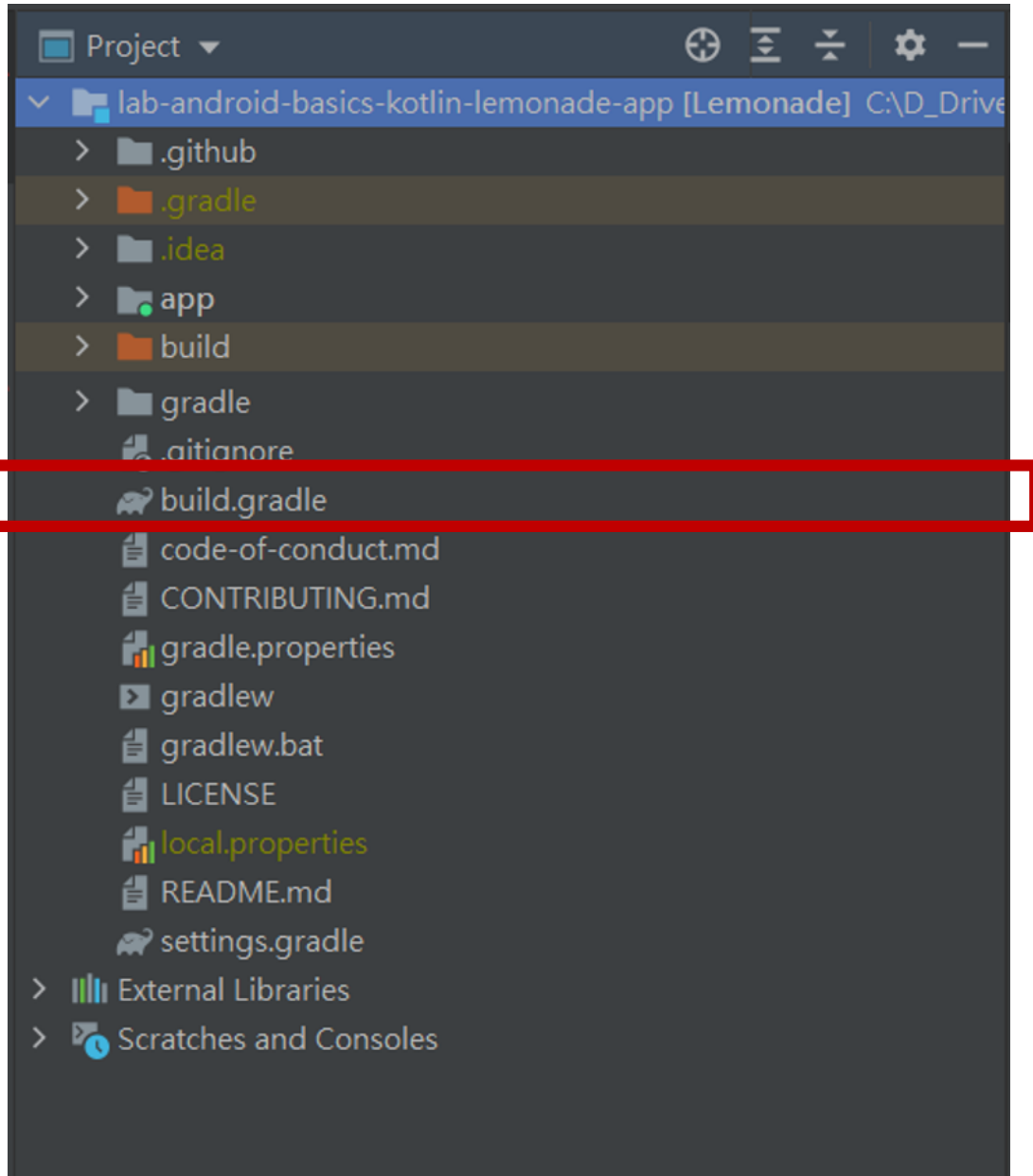
# Folders In Android App

## 2. build.gradle (project level)

- a Groovy-based DSL (domain specific language) for describing the builds.
- define our dependencies

<https://search.maven.org/>

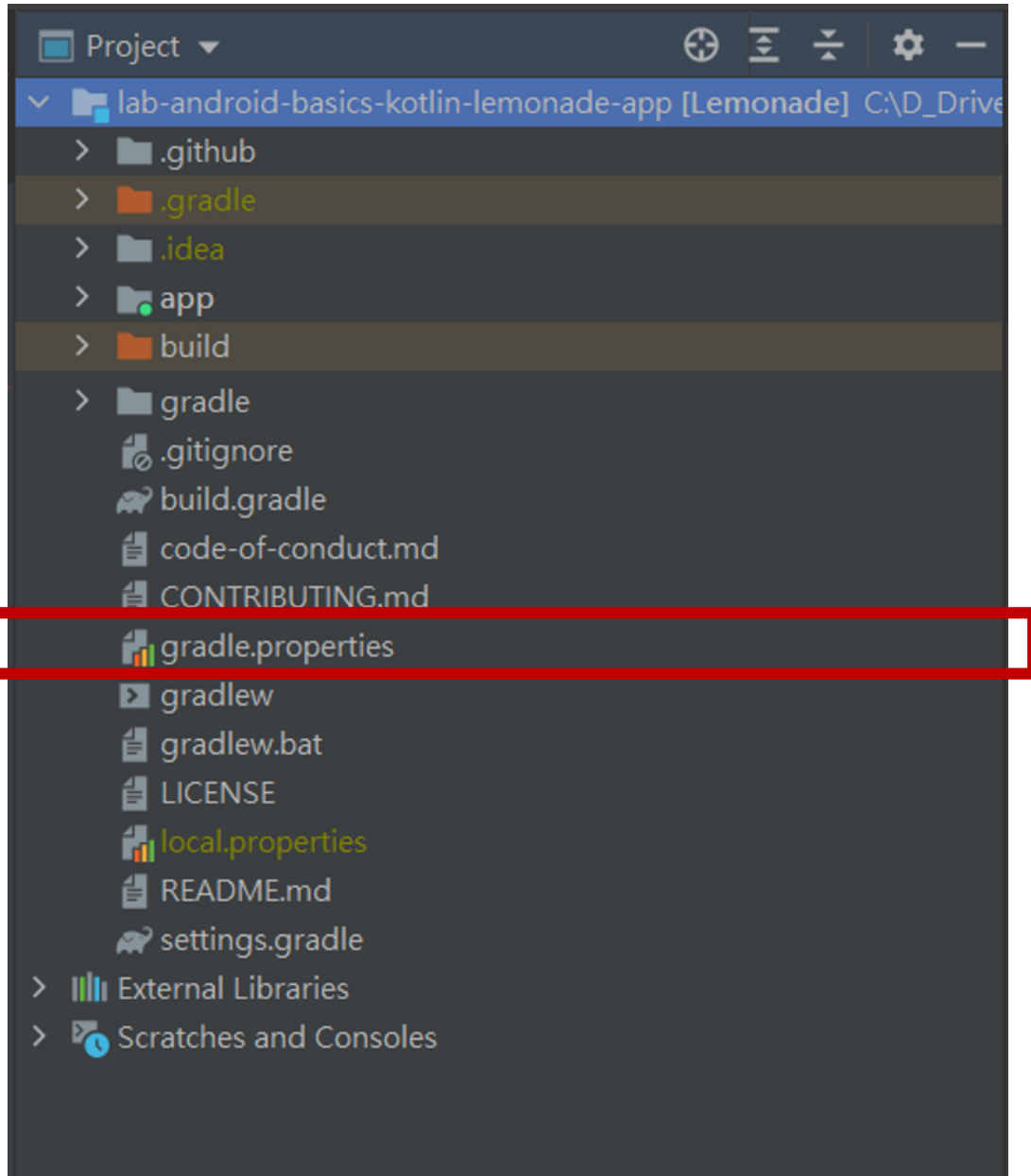
<https://maven.google.com/web/index.html>



# Folders In Android App

## 3. gradle.properties

- configure project-wide Gradle settings

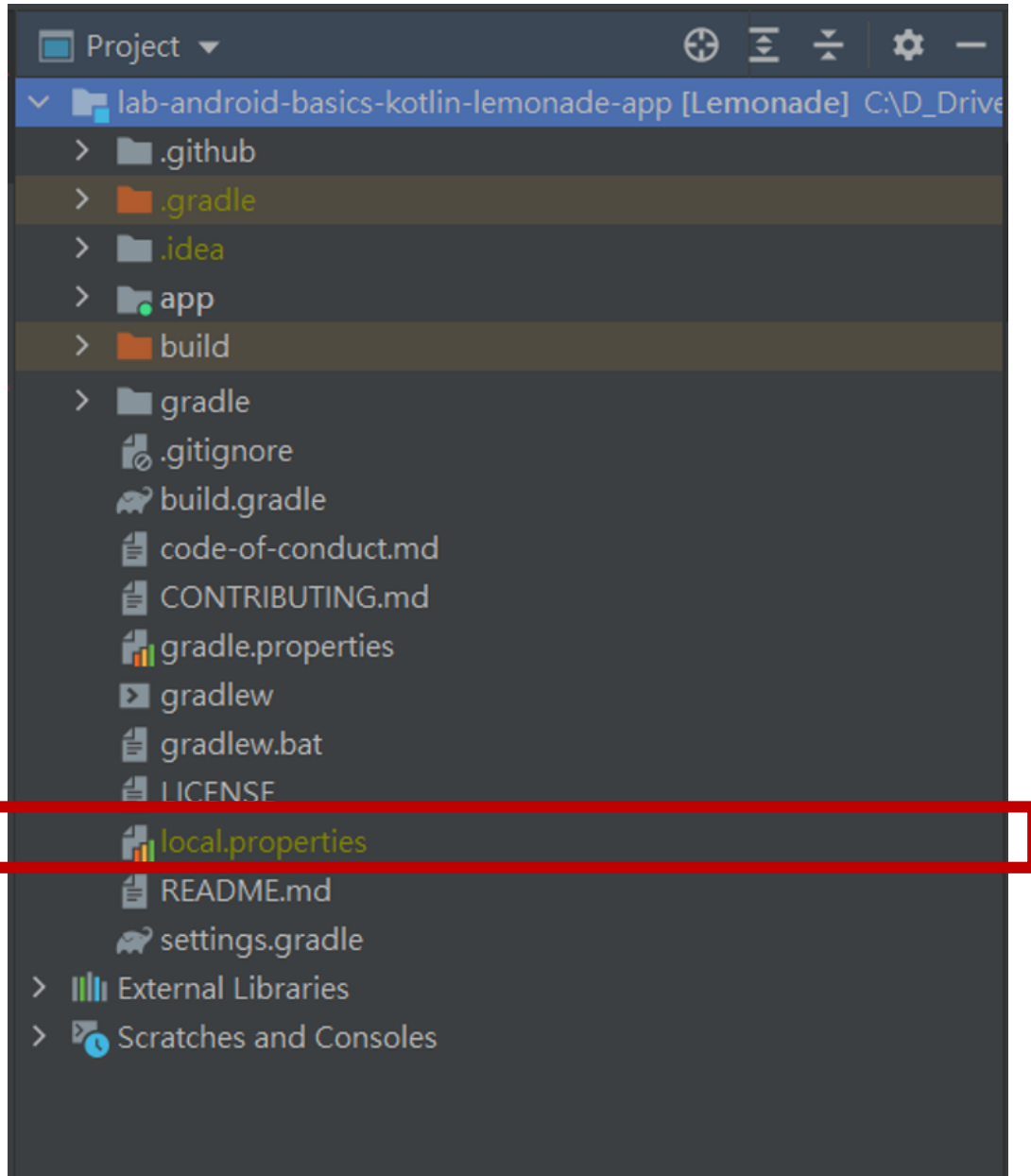




# Folders In Android App

## 4. local.properties

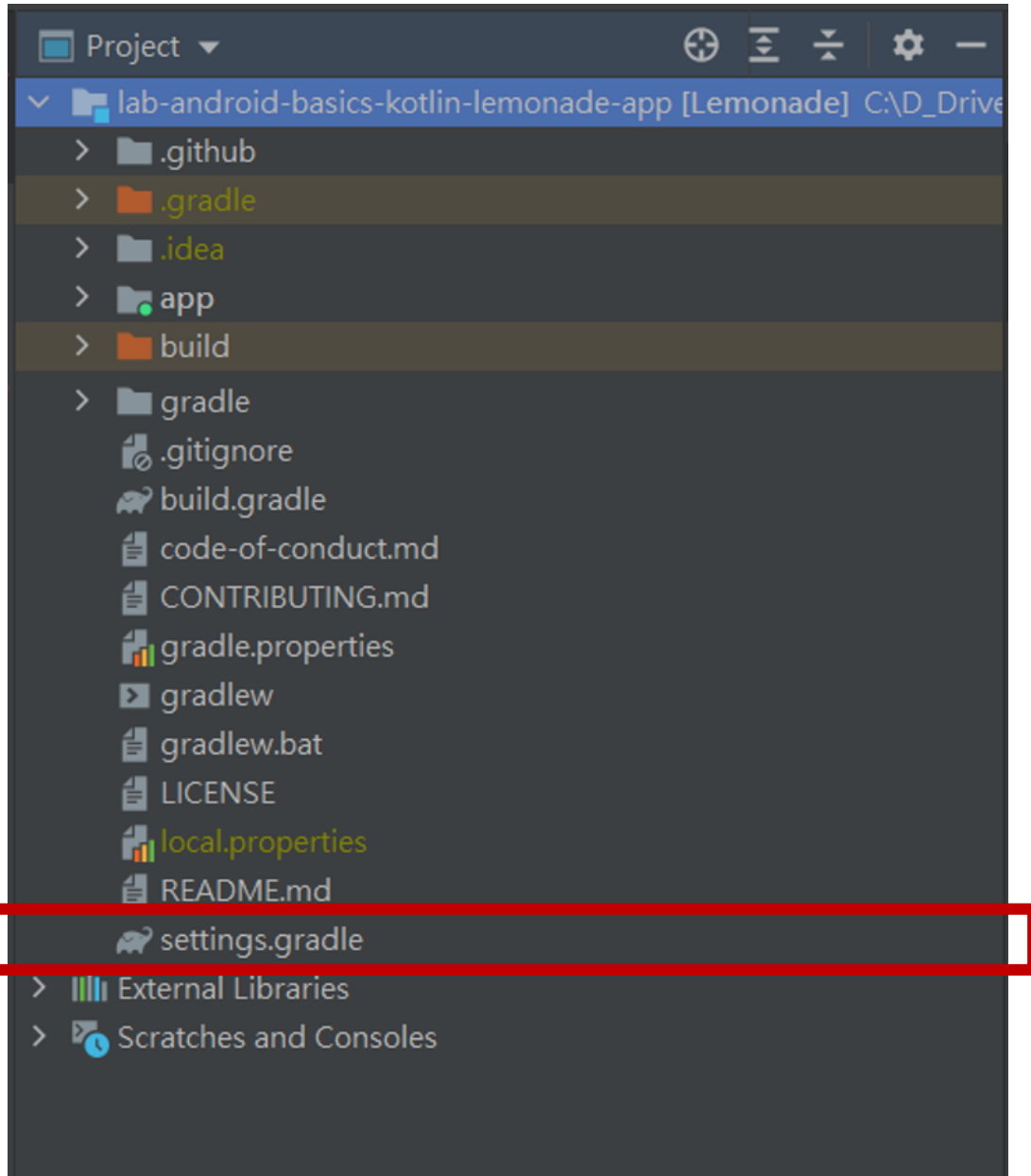
- Configures local environment properties for the build system, including the following:

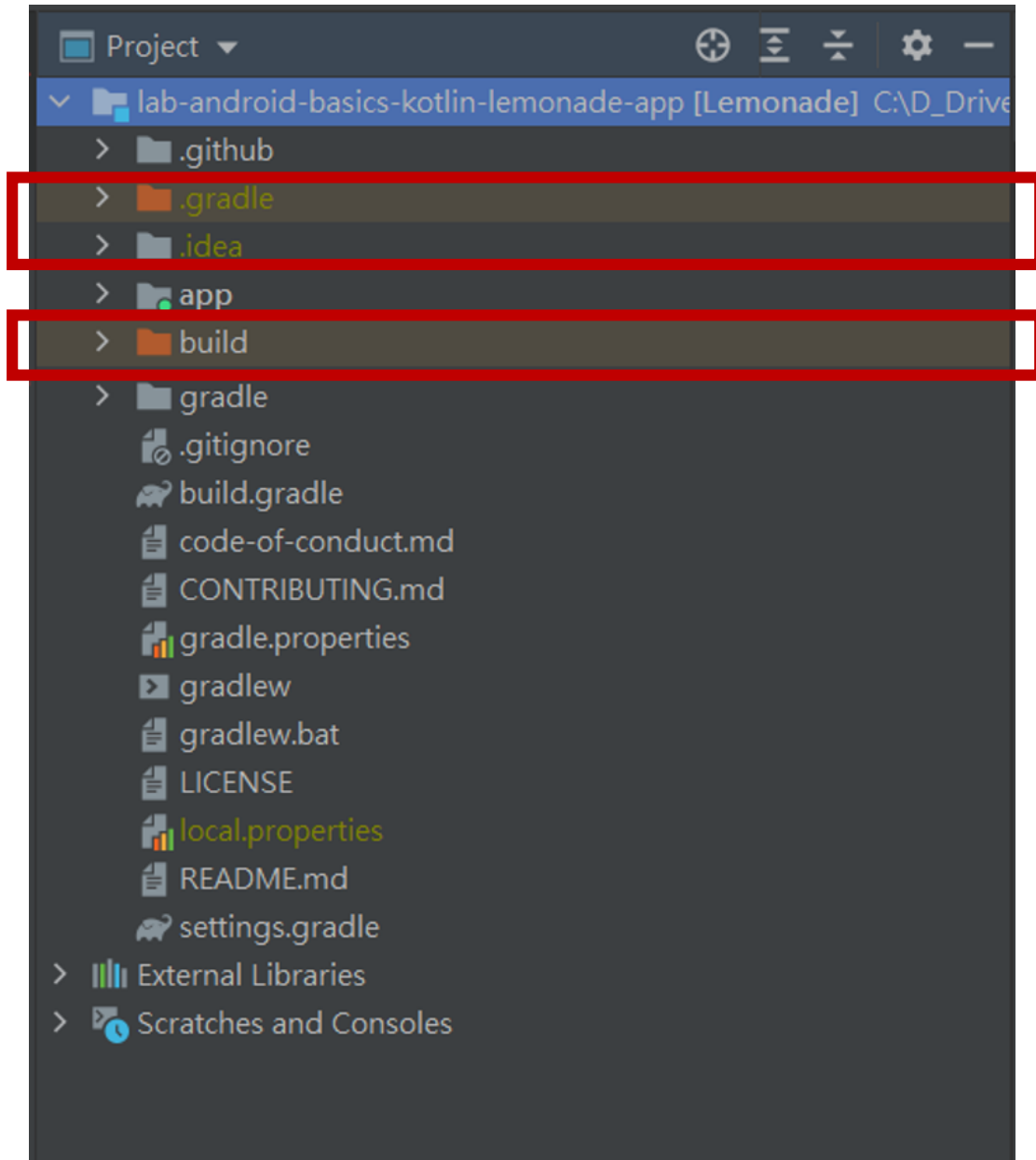


# Folders In Android App

## 5. settings.gradle

- defines project-level repository settings
- tells Gradle which modules it should include when building your app.

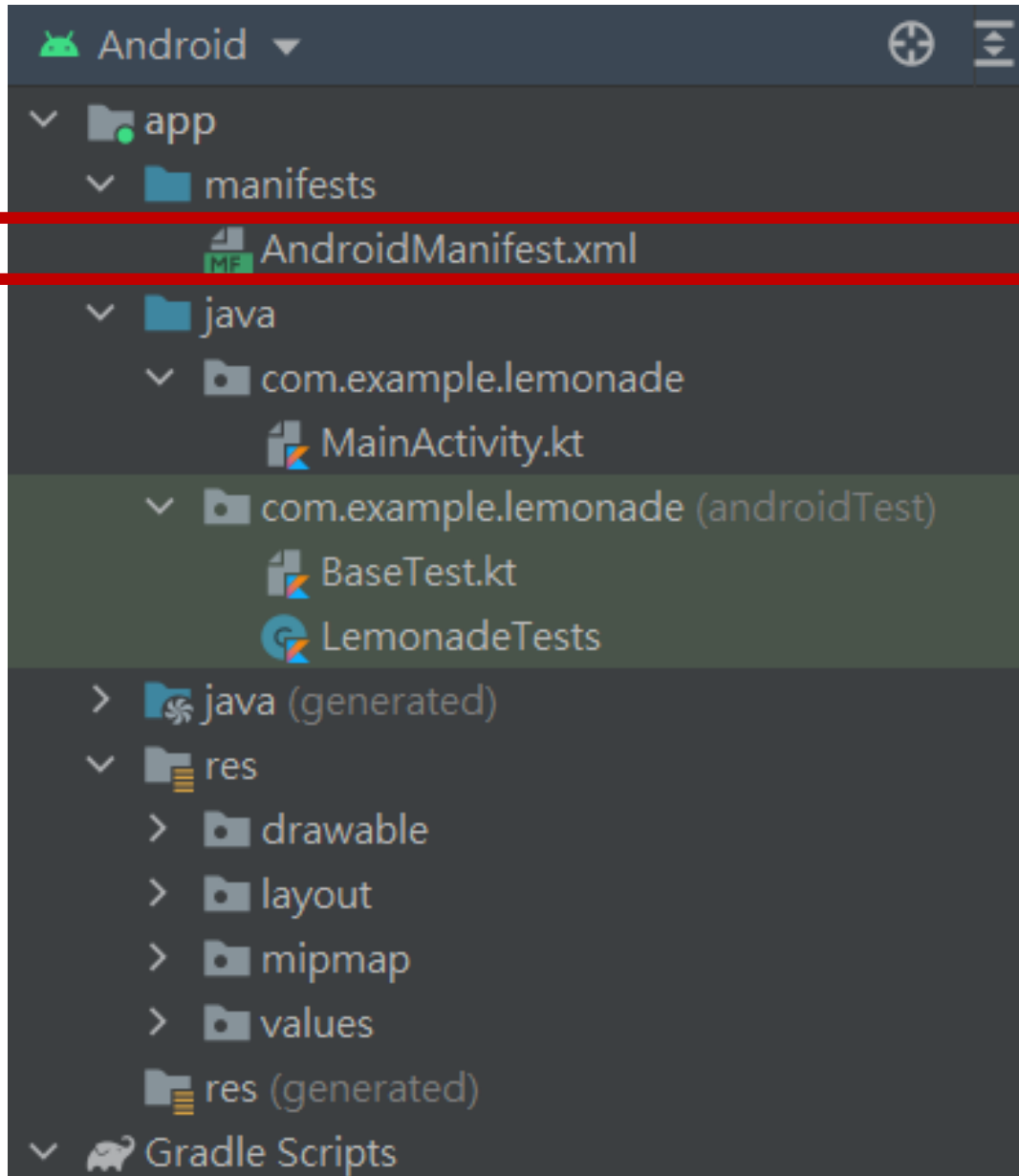




# Folders In Android App

These files can be ignored because they are generated after compile.

- .gradle
- .idea
- build



## Inside `app` folder

### 1. `manifests` declare:

- Components of the app including all **activities, services, broadcast receivers, and content providers.**
- Permissions of accessing the system or other apps.
- Hardware and software features the app requires.

# Manifests

- Activity need to registered in Manifests.
- Contains all the definitions, such as **package name, activities, services, minimum level of the Android API, linked libraries, and permissions** that allow the Android OS to launch and run the application.

# Manifests

xmlns: abbreviation of xml namespace  
avoid the problem of same element  
android: prefix of namespace

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.lemonade">

  <application
    android:allowBackup="true" Prefix:element
    android:icon="@mipmap/ic_launcher_lemonade"
    android:label="@string/app_label"
    android:roundIcon="@mipmap/ic_launcher_lemonade"
    android:supportsRtl="true"
    android:theme="@style/Theme.AppCompat"
    <activity android:name=".MainActivity"
      android:text="@string/lemon_select"
      tools:text="@string/lemon_select"
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  </application>

</manifest>
```

# Manifests

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.lemonade"> Keep this same as applicationId in build.gradle (module)

    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher_lemonade"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_lemonade_round"
        android:supportRtl="true"
        android:theme="@style/Theme.Lemonade">
        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

# Manifests

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.example.lemonade"> Due to this
```

```
<application
```

```
  android:allowBackup="true"
```

```
  android:icon="@mipmap/ic_launcher_lemonade"
```

```
  android:label="@string/app_name"
```

```
  android:roundIcon="@mipmap/ic_launcher_lemonade_round"
```

```
  android:supportsRtl="true"
```

```
  android:theme="@style/Theme.Lemonade"
```

```
  <activity android:name=".MainActivity">
```

```
    <intent-filter>
```

```
      <action android:name="android.intent.action.MAIN" />
```

```
      <category android:name="android.intent.category.LAUNCHER" />
```

```
    </intent-filter>
```

```
  </activity>
```

```
</application>
```

```
</manifest>
```

**.MainActivity resolved to  
com.example.lemonade.MainActivity**



# Manifests

```
<activity android:name=".MainActivity">  
  <intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
  
    <category android:name="android.intent.category.LAUNCHER" />  
  </intent-filter>  
</activity>
```

## Intent

Intents are like "glue" that enable different activities from different applications to work together.

# Manifests

```
<activity android:name=".MainActivity">  
  <intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
  
    <category android:name="android.intent.category.LAUNCHER" />  
  </intent-filter>  
</activity>
```

## Intent

declares an intention to do something, such as launching an activity, broadcasting a message, starting a service, display a web page, dialing or answering a phone call.

# Manifests

```
<activity android:name=".MainActivity">
  <intent-filter>
    <action android:name="android.intent.action.MAIN" />

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

## Intent-filter

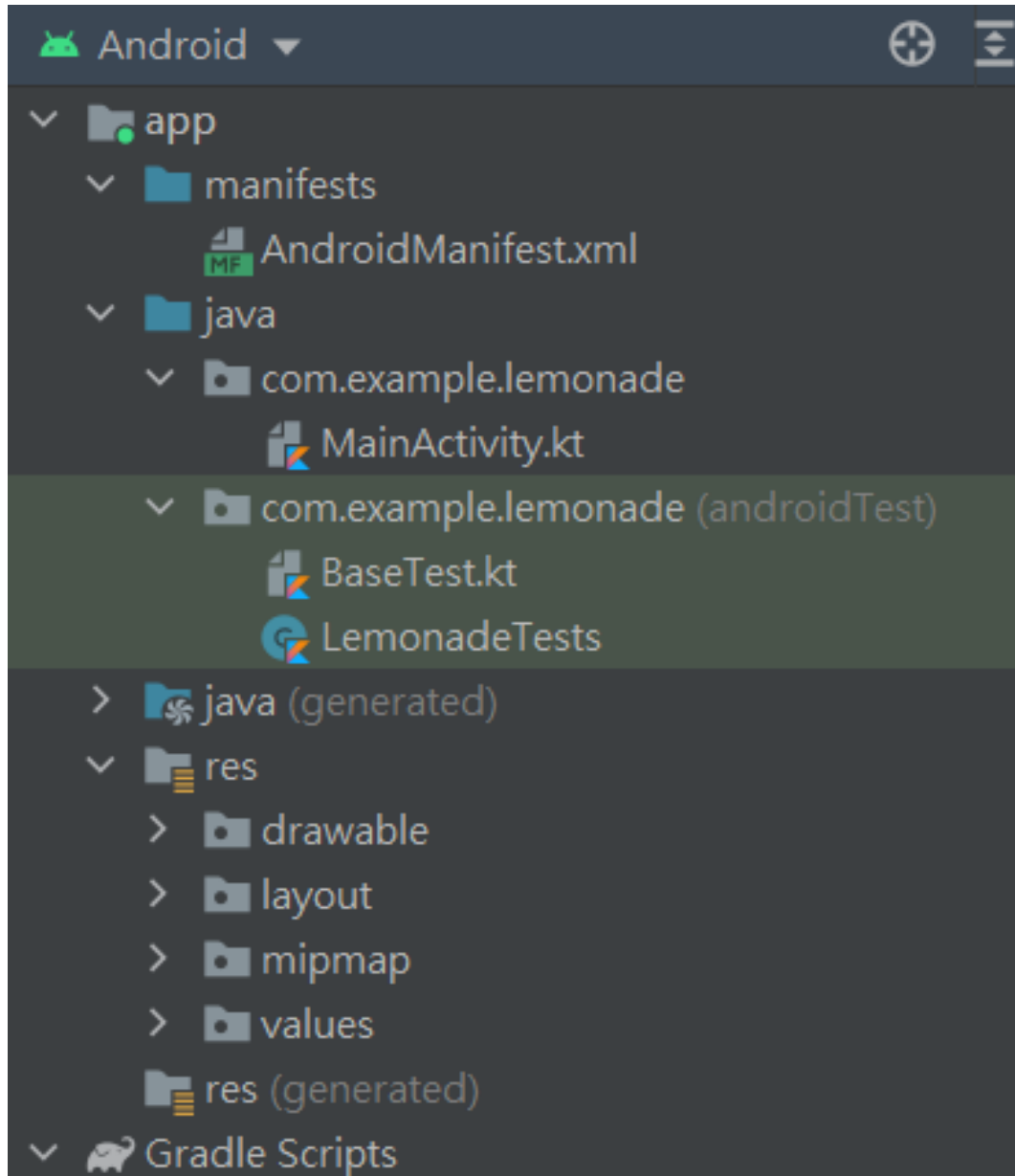
When the app launch an intent. System will scan all the intent filter and find the component.

# Manifests

```
<activity android:name=".MainActivity">  
  <intent-filter>  
    <action android:name="android.intent.action.MAIN" />  
  
    <category android:name="android.intent.category.LAUNCHER" />  
  </intent-filter>  
</activity>
```

For more information

<https://developer.android.com/guide/topics/manifest/manifest-intro>



## Inside `app` folder

### 2. Java ( where we work )

- Java & Kotlin file
- androidTest: UI test

### 3. res

- Images
- Layouts
- Strings
- Icon

# Activity

- An Android application could have one or more Activity.
- `onCreate()` is a call-back method, which is called back by the Android system when the activity is launched.

# User Interface

- There are two ways to create User Interface (UI) on Android:
  1. Write Java & Kotlin codes.
  2. Layout via XML descriptions and let the system generates the Java code for you.

# User Interface

Write Java codes.

```
public class MainActivity extends ..... {  
  
    // REPLACE the ENTIRE onCreate() method as follows:  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        TextView textView = new TextView(this); // Construct a TextView UI component  
        textView.setText("Hello, from my Java code!"); // Set the text message for TextView  
        setContentView(textView); // this Activity sets its content to the TextView  
    }  
  
    // Do not touch the rest of the codes, if any  
  
}
```



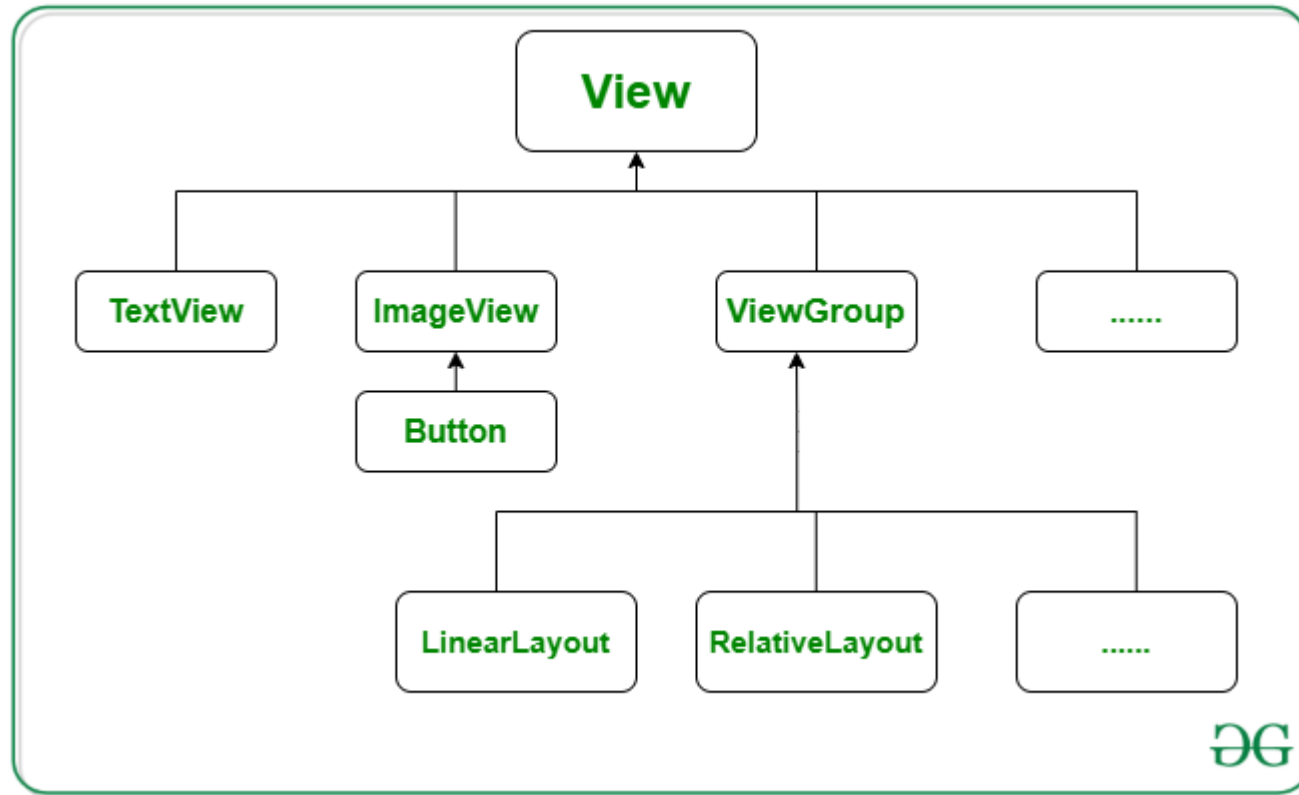
# User Interface

## Layout via XML descriptions

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
    }  
}
```

# View & View Group

- A View is a UI component (or widget, or control).

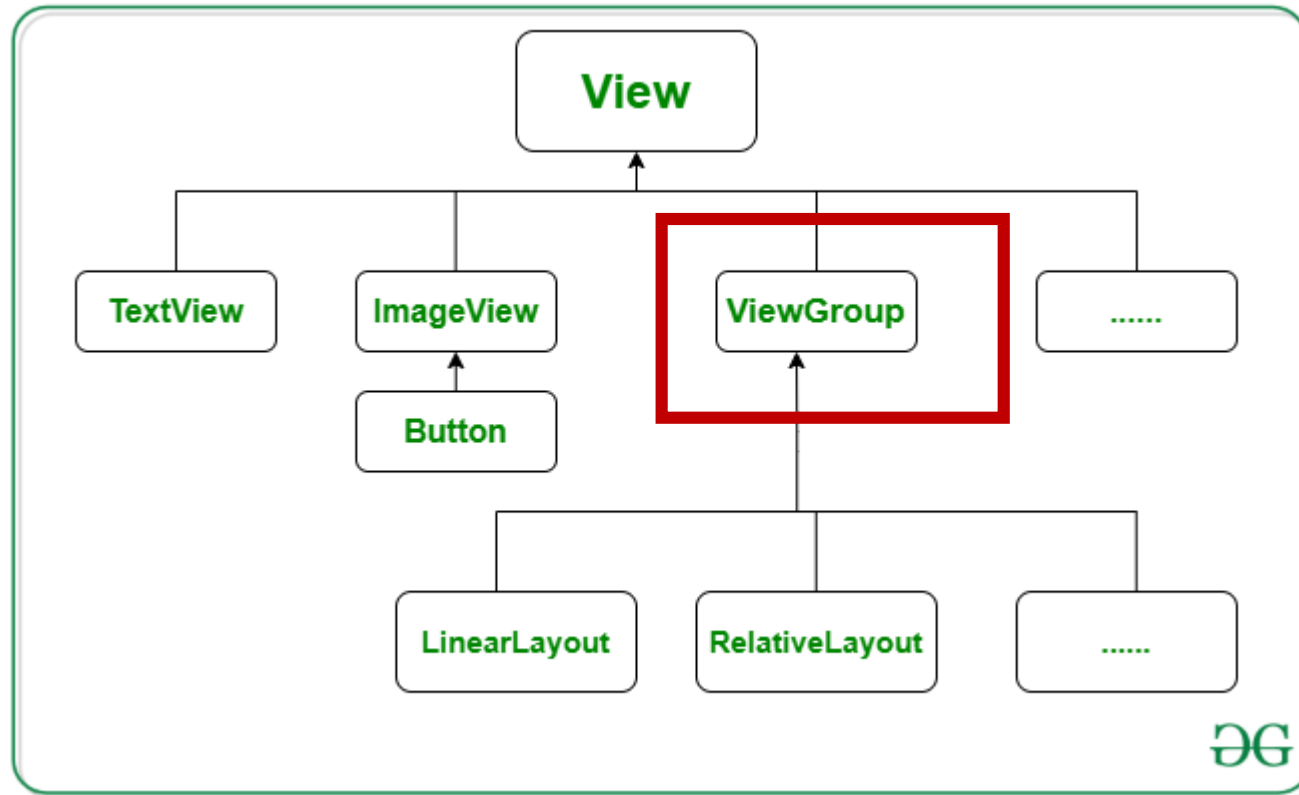


# View & View Group

- A View is a UI component (or widget, or control).
- Android provides many ready-to-use Views such as TextView, EditText, Button, RadioButton, etc, in package android.widget.

# View & View Group

- A View is a UI component (or widget, or control).



# View & View Group

- A ViewGroup is an invisible **container** used to layout the View components.
- Android provides many ready-to-use ViewGroups such as LinearLayout, RelativeLayout, TableLayout and GridLayout in package android.widget.

## Some rare cases in test

- `androidx.test.espresso.PerformException:Error`

## Some concerns



**Pipeline #33** triggered 2 days ago by

- Don't worry about it.

# keyboard shortcut

- Alt + Enter: Project quick fix
- Ctrl + O: override members
- Ctrl + Alt + L: Reformat code
- Ctrl + P: Show parameters for selected method
- Double Shift: Search everywhere