

# [GETTING STARTED WITH MOBILE APPLICATION]

Department : Math - Info  
University : Djilali Bounamaa Khemis miliana

Instructor: Kridi Ibrahim  
Email Address : i.kridi@univ-dbkm.dz

Semester 01 2024

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	What is a mobile app (mobile application)?	3
1.2	How are mobile apps built?	3
1.3	How does a mobile app work?	3
<b>2</b>	<b>Android OS</b>	<b>3</b>
<b>3</b>	<b>Android Architecture</b>	<b>4</b>
<b>4</b>	<b>Programming Language and IDE</b>	<b>4</b>
<b>5</b>	<b>Kotlin</b>	<b>4</b>
<b>6</b>	<b>Android Studio</b>	<b>4</b>
6.1	How to Install Android Studio	5
<b>7</b>	<b>Android Studio Interface</b>	<b>5</b>
7.1	Menu Part	5
7.2	Jetpack Compose: UI Design Area	5
7.3	Coding Area	6
7.4	Project Structure	6
7.5	Current Execution Part	7
7.5.1	File Structure of a Project in Android Studio	7
<b>8</b>	<b>Gradle Files in Android Studio</b>	<b>8</b>
8.1	Project-level build.gradle	9
8.2	Module-level build.gradle	10
<b>9</b>	<b>Running an Android App</b>	<b>11</b>

## Abstract

This course equips you to build Android apps. It covers the basics of mobile applications, dives into the Android OS, and introduces Kotlin and Android Studio (your development environment). Learn to install Android Studio, navigate its interface, and understand project structure. We'll also explore Gradle files used in Android app development. This course lays the groundwork for your journey to becoming an Android developer.

# 1 Introduction

## 1.1 What is a mobile app (mobile application)?

A mobile app (or mobile application) is a software application developed specifically for use on small, wireless computing devices, such as smartphones and tablets, rather than desktop or laptop computers.

Mobile apps are sometimes categorized according to whether they are web-based or native apps, which are created specifically for a given platform. A third category, cross-platform, combines elements of both native and web apps.

## 1.2 How are mobile apps built?

Mobile apps are built using a variety of programming languages and frameworks, and they can be downloaded and installed from app stores such as the Apple App Store or Google Play.

Mobile apps are designed to provide a wide range of functions and services and with consideration for the demands, constraints and capabilities of the devices they're built for.

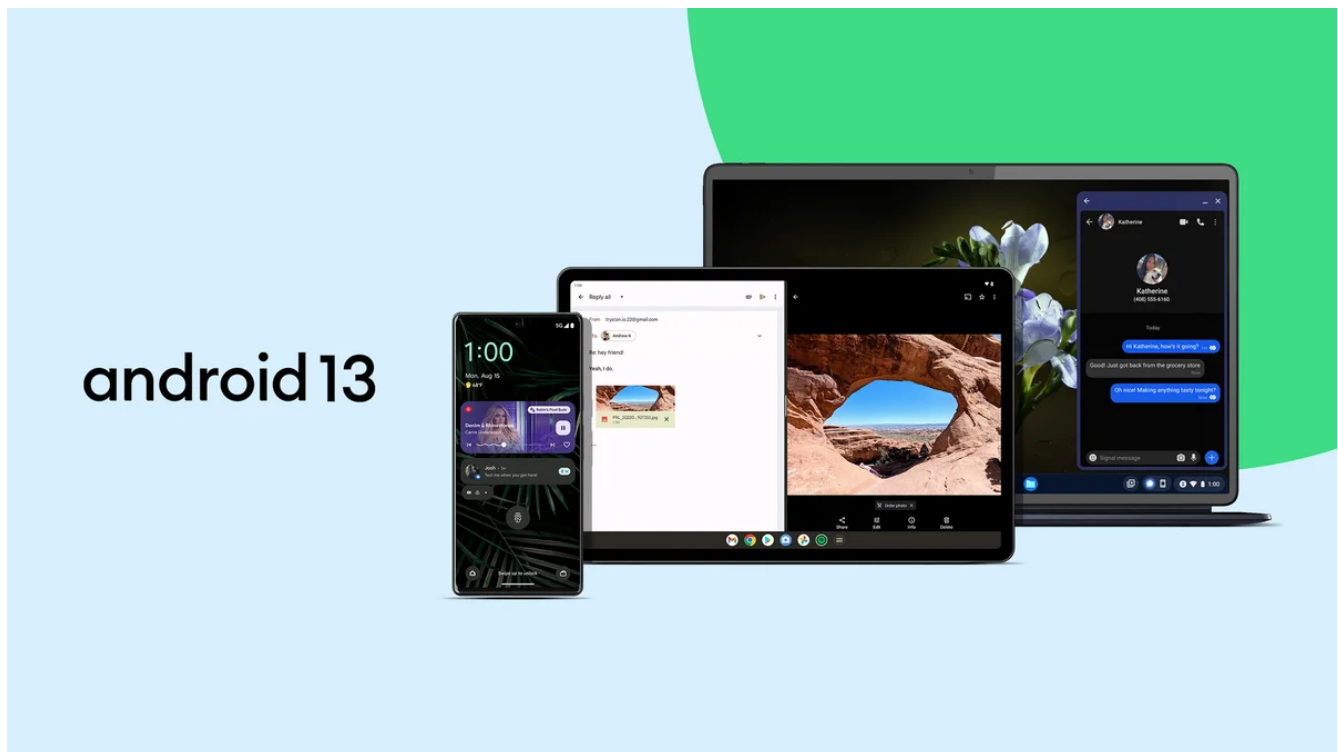
## 1.3 How does a mobile app work?

Mobile apps are designed to run on specific mobile operating systems such as iOS, Android and Windows Phone. When a mobile app is downloaded and installed on a device, it is stored in the device's memory and is launched using the device's operating system.

When a user opens a mobile app, the app communicates with the device's operating system and other built-in software components to access the device's hardware and services such as the camera, GPS and internet connection. The app then uses this information to provide its specific functions and services to the user.

# 2 Android OS

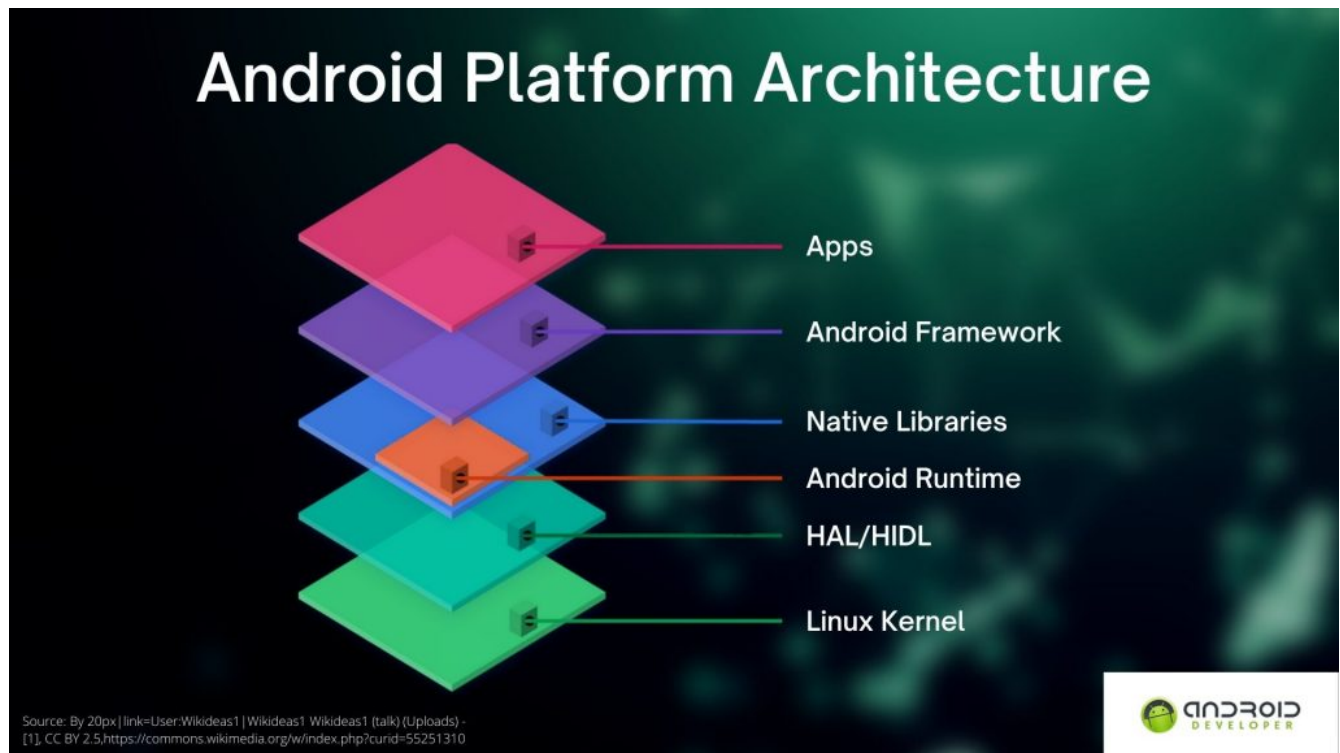
Android is an open-source operating system designed for various devices, including smartphones, tablets, smart TVs, smartwatches, and more. It powers a vast ecosystem of devices, offering diverse form factors.



### 3 Android Architecture

The Android operating system consists of key components:

- **Linux Kernel:** Provides core system services.
- **Hardware Abstraction Layer (HAL):** Bridges hardware and software.
- **Android Runtime (ART):** Manages execution and includes the Dalvik Virtual Machine (DVM).
- **Native C/C++ Libraries:** Provide essential system functionality.
- **Java API Framework:** Set of high-level APIs for building Android applications.
- **System and User Apps:** The application layer consists of pre-installed system apps and user-installed apps.



### 4 Programming Language and IDE

Java and Kotlin are the primary programming languages for Android app development. Kotlin, sponsored by Google, became an official language for Android Development in 2017.

### 5 Kotlin

Kotlin, a statically typed, general-purpose programming language developed by JetBrains, is fully interoperable with Java code. It was announced as one of the official languages for Android Development in 2017.

### 6 Android Studio

Android Studio is the official IDE for Android development, providing a powerful and feature-rich environment. It includes:

- Flexible Gradle build system.

- Emulator for testing.
- Unified environment for developing apps for all Android devices.
- Intelligent code completion and predefined code templates.
- Git integration for version control.

## 6.1 How to Install Android Studio

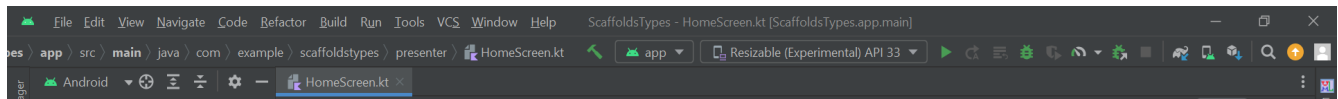
To install Android Studio, visit the [Android Studio Download page](#). Ensure your system meets the minimum requirements and follow the installation guide.

# 7 Android Studio Interface

Android Studio provides various templates for creating projects. The interface is divided into four parts:

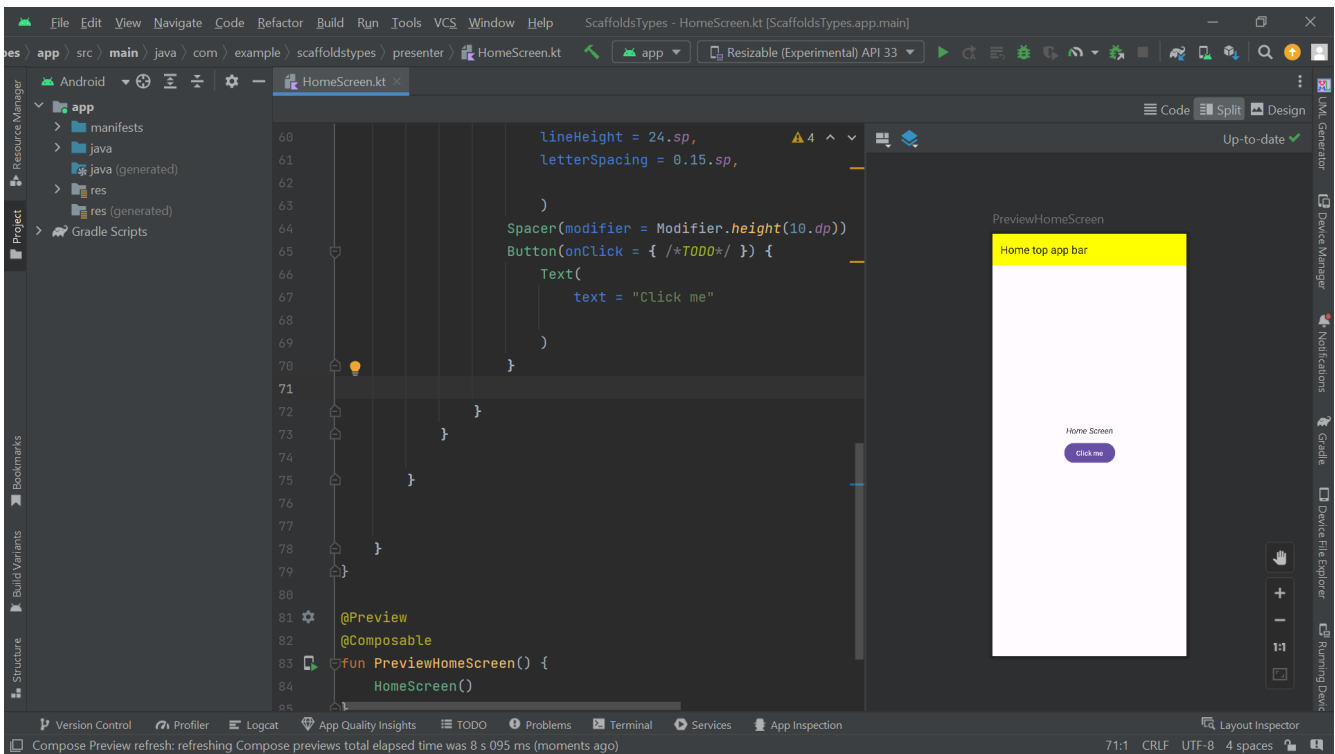
## 7.1 Menu Part

Located at the top, it provides options to create a new project, open an existing project, run the application, and select the desired device.



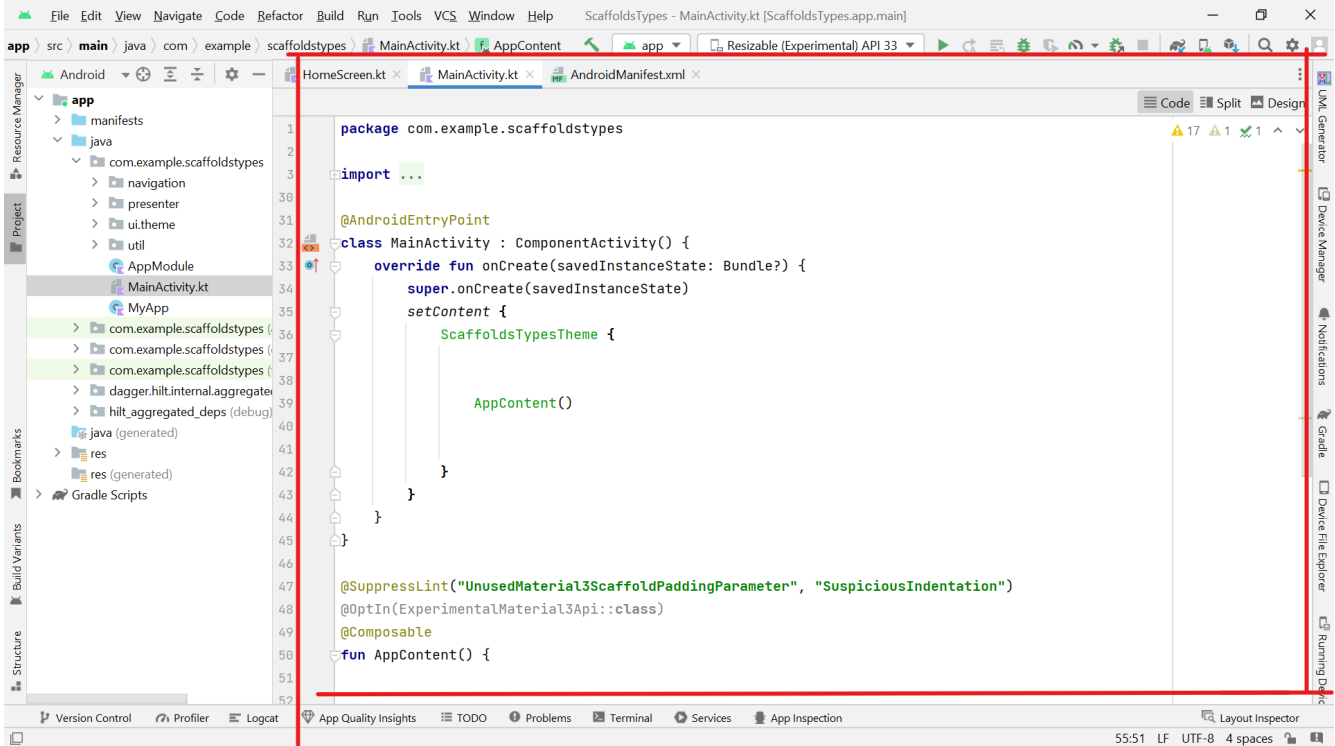
## 7.2 Jetpack Compose: UI Design Area

Jetpack Compose represents a paradigm shift in Android UI development, embracing a fully declarative approach. With Compose, UIs are defined as a function of their state, enabling developers to describe the desired UI hierarchy in a straightforward, intuitive manner using Kotlin. Forget about tedious XML layouts – in Compose, UI elements are composed using concise and expressive Kotlin code. Widgets, styles, and layouts are all defined declaratively, making it easier to understand and maintain the structure of your UI. Compose facilitates dynamic UI updates by automatically recomposing the UI based on changes in the underlying data or state. This reactive approach not only simplifies development but also enhances code readability and maintainability. With Compose, building beautiful and responsive UIs becomes a delightful, almost magical, experience.



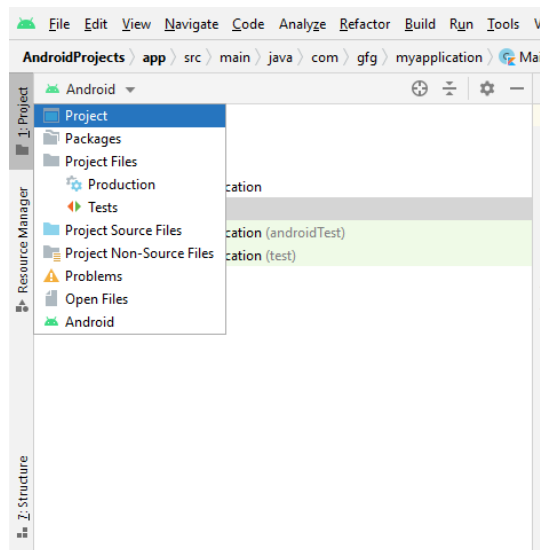
### 7.3 Coding Area

The central area is dedicated to code editing, supporting both Java and Kotlin programming languages.



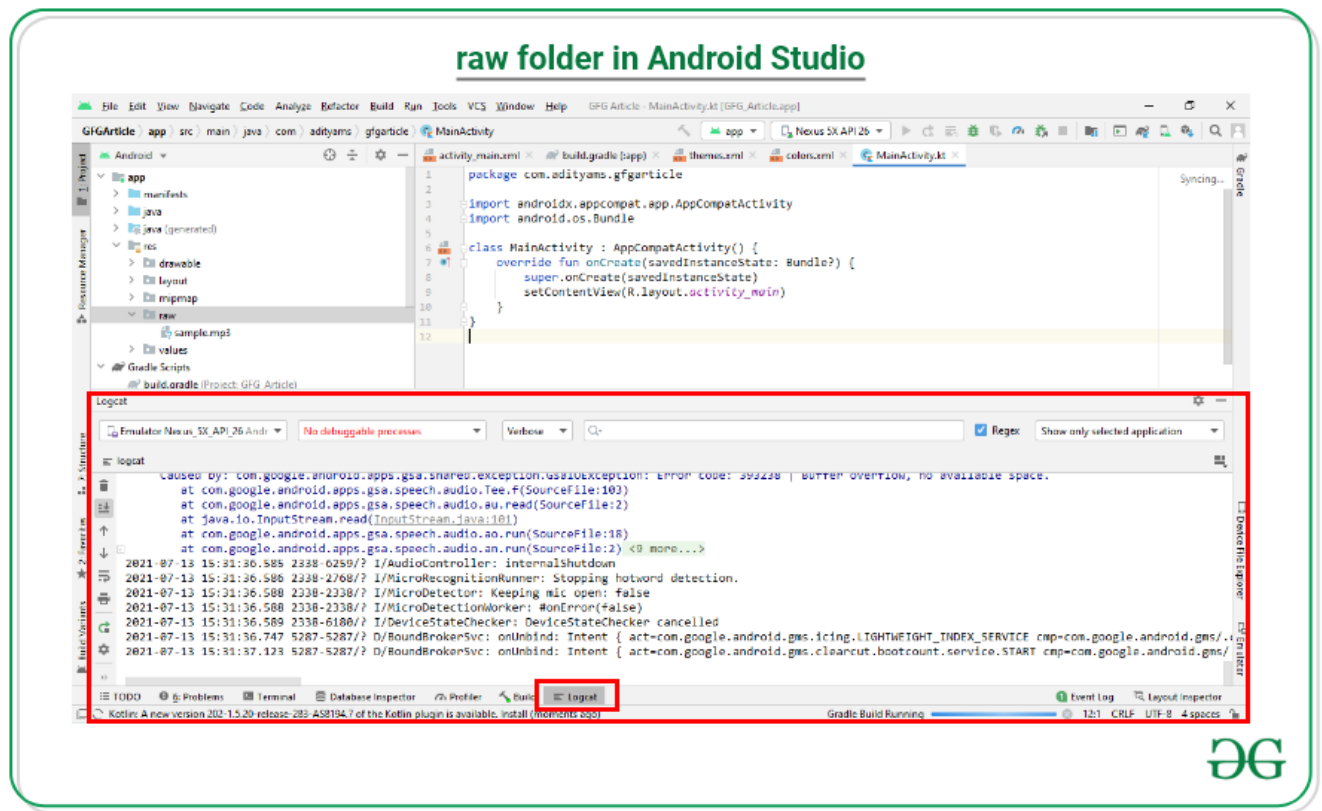
### 7.4 Project Structure

This area allows exploration of every file in the project, providing summarised and detailed views.



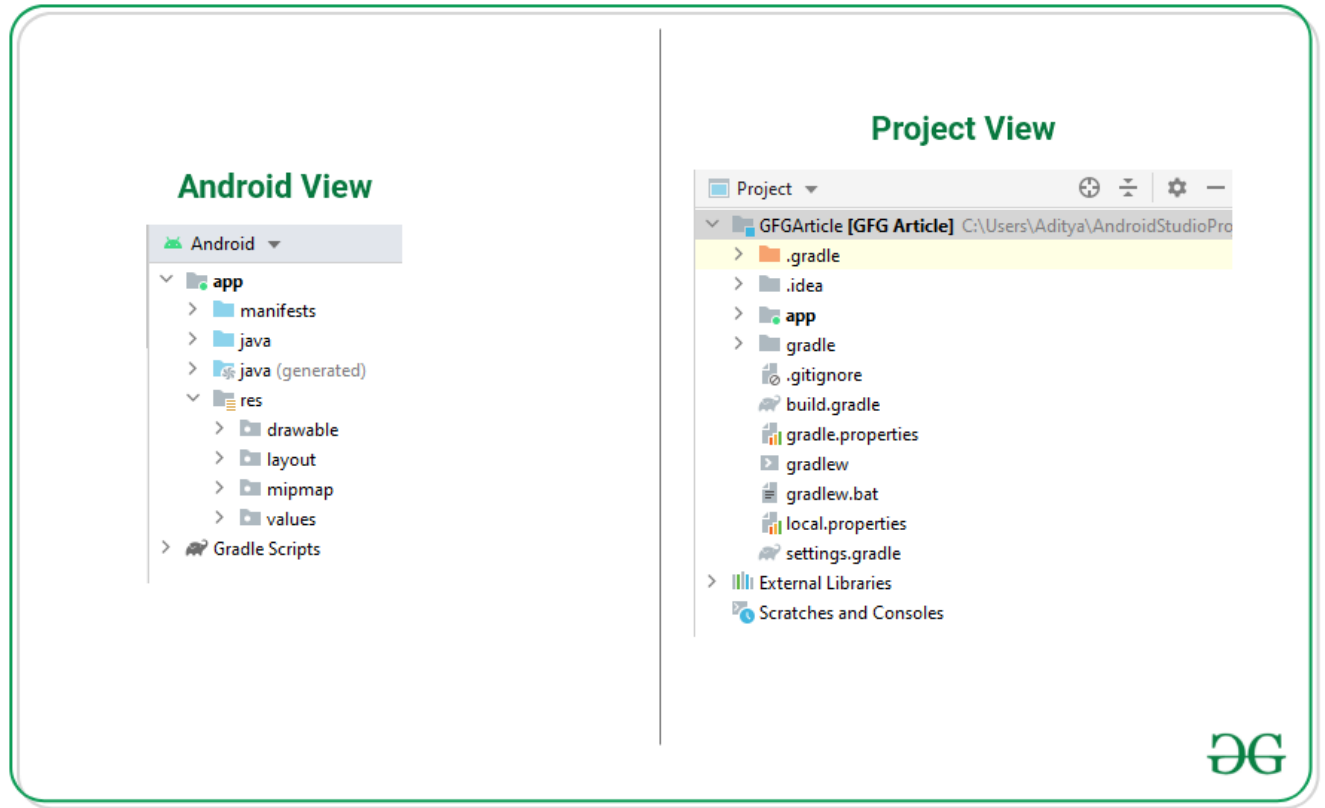
## 7.5 Current Execution Part

Provides a detailed view of the current execution process, displaying errors, build outputs, logcat results, etc.



### 7.5.1 File Structure of a Project in Android Studio

Android Studio project folder structure includes essential components like the Manifest file, application logic, drawable files, UI layout files, and more.

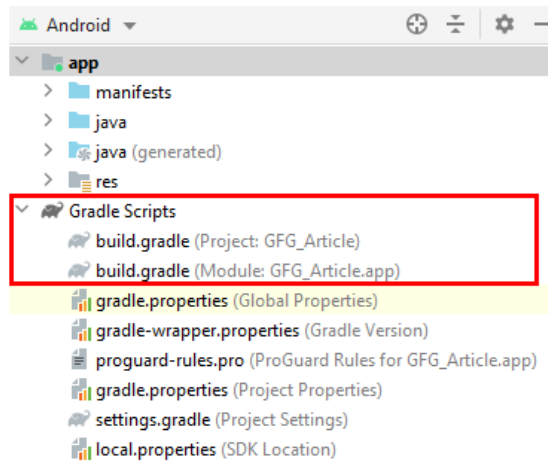


## 8 Gradle Files in Android Studio

In Android Studio, Gradle is a build automation tool used to manage project dependencies and build processes. There are project-level and module-level build.gradle files.

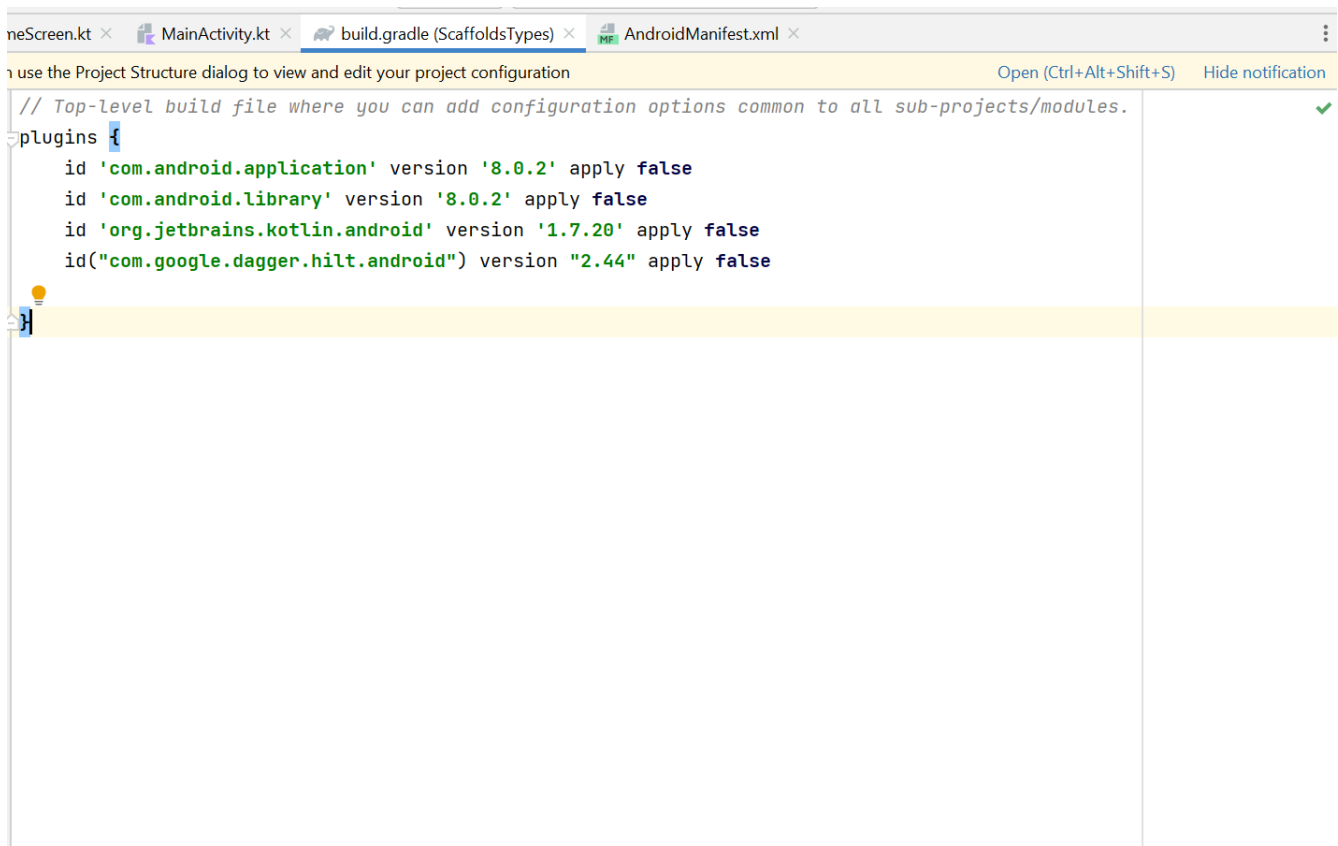


## build.gradle files in Android Studio



### 8.1 Project-level build.gradle

Located in the root of the project, it defines configurations for the entire project, specifying the buildscript, repositories, and dependencies.

A screenshot of an IDE window showing the 'build.gradle' file. The window title bar includes tabs for 'neScreen.kt', 'MainActivity.kt', 'build.gradle (ScaffoldsTypes)', and 'AndroidManifest.xml'. A yellow notification bar at the top says 'Use the Project Structure dialog to view and edit your project configuration' with 'Open (Ctrl+Alt+Shift+S)' and 'Hide notification' buttons. The code content is as follows:

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.
plugins {
    id 'com.android.application' version '8.0.2' apply false
    id 'com.android.library' version '8.0.2' apply false
    id 'org.jetbrains.kotlin.android' version '1.7.20' apply false
    id("com.google.dagger.hilt.android") version "2.44" apply false
}
```

## 8.2 Module-level build.gradle

Located in each module's directory, it contains configurations specific to the module, specifying dependencies, build types, and other settings.

// Module-level build.gradle file content  
apply plugin: 'com.android.application'

```
android {
    compileSdkVersion 30
    defaultConfig {
        applicationId "com.example.myapp"
        minSdkVersion 16
        targetSdkVersion 30
        versionCode 1
        versionName "1.0"
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-rules.pro'
        }
    }
}

dependencies {
    implementation 'com.android.support:appcompat-v7:28.0.0'
    implementation 'com.google.android.material:material:1.3.0'
    // ... other dependencies
}
```

## 9 Running an Android App

To run an application, use an emulator or a physical device connected via USB. Refer to How to install Android Virtual Device (AVD) for setting up an Android Virtual Device and How to Run the Android App on a Real Device? for running on a physical device.

