

MOBILE DEVELOPMENT

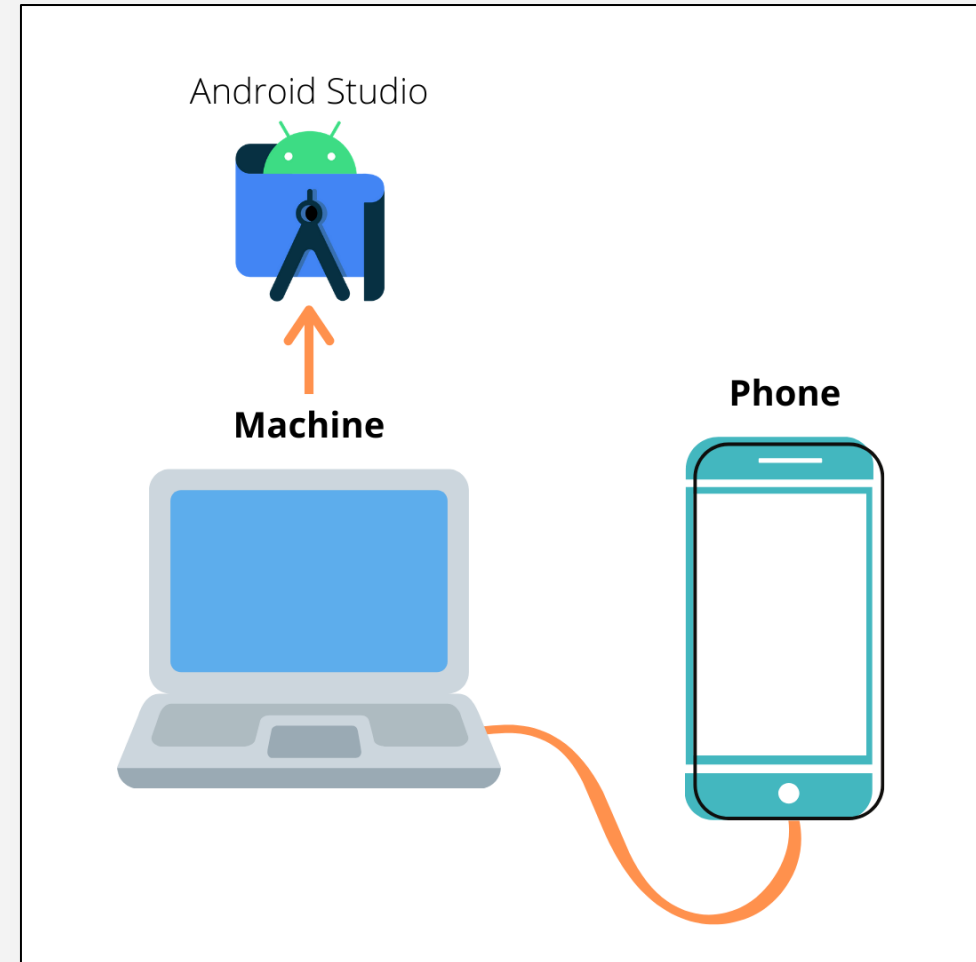
Intro to Android Studio

Outline

- Development setups
- Getting Familiar with Android Studio
- Running a Basic Application

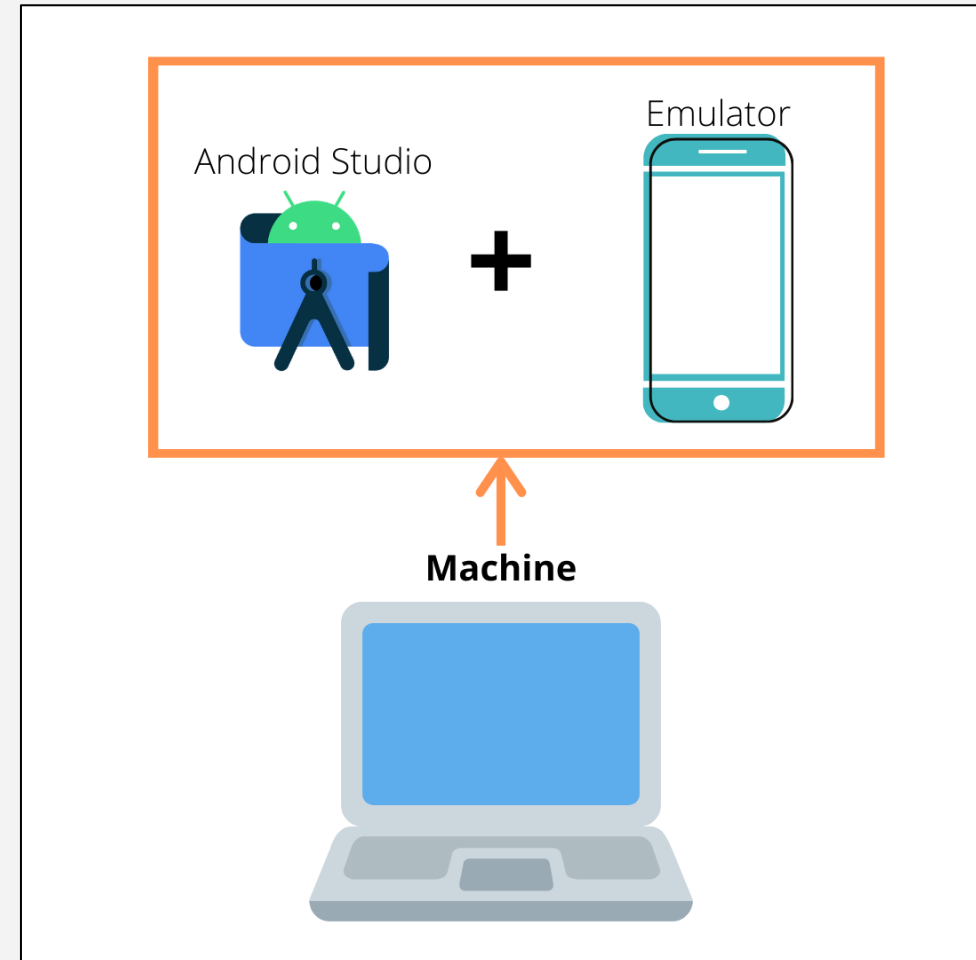
Some Setups for Development

- Machine + Physical Device
 - Shifts some computing to the phone
 - You'll need an... android phone
 - You'll be somewhat limited by the phone's OS version

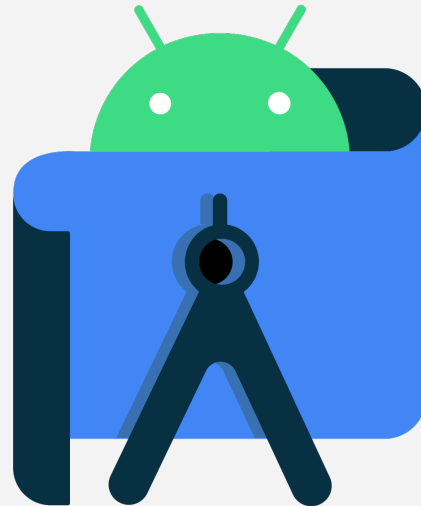


Some Setups for Development

- Machine Only
 - Resource demanding
 - Can use Android Studio's built in emulator or an alternative emulator
 - You can create virtual devices, but you'll have to download resources
 - VDs don't have some services:
 - Bluetooth, cellular service



Let's get started with **Android Studio**



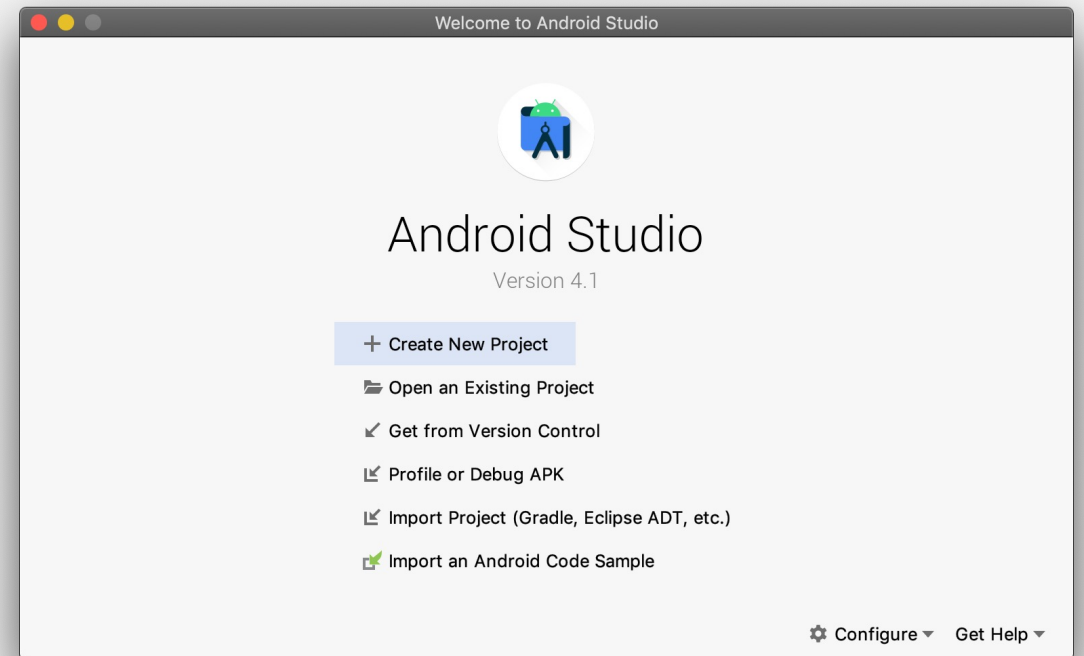
Android Studio

- While Android development can be done on most programming platforms, the official IDE is **Android Studio**
- For MOBDEVE, Android Studio will be the default IDE
 - Please consider making it your default as well
 - You won't have to worry about other packages you'll need to download, such as the Android SDK

Android Studio

- On first run, you should see something like this →
- Else, you'd see a list of previously opened projects

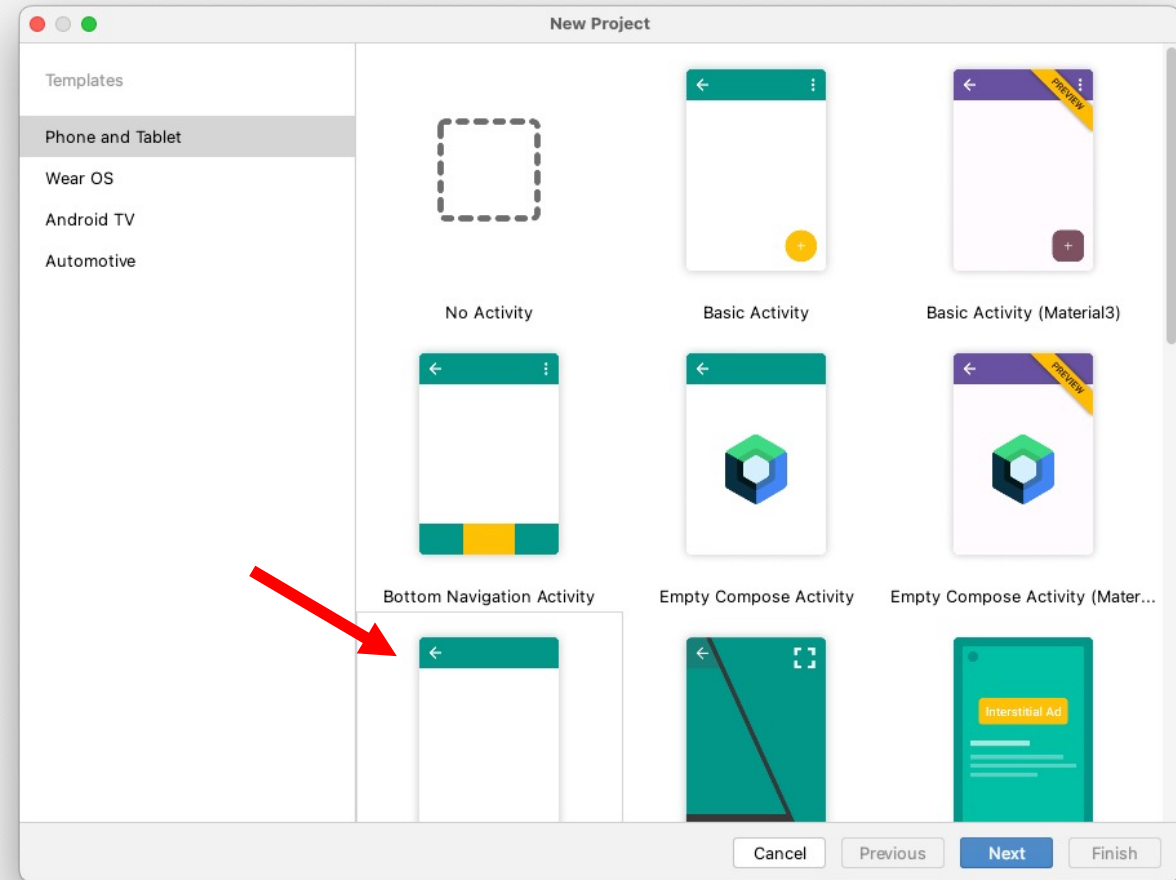
This is an older version of Android Studio...
I already have a number of projects created.



Let's start by creating a new project 😊

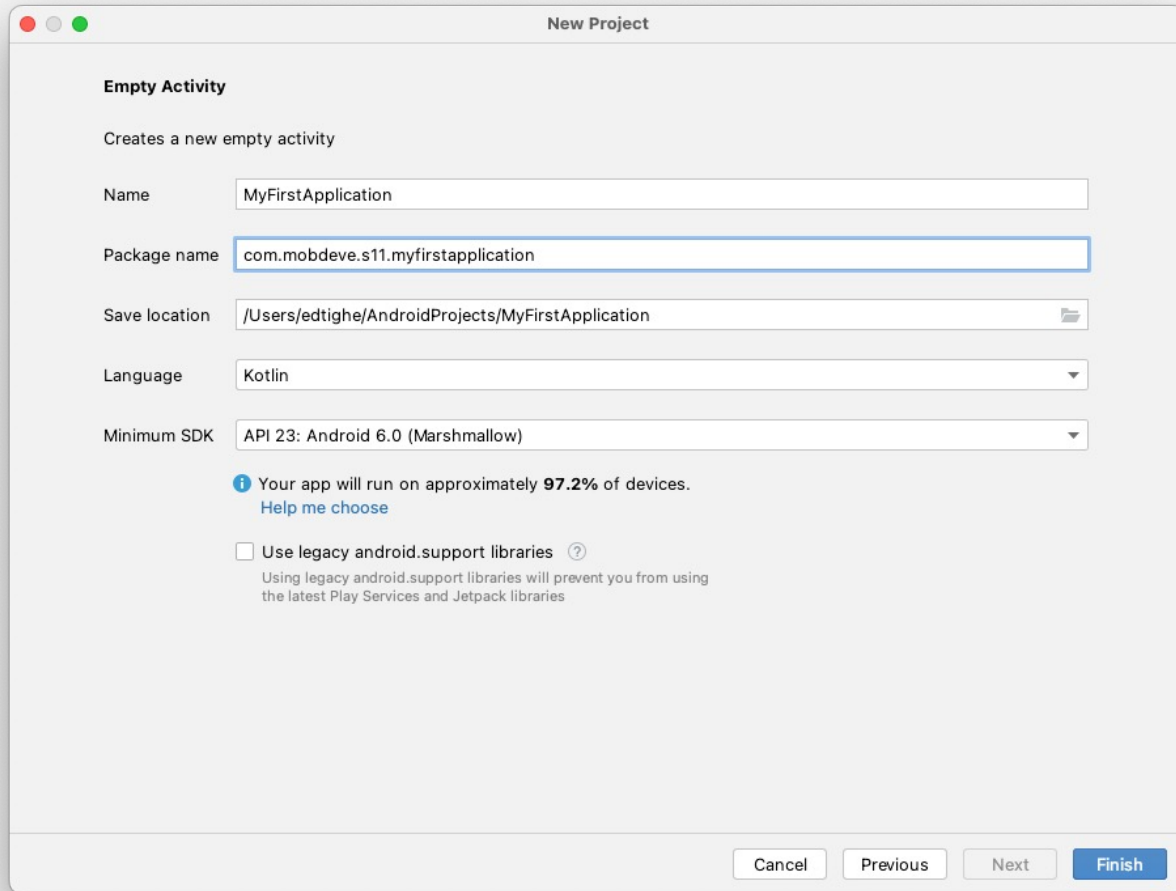
Android Studio

- Several templates are provided; however, we'll default to using the **Empty Activity** in most cases
 - Other templates have too much included
 - The Empty Activity starts us off with a blank canvas



Also, we'll discuss this further latter one, but know that an "Activity" can be thought of as a screen

Android Studio



New Project

Empty Activity

Creates a new empty activity

Name: MyFirstApplication

Package name: com.mobdeve.s11.myfirstapplication

Save location: /Users/edtighe/AndroidProjects/MyFirstApplication

Language: Kotlin

Minimum SDK: API 23: Android 6.0 (Marshmallow)

i Your app will run on approximately **97.2%** of devices.
[Help me choose](#)

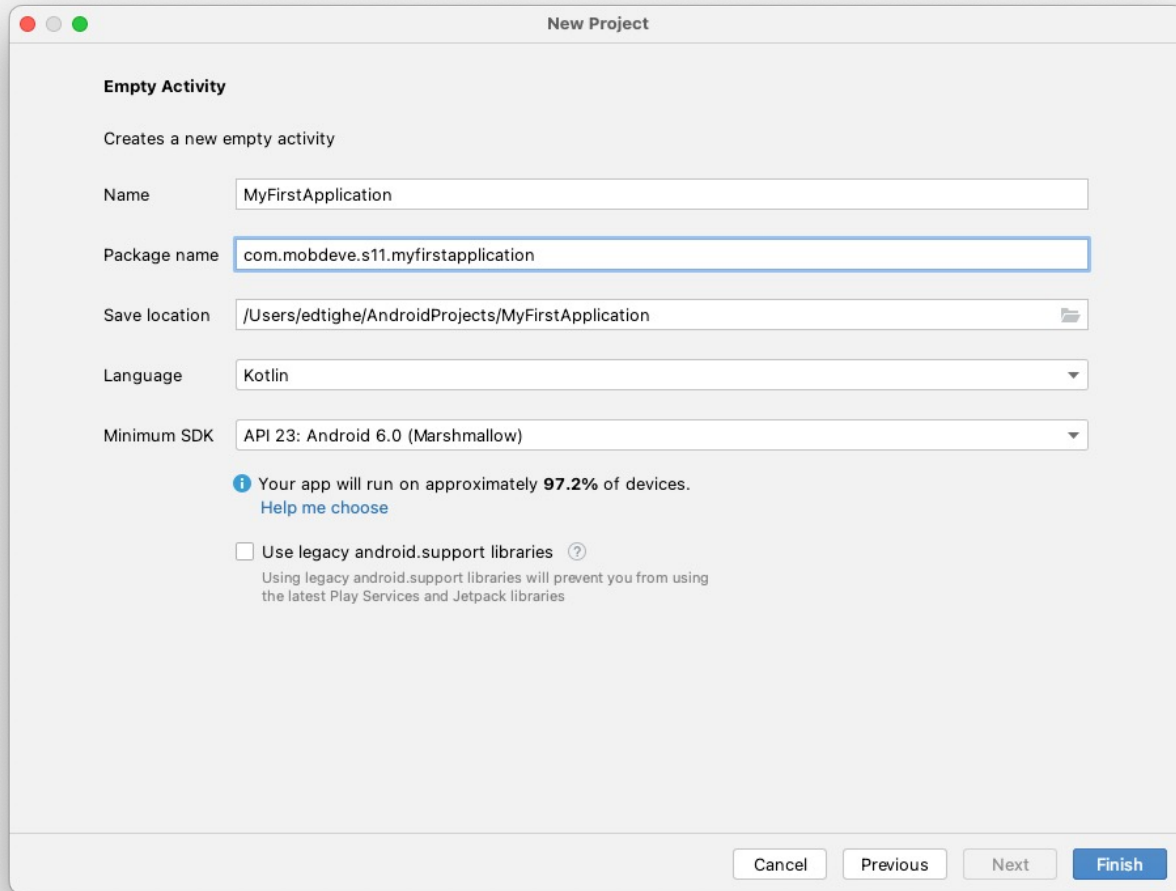
☐ Use legacy android.support libraries **?**
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Cancel Previous Next **Finish**

- **Name**

- Rather standard...
- Refers to the application's name

Android Studio

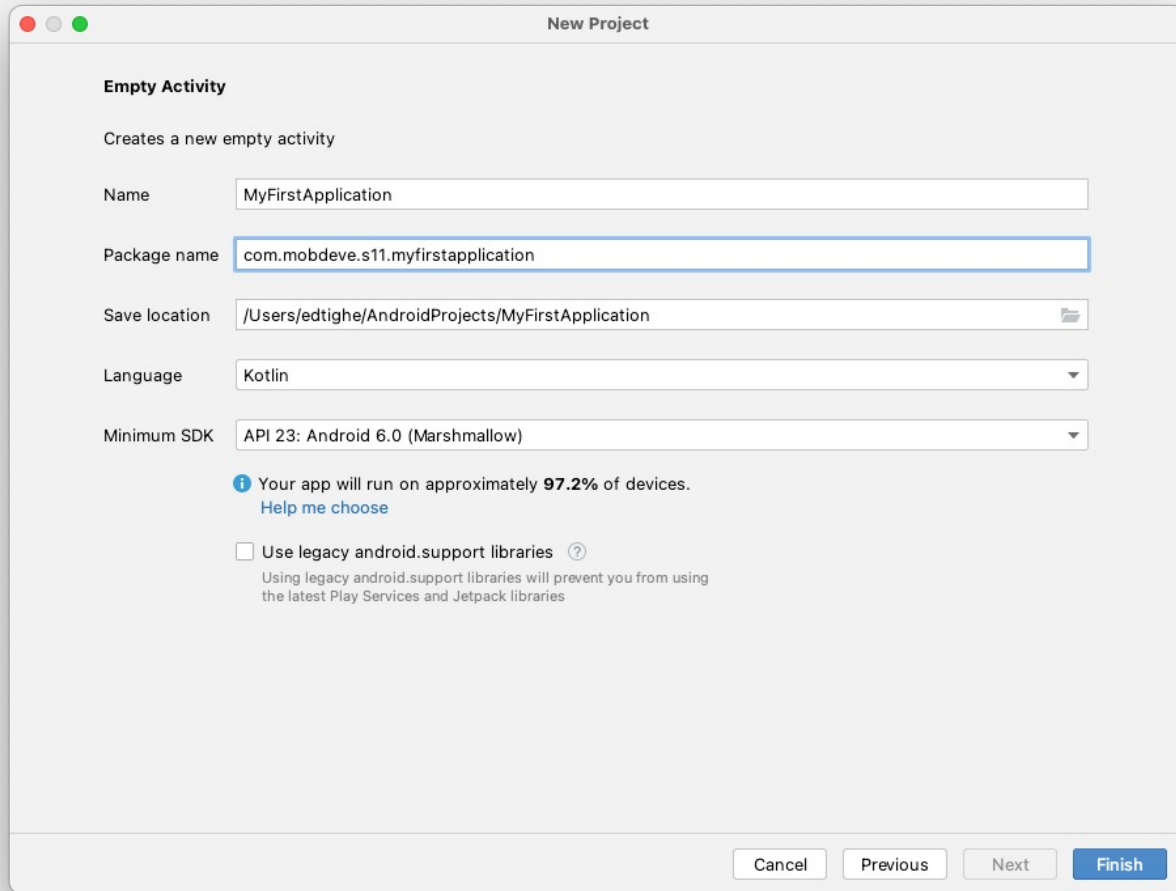


- **Package Name**

- This is a unique ID for your app
- Typical naming convention is the reverse of a domain name
 - com.mobdeve.app1 can be read as the app1 package of mobdeve.com

Additionally, you can add another layer that makes the ID unique, like your name
com.mobdeve.tighee.app1

Android Studio



New Project

Empty Activity

Creates a new empty activity

Name: MyFirstApplication

Package name: com.mobdeve.s11.myfirstapplication

Save location: /Users/edtighe/AndroidProjects/MyFirstApplication

Language: Kotlin

Minimum SDK: API 23: Android 6.0 (Marshmallow)

i Your app will run on approximately **97.2%** of devices.
[Help me choose](#)

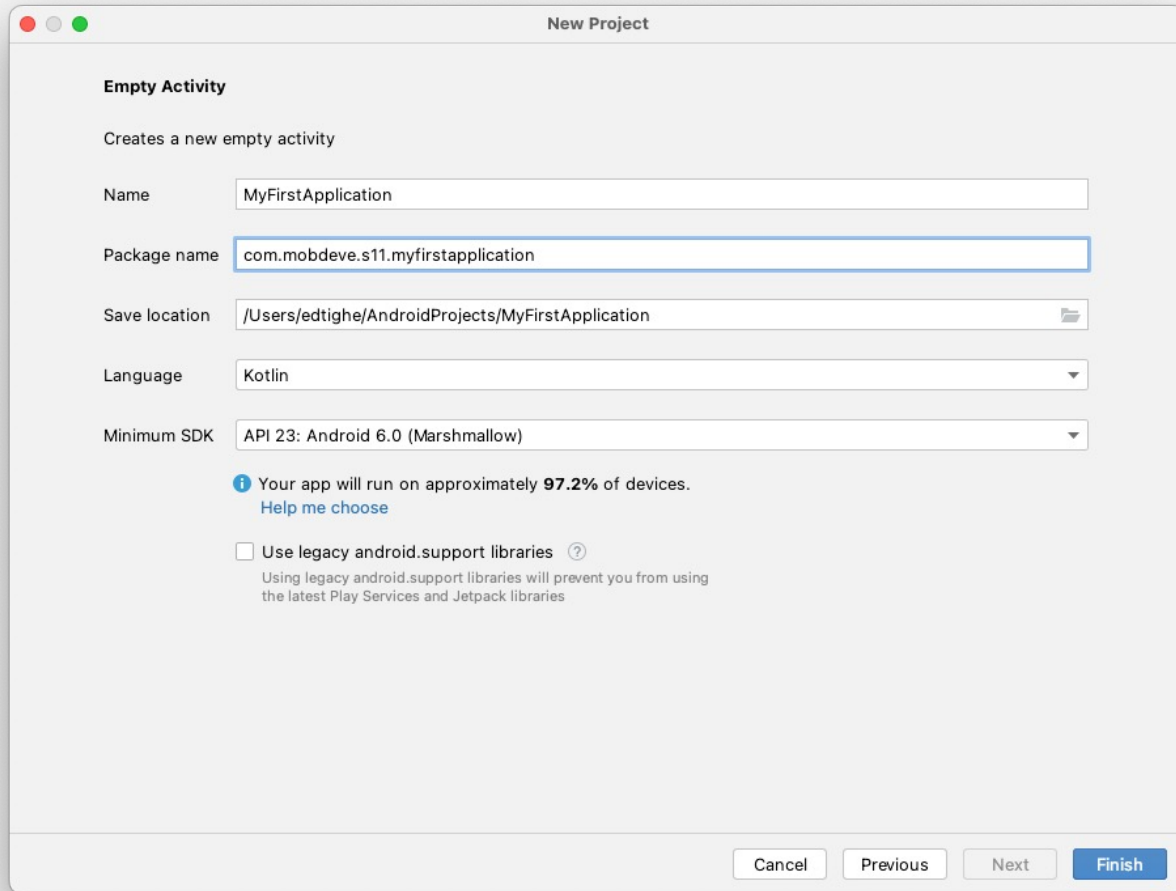
☐ Use legacy android.support libraries **?**
Using legacy android.support libraries will prevent you from using the latest Play Services and Jetpack libraries

Cancel Previous Next **Finish**

- **Save Location**

- For MOBDEVE, consider saving all your projects in a specific folder to keep things centralized

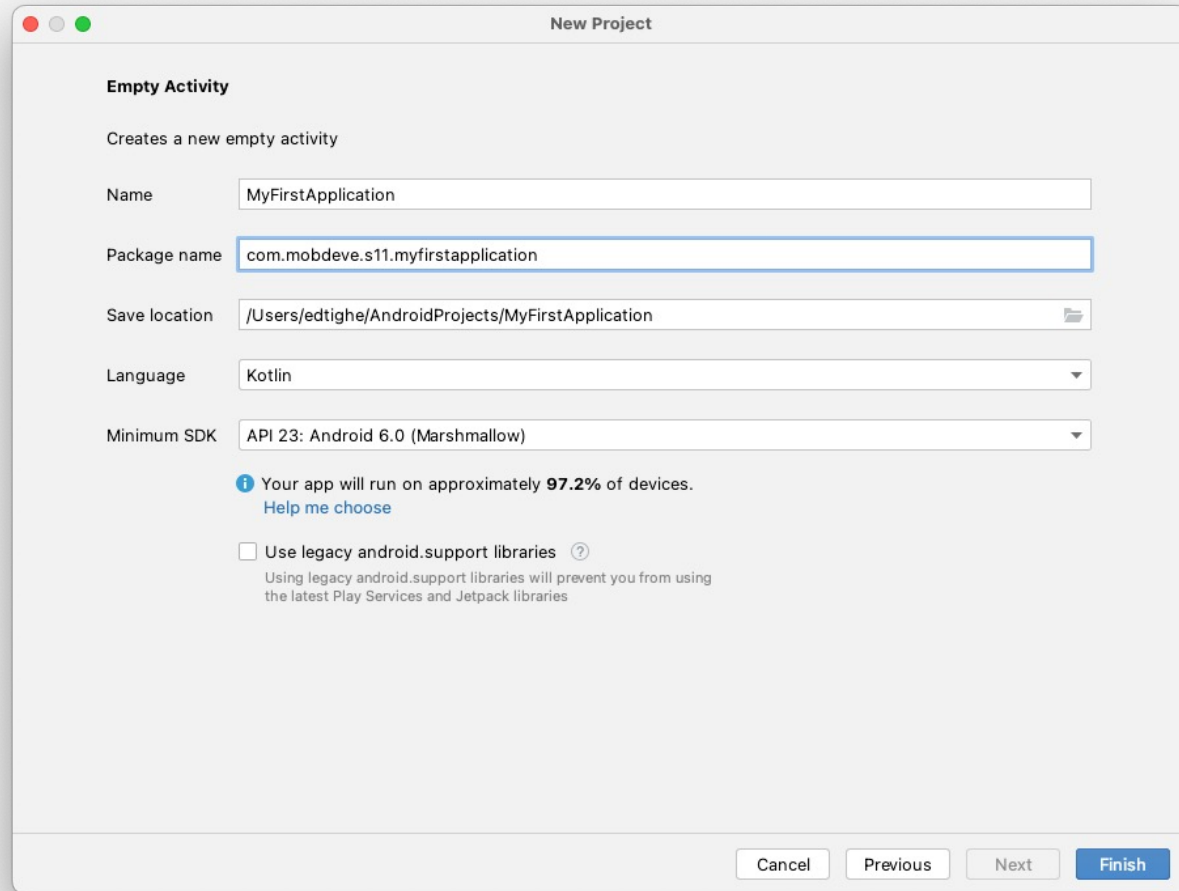
Android Studio



- Language

- You can select from either Java or Kotlin
- Either option still allows you to run the other language
 - For Java projects, you'll just have to allow Kotlin to be used
- You can also ask Android Studio to convert code for you

Android Studio



- **Minimum SDK**

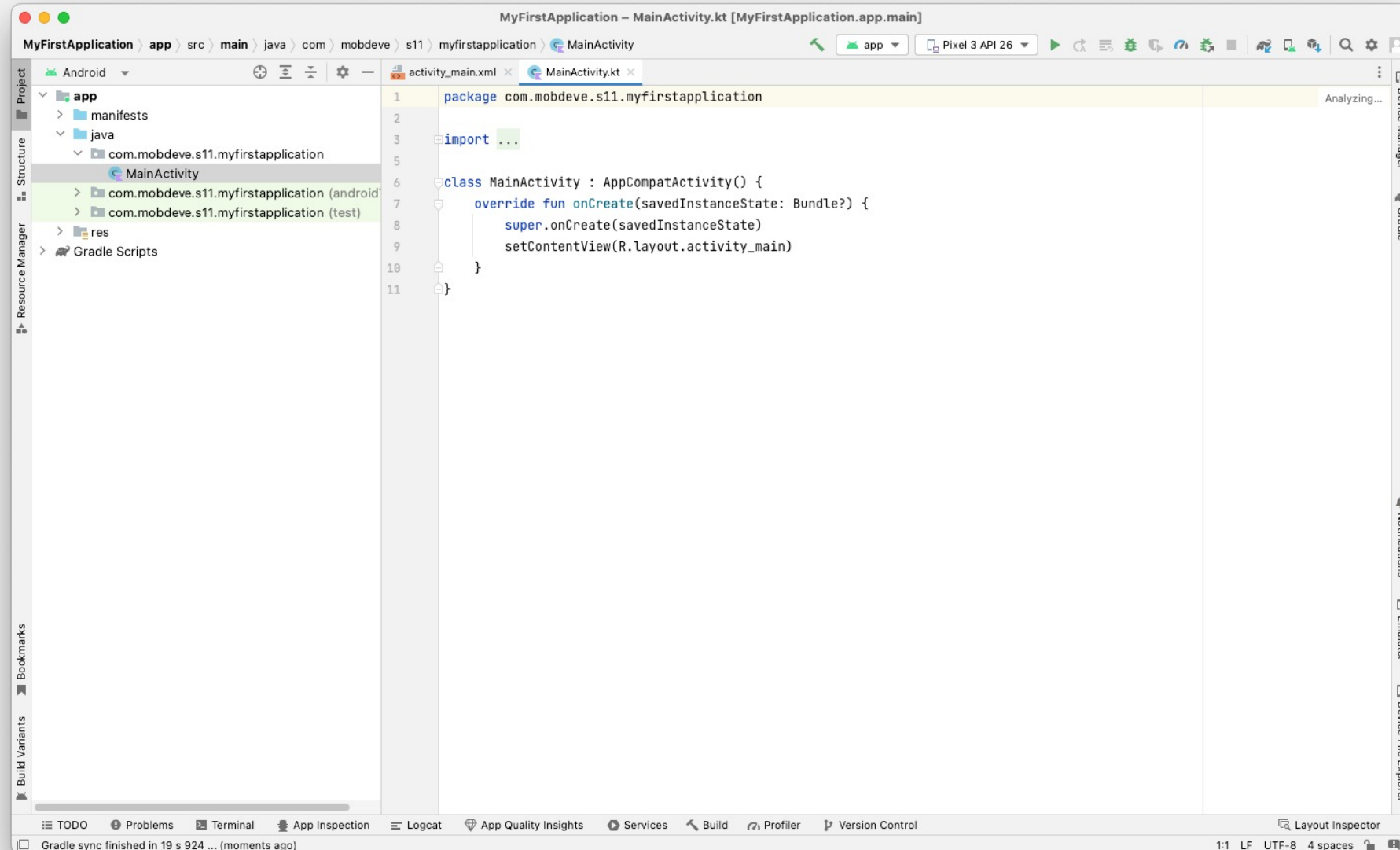
- This refers to the oldest version of Android your application will be able to run on
- This is different from the Target SDK, which your app will be tested and designed for

For our course, please use API 23

Android Studio

- After finishing the initial configuration, Android Studio should start to piece things together
 - This might take awhile for some, mainly because of the Gradle build (especially if its being built for the first time)
- Additionally, you might be prompted to download resources

Android Studio



Android Studio

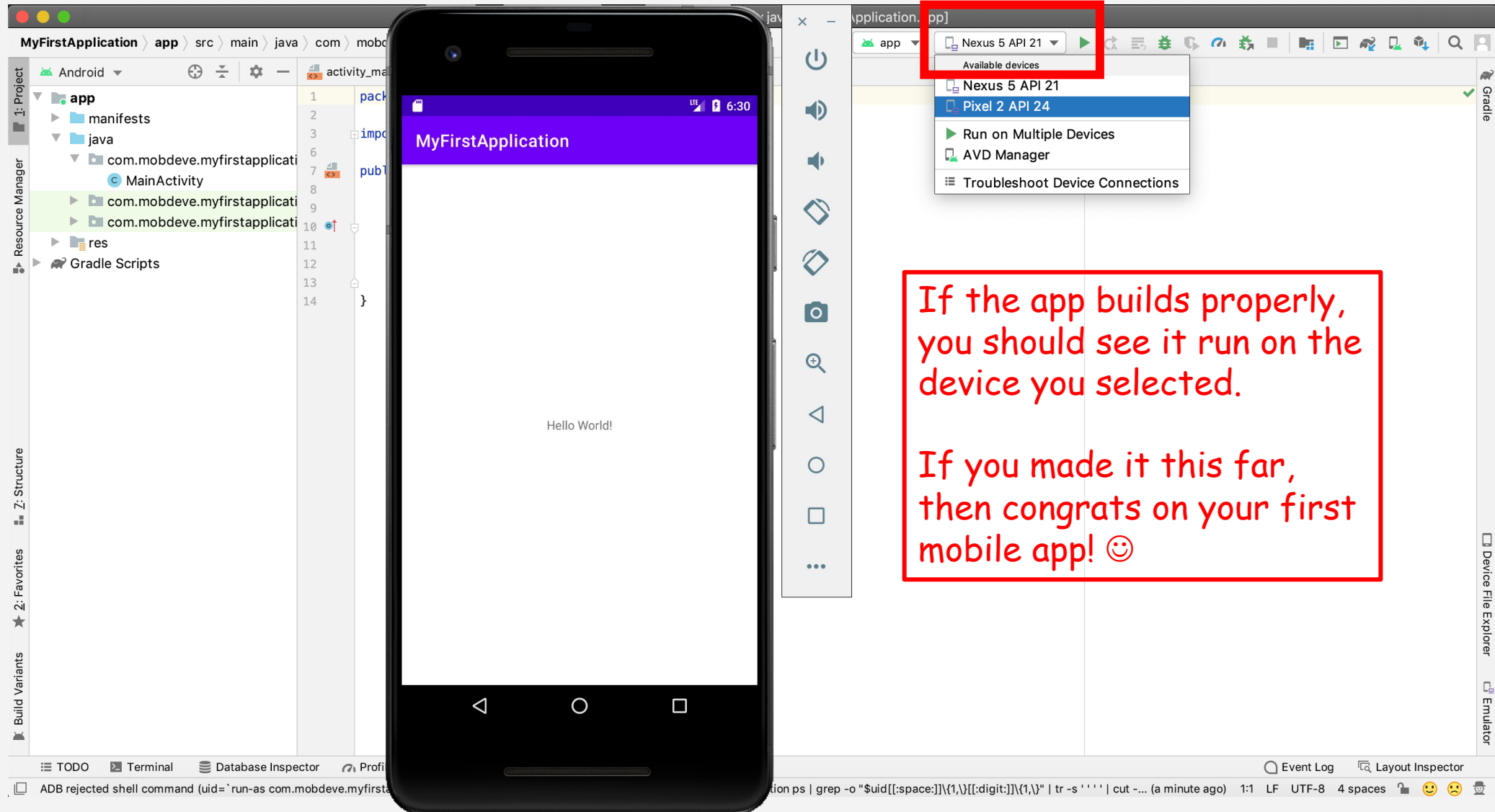
- Before running the base app, we need to make sure we have either our **physical device** ready or create a **virtual device**
- If you have a physical device, make sure to enable the developer's mode
 - Follow the steps indicated here:
<https://developer.android.com/studio/debug/dev-options>

Android Studio

- If you don't, make sure to set up a virtual device
 - Locate the **Android Virtual Device (AVD) Manager** and choose *create a virtual device*
 - Select a phone device (e.g. Pixel 2 or Nexus 5X)
 - Select a system image (e.g. API 23, 26)
 - If you don't have the appropriate system image downloaded already, this can be a long wait depending on your internet...
- Finish the setup

Android Studio

Find the device you which to run your app on and press the play button

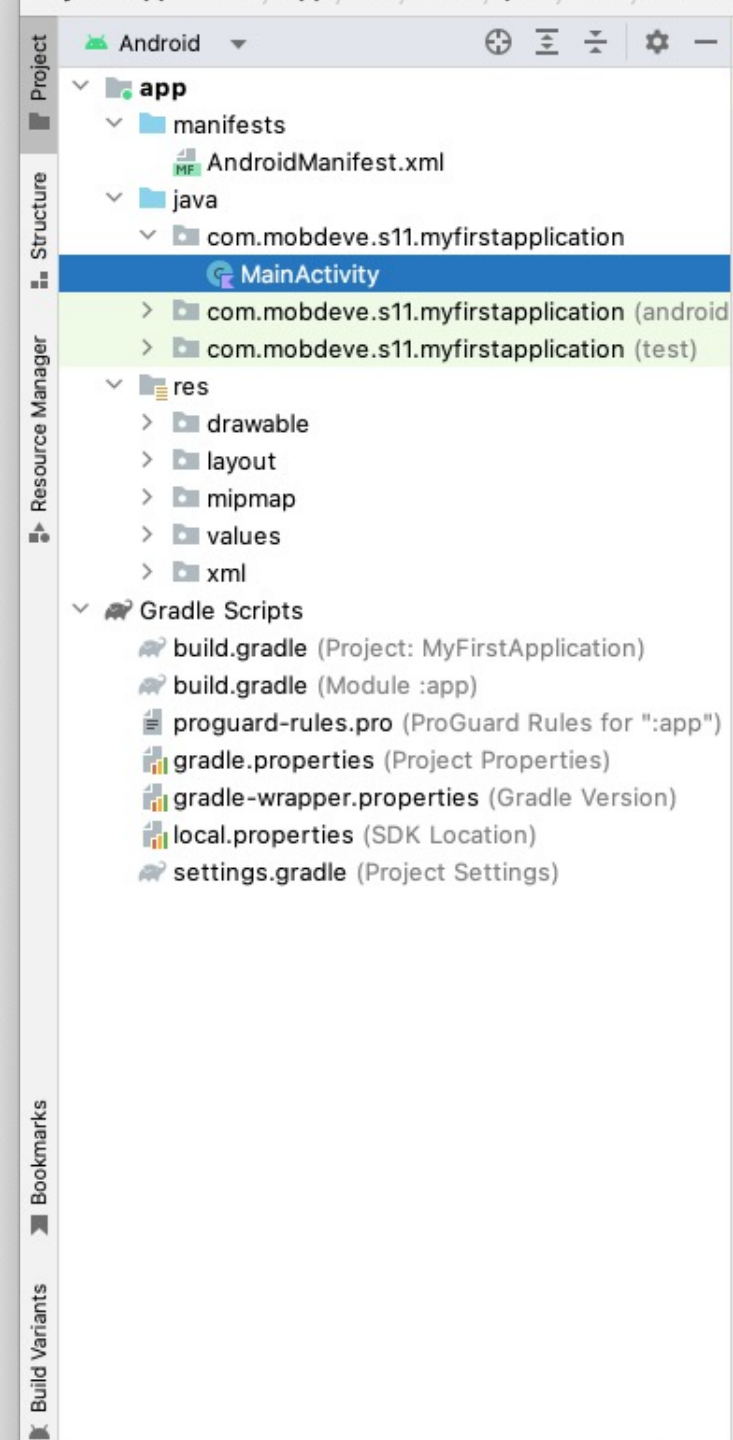


If the app builds properly, you should see it run on the device you selected.

If you made it this far, then congrats on your first mobile app! 😊

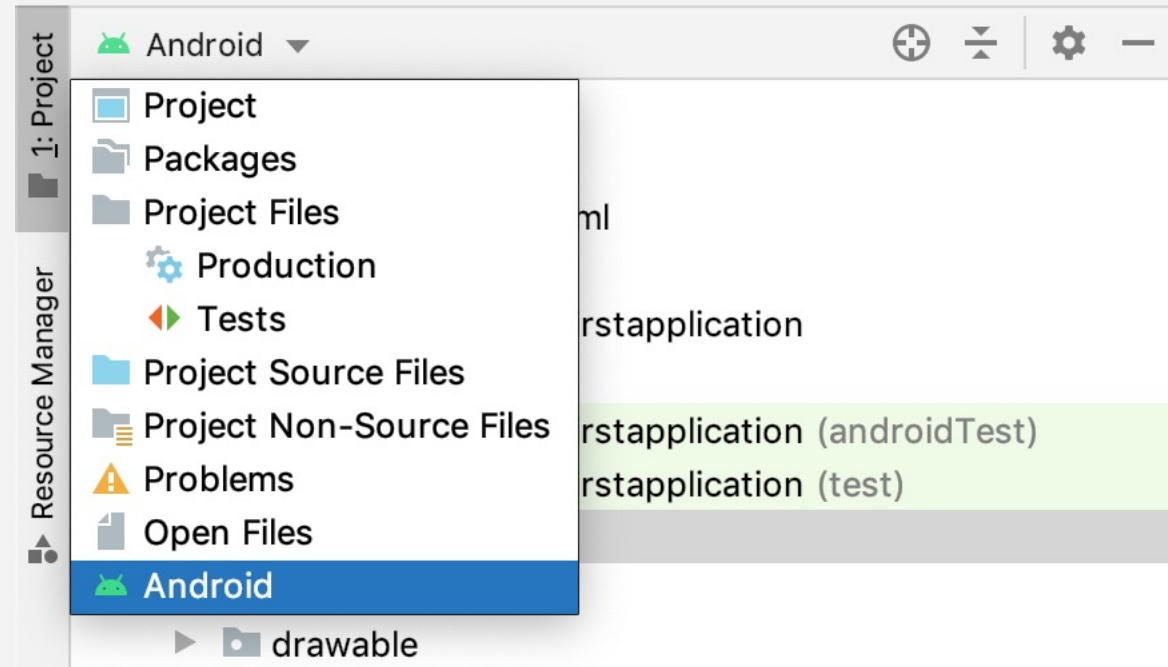
Getting Familiar with the IDE

- On the left, you'll see the **project files**



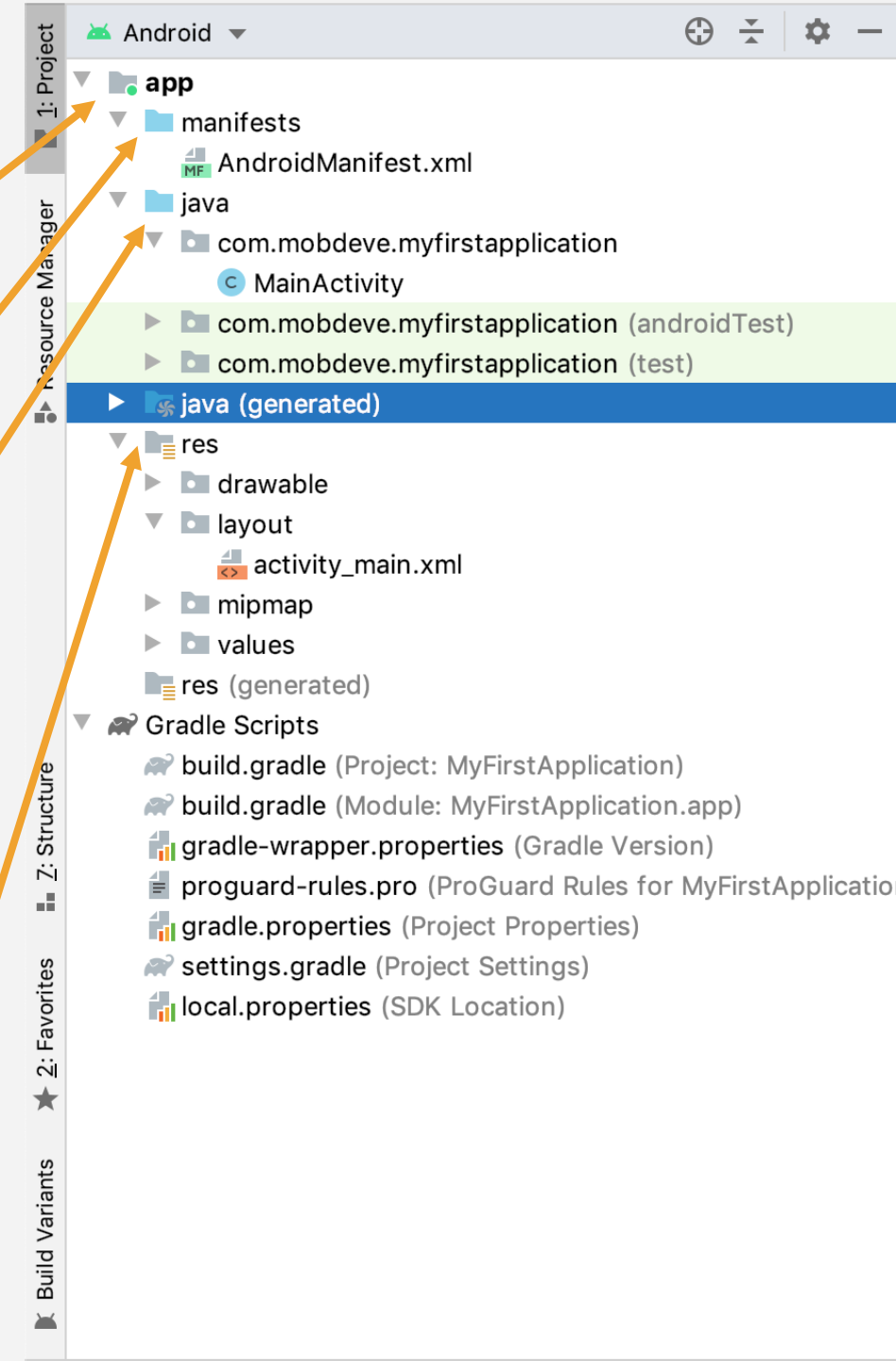
Getting Familiar with the IDE

- One of the things to note is that you're (usually) on **Android view** by default
 - Organizes / abstracts files
- Another view you might use is the **Project view**
 - Shows the entire project's file structure



Getting Familiar with the IDE

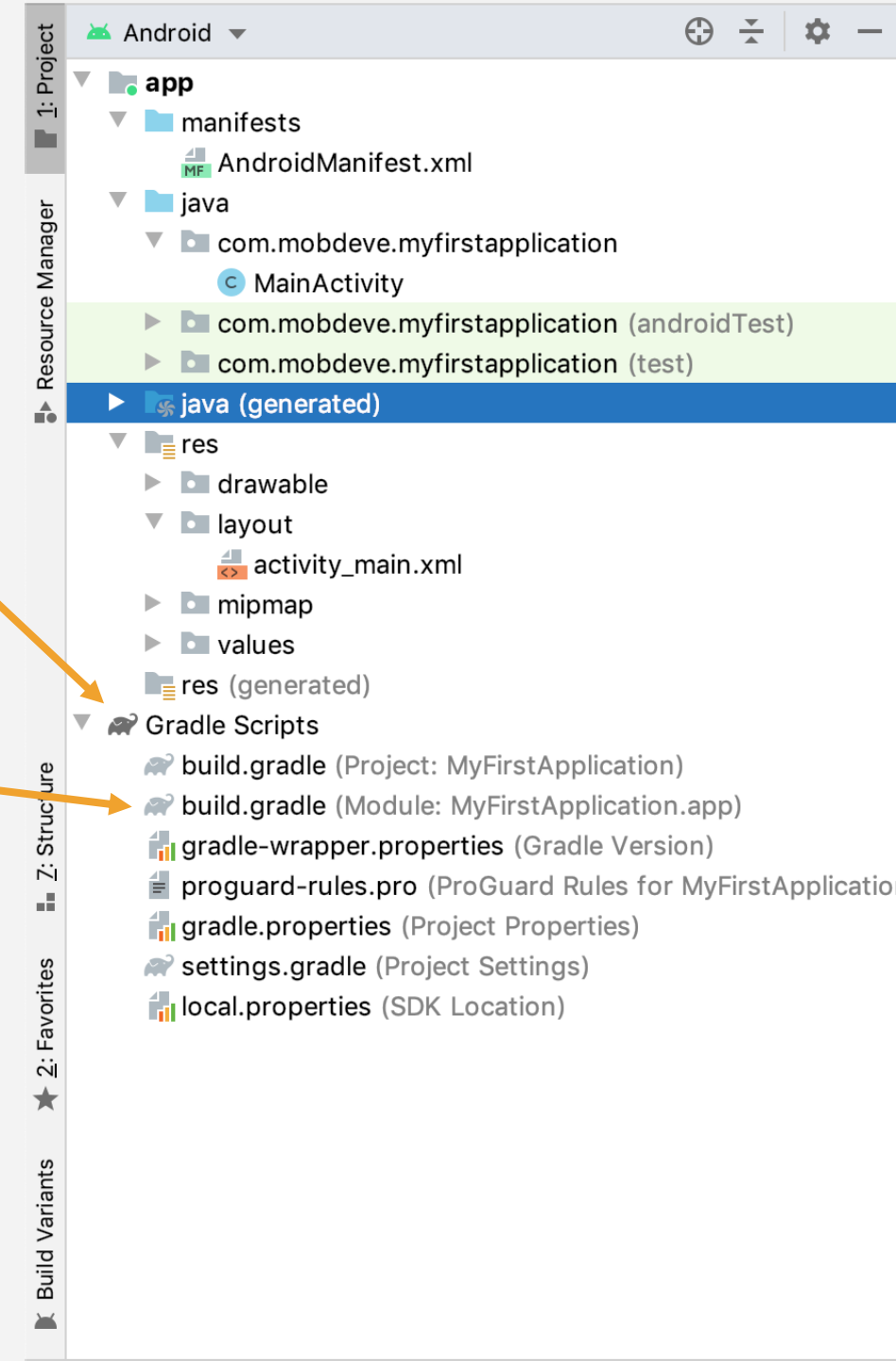
- **app** contains our application source code
 - **manifests** contains the Android Manifest, which is like a “config” file for the app
 - **java** contains the Java source code
 - **res** contains XML resource files that contain info about the layout



Getting Familiar with the IDE

- **Gradle Scripts** contains scripts that help compile the source code for installation on a device
 - The build.gradle that is normally modified is specific to the app
- Building the gradle does require internet connection when updating dependencies

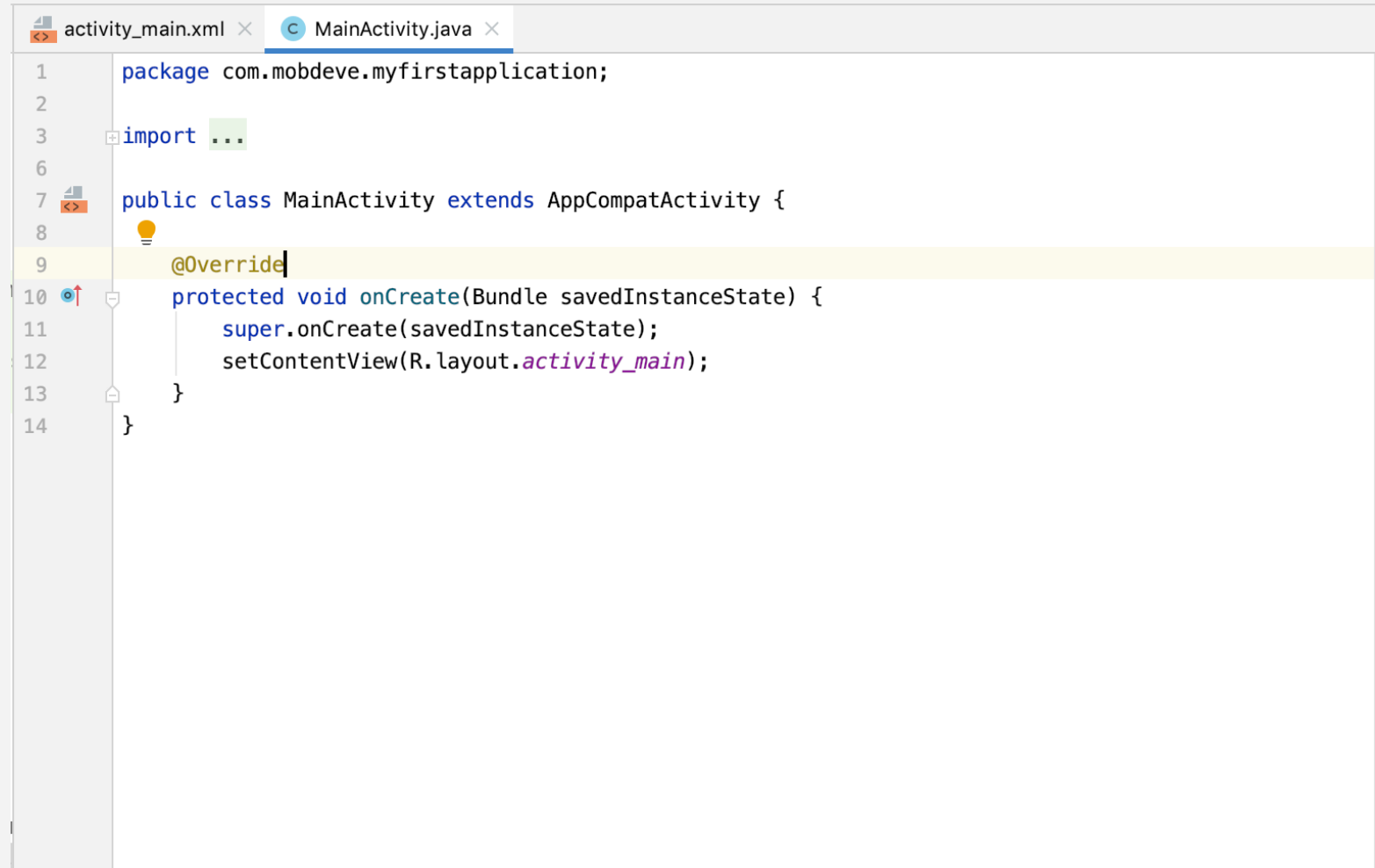
There is a work around where you can build the gradle once and work in offline mode



Getting Familiar with the IDE

- Editor area

- This doesn't need much explanation
- However, there are interesting ways to use the editor when designing the UI
 - Will tackle in the future

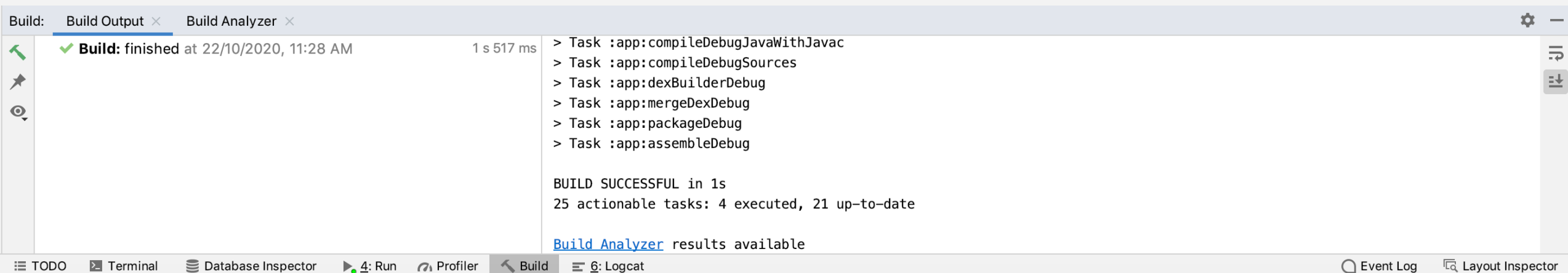
A screenshot of an IDE editor window. The top bar shows two tabs: 'activity_main.xml' and 'MainActivity.java'. The 'MainActivity.java' tab is active. The code is as follows:

```
1 package com.mobdeve.myfirstapplication;
2
3 import ...
4
5
6
7 public class MainActivity extends AppCompatActivity {
8     //
9     @Override
10    protected void onCreate(Bundle savedInstanceState) {
11        super.onCreate(savedInstanceState);
12        setContentView(R.layout.activity_main);
13    }
14 }
```

The line containing '@Override' is highlighted in yellow. There is a lightbulb icon above it, indicating a suggestion or tip. The left margin shows line numbers 1 through 14.

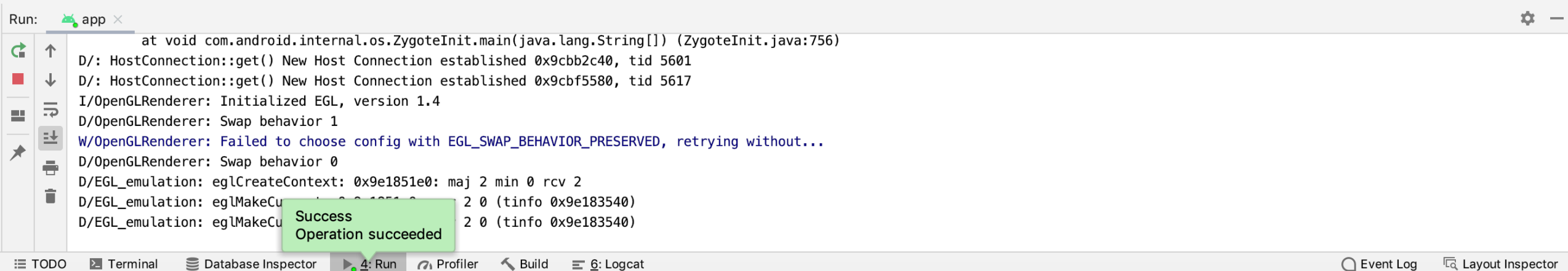
Getting Familiar with the IDE

- At the bottom of Android Studio, you'll find different tools for monitoring an application
 - If your app builds properly, it should show in **Build**
 - Else, you'll see compile time errors



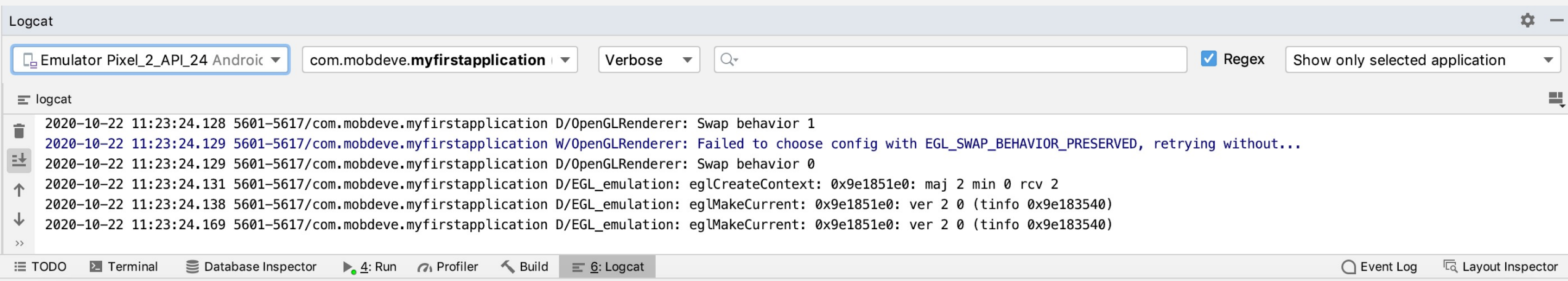
Getting Familiar with the IDE

- At the bottom of Android Studio, you'll find different tools for monitoring an application
 - Runtime errors, on the other hand, are shown in the **Run** tab

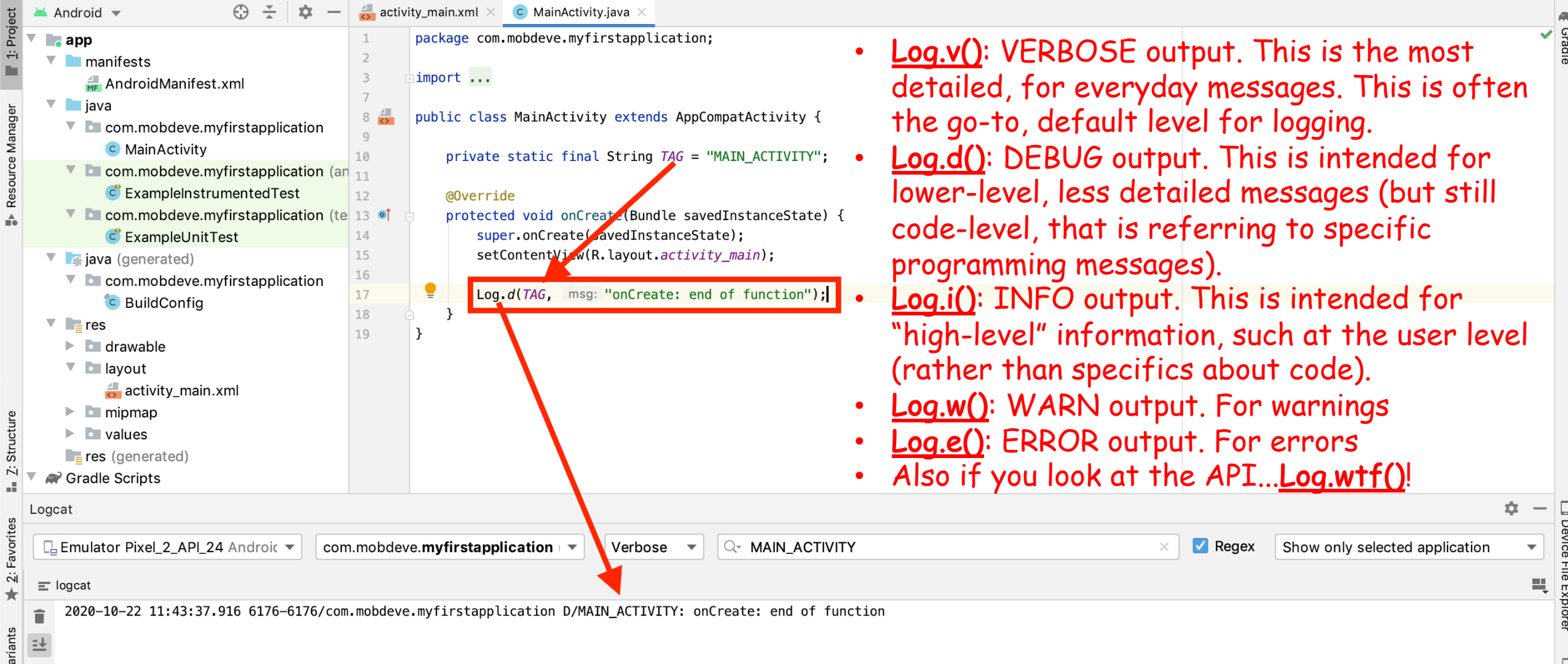


Getting Familiar with the IDE

- At the bottom of Android Studio, you'll find different tools for monitoring an application
 - And for most debugging, stick with **Logcat**
 - Here, you'll see the logs of most (if not all) applications running



Getting Familiar with the IDE



The screenshot shows an IDE with the following components:

- Project Explorer:** Shows the project structure with folders like `manifests`, `java`, `res`, and `Gradle Scripts`. The `MainActivity` class is highlighted.
- Code Editor:** Displays the `MainActivity.java` file. The `onCreate` method is visible, and the `Log.d(TAG, msg: \"onCreate: end of function\");` line is highlighted with a red box. A red arrow points from this line to the Logcat window.
- Logcat:** Shows the log output for the application. The log message `2020-10-22 11:43:37.916 6176-6176/com.mobdeve.myfirstapplication D/MAIN_ACTIVITY: onCreate: end of function` is displayed.

On the right side of the image, there is a list of log levels and their descriptions:

- Log.v(): VERBOSE output. This is the most detailed, for everyday messages. This is often the go-to, default level for logging.
- Log.d(): DEBUG output. This is intended for lower-level, less detailed messages (but still code-level, that is referring to specific programming messages).
- Log.i(): INFO output. This is intended for "high-level" information, such as at the user level (rather than specifics about code).
- Log.w(): WARN output. For warnings
- Log.e(): ERROR output. For errors
- Also if you look at the API...Log.wtf()!

Getting Familiar with the IDE



Build / make project

Device to use /
Access to AVD
Manager

Build and run
application


Run selected app
in debugging
mode

Sync Gradle

SDK manager

If there's still time...

- Add the following to the onCreate function
 - The code to the side is shown in Java
 - There is minimal different with how its written in Kotlin
- What happens? 😊



```
activity_main.xml x MainActivity.java x
1 package com.mobdeve.myfirstapplication;
2
3 import ...
8
9 public class MainActivity extends AppCompatActivity {
10
11     private static final String TAG = "MAIN_ACTIVITY";
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17
18         Toast t = Toast.makeText(
19            (getApplicationContext(),
20                 text: "Naalala mo pa ba yung PhilHealth issue?",
21                 Toast.LENGTH_LONG
22             );
23         t.show();
24
25         Log.d(TAG, msg: "onCreate: end of function");
26     }
27 }
```

The screenshot shows an IDE with two tabs: 'activity_main.xml' and 'MainActivity.java'. The 'MainActivity.java' tab is active, displaying the following Java code. A red rectangular box highlights the Toast message creation and display logic, which is added to the onCreate method. The code includes a package declaration, an import statement, a class declaration extending AppCompatActivity, a static TAG, and an overridden onCreate method. The highlighted code creates a Toast with the text 'Naalala mo pa ba yung PhilHealth issue?' and displays it for a long duration. The onCreate method also logs a message at the end.

Any questions? 😊

Next Session...

- We'll run through a **hands-on exercise**
 - Two exercises if we have time
 - Focus is on creating basic views and linking them to our code

Thanks everyone!

See you next meeting!

Android development with notepad-like editor and command line



4



My ideal is to write my code in an editor like Notepad and supplement with the command line! How can I compile my code in this environment? How do I update my R.java file or build my project's configuration?

**I DONT KNOW WHO YOU
ARE BUT I LL FIND YOU**



**AND INSTALL ANDROID STUDIO ON
YOUR PC**

memegenerator.net