Android and Android Studio

What is Android?

It is an operating system based upon a modified version of the Linux kernel, and other open source software, which can run the following devices, as well as others: watches, smart glasses, cars, homes: (heating/cooling, lighting, alarms, appliances, sprinklers, pools, home theaters, etc.), cameras, tvs, phones, games, and even (smart) mirrors!

What is Android Studio?

Android Studio was developed by Google as their "official" integrated development environment (IDE)/tool for developing Android applications. Android apps 1) provide multiple entry points, and 2) adapt to different devices.

Multiple Entry Points:

- Activity: app component that provides user interface (UI)
- "Main" activity starts when user taps app's icon
- Can also direct user to activity from elsewhere (e.g., notification, different app, etc.)

App components are application building blocks. Each component is an app entry point, via system or user. Four types of app components: https://developer.android.com/guide/components/fundamentals

1. Activities
2. Services
3. Broadcast receivers
4. Content providers

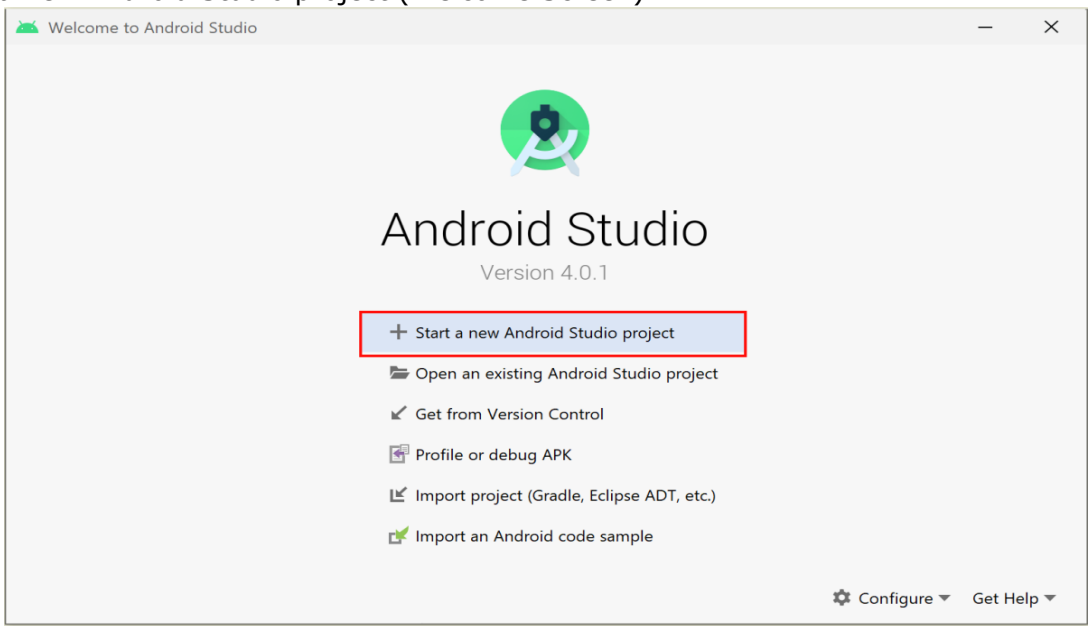
Note: Each component type serves a discrete purpose, with distinct lifecycles (i.e, when it is created and destroyed).

Adapt to Different Devices:

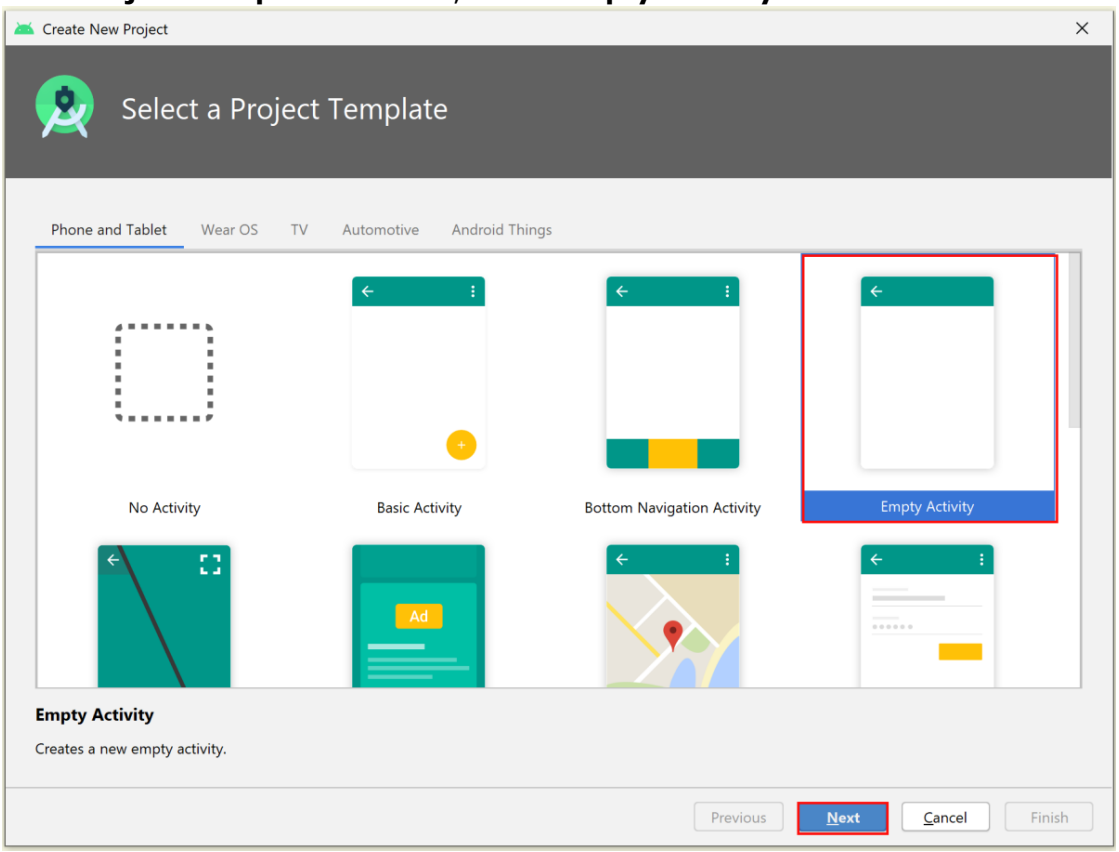
- Provide different resources for different devices
- Different layouts for different screen sizes

Create an App (Project)!

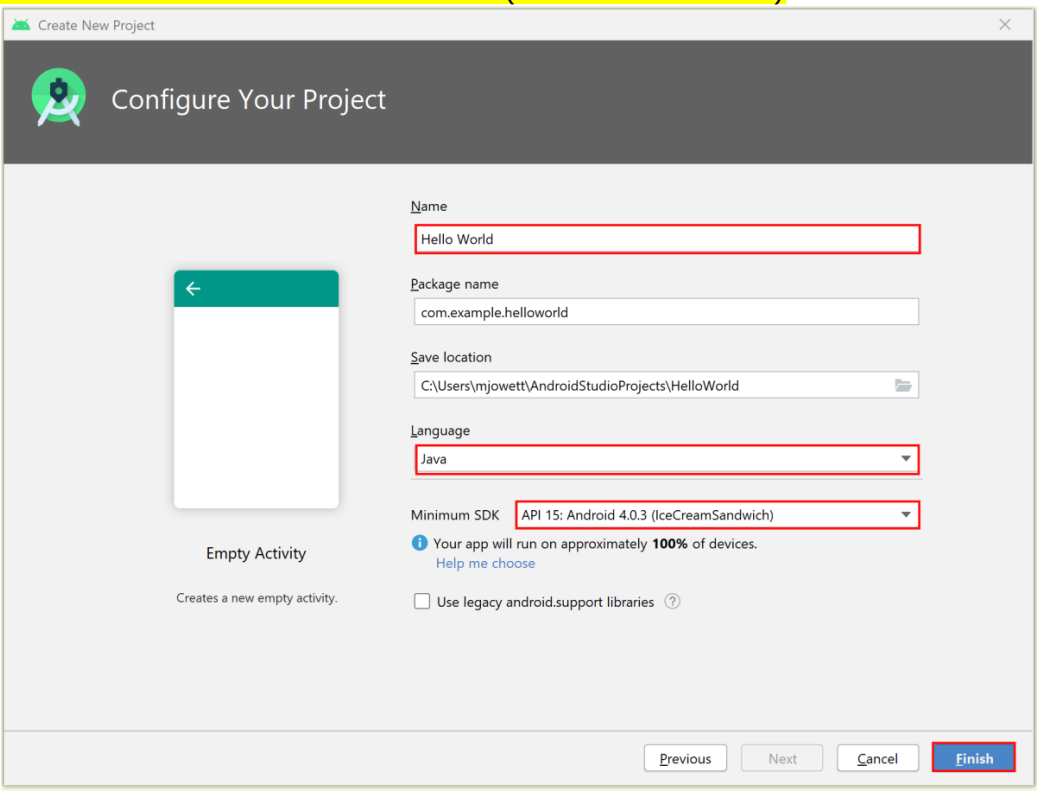
1. Install Android: https://developer.android.com/studio/
2. Start a new Android Studio project (Welcome Screen)



3. "**Select a Project Template**" window, select **Empty Activity** and click **Next**.



4. "**Configure Your Project**" window:
    a. Name: "Hello World"
    b. Language: ***Java*** (***NOT*** Kotlin!)
    c. Minimum SDK: API 15: Android 4.0.3 (IceCreamSandwich)



5. **Finish**

Project window: View > Tool Windows > Project (*Be sure* "**Android**" is selected in drop-down menu.)

app > java > com.example.myfirstapp > MainActivity
App entry point. Build and run app: system launches instance of this Activity and loads its layout.

app > res > layout > activity_main.xml
XML file defines layout for activity's user interface (UI). Contains a TextView element (widget) with text "Hello, World!"

app > manifests > AndroidManifest.xml
Manifest file describes characteristics of app, and defines its components.
Note: Manifest files define application operational requirements (i.e., structure, metadata, components).

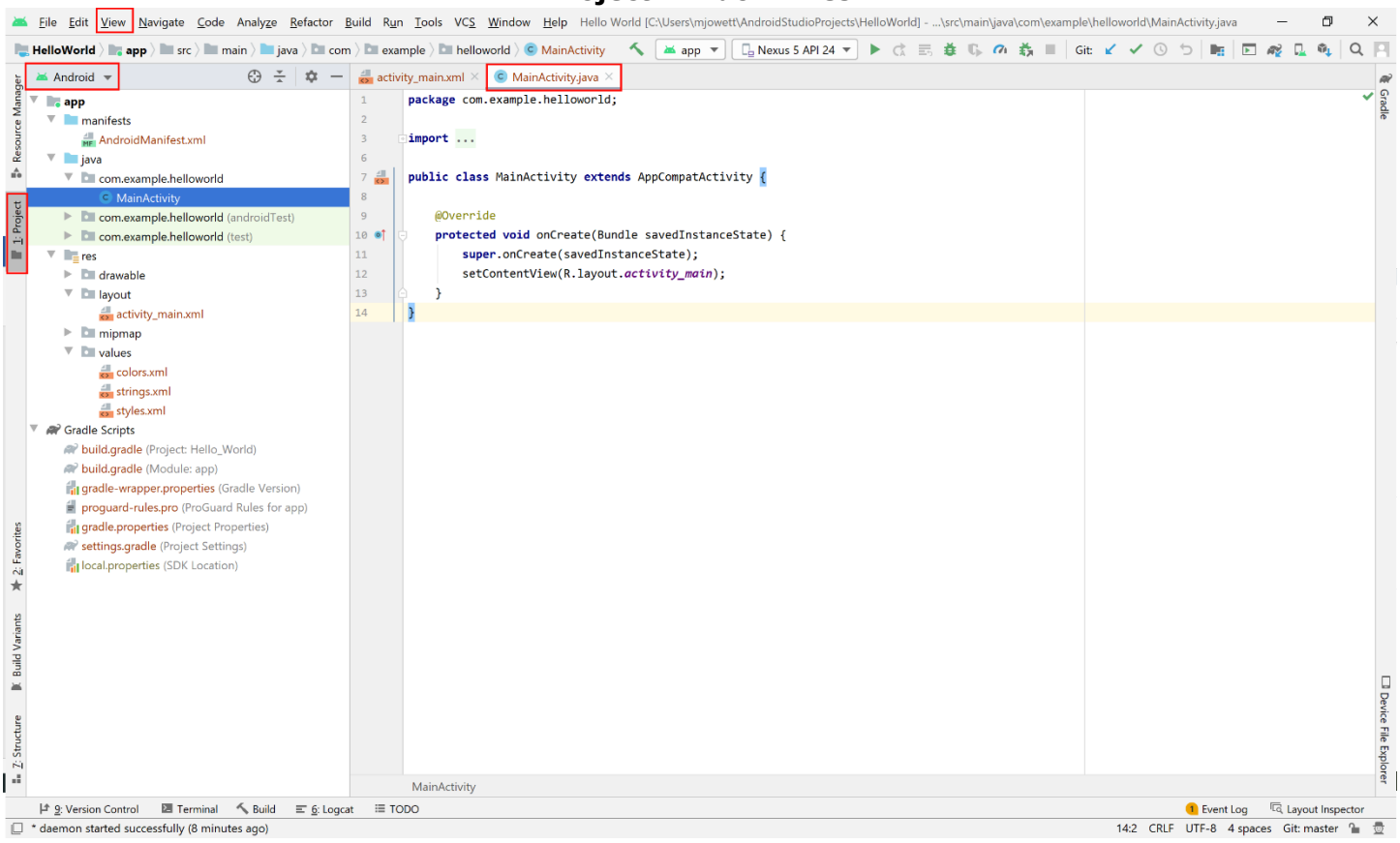
Gradle Scripts > build.gradle
Two files named "build.gradle":

1. One for project, "Project: Hello_World," and
2. One for "Module: app."

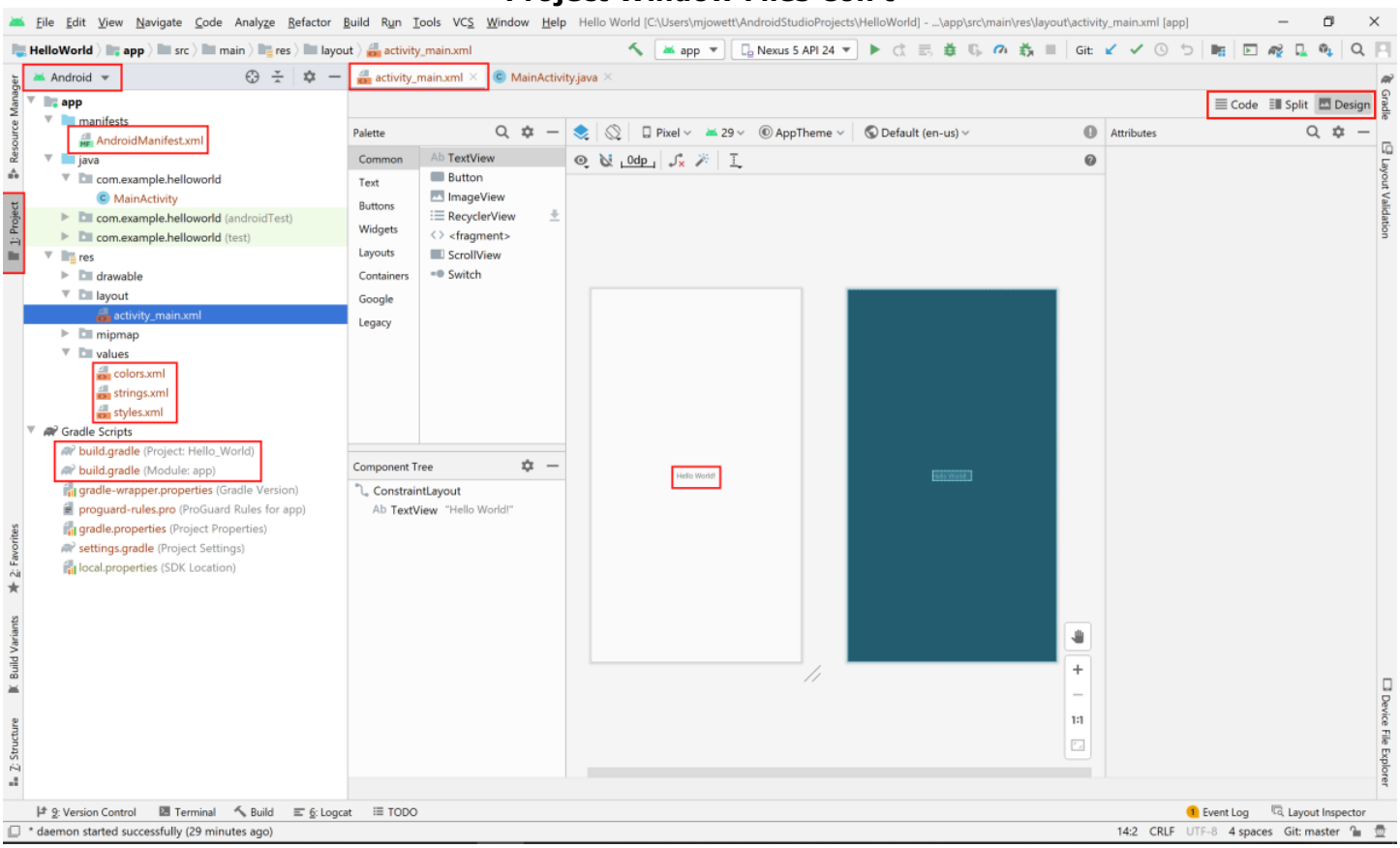
Note: *Gradle* is build system Android Studio use to compile and package apps.

Note: Each module has its own build.gradle file. This project has just one module. When a project has more than one module, use each module's build.file to control how the Gradle plugin builds app.

Project Window Files



Project Window Files Con't



Other Files and Folders:

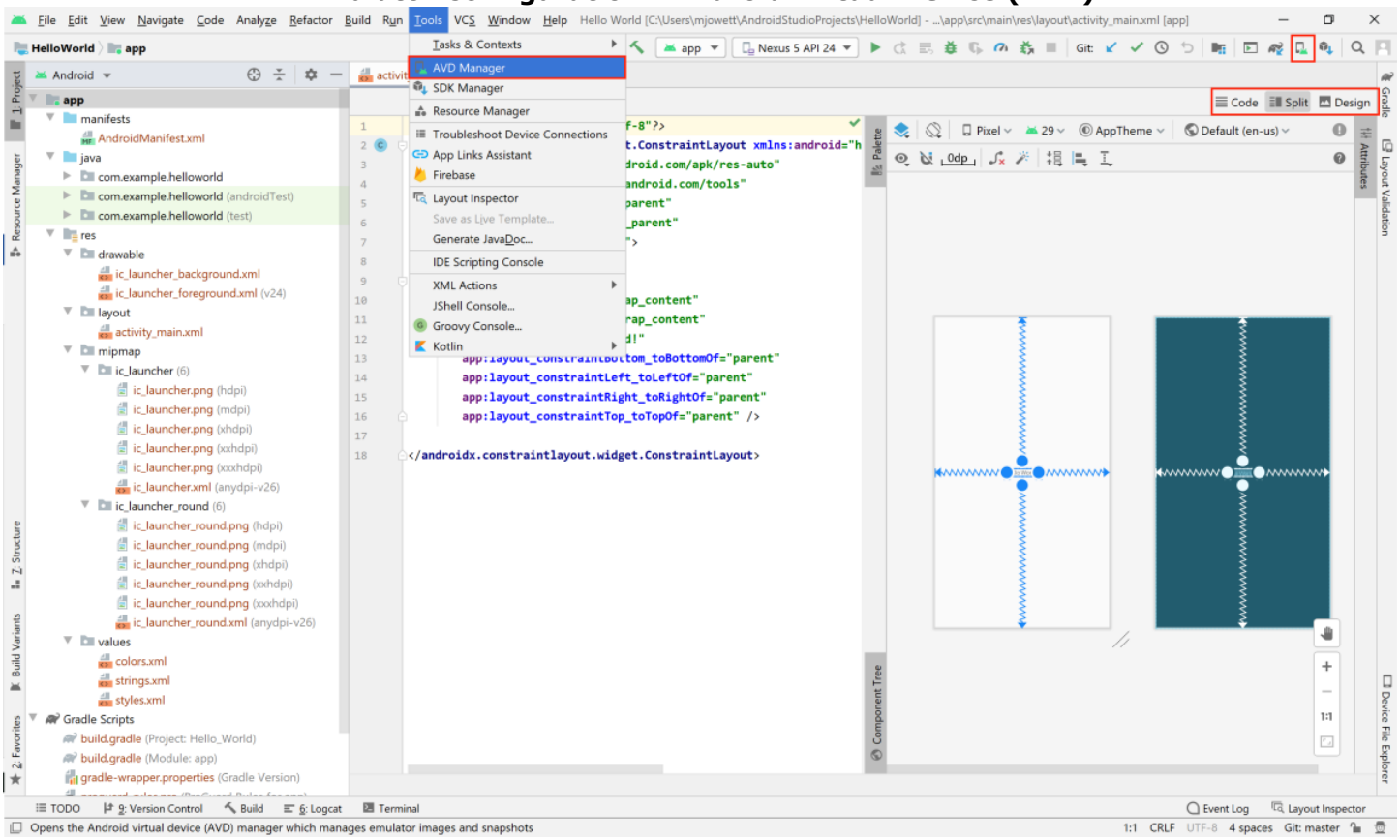
- **res** folder:
  - Stores app resource folders and files.
  - ***NO*** capital letters allowed in file names in **res** folder!
- **drawable** folder:
  - Stores images (.png): multiple subfolders (different size images for different resolutions).
  - Also, contains <u>two</u> .xml files:
    1. Describes how to display launcher image background.
    2. Describes how to display launcher image foreground.
- **mipmap** folder:
  - Contains <u>two</u> folders for <u>two</u> different types of launcher icons.
  - Each contains <u>six</u> .png files for different screen resolutions.
  - Android automatically selects appropriate image for selected device.
- **layout** folder: Contains all layouts for app user interface.
- **values** folder: contain global configuration data for app (minimizes hard-coding values)
  - colors.xml: element color values in layout
  - strings.xml: text values
  - styles.xml: defines layout look and format
  - dimens.xml: element display size
  - arrays.xml: string array element values
  - ids.xml: IDs that cannot be reused by layout elements

Emulator Configuration - Android Virtual Device (AVD):

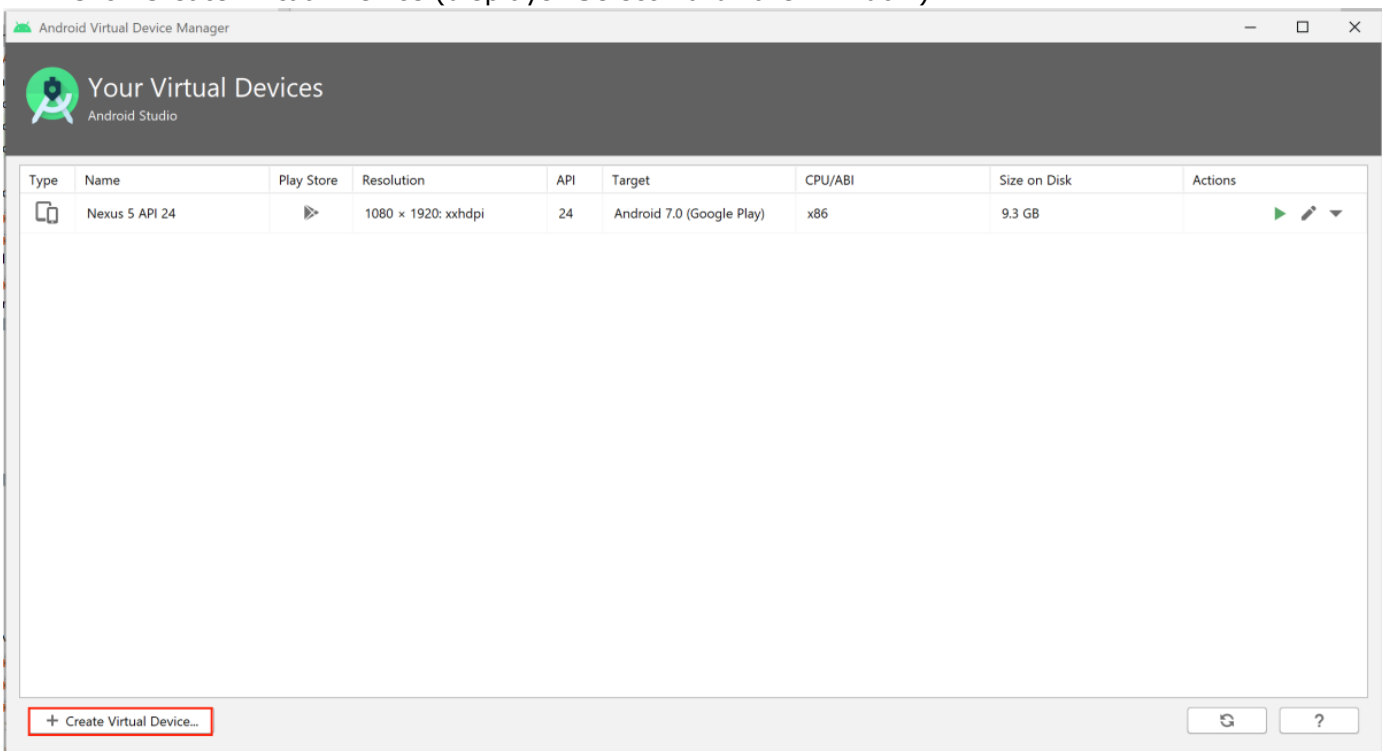
Note: multiple AVDs may be set up for testing.

1. Tools > AVD Manager (displays "Your Virtual Devices" window)

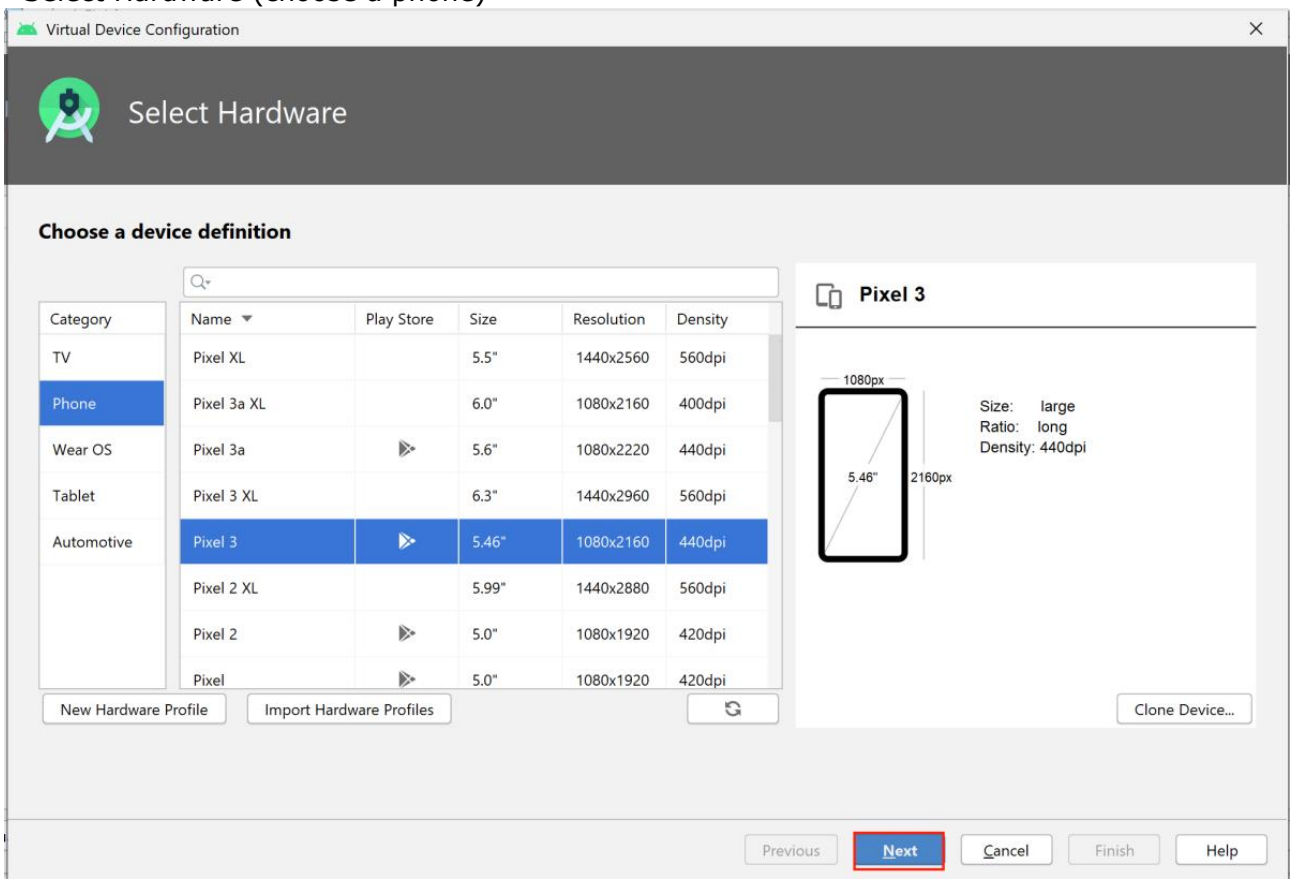
Emulator Configuration - Android Virtual Device (AVD)



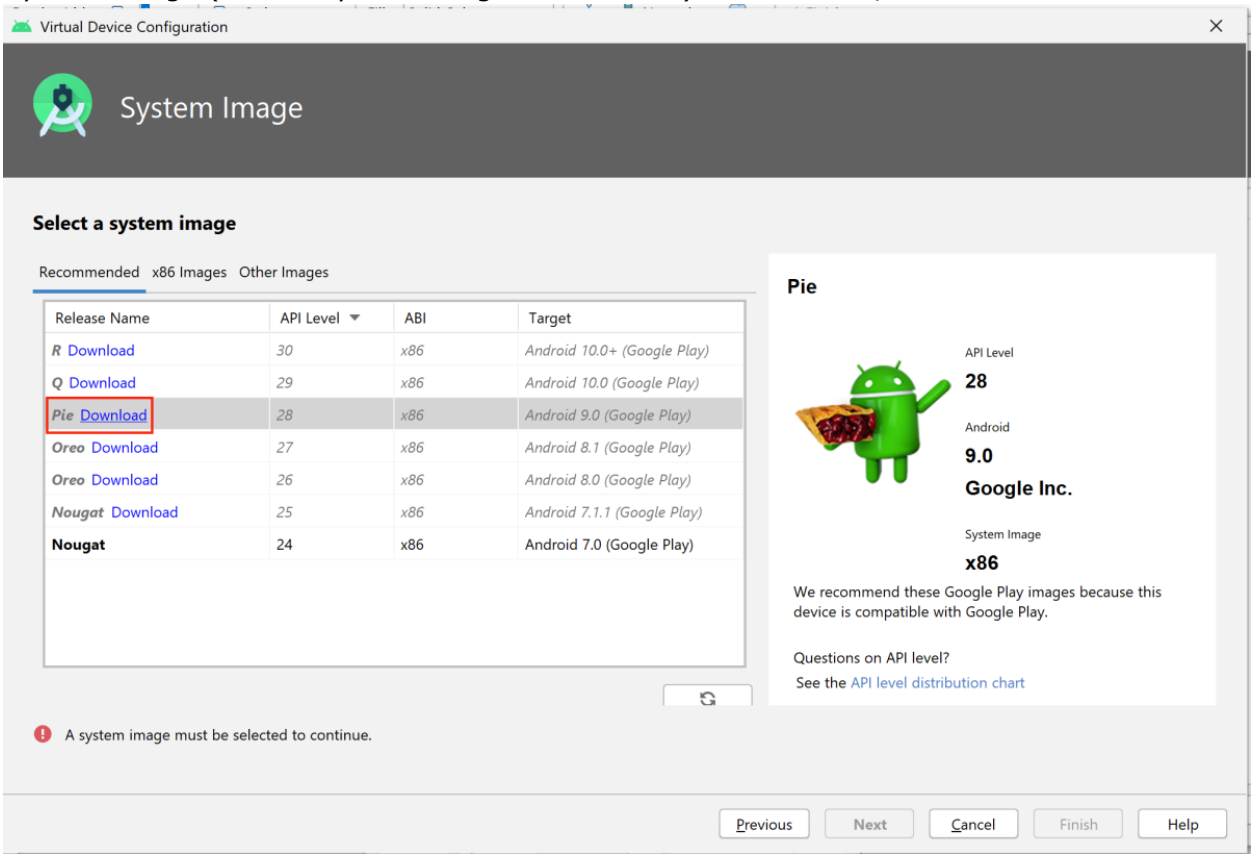
2. Click Create Virtual Device (displays "Select Hardware" window)



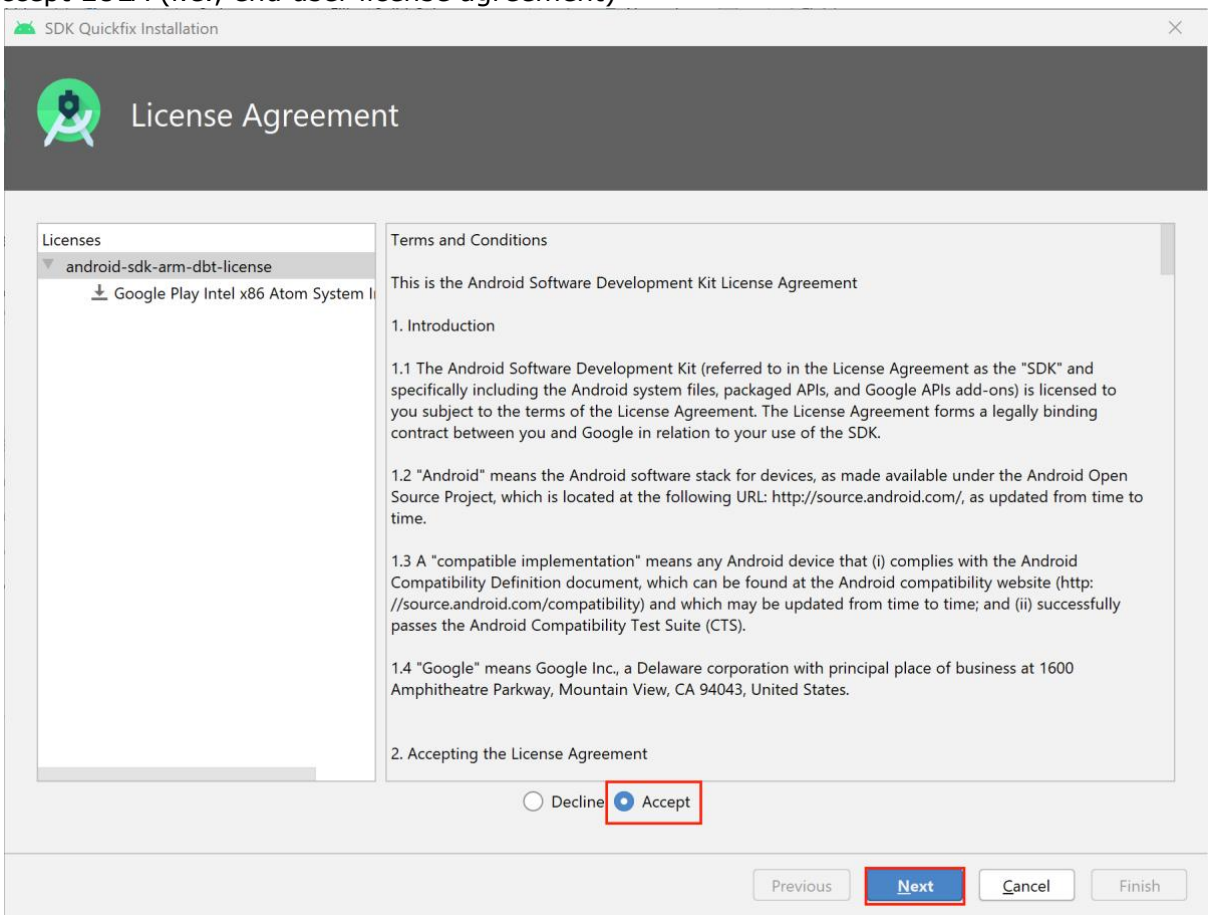
3. Select Hardware (choose a phone)



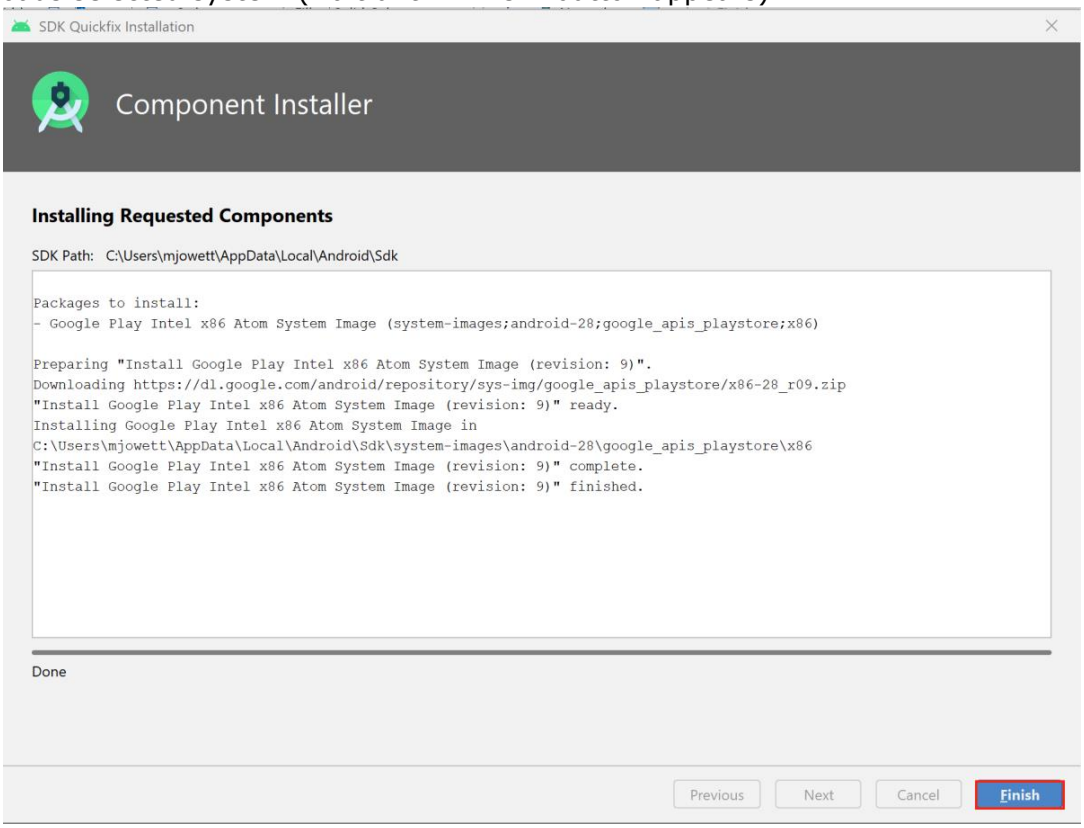
4. System Image (choose system image – Pie API 28). For Nexus 5, choose API 24.



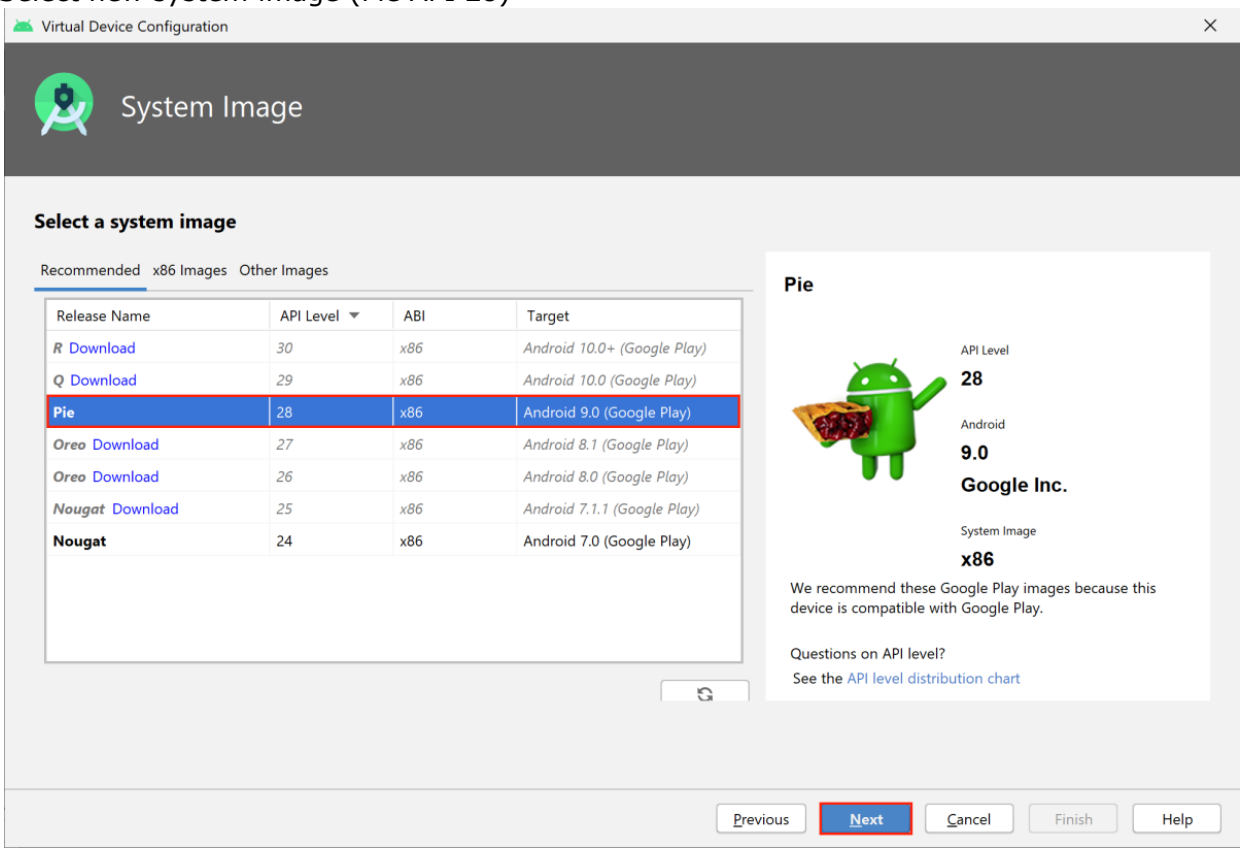
5. Accept EULA (i.e., end user license agreement)



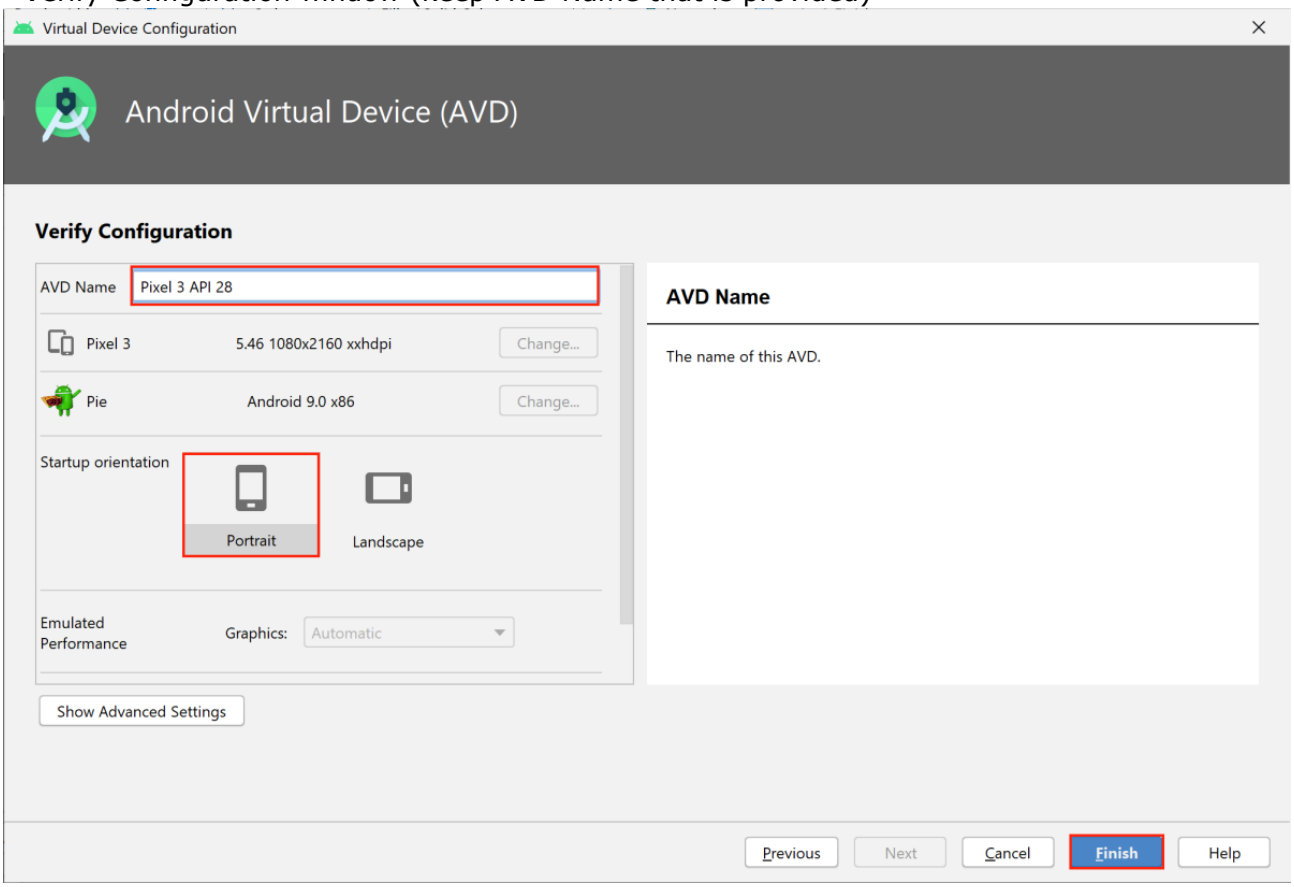
6. Downloads selected system (wait until "Finish" button appears)



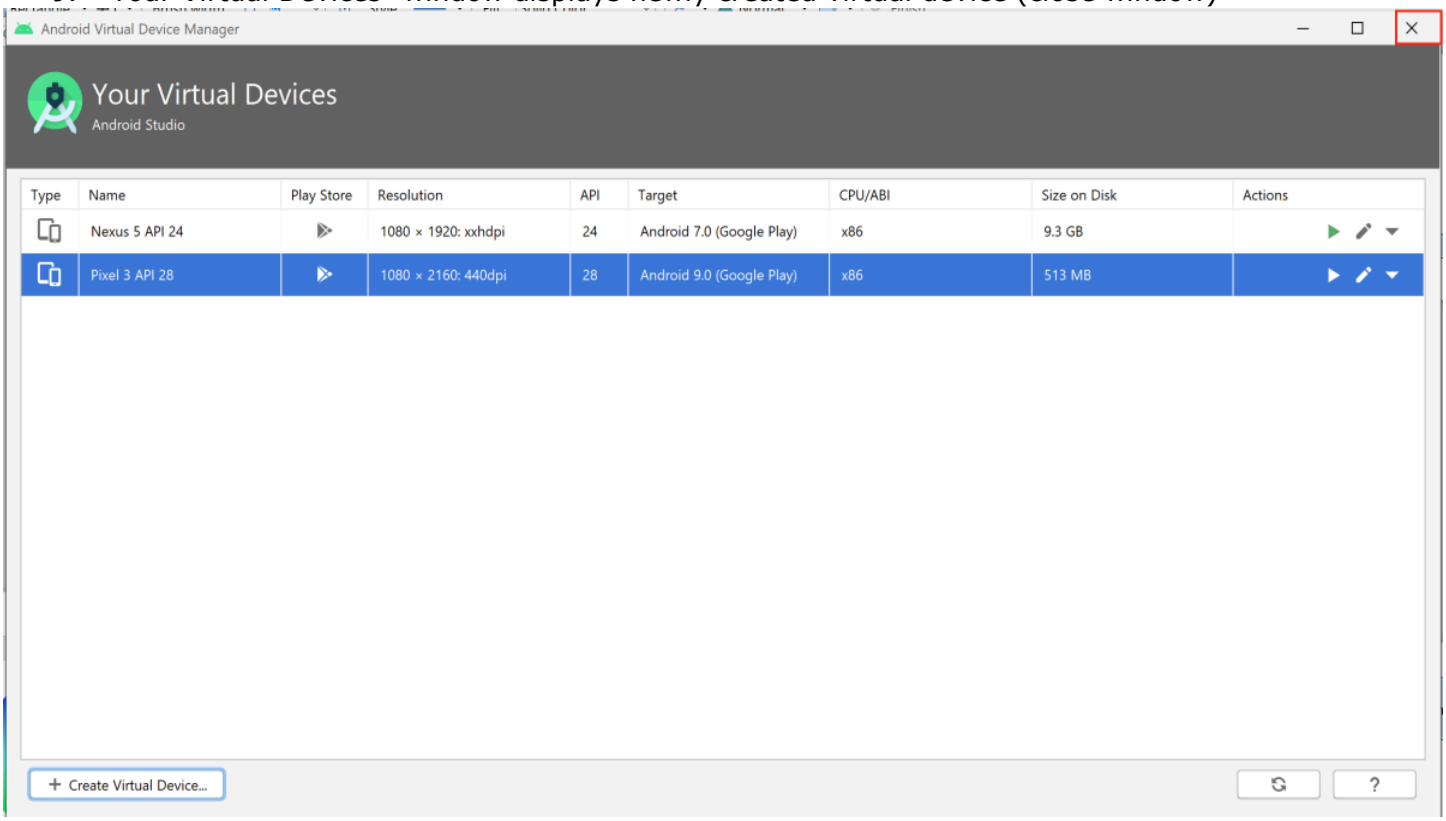
7. Select new system image (Pie API 28)



8. Verify Configuration window (keep AVD Name that is provided)

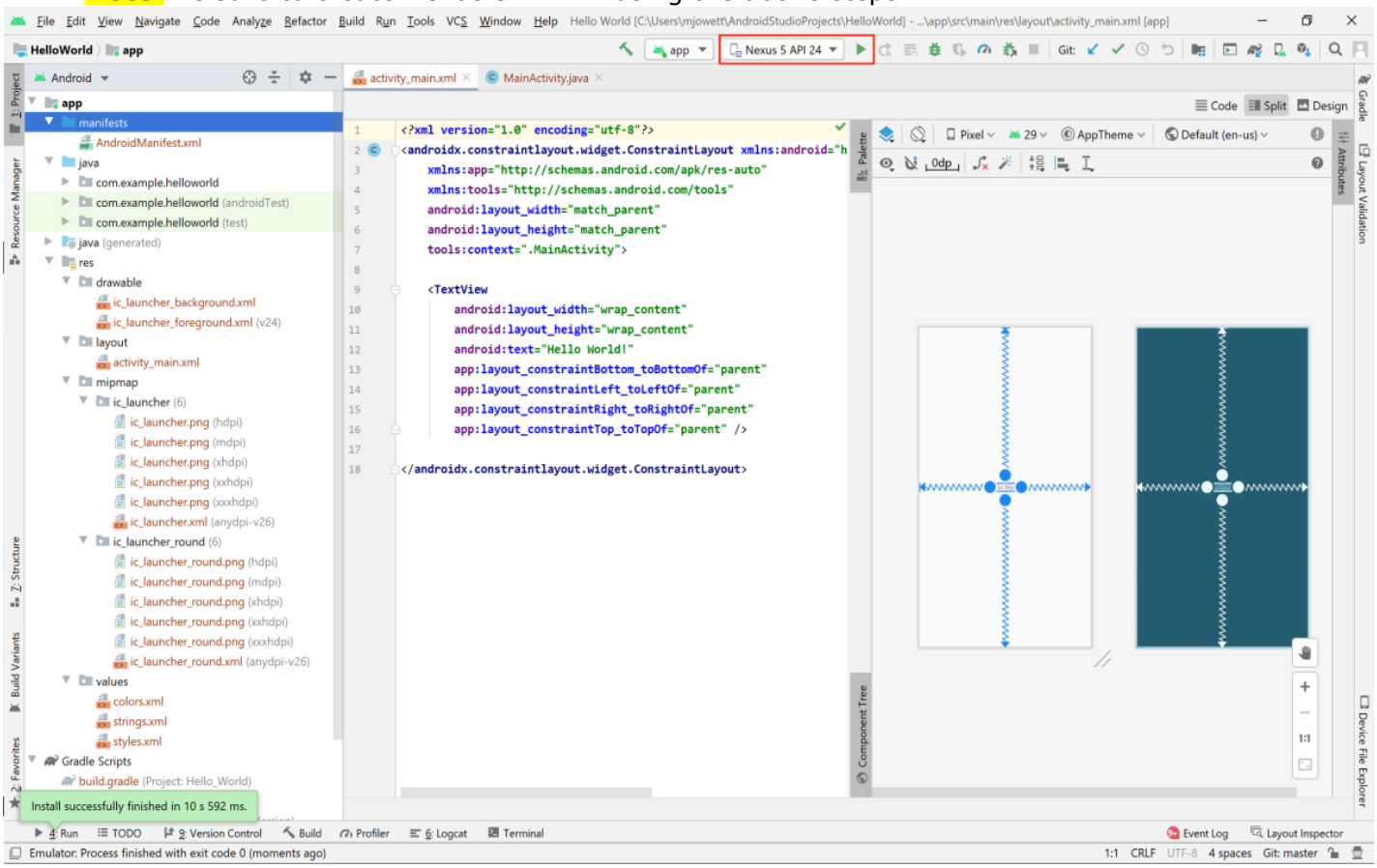


9. “Your Virtual Devices” window displays newly created virtual device (close window)

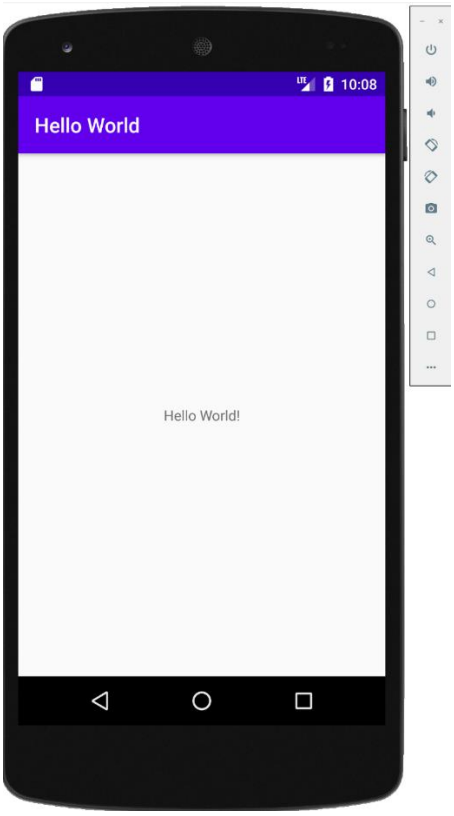


10. Select available devices (either **Nexus 5 API 24** or **Pixel 3 API 28**)

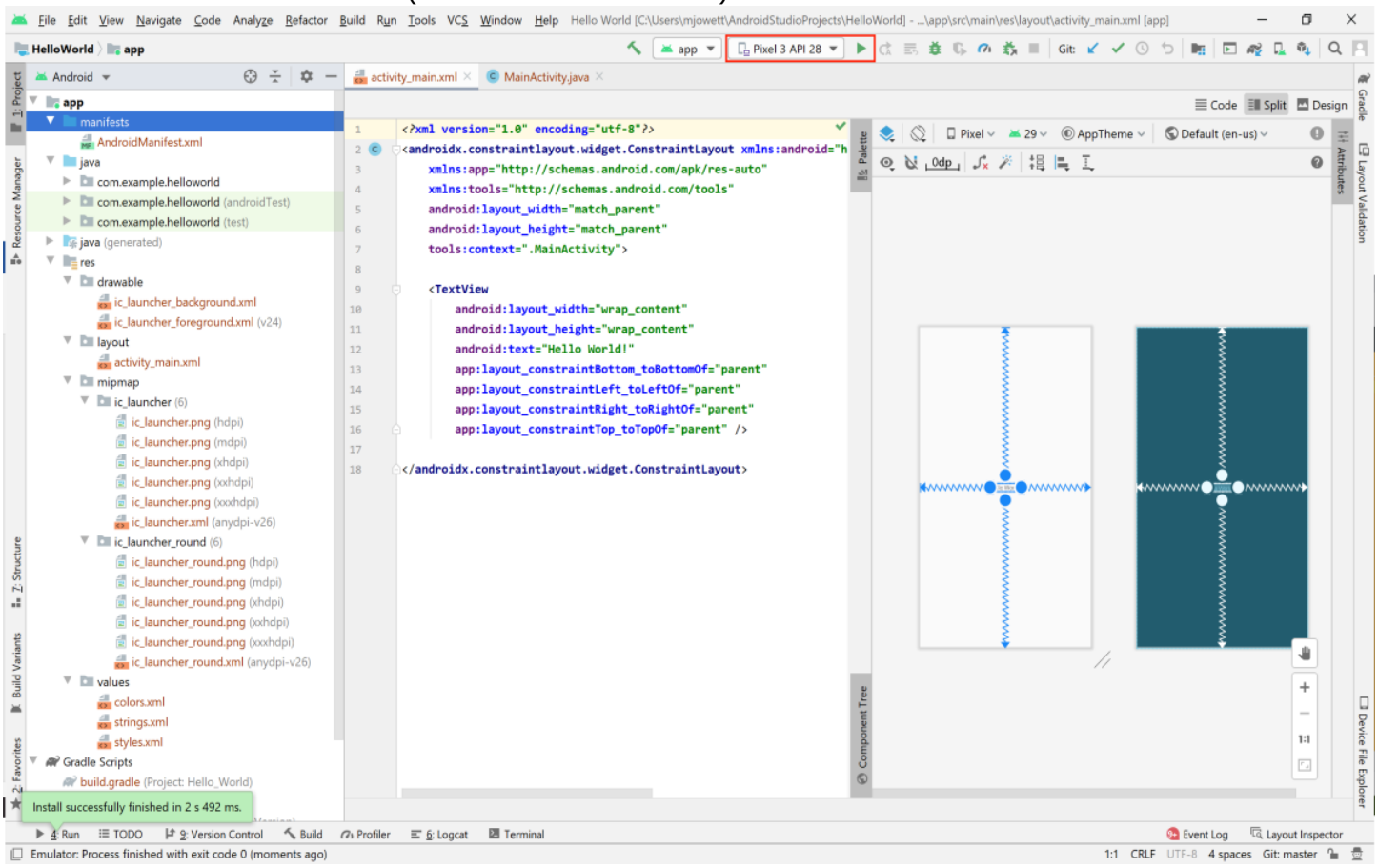
Note: Be sure to create Nexus 5 API 24 using the above steps.



11. Nexus 5 emulator:



12. Select available devices (choose **Pixel 3 API 28**)



13. Pixel 3 emulator: