Mobile Applications

Lab2. First Android app

Boni García

boni.garcia@uc3m.es

Telematic Engineering Department
School of Engineering

2023/2024

uc3m | Universidad Carlos III de Madrid

Table of contents

1. Introduction

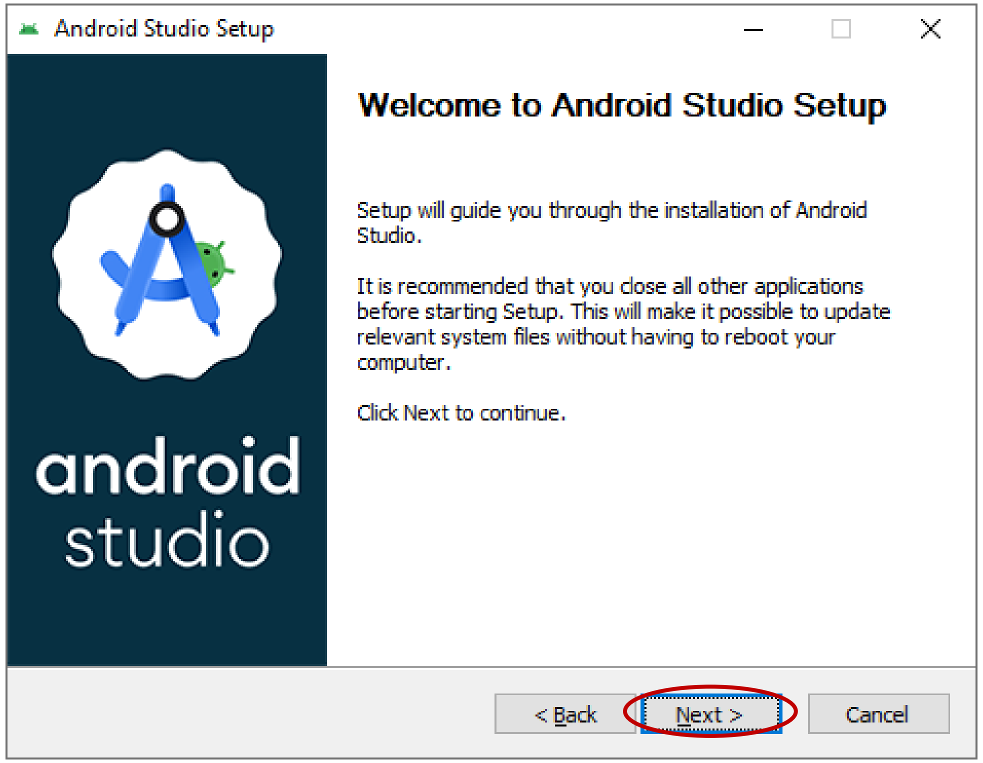
- The objective of this lab session is that each student creates and understands a very basic Android app (a "*hello world*" app) with Android Studio
  - These slides provides a guided tutorial to that aim
- The proposed procedure to achieve this objective is the following:
  1. Install Android Studio
     - Configure Software Development Kit (SDK)
     - Configure Android Virtual Device (AVD)
  2. Execute a basic app ("*hello world*")
     - Create app using the "empty activity" template
     - Run the app in the emulated device previously configured
     - Modify the app (proposed exercises)

2. Android Studio

- **Android Studio** is the development platform we use for creating Android apps in this course
  - Android Studio has been built on JetBrains' IntelliJ IDEA
- The hardware requirements for installing Android Studio are:
  - CPU 64 bits
  - 8 MB RAM
  - 8 GB disk (SSD recommended)
  - Minimum screen resolution 1280x800 (recommended 1920x1080)
- The recommended steps for installing Android Studio are:
  a) Install Android Studio using installer downloaded from: https://developer.android.com/studio
  b) Configure Android SDK. We will use Android 7.0
  c) Configure an Android Virtual Device (AVD). We will use a “Pixel 2” smartphone

2. Android Studio

- Step a) → Follow the wizard until Android Studio is installed



Android Studio Setup

Welcome to Android Studio Setup

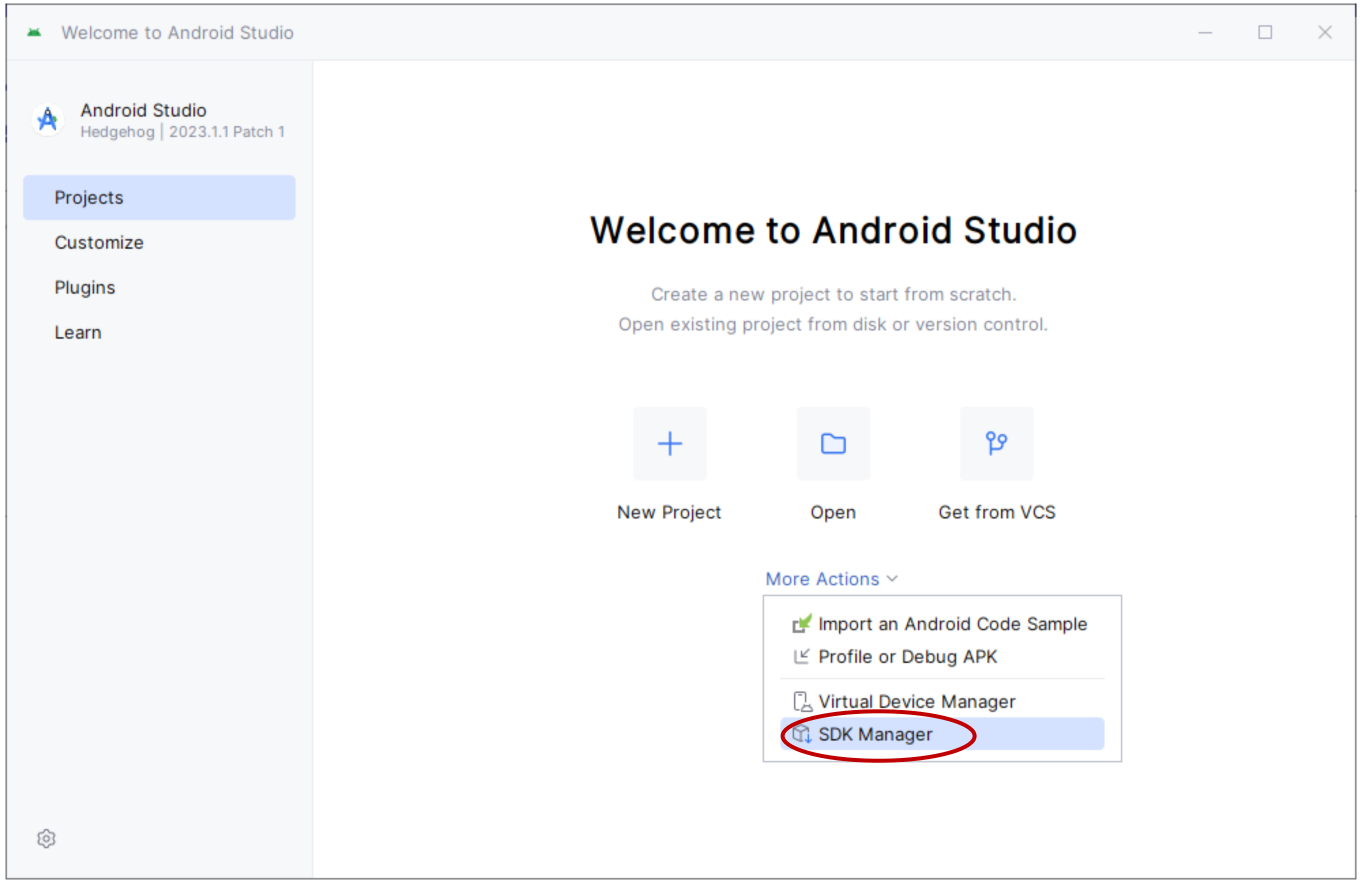
Setup will guide you through the installation of Android Studio.

It is recommended that you close all other applications before starting Setup. This will make it possible to update relevant system files without having to reboot your computer.

Click Next to continue.

2. Android Studio

- Step b) → Configure SDK
  - Using the SDK Manager

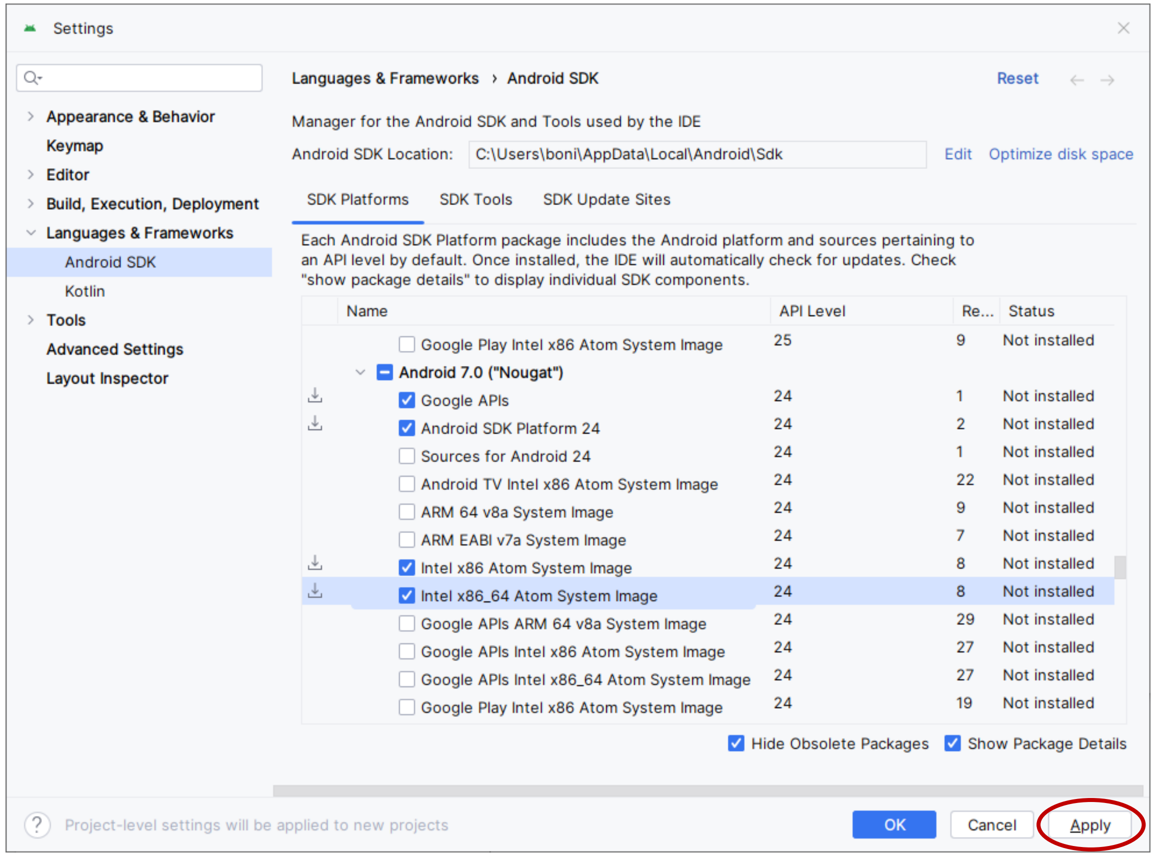


Welcome to Android Studio

Create a new project to start from scratch.
Open existing project from disk or version control.

2. Android Studio

- Step b) → Configure SDK
  - Install Android 7.0 (“Nougat”) with the following packages:
    - Google APIs
    - Android SDK Platform 24
    - Intel x86 Atom System Image
    - Intel x86_64 Atom System Image



Languages & Frameworks › Android SDK

Manager for the Android SDK and Tools used by the IDE

Android SDK Location: C:\Users\boni\AppData\Local\Android\Sdk

Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, the IDE will automatically check for updates. Check "show package details" to display individual SDK components.

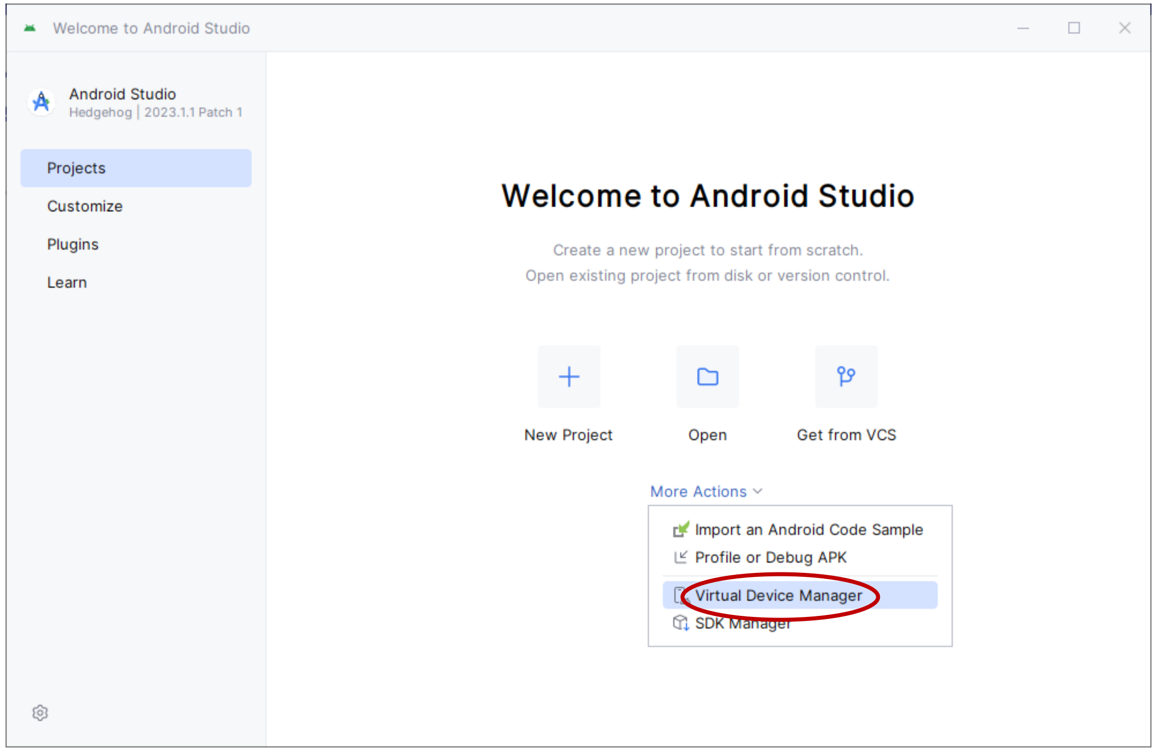
Name	API Level	Re...	Status
☐ Google Play Intel x86 Atom System Image	25	9	Not installed
Android 7.0 ("Nougat")			
✓ Google APIs	24	1	Not installed
✓ Android SDK Platform 24	24	2	Not installed
☐ Sources for Android 24	24	1	Not installed
☐ Android TV Intel x86 Atom System Image	24	22	Not installed
☐ ARM 64 v8a System Image	24	9	Not installed
☐ ARM EABI v7a System Image	24	7	Not installed
✓ Intel x86 Atom System Image	24	8	Not installed
✓ Intel x86_64 Atom System Image	24	8	Not installed
☐ Google APIs ARM 64 v8a System Image	24	29	Not installed
☐ Google APIs Intel x86 Atom System Image	24	27	Not installed
☐ Google APIs Intel x86_64 Atom System Image	24	27	Not installed
☐ Google Play Intel x86 Atom System Image	24	19	Not installed

✓ Hide Obsolete Packages ✓ Show Package Details

Project-level settings will be applied to new projects

2. Android Studio

- Step c) → create a new Android Virtual Device (AVD)
  - Virtual Device Manager, then, “Create Virtual Device” button (+)


Welcome to Android Studio

Android Studio
Hedgehog | 2023.1.1 Patch 1

Projects
Customize
Plugins
Learn

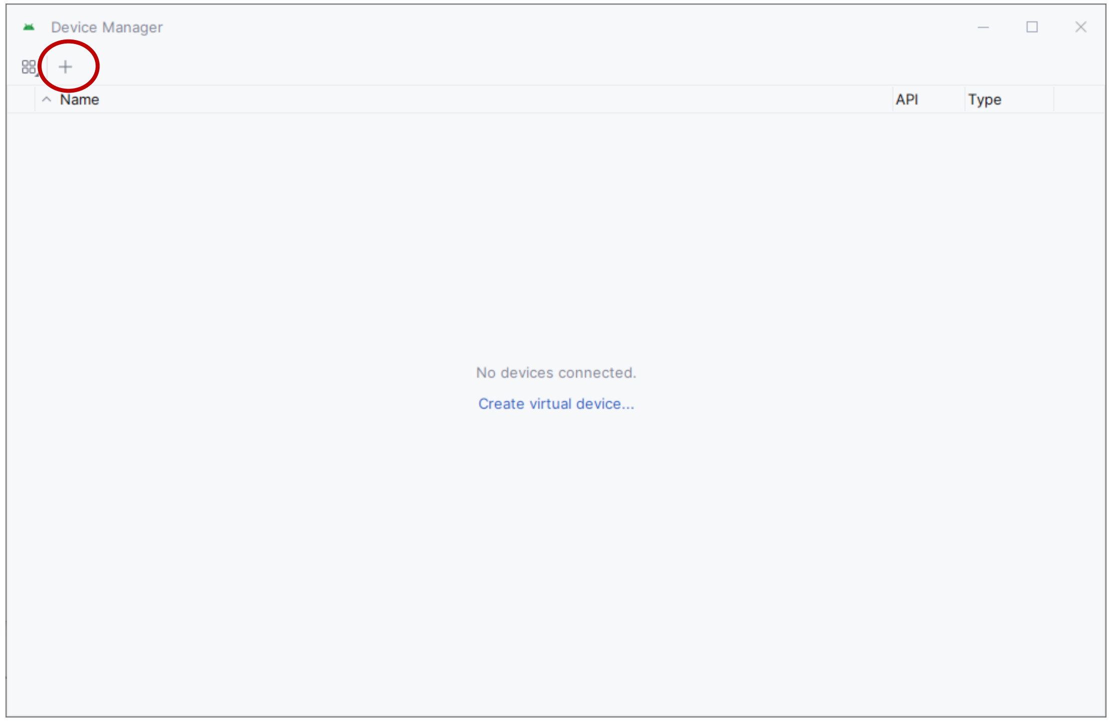
Welcome to Android Studio

Create a new project to start from scratch.
Open existing project from disk or version control.

New Project
Open
Get from VCS

More Actions

Import an Android Code Sample
Profile or Debug APK
Virtual Device Manager
SDK Manager

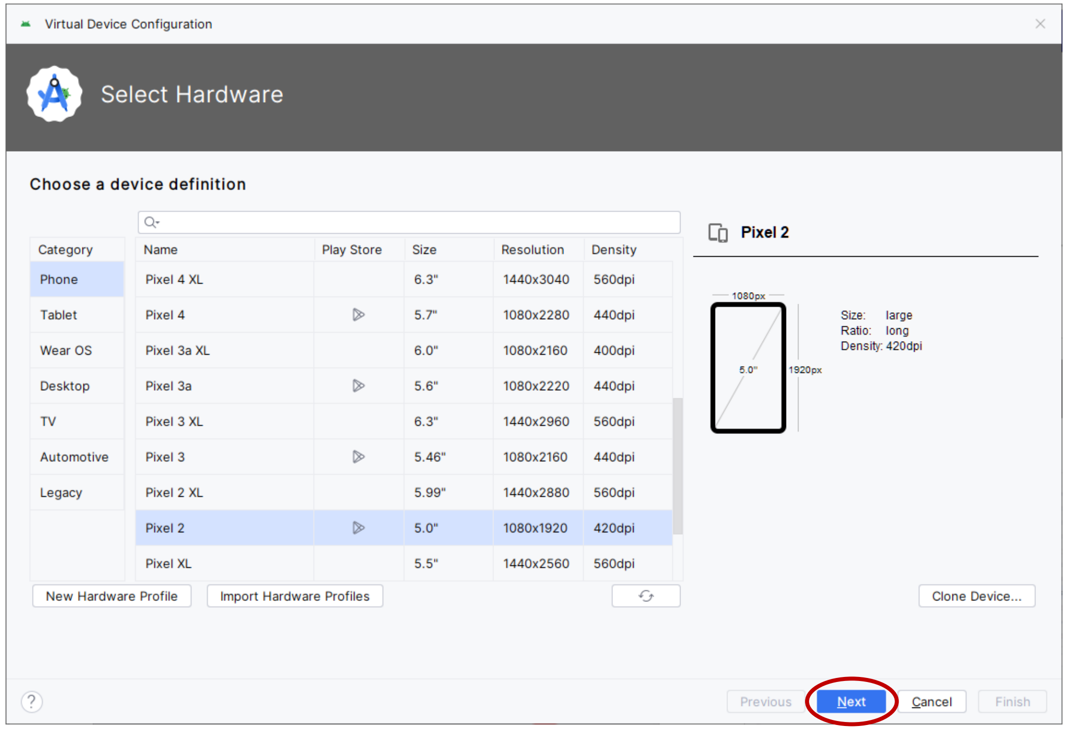

Device Manager

Name	API	Type

No devices connected.
Create virtual device...

2. Android Studio

- Step c) → create a new Android Virtual Device (AVD)
  - By default, we use a "Pixel 2" with Android 10 (API level 29 with Google Play)



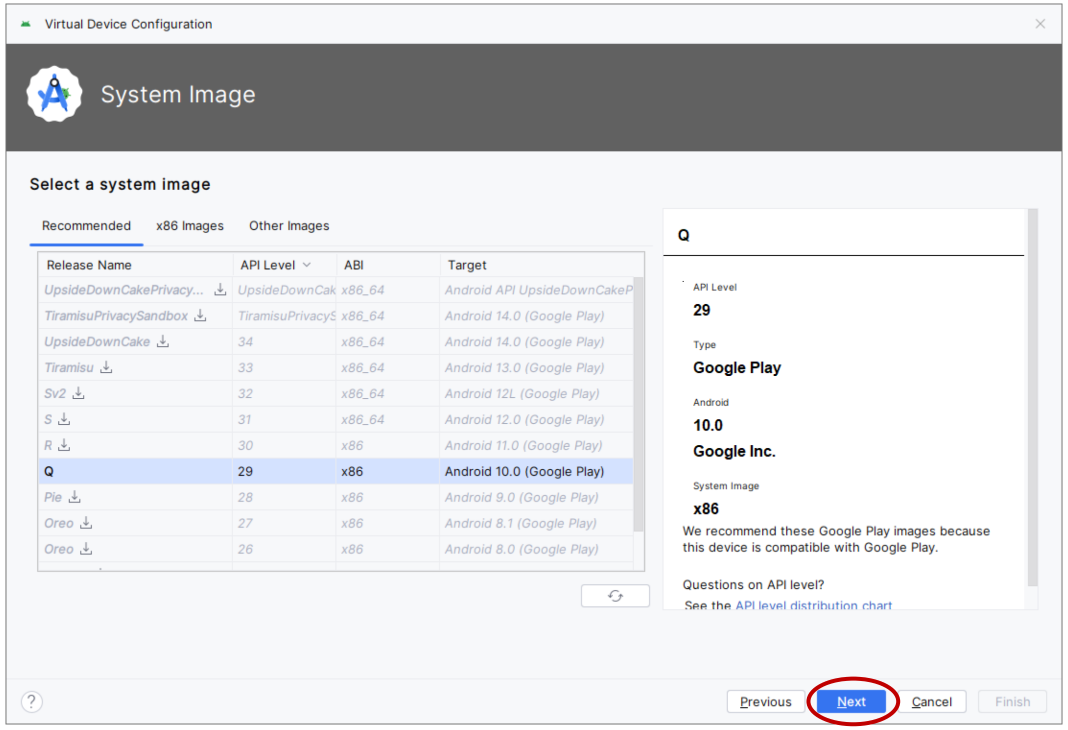
Choose a device definition

Category	Name	Play Store	Size	Resolution	Density
Phone	Pixel 4 XL		6.3"	1440x3040	560dpi
Tablet	Pixel 4	▷	5.7"	1080x2280	440dpi
Wear OS	Pixel 3a XL		6.0"	1080x2160	400dpi
Desktop	Pixel 3a	▷	5.6"	1080x2220	440dpi
TV	Pixel 3 XL		6.3"	1440x2960	560dpi
Automotive	Pixel 3	▷	5.46"	1080x2160	440dpi
Legacy	Pixel 2 XL		5.99"	1440x2880	560dpi
	Pixel 2	▷	5.0"	1080x1920	420dpi
	Pixel XL		5.5"	1440x2560	560dpi

Pixel 2

1080px 5.0" 1920px

Size: large
Ratio: long
Density: 420dpi



Select a system image

Release Name	API Level	ABI	Target
UpsideDownCakePrivacy...	UpsideDownCak	x86_64	Android API UpsideDownCakeP
TiramisuPrivacySandbox	TiramisuPrivacyS	x86_64	Android 14.0 (Google Play)
UpsideDownCake	34	x86_64	Android 14.0 (Google Play)
Tiramisu	33	x86_64	Android 13.0 (Google Play)
Sv2	32	x86_64	Android 12L (Google Play)
S	31	x86_64	Android 12.0 (Google Play)
R	30	x86	Android 11.0 (Google Play)
Q	29	x86	Android 10.0 (Google Play)
Pie	28	x86	Android 9.0 (Google Play)
Oreo	27	x86	Android 8.1 (Google Play)
Oreo	26	x86	Android 8.0 (Google Play)

Q

API Level
29

Type
Google Play

Android
10.0
Google Inc.

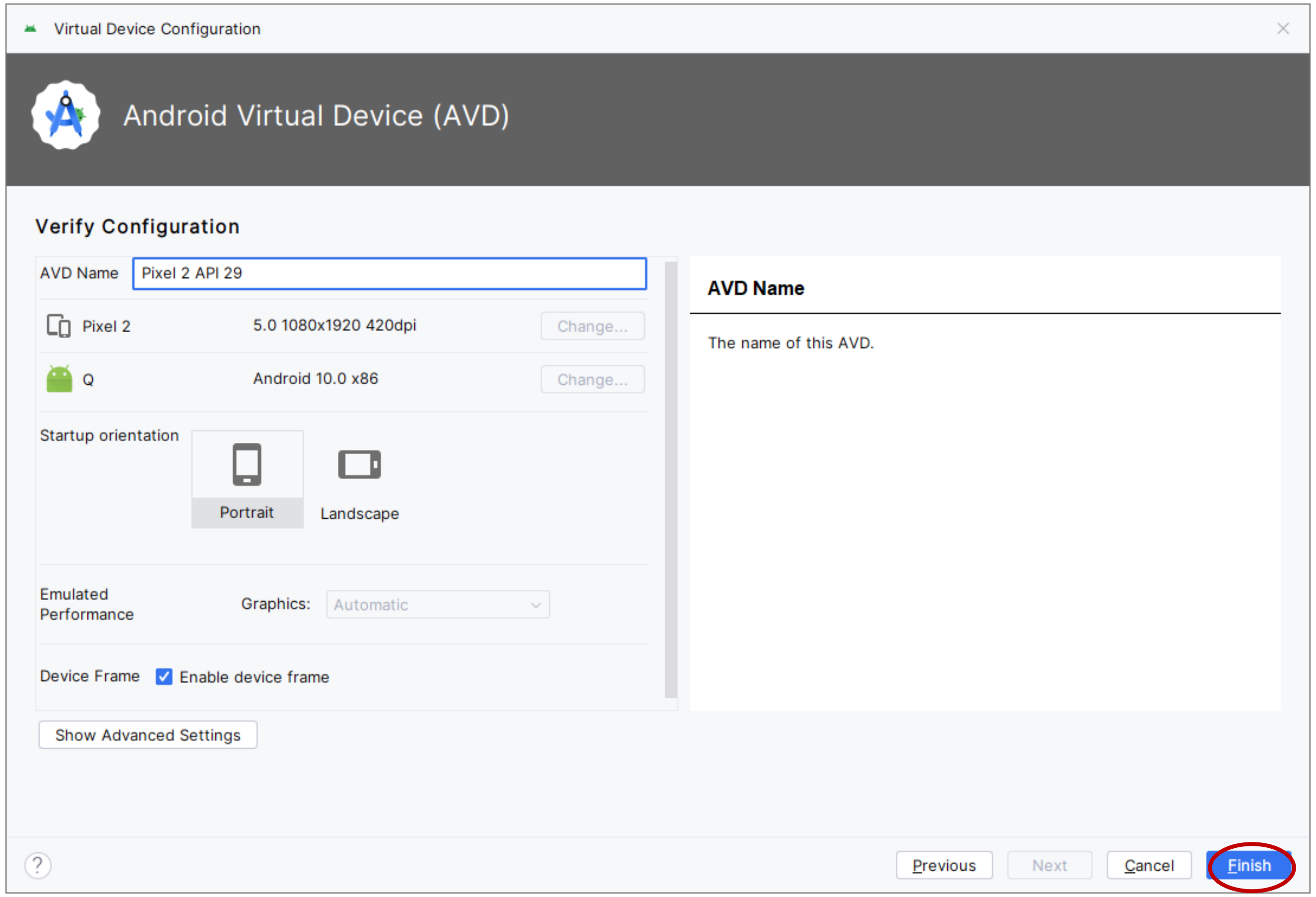
System Image
x86

We recommend these Google Play images because this device is compatible with Google Play.

Questions on API level?
See the API level distribution chart

2. Android Studio

- Step c) → create a new Android Virtual Device (AVD)



Virtual Device Configuration

Android Virtual Device (AVD)

Verify Configuration

AVD Name: Pixel 2 API 29

Pixel 2 — 5.0 1080x1920 420dpi — Change...

Q — Android 10.0 x86 — Change...

Startup orientation: [✓ Portrait] Landscape

Emulated Performance — Graphics: Automatic

Device Frame — ✓ Enable device frame

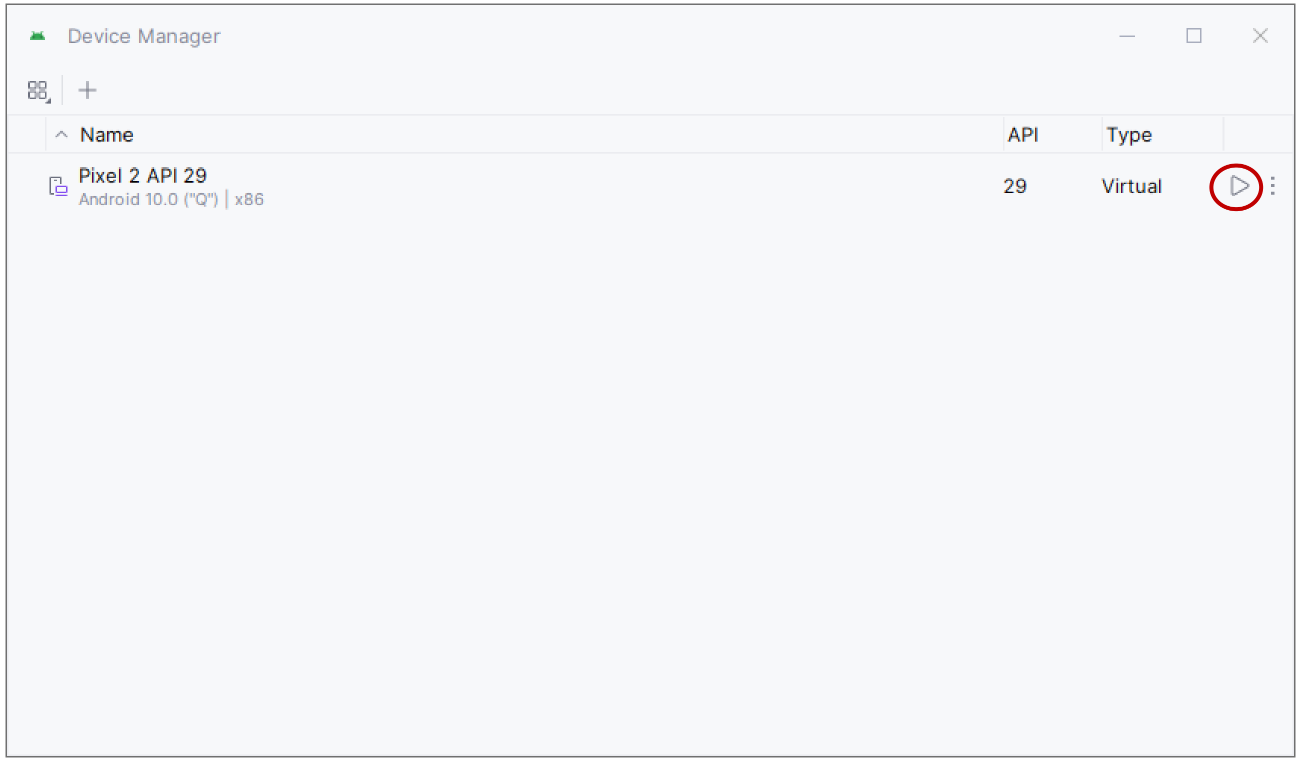
Show Advanced Settings

AVD Name

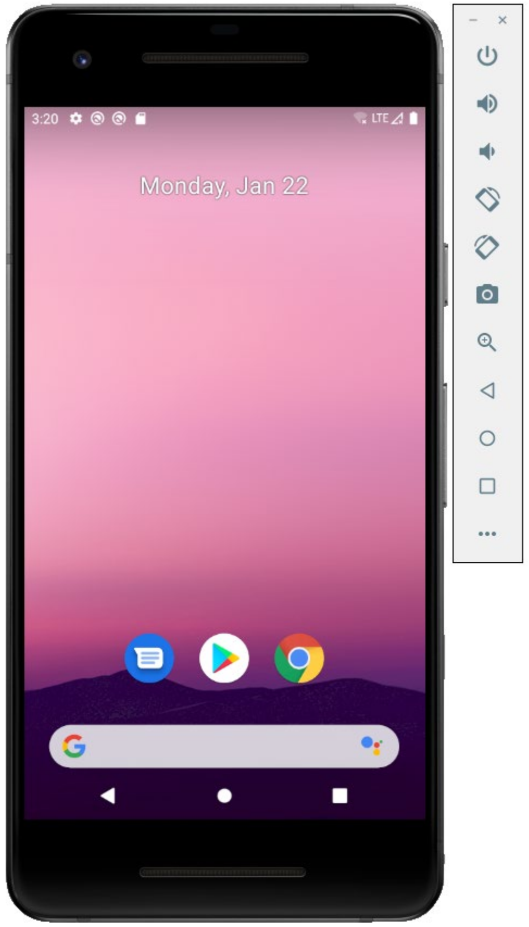
The name of this AVD.

2. Android Studio

- Step c) → create a new Android Virtual Device (AVD)

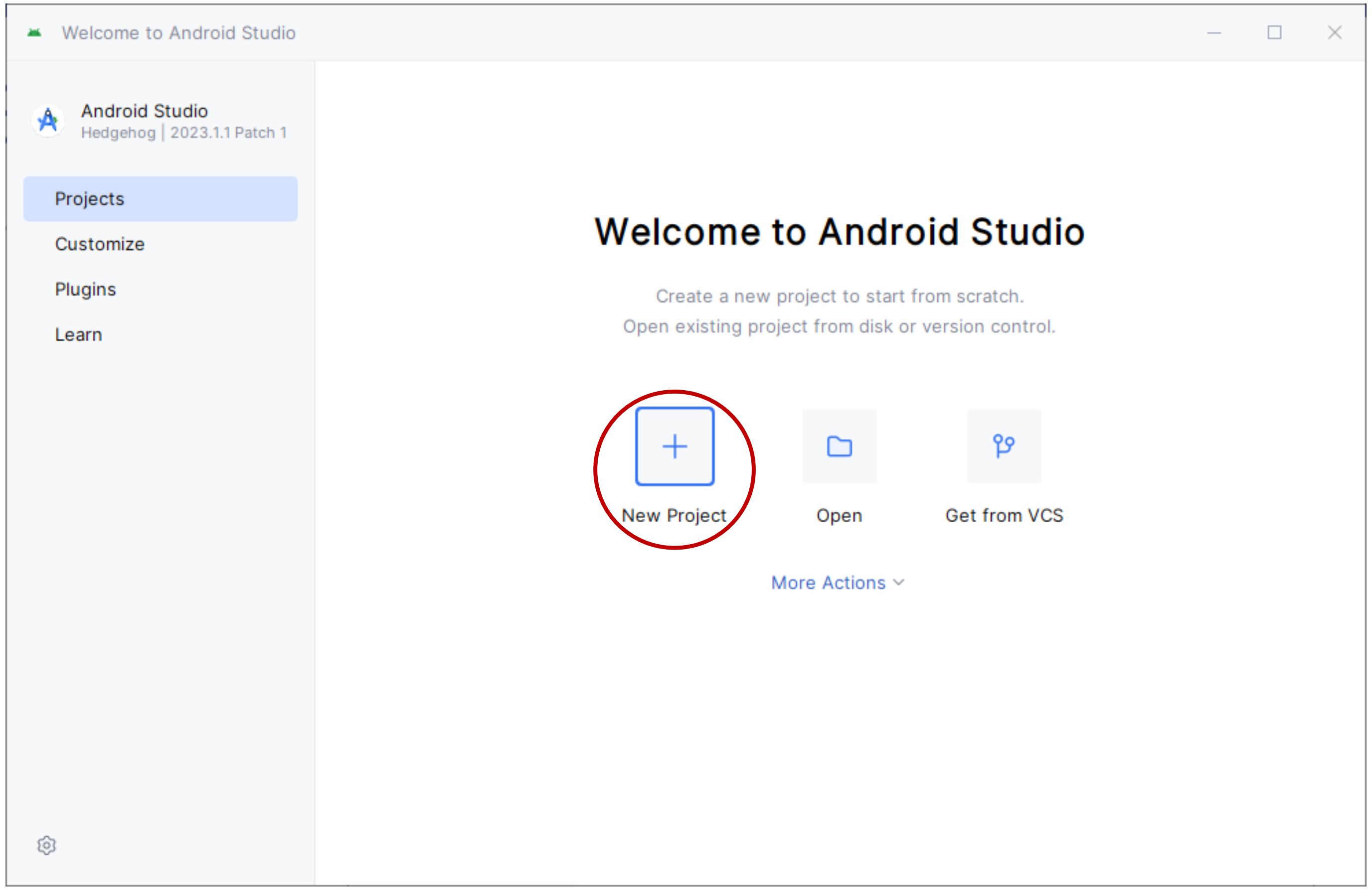

Device Manager

Name	API	Type
Pixel 2 API 29 Android 10.0 ("Q") | x86	29	Virtual

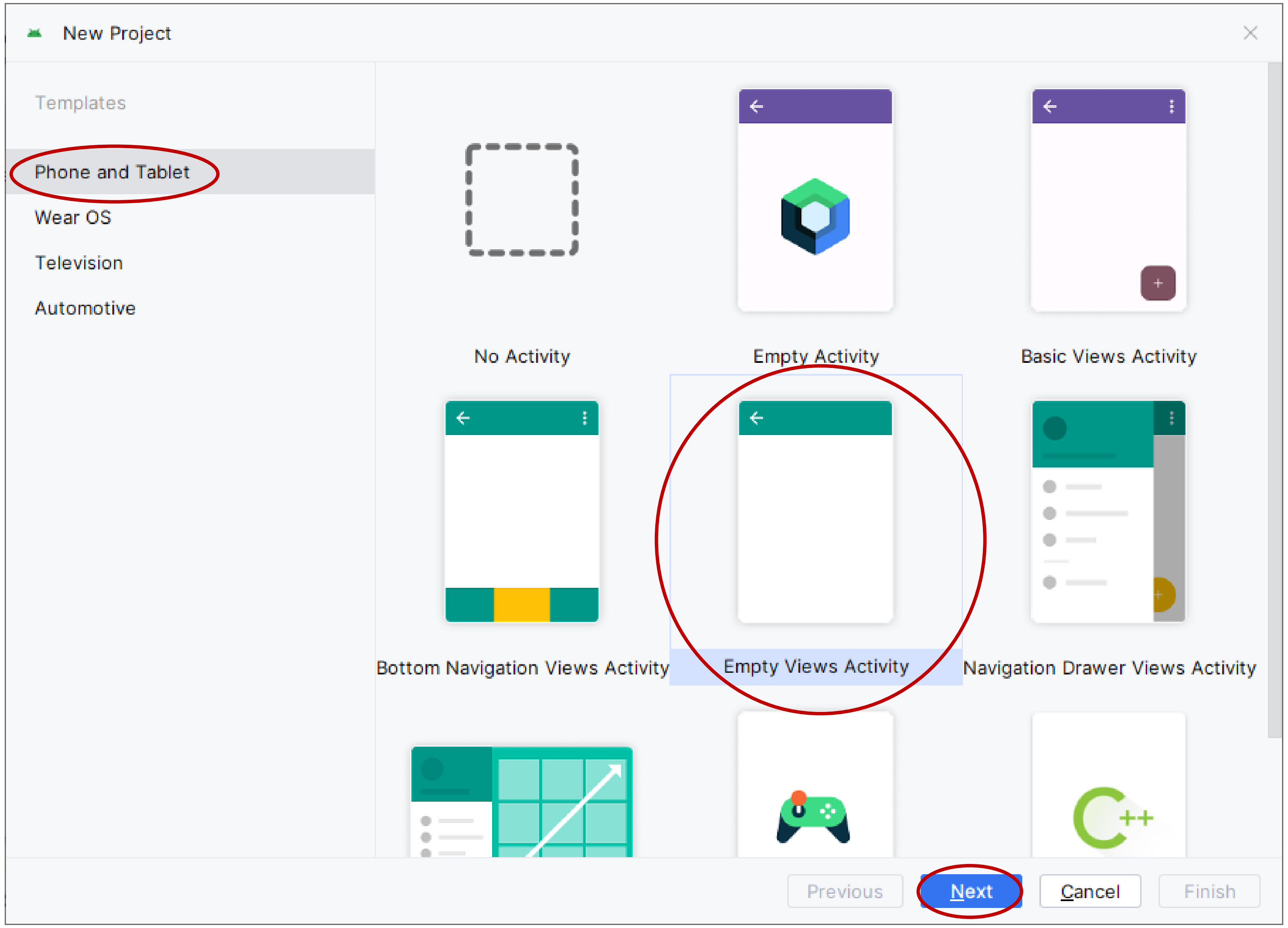


3. “Hello world” app

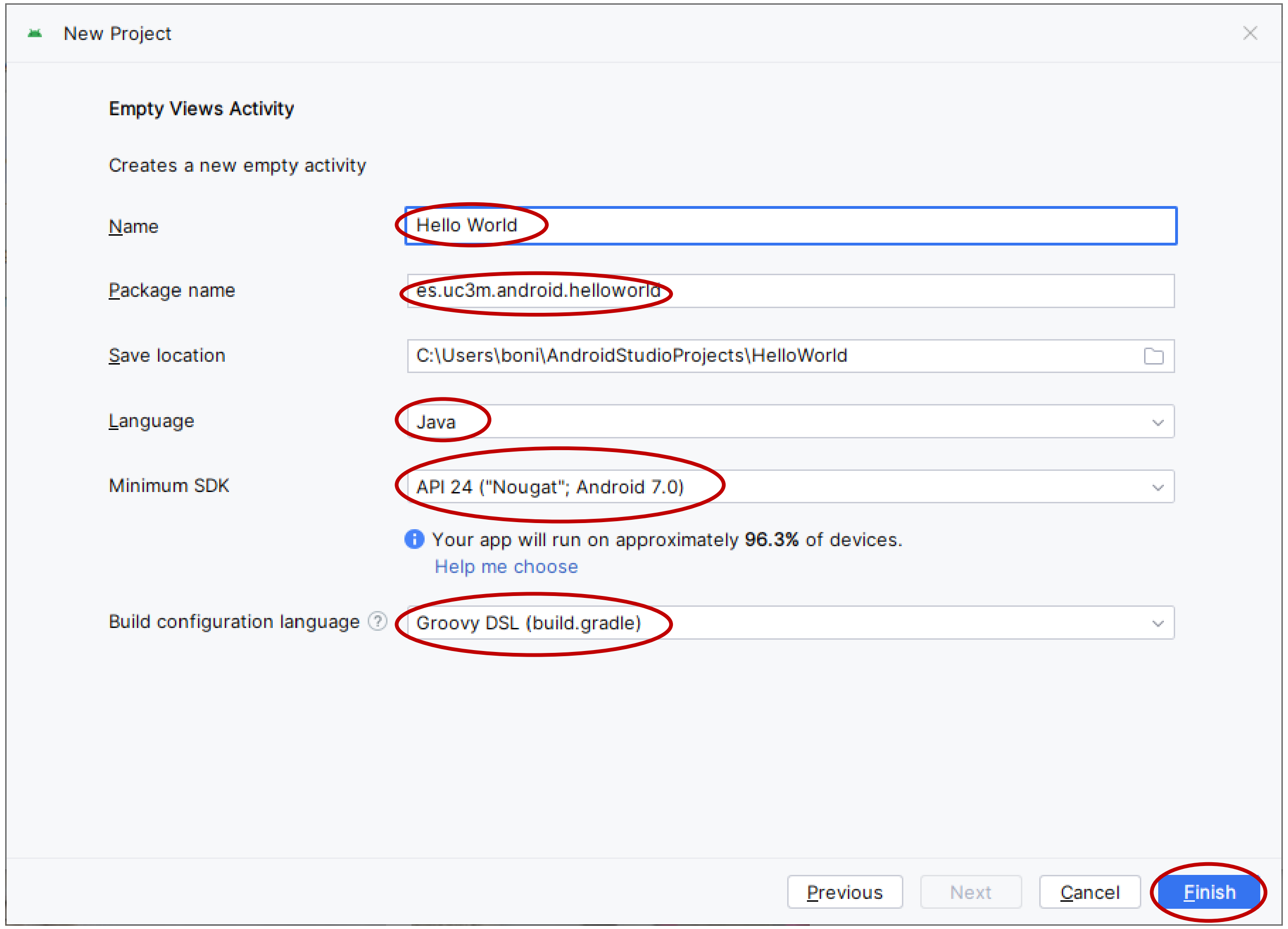
- We can create a very simple Android app as follows:


Welcome to Android Studio
Android Studio
Hedgehog | 2023.1.1 Patch 1
Projects
Customize
Plugins
Learn
Welcome to Android Studio
Create a new project to start from scratch.
Open existing project from disk or version control.
New Project
Open
Get from VCS
More Actions

3. “Hello world” app



3. "Hello world" app

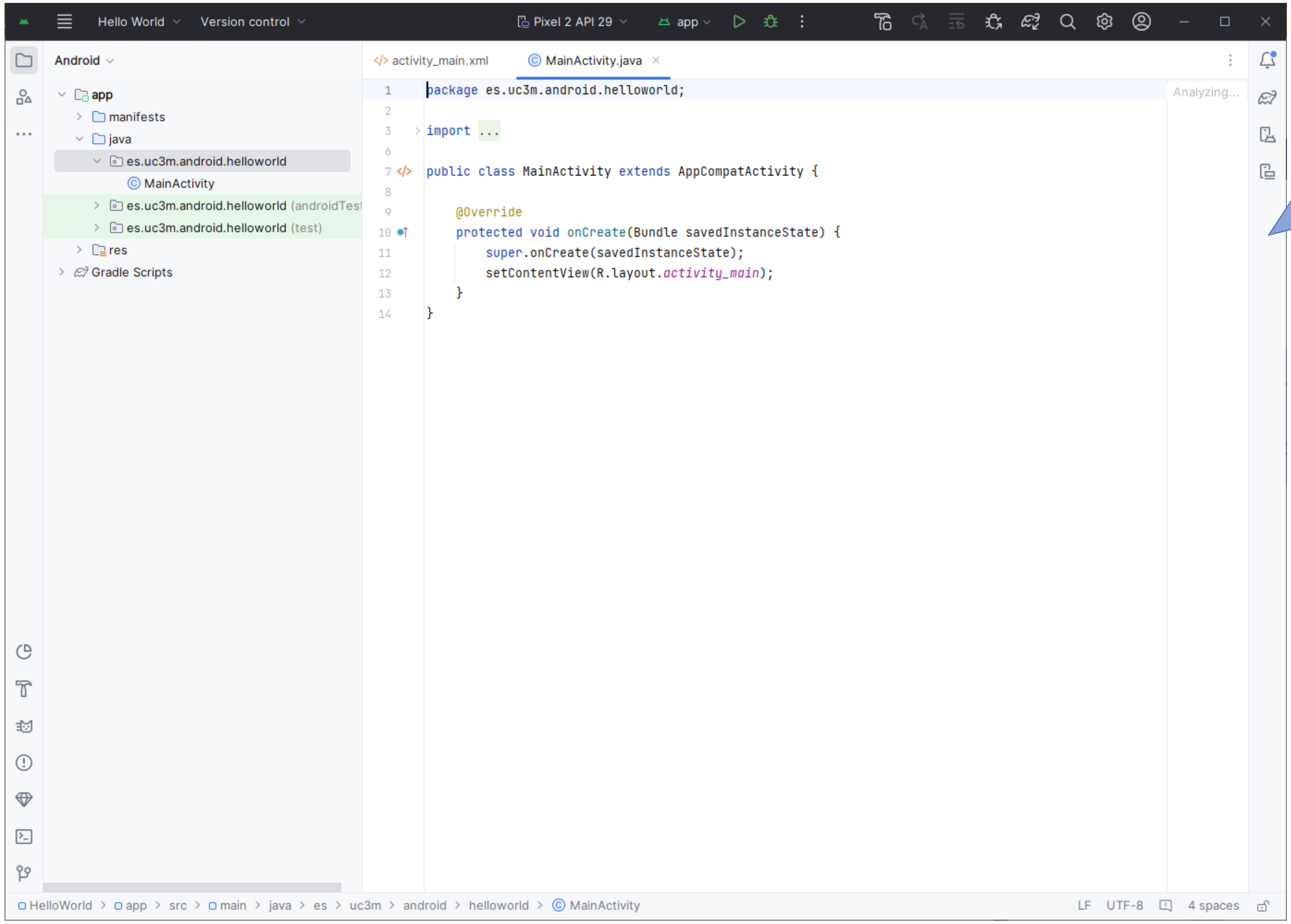

New Project

Empty Views Activity

Creates a new empty activity

Name	Hello World
Package name	es.uc3m.android.helloworld
Save location	C:\Users\boni\AndroidStudioProjects\HelloWorld
Language	Java
Minimum SDK	API 24 ("Nougat"; Android 7.0)
	Your app will run on approximately **96.3%** of devices. Help me choose
Build configuration language	Groovy DSL (build.gradle)

3. “Hello world” app


Hello World
Version control
Pixel 2 API 29
app
Android
app
manifests
java
es.uc3m.android.helloworld
MainActivity
es.uc3m.android.helloworld (androidTest)
es.uc3m.android.helloworld (test)
res
Gradle Scripts
activity_main.xml
MainActivity.java
Analyzing...
HelloWorld > app > src > main > java > es > uc3m > android > helloworld > MainActivity
LF UTF-8 4 spaces

```java
package es.uc3m.android.helloworld;

import ...

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

This is Android Studio using the “*new UI*” (beta look and feel first introduced in IntelliJ 2022.2)

3. “Hello world” app

- The **manifest file** describes different information about the app:

```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">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.HelloWorld"
        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>
```

The notation `@res` allows to use resources (i.e., included in the res folder. For instance, `@string/app_name` refers to the app name (defined as a string in `res/values/strings.xml`)

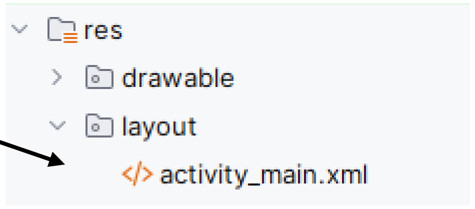
Activity (i.e., a Java or Kotlin class)

3. “Hello world” app

- Each app component (e.g., an Activity) is defined as a Java class

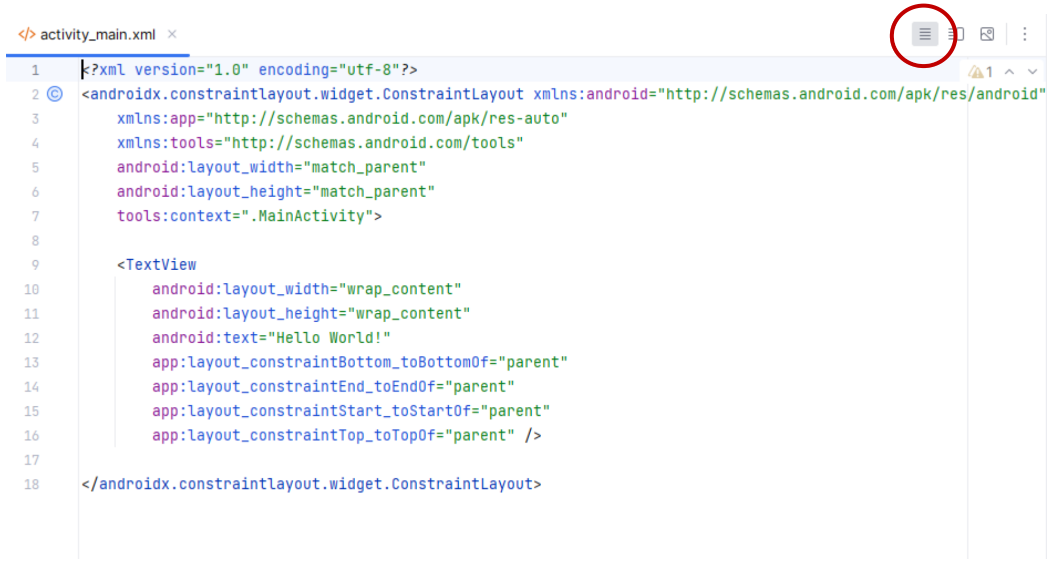
```
public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}
```

This statement allows to define the layout (*activity_main*) for this activity

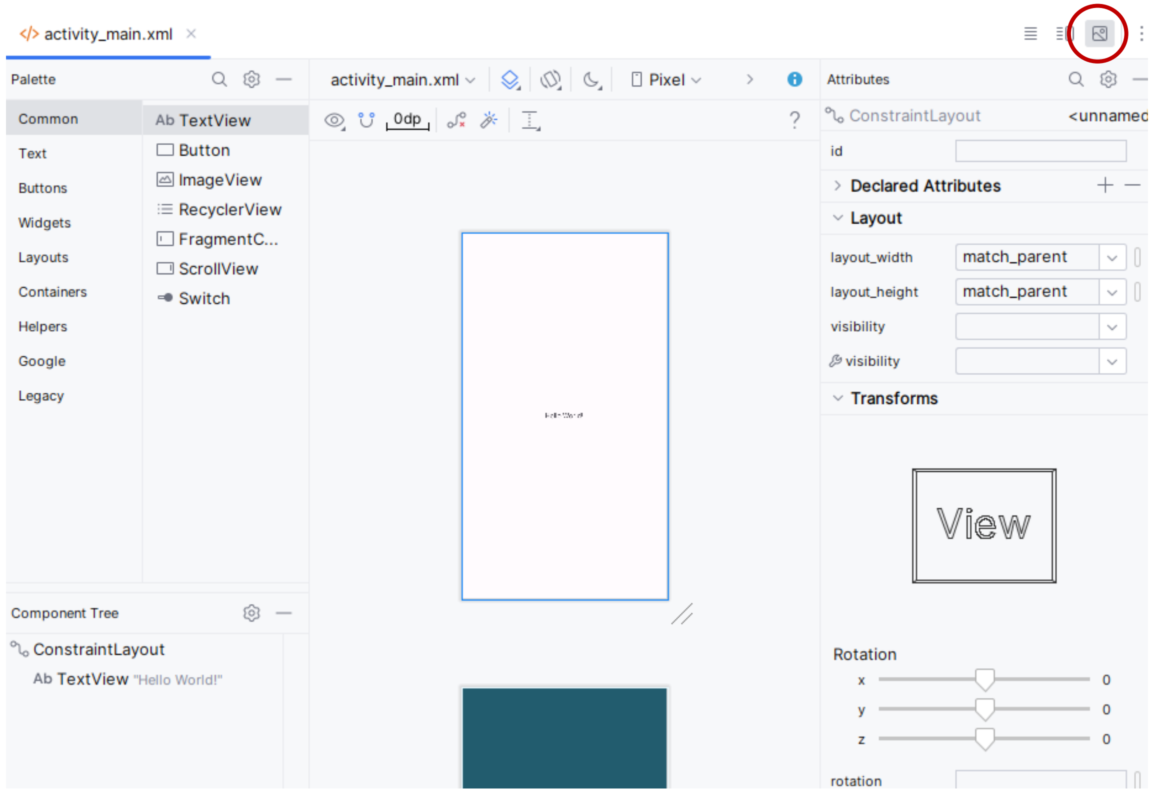


3. "Hello world" app

- The activity layout is defined in XML, although Android Studio has a graphical tool to design it:



```xml
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```



3. “Hello world” app

- Some relevant parts of this XML layout are:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

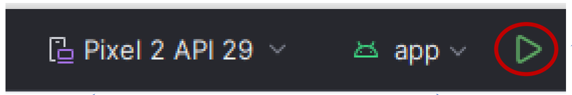
This activity uses all the available screen

This activity contains only one visual element: a `TextView`, which displays some text to the user

The content of the `TextView` is the hardcoded string `Hello World!`

3. "Hello world" app

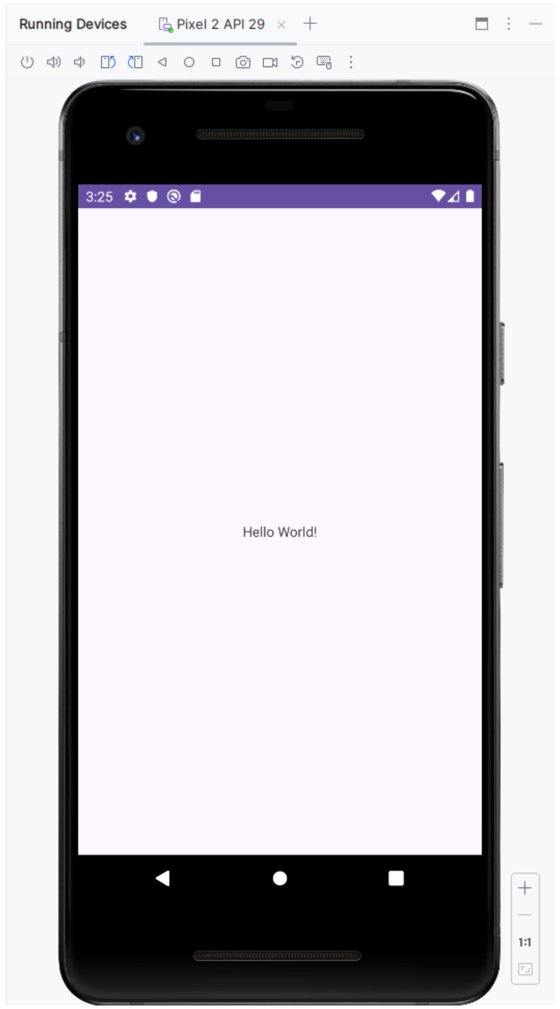
- We can run easily our app with Android Studio using the following controls:



This button executes the selected app in the selected AVD

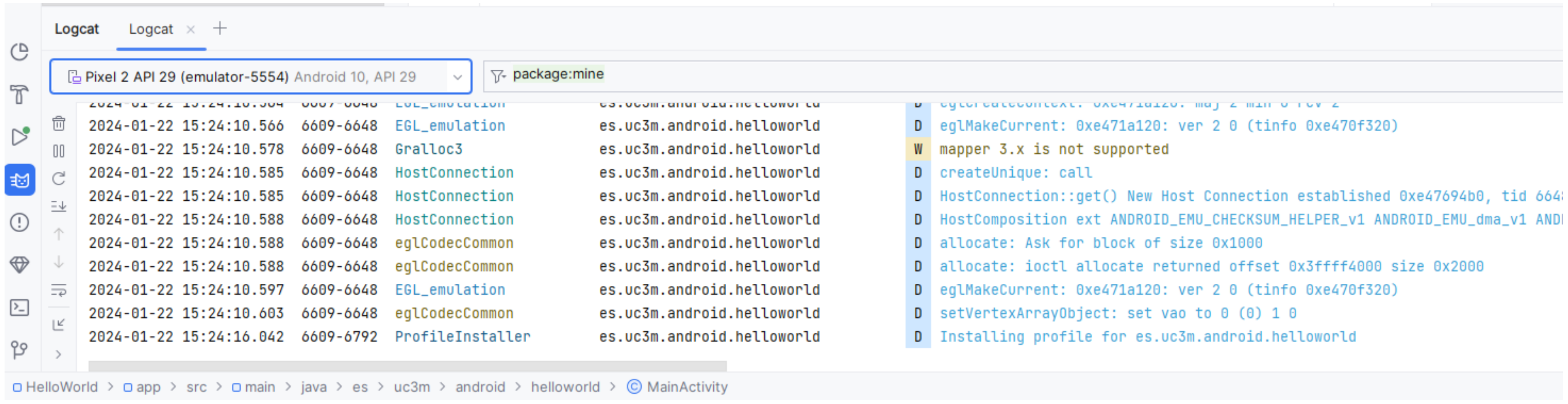
This dropdown menu allows to select the AVD in which the app is deployed

This dropdown menu allows to select the app (i.e., the Gradle module) to be executed



3. “Hello world” app

- When our app is in execution, we can see the logs in the Logcat view:



```
2024-01-22 15:24:10.566  6609-6648  EGL_emulation      es.uc3m.android.helloworld  D  eglMakeCurrent: 0xe471a120: ver 2 0 (tinfo 0xe470f320)
2024-01-22 15:24:10.578  6609-6648  Gralloc3           es.uc3m.android.helloworld  W  mapper 3.x is not supported
2024-01-22 15:24:10.585  6609-6648  HostConnection     es.uc3m.android.helloworld  D  createUnique: call
2024-01-22 15:24:10.585  6609-6648  HostConnection     es.uc3m.android.helloworld  D  HostConnection::get() New Host Connection established 0xe47694b0, tid 664
2024-01-22 15:24:10.588  6609-6648  HostConnection     es.uc3m.android.helloworld  D  HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_v1 ANDROID_EMU_dma_v1 AND
2024-01-22 15:24:10.588  6609-6648  eglCodecCommon     es.uc3m.android.helloworld  D  allocate: Ask for block of size 0x1000
2024-01-22 15:24:10.588  6609-6648  eglCodecCommon     es.uc3m.android.helloworld  D  allocate: ioctl allocate returned offset 0x3ffff4000 size 0x2000
2024-01-22 15:24:10.597  6609-6648  EGL_emulation      es.uc3m.android.helloworld  D  eglMakeCurrent: 0xe471a120: ver 2 0 (tinfo 0xe470f320)
2024-01-22 15:24:10.603  6609-6648  eglCodecCommon     es.uc3m.android.helloworld  D  setVertexArrayObject: set vao to 0 (0) 1 0
2024-01-22 15:24:16.042  6609-6792  ProfileInstaller   es.uc3m.android.helloworld  D  Installing profile for es.uc3m.android.helloworld
```

HelloWorld > app > src > main > java > es > uc3m > android > helloworld > MainActivity

We can add custom logs for debugging purposes

4. Proposed exercises

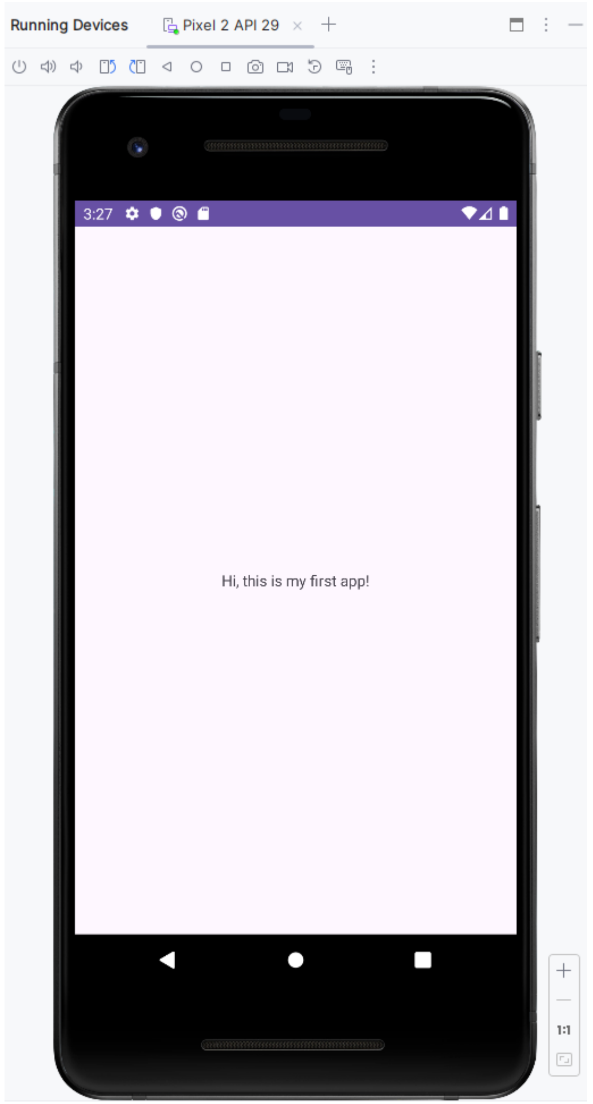
- Exercise 1. Change `TextView` text (`Hello World!`) with a string defined as a external resource:
  - Step 1. Include the following line in `strings.xml` (inside `<resources>... </resources>`):

```
<string name="hello">Hi, this is my first app!</string>
```

  - Step 2. Edit file `activity_main.xml` to use that message, as follows:

```
android:text="@string/hello"
```

  - Step 3. Execute app in the AVD



Hi, this is my first app!

4. Proposed exercises

- Exercise 2. Add log messages in the activity (Java class):
  - Step 1. Edit `MainActivity.java` including the following line at the end of the `onCreate` method:

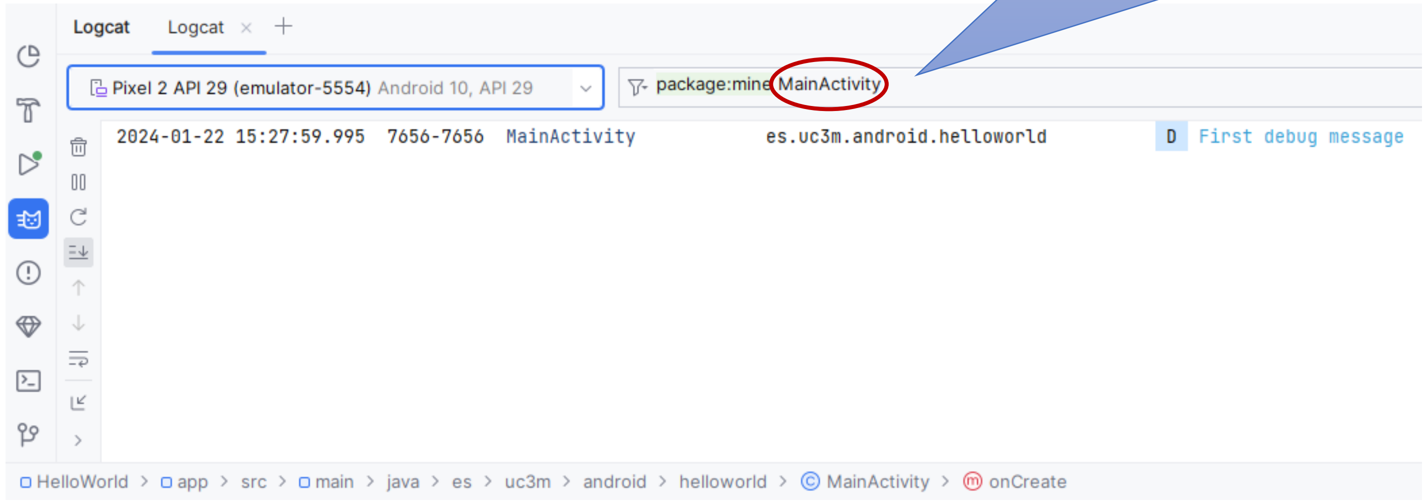
```
Log.d("MainActivity", "First debug message");
```

  - Step 2. Resolve the import:

```
import android.util.Log;
```

  - Step 3. Execute app in the AVD

We can filter the logs using the activity name (`MainActivity`) to find our debug message easily


Logcat
Logcat
Pixel 2 API 29 (emulator-5554) Android 10, API 29
package:mine MainActivity
HelloWorld > app > src > main > java > es > uc3m > android > helloworld > MainActivity > onCreate

```
2024-01-22 15:27:59.995  7656-7656  MainActivity            es.uc3m.android.helloworld           D  First debug message
```

4. Proposed exercises

- Exercise 3. Modify theme:
  - Step 1. Modify theme base in `themes.xml` (`parent` attribute):

```
<style name="Base.Theme.HelloWorld" parent="Theme.AppCompat.DayNight.DarkActionBar">
```

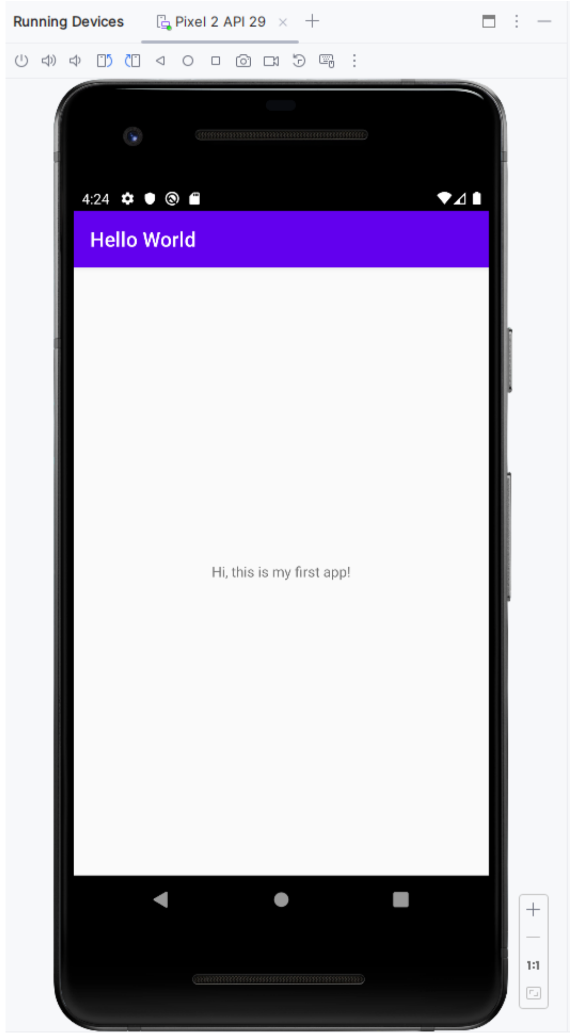
  - Step 2. Include the following line in `colors.xml` (inside `<resources>... </resources>`):

```
<color name="purple_500">#FF6200EE</color>
```

  - Step 2. Include item for primary color in `themes.xml` (inside base `<style>... </style>`):

```
<item name="colorPrimary">@color/purple_500</item>
```

  - Step 4. Execute app in the AVD


Running Devices
Pixel 2 API 29
4:24
Hello World
Hi, this is my first app!
1:1