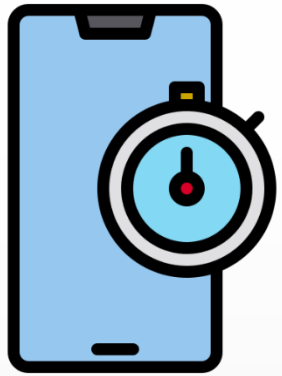
Intro to Java for Android Platform

Dmytro Zubov, PhD
dmytro.zubov@ucentralasia.org

Naryn, 12:20pm, Sept 8, 2022

Lessons learnt last time

- Outline of the course
- Syllabus
- Course project
- Intro to Mobile App development
- HTML5 offline web apps



What we gonna discuss today?

- Intro to Java programming language
- History of Android OS briefly
- Get the free software we need
- Analyzing an Android App structure
- Creating an Android app

What is Java programming language?

- Java is a general-purpose computer programming language that is **concurrent**, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible.
- It is intended to let application developers "write once, run anywhere", meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture.

https://en.wikipedia.org/wiki/Java_(programming_language)

What is Java programming language?

- In 1991, a group led by James Gosling and Patrick Naughton at Sun Microsystems designed a programming language, code-named “Green”, for use in consumer devices, such as intelligent television “set-top” boxes. The language was designed to be simple, secure, and usable for many different processor types.
- Gosling recounts that in 1994 the team realized, “We could write a really cool browser. It was one of few things in the client/server mainstream that needed some of the weird things we’d done: architecture neutral, real-time, reliable, secure.”
- Java was introduced to an enthusiastic crowd at the SunWorld exhibition in 1995, together with a browser that ran applets – Java-code that can be located anywhere on the Internet.

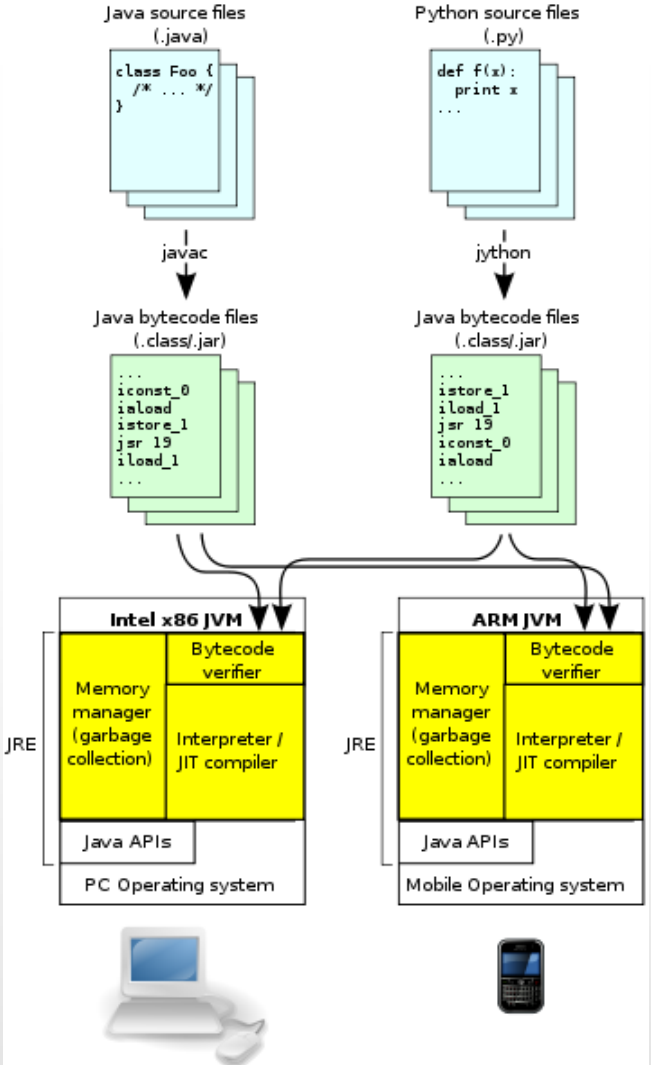
Java's Versions and Editions

- Java Card for smart-cards
- Java Platform, Micro Edition (Java ME) – targeting environments with limited resources
- Java Platform, Standard Edition (Java SE) – targeting workstation environments
- Java Platform, Enterprise Edition (Java EE) – targeting large distributed enterprise or Internet environments

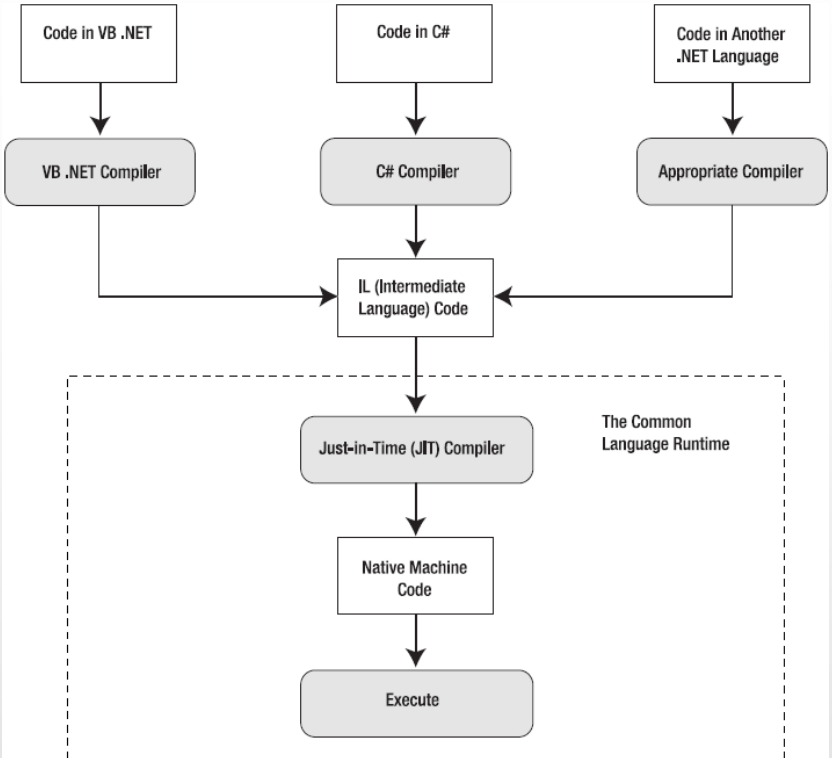
Version	Date
JDK Beta	1995
JDK1.0	January 23, 1996
JDK 1.1	February 19, 1997
J2SE 1.2	December 8, 1998
J2SE 1.3	May 8, 2000
J2SE 1.4	February 6, 2002
J2SE 5.0	September 30, 2004
Java SE 6	December 11, 2006
Java SE 7	July 28, 2011
Java SE 8	March 18, 2014
Java SE 9	September 21, 2017
Java SE 10	March 20, 2018
Java SE 11	September 25, 2018[41]
Java SE 12	March 19, 2019
Java SE 13	September 17, 2019
Java SE 14	March 17, 2020
Java SE 15	September 15, 2020
Java SE 16	March 16, 2021
Java SE 17	September 2021
Java SE 18	March 2022

https://en.wikipedia.org/wiki/Java_(programming_language)

Is the Java-technology similar to .NET-technology?

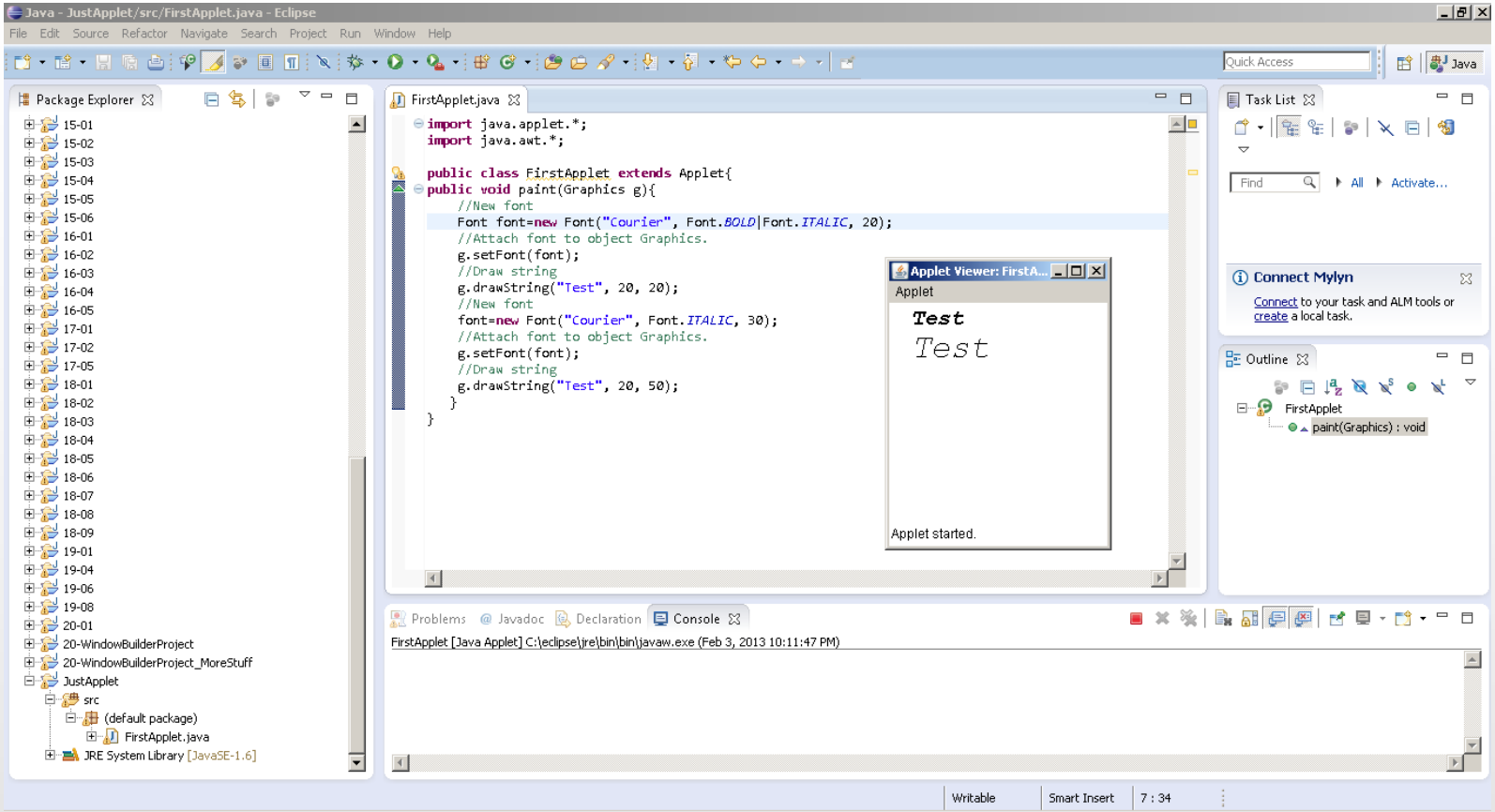


- Overview of a Java virtual machine (JVM) architecture. Source code is compiled to Java bytecode, which is verified, interpreted or JIT-compiled for the native architecture. The Java APIs and JVM together make up the Java Runtime Environment (JRE).



Java IDEs

- Eclipse, NetBeans, IntelliJ IDEA, JDeveloper, JCreator, etc.



Java applets

- Java applets are Java-programs that are embedded in other applications, typically in a web-page displayed in a web-browser (this Java-applet is equal to the above Java-applet in Eclipse):

FirstApplet.java - Notepad

File Edit Format View Help

```
import java.applet.*;
import java.awt.*;
public class FirstApplet extends Applet{
public void paint(Graphics g){
        //New font
        Font font=new Font("Courier", Font.BOLD|Font.ITALIC, 20);
        //Attach font to object Graphics.
        g.setFont(font);
        //Draw string
        g.drawString("Test", 20, 20);
        //New font
        font=new Font("Courier", Font.ITALIC, 30);
        //Attach font to object Graphics.
        g.setFont(font);
        //Draw string
        g.drawString("Test", 20, 50);
   }
}
```

Ln 1, Col 1

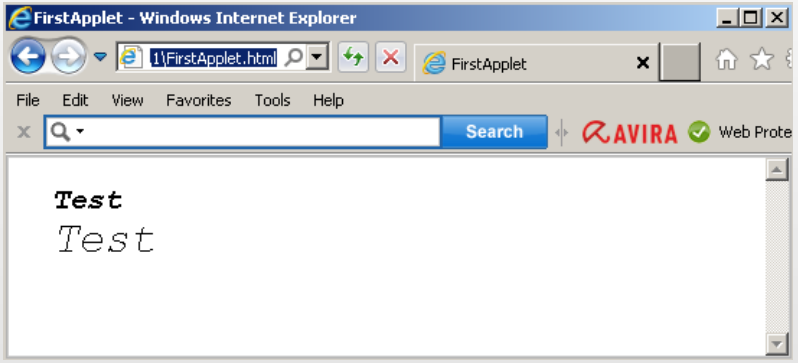
HTML-file uses .class-file which is created by javac.exe (java-compiler). javac.exe is located in the folder
“c:\Program Files (x86)\Java\jdk1.7.0_13\bin\”
Also, Eclipse can create .class file from .java-file (“bin” folder of the java-project).

FirstApplet - Notepad

File Edit Format View Help

```
<html>
<head>
<title>FirstApplet</title>
</head>
<body>
        <applet code="FirstApplet.class" width="100" height="100"></applet>
</body>
```

Ln 1, Col 1



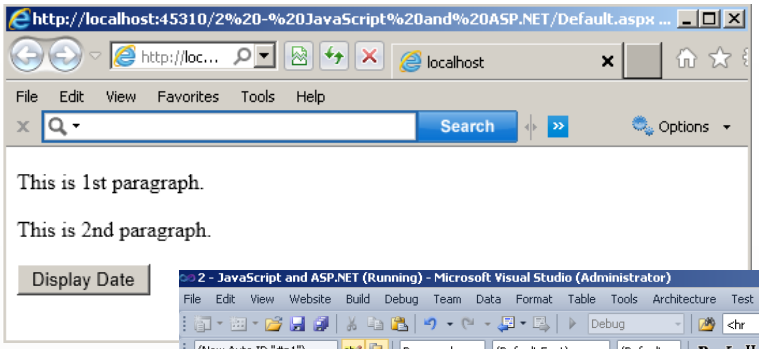
JavaScript is not Java

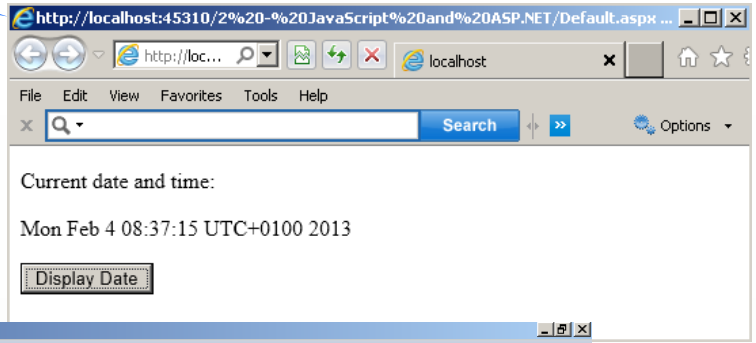
- JavaScript is a high-level programming language. It was originally designed as a scripting language for websites but became widely adopted as a general-purpose programming language and is currently the most popular programming language in use.

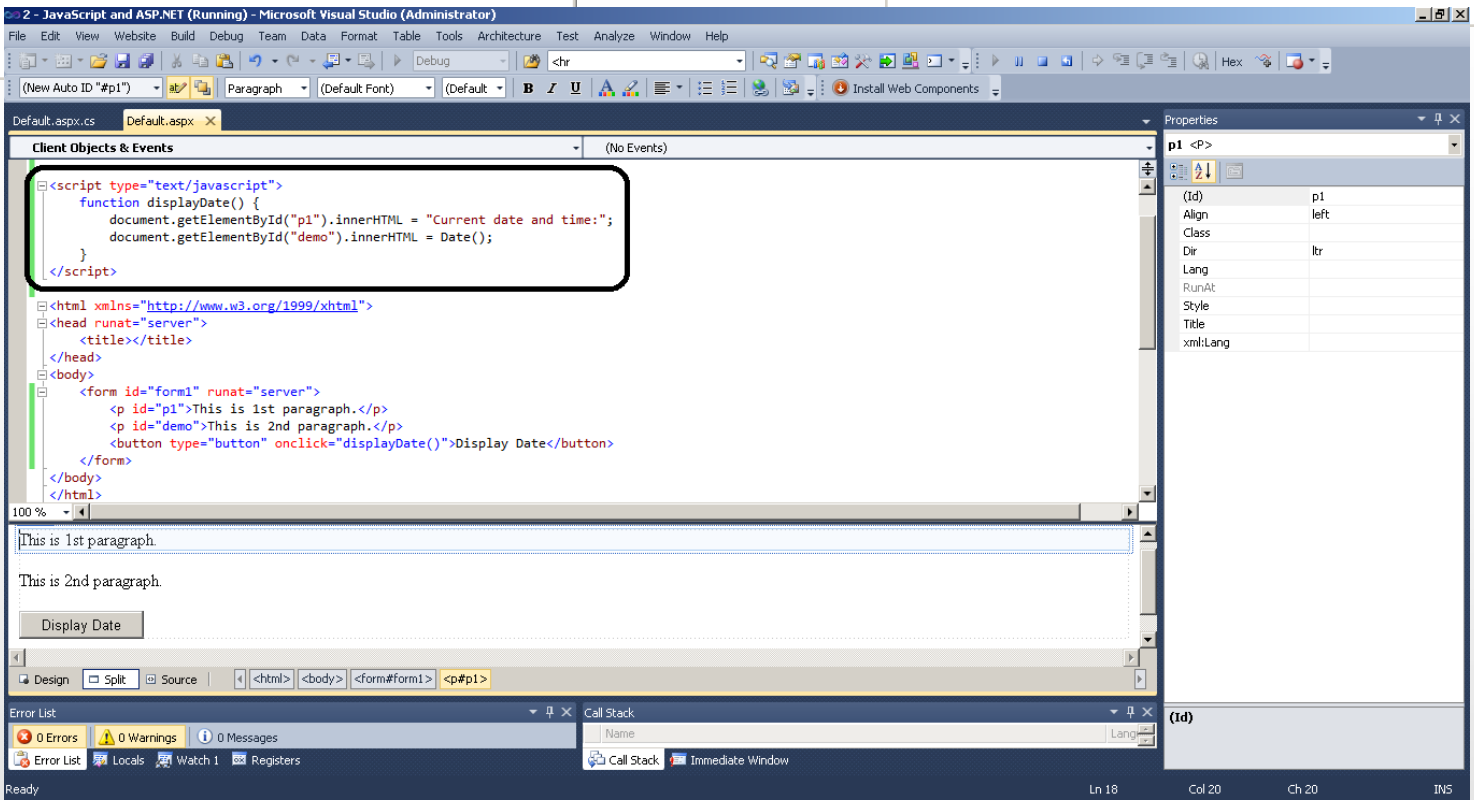
- Despite the similarities in name and syntax, JavaScript is not related to the programming language Java. Though the names of both languages are trademarks of Oracle Corporation, the two languages follow different design principles, and are actively developed by unrelated organizations.

- JavaScript is usually found running in a web browser as interactive or automated content, ranging from popup messages and live clocks to large web applications. JavaScript is also commonly used in server-side programming through platforms like Node.js, or "embedded" in non-JavaScript applications where the base programming language lacks the high-level functionality that JavaScript offers.

An example of the ASP.NET with JavaScript







History of Android OS Briefly

- Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablets. Android is developed by a consortium of developers known as the Open Handset Alliance and commercially sponsored by Google. It was unveiled in November 2007, with the first commercial Android device launched in September 2008.

- In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars, Android Wear for wrist watches, Android Glass for Google Glass, each with a specialized user interface. Variants of Android are also used on game consoles, digital cameras, PCs and other electronics.

https://en.wikipedia.org/wiki/Android_(operating_system)

Versions of Android Platform

- Android has been the best-selling OS worldwide on smartphones since 2011 and on tablets since 2013.
- As of May 2021, it has over three billion monthly active users, the largest installed base of any operating system, and as of January 2021, the Google Play Store features over 3 million apps.

Android 13, released on August 15, 2022, is the latest version.

https://en.wikipedia.org/wiki/Android_(operating_system)

Version	Marketing name	Release date	API level	Kernel	Runtime	Launched with
13			33	5.x	ART	Pixel 4, Pixel 4 XL, Pixel 5, Pixel 5a, Pixel 6, Pixel 6 Pro, Asus ZenFone 8, Lenovo P12 Pro, OnePlus 10 Pro, Oppo Find X5 Pro, Vivo X80 Pro, Realme GT2 Pro, Xiaomi 12, Xiaomi 12 Pro, Xiaomi Pad 5, Redmi K50 Pro, Sharp AQUOS sense6, Tecno Camon 19 Pro, ZTE Axon 40 Ultra[417]
12L	12	March 7, 2022	32	5.x	ART	Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, Pixel 4, Pixel 4 XL, Pixel 4a, Pixel 5, Pixel 5a
12		October 4, 2021	31	5.x	ART	Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, Pixel 4, Pixel 4 XL, Pixel 5, Pixel 6, Pixel 6 Pro, Asus ZenFone 8, Nokia X20, OnePlus 9, OnePlus 9 Pro, Oppo Find X3 Pro, iQOO 7 Legend, Realme GT,[418] TCL 20 Pro 5G, Xiaomi Mi 11, Xiaomi Mi 11 Ultra, Xiaomi Mi 11i/Mi 11X Pro,[419] Tecno Camon 17, ZTE Axon 30 Ultra
11	11	September 8, 2020	30	5.x	ART	Pixel 2, Pixel 2 XL, Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, Pixel 4, Pixel 4 XL,[420] OnePlus 8, OnePlus 8 Pro, Oppo Find X2, Oppo Find X2 Pro, Vivo NEX 3S, Xiaomi Mi 10, Xiaomi Mi 10 Pro, POCO F2 Pro,[421] Realme X50 Pro, Sharp AQUOS Zero 2
10	10	September 3, 2019	29	5.x	ART	Asus ZenFone 5Z, Essential Phone, Pixel, Pixel XL, Pixel 2, Pixel 2 XL, Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, OnePlus 6, OnePlus 6T, OnePlus 7, OnePlus 7 Pro, Oppo Reno, Sony Xperia XZ3, Vivo X27, Vivo NEX S, Vivo NEX A, Xiaomi Mi MIX 3 5G, Xiaomi Mi 9, Tecno Spark 3 Pro, Huawei Mate 20 Pro, LG G8, Nokia 8.1, Realme 3 Pro[422]
9	Pie	August 6, 2018	28	4.x	ART	Essential Phone, Pixel, Pixel XL, Pixel 2, Pixel 2 XL, Nokia 7 Plus, OnePlus 6, Oppo R15 Pro, Sony Xperia XZ2, Vivo X21UD, Vivo X21, Xiaomi Mi Mix 2S[423]
8.1	Oreo	December 5, 2017	27	4.x	ART	Pixel, Pixel XL, Nexus 6P, Nexus 5X
8.0		August 21, 2017	26	4.x	ART	—
7.1	Nougat	October 4, 2016	25	4.x	ART	Pixel, Pixel XL
7.0		August 22, 2016	24	4.x	ART	Nexus 5X, Nexus 6P, LG V20
6.0	Marshmallow	October 5, 2015	23	4.x	ART	Nexus 5X, Nexus 6P
5.1	Lollipop	March 9, 2015	22	3.x	ART	Android One
5.0		November 3, 2014	21	3.x	ART 2.1.0	Nexus 6, Nexus 9
4.4	KitKat	October 31, 2013	19	3.x	Dalvik (and ART 1.6.0)	Nexus 5
4.3	Jelly Bean	July 24, 2013	18	3.x	Dalvik	Nexus 7 2013
4.2		November 13, 2012	17	3.x	Dalvik	Nexus 4, Nexus 10
4.1		July 9, 2012	16	3.x	Dalvik	Nexus 7
4.0	Ice Cream Sandwich	October 19, 2011	15	3.x	Dalvik	Galaxy Nexus
2.3	Gingerbread	February 9, 2011	10	2.6.32	Dalvik 1.4.0	Nexus S

What is Dalvik software?

- Dalvik software is a discontinued process virtual machine in Google's Android operating system that executes applications written for Android
- Dalvik software is an integral part of the Android software stack in Android versions 4.4 "KitKat" and earlier, which is typically used on mobile devices such as mobile phones and tablet computers, and more recently on devices such as smart TVs and wearables
- Dalvik is open-source software, originally written by Dan Bornstein, who named it after the fishing village of Dalvík in Eyjafjörður, Iceland

https://en.wikipedia.org/wiki/Dalvik_(software)

What is Dalvik software? (cont.)

- Programs for Android are commonly written in Java and compiled to bytecode for the Java virtual machine, which is then translated to Dalvik bytecode and stored in .dex (Dalvik EXecutable) and .odex (Optimized Dalvik EXecutable) files; related terms odex and de-odex are associated with respective bytecode conversions
- The compact Dalvik Executable format is designed for systems that are constrained in terms of memory and processor speed

https://en.wikipedia.org/wiki/Dalvik_(software)

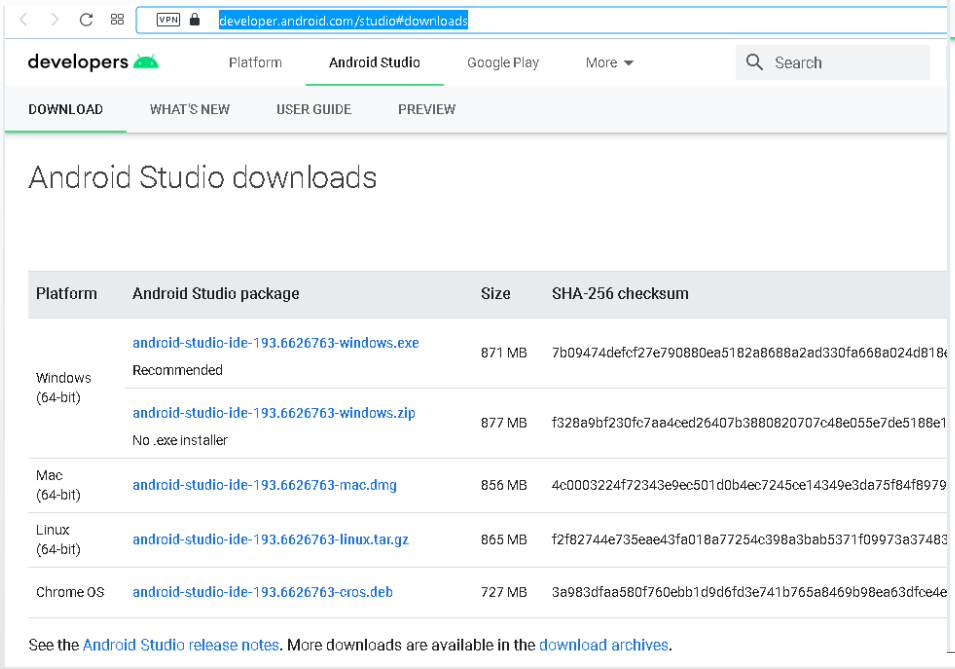
What is ART?

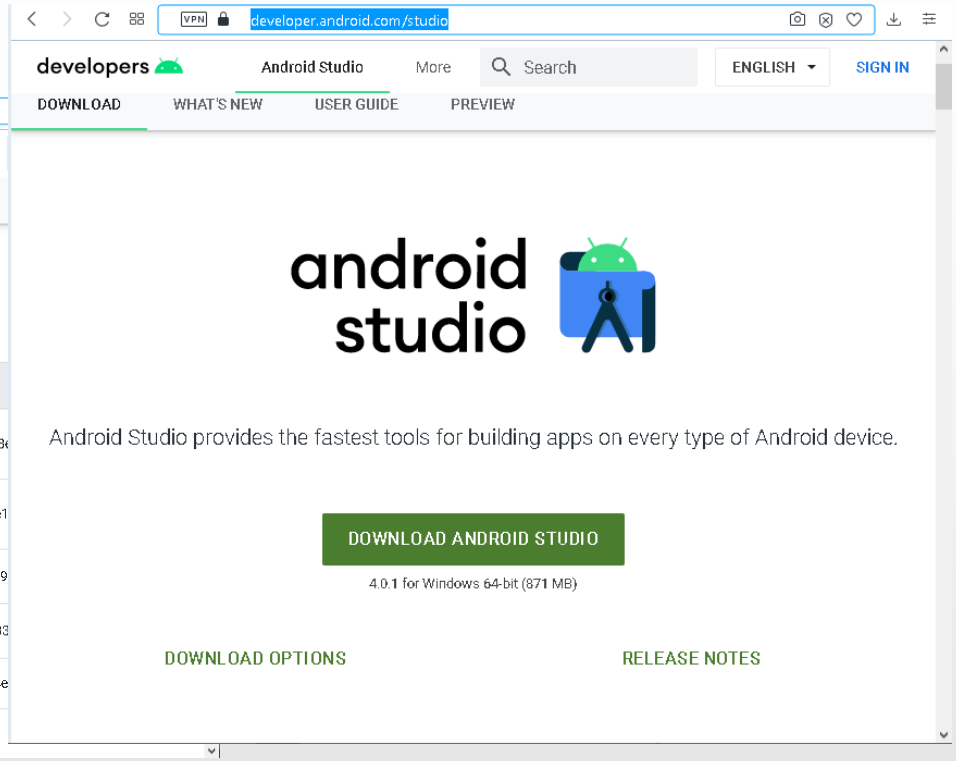
- Android Runtime (ART) is an application runtime environment used by the Android operating system. Replacing Dalvik, which is the process virtual machine originally used by Android, ART performs the translation of the application's bytecode into native instructions that are later executed by the device's runtime environment

- Unlike Dalvik, ART introduces the use of ahead-of-time (AOT) compilation by compiling entire applications into native machine code upon their installation. By eliminating Dalvik's interpretation and trace-based JIT compilation, ART improves the overall execution efficiency and reduces power consumption, which results in improved battery autonomy on mobile devices. At the same time, ART brings faster execution of applications, improved memory allocation and garbage collection (GC) mechanisms, new applications debugging features, and more accurate high-level profiling of applications.

Get the Free Software We Need

- Visit http://developer.android.com/studio and download the Android Studio IDE for your platform (Windows, Mac, Linux, Chrome OS)





Install Android Studio

https://developer.android.com/studio/install.html

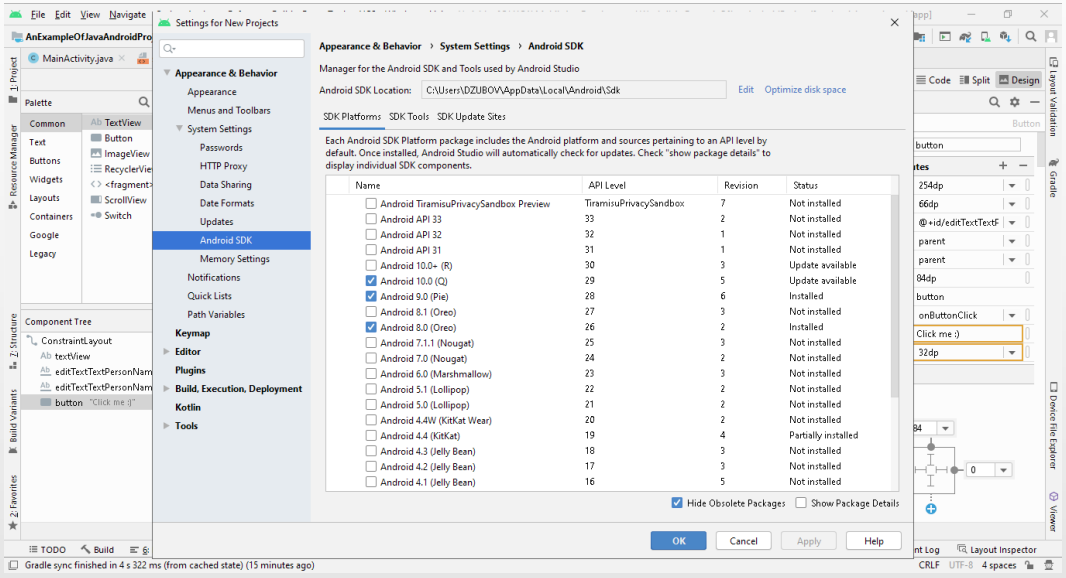
Install Android Studio (cont.)

- To install Android Studio on Windows, proceed as follows:
  - If you downloaded an .exe file (recommended), double-click to launch it.
  - If you downloaded a .zip file, unpack the ZIP, copy the android-studio folder into your Program Files folder, and then open the *android-studio > bin folder* and launch studio64.exe (for 64-bit machines) or studio.exe (for 32-bit machines).
  - Follow the setup wizard in Android Studio and install any SDK packages that it recommends.
  - As new tools and other APIs become available, Android Studio tells you with a pop-up, or you can check for updates by clicking *Help > Check for Update*.

Update the IDE and SDK Tools

- Android Studio notifies us with a small bubble dialog when an update is available for the IDE, but we can manually check for updates by clicking *Help > Check for Updates* (on Mac, Android Studio > Check for Updates).
- Update the tools with the SDK Manager

° The Android SDK Manager (*Tools > SDK*) helps us download the SDK tools, platforms, and other components we need to develop apps. Once downloaded, we can find each package in the directory indicated as the Android SDK Location



Update the IDE and SDK Tools (cont.)

- Recommended packages:
  - Android SDK Build-Tools: *Required*. Includes tools to build Android apps. See the SDK Build Tools release notes.
  - Android SDK Platform-Tools: *Required*. Includes various tools required by the Android platform, including the adb tool.
  - Android SDK Tools: *Required*. Includes essential tools such as ProGuard.
  - Android Emulator: *Recommended.* A QEMU-based device-emulation tool that we can use to debug and test our apps in an actual Android runtime environment.
- To see all available packages for each Android platform, click *Show Package Details* at the bottom of the window.

Java Android App: Developer Workflow

1. Set up the workspace: Install Android Studio and create a project

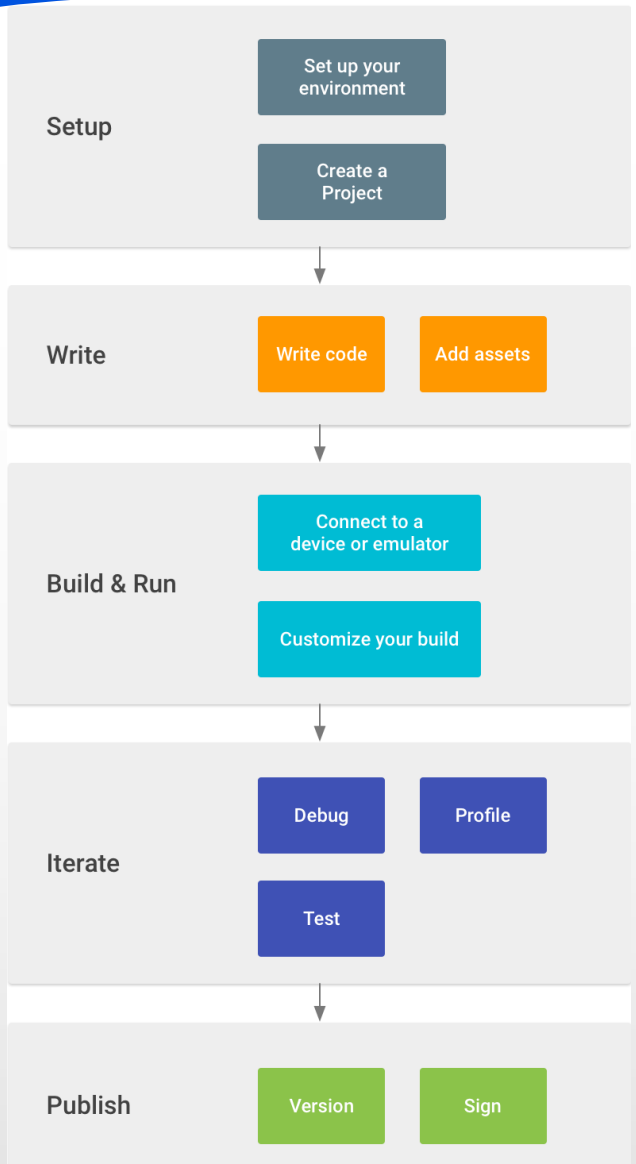
2. Write the app

APK (.apk) and OBB files are simply raw form of an Android smartphone app

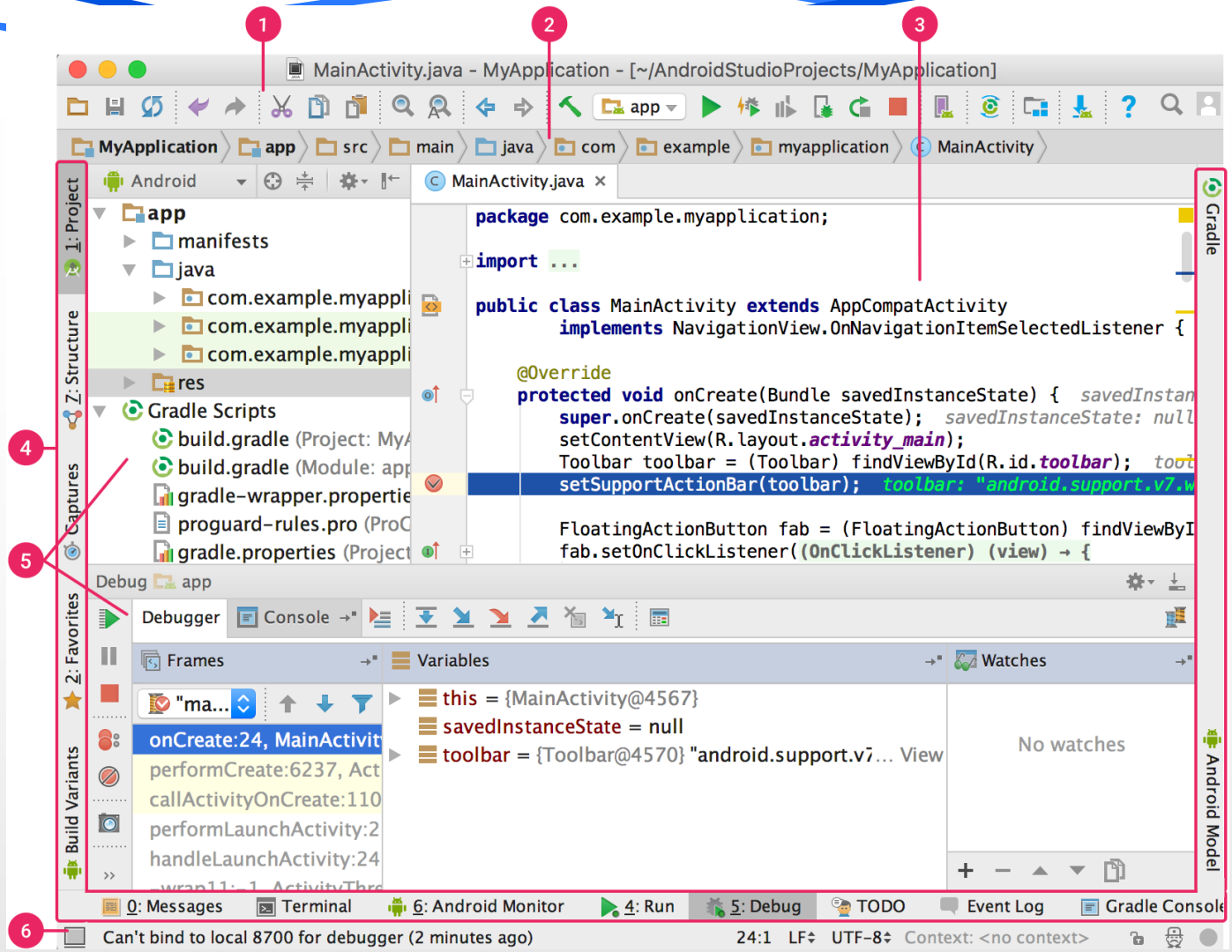
3. Build and run: During this phase, we build a project into a APK and OBB files that we can install and run on the emulator or an Android-powered device

4. Debug, profile, and test: This is the iterative phase in which we continue writing the app but with a focus on eliminating bugs and optimizing app performance

5. Publish: Deliver an app to end-users (e.g., using your own website or Google Play)



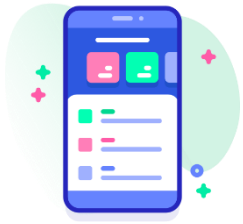
Android Studio: The User Interface



- The areas that you see on your computer screen may be different from these areas…

https://developer.android.com/studio/intro

Android Studio: The User Interface (cont.)



1. The **toolbar** lets us carry out a wide range of actions, including running the app and launching Android tools

2. The **navigation bar** helps us navigate through the project and open files for editing. It provides a more compact view of the structure visible in the Project window

3. The **editor window** is where we create and modify code. Depending on the current file type, the editor can change, e.g., when viewing a layout file, the editor displays the Layout Editor

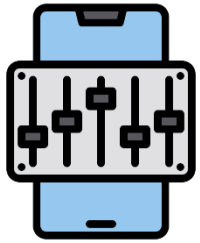
4. The **tool window bar** runs around the outside of the IDE window and contains the buttons that allow us to expand or collapse individual tool windows

5. The **tool windows** give us access to specific tasks like project management, search, version control, and more. We can expand them and collapse them.

6. The **status bar** displays the status of the project and the IDE itself, as well as any warnings or messages

https://developer.android.com/studio/intro

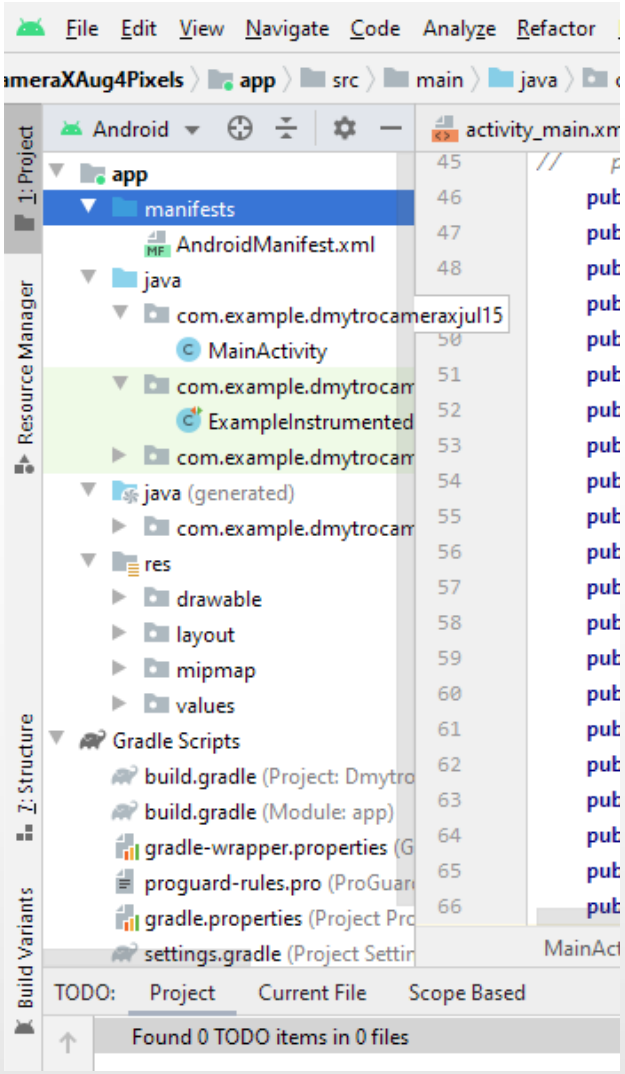
Android Studio: Gradle Build System



- Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle. We can use the features of the build system to do the following:
  - Customize, configure, and extend the build process
  - Create multiple APKs for our app, with different features using the same project and modules
  - Reuse code and resources across source sets
- By employing the flexibility of Gradle, we can achieve all of this without modifying our app's core source files. Android Studio build files are named build.gradle

Android Studio: Project files

- By default, Android Studio displays the project files in the Android view. This view doesn't reflect the actual file hierarchy on disk but is organized by modules and file types to simplify navigation between key source files of the project, hiding certain files or directories that are not commonly used.



Android Studio: Project files (cont.)

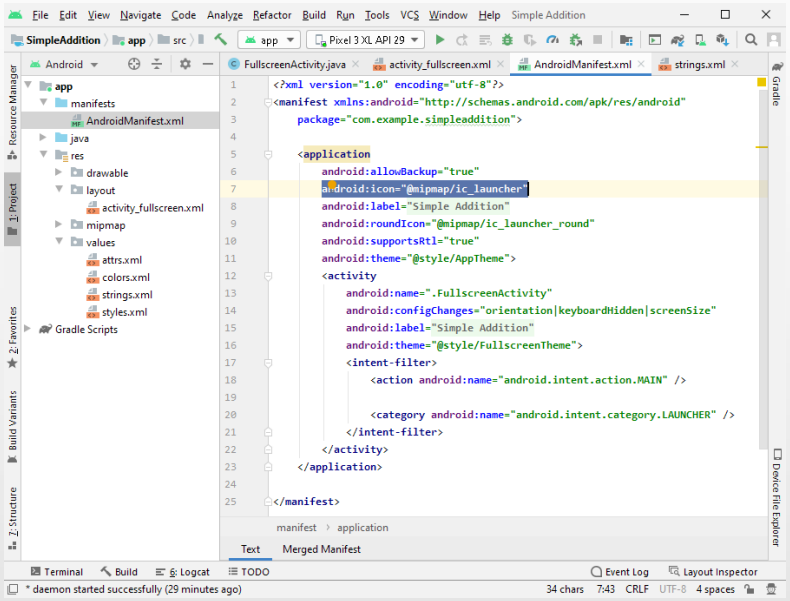
- Within each Android app module, files are shown in the following groups:
  - ***manifests***: Contains the `AndroidManifest.xml` file
  - ***java***: Contains the Java source code files, separated by package names, including JUnit test code
  - ***res***: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images, divided into corresponding sub-directories

Understanding the Android manifest

- Before the Android system starts an app component such as an Activity, the system must know that the Activity exists. It does so by reading the app's `AndroidManifest.xml` file, which describes all components of the Android app.

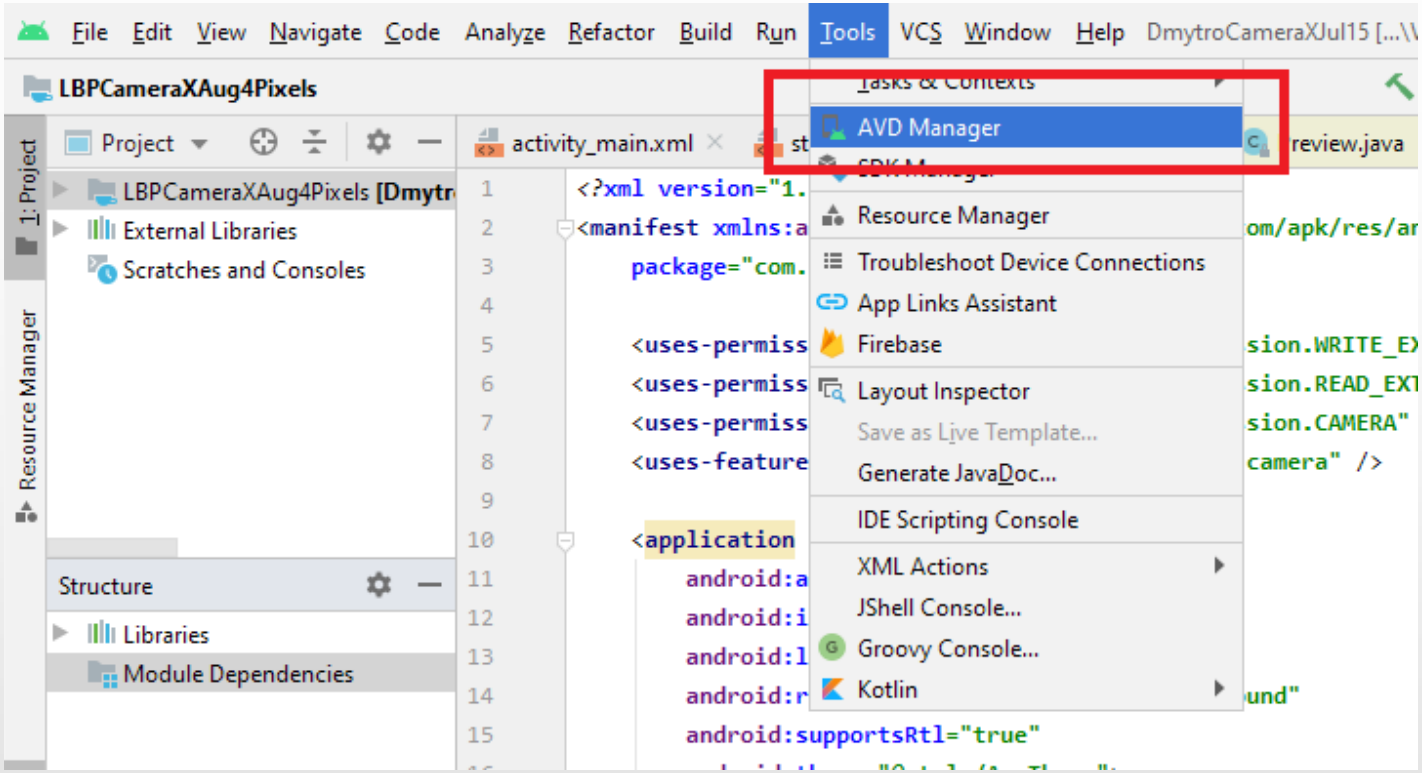
- The `android:icon` attribute sets the icon for the app:

```
android:allowBackup="true"
android:icon="@mipmap/ic_launcher"
```

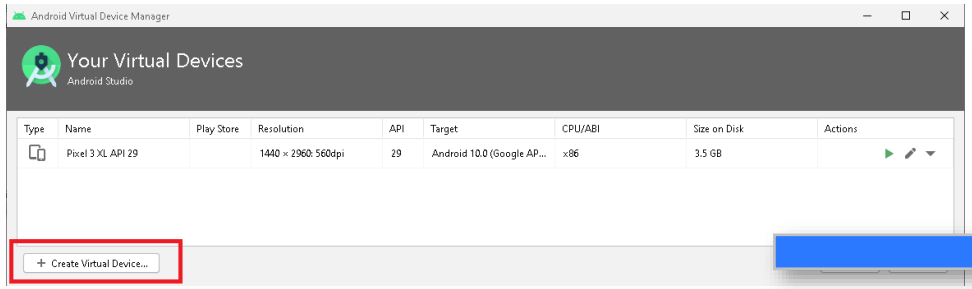


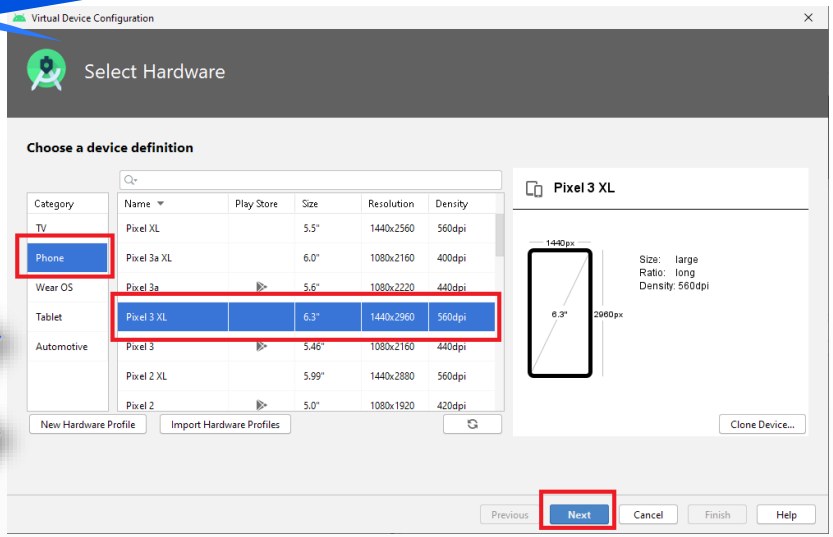
Creating an Android Virtual Device

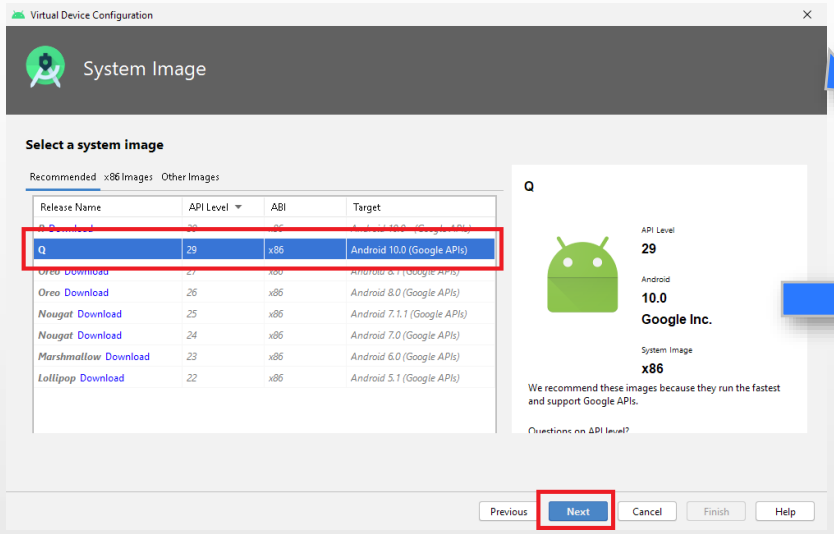
- To run the code, we must have something (a physical device or an emulated device) that can run an Android program

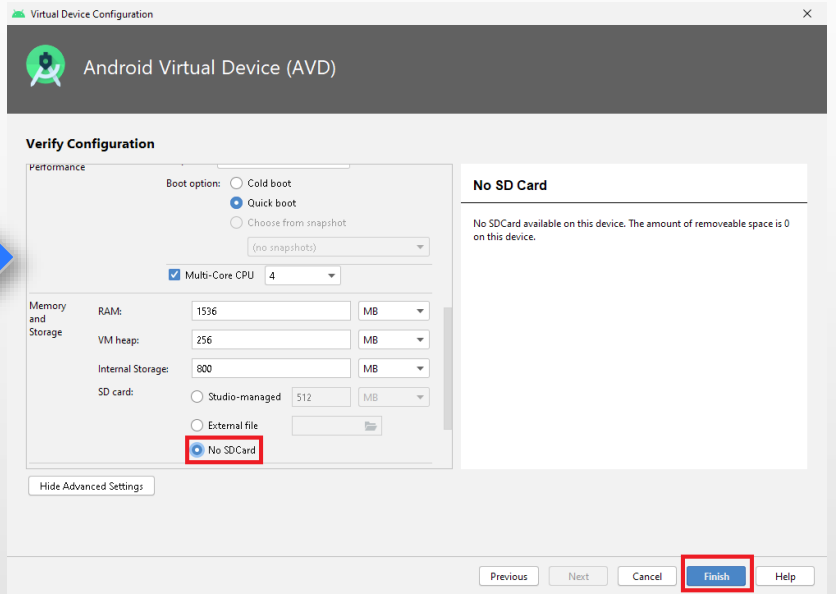


Creating an Android Virtual Device (cont.)







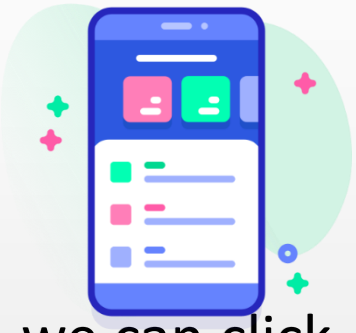


How to Run Android Studio App on the Smartphone

- Set up a device for development

1. On the device, open the Settings app, select Developer options, and then enable USB debugging

2. Set up the OS to detect the device, e.g., install a USB driver on Windows

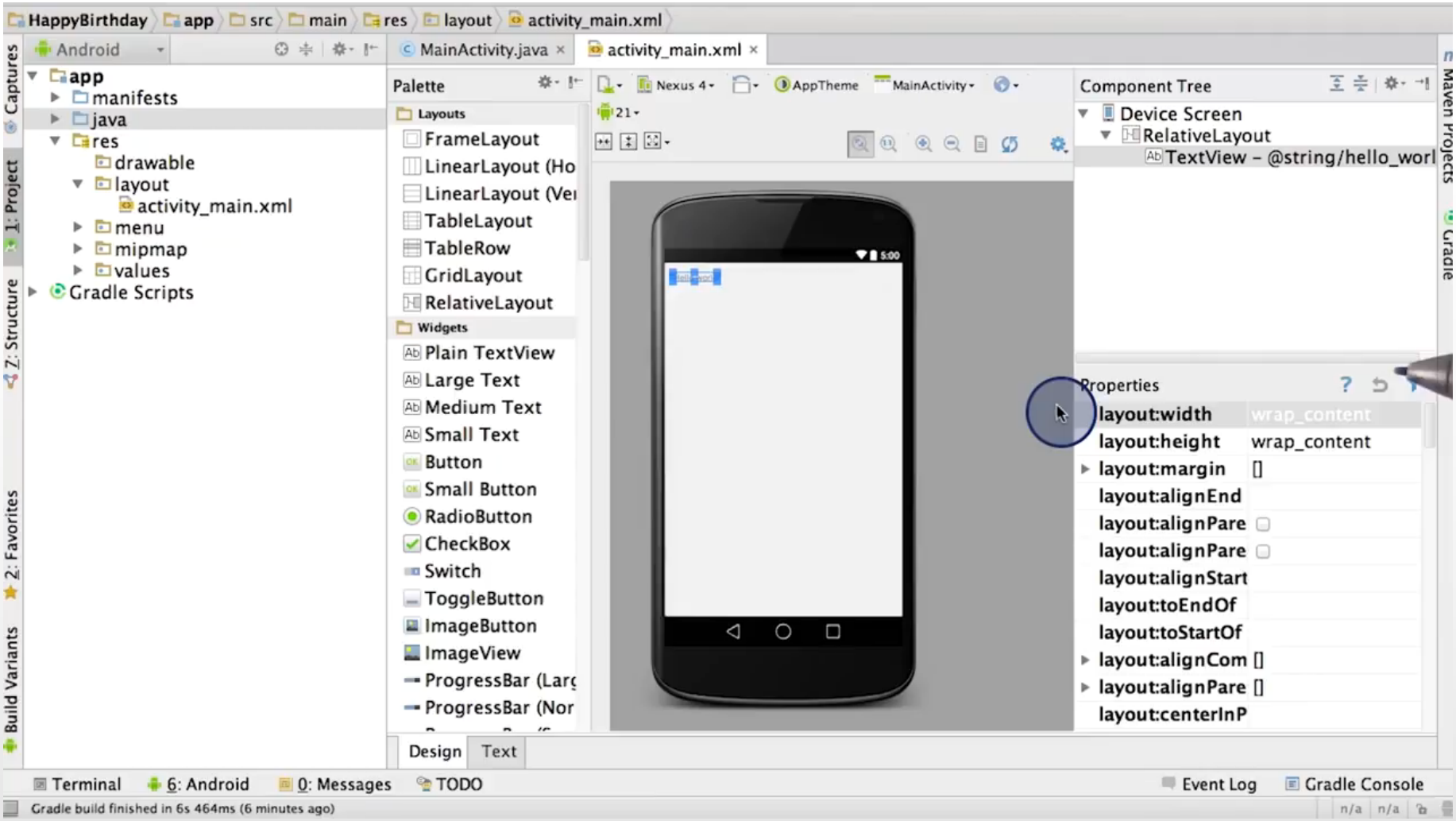


- Connect a computer to the device:

  ◦ When the device is set up and plugged in over USB, we can click **Run** in Android Studio to build and run the app on the device.

https://developer.android.com/studio/run/device

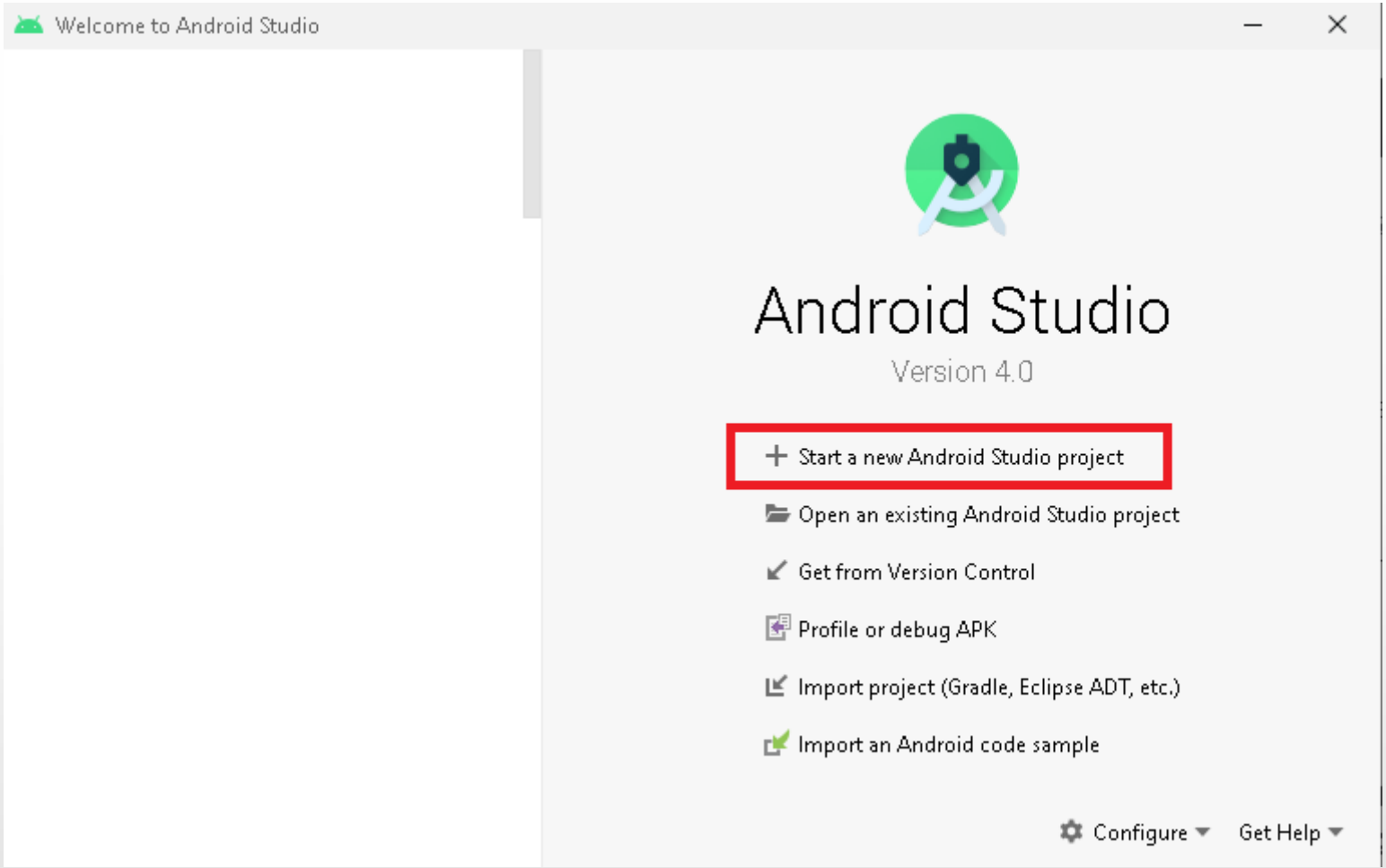
How to Run Android Studio App in the Smartphone (cont.)



https://www.youtube.com/watch?v=P6AAmcx2QXY

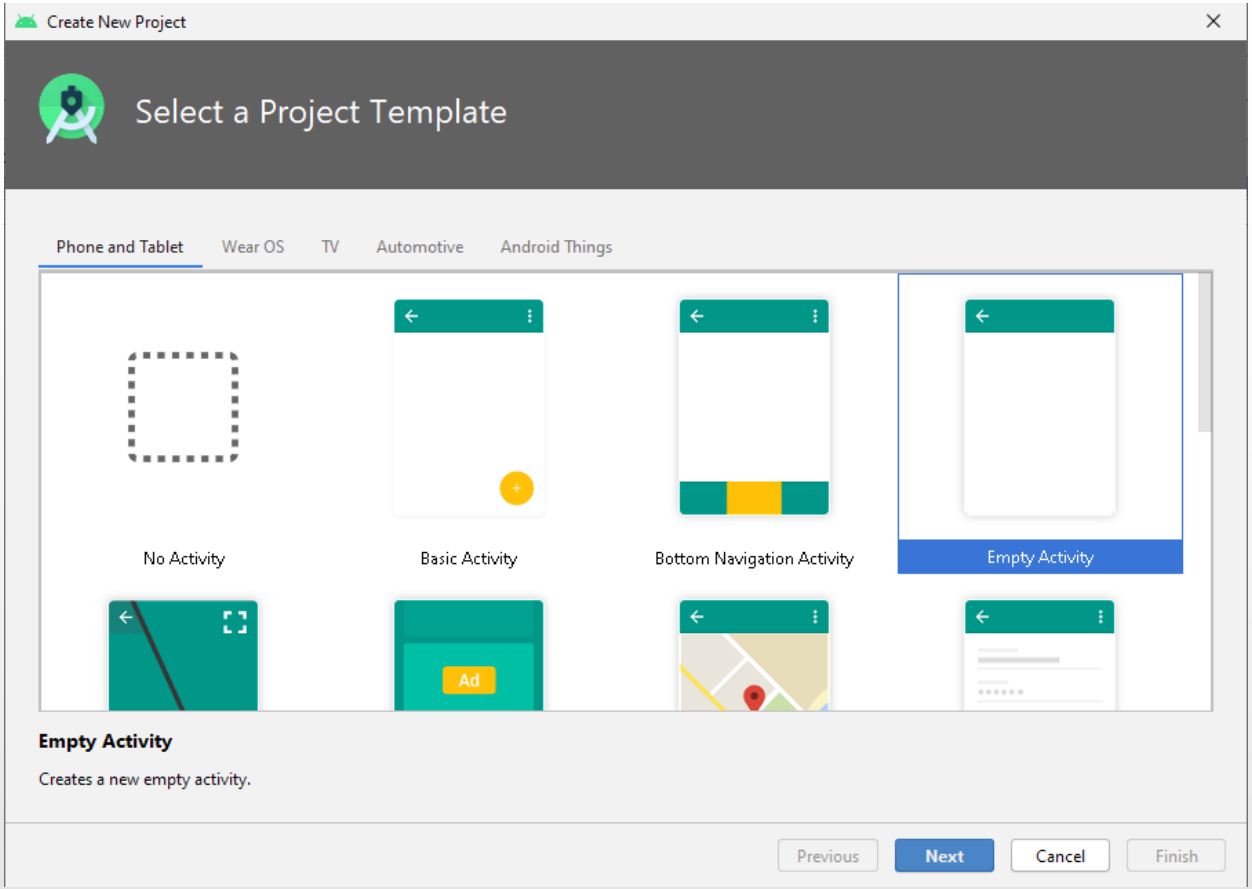
Create a project in Android Studio

- Launch Android Studio:



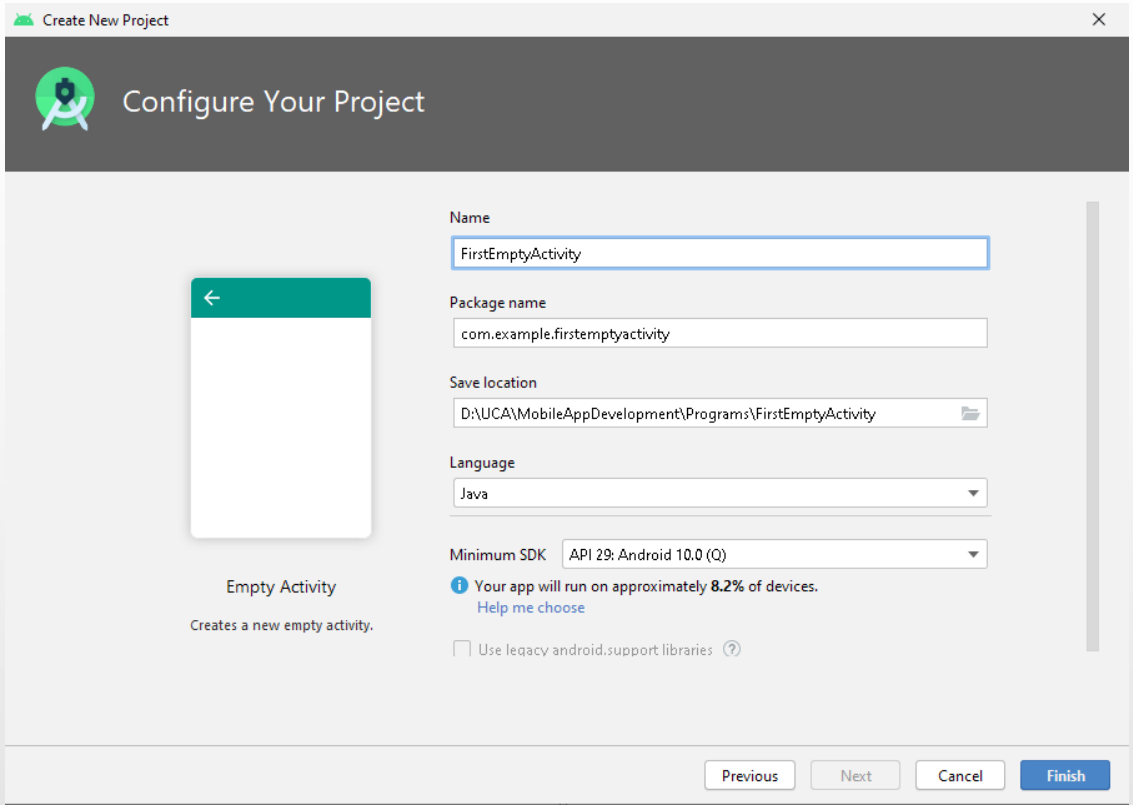
Create a project in Android Studio (cont.)

- Select the Activity option. In this example, the **Empty Activity** is chosen:



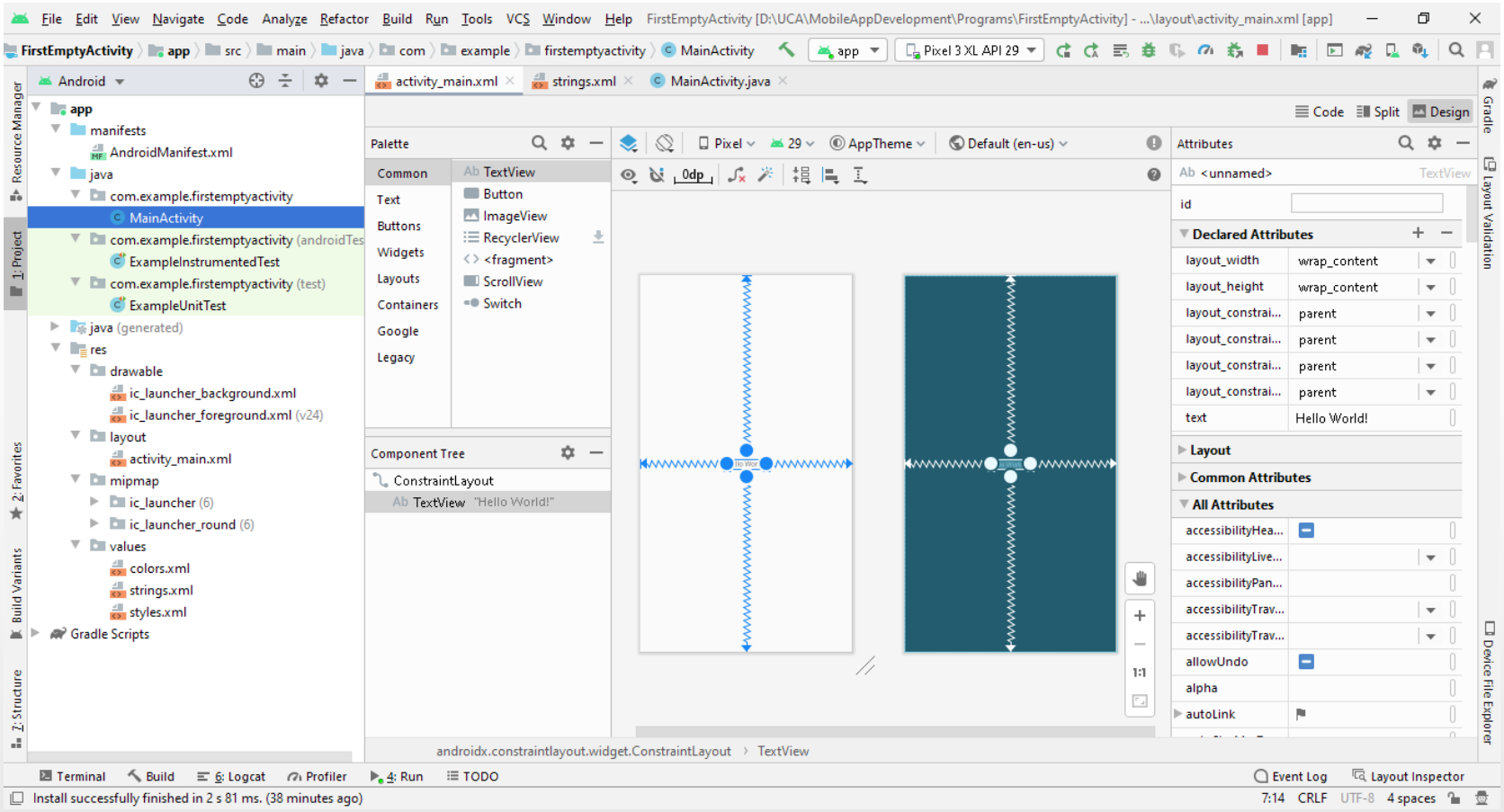
Create a project in Android Studio (cont.)

- Configure the project. In this example, we change the name to FirstEmptyActivity, location, language Java, and Minimum SDK to API 29:



Create a project in Android Studio (cont.)

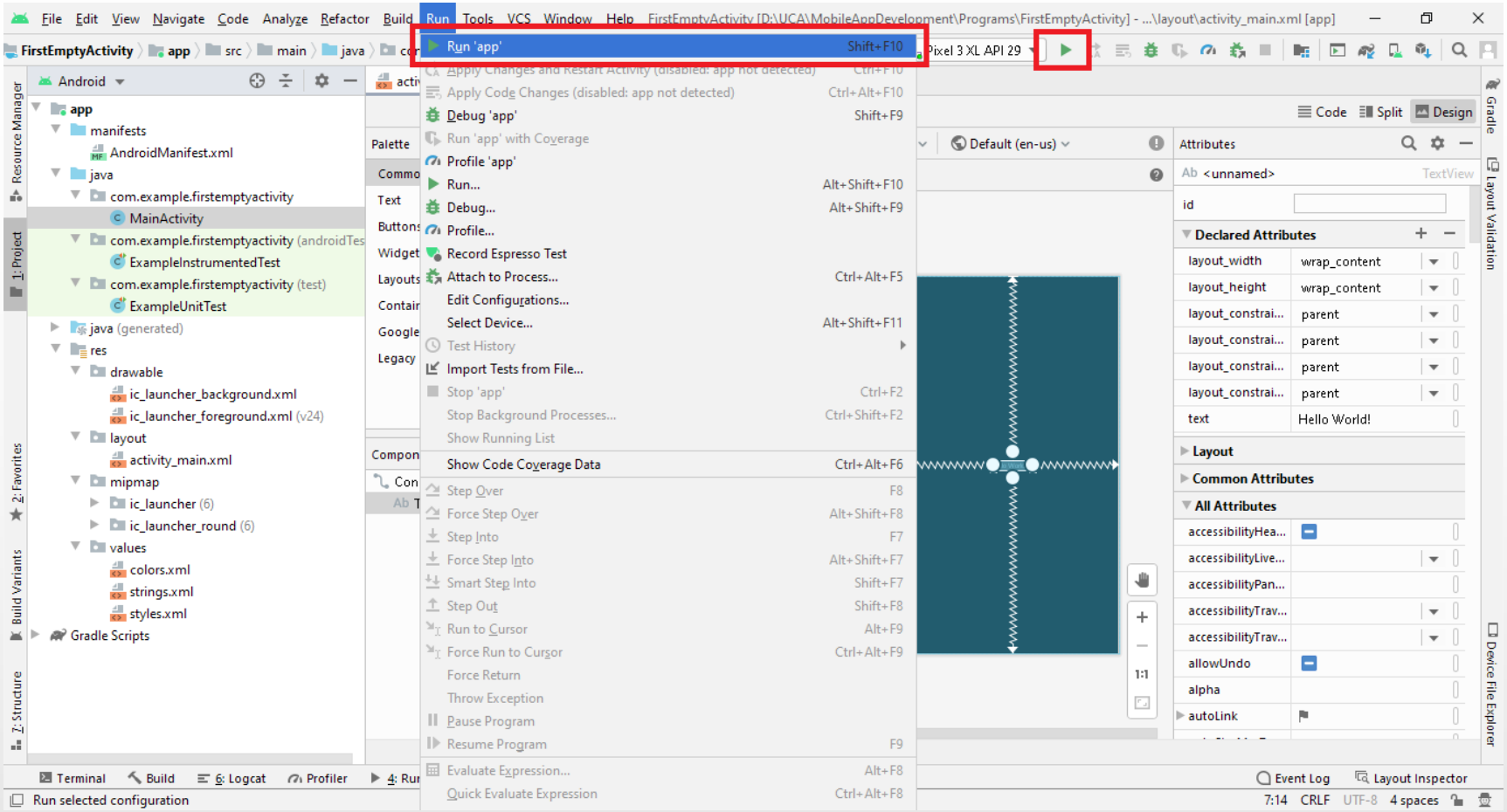
- Android Studio shows the project as follows:



Appearance can be different

Create a project in Android Studio (cont.)

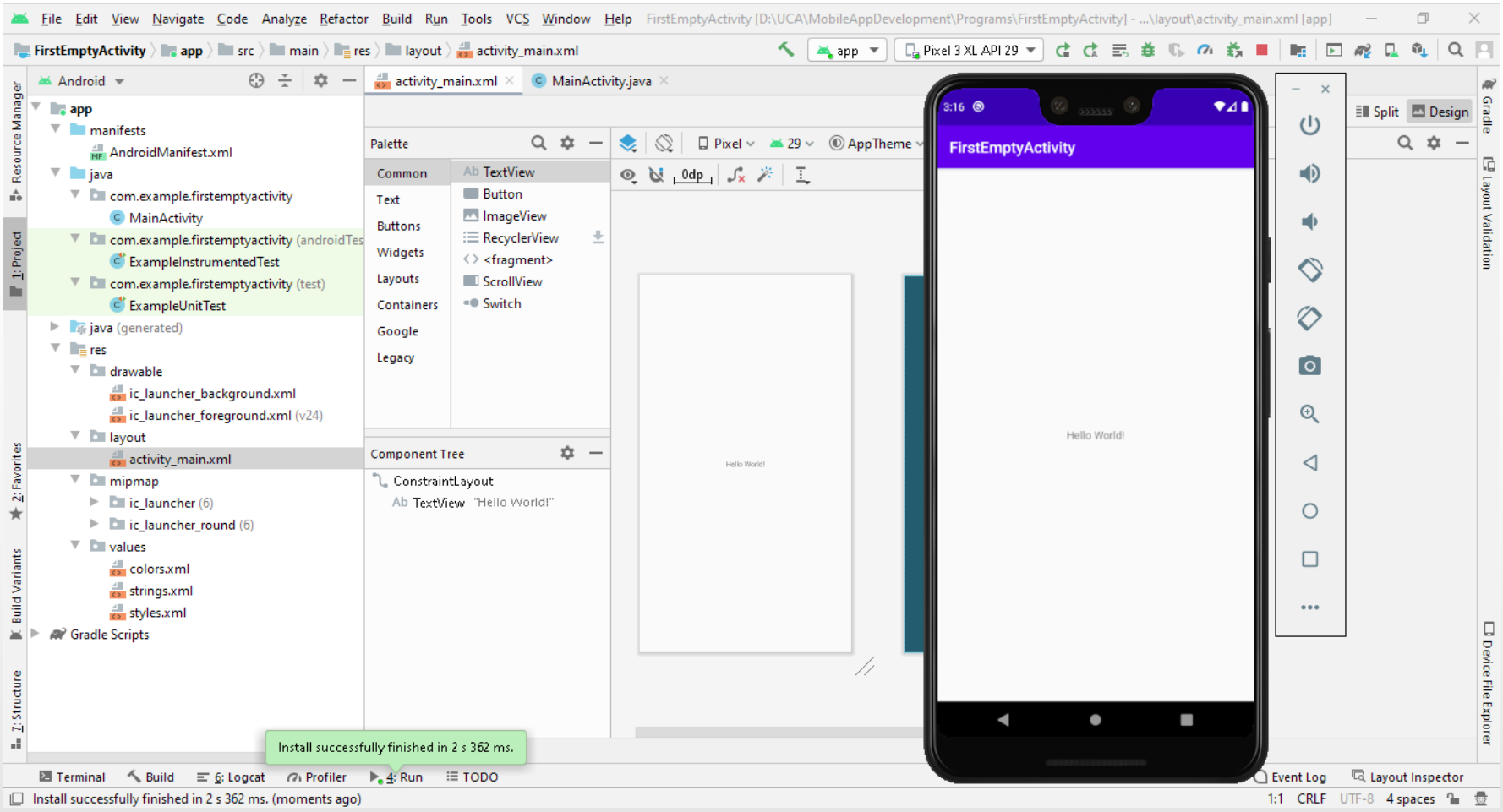
- Run the app (click triangle or use Shift+F10):



https://developer.android.com/studio/run

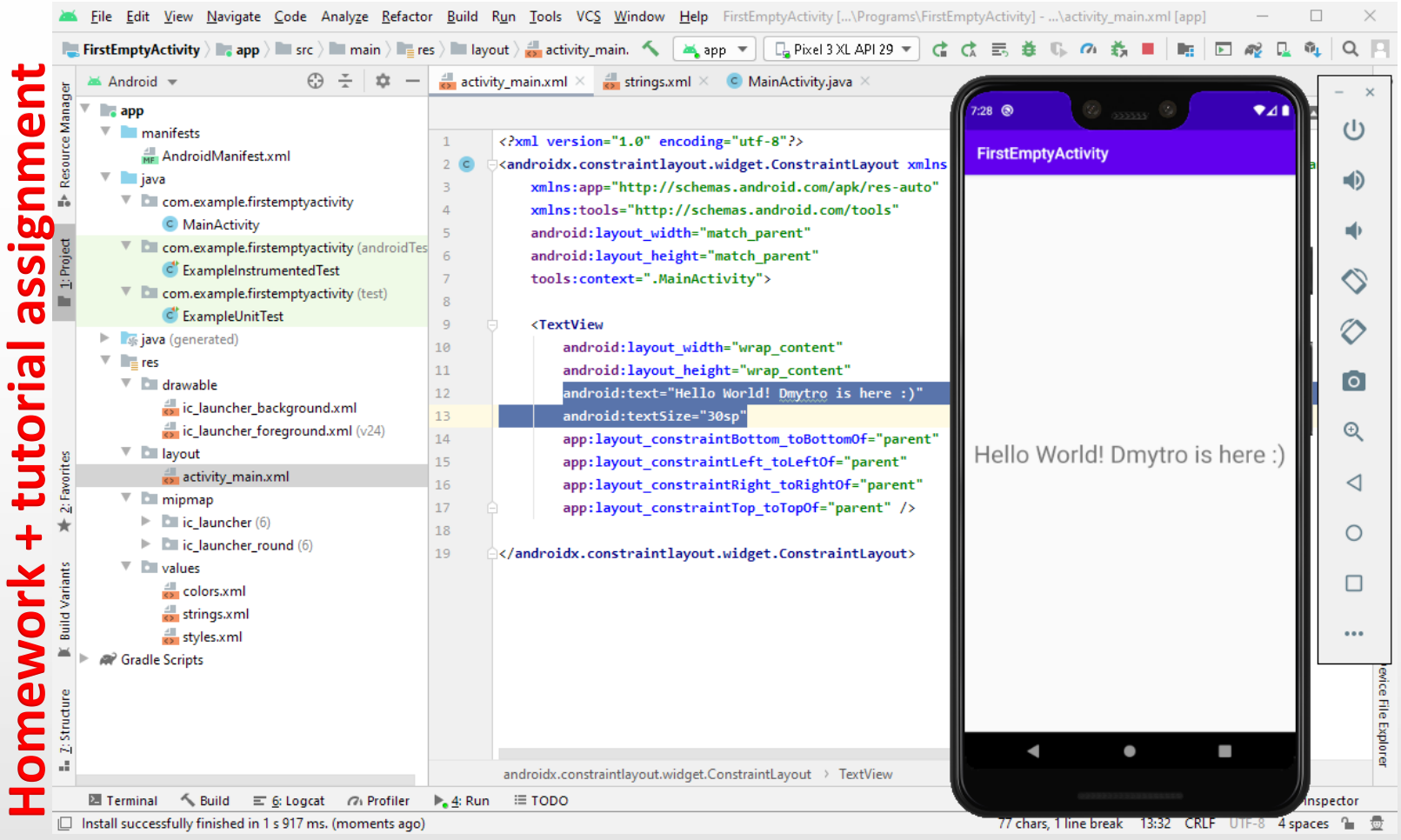
Create a project in Android Studio (cont.)

- ”Hello World!” :)



Create a project in Android Studio (cont.)

- Modify *TextView* element. You can use another name :)



What do we have inside MainActivity.java?

```
package com.allyourcode.a04_01;

import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

Inside of a package named `org.allyourcode.a04_01`...

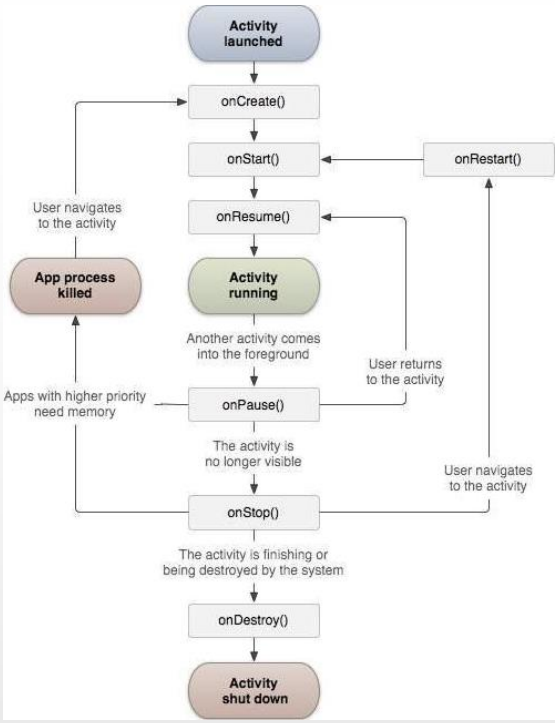
...you tell Java that, in this program, you'll be using (importing) some code that's already defined in the Android library, and then...

...you create a class named `MainActivity`.

Inside the `MainActivity` class, you make a list of instructions (a method) named `onCreate`.

Inside that `onCreate` list, you put two statements: one statement calls the `super.onCreate` method; another statement calls the `setContentView` method.

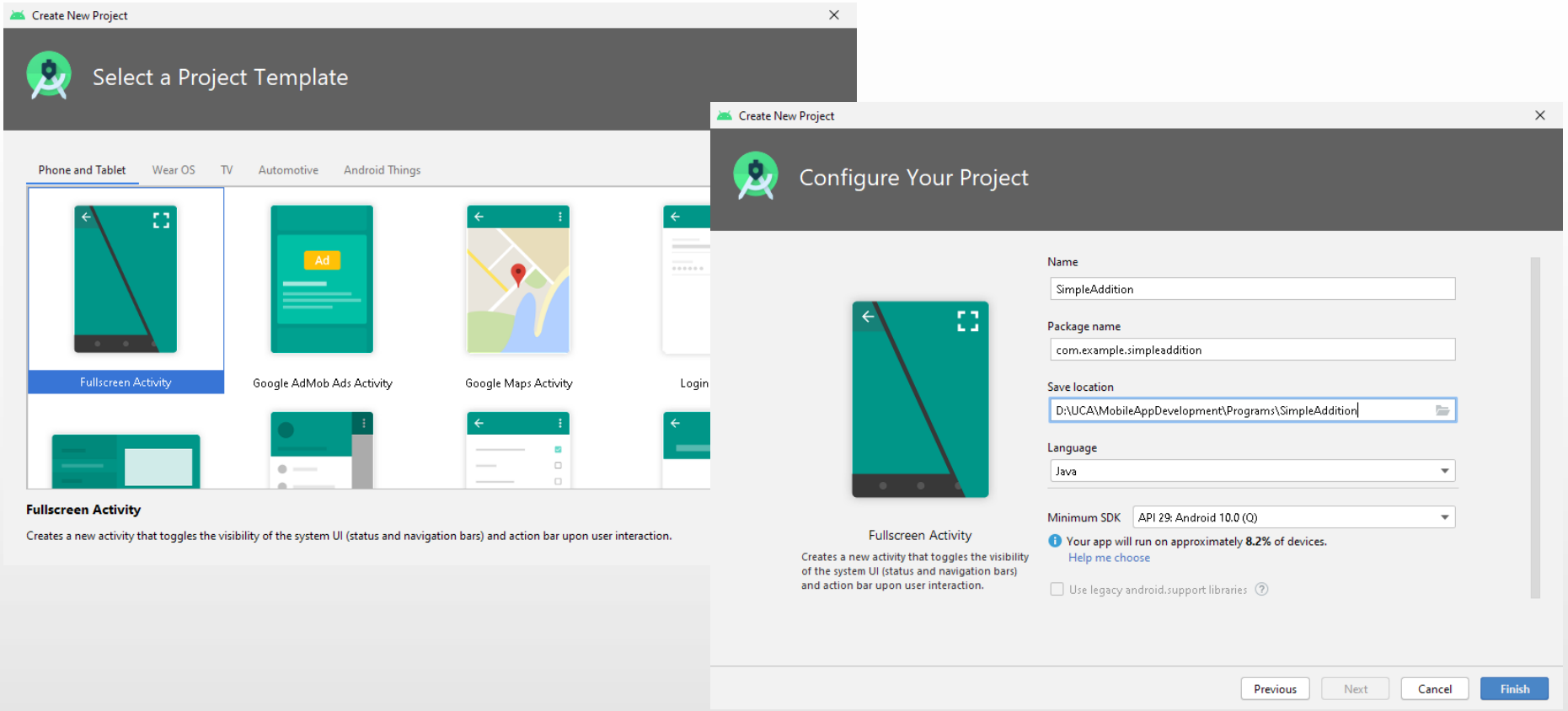
A simplified illustration of the activity lifecycle



https://www.tutorialspoint.com/android/android_acitivities.htm#:~:text=An%20activity%20represents%20a%20single,the%20subclass%20of%20ContextThemeWrapper%20class.&text=The%20Activity%20class%20defines%20the,implement%20all%20the%20callbacks%20methods.

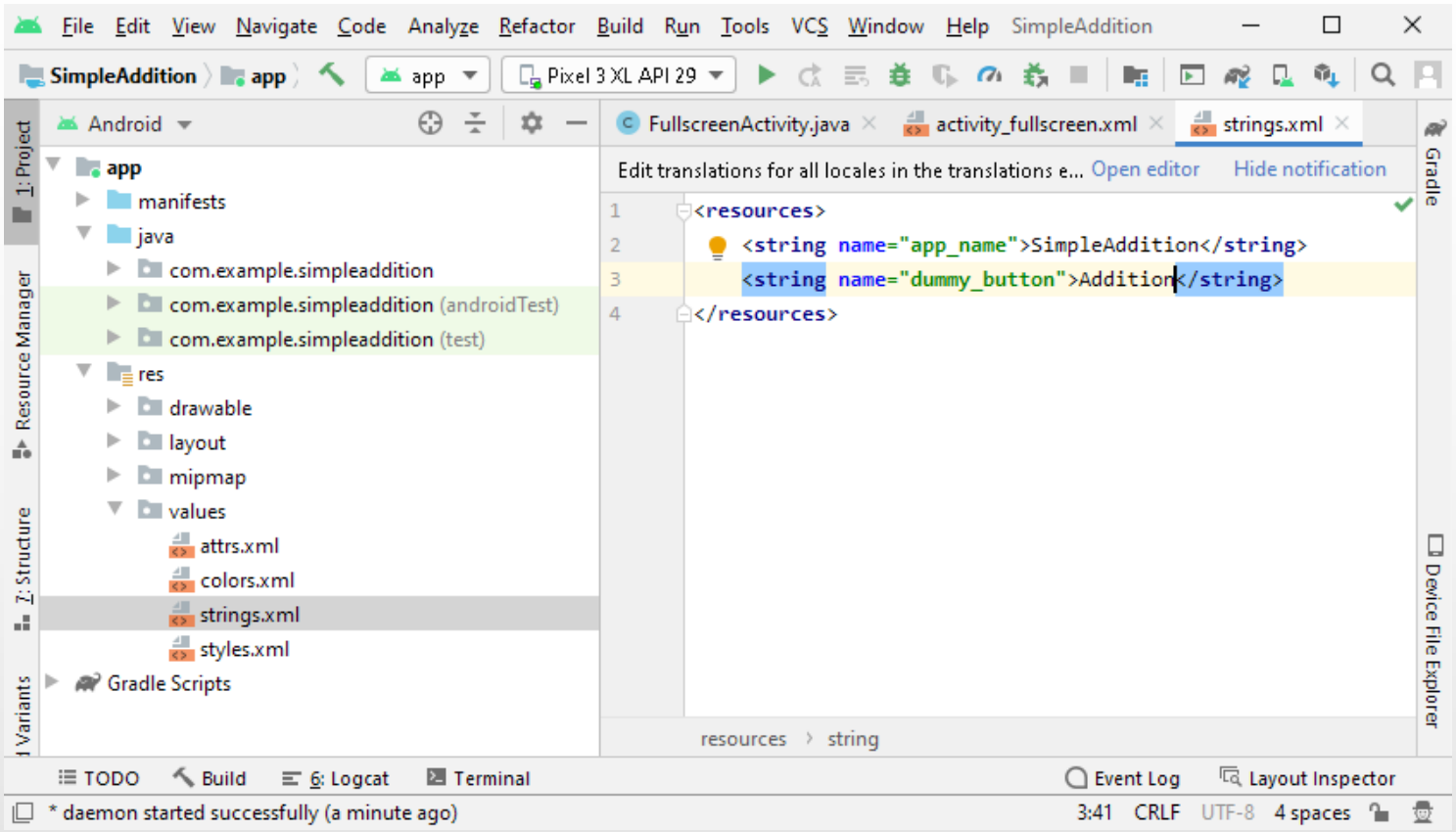
Create a project in Android Studio: 2nd example

- Create a skeletal Android app using Fullscreen activity template. Name the project “SimpleAddition”



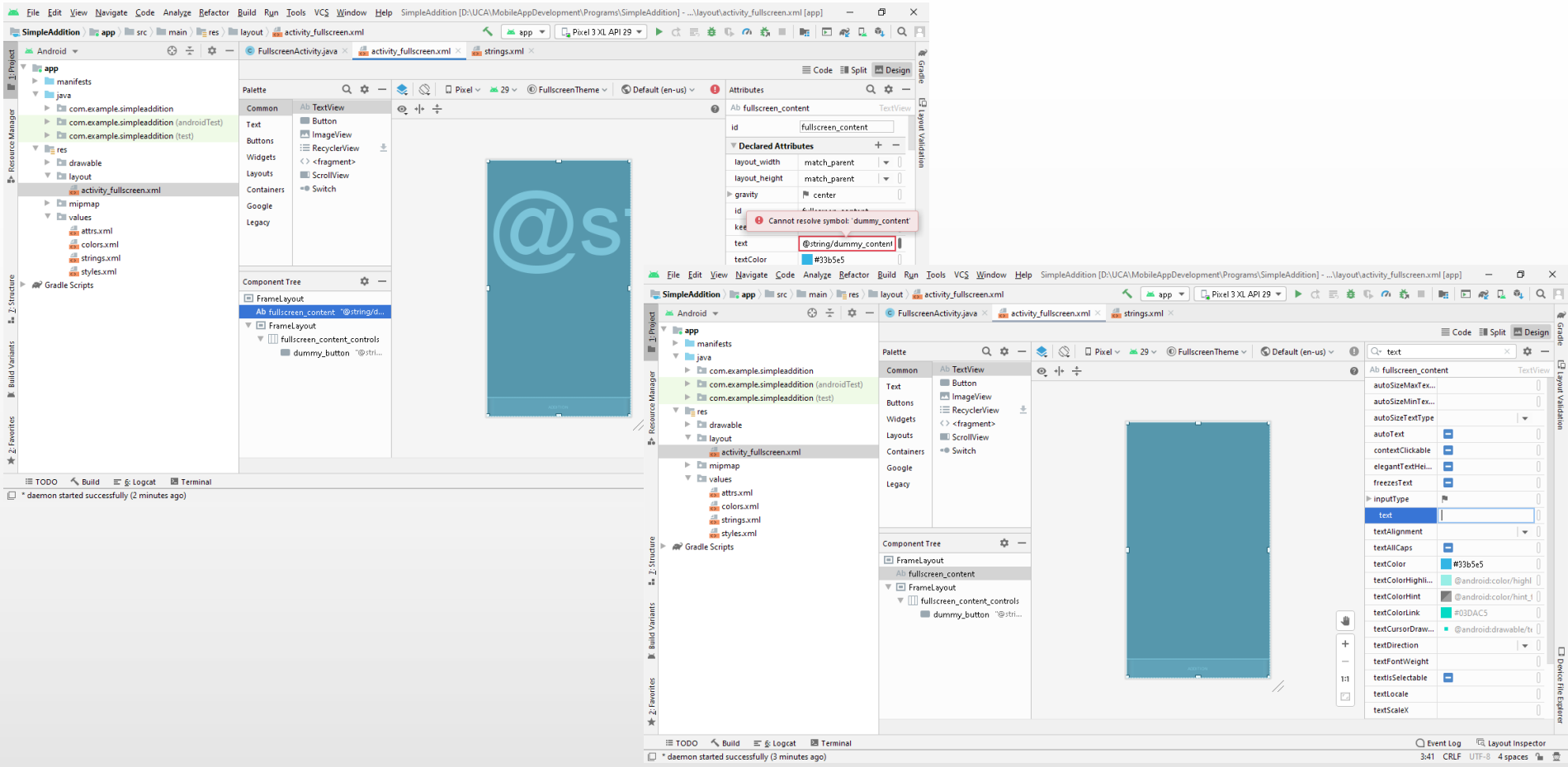
Create a project in Android Studio: 2nd example

- Change the content of the string “dummy_button” and delete “dummy_content”:



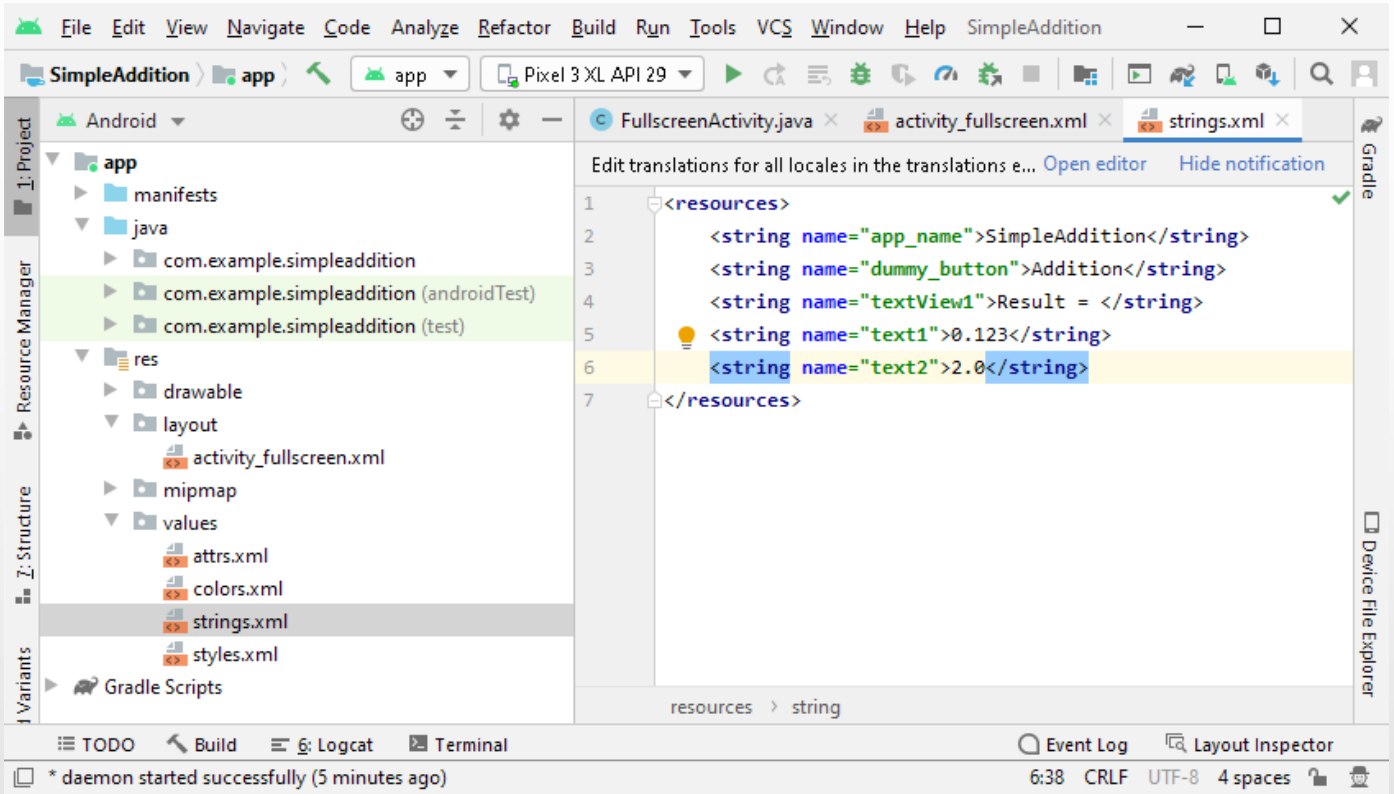
Create a project in Android Studio: 2nd example

- Delete text for the attribute *text* of *fullscreen_content*:



Create a project in Android Studio: 2nd example

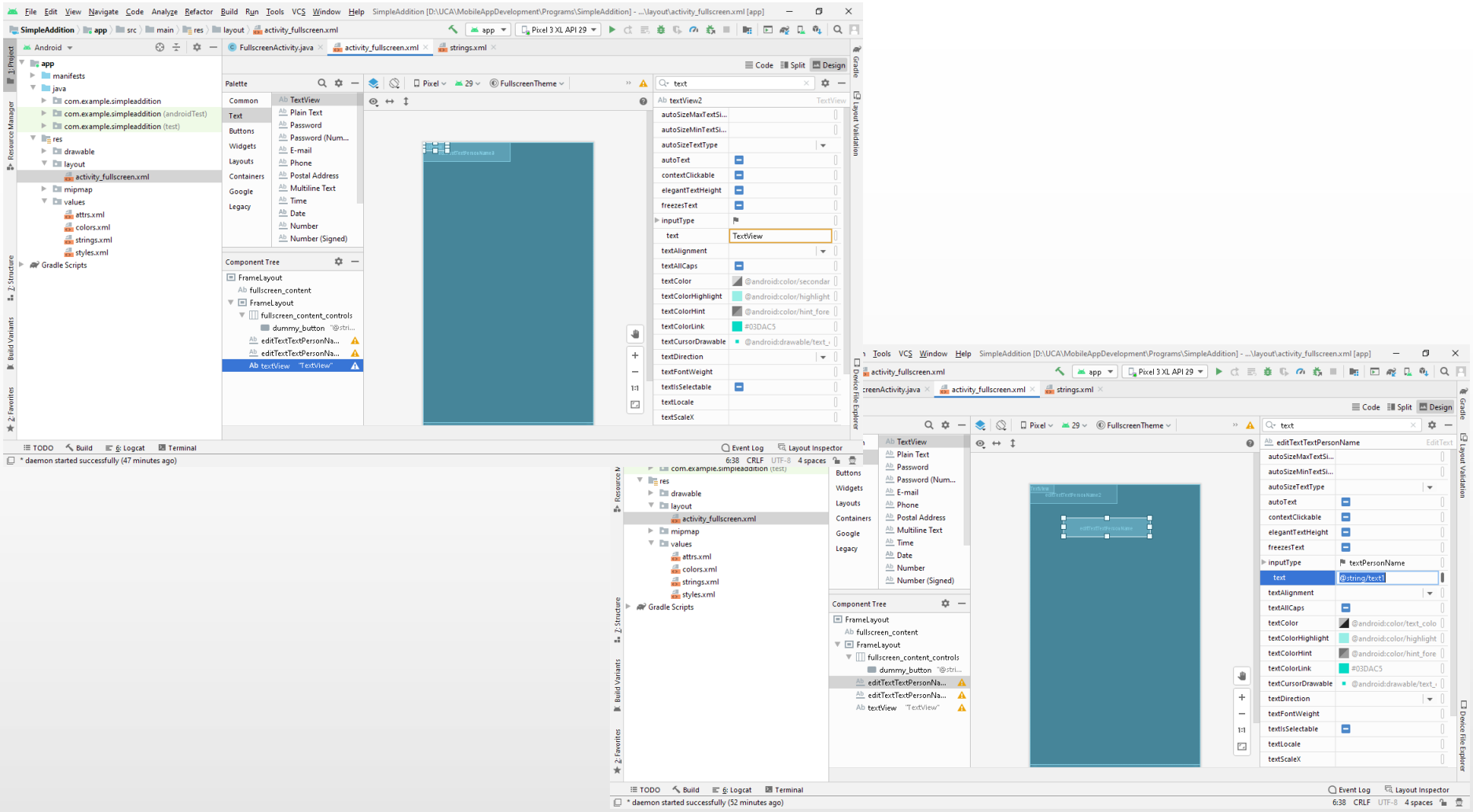
- Double-click *strings.xml* in *res/values* branch. Add strings variables *textView1* (“Result = ”), *text1* (“0.123”), and *text2* (“2.0”):



Create a project in Android Studio: 2nd example

- Drag and drop the "*Plain Text*" element from the "*Text*" palette to FullscreenActivity graphical layout **twice**. Drag and drop the "*TextView*" element from the "*Text*" palette to FullscreenActivity graphical layout.
- Select *editTextTextPersonName* element in the *Component Tree* and change the properties *layout_marginStart* and *layout_marginTop* to 80dp. Text size is 32dp. Change the property *text* to *@string/text1*
- Select *editTextTextPersonName2* element in the *Component Tree* and change the properties *layout_marginStart* and *layout_marginTop* to 80dp and 180dp, respectively. Text size is 32dp. Change the property *text* to *@string/text2*
- Select *textView* element in the *Component Tree* and change the properties *layout_marginStart* and *layout_marginTop* to 80dp and 280dp, respectively. Text size is 32dp. Change the property *text* to *@string/textView1*

Create a project in Android Studio: 2nd example



Create a project in Android Studio: 2nd example

- The *xml* code is as follows:

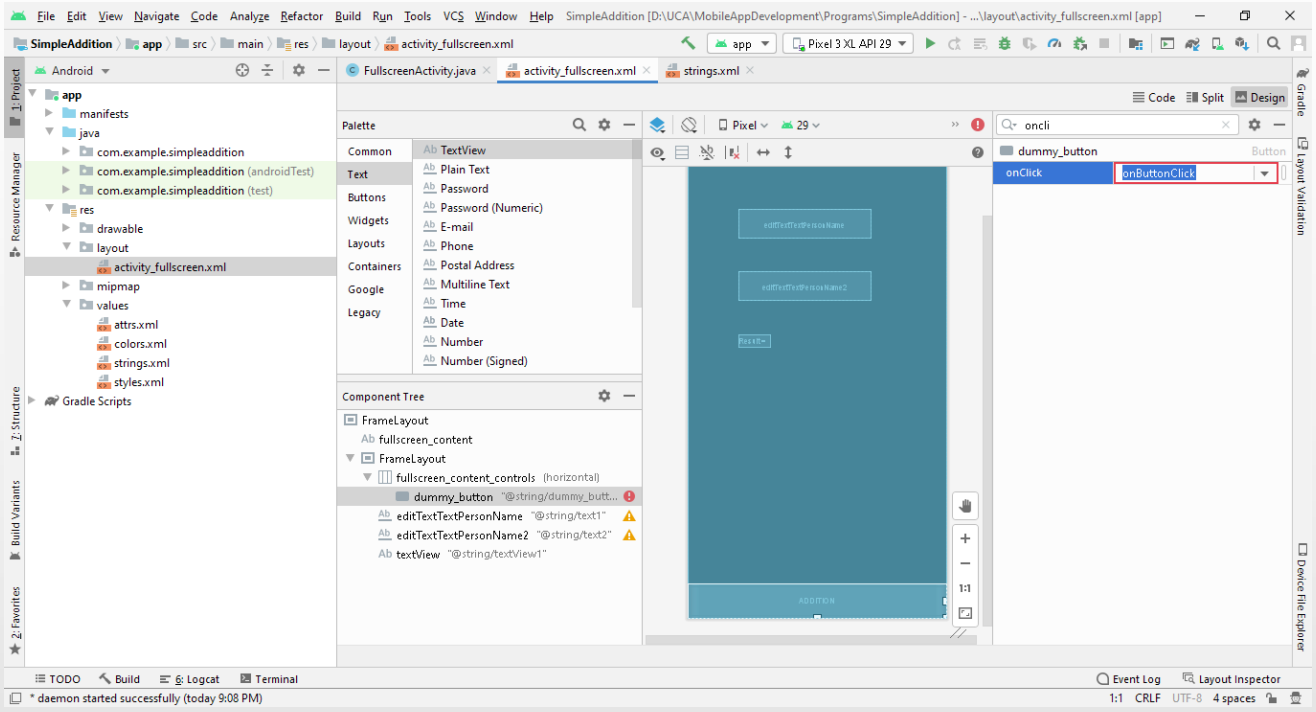
```
<EditText
    android:id="@+id/editTextTextPersonName"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginTop="80dp"
    android:ems="10"
    android:inputType="textPersonName"
    android:text="@string/text1"
    android:textSize="32dp" />

<EditText
    android:id="@+id/editTextTextPersonName2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginTop="180dp"
    android:ems="10"
    android:inputType="textPersonName"
    android:text="@string/text2"
    android:textSize="32dp" />

<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="80dp"
    android:layout_marginTop="280dp"
    android:text="@string/textView1"
    android:textSize="32dp" />
```

Create a project in Android Studio: 2nd example

- On the preview screen or in the component tree, select the *dummy_button*
- In the Properties pane, type *onButtonClick* in the *onClick* field



Create a project in Android Studio: 2nd example

- Inside the app/java branch of the Project tool window, double-click *FullscreenActivity.java*
- Modify the activity's code as follows:

```
package com.example.simpleaddition;
…
import android.widget.EditText;
import android.widget.TextView;
…
public class FullscreenActivity extends AppCompatActivity {
   EditText editText;
   EditText editText2;
   TextView textView;
…
@Override
protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_fullscreen);
      editText = (EditText) findViewById(R.id.editTextTextPersonName);
      editText2 = (EditText) findViewById(R.id.editTextTextPersonName2);
      textView = (TextView) findViewById(R.id.textView);
      …
      findViewById(R.id.dummy_button).setOnTouchListener(mDelayHideTouchListener);
}
public void onButtonClick(View view) {
      Double x,y,z;
      x=Double.parseDouble(editText.getText().toString());
      y=Double.parseDouble(editText2.getText().toString());
      z = x + y;
      textView.setText("Result = " + Double.toString(z));
}
…
```

Do you have any questions or comments?

Thank you
for your attention !

In this presentation:

- Some icons were downloaded from flaticon.com and iconscout.com