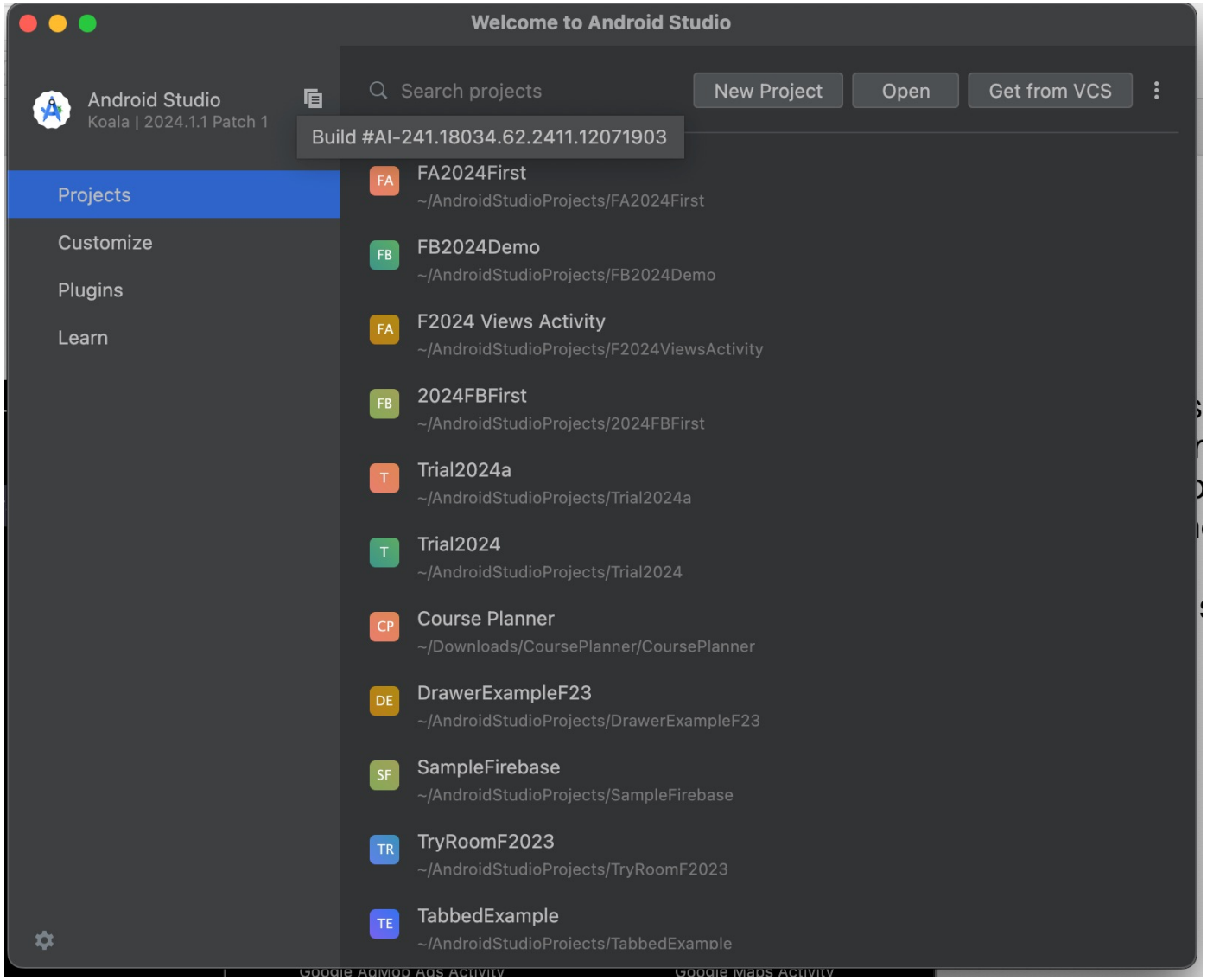
First App

Mobile Computing - Android

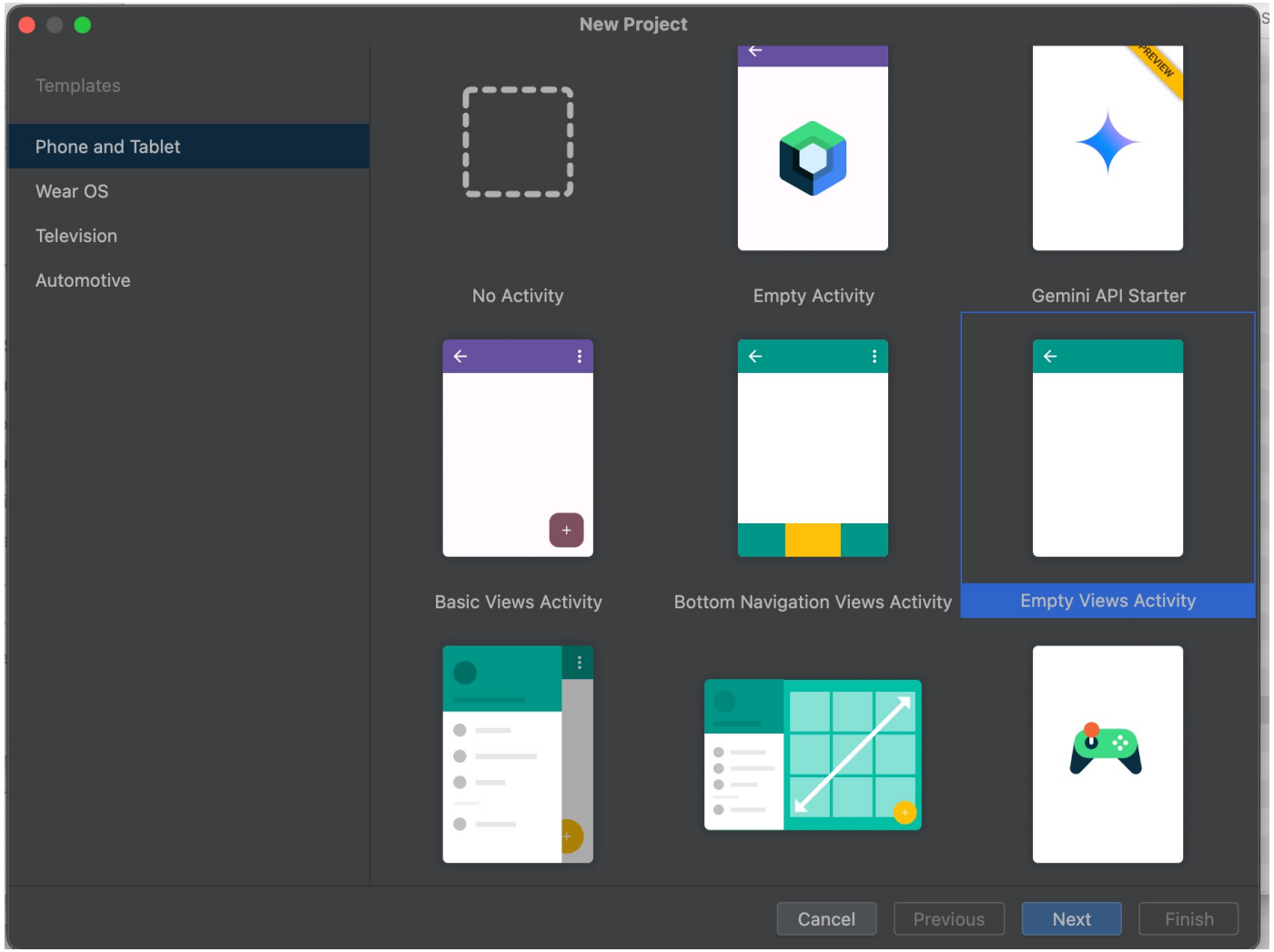
Start



Typical to start from the splash page which gives a list of your projects, most current on top.

Select **New Project**

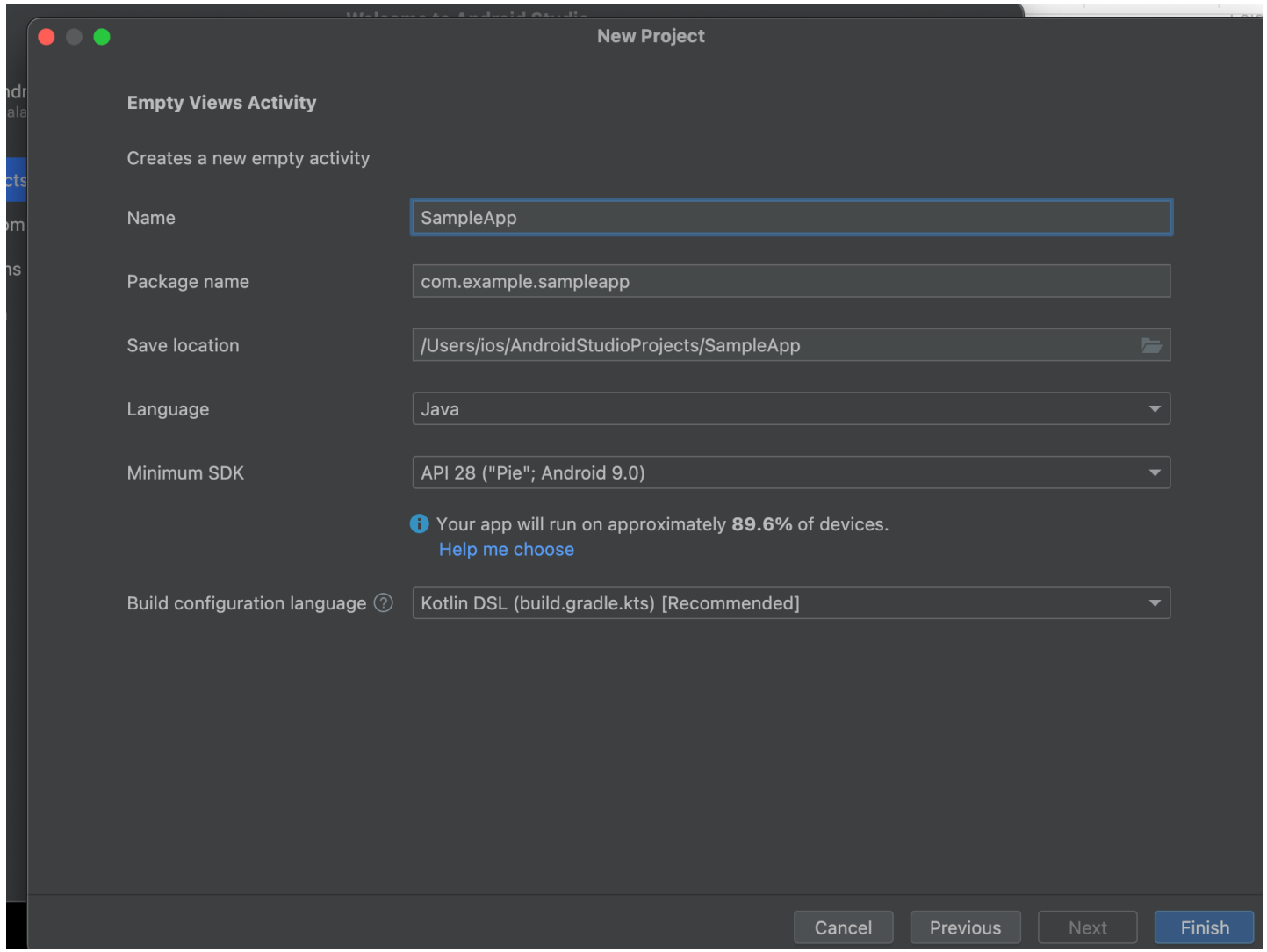
Template choice



Google provides a number of templates that are working projects and implement some basic kind of functionality.

For now, we will select **Empty Views Activity**

Basic Setup



We need to specify

The name of the project.

(Package name will autogenerate)

You can change the save location to someplace convenient.

Language – Choice is either Java or Kotlin

Minimum SDK – Pick one that supports what you need in your project balanced against how many users can run it.

Finish!

Project contents

- **Java** – Your java code
- **Res** – Your XML resources
  - Drawable = images
  - Layout = user interface
  - Menu = menu items
  - Values = strings, ints, Booleans, etc
- **AndroidManifest** = list of application components, minimum SDK, app name, icon, theme, etc.
- **R** = autogenerated file with ids for resources

Layout Files

- Can contain TextViews, EditTexts, Buttons, … the "widgets" that make up the UI and give Android its distinctive appearance.
- Layout files are in XML, but we can get a preview using the design view.
- We can edit using the design view and the attribute inspector or directly change the XML using the code view.

App Goal

- We want an App that will have two editable text areas (EditText) and a button (Button).
- The user enters a name (string) in the nameET
- The user enters an age (number) in the ageET
- The user presses a button and we report the name and age in months in the Logcat.

Process-Layout

- Create a new project using the empty activity.
- Use the design view to drag and drop an EditText (Text - Plain text) into the generated layout.
  - Center the component horizontally and vertically.
  - Move it to the upper left.
  - Give it the id nameET.
  - Give it a hint.
- Use the design view to drag and drop an EditText (Text – Number) into the generated layout.
  - Center the component horizontally and vertically.
  - Move it under the previous widget.
  - Give it the id ageET
  - Give it a hint
- Use the design view to drag and drop a Button into the generated layout.
  - Center the component horizontally and vertically.
  - Move it under the previous widget
  - Give it the id ageInDaysBTN
  - Give it text Age in Days

Process-Code

- In the MainActivity
- Import Log
- Create a new function ageInDaysAction with the following code. (Note: You will have to import classes for use in the method. You can right click on the class name in the code as a short cut that will do the correct addition.)
- Back in the layout, set the onClick property of the button to be ageInDaysAction.
- Run the App.

Method

```
import android.util.Log;
import android.view.View;
import android.widget.EditText;
```

```
public void ageInDaysAction(View v){
    EditText nameET = findViewById(R.id.nameET);
    String name = nameET.getText().toString();


    EditText ageET = findViewById(R.id.ageET);
    int age = Integer.parseInt(ageET.getText().toString());


    Log.d("TAG",
    String.format("Hello, %s, you are %d days old",
          name, age*365));
  }
```

Code Notes

- Used same name for the id and reference to the component.
- Added letters to indicate the kind of component (ET, BTN, …)
- getText() returns a character sequence which we convert to String.
- The click handling routines all have the same signature.
- Used a format string… %s and %d are replaced by the values that come after.
- Log.d() writes to the debug channel. We can use the tag to help filter from the other log messages.

Layout-Code

```
<?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:id="@+id/main"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <TextView
        android:id="@+id/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" />
```

Layout-Code

```
<EditText
 android:id="@+id/nameET"
        android:layout_width="370dp"
        android:layout_height="60dp"
        android:ems="10"
        android:hint="Enter your name here"
        android:inputType="textPersonName"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.301"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.032" />
```

Layout-Code

```
<EditText
        android:id="@+id/ageET"
        android:layout_width="370dp"
        android:layout_height="60dp"
        android:ems="10"
        android:hint="Enter your age in years"
        android:inputType="number"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.301"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.13" />
```

Layout-Code

```
<Button
        android:id="@+id/ageInDaysBTN"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:onClick="ageInDaysAction"
        android:text="Age In Days"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.111"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintVertical_bias="0.227" />

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

Code Notes

- Used hard coded size for the width of the EditTexts. It would be appropriate to make them match the width of the layout. (Use parent.)
- Should remove the “Hello World” TextEdit. (I often repurpose it if I need a display/report widget.)
- Can be more sophisticated with the constraints.
  - Constrain to other widgets instead of just the parent.
  - Use margins.
- As you give a component an ID, you may be asked to refactor. You should do so.

Questions

1. Where in the project would you find java code for your Apps activities.
2. Where in the project would you find XML code specifying the layout.
3. What are three kinds of widgets that you can add to a layout?
4. The method to handle a button click goes where in the project?
5. What does the button click handling method return?
6. What are its argument?
7. How do you get a reference to a widget in your layout?