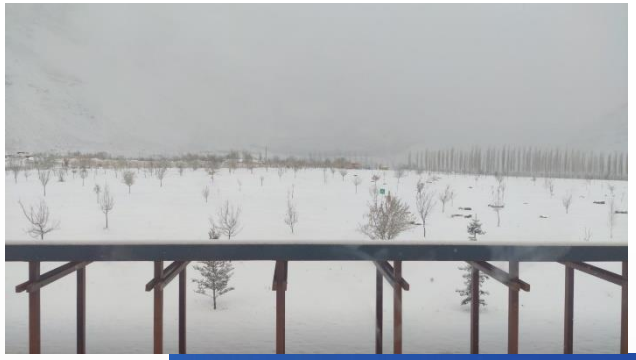
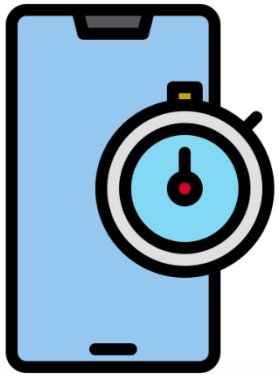
Java Arrays and Strings
Android Login Activity

Dmytro Zubov, PhD
dmytro.zubov@ucentralasia.org

Khorog, Naryn, Nov 23 – Dec 1, 2020



Lessons learnt last time



- What is Augmented Reality (AR)?
- Difference between Virtual Reality, Augmented Reality and Mixed Reality
- Types of AR
- AR applications
- Intro to ARCore
- AR Java Android App in Android Studio
- AR Java Android App in Android Studio: A Local 3D model
- AR Java Android App in Android Studio: A Local 3D model (shape)

What we gonna discuss today?

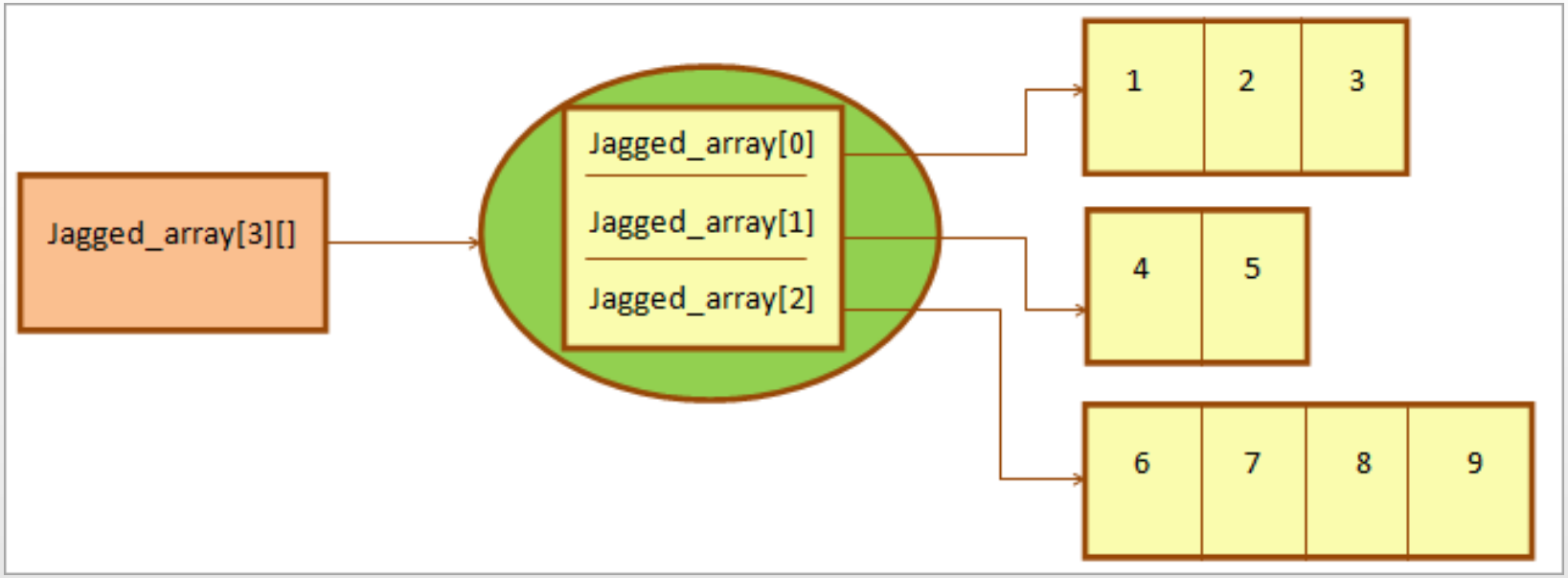
- Java arrays: jagged array & array index operator
- Java strings
- Java Android Authentication app using Firebase

Java arrays: jagged array & array index operator

- An array type is a special reference type that signifies an array, a region of memory that stores values in equal-size and contiguous slots, which are commonly referred to as elements.
  - This type consists of the element type (a primitive type or a user-defined type) and one or more pairs of square brackets that indicate the number of dimensions. A single pair of brackets signifies a one-dimensional array (a vector; e.g., int[] signifies a one-dimensional array with int as the element type), two pairs of brackets signify a two-dimensional array (a table; e.g., double[][] signifies a two-dimensional array with double as the element type), and so on.

Java arrays: jagged array & array index operator (cont.)

- Jagged (ragged) array in Java
  - It is an array of arrays where each element is an array. A special feature of this type of array is that it is a multidimensional array where each element can have different sizes
  - An example of two-dimensional jagged array:



https://www.softwaretestinghelp.com/jagged-array-in-java/

Java arrays: jagged array & array index operator (cont.)

- Jagged array in Java (cont.)
  - We can create a two-dimensional jagged array as follows:

```
int jagged_array[][] = new int[3][];
```

- In the above declaration, a two-dimensional array is declared with three rows

  - Once the array is declared, we can define it as a jagged array as shown below:

```
jagged_array[0] = new int[2];
jagged_array[1] = new int[3];
jagged_array[2] = new int[4];
```

- The first statement above indicates that the first row in the 2D array will have 2 columns. The second row will have 3 columns while the third row will have 4 columns thereby making it a jagged array.

Java arrays: jagged array & array index operator (cont.)

- Jagged array in Java (cont.)
  - Once the array is created, we can initialize it with values. If we don't explicitly initialize this array, then it will take the default values as initial values depending on the data type of the array.
  - Alternatively, we can also initialize an array as follows:

```
int jagged_array[][] = new int[][]{
    new int[] { 1, 2, 3 };
    new int[] { 4, 5, 6, 7 };
    new int[] { 8, 9 };
};
```

  - Another way of initializing a jagged array is by omitting the first `new` operator:

```
int[][] jagged_array ={
    new int[] { 1, 2, 3 };
    new int[] { 4, 5, 6, 7 };
    new int[] { 8, 9 };
};
```

Java arrays: jagged array & array index operator (cont.)

- Jagged array in Java (cont.)
  - We can also omit all the `new` operators altogether and have a declaration and initialization statement as shown below:

```
int[][] jagged_array = {
    { 1, 2, 3 },
    { 4, 5, 6, 7 },
    { 8, 9 } };
```

  - The program below initializes a ragged array by assigning initial values to 1st and 2nd rows; 3rd raw gets the values from the input.

```java
public void onButtonClick(View view) {
    // Declare a 2D jagged array with 3 rows
    Double ragged_array[][] = new Double[3][];
    // Define and initialize jagged array
    ragged_array[0] = new Double[]{1.2,2.3,3.4};
    ragged_array[1] = new Double[]{9.8,10.7,-6.75,3.1415};
    ragged_array[2] = new Double[2];
    ragged_array[2][0]=Double.parseDouble(editText.getText().toString()); ragged_array[2][1]=Double.parseDouble(editText2.getText().toString());
    // Display the jagged array
    String s = ragged_array[0][0] + "  " + ragged_array[0][1] + "  " + ragged_array[0][2] + "\r\n";
    s = s + ragged_array[1][0] + "  " + ragged_array[1][1] + "  " + ragged_array[1][2] +  "  " + ragged_array[1][3] + "\r\n";
    s = s + ragged_array[2][0] + "  " + ragged_array[2][1];
    textView.setText(s);
}
```

Java arrays: jagged array & array index operator (cont.)

- Experimenting with Array Index Operator
  - An array's length is returned by appending "`.length`" to the array variable. For instance, `ages.length` returns the length of the number of elements in the array that `ages` references. Similarly, `matrix.length` returns the number of row elements in two-dimensional array `matrix`, whereas `matrix[0].length` returns the number of column elements assigned to the first row of this array.

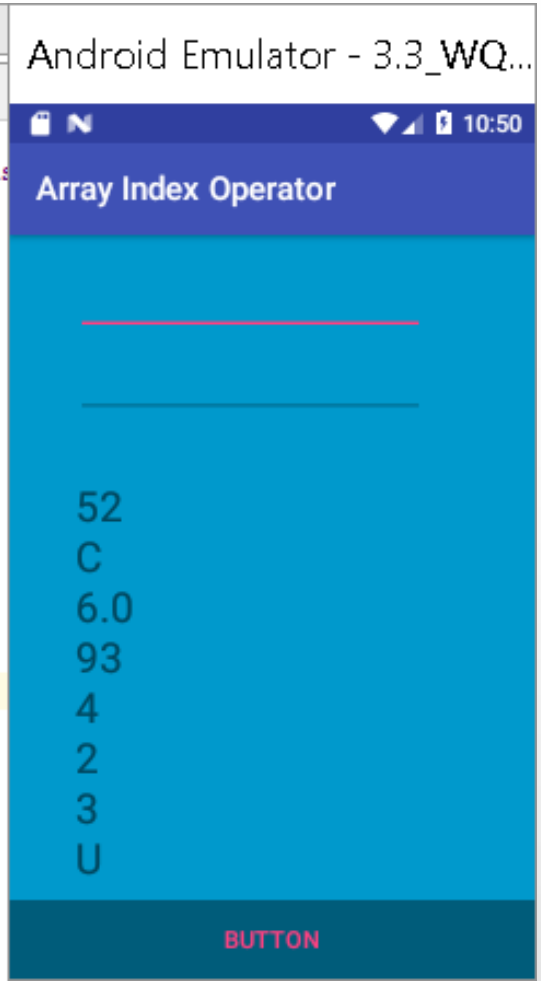
Java arrays: jagged array & array index operator (cont.)

- Experimenting with Array Index Operator (cont.)

y_fullscreen.xml × | strings.xml × | FullscreenActivity.java ×

FullscreenActivity | onButtonClick()

```
        // while interacting with the UI.
        findViewById(R.id.dummy_button).setOnTouchListener(mDelayHideTouchLis
    }

    public void onButtonClick(View view) {
        int[] ages = { 52, 28, 93, 16 };
        char gradeLetters[] = { 'A', 'B', 'C', 'D', 'F' };
        float[][] matrix = { { 1.0F, 2.0F, 3.0F }, { 4.0F, 5.0F, 6.0F }};
        String s1 = Integer.toString(ages[0]), s2="University";
        s1 = s1 + "\r\n" + gradeLetters[2];
        s1 = s1 + "\r\n" + matrix[1][2];
        s1 = s1 + "\r\n" + ages['\u0002'];
        s1 = s1 + "\r\n" + ages.length;
        s1 = s1 + "\r\n" + matrix.length;
        s1 = s1 + "\r\n" + matrix[0].length;
        s1 = s1 + "\r\n" + s2.charAt(0);
        textView.setText(s1);
    }

    @Override
    protected void onPostCreate(Bundle savedInstanceState) {
        super.onPostCreate(savedInstanceState);
```



Java strings

- String is a memory storage with textual content. The String class represents an immutable (unchangeable) sequence of Unicode (16-bit encoding) characters to store characters in any language (English, German, Japanese, and so on). Strings can store a variety of different data structures:
  - the Java String class (java.lang.String) is a utility class for storing string data that will not be modified
  - the Java StringBuilder class (java.lang.StringBuilder) is a utility class for storing string data that will be modified; used when concurrency is not an issue
  - the Java StringBuffer class (java.lang.StringBuffer) is a utility class for storing string data that will be modified; it is used when concurrency is an issue
  - an array of char primitives or Character (java.lang.Character) variables
  - an array of byte primitives or Byte (java.lang.Byte) variables
  - various other data structures and object classes can be used to store string data

https://code.tutsplus.com/tutorials/learn-java-for-android-development-string-basics--mobile-3824
https://developer.android.com/reference/java/lang/String.html

Java strings (cont.)

- The following Java variables represent a string of vowel characters in different ways (as bytes, characters, Unicode representations or sub-strings):

```
String s1, strVowels = "aeiou";
char astrVowels[] = { 'a', 'e', 'i', 'o', 'u' };
byte abyteVowels[] = { 'a', 'e', 'i', 'o', 'u' };
byte abyteVowelsU[] = { '\u0061',
'\u0065','\u0069','\u006F','\u0075' };
String uVowels = new
String("\u0061\u0065\u0069\u006F\u0075");
CharSequence csVowels = (CharSequence) new String("aeiou");
StringBuffer sbVowels = new StringBuffer("a" + "e" + "iou");
StringBuilder sVowelBuilder = new StringBuilder();
sVowelBuilder.append('a');
sVowelBuilder.append("eio");
sVowelBuilder.append('\u0075');
```

Java strings (cont.)

- Simple String Iteration
  - There are numerous ways to do the string iteration, but one simple way is to use a `for()` loop. We can use the String's `length()` method to determine how many characters we've got, and the `charAt()` method to retrieve a specific character by its index, much like we would an array index.

```
String s1, strVowels = "aeiou";
String strAppName =
getResources().getString(R.string.app_name);
s1 = strAppName + "\r\n";
for (int i = 0; i<strVowels.length(); i++)
    s1 = s1 + strVowels.charAt(i) + "\r\n";
textView.setText(s1);
```

What is the output of this program?

Java strings (cont.)

- String Modifications: Converting to Upper and Lowercase
  - A reason we might want to change the case of a string is to normalize the string to make case-insensitive searching or matching easier to implement. An example is as follows:

```
String strUpperCaseVersion = strVowels.toUpperCase();
String strLowerCaseVersion = strVowels.toLowerCase();
s1 = s1 + strUpperCaseVersion + "\r\n";
s1 = s1 + strLowerCaseVersion + "\r\n";
```

Java strings (cont.)

- String Modifications: Splitting
  - Sometimes we want to quickly parse a string into substrings. We might do this to extract the individual words from a sentence, or a delimited list of tags, etc. We can use simple regular expressions with the split() function for this purpose.

```
String s1="", someWords = "Red Orange Yellow Green Blue Indigo";
String aColors[] = someWords.split(" ");
for (int i = 0; i<aColors.length; i++)
     s1 = s1 + aColors[i] + "\r\n";
textView.setText(s1);
```

What is the output of this program?

Java strings (cont.)

- Simple String Matching
  - We can check if two strings match using the String class's methods `compareTo()` and/or `compareToIgnoreCase()`. This method will return 0 if, and only if, the two strings are identical:

```
String s1= "", strVowels = "aeiou";
if(strVowels.compareTo("AEIOU") == 0)
     s1 = s1 + "Strings match!" + "\r\n";
else
     s1 = s1 + "Strings don't match!" + "\r\n";
```

What is the output of this program?

Java strings (cont.)

- String Searching
  - There are many other ways to perform string matching and searching, allowing us to build whatever search methods we desire. We can also find specific characters or substrings using the `indexOf()` and `lastIndexOf()` methods, check if a string begins or ends with a substring using the `startsWith()` and `endsWith()` methods. Finally, the `matches()` method supports regular expression matching.
  - Here we use the `contains()` method to determine if a specific substring exists:

```
if(strVowels.contains("IOU")==true)
{
        // String contains IOU sub-string!
}
```

Java strings (cont.)

- Strings and Other Data Types
  - The String object is fundamental to Java. Every class, due to being derived from the root class called Object (`java.lang.Object`), has a `toString()` method to create a useful string representation of their value
  - Classes that don't have a string representation usually return some identifier or debug information as to the type of class. Those that do have a string representation, such as a number string from an Integer object, return the textual representation of the encapsulated number.
  - When concatenated with a String, such as with the plus (+), `toString()` method results are used by default:

```
Integer iNum =135;
String s = "The number one-three-five = " + iNum.toString();
textView.setText(s);
```

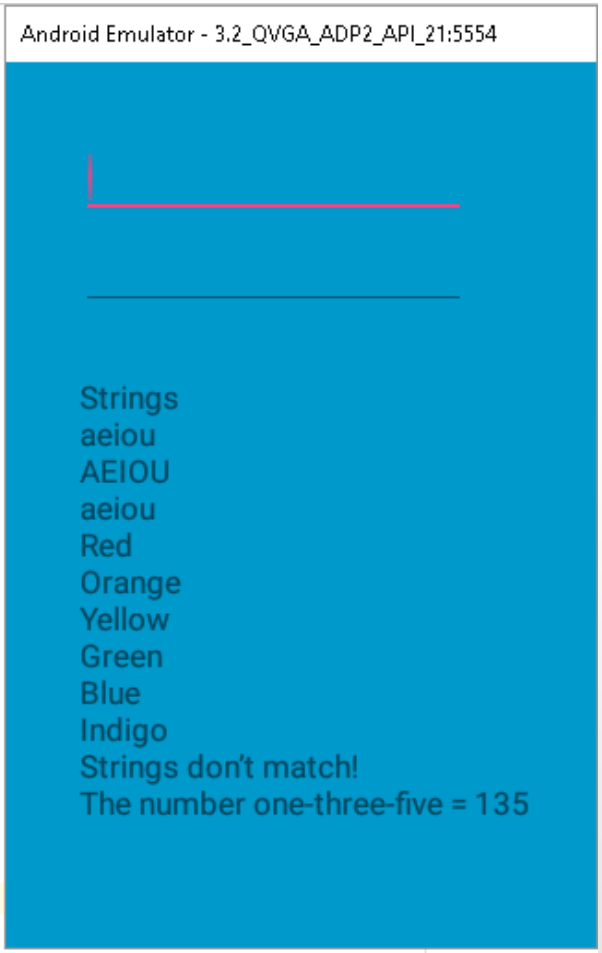
Will this example be working WITHOUT .toString?

What is the output of this program?

Java strings (cont.)

- An example

```
String strVowels10 = new String(new byte[]{ '\u0061',
        '\u0065','\u0069','\u006F','\u0075' });

// Simple String Iteration
String strAppName = "Strings";
s1 = strAppName + "\r\n";
for (int i = 0; i<strVowels.length(); i++)
    s1 = s1 + strVowels.charAt(i);
s1 = s1 + "\r\n";

// String Modifications: Converting to Upper and Lowercase
String strUpperCaseVersion = strVowels.toUpperCase();
String strLowerCaseVersion = strVowels.toLowerCase();
s1 = s1 + strUpperCaseVersion + "\r\n";
s1 = s1 + strLowerCaseVersion + "\r\n";

// String Modifications: Splitting
String someWords = "Red Orange Yellow Green Blue Indigo";
String aColors[] = someWords.split(" ");
for (int i = 0; i<aColors.length; i++)
    s1 = s1 + aColors[i] + "\r\n";

// Simple String Matching
if(strVowels.compareTo("AEIOU") == 0)
    s1 = s1 + "Strings match!" + "\r\n";
else
    s1 = s1 + "Strings don't match!" + "\r\n";

// Strings and Other Data Types
Integer iNum =135;
s1 = s1 + "The number one-three-five = " + iNum.toString();
```



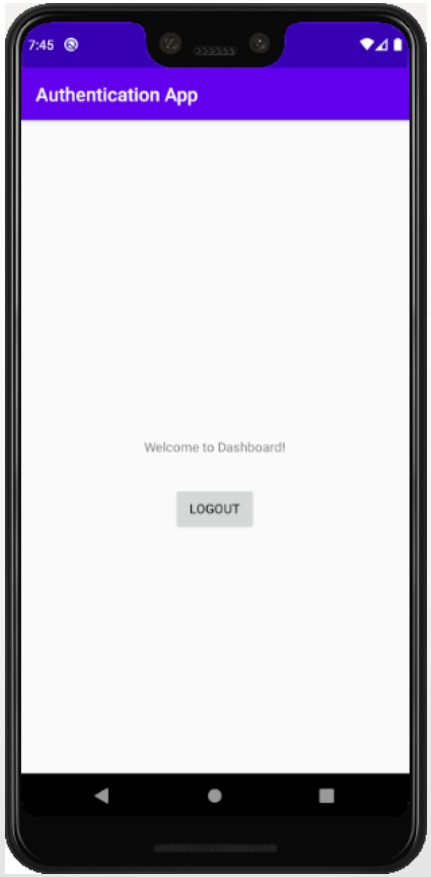
Introducing Firebase Authentication

https://www.youtube.com/watch?v=8sGY55yxicA&list=PLl-K7zZEsYLmOF_07IayrTntevxtbUxDL

Java Android Authentication App Using Firebase

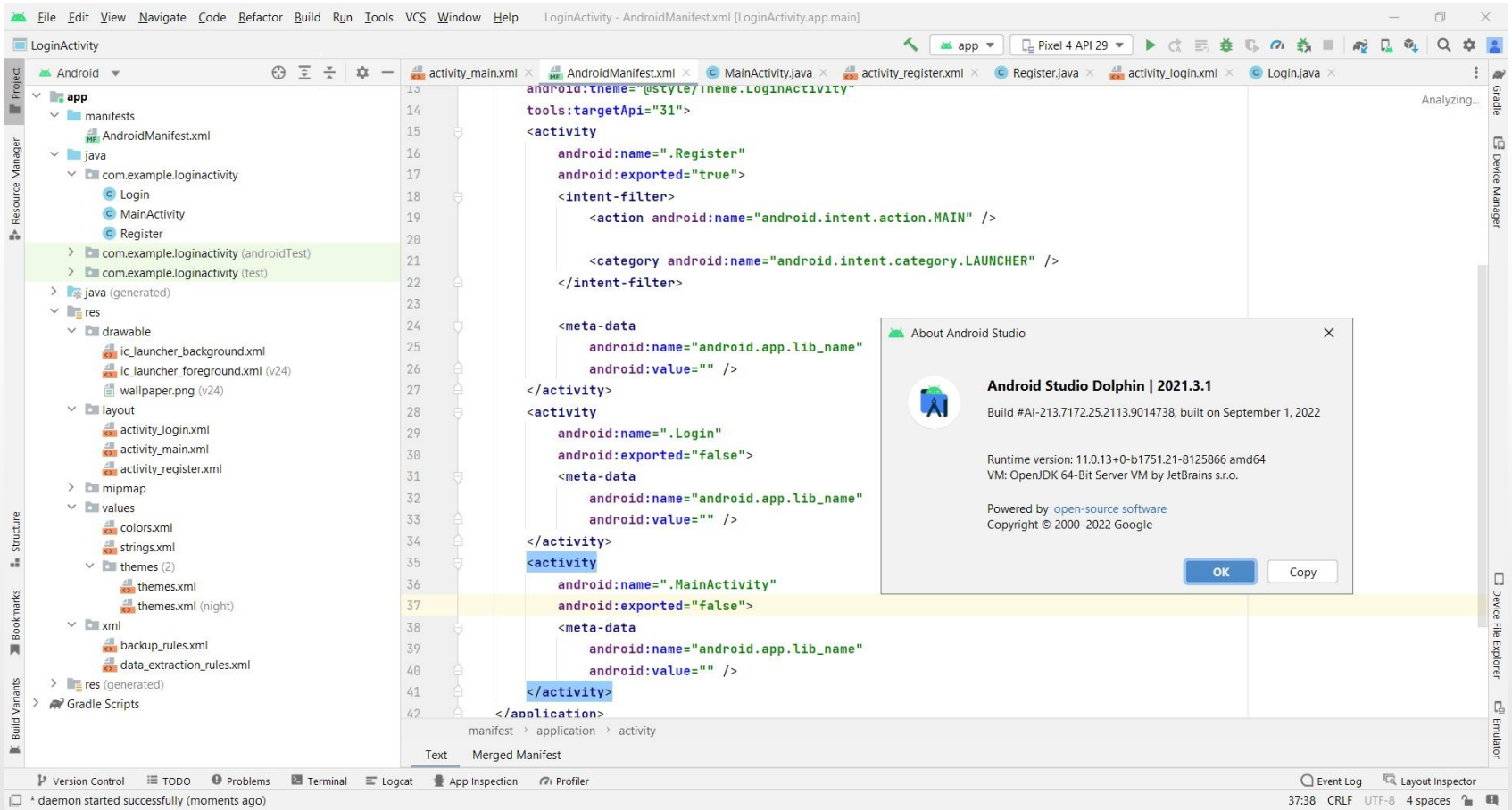
https://smallacademy.co/blog/android/login-register-using-firebase/

- In this exercise, we develop a simple Java Android Authentication app using Google Firebase



Java Android Authentication App Using Firebase

- In this exercise, we use Android Studio Dolphin 2021.3.1.



Java Android Authentication App Using Firebase (cont.)

- In this exercise, we develop a simple Java Android Authentication app using Google Firebase

console.firebase.google.com/project/[redacted]/authentication/users

Firebase

Project Overview

Develop

Authentication

Cloud Firestore

Realtime Database

Storage

Hosting

Functions

Machine Learning

Quality

Crashlytics Performance Test La

Extensions

Spark
Free $0/month

Upgrade

AutenticationApp

Go to docs

Authentication

Users | Sign-in method | Templates | Usage

Prototype and test end-to-end with the Local Emulator Suite, now with Firebase Authentication

Get started

Search by email address, phone number, or user UID

Add user

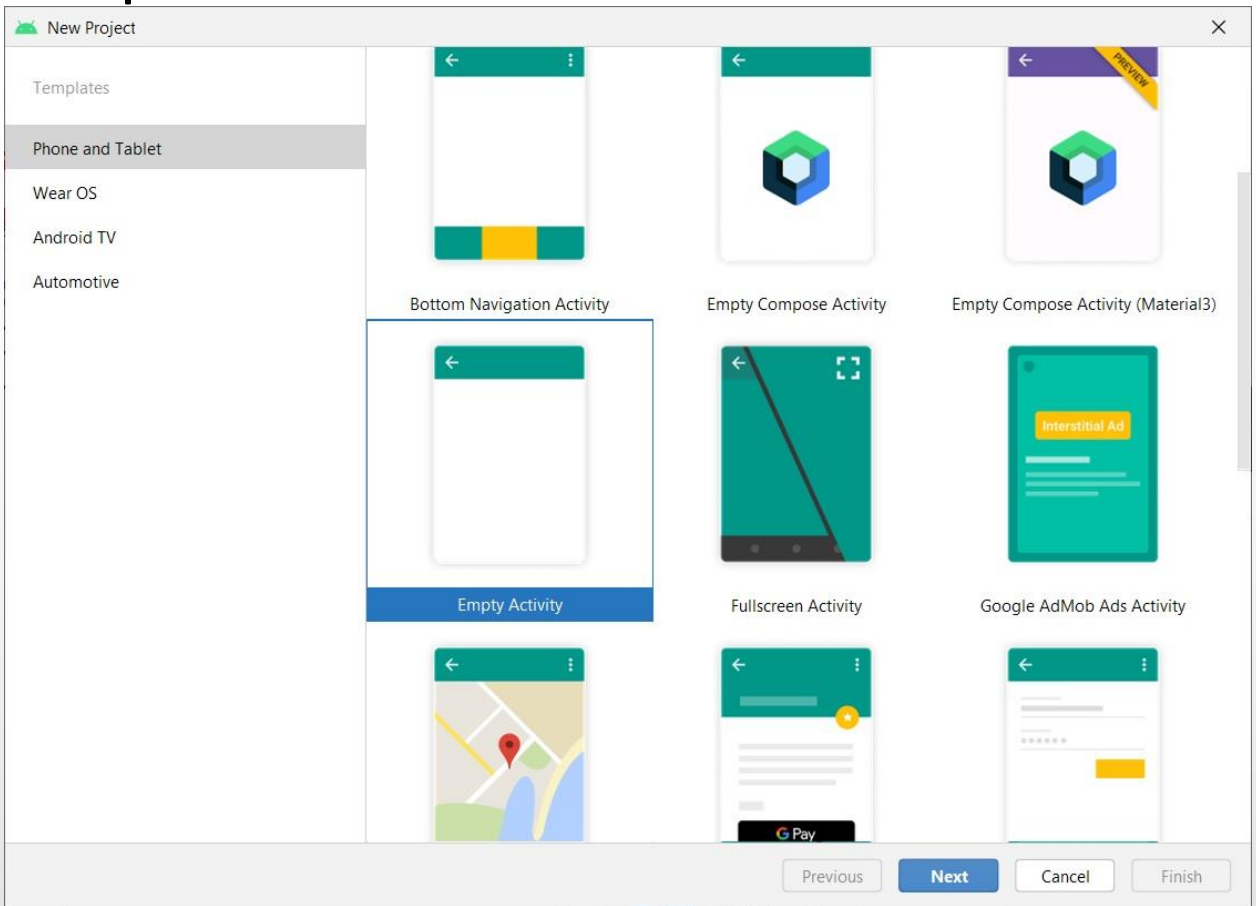
Identifier	Providers	Created	Signed In	User UID
[redacted]@yahoo.com		Nov 30,...	Dec 1, ...	[redacted]

Rows per page: 50 1 – 1 of 1

UI may look different...

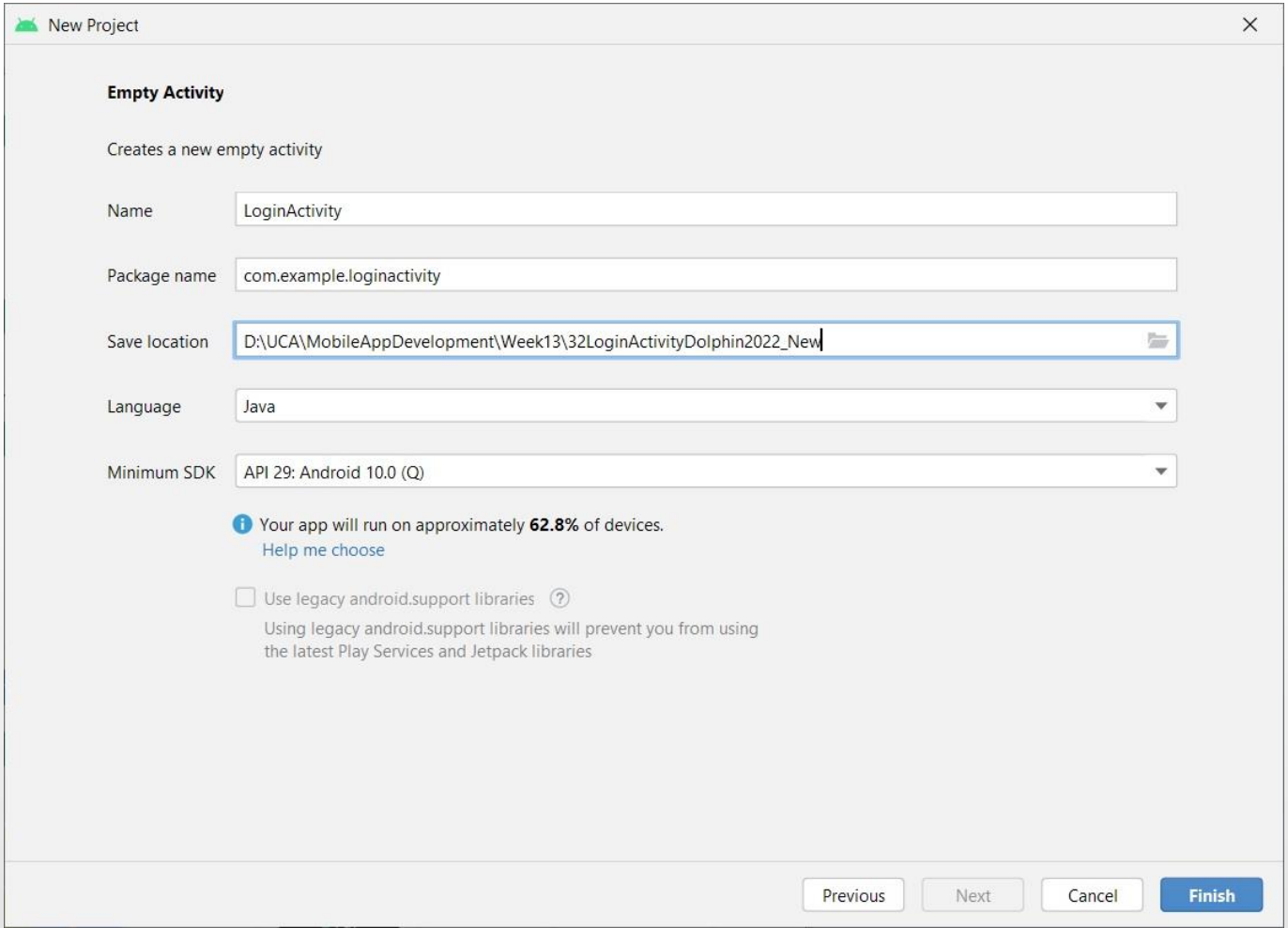
Java Android Authentication App Using Firebase (cont.)

- Create a new Android app project using the Empty Activity template:



Java Android Authentication App Using Firebase (cont.)

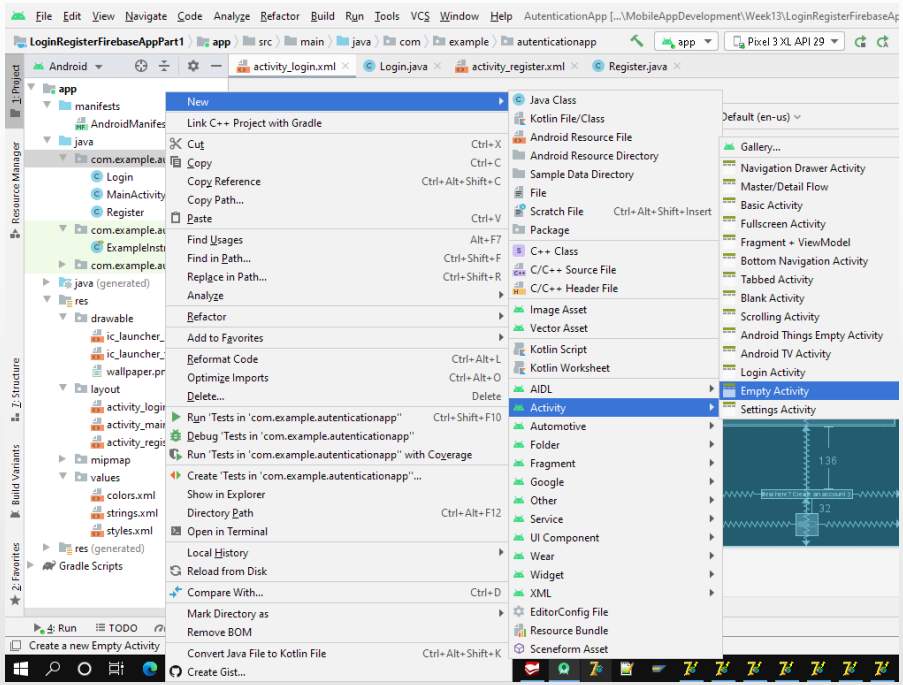
- Configure the project

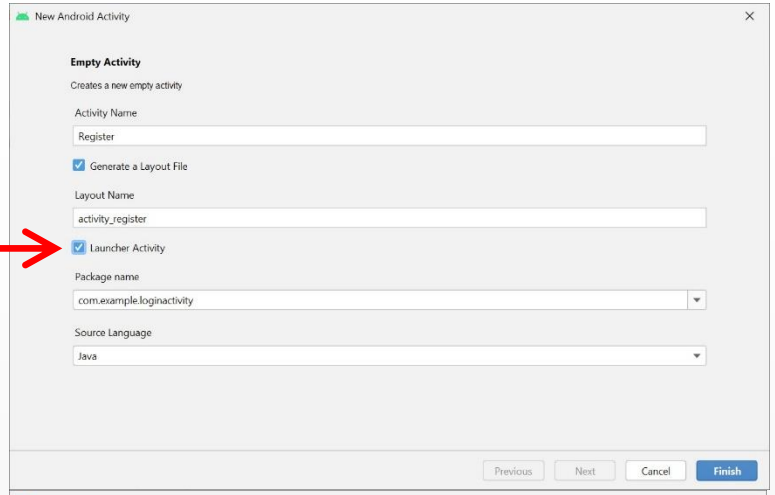


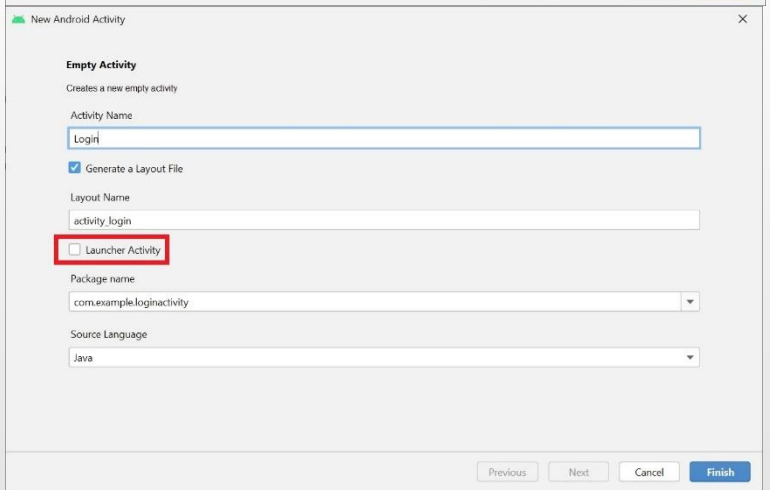
Java Android Authentication App Using Firebase (cont.)

- Add two new Empty activities “Login” and “Register” to our project:

Register activity is a Launcher Activity (this activity will be shown first to the end-user)



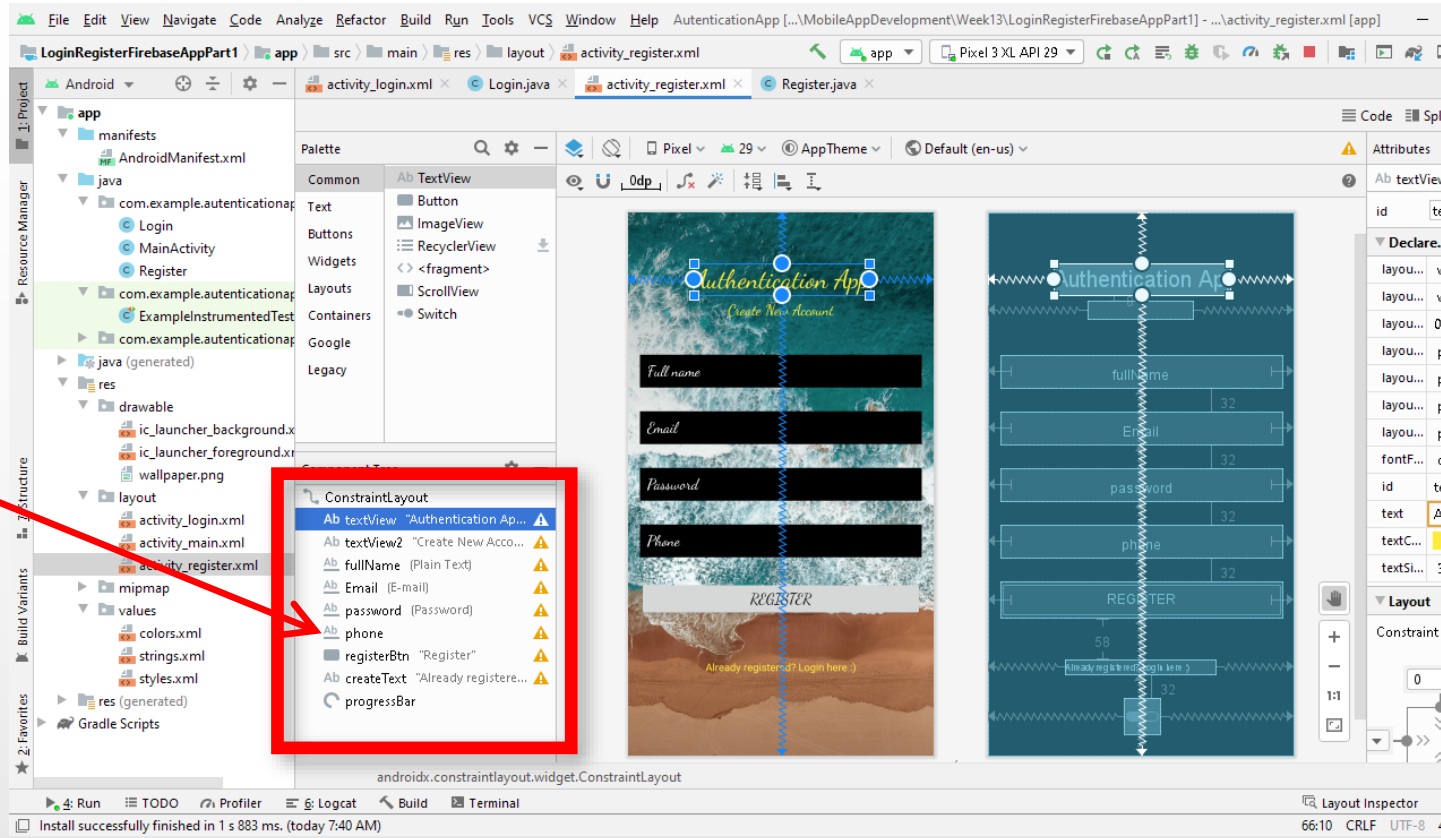




Java Android Authentication App Using Firebase (cont.)

- Design the UI of the Register activity as follows:

Drag and drop these elements from the Palette



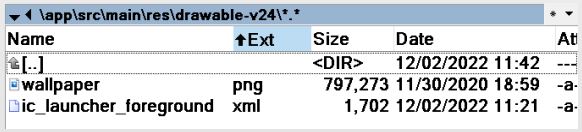
Java Android Authentication App Using Firebase (cont.)

- The XML description of the Register activity without components may look as follows:

```
<?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"
    android:background="@drawable/wallpaper"
    tools:context=".Register">

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

Here, we add the background

Here, we copy the file "wallpaper.png" into the folder "drawable-v24"



Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the title of the app in the Register activity may look as follows:

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="cursive"
    android:text="Authentication App"
    android:textColor="#FFEB3B"
    android:textSize="35sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.1" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the description of the Register activity may look as follows:

```
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:fontFamily="cursive"
    android:text="Create New Account"
    android:textColor="#FFEB3B"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the full name of the end-user in the Register activity may look as follows:

```
<EditText
    android:id="@+id/fullName"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:background="@android:color/background_dark"
    android:ems="10"
    android:fontFamily="cursive"
    android:hint="Full name"
    android:inputType="textPersonName"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2"
    app:layout_constraintVertical_bias="0.031" />
```

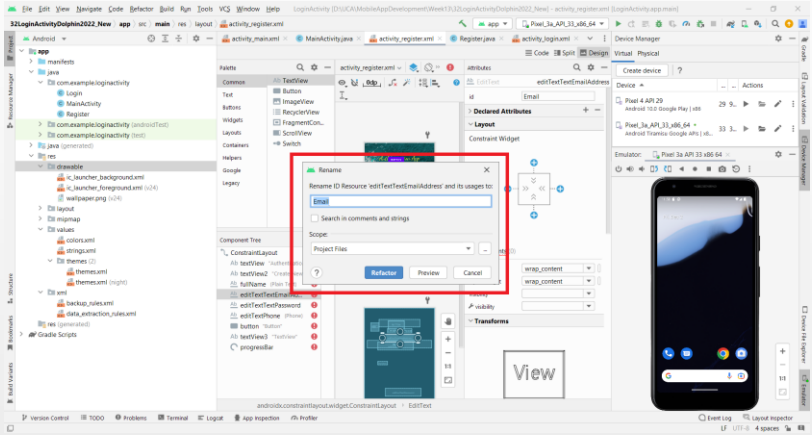
Perhaps we need to change the ID manually before copying this text ☹

Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the email of the end-user in the Register activity may look as follows:

```
<EditText
    android:id="@+id/Email"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:inputType="textEmailAddress"
    android:hint="Email"
    android:background="@android:color/background_dark"
    android:ems="10"
    android:fontFamily="cursive"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/fullName" />
```

- Perhaps we need to change the ID manually before copying this text ☹



Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the password of the end-user in the Register activity may look as follows:

```
<EditText
    android:id="@+id/password"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:ems="10"
    android:inputType="textPassword"
    android:hint="Password"
    android:background="@android:color/background_dark"
    android:fontFamily="cursive"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Email" />
```

Perhaps we need to change the ID manually before copying this text ☹

Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the phone of the end-user in the Register activity may look as follows:

```
<EditText
    android:id="@+id/phone"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:ems="10"
    android:inputType="textPhonetic"
    android:hint="Phone"
    android:background="@android:color/background_dark"
    android:fontFamily="cursive"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/password" />
```

Perhaps we need to change the ID manually before copying this text ☹

Java Android Authentication App Using Firebase (cont.)

- The XML description of the Button element with the onClickRegister callback in the Register activity may look as follows:

```
<Button
    android:id="@+id/registerBtn"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:fontFamily="cursive"
    android:onClick="onClickRegister"
    android:text="Register"
    android:textColor="#000000"
    android:textColorHint="#000000"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/phone" />
```

Perhaps we need to change the ID manually before copying this text ☹

Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the onClickLoginFromRegister callback (to start another activity "Login") in the Register activity may look as follows:

```
<TextView
    android:id="@+id/createText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="58dp"
    android:onClick="onClickLoginFromRegister"
    android:text="Already registered? Login here :)"
    android:textColor="#FFEB3B"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/registerBtn" />
```

Java Android Authentication App Using Firebase (cont.)

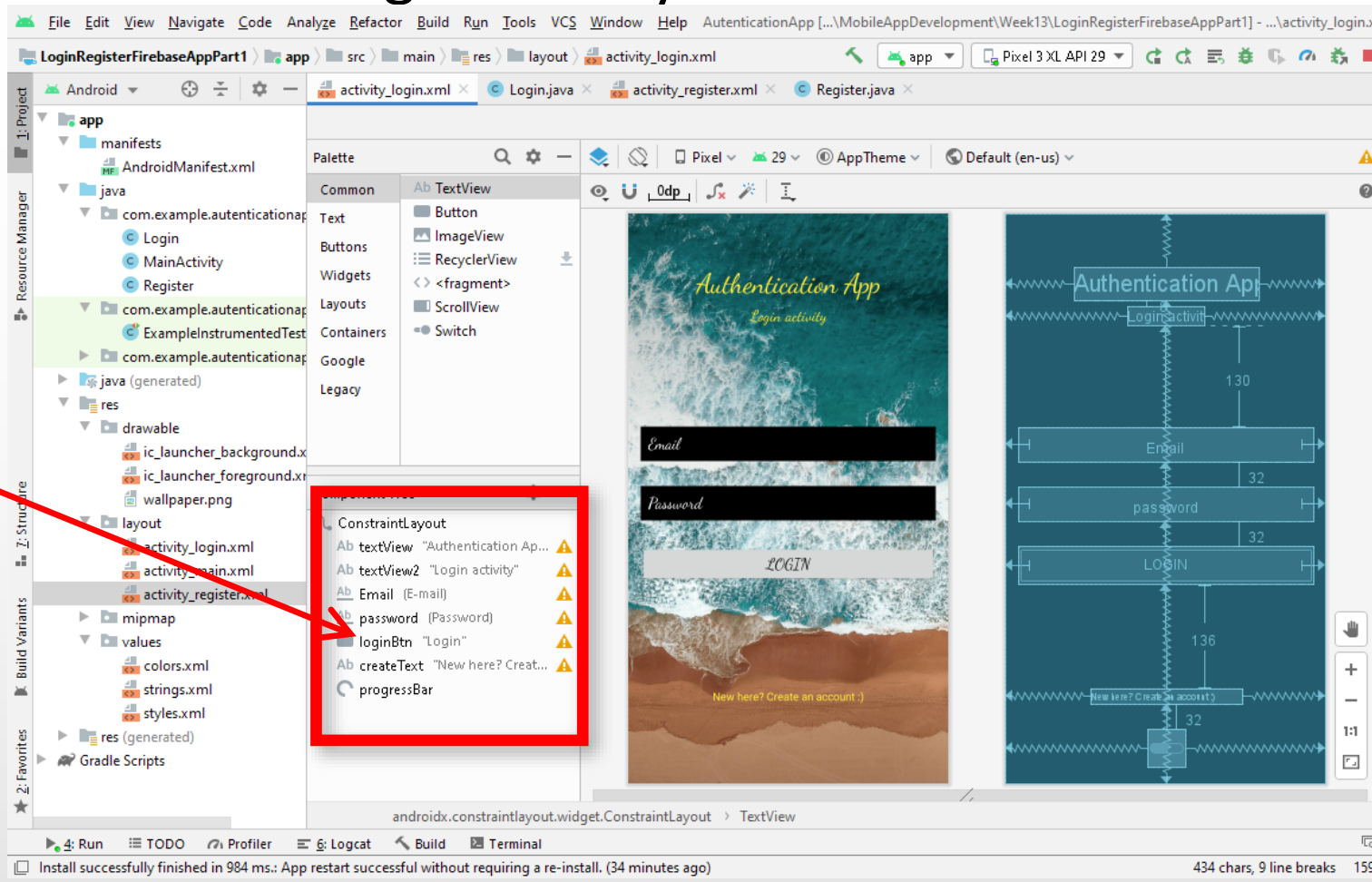
- The XML description of the ProgressBar element (this element shows the progress of the registration process; initially, this element is invisible) in the Register activity may look as follows:

```
<ProgressBar
    android:id="@+id/progressBar"
    style="?android:attr/progressBarStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:visibility="invisible"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/createText" />
```

Java Android Authentication App Using Firebase (cont.)

- Design the UI of the Login activity as follows:

Drag and drop these elements from the Palette



Java Android Authentication App Using Firebase (cont.)

- The XML description of the Login activity without components may look as follows:

```
<?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"
    android:background="@drawable/wallpaper"
    tools:context=".Login">

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

Here, we add the background

Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the title of the app in the Login activity may look as follows:

```
<TextView
    android:id="@+id/textView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:fontFamily="cursive"
    android:text="Authentication App"
    android:textColor="#FFEB3B"
    android:textSize="35sp"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.1" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the description of the Login activity may look as follows:

```
<TextView
    android:id="@+id/textView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:fontFamily="cursive"
    android:text="Login activity"
    android:textColor="#FFEB3B"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the email of the end-user in the Login activity may look as follows:

```
<EditText
    android:id="@+id/Email1"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="130dp"
    android:layout_marginEnd="16dp"
    android:background="@android:color/background_dark"
    android:ems="10"
    android:fontFamily="cursive"
    android:hint="Email"
    android:inputType="textEmailAddress"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the EditText element with the password of the end-user in the Login activity may look as follows:

```
<EditText
    android:id="@+id/Password"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:ems="10"
    android:inputType="textPassword"
    android:hint="Password"
    android:background="@android:color/background_dark"
    android:fontFamily="cursive"
    android:padding="10dp"
    android:textColor="#FFFFFF"
    android:textColorHint="@android:color/background_light"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Email1" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the Button element with the onClickLogin callback in the Login activity may look as follows:

```
<Button
    android:id="@+id/loginBtn"
    android:layout_width="0dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="16dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="16dp"
    android:fontFamily="cursive"
    android:onClick="onClickLogin"
    android:text="Login"
    android:textColor="#000000"
    android:textColorHint="#000000"
    android:textSize="20sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Password" />
```

Java Android Authentication App Using Firebase (cont.)

- The XML description of the textView element with the onClickRegisterFromLogin callback (to start another activity "Register") in the Login activity may look as follows:

```
<TextView
    android:id="@+id/createText"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="136dp"
    android:onClick="onClickRegisterFromLogin"
    android:text="New here? Create an account :)"
    android:textColor="#FFEB3B"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/loginBtn" />
```

Java Android Authentication App Using Firebase (cont.)

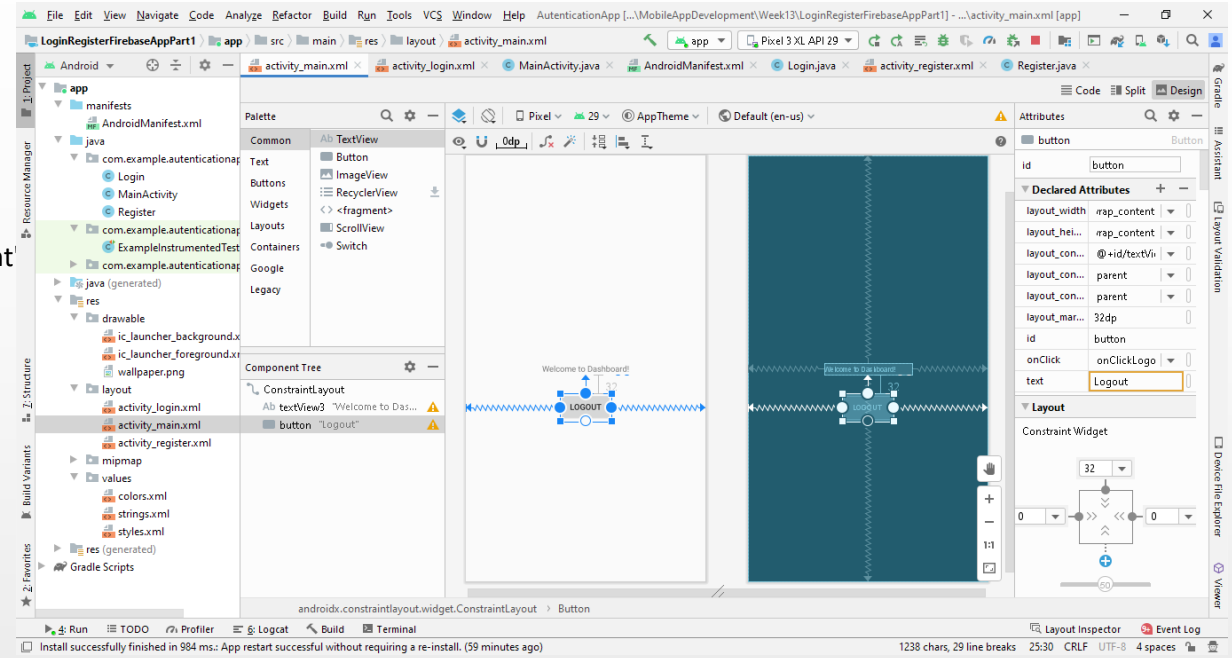
- The XML description of the ProgressBar element (this element shows the progress of the login process; initially, this element is invisible) in the Login activity may look as follows:

```
<ProgressBar
    android:id="@+id/progressBar"
    style="?android:attr/progressBarStyle"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="32dp"
    android:visibility="invisible"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/createText" />
```

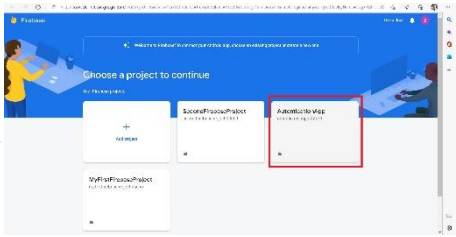
Java Android Authentication App Using Firebase (cont.)

- The XML description of the MainActivity activity may look as follows:

```
<?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:id="@+id/textView3"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Welcome to Dashboard!"
      app:layout_constraintBottom_toBottomOf="parent"
      app:layout_constraintLeft_toLeftOf="parent"
      app:layout_constraintRight_toRightOf="parent"
      app:layout_constraintTop_toTopOf="parent" />
   <Button
      android:id="@+id/button"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_marginTop="32dp"
      android:onClick="onClickLogout"
      android:text="Logout"
      app:layout_constraintEnd_toEndOf="parent"
      app:layout_constraintStart_toStartOf="parent"
      app:layout_constraintTop_toBottomOf="@+id/textView3" />

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



Java Android Authentication App Using Firebase (cont.)



- Indeed, you must have the Firebase project … ☺

console.firebase.google.com/project/authenticationtest-930a8/overview

Firebase

AuthenticationTest

Go to docs

Project Overview

Receive email updates about new Firebase features, research, and events

Sign up

Product categories

Build

Release & Monitor

Analytics

Engage

All products

AuthenticationTest

Spark plan

Get started by adding Firebase to your app

Add an app to get started

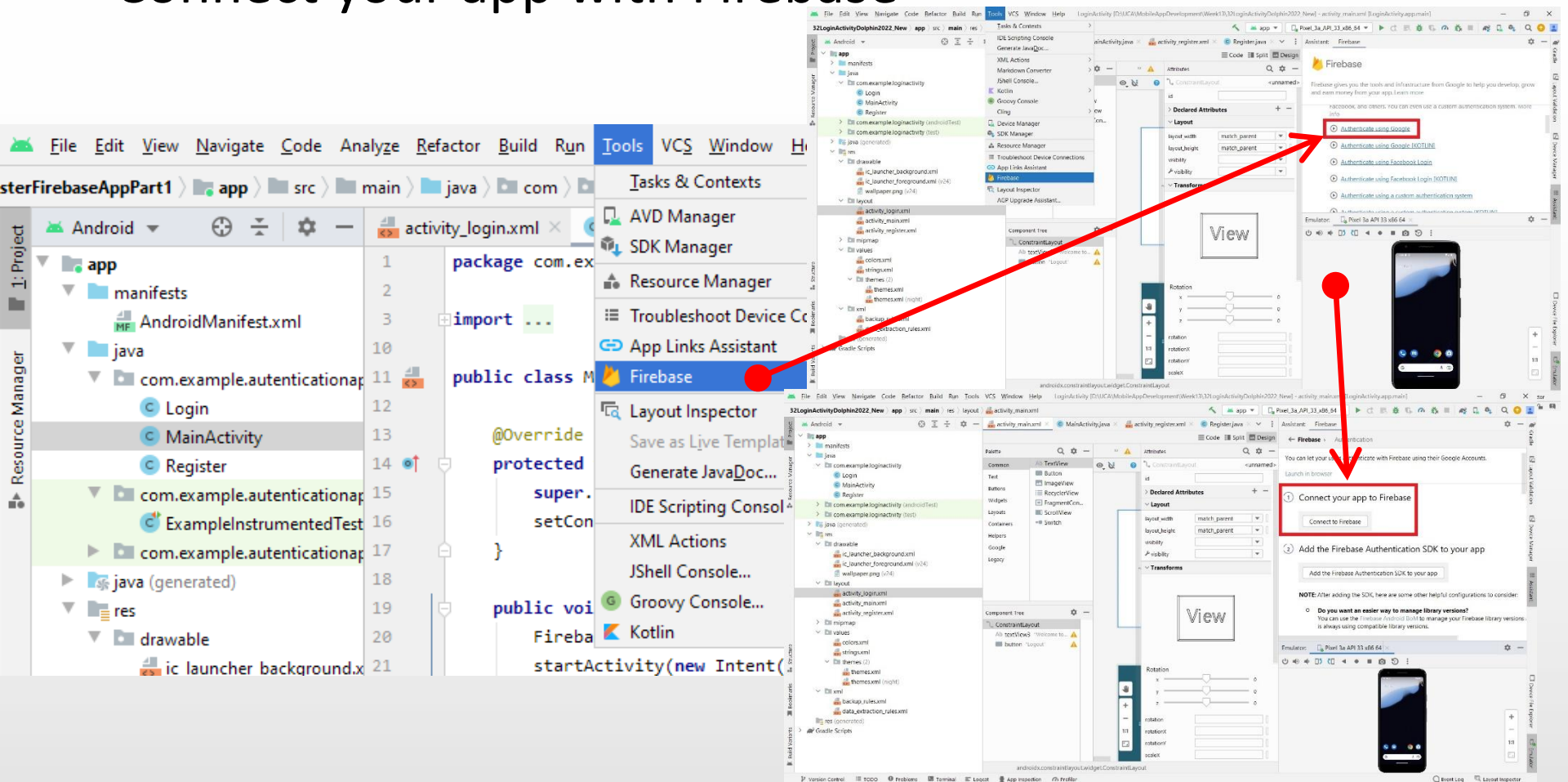
Spark
No-cost $0/month

Upgrade

Store and sync app data in milliseconds

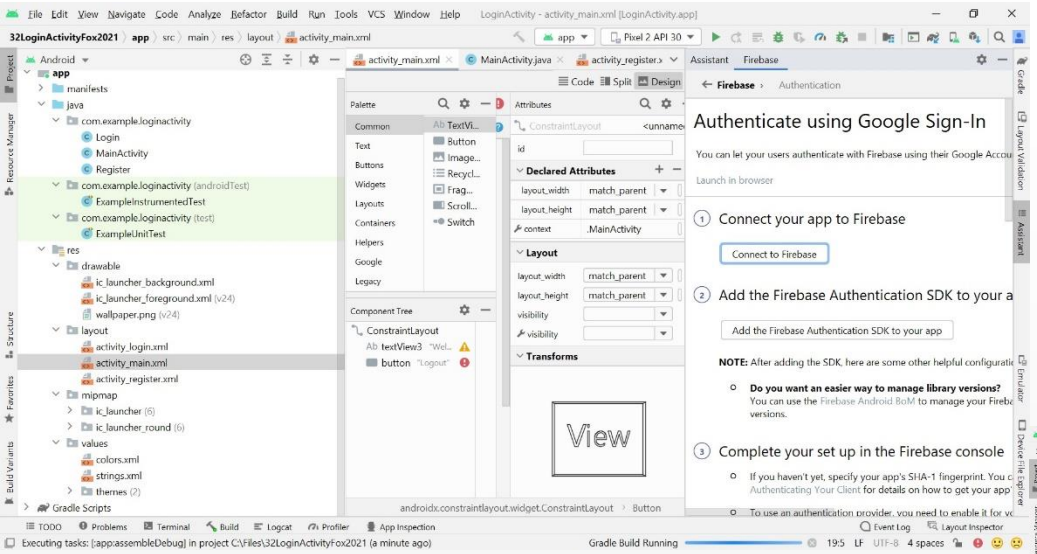
Java Android Authentication App Using Firebase (cont.)

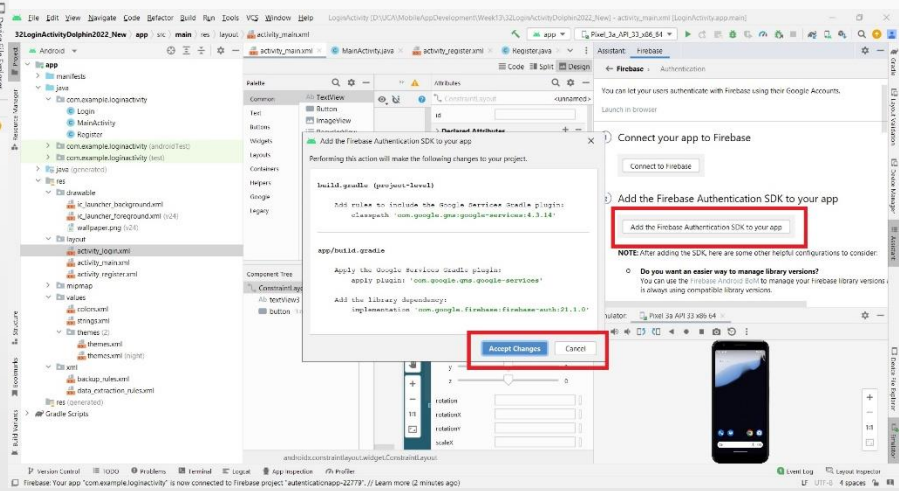
- Connect your app with Firebase



Java Android Authentication App Using Firebase (cont.)

- Connect your app with Firebase (cont.)

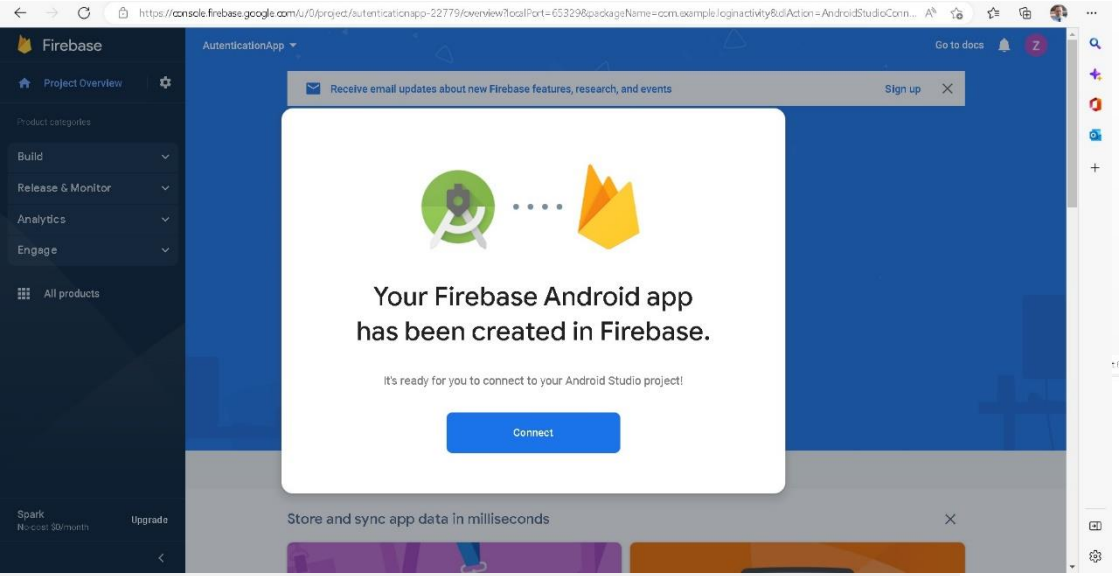


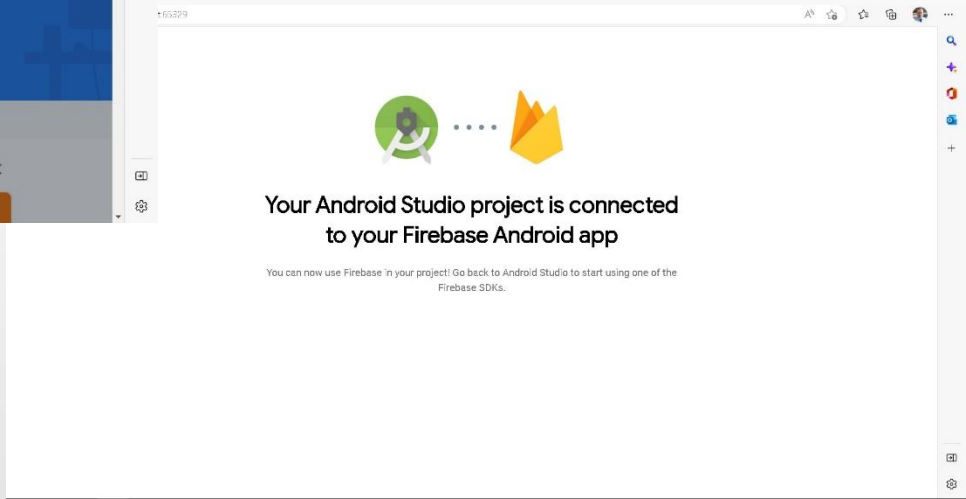


In my app's development, only 1st and 2nd steps were in use …

Java Android Authentication App Using Firebase (cont.)

- Connect your app with Firebase (cont.)





Java Android Authentication App Using Firebase (cont.)

- The XML description of the AndroidManifest.xml may look as follows:

```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">
  <uses-permission android:name="android.permission.INTERNET"/>
  <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.LoginActivity"
    tools:targetApi="31">
    <activity
      android:name=".Register"
      android:exported="true">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>

      <meta-data
        android:name="android.app.lib_name"
        android:value="" />
    </activity>
    <activity
      android:name=".Login"
      android:exported="false">
      <meta-data
        android:name="android.app.lib_name"
        android:value="" />
    </activity>
    <activity
      android:name=".MainActivity"
      android:exported="false">
      <meta-data
        android:name="android.app.lib_name"
        android:value="" />
    </activity>
  </application>

</manifest>
```

Here, we add the permission to use Internet

Here, we see that the activity "Register" will be shown first

Java Android Authentication App Using Firebase (cont.)

- MainActivity.java

```
public class MainActivity extends AppCompatActivity {

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

    public void onClickLogout(View view) {
        FirebaseAuth.getInstance().signOut();//logout
        startActivity(new Intent(getApplicationContext(),Login.class));
        finish();
    }
}
```

Java Android Authentication App Using Firebase (cont.)

- Register.java

```
public class Register extends AppCompatActivity {
    private static final String TAG = MainActivity.class.getSimpleName();
    EditText mFullName,mEmail,mPassword,mPhone;
    Button mRegisterBtn;
    TextView mLoginBtn;
    FirebaseAuth fAuth;
    ProgressBar progressBar;

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

        mFullName   = findViewById(R.id.fullName);
        mEmail      = findViewById(R.id.Email);
        mPassword   = findViewById(R.id.password);
        mPhone      = findViewById(R.id.phone);
        mRegisterBtn= findViewById(R.id.registerBtn);
        mLoginBtn   = findViewById(R.id.createText);

        fAuth = FirebaseAuth.getInstance();
        progressBar = findViewById(R.id.progressBar);

        if(fAuth.getCurrentUser() != null){
            startActivity(new Intent(getApplicationContext(),MainActivity.class));
            finish();
        }
    }
    public void onClickLoginFromRegister (View view){
        startActivity(new Intent(getApplicationContext(),Login.class));
    }
    public void onClickRegister (View view){
        final String email = mEmail.getText().toString().trim();
        String password = mPassword.getText().toString().trim();

        if(TextUtils.isEmpty(email)){
            mEmail.setError("Email is Required.");
            return;
        }

        if(TextUtils.isEmpty(password)){
            mPassword.setError("Password is Required.");
            return;
        }

        if(password.length() < 6){
            mPassword.setError("Password Must be >= 6 Characters");
            return;
        }

        progressBar.setVisibility(View.VISIBLE);

        // Now, we register the user in Firebase
        fAuth.createUserWithEmailAndPassword(email,password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Toast.makeText(Register.this, "User Created.", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(getApplicationContext(),MainActivity.class));
                }else {
                    Toast.makeText(Register.this, "Error ! " + task.getException().getMessage(), Toast.LENGTH_SHORT).show();
                    progressBar.setVisibility(View.GONE);
                }
            }
        });
    }

}
```

Java Android Authentication App Using Firebase (cont.)

- Login.java

```
public class Login extends AppCompatActivity {
    EditText mEmail, mPassword;
    Button mLoginBtn;
    TextView mCreateBtn;
    ProgressBar progressBar;
    FirebaseAuth fAuth;

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

        mEmail = findViewById(R.id.Email1);
        mPassword = findViewById(R.id.Password);
        progressBar = findViewById(R.id.progressBar);
        fAuth = FirebaseAuth.getInstance();
        mLoginBtn = findViewById(R.id.loginBtn);
        mCreateBtn = findViewById(R.id.createText);
    }

    public void onClickRegisterFromLogin(View view) {
        startActivity(new Intent(getApplicationContext(),Register.class));
    }

    public void onClickLogin(View view) {
        String email = mEmail.getText().toString().trim();
        String password = mPassword.getText().toString().trim();

        if(TextUtils.isEmpty(email)){
            mEmail.setError("Email is Required.");
            return;
        }

        if(TextUtils.isEmpty(password)){
            mPassword.setError("Password is Required.");
            return;
        }

        if(password.length() < 6){
            mPassword.setError("Password Must be >= 6 Characters");
            return;
        }

        progressBar.setVisibility(View.VISIBLE);

        // Now, we authenticate the user
        fAuth.signInWithEmailAndPassword(email,password).addOnCompleteListener(new OnCompleteListener<AuthResult>() {
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {
                if(task.isSuccessful()){
                    Toast.makeText(Login.this, "Logged in Successfully", Toast.LENGTH_SHORT).show();
                    startActivity(new Intent(getApplicationContext(),MainActivity.class));
                }else {
                    Toast.makeText(Login.this, "Error ! " + task.getException().getMessage(), Toast.LENGTH_SHORT).show();
                    progressBar.setVisibility(View.GONE);
                }

            }
        });

    }
}
```

Java Android Authentication App Using Firebase (cont.)

- Pixel API 33 AVD starts Java Android Authentication App

32LoginActivityDolphin2022_New › app › src › main › AndroidManifest.xml

app | Pixel_3a_API_33_x86_64

Android

- app
  - manifests
    - AndroidManifest.xml
  - java
    - com.example.loginactivity
      - Login
      - MainActivity
      - Register
    - com.example.loginactivity (androidTest)
    - com.example.loginactivity (test)
  - java (generated)
  - res
    - drawable
      - ic_launcher_background.xml
      - ic_launcher_foreground.xml (v24)
      - wallpaper.png (v24)
    - layout
      - activity_login.xml
      - activity_main.xml
      - activity_register.xml
    - mipmap
    - values
      - colors.xml
      - strings.xml
      - themes (2)
        - themes.xml
        - themes.xml (night)
    - xml
      - backup_rules.xml
      - data_extraction_rules.xml
  - res (generated)
- Gradle Scripts

AndroidManifest.xml | MainActivity.java | activity_register.xml | Register.java | activity_login.xml | Log

```
<uses-permission android:name="android.permission.INTERNET"/>
<application
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="LoginActivity"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.LoginActivity"
    tools:targetApi="31">
    <activity
        android:name=".Register"
        android:exported="true">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>

        <meta-data
            android:name="android.app.lib_name"
            android:value="" />
    </activity>
    <activity
        android:name=".Login"
        android:exported="false">
        <meta-data
            android:name="android.app.lib_name"
```

manifest › application › activity

Text | Merged Manifest

Run 'app' Shift+F10

Assistant: Firebase

Firebase › Authentication

Add the Firebase Authentication

Dependencies set up correctly

Emulator: Pixel 3a API 33 x86 64

Version Control | Run | TODO | Problems | Terminal | Logcat | App Inspection | Build | Profiler | Event Log | Layout Inspector

Run selected configuration

37:38 CRLF UTF-8 4 spaces

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