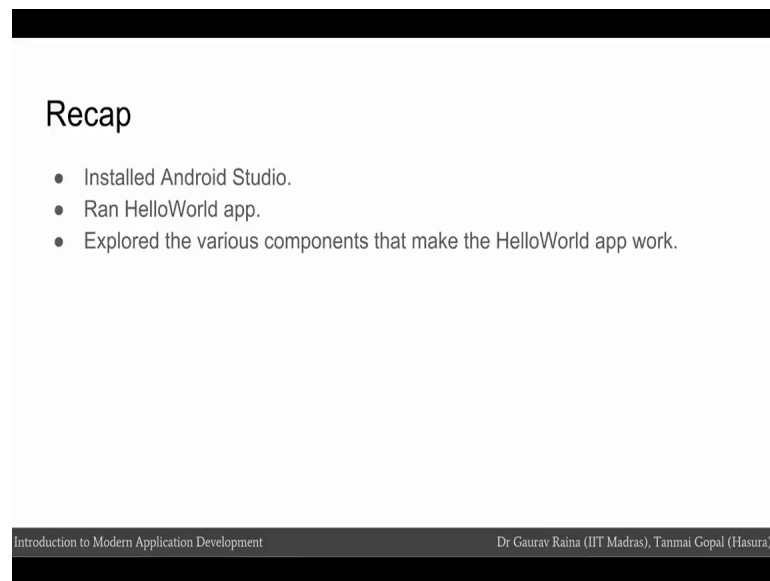


**Introduction to Modern Application Development**  
**Dr. Gaurav Raina**  
**Prof. Tanmai Gopal**  
**Department of Computer Science and Engineering**  
**Indian Institute of Technology, Madras**

**Module - P13**  
**Lecture - 31**  
**Building custom UI using XML and Logs**

(Refer Slide Time: 00:01)

A slide titled "Recap" with a list of three bullet points. The slide has a white background with a black header bar at the top and a black footer bar at the bottom. The footer bar contains the text "Introduction to Modern Application Development" on the left and "Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)" on the right.

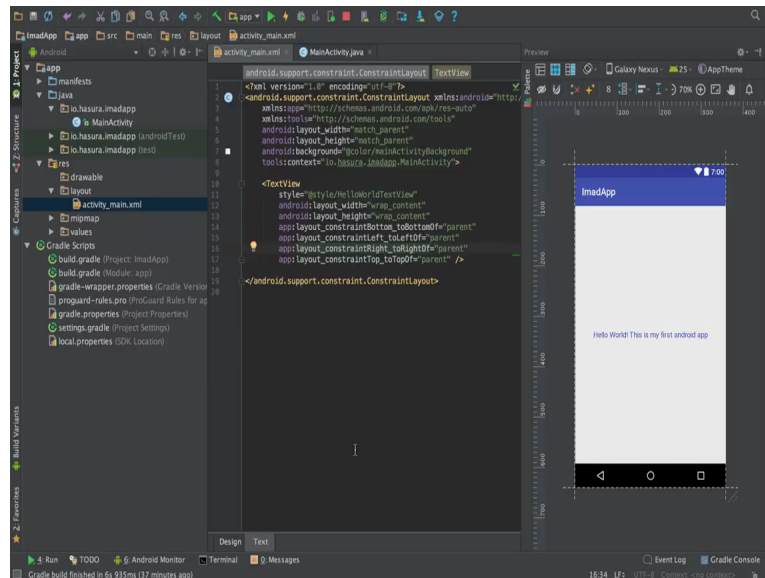
**Recap**

- Installed Android Studio.
- Ran HelloWorld app.
- Explored the various components that make the HelloWorld app work.

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

In the previous module we installed android studio on our system we created a new project and ran our first HelloWorld application. We explored the various components that made the HelloWorld application and understood how these components work. Now that we have a better understanding of how a simple HelloWorld app works let us work on tweaking it a little bit.

(Refer Slide Time: 00:17)



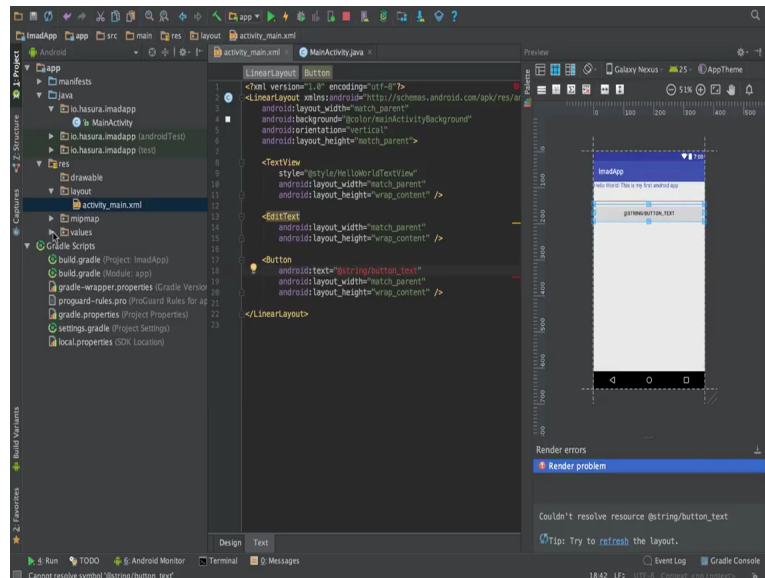
First thing let us head on to the activity underscore main dot XML file and change the UI a little bit. Currently our view uses a constraint layout as its root view apart from constraint layout android also provides many other layouts like the grid layouts, the frame layout, linear layout is of 2 types one is vertical one is horizontal and there is a relative layout table layout etcetera. The constraint layout was released recently and it is a new layout that android has brought forward I have provided a link in the slides for you to get a better understanding of constraint layout.

Also there are 2 ways to build UI in android studio you can either choose to code it using XML or use the interface provided by android the interface provide by android will seem a little confusing in the beginning, but with practice you get really quick and fast at it. I would suggest that you go through the link provided in the slides for the constraint layout and get used to the constraint layout as soon as possible. If you see this about android application development for the time being we are going to use XML to co to make our UI. So, that you get a better understanding of how it works.

So, now, I am trying to build a UI wherein I am trying to get a screen with a text view on top followed by a edit text a text box below, it a text box is something there the user can enter using the keyboard it is called the edit text in android and below the edit text I want to put a button which will basically its basically the functionality that we are trying to

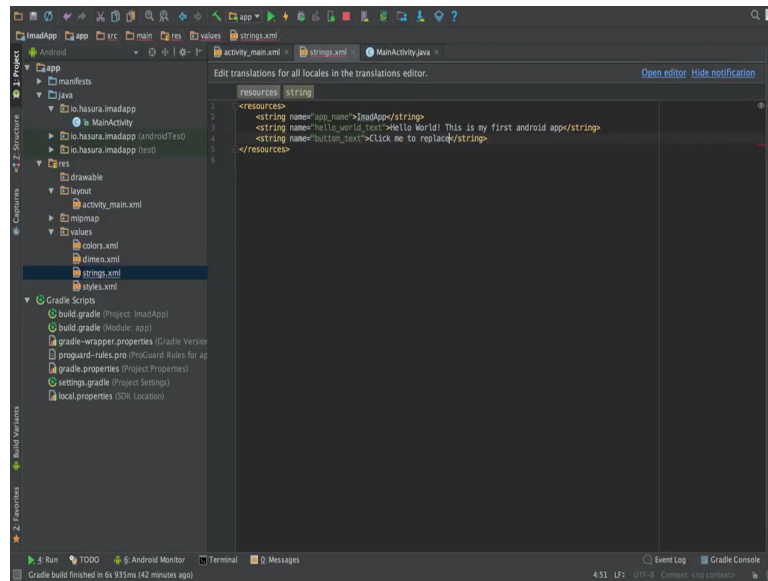
implement is that whatever text the user enters in the edit textbox and once he clicks on the button the text in the text view will be replaced by the text that the user has entered.

(Refer Slide Time: 02:30)



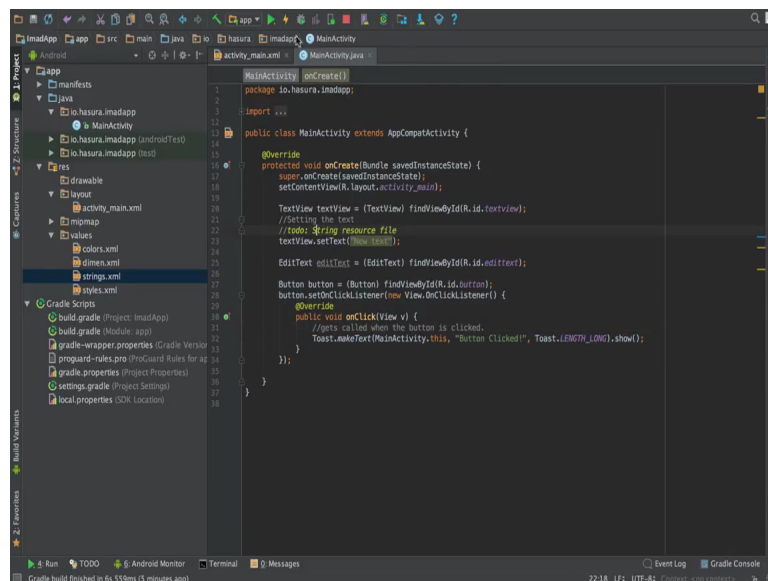
So, let us see how we can do this, let us take off everything and build it from scratch let us start with a linear layout as a root view. Here android studio is saying that the main space detoration for the android is not in this XML file to auto input this main space click on alt and enter together then we can add the layout width as match parent the layout height as match parent and also specify an orientation for this to be vertical. Now that we have a linear layout set up we can also put the background image from here. Now first we need a text view let us leave this as match parent and wrap content and give it a style of what we defined earlier. Next let us put a edit text wrap content and finally, a button which can also be match wrap and let us put a text to this which can be click me to replace and as I said earlier it is not good practice to and let us go here strings and put here, alright.

(Refer Slide Time: 04:18)



```
1 <resources>
2   <string name="app_name">ImadApp</string>
3   <string name="hello_world">Hello World! This is my first android app</string>
4   <string name="button_text">Click me to replace</string>
5 </resources>
```

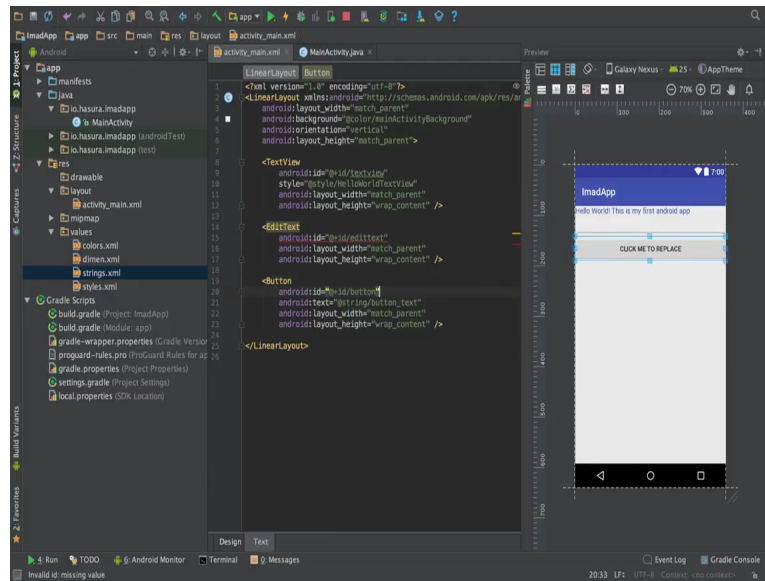
(Refer Slide Time: 04:35)



```
1 package io.hasura.imadapp;
2
3 import androidx.appcompat.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.view.View;
6 import android.view.View.OnClickListener;
7 import android.widget.Button;
8 import android.widget.EditText;
9 import android.widget.TextView;
10
11 public class MainActivity extends AppCompatActivity {
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17
18         TextView textView = (TextView) findViewById(R.id.textView);
19         //Setting the text
20         //todo: String resource file
21         textView.setText("New text!");
22
23         EditText editText = (EditText) findViewById(R.id.editText);
24
25         Button button = (Button) findViewById(R.id.button);
26         button.setOnClickListener(new View.OnClickListener() {
27             @Override
28             public void onClick(View v) {
29                 //gets called when the button is clicked.
30                 Toast.makeText(MainActivity.this, "Button Clicked", Toast.LENGTH_LONG).show();
31             }
32         });
33     }
34 }
35
36 }
```

So, now we have a basic UI set up let us run our app and see how it looks. So, our view looks good.

(Refer Slide Time: 05:08)

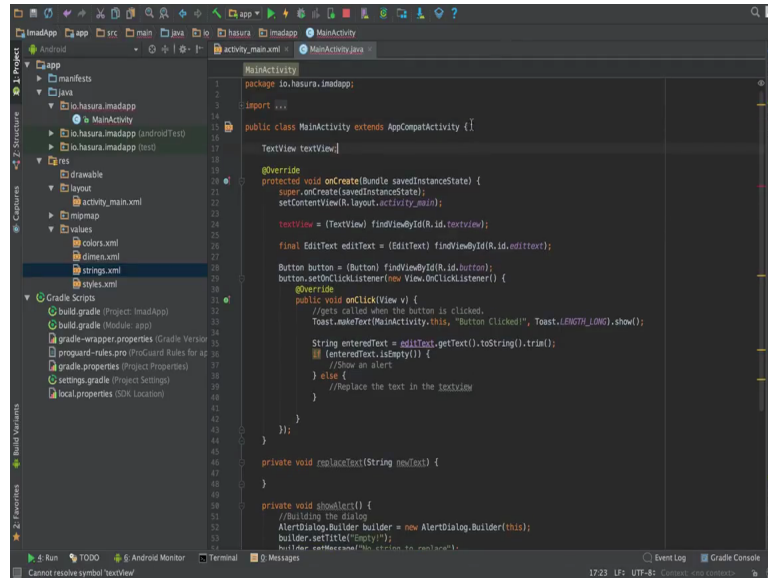


Next let us see how we can access these widgets that we have added as in web app you have to add ids for the widget that is you need. So, let us add ids for the text view it is called the text view id would be edit text and for the button it will be button and to access these in main activity we simply have to define a variable first text view and the use the id that is assigned similarly for the edit text and for the button again. So, button we have to put the button widget into the main activity or java plus that we have to press alt and enter please note that on a vendor system this might be different android studio will provide you a prompt with how to do this button.

Now, to programmatically set a text for this text view we can do something like this and to access the click of the button se we have to set an on click listener on to the button and this is basically an interface that we have to implement so that one way to do it would be this way. So, now, we have set a click listener on to this button which is an interface. So, each time the button is clicked this method will get called. Let us see if this works, again please note that you should always use (Refer Time: 07:27) for these kind of changes. Let us show toast when the button is clicked toast is an android widget which you will see in action in a few seconds. So, to create a toast you have to do toast dot make text pass it the activity context which would be main activity dot this pass in this string that you want to show button click and choose the length to which you want to show it for.

So, there are 2 options for this length short or length long ops let us show length long and your dot show all right, let us run the app and see this. As you can see button is clicked the new text has been assigned programmatically.

(Refer Slide Time: 08:38)



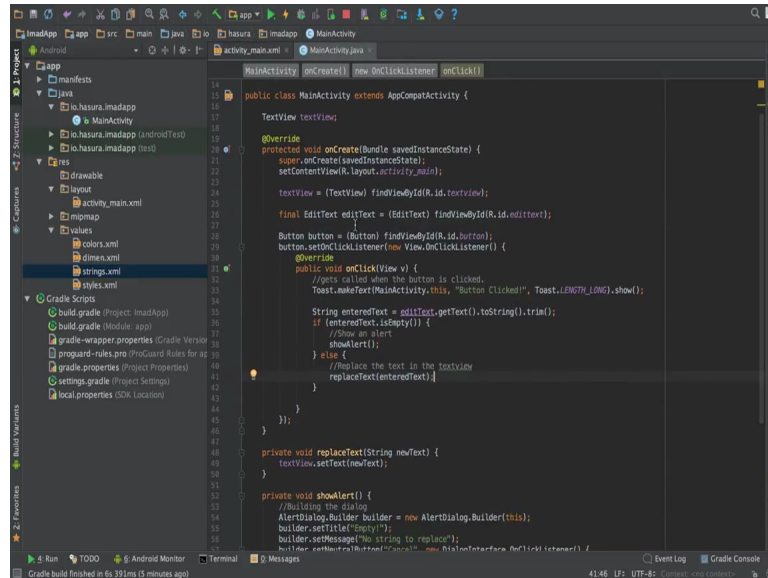
```
1 package io.hasura.imadapp;
2 import java.util.*;
3 import androidx.appcompat.app.AppCompatActivity;
4 import android.os.Bundle;
5 import android.view.View;
6 import android.view.View.OnClickListener;
7 import android.widget.Button;
8 import android.widget.EditText;
9 import android.widget.TextView;
10
11 public class MainActivity extends AppCompatActivity {
12
13     TextView textView;
14
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         textView = (TextView) findViewById(R.id.textView);
21
22         final EditText editText = (EditText) findViewById(R.id.editText);
23
24         Button button = (Button) findViewById(R.id.button);
25         button.setOnClickListener(new View.OnClickListener() {
26             @Override
27             public void onClick(View v) {
28                 //gets called when the button is clicked.
29                 Toast.makeText(MainActivity.this, "Button Clicked", Toast.LENGTH_LONG).show();
30
31                 String enteredText = editText.getText().toString().trim();
32                 if (enteredText.isEmpty()) {
33                     //Show an alert
34                 } else {
35                     //Replace the text in the textView
36                 }
37             }
38         });
39
40     private void replaceText(String newText) {
41
42     }
43
44     private void showAlert() {
45         //Building the dialog
46         AlertDialog.Builder builder = new AlertDialog.Builder(this);
47         builder.setTitle("Alert");
48         builder.setMessage("No string to replace");
49     }
50 }
```

So, now let us implement the function that we set out to implement. Let us take this out and you can see this was the toast I was talking about button clicked alright. So, on button clicked first let us check if the user has entered some text into the edit text box. So, will be equal to edit text dot get text to string, so, this is the string that we use and entered and if he has not entered anything then this would be empty string. Let us also trim it take off all these places from the closing edges, let us check if this string is empty in case this string is empty let us show an alert and if it is not replace the text with let us create a method to show an alert to show an alert you have to first create a builder and send it a context which will be this; this is to set the title to the alert set message no string to replace and let us also set a neutral button it is just cancel and (Refer Time: 10:38) on click this now.

(Refer Time: 10:43) can see. So, here when the cancel button is clicked we the method on click gets called which gives us the dialogue let us dismiss the dialogue and finally, builder dot show, to show the dialogue. This is to build the dialogue. Next let us make a method to replace the text. This can be new text and so this method receives a new text

and we sync; now we have to access this text view in this method for this we have to make this text view global.

(Refer Slide Time: 11:52)



```
public class MainActivity extends AppCompatActivity {
    TextView textView;

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

        textView = (TextView) findViewById(R.id.textview);

        final EditText editText = (EditText) findViewById(R.id.edittext);

        Button button = (Button) findViewById(R.id.button);
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //gets called when the button is clicked.
                Toast.makeText(MainActivity.this, "Button Clicked", Toast.LENGTH_LONG).show();

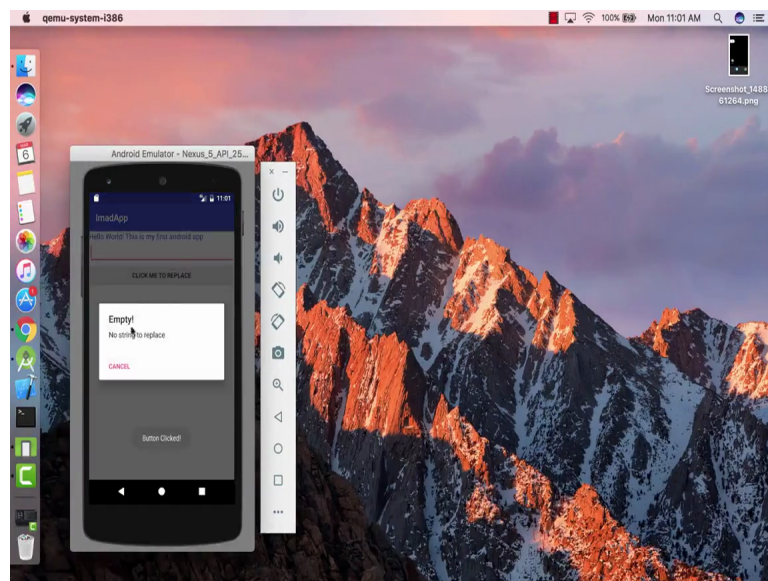
                String enteredText = editText.getText().toString().trim();
                if (enteredText.isEmpty()) {
                    //Show an alert
                    showAlert();
                } else {
                    //Replace the text in the textView
                    replaceText(enteredText);
                }
            }
        });
    }

    private void replaceText(String newText) {
        textView.setText(newText);
    }

    private void showAlert() {
        //Building the dialog
        AlertDialog.Builder builder = new AlertDialog.Builder(this);
        builder.setTitle("Empty");
        builder.setMessage("No string to replace");
        builder.setPositiveButton("OK", null);
        builder.setNegativeButton("Cancel", null);
        builder.show();
    }
}
```

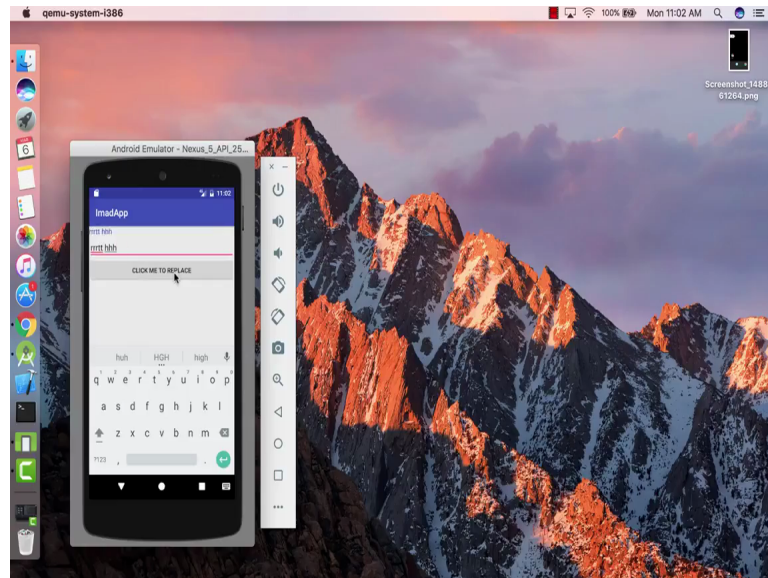
So, let us do that. This (Refer Time: 11:46). So, now, we have text view dot set text new text new text alright. Now, that can be done here is show alert or replace text with entered text. Now this should work.

(Refer Slide Time: 12:33)



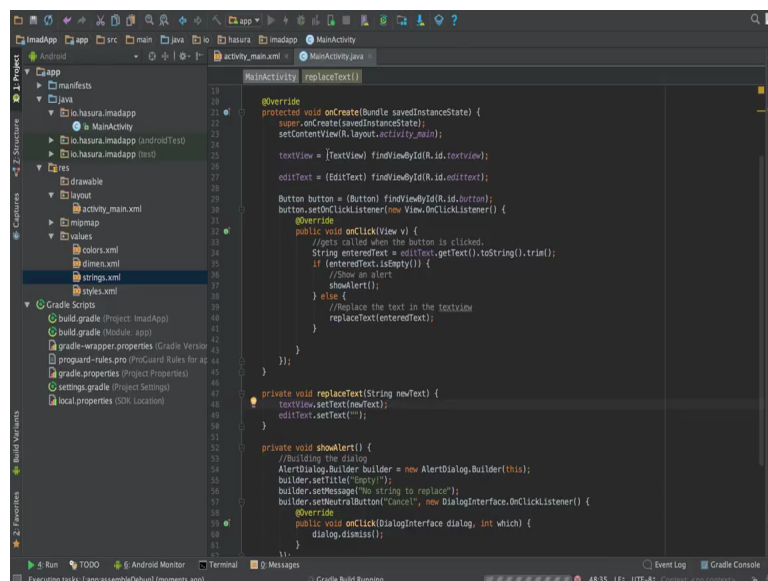
So, now let us run this app and see this in action there is alert or we still showing the toast because we have not really taken the toast out of here let us do that run the app again.

(Refer Slide Time: 12:58)



So, if you if you click the button with no string with nothing entered it shows us a alert and we can enter anything you like and that gets replaced.

(Refer Slide Time: 13:32)

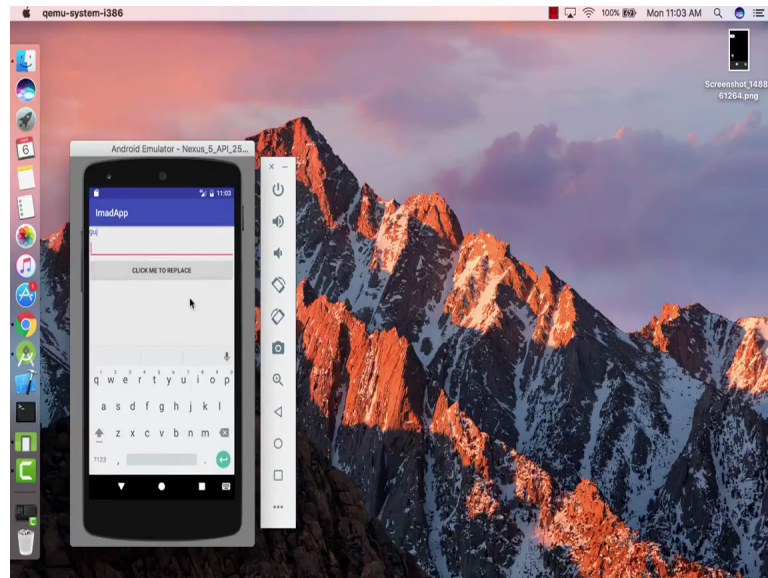


Let us also do, let us also implement feature wherein once the button is clicked the edit text is emptied for this we need to do it when the text is replaced we need to access the



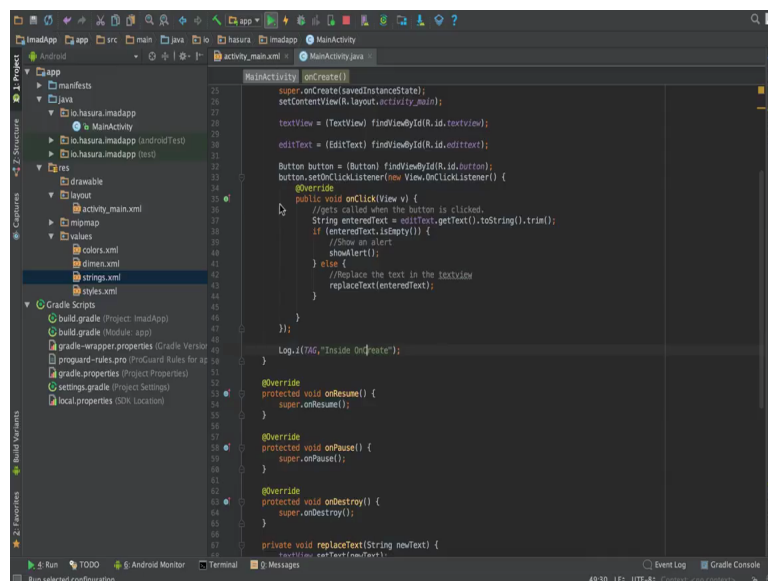
edit text globally as well. Edit text has become final because we are accessing it inside the interface. So, now, here let us do edit text or set text here now let us run the app, alright.

(Refer Slide Time: 14:03)



So, it works then I would like to talk about log statements logs are helpful to debug your code and better understand what is exactly happening in your app.

(Refer Slide Time: 14:36)



The easiest way to use set up a log is call log dot i which stands for informative apart from i we have d e v which stands for debug, error and verbose in this case let us use I, it

accepts 2 parameters one is a tag and other one is a message. So, the tag would be usually you can make the tag as the place where you add and the message can be inside on create better practice would be to set up a static string called tag for all the logs in your activity and simply replace. This good exercise that all of you should try out would be to implement the life cycles method like on resume, on pause, on destroy put the log statements in each of these and see it in action.

(Refer Slide Time: 15:05)

```

1 package io.hasura.imadapp;
2 import androidx.appcompat.app.AppCompatActivity;
3 import android.os.Bundle;
4 import android.os.Bundle;
5 import android.os.Bundle;
6 import android.os.Bundle;
7 import android.os.Bundle;
8 import android.os.Bundle;
9 import android.os.Bundle;
10 import android.os.Bundle;
11 import android.os.Bundle;
12 import android.os.Bundle;
13 import android.os.Bundle;
14 import android.os.Bundle;
15 import android.os.Bundle;
16 public class MainActivity extends AppCompatActivity {
17     TextView textView;
18     EditText editText;
19     private static final String TAG = "MainActivity";
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_main);
24         textView = (TextView) findViewById(R.id.textView);
25         editText = (EditText) findViewById(R.id.editText);
26         Button button = (Button) findViewById(R.id.button);
27         button.setOnClickListener(new View.OnClickListener() {
28             @Override
29             public void onClick(View v) {
30                 //gets called when the button is clicked
31                 String enteredText = editText.getText().toString().trim();
32                 if (enteredText.isEmpty()) {
33                     //Show an alert
34                     showAlert();
35                 } else {
36                     //Replace the text in the textView
37                     replaceText(enteredText);
38                 }
39             }
40         });
41         Log.i("MainActivity", "Inside onCreate");
42     }
43     private void showAlert() {
44         android.support.v7.app.AlertDialog.Builder builder = new android.support.v7.app.AlertDialog.Builder(this);
45         builder.setTitle("Alert");
46         builder.setMessage("Please enter text");
47         builder.setPositiveButton("OK", null);
48         builder.show();
49     }
50     private void replaceText(String newText) {
51         textView.setText(newText);
52         editText.setText("");
53     }
54 }
    
```

(Refer Slide Time: 15:56)

```

25 super.onCreate(savedInstanceState);
26 setContentView(R.layout.activity_main);
27 textView = (TextView) findViewById(R.id.textView);
28 editText = (EditText) findViewById(R.id.editText);
29 Button button = (Button) findViewById(R.id.button);
30 button.setOnClickListener(new View.OnClickListener() {
31     @Override
32     public void onClick(View v) {
33         //gets called when the button is clicked
34         String enteredText = editText.getText().toString().trim();
35         if (enteredText.isEmpty()) {
36             //Show an alert
37             showAlert();
38         }
39     }
40 });
41 Log.i("MainActivity", "Inside onCreate");
42 }
43 private void showAlert() {
44     android.support.v7.app.AlertDialog.Builder builder = new android.support.v7.app.AlertDialog.Builder(this);
45     builder.setTitle("Alert");
46     builder.setMessage("Please enter text");
47     builder.setPositiveButton("OK", null);
48     builder.show();
49 }
50 private void replaceText(String newText) {
51     textView.setText(newText);
52     editText.setText("");
53 }
54 }
    
```

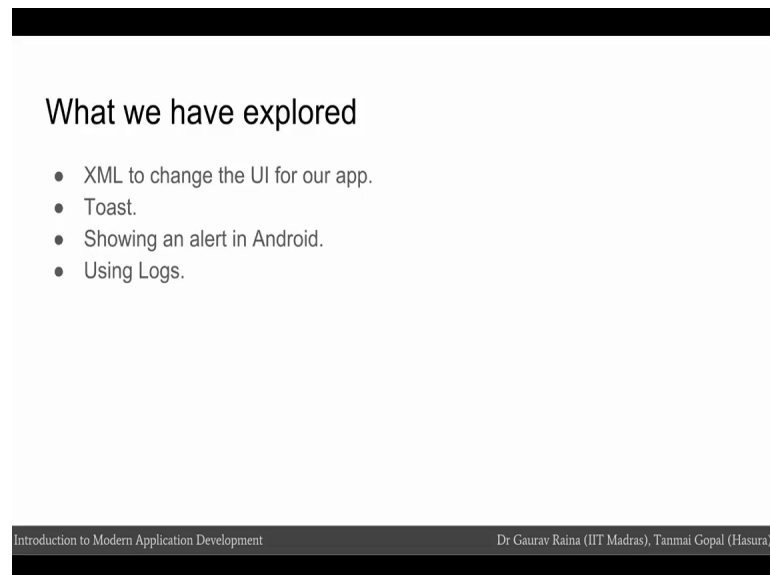
```

09-06 11:12:34.354 5288-5288/? I/art: Not late-enabling -Xcheckjni (already on)
09-06 11:12:34.357 5288-5288/? W/art: Unexpected CPU variant for X86 using defaults: x86
09-06 11:12:34.688 5288-5215/? E/art: Failed sending reply to debugger: Broken pipe
09-06 11:12:34.689 5288-5215/? I/art: Debugger is no longer active
09-06 11:12:34.689 5288-5215/? I/art: Starting a blocking GC Instrumentation
09-06 11:12:35.181 5288-5288/? W/System: ClassLoader referenced unknown path: /data/app/io.hasura.imadapp-2/lib/x86
09-06 11:12:35.244 5288-5288/? I/InstantRun: Starting Instant Run Server for io.hasura.imadapp
09-06 11:12:37.458 5288-5288/io.hasura.imadapp W/art: Before Android 4.2, method android.graphics.PorterDuffColorFilter androidx.support.graphics.drawable.VectorDrawableCompat$TintAwareColorFilterImpl.<init>()Landroid/graphics/PorterDuffColorFilter; took 128.27ms
09-06 11:12:37.613 5288-5288/io.hasura.imadapp W/art: Verification of void androidx.support.v4.app.NavUtils.<clinit>()Landroid/os/Bundle; took 128.27ms
09-06 11:12:38.017 5288-5288/io.hasura.imadapp I/MainActivity: Inside onCreate
09-06 11:12:38.133 5288-5288/io.hasura.imadapp I/OpenGLRenderer: Initialized EGL, version 1.4
09-06 11:12:38.133 5288-5288/io.hasura.imadapp D/OpenGLRenderer: Swap behavior 1
09-06 11:12:38.185 5288-5288/io.hasura.imadapp E/EGL_emulation: tid 5288: eglSurfaceAttr(1174): error 0x3089 (EGL_BAD_MATCH)
09-06 11:12:38.187 5288-5288/io.hasura.imadapp W/OpenGLRenderer: Failed to set EGL_SWAP_BEHAVIOR on surface #0d589384, error=EGL_BAD_MATCH
09-06 11:12:38.219 5288-5288/io.hasura.imadapp W/art: Before Android 4.1, method int androidx.support.v7.widget.ListViewCompat.lookForSelectablePosition(int, boolean)
    
```

To view log statements you simply have to run the app and check the android monitor just click here as you can see there is a log which says the tag and main activity which says inside on create. You can filter it based on your tags and your log also you can change the different types over here to filter them in further.

With this we come to the end of this module. In this module we have looked at how to use XML to change the UI for our app we have used android widgets like a toast and an alert. We have also started using logs in our app developer dot android dot com is the website to visit to learn more about android.

(Refer Slide Time: 16:16)



**What we have explored**

- XML to change the UI for our app.
- Toast.
- Showing an alert in Android.
- Using Logs.

Introduction to Modern Application Development Dr Gaurav Raina (IIT Madras), Tanmai Gopal (Hasura)

(Refer Slide Time: 16:33)

## Student Tasks

- Use [developer.android.com](https://developer.android.com) to learn more about Android.
- Experiment with UI. There are multiple ways of doing things, find the method best suited to your needs.
- Check out Constraint Layout - <https://developer.android.com/training/constraint-layout/index.html>
- Themes and Styles - <https://developer.android.com/guide/topics/ui/themes.html>
- Use the android community to learn and explore - stackoverflow.
- Get familiar with Android studio and Android studio shortcuts to improve development speed.
- If in doubt, do not hesitate to post in the forum or get in touch via twitter (@JaisonTitus)

It has information from how android works till how to develop our android to in great detail. Experiment with the UI there are multiple ways of doing anything that you want in a mobile app find the method that is best suited to your needs and keep experimenting. Check out constraint layout it is something new that android has released recently.

These (Refer Time: 17:02) are very important while developing a mobile application, they are present to avoid code duplication and to facilitate code reusability. To learn more about themes and styles please visit the link also start implementing themes and styles in your development practice from the start so that it becomes a habit. Use the android community to learn and explore stack overflow will almost always have answers to all your questions. Android studio is a very very powerful tool it is a very powerful id with a lot of features get familiar with android studio and the android studio shortcuts to improve your development speed and if in doubt do not hesitate to post in the forum or get in touch with me via twitter.

(Refer Slide Time: 17:50)

### Next Module

- Building a simple blog app.
- Making network API calls.
- Integrating 3rd party libraries.
- Using a Recycler View.

In the next module we will be building a simple blog app this will be similar to the web app that you guys have built. We will be making the network API calls on this blog app, we would be using the same API calls that we have assigned to you as task at the end of the module. We will also be integrating third party libraries into our apps to make our lives easier and to increase the development speed and finally, we will be using a very important widget in the android ecosystem called the recycler view.