Lab Manual: Event Handling, Toast Notification, and AlertDialog in Android

Objective

By the end of this lab, you will be able to:

1. Understand and implement event handling for different UI elements in Android.
2. Display Toast notifications to provide feedback to the user.
3. Create and customize AlertDialogs for user interaction.

Pre-requisites

- Basic knowledge of Android development and Java/Kotlin programming.
- Familiarity with Android Studio and XML layout design.

Lab Setup

- Install Android Studio.
- Create a new Android project with an empty activity.

Lab Exercise 1: Event Handling

Introduction

Event Handling in Android refers to responding to user interactions such as button clicks, text field changes, etc. In this exercise, you will learn to handle different events using listeners.

Steps:

1. **Create a New Android Project:**
    - Open Android Studio.
    - Select "New Project" -> "Empty Activity."
    - Name your project "EventHandlingDemo."
2. **Modify the Layout File (activity_main.xml):**
    - Add a Button and an EditText to the layout.
    - Below is the sample code for activity_main.xml:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:orientation="vertical"
   android:padding="16dp"
   tools:context=".MainActivity">

   <EditText
      android:id="@+id/editText"
      android:layout_width="match_parent"
      android:layout_height="wrap_content"
      android:hint="Enter Text" />

   <Button
      android:id="@+id/button"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Click Me" />
</LinearLayout>
```

3. **Implement Event Handling in MainActivity.java/MainActivity.kt:**

- Handle the button click event to display the text entered in EditText.

Java:

```
package com.example.eventhandlingdemo;

import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

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

        EditText editText = findViewById(R.id.editText);
        Button button = findViewById(R.id.button);

        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String text = editText.getText().toString();
                                    Toast.makeText(MainActivity.this, "You entered: " + text,
Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Kotlin:

```
package com.example.eventhandlingdemo

import android.os.Bundle
import android.widget.Button
import android.widget.EditText
```

```
import android.widget.Toast
import androidx.appcompat.app.AppCompatActivity

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val editText: EditText = findViewById(R.id.editText)
        val button: Button = findViewById(R.id.button)

        button.setOnClickListener {
            val text = editText.text.toString()
            Toast.makeText(this, "You entered: $text", Toast.LENGTH_SHORT).show()
        }
    }
}
```

4. **Run the Application:**
    - Click on the "Run" button in Android Studio.
    - Test the button click functionality.

Lab Exercise 2: Toast Notification

Introduction

A Toast is a small message that appears at the bottom of the screen and disappears after a few seconds. It provides feedback about an operation in a simple popup.

Steps:

1. **Modify the Code to Show a Toast Notification:**
    - Update the onClick method to display a Toast notification with a custom message.

Java:

```
Toast.makeText(MainActivity.this, "Button Clicked!", Toast.LENGTH_SHORT).show();
```

Kotlin:

```
Toast.makeText(this, "Button Clicked!", Toast.LENGTH_SHORT).show()
```

2. **Customize the Toast Position:**
    - You can change the position of the Toast by setting its gravity.

Java:

```
Toast toast = Toast.makeText(MainActivity.this, "Custom Position Toast", Toast.LENGTH_SHORT);
toast.setGravity(Gravity.CENTER, 0, 0);
toast.show();
```

Kotlin:

kotlin
Copy code

```
val toast = Toast.makeText(this, "Custom Position Toast", Toast.LENGTH_SHORT)
toast.setGravity(Gravity.CENTER, 0, 0)
toast.show()
```

3.
4. **Run the Application:**
    - Observe how the Toast message appears and disappears.

Lab Exercise 3: AlertDialog

Introduction

AlertDialog is used to show alerts to the user. It can display a message and ask for a response (e.g., OK, Cancel).

Steps:

1. **Add a Button for AlertDialog:**
    - Add a new button to activity_main.xml:

```
<Button
   android:id="@+id/alertDialogButton"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="Show Alert Dialog" />
```

2. **Implement AlertDialog in MainActivity.java/MainActivity.kt:**
    - Create an AlertDialog when the button is clicked.

Java:

```
Button alertDialogButton = findViewById(R.id.alertDialogButton);
alertDialogButton.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      new AlertDialog.Builder(MainActivity.this)
         .setTitle("Alert Dialog")
         .setMessage("This is an Alert Dialog. Do you want to continue?")
         .setPositiveButton("Yes", new DialogInterface.OnClickListener() {
            public void onClick(DialogInterface dialog, int which) {
                                                Toast.makeText(MainActivity.this, "You clicked Yes!",
Toast.LENGTH_SHORT).show();
            }
         })
         .setNegativeButton("No", null)
         .show();
   }
```

});
Kotlin:

```
val alertDialogButton: Button = findViewById(R.id.alertDialogButton)
alertDialogButton.setOnClickListener {
    AlertDialog.Builder(this)
        .setTitle("Alert Dialog")
        .setMessage("This is an Alert Dialog. Do you want to continue?")
        .setPositiveButton("Yes") { dialog, _ ->
            Toast.makeText(this, "You clicked Yes!", Toast.LENGTH_SHORT).show()
        }
        .setNegativeButton("No", null)
        .show()
}
```

3. **Run the Application:**
    - Click on the "Show Alert Dialog" button to see the AlertDialog in action.

Conclusion

In this lab, you learned how to handle various UI events, display Toast notifications, and use AlertDialogs in Android applications. These are essential components for creating interactive and user-friendly Android apps.

Assignment

1. Create a custom Toast with an image and text.
2. Implement an AlertDialog with multiple choice options.