Practical No. 4

Title: Android program for notifications and alert box

Aim: Create an application to demonstrate notifications and alert box

Introduction

Android Notification

Android Notification provides short, timely information about the action happened in the application, even it is not running. The notification displays the icon, title and some amount of the content text.

The properties of Android notification are set using NotificationCompat.Builder object. Some of the notification properties are mention below:

Method	Description
setSmallIcon()	It sets the icon of notification.
setContentTitle()	It is used to set the title of notification.
setContentText()	It is used to set the text message.
setAutoCancel()	It sets the cancelable property of notification.
setPriority()	It sets the priority of notification.

Following are steps for creating and sending the notification:

Step 1 - Create Notification Builder

As a first step is to create a notification builder using

NotificationCompat.Builder.build().

NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this);

Step 2 - Setting Notification Properties

Once you have Builder object, you can set its Notification properties using Builder object as per your requirement.

mBuilder.setSmallIcon(R.drawable.notification_icon);

mBuilder.setContentTitle("Notification Alert, Click Me!");

mBuilder.setContentText("Hi, this is Android Notification Detail!");

Step 3 - Attach Actions

This is an optional part and required if you want to attach an action with the notification. An action allows users to go directly from the notification to an Activity in your application, where they can look at one or more events or do further work.

The action is defined by a PendingIntent containing an Intent that starts an Activity in your application.

For example, if you want to start Activity when the user clicks the notification text in the notification drawer, you add the PendingIntent by calling setContentIntent(). A PendingIntent object helps you to perform an action on your applications behalf, often at a later time, without caring of whether or not your application is running.

Step 4 - Issue the notification

Finally, you pass the Notification object to the system by calling NotificationManager.notify() to send your notification. Make sure you call NotificationCompat.Builder.build() method on builder object before notifying it.

NotificationManagermNotificationManager=(NotificationManager)getSystemService(Conte
xt.NOTIFICATION_SERVICE);

// notificationID allows you to update the notification later on. mNotificationManager.notify(notificationID, mBuilder.build());

Android Alert Box

Alert Dialog shows the Alert message and gives the answer in the form of yes or no. Alert Dialog displays the message to warn you and then according to your response, the next step is processed. Android Alert Dialog is built with the use of three fields: Title, Message area, and Action Button.

Alert Dialog code has three methods:

Method	Description
setTitle()	method for displaying the Alert Dialog box Title
setMessage()	method for displaying the message
setIcon()	method is used to set the icon on the Alert dialog box

Exercise - Create android application to demonstrate notifications and alert box

Implementation:

Program:

Activity_main.xml:

```
<?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:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:text="@string/AndroidNotification"
   app:layout_constraintBottom_toBottomOf="parent"
   app:layout_constraintHorizontal_bias="0.496"
   app:layout_constraintLeft_toLeftOf="parent"
   app:layout_constraintRight_toRightOf="parent"
   app:layout_constraintTop_toTopOf="parent"
   app:layout_constraintVertical_bias="0.294"
   android:textAlignment="center"
   />
<Button
   android:id="@+id/button"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:layout_marginStart="8dp"
   android:layout_marginEnd="8dp"
   android:layout_marginBottom="320dp"
   android:text="@string/Notify"
   app:layout_constraintBottom_toBottomOf="parent"
```

```
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        />
    </androidx.constraintlayout.widget.ConstraintLayout>
```

Activity_notification_view.xml:

```
    <?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=".NotificationView">

        <TextView
            android:id="@+id/textView2"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:gravity="center"
            android:text="Detailed Message"
            app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="1.0"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintVertical_bias="0.24"
    tools:ignore="MissingConstraints"

    />

  <TextView
    android:id="@+id/textview"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    android:textAlignment="center"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@id/textView2"
    app:layout_constraintVertical_bias="0.217"
    />
```

```
  <Button
     android:id="@+id/alt"
     android:layout_width="wrap_content"
     android:layout_height="wrap_content"
     android:layout_marginTop="476dp"
     android:text="CLOSE"
     app:layout_constraintEnd_toEndOf="parent"
     app:layout_constraintStart_toStartOf="parent"
     app:layout_constraintTop_toTopOf="parent"
     />

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

MainActivity.java:

```
package com.example.notificationandalert;

import android.app.Notification;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.content.Context;
import android.content.Intent;
```

```
import android.os.Build;
import android.os.Bundle;
import android.widget.Button;

import androidx.activity.EdgeToEdge;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.NotificationCompat;
import androidx.core.app.RemoteInput;
import androidx.core.graphics.Insets;
import androidx.core.view.ViewCompat;
import androidx.core.view.WindowInsetsCompat;

public class MainActivity extends AppCompatActivity {

    private static final String KEY_TEXT_REPLY = "key_text_reply";
    Button button;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        EdgeToEdge.enable(this);
        setContentView(R.layout.activity_main);
        button = findViewById(R.id.button);
```

```
    button.setOnClickListener(v -> addNotification());
    ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main), (v, insets) -> {
        Insets systemBars = insets.getInsets(WindowInsetsCompat.Type.systemBars());
        v.setPadding(systemBars.left, systemBars.top, systemBars.right, systemBars.bottom);
        return insets;
    });
}

private void addNotification() {
    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.O){
        String name = getString(R.string.channel_name);
        String description = getString(R.string.channel_description);
        int importance = NotificationManager.IMPORTANCE_HIGH;
        NotificationChannel channel = new NotificationChannel("1", name, importance);
        channel.setDescription(description);
        channel.setShowBadge(true);
        channel.setLockscreenVisibility(Notification.VISIBILITY_PUBLIC);
        NotificationManager notificationManager = getSystemService(NotificationManager.class);
        notificationManager.createNotificationChannel(channel);
    }

    Intent notificationIntent = new Intent(this, NotificationView.class);
    notificationIntent.setFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP |
```

```
        Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);


    //notification message will get at NotificationView

    notificationIntent.putExtra("message", "This is a notification content. Your account has been credited with xxxxxx amount. Enjoy!");


    PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, notificationIntent, PendingIntent.FLAG_IMMUTABLE |

        PendingIntent.FLAG_UPDATE_CURRENT);


    //notification messsage will get a reply button

    String replyLabel = getResources().getString(R.string.reply_label);

    RemoteInput remoteInput = new RemoteInput.Builder(KEY_TEXT_REPLY).setLabel(replyLabel).build();


    PendingIntent replyPendingIntent = PendingIntent.getBroadcast(getApplicationContext(), 1, notificationIntent, PendingIntent.FLAG_MUTABLE | PendingIntent.FLAG_UPDATE_CURRENT);


    NotificationCompat.Action action =

        new NotificationCompat.Action.Builder(R.drawable.baseline_reply_24,

            getString(R.string.label), replyPendingIntent)

            .addRemoteInput(remoteInput).build();


    NotificationCompat.Builder builder =

        new NotificationCompat.Builder(this, "1")
```

```
                .setSmallIcon(R.drawable.baseline_message_24)
                .setContentTitle(getString(R.string.channel_name))
                .setContentText(getText(R.string.channel_description))
                .setAutoCancel(true)
                .setContentIntent(pendingIntent)
                //.setDefaults(DEFAULT_SOUND | DEFAULT_VIBRATE)
                .setCategory(NotificationCompat.CATEGORY_MESSAGE)
                .setPriority(NotificationCompat.PRIORITY_MAX)
                .addAction(action);

        //Add as Notification
        NotificationManager manager = (NotificationManager) getSystemService
              (Context.NOTIFICATION_SERVICE);
        manager.notify(1, builder.build());
    }
}
```

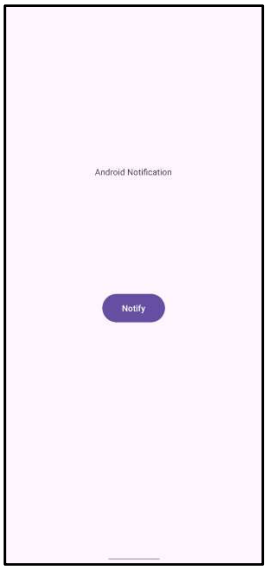
NotificationView.java:

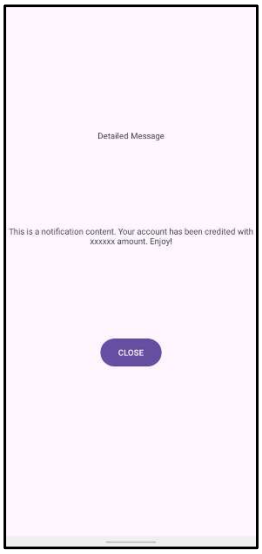
```
package com.example.notificationandalert;

import android.os.Bundle;
import android.widget.Button;
import android.widget.TextView;
```

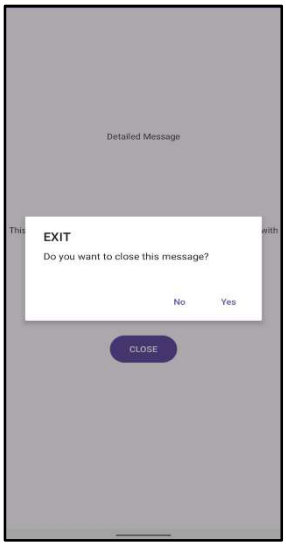
```
import android.widget.Toast;


import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;


public class NotificationView extends AppCompatActivity {
    TextView textView;
    Button alert;
    AlertDialog.Builder builder;


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


        textView = findViewById(R.id.textview);
        alert = findViewById(R.id.alt);


        // getting the notification message
        String message = getIntent().getStringExtra("message");
        textView.setText(message);


        // Initializing the AlertDialog.Builder
```

```
        builder = new AlertDialog.Builder(this);


        // Setting OnClickListener for the button
        alert.setOnClickListener(v -> {
            // Build and set up the dialog
            builder.setMessage("Do you want to close this message?")
                    .setCancelable(false)
                    .setPositiveButton("Yes", (dialog, id) -> {
                        finish(); // Closes the activity
                        Toast.makeText(getApplicationContext(), "You chose Yes!", Toast.LENGTH_SHORT).show();
                    })
                    .setNegativeButton("No", (dialog, id) -> {
                        dialog.cancel(); // Dismiss the dialog
                        Toast.makeText(getApplicationContext(), "You chose No!", Toast.LENGTH_SHORT).show();
                    });


            // Create and show the dialog when the button is clicked
            AlertDialog alertDialog = builder.create();
            alertDialog.setTitle("EXIT");
            alertDialog.show();
        });
    }
}
```

Output:







Conclusion – This program provides a basic understanding of creating notifications and alert boxes in Android apps. Building on this foundation, you can explore more advanced features like customizing notification layouts, adding progress bars, and handling user interaction with notifications.