SRM Institute of Science and Technology

Department of Computer Applications

Delhi – Meerut Road, Sikri Kalan, Ghaziabad, Uttar Pradesh – 201204

Circular – 2020-21

MCA GAI 2nd semester

Android Application Development Lab Manual

This lab manual provides a structured guide for various Android programming exercises. Each lab includes the aim, detailed procedure, source code, input requirements, and expected output to facilitate learning and practical implementation.

Lab 1: Login Page Creation with Toast Message

Title

Login Page Creation with Toast Message

Aim

To create a simple Android login page with username and password fields, a login button, and display a success or failure message using a Toast notification upon button click.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Click on "Start a new Android Studio project".
   - Select "Empty Activity" and click "Next".
   - Configure your project:
     - **Name:** `LoginPageApp`
     - **Package name:** `com.example.loginpageapp`
     - **Save location:** Choose a suitable directory.
     - **Language:** Java (or Kotlin, as preferred, but examples will be in Java).
     - **Minimum SDK:** API 21 (Android 5.0 Lollipop) or higher.
   - Click "Finish".
2. **Design the Layout (`activity_main.xml`):**
   - Open `app/src/main/res/layout/activity_main.xml`.
   - Switch to the "Code" view.
   - Add `EditText` for username, `EditText` for password, and a `Button` for login. Use `LinearLayout` or `ConstraintLayout` for arrangement.
3. **Implement Logic (`MainActivity.java`):**
   - Open `app/src/main/java/com/example/loginpageapp/MainActivity.java`.
   - Declare `EditText` and `Button` variables.
   - Initialize these variables by finding their respective views using `findViewById()`.
   - Set an `OnClickListener` for the login button.

    - Inside the `OnClickListener`, retrieve the text from the username and password fields.
    - Implement a simple check (e.g., username "admin" and password "password").
    - Display a `Toast` message indicating "Login Successful" or "Login Failed" based on the credentials.
4. **Run the Application:**
    - Connect an Android device or start an AVD (Android Virtual Device).
    - Click the "Run" button (green triangle) in Android Studio.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="16dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Login"
        android:textSize="32sp"
        android:textStyle="bold"
        android:layout_marginBottom="32dp"/>

    <EditText
        android:id="@+id/editTextUsername"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Username"
        android:inputType="text"
        android:padding="12dp"
        android:layout_marginBottom="16dp"
        android:background="@drawable/rounded_edittext_background"/>

    <EditText
        android:id="@+id/editTextPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Password"
        android:inputType="textPassword"
        android:padding="12dp"
        android:layout_marginBottom="24dp"
        android:background="@drawable/rounded_edittext_background"/>

    <Button
        android:id="@+id/buttonLogin"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Login"
        android:padding="12dp"
        android:backgroundTint="#6200EE"
        android:textColor="#FFFFFF"
        android:textSize="18sp"
        android:textStyle="bold"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml ***(Create this file)***

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

MainActivity.java

```
package com.example.loginpageapp;

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

public class MainActivity extends AppCompatActivity {

    private EditText editTextUsername;
    private EditText editTextPassword;
    private Button buttonLogin;

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

        // Initialize UI components
        editTextUsername = findViewById(R.id.editTextUsername);
        editTextPassword = findViewById(R.id.editTextPassword);
        buttonLogin = findViewById(R.id.buttonLogin);

        // Set OnClickListener for the login button
        buttonLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Get username and password from EditText fields
                String username = editTextUsername.getText().toString();
                String password = editTextPassword.getText().toString();

                // Simple validation check
                if (username.equals("admin") && password.equals("password"))
{
                    // Display success Toast message
                    Toast.makeText(MainActivity.this, "Login Successful!",
Toast.LENGTH_SHORT).show();
                } else {
                    // Display failure Toast message
                    Toast.makeText(MainActivity.this, "Login Failed. Invalid
credentials.", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Input

- **Username:** Type `admin`
- **Password:** Type `password`

Expected Output

- **Successful Login:** A short pop-up message (Toast) saying "Login Successful!" will appear at the bottom of the screen.
- **Failed Login:** A short pop-up message (Toast) saying "Login Failed. Invalid credentials." will appear at the bottom of the screen.

Lab 2: Student Registration Form with Toast Message

Title

Student Registration Form with Toast Message

Aim

To create an Android student registration form with fields for name, email, and a registration button, displaying a confirmation message using a Toast notification upon successful registration.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Click on "Start a new Android Studio project".
   - Select "Empty Activity" and click "Next".
   - Configure your project:
     - **Name:** `StudentRegistrationApp`
     - **Package name:** `com.example.studentregistrationapp`
     - **Save location:** Choose a suitable directory.
     - **Language:** Java.
     - **Minimum SDK:** API 21 or higher.
   - Click "Finish".
2. **Design the Layout (`activity_main.xml`):**
   - Open `app/src/main/res/layout/activity_main.xml`.
   - Add `EditText` fields for Name, Email, and a `Button` for registration. Use `LinearLayout` for arrangement.
3. **Implement Logic (`MainActivity.java`):**
   - Open `app/src/main/java/com/example/studentregistrationapp/MainActivity.java`.
   - Declare `EditText` and `Button` variables.
   - Initialize these variables using `findViewById()`.
   - Set an `OnClickListener` for the registration button.
   - Inside the `OnClickListener`, retrieve the text from the name and email fields.
   - Perform basic validation (e.g., check if fields are not empty).
   - Display a `Toast` message confirming the registration with the entered name, or an error message if fields are empty.
4. **Run the Application:**
   - Connect an Android device or start an AVD.
   - Click the "Run" button.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center_horizontal"
```

```xml
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Student Registration"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <EditText
        android:id="@+id/editTextName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Full Name"
        android:inputType="textPersonName"
        android:padding="14dp"
        android:layout_marginBottom="20dp"
        android:background="@drawable/rounded_edittext_background"/>

    <EditText
        android:id="@+id/editTextEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Email Address"
        android:inputType="textEmailAddress"
        android:padding="14dp"
        android:layout_marginBottom="30dp"
        android:background="@drawable/rounded_edittext_background"/>

    <Button
        android:id="@+id/buttonRegister"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Register"
        android:padding="14dp"
        android:backgroundTint="#007BFF"
        android:textColor="#FFFFFF"
        android:textSize="20sp"
        android:textStyle="bold"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml ***(Reuse or create if not present)***

```xml
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

MainActivity.java

```java
package com.example.studentregistrationapp;

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

public class MainActivity extends AppCompatActivity {

    private EditText editTextName;
    private EditText editTextEmail;
    private Button buttonRegister;

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

        // Initialize UI components
        editTextName = findViewById(R.id.editTextName);
        editTextEmail = findViewById(R.id.editTextEmail);
        buttonRegister = findViewById(R.id.buttonRegister);

        // Set OnClickListener for the register button
        buttonRegister.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Get name and email from EditText fields
                String name = editTextName.getText().toString().trim();
                String email = editTextEmail.getText().toString().trim();

                // Basic validation
                if (name.isEmpty() || email.isEmpty()) {
                    Toast.makeText(MainActivity.this, "Please fill in all
fields.", Toast.LENGTH_SHORT).show();
                } else {
                    // Display success Toast message
                    String message = "Registration successful for " + name +
" (" + email + ")";
                    Toast.makeText(MainActivity.this, message,
Toast.LENGTH_LONG).show();
                    // In a real app, you would save this data to a database
or send it to a server.
                }
            }
        });
    }
}
```

Input

- **Full Name:** Type `John Doe`
- **Email Address:** Type `john.doe@example.com`

Expected Output

- **Successful Registration:** A long pop-up message (Toast) saying "Registration successful for John Doe (john.doe@example.com)" will appear at the bottom of the screen.
- **Empty Fields:** A short pop-up message (Toast) saying "Please fill in all fields." will appear.

Lab 3: Implement Explicit Intent

Title

Implement Explicit Intent

Aim

To demonstrate the use of an Explicit Intent to navigate from one activity to another within the same application, passing data between them.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `ExplicitIntentApp`.
   - **Package name:** `com.example.explicitintentapp`
   - **Language:** Java.
2. **Create a Second Activity:**
   - Right-click on your package name (`com.example.explicitintentapp`) in the Project pane.
   - Select `New` -> `Activity` -> `Empty Activity`.
   - Name it `SecondActivity`.
   - Click "Finish". This will create `SecondActivity.java` and `activity_second.xml`.
3. **Design Layouts:**
   - **`activity_main.xml`:** Add an `EditText` for input and a `Button` to launch `SecondActivity`.
   - **`activity_second.xml`:** Add a `TextView` to display the data received from `MainActivity`.
4. **Implement Logic:**
   - **`MainActivity.java`:**
     - Get references to the `EditText` and `Button`.
     - Set an `OnClickListener` for the button.
     - Inside the listener, create an `Intent` object, specifying `MainActivity.this` as the context and `SecondActivity.class` as the target component.
     - Put the data from the `EditText` into the `Intent` using `putExtra()`.
     - Start the `SecondActivity` using `startActivity()`.
   - **`SecondActivity.java`:**
     - Get a reference to the `TextView`.
     - Retrieve the `Intent` that started this activity using `getIntent()`.
     - Extract the data from the `Intent` using `getStringExtra()`, using the same key used in `putExtra()`.
     - Set the retrieved text to the `TextView`.
5. **Run the Application:**
   - Run the app on an emulator or device.

Source Code

`activity_main.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Main Activity"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="32dp"/>

    <EditText
        android:id="@+id/editTextMessage"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter message to send"
        android:inputType="text"
        android:padding="12dp"
        android:layout_marginBottom="24dp"
        android:background="@drawable/rounded_edittext_background"/>

    <Button
        android:id="@+id/buttonSendMessage"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Go to Second Activity"
        android:padding="12dp"
        android:backgroundTint="#FF5722"
        android:textColor="#FFFFFF"
        android:textSize="18sp"
        android:textStyle="bold"/>

</LinearLayout>
```

activity_second.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".SecondActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Second Activity"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="32dp"/>

    <TextView
        android:id="@+id/textViewReceivedMessage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Message will appear here"
        android:textSize="20sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background"
        android:gravity="center"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml (Reuse or create)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

res/drawable/rounded_textview_background.xml (Create this file)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#E0F7FA"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#00BCD4" android:width="2dp"/>
</shape>
```

MainActivity.java

```
package com.example.explicitintentapp;

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

public class MainActivity extends AppCompatActivity {

    public static final String EXTRA_MESSAGE =
"com.example.explicitintentapp.MESSAGE"; // Define a constant for the key

    private EditText editTextMessage;
    private Button buttonSendMessage;

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

        editTextMessage = findViewById(R.id.editTextMessage);
        buttonSendMessage = findViewById(R.id.buttonSendMessage);

        buttonSendMessage.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String message = editTextMessage.getText().toString();

                // Create an explicit intent to start SecondActivity
                Intent intent = new Intent(MainActivity.this,
SecondActivity.class);

                // Put the message as extra data in the intent
                intent.putExtra(EXTRA_MESSAGE, message);

                // Start the SecondActivity
                startActivity(intent);
            }
        });
    }
}
```

SecondActivity.java

```
package com.example.explicitintentapp;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import android.widget.TextView;

public class SecondActivity extends AppCompatActivity {

    private TextView textViewReceivedMessage;

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

        textViewReceivedMessage = findViewById(R.id.textViewReceivedMessage);

        // Get the intent that started this activity
        Intent intent = getIntent();

        // Extract the message from the intent using the defined key
        if (intent != null && intent.hasExtra(MainActivity.EXTRA_MESSAGE)) {
            String message =
intent.getStringExtra(MainActivity.EXTRA_MESSAGE);
            textViewReceivedMessage.setText("Received: " + message);
        } else {
            textViewReceivedMessage.setText("No message received.");
        }
    }
}
```

Input

- In `MainActivity`, type any message into the `EditText` (e.g., "Hello from Main Activity!").
- Click the "Go to Second Activity" button.

Expected Output

- The application will navigate to `SecondActivity`.
- The `TextView` in `SecondActivity` will display the message you typed in `MainActivity` (e.g., "Received: Hello from Main Activity!").

Lab 4: Implement Implicit Intent

Title

Implement Implicit Intent

Aim

To demonstrate the use of an Implicit Intent to perform an action (e.g., open a web page, dial a number, send an email) by letting the Android system choose the appropriate application.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `ImplicitIntentApp`.
   - **Package name:** `com.example.implicitintentapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add a few `Button` widgets, each for a different implicit intent action (e.g., "Open Web Page", "Dial Phone", "Send Email").
3. **Implement Logic (`MainActivity.java`):**
   - Get references to the `Button` widgets.
   - Set `OnClickListener` for each button.
   - Inside each listener:
     - **Open Web Page:** Create an `Intent` with `ACTION_VIEW` and `Uri.parse("https://www.example.com")`.
     - **Dial Phone:** Create an `Intent` with `ACTION_DIAL` and `Uri.parse("tel:1234567890")`. (Note: `ACTION_CALL` requires permission and directly initiates a call, `ACTION_DIAL` opens the dialer).
     - **Send Email:** Create an `Intent` with `ACTION_SENDTO` and `Uri.parse("mailto:recipient@example.com")`. Add `putExtra` for subject and body.
     - Use `startActivity(intent)` to launch the appropriate app.
     - Wrap `startActivity` in a `try-catch` block or use `resolveActivity` to handle cases where no app can handle the intent.
4. **Add Permissions (if necessary):**
   - For `ACTION_CALL`, you would need `<uses-permission android:name="android.permission.CALL_PHONE" />` in `AndroidManifest.xml`. For `ACTION_DIAL`, it's not required.
5. **Run the Application:**
   - Run the app on an emulator or device.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
```

```xml
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Implicit Intent Examples"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <Button
        android:id="@+id/buttonOpenWeb"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Open Web Page"
        android:padding="12dp"
        android:layout_marginBottom="16dp"
        android:backgroundTint="#4CAF50"
        android:textColor="#FFFFFF"
        android:textSize="18sp"/>

    <Button
        android:id="@+id/buttonDialPhone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Dial Phone Number"
        android:padding="12dp"
        android:layout_marginBottom="16dp"
        android:backgroundTint="#2196F3"
        android:textColor="#FFFFFF"
        android:textSize="18sp"/>

    <Button
        android:id="@+id/buttonSendEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Send Email"
        android:padding="12dp"
        android:layout_marginBottom="16dp"
        android:backgroundTint="#FFC107"
        android:textColor="#333333"
        android:textSize="18sp"/>

    <Button
        android:id="@+id/buttonViewMap"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="View Location on Map"
        android:padding="12dp"
        android:backgroundTint="#9C27B0"
        android:textColor="#FFFFFF"
        android:textSize="18sp"/>

</LinearLayout>
```

MainActivity.java

```java
package com.example.implicitintentapp;

import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Toast;
```

```
public class MainActivity extends AppCompatActivity {

    private Button buttonOpenWeb;
    private Button buttonDialPhone;
    private Button buttonSendEmail;
    private Button buttonViewMap;

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

        buttonOpenWeb = findViewById(R.id.buttonOpenWeb);
        buttonDialPhone = findViewById(R.id.buttonDialPhone);
        buttonSendEmail = findViewById(R.id.buttonSendEmail);
        buttonViewMap = findViewById(R.id.buttonViewMap);

        // Open Web Page
        buttonOpenWeb.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String url = "https://www.google.com"; // Example URL
                Intent intent = new Intent(Intent.ACTION_VIEW);
                intent.setData(Uri.parse(url));
                if (intent.resolveActivity(getPackageManager()) != null) {
                    startActivity(intent);
                } else {
                    Toast.makeText(MainActivity.this, "No application can 
handle this request.", Toast.LENGTH_SHORT).show();
                }
            }
        });

        // Dial Phone Number
        buttonDialPhone.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String phoneNumber = "tel:1234567890"; // Example phone 
number
                Intent intent = new Intent(Intent.ACTION_DIAL);
                intent.setData(Uri.parse(phoneNumber));
                if (intent.resolveActivity(getPackageManager()) != null) {
                    startActivity(intent);
                } else {
                    Toast.makeText(MainActivity.this, "No application can 
handle this request.", Toast.LENGTH_SHORT).show();
                }
            }
        });

        // Send Email
        buttonSendEmail.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String recipient = "test@example.com";
                String subject = "Regarding your app";
                String body = "Hello, I have a question about...";

                Intent intent = new Intent(Intent.ACTION_SENDTO);
                intent.setData(Uri.parse("mailto:")); // Only email apps 
should handle this
                intent.putExtra(Intent.EXTRA_EMAIL, new String[]{recipient});
                intent.putExtra(Intent.EXTRA_SUBJECT, subject);
                intent.putExtra(Intent.EXTRA_TEXT, body);

                if (intent.resolveActivity(getPackageManager()) != null) {
                    startActivity(Intent.createChooser(intent, "Send Email
Using...")); // Use createChooser for better user experience
                } else {
                    Toast.makeText(MainActivity.this, "No email client
found.", Toast.LENGTH_SHORT).show();
                }
            }
        });

        // View Location on Map
        buttonViewMap.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Example: Latitude, Longitude (e.g., Googleplex)
                String geoUri = "geo:37.4220,-122.0841?z=16"; // z is zoom
level
                Intent intent = new Intent(Intent.ACTION_VIEW,
Uri.parse(geoUri));
                if (intent.resolveActivity(getPackageManager()) != null) {
                    startActivity(intent);
                } else {
                    Toast.makeText(MainActivity.this, "No map application
found.", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

Input

- Click on any of the buttons: "Open Web Page", "Dial Phone Number", "Send Email", or "View Location on Map".

Expected Output

- **Open Web Page:** The device's default web browser will open to `https://www.google.com`.
- **Dial Phone Number:** The device's dialer application will open with "1234567890" pre-filled.
- **Send Email:** A list of email client applications will appear (if multiple are installed), allowing the user to choose one to send an email to `test@example.com` with the specified subject and body.
- **View Location on Map:** The device's map application will open, centered on the specified geographical coordinates.
- If no application can handle a specific intent, a Toast message "No application can handle this request." (or similar) will appear.

Lab 5: Implement Time Picker

Title

Implement Time Picker

Aim

To implement an Android `TimePicker` dialog that allows users to select a time, and then display the selected time in a `TextView`.

Procedure

1. **Create a New Android Project:**
    - Open Android Studio.
    - Create a new "Empty Activity" project named `TimePickerApp`.
    - **Package name:** `com.example.timepickerapp`
    - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
    - Add a `Button` to trigger the `TimePicker` dialog.
    - Add a `TextView` to display the selected time.
3. **Implement Logic (`MainActivity.java`):**
    - Get references to the `Button` and `TextView`.
    - Set an `OnClickListener` for the button.
    - Inside the listener, create an instance of `TimePickerDialog`.
    - Pass the current hour and minute to the `TimePickerDialog` constructor to set the initial time.
    - Implement `TimePickerDialog.OnTimeSetListener` to handle the selected time. This listener's `onTimeSet()` method will be called when the user selects a time and clicks "OK".
    - In `onTimeSet()`, format the selected hour and minute (e.g., add leading zeros for single digits, handle AM/PM).
    - Set the formatted time to the `TextView`.
    - Call `timePickerDialog.show()` to display the dialog.
4. **Run the Application:**
    - Run the app on an emulator or device.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Time Picker Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <TextView
        android:id="@+id/textViewSelectedTime"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="No time selected"
        android:textSize="24sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background"
        android:layout_marginBottom="30dp"/>

    <Button
        android:id="@+id/buttonPickTime"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Pick Time"
        android:padding="12dp"
        android:backgroundTint="#FF9800"
        android:textColor="#FFFFFF"
        android:textSize="18sp"/>

</LinearLayout>
```

res/drawable/rounded_textview_background.xml *(Reuse or create)*

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#FFF3E0"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#FF9800" android:width="2dp"/>
</shape>
```

MainActivity.java

```
package com.example.timepickerapp;

import androidx.appcompat.app.AppCompatActivity;
import android.app.TimePickerDialog;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.TimePicker;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {

    private TextView textViewSelectedTime;
    private Button buttonPickTime;

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

        textViewSelectedTime = findViewById(R.id.textViewSelectedTime);
        buttonPickTime = findViewById(R.id.buttonPickTime);

        buttonPickTime.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                showTimePickerDialog();
            }
        });
    }

    private void showTimePickerDialog() {
        // Get current time to set as default in the picker
        final Calendar c = Calendar.getInstance();
        int hour = c.get(Calendar.HOUR_OF_DAY);
        int minute = c.get(Calendar.MINUTE);

        // Create a new instance of TimePickerDialog
        TimePickerDialog timePickerDialog = new TimePickerDialog(this,
                new TimePickerDialog.OnTimeSetListener() {
                    @Override
                    public void onTimeSet(TimePicker view, int hourOfDay, int
minute) {
                        // Format the selected time
                        String formattedTime = String.format("%02d:%02d",
hourOfDay, minute);
                        textViewSelectedTime.setText("Selected Time: " +
formattedTime);
                    }
                },
                hour, // Initial hour
                minute, // Initial minute
                false); // True for 24-hour format, false for AM/PM format

        timePickerDialog.show();
    }
}
```

Input

- Click the "Pick Time" button.
- In the `TimePicker` dialog, select an hour and minute using the clock or input fields.
- Click "OK".

Expected Output

- The `TimePicker` dialog will appear, allowing you to select a time.
- After selecting a time and clicking "OK", the `TextView` will update to display the selected time (e.g., "Selected Time: 10:30" or "Selected Time: 22:15").

Lab 6: Implement Date Picker

Title

Implement Date Picker

Aim

To implement an Android `DatePicker` dialog that allows users to select a date, and then display the selected date in a `TextView`.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `DatePickerApp`.
   - **Package name:** `com.example.datepickerapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add a `Button` to trigger the `DatePicker` dialog.
   - Add a `TextView` to display the selected date.
3. **Implement Logic (`MainActivity.java`):
   - Get references to the `Button` and `TextView`.
   - Set an `OnClickListener` for the button.
   - Inside the listener, create an instance of `DatePickerDialog`.
   - Pass the current year, month, and day to the `DatePickerDialog` constructor to set the initial date.
   - Implement `DatePickerDialog.OnDateSetListener` to handle the selected date. This listener's `onDateSet()` method will be called when the user selects a date and clicks "OK".
   - In `onDateSet()`, format the selected date (e.g., "DD/MM/YYYY").
   - Set the formatted date to the `TextView`.
   - Call `datePickerDialog.show()` to display the dialog.
4. **Run the Application:**
   - Run the app on an emulator or device.

Source Code

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Date Picker Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <TextView
        android:id="@+id/textViewSelectedDate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="No date selected"
        android:textSize="24sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background"
        android:layout_marginBottom="30dp"/>

    <Button
        android:id="@+id/buttonPickDate"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Pick Date"
        android:padding="12dp"
        android:backgroundTint="#03A9F4"
        android:textColor="#FFFFFF"
        android:textSize="18sp"/>

</LinearLayout>
```

res/drawable/rounded_textview_background.xml *(Reuse or create)*

```xml
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#E1F5FE"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#03A9F4" android:width="2dp"/>
</shape>
```

MainActivity.java

```java
package com.example.datepickerapp;

import androidx.appcompat.app.AppCompatActivity;
import android.app.DatePickerDialog;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.TextView;
import java.util.Calendar;

public class MainActivity extends AppCompatActivity {

    private TextView textViewSelectedDate;
    private Button buttonPickDate;

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

        textViewSelectedDate = findViewById(R.id.textViewSelectedDate);
        buttonPickDate = findViewById(R.id.buttonPickDate);

        buttonPickDate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                showDatePickerDialog();
            }
        });
    }
    private void showDatePickerDialog() {
        // Get current date to set as default in the picker
        final Calendar c = Calendar.getInstance();
        int year = c.get(Calendar.YEAR);
        int month = c.get(Calendar.MONTH); // Month is 0-indexed
        int day = c.get(Calendar.DAY_OF_MONTH);

        // Create a new instance of DatePickerDialog
        DatePickerDialog datePickerDialog = new DatePickerDialog(this,
                new DatePickerDialog.OnDateSetListener() {
                    @Override
                    public void onDateSet(DatePicker view, int year, int
monthOfYear, int dayOfMonth) {
                        // Format the selected date (monthOfYear is 0-
indexed, so add 1)
                        String formattedDate = String.format("%02d/%02d/%d",
dayOfMonth, (monthOfYear + 1), year);
                        textViewSelectedDate.setText("Selected Date: " +
formattedDate);
                    }
                },
                year, // Initial year
                month, // Initial month
                day); // Initial day

        datePickerDialog.show();
    }
}
```

Input

- Click the "Pick Date" button.
- In the `DatePicker` dialog, select a year, month, and day.
- Click "OK".

Expected Output

- The `DatePicker` dialog will appear, allowing you to select a date.
- After selecting a date and clicking "OK", the `TextView` will update to display the selected date (e.g., "Selected Date: 23/05/2025").

Lab 7: Student Registration Form using Basic and List View

Title

Student Registration Form using Basic and List View

Aim

To create a student registration form that collects student details (Name, Roll No, Course) and displays the registered students in a `ListView`.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `StudentListApp`.
   - **Package name:** `com.example.studentlistapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add `EditText` fields for Name, Roll No, and Course.
   - Add a `Button` to "Add Student".
   - Add a `ListView` to display the registered students.
3. **Implement Logic (`MainActivity.java`):**
   - Declare `EditText` fields, `Button`, and `ListView`.
   - Initialize UI components.
   - Create an `ArrayList<String>` to hold student data (e.g., "Name: [name], Roll No: [roll], Course: [course]").
   - Create an `ArrayAdapter<String>` to bridge the `ArrayList` data to the `ListView`. Use `android.R.layout.simple_list_item_1` for a basic list item layout.
   - Set the adapter to the `ListView`.
   - Set an `OnClickListener` for the "Add Student" button:
     - Retrieve text from `EditText` fields.
     - Perform basic validation (e.g., check for empty fields).
     - If valid, construct a string with student details.
     - Add this string to the `ArrayList`.
     - Notify the adapter that the data set has changed using `adapter.notifyDataSetChanged()`.
     - Clear the `EditText` fields.
     - Display a Toast message.
4. **Run the Application:**
   - Run the app on an emulator or device.

Source Code

activity_main.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:padding="24dp"
    tools:context=".MainActivity">
```

```xml
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Student Registration"
    android:textSize="28sp"
    android:textStyle="bold"
    android:layout_gravity="center_horizontal"
    android:layout_marginBottom="30dp"/>

<EditText
    android:id="@+id/editTextStudentName"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Student Name"
    android:inputType="textPersonName"
    android:padding="12dp"
    android:layout_marginBottom="12dp"
    android:background="@drawable/rounded_edittext_background"/>

<EditText
    android:id="@+id/editTextRollNo"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Roll Number"
    android:inputType="number"
    android:padding="12dp"
    android:layout_marginBottom="12dp"
    android:background="@drawable/rounded_edittext_background"/>

<EditText
    android:id="@+id/editTextCourse"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:hint="Course"
    android:inputType="text"
    android:padding="12dp"
    android:layout_marginBottom="20dp"
    android:background="@drawable/rounded_edittext_background"/>

<Button
    android:id="@+id/buttonAddStudent"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:text="Add Student"
    android:padding="12dp"
    android:layout_marginBottom="30dp"
    android:backgroundTint="#673AB7"
    android:textColor="#FFFFFF"
    android:textSize="18sp"/>

<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Registered Students:"
    android:textSize="20sp"
    android:textStyle="bold"
    android:layout_marginBottom="10dp"/>

<ListView
    android:id="@+id/listViewStudents"
    android:layout_width="match_parent"
    android:layout_height="0dp"
    android:layout_weight="1"
    android:background="@drawable/rounded_listview_background"
    android:padding="8dp"/>
```

```
</LinearLayout>
```

res/drawable/rounded_edittext_background.xml *(Reuse or create)*

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

res/drawable/rounded_listview_background.xml *(Create this file)*

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#E8EAF6"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#7986CB" android:width="2dp"/>
</shape>
```

MainActivity.java

```
package com.example.studentlistapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.Toast;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity {

    private EditText editTextStudentName;
    private EditText editTextRollNo;
    private EditText editTextCourse;
    private Button buttonAddStudent;
    private ListView listViewStudents;

    private ArrayList<String> studentList;
    private ArrayAdapter<String> adapter;

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

        // Initialize UI components
        editTextStudentName = findViewById(R.id.editTextStudentName);
        editTextRollNo = findViewById(R.id.editTextRollNo);
        editTextCourse = findViewById(R.id.editTextCourse);
        buttonAddStudent = findViewById(R.id.buttonAddStudent);
        listViewStudents = findViewById(R.id.listViewStudents);

        // Initialize ArrayList and ArrayAdapter
        studentList = new ArrayList<>();
        adapter = new ArrayAdapter<>(this,
android.R.layout.simple_list_item_1, studentList);
        listViewStudents.setAdapter(adapter);

        // Set OnClickListener for the Add Student button
        buttonAddStudent.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                addStudent();
            }
        });
    }

    private void addStudent() {
        String name = editTextStudentName.getText().toString().trim();
        String rollNo = editTextRollNo.getText().toString().trim();
        String course = editTextCourse.getText().toString().trim();

        if (name.isEmpty() || rollNo.isEmpty() || course.isEmpty()) {
            Toast.makeText(this, "Please fill in all student details.",
Toast.LENGTH_SHORT).show();
            return;
        }

        // Construct student data string
        String studentData = "Name: " + name + "\nRoll No: " + rollNo +
"\nCourse: " + course;

        // Add to list and notify adapter
        studentList.add(studentData);
        adapter.notifyDataSetChanged();

        // Clear input fields
        editTextStudentName.setText("");
        editTextRollNo.setText("");
        editTextCourse.setText("");

        Toast.makeText(this, "Student added successfully!",
Toast.LENGTH_SHORT).show();
    }
}
```

Input

- Enter student details in the "Student Name", "Roll Number", and "Course" fields.
- Click the "Add Student" button.
- Repeat for multiple students.

Expected Output

- Each time you click "Add Student" with valid input, a Toast message "Student added successfully!" will appear.
- The `ListView` below the form will update to display the details of the newly added student as a new item in the list. Each item will show "Name: [name]\nRoll No: [roll]\nCourse: [course]".

Lab 8: Implement Context Menu

Title

Implement Context Menu

Aim

To implement a Context Menu (floating contextual menu) that appears when a user long-presses on a specific UI element (e.g., a `TextView` or an item in a `ListView`), providing actions related to that element.

Procedure

1. **Create a New Android Project:**
    - Open Android Studio.
    - Create a new "Empty Activity" project named `ContextMenuApp`.
    - **Package name:** `com.example.contextmenuapp`
    - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
    - Add a `TextView` or a `ListView` (for a more practical example) that will trigger the context menu. For simplicity, we'll use a `TextView`.
3. **Create the Menu Resource (`res/menu/context_menu.xml`):**
    - Right-click on `res` -> `New` -> `Android Resource Directory`.
    - Select "menu" as the Resource type. Click "OK".
    - Right-click on the newly created `menu` directory -> `New` -> `Menu resource file`.
    - Name it `context_menu`.
    - Add `item` tags for the desired menu options (e.g., "Edit", "Delete", "Share").
4. **Implement Logic (`MainActivity.java`):**
    - Get a reference to the `TextView`.
    - **Register the view for context menu:** Call `registerForContextMenu(textView)`.
    - **Override `onCreateContextMenu()`:** This method is called when the registered view is long-pressed. Inflate your `context_menu.xml` here.
    - **Override `onContextItemSelected()`:** This method is called when an item from the context menu is selected. Use `item.getItemId()` to identify which menu item was clicked and perform the corresponding action (e.g., display a Toast message for each action).
5. **Run the Application:**
    - Run the app on an emulator or device.
    - Long-press on the `TextView`.

Source Code

`activity_main.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Context Menu Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <TextView
        android:id="@+id/textViewContext"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Long press me for Context Menu"
        android:textSize="22sp"
        android:textColor="#333333"
        android:padding="20dp"
        android:background="@drawable/rounded_textview_background_context"
        android:gravity="center"
        android:clickable="true"
        android:focusable="true"/>

</LinearLayout>
```

res/drawable/rounded_textview_background_context.xml ***(Create this file)***

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F3E5F5"/>
    <corners android:radius="16dp"/>
    <stroke android:color="#9C27B0" android:width="2dp"/>
</shape>
```

res/menu/context_menu.xml ***(Create this file)***

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/menu_edit"
        android:title="Edit" />
    <item
        android:id="@+id/menu_delete"
        android:title="Delete" />
    <item
        android:id="@+id/menu_share"
        android:title="Share" />
</menu>
```

MainActivity.java

```
package com.example.contextmenuapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.ContextMenu;
import android.view.MenuItem;
import android.view.View;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private TextView textViewContext;

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

        textViewContext = findViewById(R.id.textViewContext);

        // Register the TextView for a context menu
        registerForContextMenu(textViewContext);
    }

    // Called when the context menu is being built
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenu.ContextMenuInfo menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        // Inflate the menu resource
        getMenuInflater().inflate(R.menu.context_menu, menu);
        menu.setHeaderTitle("Choose an action"); // Set a header for the menu
    }

    // Called when a context menu item is selected
    @Override
    public boolean onContextItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.menu_edit:
                Toast.makeText(this, "Edit selected",
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.menu_delete:
                Toast.makeText(this, "Delete selected",
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.menu_share:
                Toast.makeText(this, "Share selected",
Toast.LENGTH_SHORT).show();
                return true;
            default:
                return super.onContextItemSelected(item);
        }
    }
}
```

Input

- Long-press on the "Long press me for Context Menu" `TextView`.
- Select one of the options from the context menu (Edit, Delete, Share).

Expected Output

- Upon long-pressing the `TextView`, a floating context menu will appear with "Edit", "Delete", and "Share" options.
- When you tap on an option (e.g., "Edit"), a Toast message indicating the selected action (e.g., "Edit selected") will appear.

Lab 9: Implement Option Menu

Title

Implement Option Menu

Aim

To implement an Option Menu (also known as the Action Bar menu or Overflow menu) that provides a set of general actions for the current activity, typically displayed at the top right of the screen.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `OptionMenuApp`.
   - **Package name:** `com.example.optionmenuapp`
   - **Language:** Java.
2. **Create the Menu Resource (`res/menu/option_menu.xml`):**
   - Right-click on `res/menu` directory -> `New` -> `Menu resource file`.
   - Name it `option_menu`.
   - Add `item` tags for your menu options (e.g., "Settings", "About", "Exit").
   - Use `app:showAsAction="ifRoom|withText"` or `always` to make items appear directly in the Action Bar, or `never` to place them in the overflow menu. Remember to add `xmlns:app="http://schemas.android.com/apk/res-auto"` to the `<menu>` tag if using `app:` attributes.
3. **Implement Logic (`MainActivity.java`):**
   - **Override `onCreateOptionsMenu()`:** This method is called when the options menu is first created. Inflate your `option_menu.xml` here.
   - **Override `onOptionsItemSelected()`:** This method is called when an item from the options menu is selected. Use `item.getItemId()` to identify which menu item was clicked and perform the corresponding action (e.g., display a Toast message).
4. **Run the Application:**
   - Run the app on an emulator or device.
   - Look for the three-dot icon (overflow menu) or direct icons in the Action Bar.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Option Menu Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tap the three dots (overflow icon) in the top right
corner to see the Option Menu."
        android:textSize="18sp"
        android:textColor="#555555"
        android:gravity="center"
        android:padding="16dp"/>

</LinearLayout>
```

res/menu/option_menu.xml (Create this file)

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/menu_settings"
        android:title="Settings"
        app:showAsAction="never" /> <item
        android:id="@+id/menu_about"
        android:title="About"
        app:showAsAction="never" /> <item
        android:id="@+id/menu_exit"
        android:title="Exit"
        app:showAsAction="never" /> <item
        android:id="@+id/menu_search"
        android:title="Search"
        android:icon="@android:drawable/ic_menu_search"
        app:showAsAction="ifRoom" /> </menu>
```

MainActivity.java

```
package com.example.optionmenuapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

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

    // Called to create the options menu
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is
present.
        getMenuInflater().inflate(R.menu.option_menu, menu);
        return true;
    }

    // Called when an item in the options menu is selected
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        switch (item.getItemId()) {
            case R.id.menu_settings:
                Toast.makeText(this, "Settings selected",
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.menu_about:
                Toast.makeText(this, "About selected",
Toast.LENGTH_SHORT).show();
                return true;
            case R.id.menu_exit:
                Toast.makeText(this, "Exit selected",
Toast.LENGTH_SHORT).show();
                finish(); // Close the activity
                return true;
            case R.id.menu_search:
                Toast.makeText(this, "Search selected",
Toast.LENGTH_SHORT).show();
                // Implement search functionality here
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

Input

- Tap the three-dot icon (overflow menu) in the top right corner of the screen (or the search icon if it appears directly).
- Select one of the options (e.g., "Settings", "About", "Exit", "Search").

Expected Output

- The options menu will appear.
- When you tap on an option (e.g., "Settings"), a Toast message indicating the selected action (e.g., "Settings selected") will appear.
- If "Exit" is selected, the application will close.

Lab 10: Shared Preferences

Title

Shared Preferences

Aim

To demonstrate how to use `SharedPreferences` to store and retrieve simple key-value pair data (e.g., user settings, last logged-in username) persistently within an Android application.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `SharedPreferencesApp`.
   - **Package name:** `com.example.sharedpreferencesapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add an `EditText` for user input (e.g., "Enter your name").
   - Add a `Button` to "Save Data".
   - Add a `Button` to "Load Data".
   - Add a `TextView` to display the loaded data.
3. **Implement Logic (`MainActivity.java`):**
   - Declare `EditText`, `Button`s, and `TextView`.
   - Initialize UI components.
   - **Saving Data:**
     - Set an `OnClickListener` for the "Save Data" button.
     - Get an instance of `SharedPreferences` using `getSharedPreferences("MyPrefs", MODE_PRIVATE)` or `PreferenceManager.getDefaultSharedPreferences(this)`.
     - Get an `Editor` object from `SharedPreferences`.
     - Use `editor.putString("key", value)` (or `putInt`, `putBoolean`, etc.) to store data.
     - Call `editor.apply()` to save the changes asynchronously (preferred) or `editor.commit()` synchronously.
     - Display a Toast message.
   - **Loading Data:**
     - Set an `OnClickListener` for the "Load Data" button.
     - Get an instance of `SharedPreferences` (same as saving).
     - Use `sharedPreferences.getString("key", defaultValue)` to retrieve data.
     - Set the retrieved data to the `TextView`.
     - Display a Toast message.
   - **Initial Load (Optional but good practice):** Load data when the activity is created (in `onCreate`) to display any previously saved data.
4. **Run the Application:**
   - Run the app on an emulator or device.
   - Enter text, save, close the app, and reopen to see if data persists.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Shared Preferences Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <EditText
        android:id="@+id/editTextData"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter text to save"
        android:inputType="text"
        android:padding="12dp"
        android:layout_marginBottom="20dp"
        android:background="@drawable/rounded_edittext_background"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_marginBottom="30dp">

        <Button
            android:id="@+id/buttonSave"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Save Data"
            android:padding="12dp"
            android:layout_marginEnd="8dp"
            android:backgroundTint="#FF5722"
            android:textColor="#FFFFFF"
            android:textSize="18sp"/>

        <Button
            android:id="@+id/buttonLoad"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Load Data"
            android:padding="12dp"
            android:layout_marginStart="8dp"
            android:backgroundTint="#4CAF50"
            android:textColor="#FFFFFF"
            android:textSize="18sp"/>
    </LinearLayout>

    <TextView
        android:id="@+id/textViewLoadedData"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Loaded Data: (None)"
        android:textSize="22sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background_sp"
        android:gravity="center"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml (Reuse or create)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

res/drawable/rounded_textview_background_sp.xml (Create this file)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#E0F2F7"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#00BCD4" android:width="2dp"/>
</shape>
```

MainActivity.java

```
package com.example.sharedpreferencesapp;

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

public class MainActivity extends AppCompatActivity {

    private EditText editTextData;
    private Button buttonSave;
    private Button buttonLoad;
    private TextView textViewLoadedData;

    private static final String PREF_NAME = "MySharedPrefs";
    private static final String KEY_DATA = "my_data_key";

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

        editTextData = findViewById(R.id.editTextData);
        buttonSave = findViewById(R.id.buttonSave);
        buttonLoad = findViewById(R.id.buttonLoad);
        textViewLoadedData = findViewById(R.id.textViewLoadedData);

        // Load data when the activity is created (if any exists)
        loadData();

        buttonSave.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                saveData();
            }
        });

        buttonLoad.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                loadData();
            }
        });
    }

    private void saveData() {
        String dataToSave = editTextData.getText().toString();

        // Get a SharedPreferences instance
        SharedPreferences sharedPreferences = getSharedPreferences(PREF_NAME,
MODE_PRIVATE);

        // Get an editor to put data
        SharedPreferences.Editor editor = sharedPreferences.edit();

        // Put the string data with a key
        editor.putString(KEY_DATA, dataToSave);

        // Apply the changes asynchronously
        editor.apply();

        Toast.makeText(this, "Data saved!", Toast.LENGTH_SHORT).show();
    }

    private void loadData() {
        // Get a SharedPreferences instance
        SharedPreferences sharedPreferences = getSharedPreferences(PREF_NAME,
MODE_PRIVATE);

        // Retrieve the string data using the key, provide a default value if
not found
        String loadedData = sharedPreferences.getString(KEY_DATA, "No data
found.");

        // Set the loaded data to the TextView
        textViewLoadedData.setText("Loaded Data: " + loadedData);

        Toast.makeText(this, "Data loaded!", Toast.LENGTH_SHORT).show();
    }
}
```

Input

- Type some text into the `EditText` (e.g., "This is my saved text.").
- Click the "Save Data" button.
- (Optional) Close the app and reopen it.
- Click the "Load Data" button.

Expected Output

- After clicking "Save Data", a Toast "Data saved!" will appear.
- After clicking "Load Data", a Toast "Data loaded!" will appear, and the `TextView` will display "Loaded Data: This is my saved text." (or whatever text you saved).

- If you close the app and reopen it, the previously saved data should automatically appear in the `TextView` (due to `loadData()` in `onCreate()`) or when you click "Load Data".

Lab 11: Storing Data to File in Internal Storage

Title

Storing Data to File in Internal Storage

Aim

To demonstrate how to store and retrieve text data to/from a private file in the application's internal storage, which is accessible only by the app itself.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `InternalStorageApp`.
   - **Package name:** `com.example.internalstorageapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add an `EditText` for user input.
   - Add a `Button` to "Save to File".
   - Add a `Button` to "Load from File".
   - Add a `TextView` to display the loaded data.
3. **Implement Logic (`MainActivity.java`):**
   - Declare UI components.
   - Initialize UI components.
   - Define a `FILENAME` constant for your file.
   - **Saving Data:**
     - Set an `OnClickListener` for the "Save to File" button.
     - Get the text from the `EditText`.
     - Use `openFileOutput(FILENAME, MODE_PRIVATE)` to get a `FileOutputStream`.
     - Wrap it in an `OutputStreamWriter` and `BufferedWriter` for efficient text writing.
     - Write the data using `writer.write(data)`.
     - Close the writer.
     - Handle `IOException` with a `try-catch` block.
     - Display a Toast message.
   - **Loading Data:**
     - Set an `OnClickListener` for the "Load from File" button.
     - Use `openFileInput(FILENAME)` to get a `FileInputStream`.
     - Wrap it in an `InputStreamReader` and `BufferedReader` for efficient text reading.
     - Read lines using `reader.readLine()` until null.
     - Append lines to a `StringBuilder`.
     - Close the reader.
     - Handle `IOException` and `FileNotFoundException`.
     - Set the retrieved data to the `TextView`.
     - Display a Toast message.
4. **Run the Application:**
   - Run the app on an emulator or device.
   - Enter text, save, close the app, and reopen to see if data persists.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Internal Storage Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginBottom="40dp"/>

    <EditText
        android:id="@+id/editTextFileContent"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Enter text to save to file"
        android:inputType="textMultiLine"
        android:lines="4"
        android:gravity="top"
        android:padding="12dp"
        android:layout_marginBottom="20dp"
        android:background="@drawable/rounded_edittext_background"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_marginBottom="30dp">

        <Button
            android:id="@+id/buttonSaveFile"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Save to File"
            android:padding="12dp"
            android:layout_marginEnd="8dp"
            android:backgroundTint="#FF9800"
            android:textColor="#FFFFFF"
            android:textSize="18sp"/>

        <Button
            android:id="@+id/buttonLoadFile"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Load from File"
            android:padding="12dp"
            android:layout_marginStart="8dp"
            android:backgroundTint="#607D8B"
            android:textColor="#FFFFFF"
            android:textSize="18sp"/>
    </LinearLayout>
    <TextView
        android:id="@+id/textViewFileContent"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="File Content: (None)"
        android:textSize="20sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background_file"
        android:gravity="center_vertical"
        android:minHeight="100dp"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml (Reuse or create)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

res/drawable/rounded_textview_background_file.xml (Create this file)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#ECEFF1"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#607D8B" android:width="2dp"/>
</shape>
```

MainActivity.java

```
package com.example.internalstorageapp;

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

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;

public class MainActivity extends AppCompatActivity {

    private EditText editTextFileContent;
    private Button buttonSaveFile;
    private Button buttonLoadFile;
    private TextView textViewFileContent;

    private static final String FILENAME = "my_internal_file.txt";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        editTextFileContent = findViewById(R.id.editTextFileContent);
        buttonSaveFile = findViewById(R.id.buttonSaveFile);
        buttonLoadFile = findViewById(R.id.buttonLoadFile);
        textViewFileContent = findViewById(R.id.textViewFileContent);

        buttonSaveFile.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                saveToFile();
            }
        });

        buttonLoadFile.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                loadFromFile();
            }
        });

        // Optional: Load content on app start if file exists
        loadFromFile();
    }

    private void saveToFile() {
        String data = editTextFileContent.getText().toString();
        FileOutputStream fos = null;
        try {
            // Open a private file output stream
            fos = openFileOutput(FILENAME, MODE_PRIVATE);
            // Write data to the file
            fos.write(data.getBytes());
            Toast.makeText(this, "Saved to " + getFilesDir() + "/" +
FILENAME, Toast.LENGTH_LONG).show();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
            Toast.makeText(this, "File not found: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            e.printStackTrace();
            Toast.makeText(this, "Error saving file: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        } finally {
            if (fos != null) {
                try {
                    fos.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }

    private void loadFromFile() {
        FileInputStream fis = null;
        try {
            // Open a private file input stream
            fis = openFileInput(FILENAME);
            InputStreamReader isr = new InputStreamReader(fis);
            BufferedReader br = new BufferedReader(isr);
            StringBuilder sb = new StringBuilder();
            String line;
            while ((line = br.readLine()) != null) {
                sb.append(line).append("\n"); // Append each line and a
newline
            }
            textViewFileContent.setText("File Content:\n" +
sb.toString().trim()); // Trim to remove trailing newline
            Toast.makeText(this, "Loaded from " + getFilesDir() + "/" +
FILENAME, Toast.LENGTH_LONG).show();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
            textViewFileContent.setText("File Content: (No data saved yet)");
            Toast.makeText(this, "File not found. Save some data first.",
Toast.LENGTH_SHORT).show();
        } catch (IOException e) {
            e.printStackTrace();
            Toast.makeText(this, "Error loading file: " + e.getMessage(),
Toast.LENGTH_SHORT).show();
        } finally {
            if (fis != null) {
                try {
                    fis.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        }
    }
}
```

Input

- Type some multi-line text into the `EditText` (e.g., "Line 1\nLine 2\nLine 3").
- Click the "Save to File" button.
- (Optional) Close the app and reopen it.
- Click the "Load from File" button.

Expected Output

- After clicking "Save to File", a Toast message indicating successful saving (e.g., "Saved to /data/user/0/com.example.internalstorageapp/files/my_internal_file.txt") will appear.
- After clicking "Load from File", a Toast message indicating successful loading will appear, and the `TextView` will display "File Content:\nLine 1\nLine 2\nLine 3" (or whatever text you saved).
- If you close the app and reopen it, the previously saved data should automatically appear in the `TextView` (due to `loadFromFile()` in `onCreate()`) or when you click "Load from File".

Lab 12: SQLite Database

Title

SQLite Database

Aim

To implement a simple Android application that uses an SQLite database to store, retrieve, update, and delete student records (ID, Name, Age).

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `SQLiteDatabaseApp`.
   - **Package name:** `com.example.sqlitedatabaseapp`
   - **Language:** Java.
2. **Design the Layout (`activity_main.xml`):**
   - Add `EditText` fields for Student Name and Age.
   - Add `EditText` for Student ID (for update/delete operations).
   - Add `Button`s for "Add", "View All", "Update", and "Delete".
   - Add a `TextView` to display messages or fetched data.
3. **Create a Database Helper Class (`DatabaseHelper.java`):**
   - Create a new Java class named `DatabaseHelper` that extends `SQLiteOpenHelper`.
   - Implement the constructor, `onCreate()`, and `onUpgrade()` methods.
     - `onCreate()`: Define the SQL query to create your student table (`CREATE TABLE students (ID INTEGER PRIMARY KEY AUTOINCREMENT, NAME TEXT, AGE INTEGER)`).
     - `onUpgrade()`: Handle database schema upgrades (e.g., drop existing table and recreate).
   - Add methods for CRUD (Create, Read, Update, Delete) operations:
     - `addStudent(String name, int age)`: Inserts a new student record.
     - `getAllStudents()`: Retrieves all student records.
     - `updateStudent(int id, String newName, int newAge)`: Updates an existing record.
     - `deleteStudent(int id)`: Deletes a record by ID.
4. **Implement Logic (`MainActivity.java`):**
   - Declare UI components and an instance of `DatabaseHelper`.
   - Initialize UI components and `DatabaseHelper`.
   - Set `OnClickListener` for each button:
     - **Add Button:** Get name and age from `EditText`. Call `dbHelper.addStudent()`. Display Toast.
     - **View All Button:** Call `dbHelper.getAllStudents()`. Iterate through the `Cursor` to build a string of all students and display it in the `TextView`. Handle empty results.
     - **Update Button:** Get ID, new name, new age. Call `dbHelper.updateStudent()`. Display Toast.
     - **Delete Button:** Get ID. Call `dbHelper.deleteStudent()`. Display Toast.
   - Handle `NumberFormatException` for ID/Age inputs.

5. **Run the Application:**
    - Run the app on an emulator or device.
    - Add students, then view them. Try updating and deleting.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:padding="24dp"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="SQLite Database Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_gravity="center_horizontal"
        android:layout_marginBottom="30dp"/>

    <EditText
        android:id="@+id/editTextStudentName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Student Name"
        android:inputType="textPersonName"
        android:padding="12dp"
        android:layout_marginBottom="12dp"
        android:background="@drawable/rounded_edittext_background"/>

    <EditText
        android:id="@+id/editTextStudentAge"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Student Age"
        android:inputType="number"
        android:padding="12dp"
        android:layout_marginBottom="12dp"
        android:background="@drawable/rounded_edittext_background"/>

    <EditText
        android:id="@+id/editTextStudentId"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:hint="Student ID (for Update/Delete)"
        android:inputType="number"
        android:padding="12dp"
        android:layout_marginBottom="20dp"
        android:background="@drawable/rounded_edittext_background"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_marginBottom="20dp">

        <Button
            android:id="@+id/buttonAdd"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Add"
            android:padding="10dp"
            android:layout_marginEnd="8dp"
            android:backgroundTint="#4CAF50"
            android:textColor="#FFFFFF"
            android:textSize="16sp"/>

        <Button
            android:id="@+id/buttonViewAll"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="View All"
            android:padding="10dp"
            android:layout_marginEnd="8dp"
            android:backgroundTint="#2196F3"
            android:textColor="#FFFFFF"
            android:textSize="16sp"/>

        <Button
            android:id="@+id/buttonUpdate"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Update"
            android:padding="10dp"
            android:layout_marginEnd="8dp"
            android:backgroundTint="#FFC107"
            android:textColor="#333333"
            android:textSize="16sp"/>

        <Button
            android:id="@+id/buttonDelete"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1"
            android:text="Delete"
            android:padding="10dp"
            android:backgroundTint="#F44336"
            android:textColor="#FFFFFF"
            android:textSize="16sp"/>
    </LinearLayout>

    <TextView
        android:id="@+id/textViewResult"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:text="Database Operations Result:"
        android:textSize="16sp"
        android:textColor="#333333"
        android:padding="16dp"
        android:background="@drawable/rounded_textview_background_db"
        android:scrollbars="vertical"/>

</LinearLayout>
```

res/drawable/rounded_edittext_background.xml (Reuse or create)

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#F0F0F0"/>
    <corners android:radius="8dp"/>
    <stroke android:color="#CCCCCC" android:width="1dp"/>
</shape>
```

res/drawable/rounded_textview_background_db.xml *(Create this file)*

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android">
    <solid android:color="#E8F5E9"/>
    <corners android:radius="12dp"/>
    <stroke android:color="#81C784" android:width="2dp"/>
</shape>
```

DatabaseHelper.java

```
package com.example.sqlitedatabaseapp;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DatabaseHelper extends SQLiteOpenHelper {

    private static final String DATABASE_NAME = "Students.db";
    private static final int DATABASE_VERSION = 1;

    public static final String TABLE_NAME = "students";
    public static final String COL_ID = "ID";
    public static final String COL_NAME = "NAME";
    public static final String COL_AGE = "AGE";

    public DatabaseHelper(Context context) {
        super(context, DATABASE_NAME, null, DATABASE_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {
        // SQL query to create the students table
        String CREATE_TABLE = "CREATE TABLE " + TABLE_NAME + " (" +
                COL_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                COL_NAME + " TEXT, " +
                COL_AGE + " INTEGER" +
                ")";
        db.execSQL(CREATE_TABLE);
    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion)
{
        // Drop older table if existed
        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME);
        // Create tables again
        onCreate(db);
    }

    // Method to add a new student
    public boolean addStudent(String name, int age) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_NAME, name);
        contentValues.put(COL_AGE, age);

        long result = db.insert(TABLE_NAME, null, contentValues);
        db.close();
        return result != -1; // Returns true if data is inserted successfully
    }
    // Method to get all students
    public Cursor getAllStudents() {
        SQLiteDatabase db = this.getWritableDatabase();
        Cursor cursor = db.rawQuery("SELECT * FROM " + TABLE_NAME, null);
        return cursor;
    }

    // Method to update a student record
    public boolean updateStudent(int id, String newName, int newAge) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues contentValues = new ContentValues();
        contentValues.put(COL_NAME, newName);
        contentValues.put(COL_AGE, newAge);

        int result = db.update(TABLE_NAME, contentValues, COL_ID + " = ?",
new String[]{String.valueOf(id)});
        db.close();
        return result > 0; // Returns true if record is updated
    }

    // Method to delete a student record
    public int deleteStudent(int id) {
        SQLiteDatabase db = this.getWritableDatabase();
        int result = db.delete(TABLE_NAME, COL_ID + " = ?", new
String[]{String.valueOf(id)});
        db.close();
        return result; // Returns number of rows deleted
    }
}
```

MainActivity.java

```
package com.example.sqlitedatabaseapp;

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

public class MainActivity extends AppCompatActivity {

    private DatabaseHelper dbHelper;

    private EditText editTextStudentName, editTextStudentAge,
editTextStudentId;
    private Button buttonAdd, buttonViewAll, buttonUpdate, buttonDelete;
    private TextView textViewResult;

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

        dbHelper = new DatabaseHelper(this);

        editTextStudentName = findViewById(R.id.editTextStudentName);
        editTextStudentAge = findViewById(R.id.editTextStudentAge);
        editTextStudentId = findViewById(R.id.editTextStudentId);
        buttonAdd = findViewById(R.id.buttonAdd);
        buttonViewAll = findViewById(R.id.buttonViewAll);
        buttonUpdate = findViewById(R.id.buttonUpdate);
        buttonDelete = findViewById(R.id.buttonDelete);
        textViewResult = findViewById(R.id.textViewResult);

        addListeners();
    }

    private void addListeners() {
        buttonAdd.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                addStudent();
            }
        });

        buttonViewAll.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                viewAllStudents();
            }
        });

        buttonUpdate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                updateStudent();
            }
        });

        buttonDelete.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                deleteStudent();
            }
        });
    }

    private void addStudent() {
        String name = editTextStudentName.getText().toString().trim();
        String ageStr = editTextStudentAge.getText().toString().trim();

        if (name.isEmpty() || ageStr.isEmpty()) {
            Toast.makeText(this, "Please enter name and age.",
Toast.LENGTH_SHORT).show();
            return;
        }

        try {
            int age = Integer.parseInt(ageStr);
            boolean isInserted = dbHelper.addStudent(name, age);
            if (isInserted) {
                Toast.makeText(this, "Student Added Successfully!",
Toast.LENGTH_SHORT).show();
                editTextStudentName.setText("");
                editTextStudentAge.setText("");
            } else {
                Toast.makeText(this, "Failed to Add Student.",
Toast.LENGTH_SHORT).show();
            }
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Please enter a valid age.",
Toast.LENGTH_SHORT).show();
        }
    }

    private void viewAllStudents() {
        Cursor res = dbHelper.getAllStudents();
        if (res.getCount() == 0) {
            textViewResult.setText("Database Operations Result:\nNo Students
Found.");
            Toast.makeText(this, "No Students Found.",
Toast.LENGTH_SHORT).show();
            return;
        }

        StringBuilder buffer = new StringBuilder();
        buffer.append("Database Operations Result:\n\n");
        while (res.moveToNext()) {
            buffer.append("ID: ").append(res.getString(0)).append("\n");
            buffer.append("Name: ").append(res.getString(1)).append("\n");
            buffer.append("Age: ").append(res.getString(2)).append("\n\n");
        }
        textViewResult.setText(buffer.toString());
        res.close(); // Close the cursor
    }

    private void updateStudent() {
        String idStr = editTextStudentId.getText().toString().trim();
        String name = editTextStudentName.getText().toString().trim();
        String ageStr = editTextStudentAge.getText().toString().trim();

        if (idStr.isEmpty() || name.isEmpty() || ageStr.isEmpty()) {
            Toast.makeText(this, "Please enter ID, Name, and Age to update.",
Toast.LENGTH_SHORT).show();
            return;
        }

        try {
            int id = Integer.parseInt(idStr);
            int age = Integer.parseInt(ageStr);
            boolean isUpdated = dbHelper.updateStudent(id, name, age);
            if (isUpdated) {
                Toast.makeText(this, "Student Updated Successfully!",
Toast.LENGTH_SHORT).show();
                editTextStudentId.setText("");
                editTextStudentName.setText("");
                editTextStudentAge.setText("");
            } else {
                Toast.makeText(this, "Failed to Update Student. Check ID.",
Toast.LENGTH_SHORT).show();
            }
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Please enter valid ID and Age.",
Toast.LENGTH_SHORT).show();
        }
    }

    private void deleteStudent() {
        String idStr = editTextStudentId.getText().toString().trim();
        if (idStr.isEmpty()) {
            Toast.makeText(this, "Please enter Student ID to delete.",
Toast.LENGTH_SHORT).show();
            return;
        }

        try {
            int id = Integer.parseInt(idStr);
            int deletedRows = dbHelper.deleteStudent(id);
            if (deletedRows > 0) {
                Toast.makeText(this, "Student Deleted Successfully!",
Toast.LENGTH_SHORT).show();
                editTextStudentId.setText("");
            } else {
                Toast.makeText(this, "Failed to Delete Student. ID not
found.", Toast.LENGTH_SHORT).show();
            }
        } catch (NumberFormatException e) {
            Toast.makeText(this, "Please enter a valid ID.",
Toast.LENGTH_SHORT).show();
        }
    }
}
```

Input

- **Add Student:** Enter "John Doe" for Name, "20" for Age. Click "Add". Repeat for "Jane Smith", "22".
- **View All:** Click "View All".
- **Update Student:** Enter an existing ID (e.g., "1") for ID, "Johnny D." for Name, "21" for Age. Click "Update".
- **Delete Student:** Enter an existing ID (e.g., "2") for ID. Click "Delete".

Expected Output

- **Add:** Toast "Student Added Successfully!".
- **View All:** The `textViewResult` will display a list of all students with their ID, Name, and Age.
- `Database Operations Result:`
-
- `ID: 1`
- `Name: John Doe`
- `Age: 20`
-
- `ID: 2`
- `Name: Jane Smith`
- `Age: 22`

  (IDs are auto-incremented)

- **Update:** Toast "Student Updated Successfully!". `View All` will show the updated record.
- **Delete:** Toast "Student Deleted Successfully!". `View All` will show the record removed.
- Error Toasts will appear for invalid inputs or operations (e.g., "Please enter name and age.").

Lab 13: Simulate Paintbrush Applications

Title

Simulate Paintbrush Applications

Aim

To create a basic Android application that simulates a simple paintbrush, allowing users to draw lines on a custom `View` using touch input.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `PaintbrushApp`.
   - **Package name:** `com.example.paintbrushapp`
   - **Language:** Java.
2. **Create a Custom View Class (`DrawingView.java`):**
   - Create a new Java class named `DrawingView` that extends `View`.
   - **Constructor:** Initialize `Paint` and `Path` objects. `Paint` defines drawing style (color, stroke width), `Path` stores the lines drawn.
   - **`onDraw(Canvas canvas)`:** This method is called to draw the view. Draw the `Path` onto the `Canvas` using the `Paint` object.
   - **`onTouchEvent(MotionEvent event)`:** Override this method to handle touch events:
     - `ACTION_DOWN`: Start a new path at the touch coordinates.
     - `ACTION_MOVE`: Extend the current path to the new touch coordinates.
     - `ACTION_UP`: End the path.
     - After modifying the path, call `invalidate()` to force the view to redraw itself (`onDraw` will be called).
3. **Design the Layout (`activity_main.xml`):**
   - Add your custom `DrawingView` to the layout. Ensure it fills the parent or has a defined size.
   - Add buttons for changing color or clearing the canvas (optional, but good for a full app). For simplicity, we'll focus on drawing.
4. **Implement Logic (`MainActivity.java`):**
   - Get a reference to your `DrawingView`.
   - (Optional) Implement button listeners for clear, color change, etc., and call methods on `DrawingView` to handle them.
5. **Run the Application:**
   - Run the app on an emulator or device.
   - Use your finger or mouse to draw on the screen.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
```

```xml
    android:gravity="center_horizontal"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Simple Paintbrush"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="20dp"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:gravity="center"
        android:layout_marginBottom="10dp">

        <Button
            android:id="@+id/buttonRed"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Red"
            android:backgroundTint="#F44336"
            android:textColor="#FFFFFF"
            android:layout_marginEnd="8dp"/>
        <Button
            android:id="@+id/buttonBlue"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Blue"
            android:backgroundTint="#2196F3"
            android:textColor="#FFFFFF"
            android:layout_marginEnd="8dp"/>
        <Button
            android:id="@+id/buttonGreen"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="Green"
            android:backgroundTint="#4CAF50"
            android:textColor="#FFFFFF"/>
    </LinearLayout>

    <Button
        android:id="@+id/buttonClear"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Clear Canvas"
        android:backgroundTint="#FF9800"
        android:textColor="#FFFFFF"
        android:layout_marginBottom="10dp"/>

    <com.example.paintbrushapp.DrawingView
        android:id="@+id/drawingView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="#FFFFFF"
        android:layout_margin="16dp"
        android:padding="4dp"
        android:elevation="4dp"
        android:outlineProvider="background"
        android:backgroundTint="#F5F5F5"/>

</LinearLayout>
```

DrawingView.java

```
package com.example.paintbrushapp;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.view.MotionEvent;
import android.view.View;
import java.util.ArrayList;

public class DrawingView extends View {

    private Paint drawPaint;
    private Path drawPath;
    private ArrayList<Path> paths = new ArrayList<>();
    private ArrayList<Paint> paints = new ArrayList<>();

    private int paintColor = Color.BLACK; // Default color
    private int strokeWidth = 10; // Default stroke width

    public DrawingView(Context context, AttributeSet attrs) {
        super(context, attrs);
        setupDrawing();
    }

    private void setupDrawing() {
        drawPath = new Path();
        drawPaint = new Paint();

        drawPaint.setColor(paintColor);
        drawPaint.setAntiAlias(true);
        drawPaint.setStrokeWidth(strokeWidth);
        drawPaint.setStyle(Paint.Style.STROKE);
        drawPaint.setStrokeJoin(Paint.Join.ROUND);
        drawPaint.setStrokeCap(Paint.Cap.ROUND);
    }

    // This method is called when the view is drawn
    @Override
    protected void onDraw(Canvas canvas) {
        // Draw all existing paths
        for (int i = 0; i < paths.size(); i++) {
            canvas.drawPath(paths.get(i), paints.get(i));
        }
    }

    // This method is called when a touch event occurs
    @Override
    public boolean onTouchEvent(MotionEvent event) {
        float touchX = event.getX();
        float touchY = event.getY();

        switch (event.getAction()) {
            case MotionEvent.ACTION_DOWN:
                // Start a new path and add current paint settings
                drawPath = new Path();
                paths.add(drawPath);
                paints.add(getNewPaint()); // Store a copy of current paint
settings
                drawPath.moveTo(touchX, touchY);
                break;
            case MotionEvent.ACTION_MOVE:
                drawPath.lineTo(touchX, touchY);
                break;
            case MotionEvent.ACTION_UP:
                // Nothing specific needed here, path is already complete
                break;
            default:
                return false;
        }
        invalidate(); // Request a redraw of the view
        return true;
    }

    // Method to set drawing color
    public void setColor(int newColor) {
        paintColor = newColor;
        // Update the current drawing paint's color for new strokes
        drawPaint.setColor(paintColor);
    }

    // Method to clear the canvas
    public void clearCanvas() {
        paths.clear(); // Clear all paths
        paints.clear(); // Clear all paints
        invalidate(); // Redraw the view
    }

    // Helper to create a new Paint object with current settings for each new
path
    private Paint getNewPaint() {
        Paint newPaint = new Paint();
        newPaint.setColor(paintColor);
        newPaint.setAntiAlias(true);
        newPaint.setStrokeWidth(strokeWidth);
        newPaint.setStyle(Paint.Style.STROKE);
        newPaint.setStrokeJoin(Paint.Join.ROUND);
        newPaint.setStrokeCap(Paint.Cap.ROUND);
        return newPaint;
    }
}
```

MainActivity.java

```
package com.example.paintbrushapp;

import androidx.appcompat.app.AppCompatActivity;
import android.graphics.Color;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;

public class MainActivity extends AppCompatActivity {

    private DrawingView drawingView;
    private Button buttonRed, buttonBlue, buttonGreen, buttonClear;

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

        drawingView = findViewById(R.id.drawingView);
        buttonRed = findViewById(R.id.buttonRed);
        buttonBlue = findViewById(R.id.buttonBlue);
        buttonGreen = findViewById(R.id.buttonGreen);
        buttonClear = findViewById(R.id.buttonClear);

        buttonRed.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                drawingView.setColor(Color.RED);
            }
        });

        buttonBlue.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                drawingView.setColor(Color.BLUE);
            }
        });

        buttonGreen.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                drawingView.setColor(Color.GREEN);
            }
        });

        buttonClear.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                drawingView.clearCanvas();
            }
        });
    }
}
```

Input

- Touch and drag your finger across the white drawing area.
- Tap the "Red", "Blue", or "Green" buttons to change the drawing color.
- Tap the "Clear Canvas" button to erase everything.

Expected Output

- As you drag your finger, lines will be drawn on the screen in the selected color.
- Changing the color will make subsequent lines appear in that new color.
- Tapping "Clear Canvas" will erase all drawn lines, leaving a blank white canvas.

Lab 14: Draw an Object

Title

Draw an Object

Aim

To demonstrate how to draw a predefined geometric object (e.g., a rectangle, circle, or triangle) on a custom `View` using Android's `Canvas` and `Paint` classes.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `DrawObjectApp`.
   - **Package name:** `com.example.drawobjectapp`
   - **Language:** Java.
2. **Create a Custom View Class (`DrawingObjectView.java`):**
   - Create a new Java class named `DrawingObjectView` that extends `View`.
   - **Constructor:** Initialize a `Paint` object to define the color and style for drawing.
   - **`onDraw(Canvas canvas)`:** This is where the drawing logic goes.
     - Use `canvas.drawRect()`, `canvas.drawCircle()`, `canvas.drawPath()`, etc., along with your `Paint` object, to draw the desired shape.
     - You can get the width and height of the view using `getWidth()` and `getHeight()` to position your object dynamically.
3. **Design the Layout (`activity_main.xml`):**
   - Add your custom `DrawingObjectView` to the layout. Ensure it fills the parent or has a defined size.
4. **Implement Logic (`MainActivity.java`):**
   - Simply set the content view to your layout containing `DrawingObjectView`. No complex logic is needed in `MainActivity` for a static drawing.
5. **Run the Application:**
   - Run the app on an emulator or device.

Source Code

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    android:gravity="center_horizontal"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Draw a Simple Object"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_marginTop="20dp"
        android:layout_marginBottom="20dp"/>
    <com.example.drawobjectapp.DrawingObjectView
        android:id="@+id/drawingObjectView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:background="#F0F0F0"
        android:layout_margin="16dp"
        android:padding="4dp"
        android:elevation="4dp"
        android:outlineProvider="background"/>

</LinearLayout>
```

DrawingObjectView.java

```
package com.example.drawobjectapp;

import android.content.Context;
import android.graphics.Canvas;
import android.graphics.Color;
import android.graphics.Paint;
import android.graphics.Path;
import android.util.AttributeSet;
import android.view.View;

public class DrawingObjectView extends View {

    private Paint drawPaint;

    public DrawingObjectView(Context context, AttributeSet attrs) {
        super(context, attrs);
        setupPaint();
    }

    private void setupPaint() {
        drawPaint = new Paint();
        drawPaint.setColor(Color.BLUE); // Set initial color
        drawPaint.setAntiAlias(true);
        drawPaint.setStrokeWidth(10);
        drawPaint.setStyle(Paint.Style.STROKE); // Draw outline
        drawPaint.setStrokeJoin(Paint.Join.ROUND);
        drawPaint.setStrokeCap(Paint.Cap.ROUND);
    }

    @Override
    protected void onDraw(Canvas canvas) {
        super.onDraw(canvas);

        int width = getWidth();
        int height = getHeight();

        // Draw a Rectangle
        drawPaint.setColor(Color.RED);
        drawPaint.setStyle(Paint.Style.FILL); // Fill the shape
        canvas.drawRect(width / 4, height / 4, width * 3 / 4, height * 3 / 4,
drawPaint);

        // Draw a Circle
        drawPaint.setColor(Color.GREEN);
        drawPaint.setStyle(Paint.Style.STROKE); // Draw outline
        drawPaint.setStrokeWidth(15);
        canvas.drawCircle(width / 2, height / 2, Math.min(width, height) / 4,
drawPaint);

        // Draw a Triangle using Path
        drawPaint.setColor(Color.MAGENTA);
        drawPaint.setStyle(Paint.Style.FILL_AND_STROKE); // Fill and outline
        drawPaint.setStrokeWidth(8);

        Path trianglePath = new Path();
        trianglePath.moveTo(width / 2, height / 8); // Top point
        trianglePath.lineTo(width / 8, height * 7 / 8); // Bottom-left point
        trianglePath.lineTo(width * 7 / 8, height * 7 / 8); // Bottom-right
point
        trianglePath.close(); // Connects the last point to the first point
        canvas.drawPath(trianglePath, drawPaint);

        // Draw some text
        drawPaint.setColor(Color.BLACK);
        drawPaint.setTextSize(40);
        drawPaint.setStyle(Paint.Style.FILL);
        canvas.drawText("Shapes!", width / 2 - 80, height / 2 + 20,
drawPaint);
    }
}
```

MainActivity.java

```
package com.example.drawobjectapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;

public class MainActivity extends AppCompatActivity {

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

Input

- Simply launch the application. No user input is required for this static drawing.

Expected Output

- The application will display a screen with a background.
- On this background, you will see a red filled rectangle, a green outlined circle, a magenta filled and outlined triangle, and black text "Shapes!". The exact positioning will be relative to the view's size.

Lab 15: Implement WebView

Title

Implement WebView

Aim

To implement an Android `WebView` component to display web content (local HTML files or remote web pages) directly within the application.

Procedure

1. **Create a New Android Project:**
   - Open Android Studio.
   - Create a new "Empty Activity" project named `WebViewApp`.
   - **Package name:** `com.example.webviewapp`
   - **Language:** Java.
2. **Add Internet Permission:**
   - Open `app/src/main/AndroidManifest.xml`.
   - Add the following permission *outside* the `<application>` tag but inside the `<manifest>` tag:
   - `<uses-permission android:name="android.permission.INTERNET" />`
3. **Design the Layout (`activity_main.xml`):**
   - Add a `WebView` widget to fill the entire layout.
4. **Implement Logic (`MainActivity.java`):**
   - Get a reference to the `WebView`.
   - **Enable JavaScript:** `webView.getSettings().setJavaScriptEnabled(true);` (Crucial for most modern websites).
   - **Set a `WebViewClient`:** This prevents the default behavior of opening external links in the device's browser and keeps navigation within your `WebView`. Override `shouldOverrideUrlLoading()`.
   - **Load URL:** Use `webView.loadUrl("https://www.google.com");` (or any other URL).
   - **Handle Back Button (Optional but Recommended):** Override `onBackPressed()` to allow the `WebView` to navigate back in its history if possible, instead of closing the activity.
5. **Run the Application:**
   - Run the app on an emulator or device. Ensure you have an active internet connection if loading a remote URL.

Source Code

`AndroidManifest.xml` (Important: Add Internet Permission)

```
<?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.WebViewApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout 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:orientation="vertical"
    tools:context=".MainActivity">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="WebView Example"
        android:textSize="28sp"
        android:textStyle="bold"
        android:layout_gravity="center_horizontal"
        android:layout_marginTop="16dp"
        android:layout_marginBottom="16dp"/>

    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="0dp"
        android:layout_weight="1"
        android:layout_margin="8dp"
        android:background="#FFFFFF"/>

</LinearLayout>
```

MainActivity.java

```
package com.example.webviewapp;

import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.Toast;

public class MainActivity extends AppCompatActivity {

    private WebView webView;

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

        webView = findViewById(R.id.webView);

        // Enable JavaScript (important for most modern websites)
        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);

        // Set a WebViewClient to keep links opening inside your app
        webView.setWebViewClient(new WebViewClient() {
            @Override
            public boolean shouldOverrideUrlLoading(WebView view, String url)
{
                // Return false to let WebView handle the URL itself
                return false;
            }

            @Override
            public void onReceivedError(WebView view, int errorCode, String
description, String failingUrl) {
                super.onReceivedError(view, errorCode, description,
failingUrl);
                Toast.makeText(MainActivity.this, "Error loading page: " +
description, Toast.LENGTH_LONG).show();
            }
        });

        // Load a URL
        // Make sure you have internet permission in AndroidManifest.xml
        webView.loadUrl("https://www.google.com");
        // You can also load local HTML files from the 'assets' folder:
        // webView.loadUrl("file:///android_asset/my_local_page.html");
    }

    // Handle back button press to navigate WebView history
    @Override
    public void onBackPressed() {
        if (webView.canGoBack()) {
            webView.goBack();
        } else {
            super.onBackPressed();
        }
    }
}
```

Input

- Launch the application.
- (Optional) If the loaded page has links, tap on them to navigate within the `WebView`.
- Press the device's back button.

Expected Output

- The `WebView` will load and display the content of `https://www.google.com` (or the URL you specified) within the application.
- If you navigate to other pages within the `WebView`, pressing the device's back button will take you to the previous page in the `WebView`'s history. If there's no history, the app will close.

- If there's a network error or the URL is invalid, a Toast message indicating the error will appear.