* **Input Controls**

Below is a comprehensive guide to various input controls and UI components in Android. Each section includes an explanation and a simple code example. You can mix and match these concepts to build robust and interactive user interfaces.

**1. Buttons**

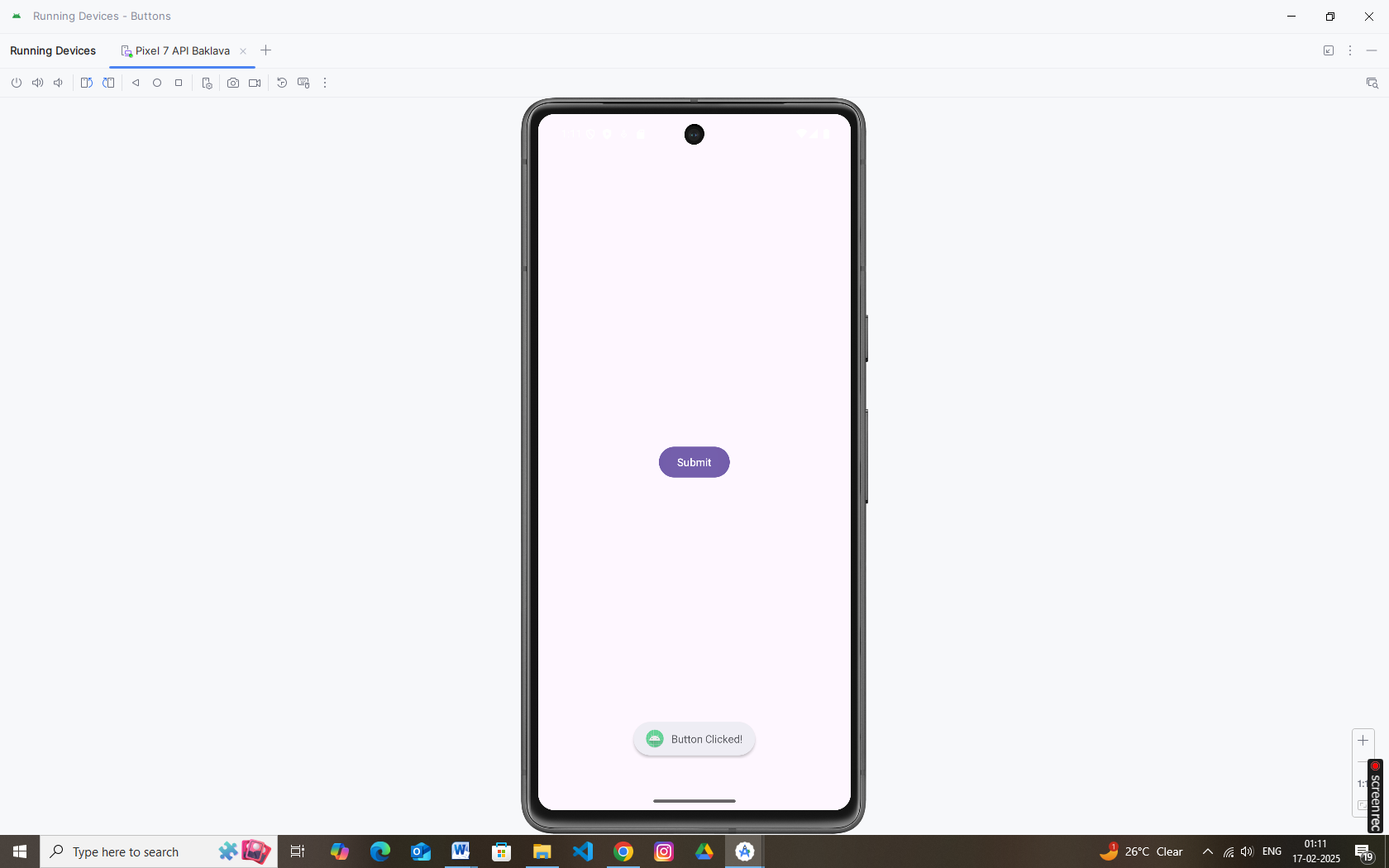
**Description:**  
Buttons trigger actions when clicked.

**XML Example:**

*<?*xml version="1.0" encoding="utf-8"*?>  
<!-- res/layout/activity\_main.xml -->*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="16dp">  
  
 <Button  
 android:id="@+id/btnSubmit"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Submit" />  
  
  
</LinearLayout>

**Java Example:**

package com.example.buttons;  
  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
 Button btnSubmit = findViewById(R.id.*btnSubmit*);  
 btnSubmit.setOnClickListener(v ->  
 Toast.*makeText*(this, "Button Clicked!", Toast.*LENGTH\_SHORT*).show()  
 );  
  
 }  
}



**2. Checkboxes**

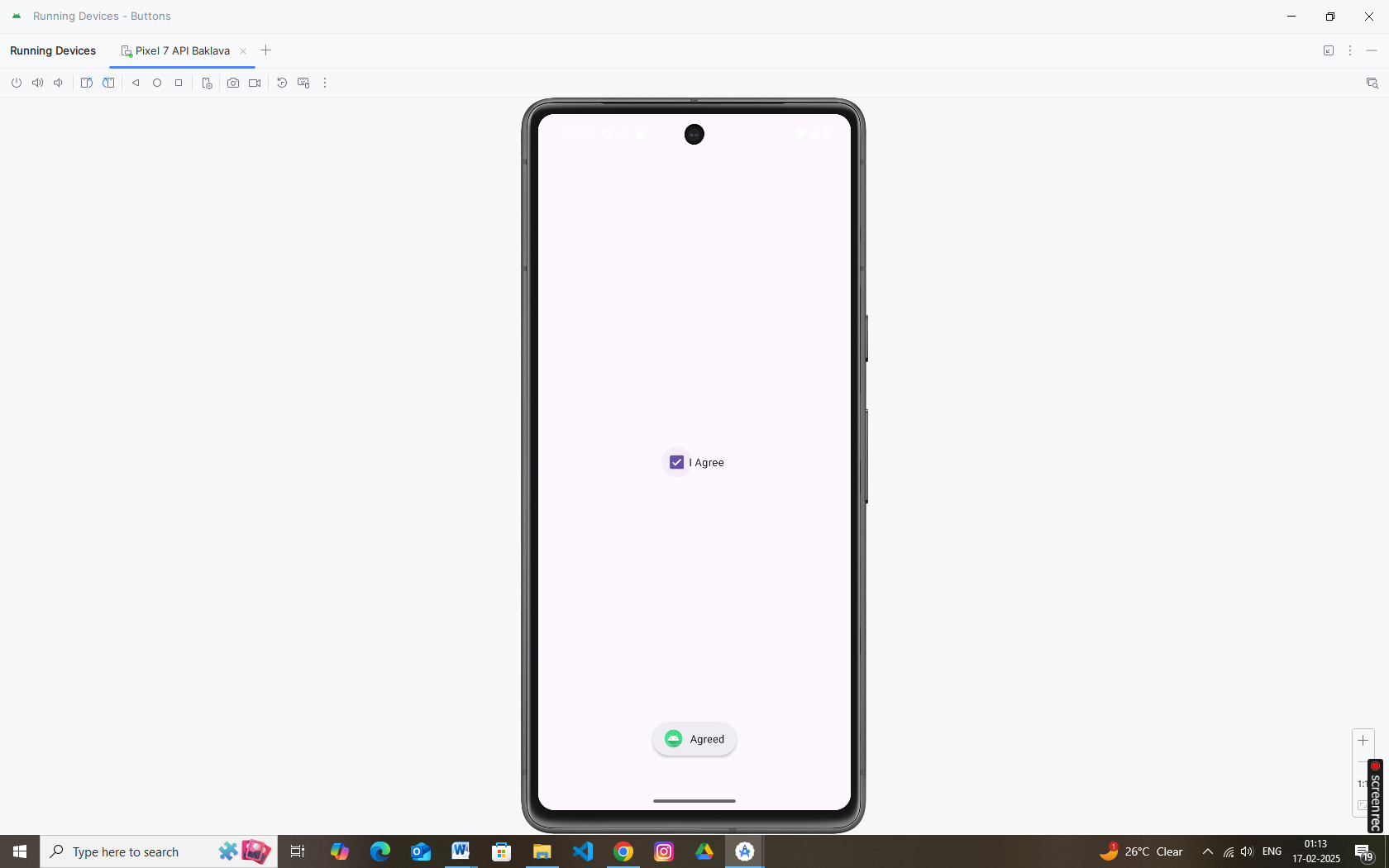
**Description:**  
Checkboxes allow users to select one or more options.

**XML Example:**

*<?*xml version="1.0" encoding="utf-8"*?>  
<!-- res/layout/activity\_main.xml -->*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="16dp">  
  
 <CheckBox  
 android:id="@+id/checkboxAgree"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="I Agree" />  
  
  
</LinearLayout>

**Java Example:**

package com.example.buttons;  
  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.CheckBox;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 CheckBox checkboxAgree = findViewById(R.id.*checkboxAgree*);  
 checkboxAgree.setOnCheckedChangeListener((buttonView, isChecked) -> {  
 String msg = isChecked ? "Agreed" : "Not Agreed";  
 Toast.*makeText*(this, msg, Toast.*LENGTH\_SHORT*).show();  
 });  
  
  
 }  
}



**3. Radio Buttons**

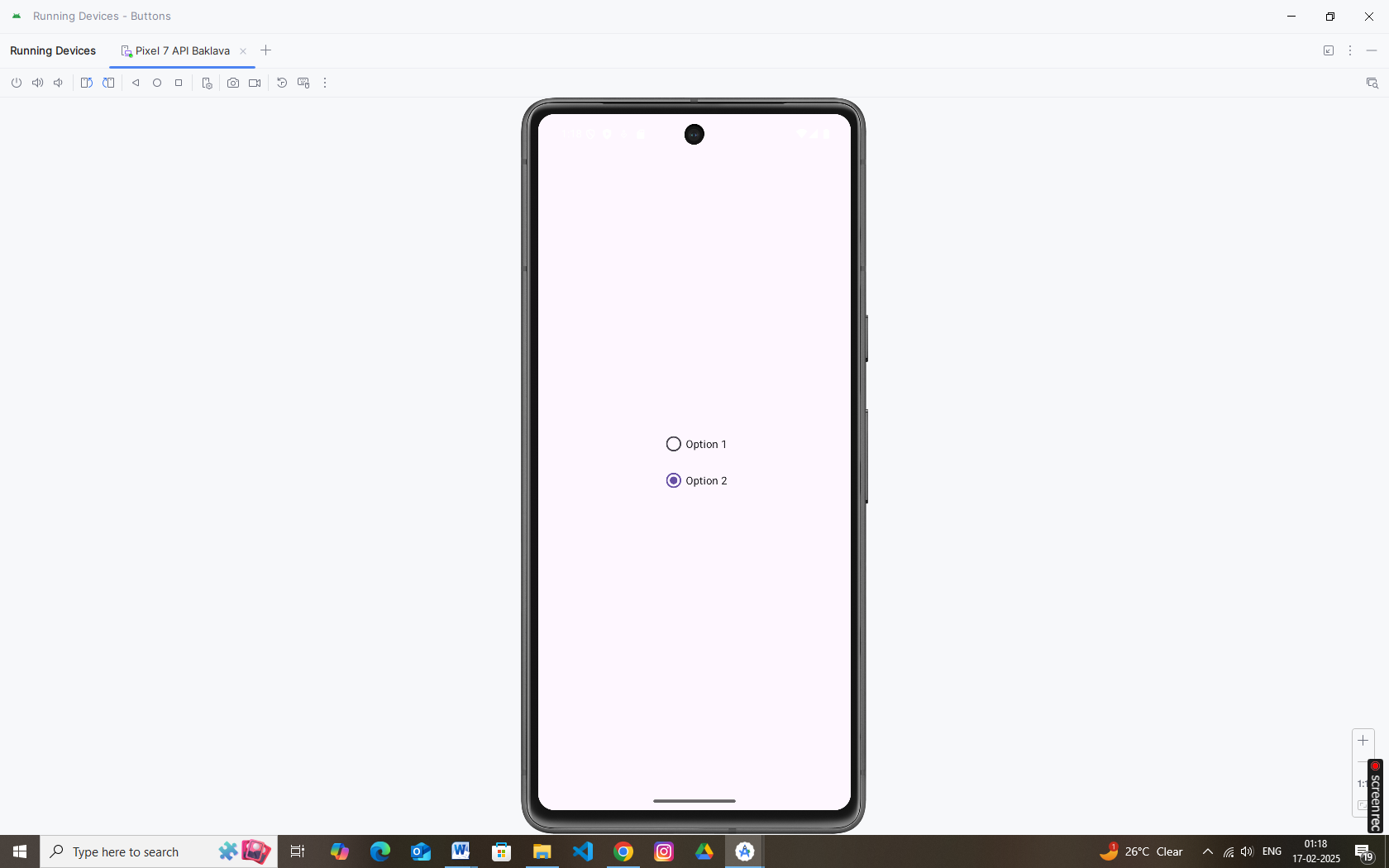
**Description:**  
Radio buttons allow users to select one option among a group.

**XML Example (with RadioGroup):**

*<?*xml version="1.0" encoding="utf-8"*?>  
<!-- res/layout/activity\_main.xml -->*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="16dp">  
  
 <RadioGroup  
 android:id="@+id/radioGroup"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content">  
  
 <RadioButton  
 android:id="@+id/radioOption1"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Option 1" />  
  
 <RadioButton  
 android:id="@+id/radioOption2"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Option 2" />  
 </RadioGroup>  
  
  
  
</LinearLayout>

**Java Example:**

package com.example.buttons;  
  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.CheckBox;  
import android.widget.RadioButton;  
import android.widget.RadioGroup;  
import android.widget.Toast;  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 RadioGroup radioGroup = findViewById(R.id.*radioGroup*);  
 radioGroup.setOnCheckedChangeListener((group, checkedId) -> {  
 RadioButton radioButton = findViewById(checkedId);  
 Toast.*makeText*(this, "Selected: " + radioButton.getText(), Toast.*LENGTH\_SHORT*).show();  
 });  
  
  
 }  
}



**4. Toggle Buttons**

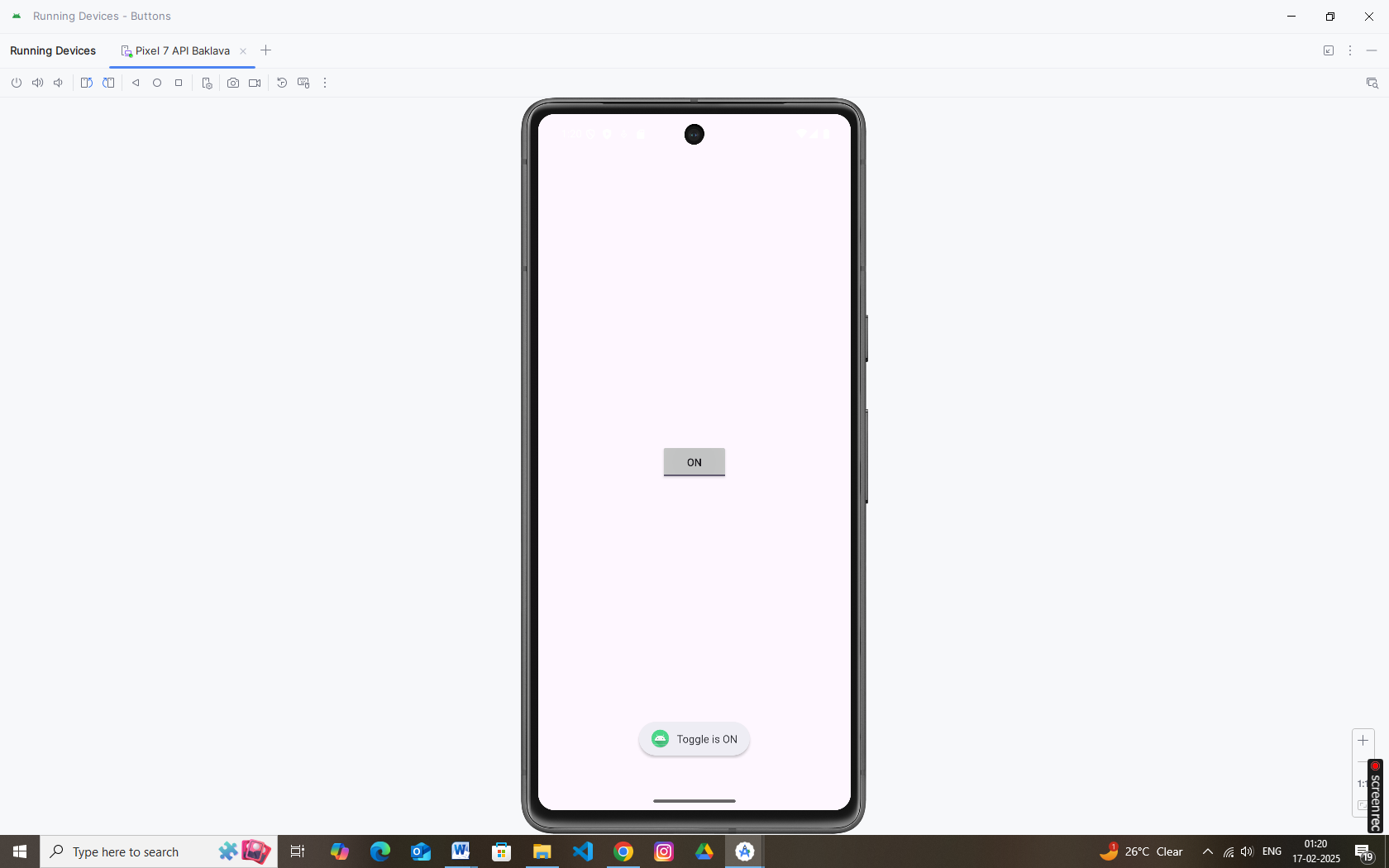
**Description:**  
Toggle buttons act like switches that change state when pressed.

**XML Example:**

*<?*xml version="1.0" encoding="utf-8"*?>  
<!-- res/layout/activity\_main.xml -->*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="16dp">  
  
 <ToggleButton  
 android:id="@+id/toggleButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textOn="ON"  
 android:textOff="OFF" />  
  
  
</LinearLayout>

**Java Example:**

package com.example.buttons;  
  
import android.os.Bundle;  
import android.widget.Button;  
import android.widget.CheckBox;  
import android.widget.RadioButton;  
import android.widget.RadioGroup;  
import android.widget.Toast;  
import android.widget.ToggleButton;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 ToggleButton toggleButton = findViewById(R.id.*toggleButton*);  
 toggleButton.setOnCheckedChangeListener((buttonView, isChecked) -> {  
 String status = isChecked ? "Toggle is ON" : "Toggle is OFF";  
 Toast.*makeText*(this, status, Toast.*LENGTH\_SHORT*).show();  
 });  
  
  
 }  
}



**5. Spinners**

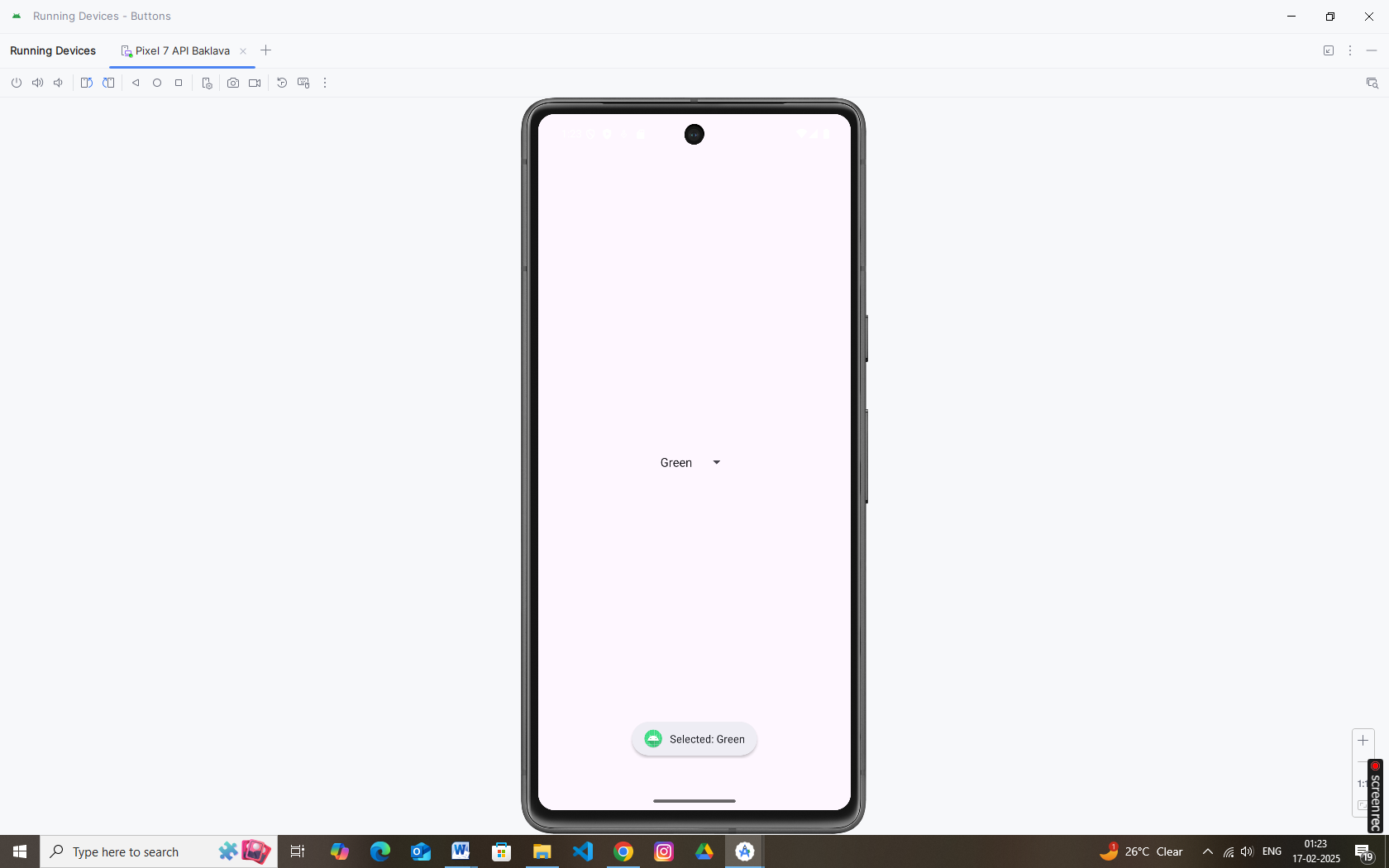
**Description:**  
Spinners provide a dropdown list of options.

**XML Example:**

*<?*xml version="1.0" encoding="utf-8"*?>  
<!-- res/layout/activity\_main.xml -->*<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"  
 android:gravity="center"  
 android:padding="16dp">  
  
 <Spinner  
 android:id="@+id/spinnerOptions"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" />  
  
  
  
</LinearLayout>

**Java Example:**

package com.example.buttons;  
  
import android.os.Bundle;  
import android.view.View;  
import android.widget.AdapterView;  
import android.widget.ArrayAdapter;  
import android.widget.Spinner;  
import android.widget.Toast;  
  
import androidx.appcompat.app.AppCompatActivity;  
  
public class MainActivity extends AppCompatActivity {  
  
 @Override  
 protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_main*);  
  
 Spinner spinner = findViewById(R.id.*spinnerOptions*);  
 String[] options = {"Red", "Green", "Blue"};  
 ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.*simple\_spinner\_item*, options);  
 adapter.setDropDownViewResource(android.R.layout.*simple\_spinner\_dropdown\_item*);  
 spinner.setAdapter(adapter);  
  
 spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {  
 @Override  
 public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {  
 Toast.*makeText*(getApplicationContext(), "Selected: " + options[position], Toast.*LENGTH\_SHORT*).show();  
 }  
 @Override  
 public void onNothingSelected(AdapterView<?> parent) {}  
 });  
  
  
 }  
}



**6. Input Events**

**Description:**  
Input events (such as touch, click, or key events) allow you to respond to user interactions.

**Example (Button Click):**

Button btnAction = findViewById(R.id.*btnSubmit*);  
btnAction.setOnClickListener(v -> {  
 *// Handle click event here* Toast.*makeText*(this, "Button was clicked!", Toast.*LENGTH\_SHORT*).show();  
});

You can also override other event methods (like onTouchEvent or onKeyDown) in your activity for more granular control.

**7. Menus**

**Description:**  
Menus (options, context, or popup) provide actions that the user can perform.

**Menu XML (res/menu/main\_menu.xml):**

Xml

*<?*xml version="1.0" encoding="utf-8"*?>*<menu xmlns:android="http://schemas.android.com/apk/res/android">  
 <item  
 android:id="@+id/action\_settings"  
 android:title="Settings" />  
 <item  
 android:id="@+id/action\_about"  
 android:title="About" />  
</menu>

**Java Example (Activity):**

public boolean onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.main\_menu, menu);  
 return true;  
}  
  
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
 switch (item.getItemId()) {  
 case R.id.*action\_settings*:  
 Toast.*makeText*(this, "Settings selected", Toast.*LENGTH\_SHORT*).show();  
 return true;  
 case R.id.*action\_about*:  
 Toast.*makeText*(this, "About selected", Toast.*LENGTH\_SHORT*).show();  
 return true;  
 default:  
 return super.onOptionsItemSelected(item);  
 }

**8. Toast**

**Description:**  
Toasts display a short message to the user that automatically disappears.

**Java Example:**

Toast.*makeText*(getApplicationContext(), "This is a Toast message", Toast.*LENGTH\_SHORT*).show();

**9. Dialogs**

**Description:**  
Dialogs prompt the user with alerts or choices.

**Example: AlertDialog**

new AlertDialog.Builder(this)  
 .setTitle("Exit")  
 .setMessage("Are you sure you want to exit?")  
 .setPositiveButton("Yes", (dialog, which) -> finish())  
 .setNegativeButton("No", null)  
 .show();

**10. Styles and Themes**

**Description:**  
Styles and themes let you define the look and feel of your app consistently across multiple components.

**Define a Style (res/values/styles.xml):**

<resources>  
 <style name="CustomButtonStyle">  
 <item name="android:background">#FF5722</item>  
 <item name="android:textColor">#FFFFFF</item>  
 <item name="android:padding">10dp</item>  
 </style>  
</resources>

**Apply Style in XML:**

<Button  
 style="@style/CustomButtonStyle"  
 android:id="@+id/styledButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Styled Button" />

**Themes:**  
Themes are applied at the app or activity level (see the AndroidManifest.xml or your styles.xml for your app theme).

**11. Creating Lists (Using ListView)**

**Description:**  
ListView displays a vertically scrollable list of items.

**XML Layout (res/layout/activity\_list.xml):**

<ListView  
 android:id="@+id/listView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>

**Java Example:**

ListView listView = findViewById(R.id.*listView*);  
String[] items = {"Apple", "Banana", "Cherry", "Date", "Elderberry"};  
ArrayAdapter<String> adapter = new ArrayAdapter<>(this,  
 android.R.layout.*simple\_list\_item\_1*, items);  
listView.setAdapter(adapter);

**12. Custom Lists (Using Custom Adapter for ListView)**

**Description:**  
Custom lists let you design each row with a custom layout and control how data is bound.

**Custom Row Layout (res/layout/custom\_row.xml):**

<Button  
 style="@style/CustomButtonStyle"  
 android:id="@+id/styledButton"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:text="Styled Button" />  
  
  
<ListView  
 android:id="@+id/listView"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"/>  
  
  
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="horizontal"  
 android:padding="10dp">  
  
 <ImageView  
 android:id="@+id/imgIcon"  
 android:layout\_width="40dp"  
 android:layout\_height="40dp"  
 android:src="@mipmap/ic\_launcher" />  
  
 <TextView  
 android:id="@+id/tvTitle"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="10dp"  
 android:textSize="18sp" />  
</LinearLayout>

