

MORE ANDROID VIEWS



ANDROID UI ELEMENTS

- Image View
- Switch
- Toggle button
- Radio Button
- Check box
- Spinner
- List View
- **Seek bar**

IMAGE VIEW

```
<ImageView  
    android:id="@+id/imageView"  
    android:layout_width="171dp"  
    android:layout_height="wrap_content"  
    android:layout_alignParentTop="true"  
    android:layout_alignParentEnd="true"  
    android:layout_marginTop="117dp"  
    android:layout_marginEnd="127dp"  
    tools:src="@tools:sample/avatars" />
```

Displays image resources, for example Bitmap or Drawable resources. `ImageView`

Toggle Button

Adding toggle button to xml

- Allows the user to change a setting between two states

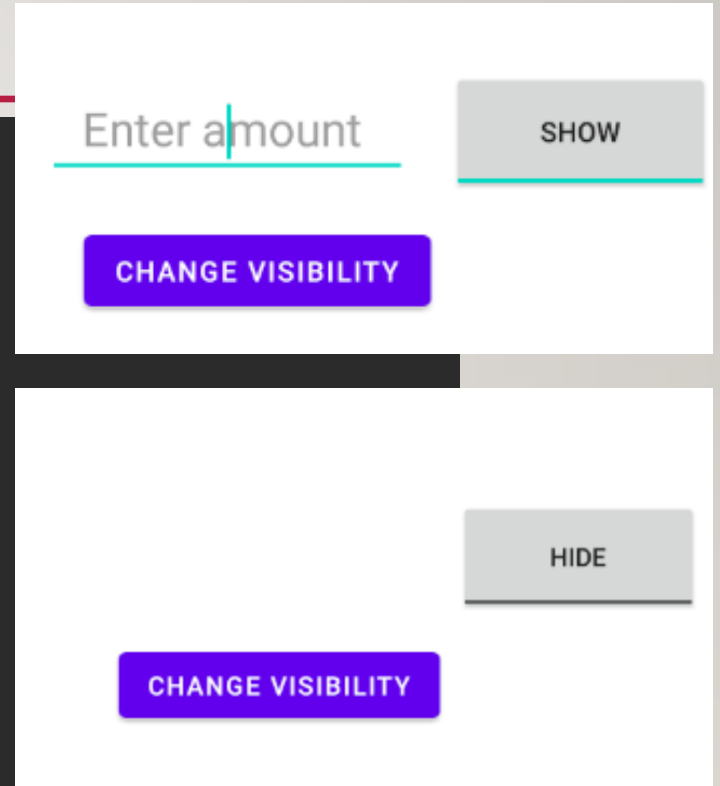
<ToggleButton

```
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:id="@+id/toggle_button"  
    android:textOn="On"  
    android:textOff="Off"  
  
    android:onClick="myMethod"
```

/>

//example :- Hide / show Edit Text using toggle button

```
btn.setOnClickListener(e -> {  
    boolean show = toggleButton.isChecked();  
    if(show){  
        editText.setVisibility(View.VISIBLE);  
    }else{  
        editText.setVisibility(View.INVISIBLE);  
    }  
  
    Toast.makeText(this, ""+toggleButton.getText().toString(),  
    Toast.LENGTH_LONG).show() ;  
});
```



RADIO BUTTON

- Radio buttons allow the user to select one option from a set.
- You should use radio buttons for optional sets that are mutually exclusive
- you must group them together inside a RadioGroup

ATTENDING?

☒ Yes ☐ Maybe ☐ No

☒ Yes

☐ Maybe

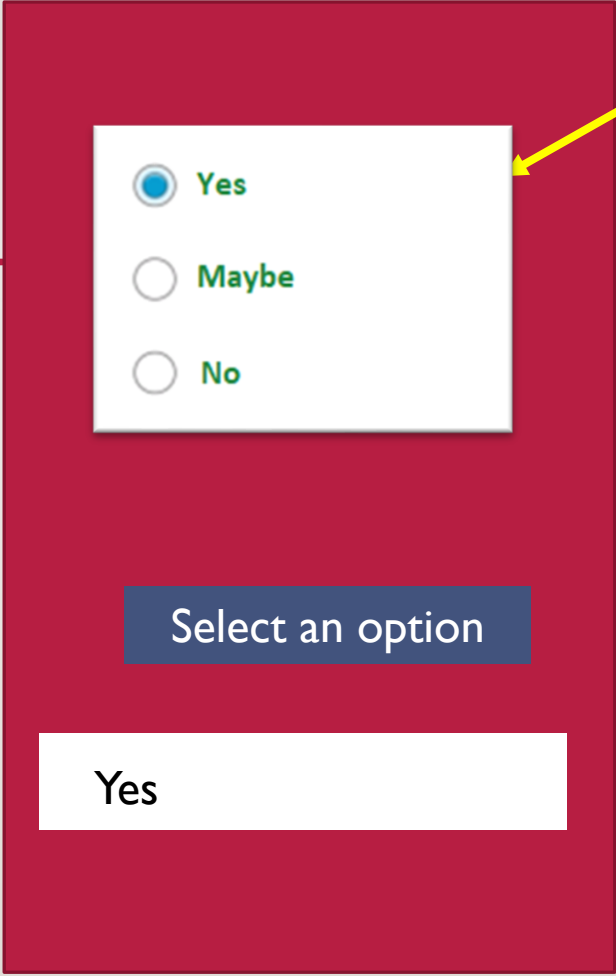
☐ No

EXAMPLE FOR RADIO BUTTONS

Build android app as shown in the figure

The app should do the following :-

- **Allow the user to select an option (Yes, Maybe or No using radio buttons)**
- **display the selected color when the button is clicked.**



The image shows a mockup of an Android application interface. It features a dark red background. At the top, there is a white dialog box containing three radio button options: 'Yes' (selected), 'Maybe', and 'No'. Below this dialog box is a blue button labeled 'Select an option'. At the bottom, there is a white rectangular box displaying the word 'Yes'. A yellow arrow points from the text 'Radio buttons' to the radio button options in the dialog box.

Radio buttons

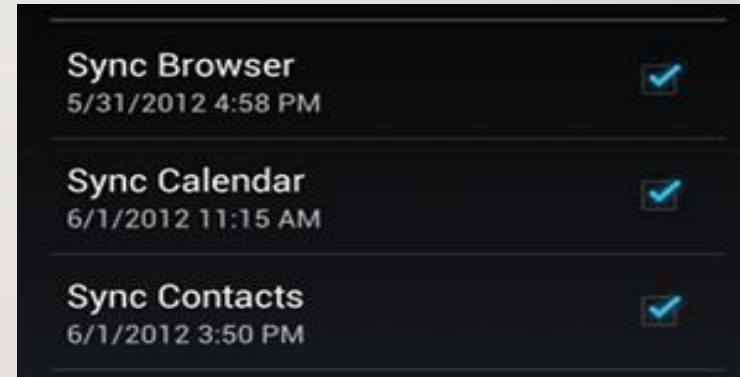
```
<RadioGroup
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">
    <RadioButton android:id="@+id/radio_btn1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/radBtn1"
        android:onClick="onRadioButtonClicked"/>
    <RadioButton android:id="@+id/radio_btn2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/radBtn2"
        android:onClick="onRadioButtonClicked"/>
</RadioGroup>
```

Can add
android:orientation="horizontal"
for horizontally laid out buttons


```
public void onRadioButtonClicked(View view) {  
    // Is the button now checked?  
    boolean checked = ((RadioButton) view).isChecked();  
  
    // Check which radio button was clicked  
    switch(view.getId()) {  
        case R.id.radio_btn1:  
            if (checked)  
                // radio btn1 is selected  
                break;  
        case R.id.radio_btn2:  
            if (checked)  
                // radio btn2 is selected  
                break;  
    }  
}
```

CHECK BOX BUTTON

- Checkboxes allow the user to select one or more options from a set.
- For each option create a checkbox
- Checkbox is managed separately
- and you must register a click listener for each

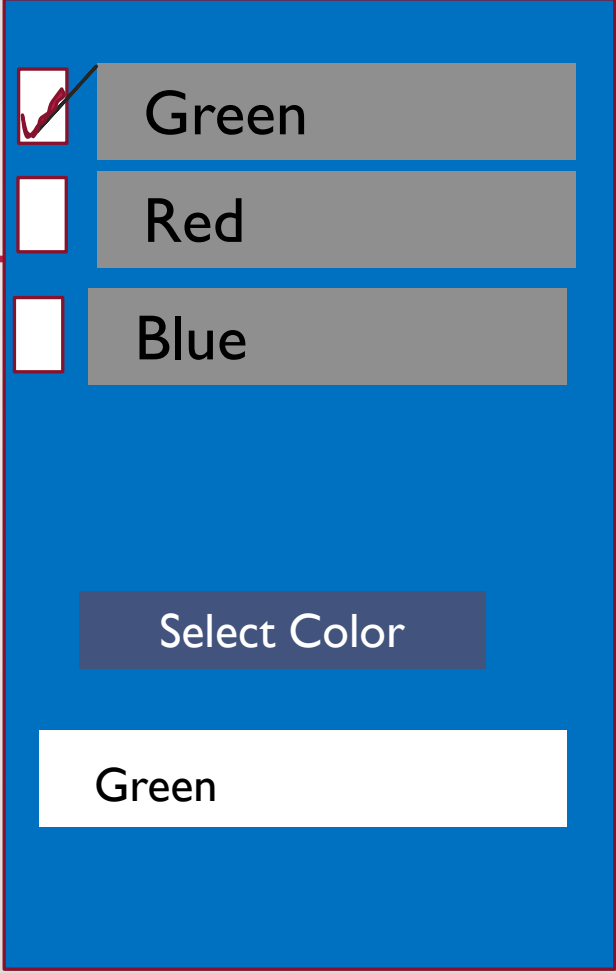


EXAMPLE FOR CHECK BOX

Build android app as shown in the figure

The app should do the following :-

- **Allow the user to select favorite color(s) (Using check boxes)**
- **display the selected colors when the button is clicked.**



The image shows a mobile application interface with a blue background. At the top, there are three rows, each consisting of a checkbox and a text label. The first row has a checked checkbox and the label 'Green'. The second row has an unchecked checkbox and the label 'Red'. The third row has an unchecked checkbox and the label 'Blue'. Below these rows is a dark blue button with the text 'Select Color'. At the bottom, there is a white rectangular box containing the text 'Green'. An arrow points from the label 'Checkboxes' to the first checkbox.

Checkboxes

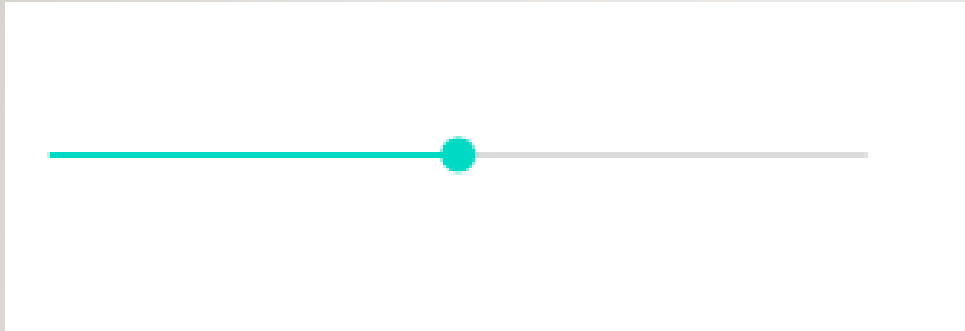
ADDING CHECK BOX IN XML

```
<CheckBox android:id="@+id/checkbox_green"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/green"
    android:onClick="onCheckboxClicked"/>
<CheckBox android:id="@+id/checkbox_red"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="@string/red"
    android:onClick="onCheckboxClicked"/>
```

RESPONDING TO CLICK EVENTS

```
public void onCheckboxClicked(View view) {  
    // Is the view now checked?  
    boolean checked = ((CheckBox) view).isChecked();  
  
    // Check which checkbox was clicked  
    switch(view.getId()) {  
        case R.id.checkbox_green :  
            if (checked)  
                // green is selected = true  
            else  
                // green is not selected  
            break;  
        case R.id.checkbox_red:  
            if (checked)  
                // red is selected = true  
            else  
                // red is not selected = false  
            break;  
        // TODO: check if blue is selected  
    }  
}
```


SEEK BAR



Type of ProgressBar with draggable thumb. The end user can drag the thumb left and right to move the progress of song, movie etc.

We can handle Seekbar vents using `SeekBar.OnSeekBarChangeListener` interface

```
SeekBar sb = findViewById(R.id.seekBar);

sb.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int i, boolean b) {
        // Toast.makeText(MainActivity.this, "Value = "+ i , Toast.LENGTH_LONG).show() ;
    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        Toast.makeText( context: MainActivity.this, text: "Value = "+ seekBar.getProgress() , Toast.LENGTH_LONG).show() ;
    }
});
```

LIST VIEW

```
<ListView  
    android:id="@+id/list_view"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent" />
```

Displays a vertically-scrollable collection of views

My Application	
Item 1	
Item 2	
Item 3	
Item 4	

```
String [] myArray = {"item 1" , "Item 2"  
    , "Item 3" , "Item 4"};  
ArrayAdapter adapter = new  
    ArrayAdapter<String>(this,  
  
    android.R.layout.simple_list_item_1,  
    myArray);  
  
ListView listView =  
    findViewById(R.id.list_view);  
listView.setAdapter(adapter);
```

REFERENCES

- <https://developer.android.com/guide/topics/ui/controls/togglebutton>
- <https://developer.android.com/guide/topics/ui/controls/radiobutton>
- <https://developer.android.com/guide/topics/ui/controls/spinner>
- <http://developer.android.com/index.html>