Android Controls

Android UI Layouts:

Containers:

A Layout defines the visual structure for a user interface.Depending upon the Requirement of UI Categorized into Five Standard Layouts. They are:

o Linear Layout

o Relative Layout

o Frame Layout

o Table layout

o Grid Layout

1. **Linear Layout:**

A Layout that arranges its children in a single column or a single row. The direction of the row can be set by calling [setOrientation()](http://developer.android.com/reference/android/widget/LinearLayout.html" \l "setOrientation(int)). You can also specify gravity, which specifies the alignment of all the child elements by calling [setGravity()](http://developer.android.com/reference/android/widget/LinearLayout.html" \l "setGravity(int)) or specify that specific children grow to fill up any remaining space in the layout by setting the weight member of [LinearLayout.LayoutParams](http://developer.android.com/reference/android/widget/LinearLayout.LayoutParams.html). The default orientation is horizontal.

* Linear Layout is a box model – widgets or child containers are lined up in a column or row.
* –Orientation:  horizontal or vertical
* –Fill model: wrap\_content or fill\_parent or dimensions
* –Weight: how to spit the available free space
* –Gravity: top, bottom, right, left, center, center\_vertical, center\_horizontal etc
* –Padding: increase space inner widgets

Example:

<http://androidexample.com/Linear_Layout_Basic-_Android_Example/index.php?view=article_discription&aid=72&aaid=96>

1. **Relative Layout:**

RelatvieLayout is a layout, where the position of children can be described relative to each other or to the parent.

**Positions Relative to Container:**    layout\_alignParentTop, layout\_alignParentBottom, layout\_alignParentLeft, layout\_alignParentRight, layout\_centerInParent etc

**Positions Relative to other child elements:**   layout\_above, layout\_below, layout\_toLeft, layout\_toRight, layout\_alignBaseline etc.

Cannot have circular dependency.

**Example:**

<http://androidexample.com/Relative_Layout_Basics_-_Android_Example/index.php?view=article_discription&aid=73&aaid=97>

1. **Frame Layout:**

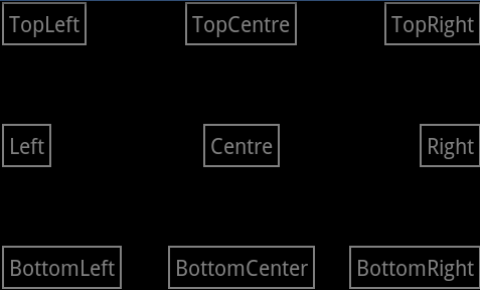
Frame layout is used to display a single view at a time. The view can contain many widgets but only one will appear at a time.

You can have multiple elements within a FrameLayout but each element will be positioned based on the top left of the screen.

Elements that overlap will be displayed overlapping

**Frame Layout** can become more useful when elements are hidden and displayed programmatically. You can use the attribute android:visibility in the XML to hide specific elements.

gravity: Top, bottom, right, left, center, center\_vertical, center\_horizontal and etc.



**Example:**

<?xml version="1.0" encoding="utf-8"?>

<FrameLayout

    android:layout\_width="fill\_parent"

    android:layout\_height="fill\_parent"

    xmlns:android="<http://schemas.android.com/apk/res/android>">

    <ImageView

        android:src="@drawable/onlymobilepro"

        android:scaleType="fitCenter"

        android:layout\_height="fill\_parent"

        android:layout\_width="fill\_parent"/>

    <TextView

        android:text="This is Frame Layout!!!"

        android:textSize="24px"

        android:textColor="#cc0000"

        android:layout\_height="fill\_parent"

        android:layout\_width="fill\_parent"

        android:gravity="top"/>

    <TextView

        android:text="Learn Android Development At onlyMobilePro.com"

        android:textSize="24px"

        android:textColor="#00dd00"

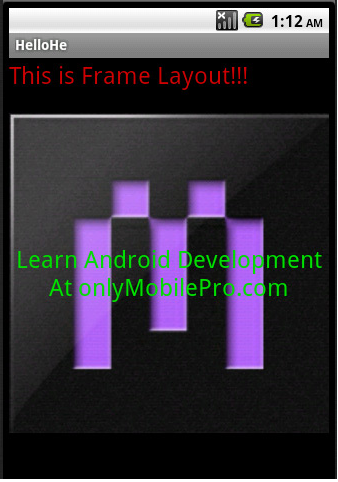
        android:layout\_height="fill\_parent"

        android:layout\_width="fill\_parent"

        android:gravity="center"/>

</FrameLayout>

**Output:**



**4.Table Layout:**

A layout that arranges its children into rows and columns. A TableLayout consists of a number of [TableRow](http://developer.android.com/reference/android/widget/TableRow.html) objects, each defining a row (actually, you can have other children, which will be explained below). TableLayout containers do not display border lines for their rows, columns, or cells. Each row has zero or more cells; each cell can hold one [View](http://developer.android.com/reference/android/view/View.html) object. The table has as many columns as the row with the most cells. A table can leave cells empty. Cells can span columns, as they can in HTML.

\_Layout Views using a grid of rows and columns.

\_Tables can span multiple rows and columns, and columns can be set to shrink or grow.

**Example:**

<http://androidexample.com/Table_Layout_-_Android_Example/index.php?view=article_discription&aid=74&aaid=98>

**5.Grid Layout:**

A layout that places its children in a rectangular grid. It is introduced from 4.0(Ice Cream Sandwich)

The number of rows and columns within the grid can be declared using the **android:rowCount** and **android:columnCount** properties.

**Example:**

http://www.informit.com/articles/article.aspx?p=2007353&seqNum=8

The **View** class represents the basic building for user interface components. A View instance occupies a rectangular are on the display, and it is responsible for drawing itself and handling events. The View components are used for both displaying information to the user and getting user input. In this section, we refer to these two View types as **output controls** and **input controls,** respectively.

**Input Controls:**

Input controls are the interactive components in your app's user interface. Android provides a wide variety of controls you can use in your UI, such as buttons, text fields, seek bars, check box, zoom buttons, toggle buttons, and many more.

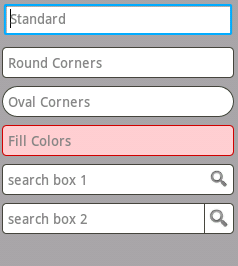
Input controls are the interactive view components that can enable the user to interact with the application by providing input, such as a button. The Android platform provides a variet;y of input controls that can be used in applications.

**List of input Controls :**

* EditText
* Buttons
* CheckBoxes
* Radio Buttons
* Toggle Buttons
* Spinners
* SeekBar
* VideoView
* [TimePicker](http://sampleprogramz.com/android/timepicker.php)
* [DatePicker](http://sampleprogramz.com/android/datepicker.php)
* [CalendarView](http://sampleprogramz.com/android/calendarview.php)

**EditText:**

A EditText is an overlay over TextView that configures itself to be editable. It is the predefined subclass of TextView that includes rich editing capabilities.



**Styles of edittext**

## EditText Attributes:

Following are the important attributes related to EditText control.

Inherited from **android.widget.TextView** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:autoText | If set, specifies that this TextView has a textual input method and automatically corrects some common spelling errors. |
| android:drawableBottom | This is the drawable to be drawn below the text. |
| android:drawableRight | This is the drawable to be drawn to the right of the text. |
| android:editable | If set, specifies that this TextView has an input method. |
| android:text | This is the Text to display. |

Inherited from **android.view.View** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

Example:

<EditText

android:id="@+id/edittext"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/button"

android:layout\_below="@+id/textView1"

android:layout\_marginTop="61dp"

android:ems="10"

android:text="@string/enter\_text" android:inputType="text" />

**AutoCompleteTextView:**

A AutoCompleteTextView is a view that is similar to EditText, except that it shows a list of completion suggestions automatically while the user is typing.

## **AutoCompleteTextView Attributes**

Following are the important attributes related to AutoCompleteTextView control. You can check Android official documentation for complete list of attributes and related methods which you can use to change these attributes are run time.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:completionHint | This defines the hint displayed in the drop down menu. |
| android:completionHintView | This defines the hint view displayed in the drop down menu. |
| android:completionThreshold | This defines the number of characters that the user must type before completion suggestions are displayed in a drop down menu. |
| android:dropDownAnchor | This is the View to anchor the auto-complete dropdown to. |
| android:dropDownHeight | This specifies the basic height of the dropdown. |
| android:dropDownHorizontalOffset | The amount of pixels by which the drop down should be offset horizontally. |
| android:dropDownSelector | This is the selector in a drop down list. |
| android:dropDownVerticalOffset | The amount of pixels by which the drop down should be offset vertically. |
| android:dropDownWidth | This specifies the basic width of the dropdown. |
| android:popupBackground | This sets the background. |

<AutoCompleteTextView

android:id="@+id/autoCompleteTextView1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignLeft="@+id/textView2"

android:layout\_below="@+id/textView2"

android:layout\_marginTop="54dp"

android:ems="10" />

**ImageButton:**

A ImageButton is a AbsoluteLayout which enables you to specify the exact location of its children. This shows a button with an image (instead of text) that can be pressed or clicked by the user.

## **ImageButton Attributes:**

Following are the important attributes related to ImageButton control. Inherited from **android.widget.ImageView** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:adjustViewBounds | Set this to true if you want the ImageView to adjust its bounds to preserve the aspect ratio of its drawable. |
| android:baseline | This is the offset of the baseline within this view. |
| android:baselineAlignBottom | If true, the image view will be baseline aligned with based on its bottom edge. |
| android:cropToPadding | If true, the image will be cropped to fit within its padding. |
| android:src | This sets a drawable as the content of this ImageView. |

Inherited from **android.view.View** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

## Example:

<ImageButton

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/imageButton"

android:layout\_centerVertical="true"

android:layout\_centerHorizontal="true"

android:src="@drawable/abc"/>

**CheckBox :**

A CheckBox is an on/off switch that can be toggled by the user. You should use check-boxes when presenting users with a group of selectable options that are not mutually exclusive.

It extends From CompoundButton Class, below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.TextView

↓

android.widget.Button

↓

android.widget.CompoundButton

↓

android.widget.Checkbox.

## CheckBox Attributes:

Following are the important attributes related to CheckBox control. You can check Android official documentation for complete list of attributes and related methods which you can use to change these attributes are run time.

Inherited from **android.widget.TextView** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:autoText | If set, specifies that this TextView has a textual input method and automatically corrects some common spelling errors. |
| android:drawableBottom | This is the drawable to be drawn below the text. |
| android:drawableRight | This is the drawable to be drawn to the right of the text. |
| android:editable | If set, specifies that this TextView has an input method. |
| android:text | This is the Text to display. |

Inherited from **android.view.View** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

## Example:

This example show CheckBox.

<CheckBox

android:id="@+id/checkBox1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Do you like Tutorials Point"

android:layout\_above="@+id/button"

android:layout\_centerHorizontal="true" />

Refferal Link:

<http://developer.android.com/reference/android/widget/CheckBox.html>

**Button:**

A Button is a Push-button which can be pressed, or clicked, by the user to perform an action.

**Button Attributes:**

Following are the important attributes related to Button control.

Inherited from **android.widget.TextView** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:autoText | If set, specifies that this TextView has a textual input method and automatically corrects some common spelling errors. |
| android:drawableBottom | This is the drawable to be drawn below the text. |
| android:drawableRight | This is the drawable to be drawn to the right of the text. |
| android:editable | If set, specifies that this TextView has an input method. |
| android:text | This is the Text to display. |

Inherited from **android.view.View** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

## **Example:**

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button"

android:id="@+id/button"

android:layout\_alignTop="@+id/editText"

android:layout\_alignLeft="@+id/textView1"

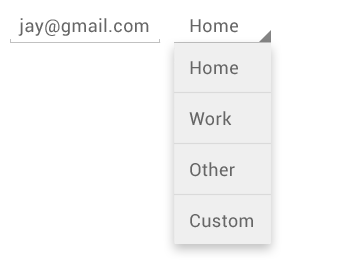
android:layout\_alignStart="@+id/textView1"

android:layout\_alignRight="@+id/editText"

android:layout\_alignEnd="@+id/editText" />

Spinners:

Spinners provide a quick way to select one value from a set. In the default state, a spinner shows its currently selected value. Touching the spinner displays a dropdown menu with all other available values, from which the user can select a new one.



You can add a spinner to your layout with the [Spinner](http://developer.android.com/reference/android/widget/Spinner.html) object. You should usually do so in your XML layout with a <Spinner> element. For example:

<Spinner android:id="@+id/planets\_spinner"

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"/>

To populate the spinner with a list of choices, you then need to specify a [SpinnerAdapter](http://developer.android.com/reference/android/widget/SpinnerAdapter.html) in your [Activity](http://developer.android.com/reference/android/app/Activity.html) or [Fragment](http://developer.android.com/reference/android/app/Fragment.html) source code.

<?xml version="1.0" encoding="utf-8"?>

<resources>

<string-arrayname="planets\_array">

<item>Mercury</item>

<item>Venus</item>

<item>Earth</item>

<item>Mars</item>

<item>Jupiter</item>

<item>Saturn</item>

<item>Uranus</item>

<item>Neptune</item>

</string-array>

</resources>

Spinner spinner = (Spinner) findViewById(R.id.spinner);// Create an ArrayAdapter using the string array and a default spinner layout

ArrayAdapter<CharSequence> adapter = ArrayAdapter.createFromResource(this,  
 R.array.planets\_array, android.R.layout.simple\_spinner\_item);// Specify the layout to use when the list of choices appears  
adapter.setDropDownViewResource(android.R.layout.simple\_spinner\_dropdown\_item);// Apply the adapter to the spinner  
spinner.setAdapter(adapter);

It extends from [AbsSpinner](http://developer.android.com/reference/android/widget/AbsSpinner.html)

implements [DialogInterface.OnClickListener](http://developer.android.com/reference/android/content/DialogInterface.OnClickListener.html)

Java.lang.object

↓

android.view.View

↓

android.widget.ViewGroup

↓

android.widget.AdapterView<android.widget.sppinnerAdapter>

↓

[android.widget.AbsSpinner](http://developer.android.com/reference/android/widget/AbsSpinner.html)

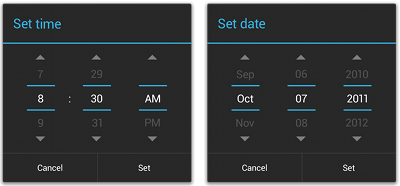
↓

android.widget.Spinner

Refferal Link : http://developer.android.com/reference/android/widget/Spinner.html

**Pickers:**

Android provides controls for the user to pick a time or pick a date as ready-to-use dialogs. Each picker provides controls for selecting each part of the time (hour, minute, AM/PM) or date (month, day, year). Using these pickers helps ensure that your users can pick a time or date that is valid, formatted correctly, and adjusted to the user's locale.



**TimePicker :** A widget for selecting the time of day, in either 24-hour or AM/PM mode.

It extends from FrameLayout

implements [DialogInterface.OnClickListener](http://developer.android.com/reference/android/content/DialogInterface.OnClickListener.html)

Java.lang.object

↓

android.view.View

↓

android.widget.ViewGroup

↓

android.widget.FrameLayout

↓

[android.widget.T](http://developer.android.com/reference/android/widget/AbsSpinner.html)imepicker

Interface: TimePicker.OnTimeChangedListener.

|  |  |  |
| --- | --- | --- |
| XML Attributes | | |
| Attribute Name | Related Method | Description |
| [android:timePickerMode](http://developer.android.com/reference/android/widget/TimePicker.html#attr_android:timePickerMode) |  | Defines the look of the widget. |

Refferal Link : <http://developer.android.com/reference/android/widget/TimePicker.html>

**DatePicker :** Provides a widget for selecting a date.

extends [FrameLayout](http://developer.android.com/reference/android/widget/FrameLayout.html)

Java.lang.object

↓

android.view.View

↓

android.widget.ViewGroup

↓

android.widget.FrameLayout

↓

[android.widget.Date](http://developer.android.com/reference/android/widget/AbsSpinner.html)picker

Interface: DatePicker.OnDateChangedListener.

|  |  |  |
| --- | --- | --- |
| XML Attributes | | |
| Attribute Name | Related Method | Description |
| [android:calendarTextColor](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:calendarTextColor) |  | The text color list of the calendar. |
| [android:calendarViewShown](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:calendarViewShown) |  | Whether the calendar view is shown. |
| [android:datePickerMode](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:datePickerMode) |  | Defines the look of the widget. |
| [android:dayOfWeekBackground](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:dayOfWeekBackground) |  | The background color for the header's day of week. |
| [android:dayOfWeekTextAppearance](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:dayOfWeekTextAppearance) |  | The text color for the header's day of week. |
| [android:endYear](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:endYear) |  | The last year (inclusive), for example "2010". |
| [android:firstDayOfWeek](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:firstDayOfWeek) | [setFirstDayOfWeek(int)](http://developer.android.com/reference/android/widget/DatePicker.html#setFirstDayOfWeek(int)) | The first day of week according to [Calendar](http://developer.android.com/reference/java/util/Calendar.html). |
| [android:headerBackground](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:headerBackground) |  | The background for the selected date header. |
| [android:headerDayOfMonthTextAppearance](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:headerDayOfMonthTextAppearance) |  | The text appearance for the day of month  (ex. |
| [android:headerMonthTextAppearance](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:headerMonthTextAppearance) |  | The text appearance for the month (ex. |
| [android:headerYearTextAppearance](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:headerYearTextAppearance) |  | The text appearance for the year (ex. |
| [android:maxDate](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:maxDate) |  | The maximal date shown by  this calendar view in mm/dd/yyyy format. |
| [android:minDate](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:minDate) |  | The minimal date shown by this calendar view in mm/dd/yyyy format. |
| [android:spinnersShown](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:spinnersShown) |  | Whether the spinners are shown. |
| [android:startYear](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:startYear) |  | The first year (inclusive), for example "1940". |
| [android:yearListItemTextAppearance](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:yearListItemTextAppearance) |  | The list year's text appearance in the list. |
| [android:yearListSelectorColor](http://developer.android.com/reference/android/widget/DatePicker.html#attr_android:yearListSelectorColor) |  | The list year's selected circle color in the list. |

**ToggleButton :**

ToggleButton and switch allow the user to change a setting between two states, such as on/off

Although all of these controls fundamentally do same thing. They provide different visual interfaces to suit different usecases.

ToggleButton has a button-like look and feel. The Switch is slider-based control.

ToggleButton displays checked/unchecked states as a button. It is basically an on/off button with a light indicator.

It extends From CompoundButton Class, below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.TextView

↓

android.widget.Button

↓

android.widget.CompoundButton

↓

android.widget.ToggleButton .

## ToggleButton Attributes:

Following are the important attributes related to ToggleButton control. You can check Android official documentation for complete list of attributes and related methods which you can use to change these attributes are run time.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:disabledAlpha | This is the alpha to apply to the indicator when disabled. |
| android:textOff | This is the text for the button when it is not checked. |
| android:textOn | This is the text for the button when it is checked. |

Inherited from **android.widget.TextView** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:autoText | If set, specifies that this TextView has a textual input method and automatically corrects some common spelling errors. |
| android:drawableBottom | This is the drawable to be drawn below the text. |
| android:drawableRight | This is the drawable to be drawn to the right of the text. |
| android:editable | If set, specifies that this TextView has an input method. |
| android:text | This is the Text to display. |

Inherited from **android.view.View** Class:

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

This example show ToggleButton.

<ToggleButton

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="On"

android:id="@+id/toggleButton"

android:checked="true"

android:layout\_below="@+id/imageButton"

android:layout\_toLeftOf="@+id/imageButton"

android:layout\_toStartOf="@+id/imageButton" />

<Switch

android:id="@+id/switch1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_centerInParent="true"

android:text="Switch" />

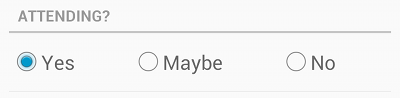


Referral Link:

http://developer.android.com/reference/android/widget/ToggleButton.html

**RadioButton :**

A RadioButton has two states: either checked or unchecked.This allows the user to select one option from a set.



This example show RadioButton.

<RadioButton

android:layout\_width="142dp"

android:layout\_height="wrap\_content"

android:text="JAVA"

android:id="@+id/radioButton"

android:textSize="25dp"

android:textColor="@android:color/holo\_red\_light"

android:checked="false"

android:layout\_gravity="center\_horizontal" />

It extends From CompoundButton Class, below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.TextView

↓

android.widget.Button

↓

android.widget.CompoundButton

↓

android.widget.RadioButton .

**RadioGroup :**

A RadioGroup class is used for set of radio buttons.

If we check one radio button that belongs to a radio group, it automatically unchecks any previously checked radio button within the same group.

These view class extends from LinearLayout. Its like a container for RadioButtons and below find hierarchy..

Java.lang.object

↓

android.view.View

↓

android.widget.ViewGroup

↓

android.widget.LinerLayout

↓

android.widget.RadioGroup

## **RadioGroup Attributes**

Following are the important attributes related to RadioGroup control. You can check Android official documentation for complete list of attributes and related methods which you can use to change these attributes are run time.

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:checkedButton | This is the id of child radio button that should be checked by default within this radio group. |

Inherited from **android.view.View** Class −

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:background | This is a drawable to use as the background. |
| android:contentDescription | This defines text that briefly describes content of the view. |
| android:id | This supplies an identifier name for this view, |
| android:onClick | This is the name of the method in this View's context to invoke when the view is clicked. |
| android:visibility | This controls the initial visibility of the view. |

This example show Radio Group.

<RadioGroup

android:layout\_width="fill\_parent"

android:layout\_height="90dp"

android:layout\_below="@+id/imageView"

android:layout\_marginTop="58dp"

android:weightSum="1"

android:id="@+id/radioGroup"

android:layout\_alignLeft="@+id/textView2"

android:layout\_alignStart="@+id/textView2"

android:layout\_alignRight="@+id/textView3"

android:layout\_alignEnd="@+id/textView3">

<RadioButton

android:layout\_width="wrap\_content"

android:layout\_height="55dp"

android:text="Male"

android:id="@+id/radioButton"

android:layout\_gravity="center\_horizontal"

android:checked="false"

android:textSize="25dp" />

<RadioButton

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Female"

android:id="@+id/radioButton2"

android:layout\_gravity="center\_horizontal"

android:checked="false"

android:textSize="25dp"

android:layout\_weight="0.13" />

</RadioGroup>

**Output Controls:**

The output Controls are View Subclasses used to presesnt information to the user. Depending on the type of the Vew Object, the displaed information can either be static , such as a text or interactive, such as a progress bar.

**List of output Controls :**

* TextView
* ImageView
* ListView
* [Gallery](http://sampleprogramz.com/android/gallery.php)View
* [Chronometer](http://sampleprogramz.com/android/chronometer.php)
* [Analog Clock](http://sampleprogramz.com/android/analogclock.php)
* [Digital Clock](http://sampleprogramz.com/android/digitalclock.php)
* ProgressBar.

**TextView**: A TextView displays text to the user and optionally allows them to edit it. A TextView is a complete text editor, however the basic class is configured to not allow editing.

**Example:**

<TextView

android:id="@+id/text\_id"

android:layout\_width="300dp"

android:layout\_height="200dp"

android:capitalize="characters"

android:text="hello\_world"

android:textColor="@android:color/holo\_blue\_dark"

android:textColorHighlight="@android:color/primary\_text\_dark"

android:layout\_centerVertical="true"

android:layout\_alignParentEnd="true"

android:textSize="50dp"/>

We can create dynamically by creating TextView object. It TextView Extends from View class like below:

Java.lang.Object

↓

[android.view.View](http://developer.android.com/reference/android/view/View.html)

↓

android.widget.TextView

**Sample Code:**

Textview textview=new TextView(Providing Activity context);

SetContentView( textview);

**The TextView supports a long list of attributes. The most notable ones are:**

|  |  |
| --- | --- |
| **Attribute** | **Description** |
| android:id | This is the ID which uniquely identifies the control. |
| android:capitalize | If set, specifies that this TextView has a textual input method and should automatically capitalize what the user types.   * Don't automatically capitalize anything - 0 * Capitalize the first word of each sentence - 1 * Capitalize the first letter of every word - 2 * Capitalize every character - 3 |
| android:cursorVisible | Makes the cursor visible (the default) or invisible. Default is false. |
| android:editable | If set to true, specifies that this TextView has an input method. |
| android:fontFamily | Font family (named by string) for the text. |
| android:gravity | Specifies how to align the text by the view's x- and/or y-axis when the text is smaller than the view. |
| android:hint | Hint text to display when the text is empty. |
| android:inputType | The type of data being placed in a text field. Phone, Date, Time, Number, Password etc. |
| android:maxHeight | Makes the TextView be at most this many pixels tall. |
| android:maxWidth | Makes the TextView be at most this many pixels wide. |
| android:minHeight | Makes the TextView be at least this many pixels tall. |
| android:minWidth | Makes the TextView be at least this many pixels wide. |
| android:password | Whether the characters of the field are displayed as password dots instead of themselves. Possible value either "true" or "false". |
| android:phoneNumber | If set, specifies that this TextView has a phone number input method. Possible value either "true" or "false". |
| android:text | Text to display. |
| android:textAllCaps | Present the text in ALL CAPS. Possible value either "true" or "false". |
| android:textColor | Text color. May be a color value, in the form of "#rgb", "#argb", "#rrggbb", or "#aarrggbb". |
| android:textColorHighlight | Color of the text selection highlight. |
| android:textColorHint | Color of the hint text. May be a color value, in the form of "#rgb", "#argb", "#rrggbb", or "#aarrggbb". |
| android:textIsSelectable | Indicates that the content of a non-editable text can be selected. Possible value either "true" or "false". |
| android:textSize | Size of the text. Recommended dimension type for text is "sp" for scaled-pixels (example: 15sp). |
| android:textStyle | Style (bold, italic, bolditalic) for the text. You can use or more of the following values separated by '|'.   * normal - 0 * bold - 1 * italic - 2 |
| android:typeface | Typeface (normal, sans, serif, monospace) for the text. You can use or more of the following values separated by '|'.   * normal - 0 * sans - 1 * serif - 2 * monospace - 3 |

**Known Direct Sub classes :**

[AppCompatTextView](http://developer.android.com/reference/android/support/v7/widget/AppCompatTextView.html), [Button](http://developer.android.com/reference/android/widget/Button.html), [CheckedTextView](http://developer.android.com/reference/android/widget/CheckedTextView.html), [Chronometer](http://developer.android.com/reference/android/widget/Chronometer.html), [DigitalClock](http://developer.android.com/reference/android/widget/DigitalClock.html), [EditText](http://developer.android.com/reference/android/widget/EditText.html), [RowHeaderView](http://developer.android.com/reference/android/support/v17/leanback/widget/RowHeaderView.html), [TextClock](http://developer.android.com/reference/android/widget/TextClock.html)

**Known Indirect Sub classes :**

[AppCompatAutoCompleteTextView](http://developer.android.com/reference/android/support/v7/widget/AppCompatAutoCompleteTextView.html), [AppCompatButton](http://developer.android.com/reference/android/support/v7/widget/AppCompatButton.html), [AppCompatCheckBox](http://developer.android.com/reference/android/support/v7/widget/AppCompatCheckBox.html), [AppCompatCheckedTextView](http://developer.android.com/reference/android/support/v7/widget/AppCompatCheckedTextView.html), [AppCompatEditText](http://developer.android.com/reference/android/support/v7/widget/AppCompatEditText.html),[AppCompatMultiAutoCompleteTextView](http://developer.android.com/reference/android/support/v7/widget/AppCompatMultiAutoCompleteTextView.html), [AppCompatRadioButton](http://developer.android.com/reference/android/support/v7/widget/AppCompatRadioButton.html), [AutoCompleteTextView](http://developer.android.com/reference/android/widget/AutoCompleteTextView.html), [CheckBox](http://developer.android.com/reference/android/widget/CheckBox.html), [CompoundButton](http://developer.android.com/reference/android/widget/CompoundButton.html), [ExtractEditText](http://developer.android.com/reference/android/inputmethodservice/ExtractEditText.html), [GuidedActionEditText](http://developer.android.com/reference/android/support/v17/leanback/widget/GuidedActionEditText.html),[MultiAutoCompleteTextView](http://developer.android.com/reference/android/widget/MultiAutoCompleteTextView.html), [RadioButton](http://developer.android.com/reference/android/widget/RadioButton.html), [SearchEditText](http://developer.android.com/reference/android/support/v17/leanback/widget/SearchEditText.html), [Switch](http://developer.android.com/reference/android/widget/Switch.html), [SwitchCompat](http://developer.android.com/reference/android/support/v7/widget/SwitchCompat.html), [ToggleButton](http://developer.android.com/reference/android/widget/ToggleButton.html)

**Refferal Link :**



**ImageView:**

The ImageView class allows displaying an image to the user, such as an icon. The Imageview Handles the scaling and tinting of the image in order to display it properly on the screen.

It extends From View Class,below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.ImageView

**Updating the Progress::**

During the runtime, the application can change the progress at any time, using the setProgress method, as shown below:

Updating the progress value of progressBar Object.

progressBar.setProgress(40);

**Chronometer:**

Class that implements a simple timer.

You can give it a start time in the [elapsedRealtime()](http://developer.android.com/reference/android/os/SystemClock.html#elapsedRealtime()) timebase, and it counts up from that, or if you don't give it a base time, it will use the time at which you call[start()](http://developer.android.com/reference/android/widget/Chronometer.html#start()). By default it will display the current timer value in the form "MM:SS" or "H:MM:SS", or you can use [setFormat(String)](http://developer.android.com/reference/android/widget/Chronometer.html#setFormat(java.lang.String)) to format the timer value into an arbitrary string.

It extends TextView , below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.TextView

↓

android.widget.Chronometer

Interface : [Chronometer.OnChronometerTickListener](http://developer.android.com/reference/android/widget/Chronometer.OnChronometerTickListener.html) ,A callback that notifies when the chronometer has incremented on its own.

|  |  |  |
| --- | --- | --- |
| XML Attributes | | |
| Attribute Name | Related Method | Description |
| [android:format](http://developer.android.com/reference/android/widget/Chronometer.html#attr_android:format) |  | Format string: if specified, the Chronometer will display this string,  with the first "%s" replaced by  the current timer value in "MM:SS" or "H:MM:SS" form. |

**Space:**

Space is a lightweight View subclass that may be used to create gaps between components in general purpose layouts.

It supports from API Level 23.

It extends From View Class, below find hierarchy.

Java.lang.object

↓

android.view.View

↓

android.widget.Space

Refferal Url:

http://developer.android.com/reference/android/widget/Space.html