Android Layout

An Android layout is a class that handles arranging the way its children appear on the screen.  Anything that is a View (or inherits from View) can be a child of a layout. All of the layouts inherit from ViewGroup (which inherits from View).  We could also create our own custom layout by making a class that inherits from ViewGroup.

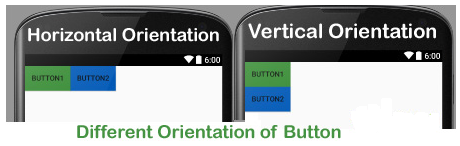
The standard Layouts are:

[LinearLayout](http://www.learn-android.com/2010/01/05/android-layout-tutorial/4/)  
[RelativeLayout](http://www.learn-android.com/2010/01/05/android-layout-tutorial/5/)

[FrameLayout](http://www.learn-android.com/2010/01/05/android-layout-tutorial/3/)  
[AbsoluteLayout](http://www.learn-android.com/2010/01/05/android-layout-tutorial/2/)  
[TableLayout](http://www.learn-android.com/2010/01/05/android-layout-tutorial/6/)

Linear Layout with examples

Linear layout is a simple layout used in android for layout designing. In the Linear layout all the elements are displayed in linear fashion means all the childs/elements of a linear layout are displayed according to its orientation. The value for orientation property can be either horizontal or vertical.



**Table Of Contents**

* [Types Of Linear Layout Orientation](http://abhiandroid.com/ui/linear-layout#Types_Of_Linear_Layout_Orientation)
* [Main Attributes In Linear Layout](http://abhiandroid.com/ui/linear-layout#Main_Attributes_In_Linear_Layout)
* [Example of Linear Layout](http://abhiandroid.com/ui/linear-layout#Example_of_Linear_Layout)

**Types of Linear Layout Orientation**

There are two types of linear layout orientation:

1. Vertical
2. Horizontal

As the name specified these two orientations are used to arrange there child one after the other, in a line, either vertically or horizontally. Let’s we describe these in detail.

1. **Vertical:**

In this all the child are arranged vertically in a line one after the other. In below code snippets we have specified orientation “vertical” so the childs/views of this layout are displayed vertically.

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

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical"><!-- Vertical Orientation set -->

<!-- Child Views(In this case 2 Button) are here -->

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button1"

android:id="@+id/button"

android:background="#358a32" />

<Button

android:layout\_width="wrap\_content"

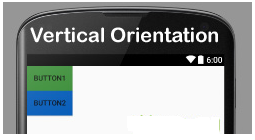
android:layout\_height="wrap\_content"

android:text="Button2"

android:id="@+id/button2"

android:background="#0058b6" />

</LinearLayout>



**2. Horizontal:**

In this all the child are arranged horizontally in a line one after the other. In below code snippets we have specified orientation “horizontal” so the childs/views of this layout are displayed horizontally.

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

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"><!-- Horizontal Orientation set -->

<!-- Child Views(In this case 2 Button) are here -->

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button1"

android:id="@+id/button"

android:background="#358a32" />

<Button

android:layout\_width="wrap\_content"

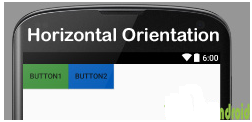
android:layout\_height="wrap\_content"

android:text="Button2"

android:id="@+id/button2"

android:background="#0058b6" />

</LinearLayout>



**Important Note:** All of the layout managers can be nested. This means that you can put a [Relative Layout](http://abhiandroid.com/ui/relative-layout) or [Frame Layout](http://abhiandroid.com/ui/framelayout) as a child to Linear Layout.

**Main Attributes In Linear Layout:**

Now let’s  we discuss about the attributes that helps us to configure a linear layout and its child controls. Some of the most important attributes you will use with linear layout [include](http://abhiandroid.com/ui/include-merge-tag):

1. **orientation:** The orientation attribute used to set the childs/views horizontally or vertically. In Linear layout default orientation is vertical.

**Example:  Orientation vertical:**

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

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical"><!-- Vertical Orientation set -->

<!-- Put Child Views like Button here -->

</LinearLayout>

**Example: Orientation Horizontal:**

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

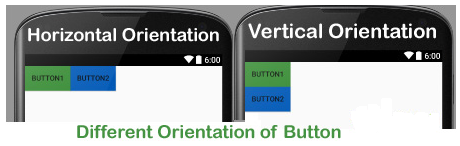
android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"><!-- Horizontal Orientation set -->

<!-- Child Views are here -->

</LinearLayout>



1. **gravity:** The gravity attribute is an optional attribute which is used to control the alignment of the layout like left, right, center, top, bottom etc.

**Example: We have set gravity right for linear layout. So the buttons gets align from right side in Horizontal orientation.**

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="right"

android:orientation="horizontal">

<!--Button Child View Here--->

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button2"

android:id="@+id/button2"

android:background="#0e7d0d" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Button1"

android:id="@+id/button"

android:background="#761212" />

</LinearLayout>



1. **layout\_weight:** The layout weight attribute specify each child control’s relative importance within the parent linear layout.

**Example: weight property for**[**button**](http://abhiandroid.com/ui/button)**in linear layout. In the below example one button is of weight 2 and other is of weight 1.**

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

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<!--Add Child View Here--->

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Weight 2"

android:background="#761212"

android:layout\_margin="5dp"

android:id="@+id/button"

android:layout\_weight="2" />

<Button

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:background="#761212"

android:layout\_margin="5dp"

android:layout\_weight="1"

android:text="Weight 1" />

</LinearLayout>

**In the layout image you can notice Button with weight 2 gets more size related the other.**



1. **weightSum:** weightSum is the sum up of all the child attributes weight. This attribute is required if we define weight property of the childs.

**Example: In the same above example of weight, we can define weightSum value 3.**

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:weightSum="3"

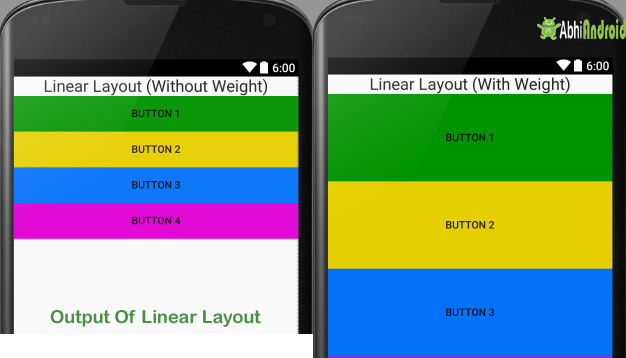
android:orientation="horizontal">

<!--Add Child View Here--->

</LinearLayout>

**Example of Linear Layout:**

Now lets design 2 linear layout UI. First we have designed using weight attribute and second without using it. So below layout output will clear the difference between them:



**Example 1:** First we will design Android Linear Layout without using weight property

In this example we have used one [TextView](http://abhiandroid.com/ui/textview" \t "_self) and 4 Button. The orientation is set to vertical.

**Below is the code of activity\_main.**[**xml**](http://abhiandroid.com/ui/xml)

<!-- Vertical Orientation is set -->

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<!-- Text Displayed At Top -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="Linear Layout (Without Weight)"

android:id="@+id/textView"

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

<!-- Button Used -->

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 1"

android:background="#009300" />

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 2"

android:background="#e6cf00" />

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 3"

android:background="#0472f9" />

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 4"

android:background="#e100d5" />

</LinearLayout>

**Output Screen:**



**Example 2:** In this example of linear layout we have used weight property.

**Below is the code of activity\_main.**[**xml**](http://abhiandroid.com/ui/xml)**with explanation included**

<!-- Vertical Orientation is set with weightSum-->

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

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:weightSum="5"

android:orientation="vertical">

<!-- Text Displayed At Top -->

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:textAppearance="?android:attr/textAppearanceLarge"

android:text="Linear Layout (With Weight)"

android:id="@+id/textView"

android:layout\_gravity="center\_horizontal"

android:layout\_weight="0"/>

<!-- Button Used -->

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 1"

android:background="#009300"

android:layout\_weight="1"/>

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 2"

android:background="#e6cf00"

android:layout\_weight="1"/>

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 3"

android:background="#0472f9"

android:layout\_weight="1"/>

<Button

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Button 4"

android:background="#e100d5"

android:layout\_weight="1"/>

</LinearLayout>

**Output Screen:**

