Day 6: Linear Layout

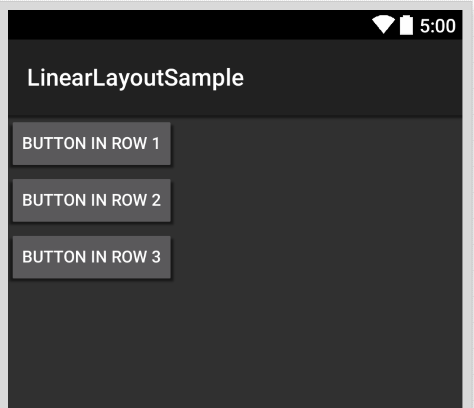
Linear Layout in Android is used to put Views on a screen in linear fashion either Horizontally or Vertically.

Let’s start digging into the XML code that is used to create LinearLayout with a really basic example.

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  android:orientation="vertical">  <Button  android:id="@+id/button1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button in Row 1" />  <Button  android:id="@+id/button2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button in Row 2" />  <Button  android:id="@+id/button3"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Button in Row 3" />  </LinearLayout> |

Gist file: <https://gist.github.com/vkoppaka/bd3cca1a422f71d76796>

If you notice the above snippet carefully, you will see that we have a <LinearLayout> XML node that we are using to specify as ViewGroup and three Buttons (Views) are placed inside the LinearLayout. The most important property for the Linear Layout is the “**android:orientation**” property which we currently set to “vertical”. This defines the linear flow of the layout. So, going by the example, the three buttons on the screen will be placed vertically, one after the other. Here is how the UI looks in designer.

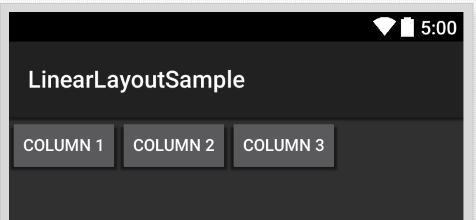


Now if I were to switch the orientation property of the Linear Layout to “horizontal” the three buttons would be placed horizontally, one next to the other.

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  android:orientation="horizontal">  <Button  android:id="@+id/button1"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Column 1" />  <Button  android:id="@+id/button2"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Column 2" />  <Button  android:id="@+id/button3"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:text="Column 3" />  </LinearLayout> |

Gist file: <https://gist.github.com/vkoppaka/e7cad7e2a30d77258462>

And the UI would look like:



Now, you might be wondering what the rest of the XML Attributes like layout\_width, layout\_height, id, text all mean. We will cover individual view attributes when we cover different Views in android but let’s explore layout\_width and layout\_height.

## Layout Options

The layout\_width and layout\_height are used to specify width and height of the layout (ViewGroup).

You can specify the height and width using any of the predefined units like px (pixels), dp(density-independent pixels), sp(scaled pixels), in (inches), mm (millimters). SP and DP are the most commonly used units.

There are constant values that you can set height and width properties to, which are –

1. match\_parent
2. wrap\_content

### match\_parent

match\_parent indicates a layout element should be as its parent, and if the layout element is the root, then it would be as big as the device height and width. One thing to note here is padding set for the layout will be deducted in the height and width calculations when match\_parent is set.

### wrap\_content

wrap\_content indicates that a layout element should only be as big as it needs to enclose its content plus padding.

That’s it for today. Tomorrow we will explore more Layout options.

Additional Note:

## AXML Intellisense in Visual Studio

If you are having any trouble getting XML intellisense for the AXML files in Visual Studio, please follow the steps below.

1. Open an (any) .axml file
2. From the menu bar, select XML -> Schemas...
3. In the XML Schemas dialog, click on Add
4. Navigate to C:\Program Files (x86)\MSBuild\Xamarin\Android
5. Select android-layout-xml.xsd and click on Open.
6. Click on OK.

Venkata