Day 8: Grid and Table Layout

# Grid Layout

Grid layout is one of the new kids in the block, it got introduced with Ice Cream Sandwich (API Level 14). If you are a .NET Programmer who is familiar with Windows Phone (or Windows’) Grid Definitions, this should feel right at home to you.

Grid Layout is used to put views into rows and columns reminiscent of tables. As with any ViewGroup Grid Layout can also hold other types of ViewGroups.

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

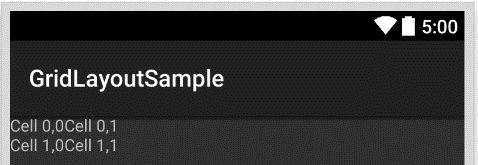
|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:rowCount="2"  android:columnCount="2">  <TextView  android:text="Cell 0,0"  android:layout\_row="0"  android:layout\_column="0" />  <TextView  android:text="Cell 0,1"  android:layout\_row="0"  android:layout\_column="1" />  <TextView  android:text="Cell 1,0"  android:layout\_row="1"  android:layout\_column="0" />  <TextView  android:text="Cell 1,1"  android:layout\_row="1"  android:layout\_column="1" />  </GridLayout> |

Gist file: <https://gist.github.com/vkoppaka/14811a86ccfff6cab45e>

In the above snippet we see a number of new android: attributes specifically starting with layout\_. Let’s explore them –

1. **rowCount**: Row Count tells Android on how many rows does the current Grid Layout has.
2. **columnCount**: Column Count tells Android on how many columns does the current Grid Layout has.
3. **layout\_row**: Specifies the row position of the view in the Grid Layout.
4. **layout\_column**: specifies the column position of the view in the Grid Layout.

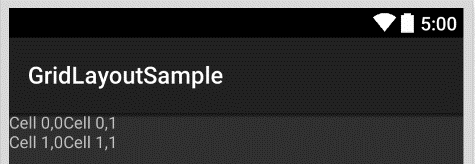
If we were to run the application and see you will a 2X2 table layout –



You can also “skip” specifying the layout\_row and layout\_column properties and Android is smart enough to place the cells in the correct order. Let’s see that in action –

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:rowCount="2"  android:columnCount="2">  <TextView  android:text="Cell 0,0" />  <TextView  android:text="Cell 0,1" />  <TextView  android:text="Cell 1,0" />  <TextView  android:text="Cell 1,1" />  </GridLayout> |

In the above snippet we are not really specifying the row and column properties but if you were to just run the application, Android puts them in the right places.



## Orientation Property

Another important property for Grid Layout is the Orientation Property which is set using android:orientation attribute. You can switch between “horizontal” and “vertical” as options and the layout is put on the screen going horizontal or vertical way for the next cell respectively.

## RowSpan and ColumnSpan

As you would expect from any Grid layout system, there are options to specify the “span” of each cell. This is achieved again by attributes, and the attributes are –

android:rowSpan: Specifies the number of rows the cell has to span.

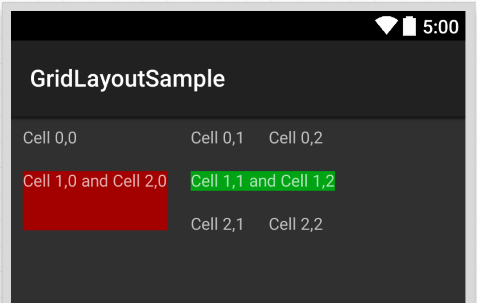
android:columnSpan: Specifies the number of columns the cell has to span.

Let’s take a look at an example –

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <GridLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent"  android:rowCount="3"  android:columnCount="3">  <TextView  android:text="Cell 0,0"  android:layout\_row="0"  android:layout\_column="0"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 0,1"  android:layout\_row="0"  android:layout\_column="1"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 0,2"  android:layout\_row="0"  android:layout\_column="2"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 1,0 and Cell 2,0"  android:background="#a30000"  android:layout\_height="50dp"  android:layout\_row="1"  android:layout\_column="0"  android:layout\_rowSpan="2"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 1,1 and Cell 1,2"  android:background="#00a313"  android:layout\_row="1"  android:layout\_column="1"  android:layout\_columnSpan="2"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 2,1"  android:layout\_row="2"  android:layout\_column="1"  android:layout\_margin="10dp" />  <TextView  android:text="Cell 2,2"  android:layout\_row="2"  android:layout\_column="2"  android:layout\_margin="10dp" />  </GridLayout> |

Gist file link: <https://gist.github.com/vkoppaka/77c9a66cb5730d1fbc36>

And if you run this application, your UI would look like –



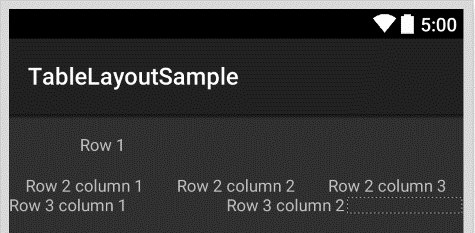
# Table Layout:

Table Layout is pretty similar to Grid Layout in terms of display. Grid Layout is more versatile in memory management. Let’s look at an example of Table Layout

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <TableLayout xmlns:android="http://schemas.android.com/apk/res/android"  android:layout\_width="match\_parent"  android:layout\_height="match\_parent">  <TableRow  android:layout\_height="wrap\_content"  android:layout\_width="fill\_parent"  android:gravity="center\_horizontal">  <TextView  android:layout\_width="match\_parent"  android:layout\_height="wrap\_content"  android:text="Row 1"  android:layout\_span="3"  android:padding="18dip" />  </TableRow>  <TableRow  android:id="@+id/tableRow1"  android:layout\_height="wrap\_content"  android:layout\_width="match\_parent">  <TextView  android:id="@+id/TextView04"  android:text="Row 2 column 1"  android:layout\_weight="1"  android:gravity="center" />  <TextView  android:id="@+id/TextView04"  android:text="Row 2 column 2"  android:layout\_weight="1"  android:gravity="center" />  <TextView  android:id="@+id/TextView04"  android:text="Row 2 column 3"  android:layout\_weight="1"  android:gravity="center" />  </TableRow>  <TableRow  android:layout\_height="wrap\_content"  android:layout\_width="fill\_parent"  android:gravity="center\_horizontal">  <TextView  android:id="@+id/TextView04"  android:text="Row 3 column 1"  android:layout\_weight="1" />  <TextView  android:id="@+id/TextView04"  android:text="Row 3 column 2" />  </TableRow>  </TableLayout> |

Gist file link: <https://gist.github.com/vkoppaka/8a529463a672c6829ddf>

And if you were to run the application, you will see a Tabular layout –



That’s it for today. See you all tomorrow.

Venkata