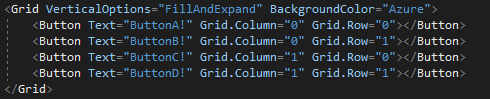
Soft 262 Task 1: Grid tutorial

**How to lay out controls in a grid:**

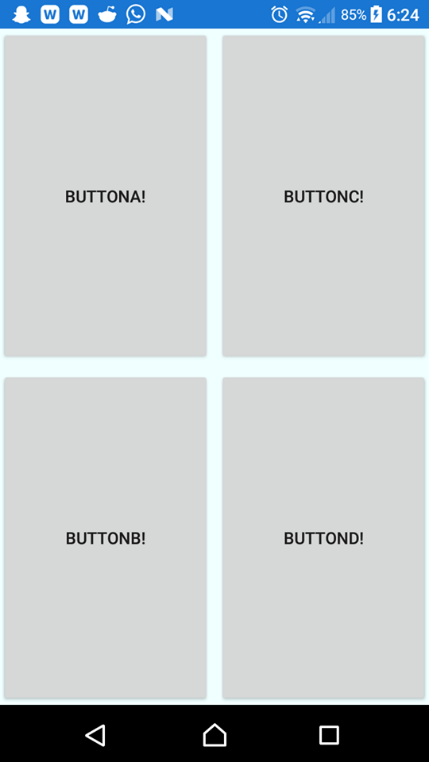
<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.grid?view=xamarin-forms>

The best way to layout your apps controls within a grid is by defining what row and column you want each control to be within using the Grid.Column and Grid.Row property on each control.

Example:



This produces the following:



This method is easily expandable, and you can essentially as many rows and columns as you want. You can also have controls span more than one row/ column as shown in the next part of the tutorial.

**How to span columns and rows:**

<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.grid.columnspanproperty?view=xamarin-forms>

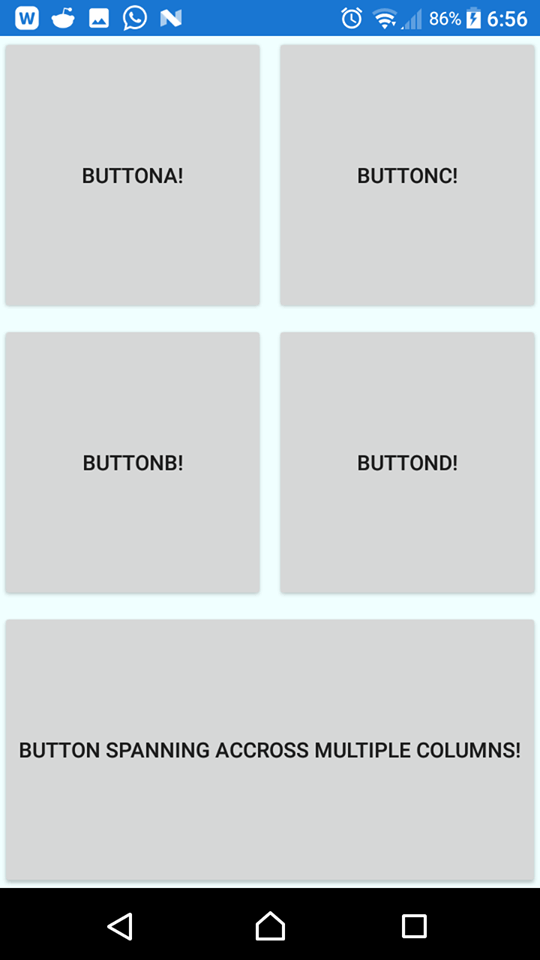
<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.grid.rowspanproperty?view=xamarin-forms>

In order to get a control to span across several rows/ columns you can simply use the Grid.ColumnSpan or Grid.RowSpan property, you can set there’s to span across as many rows/ columns as you want.

Example:



This produces the following:



**Using Margin and Padding for layout**

<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.layout.padding?view=xamarin-forms#Xamarin_Forms_Layout_Padding>

<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.view.margin?view=xamarin-forms#Xamarin_Forms_View_Margin>

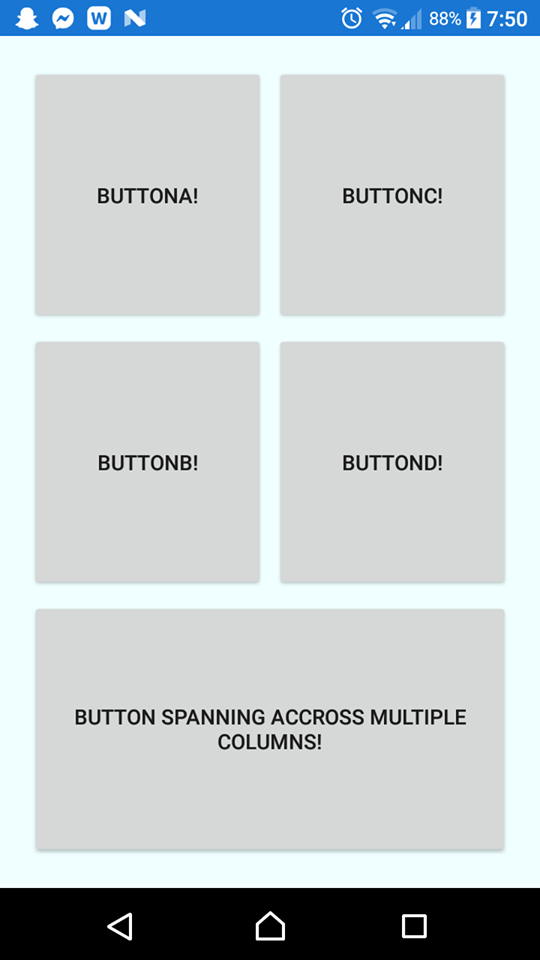
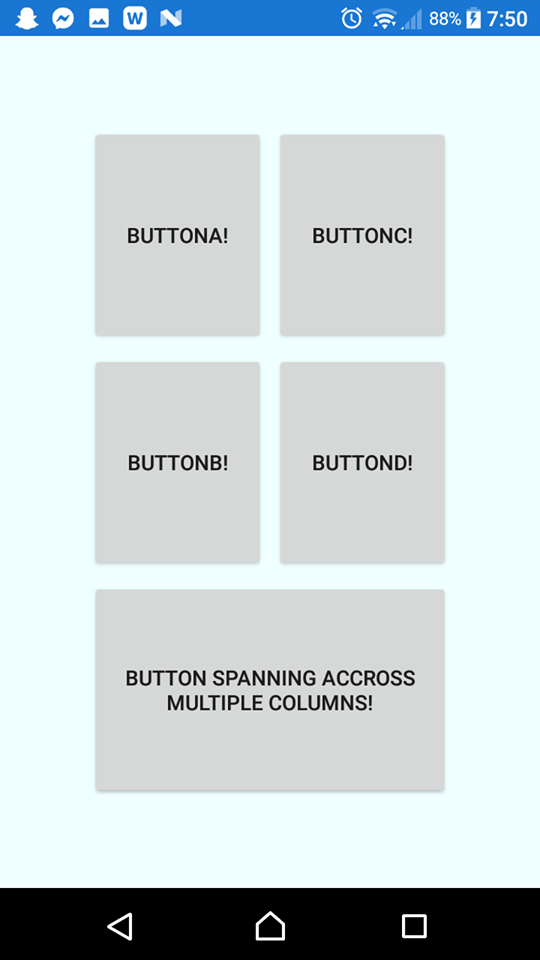
Padding and margins can be used to add spacing within and outside of containers and controls, for example padding can be added to a grid to add a gap from the edge of the screen (assuming the grid is the size of the screen) to whatever’s within it.

To add padding to an element simply add the padding property to it with how thick you want the padding

Example:



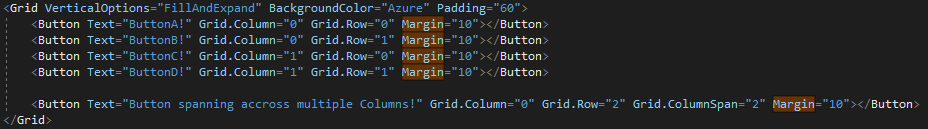
This produces the following:



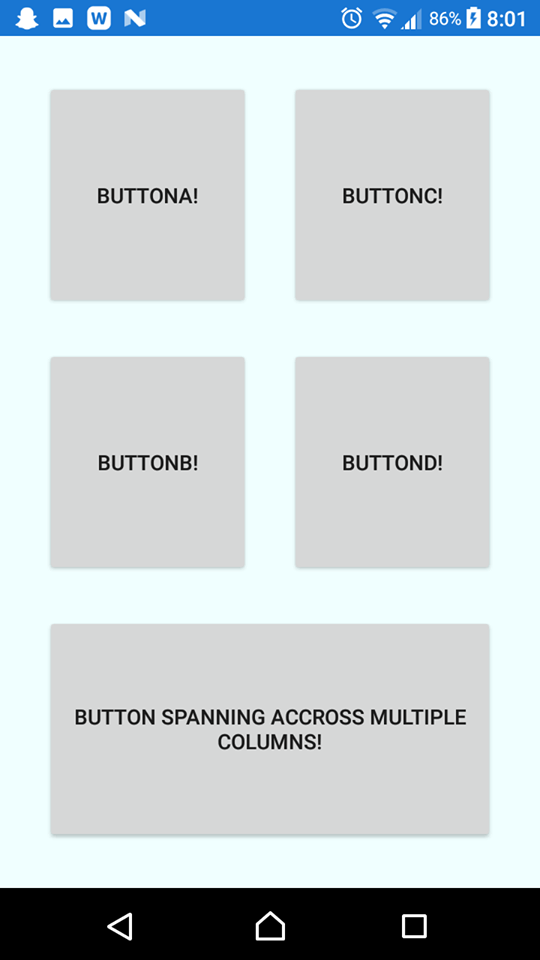
Padding of 60 to help show the effects

Where padding adds a gap within the element margin adds a gap to the outside of it, by adding a margin to each button there will be a greater gap in-between each of them.

Example:



This produces the following:



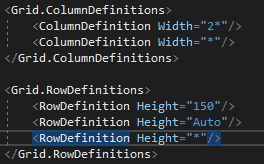
A margin of 10 has been added to each button, this will be reduced in the rest of the tutorial down to 2

**Controlling row height and column width**

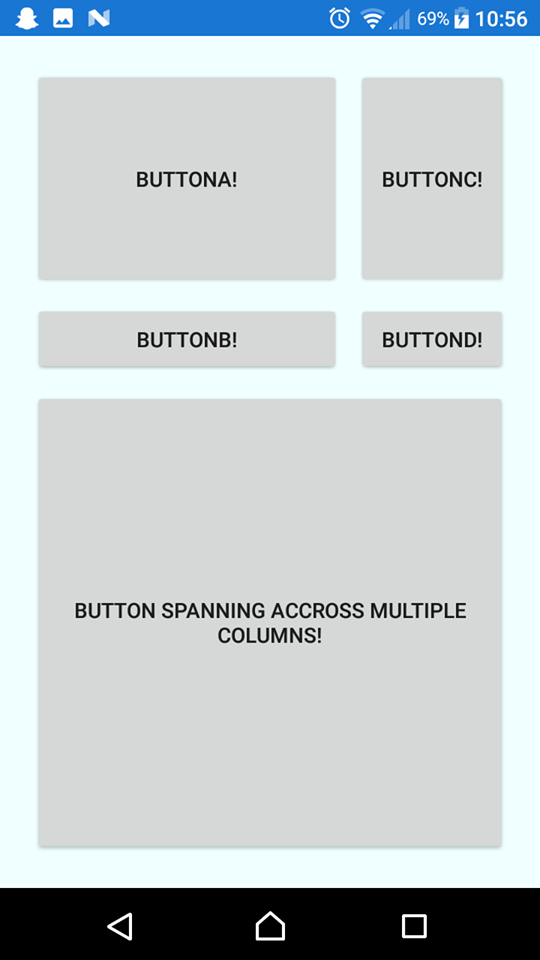
<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.rowdefinition?view=xamarin-forms>

To control the row and columns height and width you can predefine them in a grid definition. You can set them three different types, a absolute value, automatic value, or a star value. The absolute is a set height/width according to the number you use, automatic will choose a size to fit all the children within the row/column, and a star value is a proportional weight, so if one row has “2\*” and another has “\*”, the first will be twice the size as the latter.

Example



This produces the following:



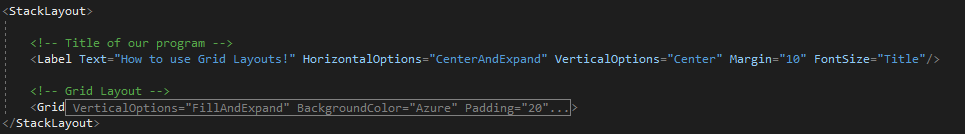
From the code you can see for the column definitions, both columns are using the star type for their width, the first of which is 2\* making it twice the width as the other.

For the Row definitions, the first is of set height 150, which will never change, the next is auto, so it’ll only take up as much space as it needs, and the finial is of height \*, which will automatically fill the rest of the space.

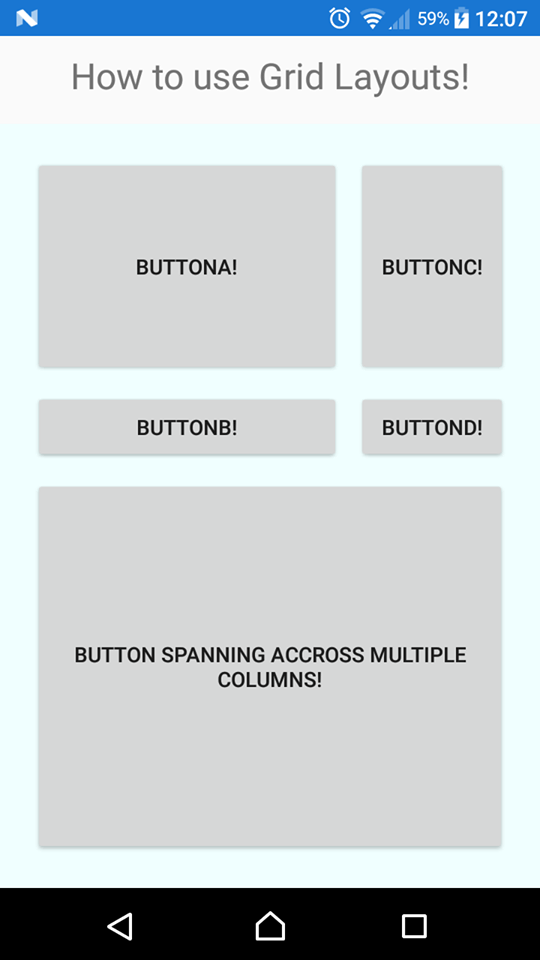
**Nesting within other layouts**

A grid layout can be nested within other layouts to help create your UI design, in the following example the grid is placed within a stack layout to help add a title to our program.

Example:



This produces the following:



**GitHub link**

<https://github.com/wmastersPlym/SOFT262Test/tree/master/Task1/SOFT262Task1>

**Microsoft documentation**

<https://docs.microsoft.com/en-us/dotnet/api/xamarin.forms.grid?view=xamarin-forms>