## UWP-011 - Laudable Layout Challenge

The challenges I created on LearnVisualStudio.NET are easily the most popular feature of the site. The challenges force you to learn by doing, to exercise what you just learned in the previous lessons. This will force you to struggle a bit with these new concepts. And struggle is good. It's how you learn. No struggle, no learning.

It might take you 30 minutes, an hour. You might have to go back and review some lessons or reference MSDN for some help but I think you can do this based on what we've already learned so far. There's actually three challenges in a row and they're all related to the topic of layout what we've already learned.

The first challenge I'm calling the “Laudable Layout Challenge”. If you can see that we're building a simple app -- only the user interface. So there’s no real functionality in this app whatsoever.

First of all, this example requires a single grid only. That's part of the requirements we'll talk about them in a moment.


And you can see that we're just capturing in TextBoxes the first name, last name, and the email address for a sales prospect and then we have a save button and again, it doesn't do anything at all.

The first column holds the TextBlocks, the second column holds the TextBoxes and Button, the third column is used for padding. You can also see there's a number of rows but I'm going to let you figure out how many to use.

See the document in the zipped folder associated with this lesson called UWP-11-Instructions.txt.

First of all, you can only use a grid control.

Secondly, this is more of a tip, whenever I was designing the application, I used the designer of a five-inch phone 1920 by 1080 300% scale.

The large TextBlock at the top is 48 points.

Most of the margins that you see here on the left and here between this top TextBlock and all the rest of the TextBoxes are either 10 pixels or 20 pixels ... again, I'll let you figure that out.

And then the TextBlocks for the first name, last name and email address should be centered vertically in the row. Presumably, this is a row here at the very top and you can see that the TextBox is butted up at what appears to be the top of the row and then there's some spacing below it and if you were to use your mouse cursor and trace along, you can see that the TextBlock in the left-most column is centered between the top line of the TextBox and the start of the next TextBox in the next row.

The tricky part: requirement number six. You're going to need to figure out how to allow your TextBlock to span multiple columns. When you have a need to tell the grid how to do something, what do you use? Use an attached property, so you should use IntelliSense to figure out the correct attached property to make your TextBlocks span multiple columns.

If you ever get stuck, you can take a look at a screenshot that I provide as guidance: UWP-011-Screenshot.png.

You should attempt to solve this without my help whatsoever. If you get stuck, I don't want you to struggle too much and get frustrated. Only review enough of the next lesson to get unstuck and then continue to move forward on your own to the extent that it's possible.

And then after you finish and successfully complete the challenge, you will not only feel victory but you will also then want to review the solution lesson in total, so that you can compare the way that you approached this challenge versus the way that I approached it. Maybe you have a more elegant way of approaching something than I did. Maybe I use a different technique than you did so you can compare and contrast and hopefully by doing that, learn a little bit more.

Don't cheat, don't go on directly to the next lesson. Make sure you struggle by attempting to solve this challenge first. Spend a half hour, an hour maybe, if it takes that. Re-read the lesson, or re-watch the videos if you have to.

## UWP-012 - Laudable Layout Challenge: Solution

If you're reading this immediately after reading the previous lesson and you've not cracked open Visual Studio and written a single line of code yet, then stop what you're doing. Shame on you! Open up Visual Studio, do this for yourself. Nobody learns by just reading books or watching videos. You must get your hands "dirty in the code". That's how you learn. So, stop reading this lesson. Go off and attempt to solve the challenge. If you get stuck, come back here and read just enough of this lesson to get unstuck.

Step 1: Create a new "Blank App (Universal Windows)" project and name it LaudableLayout.

Step 2: Add RowDefinitions and ColumnDefinitions into the default Grid:

```
11          <Grid.RowDefinitions>
12              <RowDefinition Height="Auto" />
13              <RowDefinition Height="Auto" />
14              <RowDefinition Height="Auto" />
15              <RowDefinition Height="Auto" />
16              <RowDefinition Height="Auto" />
17              <RowDefinition Height="*" />
18          </Grid.RowDefinitions>
19          <Grid.ColumnDefinitions>
20              <ColumnDefinition Width="2*" />
21              <ColumnDefinition Width="3*" />
22              <ColumnDefinition Width="1*" />
23          </Grid.ColumnDefinitions>
24      </Grid>
25  </Page>
```

Step 3: Add the TextBlocks, TextBoxes and Button to the Grid's Children collection, assigning each one to the appropriate Row or Column:

```
        <TextBlock Text="ACME Sales Corp" FontSize="48" Grid.ColumnSpan="3" />

        <TextBlock Grid.Row="1" Text="First Name: " />
        <TextBox Grid.Row="1" Grid.Column="1" />

        <TextBlock Grid.Row="2" Text="Last Name: " />
        <TextBox Grid.Row="2" Grid.Column="1" />

        <TextBlock Grid.Row="3" Text="Email: " />
        <TextBox Grid.Row="3" Grid.Column="1" />

        <Button Grid.Row="4" Grid.Column="1" Content="Save" />

    </Grid>
</Page>
```

Notice the Grid.ColumnSpan Attached Property applied to the top-most TextBlock.

Step 4: Add Margins to the Grid and the XAML Controls

```
    <Grid Margin="10,0,0,0" Background="{ThemeResource ApplicationPageBackgroundThemeBrush}">
        <Grid.RowDefinitions...>
        <Grid.ColumnDefinitions...>

        <TextBlock Text="ACME Sales Corp" FontSize="48" Grid.ColumnSpan="3" Margin="0,0,0,20" />

        <TextBlock Grid.Row="1" Text="First Name: " />
        <TextBox Grid.Row="1" Grid.Column="1" Margin="0,0,0,10" />

        <TextBlock Grid.Row="2" Text="Last Name: " />
        <TextBox Grid.Row="2" Grid.Column="1" Margin="0,0,0,10" />

        <TextBlock Grid.Row="3" Text="Email: " />
        <TextBox Grid.Row="3" Grid.Column="1" Margin="0,0,0,10" />

        <Button Grid.Row="4" Grid.Column="1" Content="Save" />

    </Grid>
</Page>
```

Step 5: Align TextBlocks vertically to the Center.

```
        <TextBlock Text="ACME Sales Corp" FontSize="48" Grid.ColumnSpan="3" Margin="0,0,0,20" />

        <TextBlock Grid.Row="1" Text="First Name: " VerticalAlignment="Center" />
        <TextBox Grid.Row="1" Grid.Column="1" Margin="0,0,0,10" />

        <TextBlock Grid.Row="2" Text="Last Name: " VerticalAlignment="Center" />
        <TextBox Grid.Row="2" Grid.Column="1" Margin="0,0,0,10" />

        <TextBlock Grid.Row="3" Text="Email: " VerticalAlignment="Center" />
        <TextBox Grid.Row="3" Grid.Column="1" Margin="0,0,0,10" />

        <Button Grid.Row="4" Grid.Column="1" Content="Save" />

    </Grid>
</Page>
```

Please note: If this is something you struggled with then maybe you should come back tomorrow and try it again just to cement these ideas in your mind.