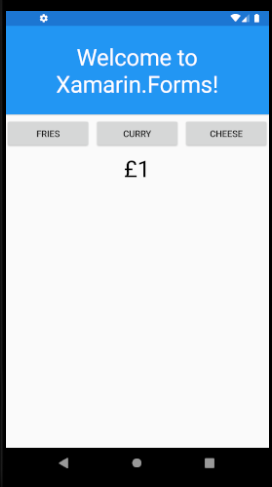
**Create a new visual studio project called Canteen App, summary of this lesson:**

* We are going to create the ordering page of your app (this is the second page in the navigation we created last week, so you need to have done that first!)
* You need a grid layout with buttons on each cell connected to a function or functions. Examples are here: [Xamarin.Forms Grid - Xamarin | Microsoft Docs](https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/grid)
* When you press each button it will add it to the total displayed in a text box at the bottom of the screen
* The grid layout and the title and the text box will be all inside of a StackLayout. Should look a bit like this, of course you can make your buttons look better.
* Make this look as close to your designs as possible, if possible!



1) You choose how you want your grid layout to look, you can make it uniform with the square the same size, to do this you just add the GRID tag and in between the opening and closing tag add the components you want to appear on the grid. In this example below there are 3 buttons (Add this grid layout code inbetween the StackLayout code as we want to have a title before the grid and a output label at the bottom):

|  |
| --- |
| <Grid>  <Button x:Name="Fries" ClassId="Fries" Clicked="AddToKart" Text="Fries" HorizontalOptions="Fill" Grid.Column="0" Grid.Row="0" ></Button>  <Button x:Name="Curry" Clicked="AddToKart" Text="Curry" HorizontalOptions="Fill" Grid.Column="1" Grid.Row="0" ></Button>  <Button x:Name="Cheese" Clicked="AddToKart" Text="Cheese" HorizontalOptions="Fill" Grid.Column="2" Grid.Row="0" ></Button>    </Grid> |

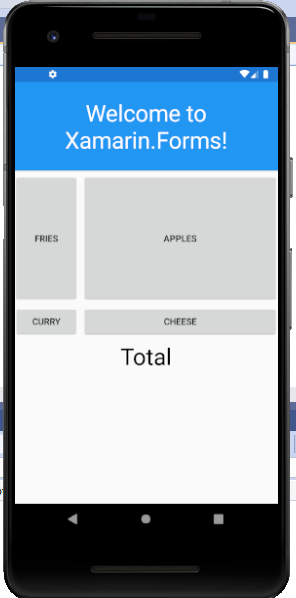
It is really important to specify what column and row your component should be on, this specifies the last button “Cheese” and it is on column 2 of row 0:

**Grid.Column**="2" **Grid.Row**="0"

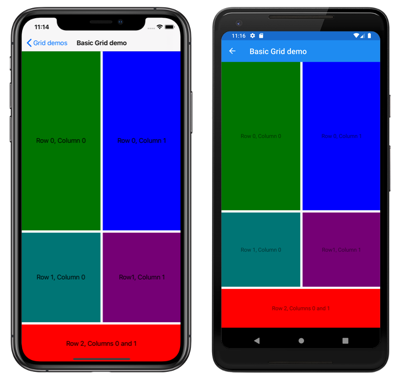
2) If you want your grid to have a more dynamic look, that is bit more tricky, you need to add Grid.RowDefinitions and Grid.ColumnDefinitions above the components as you can see in the example link at the top, and also here:

|  |
| --- |
| <Grid>  <Grid.RowDefinitions>  <RowDefinition Height="200" />  </Grid.RowDefinitions>  <Grid.ColumnDefinitions>  <ColumnDefinition Width="100"/>  </Grid.ColumnDefinitions>  <Button x:Name="Fries" ClassId="Fries" Clicked="AddToKart" Text="Fries" Grid.Column="0" Grid.Row="0" ></Button>  <Button x:Name="Apples" ClassId="Apples" Clicked="AddToKart" Text="Apples" Grid.Column="1" Grid.Row="0" ></Button>  <Button x:Name="Curry" Clicked="AddToKart" Text="Curry" Grid.Column="0" Grid.Row="1" ></Button>  <Button x:Name="Cheese" Clicked="AddToKart" Text="Cheese" Grid.Column="1" Grid.Row="1" ></Button>  </Grid> |

This example above produces buttons on row 0 that have a height 200 and width 100. All the other rows will have their default height and the width of the first row.



See more examples here [Xamarin.Forms Grid - Xamarin | Microsoft Docs](https://docs.microsoft.com/en-us/xamarin/xamarin-forms/user-interface/layouts/grid) to produce layouts like this:



3) WE are now going to create the function that will handle the button presses from ALL the buttons. To do this we need to identify what button is pressed, so you need to add a ClassID attribute to the buttons in the XAML, for example, this button has a classID of cheese and will call the function “AddToKart” when clicked:

<Button x:Name="Cheese" **ClassId="Cheese"** **Clicked="AddToKart"** Text="Cheese" Grid.Column="1" Grid.Row="1" ></Button>

4) Back in your MainPage.cs add the button function:

|  |
| --- |
| public void AddToKart(object sender, EventArgs args)  {  var button = (Button)sender; //1  var classId = button.ClassId; //2  } |

1. This gets the button that has called this function (object sender)
2. This gets the classID so we can identify the button

5) Now you need to write the code to get the correct price for each button that is pressed. You will need a global float variable called total to store the current total.

6) Finally you need to display this total. Add an output label after the grid, and update it’s value each time one of the buttons is pressed

EXTRA:

* See if you can add an image and a price to each button
* Make it look as close as possible to your design