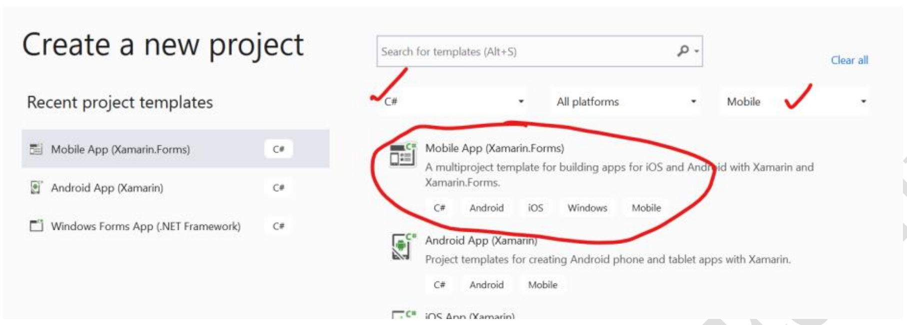
HANDS-ON EXERCISE FOR MID 1

OBJECTIVES:

- Learn how to create a standard calculator and create user defined class
- Use Grid layout to manage contents

DIRECTIONS:

1. Create a new project in Visual Studio 2019 and select Mobile App (Xamarin.Forms) on the template options. **Save it as Mid1_YOURFULLNAME**. Our objective here is to customize the Flyout template as a tool for rapid applications development.



Create a new project

Recent project templates

Mobile App (Xamarin.Forms) C#
Android App (Xamarin) C#
Windows Forms App (.NET Framework) C#

Search for templates (Alt+S)

Clear all

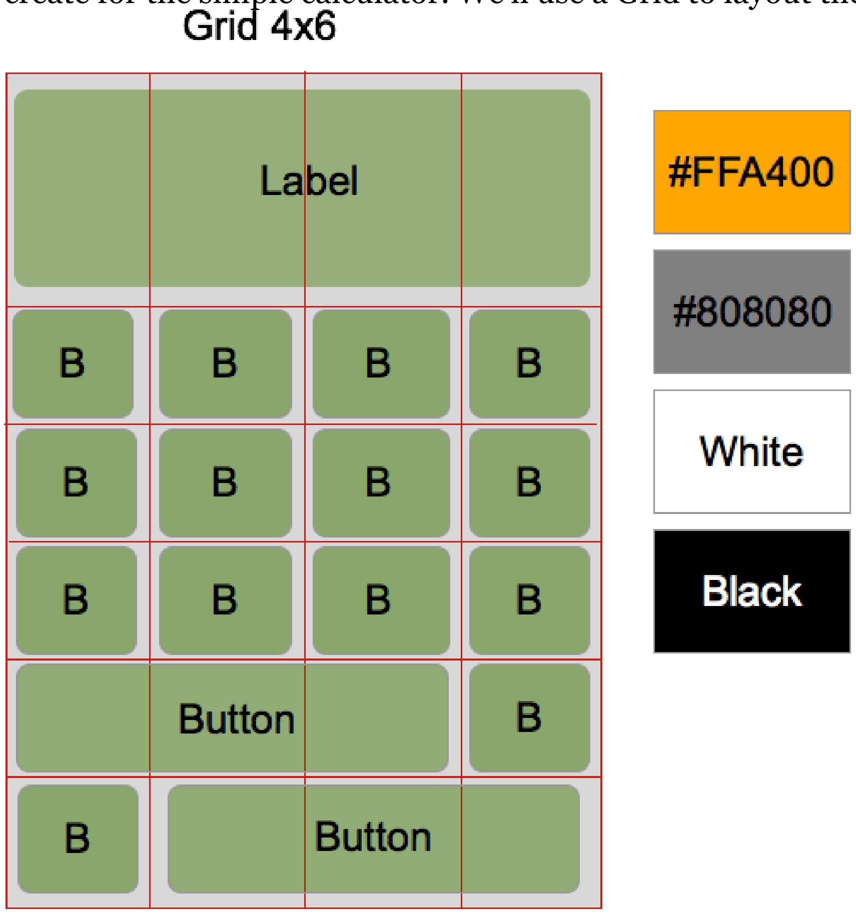
C# | All platforms | Mobile

Mobile App (Xamarin.Forms)
A multiproject template for building apps for iOS and Android with Xamarin and Xamarin.Forms.
C# Android iOS Windows Mobile

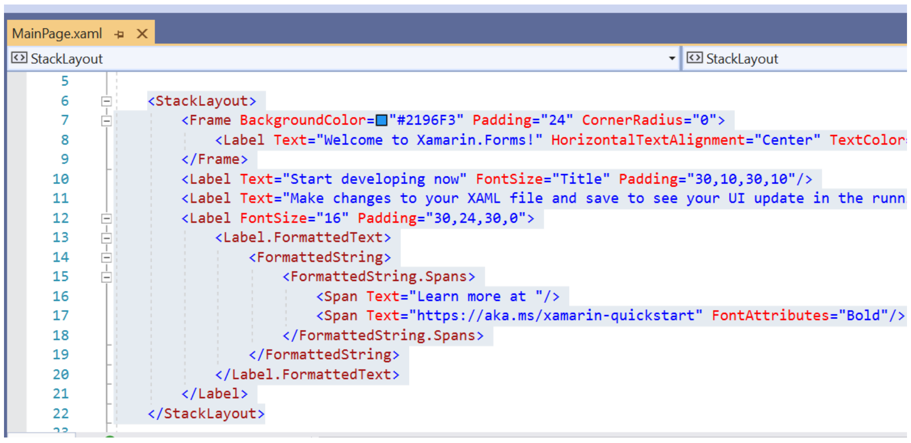
Android App (Xamarin)
Project templates for creating Android phone and tablet apps with Xamarin.
C# Android Mobile

iOS App (Xamarin)

2. Select Blank template. Take a look at the image below as guidance for the grid we will create for the simple calculator. We'll use a Grid to layout the required controls.

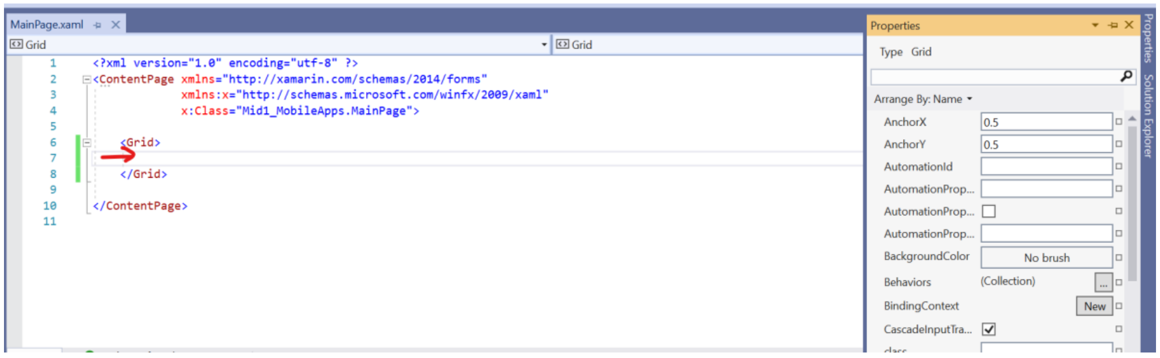


3. Open MainPage.xaml and delete lines 6-22 (StackLayout including its contents).


MainPage.xaml StackLayout StackLayout

```
    <StackLayout>
        <Frame BackgroundColor="#2196F3" Padding="24" CornerRadius="0">
            <Label Text="Welcome to Xamarin.Forms!" HorizontalTextAlignment="Center" TextColor
        </Frame>
        <Label Text="Start developing now" FontSize="Title" Padding="30,10,30,10"/>
        <Label Text="Make changes to your XAML file and save to see your UI update in the runn
        <Label FontSize="16" Padding="30,24,30,0">
            <Label.FormattedText>
                <FormattedString>
                    <FormattedString.Spans>
                        <Span Text="Learn more at "/>
                        <Span Text="https://aka.ms/xamarin-quickstart" FontAttributes="Bold"/>
                    </FormattedString.Spans>
                </FormattedString>
            </Label.FormattedText>
        </Label>
    </StackLayout>
```

4. Add a grid code <Grid> </Grid> on your page. Position your cursor in between the lines (in red arrow) and click on the Properties Window.


MainPage.xaml Grid Grid

```
<?xml version="1.0" encoding="utf-8" ?>
<ContentPage xmlns="http://xamarin.com/schemas/2014/forms"
             xmlns:x="http://schemas.microsoft.com/winfx/2009/xaml"
             x:Class="Mid1_MobileApps.MainPage">

    <Grid>

    </Grid>

</ContentPage>
```

Properties

Type Grid

Arrange By: Name

AnchorX	0.5
AnchorY	0.5
AutomationId	
AutomationProp...	
AutomationProp...	☐
AutomationProp...	
BackgroundColor	No brush
Behaviors	(Collection)
BindingContext	New
CascadeInputTra...	✓

Set the following properties of the grid.

Property	Value
Padding	"5,0"
RowSpacing	"1"
ColumnSpacing	"1"
BackgroundColor	"Black"

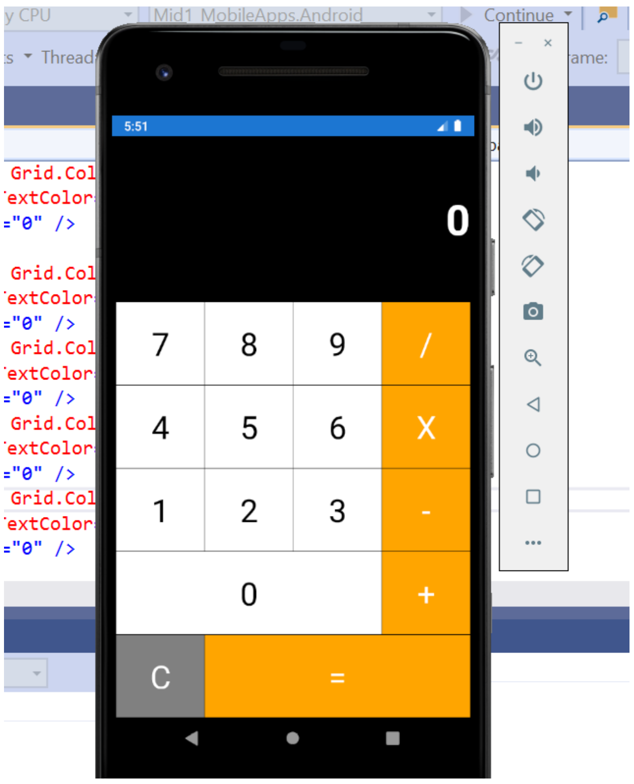
5. Customize the grid
   - Add RowDefinition and ColumnDefinition elements to create a 4x6 grid.
   - We want 4 columns and 6 rows, with the first row being 2X the size of all the others.
   - All of the columns should be equally sized.

```
<Grid Padding="5,0" RowSpacing="1" ColumnSpacing="1" BackgroundColor="Black">
        <Grid.RowDefinitions>
            <RowDefinition Height="2*" />
            <RowDefinition Height="*" />
            <RowDefinition Height="*" />
            <RowDefinition Height="*" />
            <RowDefinition Height="*" />
            <RowDefinition Height="*" />
        </Grid.RowDefinitions>

        <Grid.ColumnDefinitions>
            <ColumnDefinition Width="*" />
            <ColumnDefinition Width="*" />
            <ColumnDefinition Width="*" />
            <ColumnDefinition Width="*" />
        </Grid.ColumnDefinitions>
    </Grid>
```

6. Add a label and set of buttons right below the </Grid.ColumnDefinitions>.

```
<Label FontAttributes="Bold" FontSize="48" BackgroundColor="Black" Text="0"
                  TextColor="White" HorizontalTextAlignment="End"
VerticalTextAlignment="Center"
                  LineBreakMode="NoWrap" Grid.ColumnSpan="4" />

        <Button Text="7" Grid.Row="1" Grid.Column="0"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="8" Grid.Row="1" Grid.Column="1"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="9" Grid.Row="1" Grid.Column="2"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />

        <Button Text="4" Grid.Row="2" Grid.Column="0"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="5" Grid.Row="2" Grid.Column="1"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="6" Grid.Row="2" Grid.Column="2"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />

        <Button Text="1" Grid.Row="3" Grid.Column="0"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="2" Grid.Row="3" Grid.Column="1"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="3" Grid.Row="3" Grid.Column="2"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />
        <Button Text="0" Grid.Row="4" Grid.Column="0" Grid.ColumnSpan="3"
           BackgroundColor="White" TextColor="Black"
           FontSize="36" BorderRadius="0" />

        <Button Text="/" Grid.Row="1" Grid.Column="3"
            BackgroundColor="#FFA500" TextColor="White"
            FontSize="36" BorderRadius="0" />
        <Button Text="x" Grid.Row="2" Grid.Column="3"
            BackgroundColor="#FFA500" TextColor="White"
            FontSize="36" BorderRadius="0" />
        <Button Text="-" Grid.Row="3" Grid.Column="3"
            BackgroundColor="#FFA500" TextColor="White"
            FontSize="36" BorderRadius="0" />
        <Button Text="+" Grid.Row="4" Grid.Column="3"
            BackgroundColor="#FFA500" TextColor="White"
            FontSize="36" BorderRadius="0" />

        <Button Text="C" Grid.Row="5" Grid.Column="0"
            BackgroundColor="#808080" TextColor="White"
            FontSize="36" BorderRadius="0" />

        <Button Text="=" Grid.Row="5" Grid.Column="1" Grid.ColumnSpan="3"
            BackgroundColor="#FFA500" TextColor="White"
            FontSize="36" BorderRadius="0" />
```

Try to run your application and it should display the following:



```
Grid.Col
'extColor
="0" />

Grid.Col
'extColor
="0" />
Grid.Col
'extColor
="0" />
Grid.Col
'extColor
="0" />
Grid.Col
'extColor
="0" />
```

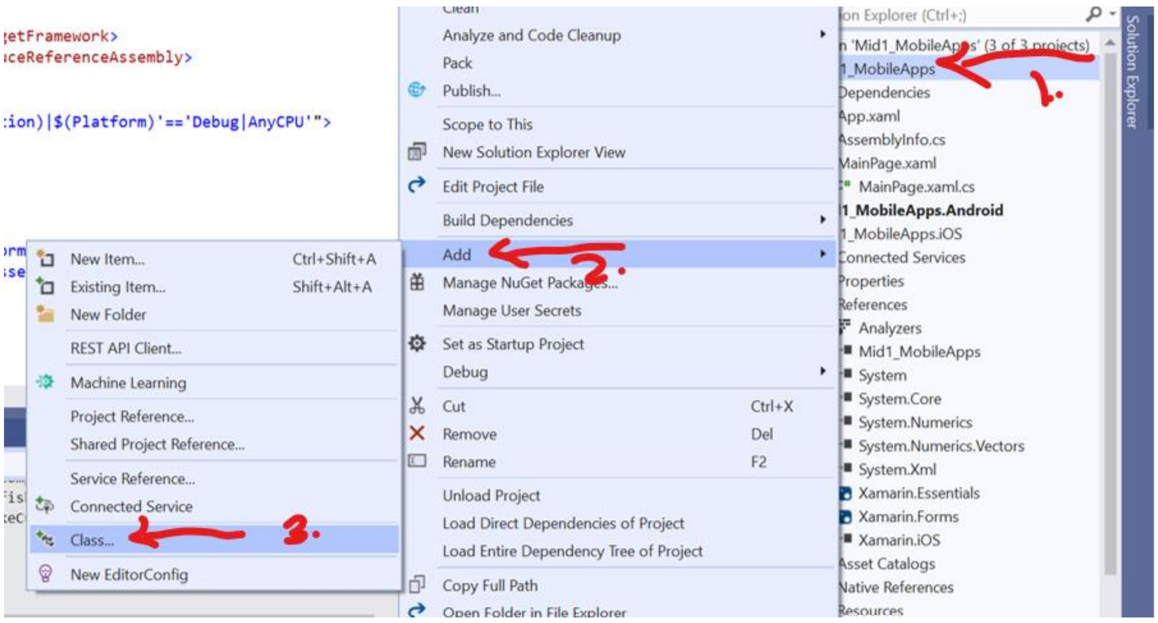
7. Add a name to the XAML label as shown below.

```
<Label x:Name="resultText" FontAttributes="Bold" FontSize="48"
    BackgroundColor="Black" Text="0" TextColor="White"
    HorizontalTextAlignment="End" VerticalTextAlignment="Center"
    LineBreakMode="NoWrap" Grid.ColumnSpan="4" />
```

8. Add event handlers to all the buttons in MainPage.xaml. We want to be able to perform calculations by pressing the calculator buttons. Wire up the behavior to the Buttons using the **specific** methods as instructed below.
   - Wire up the Clicked event in XAML on all the number (0-9) buttons to the **OnSelectNumber** method.
   - Wire up the Clicked event in XAML on all the operator (-,+,*,/) buttons to the **OnSelectOperator** method.
   - Wire up the Clicked event in XAML on the Clear button to the OnClear method.
   - Wire up the Clicked event in XAML on the Equal (=) button to the **OnCalculate** method.

```
// Examples of wiring up a Clicked event
<Button Text="1" ... Clicked="OnSelectNumber" />
<Button Text="2" ... Clicked="OnSelectNumber" />
<Button Text="9" ... Clicked="OnSelectNumber" />
. . .
<Button Text="/" ... Clicked="OnSelectOperator" />
```

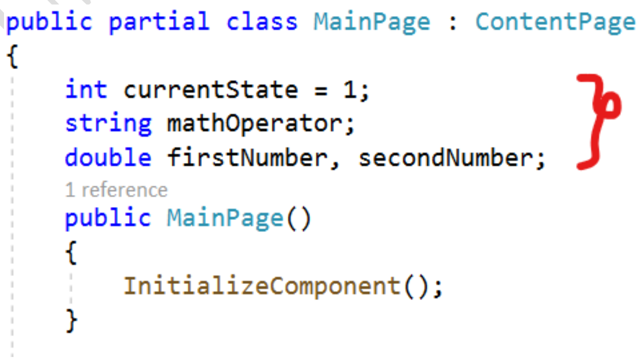
9. Go to Solution Explorer and create a new class. Right click on the main folder and click Add | Class. Save the name of your class as **Calculator.cs**.



Add the following codes inside the **class Calculator {** //paste it here **}** code.

```
public static double Calculate(double value1, double value2, string mathOperator)
{
            double result = 0;

            switch (mathOperator)
            {
                case "÷":
                    result = value1 / value2;
                    break;
                case "×":
                    result = value1 * value2;
                    break;
                case "+":
                    result = value1 + value2;
                    break;
                case "-":
                    result = value1 - value2;
                    break;
            }

            return result;
}
```

Open MainPage.xaml.cs file and insert the following variables above the **public MainPage()** method.

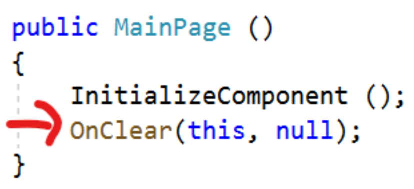


```
public partial class MainPage : ContentPage
{
    int currentState = 1;
    string mathOperator;
    double firstNumber, secondNumber;
    1 reference
    public MainPage()
    {
        InitializeComponent();
    }
```

Then add the following methods below the **public MainPage()** method.

```
void OnSelectNumber(object sender, EventArgs e)
              {
                     Button button = (Button)sender;
                     string pressed = button.Text;

                     if (this.resultText.Text == "0" || currentState < 0)
                     {
                            this.resultText.Text = "";
                            if (currentState < 0)
                                   currentState *= -1;
                     }

                     this.resultText.Text += pressed;

                     double number;
                     if (double.TryParse(this.resultText.Text, out number))
                     {
                            this.resultText.Text = number.ToString("N0");
                            if (currentState == 1)
                            {
                                   firstNumber = number;
                            }
                            else
                            {
                                   secondNumber = number;
                            }
                     }
              }

              void OnSelectOperator(object sender, EventArgs e)
              {
                     currentState = -2;
                     Button button = (Button)sender;
                     string pressed = button.Text;
                     mathOperator = pressed;
              }

              void OnClear(object sender, EventArgs e)
              {
                     firstNumber = 0;
                     secondNumber = 0;
                     currentState = 1;
                     this.resultText.Text = "0";
              }

              void OnCalculate(object sender, EventArgs e)
              {
                     if (currentState == 2)
                     {
                            var result = Calculator.Calculate(firstNumber,
secondNumber, mathOperator);

                            this.resultText.Text = result.ToString();
                            firstNumber = result;
                            currentState = -1;
                     }
              }
```

Finally, add a method call below **InitializeComponent()** to clear the Label at the start of the application.



```
public MainPage ()
{
    InitializeComponent ();
    OnClear(this, null);
}
```

10. Run your application and your little calculator should now work. Test some numbers and operations to ensure that your calculator is perfectly working.