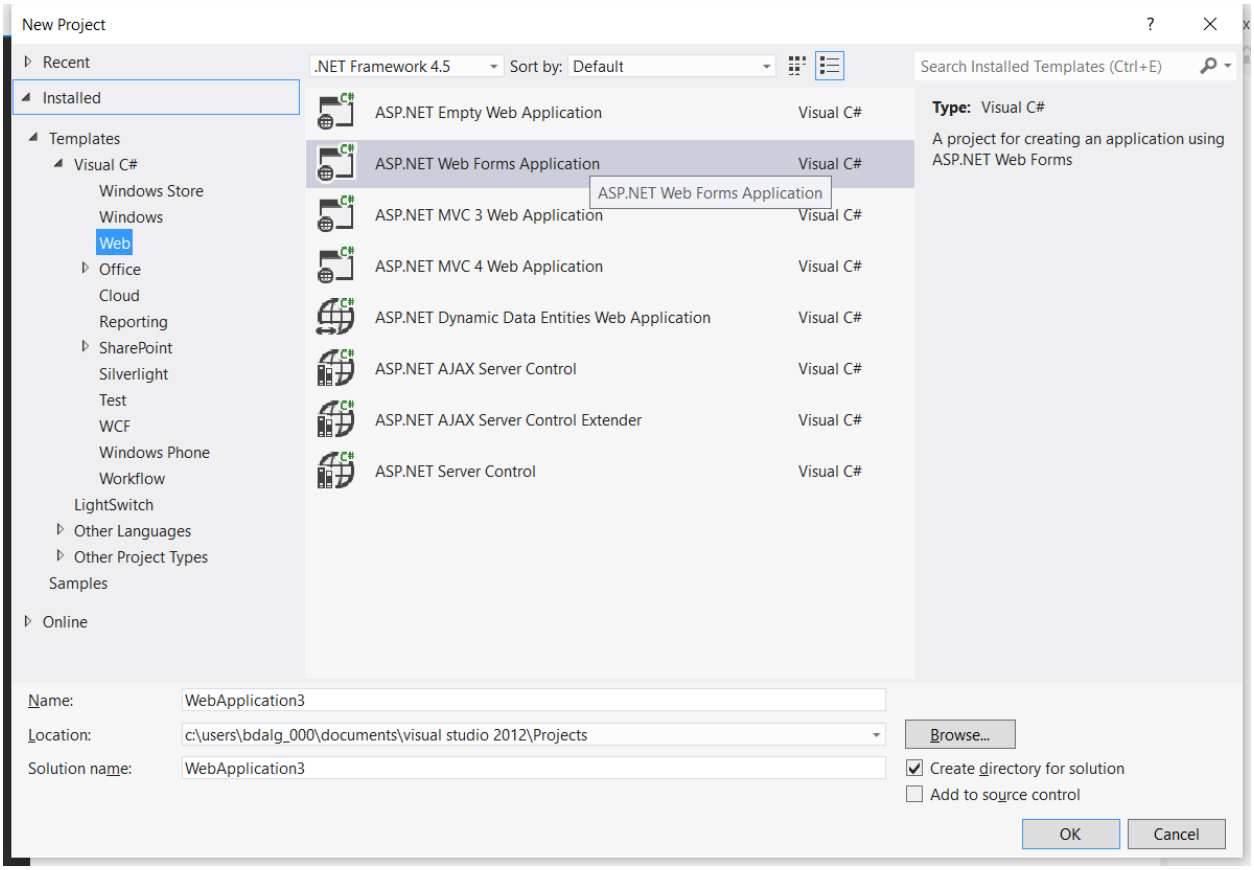
Lab: Creating your first ASP.NET Web Application

This lab will walk you through creating a simple calculator app in Visual Studio 2012.

Starting a new project

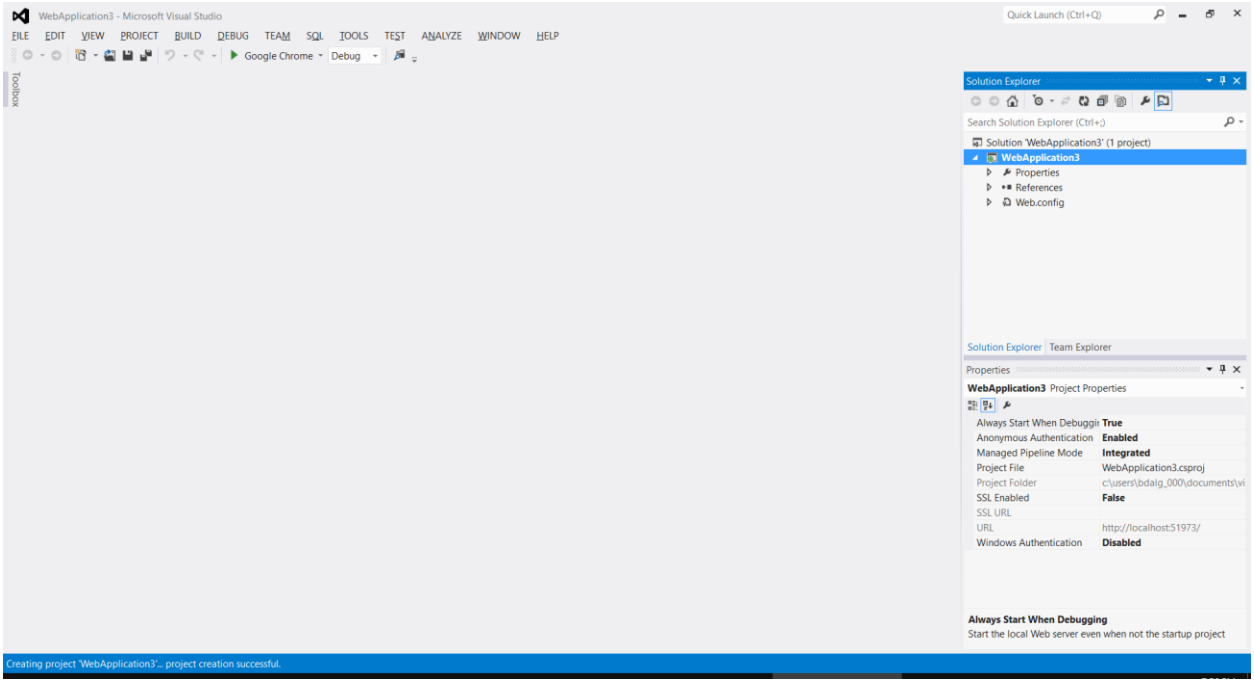
1. Open Visual Studio 2012
2. In the Menu, click *File* → *New* → *Project*
3. Choose the item Template → Visual C# → Web. Click on the ASP.NET Web Form Application.



4. Make a note of the **Name:** field and the **Location:** field. You will use these to submit your work later.

Getting familiar with Visual Studio

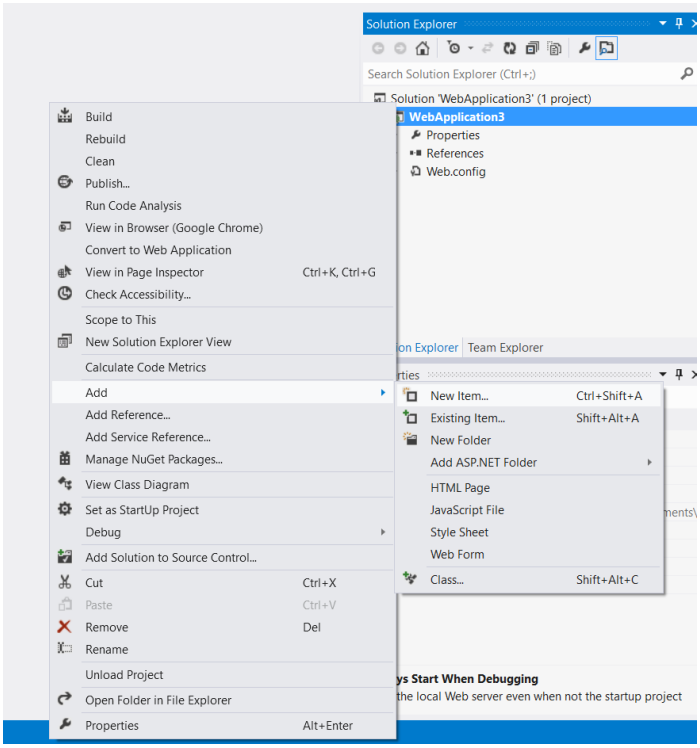
The application should be created for you and you should the Solution Explorer window. If you do not see the Solution Explorer, click on the menu item **View → Solution Explorer.**



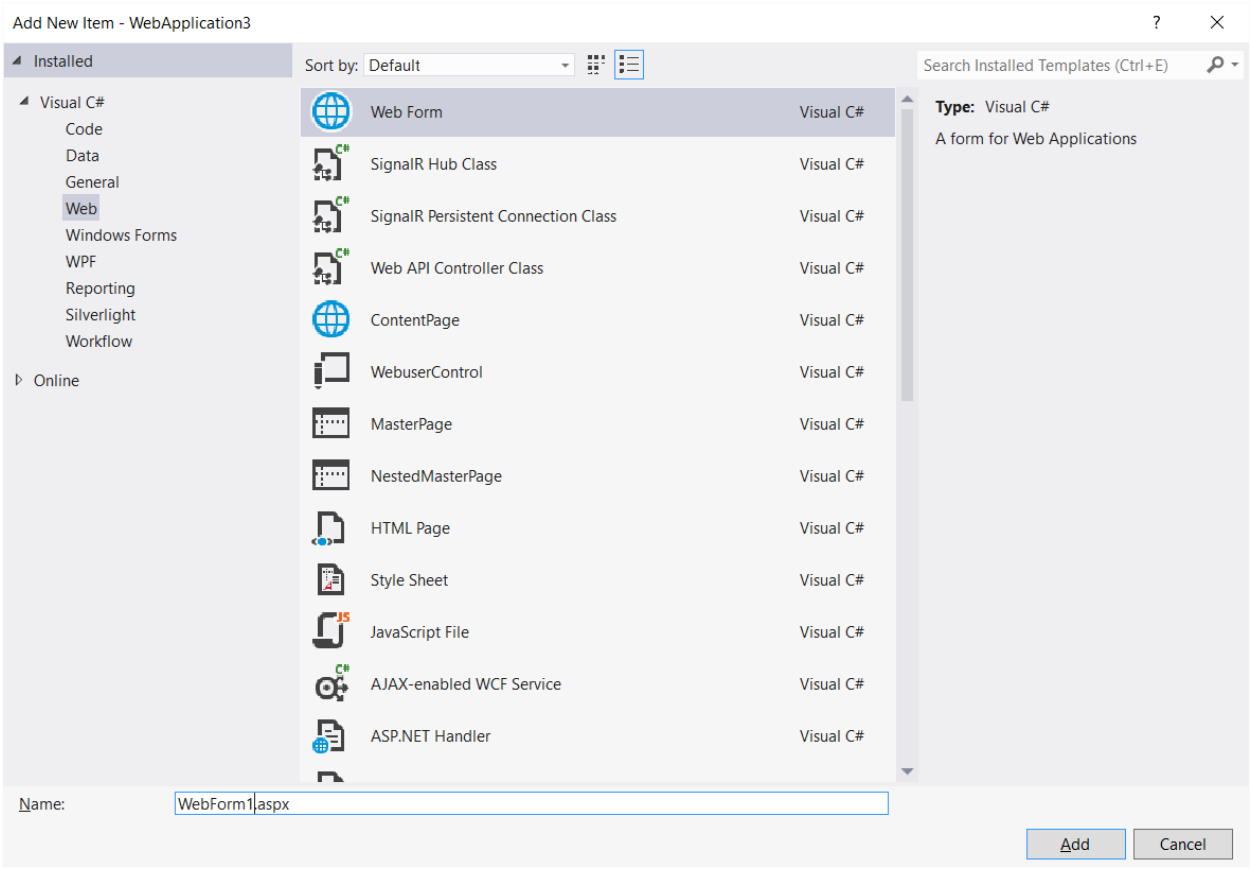
The Solution Explorer is where you can view all of you files in their project structure. There isn't much there right now and you should not edit anything that is in there yet. We will talk about the project files in later weeks.

Adding a new page

1. In the Solution Explorer, right click on your project. In the picture, notice that **WebApplication3** is clicked, not **Solution 'WebApplication3'** (our solution contains 1 project, but a solution can contain many projects).
2. Choose **Add → New Item**

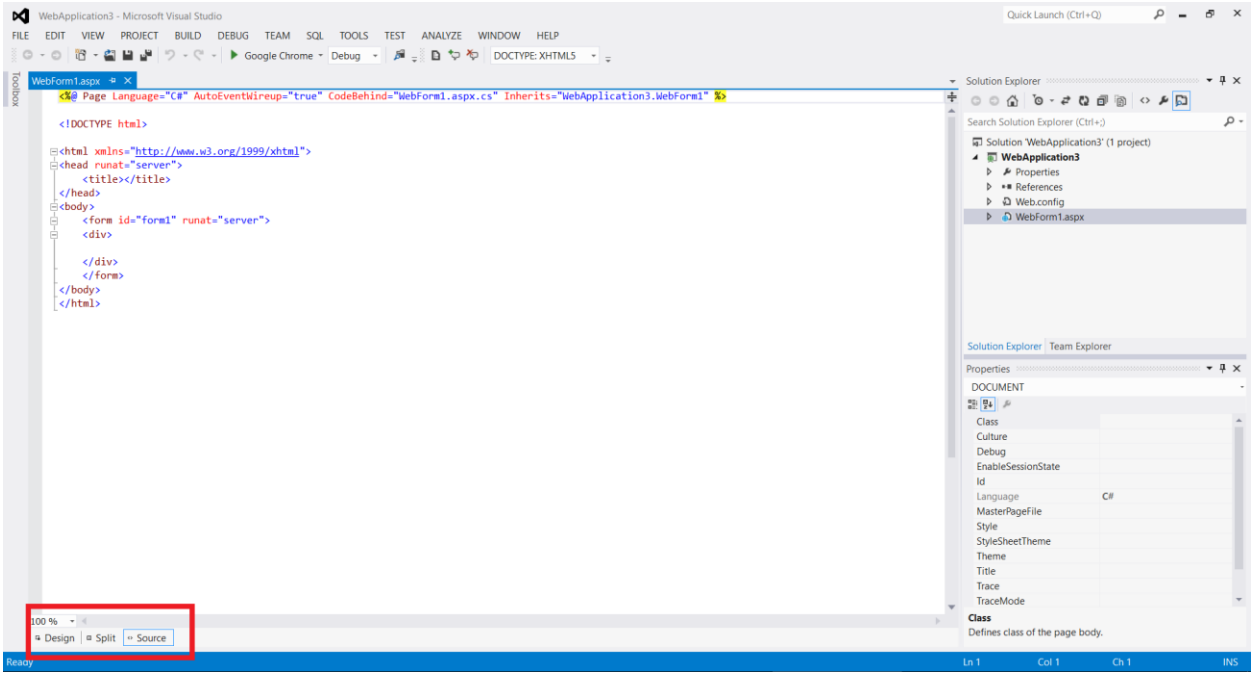


3. Choose **Web Form**. You can keep the default name *WebForm1.aspx*



Getting to know the editor

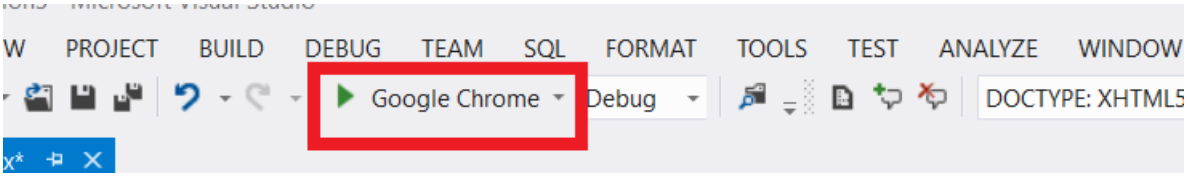
1. WebForm1.aspx should been created and should have opened. If it did not open, double click on it in the Solution Explorer.



2. There are 2 tabs at the bottom called Design and Source (and one in the middle called split).
    a. Source will show us our code
    b. Design will show us our rendered page
3. Add something simple in the empty div in the middle of the screen

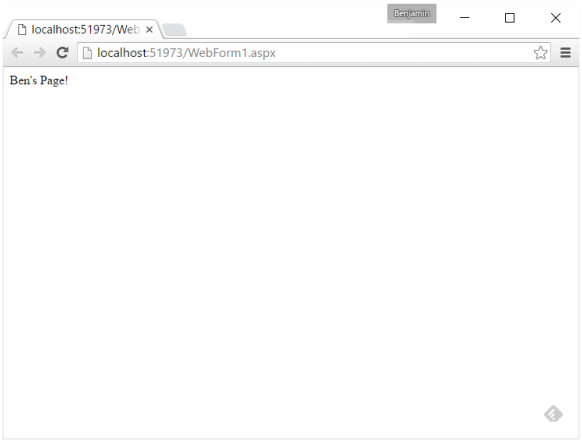
```
<html xmlns="http://www.w3.org/1999/xhtml"
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div>

        Ben's Page!

    </div>
    </form>
</body>
</html>
```

4. Run the application
    a. Click the green “play” button

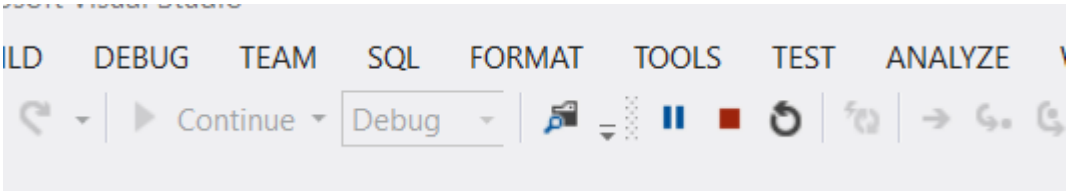


    b. Or click the menu item **Debug → Start Debugging**

5. The application should open in your default browser

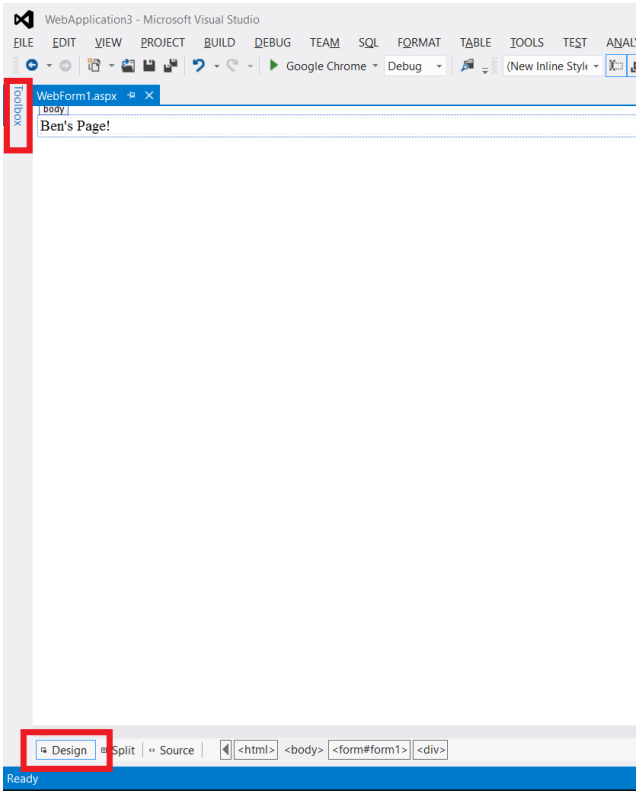


6. Press the red square “stop” button to stop debugging.

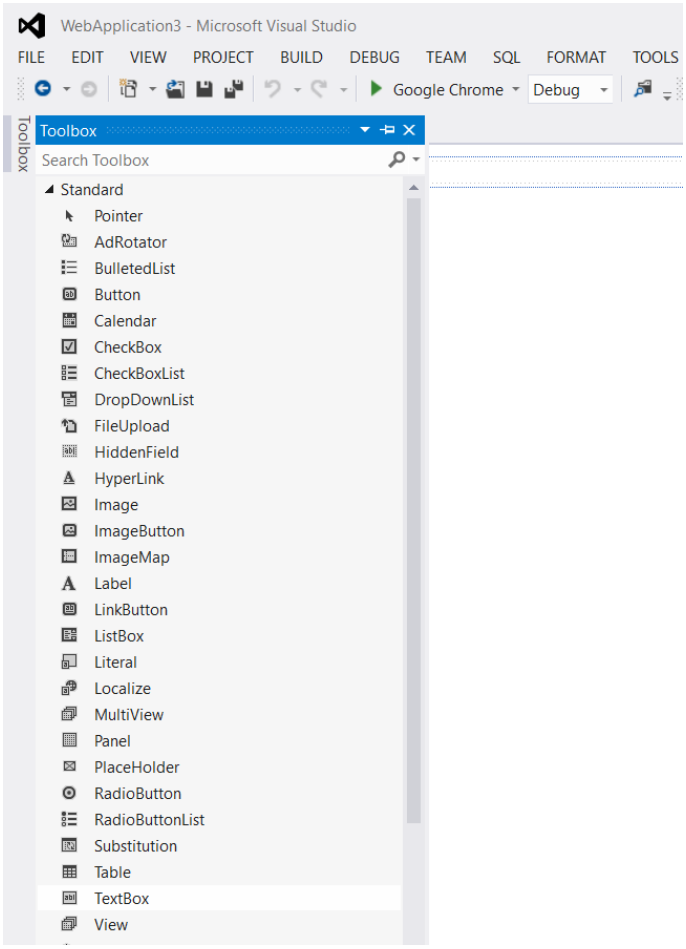


Adding Controls

1. Open WebForm1.aspx in Design view.
2. There is a Toolbox button on the left margin of the page. If you do not see the Toolbox button, click **Menu → View → Toolbox**

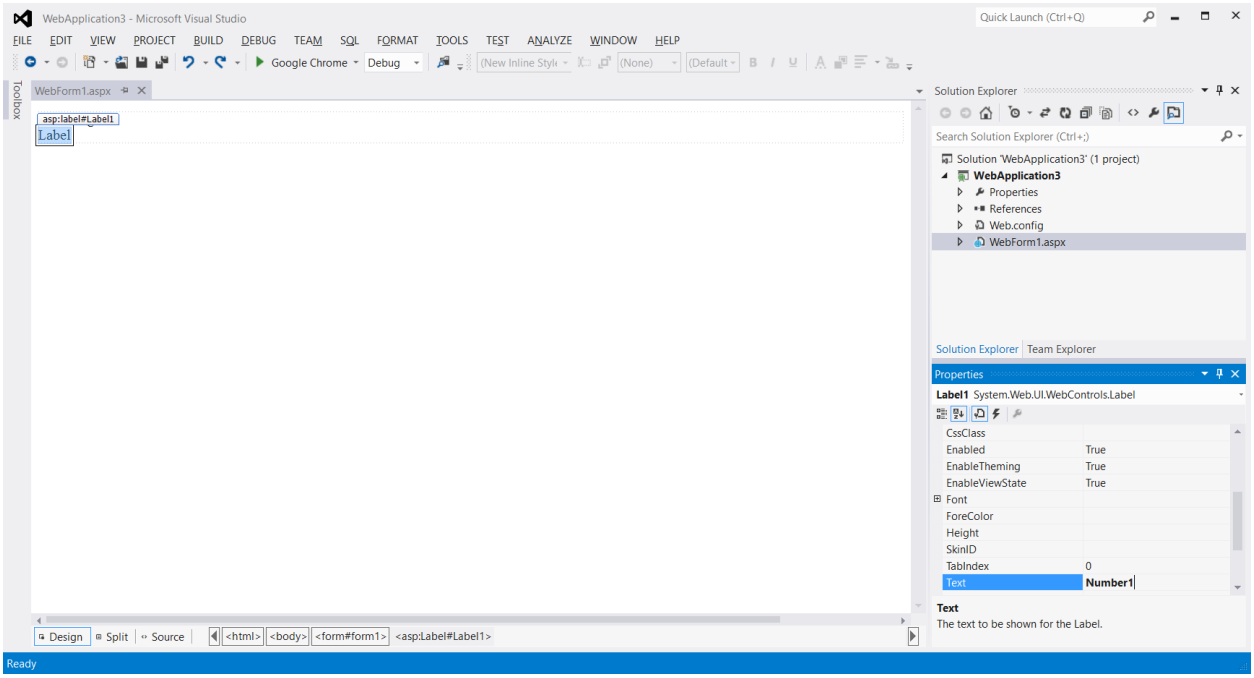


3. Inside the Toolbox, you will see some controls. These controls are similar to HTML items, but will allow us to interact with the webpage through C# code.

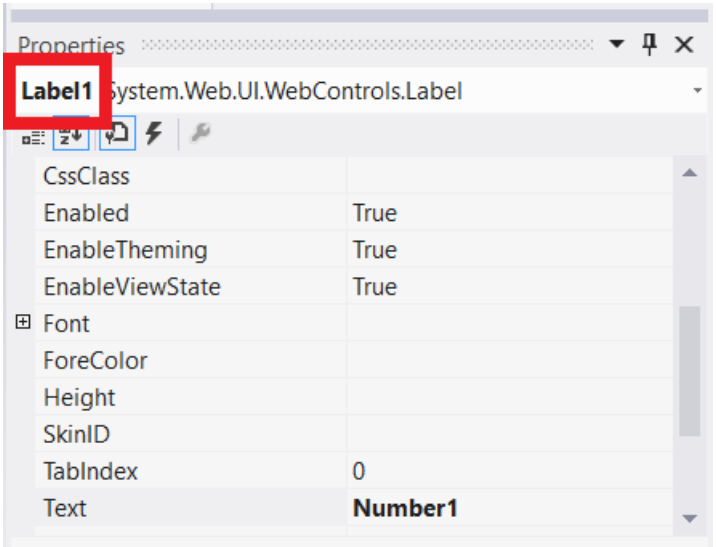


4. Click and hold on **Label** and drag it to the page.

5. In the properties panel, find the **Text** attribute for the label. Change it to something else.
    a. You can also do into the **Source** view and change it there.



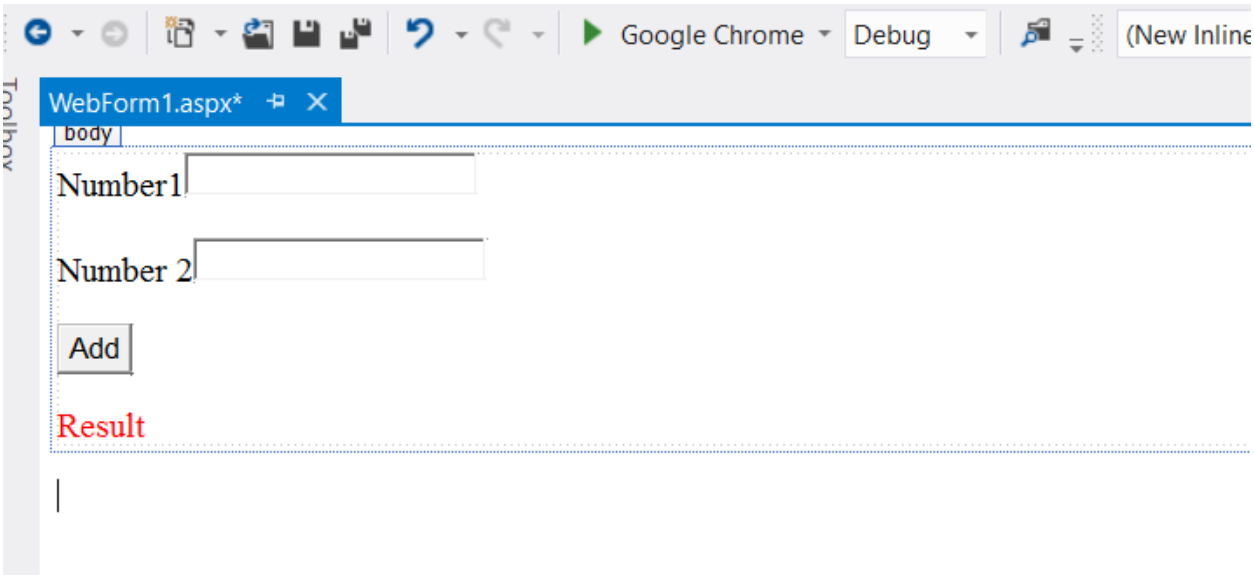
6. Notice that the ID is Label1



Creating a Page

Recreate the page below. It consists of

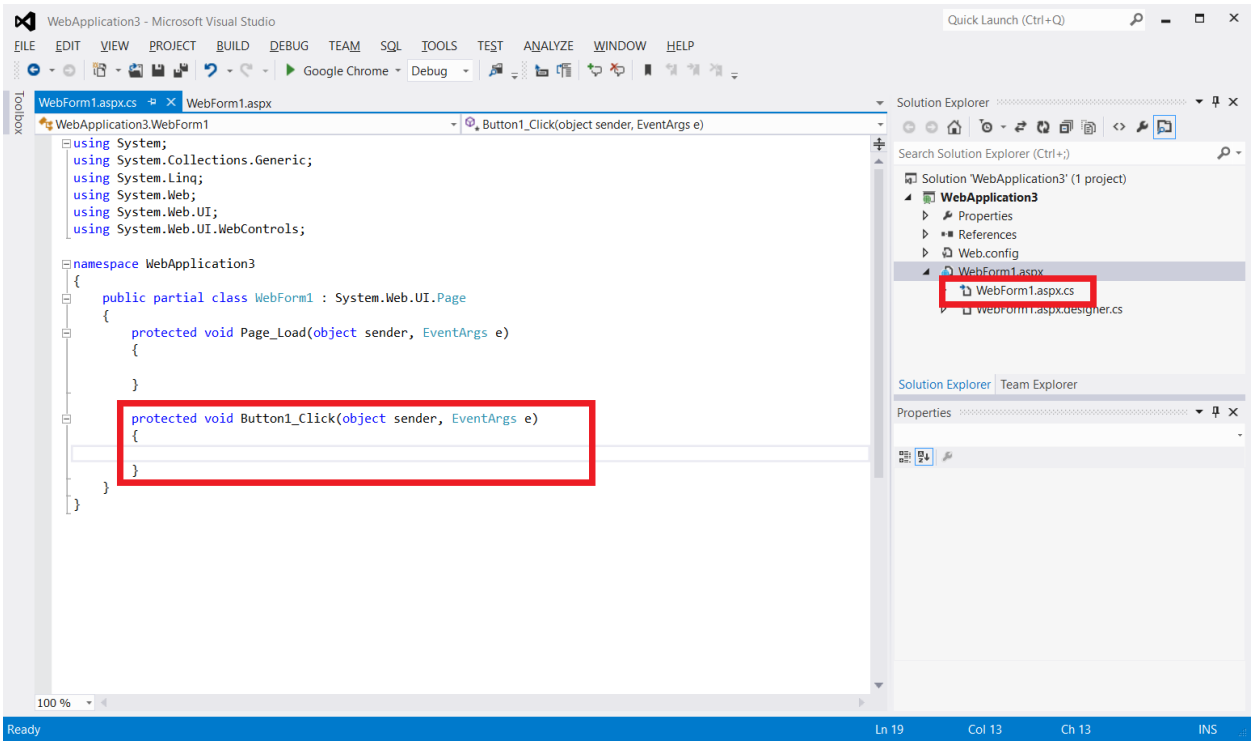
- 3 Labels (Number 1, Number 2, Result)
- 2 TextBoxes
- 1 Button



Adding an action to the button

In order to add code to the button, we must link it to the code. There is a shortcut for this in Visual Studio.

1. Double Click on the button. This should add a new empty method to your C# code and open the WebForm1.aspx.cs class (which is nested under WebForm1.aspx in the Solution Explorer).



WARNING: Do not open or change WebForm1.aspx.deisgner.cs. This is a system generated file and you should not change it unless you are absolutely sure what you are changing.

2. This C# page is the code that contains the server code.
3. If we look back at the ASPX page Source, we can see that this also changed from

```
<asp:Button ID="Button1" runat="server" Text="Add" />
```

To

```
<asp:Button ID="Button1" runat="server" OnClick="Button1_Click" Text="Add" />
```

4. The OnClick event is linked to our C# method named Button1_Click. So, when our button is clicked, the C# method will execute.

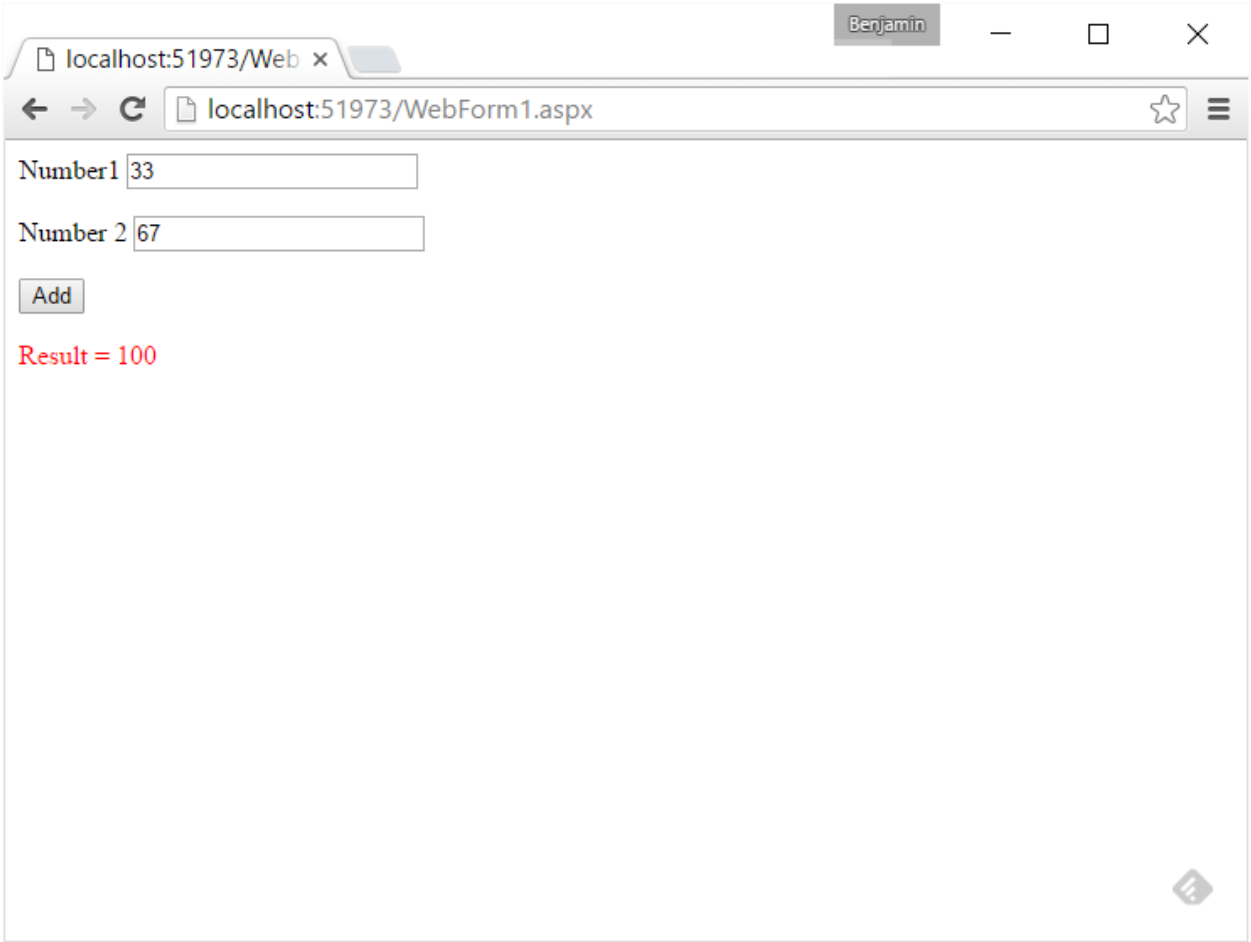
5. Let's add some code to the Button1_Click method.

```
protected void Button1_Click(object sender, EventArgs e)
{
    var number1 = int.Parse(TextBox1.Text);
    var number2 = int.Parse(TextBox2.Text);
    var result = number1 + number2;
    Label3.Text = "Result = " + result;
}
```

6. This method will do the following
    a. Grab the numbers from the page (Textbox1 and Textbox2)
    b. Parse the text into integers
    c. Add the integers
    d. Put the result onto the screen in the Result Label (Label3).

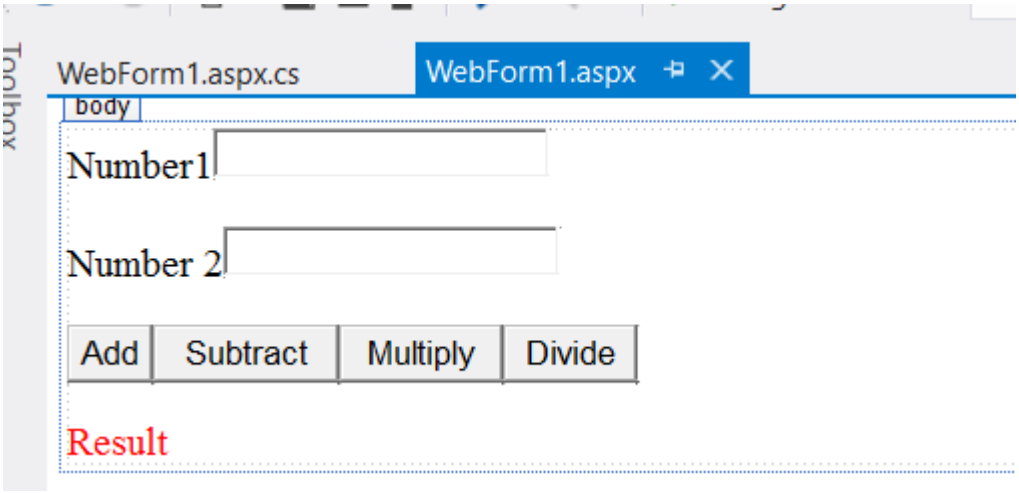
NOTE: We should note that our ID's for our controls are not very good and should be named better in the future

7. Run the program and test that it does addition correctly when the button is clicked in the browser.



Exercise

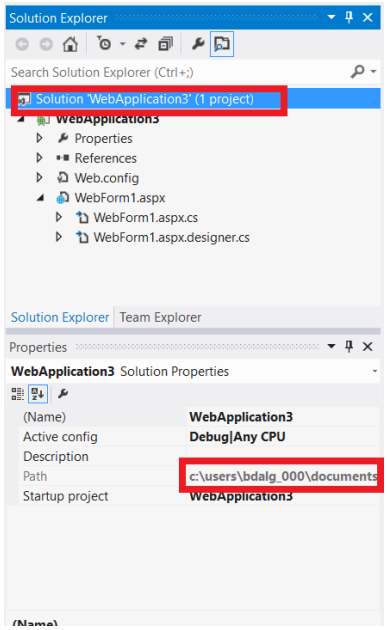
Add three more buttons to the page with 3 new methods that will Subtract, Mutliply and Divide the two numbers. You do not need to add any more Labels or TextBoxes.



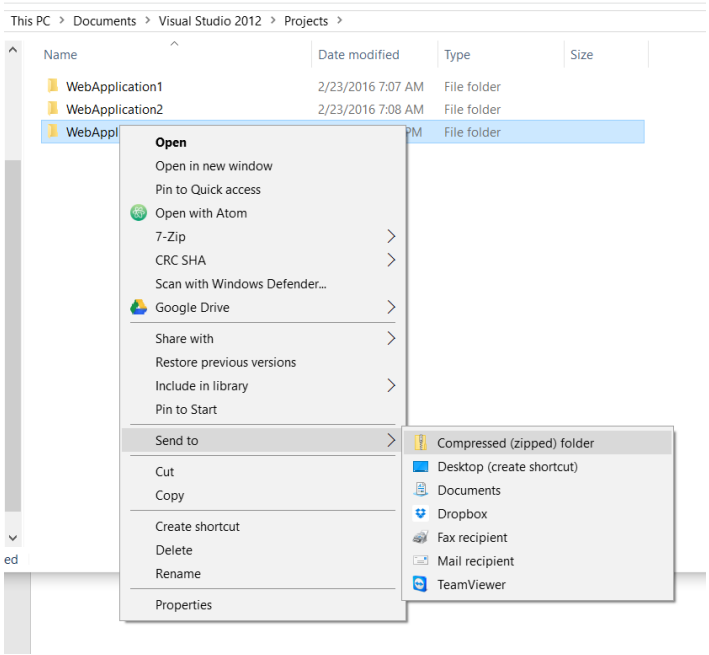
Zipping the program

In order to submit the program, you must zip the program. This makes it smaller and easy to transfer. Some programs (like email) will not accept code in an attachment so zipping programs is necessary.

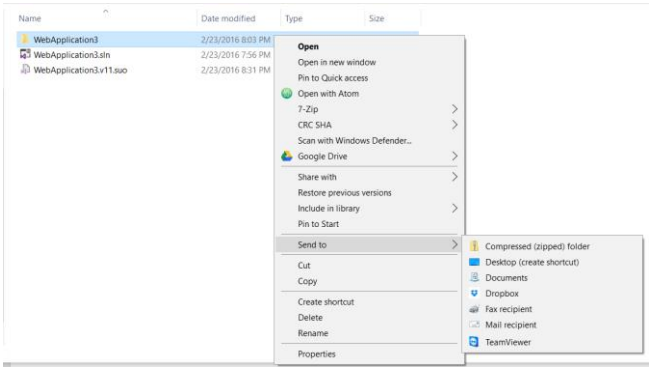
1. Locate the project in the folder you created it in. If you have forgotten where it is, you can click on the Solution Explorer, right click on the Solution file and click properties.



2. Go to the solution folder in the Explorer window. Right Click on you solution and click **Send To → Compressed (zipped) folder**



3. **Do not zip the folder inside.** If you do this, the zip file will not include the .sln file (the solution file) and you will not be able to open the solution.



4. You can use other programs to zip the code (like 7-zip), but it must always be zipped.