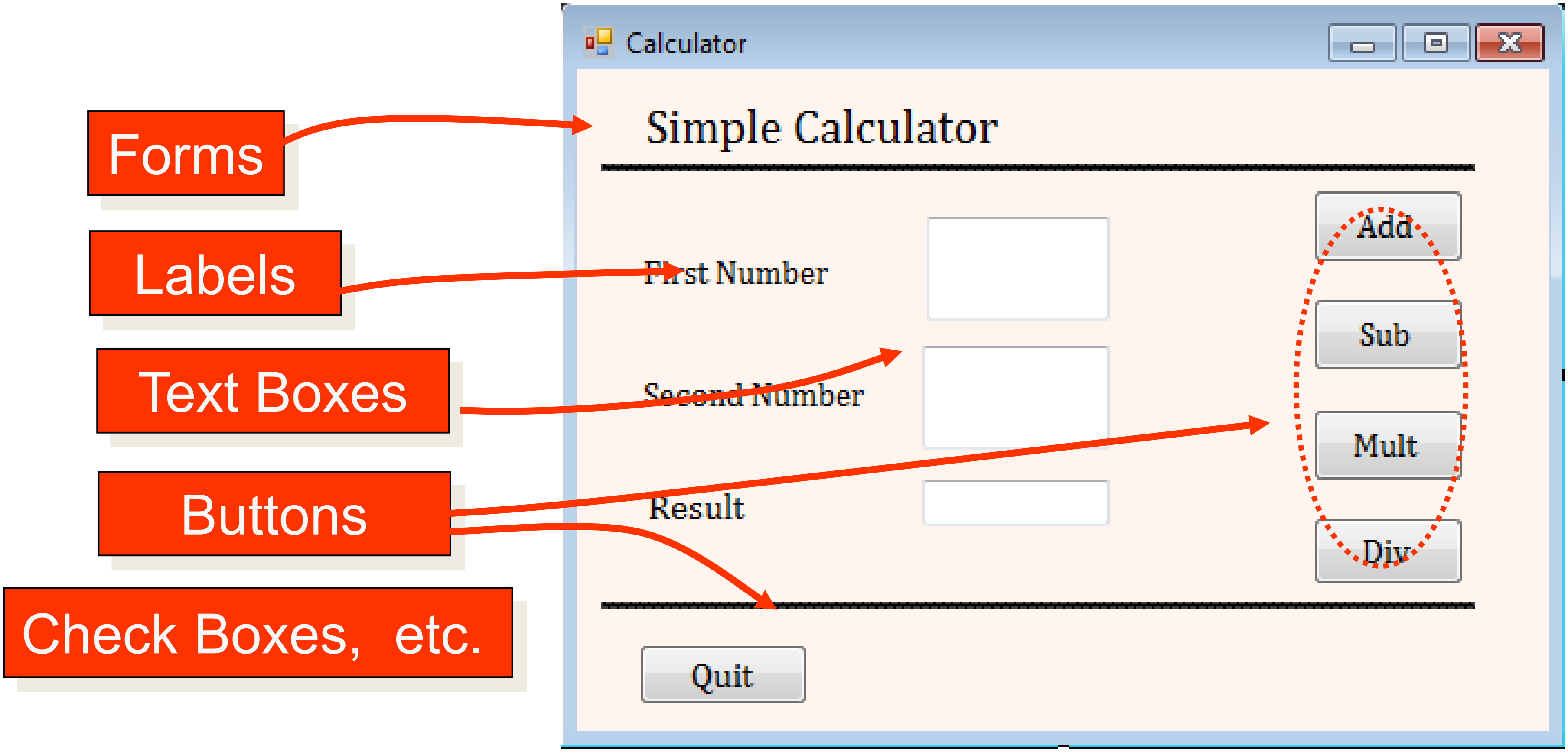
CO453 Application Programming

Week 5 – C# .NET part 1

Introduction to forms, and splash screens

.NET C# is VISUAL

.NET programs run on **MS Windows**
.NET provides ready-made **objects** to create an easy-to-use visual user interface



Forms

Labels

Text Boxes

Buttons

Check Boxes, etc.

.NET Objects have PROPERTIES

We can change the properties of any object:

Name

BackColor

BackgroundImage

BorderStyle

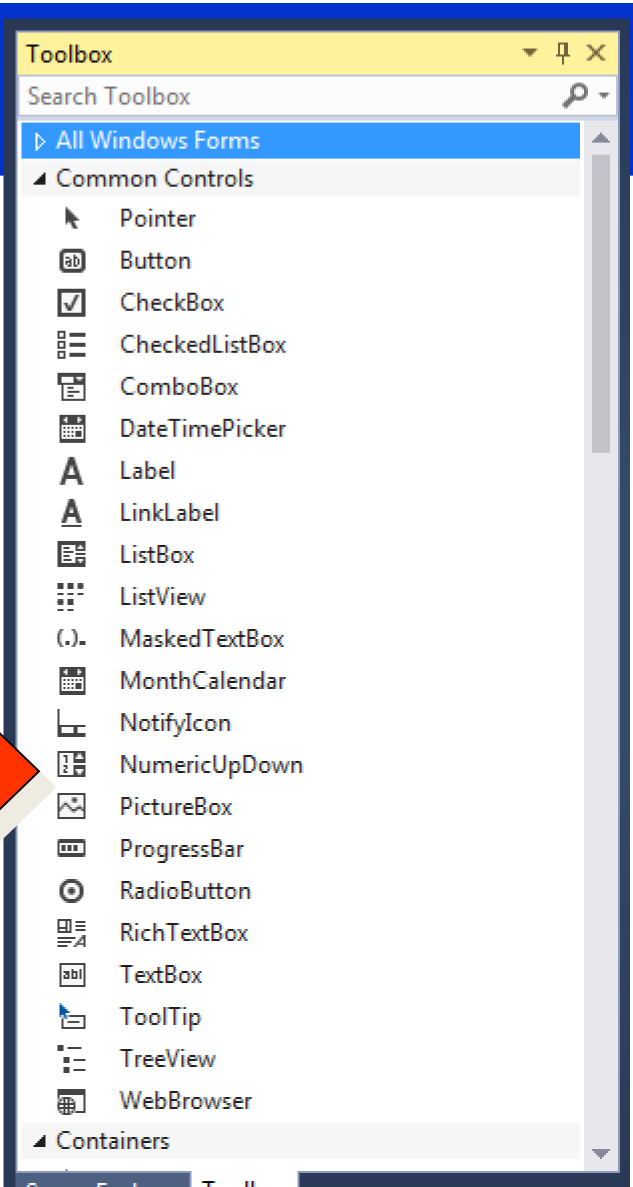
Text

Font

AutoSize

etc.

Select any object Then press the F4 key to show its <u>Properties Window</u>



C# .NET is Event-Driven

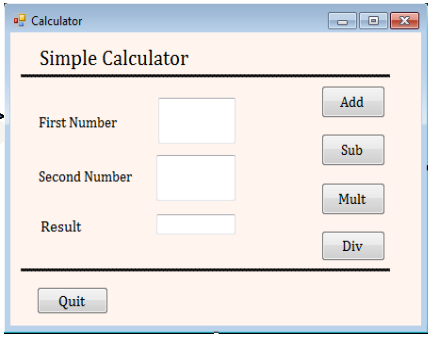
.NET programs respond to Events

a mouse-click
a key press
a form loading, etc.

.NET programs can be constructed differently:

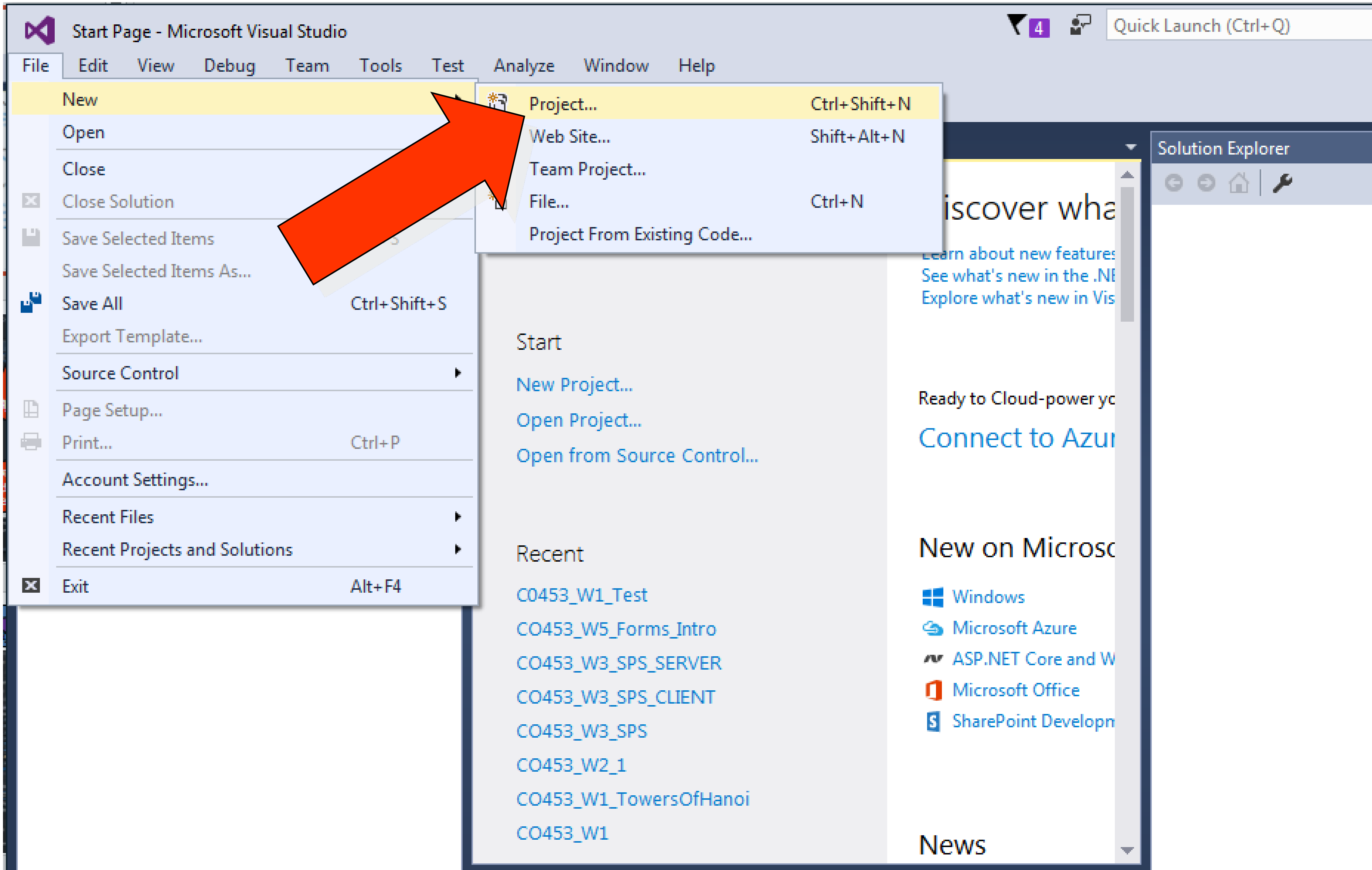
- A project can consist of several forms with objects on them
- objects have built-in properties and methods
- program code is attached to the various object methods.

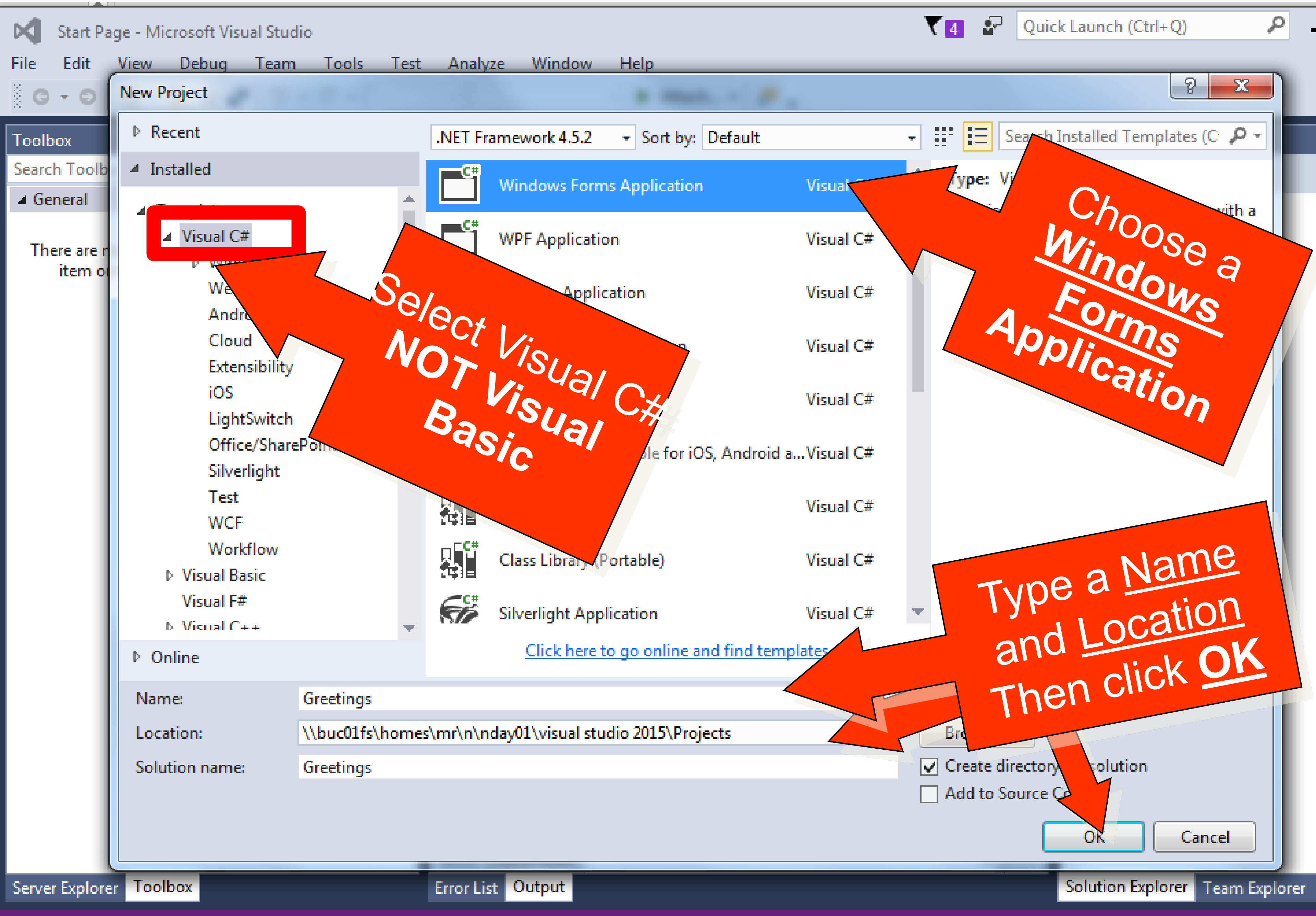
The objects together make up the interface with the user



Starting a Windows Project

Create a New Project





.NET Framework 4.5.2

Sort by: Default

Name: Greetings

Location: \\buc01fs\homes\mr\n\nday01\visual studio 2015\Projects

Solution name: Greetings

☑ Create directory solution

□ Add to Source C

Select Visual C# **NOT Visual Basic**

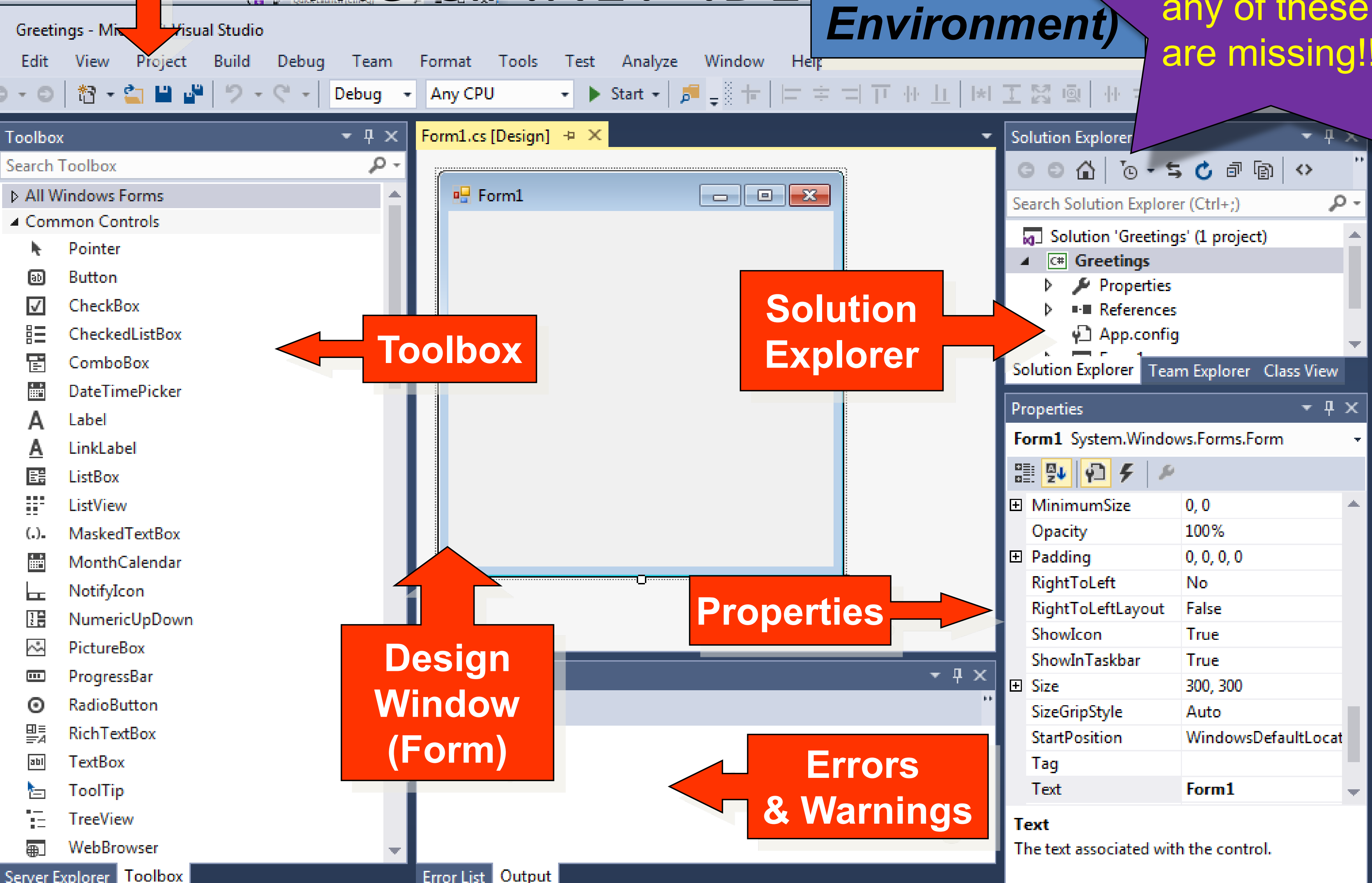
Choose a **Windows Forms Application**

Type a Name and Location Then click **OK**

The C# .NET IDE

(Integrated Developme Environment)

Use **View** if any of these are missing!!



Menu

Toolbox

Solution Explorer

Design Window (Form)

Properties

Errors & Warnings

Form1 System.Windows.Forms.Form

MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	300, 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocat
Tag	
Text	**Form1**

Text

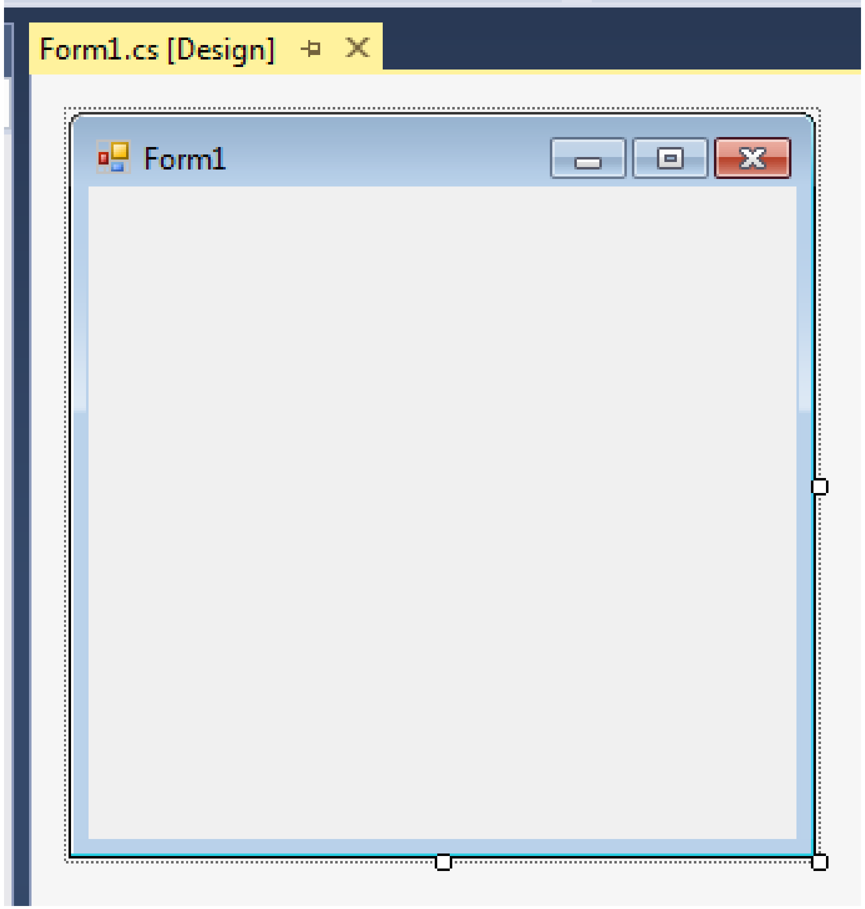
The text associated with the control.

Building A simple Windows project

(Name Entry and Message Form)

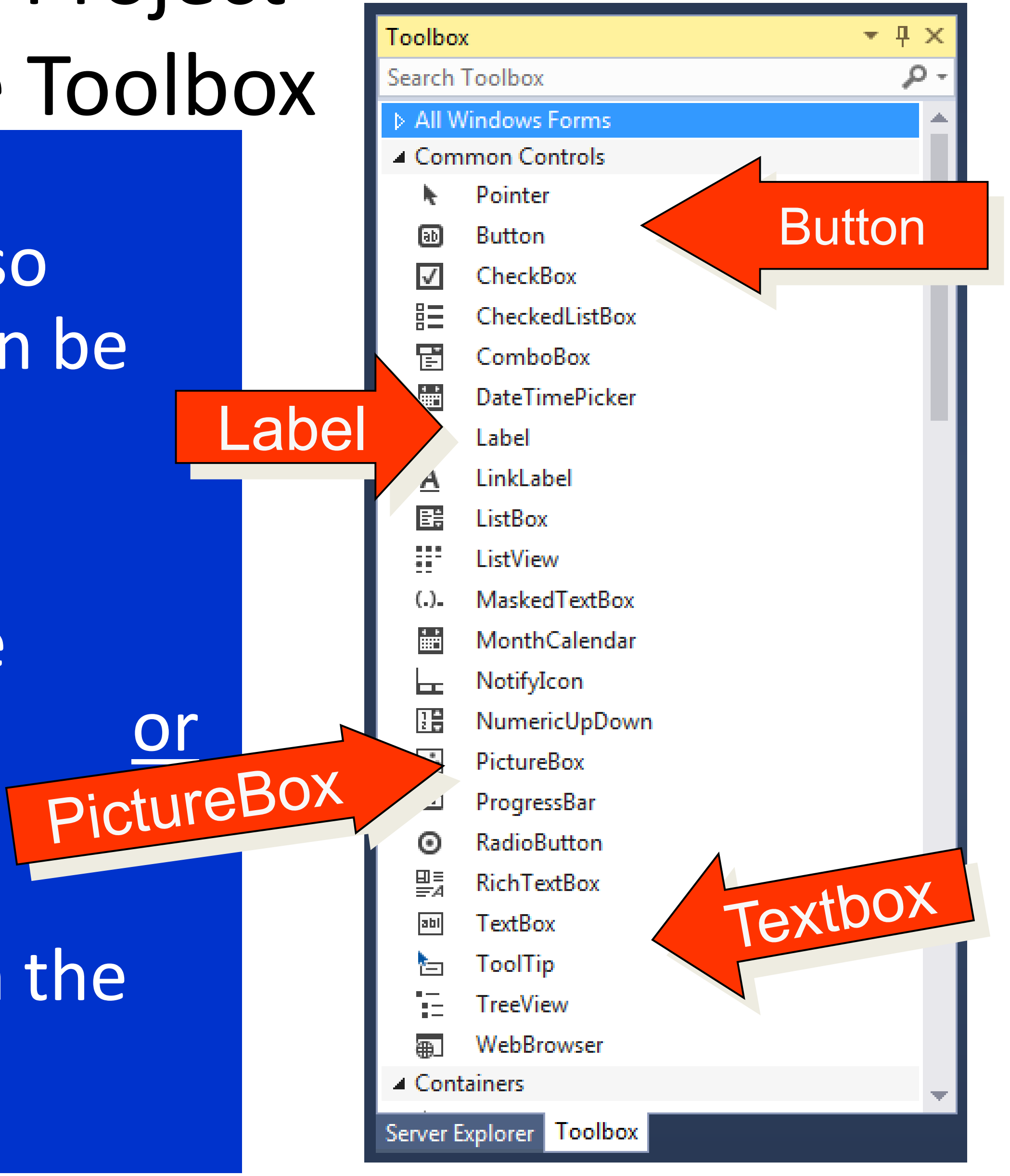
Building a Simple Project 1: The Form

- An empty form is always the start object for your .NET C# project
- You can change the form properties
- You can add other objects (buttons, textboxes, etc) to the form
- Then change their properties
- Then you can add code to object methods to make them do something useful

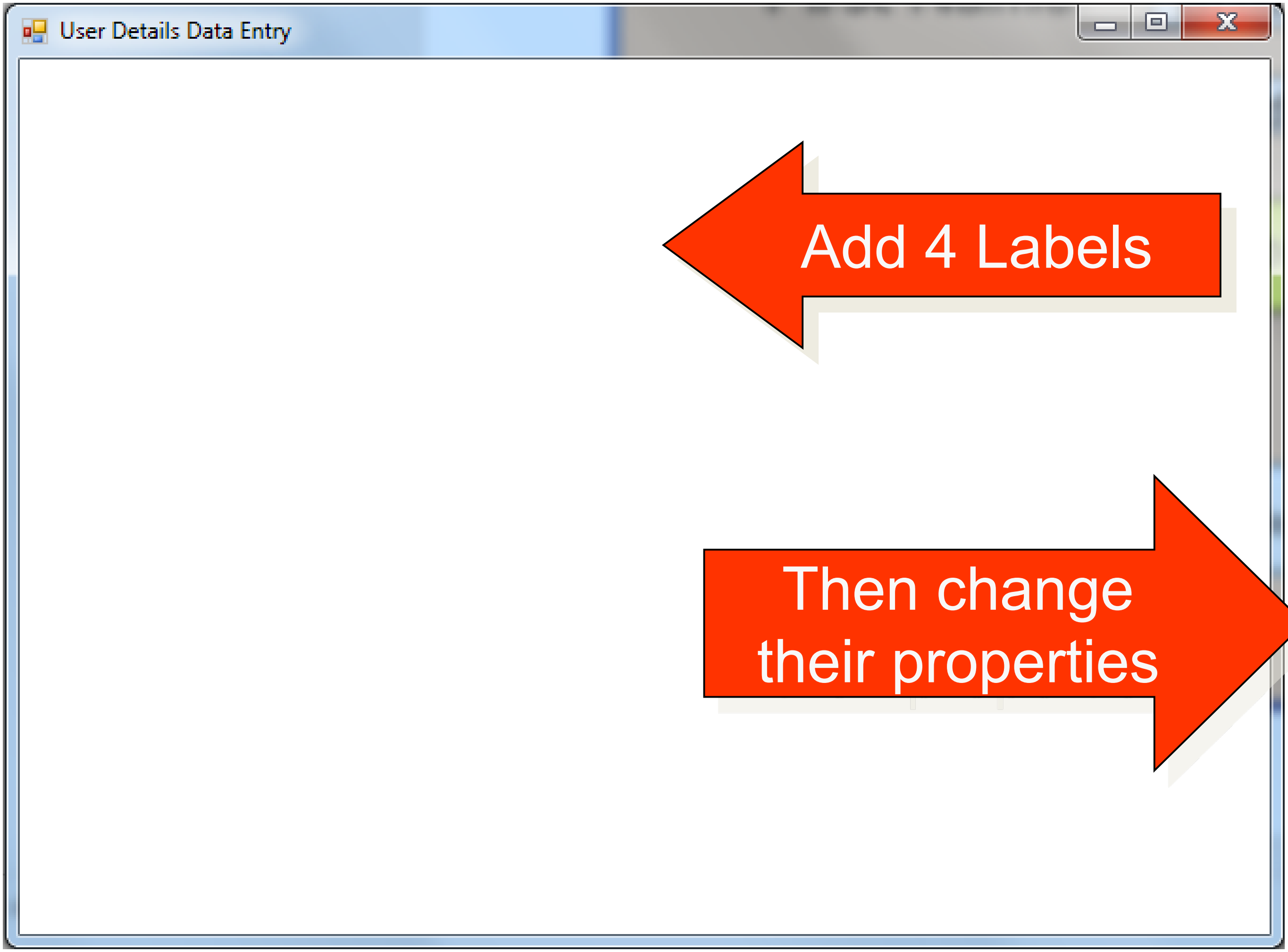


Building a Project
2 : Use the Toolbox

- The Toolbox provides a selection of objects (also called Controls) that can be added to a form
- Either double-click the appropriate tool on the toolbox or click tool and drag open an appropriate size box on the form

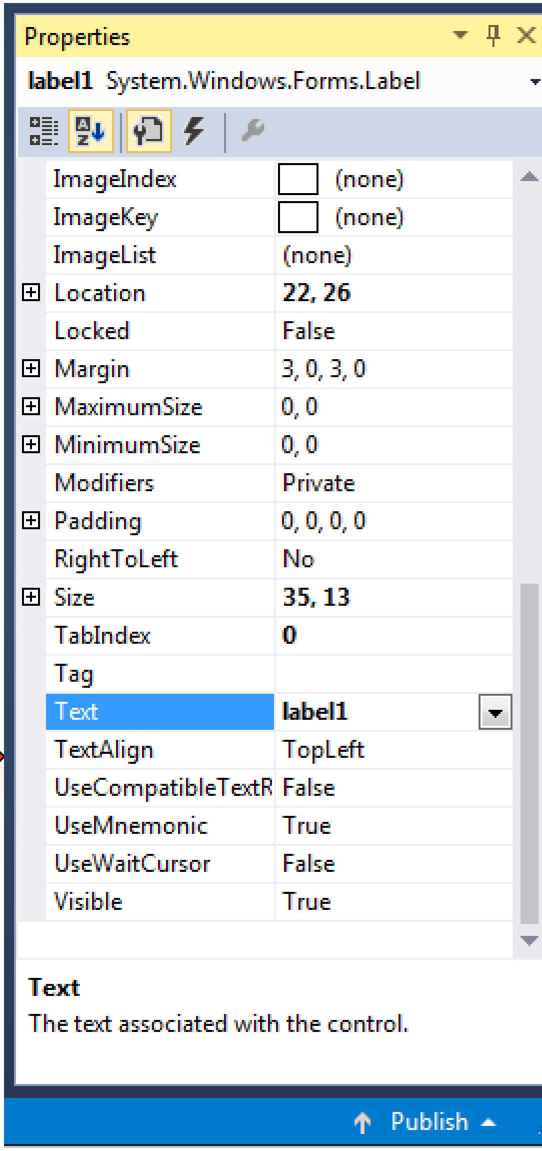


Building a Simple Project
3 : Add Some Labels



Add 4 Labels

Then change their properties

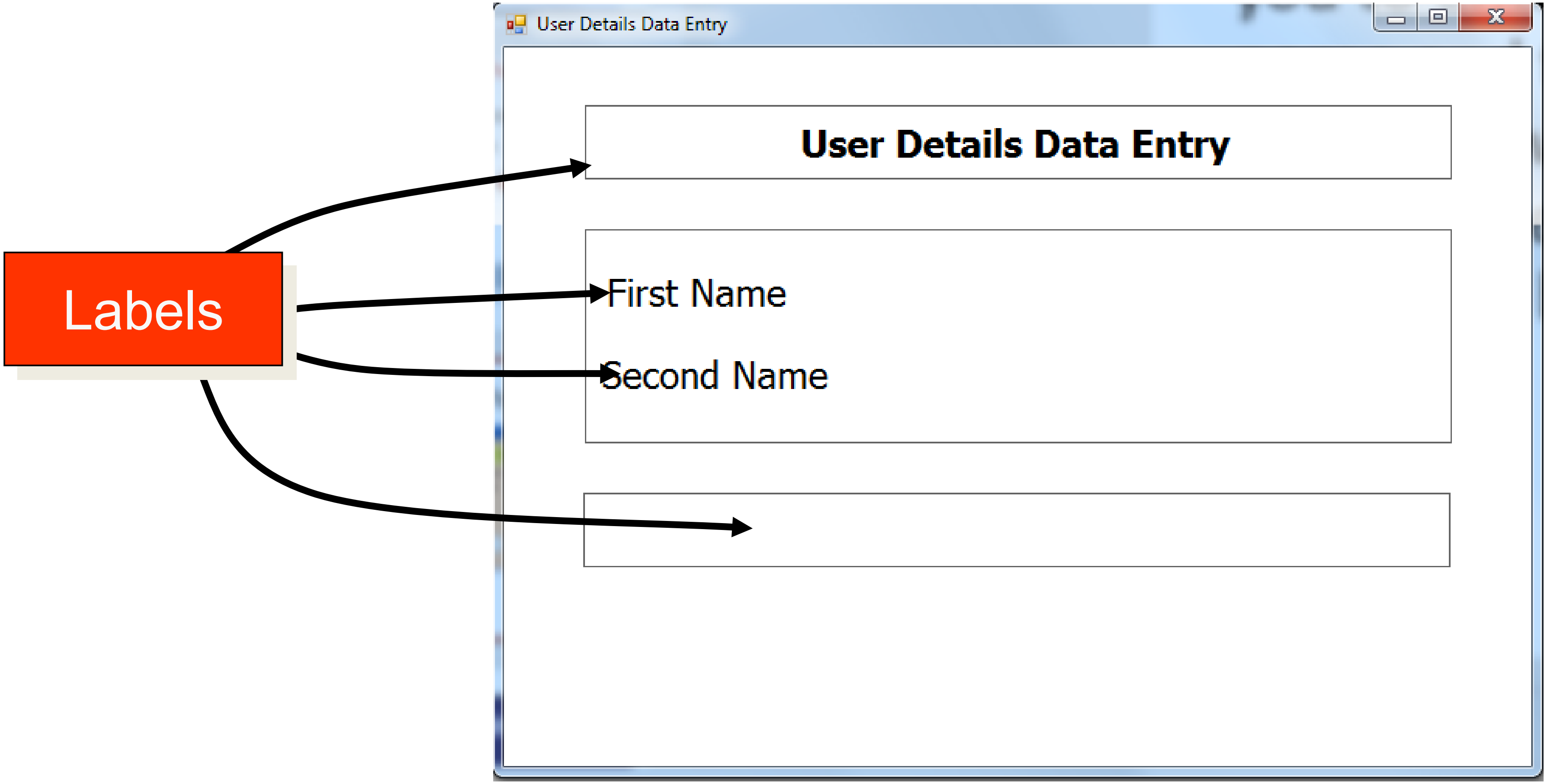

Properties

label1 System.Windows.Forms.Label

ImageIndex	□ (none)
ImageKey	□ (none)
ImageList	(none)
⊞ Location	**22, 26**
Locked	False
⊞ Margin	3, 0, 3, 0
⊞ MaximumSize	0, 0
⊞ MinimumSize	0, 0
Modifiers	Private
⊞ Padding	0, 0, 0, 0
RightToLeft	No
⊞ Size	**35, 13**
TabIndex	**0**
Tag	
Text	**label1**
TextAlign	TopLeft
UseCompatibleTextR	False
UseMnemonic	True
UseWaitCursor	False
Visible	True

Text
The text associated with the control.

Building a Simple Project
4 : Change the Properties



User Details Data Entry

First Name

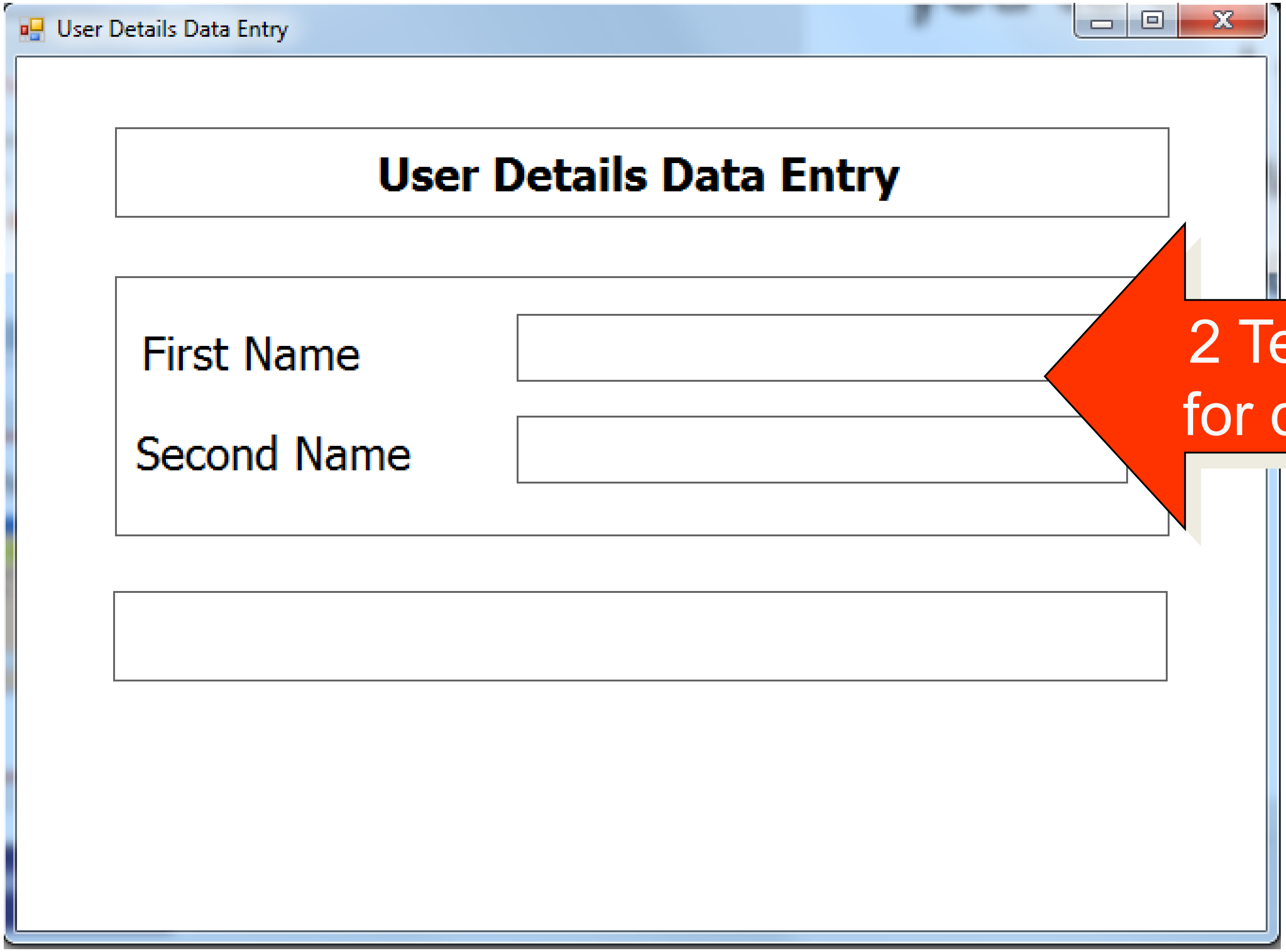
Second Name

Labels

Here we have changed the Font, Font Size, Text, BorderStyle AutoSize and BackColor properties for the 4 labels

Building a Simple Project
5 : Add Text Boxes

and change their properties



User Details Data Entry

User Details Data Entry

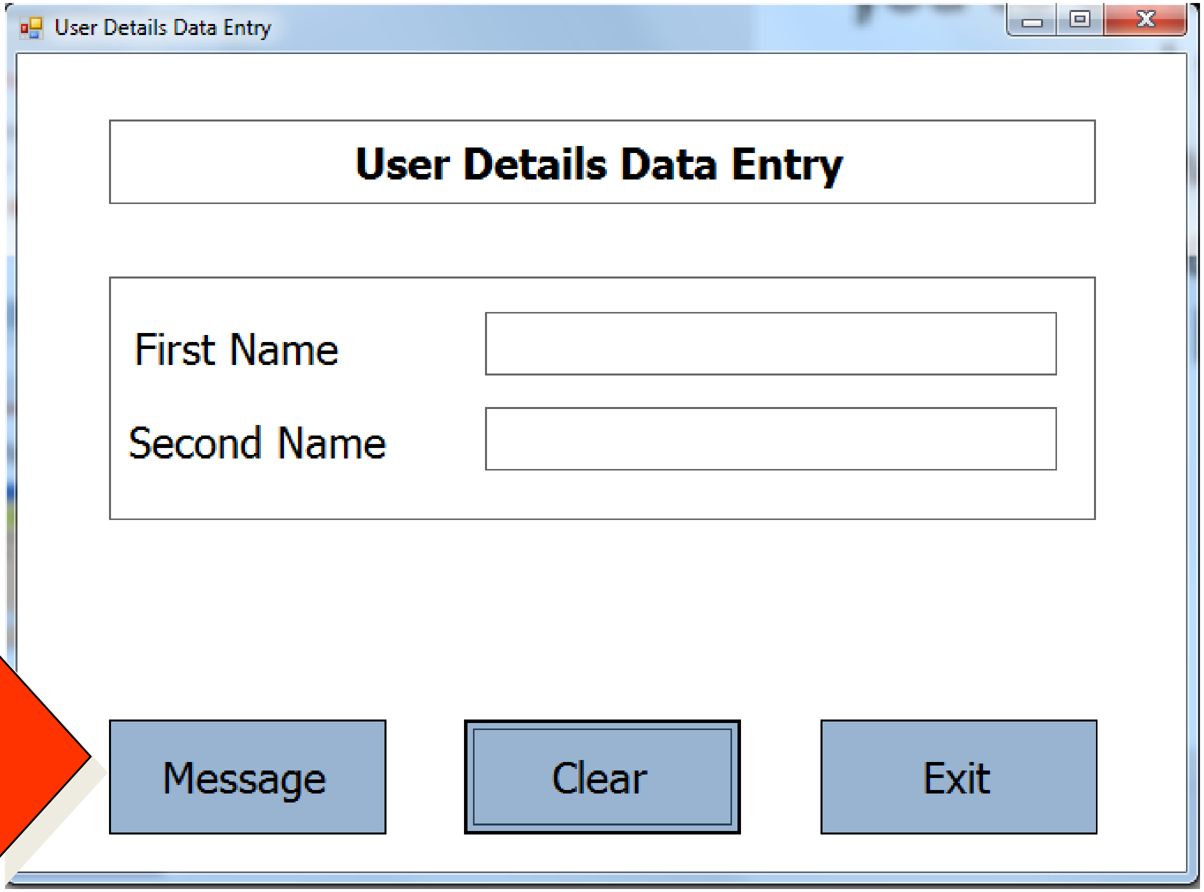
First Name

Second Name

2 Text Boxes for data entry

Building a Simple Project
6 : Add Buttons

And change their properties



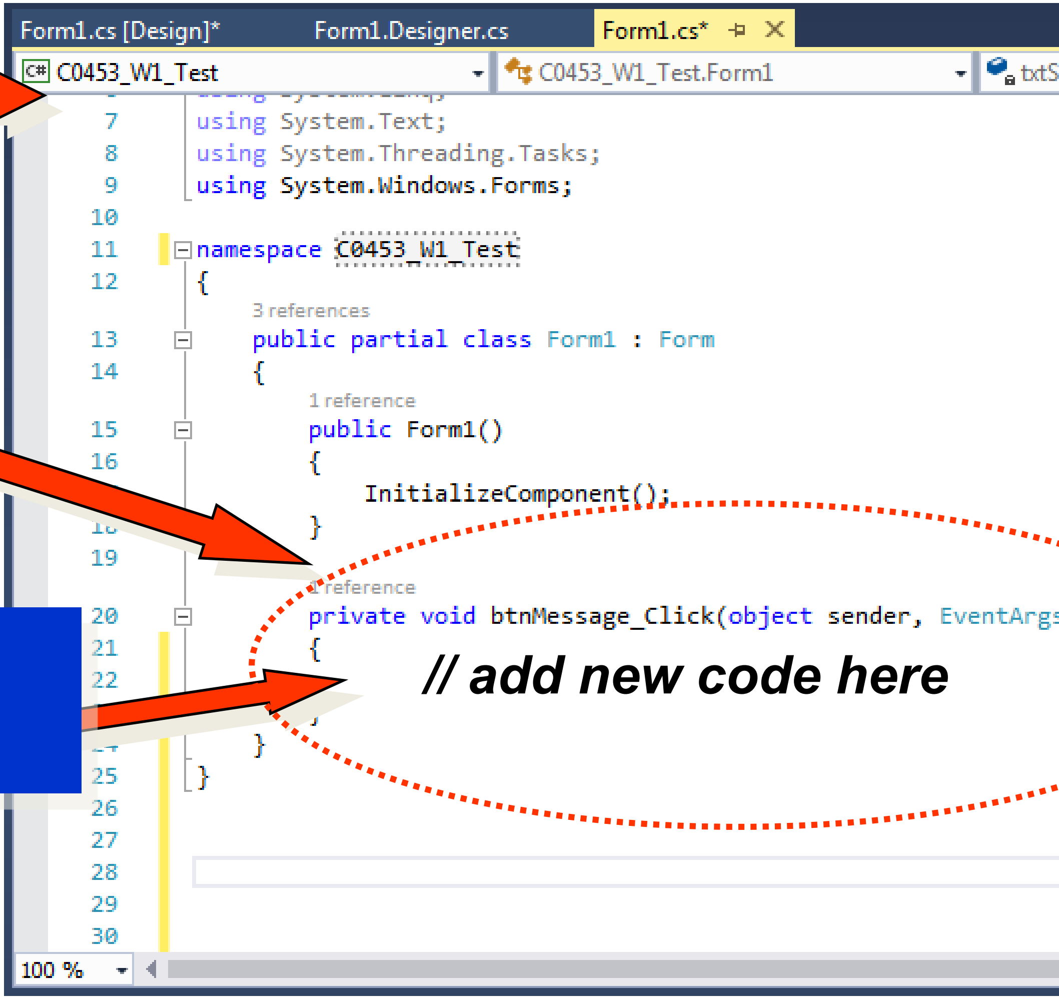
3 Buttons

Building a Simple Project
7 : Add Code to Objects

Double-Click objects to bring up the Code Window

Objects have built-in templates for methods e.g. **btnQuit_Click()**

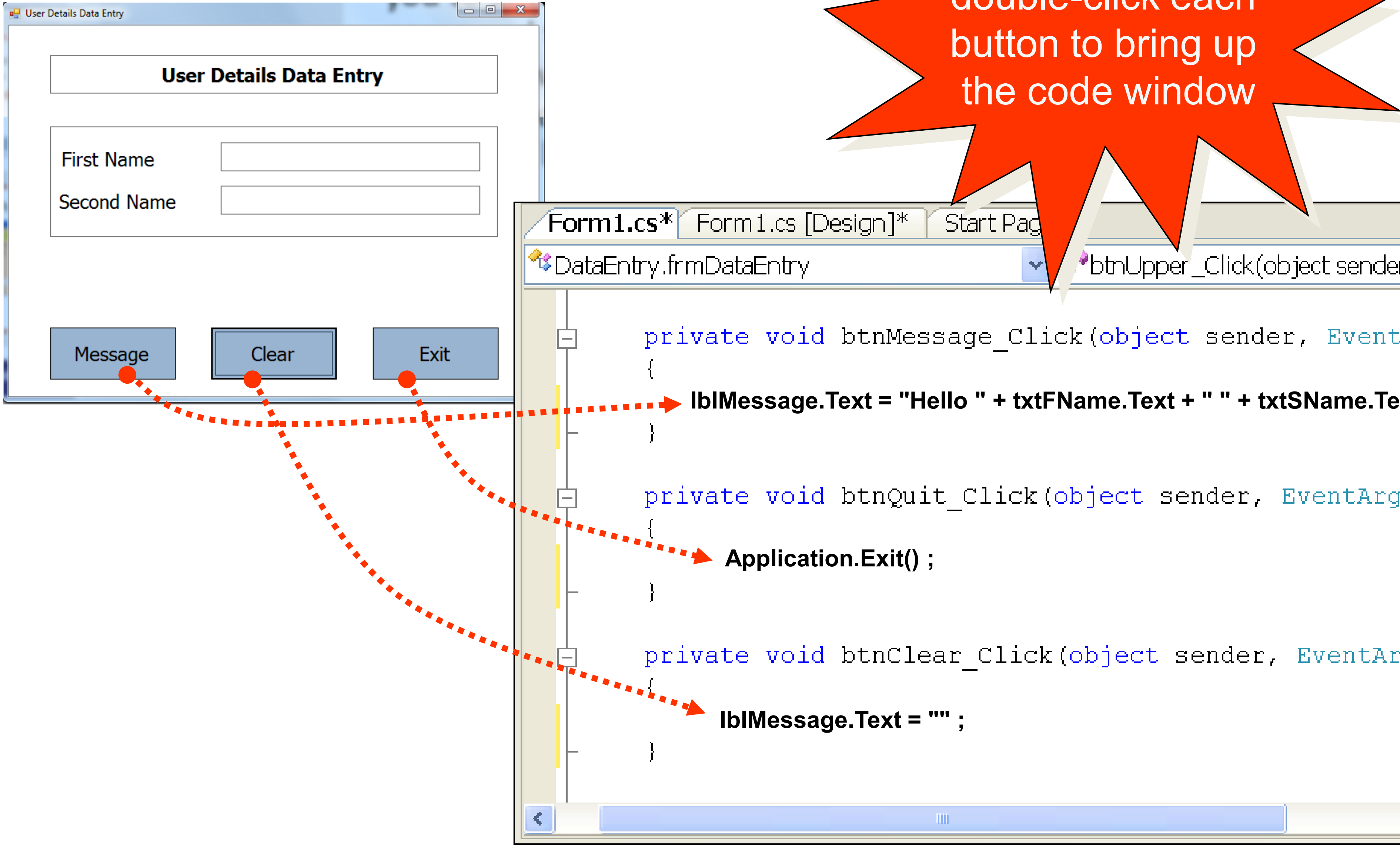
Add code to the appropriate method template



```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace C0453_W1_Test
{
    3 references
    public partial class Form1 : Form
    {
        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void btnMessage_Click(object sender, EventArgs
        {
        }
    }
}
```

// add new code here

Adding Code to Buttons



User Details Data Entry

User Details Data Entry

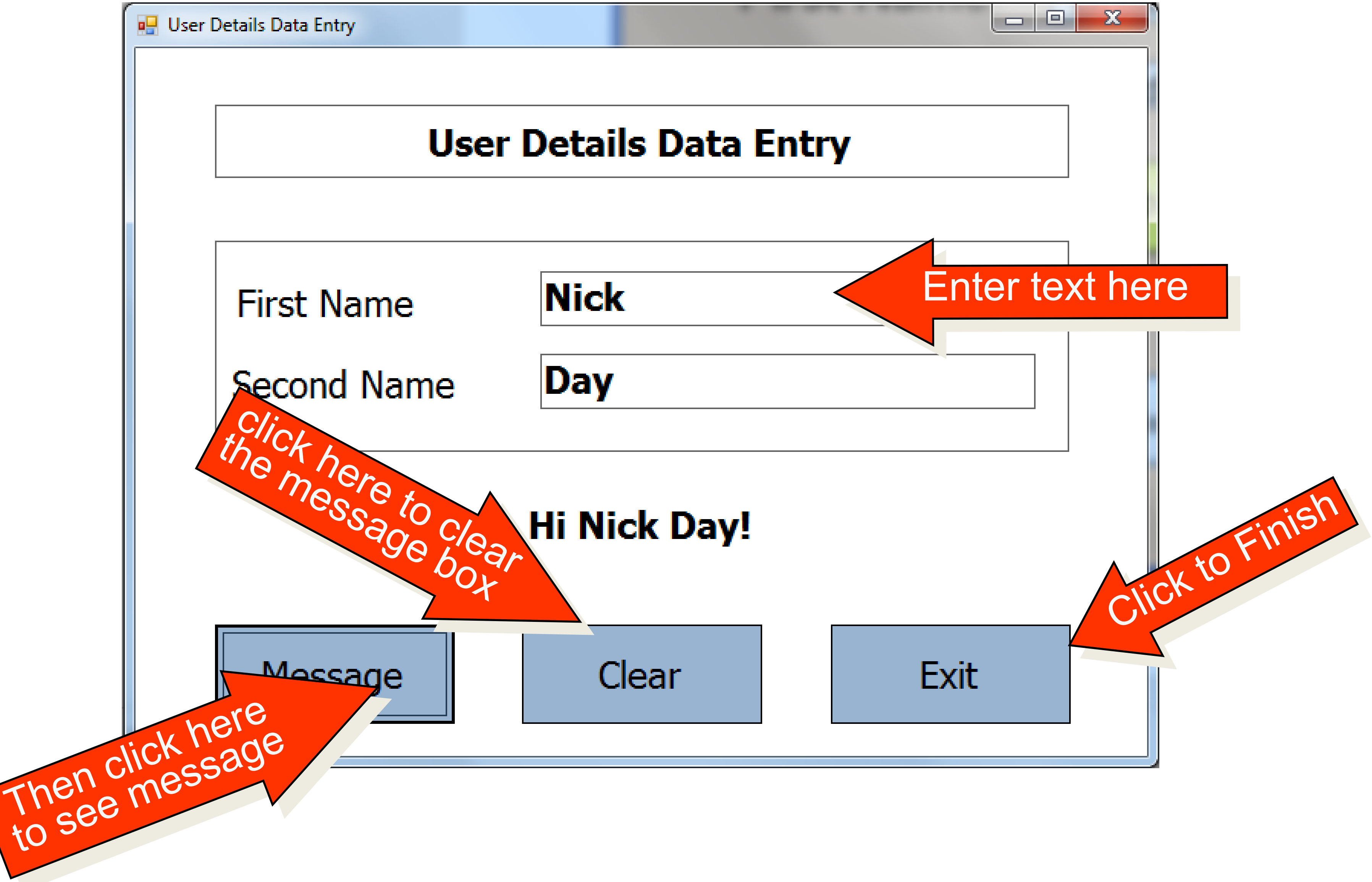
First Name

Second Name

```
    private void btnMessage_Click(object sender, Event
    {
      lblMessage.Text = "Hello " + txtFName.Text + " " + txtSName.Te
    }

    private void btnQuit_Click(object sender, EventArg
    {
        Application.Exit() ;
    }

    private void btnClear_Click(object sender, EventAr
    {
        lblMessage.Text = "" ;
    }
```

double-click each button to bring up the code window

Building a Simple Project
8 : Run it



User Details Data Entry

User Details Data Entry

First Name: Nick

Second Name: Day

Hi Nick Day!

Enter text here

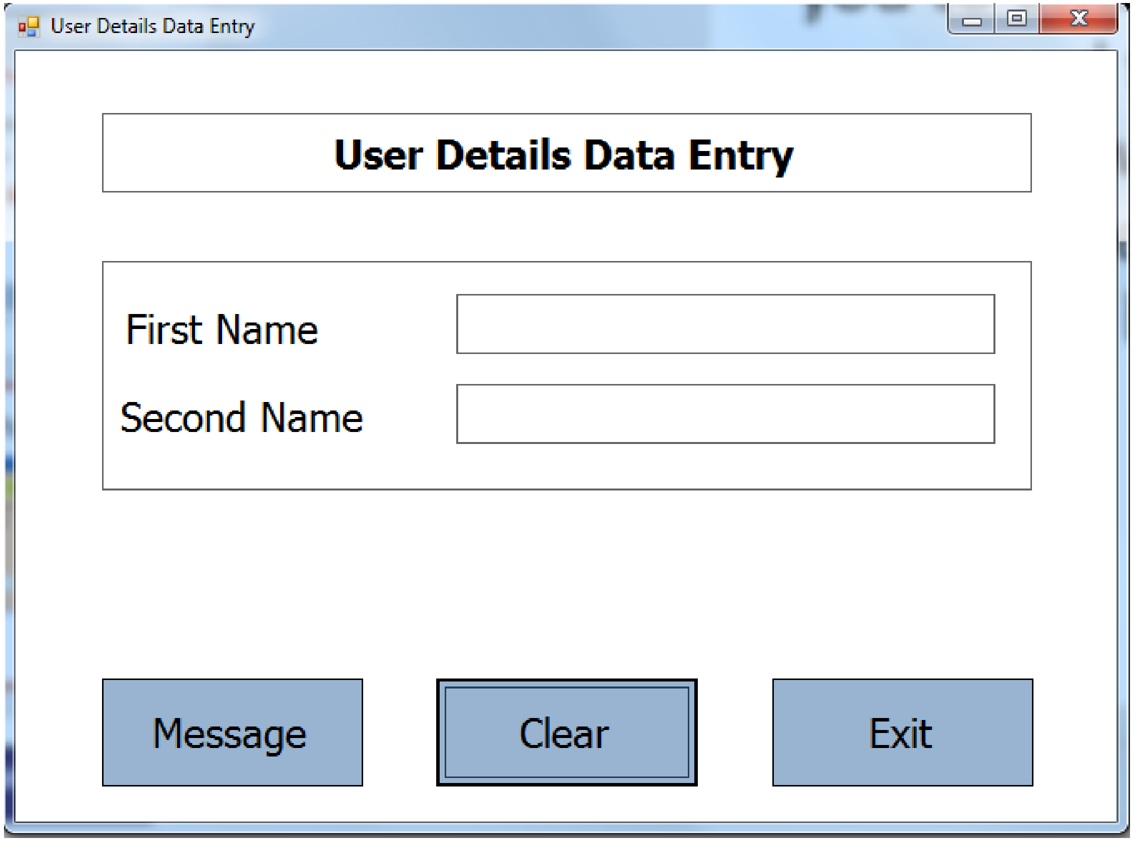
click here to clear the message box

Click to Finish

Then click here to see message

Activity

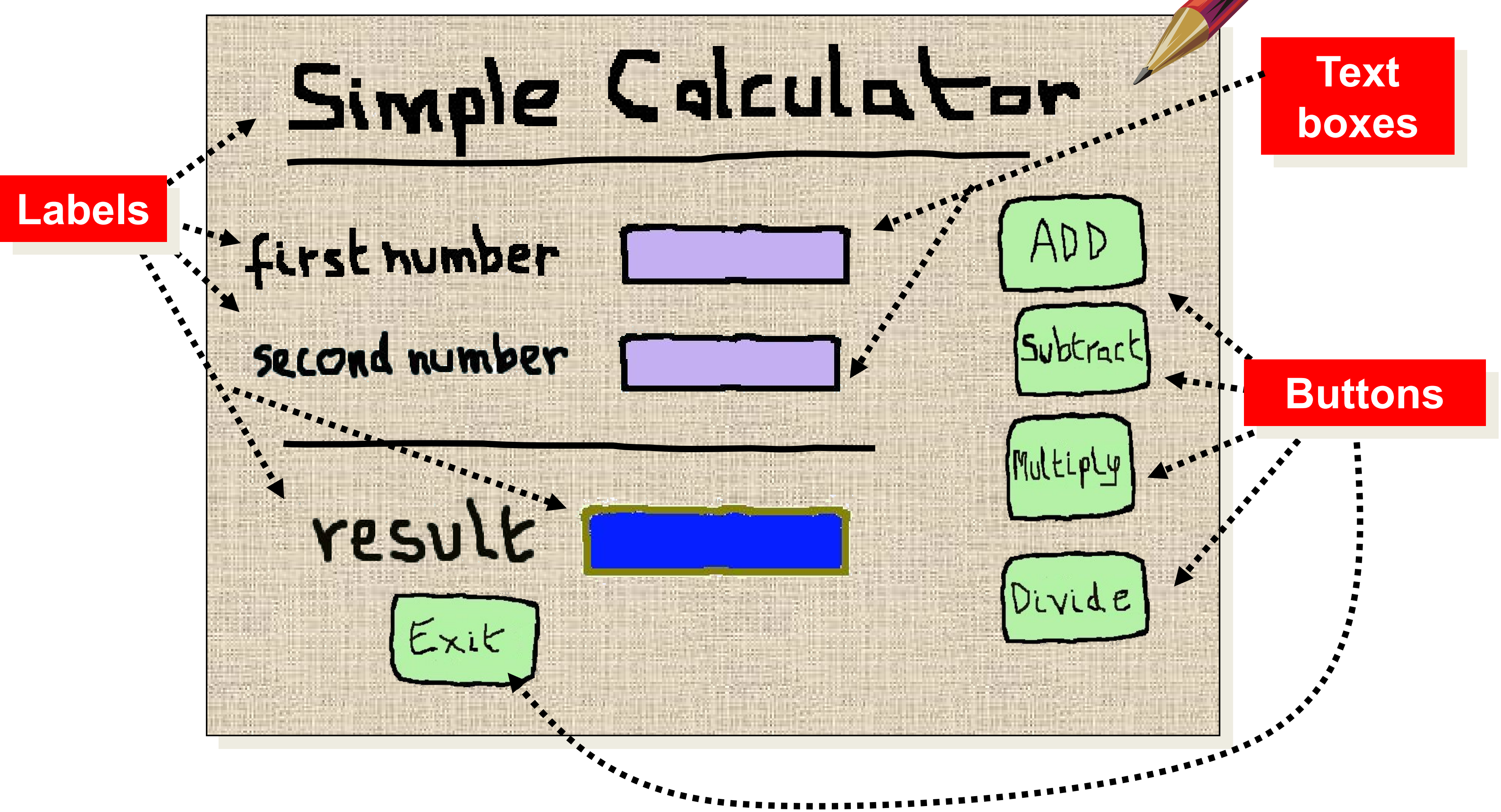
- Try and produce a similar form. The form should allow you to enter your name and then output it when clicking the ‘message’ button (Task 1.4)



Designing
a
C# Windows project

(A simple calculator)

1: Make a Sketch

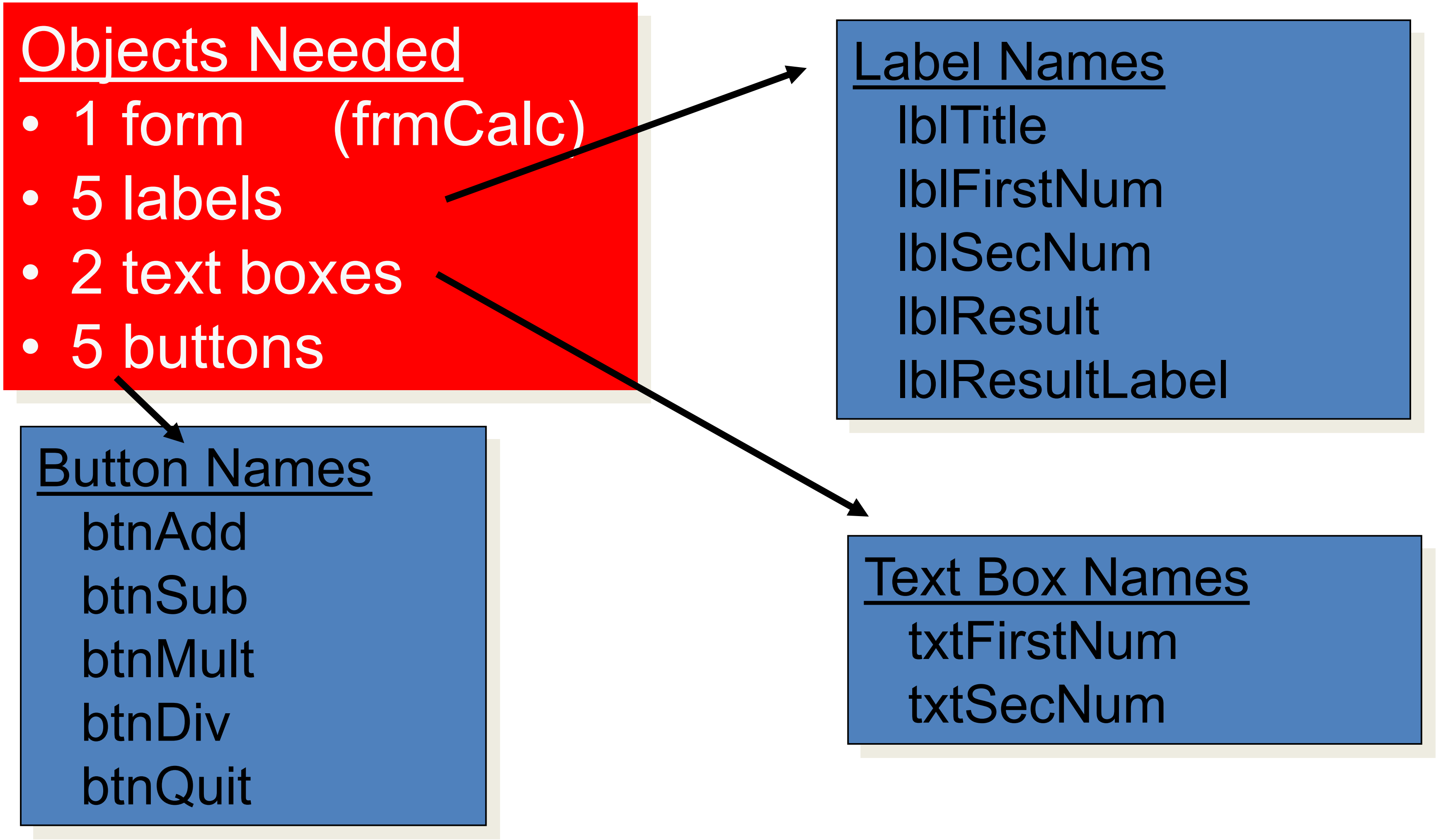


Labels

Text boxes

Buttons

2a: List Objects Needed and give them names



Objects Needed

- 1 form (frmCalc)
- 5 labels
- 2 text boxes
- 5 buttons

Label Names

lblTitle
lblFirstNum
lblSecNum
lblResult
lblResultLabel

Button Names

btnAdd
btnSub
btnMult
btnDiv
btnQuit

Text Box Names

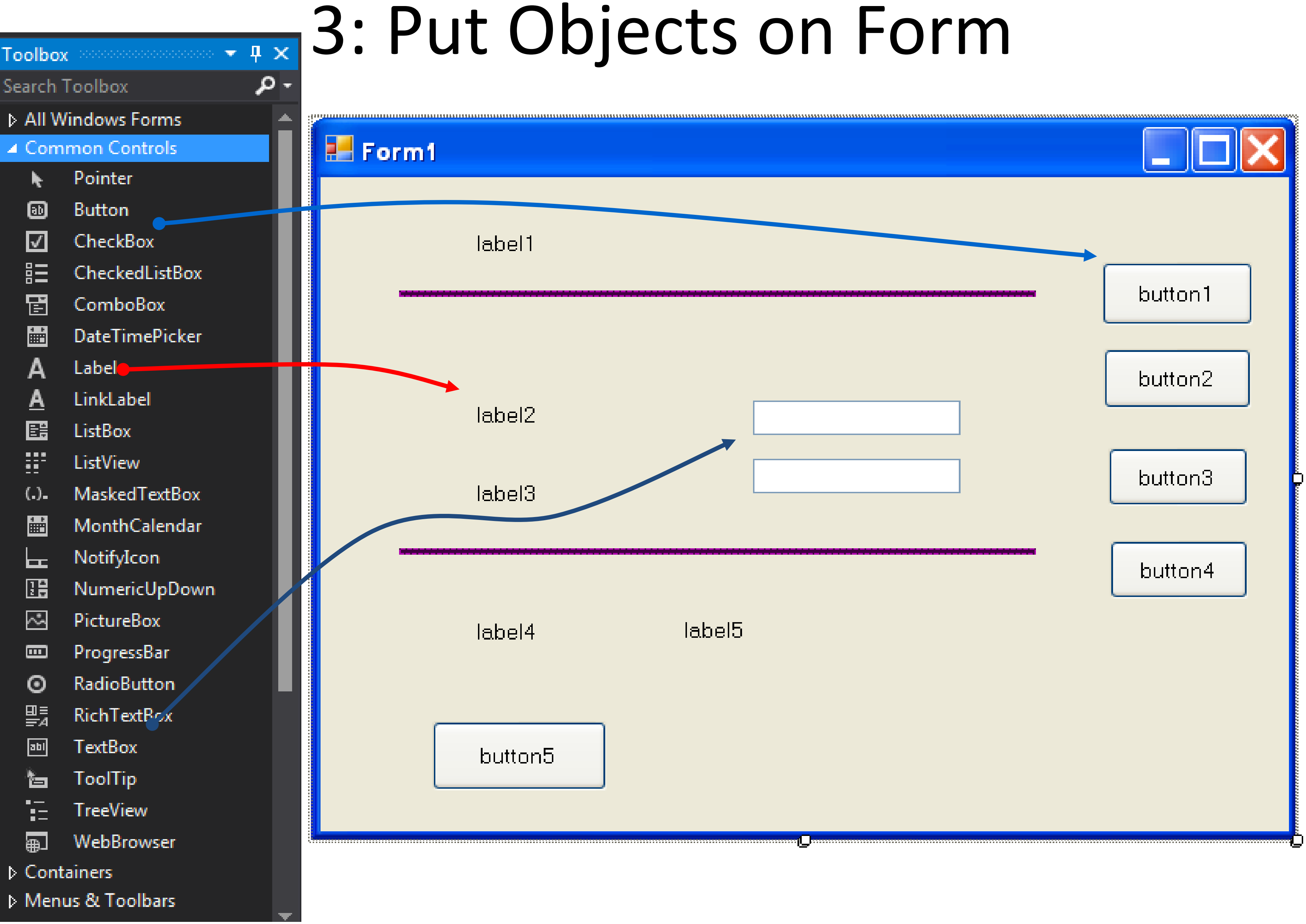
txtFirstNum
txtSecNum

2b: TOE Chart

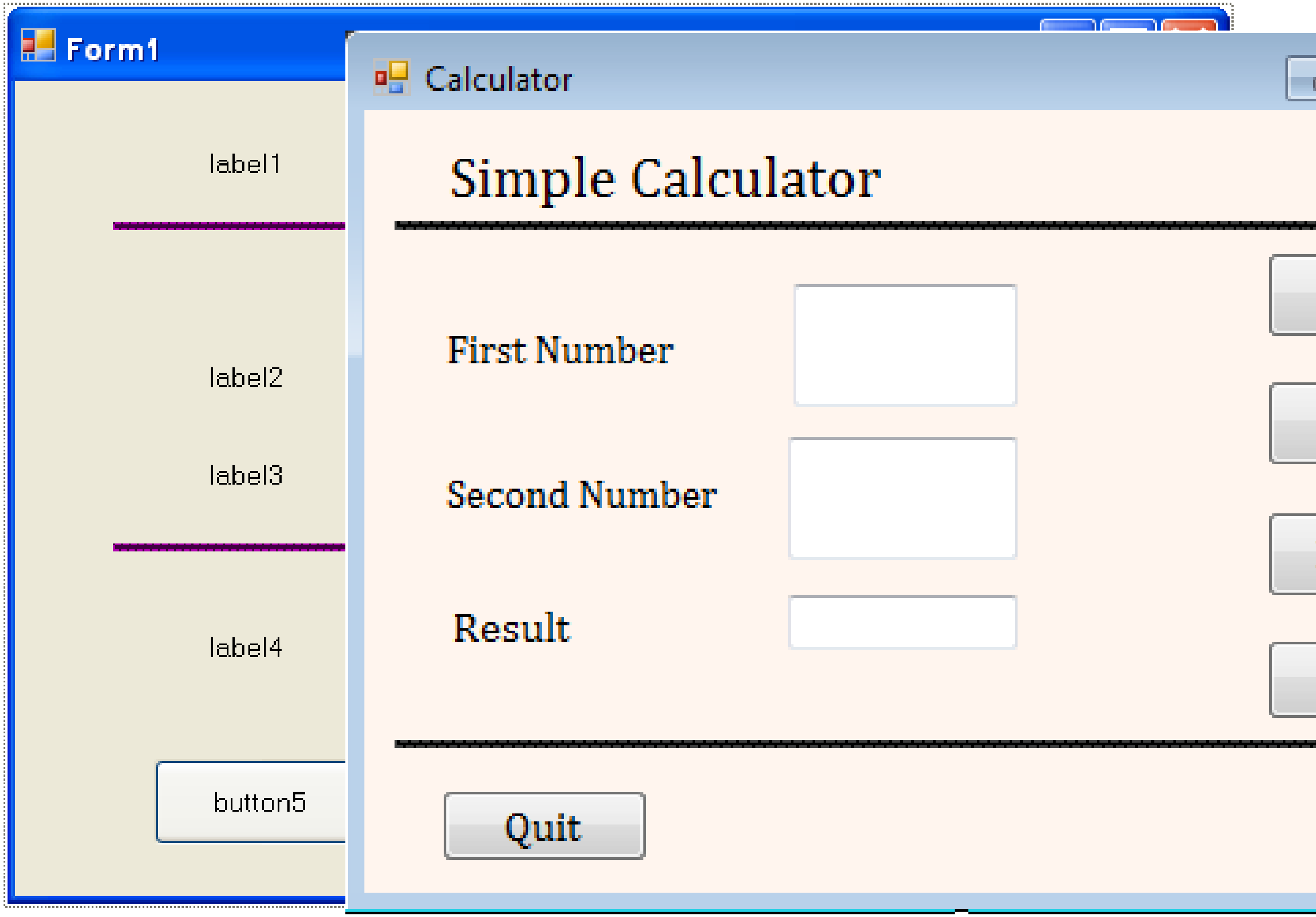
	Task	Object	Event
1	Display Interface	frmCalc	
2	Enter first number	txtFirstNum	
3	Enter second number	txtSecNum	
4	Perform addition	btnAdd	Click
5	Perform subtraction	btnSub	Click
6	Perform multiplication	btnMult	Click
7	Perform division	btnDiv	Click
8	Display Result	lblResult	
9	Quit Program	btnQuit	Click

Other objects on the form have no active role (decoration only)

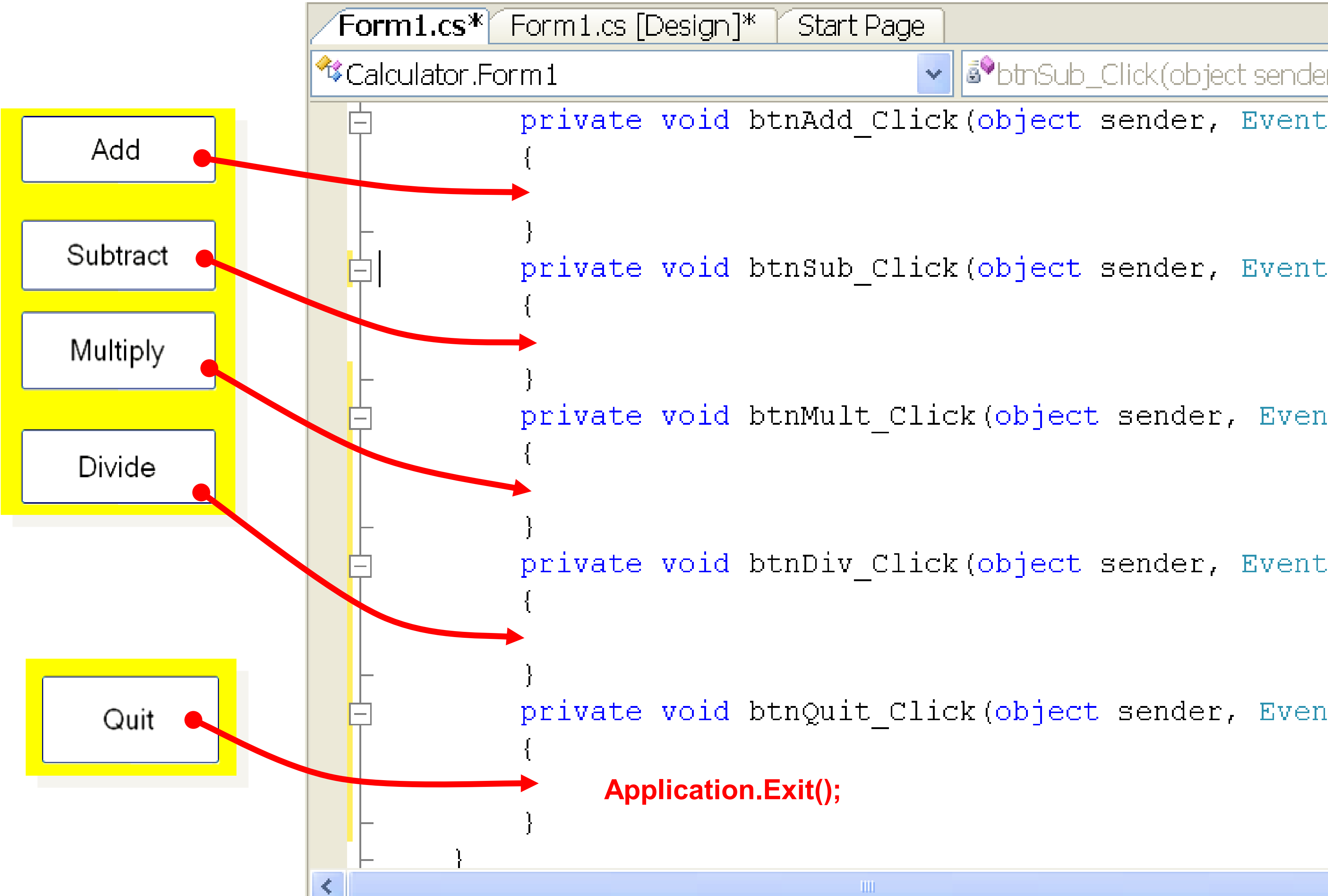
3: Put Objects on Form



4: Change Object Properties

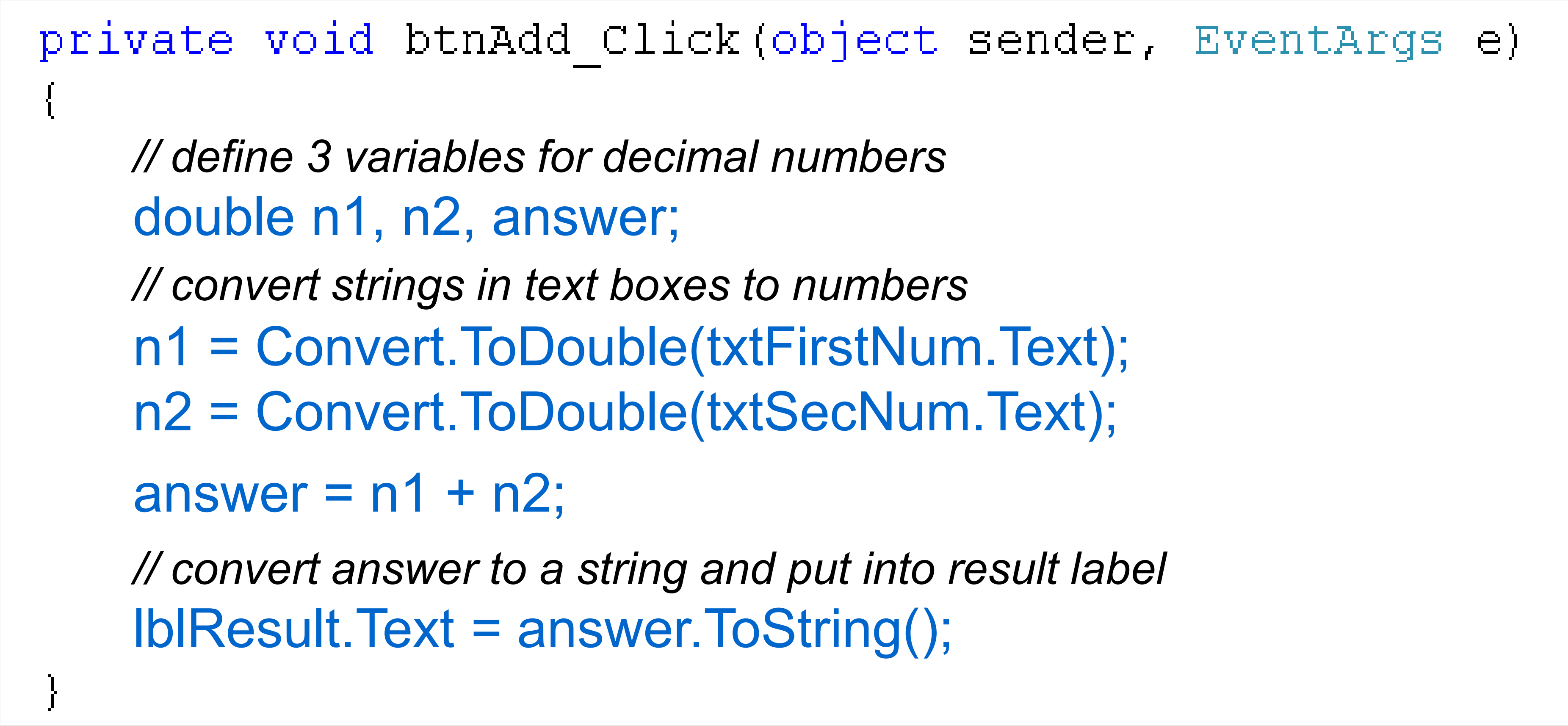


5: Add Code to Buttons



```
        private void btnAdd_Click(object sender, Event
        {

        }
        private void btnSub_Click(object sender, Event
        {

        }
        private void btnMult_Click(object sender, Even
        {

        }
        private void btnDiv_Click(object sender, Event
        {

        }
        private void btnQuit_Click(object sender, Even
        {
            Application.Exit();
        }
    }
```

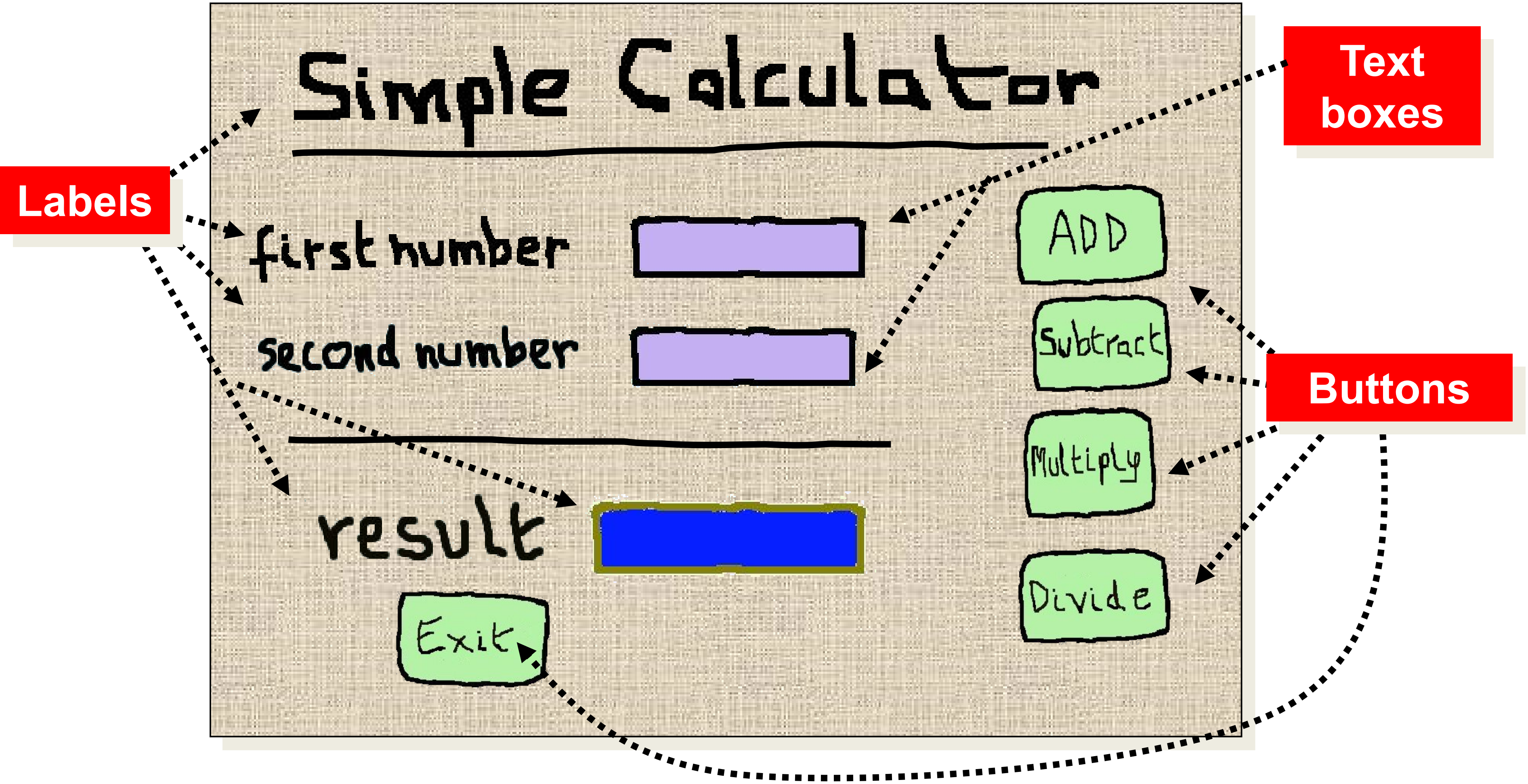
The Add button Code



```
private void btnAdd_Click(object sender, EventArgs e)
{
      // define 3 variables for decimal numbers
      double n1, n2, answer;
      // convert strings in text boxes to numbers
      n1 = Convert.ToDouble(txtFirstNum.Text);
      n2 = Convert.ToDouble(txtSecNum.Text);

      answer = n1 + n2;

      // convert answer to a string and put into result label
      lblResult.Text = answer.ToString();
}
```

Activity

Replicate the form design and functionality below:



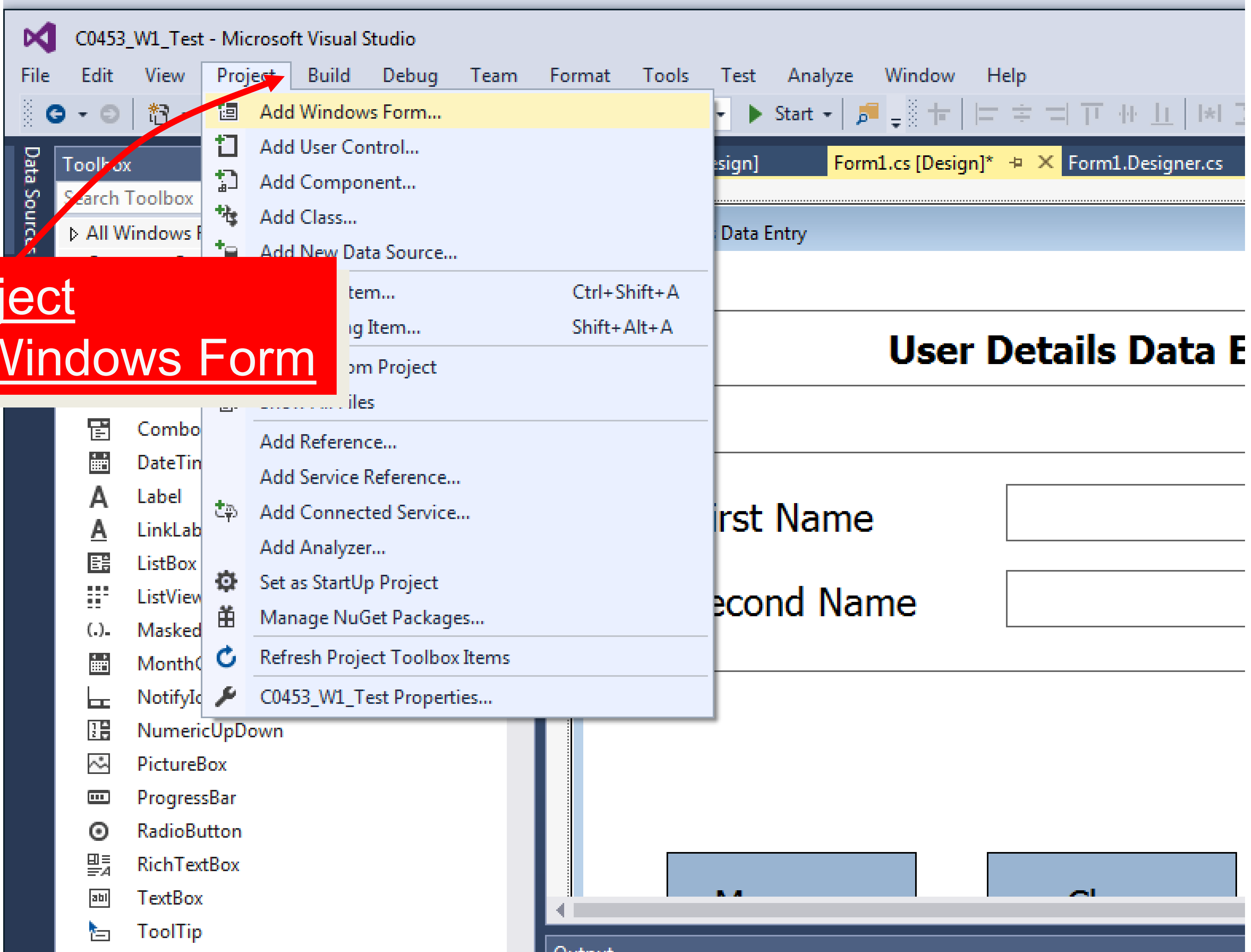
Labels

Text boxes

Buttons

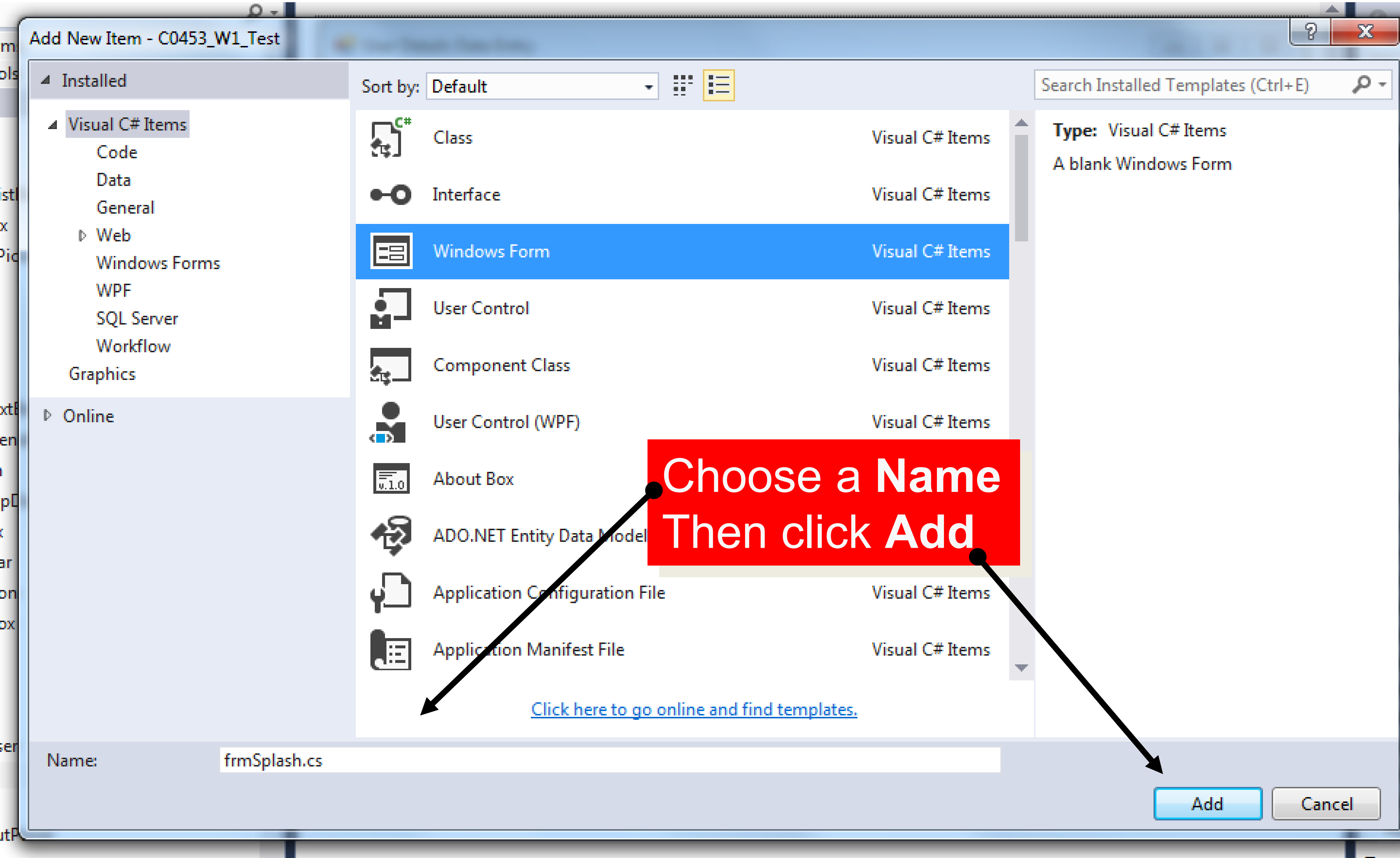
Adding
a new form
to the project
(a Splash screen)

1: Choose Add Windows Form



Select Project
then Add Windows Form

2: Name the new Form



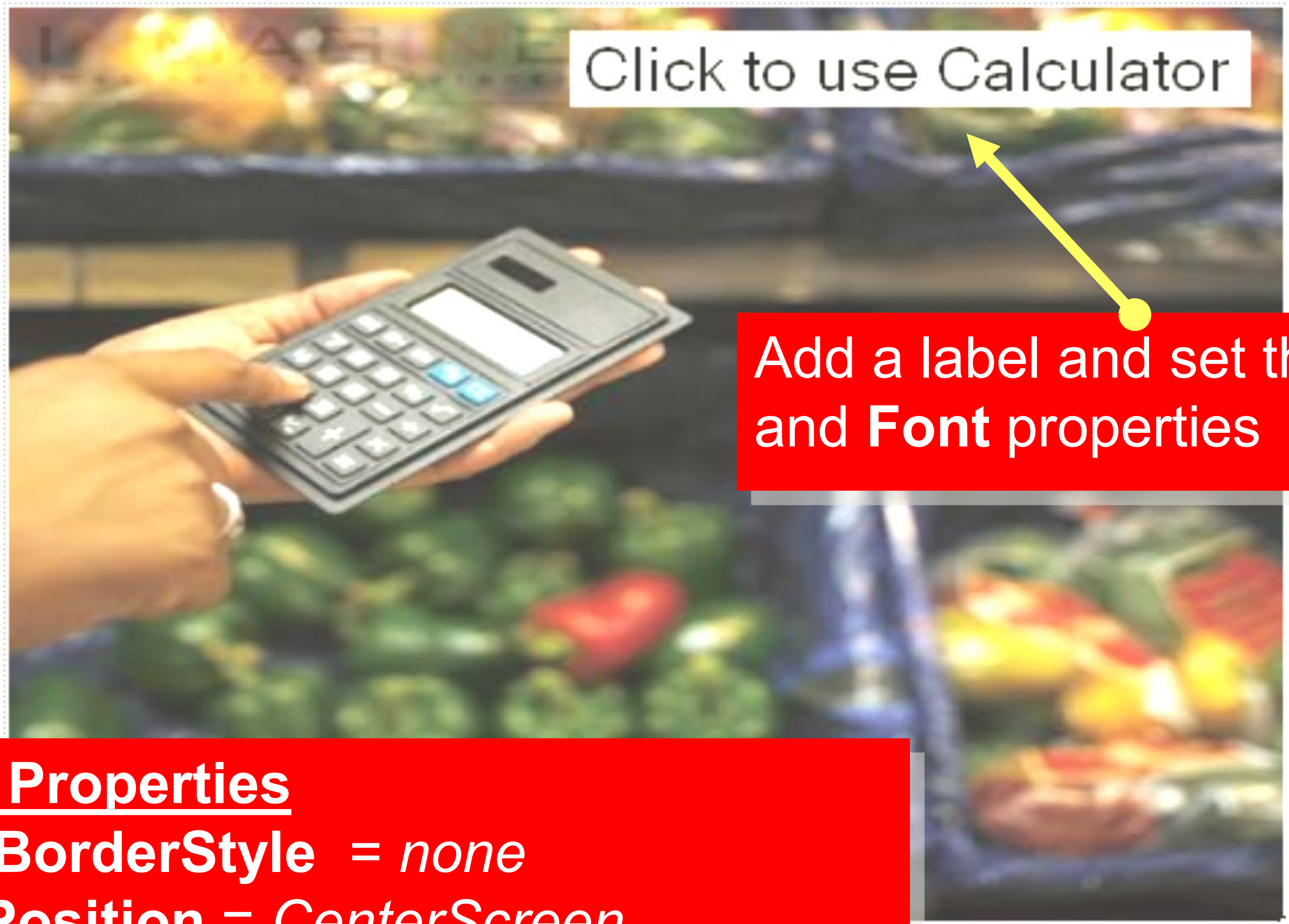
Type: Visual C# Items

A blank Windows Form

Name: frmSplash.cs

Choose a **Name**
Then click **Add**

3: Design new startup Screen



Add a label and set the **Text** and **Font** properties

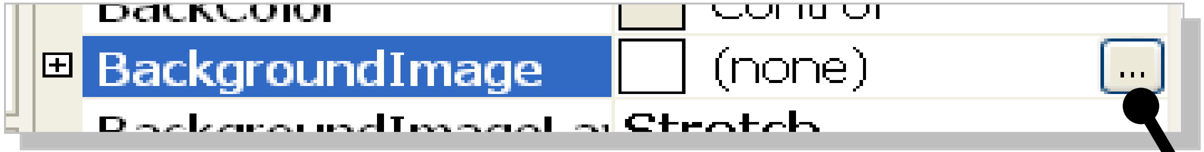
Form Properties

FormBorderStyle = *none*

StartPosition = *CenterScreen*

BackgroundImage = *< see next slide >*

4: Adding a BackgroundImage

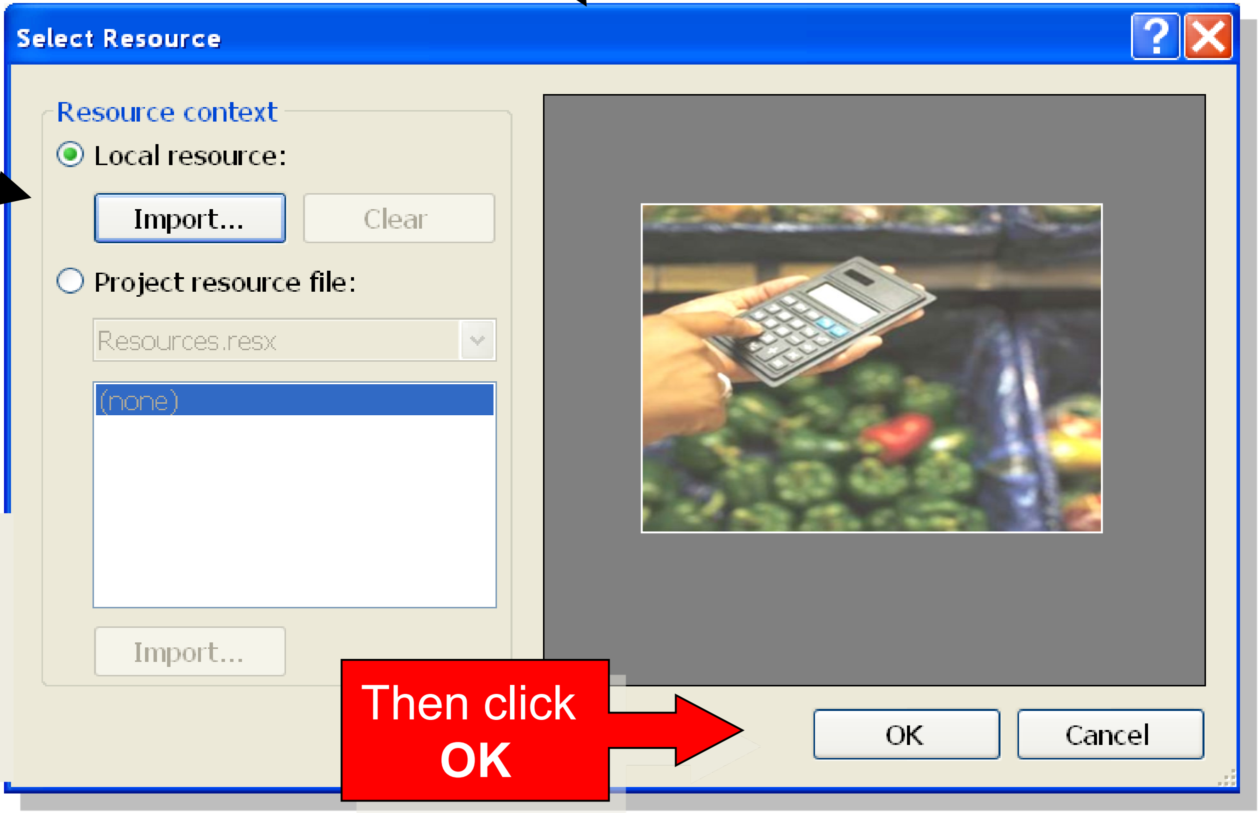


BackColor Control

⊞ BackgroundImage (none)

BackgroundImageLayout Stretch

- Click **Import** ..
- find an image file



Select Resource

Resource context

● Local resource:

○ Project resource file:

Resources.resx

(none)

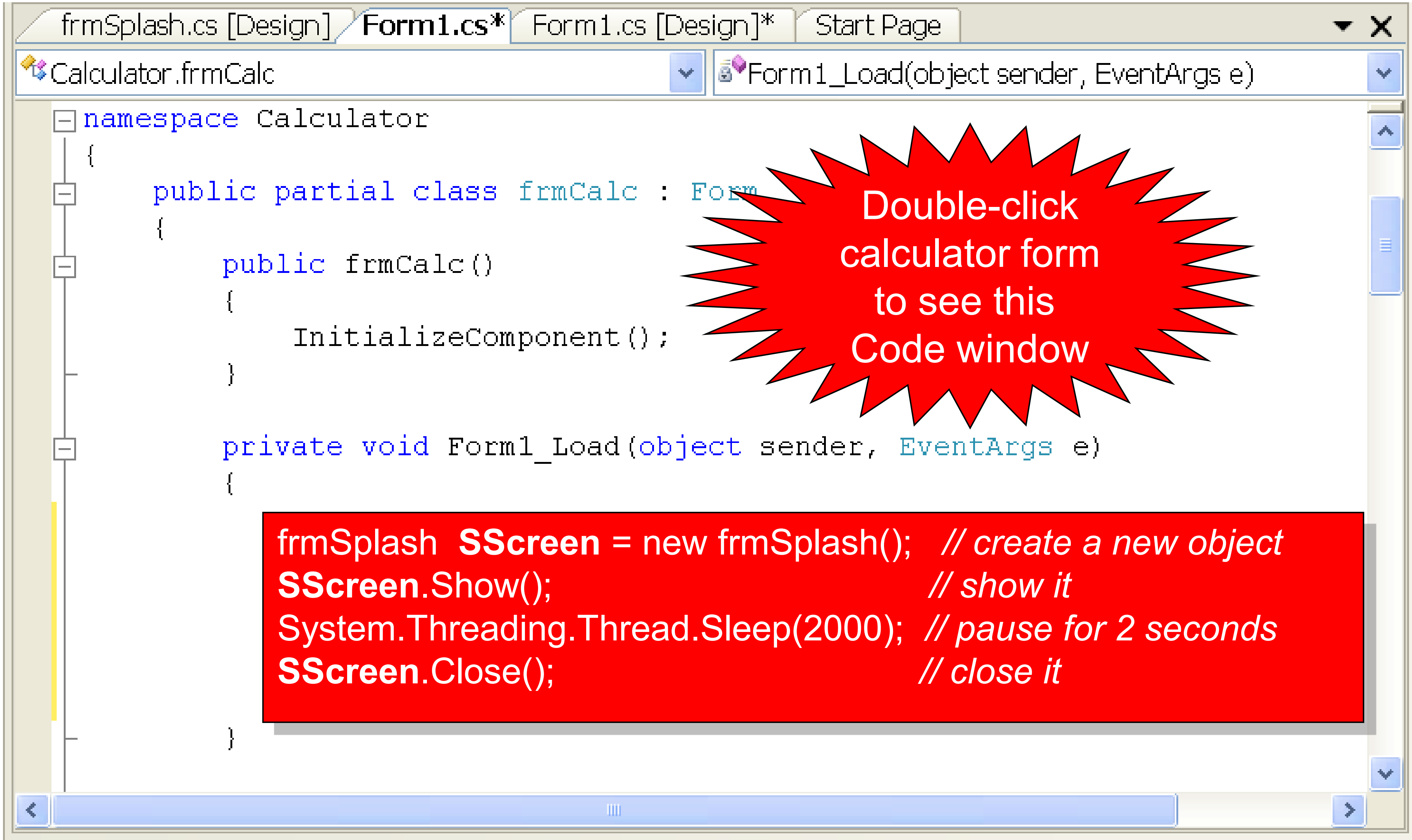
Then click **OK**

<u>Note:</u>
BackgroundImageLayout

- set this form property to: **<u>Stretch</u>** (or Tile, Center or Zoom)

5a: Coding the Calculator Form

Use Form1’s Load event to show the Splash screen first



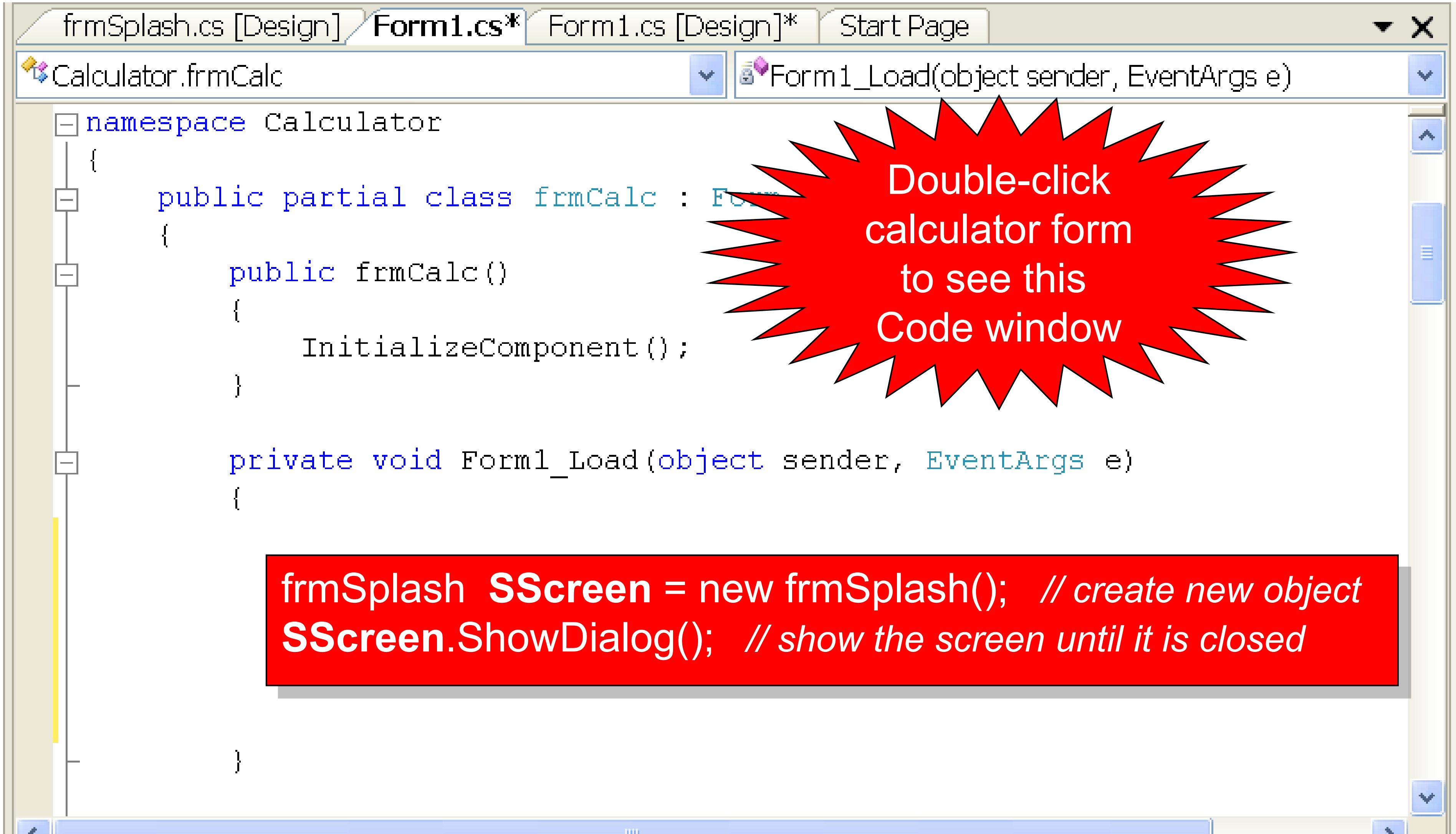
```
namespace Calculator
{
    public partial class frmCalc : Form
    {
        public frmCalc()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {

        }
```

Double-click calculator form to see this Code window

```
frmSplash  SScreen = new frmSplash();   // create a new object
SScreen.Show();                         // show it
System.Threading.Thread.Sleep(2000);    // pause for 2 seconds
SScreen.Close();                        // close it
```

5b: Alternative Coding

We want the Splash screen to show until we click it!



```
namespace Calculator
{
    public partial class frmCalc : Form
    {
        public frmCalc()
        {
            InitializeComponent();
        }

        private void Form1_Load(object sender, EventArgs e)
        {



        }
```

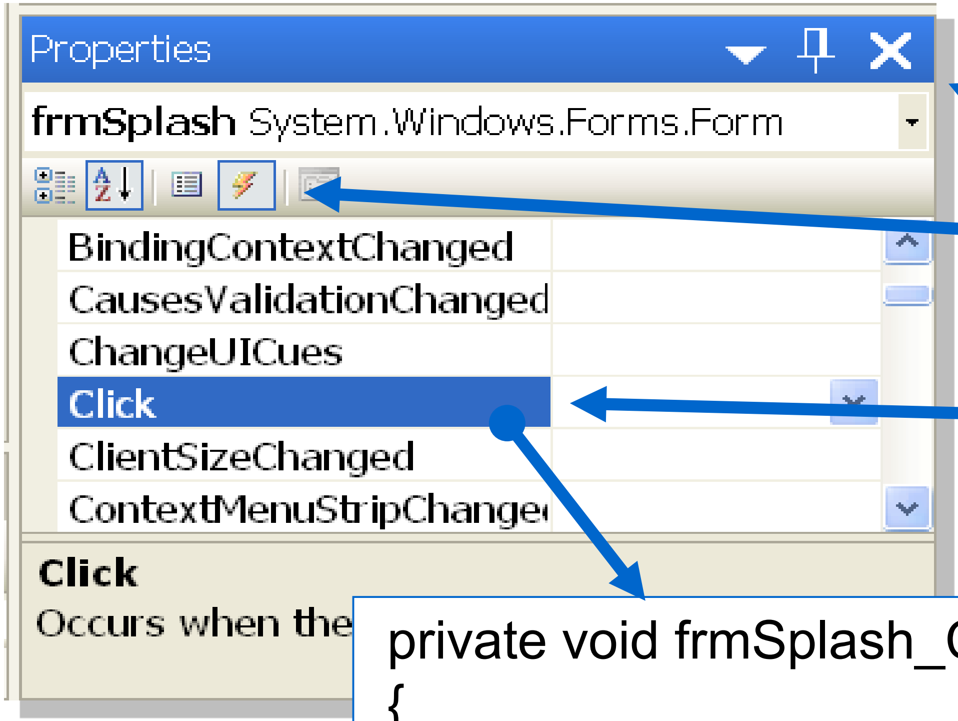
Double-click calculator form to see this Code window

frmSplash **SScreen** = new frmSplash(); *// create new object*
SScreen.ShowDialog(); *// show the screen until it is closed*

5b: Continued

Add code to the frmSplash <u>Click</u> method to close the form. But :

- **The <u>Load</u> event is a form's <u>default</u> method!!**
- **To get to its <u>Click</u> method do this:**



Properties

frmSplash System.Windows.Forms.Form

BindingContextChanged
CausesValidationChanged
ChangeUICues
Click
ClientSizeChanged
ContextMenuStripChange

Click
Occurs when the

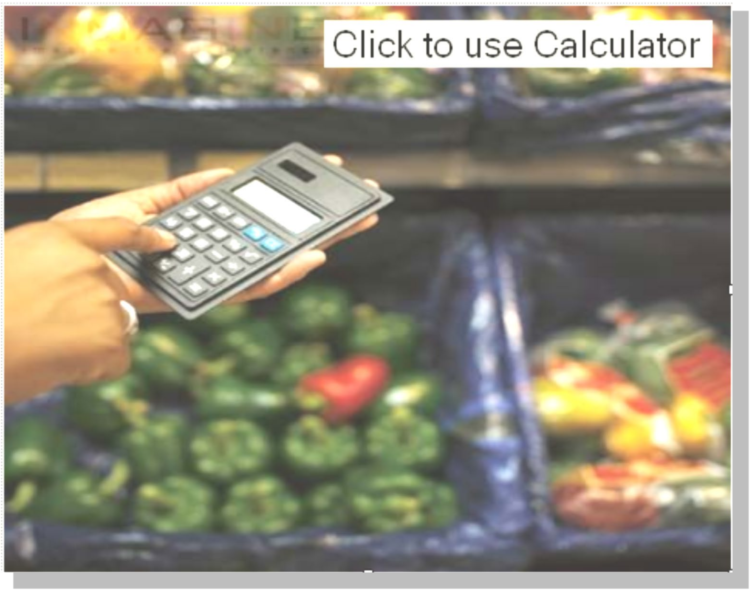
- Go to the <u>Properties</u> window
- Select the <u>Events</u> button
  (a Lightning icon)
- See a list of all form events
- Double-click the <u>Click</u> event

- Then add this code

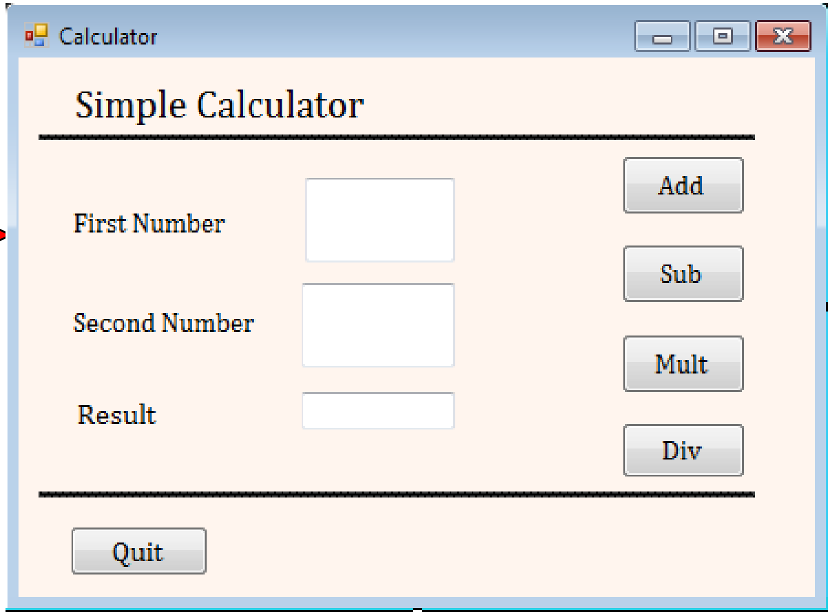
```
private void frmSplash_Click(object sender, EventArgs e)
{
        this.close();     // close this form window

}
```

Final Result

The Splash screen shows first. Click it to use the Calculator.

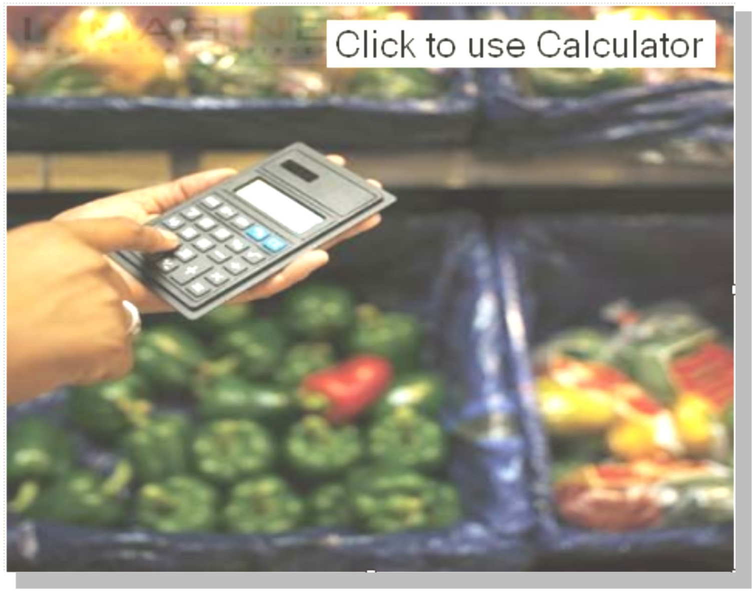


click

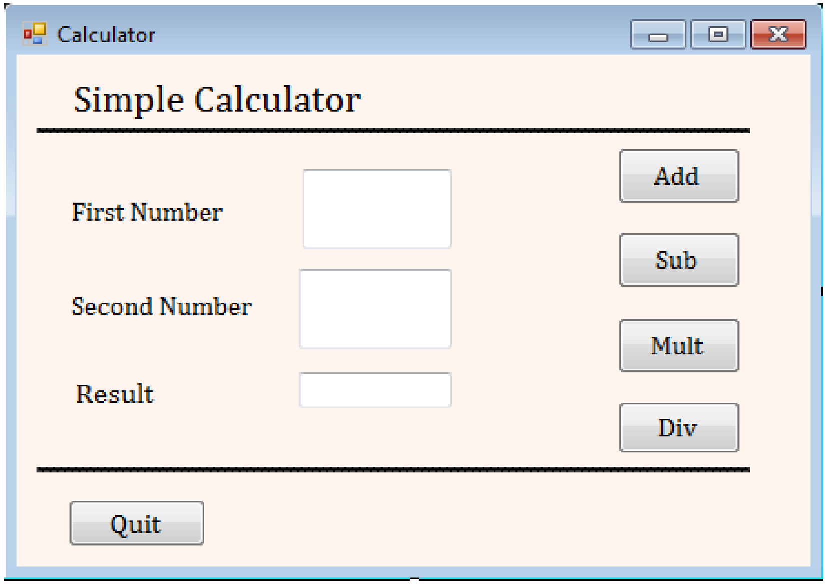


Activity

Add a splash screen to your calculator (Task 1.6)



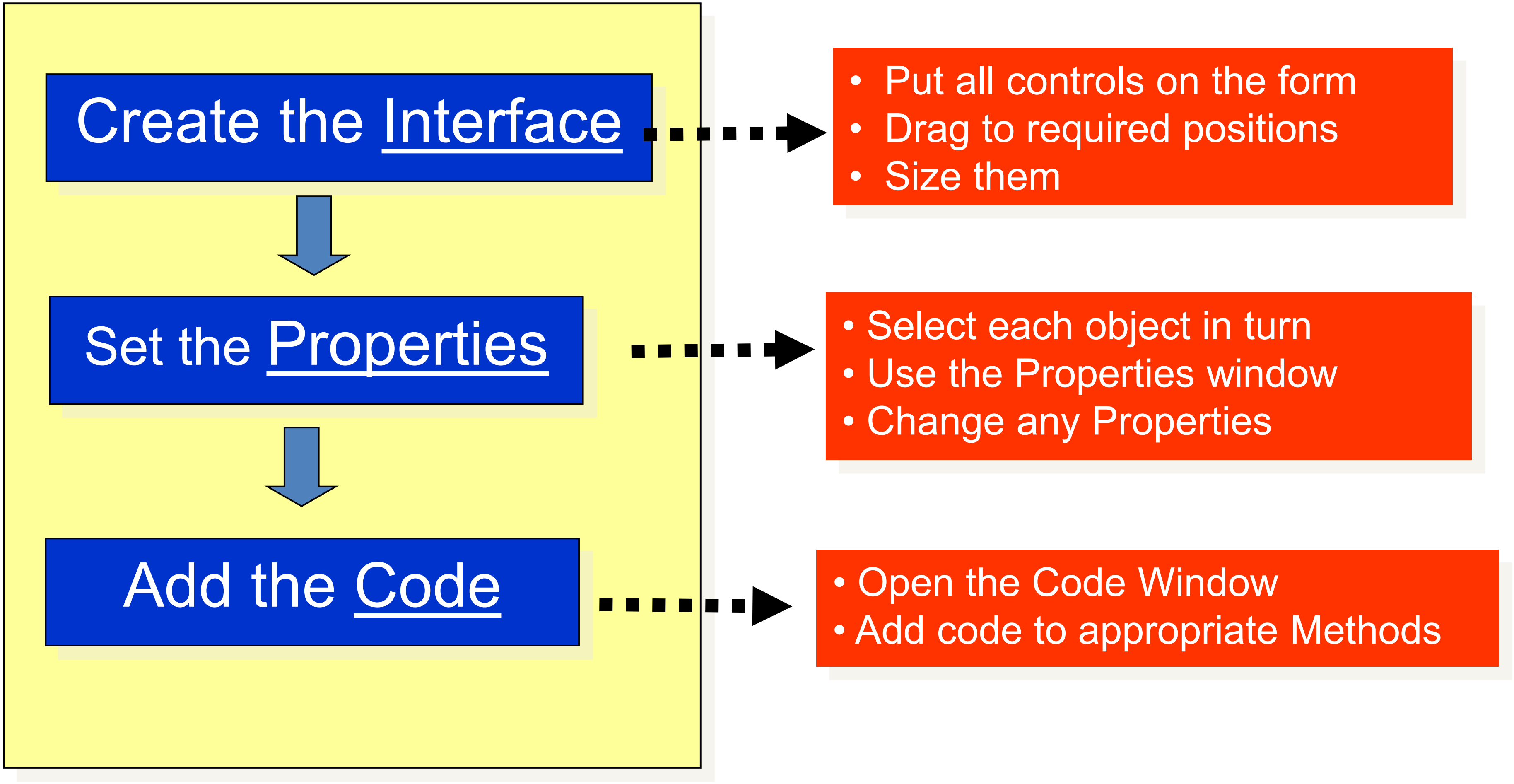
click



The Last Slide

Extra Reading

Writing a C# .NET Windows Program



Create the Interface

- Put all controls on the form
- Drag to required positions
- Size them

Set the Properties

- Select each object in turn
- Use the Properties window
- Change any Properties

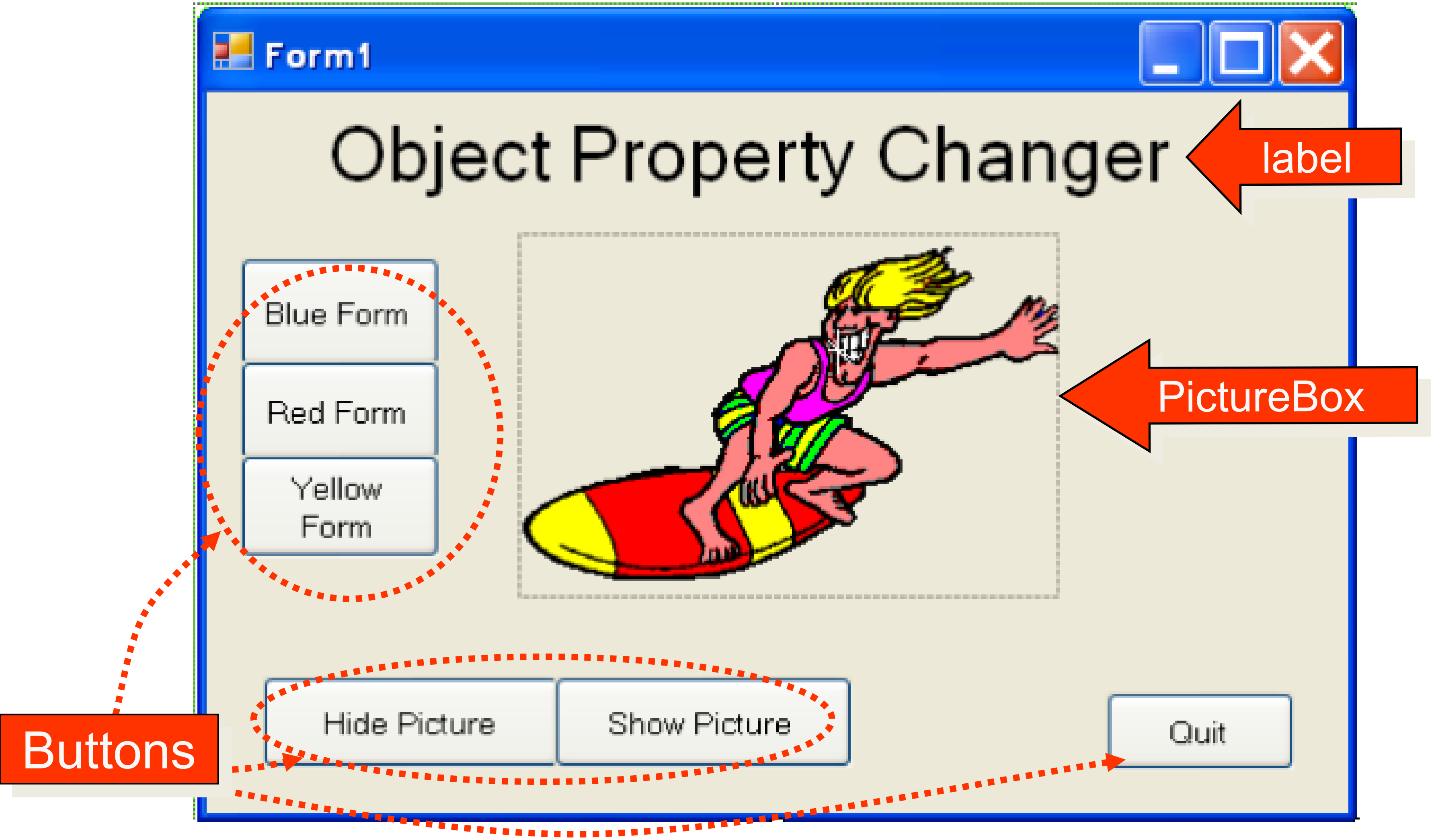
Add the Code

- Open the Code Window
- Add code to appropriate Methods

Object Properties

How to change them as the program runs (run-time)

Create the Interface and set the properties

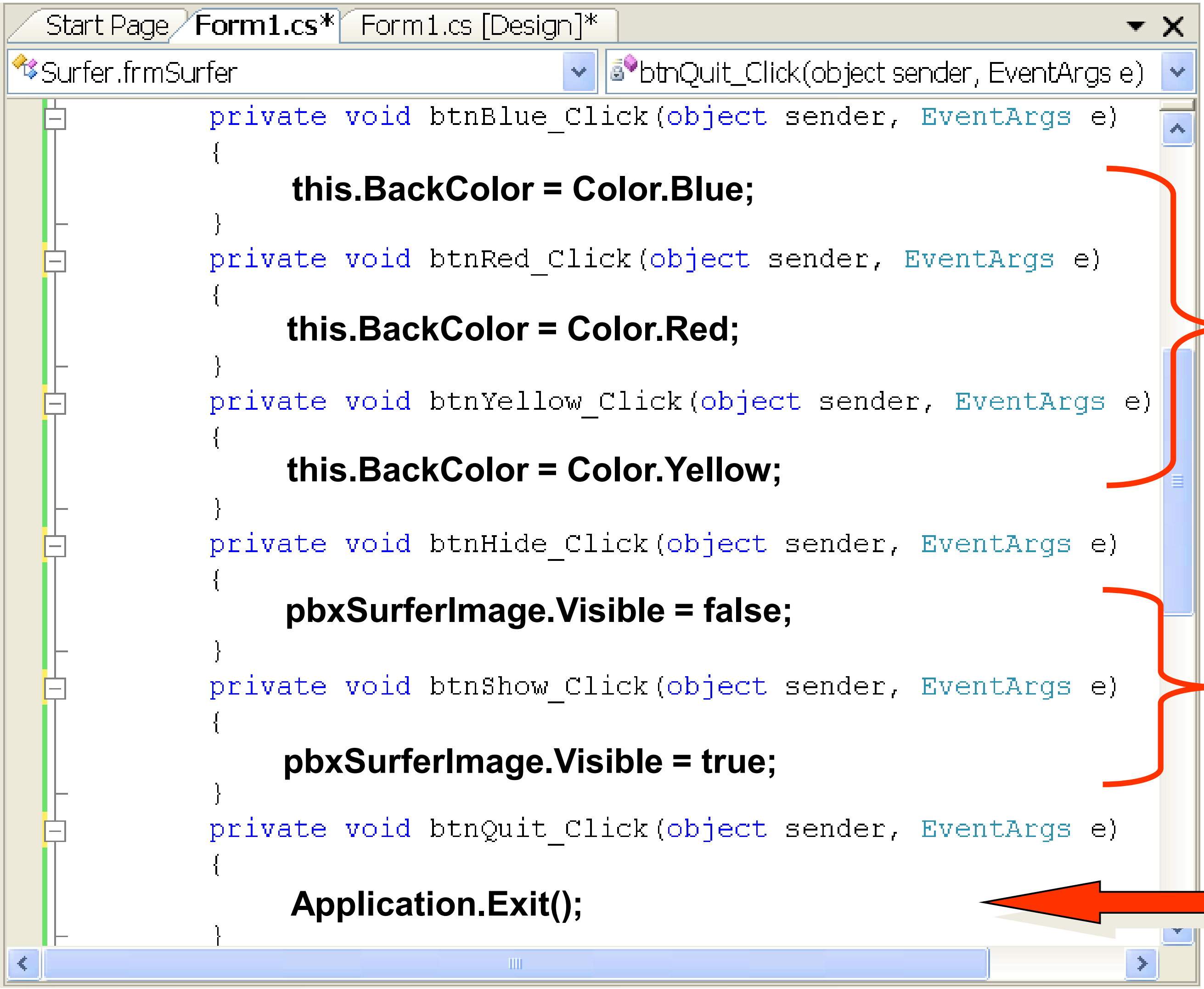


label

PictureBox

Buttons

Add Code to the Buttons



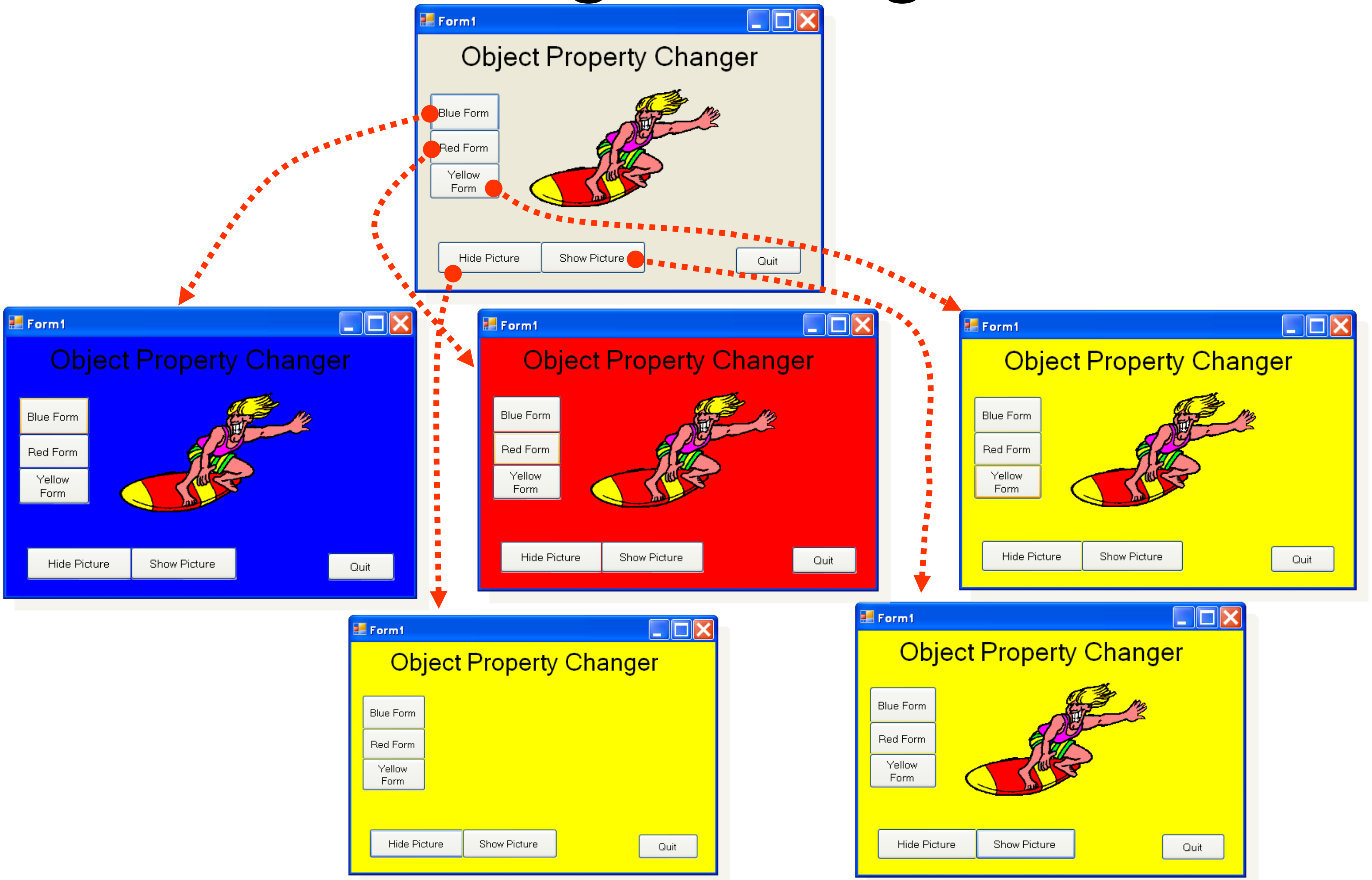
```
        private void btnBlue_Click(object sender, EventArgs e)
        {
            this.BackColor = Color.Blue;
        }
        private void btnRed_Click(object sender, EventArgs e)
        {
            this.BackColor = Color.Red;
        }
        private void btnYellow_Click(object sender, EventArgs e)
        {
            this.BackColor = Color.Yellow;
        }
        private void btnHide_Click(object sender, EventArgs e)
        {
            pbxSurferImage.Visible = false;
        }
        private void btnShow_Click(object sender, EventArgs e)
        {
            pbxSurferImage.Visible = true;
        }
        private void btnQuit_Click(object sender, EventArgs e)
        {
            Application.Exit();
        }
```

These change the BackColor property of this form

These change the Visible property of the PictureBox

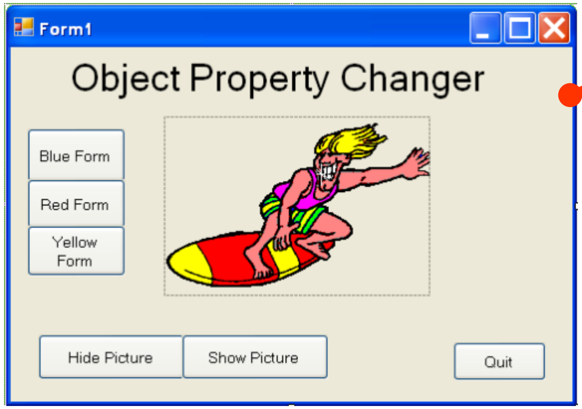
Exit the Application

Running the Program

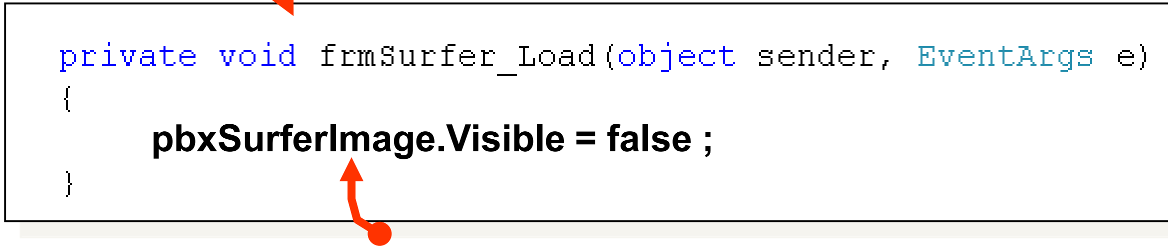


The Form’s Load Event

We often need to make something happen when a form first loads. Suppose we want the surfer to be invisible at the start



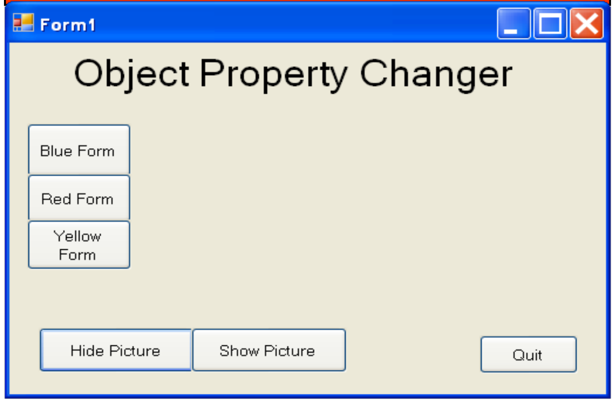
Double-click the form to bring up its code window



```
private void frmSurfer_Load(object sender, EventArgs e)
{
        pbxSurferImage.Visible = false ;
}
```

Then add code to the form’s Load event method

Now it starts like this



Extra Reading

Formatting Numeric Output

A Reminder

If you want to format a double type of number to 2 decimal places:

```
lblResult.Text = answer.ToString("0.00");
```