

# Software Design & Build Fundamentals

LEARN TO DESIGN & BUILD A WINDOWS C#  
APPLICATION

STEPHEN O'CONNOR

## Exercise files github

### Task A

#### Question 1

What is SDLC?

What role does design have?

What is level 2?

#### Question 2

Using Appendix A.

Draw diagram to describe program features using paper or P.C

#### Question 3

a. How to use Visual Studio 2013?

b. What are the C# basics?

Developing Software is Fun!

Don't be intimidated ... take a little at a time

#### Lessons

1 – 3: Workflow, Visual Studio Interface

4 – 7: C# Programming Language

8 – 11: Working with Data

12 – 16: Concert Booking Application

#### Setting-up

Download and install Visual Studio Community 2013.

Create GitHub account

Create a new repository called Csharp-Basics

Download and install the latest version of GitHub for windows version 7, 8, 8.1

On your PC in users\username\documents add new vsprojects folder

Open PowerShell if not in directory C:\> type cd c:\

```
C:\Windows\system32> cd c:\  
C:\>
```

Navigate to vsprojects folder

```
C:\> cd c:\users\username\documents\vsprojects  
C:\users\username\documents\vsprojects> git clone "HTTPS clone URL"
```

Check vsprojects folder for git clone type dir and press enter

```
C:\users\username\documents\vsprojects> dir
```

Open csharp-fundamentals, check git version

```
C:\users\username\documents\vsprojects> cd csharp-fundamentals  
C:\users\stephen\documents\vsprojects\csharp-basics [master]> git version  
git version 1.9.5.msysgit.0
```

Let's create a HelloWorld windows application.

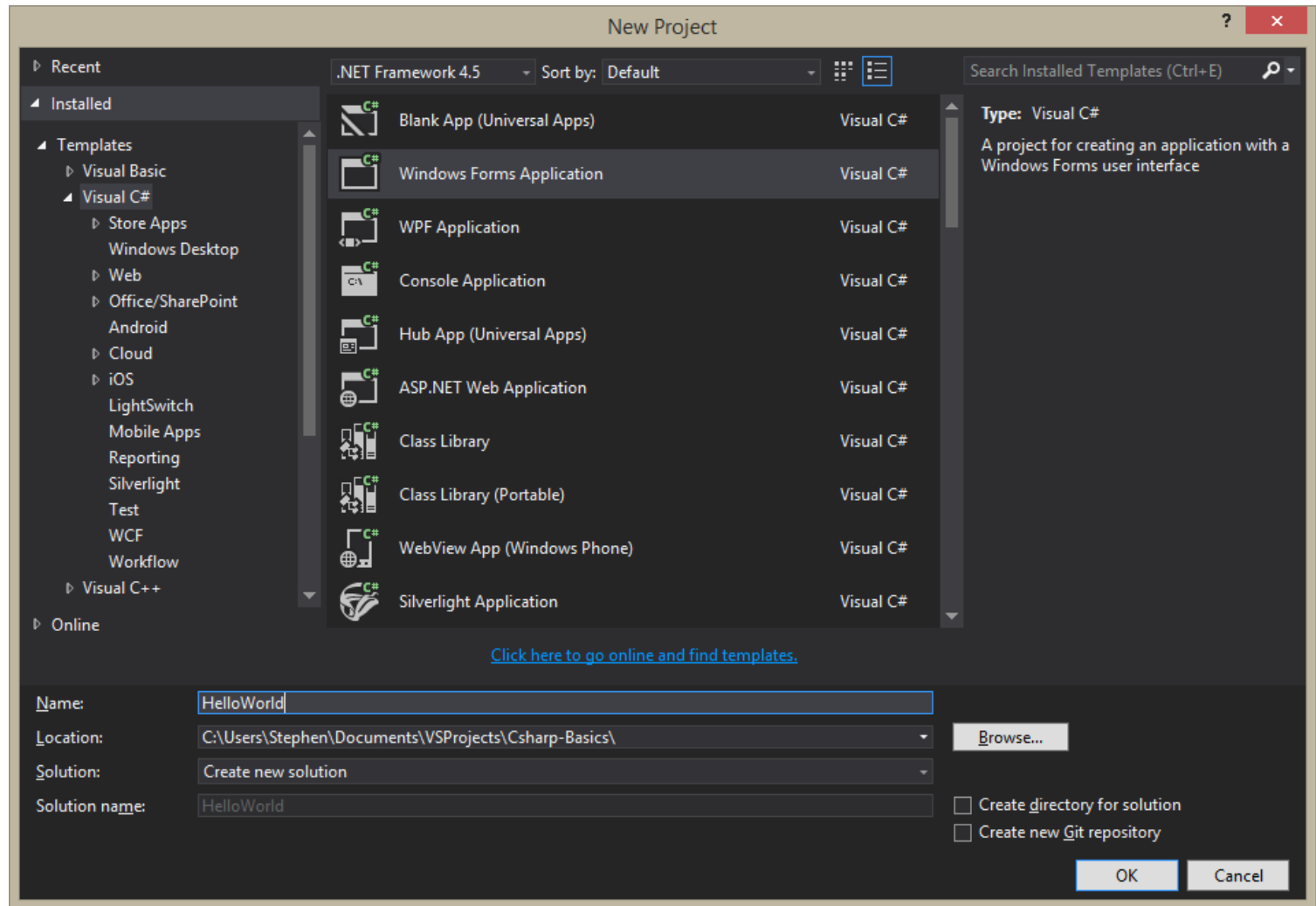
Open Visual Studio select new project

Select Visual C# and Windows form application.

In the Windows dialog change name and location.

Name: HelloWorld

Location: click Browse button navigate to vsprojects\Csharp-Basics folder – GitHub clone



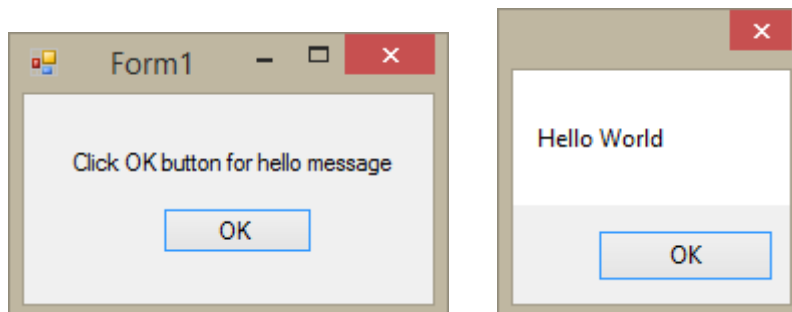
Run debug click start button, this is a basic windows form click x and close. This is a complete windows app, however it does not do anything.

Create a button that once clicked displays a message box that displays Hello World

Go to toolbox select button

Double click button and into Form1.cs type `MessageBox.Show("Hello World");`

Two tabs code design view and debug app using the start button.



Go through basic of visual studio

Go to location of app in documents

Solution explorer file description

Shut down reopen visual studio

Debugging

Set break points to step through each line of code

What are Events?

How to handle Events?

“Event Driven Programs”

Response to events like open file, exit, new file, print file.

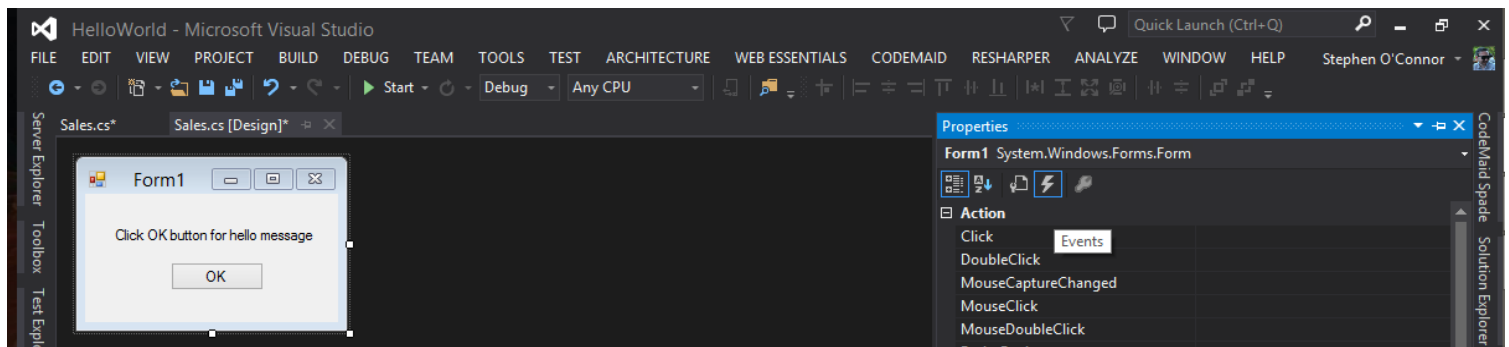
Hundreds of events that an app can react to.

Double click on the OK button an event handler is created and given a default name. When the event is triggered the end user clicks the button.

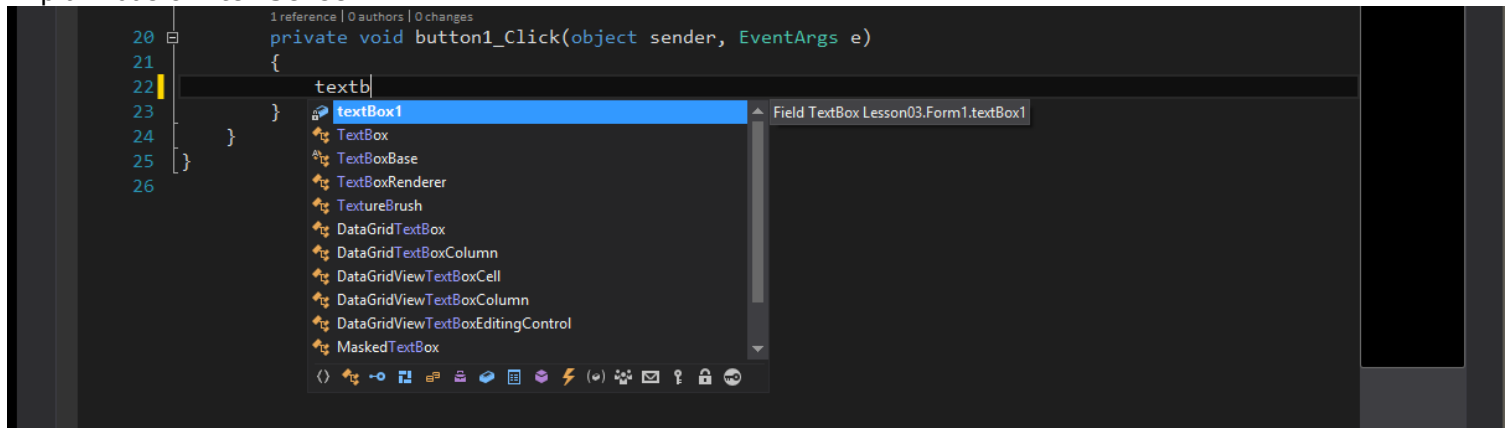
Event handler event default name button1\_Click Curly braces ... { } ... define a block of code.

Events are triggered in an app. App can respond or ignore those events. Write code in Event Handlers to handle events. Code must be written inside of code block defined by curly braces. Methods are the basic building blocks of writing code. An event Handler is a more specific type of a method.

```
private void button1_Click(object sender, EventArgs e) // event handler
{
    MessageBox.Show("Hello World"); // event
}
```



Explain basic IntelliSense



Comments and regions

```

19
20 1 reference | 0 authors | 0 changes
21 private void button1_Click(object sender, EventArgs e)
22 {
23     // this is a comment
24     textBox1.Text = "Hello World";
25
26     /*
27      * this is a multi-line
28      * comment
29      */
30
31     #region This is a region
32     // this is a region and its useful for
33     // organising your code
34     // https://msdn.microsoft.com/en-us/library/9a1ybwek.aspx
35     #endregion
36 }
37 }

```

## Design

### Design Best Practices

Arrange controls in columns and rows

Labels left input boxes, data entered on the right

Ok Cancel buttons on the right

Use standard fonts and colors

Keep it simple

Use standard, succinct descriptions (names) for controls. Don't make the user "think"

Make it easy for the user to use

### Buttons

Allow user to communicate a decision or to trigger some action

### Labels

Non-interactive descriptions or text usually displayed on the left-hand side of other controls

### Text Boxes

Allow for unstructured user input.

### Check Boxes

Allow for yes /no or on /off type user input

Used together to allow off "Check all that apply"

### Combo box

Text and list box combination

User can select item in the list, or type in a selection that is not in the list

### Menu strip

Add menu to app

Select Insert Standard Items

### Toolbar strip

Add toolbar to app.

Includes progress bars, textboxes, combo boxes & more

### Status strip

Add status bar to app to provide feedback to the user

### Tool Strip Container

Hosts other controls like the menu strip, Toll Strip and the Status Strip to provide user app customization.

Arrange toolbars top left bottom right

### Tips

Make sure the right control is selected before making any changes in the Properties window.

Set tab order. Click VIEW and Tab Order click item to make first.

Tab Order starts with the control from the left-hand corner logically to the lower right-hand corner.

## Understanding Variables and datatypes

$x = 4$

$y = x + 6$

What does y equal?

Variables represents a space in a computer's memory that is assigned to store a value.

The name of the variable is then used in code to reference the value that is stored in that memory space.

Declaring a variable is the act of allocationg space in he computer's memory for valus of a specific data type and giving the variable name.

There are many available data types in C#, including ones that can store strings, dates, numbers, and more..

### Three Basic Numeric data Types ..

Interger (int): -2,147,483,648 to +2,147,483,647

Double (double):  $\pm 5.0 \times 10^{-324}$  to  $\pm 1.7 \times 10^{308}$  up to 15 decimal places.

Boolean (bool): true or false.

C# is case sensitive.

Consider a variable a bucket and the string literal can go into the bucket.

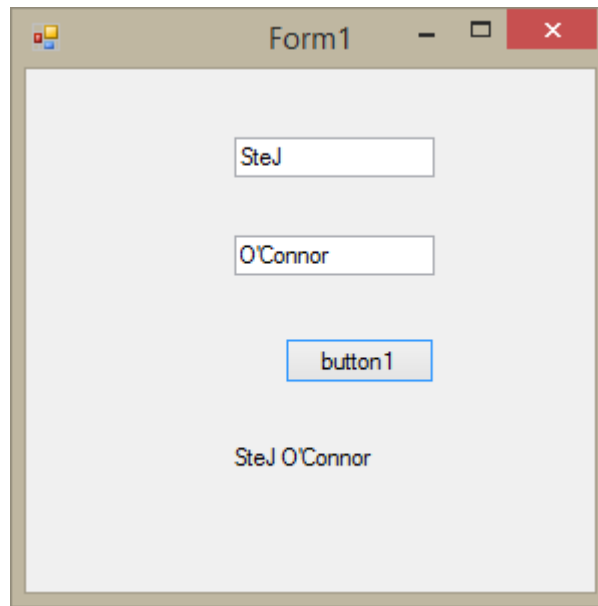
```

string hello; // hello is the bucket
hello = "hello world"; // string literal goes into the hello bucket
MessageBox.Show(hello); // hello var
MessageBox.Show("hello"); // string literal

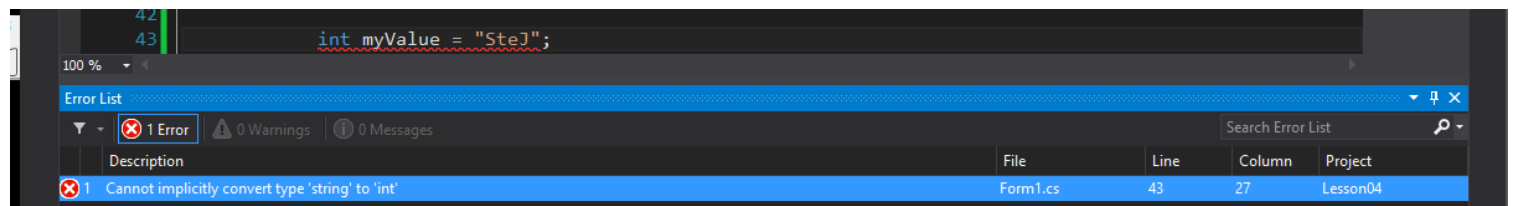
// declare two vars
string firstTextBox = textBox1.Text;
string secondTextBox = textBox2.Text;

```

```
label1.Text = firstTextBox + " " + secondTextBox;
```



Declare a Variable with the wrong a data type error with discription.

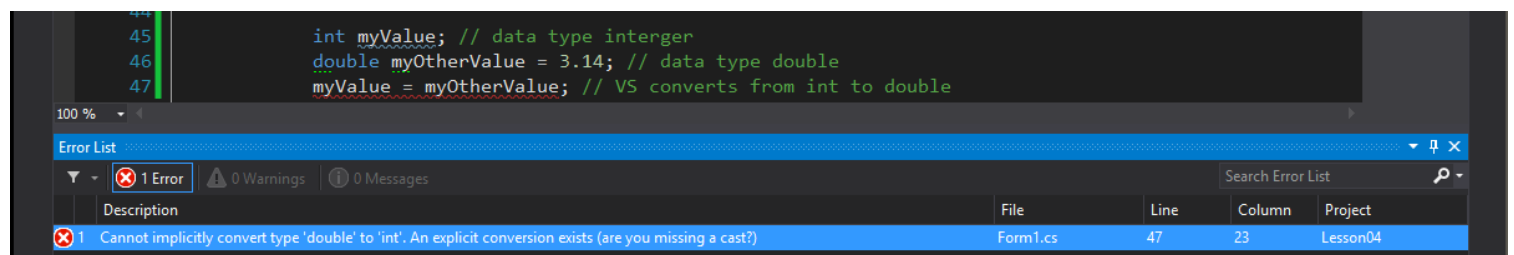


VS implicitly converts interger to double no loss of data

```
int myValue = 3; // data type integer
double myOtherValue; // data type double
myOtherValue = myValue; // VS converts from int to double
```

VS cannot convert double to interger loss of data

```
int myValue; // data type integer
double myOtherValue = 3.14; // data type double
myValue = myOtherValue; // VS cannot convert
```



### Create a basic calculator

To explicitly convert data types, let's look at the following example. With an explicit cast, either you are telling the compiler that **you know more than it does** - "please believe me, but check anyway": `textBox.Text` cannot add two numbers and display the result in `label1.Text`

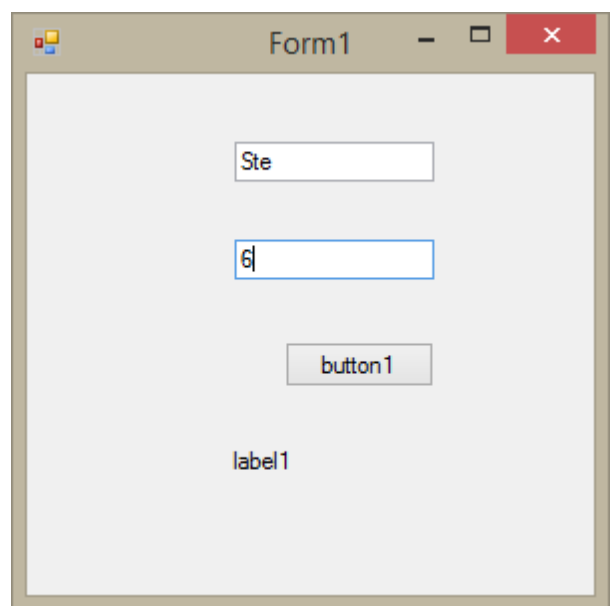
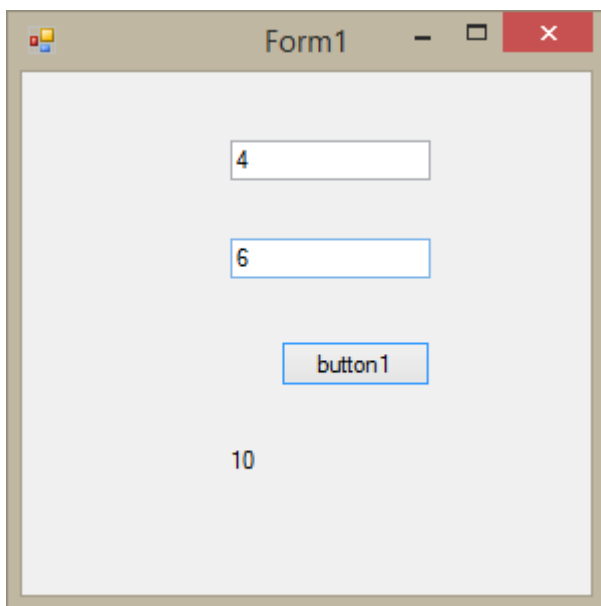
Error: Cannot implicitly convert type 'int' to 'string'

```
int firstTextBox = 0;
int secondTextBox = 0;
int result = 0;

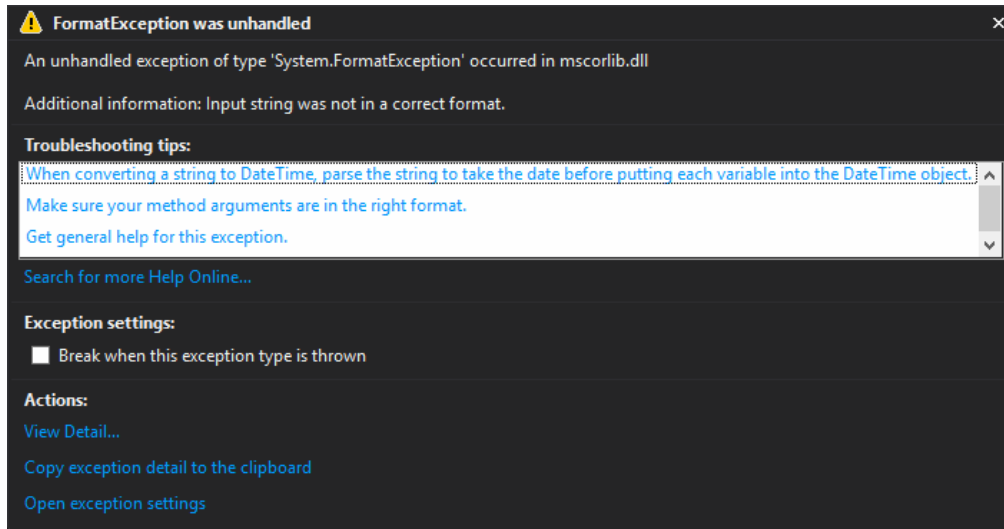
firstTextBox = int.Parse(textBox1.Text);
secondTextBox = int.Parse(textBox2.Text);

result = firstTextBox + secondTextBox;
label1.Text = result.ToString();
```

### Testing the calculator







For int → `int.Parse(myString);`  
 For double → `double.Parse(myString);`  
 For bool → `bool.Parse(myString);`  
 Expressions Versus Statements

#### Expression

This is a very basic expression `int x;`  
 Can be evaluated.

`x + 3;` // this is not a statement, it's an expression

#### Statement

This is a statement `x = x + 3;` // this is an assignment

Valid Statements Consist of

Assignment → `myInteger = 3;`  
 Call → `MessageBox.Show("Hello World");`  
 Increment → `x++;`  
 Decrement → `--x;`

**"Expressions can be evaluated.. Statements can be executed."**

```
label1.Text = firstTextBox + " " + secondTextBox; // statement
firstTextBox + " " + secondTextBox; // expression not a statement
```

Evaluating Expressions

`(3 < 2) .. true or false?`  
`(3 < 2) .. true or false?`

Given: `int x = 3;`

`(x == 3) .. true or false`  
`(x != 3) .. true or false`

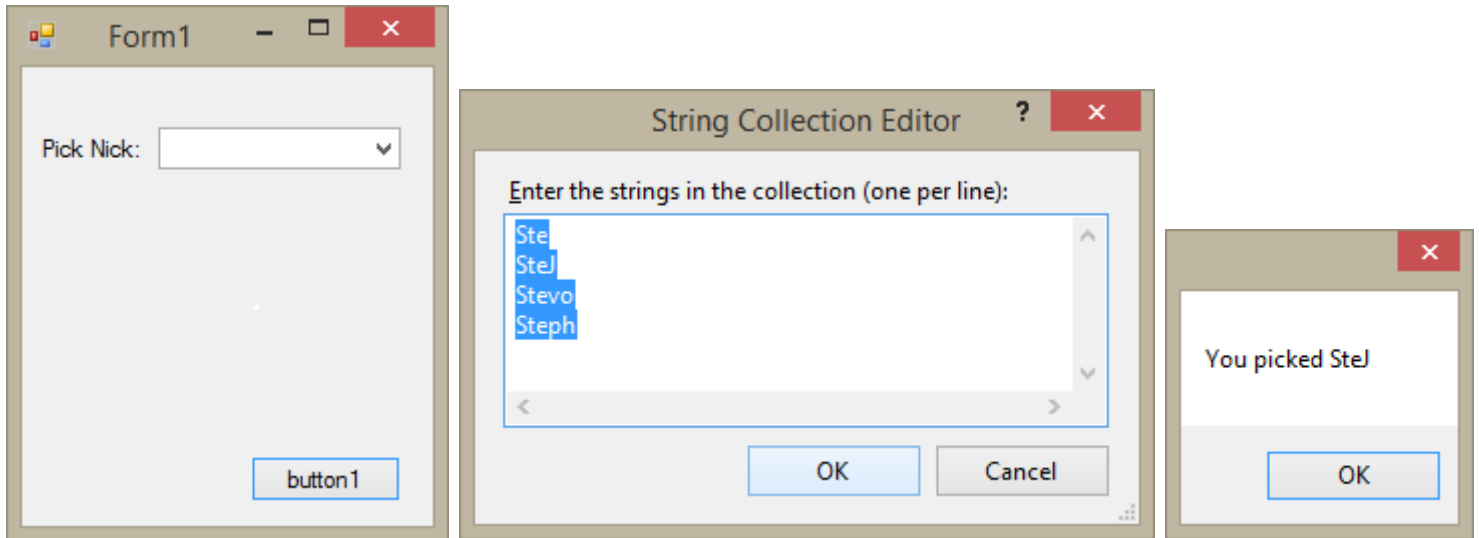
`=` Assignment telling: `x` is 4

`==` Evaluation asking: is this equal?

`!=` evaluates "not equal" 3 is not equal to 4: true

#### Using Iteration and Selection Statements in C#

Selection Statements decide whether or not to execute a block of code based on the evaluation of an expression. If condition else do this. Toolbox select and drag `comboBox` onto `form1`, select `comboBox` in design view click arrow select Edit Items. Type nicknames into dialog box, click OK



First and second 'if' statements code for comboBox

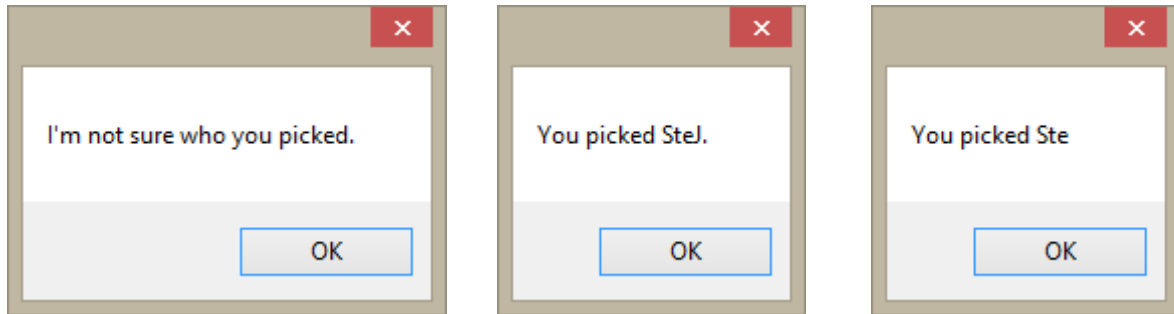
```
// 1. basic 'if' statement
if (comboBox1.Text == "SteJ")
{
    MessageBox.Show("You picked SteJ");
    comboBox1.Text = ""; // clears comboBox if selected
}

// 2. 'if' statement curly braces removed, one line of code to be executed
if (comboBox1.Text == "Steph")
    MessageBox.Show("You picked Steph"); // one line of code after 'if'
```

Third 'if' statement code for comboBox

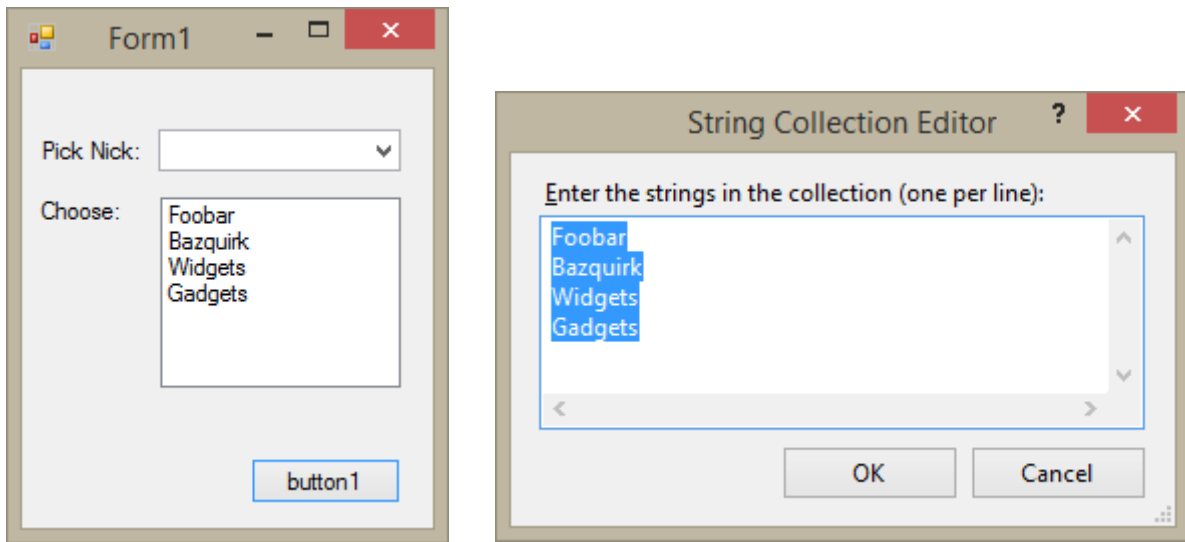
```
// 3. nested 'if' statement
if (comboBox1.Text != "Ste") // if not equal to Ste false go to else
{
    if (comboBox1.Text == "SteJ")
    {
        MessageBox.Show("You picked SteJ."); // SteJ is selected
    }
    else
    {
        MessageBox.Show("I'm not sure who you picked."); // Stevo or Steph selected
    }
}
else
{
    MessageBox.Show("You picked Ste");
}
```

In combox Select Stevo, SteJ and then Ste.



### Switch

Drag and drop listBox from toolbox onto form1. Select listBox and click arrow. Type Foobar Bazquirk, Widegets & Gadgets into dialog box.

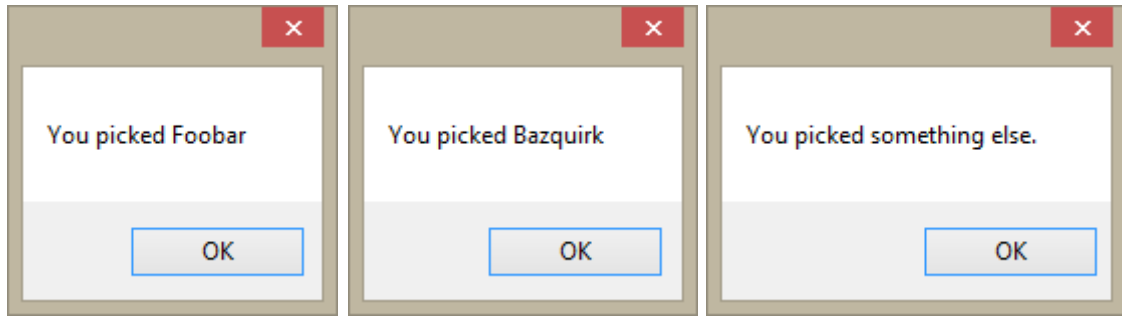


Double click listBox1 and write code for switch

```
// 'switch' statement
switch (listBox1.SelectedItem.ToString())
{
    case "Foobar":
        MessageBox.Show("You picked Foobar");
        break;

    case "Bazquirk":
        MessageBox.Show("You picked Bazquirk");
        break;

    default:
        MessageBox.Show("You picked something else.");
        break;
}
```



## Arrays

Type of collection that allows you to group together a bunch of values that are related in some way

```
// 1. Sized array set the size
string[] myArraySize = new string[2];
myArraySize[0] = "SteJ";
myArraySize[1] = "Steph";
//myArraySize[2] = "Stevo"; // causes an out of bounds exception
MessageBox.Show(myArraySize[1]);
```

All items in the array must be the same data type

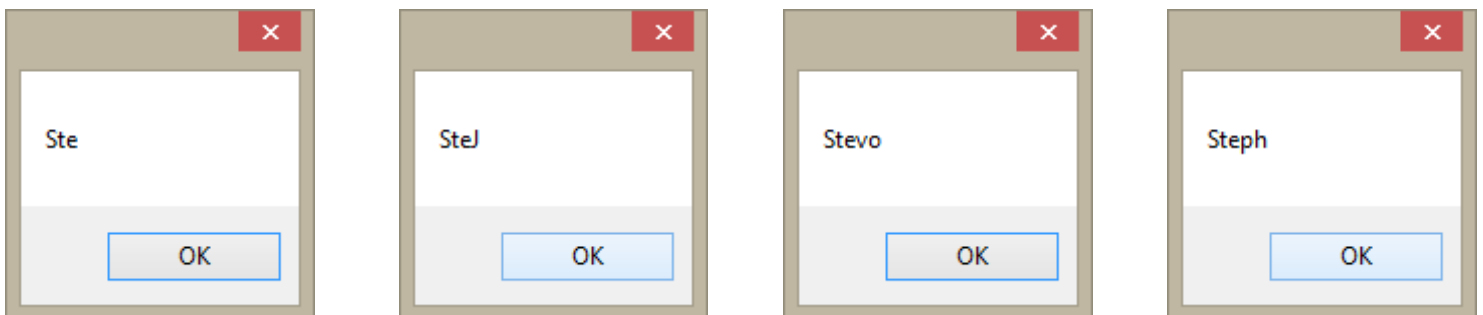
Iteration statements

Loop through, or navigate through each item in an array one at a time.

- Foreach
- For
- While

```
// 2. Initialized array begin at [0] = Ste | [1] = SteJ | [2] = Stevo | [3] = Steph
string[] myArray = {"Ste", "SteJ", "Stevo", "Steph"};
// MessageBox.Show(myArray[1]);

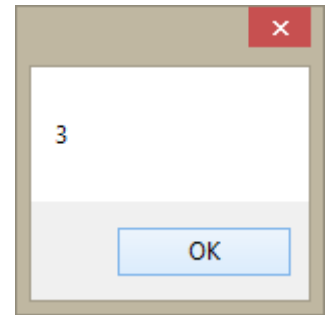
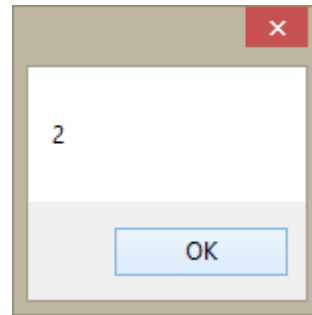
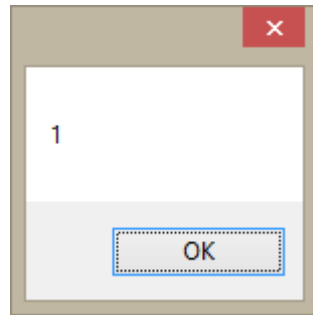
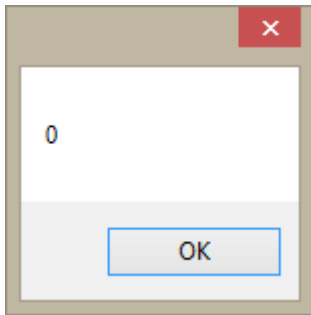
// temp value nickname
foreach (var nickname in myArray)
{
    MessageBox.Show(nickname);
}
```



For loop

```
string[] myArray = { "Ste", "SteJ", "Stevo", "Steph" };
for (int i = 0; i < myArray.Length; i++)
{
    MessageBox.Show(i.ToString());
}
```

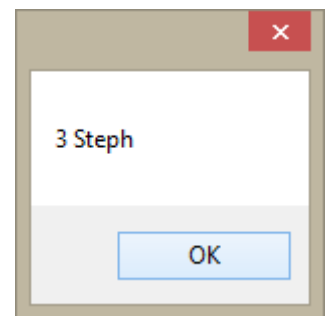
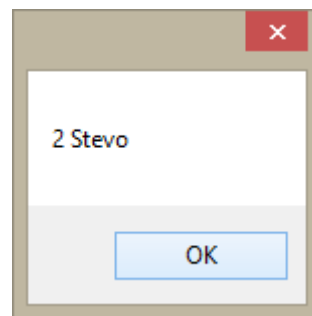
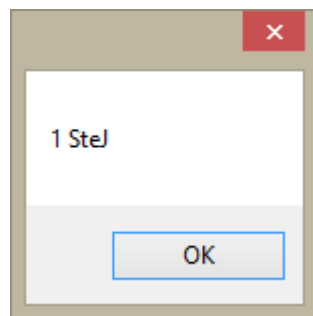
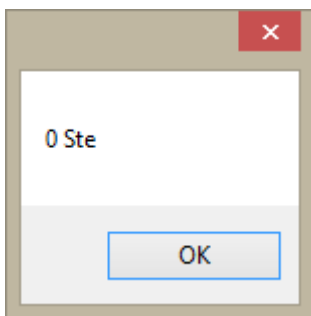
}



To get both index and name a two dimensional array is created. This a nested for loop, the second loop is a foreach.

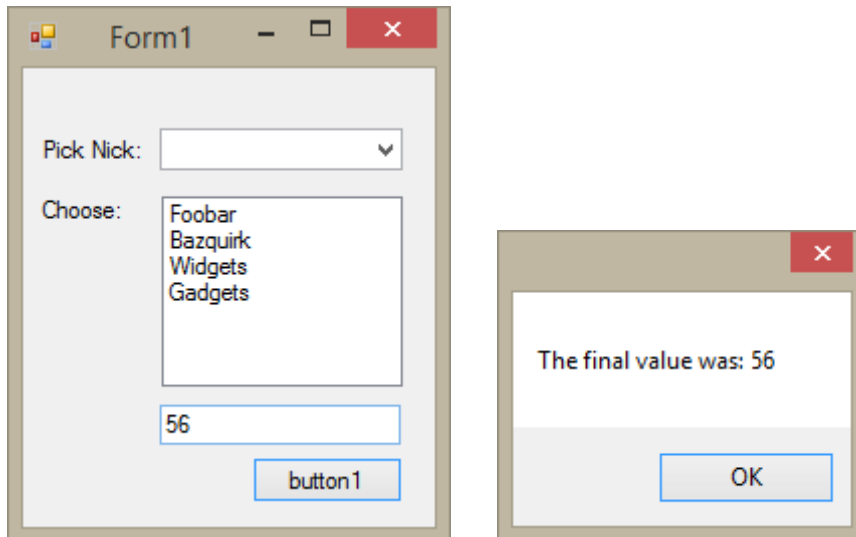
```
// two dimensional array [0] = Ste | [1] = SteJ | [2] = Stevo | [3] = Steph
string[] myArray = { "Ste", "SteJ", "Stevo", "Steph" };

for (int i = 0; i < myArray.Length;)
{
    // temp value nickname
    foreach (var nickname in myArray)
    {
        MessageBox.Show(i++ + " " + nickname);
    }
}
```



```
int i = 0;
while (i < int.Parse(textBox1.Text))
{
    i++;
}
MessageBox.Show("The final value was: " + i.ToString());
```

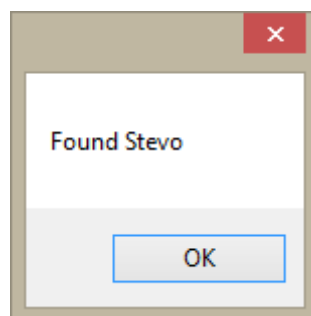
Drag and drop textBox onto Form1. Debug and enter number into textBox



Go through the array if index = Stevo Message Found Stevo

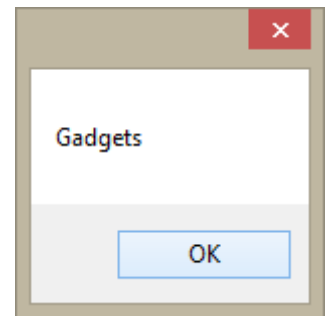
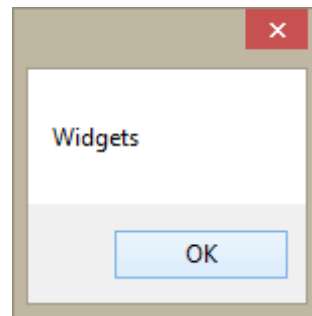
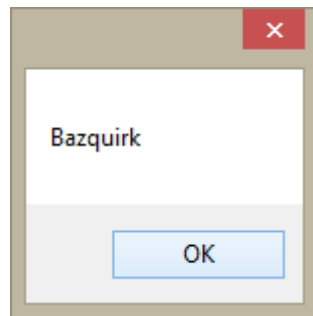
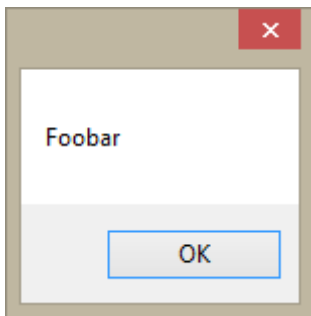
```
// Combining array, for and if
string[] myArray = {"Ste", "SteJ", "Stevo", "Steph"};

for (int i = 0; i < myArray.Length; i++)
{
    if (myArray[i] == "Stevo")
    {
        MessageBox.Show("Found Stevo");
    }
}
```



```
for (int i = 0; i < listBox1.SelectedItems.Count; i++)
{
    switch (listBox1.SelectedItems[i].ToString())
    {
        case "Foobar":
            MessageBox.Show("Foobar");
            break;
    }
}
```

```
    case "Bazquirk":  
        MessageBox.Show("Bazquirk");  
        break;  
  
    case "Widgets":  
        MessageBox.Show("Widgets");  
        break;  
  
    case "Gadgets":  
        MessageBox.Show("Gadgets");  
        break;  
    }  
}
```



Write code for "Concert Booking" program  
Operation of the booking seat plan  
Create new file  
Open existing file

Question 4  
Error handling

## Task B

### Links

GitHub  
<http://git-scm.com/book/en/v2/Getting-Started-Git-Basics>  
<https://windows.github.com/>  
<https://help.github.com/articles/set-up-git/>  
<https://github.com/blog/674-introducing-organizations>

Microsoft  
<https://www.visualstudio.com/en-us/products/visual-studio-express-vs.aspx>  
[MSDN microsoft.com](https://msdn.microsoft.com/)  
<http://www.learnvisualstudio.net/>  
<https://msdn.microsoft.com/en-us/library/hcw1s69b.aspx>

Design  
[SDLC Overview](#)