



Visual C# Introduction

Sample manual - first two chapters



TABLE OF CONTENTS (1 of 3)

1	VISUAL STUDIO PRIMER	Page
1.1	Windows Forms	5
1.2	Customising Visual Studio	5
	<i>Setting the Default Start-up Page</i>	6
	<i>Creating Appropriate Settings</i>	6
1.3	Creating Projects	6
1.4	Saving and Closing Files	7
	<i>Closing One Window</i>	8
	<i>Closing All or Nearly All Windows</i>	8
1.5	Auto-hiding windows	8
1.6	The Three Most Useful Windows	9
	<i>Displaying Windows</i>	10
	<i>Properties Window</i>	10
	<i>The Toolbox</i>	10

4	FORM EVENTS	Page
4.1	Events	18
	<i>Attaching Code to a Control's Default Event</i>	18
	<i>Creating a New Event-Handler for an Event</i>	18
	<i>Handling an Event with an Existing Routine</i>	19
4.2	Switching Between Form Design and Code View	19
	<i>Using the Keyboard</i>	20
	<i>Using Solution Explorer</i>	20
4.3	Those Strange Event Arguments	20
	<i>Argument 1 – The Object that Called the Event</i>	21
	<i>Argument 2 – The Event Arguments</i>	22

2	DRAWING FORMS	Page
2.1	Creating a New Form	11
2.2	Changing form properties	12
2.3	Form Controls	12
	<i>Definition of Controls</i>	13
	<i>Adding controls</i>	13
2.4	Selecting Controls	13
	<i>Selecting a Single Control</i>	14
	<i>Selecting Several Controls</i>	14
	<i>Selecting All Controls</i>	14
2.5	Basic Formatting	14
	<i>Resizing Controls</i>	15
	<i>Changing how Controls Look</i>	15
	<i>Moving Controls</i>	15

3	RUNNING APPLICATIONS	Page
3.1	Running a Program	16
	<i>Setting the Default Form in Program.cs</i>	16
	<i>Running and Stopping Programs</i>	16
3.2	Dealing with Errors	16
	<i>Building a Project</i>	17
	<i>Dealing with Build Errors</i>	17

TABLE OF CONTENTS (2 of 3)

5	VARIABLES AND DATA TYPES	Page
5.1	Why Use Variables?	23
5.2	Declaring Variables	23
	<i>Declaring Variables</i>	24
	<i>Creating Nullable Variables</i>	24
	<i>Using Modified Hungarian Notation</i>	25
	<i>Default Values for Variables</i>	25
	<i>Problems with Declaring Variables within Clauses</i>	25
5.3	Setting Values in Variables	26
	<i>Declaring Integer Variables and Adding/Subtracting</i>	27
	<i>Accumulating Text in String Variables</i>	27
5.4	Variable Data Types	27
	<i>Mapping C# Data Types to the CLR Runtime</i>	28
	<i>A Lazy Person's Data Types</i>	28
	<i>Logical Values</i>	29
	<i>Integers</i>	29
	<i>Decimal (Floating Point) Numbers</i>	29
	<i>Strings and Text</i>	29
	<i>Dates and Times</i>	30
	<i>Objects</i>	30
5.5	Converting Variables	30
	<i>Conversion Using Convert.To</i>	31
	<i>ToString() – Special Case for String Conversions</i>	31
	<i>Casting Data Types</i>	31
5.6	Variable Scope	31
5.7	Notes on Working with Specific Data Types	32
	<i>Working with Characters</i>	33
	<i>Working with Strings</i>	33
	<i>Escape Characters</i>	34
	<i>Verbatim Strings</i>	34
	<i>Splitting Strings</i>	34
	<i>Formatting Dates</i>	35
	<i>Working with Numbers – Possible Operations</i>	36
5.8	Constants	36
5.9	Testing Data Types	37

6	CONDITIONS	Page
6.1	Using IF for Conditions	39
6.2	Operators	39
6.3	The SWITCH statement	40
	<i>Limitations of SWITCH</i>	41
6.4	Ternary and Coalesce Operators	41
	<i>The Ternary Operator</i>	42
	<i>The Null Coalesce Operator</i>	42

7	LOOPS	Page
7.1	Looping in C#	43
	<i>Looping Over a Collection/Array</i>	43
	<i>Looping a Given Number of Times</i>	43
	<i>Looping While a Condition is True (While/Do)</i>	44
7.2	Breaking Out of Loops	45

8	ARRAYS	Page
8.1	Arrays	47
	<i>Creating single-dimensional arrays</i>	47
	<i>Populating arrays and retrieving items</i>	47
	<i>Looping over arrays</i>	47
	<i>Multi-dimensional arrays</i>	48

9	ERROR TRAPPING	Page
9.1	Try / Catch / Finally	49
	<i>Syntax of the Try / Catch Statement</i>	49
	<i>General error trapping example – validating an integer</i>	49
	<i>Catching Specific Errors</i>	50
	<i>The Finally clause</i>	51
	<i>A better alternative to finally – Using</i>	52
	<i>Throwing Exceptions</i>	53
	<i>Exceptions bubble up the call stack</i>	54

10	FILES AND FOLDERS	Page
10.1	StreamReaders and StreamWriters	55
	<i>Our Example</i>	55
	<i>Referencing the System.IO Namespace</i>	55
	<i>Reading in the Customers</i>	55
	<i>Writing out the customers' details to file</i>	56
10.2	Using FILE	57
10.3	FileInfo and DirectoryInfo	58
	<i>Useful File Properties</i>	59
	<i>Getting at folders</i>	59
	<i>Looping over files in folders</i>	60
	<i>Recursively looping over all folders and</i>	60

TABLE OF CONTENTS (3 of 3)

files

11	LISTS	Page
11.1	Overview of Lists	61
	<i>An Example of a List</i>	61
11.2	Working with Lists	61
	<i>Creating a List</i>	62
	<i>Adding Items to a List</i>	62
	<i>Counting the Items in a List</i>	62
	<i>Displaying All of the Items in a List (FOR EACH)</i>	63
	<i>Removing Items from a List</i>	63
	<i>Finding items in a list</i>	63
	<i>Lambda Expression Syntax for Find Methods</i>	64
11.3	Getting a Subset of a List	65
	<i>Method 1: Using FindAll</i>	65
	<i>Getting a Subset of a List – Method 2: Using GetRange</i>	65
11.4	Joining and Splitting String Lists	65

12	DESIGNING CLASSES	Page
12.1	Cats as Objects	67
	<i>Types, Classes and Objects</i>	67
	<i>Instantiation and Termination</i>	67
	<i>Properties</i>	68
	<i>Methods</i>	68
	<i>Encapsulation and Exposure</i>	69
	<i>Inheritance</i>	69
12.2	Our Example – Dating Agency Customers	69
	<i>Our Customer Class</i>	70
	<i>Envisaging how you will Consume a Class</i>	70

13	CREATING CLASSES	Page
13.1	Creating a Class	72
13.2	Namespaces	72
	<i>Example of a Namespace</i>	73
	<i>The Using Statement</i>	73
	<i>Removing Unused Using Statements</i>	73
	<i>Giving Aliases to Namespaces</i>	74
	<i>Our DatingAgency Namespace</i>	74
13.3	Creating a Constructor	74
	<i>Syntax of a Constructor</i>	75
	<i>Example of a Constructor</i>	75
13.4	Fields and Properties	76
	<i>Creating Fields</i>	77
	<i>Properties</i>	77
	<i>Refactoring (encapsulating) fields</i>	77
	<i>The Quickest and Best Way to Create Properties</i>	79
	<i>Properties which Perform Other Logic</i>	80
13.5	Methods	80
	<i>Void Methods</i>	81
	<i>Methods which Return Values</i>	81
	<i>Choosing between a Property and a Method</i>	82
13.6	Static Properties and Methods	82
	<i>Example of a Static Property</i>	83
	<i>Example of a Static Method</i>	83

14	USEFUL SHORT-CUT KEYS	Page
14.1	The Best Short-Cut Keys in Visual Studio	85
	<i>Going to the definition of a variable or member</i>	85
	<i>Going forward and backward using the keyboard</i>	85
	<i>Auto-formatting text</i>	86
	<i>Adding a Using statement</i>	87

CHAPTER 1 - VISUAL STUDIO PRIMER

1.1 Windows Forms

There are three main types of application you can develop using *Visual Studio* (Microsoft's development tool for .NET programmers):

Type of application	Use for
<i>WinForms (Windows Forms)</i>	Creating basic business standalone applications to run within Windows.
<i>WPF (Windows Presentation Foundation)</i>	Creating Windows applications with fancy graphics (WPF takes longer to learn but is more powerful than Windows forms).
<i>ASP.NET webforms</i>	Creating websites using forms-based ASP.NET.
<i>ASP.NET MVC / MVC Core</i>	Creating websites using the model-view-controller method.

This courseware uses WinForms exclusively, because it's the simplest of the 3 types of application (the aim of the course is to teach C#, not drawing!).

This is an application form for the *Wise Owl Dating Agency (WODA, not to be confused with YODA)*. Clicking on the button could show a message like this:

To create a Windows Forms application like this, you'll first need to learn to use Visual Studio, as shown in the rest of this chapter.



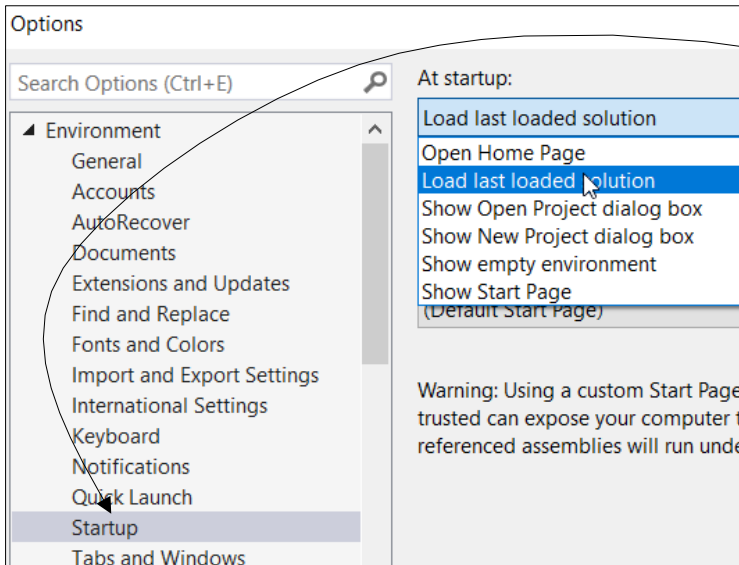
This courseware uses Visual Studio 2017, but you should find that all functionality is pretty much identical, regardless of which version of Visual Studio you use.

1.2 Customising Visual Studio

Before getting started using Visual Studio, it's a good idea to make a couple of changes.

Setting the Default Start-up Page

To change what you see each time that you go into Visual Studio, from the menu select: **T**ools **O**ptions... and then:



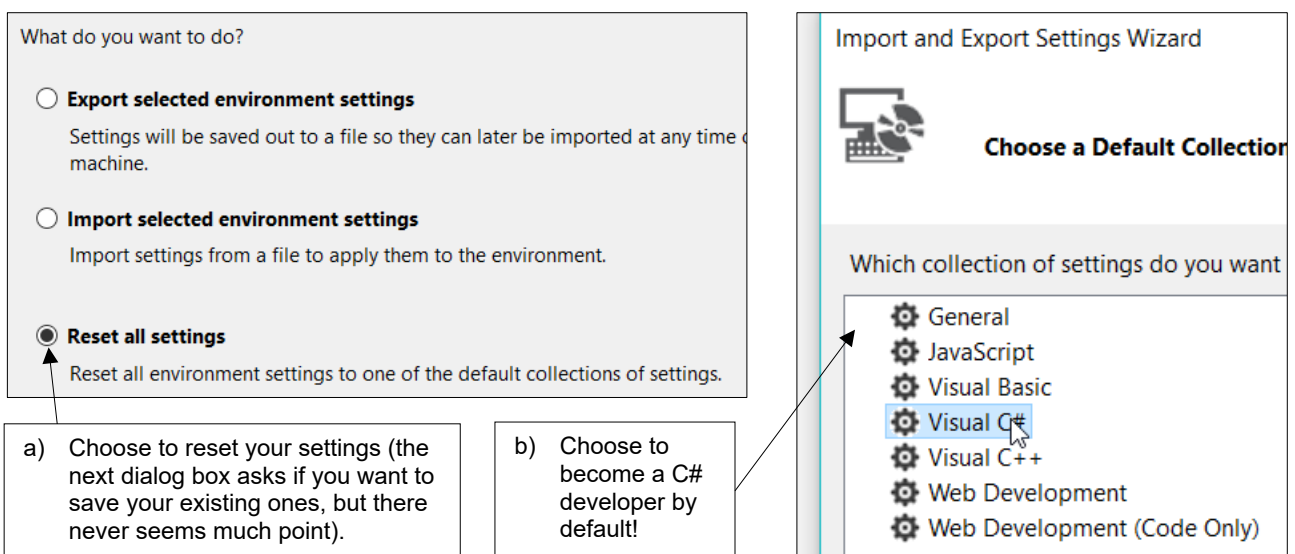
The screenshot shows the 'Options' dialog box with the 'Environment' section expanded to 'Startup'. The 'Load last loaded solution' option is highlighted. A warning message is visible at the bottom: 'Warning: Using a custom Start Page trusted can expose your computer to... referenced assemblies will run under...'

a) Select the **Startup** option under the **Environment** section.

b) Choose the option you want (the Wise Owl preference is to show the last thing you were working with, as here).

Creating Appropriate Settings

In Visual Studio you can save *settings*, letting you switch between different ways of working. From the menu choose **T**ools → **I**mport and **E**xport **S**ettings... and then:



The first dialog box asks 'What do you want to do?' with three radio button options: 'Export selected environment settings', 'Import selected environment settings', and 'Reset all settings'. The 'Reset all settings' option is selected. The second dialog box is the 'Import and Export Settings Wizard' with the title 'Choose a Default Collection'. It lists several collections: 'General', 'JavaScript', 'Visual Basic', 'Visual C#' (selected), 'Visual C++', 'Web Development', and 'Web Development (Code Only)'. A warning message is visible at the bottom: 'Warning: Using a custom Start Page trusted can expose your computer to... referenced assemblies will run under...'

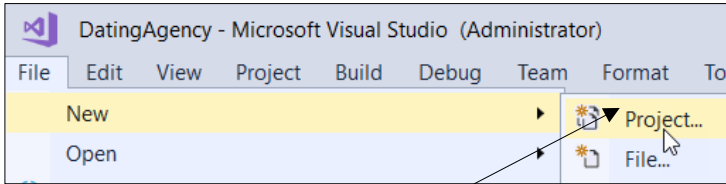
a) Choose to reset your settings (the next dialog box asks if you want to save your existing ones, but there never seems much point).

b) Choose to become a C# developer by default!

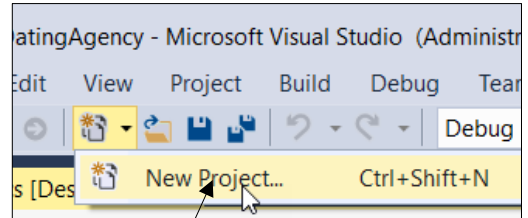
1.3 Creating Projects

A *project* is the term for the container for all of the files which make up your application. Projects can contain any number of folders and files. To create a project:

- 1) Choose one of the ways shown below to create a new project:



Either choose this menu option (you can also press **Shift** + **Ctrl** + **N** to do this) ...



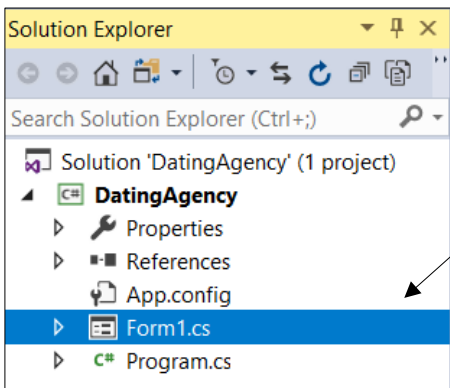
... or click on this tool to create a new project.

- 2) Complete the dialog box which appears as shown below, then select **OK**.

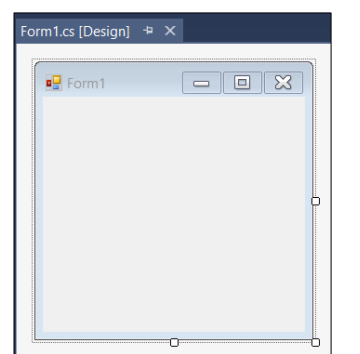
a) Choose to create a Windows application using C#.

b) Choose to create a WinForms application, not a WPF one.

c) Give your project a name, and choose where to store it.



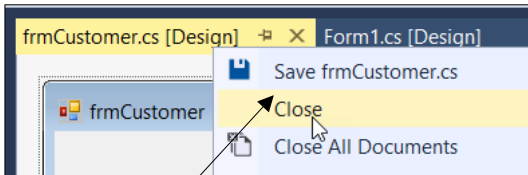
- d) Visual Studio automatically creates a new form called **Form1** in your new project, and shows this on screen:



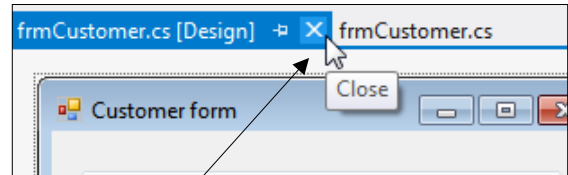
1.4 Saving and Closing Files

Closing One Window

To close a single window, either right-click on its tab or use the cross:



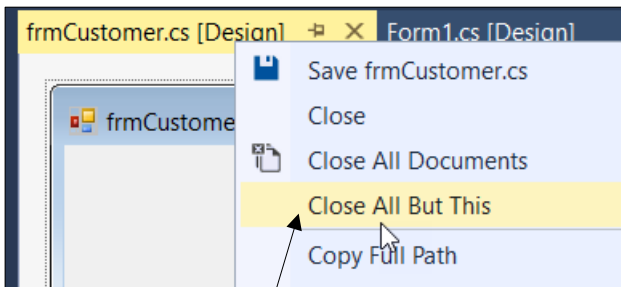
Either right-click on the window tab and choose the option shown to close it ...



... or click on this cross to close the window in question.

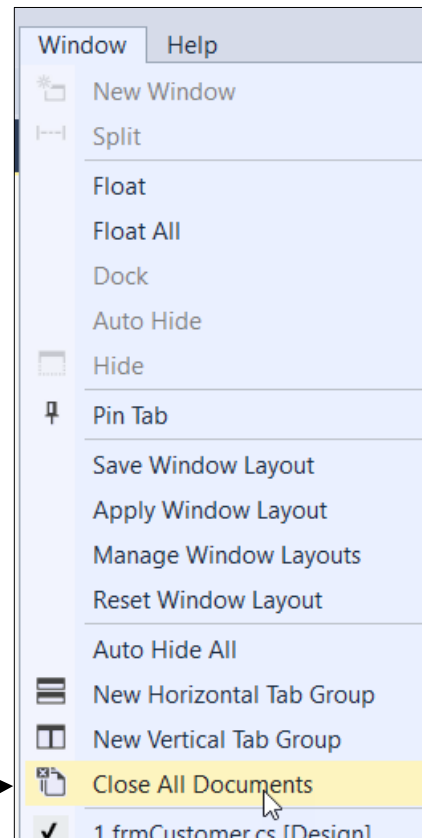
Closing All or Nearly All Windows

Windows accumulate quickly in Visual Studio – before you know where you are you have 8 or 10 open! Two quick ways to close all or nearly all of your windows are shown below:



Right-click on a window and choose this menu option to close all the windows except this one.

Select this menu option to close every window that you have open.

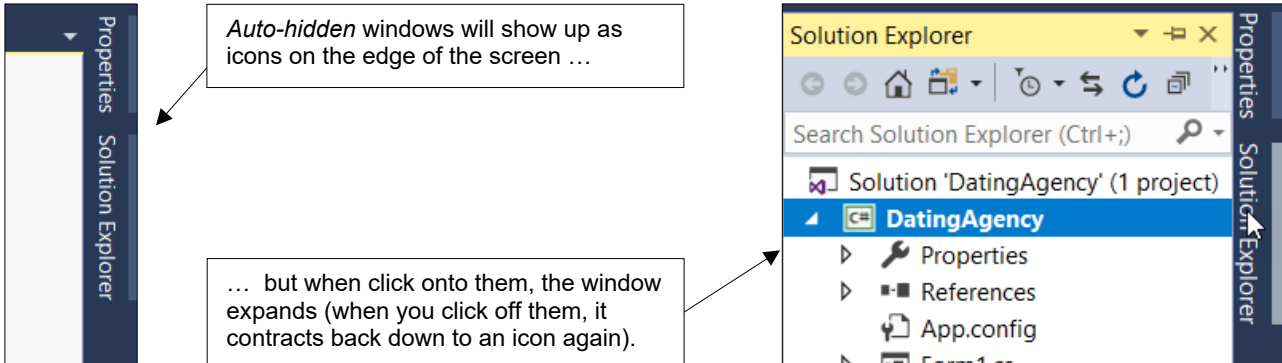


Wise Owl's Hint

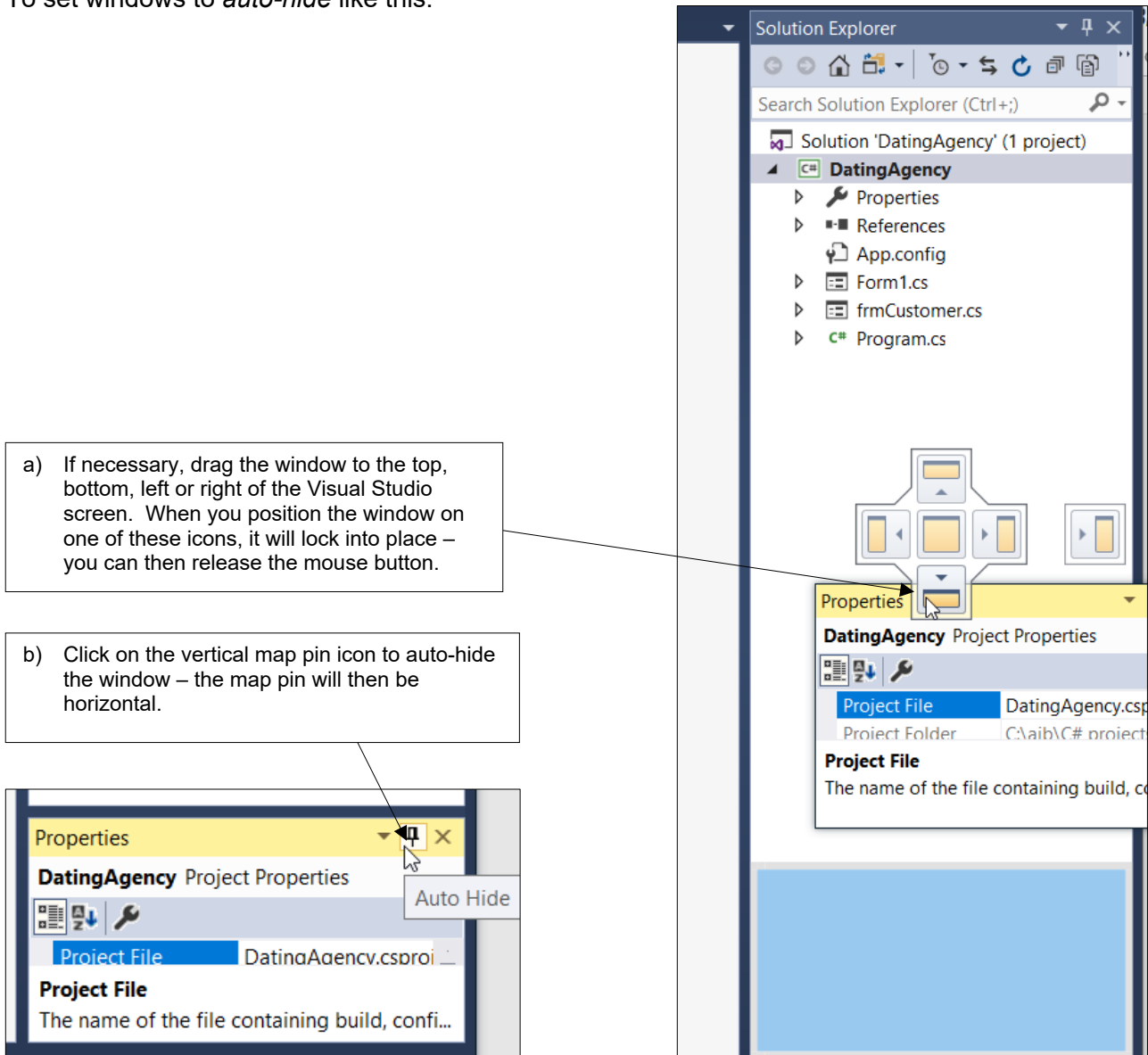
A useful short-cut is to press **Shift + Ctrl + S** to save every window that you have open – this means that you can then close all of your open windows without having to confirm that you want to save the changes for each.

1.5 Auto-hiding windows

There are a lot of windows in Visual Studio, and they can quickly clutter up the screen. The best thing to do is to *auto-hide* them, so that they show up as icons on the edge of the screen:



To set windows to *auto-hide* like this:



1.6 The Three Most Useful Windows

Displaying Windows

Visual Studio contains 3 toolbars that you'll need to have open most of the time – here's how to display them:

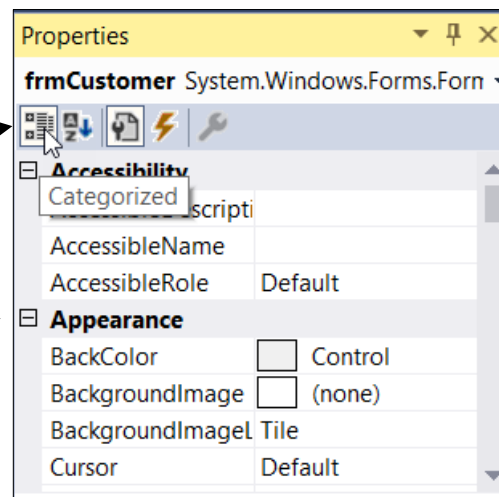
Windows	Menu option	Short-cut key
<i>Solution Explorer</i>	V iew Solution Explorer	Alt + Ctrl + L
<i>Properties</i>	V iew Properties Window	F4
<i>Toolbox</i>	V iew Toolbox	Alt + Ctrl + X

Properties Window

You can choose to display properties by category or alphabetically:

If you choose to display properties by category (the best option, this owl thinks) ...

... you'll see the properties grouped into categories (**A**ccessibility, **A**pppearance, etc.).

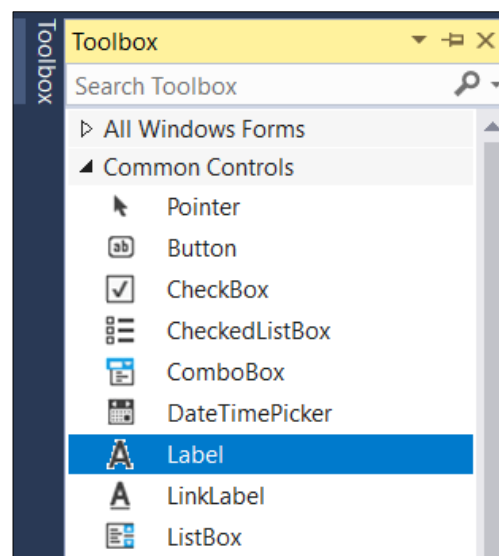


The Toolbox

The *toolbox* allows you to add things to a form quickly:

Click on any tool and drag it onto the form to add it. The main tools we'll use are:

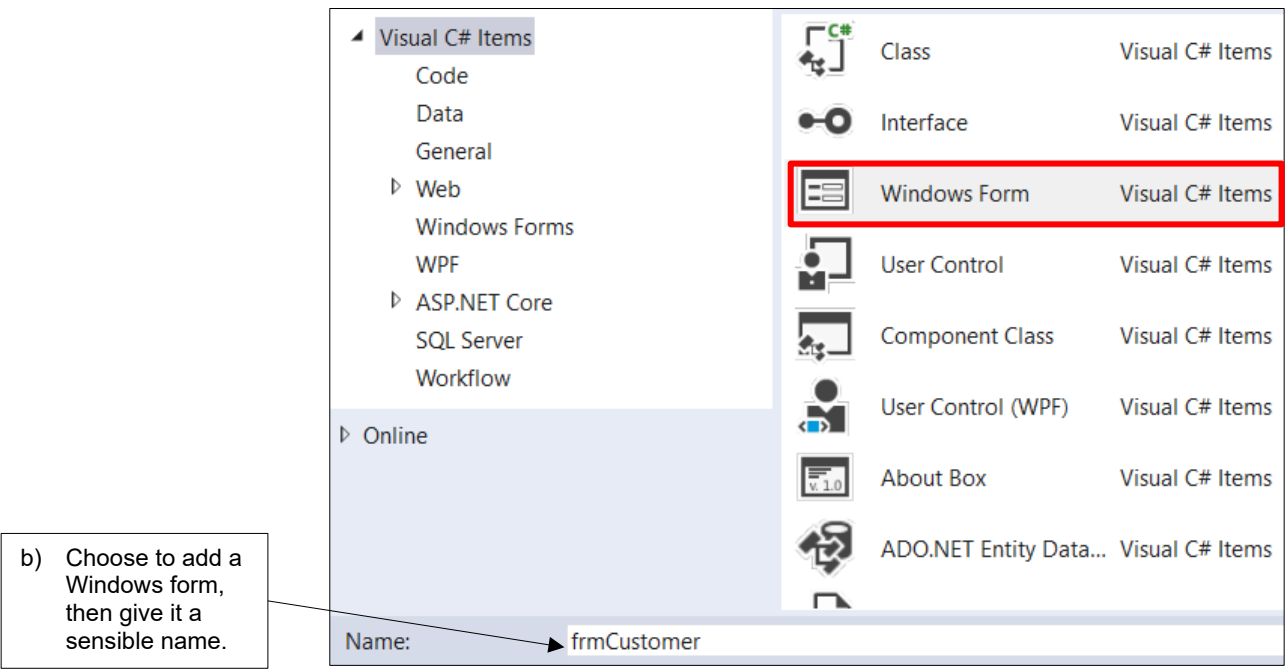
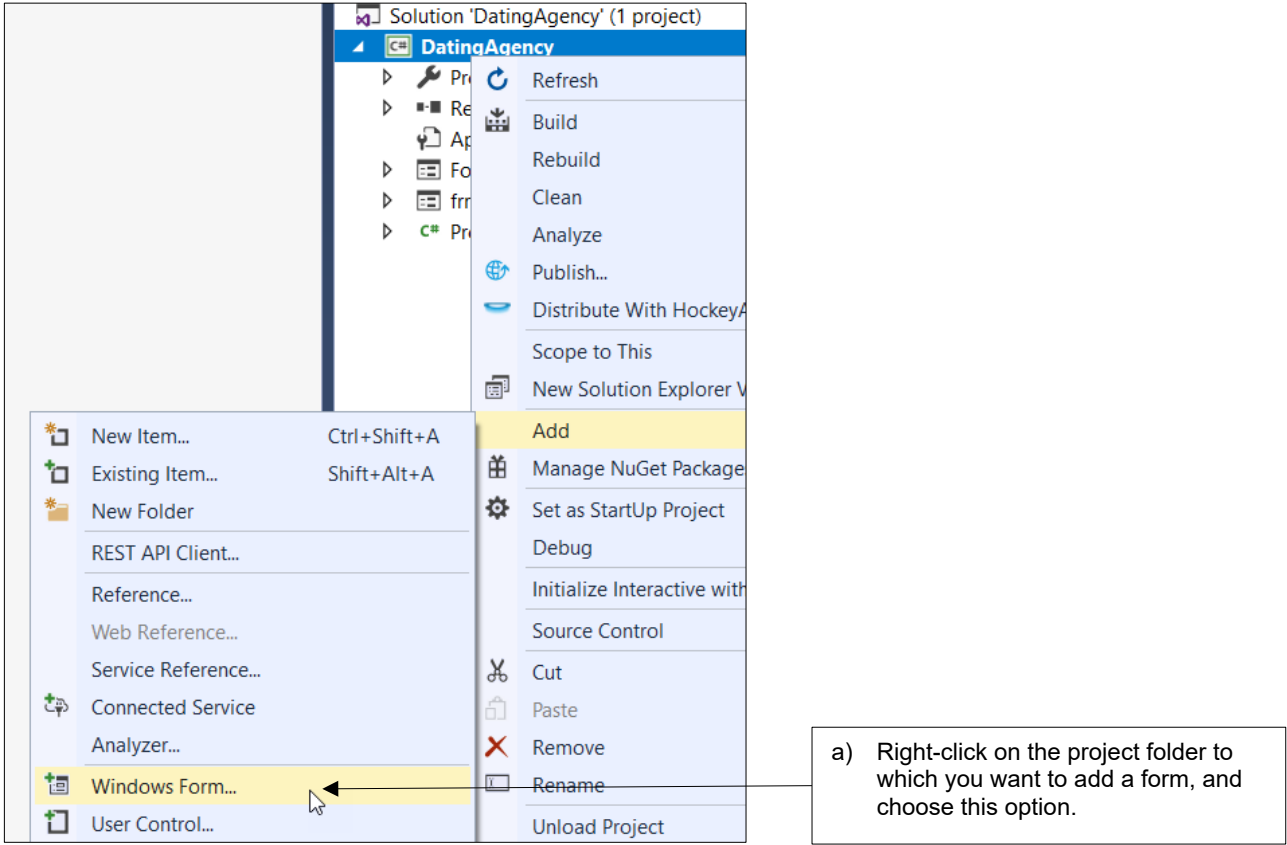
Tool	What it does
<i>Label</i>	Displays text on a form.
<i>TextBox</i>	Allows a user to type text into a box.
<i>Button</i>	A clickable button, with events attached.
<i>ComboBox</i>	A dropdown list.
<i>GroupBox</i>	Used to draw rectangles on forms.



CHAPTER 2 - DRAWING FORMS

2.1 Creating a New Form

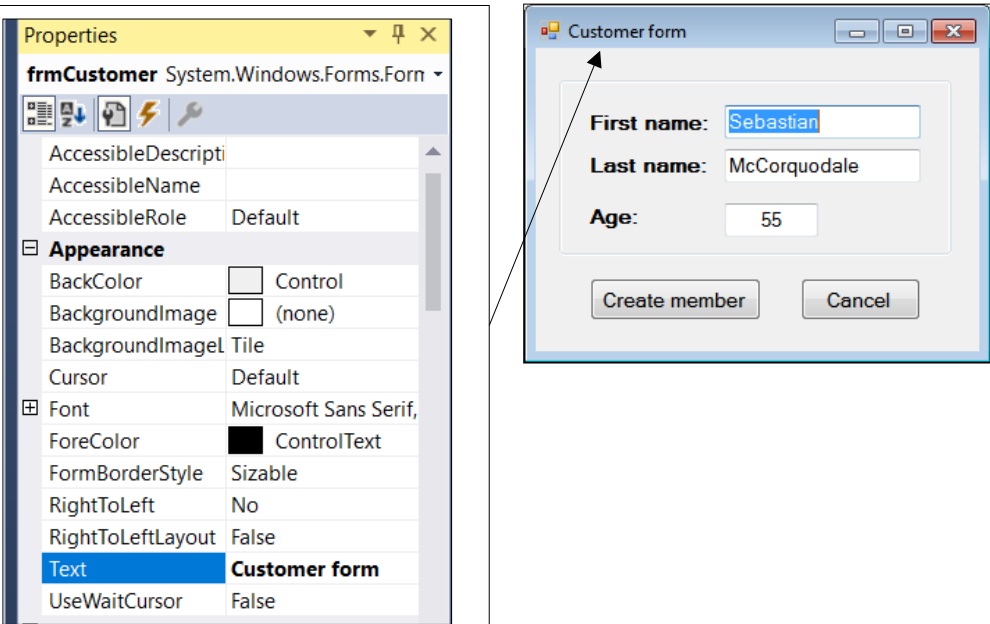
To create a new form, press **Alt** + **Ctrl** + **L** to bring up the Solution Explorer window, then:



2.2 Changing form properties

Click on the background of the form to make sure it's selected, then change any properties:

For example, the caption of the form comes from its **Text** property, which is in the **Appearance** category.



The image shows two windows. On the left is the 'Properties' window for a form named 'frmCustomer'. The 'Text' property is highlighted in blue and set to 'Customer form'. On the right is the 'Customer form' window, which has a title bar with the text 'Customer form'. Inside the window, there are three input fields: 'First name:' with the value 'Sebastian', 'Last name:' with the value 'McCorquodale', and 'Age:' with the value '55'. At the bottom of the window are two buttons: 'Create member' and 'Cancel'. An arrow points from the 'Text' property in the Properties window to the title bar of the form window.

Here are some useful properties to set for a form:

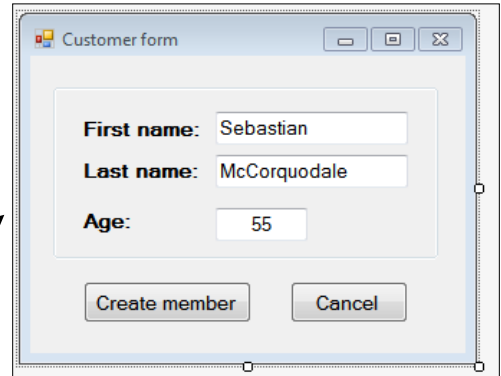
Category	Property	Notes
<i>Appearance</i>	Text	The caption for the form (as above).
<i>Layout</i>	StartPosition	Change to CenterScreen to make a form appear in the middle of the screen.
<i>Msc</i>	AcceptButton	Set to the button which you want to be selected by default if a user presses ↵ .
	CancelButton	Set to the button which you want to be selected by default if a user presses Esc .

2.3 Form Controls

Definition of Controls

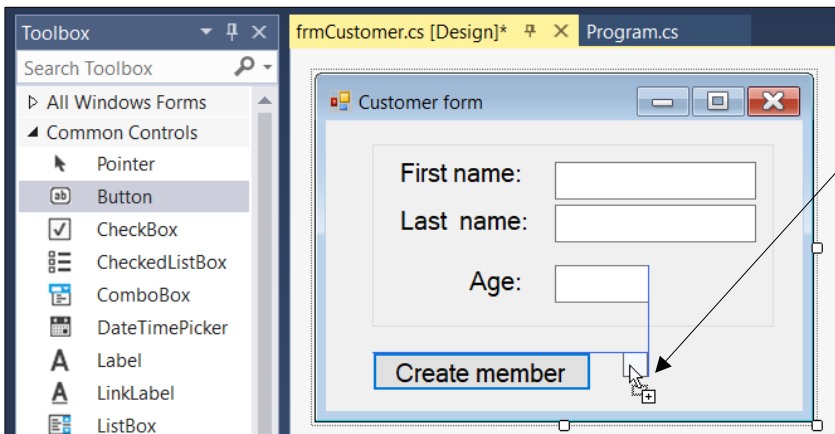
All the different widgets (labels, text boxes, command buttons, etc) that you add to a form are called *controls*:

This form contains 3 labels, 3 textboxes, 2 buttons and a groupbox (the thin rectangle).



Adding controls

The easiest way to add a control to a form is to click and drag on it:



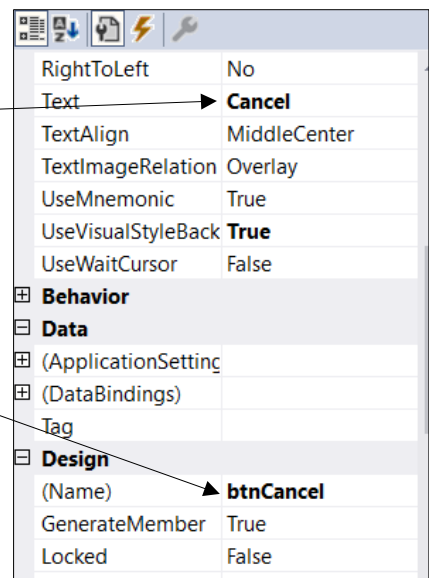
Here we're adding a clickable **Button** to a form. You can use the horizontal and vertical blue lines which appear to align controls.

You should then give your control a good name:

It's likely that you'll want to change the text displayed on a button.

The first **Design** category property of a control is always its name. Wise Owl use modified Hungarian notation (!) to name controls:

Type of control	Prefix to use
<i>TextBox</i>	txt
<i>Label</i>	lbl
<i>Button</i>	btn
<i>ComboBox</i>	cmb



2.4 Selecting Controls

Before you can move, copy, edit, delete, rename or format controls, you must first select them!

Selecting a Single Control

Here are two ways to select a single control:

Click on a control to select it ...

.. or click on the dropdown at the top of the **Properties** window to change what you have selected.

Selecting Several Controls

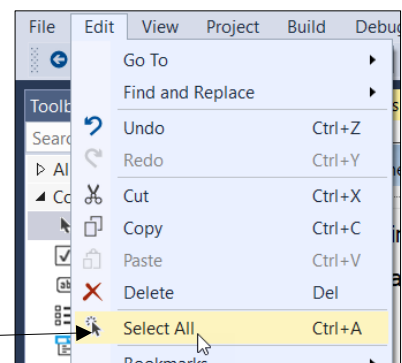
Suppose that you want to select the two buttons in the form above. Here are two ways to do this:

Method	Example diagram
Select one control, then hold down the Ctrl key and click on the others you want to add to/remove from your selection.	
Click and drag to draw a rectangle – anything it touches or encloses will be selected.	

Selecting All Controls

To do this, press **Ctrl** + **A** or select the menu option shown:

Select this option from the menu to select all of the controls on a form.



2.5 Basic Formatting

You can change how a form looks by *formatting* it. All of the examples below refer to the button, but the properties shown apply to most controls and to the form itself.



Resizing Controls

You can change the size of one or more controls using the handles surrounding them:





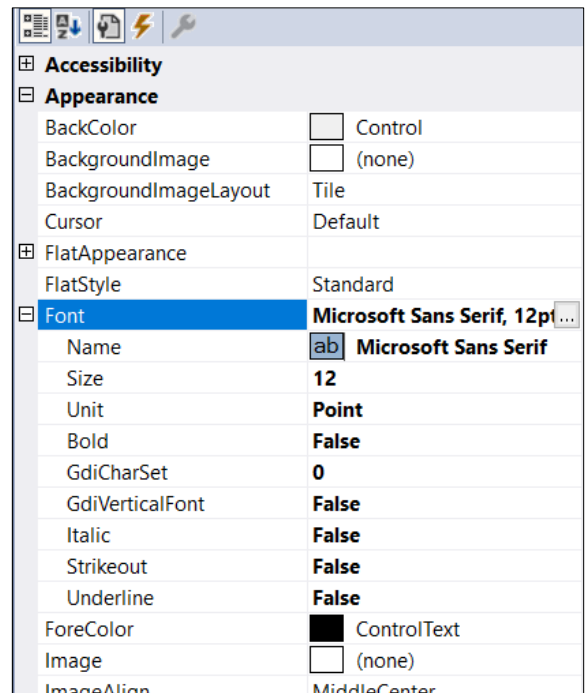
Clicking and dragging here, for example, would change the height of both buttons, since they are both selected.

Changing how Controls Look

To do this, select the control(s) and change any of their properties – here are a couple of examples:

You can change the appearance of controls using these properties – here are some ideas!

Example	Notes
	This font is set to MV Boli – and looks the worse for it!
	Changing the BackColor , ForeColor and FlatStyle properties.



Accessibility

Appearance

- BackColor: Control
- BackgroundImage: (none)
- BackgroundImageLayout: Tile
- Cursor: Default
- FlatAppearance
- FlatStyle: Standard
- Font: **Microsoft Sans Serif, 12pt...**
- Name: ab
- Size: **12**
- Unit: **Point**
- Bold: **False**
- GdiCharSet: **0**
- GdiVerticalFont: **False**
- Italic: **False**
- Strikeout: **False**
- Underline: **False**
- ForeColor: ControlText
- Image: (none)
- ImageAlign: MiddleCenter

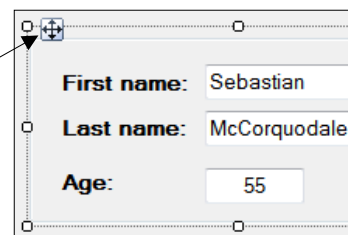
Moving Controls

To move controls, select them and then click and drag on them with the  symbol:










































Click and drag to move this textbox.

The only way to move this **GroupBox** is using this icon.



What we do!

		Basic training	Advanced training	Systems / consultancy
Office	Microsoft Excel			
	VBA macros			
	Office Scripts			
	Microsoft Access			
Business Intelligence	Power BI			
	Power Apps			
	Power Automate / PAD			
SQL Server	SQL			
	Reporting Services			
	Report Builder			
	Integration Services			
	Analysis Services			
Coding	Visual C# programming			
	VB programming			
	DAX			
	Python			



WiseOwl
Training

