Visual Basic.NET

VB.NET / Eng. Fallah NAJJAR

Iraq – An Najaf
Technical College of Management
Computer Center
2017

CONTENTS

fallahnajjar@yahoo.com
+9647807445833

fallahnajjar@yahoo.com
+9647807445833

fallahnajjar@yahoo.com
+9647807445833

fallahnajjar@yahoo.com
+9647807445833

fallahnajjar@yahoo.com
+9647807445833

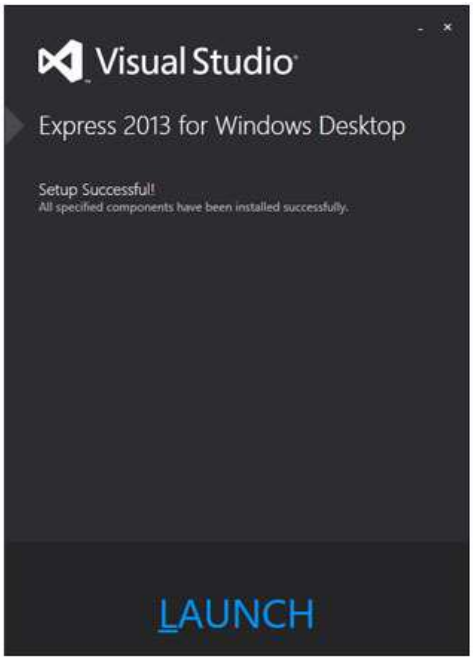
How to Install Visual Basic .NET

- Click the INSTALL link to the bottom.

- The setup process may take for a while so please be patient:

fallahnajjar@yahoo.com
+9647807445833

❖ Once you get the "Setup Successful!" message click the LAUNCH link



❖ The new Welcome dialog will offer you to **Sign in to Visual Studio** which is optional so just click **"Not now, maybe later"**.

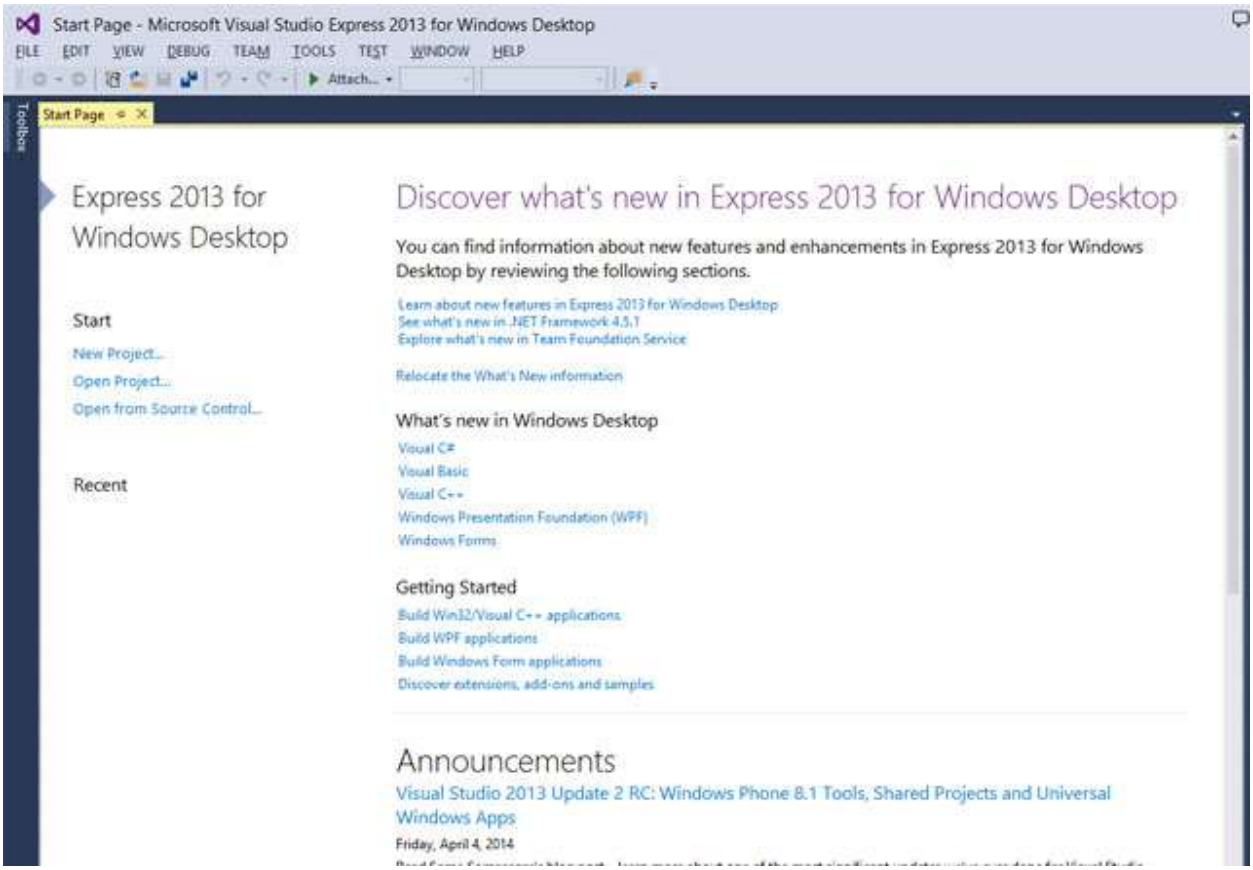
fallahnajjar@yahoo.com
+9647807445833

- After a while the new dialog will be shown up saying that **it's preparing for first use**.

- Now you have the latest Visual Studio Express for Windows Desktop running on your computer. **Congratulations!**

fallahnajjar@yahoo.com
+9647807445833

❖ By default, you'll see the **Start Page** showing Start section, Recent Projects (empty as for now), Getting Started etc.



fallahnajjar@yahoo.com
+9647807445833

Introduction

Visual Basic .NET (VB.NET) is an object-oriented computer programming language implemented on the .NET Framework. Although it is an evolution of classic Visual Basic language, it is not backwards-compatible with VB6, and any code written in the old version does not compile under VB.NET.

The following reasons make VB.Net a widely used professional language:

- Modern, general purpose.
- Object oriented.
- Component oriented.
- Easy to learn.
- Structured language.
- It produces efficient programs.
- It can be compiled on a variety of computer platforms.
- Part of .Net Framework.

VB.Net - Program Structure

A VB.Net program basically consists of the following parts:

- Namespace declaration
- A class or module
- One or more procedures
- Variables
- The Main procedure
- Statements & Expressions
- Comments

fallahnajjar@yahoo.com
+9647807445833

Let us now briefly look into what do class, object, methods and instance variables mean.

- **Object** - Objects have states and behaviors. Example: A dog has states - color, name, breed as well as behaviors - wagging, barking, eating, etc. An object is an instance of a class.
- **Class** - A class can be defined as a template/blueprint that describes the behaviors/states that objects of its type support.
- **Methods** - A method is basically a behavior. A class can contain many methods. It is in methods where the logics are written, data is manipulated and all the actions are executed.
- **Instance Variables** - Each object has its unique set of instance variables. An object's state is created by the values assigned to these instance variables.

Identifiers

An identifier is a name used to identify a class, variable, function, or any other user-defined item. The basic rules for naming classes in VB.Net are as follows:

- A name must begin with a letter that could be followed by a sequence of letters, digits (0 - 9) or underscore. The first character in an identifier cannot be a digit.
- It must not contain any embedded space or symbol like **(? - +! @ # % ^ & * () [] {} . ; : " ' / and** \). However, an underscore (_) can be used.
- It should not be a reserved keyword.

fallahnajjar@yahoo.com
+9647807445833

Data Types in VB.Net

VB.Net provides a wide range of data types. The following table shows all the data types available:

Data Type	Storage Allocation
Boolean	Depends on implementing platform
Byte	1 byte
Char	2 bytes
Date	8 bytes
Decimal	16 bytes
Double	8 bytes
Integer	4 bytes
Long	8 bytes
Object	4 bytes on 32-bit platform 8 bytes on 64-bit platform
SByte	1 byte
Short	2 bytes
Single	4 bytes
String	Depends on implementing platform
UInteger	4 bytes
ULong	8 bytes
User-Defined	Depends on implementing platform
UShort	2 bytes

fallahnajjar@yahoo.com
+9647807445833

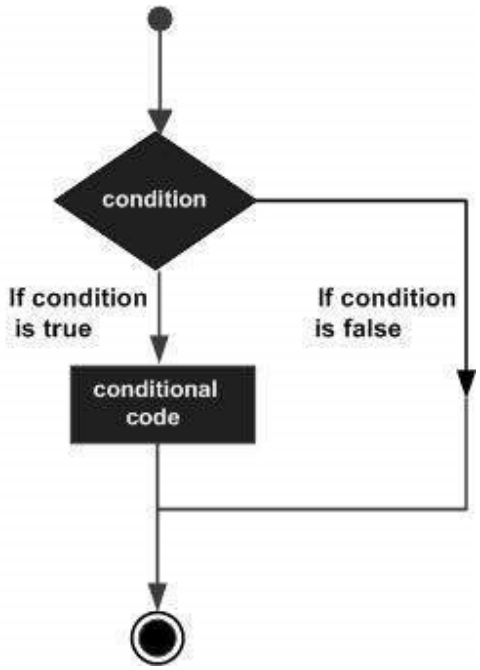
A variable is nothing but a name given to a storage area that our programs can manipulate. Each variable in VB.Net has a specific type, which determines the size and layout of the variable's memory; the range of values that can be stored within that memory; and the set of operations that can be applied to the variable.

We have already discussed various data types. The basic value types provided in VB.Net can be categorized as:

Type	Example
Integral types	SByte, Byte, Short, UShort, Integer, UInteger, Long, ULong and Char
Floating point types	Single and Double
Decimal types	Decimal
Boolean types	True or False values, as assigned
Date types	Date

Decision Making

Decision making structures require that the programmer specify one or more conditions to be evaluated or tested by the program, along with a statement or statements to be executed if the condition is determined to be true, and optionally, other statements to be executed if the condition is determined to be false.



Following is the general form of a typical decision-making structure found in most of the programming languages:

VB.Net provides the following types of decision making statements. Click the following links to check their details.

fallahnajjar@yahoo.com
+9647807445833

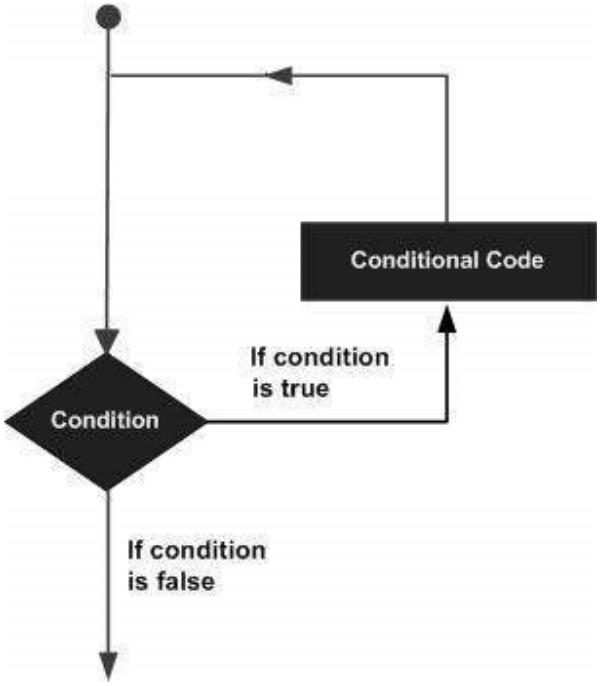
Statement	Description
If ... Then statement	An **If...Then statement** consists of a Boolean expression followed by one or more statements.
If...Then...Else statement	An **If...Then statement** can be followed by an optional **Else statement**, which executes when the Boolean expression is false.
nested If statements	You can use one **If** or **Else if** statement inside another **If** or **Else if** statement(s).
Select Case statement	A **Select Case** statement allows a variable to be tested for equality against a list of values.
nested Select Case statements	You can use one **select case** statement inside another **select case** statement(s).

Loops

There may be a situation when you need to execute a block of code several number of times. In general, statements are executed sequentially: The first statement in a function is executed first, followed by the second, and so on.

Programming languages provide various control structures that allow for more complicated execution paths.

A loop statement allows us to execute a statement or group of statements multiple times and following is the general form of a loop statement in most of the programming languages:



VB.Net provides following types of loops to handle looping requirements. Click the following links to check their details.

fallahnajjar@yahoo.com
+9647807445833

Loop Type	Description
Do Loop	It repeats the enclosed block of statements while a Boolean condition is True or until the condition becomes True. It could be terminated at any time with the Exit Do statement.
For...Next	It repeats a group of statements a specified number of times and a loop index counts the number of loop iterations as the loop executes.
For Each...Next	It repeats a group of statements for each element in a collection. This loop is used for accessing and manipulating all elements in an array or a VB.Net collection.
While... End While	It executes a series of statements as long as a given condition is True.
With... End With	It is not exactly a looping construct. It executes a series of statements that repeatedly refer to a single object or structure.
Nested loops	You can use one or more loops inside any another While, For or Do loop.

Loop Control Statements:

Loop control statements change execution from its normal sequence. When execution leaves a scope, all automatic objects that were created in that scope are destroyed.

VB.Net provides the following control statements. Click the following links to check their details.

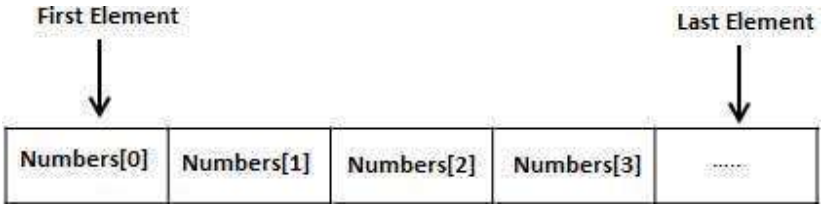
Control Statement	Description
Exit statement	Terminates the **loop** or **select case** statement and transfers execution to the statement immediately following the loop or select case.
Continue statement	Causes the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.
GoTo statement	Transfers control to the labeled statement. Though it is not advised to use GoTo statement in your program.

fallahnajjar@yahoo.com
+9647807445833

Arrays

An array stores a fixed-size sequential collection of elements of the same type. An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.

All arrays consist of contiguous memory locations. The lowest address corresponds to the first element and the highest address to the last element.



Creating Arrays in VB.Net

To declare an array in VB.Net, you use the Dim statement. For example,

```
Dim intData(30)              ' an array of 31 elements
Dim strData(20) As String        ' an array of 21 strings
Dim twoDarray(10, 20) As Integer         'a two dimensional array of integers
Dim ranges(10, 100)      'a two dimensional array
```

You can also initialize the array elements while declaring the array. For example,

```
Dim intData() As Integer = {12, 16, 20, 24, 28, 32}
Dim names() As String = {"Karthik", "Sandhya", "Shivangi", "Ashwitha", "Somnath"}
Dim miscData() As Object = {"Hello World", 12d, 16ui, "A"c}
```

Multi-Dimensional Arrays

VB.Net allows multidimensional arrays. Multidimensional arrays are also called rectangular arrays.

You can declare a 2-dimensional array of strings as:

```
Dim twoDStringArray(10, 20) As String
```

or, a 3-dimensional array of Integer variables:

```
Dim threeDIntArray(10, 10, 10) As Integer
```


fallahnajjar@yahoo.com
+9647807445833

Functions

A procedure is a group of statements that together perform a task when called. After the procedure is executed, the control returns to the statement calling the procedure. VB.Net has two types of procedures:

- Functions
- Sub procedures or Subs

Functions return a value, whereas Subs do not return a value.

Defining a Function

The Function statement is used to declare the name, parameter and the body of a function. The syntax for the Function statement is:

```
[Modifiers] Function FunctionName [(ParameterList)] As ReturnType
   [Statements]
End Function
```

Where,

- ***Modifiers***: specify the access level of the function; possible values are: Public, Private, Protected, Friend, Protected Friend and information regarding overloading, overriding, sharing, and shadowing.
- ***FunctionName***: indicates the name of the function
- ***ParameterList***: specifies the list of the parameters
- ***ReturnType***: specifies the data type of the variable the function returns

Function Returning a Value

In VB.Net, a function can return a value to the calling code in two ways:

- By using the return statement
- By assigning the value to the function name

fallahnajjar@yahoo.com
+9647807445833

Sub Procedures

As we mentioned in the previous chapter, Sub procedures are procedures that do not return any value. We have been using the Sub procedure Main in all our examples. We have been writing console applications so far in these tutorials. When these applications start, the control goes to the Main Sub procedure, and it in turn, runs any other statements constituting the body of the program.

Defining Sub Procedures

The **Sub** statement is used to declare the name, parameter and the body of a sub procedure. The syntax for the Sub statement is:

```
[Modifiers] Sub SubName [(ParameterList)]
   [Statements]
End Sub
```

Where,

- ***Modifiers***: specify the access level of the procedure; possible values are: Public, Private, Protected, Friend, Protected Friend and information regarding overloading, overriding, sharing, and shadowing.
- ***SubName***: indicates the name of the Sub
- ***ParameterList***: specifies the list of the parameters

fallahnajjar@yahoo.com
+9647807445833

Classes and Objects

Generally speaking, a class is a software component that defines and implements one or more interfaces. (Strictly speaking, a class need not implement all the members of an interface. We discuss this later when we talk about abstract members.) In different terms, a class combines data, functions, and types into a new type. Microsoft uses the term type to include classes.

Class Modules in VB.NET

Under Visual Studio.NET, a VB class module is inserted into a project using the Add Class menu item on the Project menu. This inserts a new module containing the code:

```
Public Class ClassName

End Class
```

Although Visual Studio stores each class in a separate file, this isn't a requirement. It is the Class...End Class construct that marks the beginning and end of a class definition. Thus, the code for more than one class as well as one or more code modules (which are similarly delimited by the Module...End Module construct) can be contained in a single source code file.

The C-Person class defined in the next section is an example of a VB class module.

Class Members

In VB.NET, class modules can contain the following types of members:

- **Data members**

This includes member variables (also called fields) and constants.

fallahnajjar@yahoo.com
+9647807445833

❖ **Event members**

Events are procedures that are called automatically by the Common Language Runtime in response to some action that occurs, such as an object being created, a button being clicked, a piece of data being changed, or an object going out of scope.

❖ **Function members**

This refers to both functions and subroutines. A function member is also called a method. A class' constructor is a special type of method.

❖ **Property members**

A property member is implemented as a Private member variable together with a special type of VB function that incorporates both accessor functions of the property.

❖ **Type members**

A class member can be another class, which is then referred to as a nested class.

Inheritance

One of the most important concepts in object-oriented programming is that of inheritance. Inheritance allows us to define a class in terms of another class which makes it easier to create and maintain an application. This also provides an opportunity to reuse the code functionality and fast implementation time.

When creating a class, instead of writing completely new data members and member functions, the programmer can designate that the new class should inherit the members of an existing class. This existing class is called the base class, and the new class is referred to as the derived class.

Base & Derived Classes:

A class can be derived from more than one class or interface, which means that it can inherit data and functions from multiple base classes or interfaces.

*** VB.Net supports multiple inheritance.**

fallahnajjar@yahoo.com
+9647807445833

Dialog Boxes

There are many built-in dialog boxes to be used in Windows forms for various tasks like opening and saving files, printing a page, providing choices for colors, fonts, page setup, etc., to the user of an application. These built-in dialog boxes reduce the developer's time and workload.

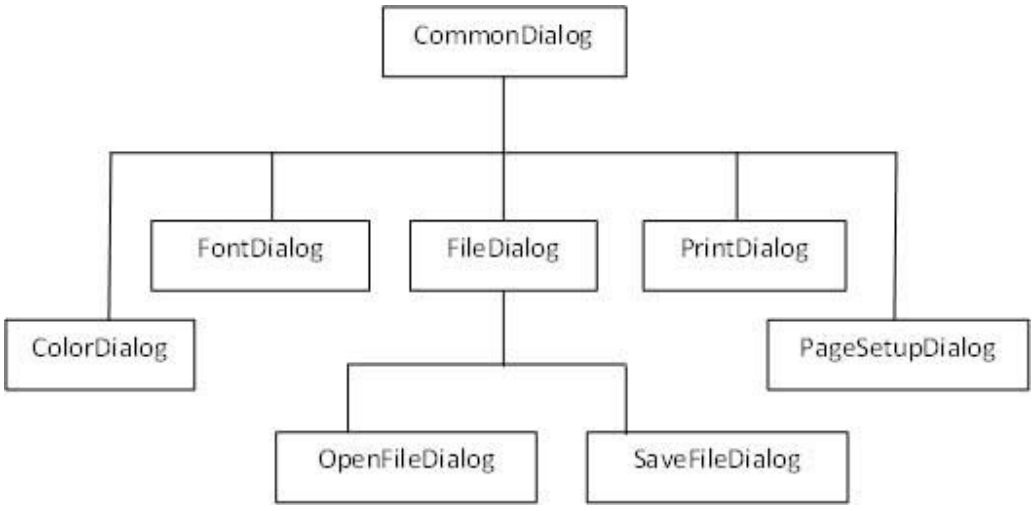
All of these dialog box control classes inherit from the **CommonDialog** class and override the *RunDialog()* function of the base class to create the specific dialog box.

The RunDialog() function is automatically invoked when a user of a dialog box calls its *ShowDialog()* function.

The **ShowDialog** method is used to display all the dialog box controls at run-time. It returns a value of the type of **DialogResult** enumeration. The values of DialogResult enumeration are:

- **Abort** - returns DialogResult.Abort value, when user clicks an Abort button.
- **Cancel**- returns DialogResult.Cancel, when user clicks a Cancel button.
- **Ignore** - returns DialogResult.Ignore, when user clicks an Ignore button.
- **No** - returns DialogResult.No, when user clicks a No button.
- **None** - returns nothing and the dialog box continues running.
- **OK** - returns DialogResult.OK, when user clicks an OK button
- **Retry** - returns DialogResult.Retry, when user clicks an Retry button
- **Yes** - returns DialogResult.Yes, when user clicks an Yes button

The following diagram shows the common dialog class inheritance:



fallahnajjar@yahoo.com
+9647807445833

All these above-mentioned classes have corresponding controls that could be added from the Toolbox during design time. You can include relevant functionality of these classes to your application, either by instantiating the class programmatically or by using relevant controls.

When you double click any of the dialog controls in the toolbox or drag the control onto the form, it appears in the Component tray at the bottom of the Windows Forms Designer, they do not directly show up on the form.

The following table lists the commonly used dialog box controls. Click the following links to check their detail:

S.N.	Control & Description
1	**ColorDialog** It represents a common dialog box that displays available colors along with controls that enable the user to define custom colors.
2	**FontDialog** It prompts the user to choose a font from among those installed on the local computer and lets the user select the font, font size, and color.
3	**OpenFileDialog** It prompts the user to open a file and allows the user to select a file to open.
4	**SaveFileDialog** It prompts the user to select a location for saving a file and allows the user to specify the name of the file to save data.
5	**PrintDialog** It lets the user to print documents by selecting a printer and choosing which sections of the document to print from a Windows Forms application.

fallahnajjar@yahoo.com
+9647807445833

Exception Handling

- An exception is a problem that arises during the execution of a program. An exception is a response to an exceptional circumstance that arises while a program is running, such as an attempt to divide by zero.

  Exceptions provide a way to transfer control from one part of a program to another. VB.Net exception handling is built upon four keywords: Try, Catch and Finally.

- **Try**: A Try block identifies a block of code for which particular exceptions will be activated. It's followed by one or more Catch blocks.
- **Catch**: A program catches an exception with an exception handler at the place in a program where you want to handle the problem. The Catch keyword indicates the catching of an exception.
- **Finally**: The Finally block is used to execute a given set of statements, whether an exception is thrown or not thrown. For example, if you open a file, it must be closed whether an exception is raised or not.

Syntax

Assuming a block will raise an exception, a method catches an exception using a combination of the Try and Catch keywords. A Try/Catch block is placed around the code that might generate an exception. Code within a Try/Catch block is referred to as protected code, and the syntax for using Try/Catch looks like the following:

```
Try
    [ tryStatements ]
    [ Exit Try ]
[ Catch [ exception [ As type ] ] [ When expression ]
    [ catchStatements ]
    [ Exit Try ] ]
[ Catch ... ]
[ Finally
    [ finallyStatements ] ]
End Try
```

You can list down multiple catch statements to catch different type of exceptions in case your try block raises more than one exception in different situations.

fallahnajjar@yahoo.com
+9647807445833

Toolbox Controls

Controls make programming a lot easier and faster. You can drag and drop a button or a text box to the designer and their codes will be generated automatically.

Pointer	ErrorProvider	NotifyIcon	SaveFileDialog
BackgroundWorker	EventLog	NumericUpDown	SerialPort
BindingNavigator	FlowLayoutPanel	OpenFileDialog	ServiceController
BindingSource	FolderBrowserDialog	PageSetupDialog	SplitContainer
Button	FontDialog	Panel	Splitter
Chart	GroupBox	PerformanceCounter	StatusStrip
CheckBox	HelpProvider	PictureBox	TabControl
CheckedListBox	HScrollBar	PrintDialog	TableLayoutPanel
ColorDialog	ImageList	PrintDocument	TextBox
ComboBox	Label	PrintPreviewControl	Timer
ContextMenuStrip	LinkLabel	PrintPreviewDialog	ToolStrip
DataGridView	ListBox	Process	ToolStripContainer
DataSet	ListView	ProgressBar	ToolTip
DateTimePicker	MaskedTextBox	PropertyGrid	TrackBar
DirectoryEntry	MenuStrip	RadioButton	TreeView
DirectorySearcher	MessageQueue	ReportViewer	VScrollBar
DomainUpDown	MonthCalendar	RichTextBox	WebBrowser

I'll describe the most common properties and events for common tools control

fallahnajjar@yahoo.com
+9647807445833

VB.Net - Form Control

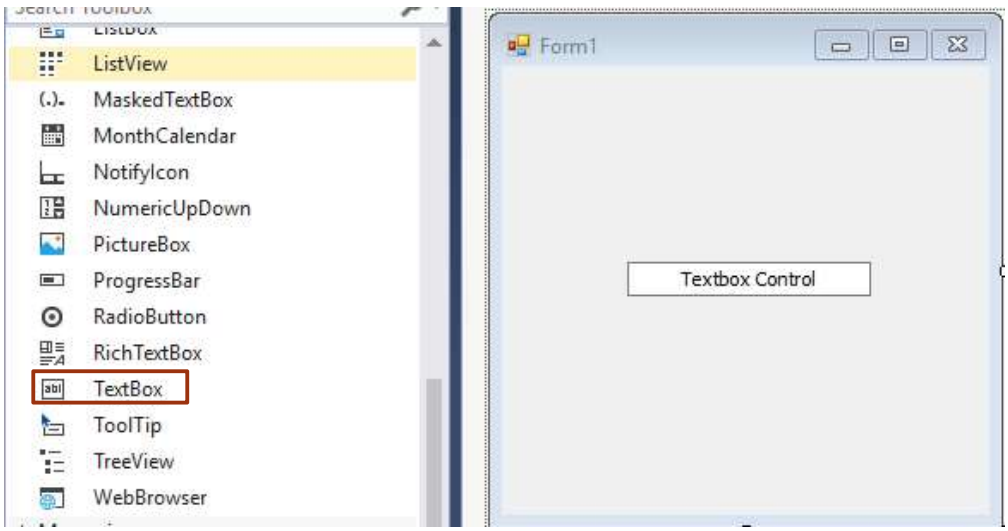
Property	Description
Name	This is the actual name of the form.
Text	The text, which will appear at the title bar of the form.

Form Events

Event	Description
Load	Occurs before a form is displayed for the first time.
Event	Description

VB.Net - TextBox Control

Let's create a text box by dragging a Text Box control from the Toolbox and dropping it on the form.



fallahnajjar@yahoo.com
+9647807445833

TextBox Properties

Property	Description
Multiline	Gets or sets a value indicating whether this is a multiline TextBox control.
PasswordChar	Gets or sets the character used to mask characters of a password in a single-line TextBox control.
Text	Gets or sets the current text in the TextBox.
TextAlign	Gets or sets how text is aligned in a TextBox control. This property has

TextBox Methods

Method	Description
Clear	Clears all text from the text box control.
ToString	Returns a string that represents the TextBox Base control.

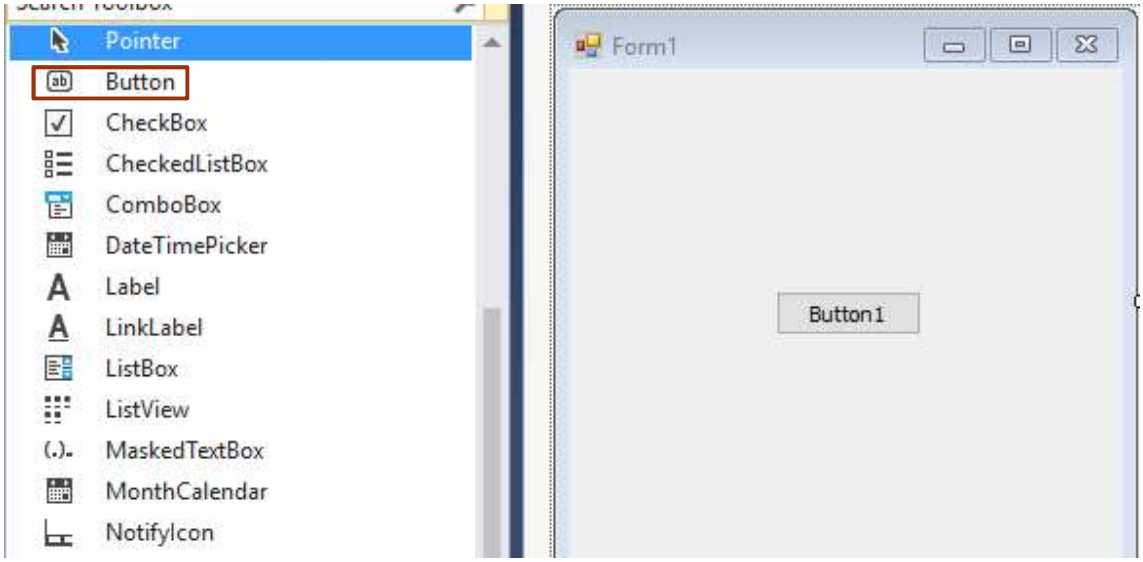
TextBox Events

Event	Description
Click	Occurs when the control is clicked.
DoubleClick	Occurs when the control is double-clicked.
TextAlignChanged	Occurs when the TextAlign property value changes.

fallahnajjar@yahoo.com
+9647807445833

VB.Net - Button Control

Let's create a label by dragging a Button control from the Toolbox ad dropping it on the form



Button Properties

Property	Description
Text	Gets or sets the text associated with this control.

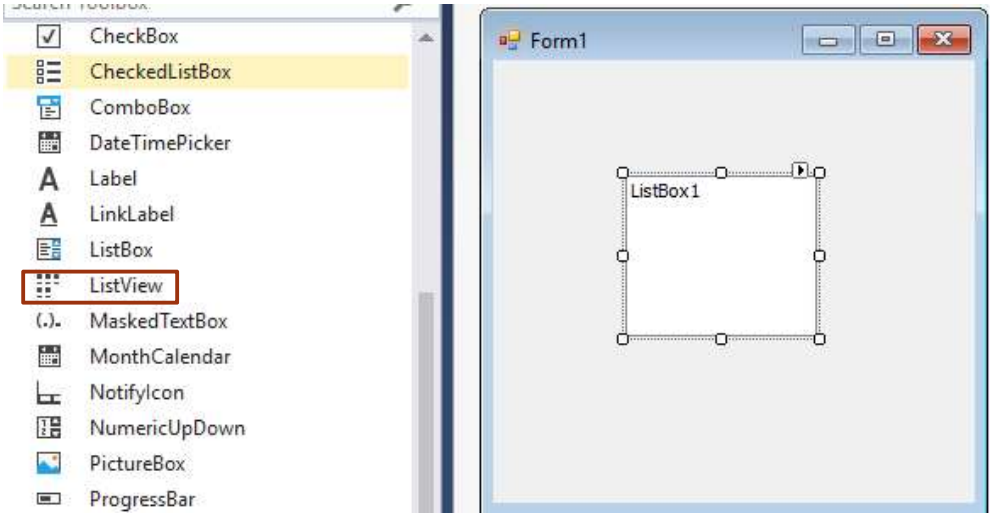
Button Events

Event	Description
Click	Occurs when the control is clicked.

fallahnajjar@yahoo.com
+9647807445833

VB.Net - ListBox Control

Let's create a list box by dragging a ListBox control from the Toolbox and dropping it on the form.



ListBox Properties

Property	Description
Items	Gets the items of the list box.
MultiColumn	Gets or sets a value indicating whether the list box supports multiple columns.
SelectedIndex	Gets or sets the zero-based index of the currently selected item in a list box.
SelectedItem	Gets or sets the currently selected item in the list box.
SelectedItems	Gets a collection containing the currently selected items in the list box.
Sorted	Gets or sets a value indicating whether the items in the list box are sorted alphabetically.
Text	Gets or searches for the text of the currently selected item in the list box.

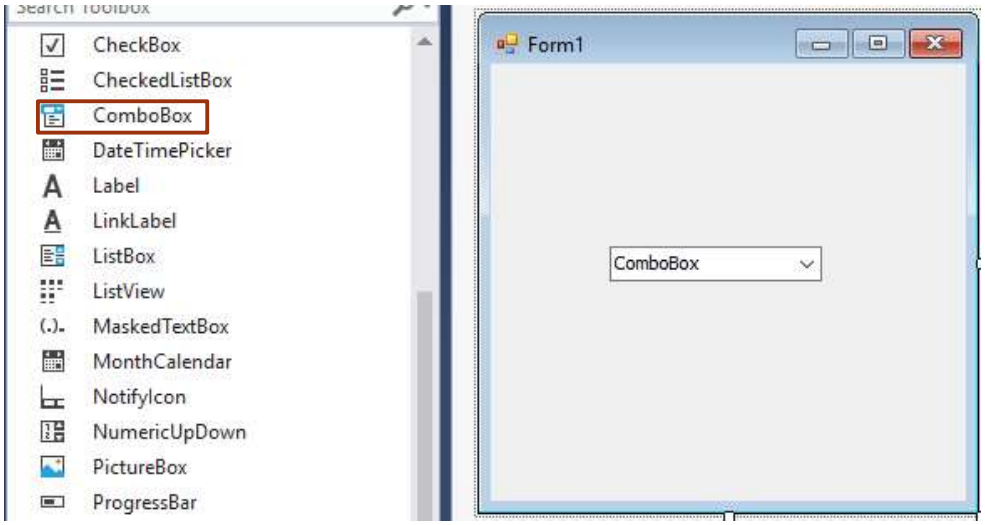
fallahnajjar@yahoo.com
+9647807445833

ListBox Events

Event	Description
Click	Occurs when a list box is selected.
SelectedIndexChanged	Occurs when the SelectedIndex property of a list box is changed.

VB.Net - ComboBox Control

Let's create a combo box by dragging a ComboBox control from the Toolbox and dropping it on the form.



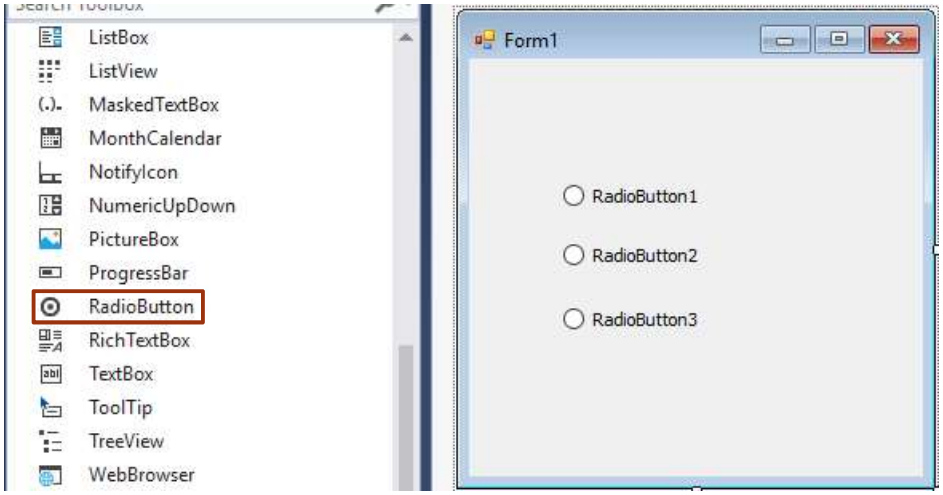
ComboBox Properties

Property	Description
Items	Gets an object representing the collection of the items contained in this ComboBox.
SelectedIndex	Gets or sets the index specifying the currently selected item.
SelectedText	Gets or sets the text that is selected in the editable portion of a ComboBox.
Sorted	Gets or sets a value indicating whether the items in the combo box are sorted.
Text	Gets or sets the text associated with this control.

fallahnajjar@yahoo.com
+9647807445833

VB.Net - RadioButton Control

Let's create three radio buttons by dragging Radio Button controls from the Toolbox and dropping on the form.



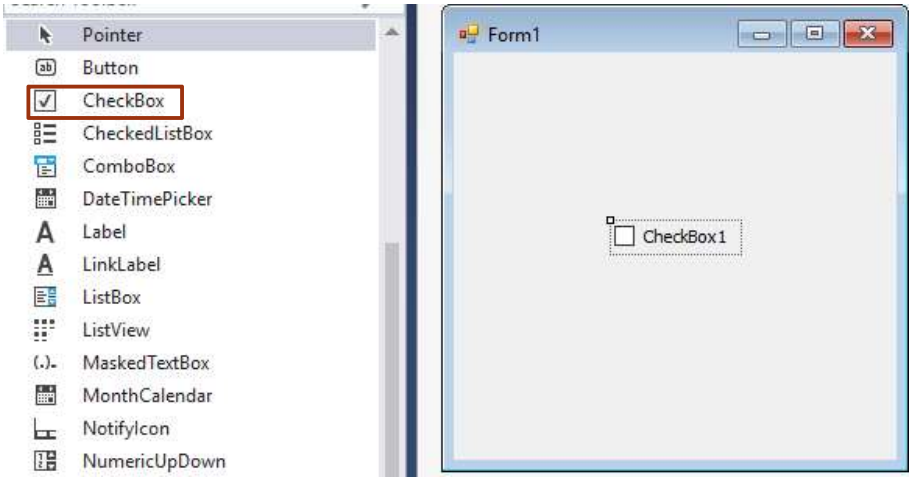
RadioButton Properties

Property	Description
Checked	Gets or sets a value indicating whether the control is checked.
Text	Gets or sets the caption for a radio button.

fallahnajjar@yahoo.com
+9647807445833

VB.Net - CheckBox Control

Let's create two check boxes by dragging Check Box controls from the Toolbox and dropping on the form.



CheckBox Properties

Property	Description
Checked	Gets or sets a value indicating whether the check box is selected.
Text	Gets or sets the caption of a check box.

Note: the previous tools with their properties, methods, and event not all things, there are many tools have many.

Now I'll start with simple projects code

fallahnajjar@yahoo.com
+9647807445833

Project 1: Simple Clock

Add to your Form project

- Label
- Timer

Timer Properties

Property	Set
Enables	False
Interval	100

Source Code

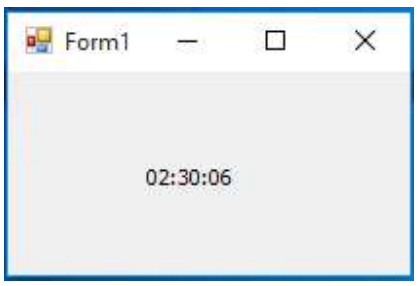
Double click (DC) on Timer1, then write

```
Me.Label1.Text = Now.ToString("HH:mm:ss")
```

Double click (DC) on Label1, then write

```
Timer1.Enabled = True
```

Then run your application (app) to get:



fallahnajjar@yahoo.com
+9647807445833

Project 2: Difference in days

Add to your Form project

- Button

Button Properties

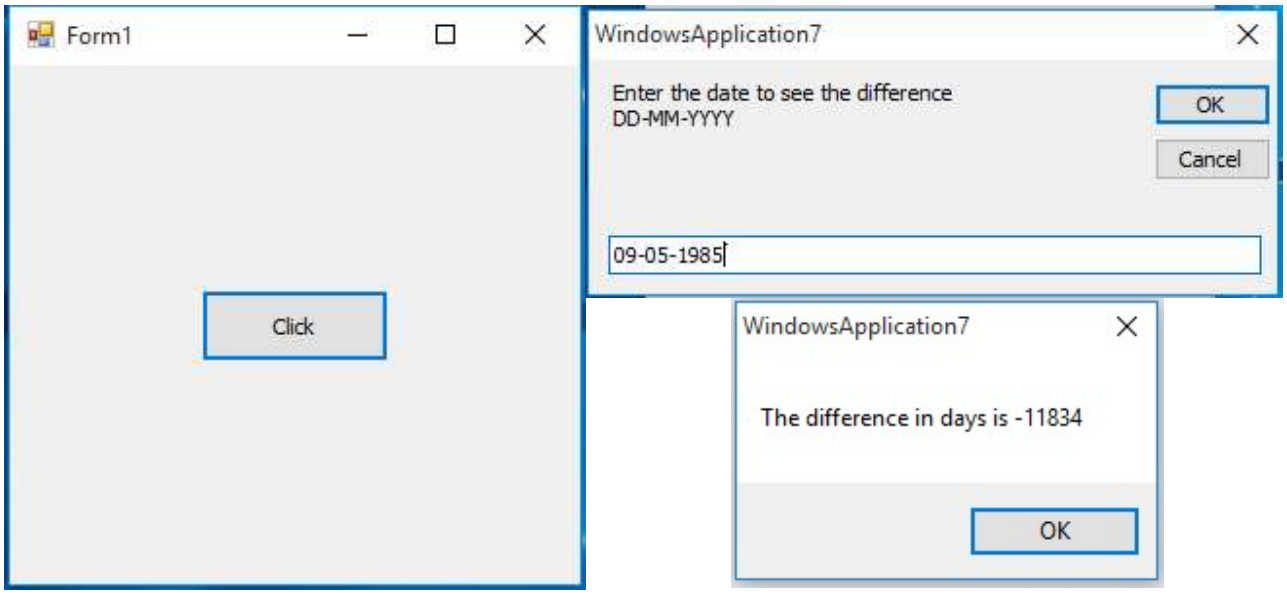
Property	Set
Text	Click

Source Code

Double click (DC) on Button, then write

```
Dim firstDate, msg As String
Dim secondDate As Date
firstDate = InputBox("Enter the date to see the difference" & vbNewLine & "DD-MM-YYYY")
secondDate = CDate(firstDatemsg = "The difference in days is " & DateDiff _
(DateInterval.Day, Now, secondDate)
MsgBox(msg)
```

Then run your application (app) to get:



fallahnajjar@yahoo.com
+9647807445833

Project 3: Check if file exists or not

Create text file and save it in C: and it name must be FileName.txt.

Add to your Form project

- Button

Button Properties

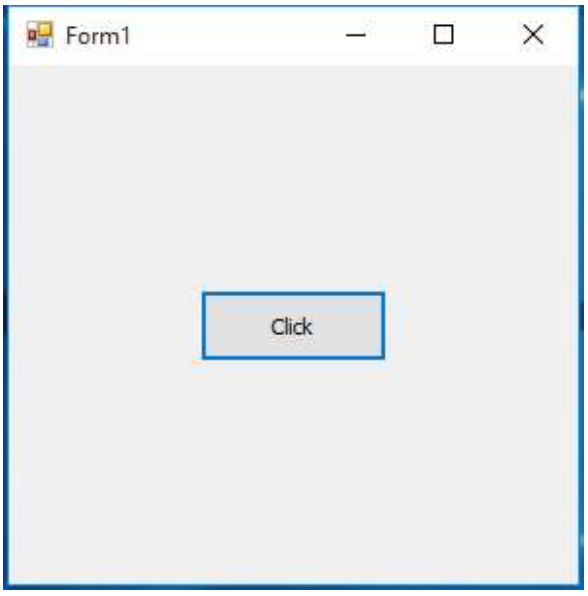
Property	Set
Text	Click

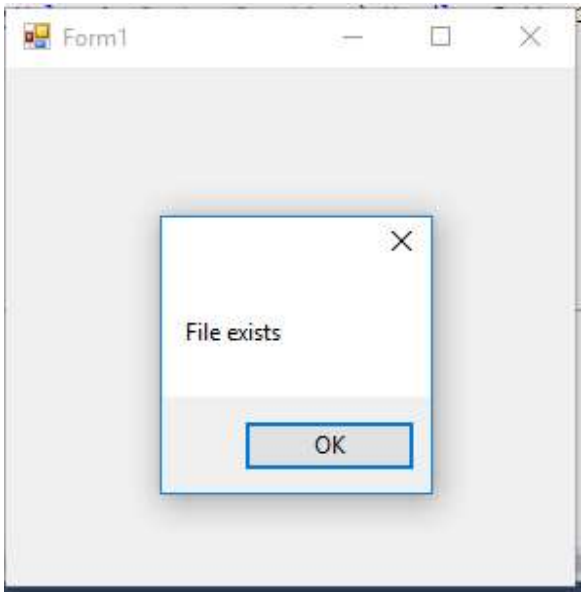
Source Code

Double click (DC) on Button, then write

```
If IO.File.Exists("C:\ NAJJAR.txt") = True Then
       MessageBox.Show("File exists")
Else
       MessageBox.Show("File does not exists")
End If
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 4: Create a file

Add to your Form project

- Button

Button Properties

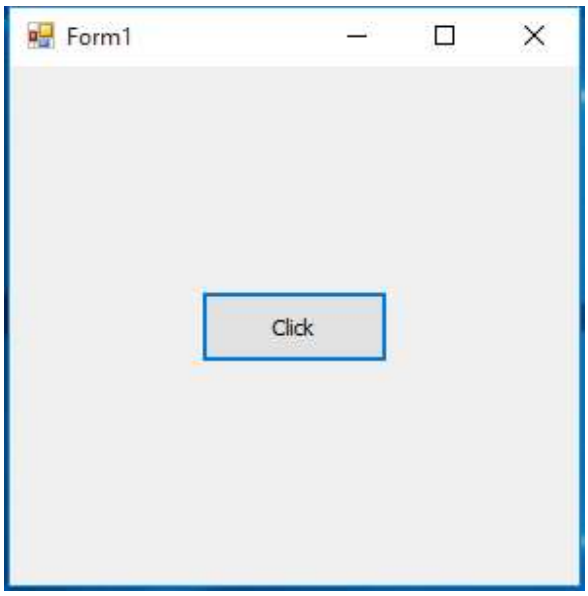
Property	Set
Text	Click

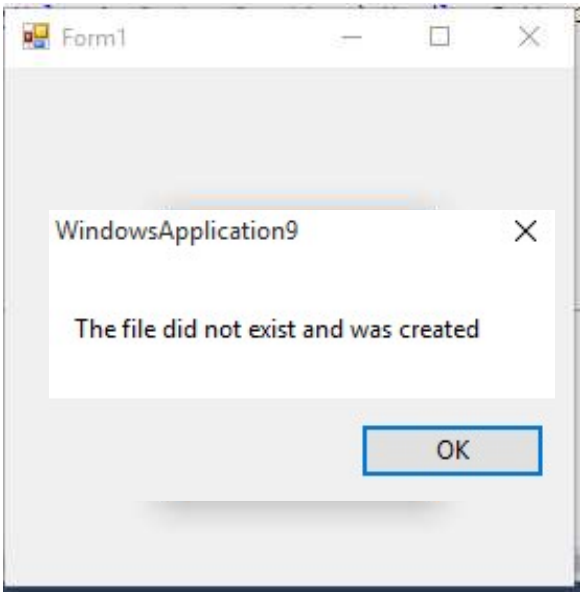
Source Code

Double click (DC) on Button, then write

```
If IO.File.Exists("F:\Najjar.txt") Then
MsgBox("file exists")
Return
End If
Dim sr As IO.StreamWriter = IO.File.CreateText("F:\ NAJJAR.txt")
sr.Close()
MsgBox("The file did not exist and was created")
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 5: Get file creation time

Add to your Form project

- Button

Button Properties

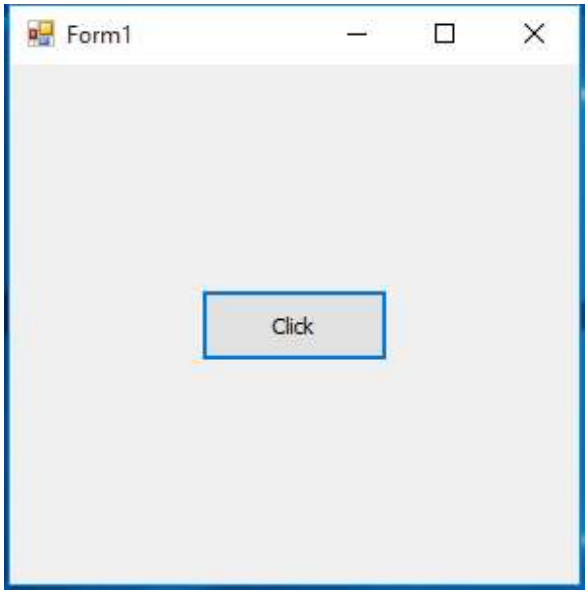
Property	Set
Text	Click

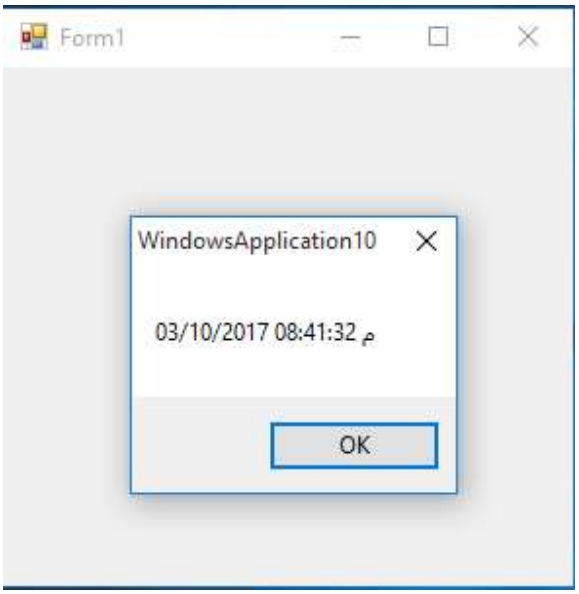
Source Code

Double click (DC) on Button, then write

```
MsgBox(IO.File.GetCreationTime("F:\ NAJJAR.txt"))
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 6: Get file last write time

Add to your Form project

- Button

Button Properties

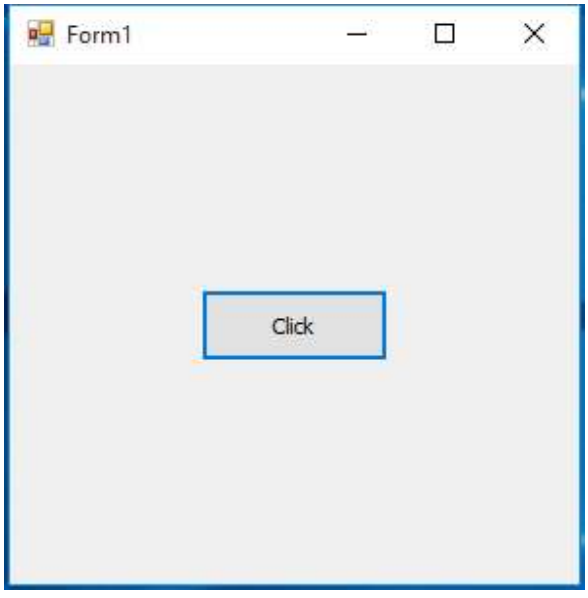
Property	Set
Text	Click

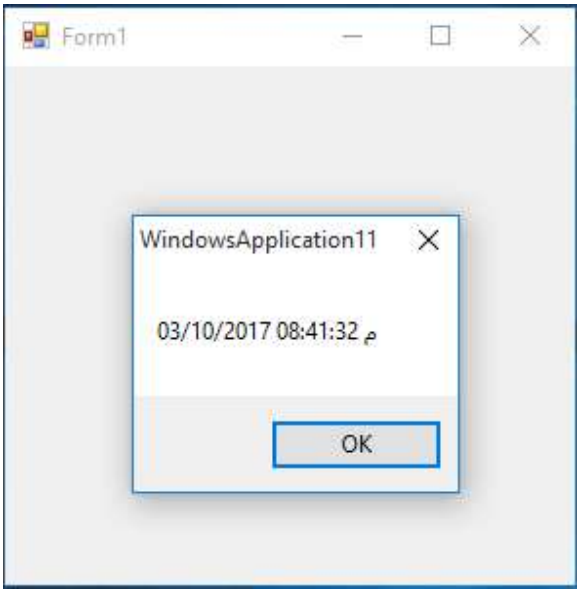
Source Code

Double click (DC) on Button, then write

```
MsgBox(IO.File.GetLastWriteTime("F:\ NAJJAR.txt"))
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 7: Get file extension

Add to your Form project

- Button

Button Properties

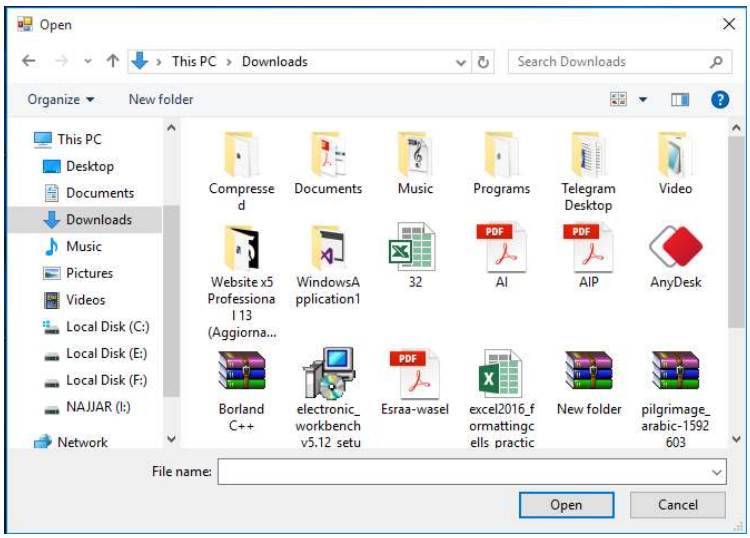
Property	Set
Text	Click

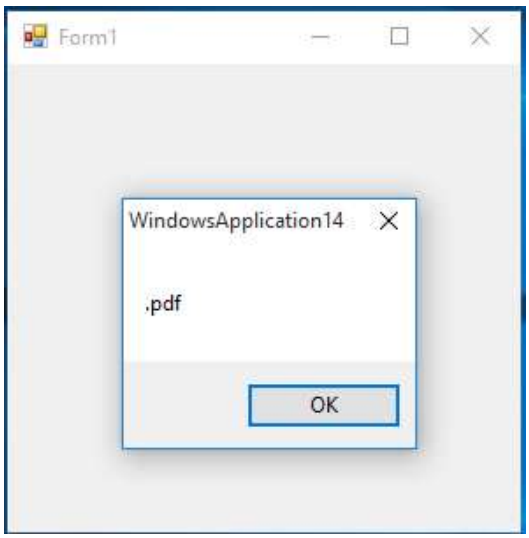
Source Code

Double click (DC) on Button, then write

```
Dim ofd As New OpenFileDialog
Dim MyFile As String
If ofd.ShowDialog = DialogResult.OK Then
MyFile = ofd.FileName
End If
Dim FI As FileInfo
FI = New FileInfo(MyFile)
MsgBox(FI.Extension)
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 8: Add text to the file

Add to your Form project

- Button
- TextBox

Button Properties

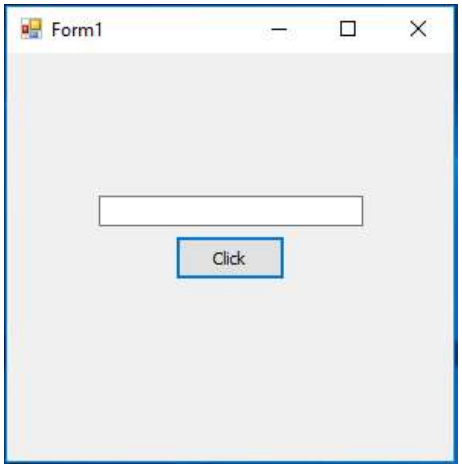
Property	Set
Text	Add Text

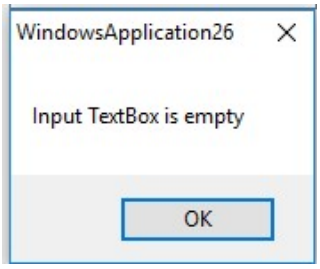
Source Code

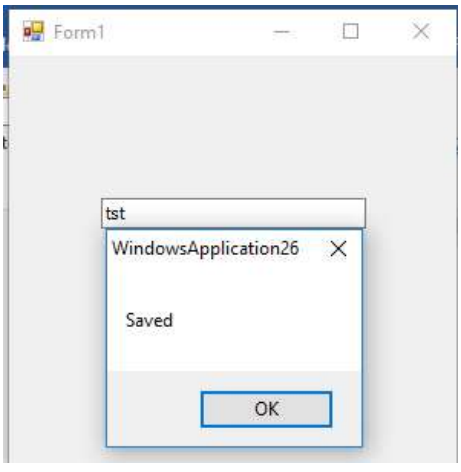
Double click (DC) on Button, then write

```
Dim Fs As IO.FileStream = New IO.FileStream("F:\ NAJJAR.txt",
IO.FileMode.Append, IO.FileAccess.Write, IO.FileShare.None)
If TextBox1.Text <> "" Then
Dim sw As New IO.StreamWriter(Fs)
sw.Write(TextBox1.Text & " ")
sw.WriteLine()
sw.Flush()
sw.Close()
Fs.Close()
MsgBox("Saved")
Else
MsgBox("Input TextBox is empty")
End If
```

Then run your application (app) to get:







You can try the following codes:

fallahnajjar@yahoo.com
+9647807445833

Double click (DC) on Button, then write

```
Kill("F:\ NAJJAR.txt")
```

```
SetAttr("F:\ NAJJAR.txt", FileAttribute.System)
```

```
File.SetAttributes("F:\FileName.txt", File.GetAttributes("F:\ NAJJAR.txt") _
Or FileAttributes.Hidden Or FileAttributes.System)
FileCopy("F:\ NAJJAR.txt", "F:\ Fallah.txt")
```

```
My.Computer.FileSystem.DeleteFile("F:\ NAJJAR.txt")
```

```
Dim sFileName As String = "F:\ NAJJAR.txt"
My.Computer.FileSystem.WriteAllText(sFileName, " لا إله إلا الله", False)
```

```
TextBox1.Text = My.Computer.FileSystem.ReadAllText("F:\ NAJJAR.txt")
Me.OpenFileDialog1.Filter = "TEXT FILES (*.TXT)|*.TXT"
Me.OpenFileDialog1.ShowDialog()
If Me.OpenFileDialog1.FileName <> "" Then
Me.TextBox1.Text = My.Computer.FileSystem.ReadAllText(Me.OpenFileDialog1.FileName)
End If
```


fallahnajjar@yahoo.com
+9647807445833

Project 9: Create Folder

Add to your Form project

- Button

Button Properties

Property	Set
Text	Click

Source Code

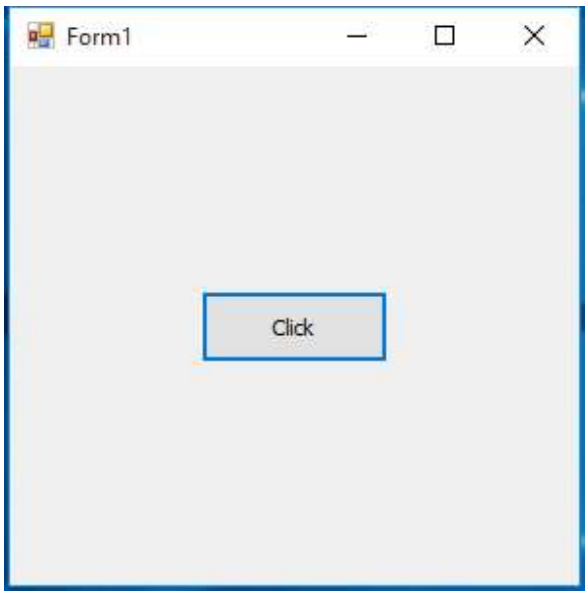
Double click (DC) on Button, then write

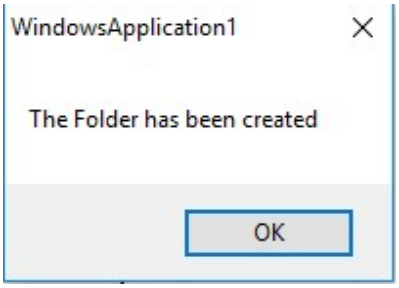
```
System.IO.Directory.CreateDirectory("F:\ NAJJAR ")
MsgBox("The Folder has been created")

Or

MkDir("F:\ NAJJAR ")
MsgBox("The Folder has been created")
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 10: Create & Delete Folder

Add to your Form project

- Tow Buttons

Buttons Properties

Tool	Property	Set
Button 1	Text	Create
Button 2	Text	Delete

Source Code

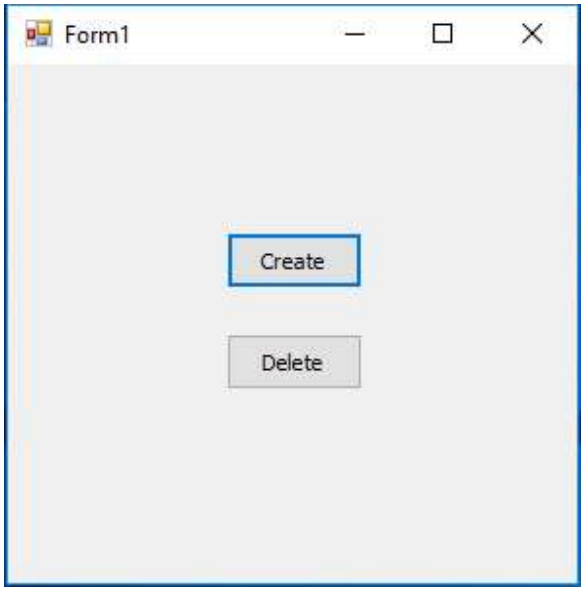
Double click (DC) on Button 1, then write

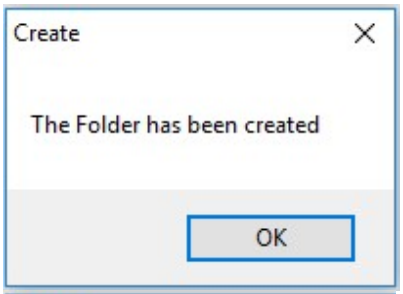
```
System.IO.Directory.CreateDirectory("F:\ NAJJAR ")
MsgBox("The Folder has been created", , "Create")
```

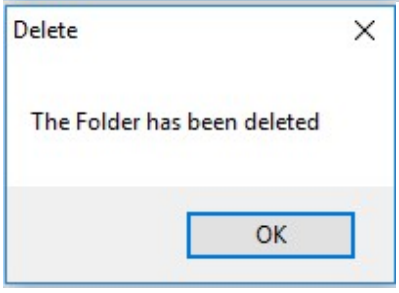
Double click (DC) on Button 2, then write

```
RmDir("F:\ NAJJAR ")
MsgBox("The Folder has been deleted", , "Delete")
```

Then run your application (app) to get:







fallahnajjar@yahoo.com
+9647807445833

Project 11: Folder Exists

Add to your Form project

- Button

Button Properties

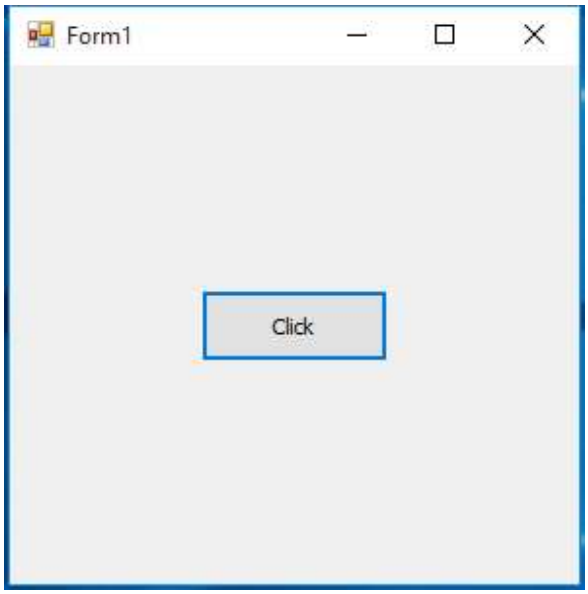
Property	Set
Text	Click

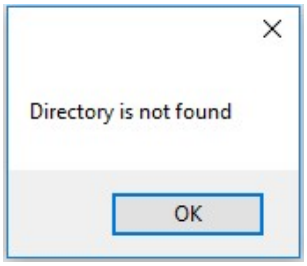
Source Code

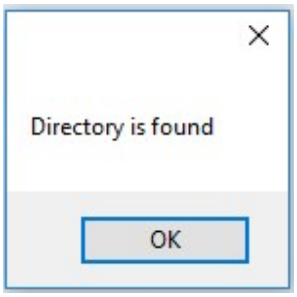
Double click (DC) on Button 1, then write

```
If My.Computer.FileSystem.DirectoryExists("F:\ NAJJAR ") = True Then
MessageBox.Show("Directory is found")
Else
MessageBox.Show("Directory is not found")
End If
```

Then run your application (app) to get:







fallahnajjar@yahoo.com
+9647807445833

Project 12: Folder Creation Time

Add to your Form project

- Button

Button Properties

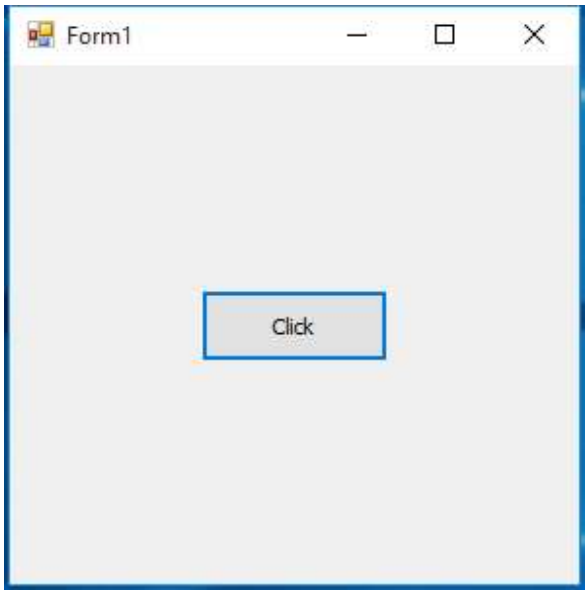
Property	Set
Text	Click

Source Code

Double click (DC) on Button 1, then write

```
MsgBox(System.IO.Directory.GetCreationTime("F:\ NAJJAR ").ToString, , "")
```

Then run your application (app) to get:



fallahnajjar@yahoo.com
+9647807445833

Project 13: Folder Last Access Time

Add to your Form project

- Button

Button Properties

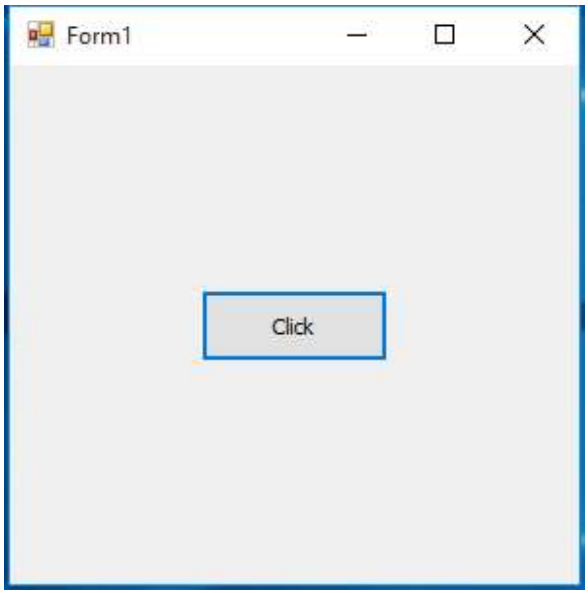
Property	Set
Text	Click

Source Code

Double click (DC) on Button 1, then write

```
MsgBox(System.IO.Directory.GetLastAccessTime("F:\NAJJAR").ToString, , "")
```

Then run your application (app) to get:



fallahnajjar@yahoo.com
+9647807445833

Project 14: Folder Size

Add to your Form project

- ❖ Button

Property	Set
Text	Click

Button Properties

Source Code

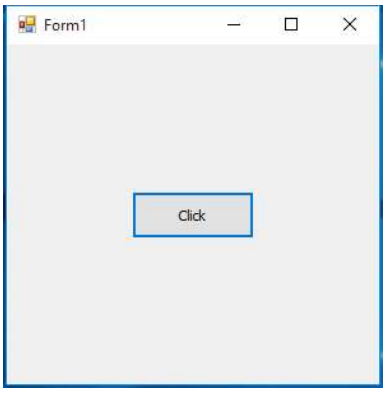
Build the following function

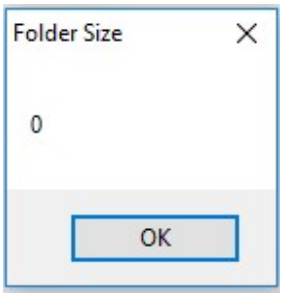
```
Function GetFolderSize(ByVal DirPath As String, Optional _
ByVal IncludeSubFolders As Boolean = True) As Long
Dim lngDirSize As Long
Dim objFileInfo As IO.FileInfo
Dim objDir As IO.DirectoryInfo = New IO.DirectoryInfo(DirPath)
Dim objSubFolder As IO.DirectoryInfo
Try
For Each objFileInfo In objDir.GetFiles()
lngDirSize += objFileInfo.Length
Next
If IncludeSubFolders Then
For Each objSubFolder In objDir.GetDirectories()
lngDirSize += GetFolderSize _
(objSubFolder.FullName)
Next
End If
Catch Ex As Exception
End Try
Return lngDirSize
End Function
```

Double click (DC) on Button 1, then write

```
MsgBox(GetFolderSize("F:\NAJJAR"))
```

Then run your application (app) to get:





fallahnajjar@yahoo.com
+9647807445833

Project 15: Folder Explorer

Add to your Form project

- Button

Button Properties

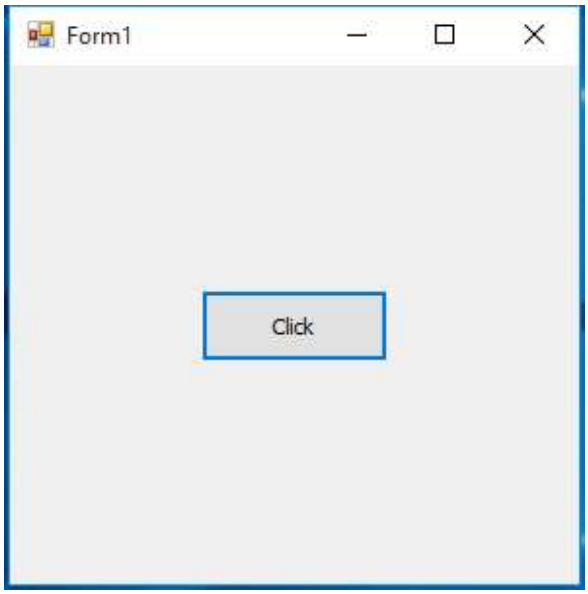
Property	Set
Text	Click

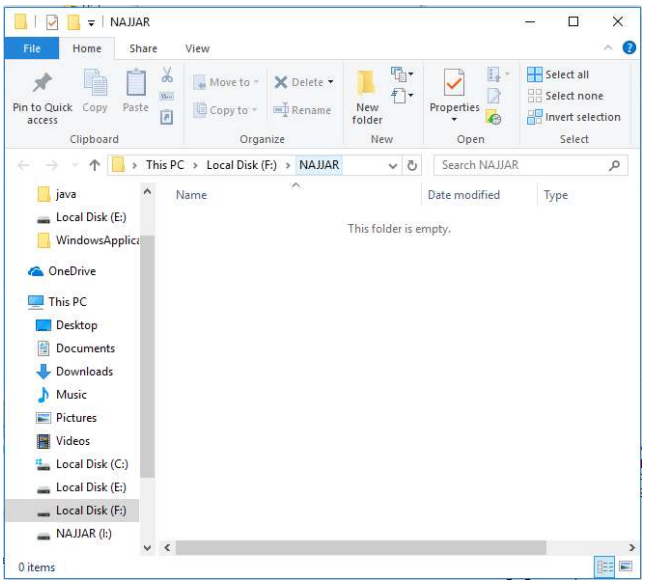
Source Code

Double click (DC) on Button 1, then write

```
Process.Start("explorer.exe", "F:\NAJJAR")
```

Then run your application (app) to get:





You can try the following codes:

fallahnajjar@yahoo.com
+9647807445833

Double click (DC) on Button, then write

```
MsgBox(System.IO.Directory.GetLastWriteTime("F:\NAJJAR").ToString)
```

```
My.Computer.FileSystem.DeleteDirectory("C:\NAJJAR ", _
 FileIO.DeleteDirectoryOption.DeleteAllContents)
```

```
My.Computer.FileSystem.DeleteDirectory("F:\NAJJAR", FileIO.UIOption.AllDialogs,_
FileIO.RecycleOption.SendToRecycleBin)
```

```
MsgBox(IO.Directory.GetDirectories("F:\NAJJAR").Length)
```

```
System.IO.Directory.Move("C:\NAJJAR", "C:\Fallah")
```


fallahnajjar@yahoo.com
+9647807445833

Project 16: Extract File Associated Icon

Add to your Form project

- Button

Button Properties

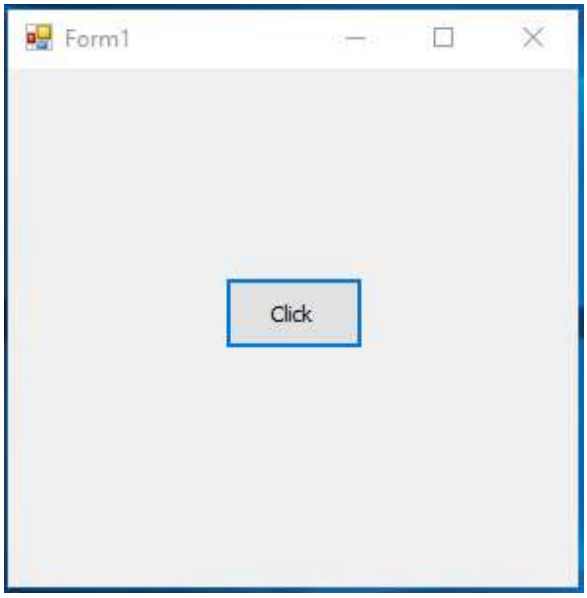
Property	Set
Text	Click

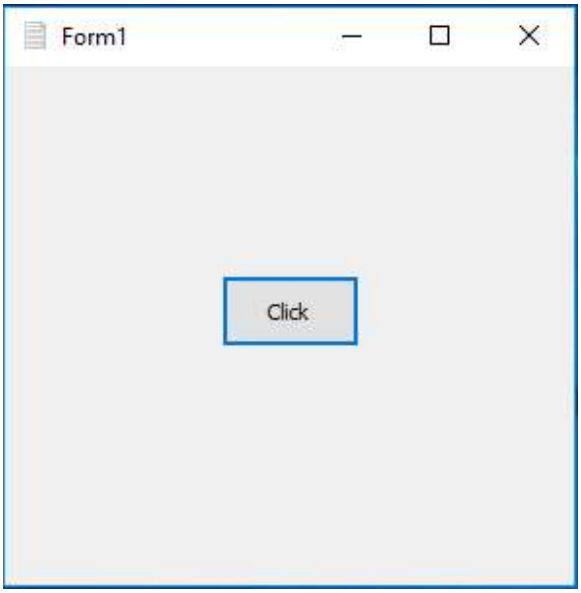
Source Code

Double click (DC) on Button 1, then write

```
Dim a As Icon
a = Icon.ExtractAssociatedIcon(Application.StartupPath & "\NAJJAR.txt")
Me.Icon = a
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 17: Set Image to PictureBox Tool

Add to your Form project

- ❖ PictureBox

Source Code

Double click (DC) on the form, then write

```
PictureBox1.Image = Image.FromFile(Application.StartupPath & "\Name.png")
```

Then run your application (app) to get:

fallahnajjar@yahoo.com
+9647807445833

Project 18: Select image & Set it to PictureBox Tool

Add to your Form project

- Button
- PictureBox

Button Properties

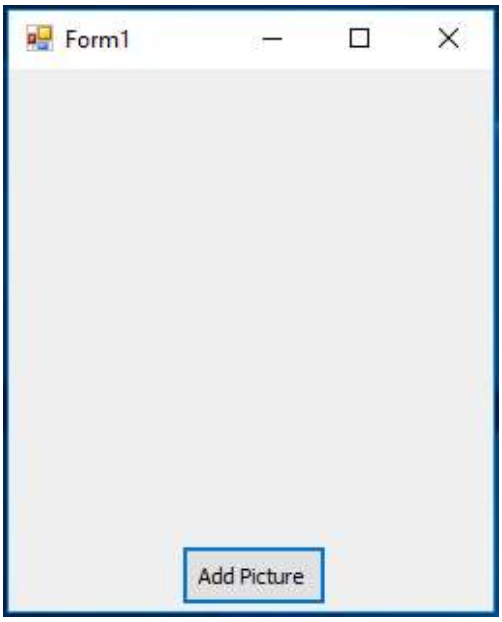
Property	Set
Text	Add Picture

Source Code

Double click (DC) on Button 1, then write

```
Dim OpenFileDialog1 As New OpenFileDialog
With OpenFileDialog1
.CheckFileExists = True
.ShowReadOnly = False
.Filter = "All Files|*.*|Bitmap Files (*)|*.png;*.bmp;*.gif;*.jpg"
.FilterIndex = 2
If .ShowDialog = DialogResult.OK Then
PictureBox1.Image = Image.FromFile(.FileName)
End If
End With
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 19: Rotate Image

Add to your Form project

- Button
- PictureBox

Tool	Property	Set
Button	Text	Rotate
PictureBox	Image	Set Any Image
PictureBox	SizeMode	StretchImage

Source Code

Double click (DC) on Button 1, then write

```
PictureBox1.Image.RotateFlip(RotateFlipType.RotateNoneFlipXY)
PictureBox1.Refresh()
```

Then run your application (app) to get

fallahnajjar@yahoo.com
+9647807445833

Project 20: Set image to PictureBox in other Form

Add 2 Forms to your project

Add to Form1

- ❖ Button
- ❖ PictureBox

Properties

Tool	Property	Set
Button	Text	Set Picture
PictureBox	Image	Set Any Image
PictureBox	SizeMode	StretchImage

Add to Form2

- ❖ PictureBox

PictureBox Properties

Property	Set
SizeMode	StretchImage

Source Code

Double click (DC) on Button1 in Form1, then write

```
Dim Form2 As New Form2
Form2.PictureBox1.Image = Me.PictureBox1.Image
Form2.ShowDialog()
```

Then run your application (app) to get:

fallahnajjar@yahoo.com
+9647807445833

Project 21: Get Color Complement

Add to your Form project

- ❖ Button
- ❖ PictureBox

Button Properties

Tool	Property	Set
Button	Text	Go
PictureBox	Image	Set Any Image

Source Code

Double click (DC) on Button 1, then write

```
Dim bmap As New Bitmap(PictureBox1.Image)
PictureBox1.Image = bmap
Dim red, green, blue As Integer
Dim i, j As Integer
With bmap
For i = 1 To .Height - 2
For j = 1 To .Width - 2
red = CInt(.GetPixel(j, i).R)
blue = CInt(.GetPixel(j, i).B)
green = CInt(.GetPixel(j, i).G)
bmap.SetPixel(j, i, Color.FromArgb(255 - red, 255 - green, 255 - blue))
Next
PictureBox1.Refresh()
Next
PictureBox1.Refresh()
End With
```

Then run your application (app) to get

fallahnajjar@yahoo.com
+9647807445833

Project 22: Add a dark layer over an image

Add to your Form project

- Button
- PictureBox

Button Properties

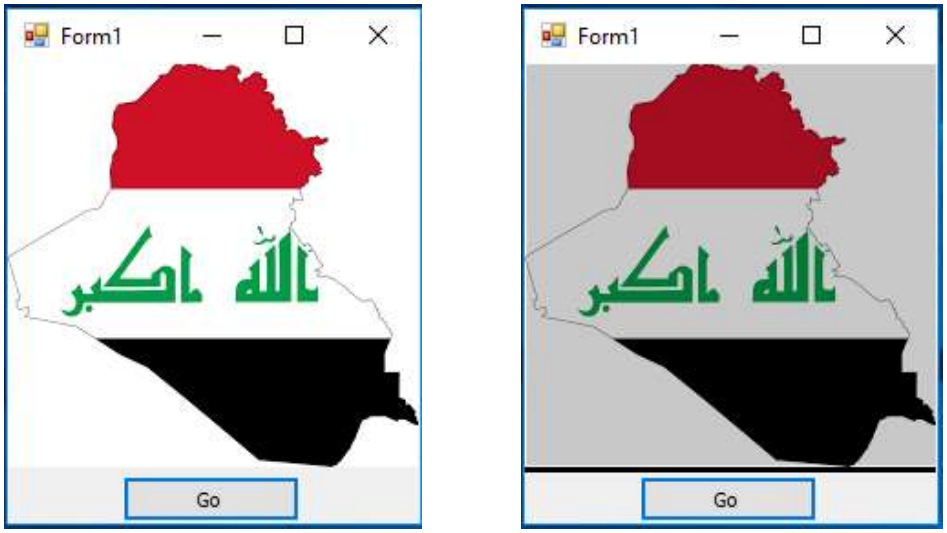
Tool	Property	Set
Button	Text	Go
PictureBox	Image	Set Any Image

Source Code

Double click (DC) on Button 1, then write

```
PictureBox1.BackColor = Color.Black
Dim bmap = New Bitmap(PictureBox1.Image)
PictureBox1.Image = bmap
Dim tempbmp As New Bitmap(PictureBox1.Image)
Dim red, green, blue As Integer
Dim i, j As Integer
With tempbmp
For i = 1 To .Height - 2
For j = 1 To .Width - 2
red = CInt(.GetPixel(j, i).R)
green = CInt(.GetPixel(j, i).G)
blue = CInt(.GetPixel(j, i).B)
bmap.SetPixel(j, i, Color.FromArgb(200, red, green, blue))
Next
PictureBox1.Refresh()
Next
End With
PictureBox1.Refresh()
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

You can try the following codes:

Try to Use the Following Code instead the Code that like it in Project 22

```
bmap.SetPixel(j, i, Color.FromArgb(50, red, green, blue))
```

```
PictureBox1.Image.RotateFlip(RotateFlipType.RotateNoneFlipY)
PictureBox1.Refresh()
```

```
PictureBox1.Image.RotateFlip(RotateFlipType.RotateNoneFlipX)
PictureBox1.Refresh()
```

```
Dim x As Bitmap = PictureBox1.Image
x.RotateFlip(RotateFlipType.Rotate180FlipY)
PictureBox1.Image = x
```


fallahnajjar@yahoo.com
+9647807445833

Project 23: Add New Button Control

Add to your Form project

- Button

Button Properties

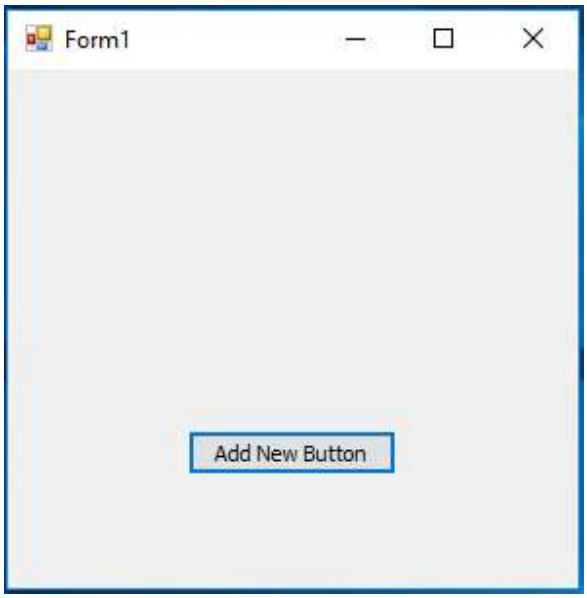
Property	Set
Text	Add New Button

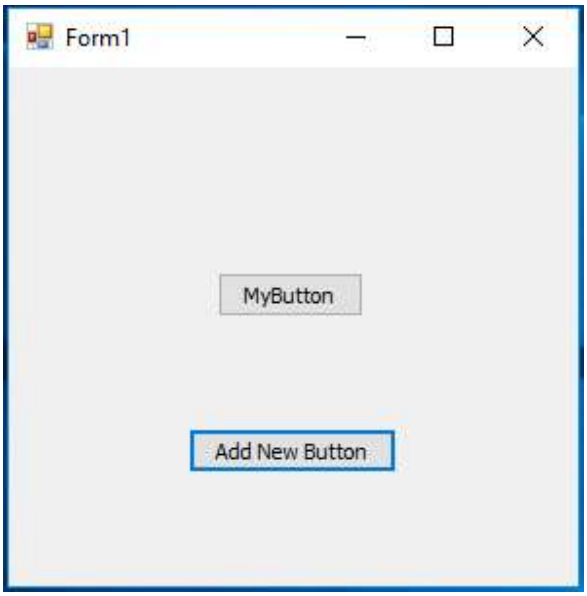
Source Code

Double click (DC) on Button 1, then write

```
Dim MyButton As New Button
MyButton.Location = New Point(107, 105)
MyButton.Text = "MyButton"
Me.Controls.Add(MyButton)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 24: Change Button Shape

Add to your Form project

- Button

Button Properties

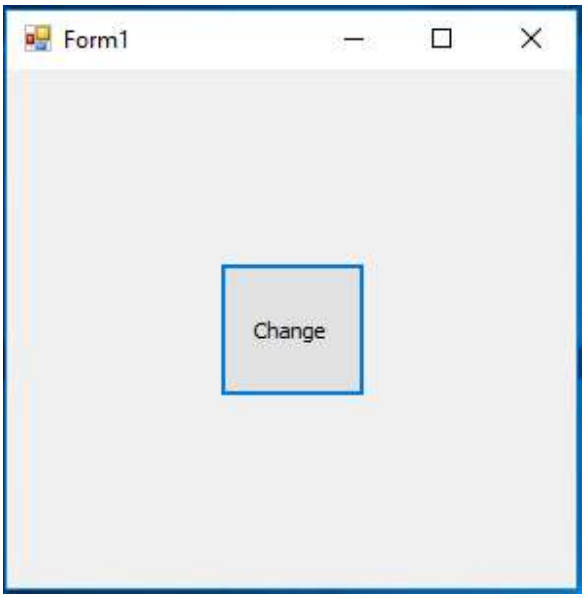
Property	Set
Text	Change

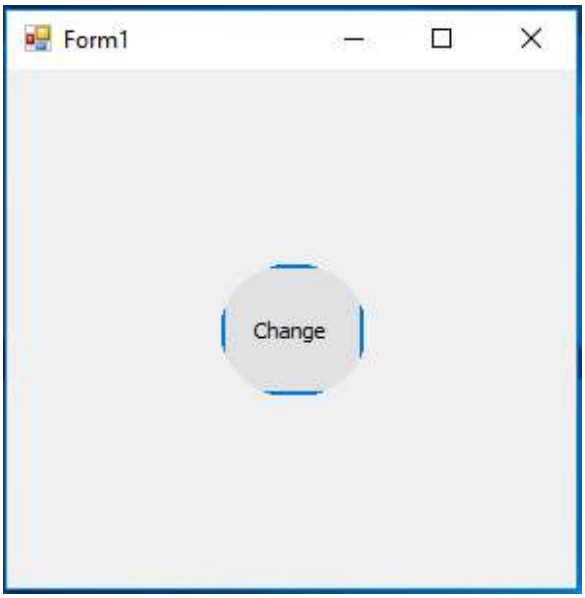
Source Code

Double click (DC) on Button 1, then write

```
Dim myGraphicsPath As New System.Drawing.
Drawing2D.GraphicsPath()
myGraphicsPath.AddEllipse(New RectangleF(0, 0, Me.Button1.Width, Me.Button1.Height))
Button1.Region = New Region(myGraphicsPath)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 25: Change Button Size

Add to your Form project

- Button

Button Properties

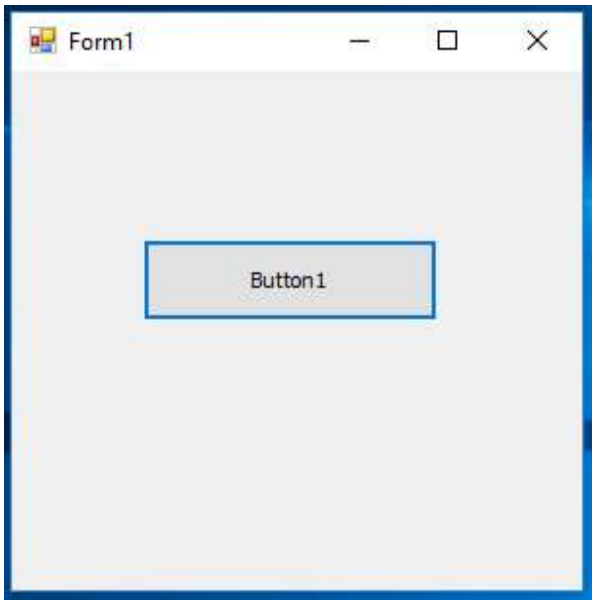
Property	Set
Size	151; 42

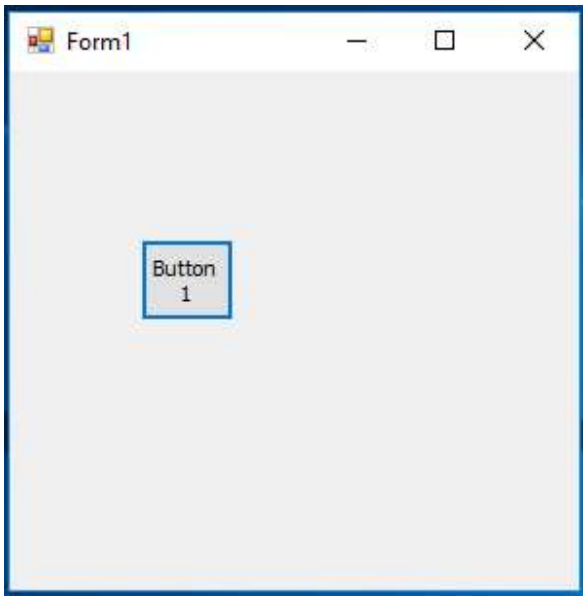
Source Code

Double click (DC) on Button 1, then write

```
Dim surface As Graphics = Button1.CreateGraphics
Dim textSize As SizeF = surface.MeasureString("  " & Button1.Text & "  ", Button1.Font)
surface.Dispose()
Button1.Width = CInt(textSize.Width)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 26: Write on Button

Add to your Form project

- Button

Button Properties

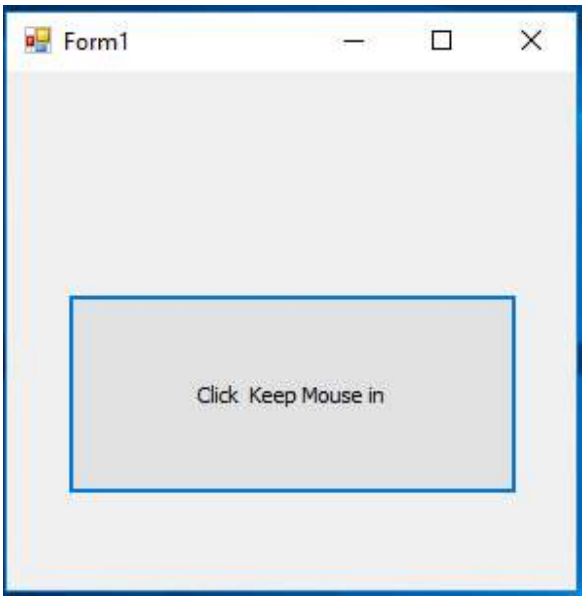
Property	Set
Text	Click & Keep Mouse in

Source Code

Double click (DC) on Button 1, then write

```
Dim g As Graphics = Button1.CreateGraphics
Dim mybrush As New Drawing2D.LinearGradientBrush _
(ClientRectangle, Color.Red, Color.Yellow, Drawing2D.LinearGradientMode.Horizontal)
Dim myFont As New Font("Times New Roman", 24)
g.DrawString("", myFont, mybrush, New RectangleF(10, 10, 100, 200))
g.DrawString("اشهد ان علياً ولي الله", myFont, mybrush, 10, 10)
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 27: Create New TextBox

Add to your Form project

- Button

Button Properties

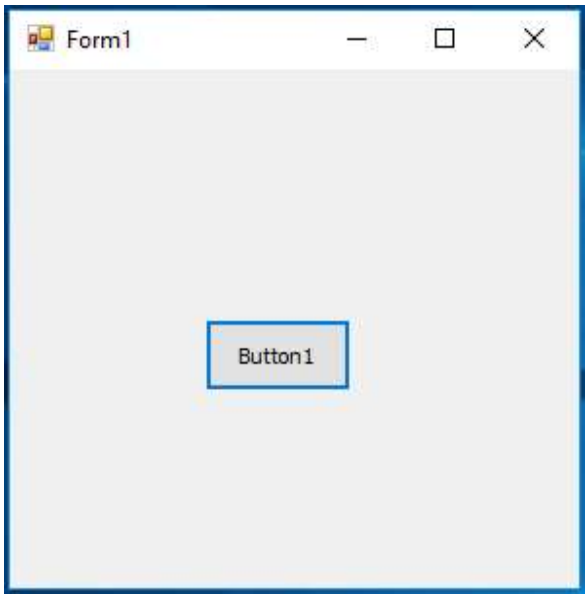
Property	Set
Text	Create New TextBox

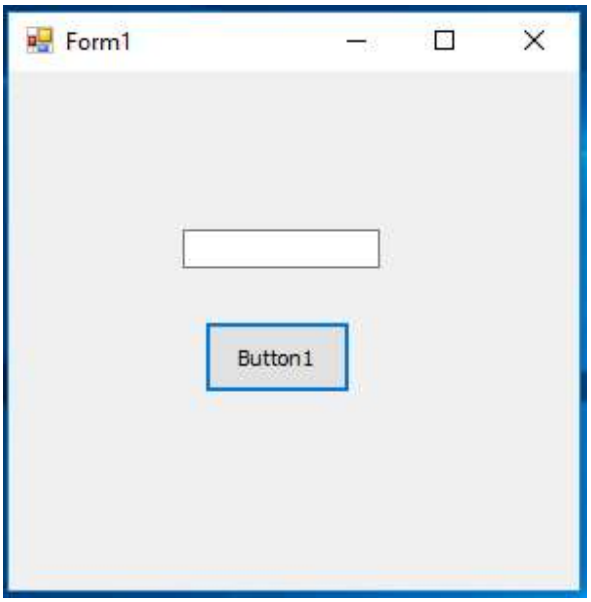
Source Code

Double click (DC) on Button 1, then write

```
Dim textBox1 As New TextBox()
textBox1.Size = New Size(101, 9)
textBox1.Location = New Point(89, 81)
Me.Controls.Add(textBox1)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 28: Write with Arabic Language only

Add to your Form project

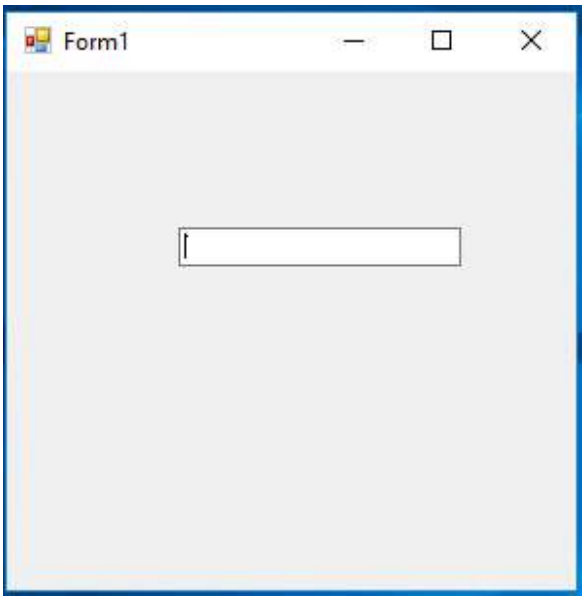
- TextBox

Source Code

View Code and change the method to TextBox1_KeyPress, then write

```
Select Case e.KeyChar
Case "ء" To "ي", ControlChars.Back, Chr(Keys.Space)
e.Handled = False
Case Else
e.Handled = True
End Select
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 29: Write with English Language only

Add to your Form project

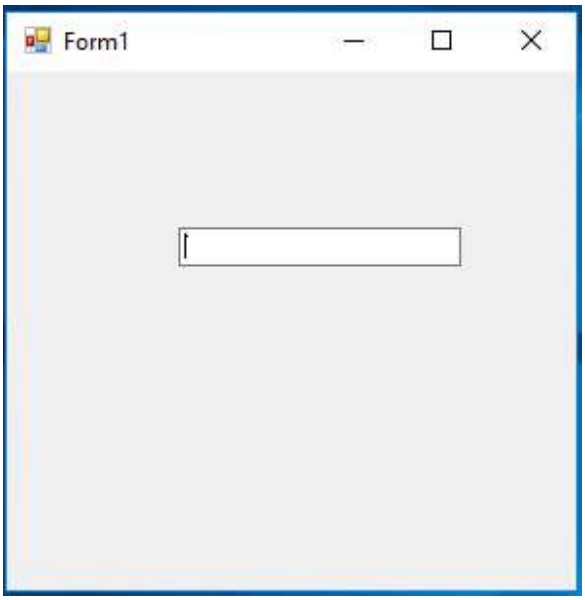
- TextBox

Source Code

View Code and change the method to TextBox1_KeyPress, then write

```
Select Case e.KeyChar
Case "a" To "z", ControlChars.Back, Chr(Keys.Space)
e.Handled = False
Case "A" To "Z", ControlChars.Back, Chr(Keys.Space)
e.Handled = False
Case Else
e.Handled = True
End Select
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 30: Write with Numbers Only

Add to your Form project

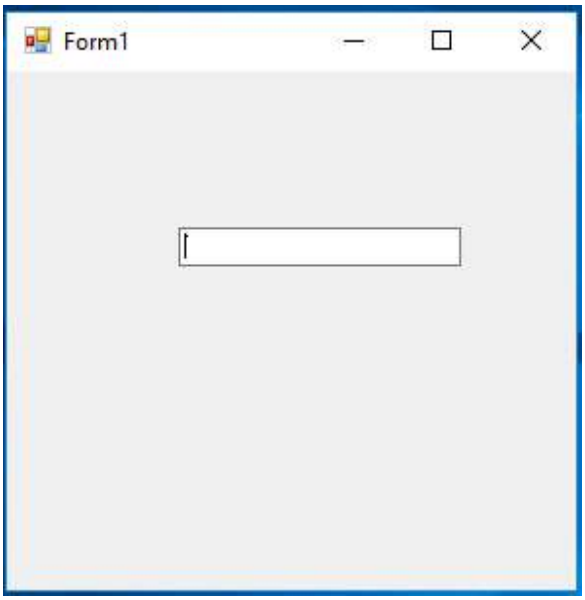
- TextBox

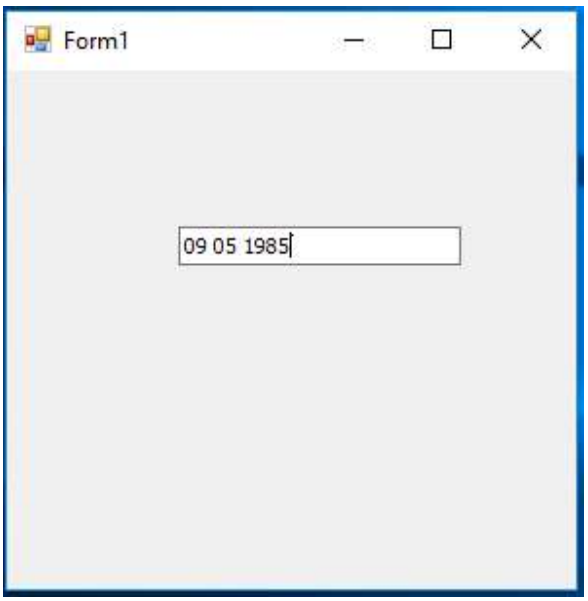
Source Code

View Code and change the method to TextBox1_KeyPress, then write

```
Select Case e.KeyChar
Case "0" To "9", ControlChars.Back, Chr(Keys.Space)
e.Handled = False
Case Else
e.Handled = True
End Select
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 31: Number of Words in TextBox

Add to your Form project

- Button
- TextBox

Button Properties

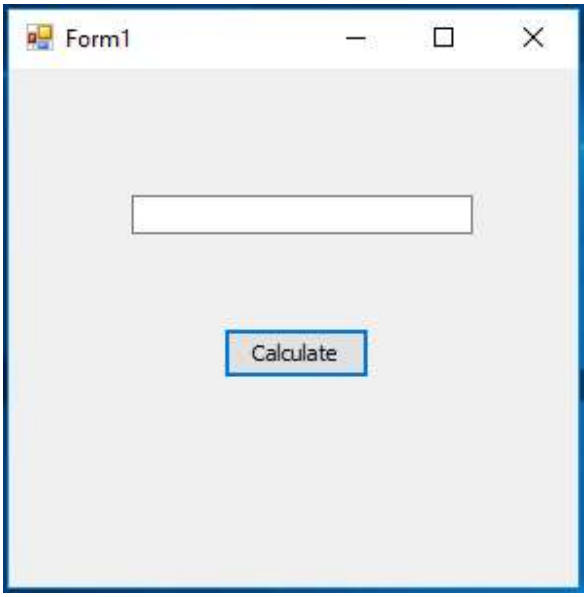
Property	Set
Text	Calculate

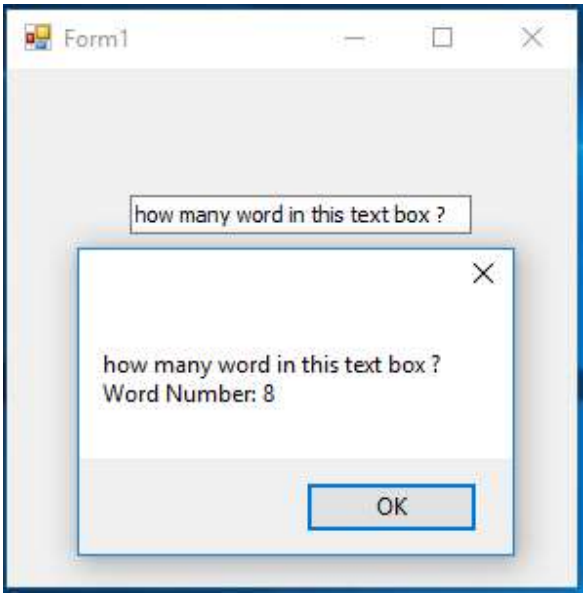
Source Code

Double click (DC) on Button 1, then write

```
Dim str As String = TextBox1.Text
If str = "" Then
Else
Dim wordCount As Integer = Strings.Split(str, Strings.Space(1)).Length
MessageBox.Show((str + Constants.vbNewLine & "Word Number: ") + wordCount.ToString)
End If
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 32: Check if the String Exist or Not in TextBox

Add to your Form project

- Button
- TextBox

Button Properties

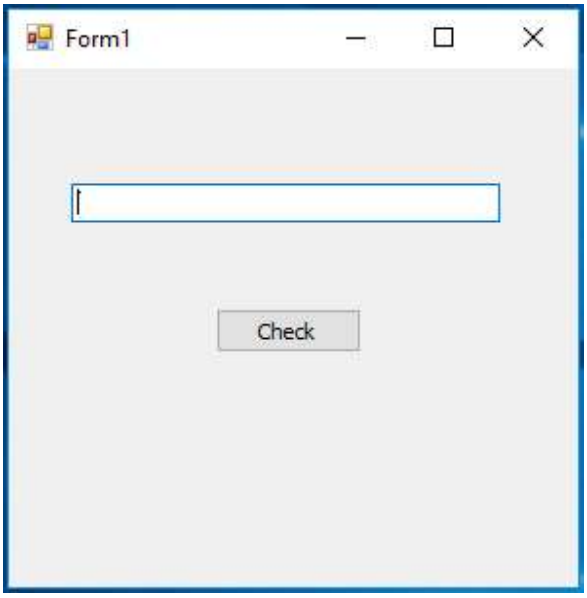
Property	Set
Text	Check

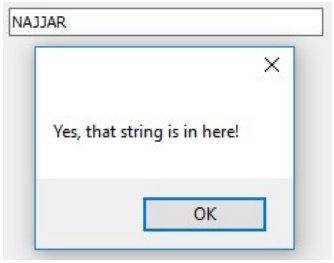
Source Code

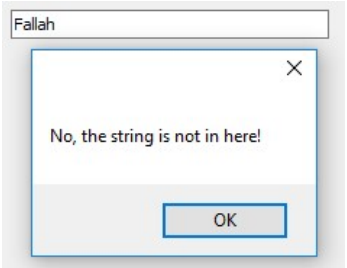
Double click (DC) on Button 1, then write

```
Dim str As String = TextBox1.Text
Dim i As Integer = str.IndexOf("NAJJAR")
If Not i = -1 Then
MessageBox.Show("Yes, that string is here!")
Else
MessageBox.Show("No, the string is not here!")
End If
```

Then run your application (app) to get







fallahnajjar@yahoo.com
+9647807445833

Project 33: Clear Textboxes

Add to your Form project

- Button
- 5 TextBox

Button Properties

Property	Set
Text	Clear All

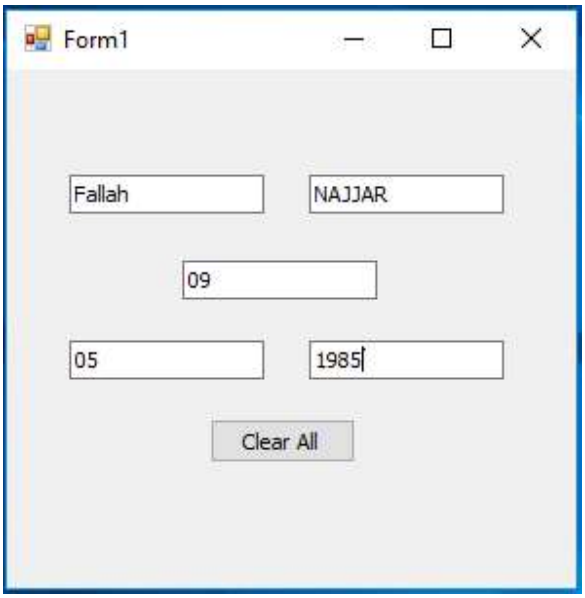
Source Code // Sub procedure

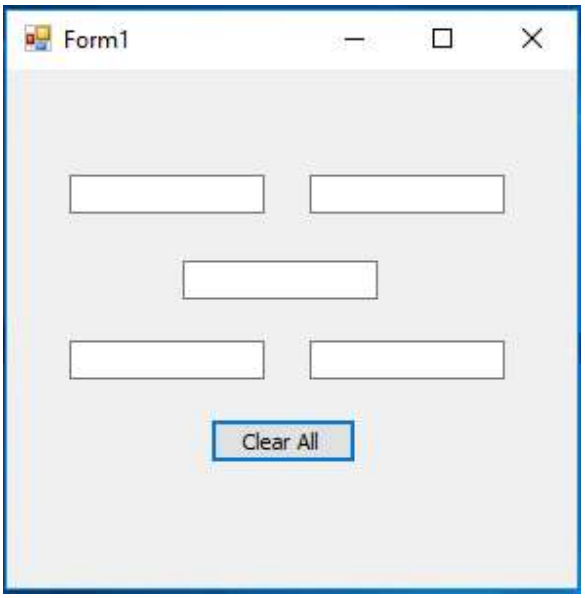
```
Public Sub ClearTextBox(ByVal Root As Control)
        For Each Ct As Control In Root.Controls
            If TypeOf Ct Is TextBox Then
                CType(Ct, TextBox).Clear()
            End If
        Next
End Sub
```

Double click (DC) on Button 1, then write

```
ClearTextBox(Me)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 34: TextBox Undo

Add to your Form project

- Button
- TextBox

Button Properties

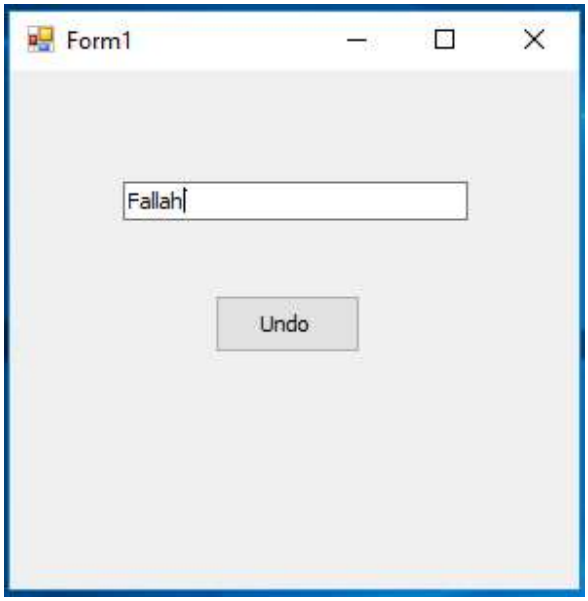
Property	Set
Text	Undo

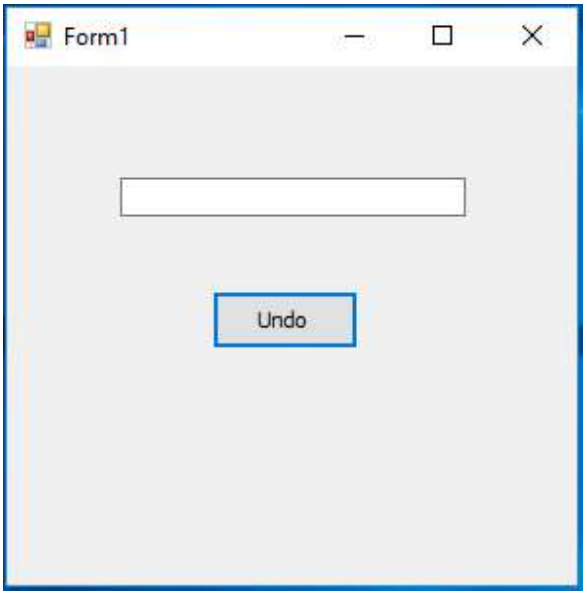
Source Code

Double click (DC) on Button 1, then write

```
If TextBox1.CanUndo Then TextBox1.Undo() End If
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 35: Create ComboBox

Add to your Form project

- Button

Button Properties

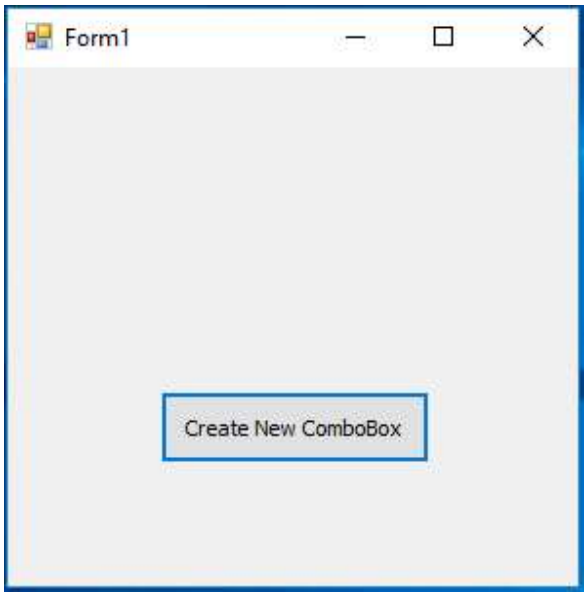
Property	Set
Text	Create New ComboBox

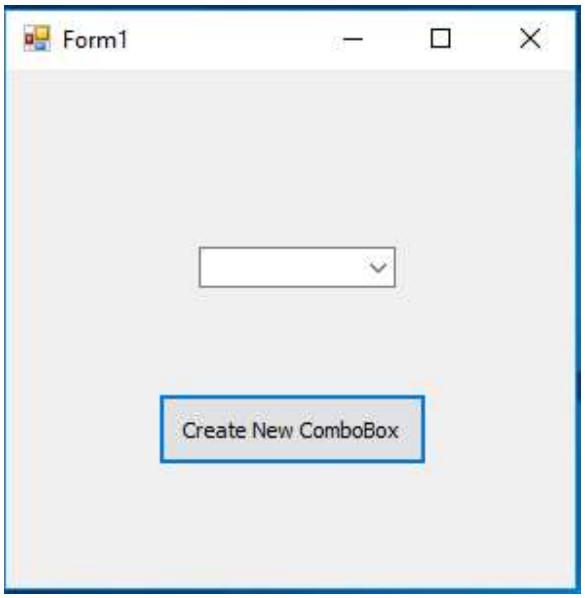
Source Code

Double click (DC) on Button 1, then write

```
Dim ComboBox1 As New ComboBox()
ComboBox1.Size = New Size(101, 9)
ComboBox1.Location = New Point(89, 81)
Me.Controls.Add(ComboBox1)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 36: Change ComboBox Style

Add to your Form project

- ComboBox
- 3 Button

Properties

Tool	Property	Set
Button1	Text	Drop Down List
Button2	Text	Drop Down
Button3	Text	Simple

Source Code

Double click (DC) on Button 1, then write

```
ComboBox1.DropDownStyle = ComboBoxStyle.DropDownList
```

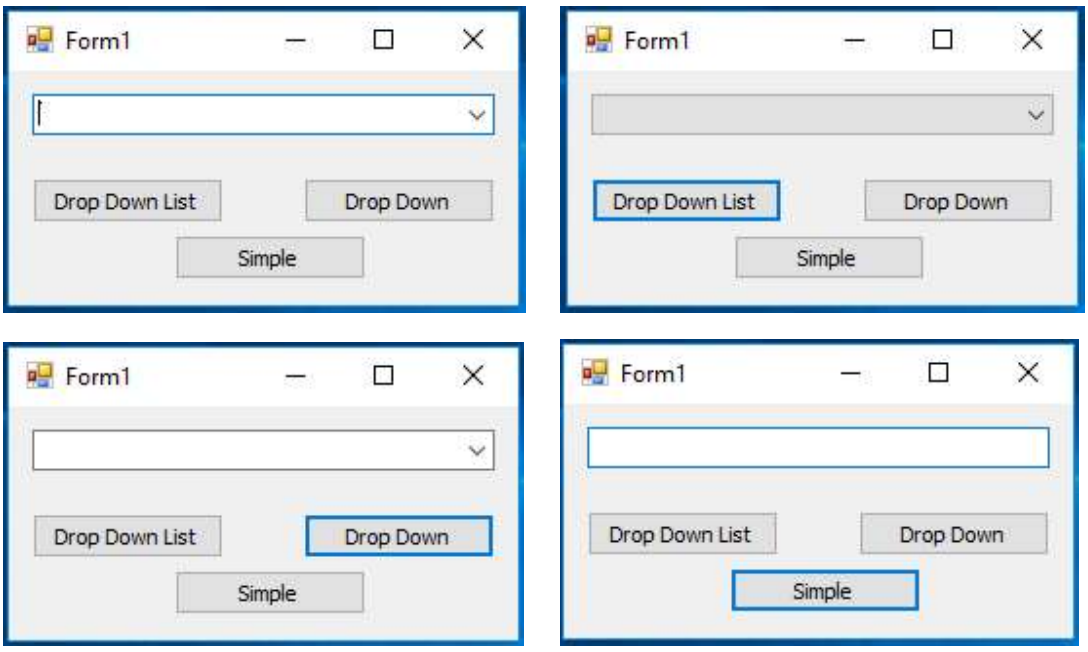
Double click (DC) on Button 1, then write

```
ComboBox1.DropDownStyle = ComboBoxStyle.DropDown
```

Double click (DC) on Button 1, then write

```
ComboBox1.DropDownStyle = ComboBoxStyle.Simple
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 37: Add Items to ComboBox

Add to your Form project

- ComboBox
- Button

Button Properties

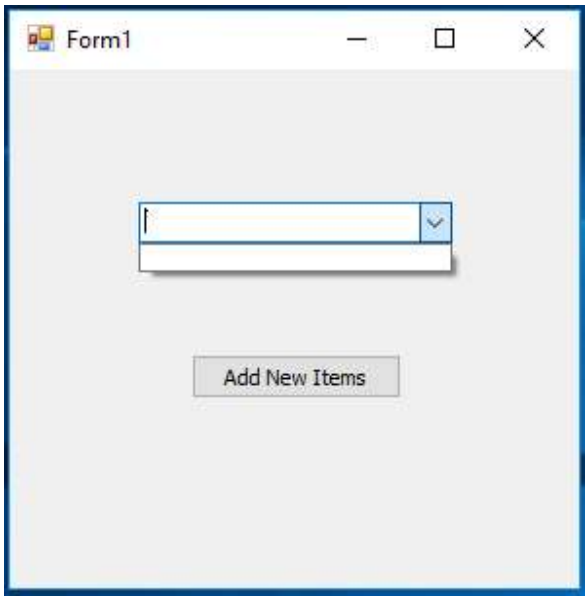
Property	Set
Text	Add New Items

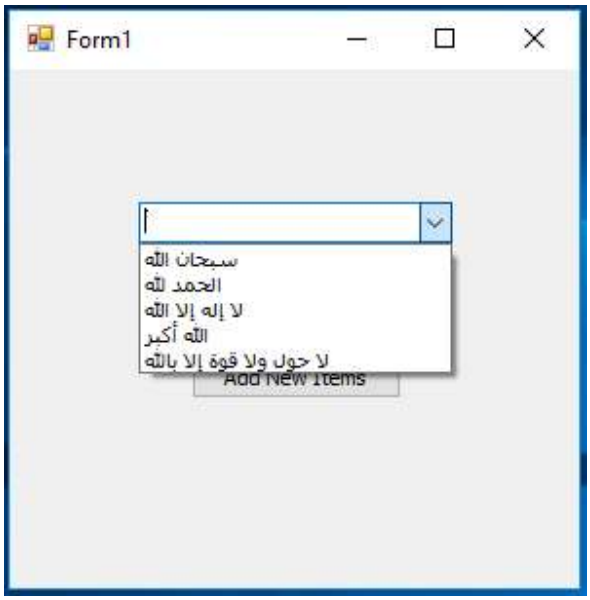
Source Code

Double click (DC) on Button 1, then write

```
ComboBox1.Items.Add("سبحان الله")
ComboBox1.Items.Add("الحمد لله")
ComboBox1.Items.Add("لا إله إلا الله")
ComboBox1.Items.Add("الله أكبر")
ComboBox1.Items.Add("لا حول ولا قوة إلا بالله")
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 38: Read / Save file to / from ComboBox

Add to your Form project

- ComboBox
- 2 Button

Tool	Property	Set
Button1	Text	Save
Button2	Text	Open

Source Code

Double click (DC) on Button 1, then write

```
Dim save As New SaveFileDialog
save.Filter = "Text File|*.txt"
If save.ShowDialog =
Windows.Forms.DialogResult.OK Then
End If
Try
FileOpen(1, save.FileName, OpenMode.Append)
For Each x In ComboBox1.Items
PrintLine(1, x)
Next
Catch
End Try
```

Double click (DC) on Button 2, then write

```
Dim openFile As New OpenFileDialog
openFile.Filter = "Text File|*.txt"
If openFile.ShowDialog = Windows.Forms. DialogResult.OK Then
Dim StreamR As New IO.StreamReader (openFile.FileName)
Dim x As String = "Fallah NAJJAR"
ComboBox1.Items.Clear()
Do
x = StreamR.ReadLine
If x <> "" Then
ComboBox1.Items.Add(x)
Else
Exit Do
End If
Loop
End If
```

You can find what are the results …

fallahnajjar@yahoo.com
+9647807445833

Project 39: Create ListBox

Add to your Form project

- Button

Button Properties

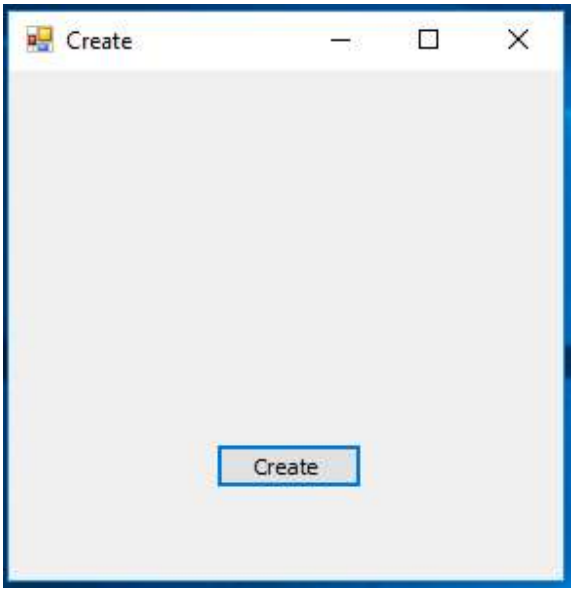
Property	Set
Text	Create

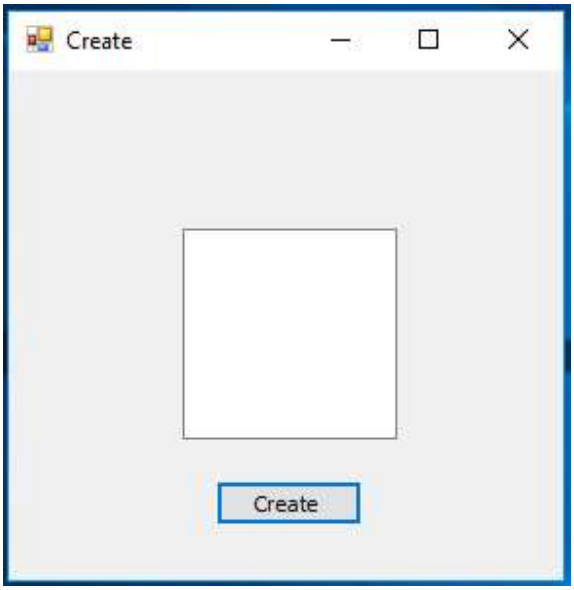
Source Code

Double click (DC) on Button 1, then write

```
Dim ListBox1 As New ListBox()
ListBox1.Size = New Size(110, 110)
ListBox1.Location = New Point(89, 81)
Me.Controls.Add(ListBox1)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 40: Add Items to ListBox

Add to your Form project

- ListBox
- Button

Button Properties

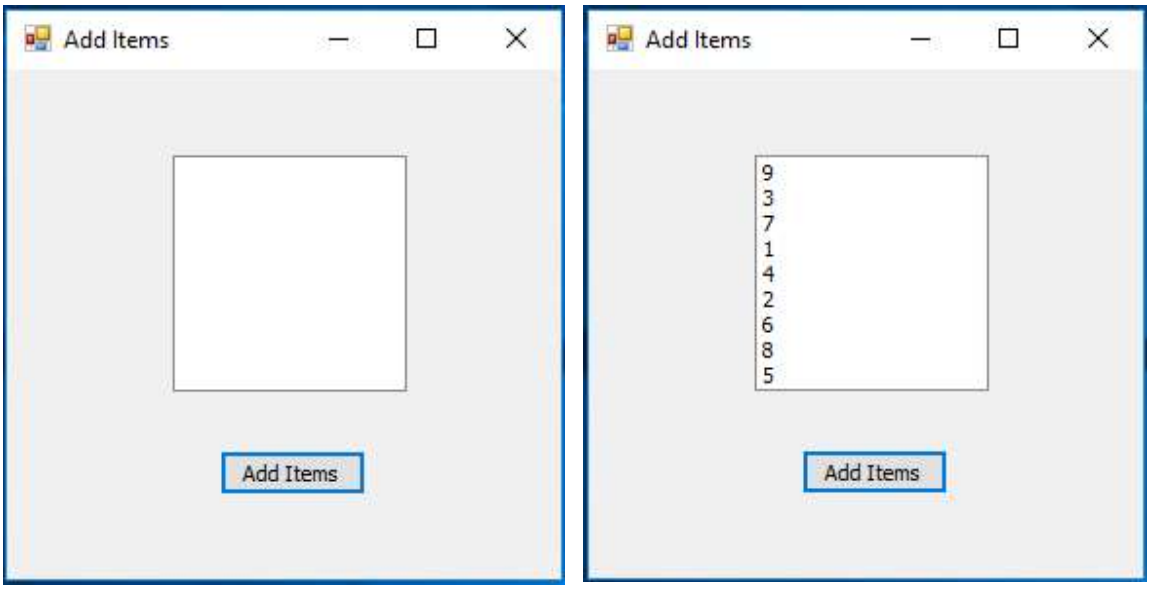
Property	Set
Text	Add Items

Source Code

Double click (DC) on Button 1, then write

```
ListBox1.Items.Add("9")
ListBox1.Items.Add("3")
ListBox1.Items.Add("7")
ListBox1.Items.Add("1")
ListBox1.Items.Add("4")
ListBox1.Items.Add("2")
ListBox1.Items.Add("6")
ListBox1.Items.Add("8")
ListBox1.Items.Add("5")
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 41: Add Items to ListBox and Sort the Items ASC

Add to your Form project

- ListBox
- 2 Buttons

Tool	Property	Set
Button1	Text	Add Items
Button2	Text	Sort
Button2	Enabled	False

Properties

Source Code

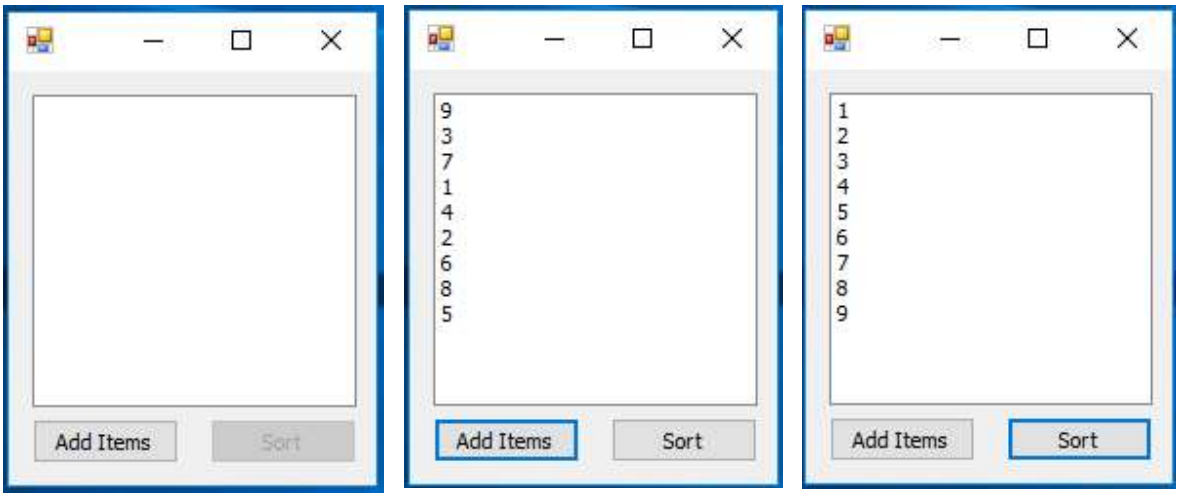
Double click (DC) on Button 1, then write

```
ListBox1.Items.Add("9")
ListBox1.Items.Add("3")
ListBox1.Items.Add("7")
ListBox1.Items.Add("1")
ListBox1.Items.Add("4")
ListBox1.Items.Add("2")
ListBox1.Items.Add("6")
ListBox1.Items.Add("8")
ListBox1.Items.Add("5")
Button2.Enabled = True
```

Double click (DC) on Button 2, then write

```
ArrayList.Adapter(ListBox1.Items).Sort()
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 42: Add Items to ListBox and Sort the Items DSC

Add to your Form project

- ListBox
- 2 Buttons

Properties

Tool	Property	Set
Button1	Text	Add Items
Button2	Text	Sort
Button2	Enabled	False

Source Code

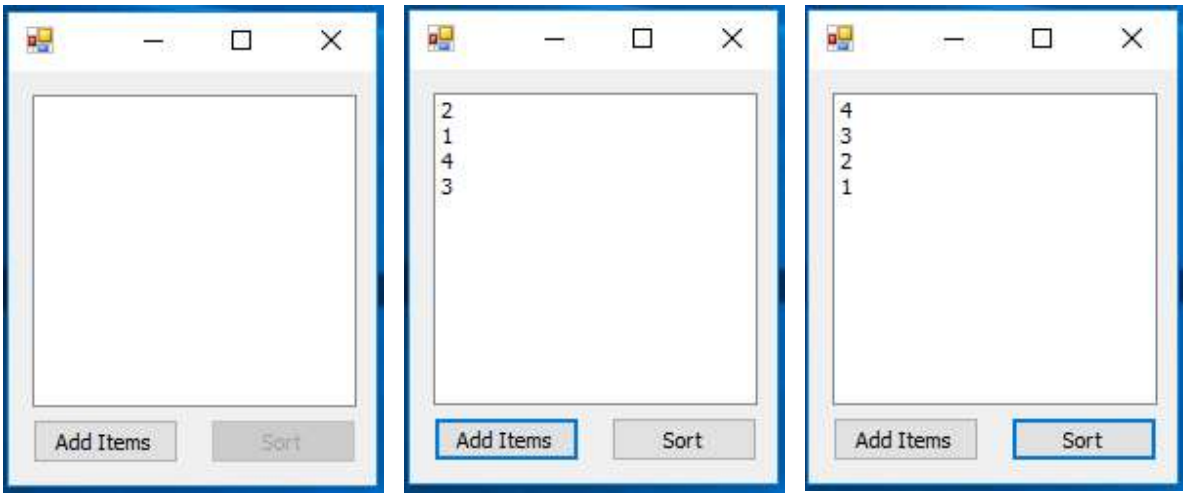
Double click (DC) on Button 1, then write

```
ListBox1.Items.Add("2")
ListBox1.Items.Add("1")
ListBox1.Items.Add("4")
ListBox1.Items.Add("3")
Button2.Enabled = True
```

Double click (DC) on Button 2, then write

```
Dim Sort_items(Me.ListBox1.Items.Count - 1) As String
For i As Integer = 0 To Me.ListBox1.Items.Count - 1
Sort_items(i) = Me.ListBox1.Items(i)
Next
Array.Sort(Sort_items)
Array.Reverse(Sort_items)
Me.ListBox1.Items.Clear()
Me.ListBox1.Items.AddRange(Sort_items)
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 43: Search in ListBox

Add to your Form project

- ListBox
- TextBox

Properties

Tool	Property	Set
ListBox	Text	ListBox1
TextBox	Text	TextBox1

Source Code

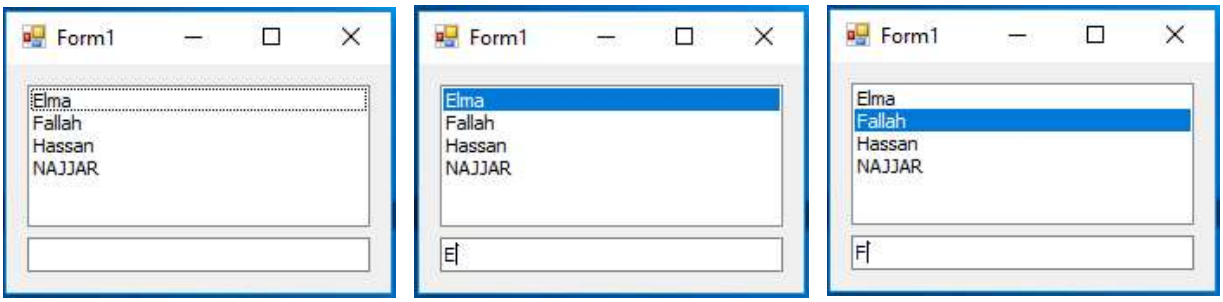
Double click (DC) on Form, then write

```
ListBox1.Items.Add("Elma")
ListBox1.Items.Add("Fallah")
ListBox1.Items.Add("Hassan")
ListBox1.Items.Add("NAJJAR")
```

Double click (DC) on TextBox 1, then write

```
ListBox1.SelectedIndex = ListBox1.FindString(TextBox1.Text)
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 44: Read Saved File to ListBox

Add to your Form project

- ListBox
- Button

Properties

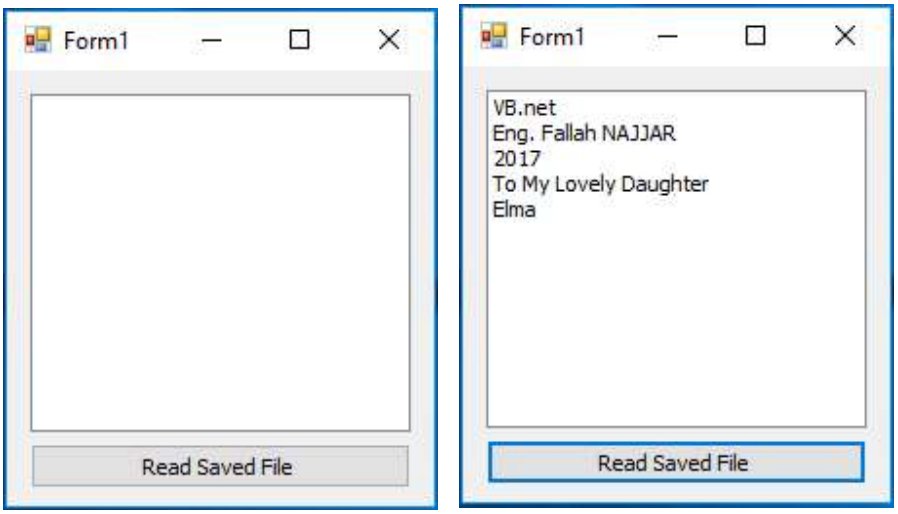
Tool	Property	Set
ListBox	Text	ListBox1
Button	Text	Read Saved File

Source Code

Double click (DC) on Button 1, then write

```
Dim Sr As IO.StreamReader
Sr = IO.File.OpenText("F:/Najjar.txt")
Do While Sr.Peek <> -1
ListBox1.Items.Add(Sr.ReadLine)
Loop
Sr.Close()
```

Then run your application (app) to



fallahnajjar@yahoo.com
+9647807445833

Project 45: Create TreeView

Add to your Form project

- Button

Button Properties

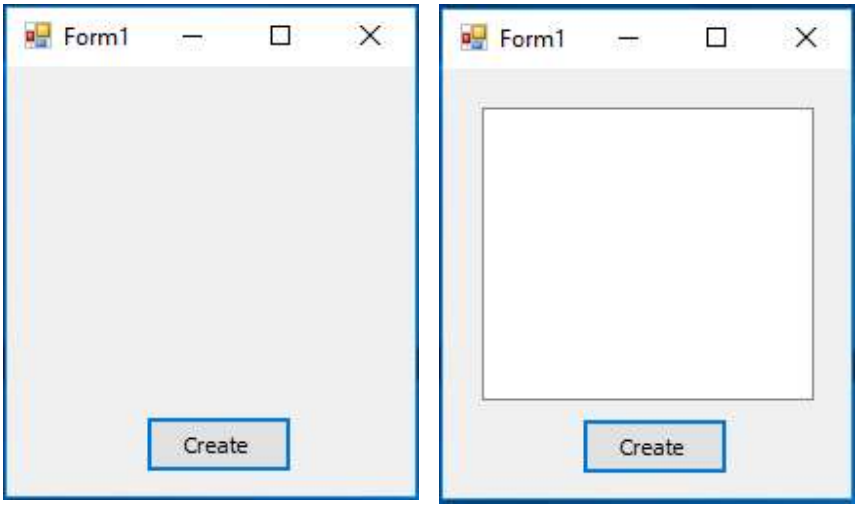
Property	Set
Text	Create

Source Code

Double click (DC) on Button 1, then write

```
Dim TreeView1 As New TreeView()
TreeView1.Size = New Size(110, 110)
TreeView1.Location = New Point(89, 81)
Me.Controls.Add(TreeView1)
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 46: Add Items & Nodes to TreeView

Add to your Form project

- TreeView
- Button

Button Properties

Property	Set
Text	Insert Node

Source Code

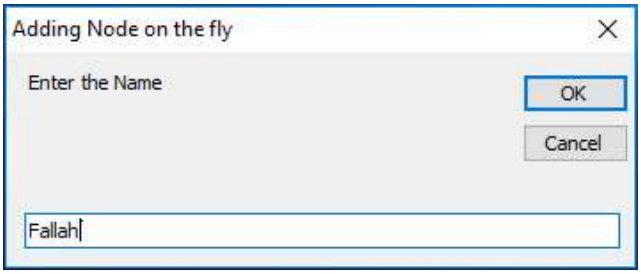
Double click (DC) on Form, then write

```
TreeView1.Nodes.Add("VB.Net")
TreeView1.Nodes.Add("Eng. Fallah NAJJAR")
TreeView1.Nodes.Add("2017")
TreeView1.Nodes.Add("To Mt Lovely Daughter")
TreeView1.Nodes.Add("Elma")
```

Double click (DC) on Button 1, then write

```
Dim strNodeName = InputBox("Enter the Name", "Adding Node on the fly", "Enter New Node")
If strNodeName <> "" Then
TreeView1.SelectedNode.Nodes.Add(strNodeName)
End If
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 47: Calculate how many Control Tools in the Form

Add to your Form project

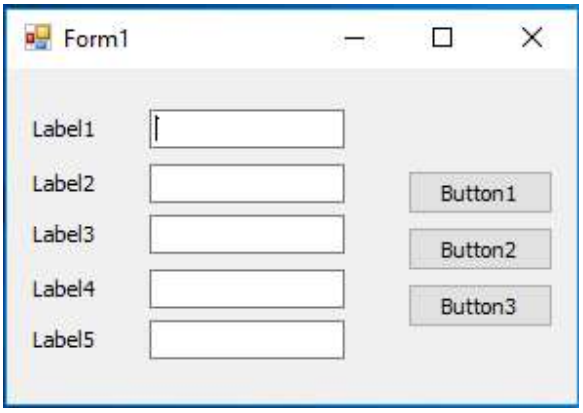
- 3 Buttons
- 5 Labels
- 5 TextBoxes

Source Code

Double click (DC) on Form, then write

```
Dim i, j, k As Integer
i = 0
j = 0
k = 0
Dim ctl As Control
For Each ctl In Me.Controls
If TypeOf ctl Is TextBox Then
i += 1
ElseIf TypeOf ctl Is Label Then
j += 1
ElseIf TypeOf ctl Is Button Then
k += 1
End If
Next
MessageBox.Show("There is " & i & " textboxes" & ControlChars.CrLf & "and " & j & " lables " &
ControlChars.CrLf & "and " & k & " buttons" & ControlChars.CrLf & "in The Form ... ")
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 48: Add Menu to the Form

Add to your Form project

- Buttons
- MenuStrip

Button Properties

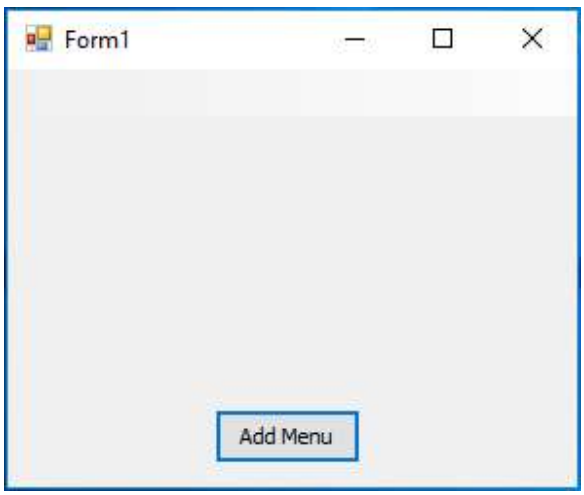
Property	Set
Text	Add Menu

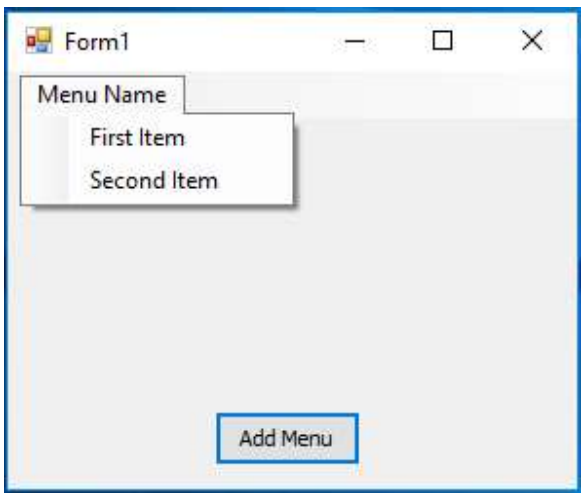
Source Code

Double click (DC) on Button 1, then write

```
Dim addedMenuStripItem As New ToolStripMenuItem
Dim firstDropDownItem As New ToolStripMenuItem
Dim secondDropDownItem As New ToolStripMenuItem
addedMenuStripItem.Text = "&Menu Name"
firstDropDownItem.Text = "&First Item"
secondDropDownItem.Text = "&Second Item"
MenuStrip1.Items.Add(addedMenuStripItem)
addedMenuStripItem.DropDownItems.Add(firstDropDownItem)
addedMenuStripItem.DropDownItems.Add(secondDropDownItem)
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 49: Working with ProgressBar

Add to your Form project

- ❖ Buttons
- ❖ ProgressBar

Property	Set
Text	Try

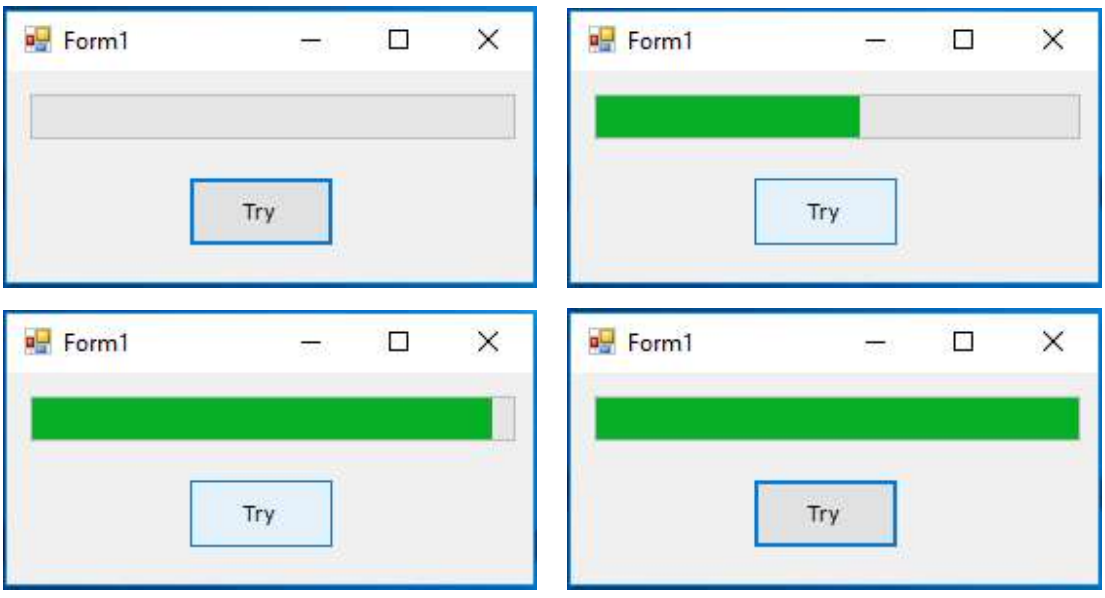
Button Properties

Source Code

Double click (DC) on Button 1, then write

```
With ProgressBar1
     .Minimum = 1
     .Maximum = 10000000
     .Value = 1
     .Step = 1
     For i As Integer = .Minimum To .Maximum
            .PerformStep()
     Next i
End With
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 50: Copy Picture to RichTextBox

Add to your Form project

- Button
- RichTextBox

Button Properties

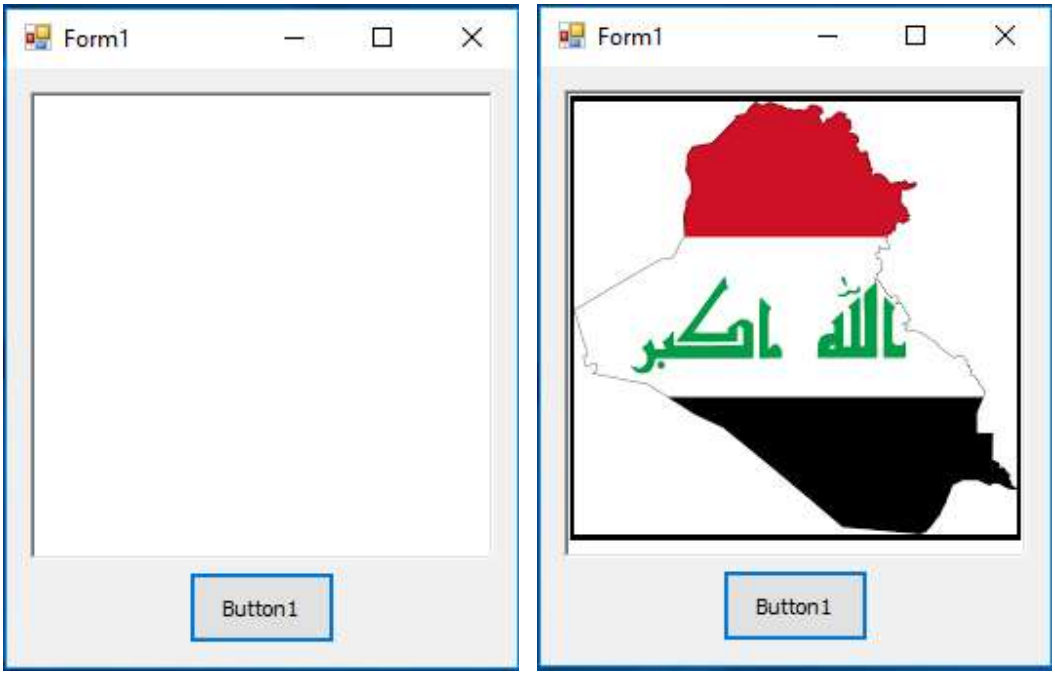
Property	Set
Text	Button1

Source Code

Double click (DC) on Button 1, then write

```
If My.Computer.Clipboard.ContainsImage = True Then
RichTextBox1.Paste()
End If
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 51: Build and Call a Procedure with your project

Add to your Form project

- Button

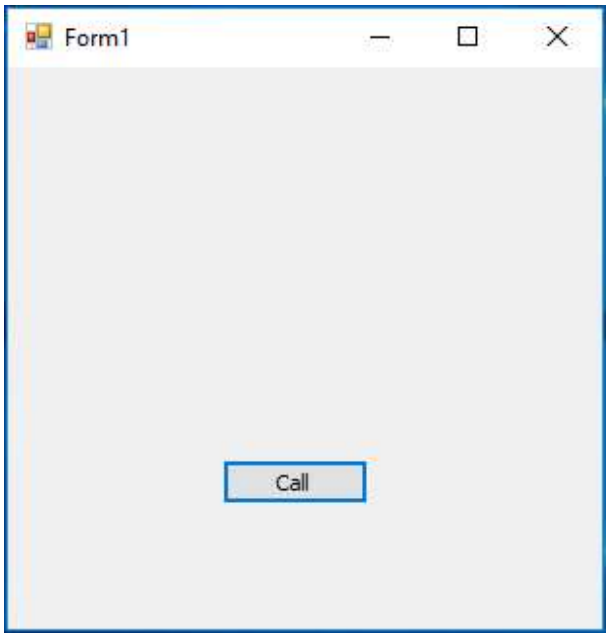
Property	Set
Name	btnCall
Text	Call

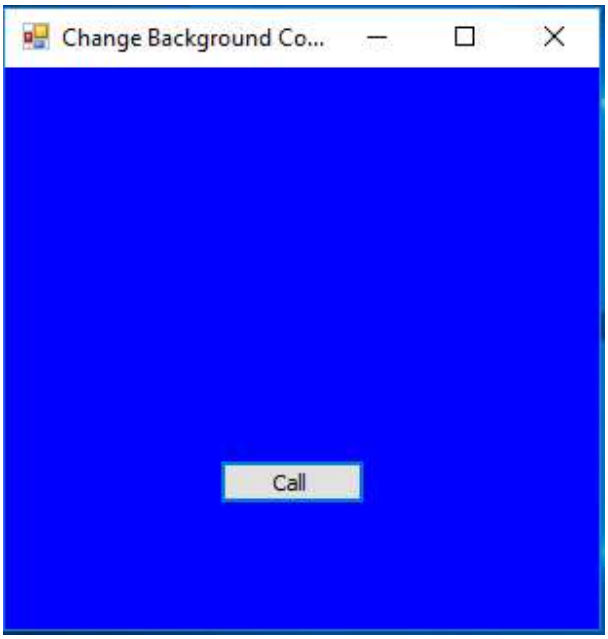
Source Code

Copy the Following code to your project

```
Public Class Form1
    Public Sub Change_Background_Color()  'sub procedure name
        With Me
            .BackColor = Color.Blue
            .Text = "Change Background Color"
        End With
    End Sub
    Private Sub btnCall_Click(sender As Object, e As EventArgs) Handles btnCall.Click
        Change_Background_Color()    'sub procedure calling in the btnCall click
    End Sub
End Class
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 52: Build and Call a Function with your project

Add to your Form project

- Button

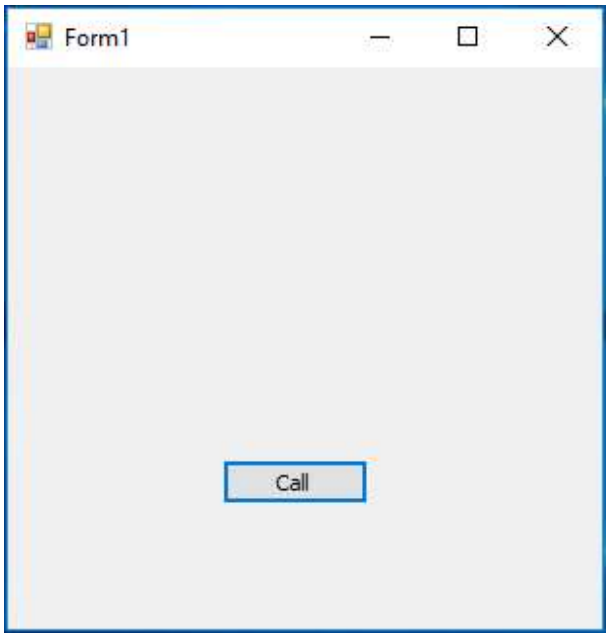
Property	Set
Name	btnCall
Text	Call

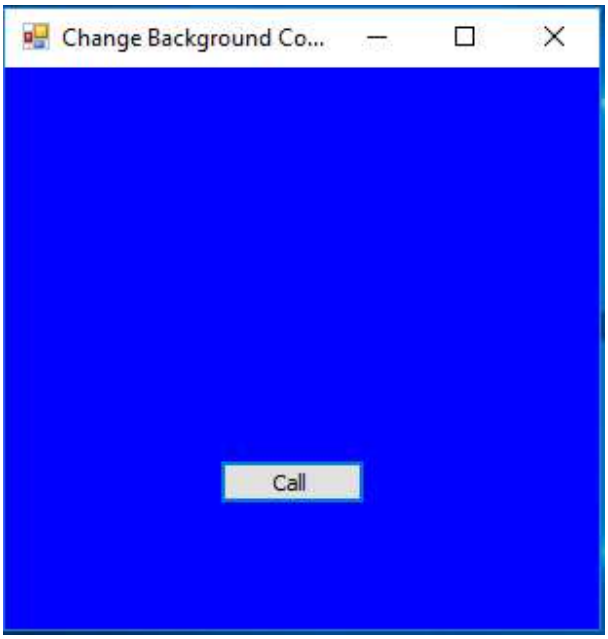
Source Code

Copy the Following code to your project

```
Public Class Form1
    Public Function Change_Background_Color() 'Function name
        With Me
            .BackColor = Color.Blue
            .Text = "Change Background Color"
        End With
        Return 0
    End Function
    Private Sub btnCall_Click(sender As Object, e As EventArgs) Handles btnCall.Click
        Change_Background_Color()    'Function calling in the btnCall click
    End Sub
End Class
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 53: Print in TextBox Using Function

Add to your Form project

- ❖ TextBox
- ❖ Button

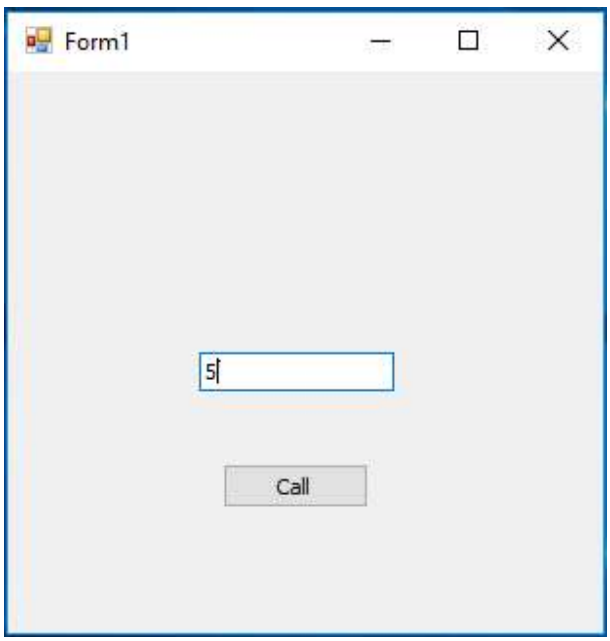
Property	Set
Name	btnCall
Text	Call

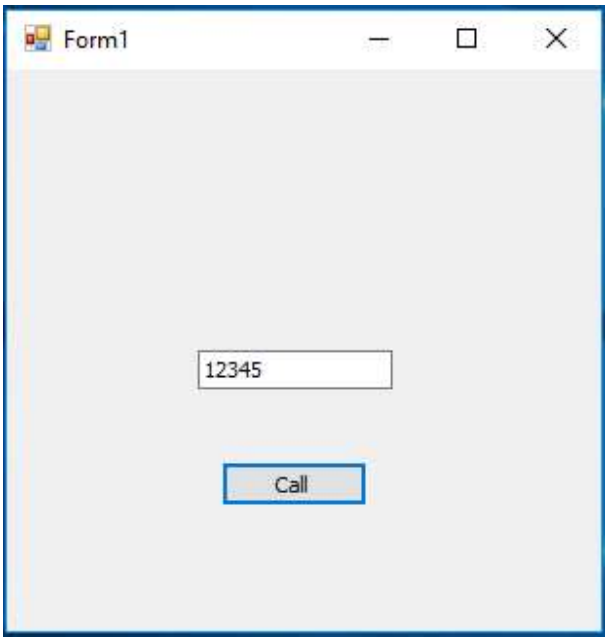
Source Code

Copy the Following code to your project

```
Public Class Form1
    Public Function test(ByVal No As Integer) As Integer ' function name
        TextBox1.Clear()                                 ' clear textbox
        For i = 1 To No                                  ' using for loop
            TextBox1.Text = TextBox1.Text & i            ' save the loop results in the text
        Next                                             ' end loop
        Return 0                                         ' return
    End Function                                         ' end function
    Private Sub btnCall_Click(sender As Object, e As EventArgs) Handles btnCall.Click
        Dim No As Integer                                ' an integer
        No = Val(TextBox1.Text)                          ' take the content of textbox
        test(No)                                         ' Call the Function
    End Sub
End Class
```

Then run your application (app) to get



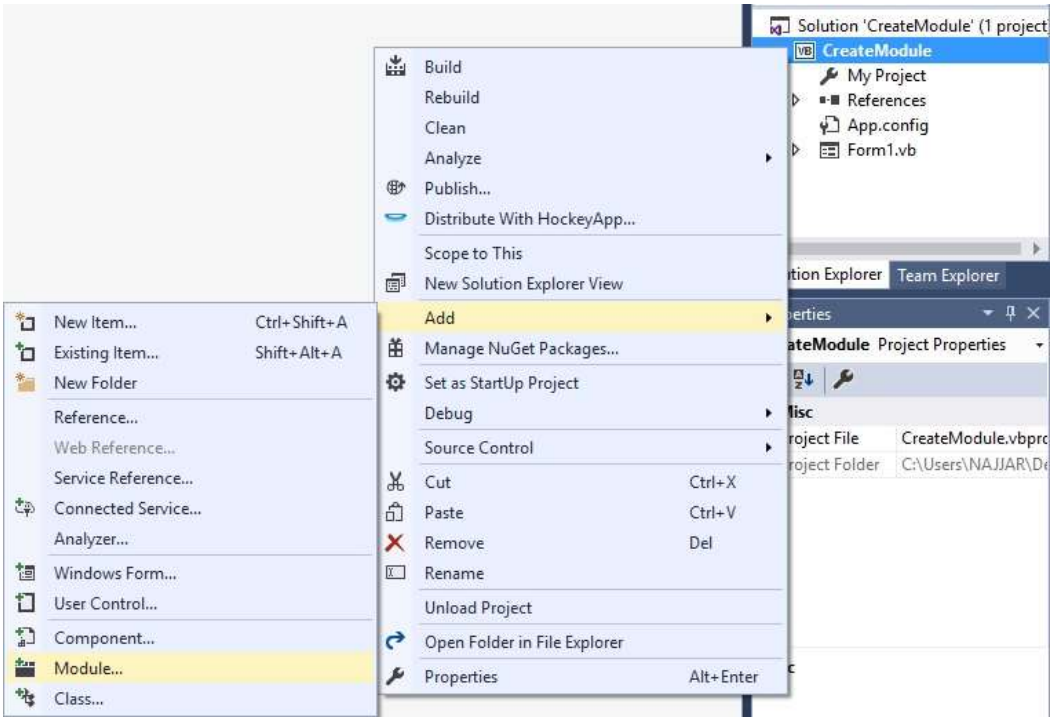


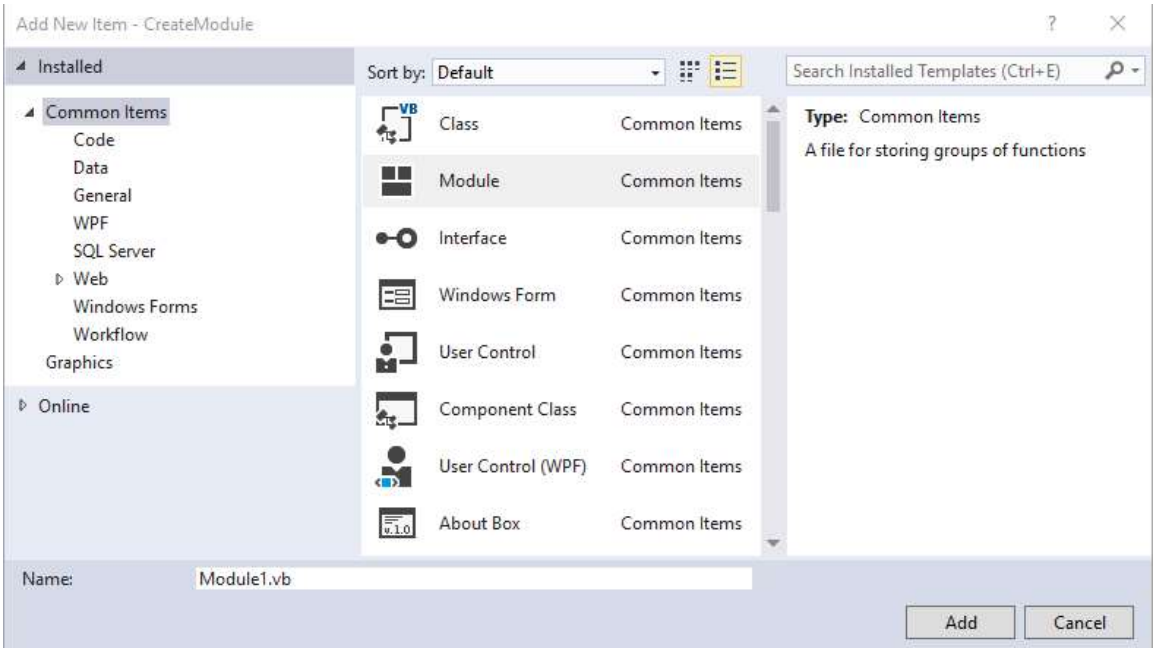
fallahnajjar@yahoo.com
+9647807445833

Project 54: Working with Module

Open New project it name is CreateModule

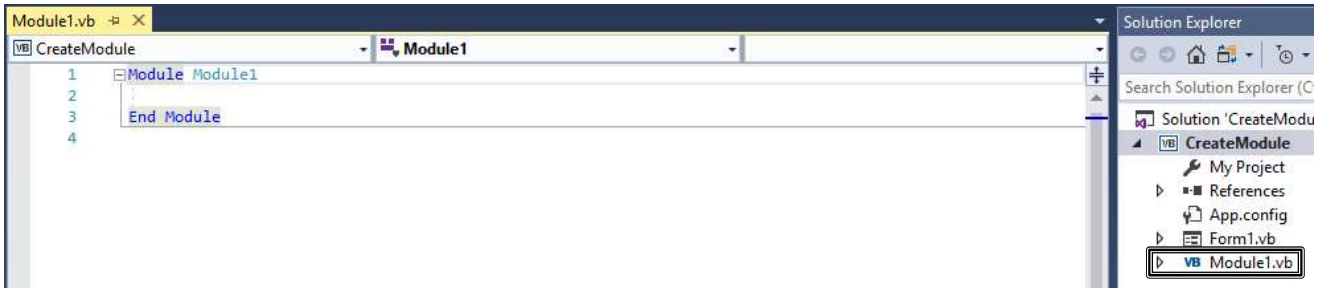
From Solution Explorer R.C. on the name of the project → Add → Module





Select Modula → Add

fallahnajjar@yahoo.com
+9647807445833



Then Add to your Form project

- ❖ TextBox
- ❖ Button

Button Properties

Property	Set
Name	btnCall
Text	Call

Source Code

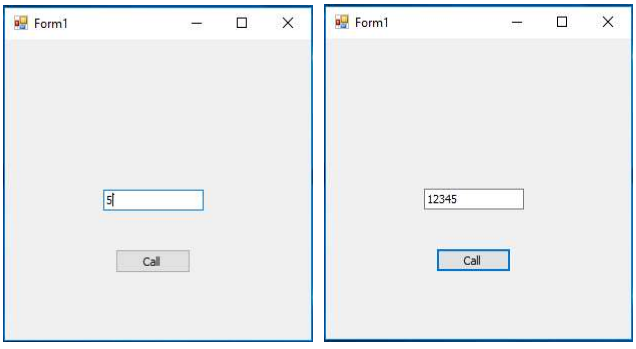
Copy the Following code to your Module

```
Module Module1
    Public No As Integer                                    ' an integer
    Public Function test(ByRef No) As Boolean               ' function name
        Form1.TextBox1.Clear()                              ' clear textbox
        For i = 1 To No                                     ' using for loop
            Form1.TextBox1.Text = Form1.TextBox1.Text & i    ' save the loop results in the text
        Next                                                ' end loop
        Return True                                         ' return
    End Function
End Module
```

Double click (DC) on Button 1, then write

```
No = Val(TextBox1.Text)                         ' take the content of textbox
test(No)                                        ' Call the Function
```

Then run your application (app) to get

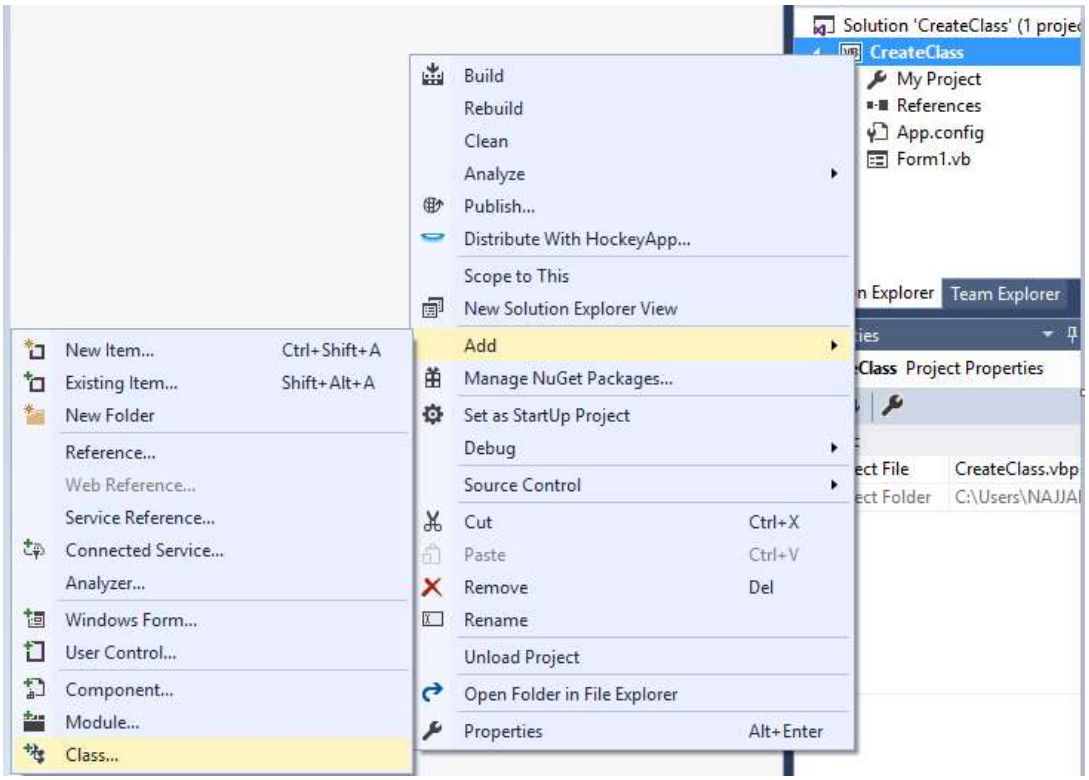


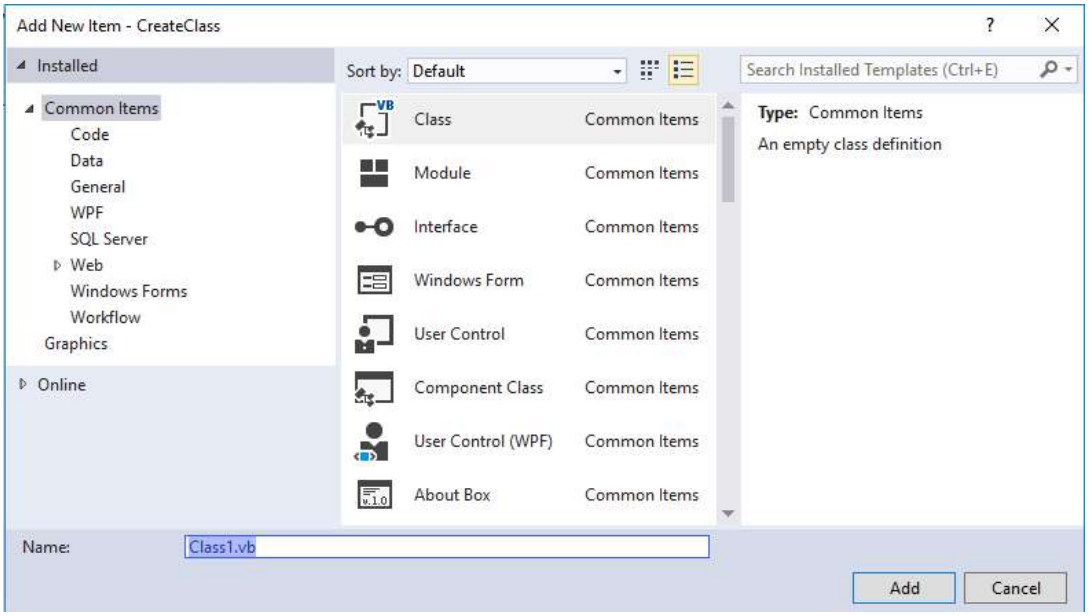
fallahnajjar@yahoo.com
+9647807445833

Project 55: Working with Class

Open New project it name is CreateClass

From Solution Explorer R.C. on the name of the project → Add → Class





Select Class → Add

fallahnajjar@yahoo.com
+9647807445833

Then Add to your Form project

- ❖ TextBox
- ❖ Button

Button Properties

Property	Set
Name	btnCall
Text	Call

Source Code

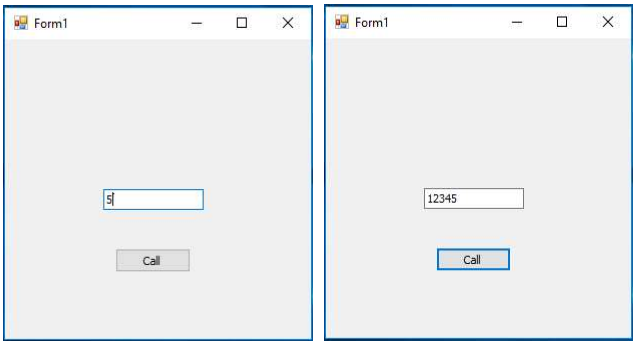
Copy the Following code to your added Class

```
Public Class Class1
    Public Function test(ByVal No) As Boolean                ' function name
        Form1.TextBox1.Clear()                               ' clear textbox
        For i = 1 To No                                      ' using for loop
            Form1.TextBox1.Text = Form1.TextBox1.Text & i    ' save the loop results in the text
        Next                                                 ' end loop
        Return True                                          ' return
    End Function
End Class
```

Replace All Code // Ctrl + A → Ctrl + V

```
Public Class Form1
    Public No As Integer                                     ' an integer
    Dim class1 As New Class1                                 ' define class
    Private Sub btnCall_Click(sender As Object, e As EventArgs) Handles btnCall.Click
        No = Val(TextBox1.Text)                              ' take the content of textbox
        class1.test(No)                                      ' Call the Function
End Sub
End Class
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 56: How to Use TabControl

Open New project it name is CreateClass
From Solution Explorer R.C. on the name of the project → Add → Class
Add to your Form project

- TabControl

  By default, when we add TabControl, two TabPages will appear each one could have its own control tools. You can run your project to see that.

- Add to TabPage1
  - ✓ 2 TextBoxes
  - ✓ 2 Buttons
  - ✓ 2 Labels
- Add to TabPage2
  - ✓ Button

Properties

Tools	Property	Set
Form1	StartPosition	CenterScreen
Form1	Text	TabControl
Form1	AcceptButton	btnLogin
Form1	CancelButton	btnCancel
TabPage1		
Button1	Name	btnLogin
Button1	Text	Login
Button2	Name	btnCancel
Button2	Text	Cancel
Label1	Text	Username:
Label2	Text	Password:
TabPage2		
Button1	Name	btnGo
Button1	Text	Go

Note that, you can add more than two TabPages just in case you need it

fallahnajjar@yahoo.com
+9647807445833

Source Code
Copy the Following code to its own tool

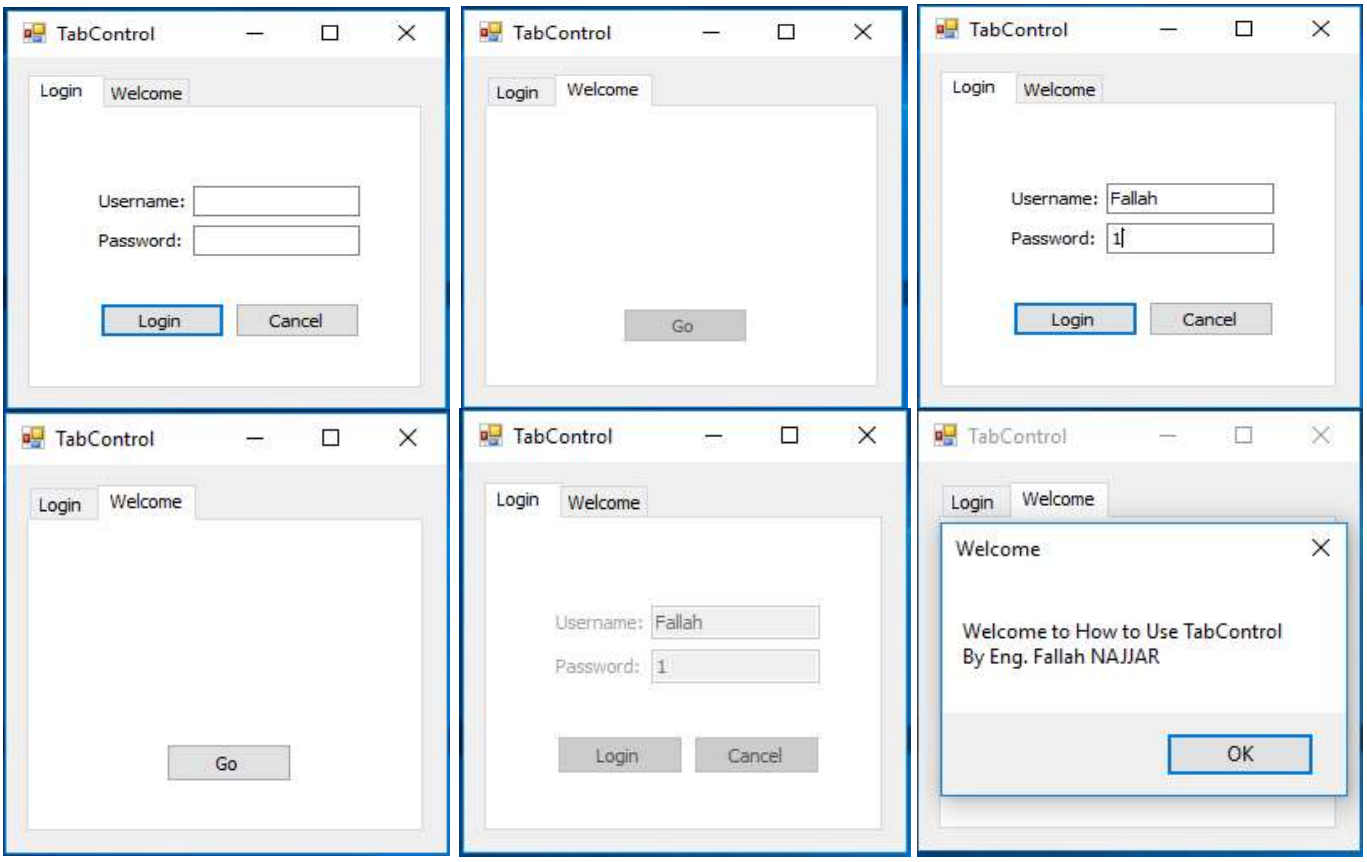
```
Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        TabControl1.TabPages("TabPage2").Enabled = False
End Sub
```

```
Private Sub btnLogin_Click(sender As Object, e As EventArgs) Handles btnLogin.Click
        If TextBox1.Text = "Fallah" And TextBox2.Text = "1" Then
            TabControl1.SelectedTab = TabPage2
            TabControl1.TabPages("TabPage2").Enabled = True
            TabControl1.TabPages("TabPage1").Enabled = False
        Else
            MsgBox("Error Username or Password",, "Error")
        End If
End Sub
```

```
Private Sub btnCancel_Click(sender As Object, e As EventArgs) Handles btnCancel.Click
        End
End Sub
```

```
Private Sub btnGo_Click(sender As Object, e As EventArgs) Handles btnGo.Click
    MsgBox("Welcome to How to Use TabControl" & vbNewLine & "By Eng. Fallah NAJJAR",, "Welcome")
End Sub
```

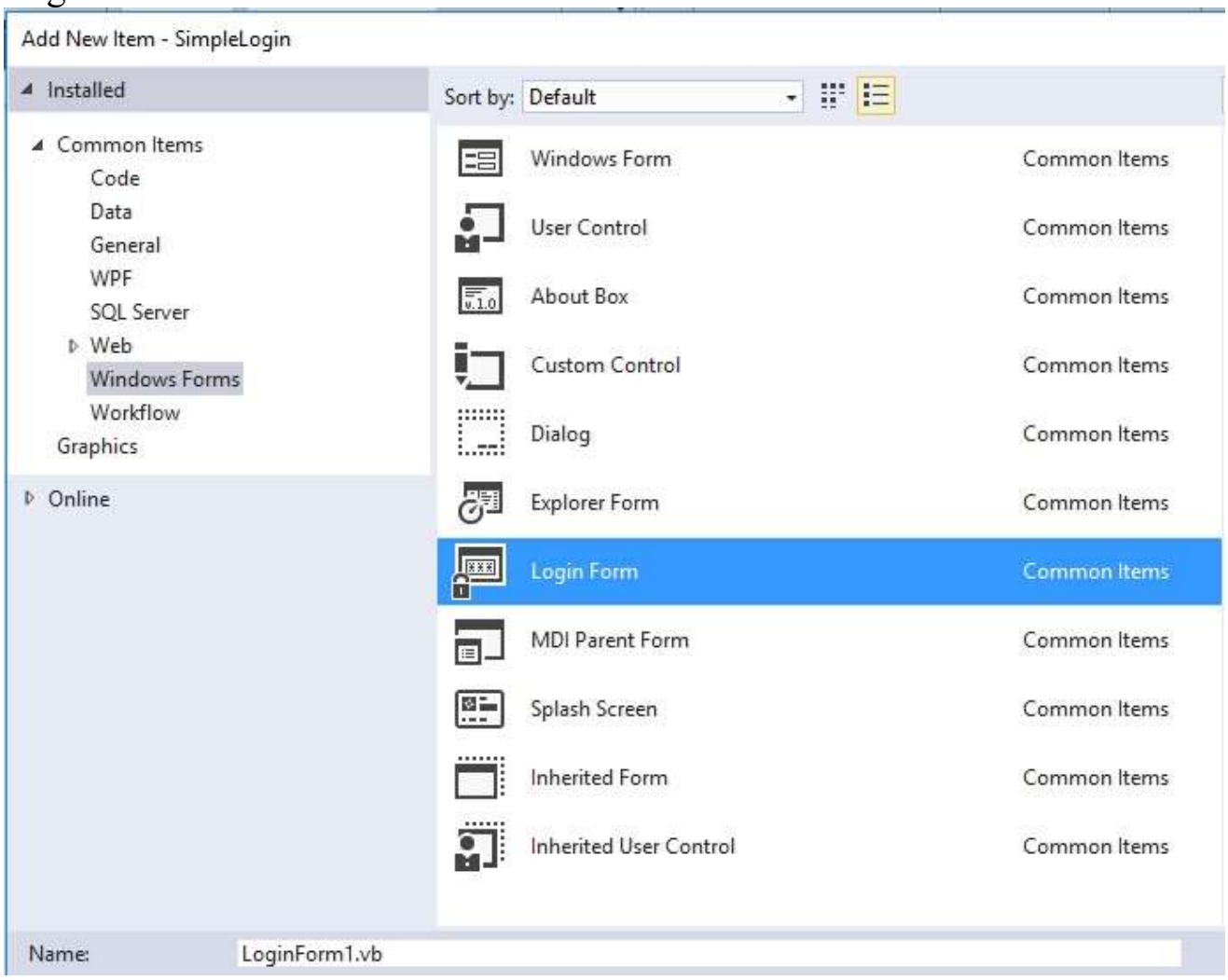
Then run your application (app) to get



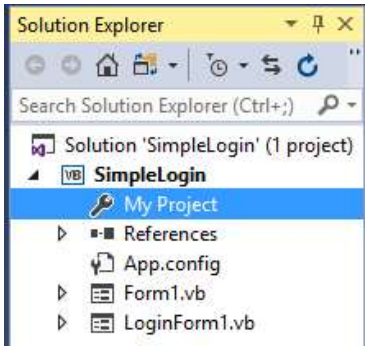
fallahnajjar@yahoo.com
+9647807445833

Project 57: Simple Login Form

Open New project its name is SimpleLogin
From Solution Explorer
R.C. on the name of the project → Add → New Item → Windows Forms → Login Form → Add

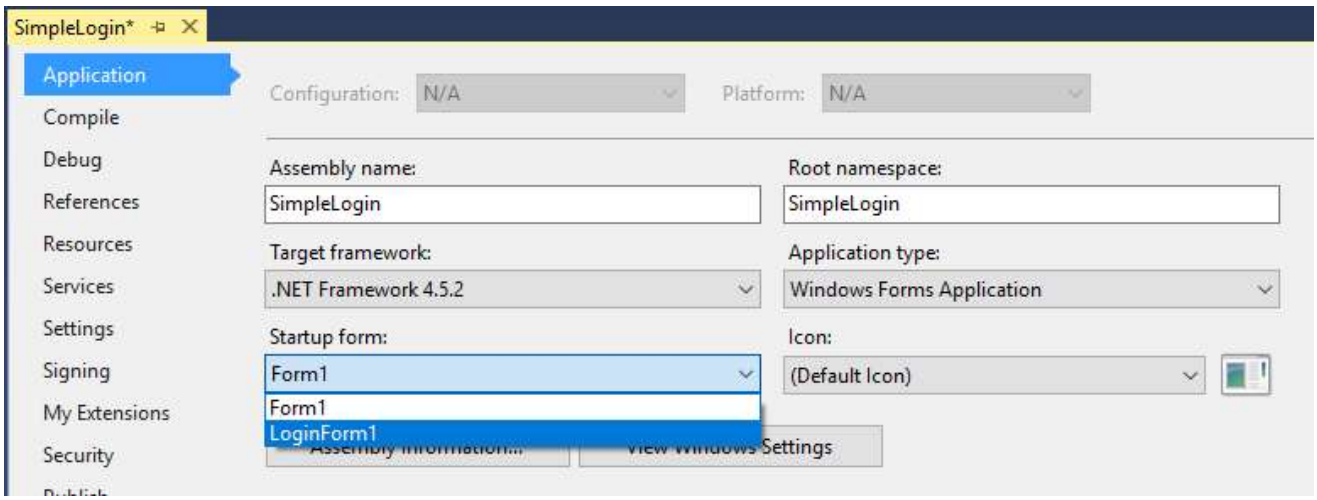


D.C. On My Project

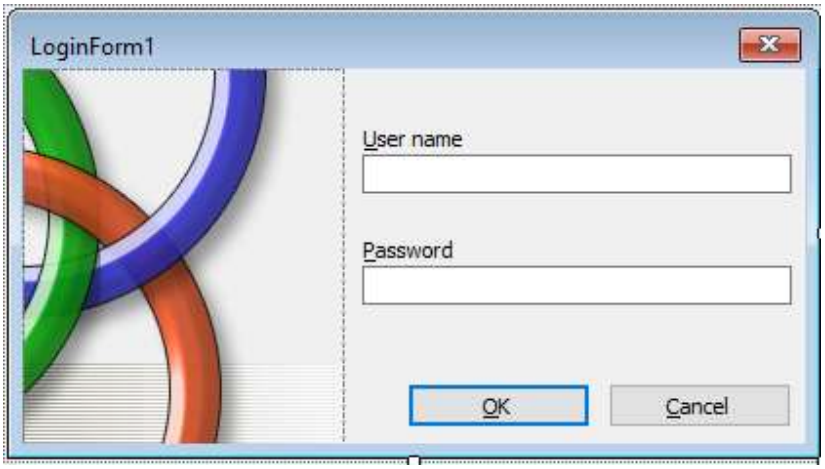


fallahnajjar@yahoo.com
+9647807445833

Application → Startup Form → LoginForm1



You'll get the following Form1



Copy this code to Ok Button

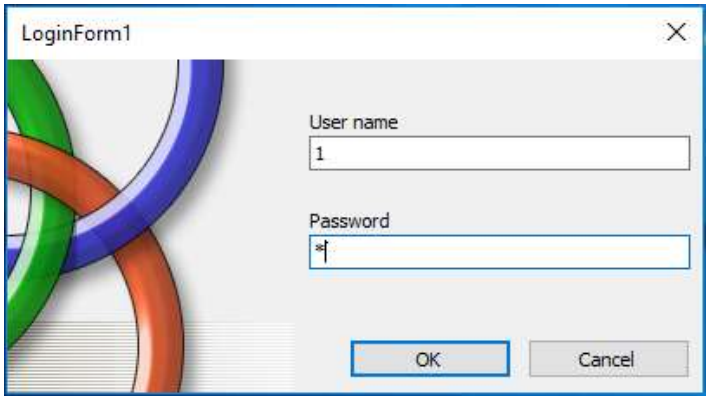
```
If UsernameTextBox.Text = "1" And PasswordTextBox.Text = "1" Then
            MsgBox("Successfully Login",, "Welcome")
            Me.Hide()
            Form1.Show()
        Else
            MsgBox("Wrong Username Or Password",, "Wrong")
End If
```

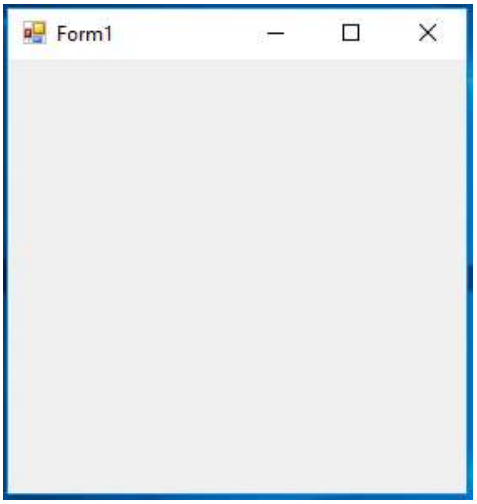
Copy this code to Cancel Button

```
Me.Close()
```


fallahnajjar@yahoo.com
+9647807445833

Then run your application (app) to get



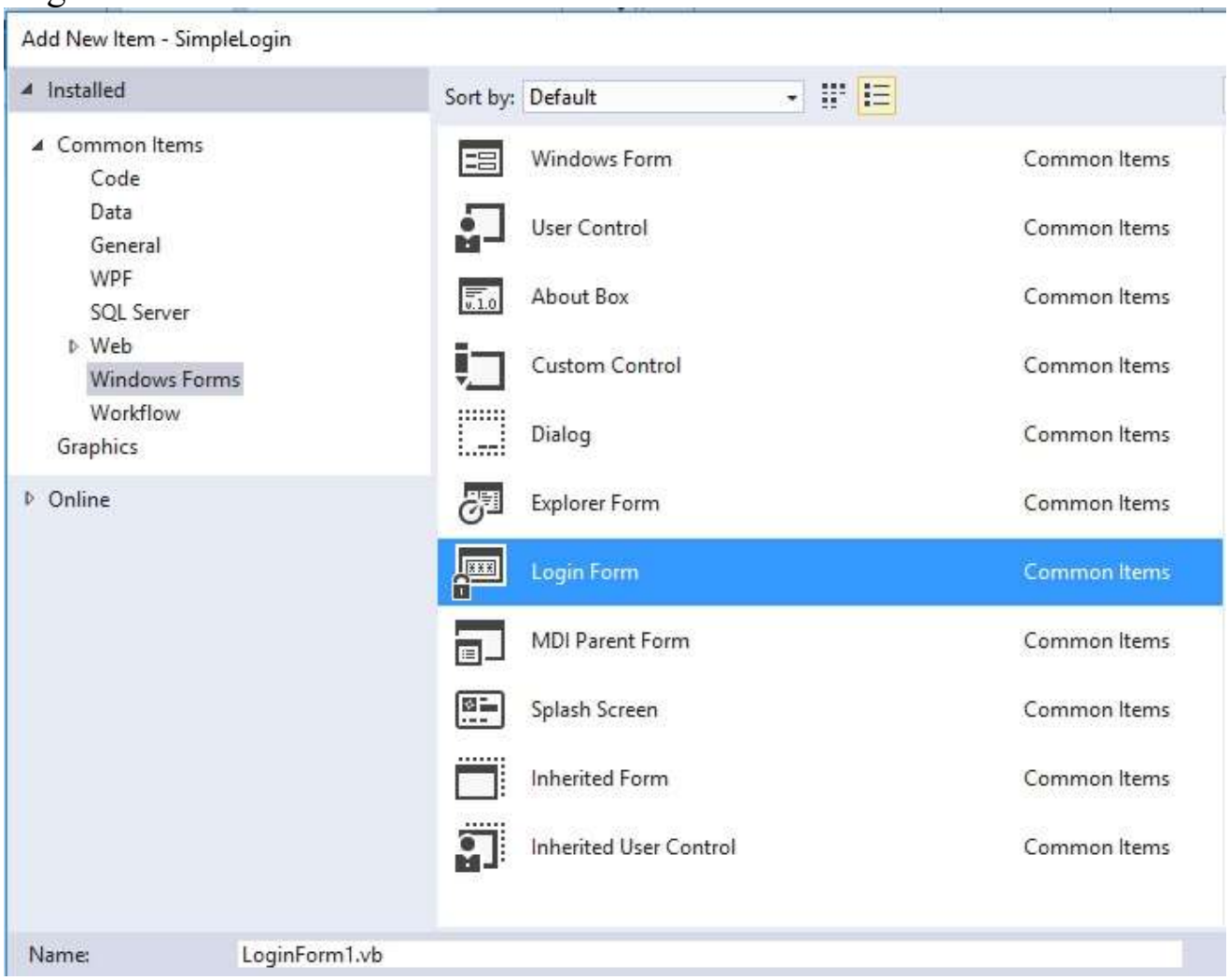


Both Username and Password Are 1.

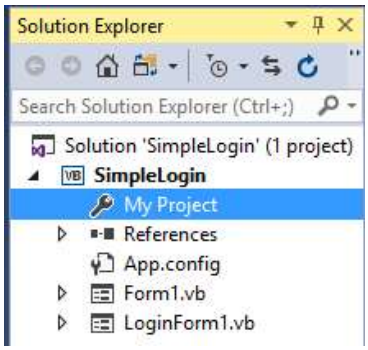
fallahnajjar@yahoo.com
+9647807445833

Project 58: Advanced Login Form

Open New project its name is SimpleLogin
From Solution Explorer
R.C. on the name of the project → Add → New Item → Windows Forms → Login Form → Add

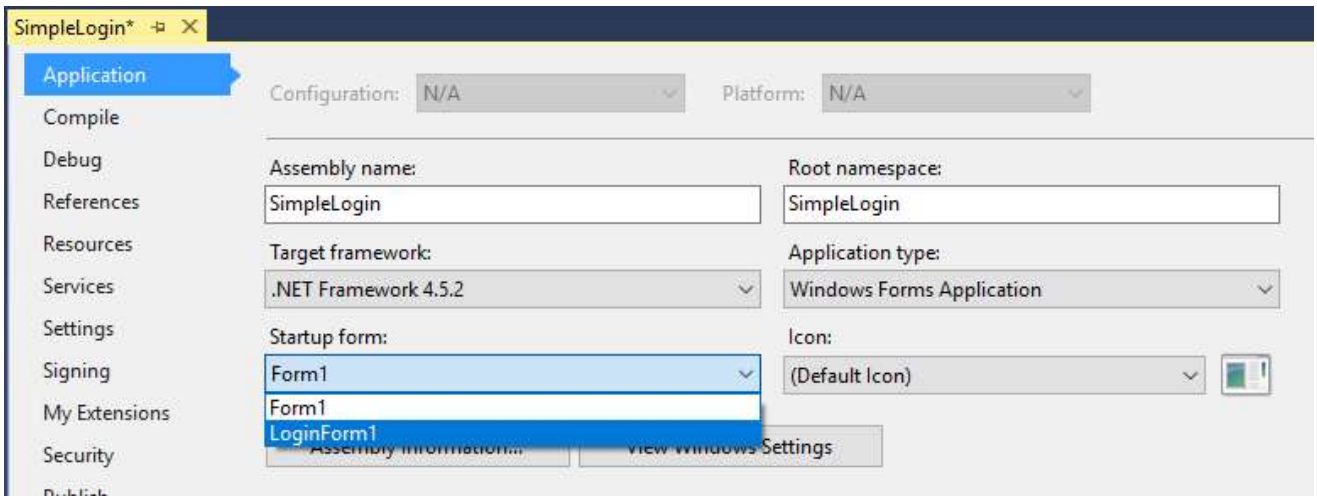


D.C. On My Project



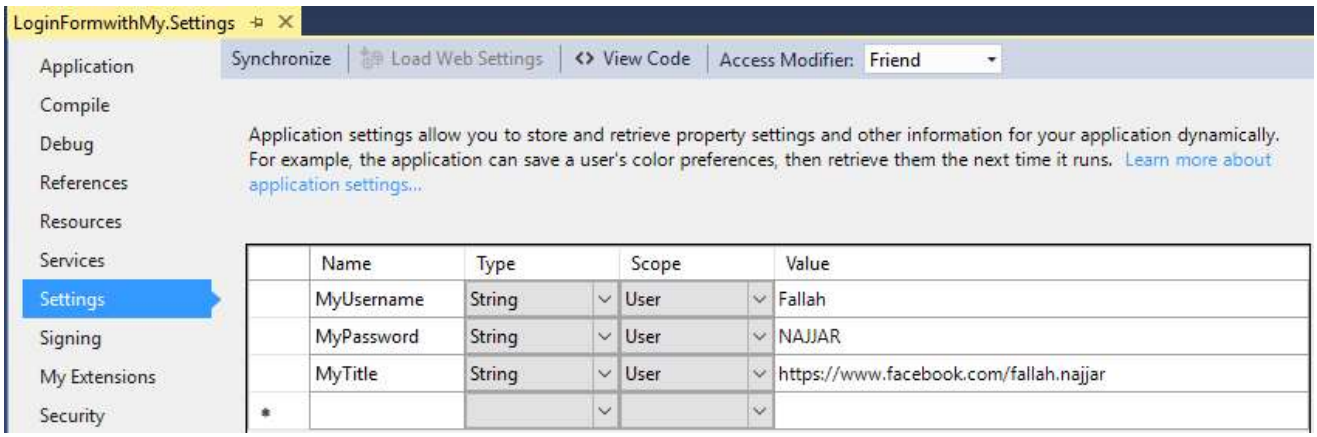
fallahnajjar@yahoo.com
+9647807445833

Application → Startup Form → LoginForm1



Then go to Settings and write

Name	Value
MyUsername	Fallah
MyPassword	NAJJAR
MyTitle	http://www.facebook.com/fallah.najjar



fallahnajjar@yahoo.com
+9647807445833

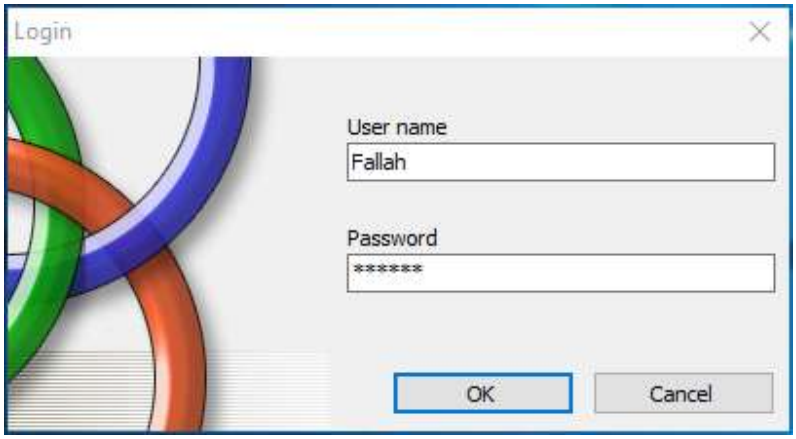
Copy this code to Ok Button

```
If txtUsername.Text = My.Settings.MyUsername And txtPassword.Text = My.Settings.MyPassword Then
MessageBox.Show("Successfully Login!", My.Settings.MyTitle, MessageBoxButtons.OK)
'clear inputs
txtUsername.Clear()
txtPassword.Clear()
Form1.Show()
Form1.txtUser.Focus()
Me.Hide()
Else
MessageBox.Show("Wrong Username or Password", "Login Failed", MessageBoxButtons.OK,
MessageBoxIcon.Error)
'clear inputs
txtUsername.Clear()
txtPassword.Clear()
txtUsername.Focus()
End If
```

Copy this code to Cancel Button

```
Me.Close()
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 59: Advanced SQL Database Connection

Add to your Form project

- ❖ 4 Labels
- ❖ 6 Buttons (Add, Save, Delete, Update, View, and Exit)
- ❖ 3 TextBoxes
- ❖ DateTimePicker
- ❖ DataGridView

Properties

Tools	Property	Set
Form1	StartPosition	CenterScreen
Form1	Text	Home
Form1	Icon	
Form1	Name	frmHome
Button1	Name	btnAdd
Button1	Text	" "
Button1	Image	
Button2	Name	btnSave
Button2	Text	" "
Button2	Image	
Button3	Name	btnUpdate
Button3	Text	" "
Button3	Image	
Button4	Name	btnDelete
Button4	Text	" "
Button4	Image	
Button5	Name	btnView
Button5	Text	" "
Button5	Image	
Button6	Name	btnExit
Button6	Text	" "
Button6	Image	
Label1	Text	Id:
Label2	Text	FName:
Label3	Text	LName:
Label4	Text	Birthday:
DGV	Name	Info_tblDataGridView

fallahnajjar@yahoo.com
+9647807445833

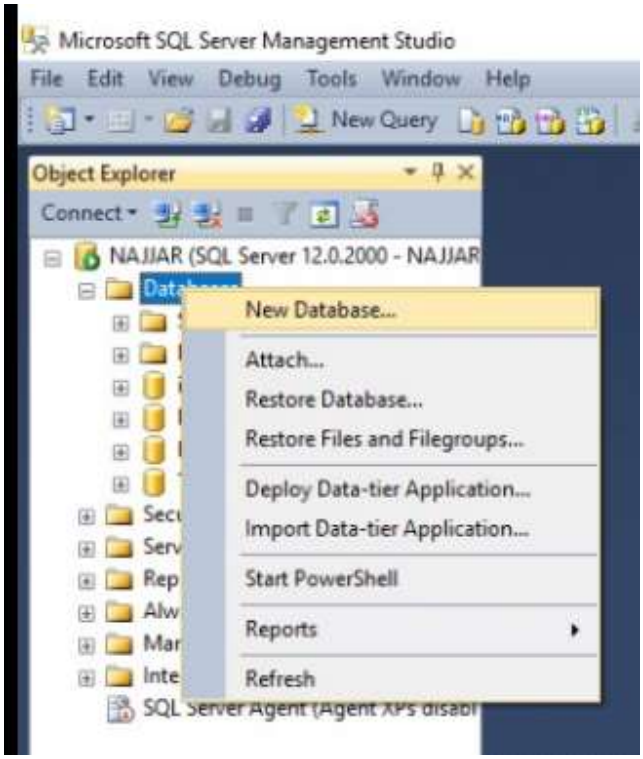
Create SQL Database as Follow

- Open Microsoft SQL Server Management Studio (any Version)

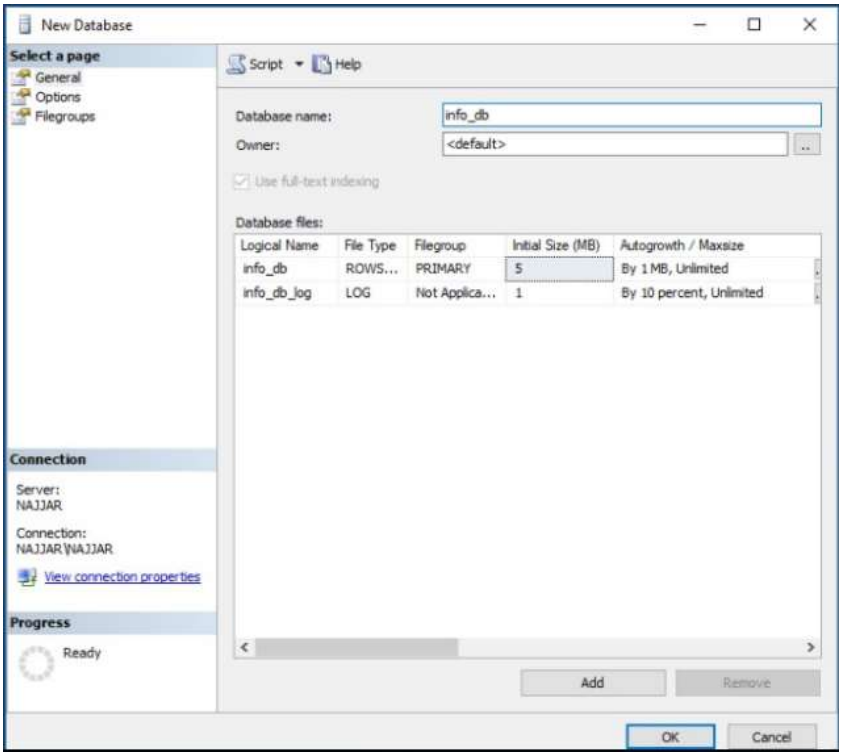
- Connect to your server

fallahnajjar@yahoo.com
+9647807445833

❖ Create New DB → R.C. on Database → New Database

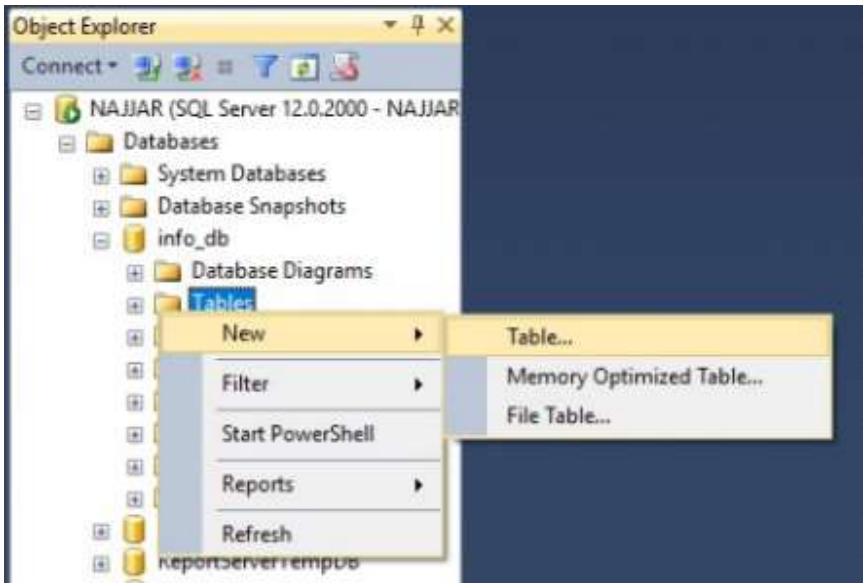


❖ Type Database Name → OK (in our case) info_db

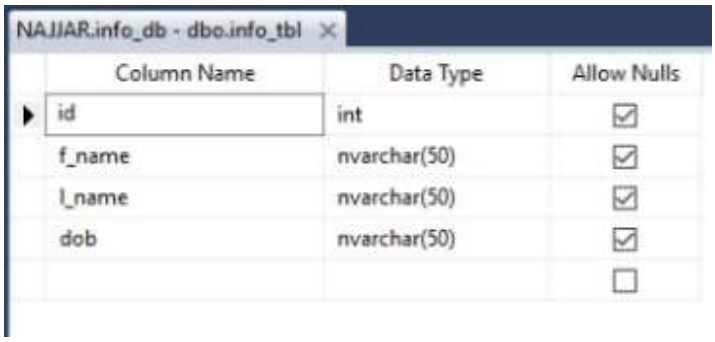


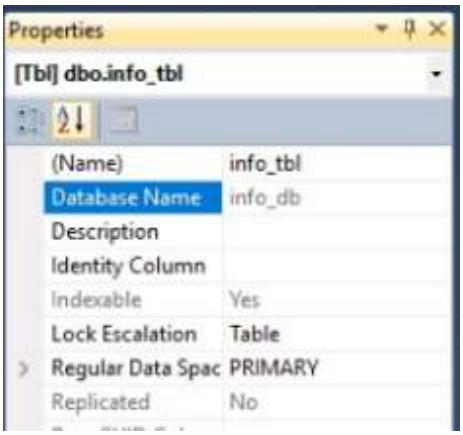
fallahnajjar@yahoo.com
+9647807445833

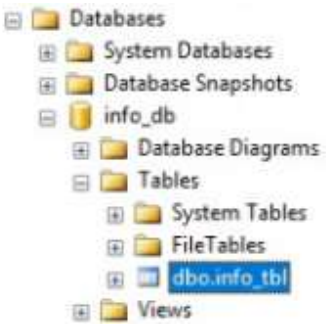
❖ Create New Table → R.C. on Tables → New → Table



❖ Design your Table Fields → save Table (in our case) info_tbl







❖ You are ready to write your code connection and control DB.

fallahnajjar@yahoo.com
+9647807445833

Copy the codes to your project

```
Imports System.Data
Imports System.Data.SqlClient
Public Class frmDB
'Connect to DB
Public conn As New SqlConnection("Server= najjar; Database = info_db; integrated security =
true")
'Get data to DGV
Public Function FillDataFun() As DataTable
Dim dtInfo As New DataTable
Dim FillData As String = "Select * From info_tbl"
Using comm1 As New SqlCommand(FillData, conn)
conn.Open()
Dim reader As SqlDataReader = comm1.ExecuteReader
dtInfo.Load(reader)
conn.Close()
End Using
Return dtInfo
End Function
' Query reader
Public Sub ExcuteQuery(query As String)
Dim comm As New SqlCommand(query, conn)
conn.Open()
comm.ExecuteNonQuery()
conn.Close()
End Sub
'Get Max id to reset id
Public Sub autoId()
Try
Dim comm As New SqlCommand("SELECT MAX(id) FROM info_tbl", conn)
conn.Open()
Dim dr As SqlDataReader = comm.ExecuteReader

If dr.Read = True Then
Me.txtID.Text = dr.Item(0) + 1
Else
Exit Sub
End If
Catch ex As Exception
End Try
conn.Close()
End Sub
'Clear all textboxes
Public Sub txtClear()
txtFname.Clear()
txtLname.Clear()
End Sub
'Load data to DGV
Private Sub frmDB_Load(sender As Object, e As EventArgs) Handles MyBase.Load
Info_tblDataGridView.DataSource = FillDataFun()
End Sub
```

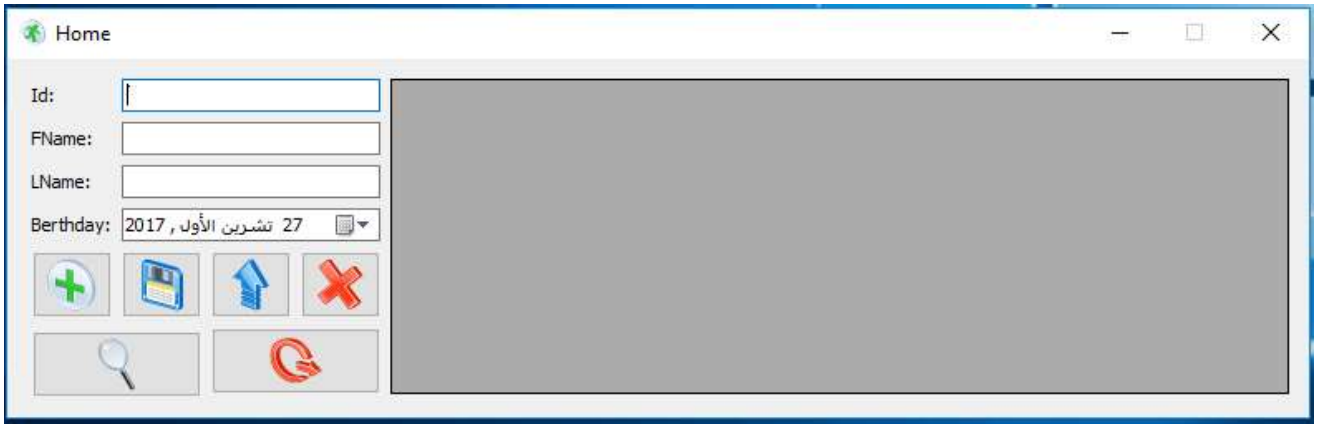

fallahnajjar@yahoo.com
+9647807445833

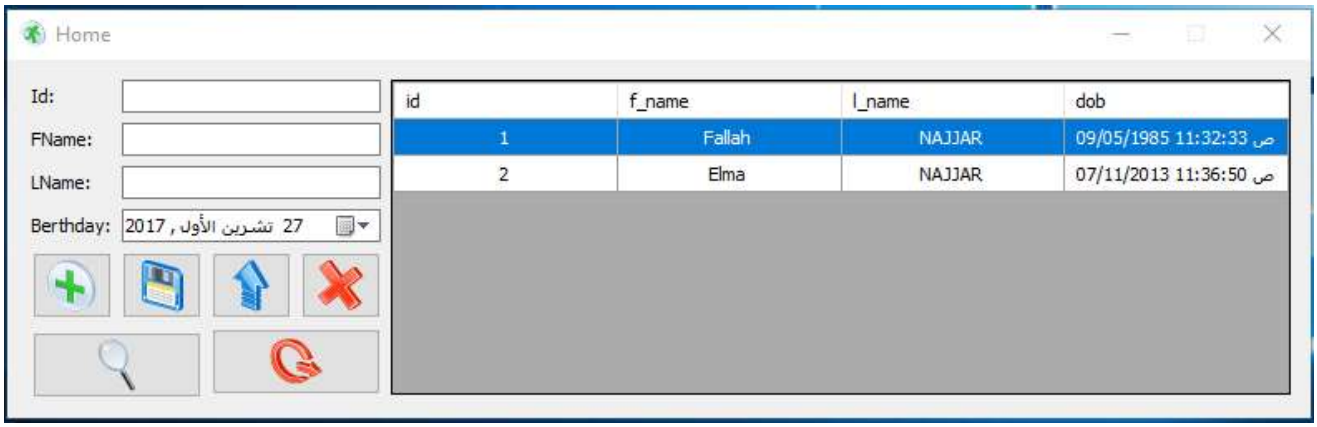
```
'Save new record
Private Sub btnSave_Click(sender As Object, e As EventArgs) Handles btnSave.Click
If txtFname.Text = "" Or txtLname.Text = "" Then
MsgBox("Enter Required Data",, "Worning")
Else
Dim InsertQuery As String = "INSERT INTO info_tbl (id,f_name,l_name,dob) VALUES('" & txtID.Text
& "','" & txtFname.Text & "','" & txtLname.Text & "','" & dtDOB.Value.ToString & "')"
ExcuteQuery(InsertQuery)
Info_tblDataGridView.DataSource = FillDataFun()
txtClear()
dtDOB.Value = Today
End If
txtID.Focus()
Call autoId()
End Sub
'Delete record
Private Sub btnDelete_Click(sender As Object, e As EventArgs) Handles btnDelete.Click
Dim DeleteQuery As String = "DELETE FROM info_tbl WHERE id='" & txtID.Text & "'"
ExcuteQuery(DeleteQuery)
Info_tblDataGridView.DataSource = FillDataFun()
txtClear()
End Sub
'Update record
Private Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
Dim UpdateQuery As String = "UPDATE info_tbl SET id= '" & txtID.Text & "',f_name= '" &
txtFname.Text & "',l_name ='" & txtLname.Text & "',dob = '" & dtDOB.Value.ToString & "'  WHERE
id = '" & txtID.Text & "'"
ExcuteQuery(UpdateQuery)
Info_tblDataGridView.DataSource = FillDataFun()
txtClear()
End Sub
'Get data from DGV to textboxes
Private Sub Info_tblDataGridView_CellClick(sender As Object, e As DataGridViewCellEventArgs)
Handles Info_tblDataGridView.CellClick
Dim index As Integer
index = e.RowIndex
Dim SelectedRow As DataGridViewRow
Info_tblDataGridView.SelectionMode = DataGridViewSelectionMode.FullRowSelect
Try
SelectedRow = Info_tblDataGridView.Rows(index)
txtID.Text = SelectedRow.Cells(0).Value.ToString()
txtFname.Text = SelectedRow.Cells(1).Value.ToString()
txtLname.Text = SelectedRow.Cells(2).Value.ToString()
dtDOB.Value = Convert.ToDateTime(SelectedRow.Cells(3).Value)
Catch ex As Exception
End Try
End Sub
Private Sub btnExit_Click(sender As Object, e As EventArgs) Handles btnExit.Click
End
End Sub
Private Sub btnAdd_Click(sender As Object, e As EventArgs) Handles btnAdd.Click
txtClear()
autoId()
End Sub
```


fallahnajjar@yahoo.com
+9647807445833

```
Private Sub btnView_Click(sender As Object, e As EventArgs) Handles btnView.Click
Dim CountQuery As String = "Select  COUNT(id) From info_tbl"
conn.Open()
Dim Command = New SqlCommand(CountQuery, conn)
Dim sqlReader As SqlDataReader = Command.ExecuteReader()
While sqlReader.Read()
MsgBox("Count =" & sqlReader.Item(0))
End While
sqlReader.Close()
Command.Dispose()
conn.Close()
End Sub
End Class
```

Then run your application (app) to get





fallahnajjar@yahoo.com
+9647807445833

Project 60: Working with Excel

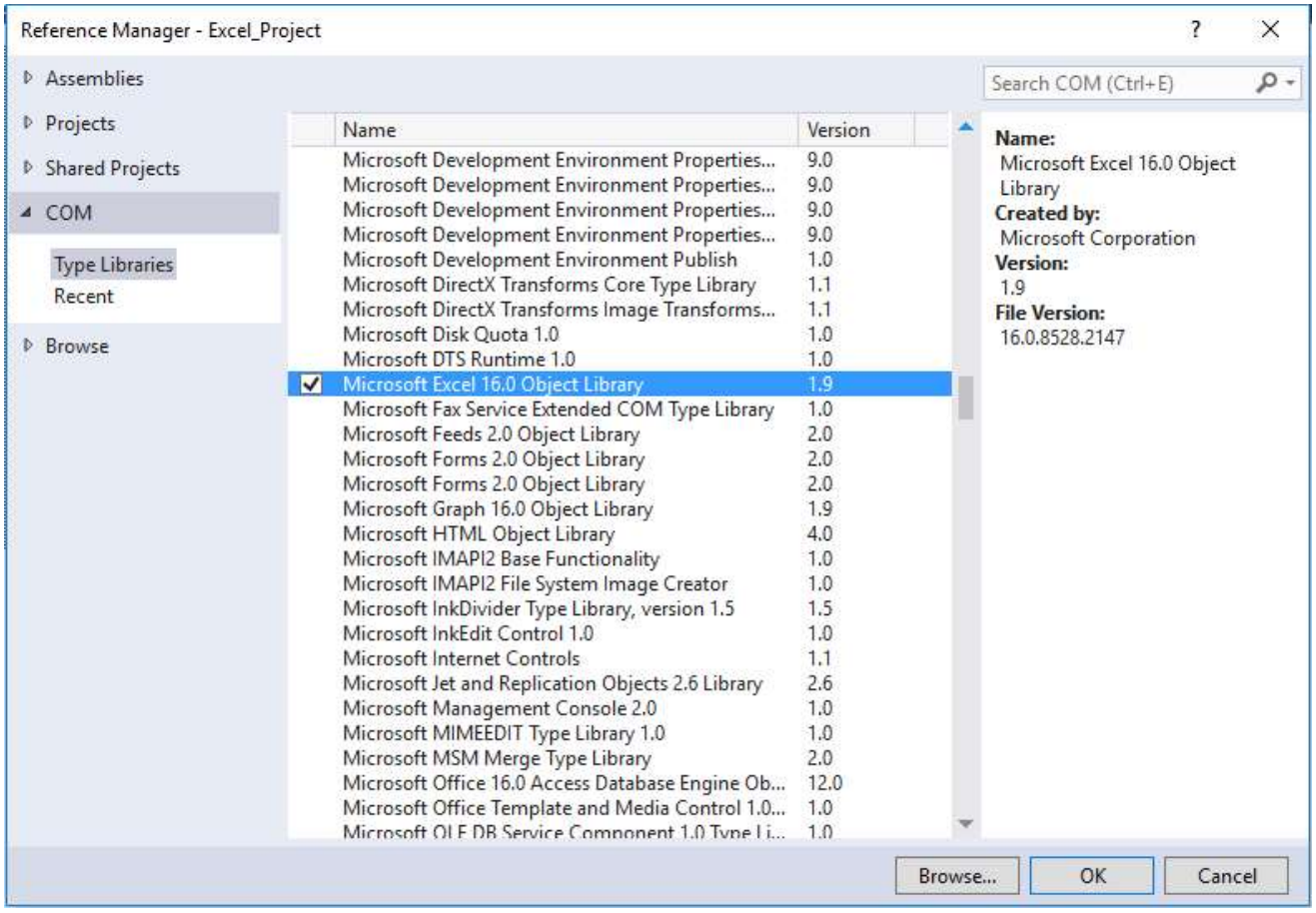
Add to your Form project

- ❖ Button1

Property	Set
Name	Button1
Text	Open

Add a reference to Microsoft Excel Object Library to your project.

R.C. on references → Add reference → COM → Microsoft Excel 16.0 → OK



fallahnajjar@yahoo.com
+9647807445833

Source Code

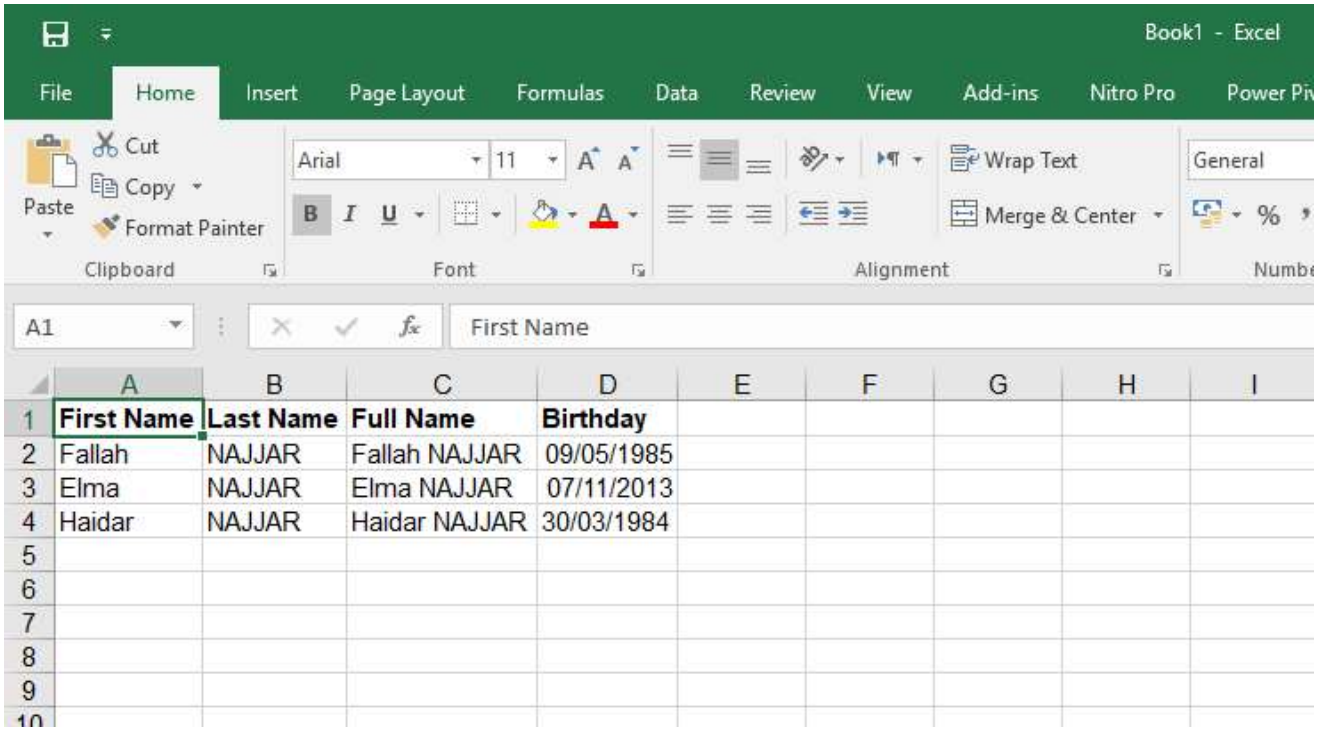
Copy the Following code to your project

```
Imports Excel = Microsoft.Office.Interop.Excel
Public Class Form1
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Dim AppEx As Excel.Application
        Dim WBEx As Excel.Workbook
        Dim SHTEx As Excel.Worksheet
        Dim RngEx As Excel.Range
        AppEx = CreateObject("Excel.Application")    ' Start Excel and get Application object.
        AppEx.Visible = True
        WBEx = AppEx.Workbooks.Add                   ' Add a new workbook.
        SHTEx = WBEx.ActiveSheet
        SHTEx.Cells(1, 1).Value = "First Name"       ' Add table headers going cell by cell.
        SHTEx.Cells(1, 2).Value = "Last Name"
        SHTEx.Cells(1, 3).Value = "Full Name"
        SHTEx.Cells(1, 4).Value = "Birthday"
        With SHTEx.Range("A1", "D1")        ' Format A1:D1 as bold, vertical alignment = center.
            .Font.Bold = True
            .VerticalAlignment = Excel.XlVAlign.xlVAlignCenter
        End With
        Dim Info(3, 2) As String         ' Create an array to set multiple values at once.
        Info(0, 0) = "Fallah"
        Info(0, 1) = "NAJJAR"
        Info(1, 0) = "Elma"
        Info(1, 1) = "NAJJAR"
        Info(2, 0) = "Haidar"
        Info(2, 1) = "NAJJAR"
        ' Fill A2:B6 with an array of values (First and Last Names).
        SHTEx.Range("A2", "B4").Value = Info
        ' Fill C2:C4 with a relative formula (=A2 & " " & B2).
        RngEx = SHTEx.Range("C2", "C4")
        RngEx.Formula = "=A2 & "" "" & B2"
        With SHTEx                          ' Fill Cell (D2:D4) values
            .Cells(2, 4).Value = "09/05/1985"
            .Cells(3, 4).Value = "07/11/2013"
            .Cells(4, 4).Value = "30/03/1984"

        End With
        ' AutoFit columns A:D.
        RngEx = SHTEx.Range("A1", "D1")
        RngEx.EntireColumn.AutoFit()
        ' Make sure Excel is visible and give the user control of Excel's lifetime.
        AppEx.Visible = True
        AppEx.UserControl = True
        RngEx = Nothing                     ' Release object references.
        SHTEx = Nothing
        WBEx = Nothing
        AppEx.Quit()
        AppEx = Nothing
        Exit Sub
Err_Handler:
        MsgBox(Err.Description, vbCritical, "Error: " & Err.Number)
    End Sub
End Class
```


fallahnajjar@yahoo.com
+9647807445833

Then run your application (app) to get



	A	B	C	D	E	F	G	H	I
1	**First Name**	**Last Name**	**Full Name**	**Birthday**					
2	Fallah	NAJJAR	Fallah NAJJAR	09/05/1985					
3	Elma	NAJJAR	Elma NAJJAR	07/11/2013					
4	Haidar	NAJJAR	Haidar NAJJAR	30/03/1984					
5									
6									
7									
8									
9									
10									

fallahnajjar@yahoo.com
+9647807445833

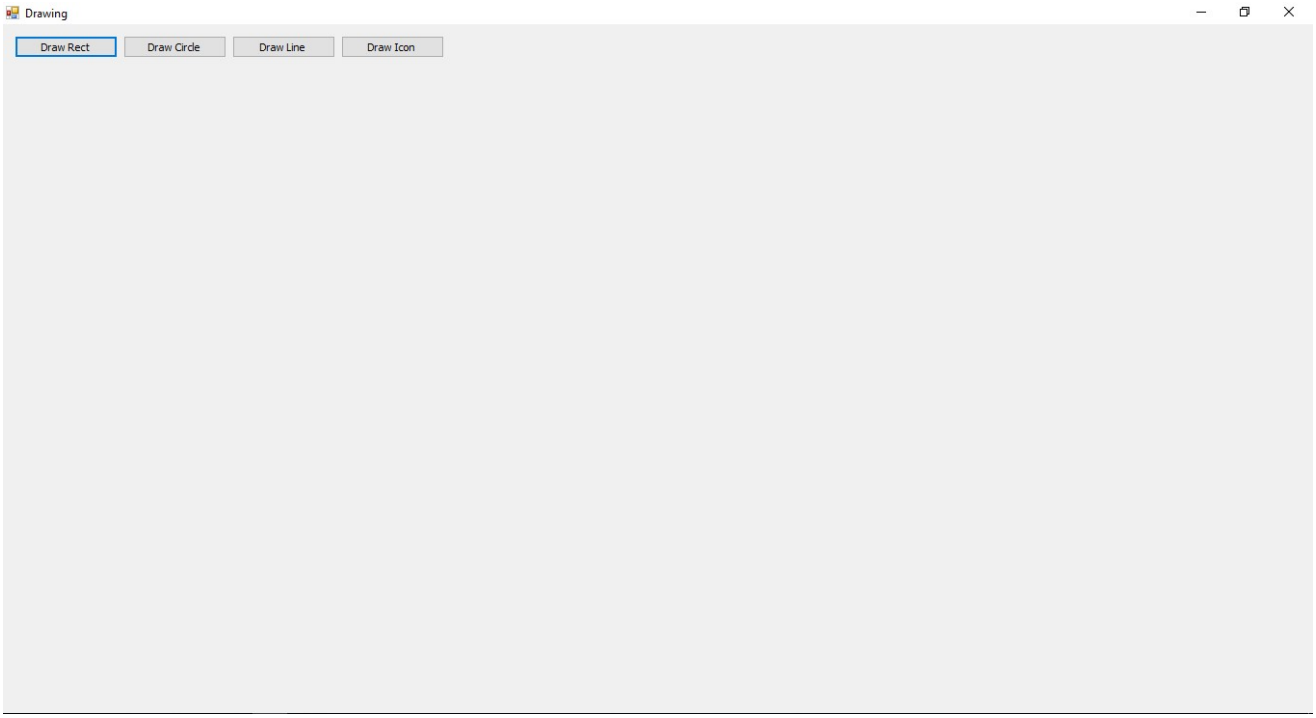
Project 61: Drawing

Add to your Form project

- ❖ 4 Buttons

Properties

Tools	Property	Set
Form1	WindowState	Maximized
Form1	Text	Drawing
Button1	Name	btnDrawRect
Button1	Text	Draw Rect
Button2	Name	btnDrawCircle
Button2	Text	Draw Circle
Button3	Name	btnDrawLine
Button3	Text	Draw Line
Button4	Name	btnDrawIcon
Button4	Text	Draw Icon



fallahnajjar@yahoo.com
+9647807445833

Source Code

Copy the Following code to your project

```
Private Sub btnDrawRect_Click(sender As Object, e As EventArgs) Handles btnDrawRect.Click
Dim rnd As New Random(), rndCol As New Random(DateTime.Now.Ticks Mod ((rnd.Next) + 1))
Dim numShapes = rnd.Next(1, 1), bIsCircleOrSquare As Boolean = (rnd.Next Mod 2)
Using g = Me.CreateGraphics()
Dim diam = rnd.Next(55, Math.Min(255, ClientSize.Width))
While numShapes > 0
Using b As New SolidBrush(Color.FromArgb(rndCol.Next(100, 256), rndCol.Next(256),
rndCol.Next(256), rndCol.Next(256)))
Dim x = rnd.Next(ClientSize.Width - diam), y = rnd.Next(ClientSize.Height - diam)
g.FillRectangle(b, x, y, diam, diam)
g.DrawRectangle(Pens.Black, x, y, diam, diam)
End Using
numShapes -= 1
End While
End Using
End Sub
```

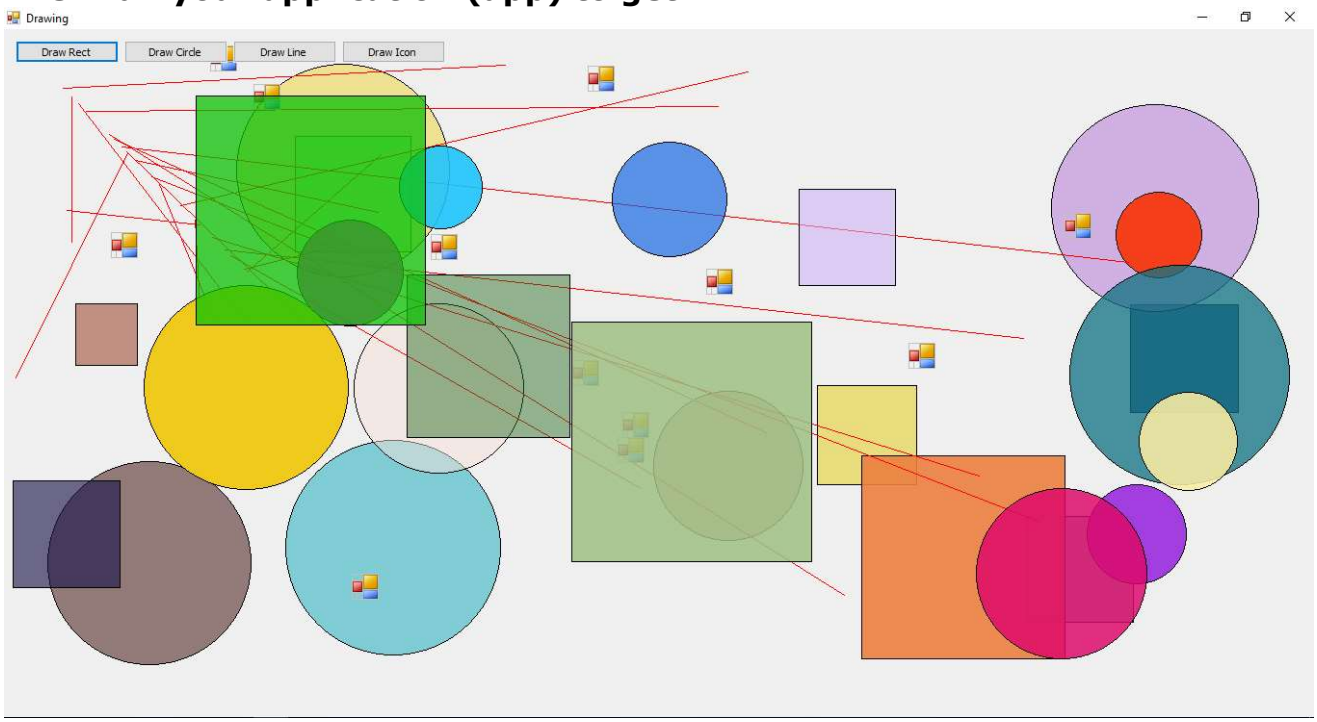
```
Private Sub btnDrawCircle_Click(sender As Object, e As EventArgs) Handles btnDrawCircle.Click
Dim rnd As New Random(), rndCol As New Random(DateTime.Now.Ticks Mod ((rnd.Next) + 1))
Dim numShapes = rnd.Next(1, 1), bIsCircleOrSquare As Boolean = (rnd.Next Mod 2)
Using g = Me.CreateGraphics()
Dim diam = rnd.Next(55, Math.Min(255, ClientSize.Width))
While numShapes > 0
Using b As New SolidBrush(Color.FromArgb(rndCol.Next(100, 256), rndCol.Next(256),
rndCol.Next(256), rndCol.Next(256)))
Dim x = rnd.Next(ClientSize.Width - diam), y = rnd.Next(ClientSize.Height - diam)
g.FillEllipse(b, x, y, diam, diam)
g.DrawEllipse(Pens.Black, x, y, diam, diam)
End Using
numShapes -= 1
End While
End Using
End Sub
```

```
Private Sub btnDrawLine_Click(sender As Object, e As EventArgs) Handles btnDrawLine.Click
Dim rnd As New Random(), rndCol As New Random(DateTime.Now.Ticks Mod ((rnd.Next) + 1))
Dim numShapes = rnd.Next(1, 1), bIsCircleOrSquare As Boolean = (rnd.Next Mod 2)
Using g = Me.CreateGraphics()
Dim diam = rnd.Next(55, Math.Min(255, ClientSize.Width))
While numShapes > 0
Using b As New SolidBrush(Color.FromArgb(rndCol.Next(100, 256), rndCol.Next(256),
rndCol.Next(256), rndCol.Next(256)))
Dim x = rnd.Next(ClientSize.Width - diam), y = rnd.Next(ClientSize.Height - diam)
g.DrawLine(Pens.Red, x, y, diam, diam)
End Using
numShapes -= 1
End While
End Using
End Sub
```


fallahnajjar@yahoo.com
+9647807445833

```
Private Sub btnDrawIcon_Click(sender As Object, e As EventArgs) Handles btnDrawIcon.Click
Dim rnd As New Random(), rndCol As New Random(DateTime.Now.Ticks Mod ((rnd.Next) + 1))
Dim numShapes = rnd.Next(1, 1), bIsCircleOrSquare As Boolean = (rnd.Next Mod 2)
Dim icon1 As Icon = Me.Icon
Using g = Me.CreateGraphics()
Dim diam = rnd.Next(55, Math.Min(255, ClientSize.Width))
While numShapes > 0
Using b As New SolidBrush(Color.FromArgb(rndCol.Next(100, 256), rndCol.Next(256),
rndCol.Next(256), rndCol.Next(256)))
Dim x = rnd.Next(ClientSize.Width - diam), y = rnd.Next(ClientSize.Height - diam)
g.DrawIcon(icon1, x, y)
End Using
numShapes -= 1
End While
End Using
End Sub
```

Then run your application (app) to get



fallahnajjar@yahoo.com
+9647807445833

Project 62: ProgressBarEx

Add to your Form project

- ProgressBarEx

You can download the ProgressBarEx.DLL from here and add it to your project as follow:

Right click on Toolbox → Choose Items → COM Components → Browse → ProgressBarEx.DLL

- Button

Properties

Tools	Property	Set
ProgressBarEx1	Background Color	Goldenrod
ProgressBarEx1	Border Color	DarkRed
ProgressBarEx1	Progress Color	Lime
ProgressBarEx1	Fore Color	MediumVioletRed
ProgressBarEx1	Show Percentage	True
ProgressBarEx1	Show Text	True
ProgressBarEx1	Text	Progress
Button1	Name	btnStart
Button1	Text	Start

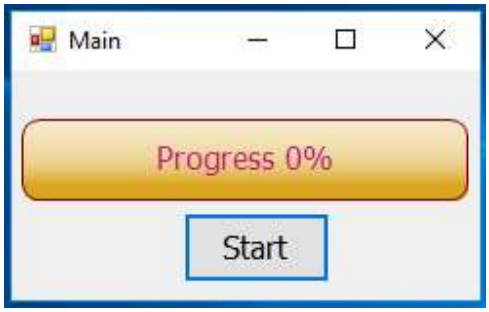
Source Code

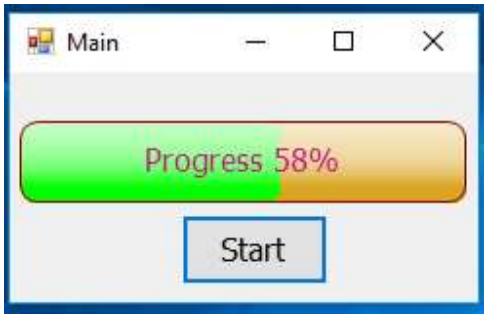
Double click (DC) on the Form, then replase all with the following code

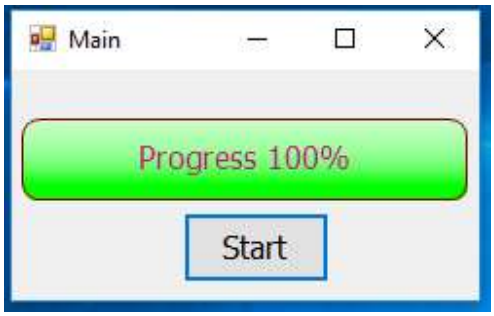
```
Public Class frmMain
    Private WithEvents tmr As New Timer With {.Interval = 75}
    Private Sub tmr_Tick(ByVal sender As Object, ByVal e As System.EventArgs) Handles tmr.Tick
        ProgressBarEx1.Value += 1
        If ProgressBarEx1.Value = 100 Then
            tmr.Stop()
        End If
    End Sub
    Private Sub btnStart_Click(sender As Object, e As EventArgs) Handles btnStart.Click
        ProgressBarEx1.Value = 0
        tmr.Start()
    End Sub
End Class
```


fallahnajjar@yahoo.com
+9647807445833

Then run your application (app) to get







fallahnajjar@yahoo.com
+9647807445833

About Me

Fallah H. N. NAJJAR 09/05/1985

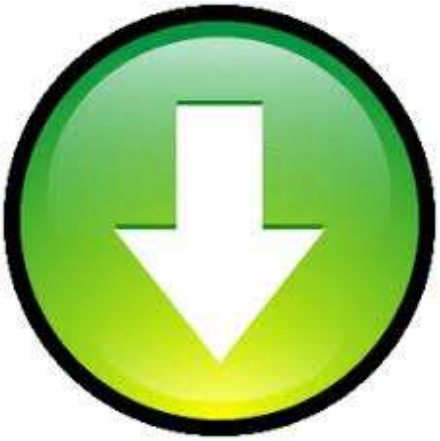
Diploma Computer Diploma / Technical Institute / Najaf

B.Sc. Software Engineer / Imam Jafar Al-Sadiq University

M.Sc. Computer Science / Kufa University

With greetings / Eng. Fallah Najjar

download each project from the download picture above each project name
or download all projects from below



fallahnajjar@yahoo.com
+9647807445833