

.NET PROGRAMMING (C#)

(18 MCA 4 2 C)

UNIT – IV

WINDOW BASED PROGRAMMING

FACULTY:

Dr. K. Arthi

Assistant Professor,

Post Graduate And Research Department Of Computer Applications,

Government Arts College (Autonomous), Coimbatore 641 018.

CONTENT

- **Window based Programming**
 - **Win-Forms**
 - **Text Box**
 - **Buttons**
 - **Message Box**
 - **List Box**
 - ~~**Handling Events**~~




WIN-FORMS



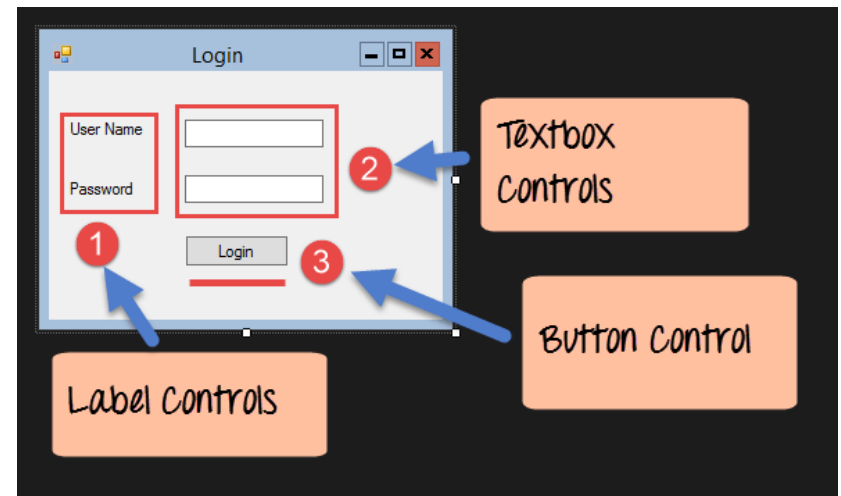
INTRODUCTION



- Windows Forms is a Graphical User Interface(GUI) class library which is bundled in .NET Framework.
- Its main purpose is to provide an easier interface to develop the applications for desktop, tablet, PCs. It is also termed as the WinForms.

- 
- The applications which are developed by using Windows Forms or WinForms are known as the Windows Forms Applications that runs on the desktop computer.
 - WinForms can be used only to develop the Windows Forms Applications not web applications.
 - WinForms applications can contain the different type of controls like labels, list boxes, tooltip etc.

EXAMPLE OF A SIMPLE WINDOWS FORM APPLICATION

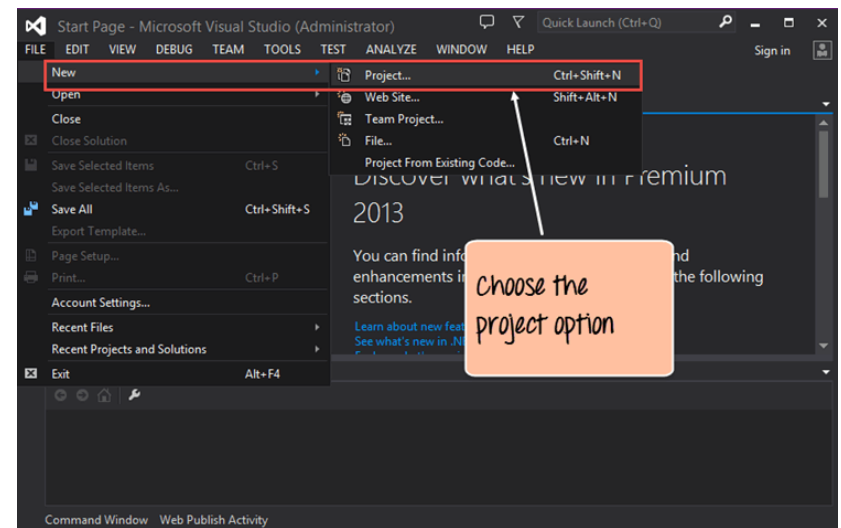
- It shows a simple Login screen, which is accessible by the user.



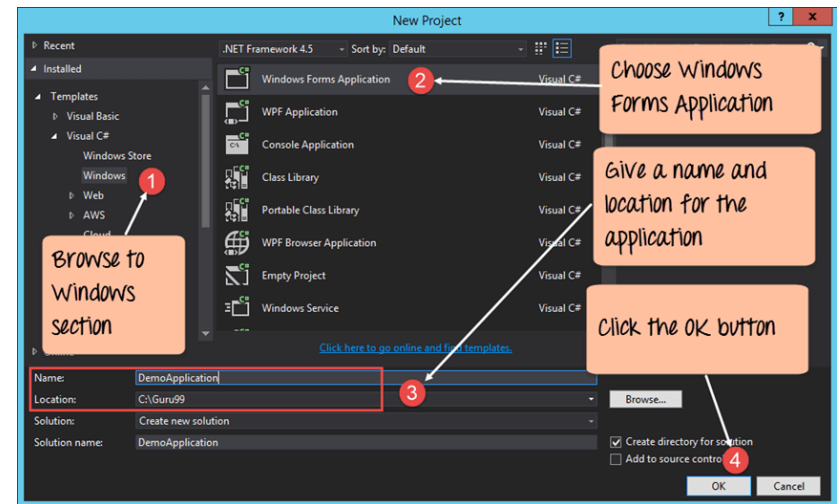
- 
- 
- This is a collection of label controls which are normally used to describe adjacent controls.
 - we have 2 textboxes, and the labels are used to tell the user that one textbox is for entering the user name and the other for the password.
 - The 2 textboxes are used to hold the username and password which will be entered by the user.
 - We have the button control.
 - The button control will normally have some code attached to perform a certain set of actions.
 - The button perform an action of validating the user name and password which is entered by the user.

EXAMPLE OF HOW WE CAN IMPLEMENT A SIMPLE 'HELLO WORLD' APPLICATION IN VISUAL STUDIO.

- Step 1) Creation of a new project in Visual Studio. Choose the menu option New->Project.

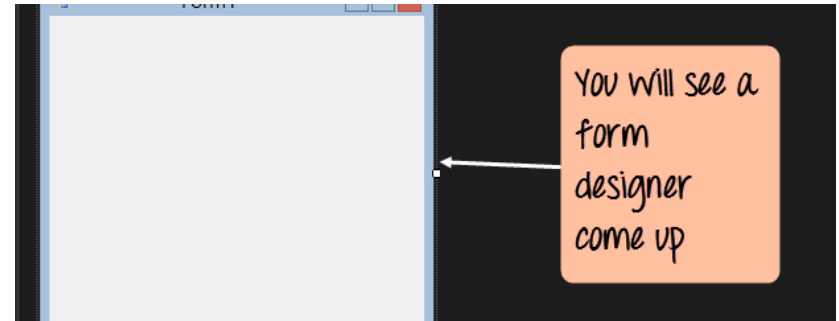


- Step 2) The next step is to choose the project type as a Windows Forms application. Here we also need to mention the name and location of our project.

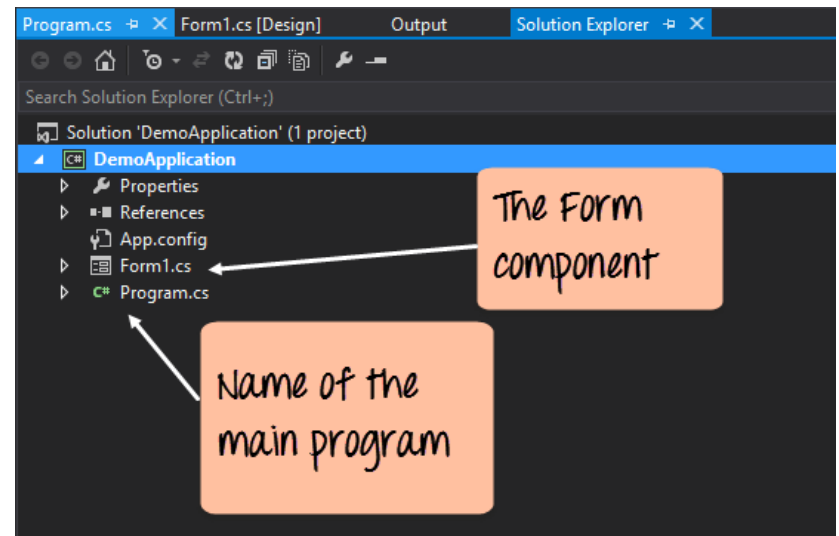


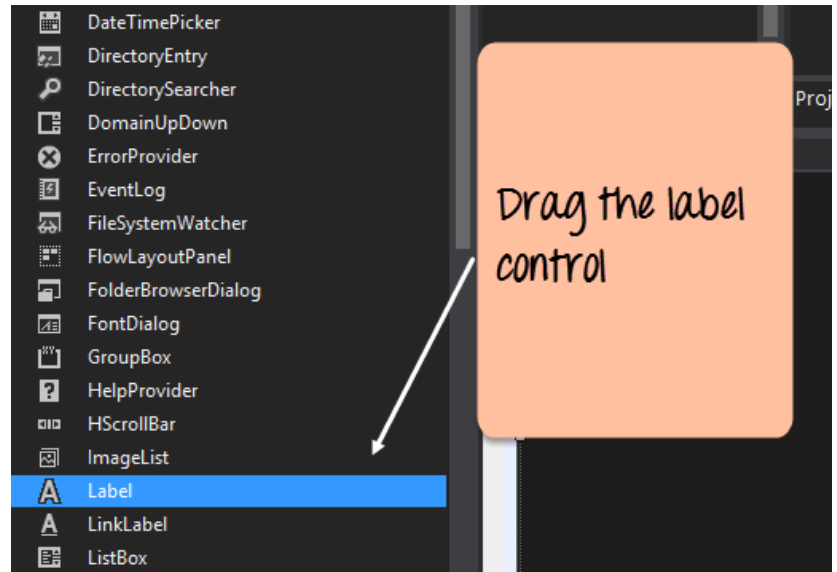
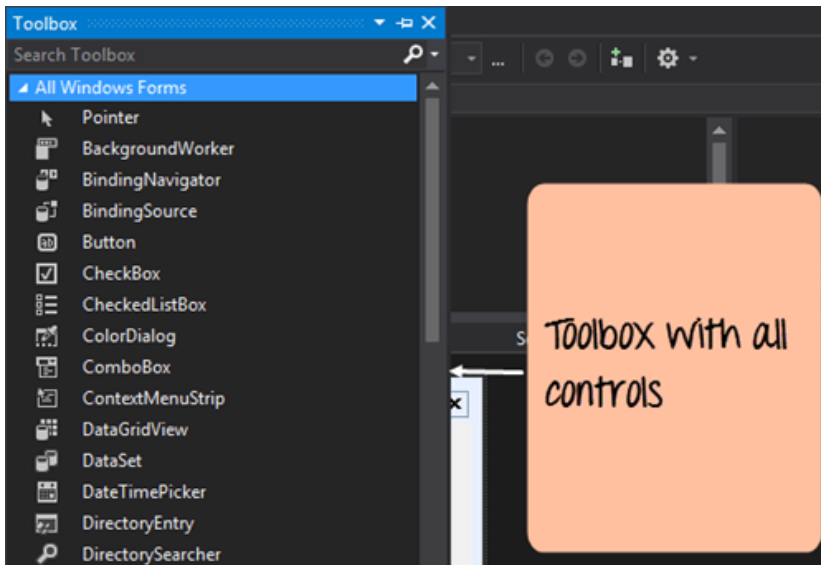
OUTPUT:

- You will see a Form Designer displayed in Visual Studio. It's in this Form Designer that you will start building your Windows Forms application.

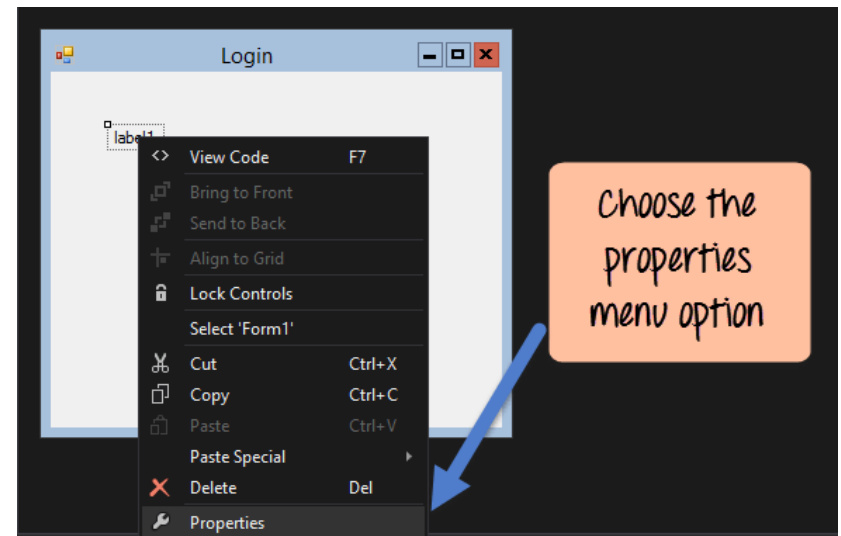
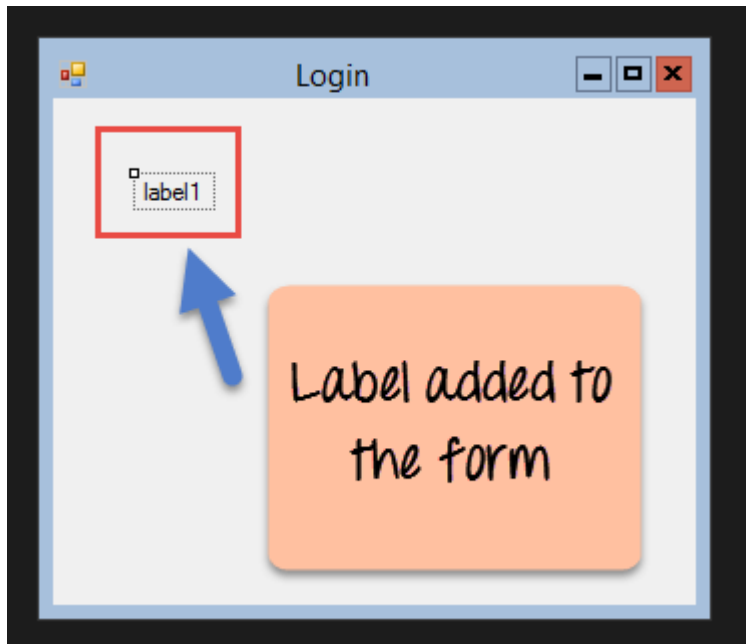


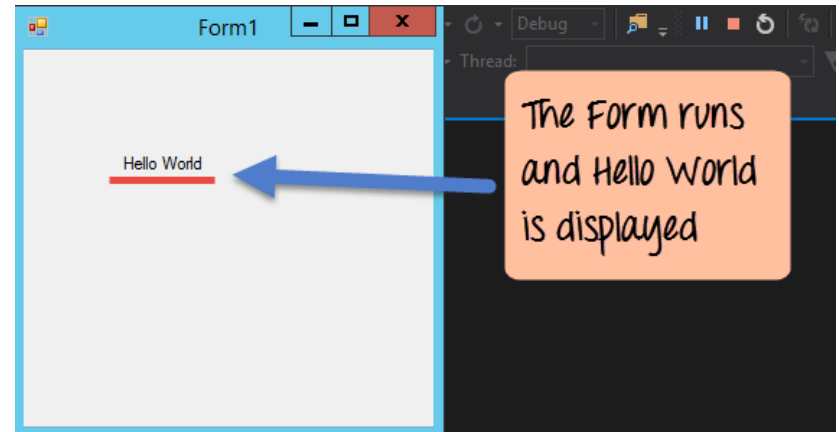
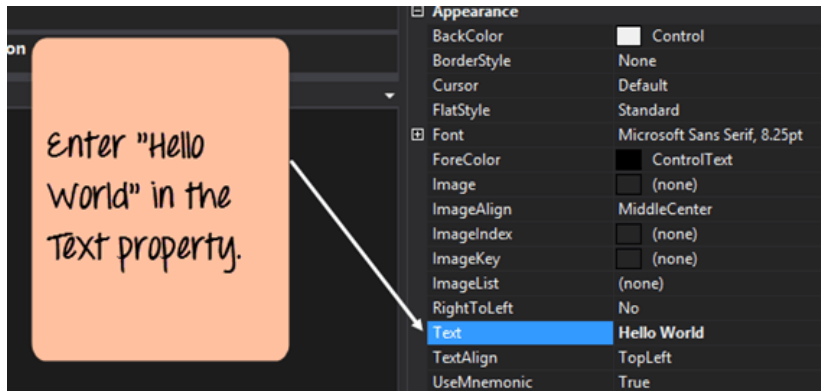
- A Form application called Form1.cs. This file will contain all of the code for the Windows Form application.
- The Main program called Program.cs is default code file which is created when a new application is created in Visual Studio. This code will contain the startup code for the application as a whole.





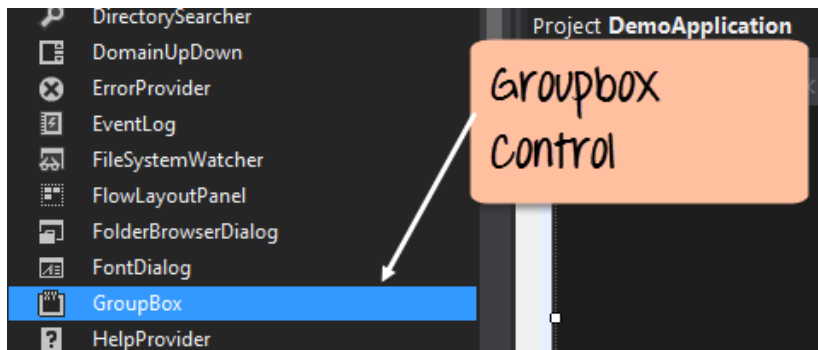
STEP 4) THE NEXT STEP IS TO GO TO THE PROPERTIES OF THE CONTROL AND CHANGE THE TEXT TO 'HELLO WORLD'.



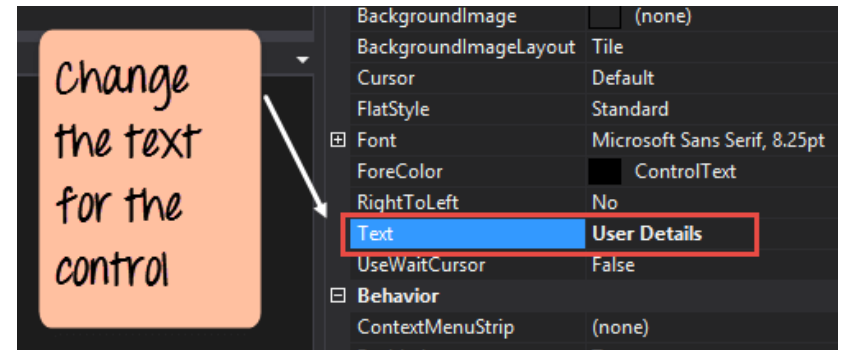


GROUP BOX

Step 1) The first step is to drag the Groupbox control onto the Windows Form from the toolbox as shown below



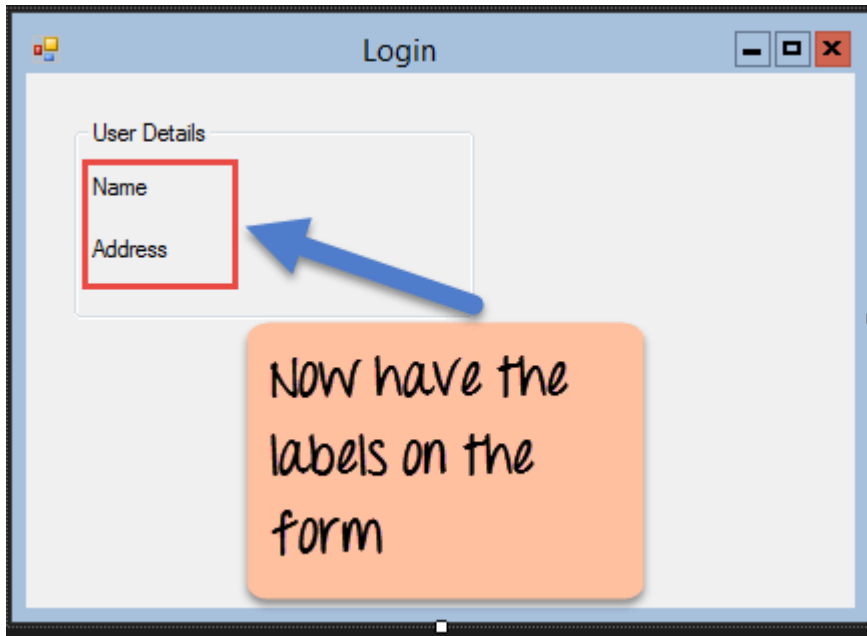
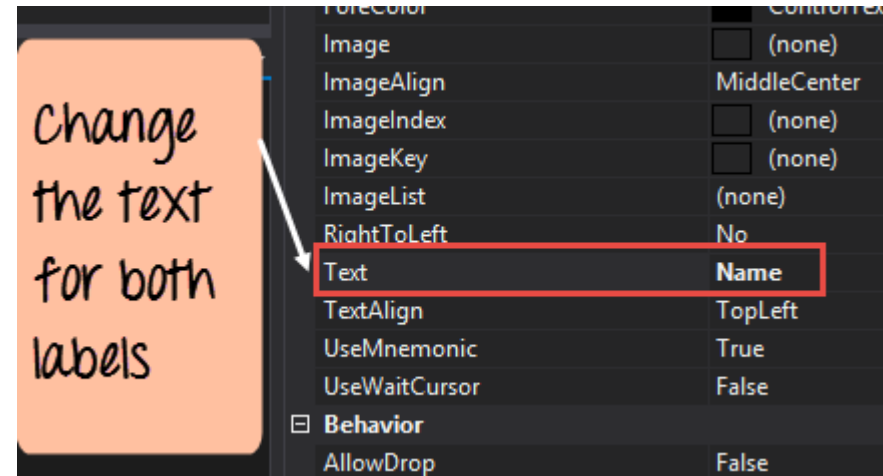
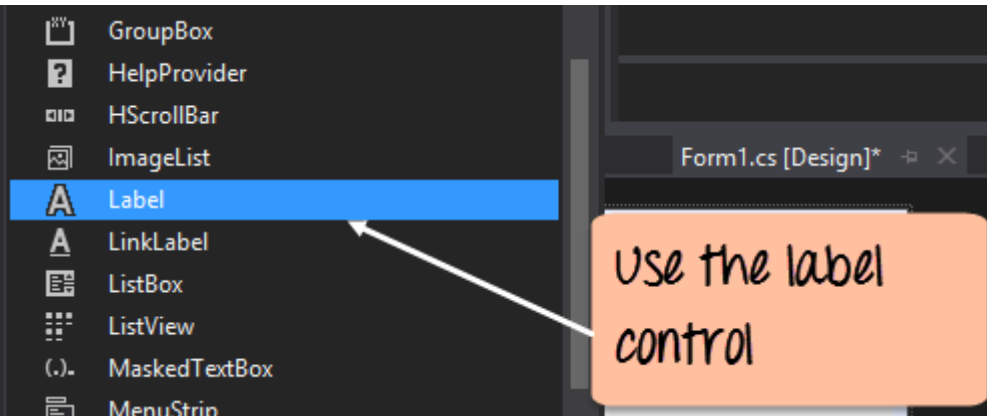
Step 2) The groupbox has been added, go to the properties window clicking the groupbox control. In the properties window, go to the Text property and change it to "User Details".





The image shows a screenshot of a software application window titled "Login". The window has a light blue title bar with standard minimize, maximize, and close buttons. Inside the window, there is a light grey area containing a white rectangular box with a red border. This box is labeled "User Details" at its top-left corner. To the right of the window, there is a handwritten note on an orange background that says "GroupBOX on the screen". A blue arrow points from this note to the red-bordered box.

Label Control



Textbox

A screenshot of the Visual Studio Properties window for a text box control. The window is divided into several sections: Data, Design, Focus, and Layout. The Design section is expanded, and the (Name) property is highlighted with a red box, showing the value txtName. An orange callout box with the text "Name to the textbox" has an arrow pointing to the (Name) property.

WordWrap	True
Data	
(ApplicationSettings)	
(DataBindings)	
Tag	
Design	
(Name)	txtName
GenerateMember	True
Locked	False
Modifiers	Private
Focus	
CausesValidation	True
Layout	

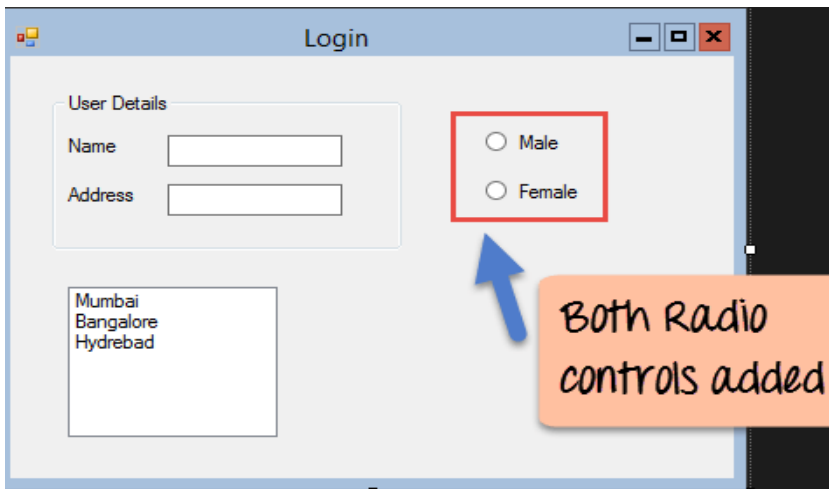
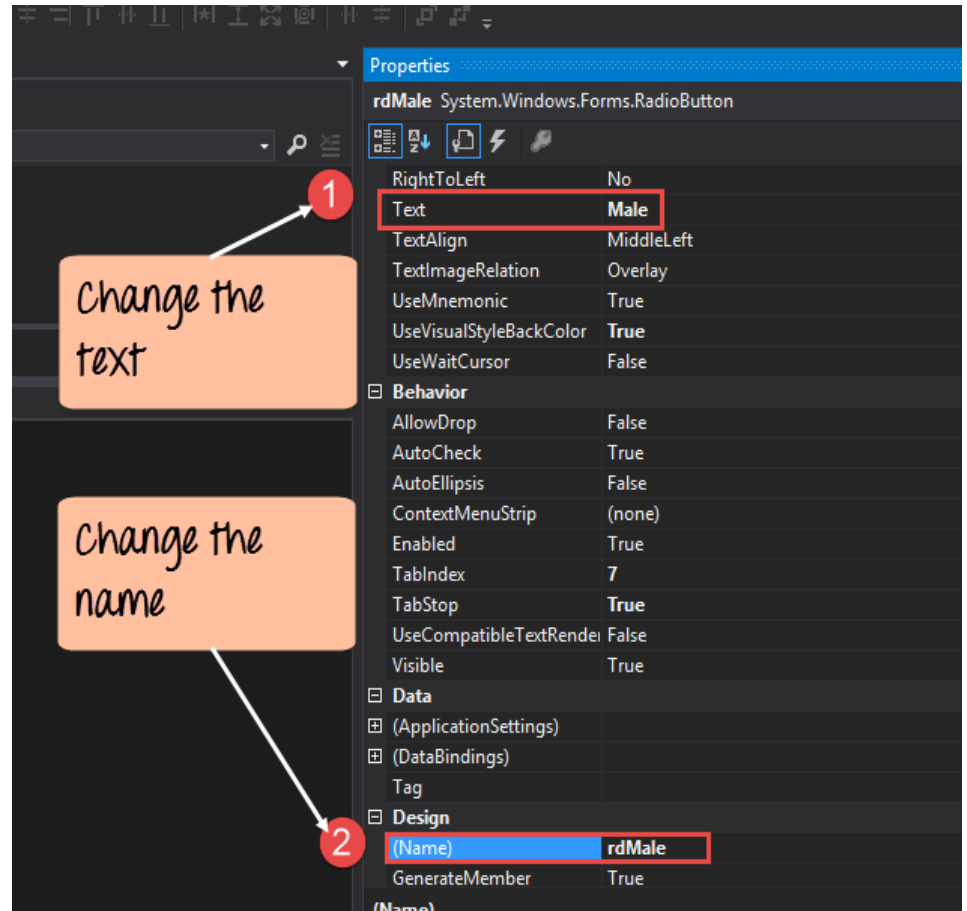
A screenshot of a Windows form titled "Login". The form contains a "User Details" group box with two text boxes: "Name" and "Address". The "Name" text box is highlighted with a red border. An orange callout box with the text "text box controls on the form" has an arrow pointing to the "Name" text box.

List box

The image illustrates the process of configuring a list box in a WPF application. It shows three main components:

- String Collection Editor:** A dialog box where city names are entered. The list contains "Mumbai", "Bangalore", and "Hydrebad". A red box highlights the list, and a blue arrow points to it with the number 3. A note says "Enter the city names".
- Properties Window:** Shows the configuration for the list box. The "Items" property is set to "(Collection)". A red box highlights this property, and a blue arrow points to it with the number 2. A note says "Click on the items property". The "Name" property is set to "IstCity", highlighted with a red box and a blue arrow with the number 1. A note says "Change the name of the control".
- Login Window:** Shows the final result. The list box contains the city names "Mumbai", "Bangalore", and "Hydrebad". A red box highlights the list box, and a blue arrow points to it with the number 4. A note says "The listbox with values".

RadioButton



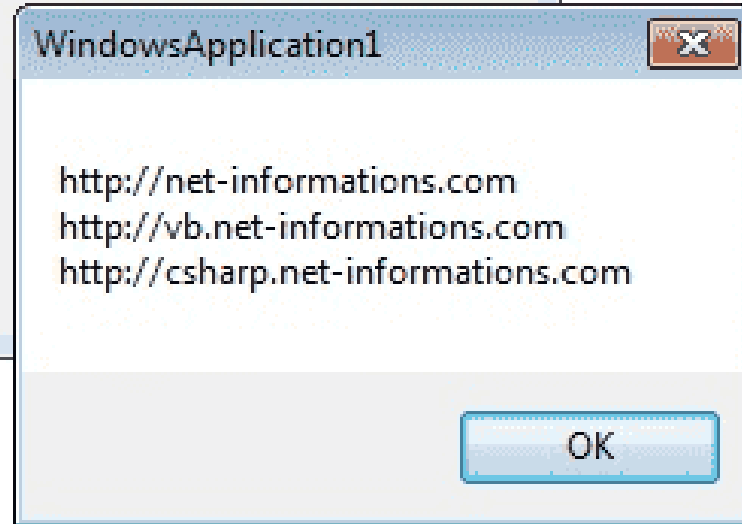
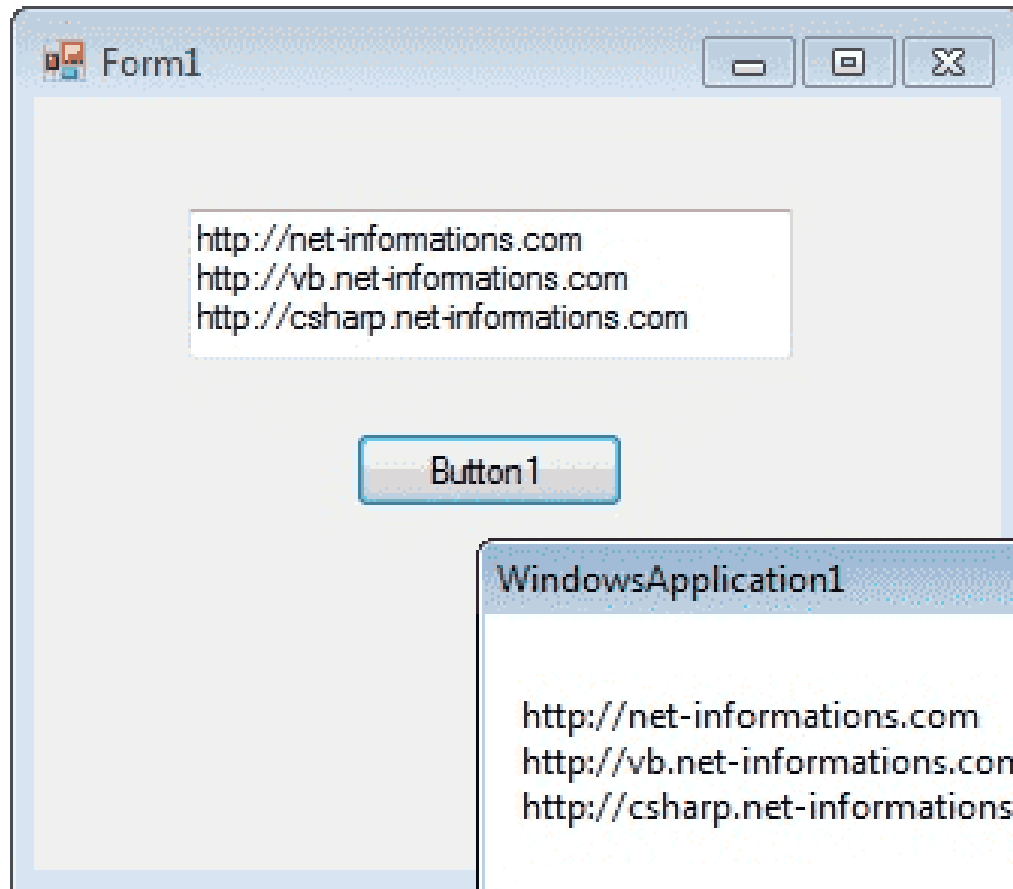



TEXT BOX




TEXTBOX

- A Textbox control is used to display, or accept as input, a single line of text.
- This control has additional functionality that is not found in the standard Windows text box control, including multiline editing and password character masking.



- 
- A text box object is used to display text on a form or to get user input while a C# program is running.
 - In a text box, a user can type data or paste it into the control from the clipboard.
 - For displaying a text in a TextBox control , you can code like this.
 - `textBox1.Text = "VisualStudio.Microsoft.com";`

- 
- You can also collect the input value from a `TextBox` control to a variable like this way.
 - `string var;`
 - `var = textBox1.Text;`

THERE ARE DIFFERENT TYPES IN TEXTBOX

-
- Textbox Properties
- Textbox BorderStyle
- Textbox Event
- Textbox ChangedEvent
- Textbox ReadOnly
- Textbox Multiline
- Textbox Password Character

TEXTBOX PROPERTIES

- You can set TextBox properties through Property window or through program.
- You can open Properties window by pressing F4 or right click on a control and select Properties menu item.


Properties



Form1 System.Windows.Forms.Form



RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	300, 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocati
Tag	
Text	Form1
TopMost	False
TransparencyKey	<input type="checkbox"/>
UseWaitCursor	False

- 
- The below code set a textbox width as 250 and height as 50 through source code.
 - `textBox1.Width = 250;`
 - `textBox1.Height = 50;`
 - Background Color and Foreground Color
 - You can set background color and foreground color through property window and programmatically.
 - `textBox1.BackColor = Color.Blue;`
 - `textBox1.ForeColor = Color.White;`

TEXTBOX BORDERSTYLE

- You can set 3 different types of border style for textbox, they are None, FixedSingle and fixed3d.
 - `textBox1.BorderStyle = BorderStyle.Fixed3D;`

TEXTBOX EVENTS

```
private void textBox1_KeyDown(object sender, KeyEventArgs e)
{
    if (e.KeyCode == Keys.Enter)
    {
        MessageBox.Show("You press Enter Key");
    }
    if (e.KeyCode == Keys.CapsLock)
    {
        MessageBox.Show("You press Caps Lock Key");
    }
}
```

TEXTCHANGED EVENT

- When user input or setting the Text property to a new value raises the

```
private void textBox1_TextChanged(object sender, EventArgs e)
{
    label1.Text = textBox1.Text;
}
```


TEXTBOX MAXIMUM LENGTH

- Sets the maximum number of characters or words the user can input into the text box control.
 - `textBox1.MaxLength = 40;`

TEXTBOX READONLY

- When a program wants to prevent a user from changing the text that appears in a text box, the program can set the controls Read-only property is to True.
 - `textBox1.ReadOnly = true;`

TEXTBOX MULTILINE

- You can use the Multiline and ScrollBars properties to enable multiple lines of text to be displayed or entered.
 - `textBox1.Multiline = true;`

TEXTBOX PASSWORD CHARACTER

- TextBox controls can also be used to accept passwords and other sensitive information.
- You can use the PasswordChar property to mask characters entered in a single line version of the control
 - `textBox1.PasswordChar = '*';`
- The above code set the PasswordChar to * ,so when the user enter password then it display only * instead of typed characters.

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            textBox1.Width = 250;
            textBox1.Height = 50;
            textBox1.Multiline = true;
            textBox1.BackColor = Color.Blue;
            textBox1.ForeColor = Color.White;
            textBox1.BorderStyle = BorderStyle.Fixed3D;
        }
        private void button1_Click(object sender, EventArgs e)
        {
            string var;
            var = textBox1.Text;
            MessageBox.Show(var);
        }
    }
}
```

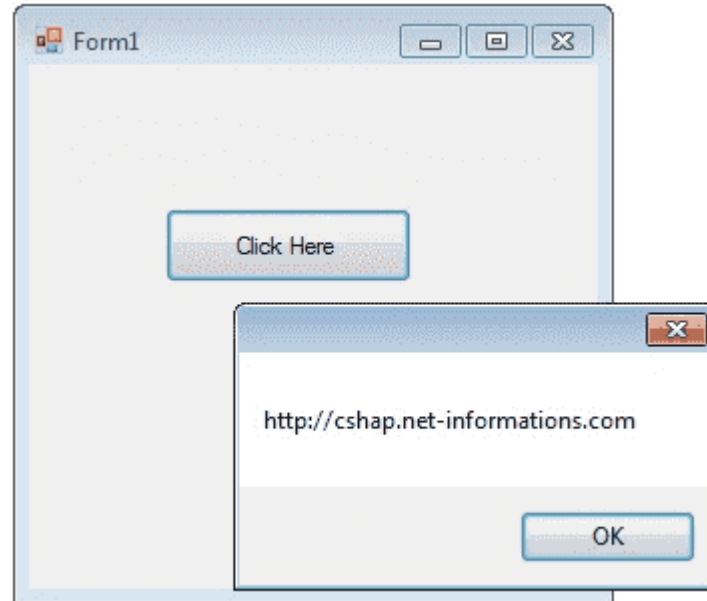



BUTTONS



C# BUTTON CONTROL

- Windows Forms controls are reusable components that encapsulate user interface functionality and are used in client side Windows applications.
- A button is a control, which is an interactive component that enables users to communicate with an application.
- The Button class inherits directly from the ButtonBase class.
- A Button can be clicked by using the mouse, ENTER key, or SPACEBAR if the button has focus.




- 
- When you want to change display text of the Button , you can change the Text property of the button.
 - `button1.Text = "Click Here";`
 - Similarly if you want to load an Image to a Button control , you can code like this.
 - `button1.Image = Image.FromFile("C:\\testimage.jpg");`
 - The following C# source code shows how to change the button Text property while Form loading event and to display a message box when pressing a Button Control.

```
using System;
using System.Drawing;
using System.Windows.Forms;
namespace WindowsFormsApplication1
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }
        private void Form1_Load(object sender, EventArgs e)
        {
            button1.Text = "Click Here";
        }
        private void button1_Click(object sender, EventArgs e)
        {
            MessageBox.Show("http://cshap.net-informations.com");
        }
    }
}
```



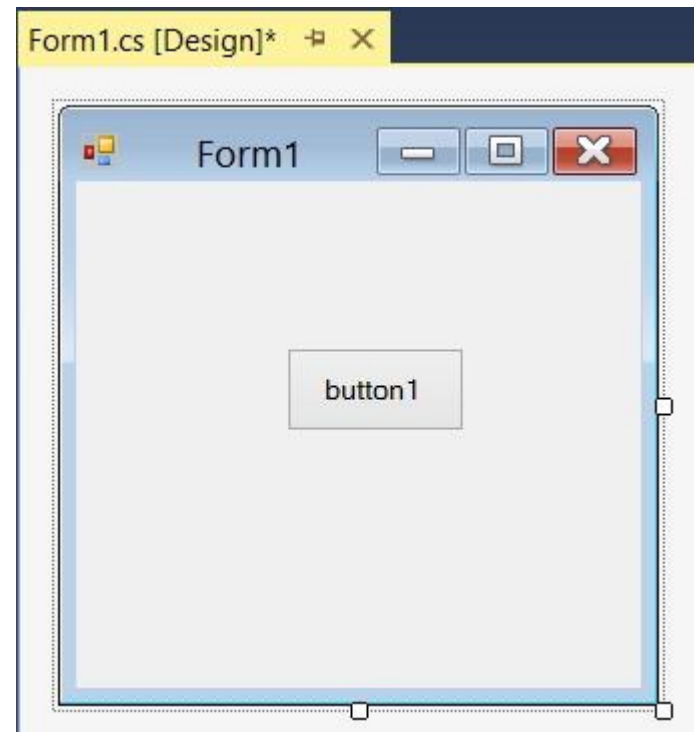
MESSAGEBOX



- 
- MessageBox is a class in C# and Show is a method that displays a message in a small window in the center of the Form.
 - MessageBox is used to provide confirmations of a task being done or to provide warnings before a task is done.

MESSAGEBOX.SHOW():

- Create a Windows Forms app in Visual Studio and add a button on it.



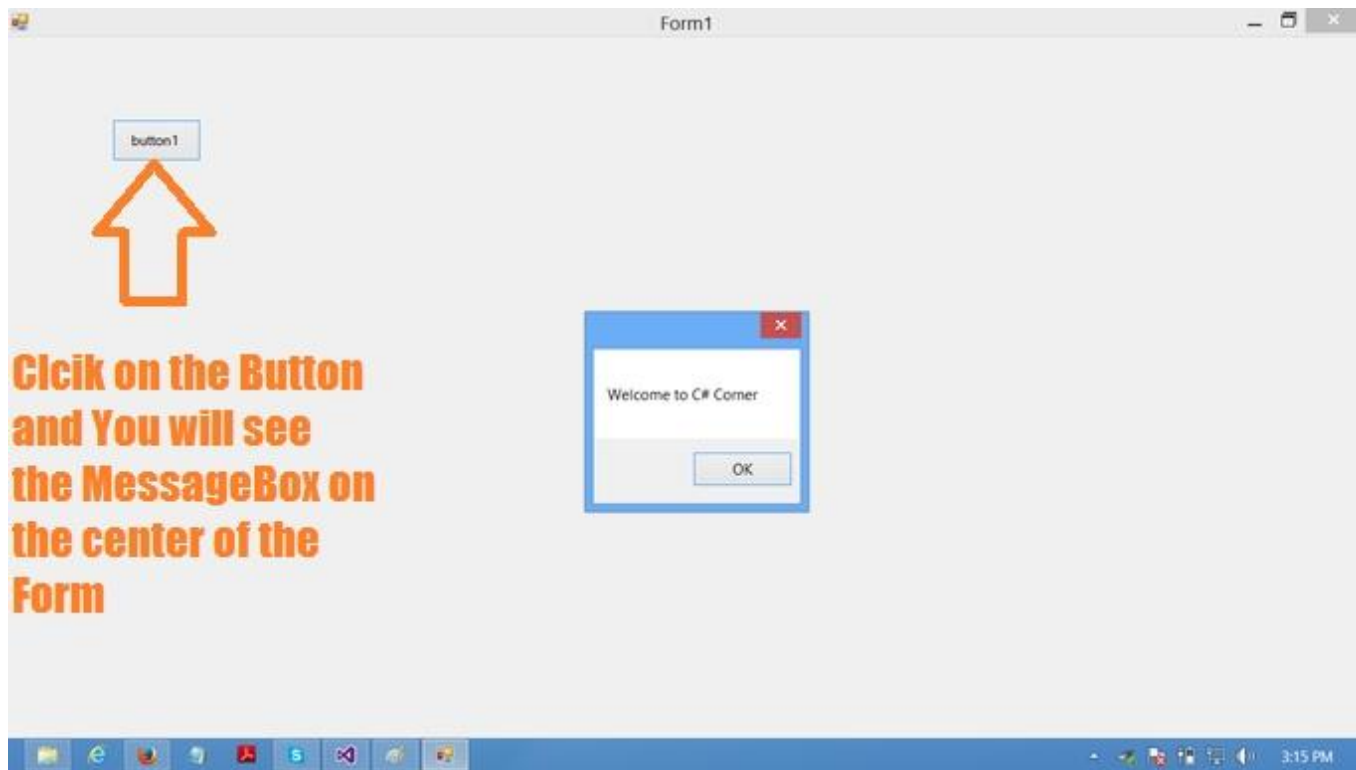
MESSAGE BOX CODING:

Form1.cs:

```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Welcome to C# corner");
}
```

▲ 1 of 21 ▼ System.Windows.Forms.DialogResult MessageBox.Show(**string text**)
Displays a message box with specified text.
text: The text to display in the message box.

OUTPUT:



MESSAGEBOX WITH TITLE

- Right click that form ->go to properties->events ->closing event->enter
- `string message = "Simple MessageBox";`
- `string title = "Title";`
- `MessageBox.Show(message, title);`
- (or)
- `MessageBox.Show("simple message", "title");`

OUTPUT:



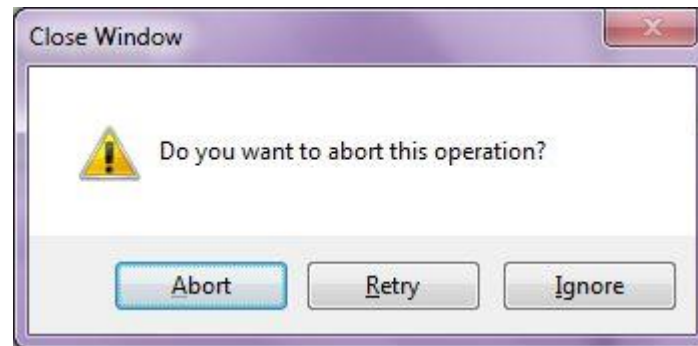
MESSAGEBOX WITH BUTTONS:


- A MessageBox can have different button combinations such as YesNo and OKCancel.
- OK->default
- OKCancel
- AbortRetryIgnore
- YesNoCancel
- YesNo
- RetryCancel

MESSAGEBOX WITH ICON:

- A MessageBox can display an icon on the dialog.
- None
- Hand
- Question
- Exclamation
- Asterisk
- Stop
- Error
- Warning
- Information
- `MessageBox.show("do you want to abort this operation", "close window", messageBoxButtons.abortretryignore, messageBoxIcon.warning);`

EG :



- 
- `DialogResult res=MessageBox.show(“do you want to abort this operation”,“close window”,messageBoxButtons.yesno, messageBoxIcon.warning);`
 - `If(res=DialogResult.No)`
 - `{`
 - `e.cancel=true;`
 - `}`



LIST BOX

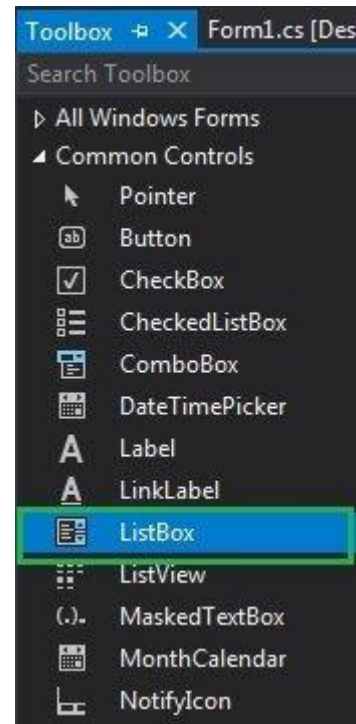


LISTBOX

- The ListBox represents a Windows control to display a list of items to a user. A user can select an item from the list. It allows the programmer to add items at design time by using the properties window or at the runtime.
- You can set this property in two different ways:
 - Design time
 - Run time

DESIGN TIME:

- Drag the ListBox control from the ToolBox and drop it on the windows form.



Properties

listBox1 System.Windows.Forms.ListBox

HorizontalScrollbar False

ImeMode NoControl

IntegralHeight True

ItemHeight 13

Items (Collection)

Location 478, 29

Locked False

Margin 3, 3, 3, 3

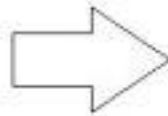
MaximumSize 0, 0

MinimumSize 0, 0

Modifiers Private

MultiColumn False

RightToLeft No



String Collection Editor

Enter the strings in the collection (one per line):

Welcome

to


Geeks

for

Geeks

OK Cancel

OUTPUT:



Welcome
to
Geeks
for
Geeks

RUN TIME:

```
ListBox mylist = new ListBox();  
    mylist.Location = new Point(287, 109);  
    mylist.Size = new Size(120, 95);  
    mylist.BorderStyle = BorderStyle.Fixed3D;  
    mylist.Items.Add("Geeks");  
    mylist.Items.Add("GFG");  
    mylist.Items.Add("GeeksForGeeks");  
    mylist.Items.Add("gfg");
```

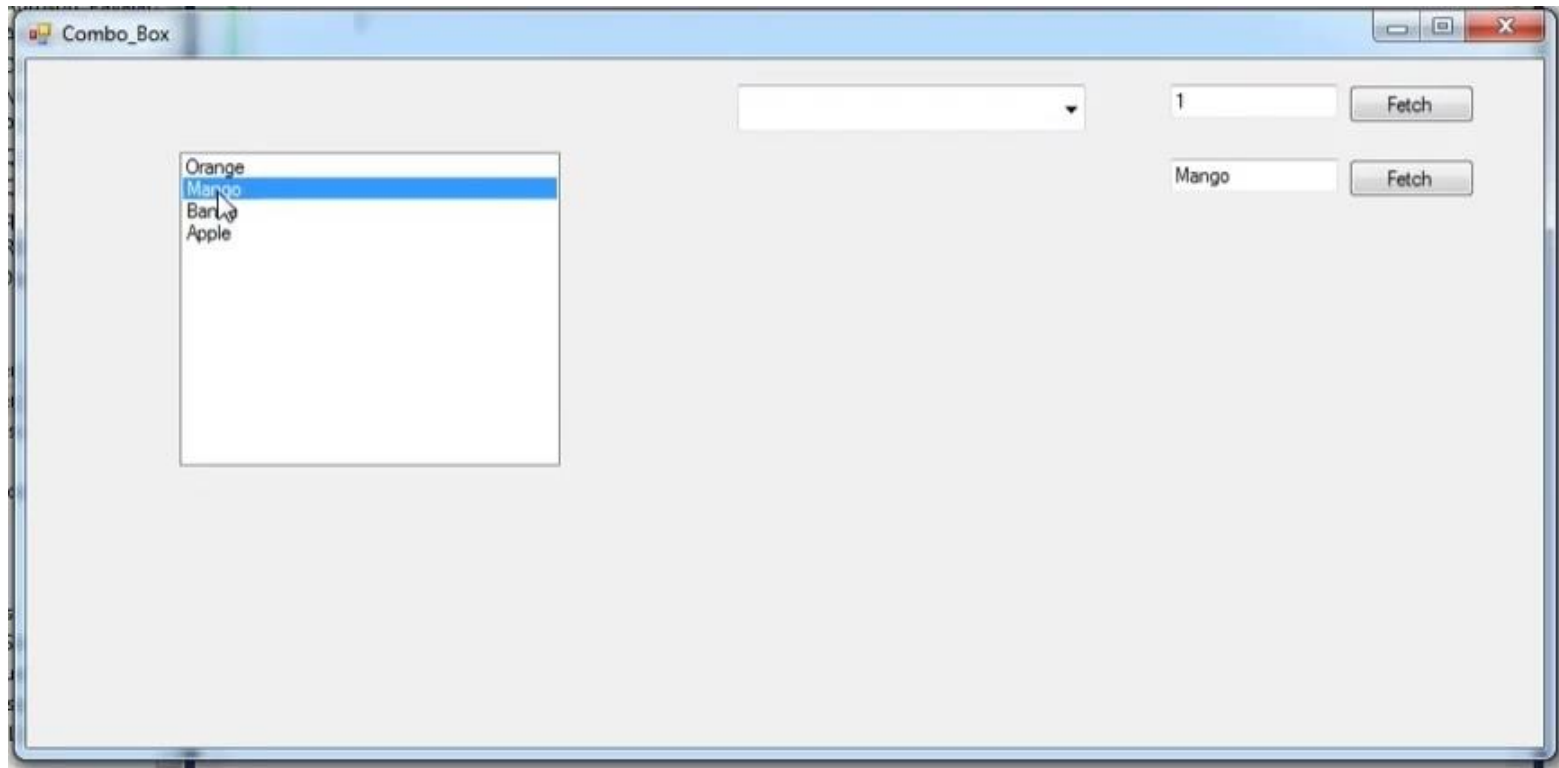
OUTPUT:

```
Geeks
GFG
GeeksForGeeks
gfg
```

ADD AND FETCH ITEMS IN RUNTIME:

- Click->form
 - `Listbox1.items.clear();`
 - `Listbox1.items.add("orange");`
 - `Listbox1.items.add("mango");`
 - `Listbox1.items.add("banana");`
 - `Listbox1.items.add("Apple");`
- Click listbox->coding
 - `textbox1.text=Listbox1.selectedIndex.ToString();`
 - `textbox2.text=Listbox1.text;`

OUTPUT:





THANKYOU

THE CONTENTS IN THIS E-MATERIAL IS TAKEN FROM THE TEXTBOOKS AND
REFERENCE BOOKS GIVEN IN THE SYLLABUS

