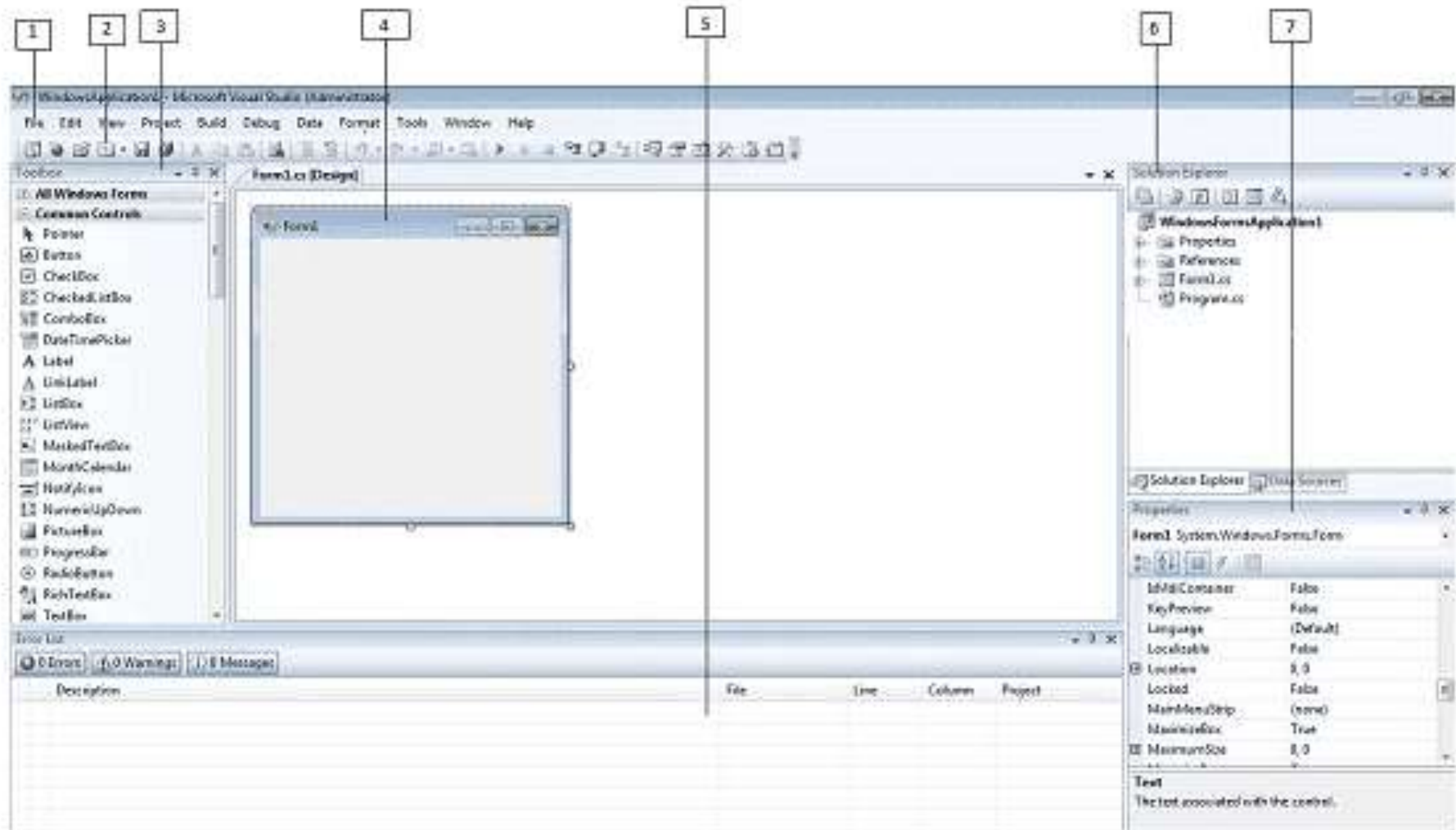


# Window Controls in C#.Net



1. Menu Bar
2. Standard Toolbar
3. ToolBox
4. Forms Designer
5. Output Window
6. Solution Explorer
7. Properties Window

# MessageBox Class

# MessageBox Class

- The `System.Windows.Forms.MessageBox` is a static class that is used to show message boxes for prompting, confirmation and warning users.
- To show a message box, simply call the `Show` method of the `MessageBox` class.
- The simplest version of the `Show` method is the one that accepts a string message as an argument.

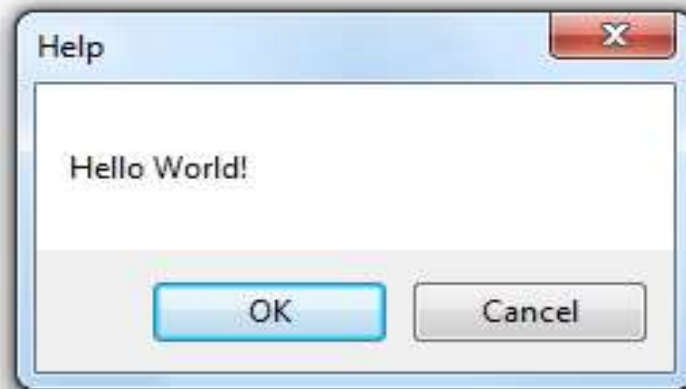
```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello World!");
}
```



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



```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Hello World!", "Help", MessageBoxButtons.OKCancel);
}
```







- The table below shows the members of the MessageBoxButtons enumeration.

Member	Buttons Shown
AbortRetryIgnore	Abort, Retry, Ignore
OK	OK
OKCancel	OK, Cancel
RetryCancel	Retry, Cancel
YesNo	Yes, No
YesNoCancel	Yes, No, Cancel



```
MessageBox.Show("Hello World!", "Help", MessageBoxButtons.OK, MessageBoxIcon.Information);
```



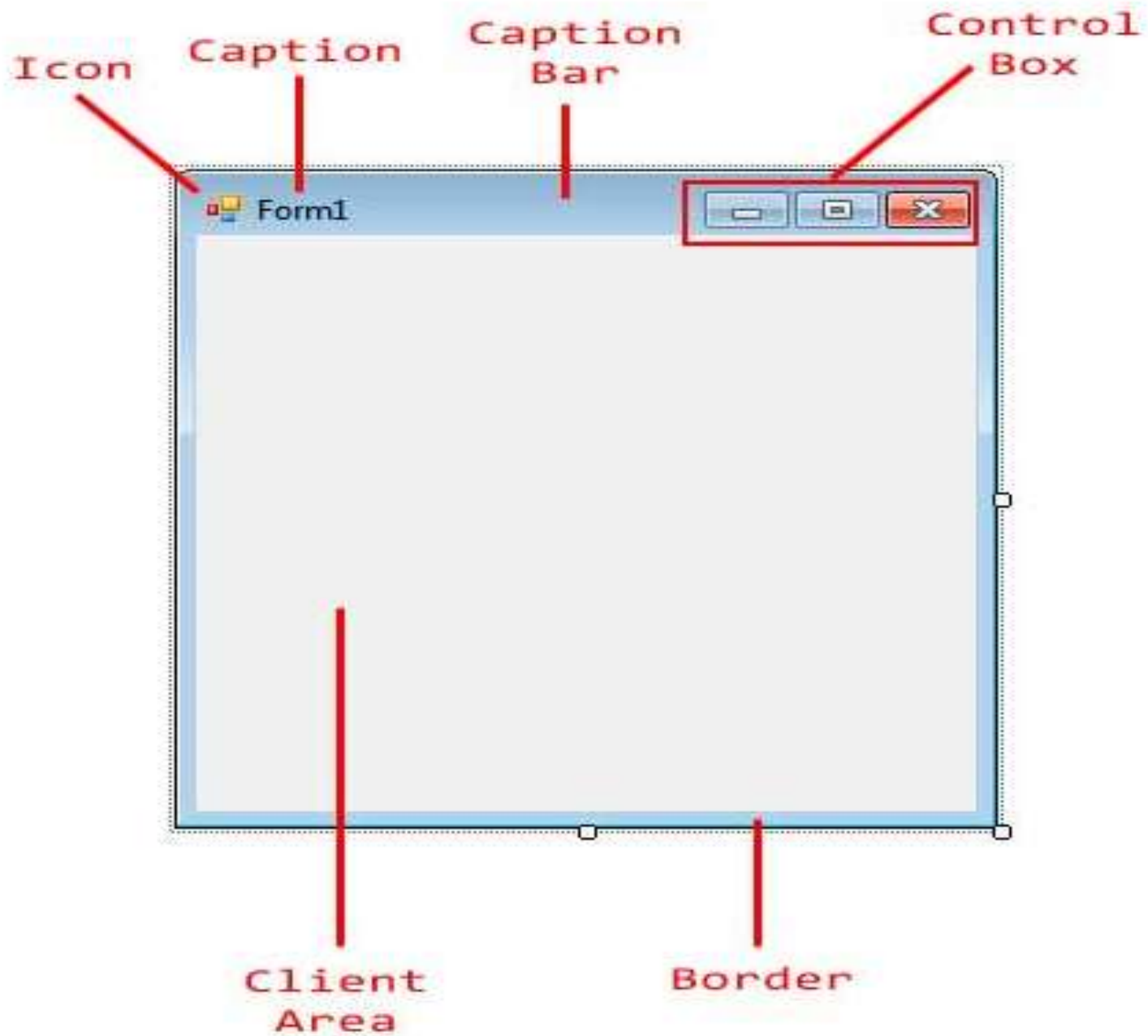
Icon	Member	Usage
	Asterisk Information	Used when showing information to the user.
	Error Hand Stop	Used when showing error messages.
	Exclamation Warning	Used when showing warning messages.
	Question	Used when asking a question to the user.

# Form In C# window Application



# Windows Form

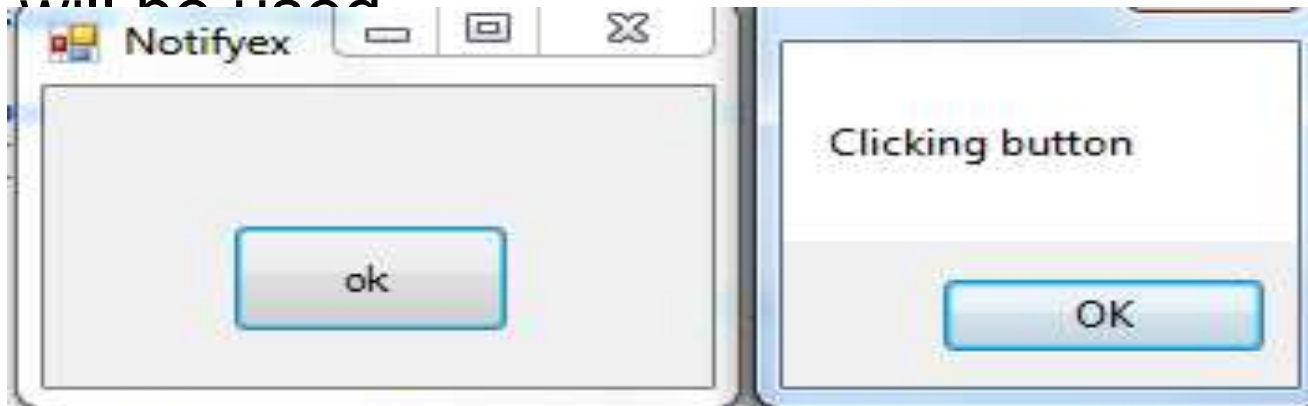
- **Windows Forms** (or simply **forms**) are the windows you see in a Windows Application. You can create multiple forms in a single application.
- Each form inherits the properties and methods of the `System.Windows.Forms.Form` class.
- The namespace `System.Windows.Forms` contains components you will need for creating forms and controls.



# Button Control

# Button control

- The **Button control** (`System.Windows.Forms.Button`) is commonly used to execute commands when it is clicked.
- When a button is clicked, you specify codes that will be used



```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show("Clicking button");
}
```

# Label control

# Label & LinkLabel control

- The **Label control** (`System.Windows.Forms.Label`) is used to add text to a form that can be used to show messages, or add labels to identify what other controls' functionality is. Drag a label control from the toolbox to the form.
- By default, it will have an initial text.
- A **LinkLabel control** is a label control that can display a hyperlink.

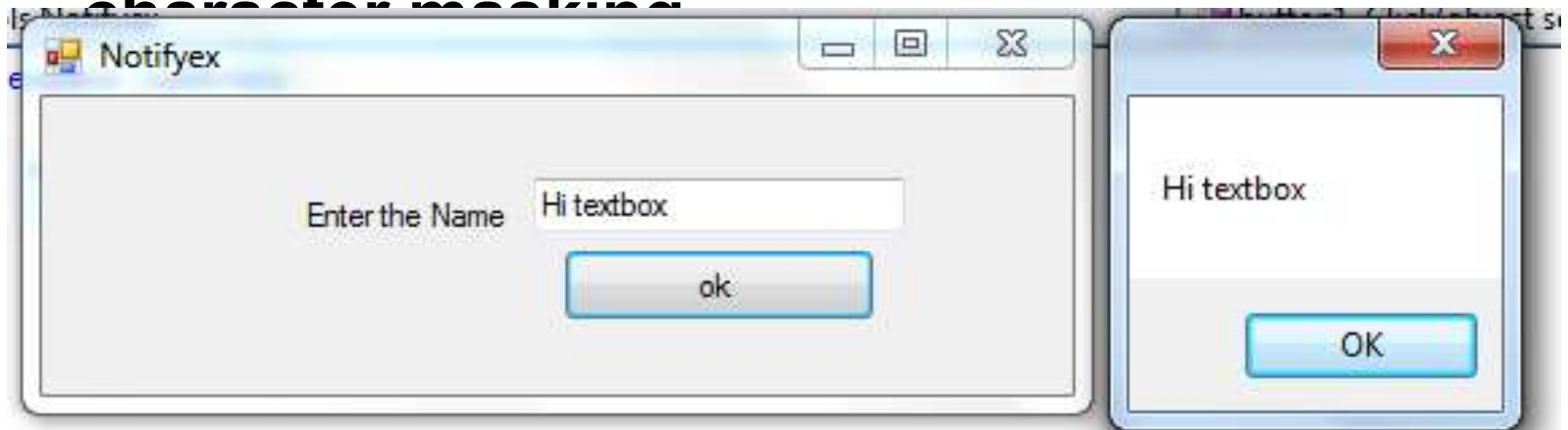


# TextBox Control



# Textbox control

- 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**

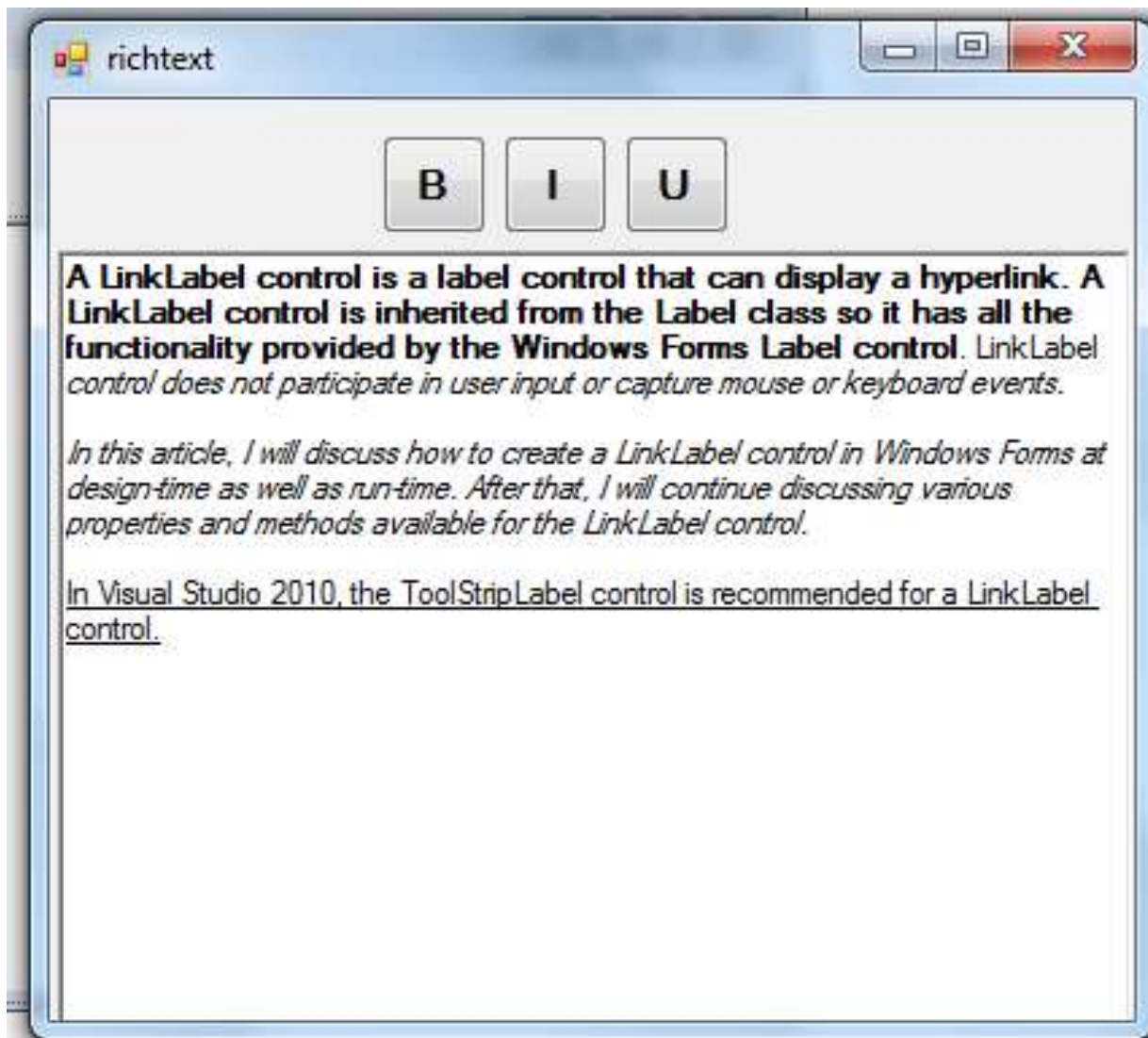


```
private void button1_Click(object sender, EventArgs e)
{
    MessageBox.Show(textBox1.Text);
}
```

# RichTextBox Control

# RichTextBox Control

- The RichTextBox control (System.Windows.Forms.RichTextBox) is similar to a TextBox control, but it allows you to format different parts of the text inside it.
- The TextBox control is typically used to accept text input from the user while the RichTextBox control is used to show formatted text and save it in Rich Text Format (RTF).



```

Font oldFont;
Font newFont;
// Get the font that is being used in the selected text
oldFont = this.richTextBox1.SelectionFont;

// If the font is using Italic style now, we should remove it
if (oldFont.Italic)
    newFont = new Font(oldFont, oldFont.Style & ~FontStyle.Underline);
else
    newFont = new Font(oldFont, oldFont.Style | FontStyle.Underline);

// Insert the new font
this.richTextBox1.SelectionFont = newFont;
this.richTextBox1.Focus();
}

private void button1_Click(object sender, EventArgs e)
{
    Font oldfont;
    Font newfont;
    oldfont = this.richTextBox1.SelectionFont;
    if (oldfont.Bold)
    {
        newfont = new Font(oldfont, oldfont.Style & ~FontStyle.Bold);
    }
    else
    {
        newfont = new Font(oldfont, oldfont.Style | FontStyle.Bold);
    }
    this.richTextBox1.SelectionFont = newfont;
    this.richTextBox1.Focus();
}

```

# CheckBox And RadioButton Control

# CheckBox And RadioButton

## Control

- **CheckBox**
- Checkboxes allow the user to make multiple selections from a number of options.
- CheckBox to give the user an option, such as true/false or yes/no.
- You can click a check box to select it and click it again to deselect it.
- **RadioButton**
- A radio button or option button enables the user to select a single option from a group of choices when paired with other RadioButton controls.
- When a user clicks on a radio button, it becomes checked, and all other radio buttons with same group become unchecked



checkboxex



## CheckBox

Choose the product to buy

☐

Pencil

☐

Sharpener

☐

Pen

Buy

## RadioButton

☐

Male

☐

Female

Click



```
private void button1_Click(object sender, EventArgs e)
{
    string items = String.Empty;

    if (checkBox1.Checked)
        items += "\n Pencil";
    if (checkBox2.Checked)
        items += "\n Sharpener";
    if (checkBox3.Checked)
        items += "\n Pen.";

    MessageBox.Show("You have bought: " + items);
}

private void button2_Click(object sender, EventArgs e)
{
    string Gender;
    if (radioButton1.Checked == true)
    {
        Gender = "Male";
    }
    else
    {
        Gender = "Female";
    }
    MessageBox.Show(Gender.ToString());
}
```

# NumericUpDown, PictureBox , ProgressBar, ComboBox , MonthlyCalendar And DateTimePicker Control

# NumericUpDown

- The NumericUpDown control is typically used to get numeric inputs and automatically restricts user for giving invalid non-numeric values.
- The NumericUpDown control appears like a TextBox control, but there are arrow buttons on its right or left side that is used to increment or decrement the value of the control.

Control	Property	Value
button1	Name	buttonCalculate
numericUpDown1	Name	numericUpDownPrice
	Decimal	2
	Increment	0.50
	Maximum	10000
numericUpDown2	Name	numericUpDownQuantity
	Maximum	100

## NumericUpDown

Price 4.50

Quantity 2

Calculate

The total price is \$9.00

OK

```
private void buttonCalculate_Click(object sender, EventArgs e)
{
    decimal price = numericUpDownPrice.Value;
    int quantity = (int)numericUpDownQuantity.Value;
    decimal total;

    total = price * quantity;

    MessageBox.Show(String.Format("The total price is {0:C}", total));
}
```

# PictureBox Control

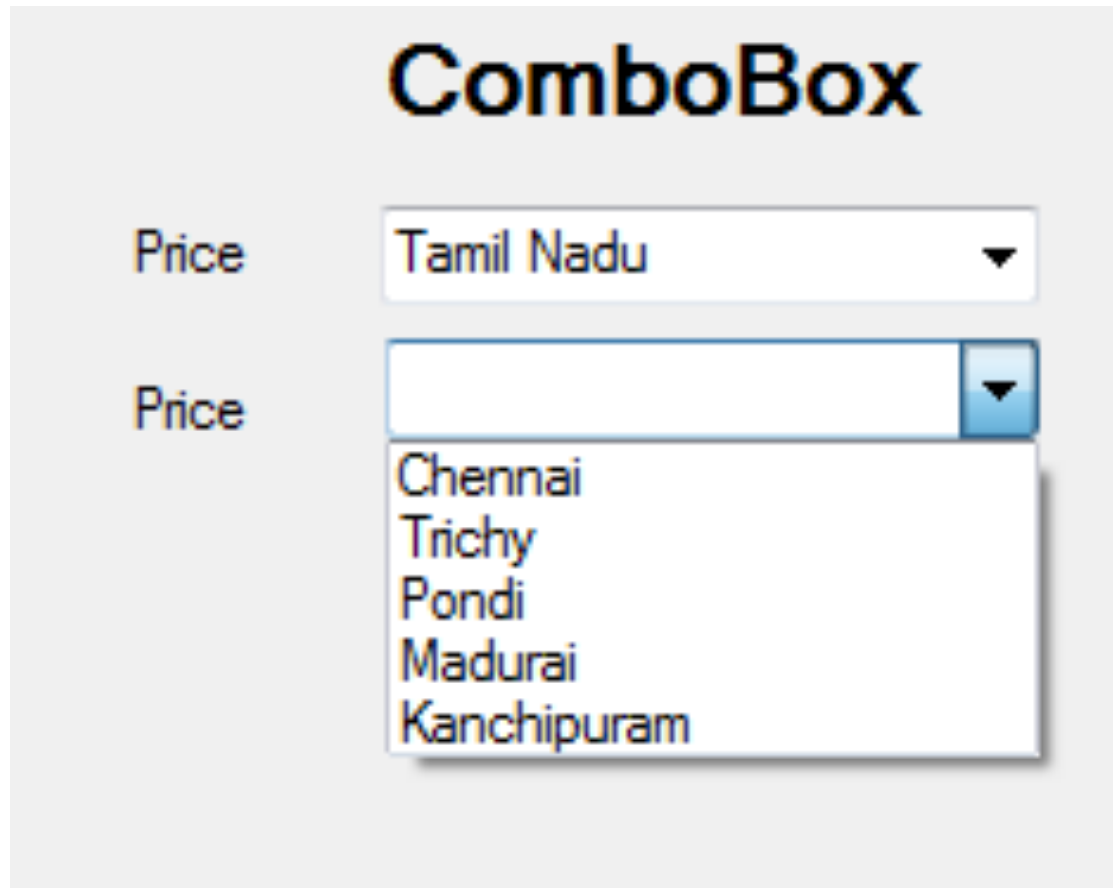
- The Windows Forms PictureBox control is used to display images in bitmap, GIF , icon , or JPEG formats.



```
private void pictureBox1_Click(object sender, EventArgs e)
{
    label1.Visible = false;
    OpenFileDialog o = new OpenFileDialog();
    if (o.ShowDialog() == DialogResult.OK)
    {
        pictureBox1.Image = new Bitmap(o.FileName);
        pictureBox1.SizeMode = PictureBoxSizeMode.StretchImage;
    }
}
```

# ComboBox control

- The **ComboBox** control is another way of allowing a user to choose from a set of options.



```
private void Form1_Load(object sender, EventArgs e)
{
    string[] names = { "Tamil Nadu", "Kerala", "Telugana", "Andhara", "Delhi" };

    foreach (string name in names)
    {
        comboBox1.Items.Add(name);
    }
}

private void comboBox1_SelectedIndexChanged(object sender, EventArgs e)
{
    comboBox2.Items.Clear();
    if (comboBox1.SelectedItem.ToString() == "Tamil Nadu")
    {
        comboBox2.Items.Add( "Chennai");
        comboBox2.Items.Add( "Trichy");
        comboBox2.Items.Add("Pondi");
        comboBox2.Items.Add( "Madurai");
        comboBox2.Items.Add("Kanchipuram");
    }
    if (comboBox1.SelectedItem.ToString() == "Kerala")
    {
        comboBox2.Items.Add("Kolam");
        comboBox2.Items.Add("Cochin");
        comboBox2.Items.Add("Thiruvandhapuram");
    }
}
```

# DateTimePicker Control

- DateTimePicker is ideal for choosing a single date and/or time value and requires the same amount of space as an ordinary drop-down list box.
- When the user clicks the drop-down button, a month calendar appears.
- The operation of the control from this point is exactly the same as the Mon





# MonthCalendar control

- The MonthCalendar control presents an intuitive graphical interface for users to view and set date information



## DateTimePicker

Thursday , June 30, 2016

## MonthlyCalendar

August, 2016						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10
Today: 7/8/2016						

DateTimePicker Date: Thursday, June 30, 2016

Month Calendar Date : Thursday, August 25, 2016

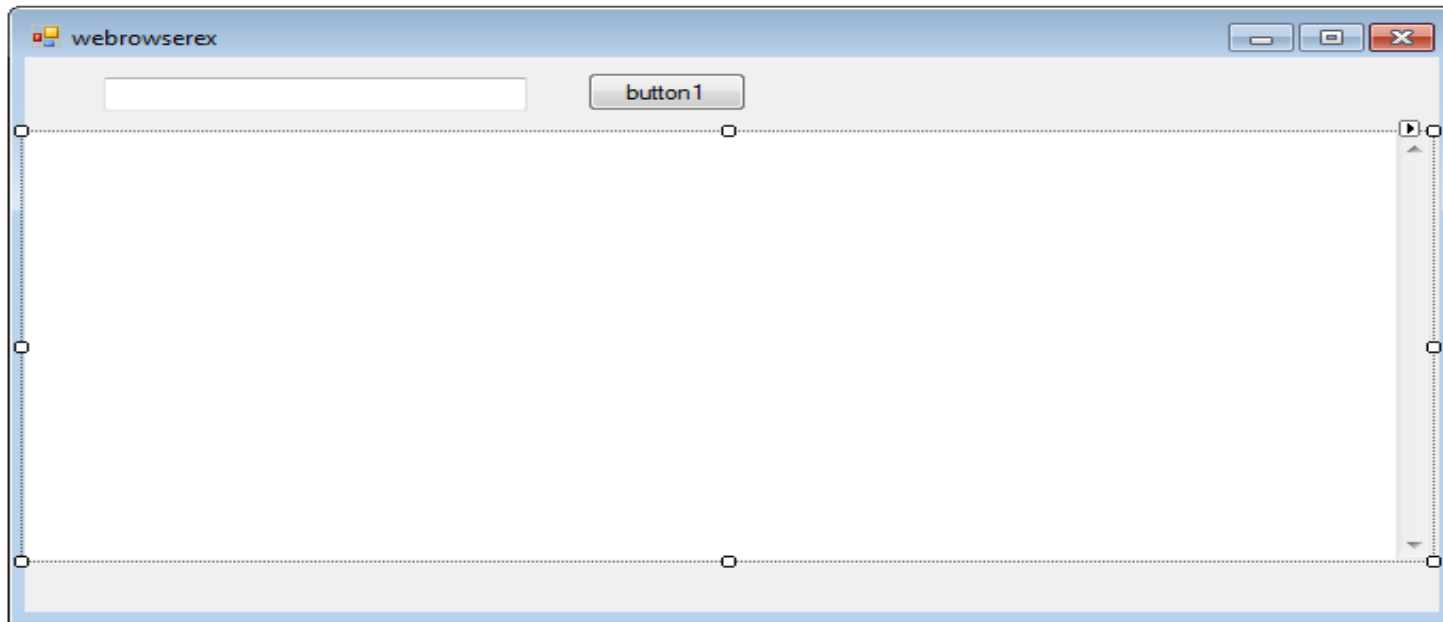
```
private void dateTimePicker1_ValueChanged(object sender, EventArgs e)
{
    label12.Text = "DateTimePicker Date: " + dateTimePicker1.Text;
}

private void monthCalendar1_DateChanged(object sender, DateRangeEventArgs e)
{
    label11.Text = "Month Calendar Date : " + monthCalendar1.SelectionStart.ToLongDateString();
}
```

# WebBrowser Control

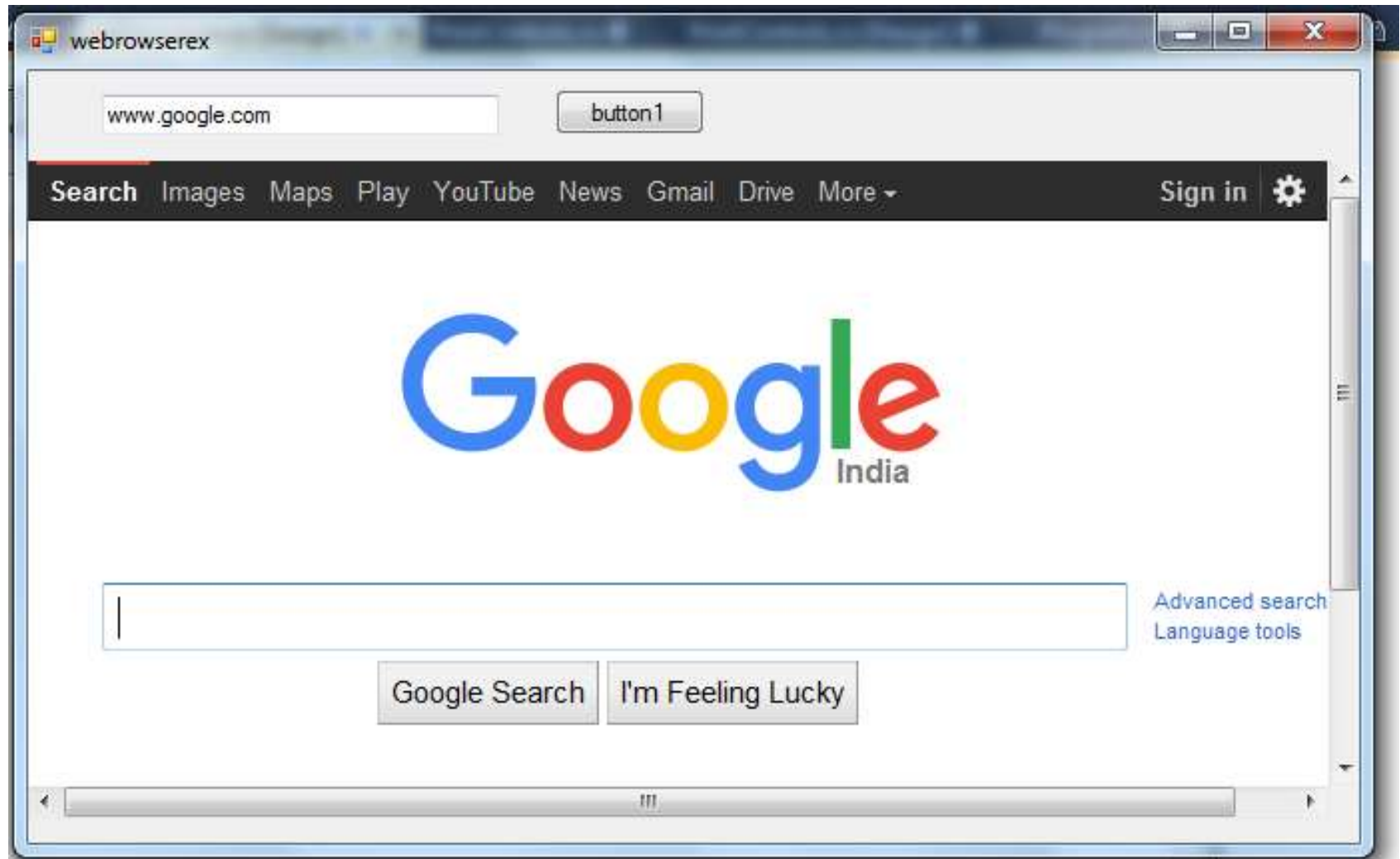
# Web Browser Control

- WebBrowser control allows developers to build Web browsing capability within Windows Forms



```
private void button1_Click(object sender, EventArgs e)
{
    webBrowser1.Navigate(textBox1.Text);
}
```

# Web Browser Output



# Dialog Controls

- ColorDialog Control
- FontDialog Control
- FolderBrowserDialog Control
- OpenFileDialog Control
- SaveFileDialog control

# The ColorDialog Control

- The ColorDialog (System.Windows.Forms.ColorDialog ) is used when you want to pick different colors.
- **For example**, when you want to pick a color of the font or a background color for the form, you can use the ColorDialog control
- The following are some of the useful properties of the ColorDialog control.

Properties	Description
AllowFullOpen	Specifies whether the user can choose custom colors.
Color	The color that the user selected.
CustomColors	A collection of custom colors picked by the user.
FullOpen	Specifies whether the part used to pick custom colors are automatically open.

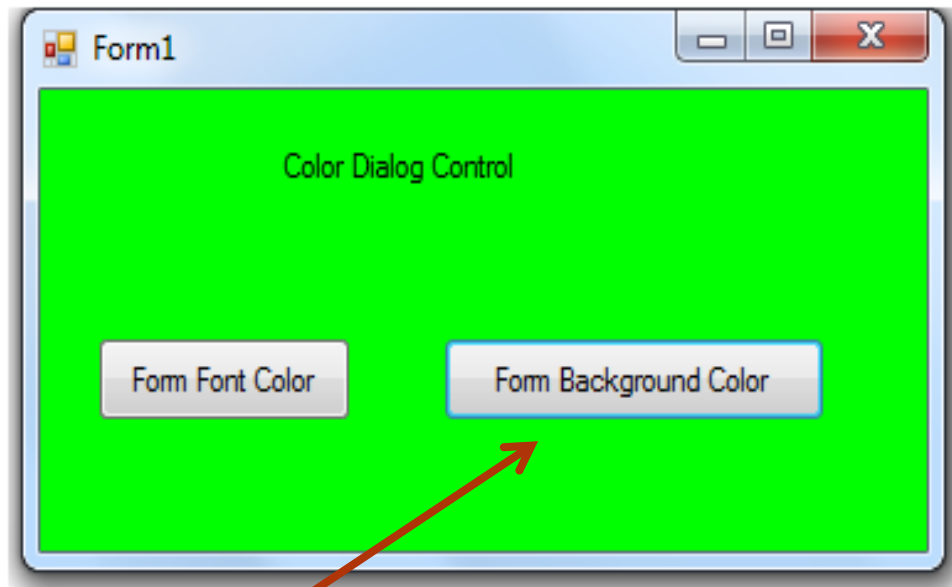


# Changing Form Background Color

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Controls
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button2_Click(object sender, EventArgs e)
        {
            ColorDialog color = new ColorDialog();
            if (color.ShowDialog() == DialogResult.OK)
            {
                this.BackColor = color.Color;
            }
        }
    }
}
```

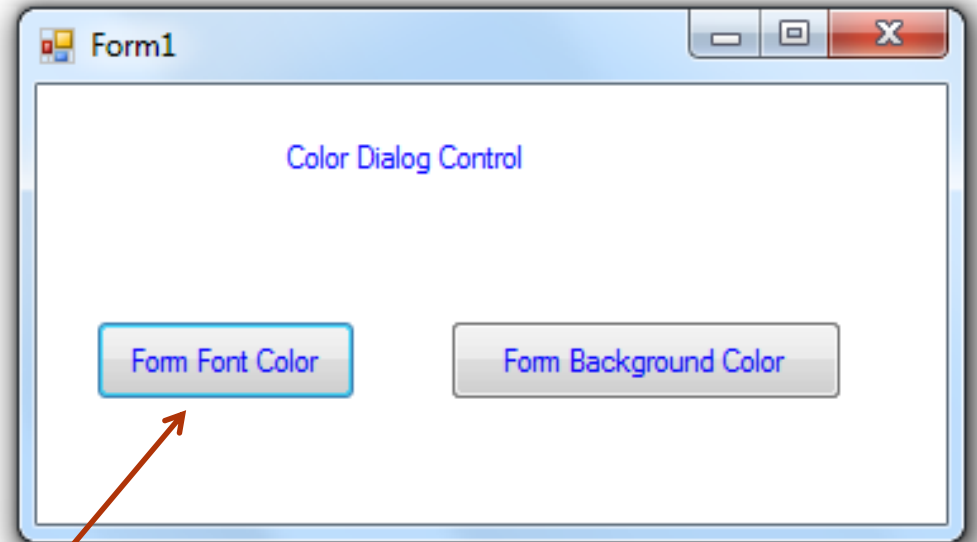


# Changing Form Font Color

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Controls
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

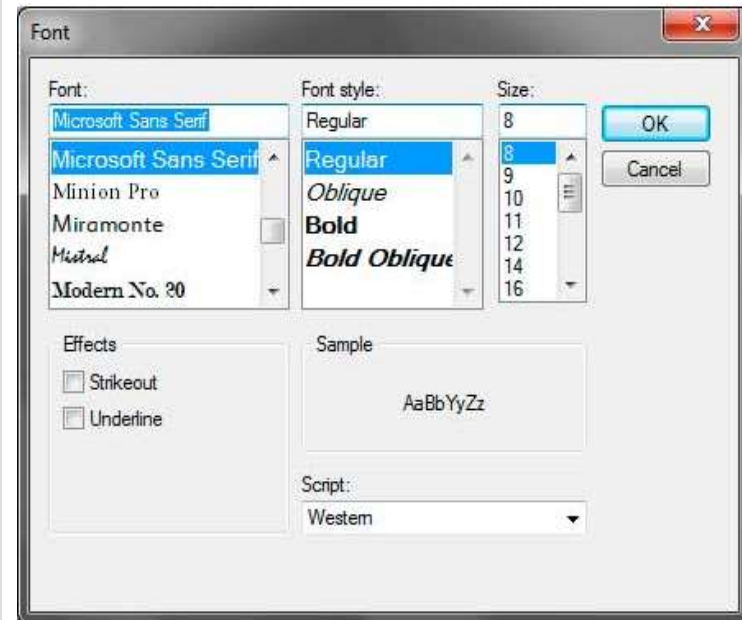
        private void button1_Click(object sender, EventArgs e)
        {
            ColorDialog color = new ColorDialog();
            if (color.ShowDialog() == DialogResult.OK)
            {
                this.ForeColor = color.Color;
            }
        }
    }
}
```



# The FontDialog Control

- The FontDialog control (System.Windows.Forms.FontDialog) is a handy control for selecting different kinds of font and font-

Properties	Description
Color	The selected color of the user.
Font	The resulting font constructed using the font dialog.
MaxSize	The maximum size the dialog can provide.
MinSize	The minimum size the dialog can provide.
ShowApply	Indicates whether to show the Apply button.
ShowColor	Indicates whether to show the Color option.
ShowEffects	Indicates whether to show the Effects option.



```

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;

namespace Controls
{
    public partial class Fontdialogex : Form
    {
        public Fontdialogex()
        {
            InitializeComponent();
        }

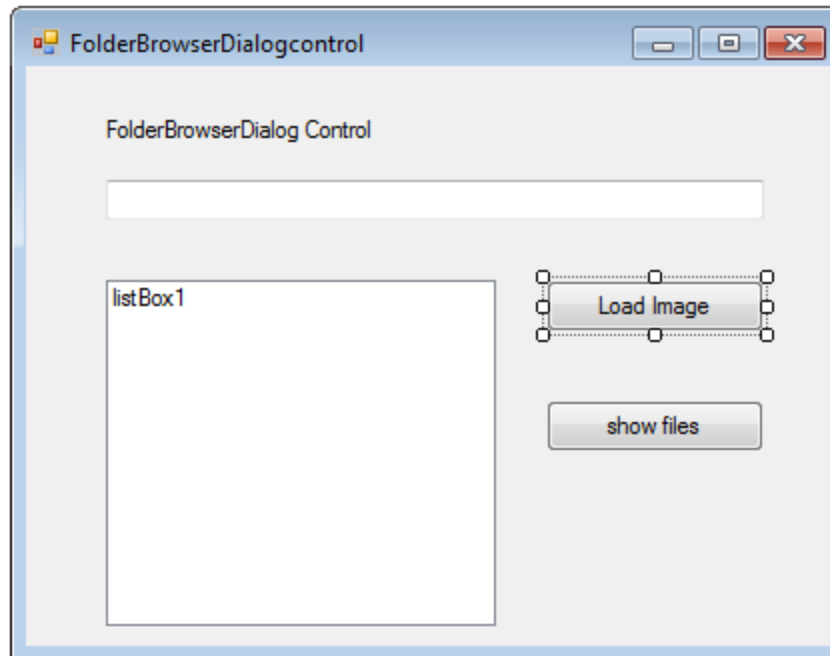
        private void button1_Click(object sender, EventArgs e)
        {
            FontDialog font = new FontDialog();
            if (font.ShowDialog() == DialogResult.OK)
            {
                label1.Font = font.Font;
            }
        }
    }
}

```



# The FolderBrowserDialog Control

- **Folder Browser Dialog** is a **.NET control** that prompts user to browse and select a folder location. Using FolderBroserDialog user can only select folders and not files.
- The FolderBrowserDialog control (System.Windows.Forms.FolderBrowserDialog) allows you to use it in your system.



```

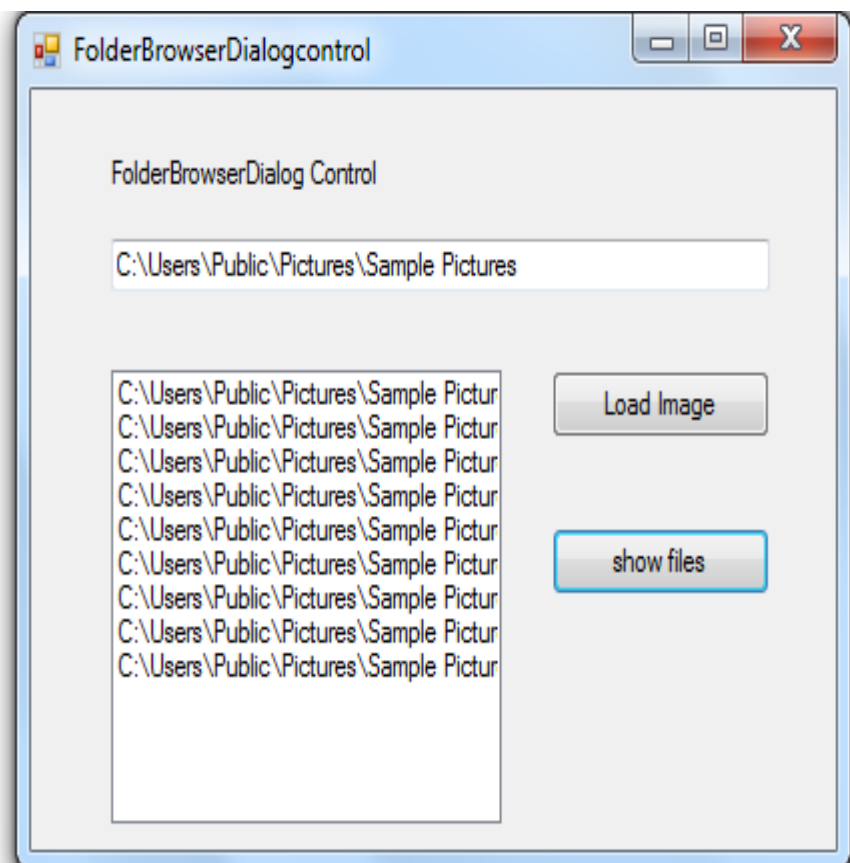
}

private void button1_Click(object sender, EventArgs e)
{
    // Create a new instance of FolderBrowserDialog.
    FolderBrowserDialog fbd = new FolderBrowserDialog();

    //Show FolderBrowserDialog
    if ( fbd.ShowDialog()==DialogResult.OK)
    {
        //Show selected folder path in textbox1.
        textBox1.Text = fbd.SelectedPath;
    }
}

private void button2_Click(object sender, EventArgs e)
{
    if (!textBox1.Text.Equals(String.Empty))
    {
        if (Directory.GetFiles(textBox1.Text).Length > 0)
        {
            foreach (string file in Directory.GetFiles(textBox1.Text))
            {
                //Add file in ListBox.
                listBox1.Items.Add(file);
            }
        }
        else
        {
            listBox1.Items.Add(String.Format("No files Found at location : {0}", textBox1.Text));
        }
    }
}

```



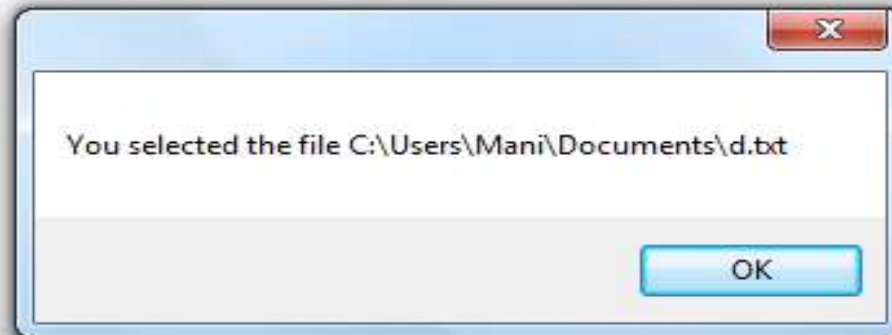
# OpenFileDialog Control

- An OpenFileDialog control is used to browse and select a file on a computer.

```
namespace Controls
{
    public partial class Opendia : Form
    {
        public Opendia()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            OpenFileDialog dlg = new OpenFileDialog();

            if (dlg.ShowDialog() == DialogResult.OK)
            {
                MessageBox.Show("You selected the file " + dlg.FileName);
            }
        }
    }
}
```





# SaveFileDialog control

- A SaveFileDialog control is used to save a file using Windows Save File Dialog.

