Windows Control Library

Windows control library is used to create your own control that can be used in windows applications. You can create a new control in any of the following three ways.

1. Extending An Existing Control
2. Composite Control and
3. Create From Scratch

Extending An Existing Control

This method is nothing but creating a new control by creating additional properties, methods and events to an existing control.

Example : The following example extends TextBox control by adding two properties, “EnterColor” and “RestrictTo”. EnterColor property is used to specify the color to be displayed as backcolor to the textbox whenever it got the focus and RestrictTo property is used to restrict the textbox to alphabets, digits or both.

1. Create a new project by selecting “Windows Forms Control Library” template and by providing the name as “MyTextBox”.

2. Within the solution explorer rename the file “usercontrol1.cs” to “Mytextbox.cs.

3. By default usercontrols are inherited from the class “usercontrol”. As we want to extend the textbox control, change the base class name of the user control “mytextbox” to “TextBox”.

4. As we changed the base class, there may be errors in the application. Hence build the solution using the shortcut “ctrl+shift+b” and delete all the statements that causes error.

5. Create a property within the class of the user control with the name “EnterColor” by creating a private variable with the name “EColor” as follows.

```
Color EColor;
public Color EnterColor
{
        get { return EColor; }
        set { EColor = value; }
}
```

6. To provide functionality for the property “EnterColor”, write the code in “Enter” and “Leave” events of the user control as follows.

```
private void MyTextBox_Enter(object sender, EventArgs e)
{
    this.BackColor = EColor;
}

private void MyTextBox_Leave(object sender, EventArgs e)
{
    this.BackColor = Color.White;
}
```

7. Create an enumeration with the name “Restrictions” as follows for the property “RestrictTo” and then create the property “RestrctTo” by creating a private variable with the name “Restrict” as follows.

```
public enum Restrictions
{
    None,
    Alphabets,
    Digits,
    Both
}
Restrictions Restrict;
public Restrictions RestrictTo
{
    get { return Restrict; }
    set { Restrict = value; }
}
```

8. Write the following code in “KeyPress” event of the user control to provide functionality for the property “RestrictTo”.

```
private void MyTextBox_KeyPress(object sender, KeyPressEventArgs e)
{
switch (Restrict)
{
case Restrictions.Alphabets:
if (char.IsLetter(e.KeyChar) == false && e.KeyChar != (char)Keys.Back)
{
      e.Handled = true;
}
break;
case Restrictions.Digits:
if (char.IsDigit(e.KeyChar) == false && e.KeyChar != (char)Keys.Back)
{
      e.Handled = true;
}
break;
case Restrictions.Both:
if (char.IsLetterOrDigit(e.KeyChar) == false && e.KeyChar !=
                                                      (char)Keys.Back)
{
      e.Handled = true;
}
break;
}
}
```

9. Build the solution using the shortcut “ctrl+shift+b” after changing the solution configuration using solution configuration combo box in the toolbar to “Release”. This will create a DLL file for the control within the release folder of bin folder of the project.

Using The Windows Control

After the control was created with windows control library, to use the control in a windows application, first the control must be added to “ToolBox”. To add the control to toolbox, right click on toolbox and choose “Choose Items” that opens “choose items” dialog box. In this dialog box click on browse button, select the DLL file of the Windows control you want to add to the toolbox and click on ok button. Once the control is added to the toolbox, you can use it same as other windows controls.

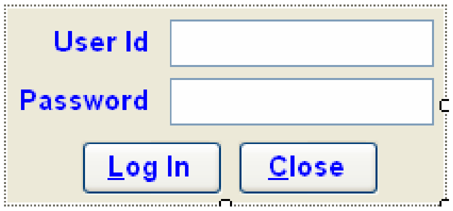
Composite Control

You can create a new control by using multiple existing controls and in this case the control is called as composite control as it is created from multiple controls.

Example : The following example creates a control with the name “security” for accepting user id and password.

1. Create a new project by selecting “windows forms control library” and by specifying the name as “Security” and rename the file “usercontrol1.cs” in solution explorer to “Security.cs”.

2. Design the user control as follows and set the following properties to the controls on the user control.



TextBox1	Name	: TxtUid
TextBox2	Name	: TxtPwd
Button1	Name	: BtnLogin
	Text	: &Log In
Button2	Name	: BtnClose
	Text	: &Close

3. Write the following code within the "resize" event of the user control to restrict the user from resizing the control after it was placed on the form.

```
private void Security_Resize(object sender, EventArgs e)
{
    this.Width = TxtUid.Left + TxtUid.Width + 5;
    this.Height = BtnLogin.Top + BtnLogin.Height + 5;
}
```

4. Create two properties "UserId" and "Password" as follows to provide access to the text boxes txtuid and txtpwd.

```
public string UserID
{
    get { return TxtUid.Text; }
    set { TxtUid.Text = value; }
}
public string Password
{
    get { return TxtPwd.Text; }
    set { TxtPwd.Text = value; }
}
```

5. Create an event with the name "LoginClick" as follows that will be raised when user clicks on Login button.

```
public event EventHandler LogInClick;
protected void OnLoginClick(object sender, EventArgs e)
{
    LogInClick(sender, e);
}
```

```
private void button1_Click(object sender, EventArgs e)
{
    OnLoginClick(sender, e);
}
```

6. Create an event with the name “CloseClick” as follows that will be raised when user clicks on “close” button.

```
public event EventHandler CloseClick;
protected void OnCloseClick(object sender, EventArgs e)
{
    CloseClick(sender, e);
}

private void button2_Click(object sender, EventArgs e)
{
    OnCloseClick(sender, e);
}
```

7. Build the solution using the shortcut “ctrl+shift+b” after changing the solution configuration to “Release” to generate a DLL file for the control in Release folder of bin folder of the project.

Creating From Scratch

In this method control will be created completely from scratch without using any existing control. For this you need to use the graphics programming available in .net. by using 3d graphics you have to draw the control and create each and every property, method and event manual for the control. For example you may want a round button in your application, which is not available in the windows controls provided in .net by Microsoft. In this case you have to manually create the round button control using Windows forms control library.