

Chương 4: Giao diện người dùng (User Interface)

Gv: Đặng Hữu Nghị

Nội dung

4.1. Giới thiệu

4.1.1. Ứng dụng Windows Forms

4.1.2. Thanh công cụ (Toolbox)

4.2. Biểu mẫu (Form)

4.3. Các điều khiển thông thường

4.4. Các điều khiển đặc biệt

4.4. Các điều khiển đặc biệt

4.4.1. Điều khiển Tooltip, HelpProvider, ErrorProvider

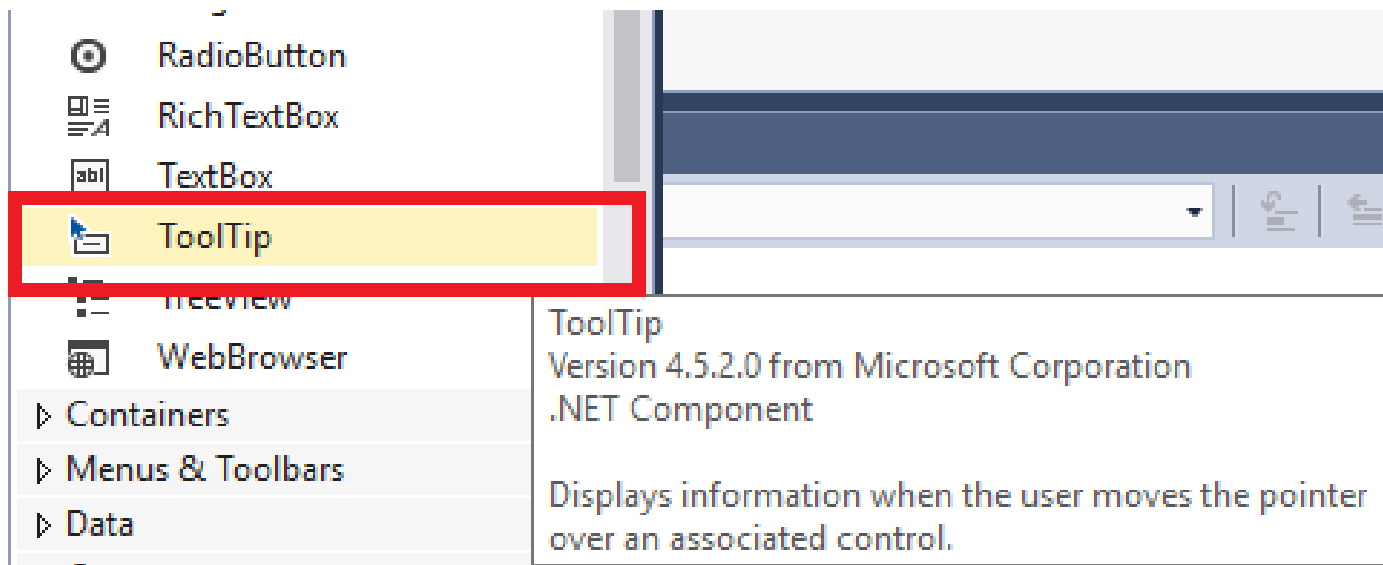
4.4.2. Điều khiển ProgressBar và Timer

4.4.3. Điều khiển ListView

4.4.5. Điều khiển DateTimePicker, MonthlyCalendar

4.4.1.1. Điều khiển Tooltip

- Cho phép hiển thị các thông tin chú thích khi người dùng đưa chuột qua các điều khiển có thiết lập *Tooltip*



4.4.1.1. Điều khiển Tooltip

- *Bảng mô tả các thuộc tính của Tooltip*

Thuộc tính	Mô tả
Active	Mang giá trị True hoặc False, nếu thiết lập True thì Tooltip có hiệu lực hiển thị thông báo, nếu mang giá trị False thì Tooltip không hiển thị được thông báo.
AutomaticDelay	Thiết lập thời gian xuất hiện Tooltip khi vừa đưa chuột đến điều khiển, thời gian tính bằng mili giây
AutoPopDelay	Thời gian hiển thị Tooltip cho đến khi kết thúc khi người dùng đã đưa chuột đến điều khiển, thời gian tính bằng mili giây
IsBalloon	Quy định kiểu hiển thị của Tooltip

4.4.1.1. Điều khiển Tooltip

- *Bảng mô tả các thuộc tính của Tooltip*

Thuộc tính	Mô tả
ReshowDelay	Thời gian mà Tooltip tắt từ khi người dùng đưa chuột ra khỏi điều khiển, thời gian tính bằng mili giây
ToolTipIcon	Biểu tượng xuất hiện bên cạnh chuỗi khai báo trong thuộc tính TooltipTitle
ToolTipTitle	Chuỗi hiện thị bên cạnh biểu tượng ToolTipIcon
UseAnimation	Thiết lập hiệu ứng ảnh động được biểu diễn khi Tooltip được hiển thị
UseFading	Thiết lập hiệu ứng mờ dần được biểu diễn khi Tooltip hiển thị

4.4.1.1. Điều khiển Tooltip

- Một số phương thức thường dùng của *Tooltip*:

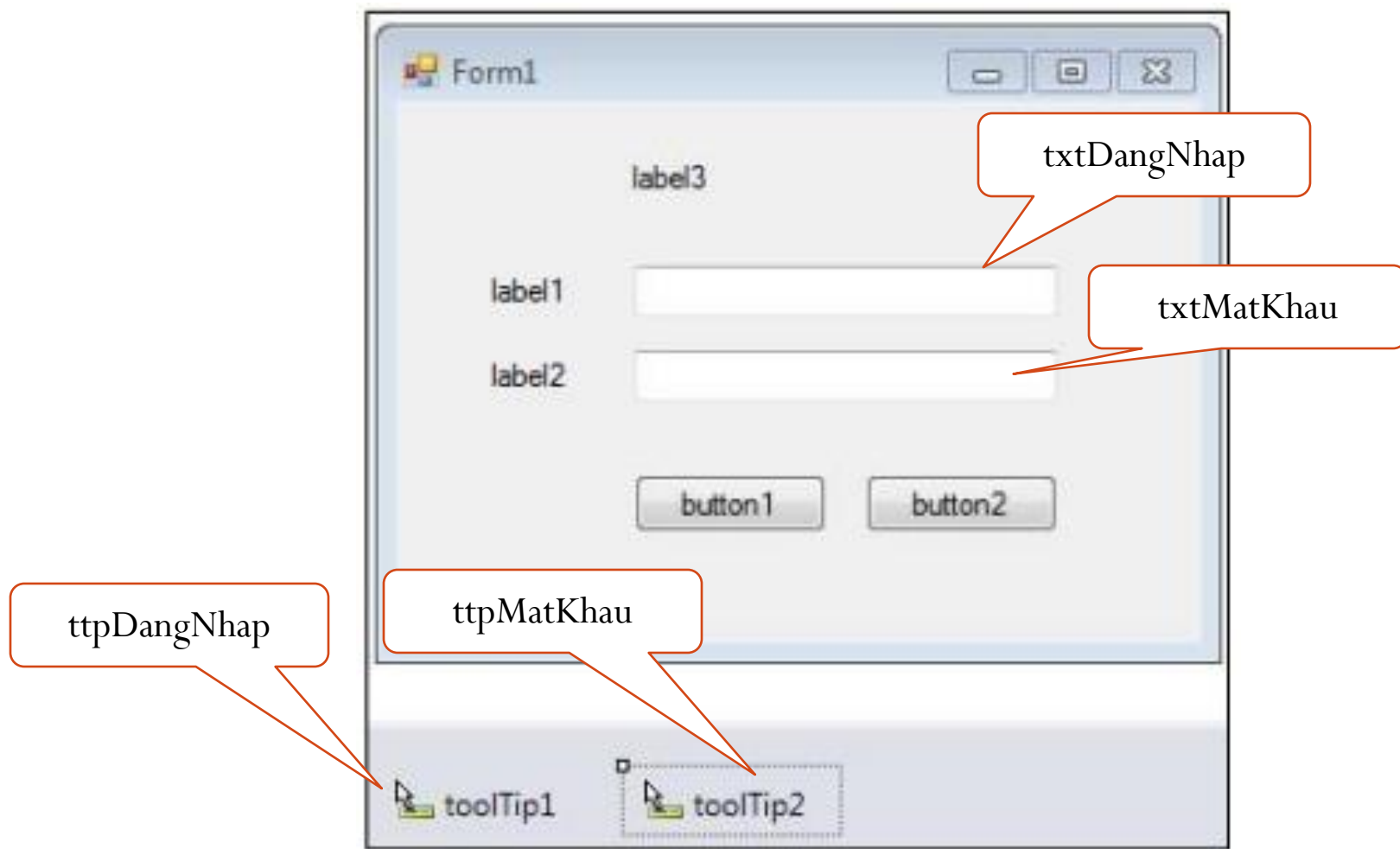
Phương thức	Mô tả
SetTooltip()	Thiết lập chuỗi hiển thị của Tooltip trên điều khiển
GetTooltip()	Lấy nội dung chuỗi hiển thị trên Tooltip
Clear()	Loại bỏ tất cả TooltipText cho các điều khiển trên form

4.4.1.1. Điều khiển Tooltip

- Ví dụ: Viết chương trình tạo giao diện form đăng nhập và thực hiện yêu cầu chức năng như hình

The image shows a Windows-style dialog box titled "Đăng nhập" (Login). It contains two text input fields: "Tên đăng nhập:" (Username) and "Mật khẩu:" (Password). Below the fields are two buttons: "Đăng nhập" (Login) and "Thoát" (Exit). Two callout boxes on the right side of the form point to the input fields, labeled "TextBox1" for the username field and "TextBox2" for the password field.

4.4.1.1. Điều khiển Tooltip



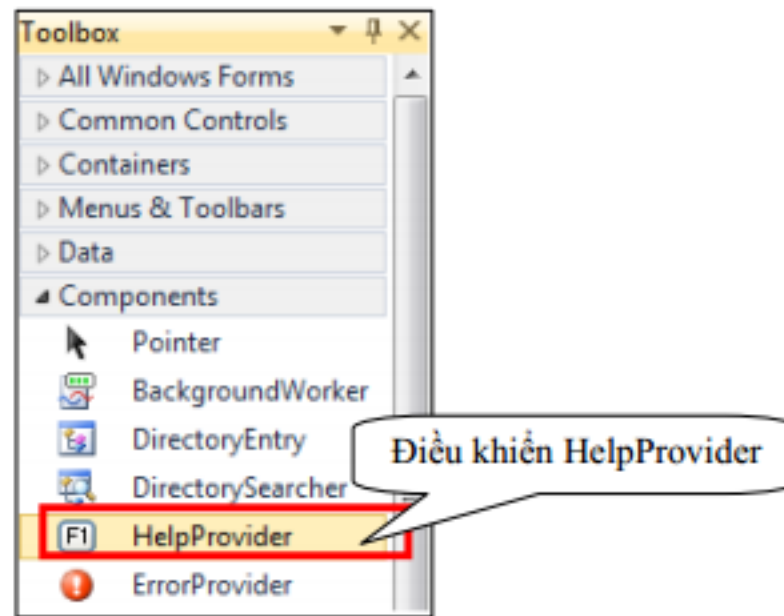
4.4.1.1. Điều khiển Tooltip

- Sự kiện *Load* của Form1:

```
private void Form1_Load(object sender, EventArgs e)
{
    ttpDangNhap.SetToolTip(txtDangNhap,
        "Nhập chuỗi ký không dấu, không khoảng trắng");
    ttpMatKhau.SetToolTip(txtMatKhau,
        "Nhập ít nhất 6 ký tự, nhiều nhất 10 ký tự");
}
```

4.4.1.2. Điều khiển HelpProvider

- Điều khiển *HelpProvider* cung cấp cửa sổ trợ giúp cho điều khiển. Với những ứng dụng có sử dụng *HelpProvider*, người dùng có thể gọi sự trợ giúp bằng cách ấn phím F1




4.4.1.2. Điều khiển HelpProvider

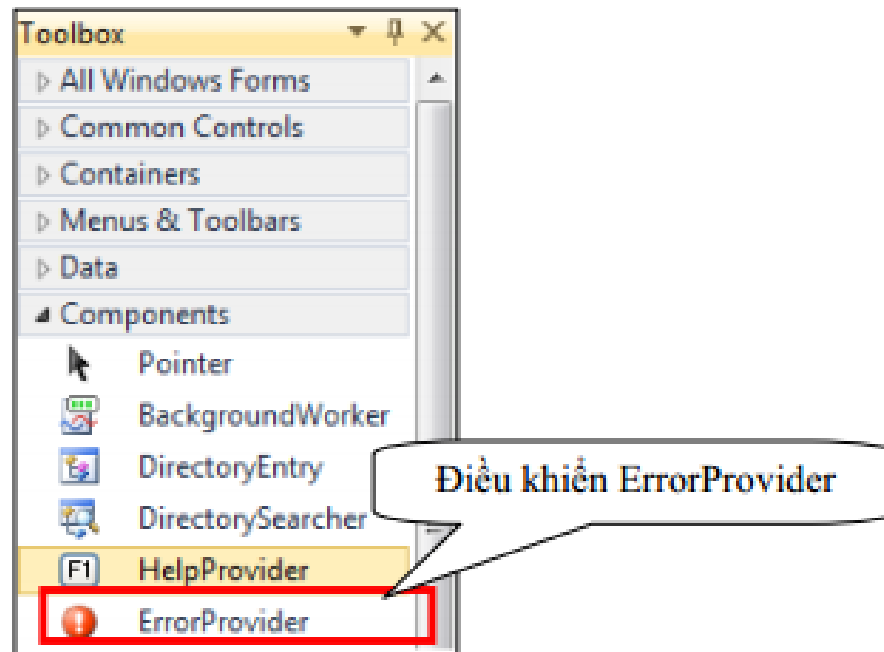
- Thuộc tính thường dùng của *HelpProvider*

Thuộc tính	Mô tả
<i>HelpNamespace</i>	Chỉ định tên tập trình trợ giúp định dạng chm hoặc html.

- Điểm đặc biệt là khi thêm điều khiển *HelpProvider* vào form thì một số thuộc tính như: *HelpKeyword on helpProvider*, *HelpNavigator on helpProvider*, *HelpStringon helpProvider* và *ShowHelp on helpProvider* sẽ xuất hiện trên tất cả các điều khiển có trên form.

4.4.1.3. Điều khiển ErrorProvider

- ErrorProvider giúp báo cho người dùng biết thông tin lỗi của điều khiển trên form. Thông thường khi điều khiển trên form lỗi, ErrorProvider sẽ cung cấp một biểu tượng 



4.4.1.3. Điều khiển **ErrorProvider**

- Một số thuộc tính thường dùng của *ErrorProvider*.

Thuộc tính	Mô tả
<i>Icon</i>	Chọn biểu tượng thể hiện lỗi của điều khiển
<i>BlinkRate</i>	Tốc độ nhấp nháy của biểu tượng trong thuộc tính <i>Icon</i> . Tốc độ tính theo mili giây
<i>BlinkStyle</i>	Kiểu nhấp nháy của biểu tượng. Nếu thiết lập giá trị <i>NeverBlink</i> thì biểu tượng sẽ hiển thị mà không nhấp nháy.

4.4.1.3. Điều khiển **ErrorProvider**

- Một số phương thức thường dùng của *ErrorProvider*.

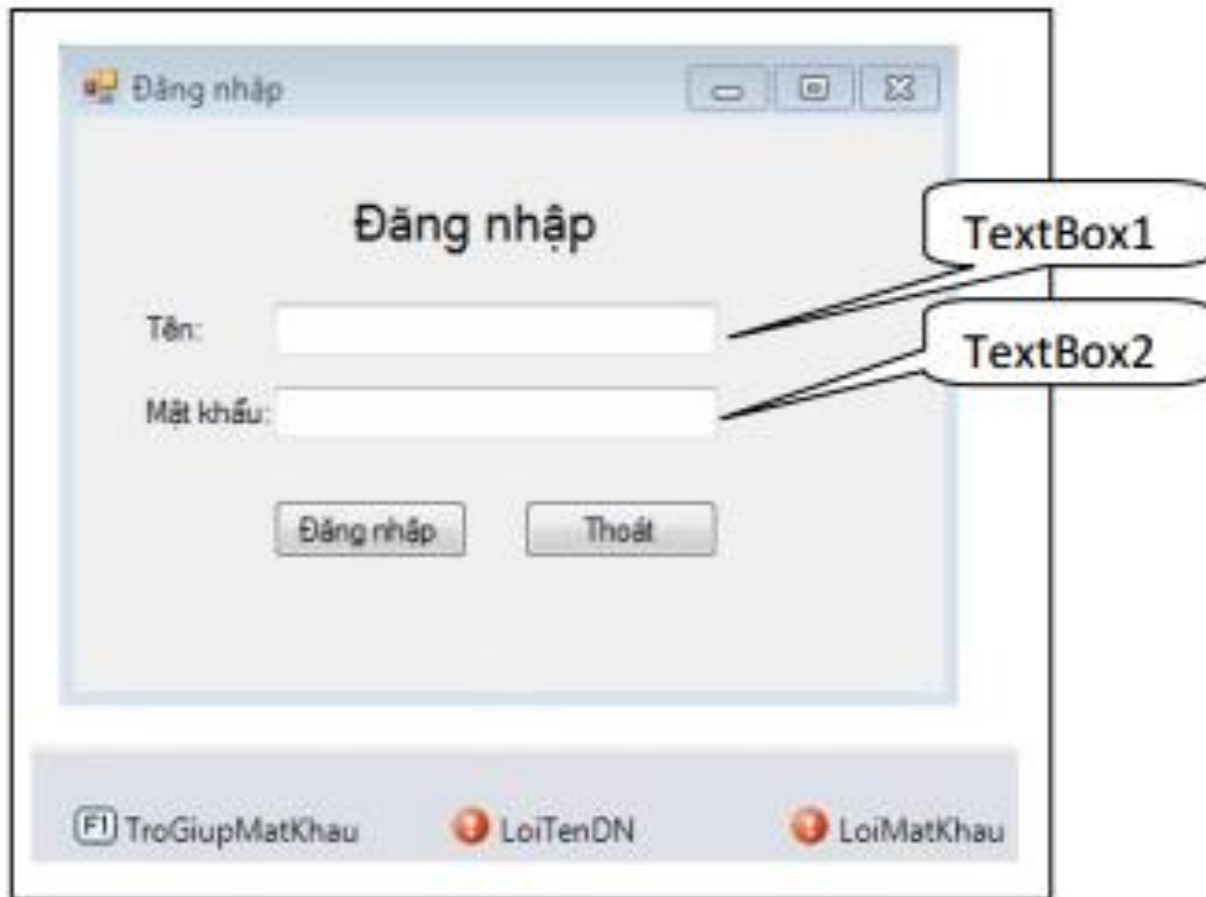
Phương thức	Mô tả
<i>SetError</i> (<Điều khiển>, <Thông báo lỗi>)	Giúp hiển thị lỗi và thông báo lỗi của điều khiển. Thông báo lỗi hiển thị dưới dạng Tooltip
<i>Clear</i> ()	Xóa biểu tượng ErrorProvider của điều khiển tương ứng trên form.
<i>GetError</i> ()	Lấy chuỗi thông báo lỗi của điều khiển.

4.4.1.3. Điều khiển **ErrorProvider**

- Ví dụ: Viết chương trình tạo form đăng nhập như hình. Yêu cầu ở Textbox nhập tên tài khoản không được có khoảng trắng; Textbox mật khẩu phải là ký tự số và không được để trống; Hiện thị trợ giúp cho điều khiển Textbox tên tài khoản, cụ thể khi nhấn F1 sẽ hiện trợ giúp tạo mật khẩu từ website:

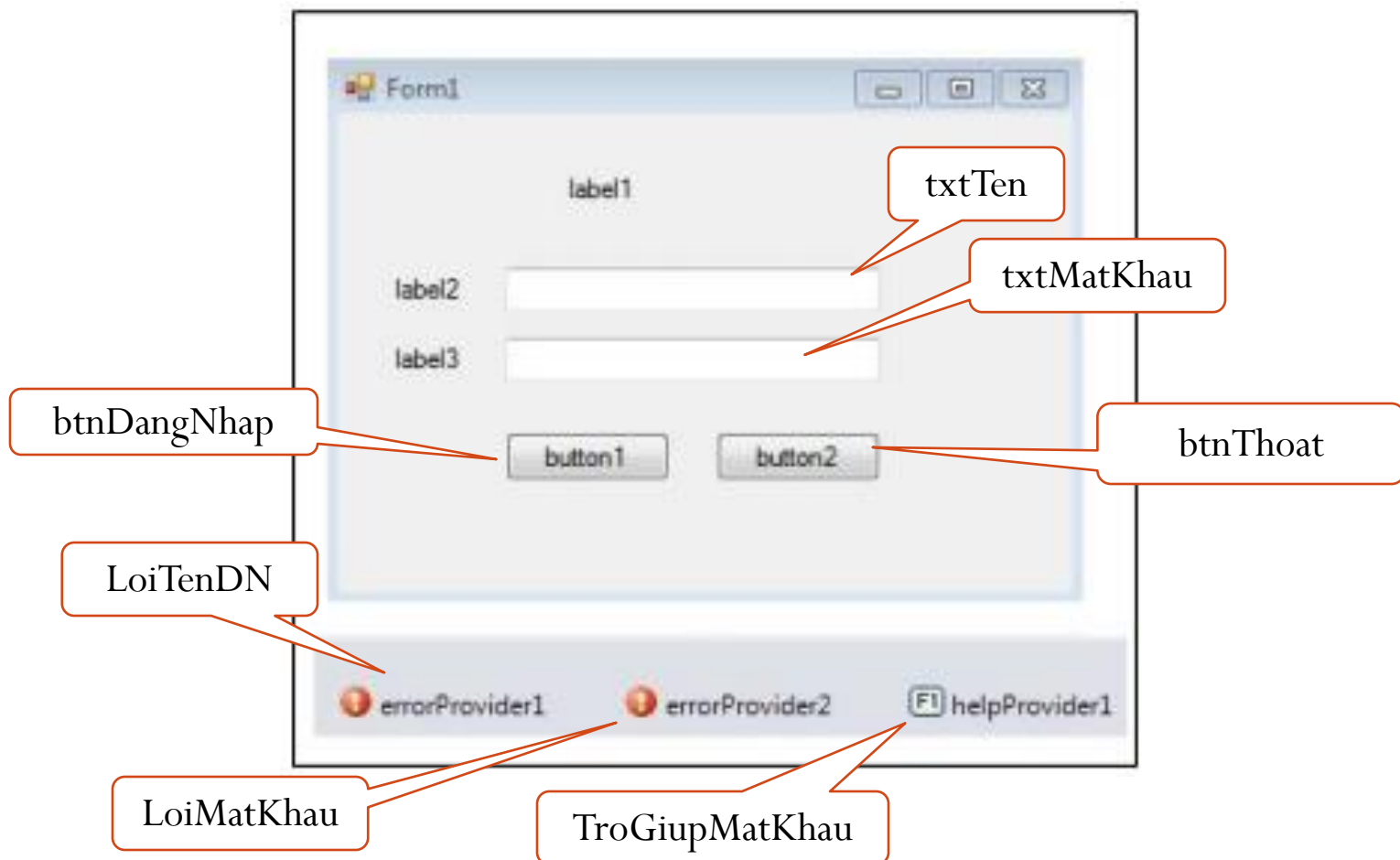
<http://phunutoday.vn/kham-pha-cong-nghe/cac-nguyen-tac-tao-mat-khau-an-toan-33828.html>

4.4.1.3. Điều khiển **ErrorProvider**



4.4.1.3. Điều khiển **ErrorProvider**

- Bước 1: Thiết kế giao diện ban đầu như hình



4.4.1.3. Điều khiển `ErrorProvider`

- Sự kiện *Load* của `Form1`:

```
private void Form1_Load(object sender, EventArgs e)
{
    //TabIndex giúp thiết lập thứ tự điều khiển khi ấn phím
    //tab
    txtTen.TabIndex = 0;
    txtMatKhai.TabIndex = 1;
    btnDangNhap.TabIndex = 2;
    btnThoat.TabIndex = 3;
    TroGiupMatKhai.SetShowHelp(txtTen, true);
    TroGiupMatKhai.HelpNamespace =
        "http://phunutoday.vn/kham-pha-cong-nghe/cac-nguyen-
        tactao-mat-khai-an-toan-33828.html";
}
```

4.4.1.3. Điều khiển `ErrorProvider`

- Sự kiện `TextChanged` của `TextBox` `txtMatKhai`

```
private void txtMatKhai_TextChanged(object sender, EventArgs e)
{
    long so = 0;
    try
    {
        so = Convert.ToInt64(txtMatKhai.Text);
        LoiMatKhai.Clear();
    }
    catch (Exception ex)
    {
        LoiMatKhai.SetError(txtMatKhai, "Phải nhập ký tự  
số và không được để trống");
    }
}
```

4.4.1.3. Điều khiển **ErrorProvider**

- Sự kiện *TextChanged* của TextBox txtTen

```
private void txtTen_TextChanged(object sender, EventArgs e)
{
    if (txtTen.Text.IndexOf(' ') != -1)
        LoiTenDN.SetError(txtTen, "Nhập tên không được  
có khoảng trắng");
    else
        LoiTenDN.Clear();
}
```

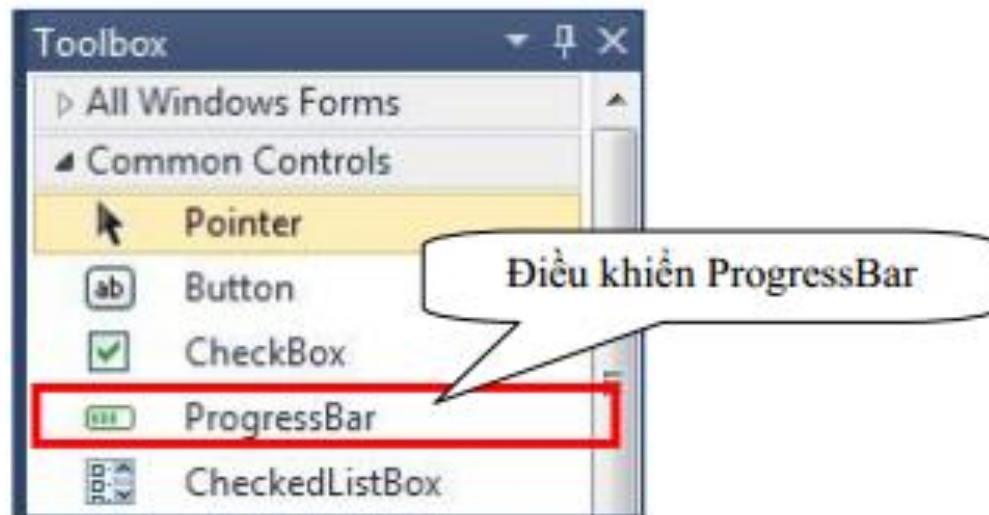
4.4.2. Điều khiển ProgressBar và Timer

4.4.2.1. Điều khiển ProgressBar

4.4.2.2. Điều khiển Timer

4.4.2.1. Điều khiển ProgressBar

- *ProgressBar* sử dụng để hiển thị thời gian thực hiện của một công việc nào đó



4.4.2.1. Điều khiển ProgressBar

- Một số thuộc tính thường dùng của *ProgressBar*

Thuộc tính	Mô tả
<i>Maximum</i>	Giá trị tối đa của <i>ProgressBar</i> . Khi <i>ProgressBar</i> được lấp đầy nghĩa là <i>ProgressBar</i> đã đạt giá trị <i>Maximum</i> .
<i>Minimum</i>	Giá trị nhỏ nhất của <i>ProgressBar</i> . Khi <i>ProgressBar</i> trống rỗng nghĩa là <i>ProgressBar</i> đang có giá trị <i>Minimum</i> .
<i>Value</i>	Giữ giá trị hiện tại của <i>ProgressBar</i> , giá trị này nằm trong đoạn <i>Minimum</i> và <i>Maximum</i> .
<i>Style</i>	Kiểu hiển thị của <i>ProgressBar</i> .
<i>Step</i>	Lượng giá trị thêm vào <i>Value</i> khi phương thức <i>PerformStep()</i> được gọi.

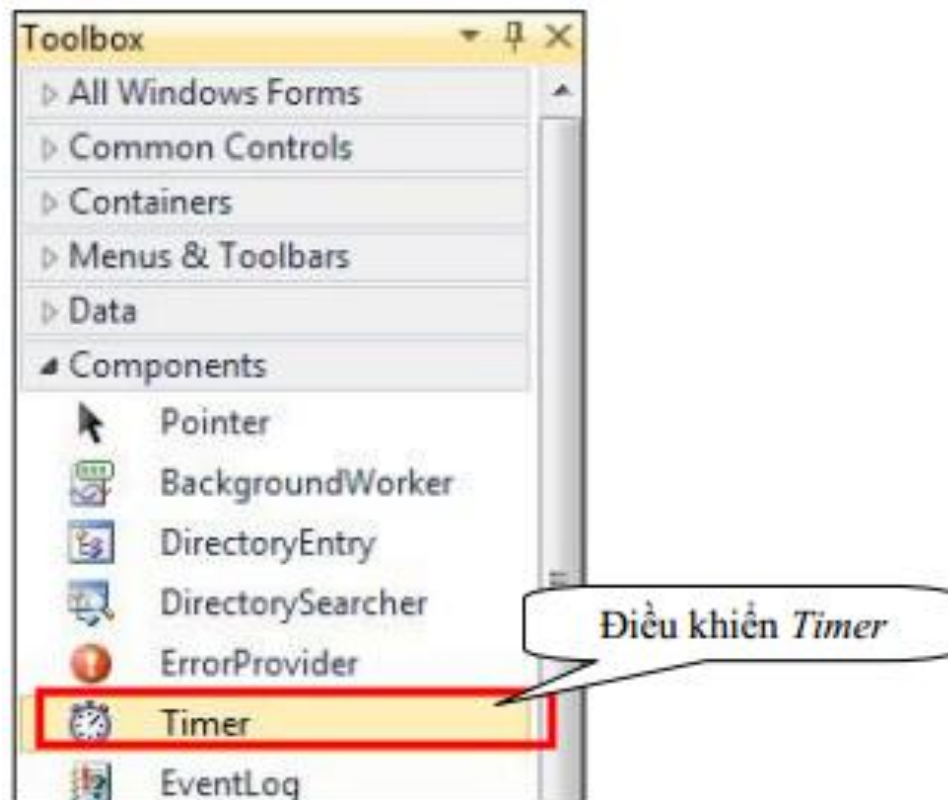
4.4.2.1. Điều khiển ProgressBar

- Một số phương thức thường dùng của *ProgressBar*.

Phương thức	Mô tả
<i>PerformStep()</i>	Phương thức giúp tăng <i>ProgressBar</i> . Giá trị tăng là giá trị được thiết lập trong thuộc tính <i>Step</i> .
<i>Increment(<giá trị>)</i>	Phương thức giúp tăng <i>ProgressBar</i> . Giá trị tăng là tham số đầu vào <giá trị> của phương thức.

4.4.2.2. Điều khiển Timer

- Điều khiển *Timer* cho phép thực thi lại một hành động sau một khoảng thời gian xác định



4.4.2.2. Điều khiển Timer

- Một số thuộc tính thường dùng của *Timer*

Thuộc tính	Mô tả
<i>Interval</i>	Thiết lập giá trị là một số nguyên. Giá trị nguyên này là thời lượng của một chu kỳ (tính bằng đơn vị mili giây).
<i>Enable</i>	Thiết lập giá trị <i>True</i> hoặc <i>False</i> . Nếu là giá trị <i>True</i> thì điều khiển <i>Timer</i> hoạt động, nếu là <i>False</i> thì điều khiển <i>Timer</i> không hoạt động.

4.4.2.2. Điều khiển Timer

- Một số phương thức thường dùng của *Timer*

Phương thức	Mô tả
<i>Start()</i>	Kích hoạt điều khiển <i>Timer</i> hoạt động. Phương thức này tương ứng với việc thiết lập giá thuộc tính <i>Enable</i> là <i>True</i>
<i>Stop()</i>	Dừng hoạt động của điều khiển <i>Timer</i> . Phương thức này tương ứng với việc thiết lập giá thuộc tính <i>Enable</i> là <i>False</i> .

- Một số sự kiện thường dùng của *Timer*

Sự kiện	Mô tả
<i>Tick</i>	Sự kiện được gọi trong mỗi chu kỳ Interval

4.4.2.2. Điều khiển Timer

- Ví dụ: Viết chương trình hỗ trợ người dùng học giải phương trình bậc nhất: $ax + b = 0$. Thiết kế giao diện form như hình



The image shows a Windows application window titled "Form1". Inside the window, the text "Phương trình bậc nhất: $ax + b = 0$ " is displayed. Below this text are three input fields: "Hệ số a:", "Hệ số b:", and "Nhập nghiệm:". At the bottom of the form are two buttons labeled "Kiểm tra" and "Thoát". A status bar at the bottom of the window contains a clock icon and the text "Thời Gian".

4.4.2.2. Điều khiển Timer

- Yêu cầu:

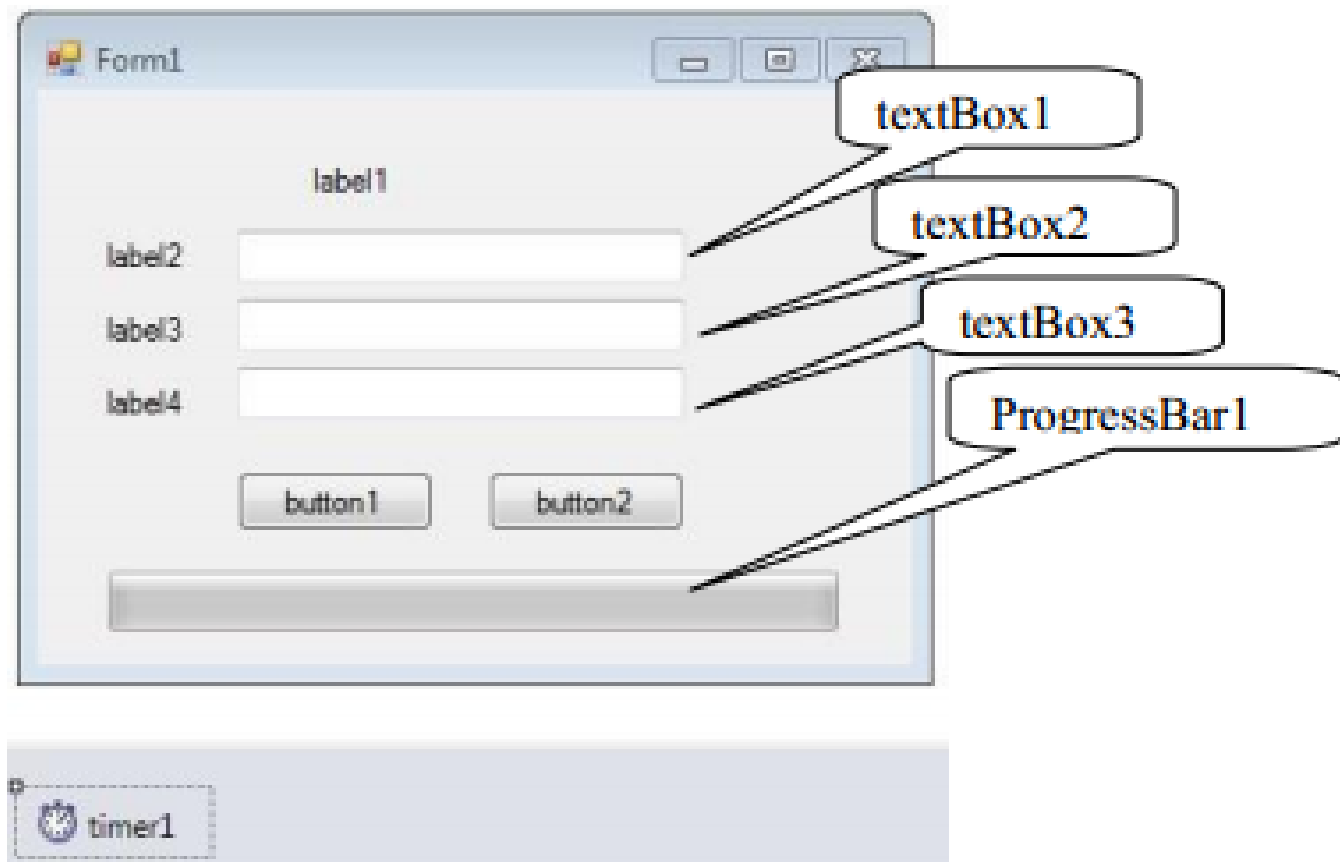
Phát sinh ngẫu nhiên hệ số a và b của phương trình. Sau đó người dùng nhập kết quả và ấn nút trả lời. Nếu trả lời đúng thì hiện MessageBox với nội dung “Bạn đã làm đúng”, nếu trả lời sai thì hiện MessageBox với nội dung “Bạn đã trả lời sai”.

- Lưu ý:

Thời gian để hoàn thành giải phương trình là 30 giây hiển thị tương ứng với ProgressBar, trong khoảng thời gian hết 30 giây người dùng không giải được sẽ hiển thị MessageBox với nội dung “Hết giờ làm bài”.

4.4.2.2. Điều khiển Timer

- Bước 1: Thiết kế giao diện ban đầu như hình



4.4.2.2. Điều khiển Timer

- Bước 2: Thiết lập giá trị thuộc tính trong cửa sổ Properties cho điều khiển
 - ✓ label1:
 - Thuộc tính Text: “Phương trình bậc nhất: $ax + b = 0$ ”
 - Thuộc tính Size: 14
 - ✓ label2:
 - Thuộc tính Text: “Hệ số a:”
 - ✓ label3:
 - Thuộc tính Text: “Hệ số b:”
 - ✓ label4:
 - Thuộc tính Text: “Nhập nghiệm:”
 - ✓ textBox1:
 - Thuộc tính Name: txtA
 - Thuộc tính Enable: False

4.4.2.2. Điều khiển Timer

✓ textBox2:

- Thuộc tính Name: txtB
- Thuộc tính Enable: False

✓ textBox3:

- Thuộc tính Name: txtX

✓ button1:

- Thuộc tính Name: btnKiemTra
- Thuộc tính Text: “Kiểm tra”

✓ button2:

- Thuộc tính Name: btnThoat
- Thuộc tính Text: Thoát

4.4.2.2. Điều khiển Timer

✓ ProgressBar1:

- Thuộc tính Name: ProGressTG
- Thuộc tính Minimum: 0
- Thuộc tính Maximum: 30000
- Thuộc tính Step: 1000
- Thuộc tính Style: Blocks

✓ timer1:

- Thuộc tính Name: ThoiGian
- Thuộc tính Enable: True
- Thuộc tính Interval: 1000

4.4.2.2. Điều khiển Timer

- Bước 3: Viết mã lệnh cho các điều khiển
 - ✓ Khai báo các biến

```
private int a=0;  
private int b=0;  
private float x = 0;  
Random rd = new Random();
```

- ✓ Sự kiện *Load* của Form1:

```
private void Form1_Load(object sender, EventArgs e)  
{  
    a = rd.Next(-10, 10);  
    txtA.Text = a.ToString();  
    b = rd.Next(-10, 10);  
    txtB.Text = b.ToString();  
    x = -b / (float)a;  
}
```

4.4.2.2. Điều khiển Timer

✓ Sự kiện *Tick* của *Timer* ThoiGian

```
private void ThoiGian_Tick(object sender, EventArgs e)
{
    if (ProGressTG.Value == 30000)
    {
        ThoiGian.Enabled = false;
        MessageBox.Show("Hết giờ làm bài");
    }
    ProGressTG.PerformStep();
}
```

4.4.2.2. Điều khiển Timer

✓ Sự kiện *Click* của nút btnKiemTra

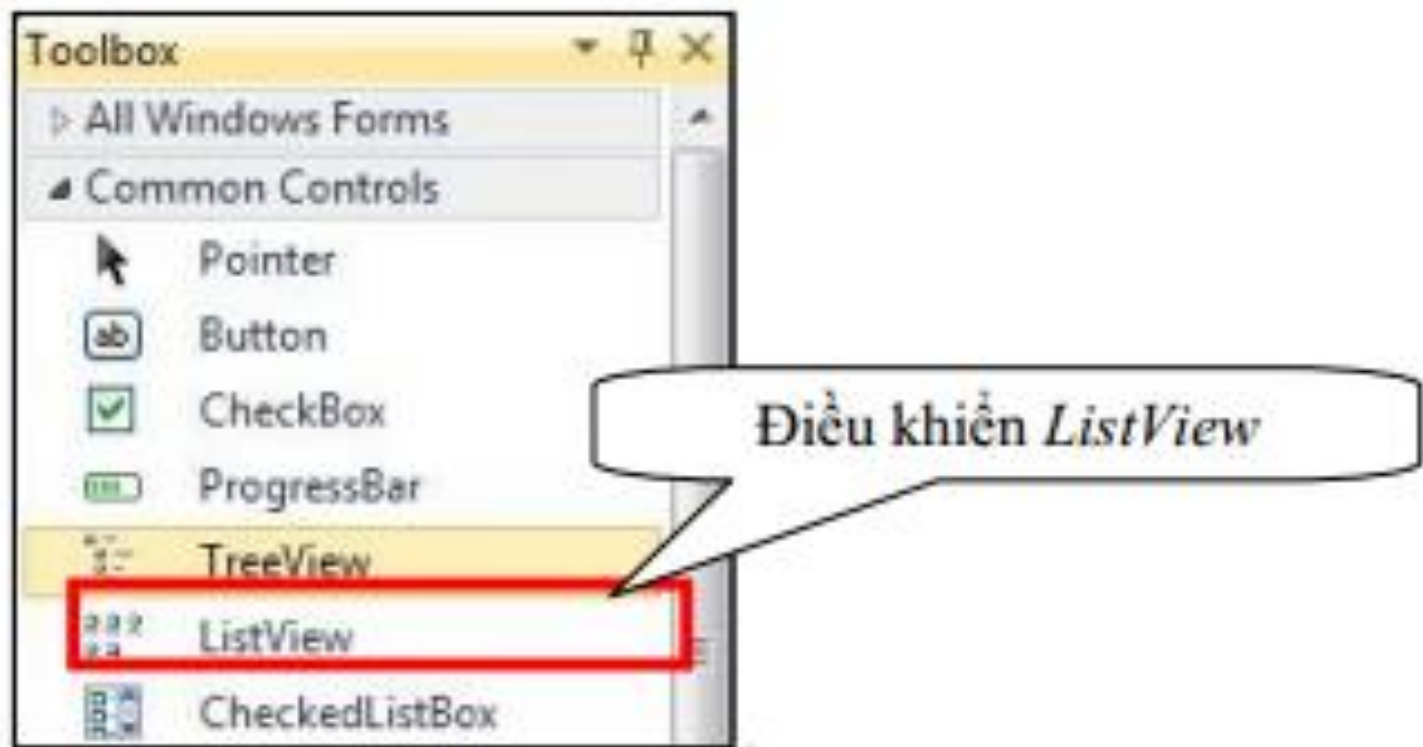
```
private void btnKiemtra_Click(object sender, EventArgs e)
{
    float kq = float.Parse(txtX.Text);
    if (Math.Abs(kq - x) < 0.01)
    {
        MessageBox.Show("Bạn đã làm đúng");
        Close();
    }
    else
        MessageBox.Show("Bạn đã trả lời sai");
}
```

4.4.3. Điều khiển ListView

- ListView là điều khiển cho phép hiển thị danh sách các đối tượng.
- Mỗi đối tượng hiển thị trong ListView được gọi là Item.
- Item là đối tượng được tạo từ lớp ListViewItem. Mỗi Item có thuộc tính Text là chuỗi ký tự hiển thị ở cột đầu tiên trong ListView, mỗi Item có các SubItem hiển thị ở các cột tiếp theo trong ListView

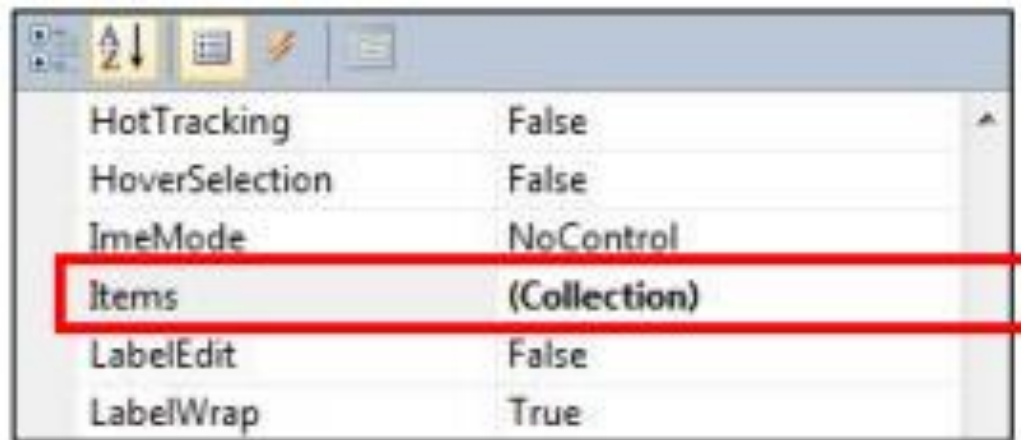
4.4.3. Điều khiển ListView

- Điều khiển ListView đặt trong Common Controls của cửa sổ Toolbox như hình



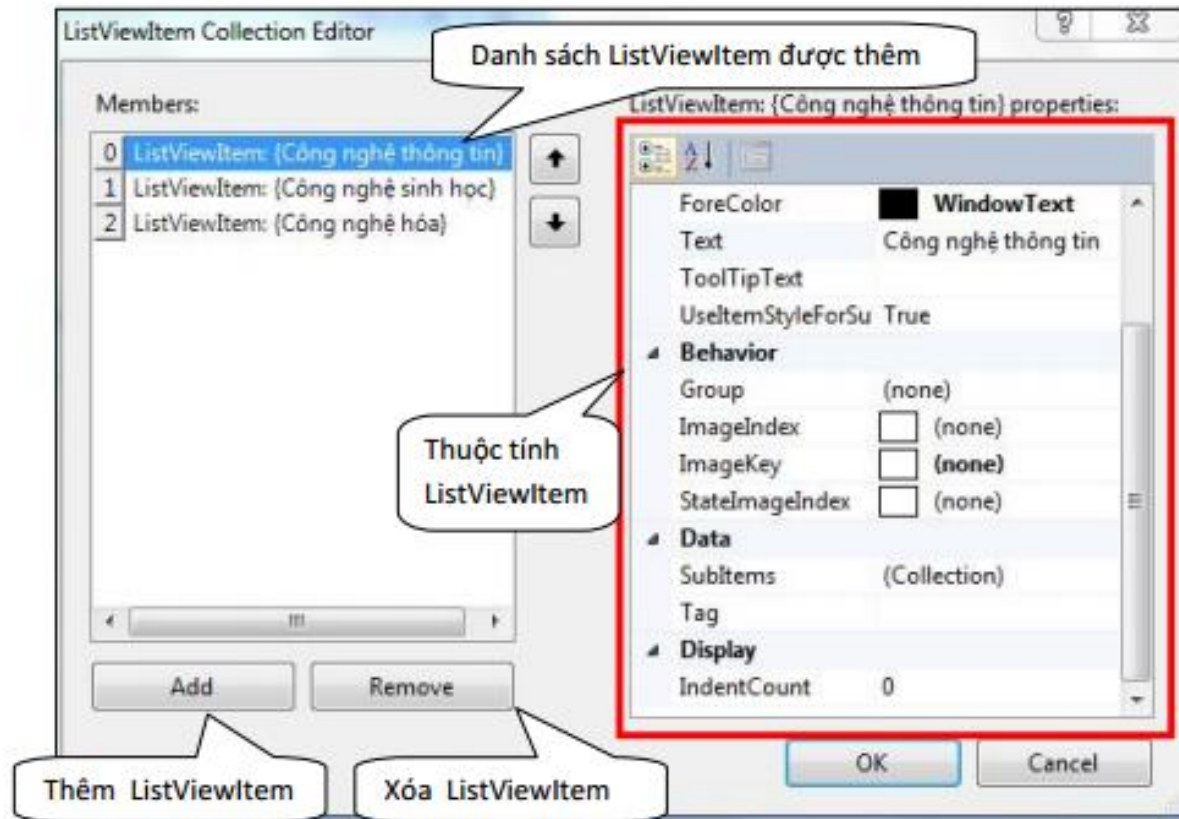
4.4.3. Điều khiển ListView

- Ta có thể thêm ListViewItem vào ListView bằng cách chọn thuộc tính Items trong cửa Properties của ListView



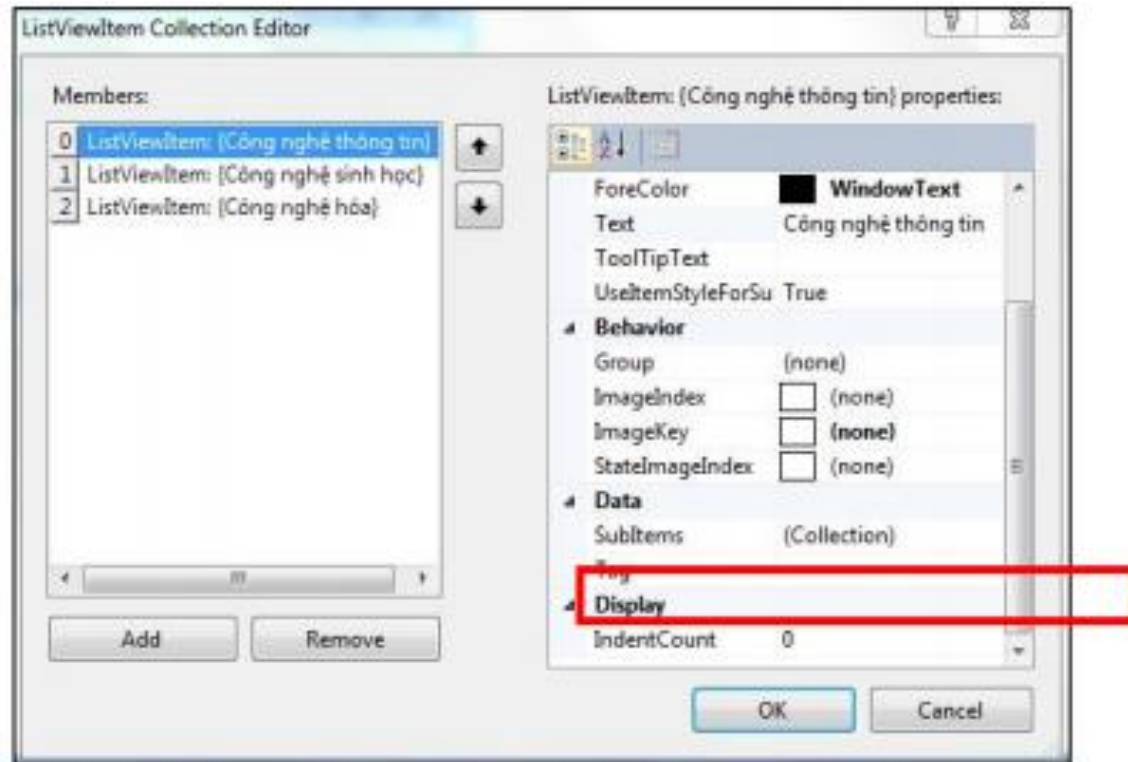
4.4.3. Điều khiển ListView

- Sau khi chọn thuộc tính Items trong cửa sổ Properties, cửa sổ ListViewItem Collection Editor sẽ xuất hiện. Ta có thể thêm hoặc xóa ListViewItem trong ListView bằng cách nhấn nút Add hoặc Remove.



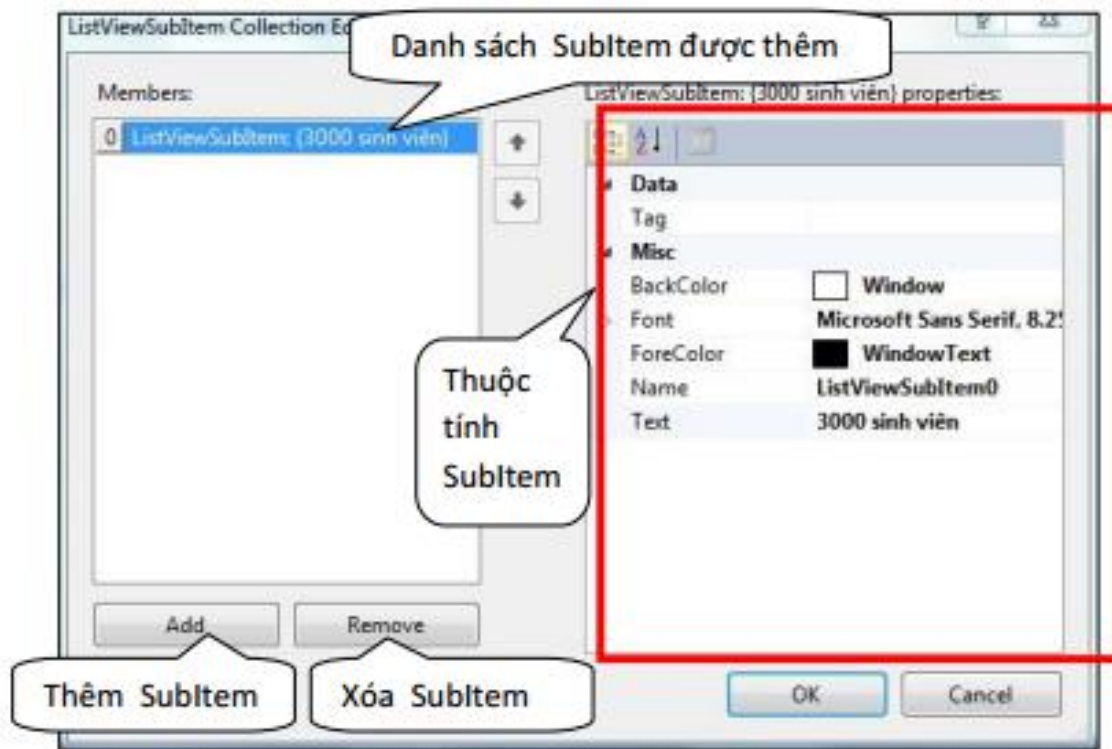
4.4.3. Điều khiển ListView

- Thêm các SubItem của Item trong ListViewItem bằng cách chọn thuộc tính SubItems trong cửa sổ *ListViewItem Collection Editor*



4.4.3. Điều khiển ListView

- Sau khi chọn thuộc tính SubItems trong cửa sổ *ListViewItem Collection Editor*, cửa sổ *ListViewSubItem Collection Editor* được hiển thị cho phép ta thêm hoặc xóa các SubItem



4.4.3. Điều khiển ListView

- Một số thuộc tính thường dùng của *ListView*

Thuộc tính	Mô tả
<i>View</i>	<p>Thuộc tính <i>View</i> qui định cách hiển thị các <i>Item</i> trong <i>ListView</i>. Thuộc tính <i>View</i> có 5 giá trị:</p> <ul style="list-style-type: none">- <i>Detail</i>: Một Icon (Icon lấy từ <i>ImageList</i>) và <i>Text</i> được hiển thị ở cột đầu tiên. Tiếp theo là các <i>SubItem</i> được hiển thị ở các cột tiếp theo. Tuy nhiên để hiển thị Item dạng <i>Detail</i> thì tạo thêm <i>Column Header</i> cho <i>ListView</i>- <i>LargeIcons</i>: Một biểu tượng lớn biểu diễn cho mỗi- <i>Item</i> cùng với một nhãn ngay dưới icon. Các biểu tượng lớn này được lấy từ điều khiển <i>ImageList</i>, và được thiết lập trong thuộc tính <i>LargeImageList</i> của <i>ListView</i>.

4.4.3. Điều khiển ListView

Thuộc tính	Mô tả
<i>View</i>	<ul style="list-style-type: none">- <i>List</i>: Mỗi <i>Item</i> sẽ được hiển thị như một biểu tượng nhỏ với một nhãn ở bên phải. Các biểu tượng trong <i>ListView</i> được sắp xếp theo các cột.- <i>SmallIcons</i>: Mỗi <i>Item</i> nằm trong một cột gồm có biểu tượng nhỏ cùng với nhãn. Các biểu tượng lớn này được lấy từ điều khiển <i>ImageList</i>, và được thiết lập trong thuộc tính <i>SmallImageList</i> của <i>ListView</i>.- <i>Tiles</i>: Mỗi một <i>Item</i> sẽ hiển thị với biểu tượng có kích thước là tối đa cùng với một <i>label</i> và các <i>subitem</i> sẽ hiển thị các cột bên phải.
<i>Items</i>	<p>Trả về các <i>Item</i> chứa trong <i>ListView</i>. Một số phương thức và thuộc tính thường dùng của <i>ListView.Items</i>:</p> <ul style="list-style-type: none">- <i>Count</i>: Đếm số lượng <i>Item</i> trong <i>ListView</i>- <i>Insert(i, <Item mới>)</i>: Chèn thêm <i>Item</i> < <i>Item mới</i>> vào vị trí <i>i</i> trong <i>ListView</i>

4.4.3. Điều khiển ListView

Thuộc tính	Mô tả
<i>Items</i>	<ul style="list-style-type: none">- <i>Add(<Item mới>)</i>: thêm Item <Item mới> vào cuối ListView- <i>Remove(<Item cần xóa>)</i>: Xóa Item <Item cần xóa> khỏi ListView- <i>RemoveAt(i)</i>: Xóa Item có chỉ số i khỏi ListView- <i>Contains(<Item cần tìm>)</i>: Trả về <i>True</i> nếu tìm thấy <Item cần tìm> trong ListView, trả về <i>False</i> nếu không có trong ListView- <i>IndexOf(<Item cần tìm>)</i>: Nếu <Item cần tìm> có trong ListView thì trả về chỉ số của Item tìm thấy trong ListView, nếu không tìm thấy sẽ trả về -1
<i>MultiSelect</i>	True/ False: Cho phép hoặc không cho phép chọn một lúc nhiều Item trong ListView
<i>FullRowSelect</i>	Khi chọn dòng dữ liệu highlighted cả dòng hay chỉ ô được chọn

4.4.3. Điều khiển ListView

Thuộc tính	Mô tả
<i>GridLines</i>	Nếu thiết lập True sẽ hiển thị các dòng và cột dạng lưới, thiết lập False không hiển thị dạng lưới
<i>SelectedItems</i>	Trả về tập các Items được chọn trong <i>ListView</i>
<i>LargeImageIcon</i>	Gán đối tượng <i>ImageList</i> cho <i>ListView</i>
<i>SmallImageIcon</i>	Gán đối tượng <i>ImageList</i> cho <i>ListView</i>
<i>FocusedItem.Index</i>	Trả về chỉ số dòng được chọn trong <i>ListView</i>
<i>SelectedIndices.Count</i>	Trả về số lượng Item được chọn trong <i>ListView</i>
<i>SelectedIndices</i>	Trả về danh sách chỉ mục các Item được chọn. Ví dụ: <code>myListView.SeletedIndices[0]</code> : trả về chỉ mục của Item đầu tiên được chọn trong danh sách các Item được chọn trong <i>ListView</i>

4.4.3. Điều khiển ListView

- Một số phương thức thường dùng của *ListView*

Phương thức	Mô tả
<i>Clear()</i>	Xóa tất cả các <i>Item</i> và <i>Column</i> trong <i>ListView</i>
<i>Sort()</i>	Sắp xếp các <i>Item</i> trong <i>ListView</i>
<i>GetItemAt(x,y)</i>	Lấy <i>Item</i> tại vị trí toạ độ x và y (x và y có thể lấy được thông qua sự kiện <i>Click</i> chuột)

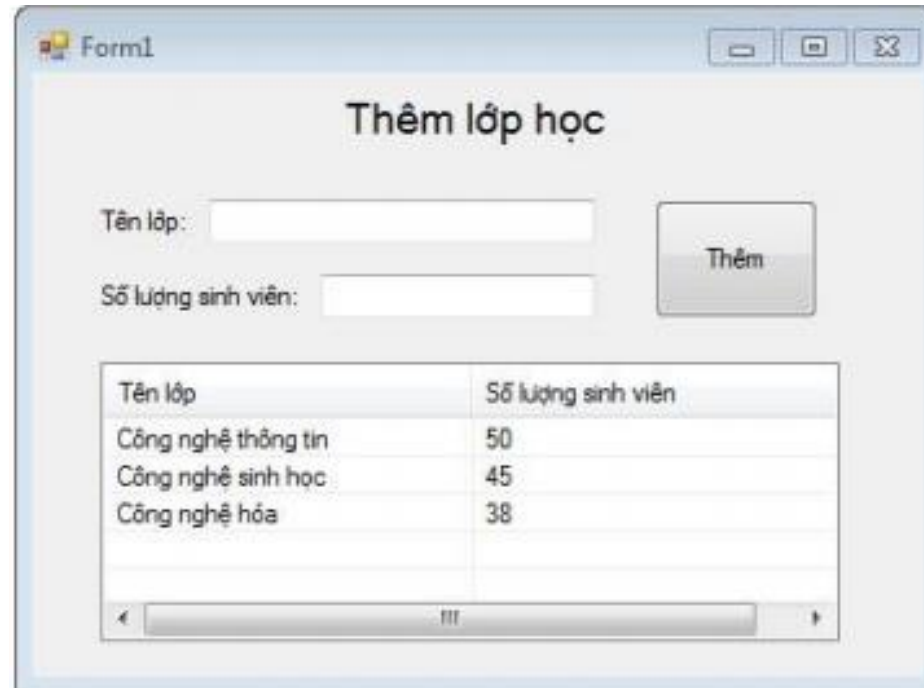
4.4.3. Điều khiển ListView

- Một số sự kiện thường dùng của *ListView*

Sự kiện	Mô tả
<i>SelectedIndexChanged</i>	Sự kiện phát sinh khi có sự thay đổi về chỉ mục được chọn của <i>Item</i> trên <i>ListView</i>
<i>ItemSelectionChanged</i>	Sự kiện phát sinh khi có sự thay đổi lựa chọn một <i>Item</i> trên <i>ListView</i>
<i>ItemCheck</i>	Xảy ra khi trạng thái chọn của <i>Item</i> thay đổi
<i>ColumnClick</i>	Sự kiện phát sinh khi một <i>column</i> trong <i>ListView</i> được <i>click</i>
<i>MouseClicked</i>	Sự kiện phát sinh khi nhấp chuột chọn một <i>Item</i> trong <i>ListView</i>

4.4.3. Điều khiển ListView

- Ví dụ: Thiết kế giao diện form như hình



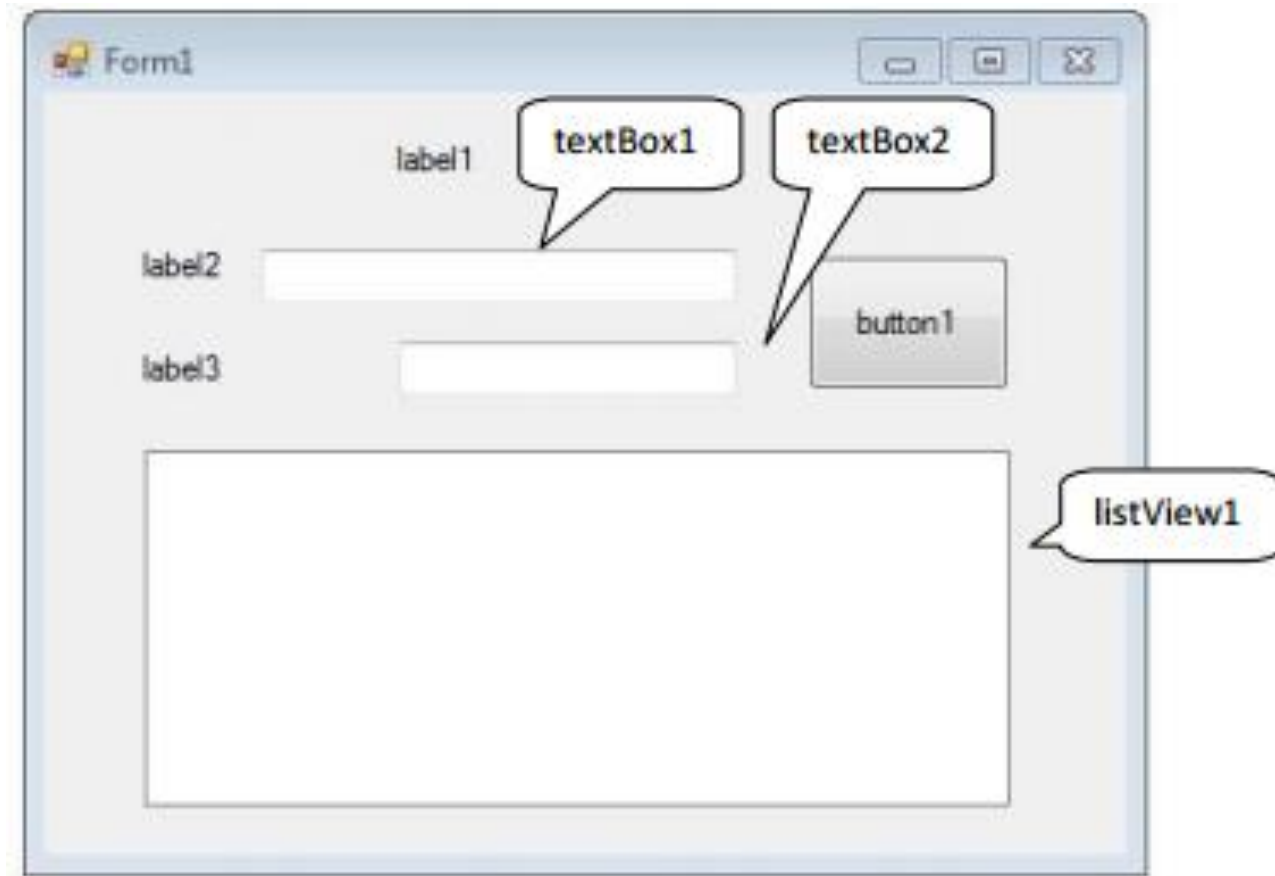
The screenshot shows a Windows application window titled 'Form1'. Inside the window, the title 'Thêm lớp học' is centered at the top. Below the title, there are two text input fields: 'Tên lớp:' and 'Số lượng sinh viên:'. To the right of these fields is a button labeled 'Thêm'. Below the input fields is a ListView control displaying a table with two columns: 'Tên lớp' and 'Số lượng sinh viên'. The table contains three rows of data: 'Công nghệ thông tin' with 50 students, 'Công nghệ sinh học' with 45 students, and 'Công nghệ hóa' with 38 students. The ListView has a scrollbar at the bottom.

Tên lớp	Số lượng sinh viên
Công nghệ thông tin	50
Công nghệ sinh học	45
Công nghệ hóa	38

- Yêu cầu: Khi người dùng nhập xong tên lớp và số lượng sinh viên, sau đó nhấn nút “Thêm” thì trong ListView sẽ chèn một dòng vào cuối với tên lớp và số lượng vừa nhập.

4.4.3. Điều khiển ListView

- Bước 1: Thiết kế giao diện ban đầu như hình



4.4.3. Điều khiển ListView

✓ button1:

- Thuộc tính Text: “Thêm”
- Thuộc tính Name: btnThem

✓ listView1:

- Thuộc tính Name: myListView
- Thuộc tính View: Details
- Thuộc tính GridLines: True

✓ textBox1:

- Thuộc tính Name: txtTenLop

✓ textBox2:

- Thuộc tính Name: txtSoLuong

4.4.3. Điều khiển ListView

- Sự kiện *Click* của nút btnThem

```
private void btnThem_Click(object sender, EventArgs e)
{
    if (txtSoLuong.Text != "" && txtTenLop.Text != "" )
    {
        ListViewItem LVItem = new
        ListViewItem(txtTenLop.Text);
        ListViewItem.ListViewSubItem LVSubItem = new
        ListViewItem.ListViewSubItem(LVItem, txtSoLuong.Text);
        LVItem.SubItems.Add(LVSubItem);
        myListView.Items.Add(LVItem);
        txtSoLuong.Text = "";
        txtTenLop.Text = "";
    }
}
```

4.4.3. Điều khiển ListView

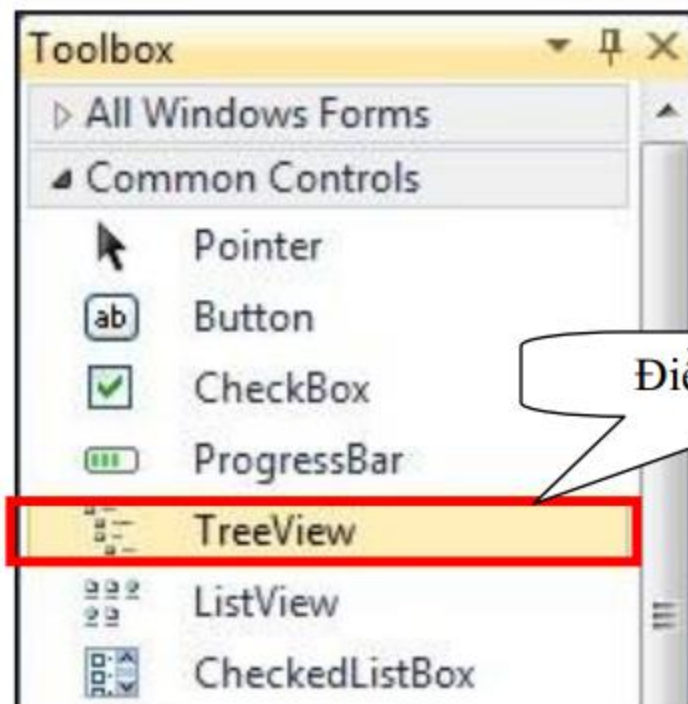
- Sự kiện *Load* của Form1

```
private void Form1_Load(object sender, EventArgs e)
{
    myListView.Columns.Add("Tên lớp", 160);
    myListView.Columns.Add("Số lượng sinh viên", 180);
}
```

4.4.4. Điều khiển TreeView

- TreeView là điều khiển dùng để hiển thị danh sách các đối tượng dưới dạng phân cấp.
- Đối tượng trong TreeView thường được gọi là node và cấu trúc phân cấp của TreeView được biểu diễn bởi lớp TreeNode.
- Mỗi một node trong TreeView có thể chứa các node khác. Node chứa một node khác gọi là node cha (RootNode) và node được chứa gọi là node con (ChildNode).
- Việc sử dụng điều khiển TreeView để hiển thị rất hữu ích, vì trình bày theo dạng phân cấp giúp việc hiển thị được rõ ràng và có hệ thống hơn

4.4.4. Điều khiển TreeView



Điều khiển *TreeView*

4.4.4. Điều khiển TreeView

- Một số thuộc tính thường dùng của *TreeView*

Thuộc tính	Mô tả
<i>Node</i>	Trả về một đối tượng thuộc lớp <i>TreeNode</i>
<i>SelectedNode</i>	Trả về node đang được chọn trong <i>TreeView</i>
<i>ShowPlusMinus</i>	Hiển thị dấu + và – trước mỗi <i>TreeNode</i>
<i>ShowRootLines</i>	Hiển thị đường thẳng nối giữa các <i>Root Node</i> trong một <i>TreeView</i>
<i>ImageList</i>	Hiển thị hình trước mỗi node trong <i>TreeView</i> . Lưu ý: Phải sử dụng thêm điều khiển <i>ImageList</i> , và gán tên đối tượng của điều khiển <i>ImageList</i> cho thuộc tính <i>ImageList</i> của <i>TreeView</i>

4.4.4. Điều khiển TreeView

Thuộc tính	Mô tả
<i>ImageIndex</i>	Giá trị của thuộc tính ImageIndex là chỉ số của hình trong điều khiển ImageList. Khi gán chỉ số cho thuộc tính ImageIndex thì hình hiển thị trước mỗi node sẽ là hình có chỉ số tương ứng. Lưu ý: Phải sử dụng thuộc tính ImageList trước
<i>SelectedImageIndex</i>	Giá trị của thuộc tính SelectImageIndex là chỉ số của hình trong điều khiển ImageList. Khi người dùng chọn node nào thì node đó sẽ có hình tương ứng như thuộc tính <i>SelectedImageIndex</i> chỉ định

4.4.4. Điều khiển TreeView

- Một số phương thức thường dùng của *TreeView*

Phương thức	Mô tả
<i>GetNodeCount()</i>	Đếm số node trong một TreeView
<i>ExpandAll()</i>	Hiển thị tất cả các node trên TreeView
<i>CollapseAll()</i>	Thu gọn tất cả các node trên TreeView
<i>GetNodeAt(x,y)</i>	Lấy một node tại một vị trí có tọa độ (x, y) trên màn hình. Lưu ý: Thường sử dụng sự kiện MouseDown hoặc NodeMouseClicked

4.4.4. Điều khiển TreeView

- Một số sự kiện thường dùng của *TreeView*

Sự kiện	Mô tả
<i>AfterCollapse</i>	Phát sinh khi thu gọn một <i>TreeNode</i>
<i>AfterExpand</i>	Phát sinh khi hiển thị các node trong <i>TreeNode</i>
<i>AfterSelect</i>	Phát sinh khi chọn một <i>TreeNode</i>
<i>NodeMouseClick</i>	Phát sinh khi chọn một node

- *TreeView* là điều khiển để hiển thị các node, tuy nhiên việc hiển thị này thực chất là do *TreeNode* tạo ra. Do đó để làm việc với các node cần sử dụng các thuộc tính và phương thức của lớp *TreeNode*

4.4.4. Điều khiển TreeView

- Một số thuộc tính thường dùng của *TreeNode*

Thuộc tính	Mô tả
<i>Nodes</i>	Trả về tập các node
<i>Text</i>	Đọc/ gán chuỗi ký tự người dùng sẽ nhìn thấy ở mỗi node
<i>FirstNode</i>	Trả về node đầu tiên
<i>LastNode</i>	Trả về node cuối cùng
<i>NextNode</i>	Chuyển đến node tiếp theo
<i>PrevNode</i>	Lùi lại node trước đó
<i>Parent</i>	Trả về node cha của node hiện tại
<i>Index</i>	Trả về chỉ số của node

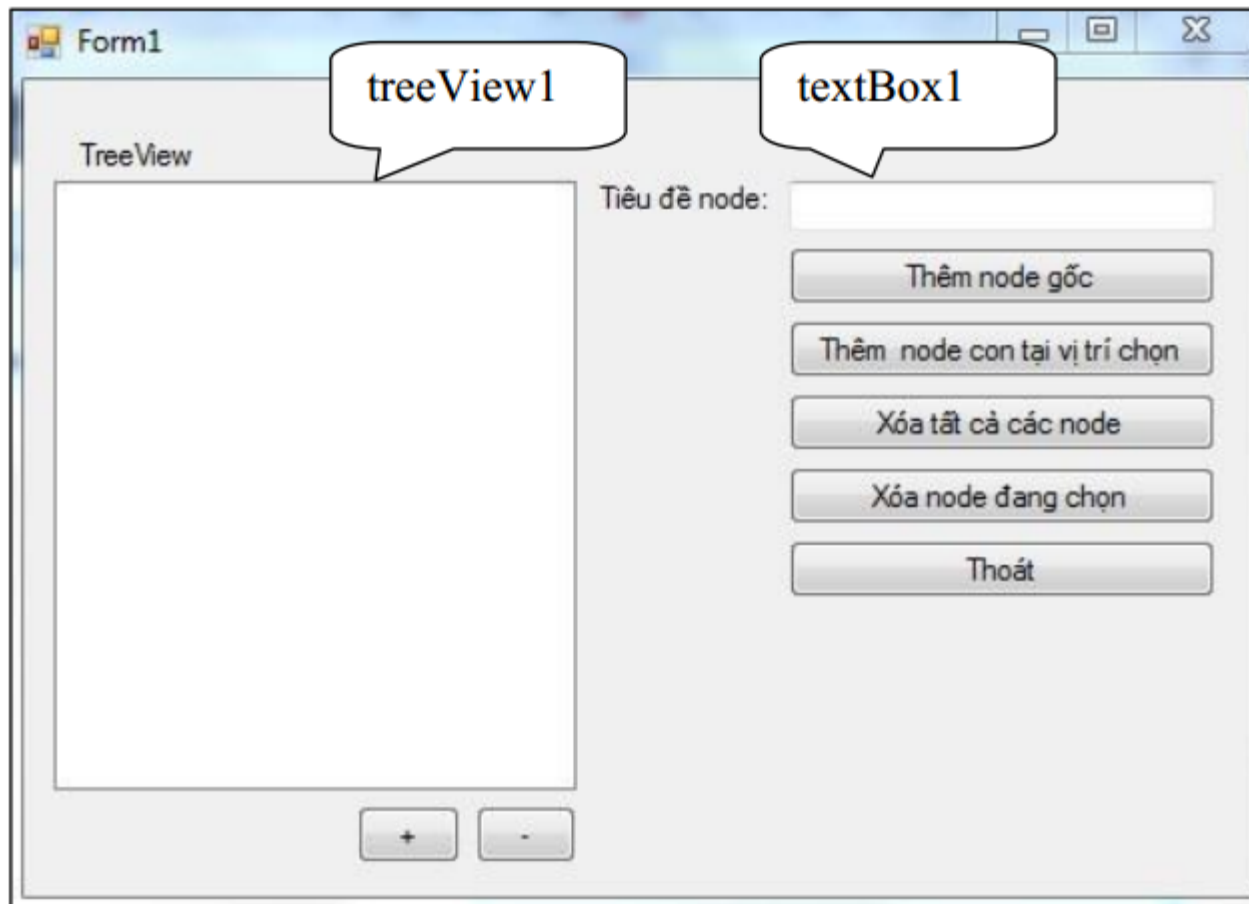
4.4.4. Điều khiển TreeView

- Một số phương thức thường dùng của *TreeNode*

Phương thức	Mô tả
<i>Nodes.Add</i>	Thêm một node
<i>Nodes.Remove</i>	Xóa một node
<i>Nodes.Insert</i>	Chèn vào một node (chèn trước, chèn sau một node)
<i>Nodes.Clear</i>	Xóa tất cả các node con và node hiện tại

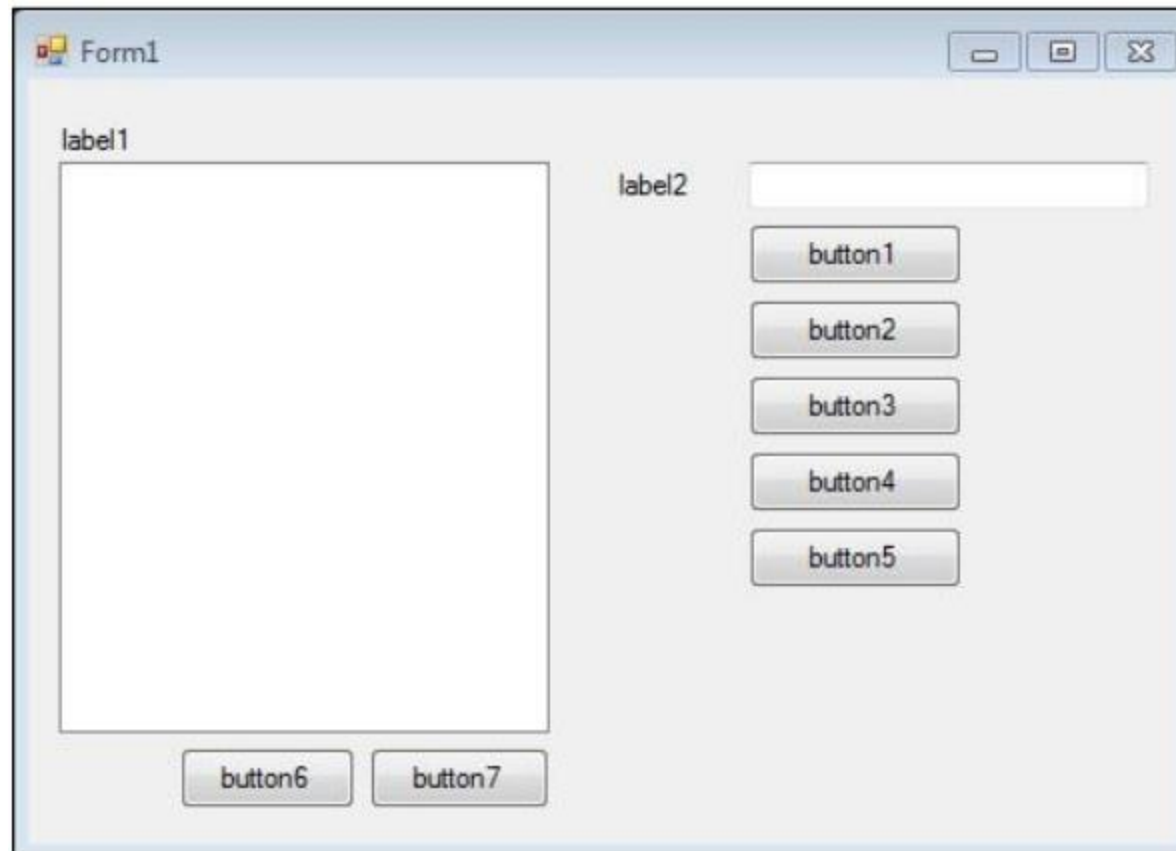
4.4.4. Điều khiển TreeView

- Ví dụ: Viết chương trình minh họa việc thêm sửa xóa các node trong một TreeView



4.4.4. Điều khiển TreeView

- Bước 1: Thiết kế giao diện ban đầu như hình



- label1:
 - Thuộc tính Text: “TreeView”
- label2:
 - Thuộc tính Text: “Tiêu đề node:”
- treeView1:
 - Thuộc tính Name: TV_Test
- textBox1:
 - Thuộc tính Name: txtTieuDe
- button1:
 - Thuộc tính Text: “Thêm node gốc”
 - Thuộc tính Name: btnThemGoc
- button2:
 - Thuộc tính Text: “Thêm node con tại vị trí”
 - Thuộc tính Name: btnThemCon

- button3:
 - Thuộc tính Text: “Xóa tất cả các node”
 - Thuộc tính Name: btnXoaTatCa
- button4:
 - Thuộc tính Text: “Xóa node đang chọn”
 - Thuộc tính Name: btnXoaChon
- button5:
 - Thuộc tính Text: “Thoát”
 - Thuộc tính Name: btnThoat
- button6:
 - Thuộc tính Text: “+”
 - Thuộc tính Name: btnMoRong
- button7:
 - Thuộc tính Text: “-”
 - Thuộc tính Name: btnThuNho

4.4.4. Điều khiển TreeView

- Sự kiện *Click* của nút btnThemGoc

```
private void btnThemGoc_Click(object sender, EventArgs e)
{
    TV_Test.Nodes.Add(txtTieuDe.Text);
    txtTieuDe.Text = "";
}
```

- Sự kiện Click của nút btnThemCon

```
private void btnThemCon_Click(object sender, EventArgs e)
{
    TV_Test.SelectedNode.Nodes.Add(txtTieuDe.Text);
    txtTieuDe.Text = "";
    TV_Test.ExpandAll();
}
```

4.4.4. Điều khiển TreeView

- Sự kiện *Click* của nút btnXoaTaCa

```
private void btnXoaTaCa_Click(object sender, EventArgs e)
{
    TV_Test.Nodes.Clear();
}
```

- Sự kiện *Click* của nút btnXoaChon

```
private void btnXoaChon_Click(object sender, EventArgs e)
{
    TV_Test.SelectedNode.Remove();
}
```

4.4.4. Điều khiển TreeView

- Sự kiện *Click* của nút btnMoRong

```
private void btnMoRong_Click(object sender, EventArgs e)
{
    TV_Test. ExpandAll();
}
```

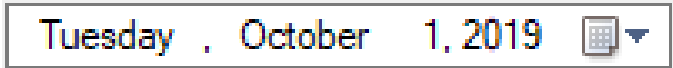
- Sự kiện *Click* của nút btnThuNho

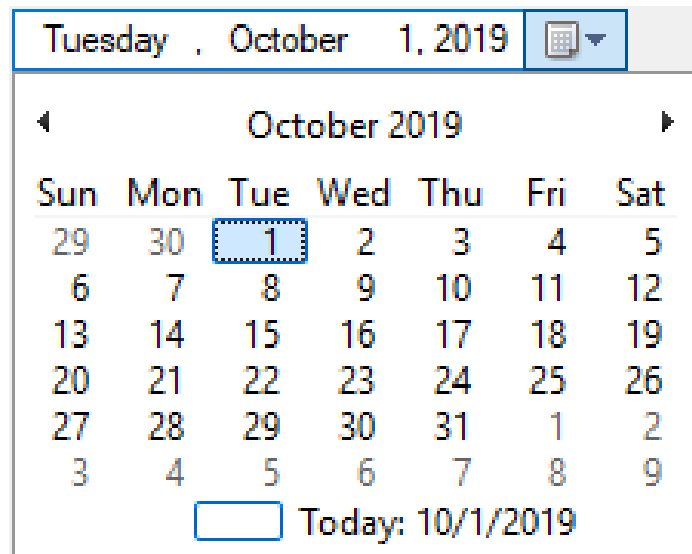
```
private void btnThuNho_Click(object sender, EventArgs e)
{
    TV_Test.CollapseAll();
}
```

4.4.5. Điều khiển DateTimePicker, MonthlyCalendar

- 4.4.5.1. Điều khiển DateTimePicker
- 4.4.5.2. Điều khiển MonthlyCalendar

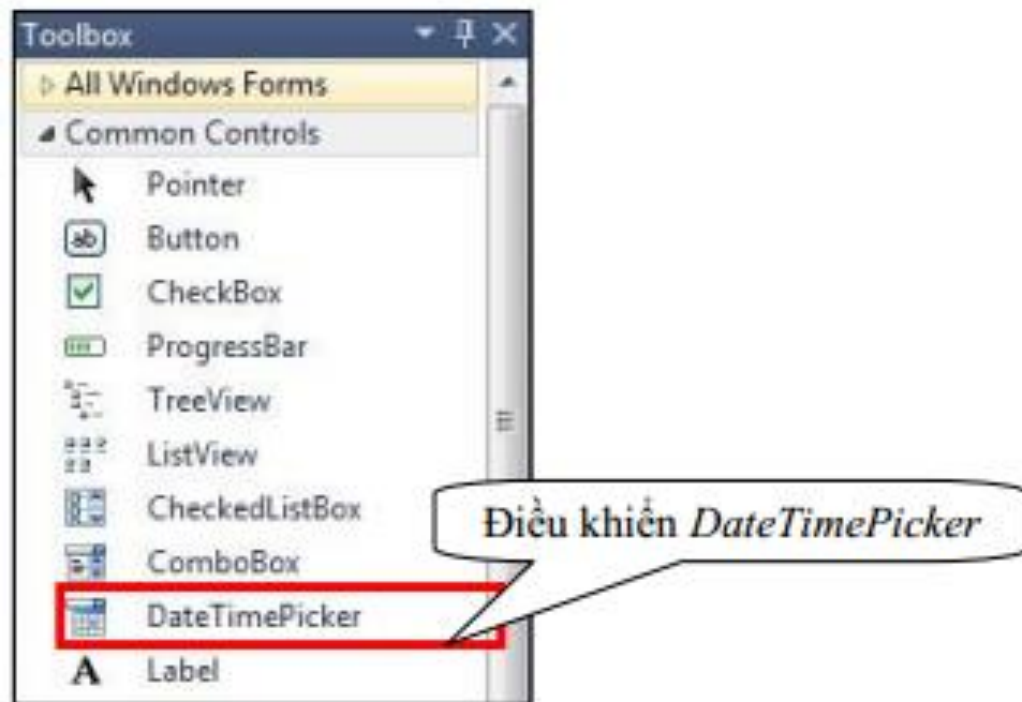
4.4.5.1. Điều khiển DateTimePicker

- Điều khiển *DateTimePicker* cho phép người dùng chọn ngày tháng như một lịch biểu nhưng biểu diễn ở dạng ComboBox 
- Khi người dùng nhấp chuột vào ComboBox sẽ sổ xuống lịch biểu như hình



4.4.5.1. Điều khiển DateTimePicker

- Các đối tượng ngày tháng biểu diễn trong DateTimePicker thực chất là các đối tượng thuộc lớp DateTime



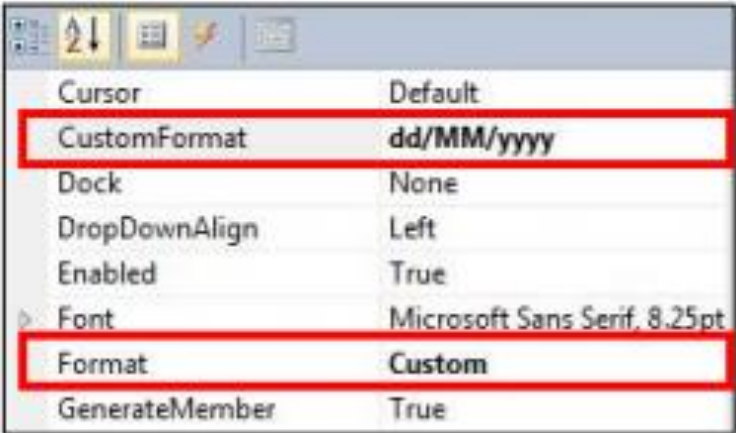
4.4.5.1. Điều khiển DateTimePicker

- Một số thuộc tính thường dùng của *DateTimePicker*

Thuộc tính	Mô tả
<i>Format</i>	Định dạng kiểu hiển thị của ngày tháng. Lưu ý: Thường sử dụng giá trị kiểu Short
<i>Value</i>	Trả về giá trị hiện thời của điều khiển DateTimePicker
<i>Value.Date</i>	Trả về ngày tháng năm
<i>Value.Day</i>	Trả về ngày của tháng
<i>Value.Month</i>	Trả về tháng
<i>Value.Year</i>	Trả về năm
<i>Value.DateOfWeek</i>	Trả về ngày của tuần (0 là chủ nhật, 1 là thứ 2, 2 là thứ 3, ... 6 là thứ 7)

4.4.5.1. Điều khiển DateTimePicker

Thuộc tính	Mô tả
<i>Value.DateOfYear</i>	Trả về ngày thứ bao nhiêu của năm
<i>CustomFormat</i>	<p>Cho phép lập trình viên tạo ra một định dạng khác về ngày tháng.</p> <p>Lưu ý: Định dạng ngày tháng năm như kiểu Việt Nam thì kiểu định dạng phải là dd/MM/yyyy. Khi đó thuộc tính format phải thiết lập là Custom.</p>



The screenshot shows the Properties window for a DateTimePicker control. The 'CustomFormat' property is set to 'dd/MM/yyyy' and the 'Format' property is set to 'Custom'. Both are highlighted with red rectangles. Other visible properties include Cursor (Default), Dock (None), DropDownAlign (Left), Enabled (True), Font (Microsoft Sans Serif, 8.25pt), and GenerateMember (True).

4.4.5.1. Điều khiển DateTimePicker

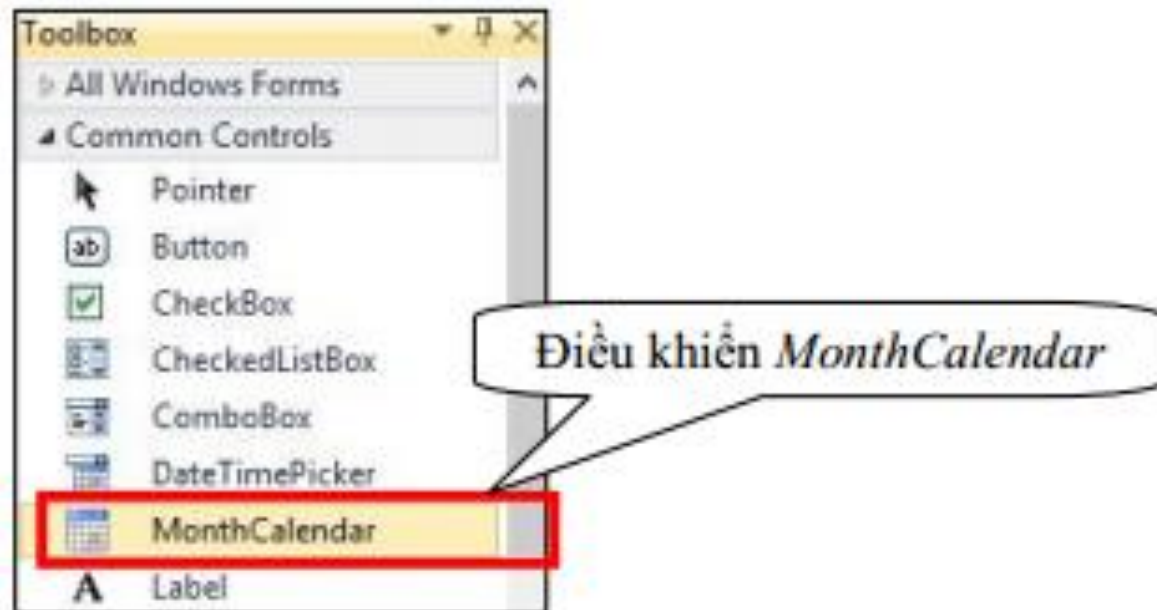
Thuộc tính	Mô tả
<i>MaxDate</i>	Thiết lập ngày lớn nhất cho phép người dùng chọn trên điều khiển <i>DateTimePicker</i>
<i>MinDate</i>	Thiết lập ngày nhỏ nhất cho phép người dùng chọn trên điều khiển <i>DateTimePicker</i>
<i>Text</i>	Trả về ngày hiển thị

- Một số sự kiện thường dùng của *DateTimePicker*.

Sự kiện	Mô tả
<i>ValueChanged</i>	Phát sinh khi người dùng chọn giá trị khác với giá trị trước đó trên điều khiển <i>DateTimePicker</i>
<i>CloseUp</i>	Phát sinh người dùng kết thúc việc chọn ngày trên điều khiển <i>DateTimePicker</i>

4.4.5.2. Điều khiển *MonthlyCalendar*

- *MonthCalendar* là điều khiển hiển thị lịch dưới dạng một lịch biểu cho phép người dùng chọn ngày tháng. Nhưng khác biệt là *MonthCalendar* cho phép người dùng có thể chọn một tập các ngày hay nói cách khác là một tập các đối tượng thuộc lớp *DateTime*



4.4.5.2. Điều khiển *MonthlyCalendar*

- Một số thuộc tính thường dùng của *MonthCalendar*

Thuộc tính	Mô tả
<i>MaxDate</i>	Thiết lập ngày lớn nhất cho phép người dùng chọn trên điều khiển <i>MonthCalendar</i>
<i>MinDate</i>	Thiết lập ngày nhỏ nhất cho phép người dùng chọn trên điều khiển <i>MonthCalendar</i>
<i>SelectionRange</i>	Trả về một dãy các ngày liên tục được chọn bởi người dùng
<i>SelectionStart</i>	Trả về ngày đầu tiên trong dãy tại thuộc tính <i>SelectionRange</i>

4.4.5.2. Điều khiển **MonthlyCalendar**

Thuộc tính	Mô tả
<i>SelectionEnd</i>	Trả về ngày cuối cùng trong dãy tại thuộc tính <i>SelectionRange</i>
<i>AnnuallyBoldedDates</i>	Chứa một mảng các ngày. Trong mỗi năm, các ngày trong mảng sẽ được bôi đen <i>MonthCalendar</i>
<i>BoldedDates</i>	Chứa mảng các ngày. Các ngày này sẽ được bôi đen trên điều khiển <i>MonthCalendar</i> tại những năm chỉ định.
<i>MaxSelectCount</i>	Thiết lập số lượng ngày tối đa mà người dùng có thể chọn
<i>MonthlyBoldedDates</i>	Chứa mảng các ngày. Trong mỗi tháng, các ngày trong mảng sẽ được bôi đen trên <i>MonthCalendar</i>

4.4.5.2. Điều khiển **MonthlyCalendar**

- Người dùng có thể chọn một ngày nào đó trên *MonthCalendar* bằng cách nhấp chuột chọn ngày đó hoặc chọn một dãy nhiều ngày liên tiếp bằng cách nhấp chuột chọn ngày đầu tiên và giữ phím shift đồng thời chọn ngày cuối cùng của dãy.
- Số lượng các ngày được chọn trong dãy phải nhỏ hơn giá trị thiết lập trong thuộc tính `MaxSelectCount`
- Một số sự kiện thường dùng của *MonthCalendar*

Sự kiện	Mô tả
<i>DateChanged</i>	Được phát sinh một ngày mới hoặc một dãy các ngày mới được chọn

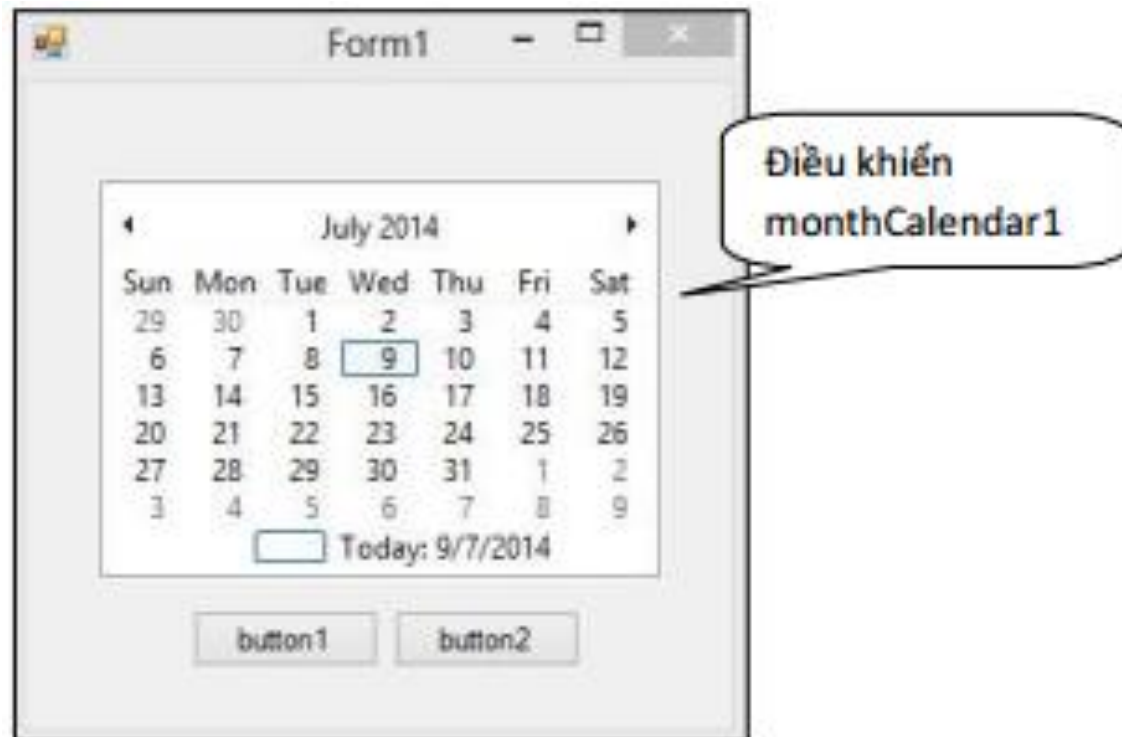
4.4.5.2. Điều khiển MonthlyCalendar

- Ví dụ: Viết chương trình minh họa việc hiển thị lịch, thiết kế giao diện form. Yêu cầu: khi nhấp nút hiển thị thì các ngày được chọn sẽ hiển thị trên MessageBox



4.4.5.2. Điều khiển MonthlyCalendar

- Bước 1: Thiết kế giao diện ban đầu như hình



4.4.5.2. Điều khiển MonthlyCalendar

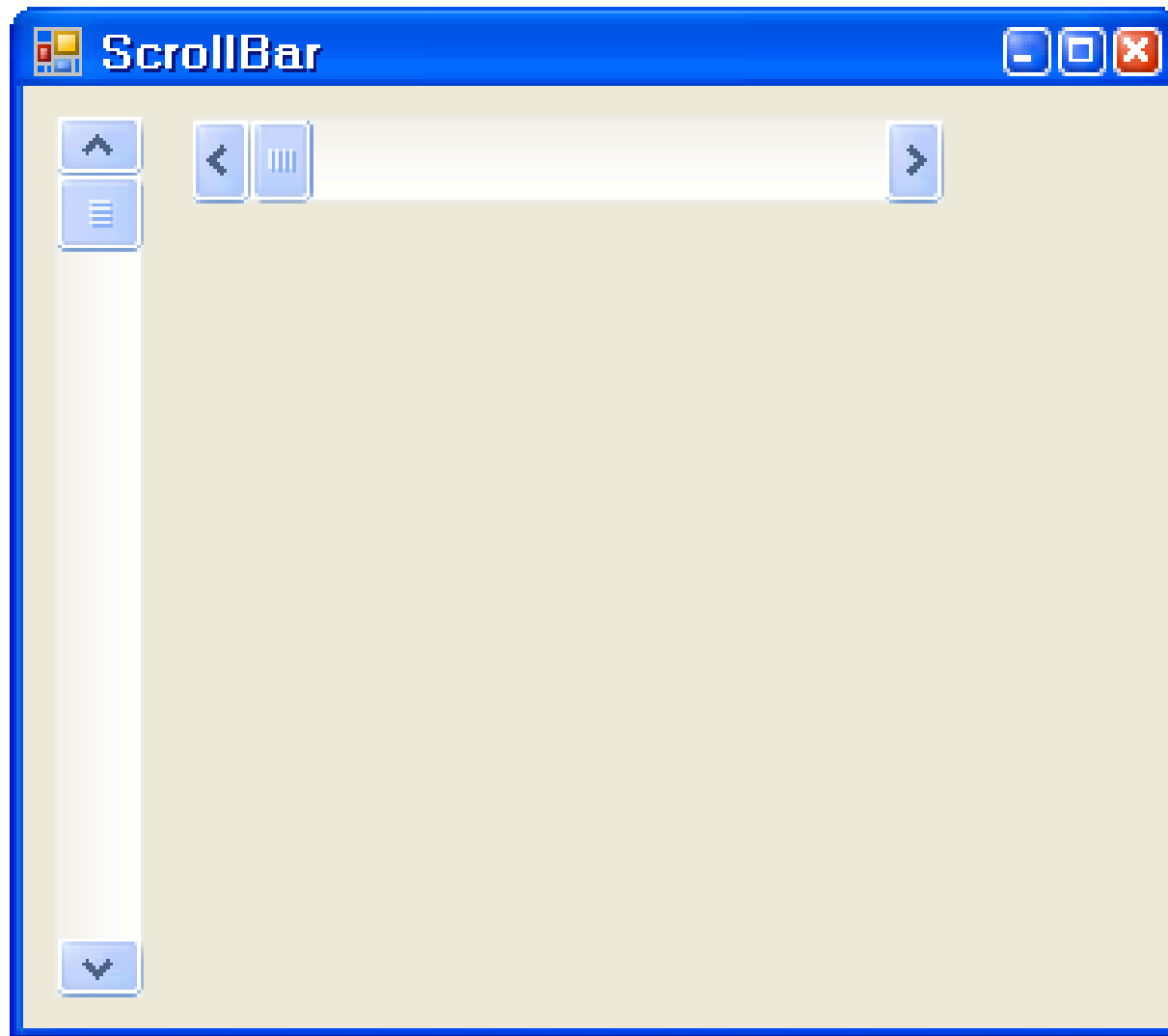
- Bước 2: Thiết lập giá trị cho thuộc tính trong cửa sổ Properties
 - ✓ button1:
 - Thuộc tính Text: “Hiển thị”
 - Thuộc tính Name: btnHienThi
 - ✓ button2:
 - Thuộc tính Text: “Thoát”
 - Thuộc tính Name: btnThoat
 - ✓ monthCalendar1:
 - Thuộc tính Name: MyMCalendar

4.4.5.2. Điều khiển MonthlyCalendar

- Sự kiện *Click* của nút btnHienThi

```
private void btnHienThi_Click(object sender, EventArgs e)
{
    string strngay="";
    DateTime i=new DateTime();
    for ( i = MyMCalendar.SelectionStart;
    i<= MyMCalendar.SelectionEnd; i=i.AddDays(1.0))
    {
        strngay += i.ToLongDateString() + "\n";
    }
    MessageBox.Show(strngay);
}
```

4.4.6 Điều khiển HScrollBar và VScrollBar



4.4.6 Điều khiển HScrollBar và VScrollBar

- Một số thuộc tính thường dùng của HScrollBar và VScrollBar

Thuộc tính	Mô tả
Value	
Maximum	
Minimum	
SmallChange	
LargeChange	

4.4.6 Điều khiển HScrollBar và VScrollBar

- Một số sự kiện thường dùng của HScrollBar và VScrollBar

Sự kiện	Mô tả
Scroll	
ValueChanged	