

**CONTROLS**

**TS. ĐẶNG THÀNH TRUNG**

## + Nội dung chính

2

- Bổ sung Control vào Window Form
- Thay đổi thuộc tính của control
- Bắt các sự kiện của Control
- Các hộp thoại
- Một số Window Form Control
- Tạo ra Menu
- Xây dựng ứng dụng MDI



# THÊM ĐIỀU KHIỂN VÀO Window Form

# + SỬ DỤNG

## Windows Form Designer

- Thành phần điều khiển (Control) được đặt lên bề mặt của đối tượng Container.
  - Đối tượng container bao gồm: Window Form, Panel, Group Box.
- Window Form Designer cung cấp công cụ ToolBox (chọn View/ToolBox hoặc Ctrl+Alt+X) có chứa các loại điều khiển khác nhau.
  - Kéo thả các điều khiển từ ToolBox lên Form và sắp xếp chúng theo ý muốn.



# BỔ SUNG CONTROL BẰNG MÃ LỆNH

- Bổ sung điều khiển lên Form bằng mã lệnh bao gồm 3 bước như sau:
  - Tạo ra biến private đại diện cho từng loại điều khiển mà ta muốn đưa vào Form.
  - Trong form, viết mã lệnh để khởi tạo từng điều khiển và thay đổi nó bằng cách sử dụng thuộc tính, phương thức và sự kiện.
  - Bổ sung từng điều khiển vào Control Collection của Form



+

**TÙY BIẾN ĐIỀU KHIỂN**

# + TÙY BIẾN Control

- Sử dụng cửa sổ Properties:
  - Chọn Control cần thay đổi thuộc tính.
  - Trên cửa sổ Properties, chọn thuộc tính cần thay đổi và gán lại giá trị thích hợp.
- Sử dụng mã lệnh:
  - Sử dụng cú pháp `ControlObject.PropertyName` để truy nhập vào thuộc tính của đối tượng control.
    - `ControlObject` là một thể hiện của `Control`
    - `PropertyName` là một thuộc tính của control.

# + MỘT SỐ THUỘC TÍNH CỦA Control

8

<b>Anchor</b>	Bám Control theo các cạnh của Form.
<b>Dock</b>	Kéo rộng Control theo các phía của Parent Control.
<b>Enabled</b>	Để xác định liệu control có thể đáp ứng tương tác của người sử dụng không.
<b>Font</b>	Gán Font chữ của văn bản hiển thị trên control. Giá trị của thuộc tính Font là một đối tượng của lớp Font. Không được thay đổi trong mã lệnh
<b>Location</b>	Xác định vị trí góc trên bên trái của control với vị trí góc trên bên trái của container control. Giá trị của thuộc tính Location là kiểu Point. 4 thuộc tính phụ thuộc vào Location là: Left (Location.X), Right (Location.X+Width), Top (Location.Y) và Bottom (Location.Y+Height).
<b>Size</b>	Xác định chiều cao và chiều rộng của control. Giá trị của thuộc tính này là kiểu Size - là một struct gồm 2 thuộc tính Height và Width.
<b>TabIndex và TabStop</b>	Xác định thứ tự nhận focus của control khi người sử dụng nhấn phím Tab. Nếu không muốn control nhận focus khi người sử dụng dùng phím Tab, gán thuộc tính TabStop của control đó bằng false.
<b>Text</b>	Là xâu ký tự có liên quan tới control. Các control khác nhau sử dụng thuộc tính Text theo các cách khác nhau:
<b>Visible</b>	Xác định khả năng nhìn thấy control vào lúc chạy chương trình.



# + BẮT SỰ KIỆN CỦA Control

- Sự kiện của control
  - Kế thừa các sự kiện của lớp System.Windows.Forms.Control
  - các sự kiện riêng của Control
- Control có một sự kiện mặc định.
  - Sự kiện Click với Button
  - Sự kiện Load của Form
  - Sự kiện CheckedChanged của CheckBox
  - ...

# + CÁC HỘP THOẠI

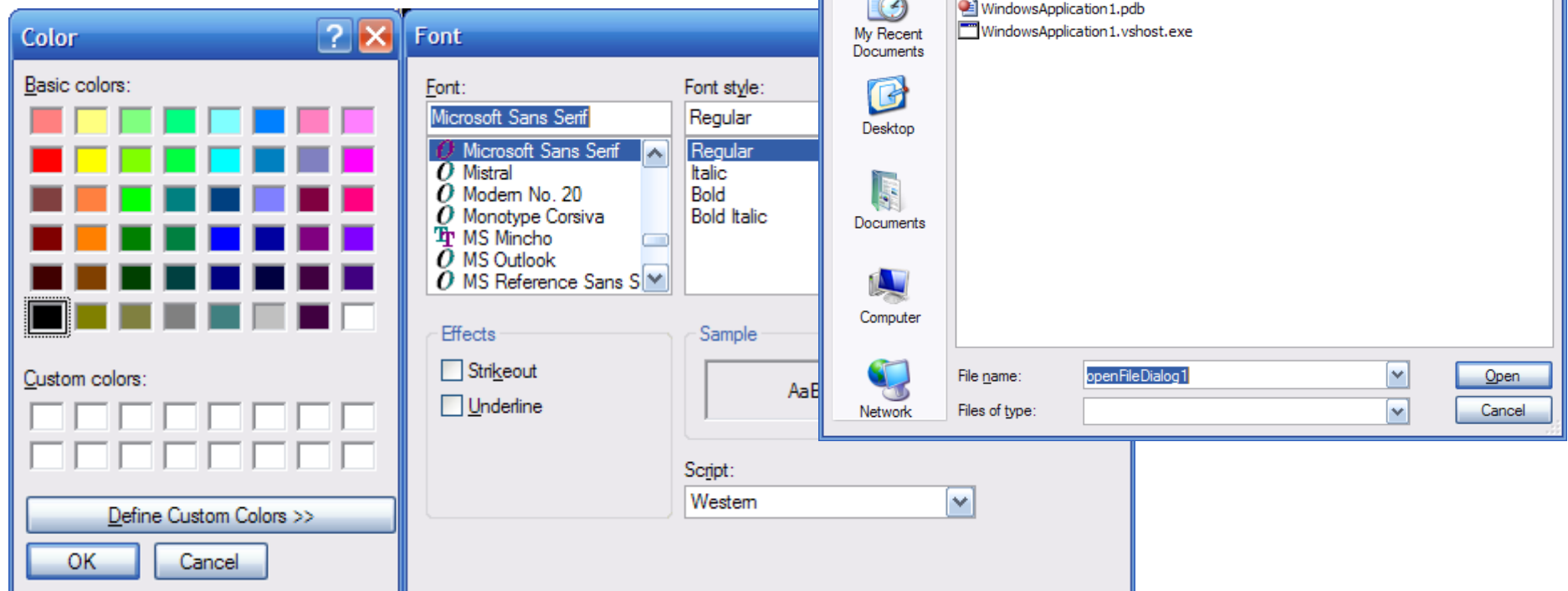
10

- Thư viện Windows Forms cung cấp các lớp Dialog Box sau:
  - ColorDialog: hiển thị các màu và trả về màu được chọn
  - FontDialog: hiển thị các font và trả về font, kích thước, ... được chọn
  - OpenFileDialog: cho phép người sử dụng duyệt file và thư mục, sau đó chọn một hoặc nhiều file.
  - PageSetupDialog: cho phép người sử dụng lựa chọn các thiết lập liên quan tới cách bố trí trang.
  - PrintDialog: cho phép người sử dụng lựa chọn các vấn đề liên quan đến in và gửi tài liệu tới máy in đã chọn.
  - PrintPreviewDialog: cho phép người sử dụng xem file trước khi in.
  - SaveFileDialog: cho phép người sử dụng duyệt file và thư mục, sau đó chọn file cần lưu.

# + CÁC HỘP THOẠI

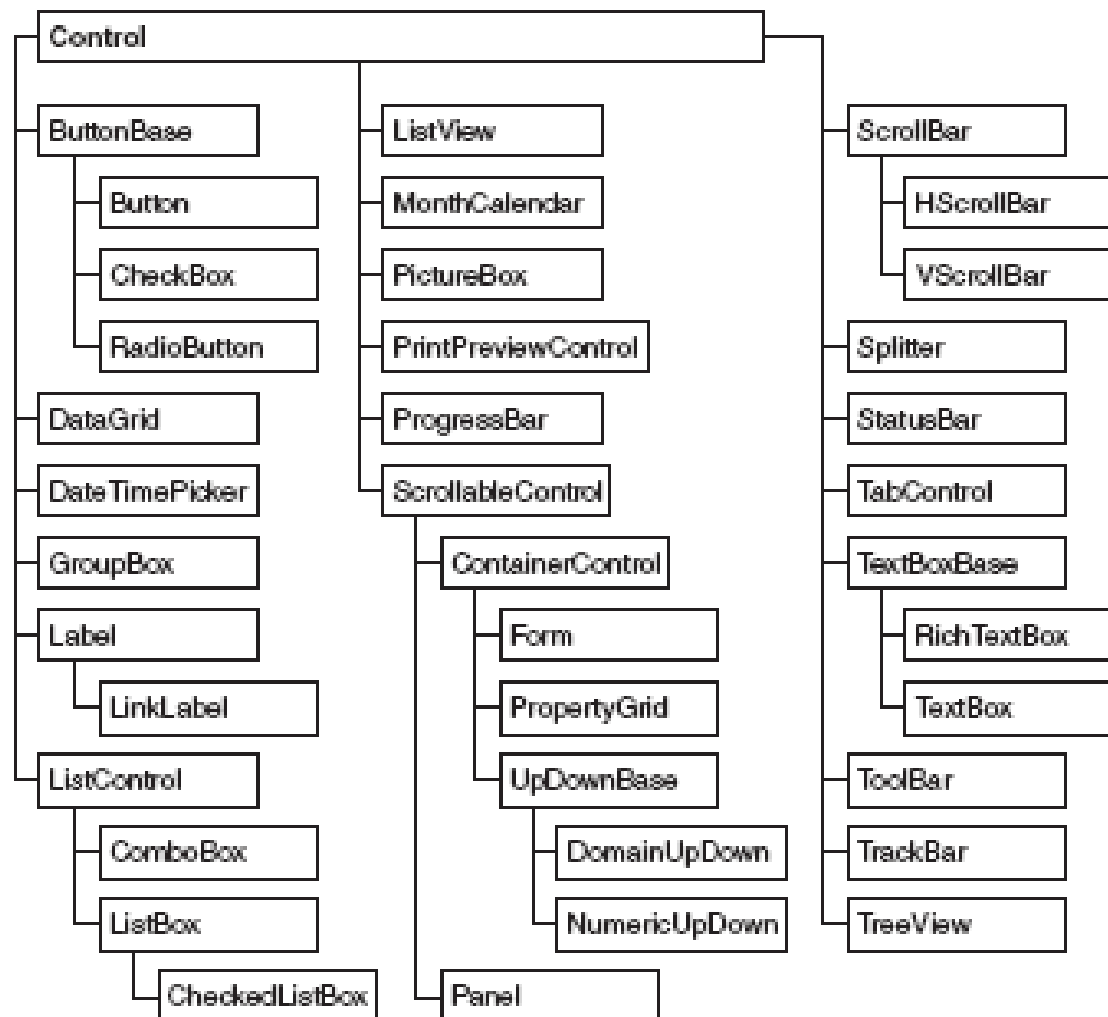
11

- Hộp thoại được chia thành 2 loại:
  - Modal - ShowDialog()
  - Modeless - Show()



# + MỘT SỐ ĐIỀU KHIỂN

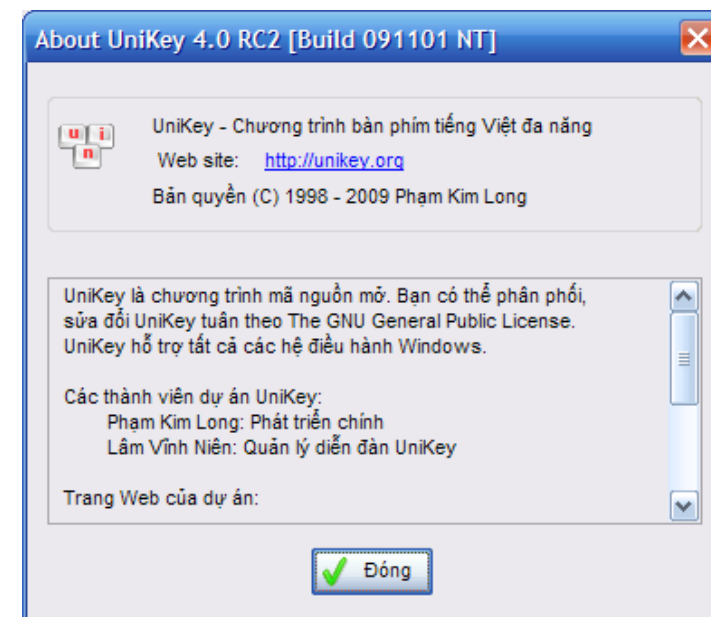
12



# + Label và LinkLabel

13

- **Label** được sử dụng để hiển thị những thông tin read-only (VD: text và image).
- **LinkLabel** thừa kế từ Label và bổ sung thêm các chức năng liên quan tới hyperlink.
  - **ActiveLinkColor**: xác định màu để hiển thị điểm link.
  - **DisabledLinkColor**: xác định màu của điểm link đã được sử dụng.
  - **LinkArea**: xác định vị trí đặt text
  - **Links**: trả về tập các đối tượng Link trong LinkLabel.
    - Lớp Link chứa các thông tin về hyperlink.
    - Thuộc tính LinkData của lớp Link cho phép ta gắn URL với hyperlink
    - LinkLabel được coi như một phần của điểm link

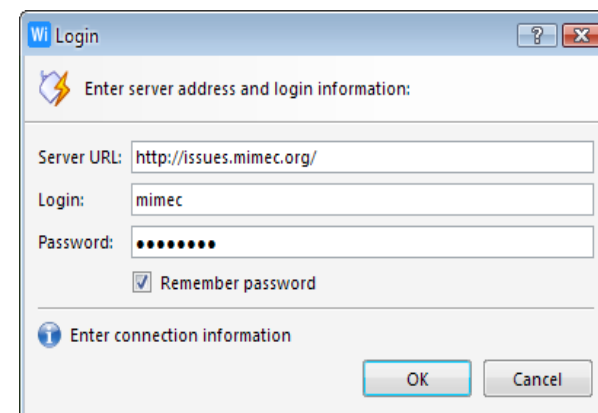


```
private void onLinkClicked(object sender, LinkLabelLinkClickedEventArgs e)
{
    System.Diagnostics.Process.Start(e.Link.LinkData.ToString());
}
```

# + TextBox và RichTextBox

14

- TextBox và RichTextBox đều thừa kế từ lớp TextBoxBase.
- TextBox cho phép người sử dụng nhập hoặc hiển thị dữ liệu.
- Một số thuộc tính của lớp TextBox:
  - AcceptReturn: chỉ áp dụng khi TextBox hiển thị nhiều dòng. Nếu có giá trị **true** cho phép xuống dòng khi nhấn phím Enter.
  - CharacterCasing: xác định ký tự nhập vào có cần thay đổi không (Lower, Normal, Upper).
  - MultiLine: xác định TextBox có hiển thị nhiều dòng không.
  - PasswordChar: ký tự hiển thị khi TextBox làm nơi nhập mật khẩu.
  - ReadOnly: không cho nhập vào TextBox.
  - ScrollBars: hiển thị thanh trượt khi TextBox có nhiều dòng.
  - Text: nội dung của TextBox.
  - WordWrap: xác định text có tự động xuống dòng hay không
  - Sự kiện TextChanged xảy ra khi nội dung của thuộc tính Text thay đổi.



# + TextBox và RichTextBox ...

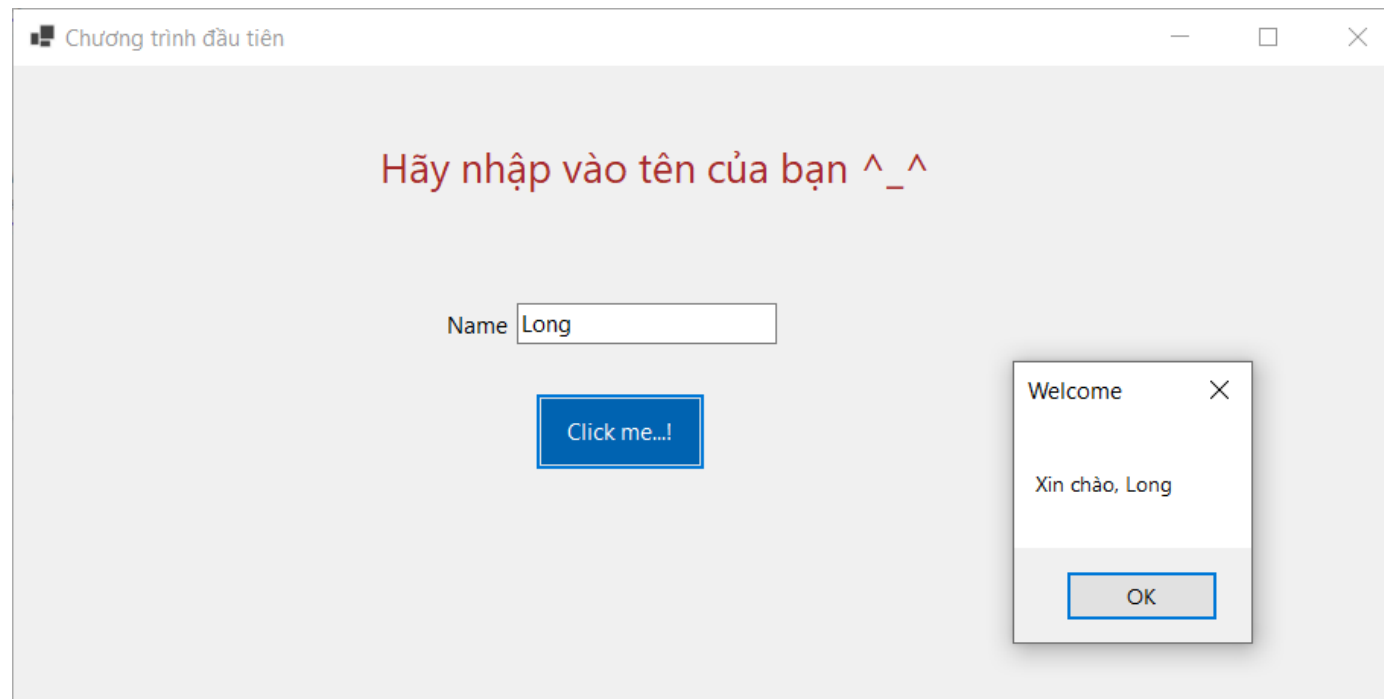
15

- RichTextBox tương tự như TextBox, nhưng nó có khả năng hiển thị nội dung phong phú hơn.
- Một số thuộc tính quan trọng của RichTextBox:
  - DetectUrls: xác định liệu control có tự phát hiện và định dạng URLs.
  - Rtf: xác định nội dung của RichTextBox bao gồm cả mã RTF.
  - SelectionColor: xác định màu của đoạn text được chọn.
  - SelectionFont: xác định font của đoạn text được chọn.
  - SelectedRtf: xác định nội dung RTF được chọn.
  - ZoomFactor: xác định mức zoom hiện tại

# + BÀI TẬP THỰC HÀNH

16

- **Bài 4.1:** Thiết lập sự kiện Click cho button Click me, khi nhấn vào sẽ hiển thị hộp thoại thông báo với lời chào kèm theo tên mà người dùng nhập.

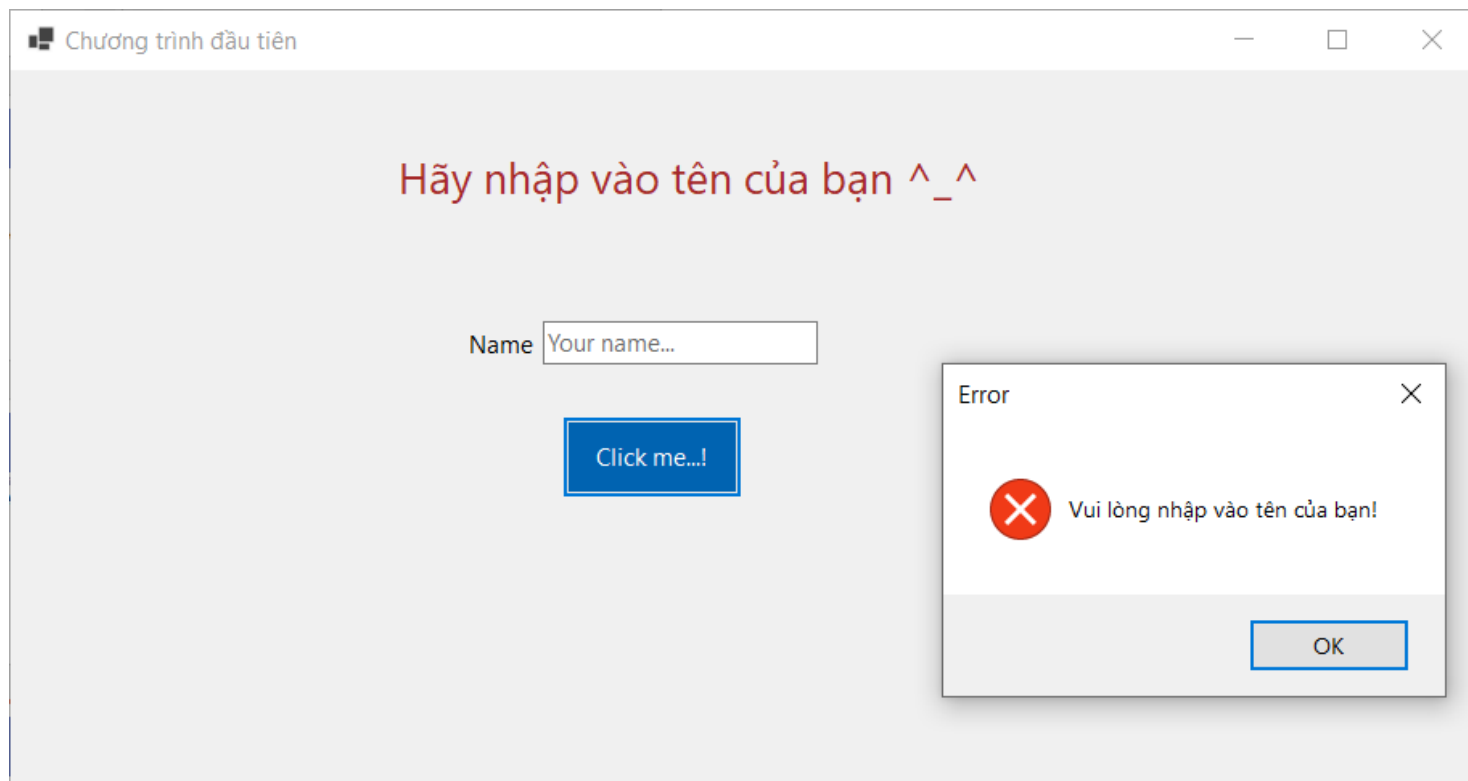




# + BÀI TẬP THỰC HÀNH

17

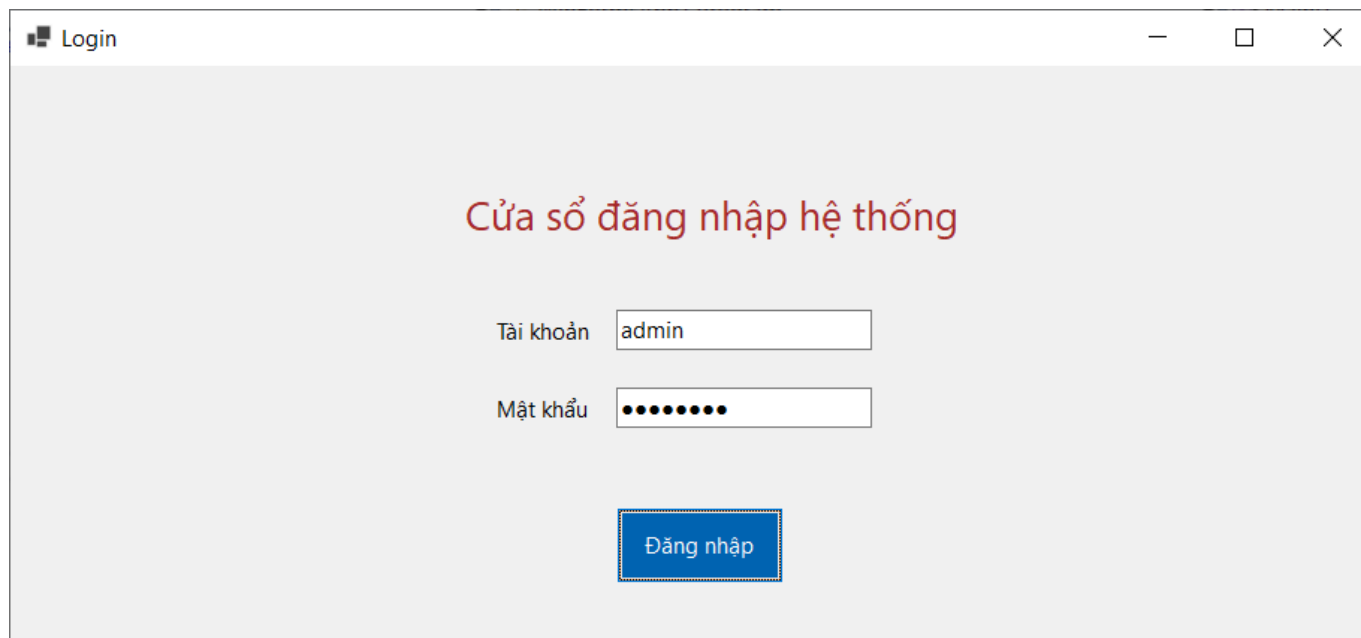
- **Bài 4.1:** Nếu người dùng để trống họ tên, hiển thị hộp thoại thông báo lỗi.



# + BÀI TẬP THỰC HÀNH

18

- **Bài 4.2:** Tạo cửa sổ đăng nhập như hình dưới. Khi người dùng nhấn nút đăng nhập, kiểm tra dữ liệu:
  - Tài khoản và mật khẩu không được để trống.
  - Tài khoản và mật khẩu có độ dài từ 6-30 ký tự.



Login

Cửa sổ đăng nhập hệ thống

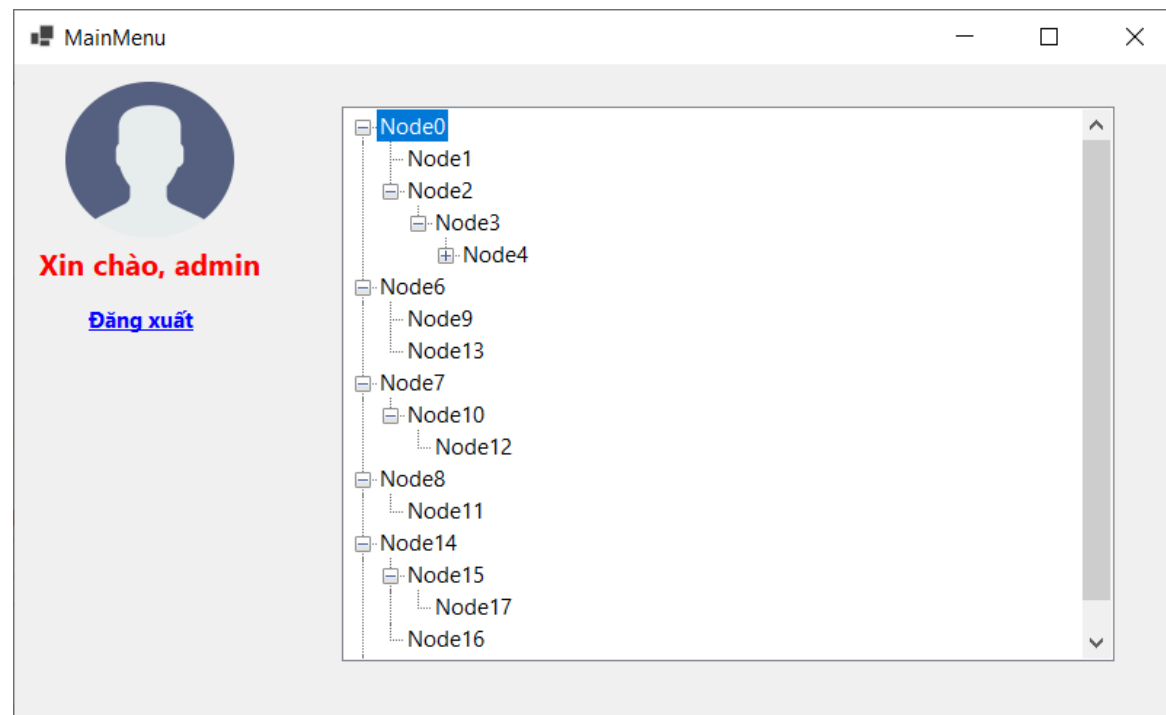
Tài khoản

Mật khẩu

# + BÀI TẬP THỰC HÀNH

19

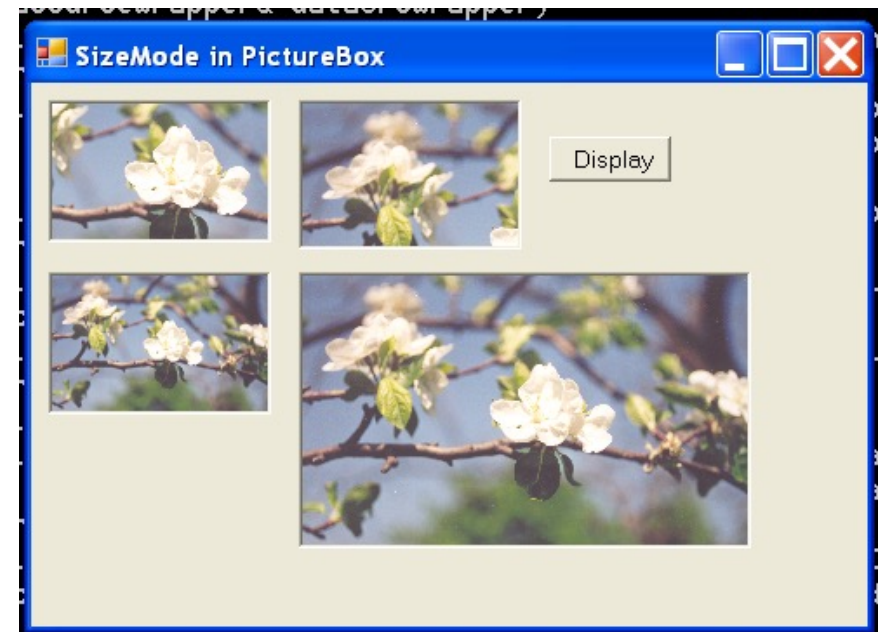
- **Bài 4.2:** Tạo cửa sổ đăng nhập như hình dưới. Khi người dùng nhấn nút đăng nhập, kiểm tra dữ liệu:
  - ❖ Nếu tài khoản = "admin", mật khẩu = "123456789" thì hiển thị màn hình trang chủ ứng dụng.



# + PictureBox

20

- Để hiển thị ảnh từ các file: metafile, icon, bitmap, JPEG, PNG và GIF.
- Một số các thành phần quan trọng của lớp PictureBox:
  - Image: ảnh hiển thị trên PictureBox.
  - SizeMode: cách hiển thị ảnh
  - PictureBoxSize
    - AutoSize, CenterImage, Normal, StretchImage.

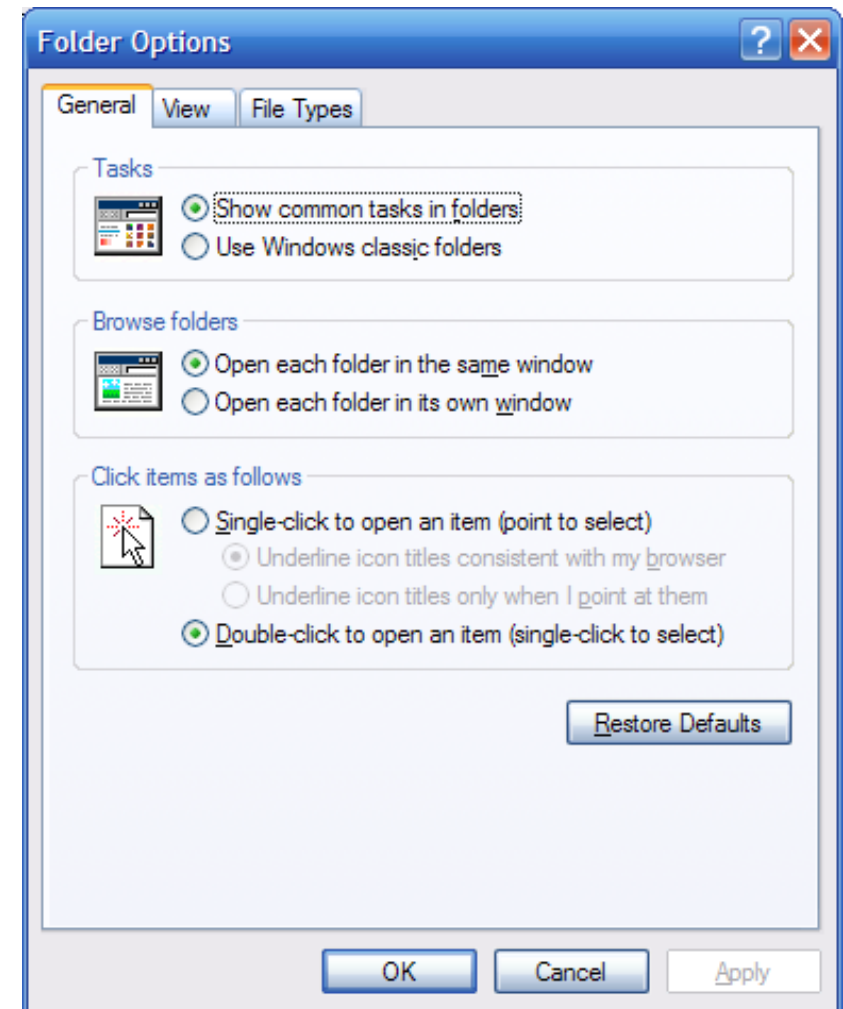


```
pbImage.Image =  
Image.FromFile(<filename>);
```

# + GroupBox và Panel

21

- GroupBox là container control, có khả năng chứa các control khác.
  - Được sử dụng để sắp xếp các control và nhóm các control tương tự nhau.
    - Controls: xác định tập các control chứa trong GroupBox
    - Text: xác định nội dung hiển thị trên GroupBox
- Panel là container control.
  - AutoScroll: xác định thanh trượt có được hiển thị không
  - Controls: xác định tập các control chứa trong Panel





# Button, CheckBox, RadioButton

22

- Các lớp Button, CheckBox, RadioButton đều thừa kế từ lớp `ButtonBase`.
- Sử dụng `Button` để khởi tạo một hành động cụ thể khi người sử dụng click nó.
- `CheckBox` và `RadioButton` thường được sử dụng để lưu trạng thái (chọn hoặc không chọn).
  - Thường được nhóm lại.
  - `CheckBox` cho phép chọn 0/1/n lựa chọn từ một tập hợp
  - `RadioButton` cho phép chọn 1 lựa chọn từ một tập hợp.
    - Nếu muốn chọn nhiều `RadioButton` thì phải đặt chúng trên các container control khác nhau.

# + MỘT SỐ THUỘC TÍNH CỦA Button

23

- Image: xác định ảnh hiển thị trên Button
- Text: xác định văn bản hiển thị trên Button
- Sự kiện Click xảy ra khi người sử dụng click chuột.

The screenshot shows a Java Swing window titled "Checkboxes and Radio Buttons". It contains two panels for movie preferences. The left panel, titled "What Type Of Movies Do You Like?", contains five checkboxes for "Comedy", "Action", "Science Fiction", "Romance", and "Animation". The right panel, titled "And Your Favourite Is?", contains five radio buttons for the same categories. Below each panel is a button: "Selected Movies" under the checkboxes and "Favourite Movie" under the radio buttons.

# + MỘT SỐ THUỘC TÍNH CỦA CheckBox

- Checked: trả về true nếu chọn CheckBox; ngược lại trả về false
- CheckState: xác định trạng thái của Checkbox, giá trị thuộc kiểu enum CheckState (Checked, Unchecked, Indeterminate).
  - Khi chọn Indeterminate thì Checked vẫn trả về true
  - Nên sử dụng CheckState để xác định trạng thái của CheckBox.
- ThreeState
  - Bằng true thì CheckBox có 3 trạng thái.
  - Bằng false thì giá trị Indeterminate chỉ được gán thông qua mã lệnh, không phải do người sử dụng.
- Text: xác định văn bản hiển thị trên CheckBox.
- Sự kiện CheckedChanged xảy ra khi giá trị Checked bị thay đổi.





## MỘT SỐ THUỘC TÍNH CỦA RadioButton

25

- Checked: trả về true hoặc false để xác định RadioButton có được chọn hay không.
- Text: xác định văn bản hiển thị trên CheckBox.
- Sự kiện CheckedChanged xảy ra khi giá trị Checked bị thay đổi.

# + BÀI TẬP THỰC HÀNH

26

- **Bài 4.3:** Giải phương trình bậc 1 và phương trình bậc 2.
  - Khi chọn radio PT bậc 1 thì khóa khung nhập hệ số c
  - Khi chọn radio PT bậc 2 thì hiển thị đầy đủ khung nhập

The image displays two side-by-side screenshots of a software application titled "Giải Phương Trình Bậc 1 - 2". Both windows have a title bar with standard Windows controls (minimize, maximize, close) and a red close button.

**Left Window (Bậc 1 selected):**

- Title:** Giải Phương Trình
- Radio Buttons:** "Bạn vui lòng chọn" with two options: "Giải Phương Trình Bậc 1" (selected) and "Giải Phương Trình Bậc 2".
- Inputs:** "Nhập a:" with value 4, "Nhập b:" with value 2, and "Nhập c:" which is empty.
- Buttons:** A blue "Giải" button and a grey "Thoát" button.
- Output:** "Kết quả:" with the text "Pt có nghiệm: -0,5".

**Right Window (Bậc 2 selected):**

- Title:** Giải Phương Trình
- Radio Buttons:** "Bạn vui lòng chọn" with two options: "Giải Phương Trình Bậc 1" and "Giải Phương Trình Bậc 2" (selected).
- Inputs:** "Nhập a:" with value 1, "Nhập b:" with value 2, and "Nhập c:" with value -3.
- Buttons:** A blue "Giải" button and a grey "Thoát" button.
- Output:** "Kết quả:" with the text "PT có 2 nghiệm: x1=1 và x2=-3".

# + ListBox, CheckedListBox và ComboBox

The screenshot shows a Windows application window titled "Get Checked Items". It contains two main panels, each with a list box and a button.

**Left Panel: Using List(Of Product)**

- Using List(Of Product):** A list box containing the following items with checkboxes:
  - ☐ Chai
  - ☐ Chang
  - ☒ Guaraná Fantástica
  - ☐ Sasquatch Ale
  - ☒ Steeleye Stout
  - ☐ Côte de Blaye
  - ☐ Chartreuse verte
  - ☐ Ipoh Coffee
  - ☒ Laughing Lumberjack Lager
  - ☐ Outback Lager
  - ☐ Rhönbräu Klosterbier
  - ☐ Lakkalikööri
- Selected Items:** A list box showing the selected items: Guaraná Fantástica, Steeleye Stout, and Laughing Lumberjack Lager.
- Buttons:** "Get Checked products" and "Use standard method" (checkbox).

**Right Panel: List(Of String)**

- List(Of String):** A list box containing the months of the year with checkboxes:
  - ☐ January
  - ☐ February
  - ☐ March
  - ☐ April
  - ☒ May
  - ☒ June
  - ☒ July
  - ☐ August
  - ☒ September
  - ☐ October
  - ☐ November
  - ☐ December
- Selected Items:** A list box showing the selected months: May, June, July, and September.
- Buttons:** "Get Checked months".

# + ListBox, CheckedListBox và ComboBox

- ListBox cho phép ta lựa chọn một hoặc nhiều giá trị từ một danh sách các giá trị cho trước.
  - ColumnWidth: xác định độ rộng của một cột trong ListBox có nhiều cột.
  - ItemHeight: xác định chiều cao của một item trong ListBox.
  - Items: tập hợp các đối tượng biểu diễn danh sách các item trong ListBox.
  - MultiColumn: thuộc tính xác định liệu ListBox có được chia thành nhiều cột hay không.
  - SelectedIndex: xác định index của item đang được chọn.
  - SelectedIndices: xác định tập các index của tất cả các item được chọn.
  - SelectedItem: xác định item được chọn
  - SelectedItems: xác định tập hợp các item được chọn
  - SelectionMode: xác định số lượng các item được chọn. Giá trị của nó là kiểu enum SelectionMode.
  - Sorted: xác định ListBox có được sắp xếp theo thứ tự.
  - Sự kiện SelectedIndexChanged xảy ra khi index bị thay đổi.
  - FindString(): phương thức xác định item đầu tiên trong ListBox có nội dung bắt đầu với một string cho trước.
  - FindStringExact(): phương thức xác định item đầu tiên trong ListBox có nội dung giống hệt nội dung của một string cho trước.

# + **CheckedListBox**

29

- CheckListBox thừa kế từ ListBox
  - Hiển thị một tập các item, mỗi item tương ứng với một CheckBox.
- Một số thành phần chính của CheckListBox:
  - CheckedIndices: xác định tập các index của tất cả các item được chọn.
  - CheckedItems xác định tập hợp các item được chọn
  - SelectionMode: xác định số lượng các item được chọn. Giá trị của nó là kiểu enum SelectionMode:
  - Sorted: xác định ListBox có được sắp xếp theo thứ tự.
  - Sự kiện ItemCheck xảy ra khi thay đổi thuộc tính checked của một item

# + ComboBox

30

- ComboBox tương tự như ListBox, ngoại trừ việc có thể nhập vào text và chỉ được chọn một item.
- Một số thành phần của ComboBox:
  - DrawMode: xác định cách vẽ các item của ComboBox. Giá trị của nó là kiểu enum DrawMode.
  - DropDownStyle: biểu diễn kiểu của ComboBox. Giá trị của nó được xác định bởi enum DropDownStyle.
  - DropDownWidth: xác định chiều rộng của phần drop-down trên ComboBox.
  - Items: xác định tập các item trong ComboBox.
  - MaxDropDownItems: xác định số item lớn nhất có thể có trên drop-down; nếu số lượng item lớn hơn thì sẽ xuất hiện thanh cuộn.
  - MaxLength: xác định chiều dài xâu lớn nhất được nhập vào ComboBox.
  - SelectedIndex: xác định index của item hiện thời được chọn.
    - Trả về -1 nếu không có item nào được chọn.
  - SelectedItem: xác định item hiện thời được chọn
  - SelectedText: xác định string trên phần edit.
  - Sorted: xác định các item có được sắp xếp hay không.
  - Sự kiện SelectedIndexChanged xảy ra khi thuộc tính SelectedIndex thay đổi.

# + BÀI TẬP THỰC HÀNH

31

- **Bài 4.4:** Viết chương trình Dịch vụ khám bệnh.
  - Nhập thông tin bệnh nhân, tích chọn dịch vụ
  - Nhấn nút “Chọn” để hiển thị thông tin

Tâm Gà - Dịch vụ khám bệnh

## Dịch Vụ Khám Bệnh

Họ tên bệnh nhân : Nguyễn Thanh Tâm

Ngày 23 Tháng 09 Năm 1985

Chọn dịch vụ : Danh sách đã chọn :

Siêu Âm	X-Quang
<b>Thử Máu</b>	Nội Soi
Nội Soi	<b>Thử Máu</b>
X-Quang	

Chọn Reset Thoát

Họ tên: Nguyễn Thanh Tâm  
Ngày sinh: 23/09/1985  
Dịch vụ đã chọn: X-Quang, Nội Soi, Thử Máu,

# + DomainUpDown và NumericUpDown

32

Progress Currency Converter

Amount:  From:  To:

10 USD = 8.8720 EUR      1 USD = 0.8872EUR, 1 EUR = 1.1271 USD

Exchange Rates						1 USD
EUR	USD	GBP	CHF	RUB	CAD	
0.8872	1	0.7614	1.0074	65.6900	1.3372	



## + DomainUpDown và NumericUpDown

- Thừa kế từ lớp System.Windows.FormsUpDownBase.
- Sử dụng bằng cách:
  - Lựa chọn các giá trị từ một tập hợp các giá trị đã được sắp xếp bằng cách nhấn các nút up/down.
  - Nhập giá trị vào control nếu thuộc tính ReadOnly của nó được gán bằng false.
- DomainUpDown cho phép lựa chọn từ một tập các đối tượng.
  - Khi một item được chọn, đối tượng sẽ được convert thành String và hiển thị.
- NumericUpDown chứa tập hợp giá trị số.

# + DomainUpDown

34

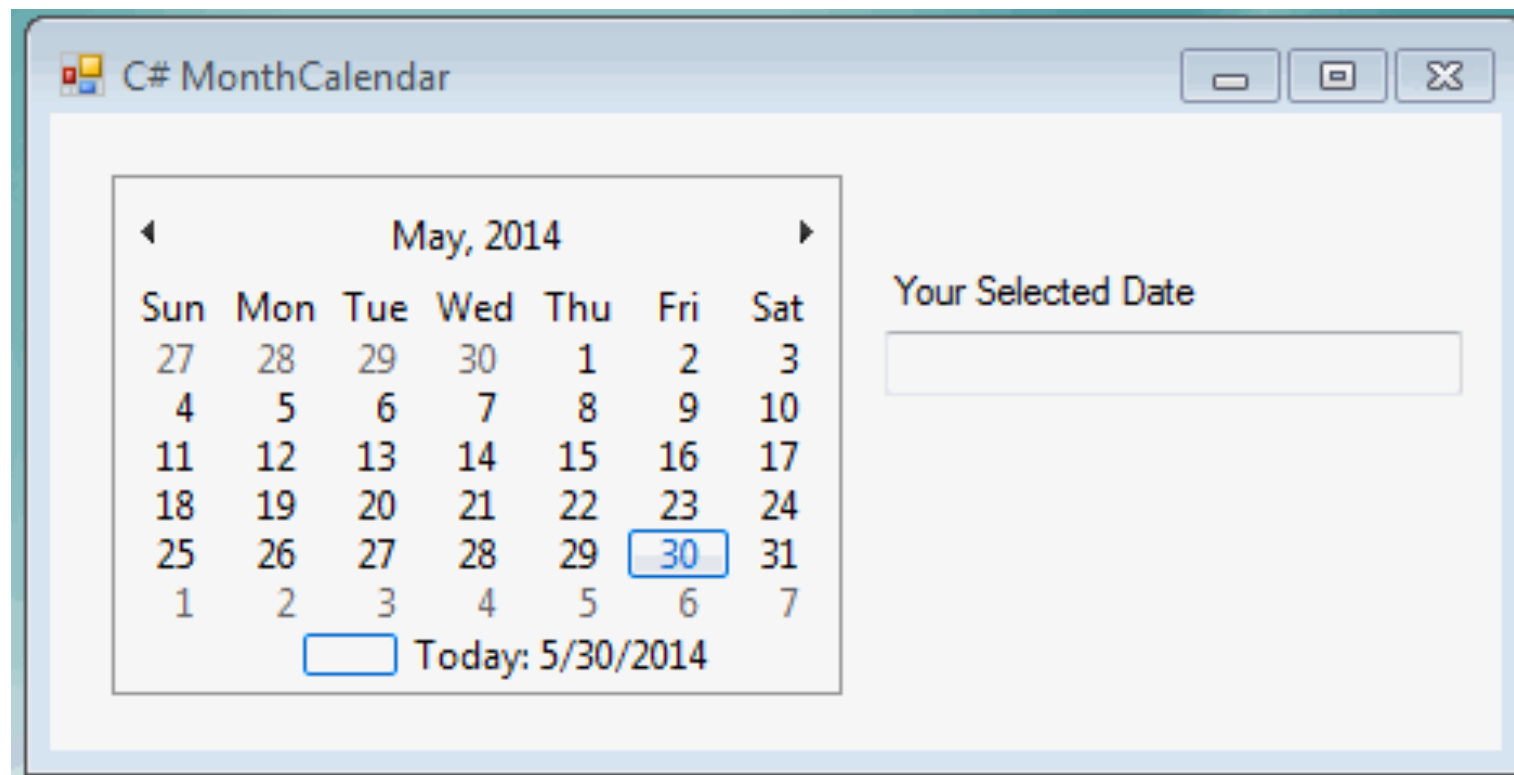
- **Items**: biểu diễn tập đối tượng được gán cho control.
- **ReadOnly**: xác định có thể nhập giá trị hay không
- **SelectedIndex**: xác định giá trị index của item được chọn
- **SelectedItem**: xác định giá trị của các item được chọn
- **Sorted**: xác định tập hợp các item có được sắp xếp hay không.
- **Wrap**: xác định thuộc tính **SelectedIndex** có được điều chỉnh lại tới item đầu tiên hoặc cuối cùng nếu người sử dụng tiếp tục di chuyển đến cuối danh sách.
- Sự kiện **SelectedItemChanged**: xảy ra khi thuộc tính **SelectedIndex** bị thay đổi.

# + NumericUpDown

35

- **Increment**: xác định bước tăng hoặc giảm.
- **Maximum**: xác định giá trị lớn nhất được phép
- **Minimum**: xác định giá trị nhỏ nhất được phép
- **ReadOnly**: xác định có thể nhập lại giá trị
- **ThousandsSeparator**: xác định dấu phân biệt hàng nghìn
- **Value**: xác định giá trị được gán cho control
- Sự kiện **ValueChanged** xảy ra khi thuộc tính **Value** bị thay đổi.

# + MonthCalendar



# + MonthCalendar

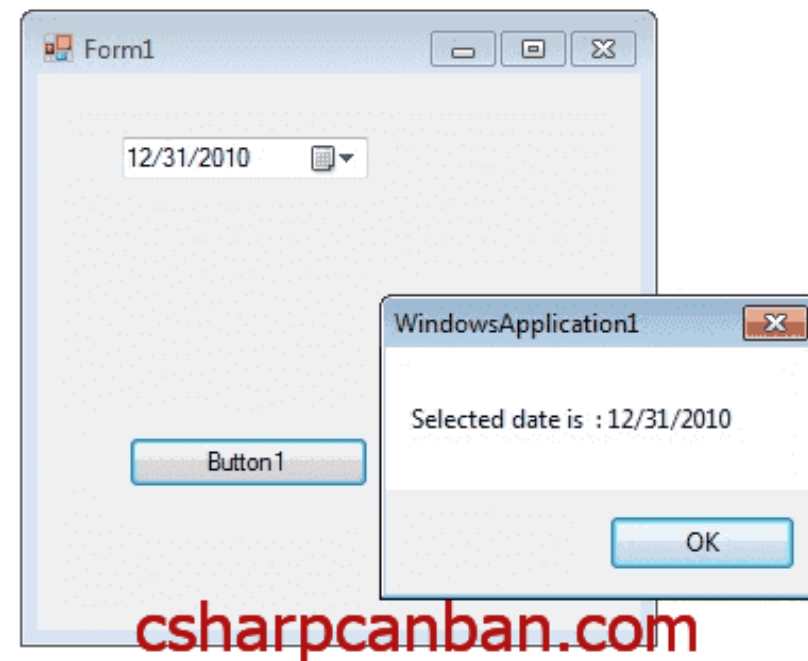
37

- MonthCalendar cung cấp giao diện rất thân thiện để người sử dụng lựa chọn thời gian.
  - **CalendarDimensions**: xác định số cột hoặc dòng của tháng để hiển thị.
  - **FirstDayOfWeek**: biểu diễn ngày đầu tiên trong tuần.
  - **MaxDate**: ngày lớn nhất có thể hiển thị.
  - **MaxSelectionCount**: xác định số lượng ngày lớn nhất được chọn.
  - **MinDate**: ngày nhỏ nhất có thể hiển thị.
  - **SelectionEnd**: xác định ngày cuối cùng trong những ngày đã chọn.
  - **SelectionRange**: xác định các ngày đã chọn.
  - **SelectionStart**: xác định ngày đầu tiên trong những ngày đã chọn.
  - **ShowToday**: hiển thị hay không ngày hiện thời ở phía dưới control
  - **ShowWeekNumbers**: hiển thị hay không thứ tự của tuần (1-52)
  - **TodayDate**: cho biết ngày hiện tại
  - Sự kiện **DateChanged** xảy ra khi chọn lại ngày trên control.
  - Sự kiện **DateSelected** xảy ra khi chọn date trên form.

# + DateTimePicker

38

- Cho phép người sử dụng lựa chọn ngày tháng và thời gian theo những định dạng khác nhau.
  - **CustomFormat**: biểu diễn xâu định dạng date/time. Chỉ có tác dụng khi Format = Custom
  - **Format**: xác định định dạng date/time dựa trên các giá trị của kiểu enum **DateTimePickerFormat**.
  - **MaxDate** là date/time lớn nhất được lựa chọn
  - **MinDate** là date/time nhỏ nhất được lựa chọn
  - **ShowCheckBox**: CheckBox có được hiển thị ở phía bên trái ngày được chọn.
  - **ShowUpDown**: UpDown có được hiển thị
  - **Value**: biểu diễn giá trị date/time được chọn
  - Sự kiện **ValueChanged** xảy ra khi thuộc tính Value bị thay đổi.
  - Sự kiện **FormatChanged** xảy ra khi thuộc tính Format thay đổi



# + BÀI TẬP THỰC HÀNH

39

- **Bài 4.5:** Viết chương trình Công ty du lịch.
- Nhập thông tin khách hàng, tích chọn dịch vụ, chọn ngày tháng
- Nhấn nút “Nhập” để hiển thị thông tin, tính tổng số ngày đi

**Công Ty Du Lịch Tâm Gà**

Khách hàng : Tâm Gà  
Ngày sinh : 23/09/1985  
Địa chỉ : Việt Nam

Địa điểm đi : TPHCM  
Địa điểm đến : Hàn Quốc

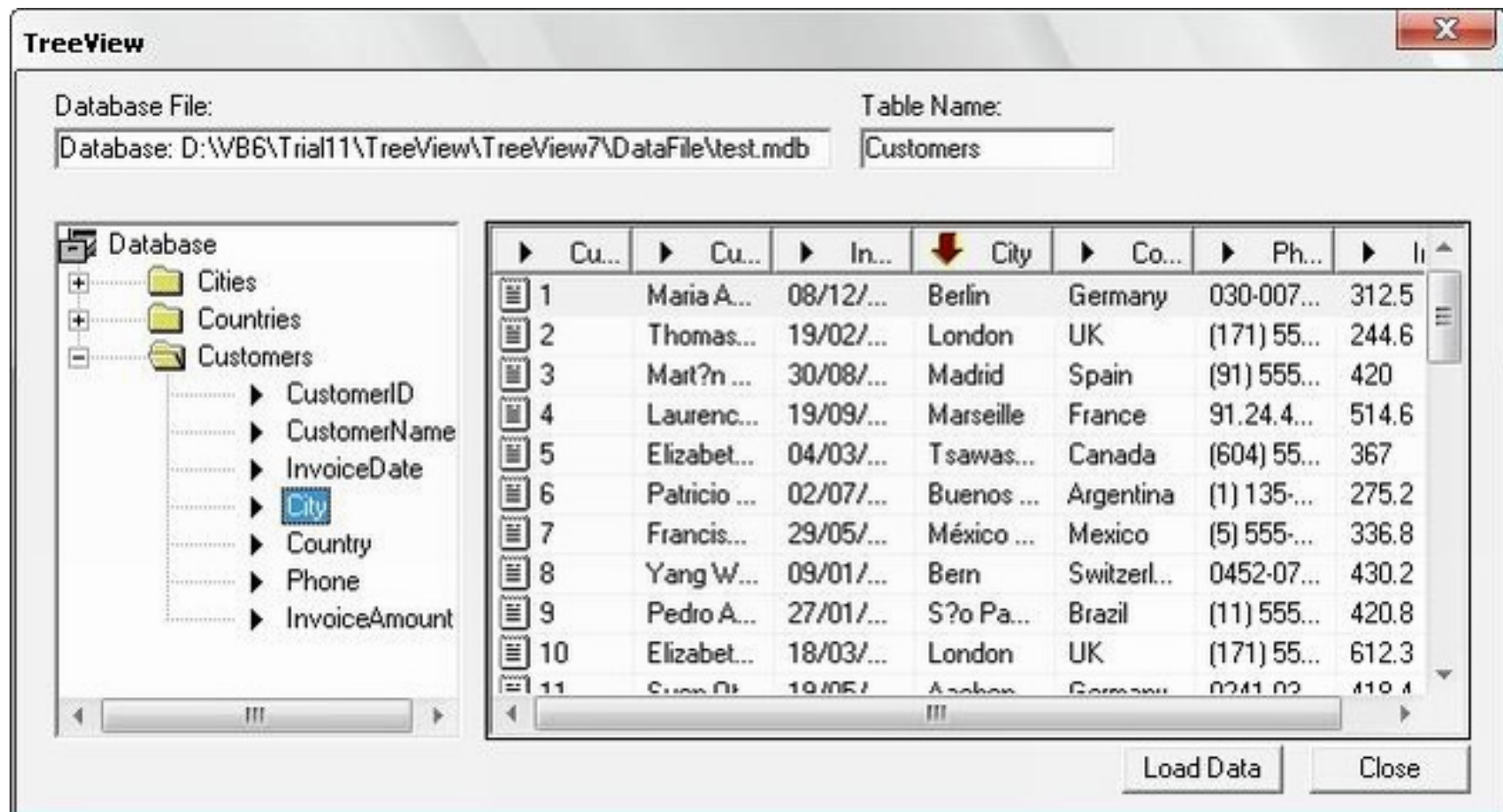
Ngày đi : 01/11/2012  
Ngày về : 01/01/2013

Tổng số ngày : 61

Nhập Reset Thoát

# + TreeView và ListView

40





# + TreeView

41

- TreeView được sử dụng để hiển thị tập hợp các nút phân cấp.
- Mỗi nút là một đối tượng TreeNode.
  - Sự kiện AfterCheck xảy ra khi TreeNode bị check
  - Sự kiện AfterCollapse xảy ra khi TreeNode bị co lại
  - Sự kiện AfterExpand xảy ra khi TreeNode được mở ra
  - Sự kiện AfterSelect xảy ra khi TreeNode bị select
  - **CheckBoxes**: xác định check box có được hiển thị bên cạnh TreeNode
  - **ImageList**: chứa các icon của TreeNode
  - **Nodes**: xác định tập hợp các TreeNode
  - **Scrollable**: thanh trượt có được hiển thị hay không
  - **SelectedNode**: biểu diễn TreeNode được select
  - **Sorted**: TreeNode có được sắp xếp hay không

# + ListView

42

- ListView được sử dụng để hiển thị danh sách các item.
- Mỗi item trong danh sách có tên và có icon tương ứng với nó.
  - **Activation**: xác định cách active một item, kiểu giá trị là kiểu enum ItemActivation.
  - **CheckBoxes**: xác định Check Box có được hiển thị bên cạnh item
  - **CheckedIndices**: xác định tập hợp các index của các item được check.
  - **CheckedItems** xác định tập hợp các item được check.
  - **Items**: xác định tập các item của ListView
  - **LargeImageList** biểu diễn ImageList được sử dụng để hiển thị icon lớn.
  - **MultiSelect**: cho phép chọn nhiều item.
  - **Scrollable**: thanh trượt xuất hiện khi kích thước của item vượt quá kích thước của client area.
  - **SelectedIndices**: xác định tập các index của các item được select.
  - **SelectedItems**: xác định tập các item được select.
  - **SmallImageList**: biểu diễn ImageList được sử dụng để hiển thị icon nhỏ
  - **Sorting**: sắp xếp các item theo thứ tự của enum SortOrder.
  - **View**: cách hiển thị các item, sử dụng các giá trị của kiểu enum View
  - Sự kiện **ItemActivate** xảy ra khi item được activate.
  - Sự kiện **ItemCheck** xảy ra khi trạng thái check của item bị thay đổi.
  - Sự kiện **SelectedIndexChanged** xảy ra khi thay đổi index.

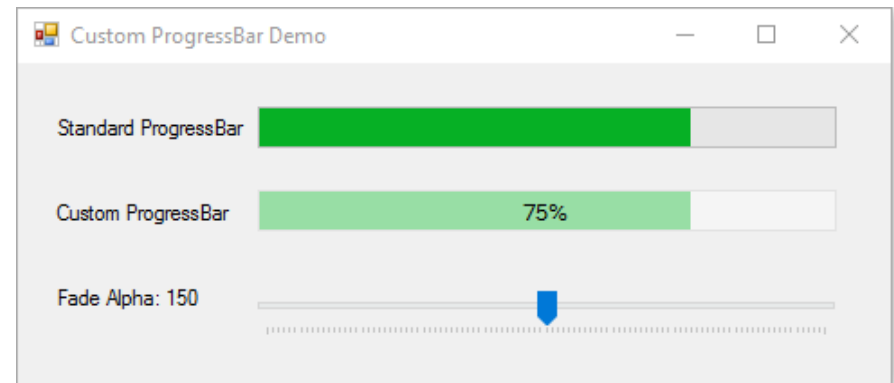
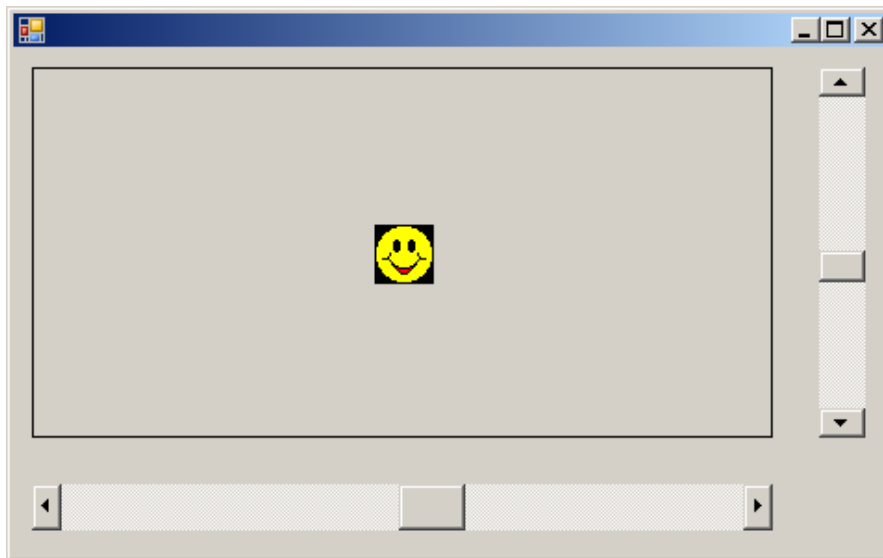
# + Timer

43

- Timer được sử dụng khi một sự kiện được tạo ra trong một khoảng thời gian cho trước.
- Lớp System.Timers.Timer kích hoạt sự kiện Tick từ một thread khác.
  - **Enabled**: xác định Timer có hoạt động không.
  - **Interval**: khoảng thời gian kích hoạt sự kiện (tính bằng ms).
  - Phương thức Start() khởi tạo Timer
  - Phương thức Stop() khoá Timer
  - Sự kiện Tick xảy ra khi đến khoảng thời gian Interval.

# + TrackBar, ProgressBar, HScrollBar, VScrollBar

44



# + TrackBar

45

- TrackBar cung cấp một cách trực quan để lựa chọn giá trị từ một chuỗi cho trước thông qua hộp cuộn và giá trị tỷ lệ.
  - **LargeChange**: xác định giá trị lớn nhất khi di chuyển hộp cuộn.
  - **Maximum**: xác định phạm vi lớn nhất của TrackBar
  - **Minimum**: xác định phạm vi nhỏ nhất của TrackBar
  - **Orientation**: biểu diễn hướng ngang và dọc của TrackBar.
  - **SmallChange**: xác định giá trị nhỏ nhất khi di chuyển hộp cuộn.
  - **TickFrequency**: biểu diễn tần suất vẽ đường di chuyển.
  - **TickStyle** xác định cách hiển thị của đường dịch chuyển, giá trị kiểu enum TickStyle.
  - **Value**: biểu diễn vị trí hiện thời của hộp cuộn.
  - **ValueChanged**: xảy ra khi thuộc tính Value thay đổi thông qua sự kiện Scroll hoặc lập trình.
  - Sự kiện **Scroll** xảy ra khi hộp cuộn bị di chuyển bởi chuột hoặc bàn phím.

# + ProgressBar

46

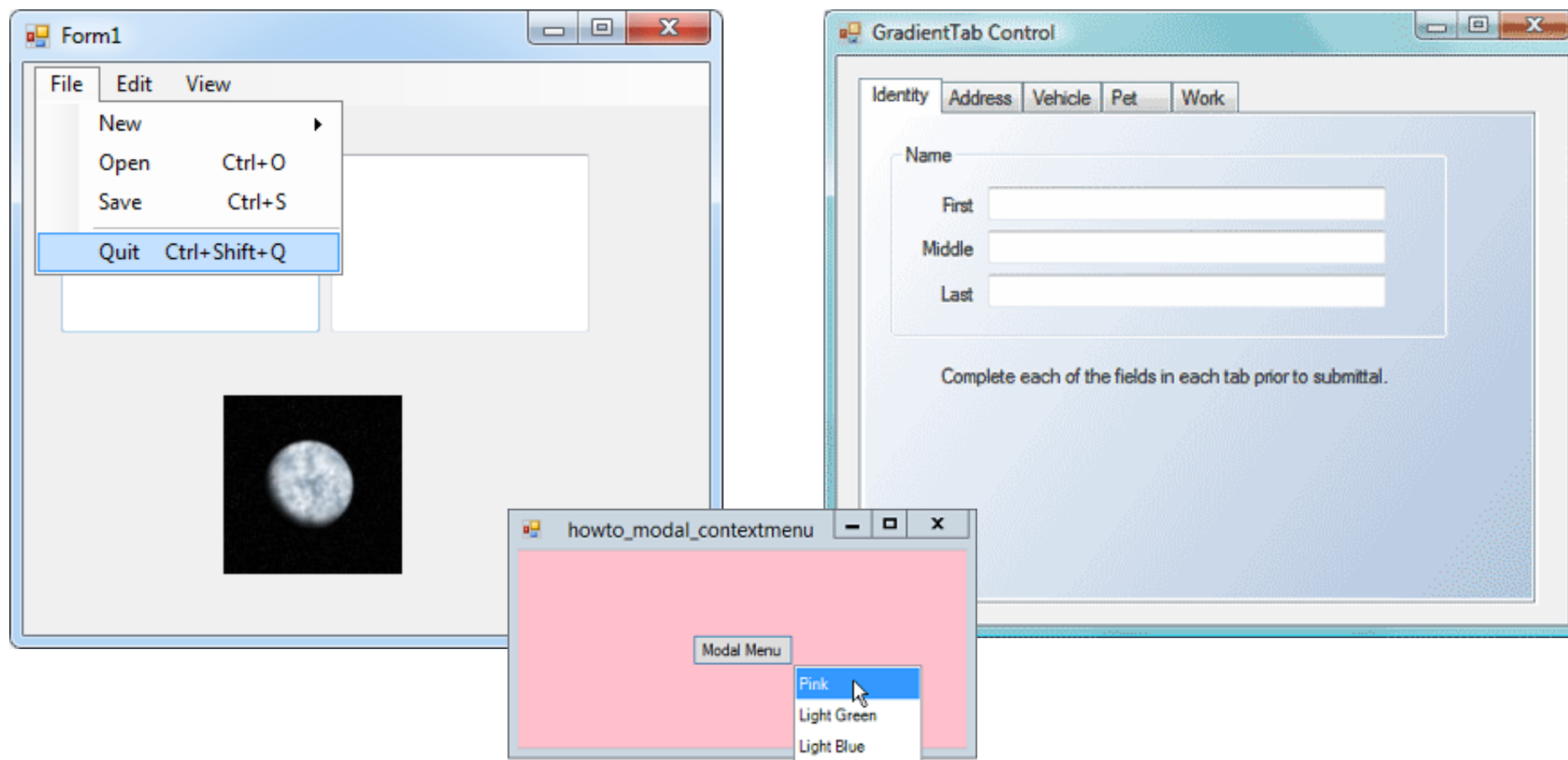
- Được sử dụng để chỉ ra trạng thái về thời gian của một thao tác như: cài đặt, copy file ..
  - Maximum: xác định biên trên của tiến trình
  - Minimum: xác định biên dưới của tiến trình
  - Value: biểu diễn giá trị hiện tại của ProgressBar

## + HscrollBar và VscrollBar

47

- HScrollBar và VScrollBar hiển thị thanh trượt ngang và dọc.
- Lớp HScrollBar và VScrollBar thừa kế từ lớp ScrollBar.
  - **LargeChange**: chỉ ra số lượng để khi Value thay đổi thì xuất hiện thanh trượt.
  - **Maximum**: xác định phạm vi trên của thanh trượt.
  - **Minimum**: xác định phạm vi trên của thanh trượt.
  - **SmallChange**: chỉ ra số lượng để khi Value thay đổi thì bỏ thanh trượt.
  - **Value**: biểu diễn vị trí hiện thời của thanh trượt.
- Sự kiện **ValueChanged** xảy ra khi thuộc tính Value thay đổi thông qua lập trình hoặc thông qua sự kiện Scroll.
- Sự kiện Scroll xảy ra khi trượt thanh.

# + TabControl và Menu





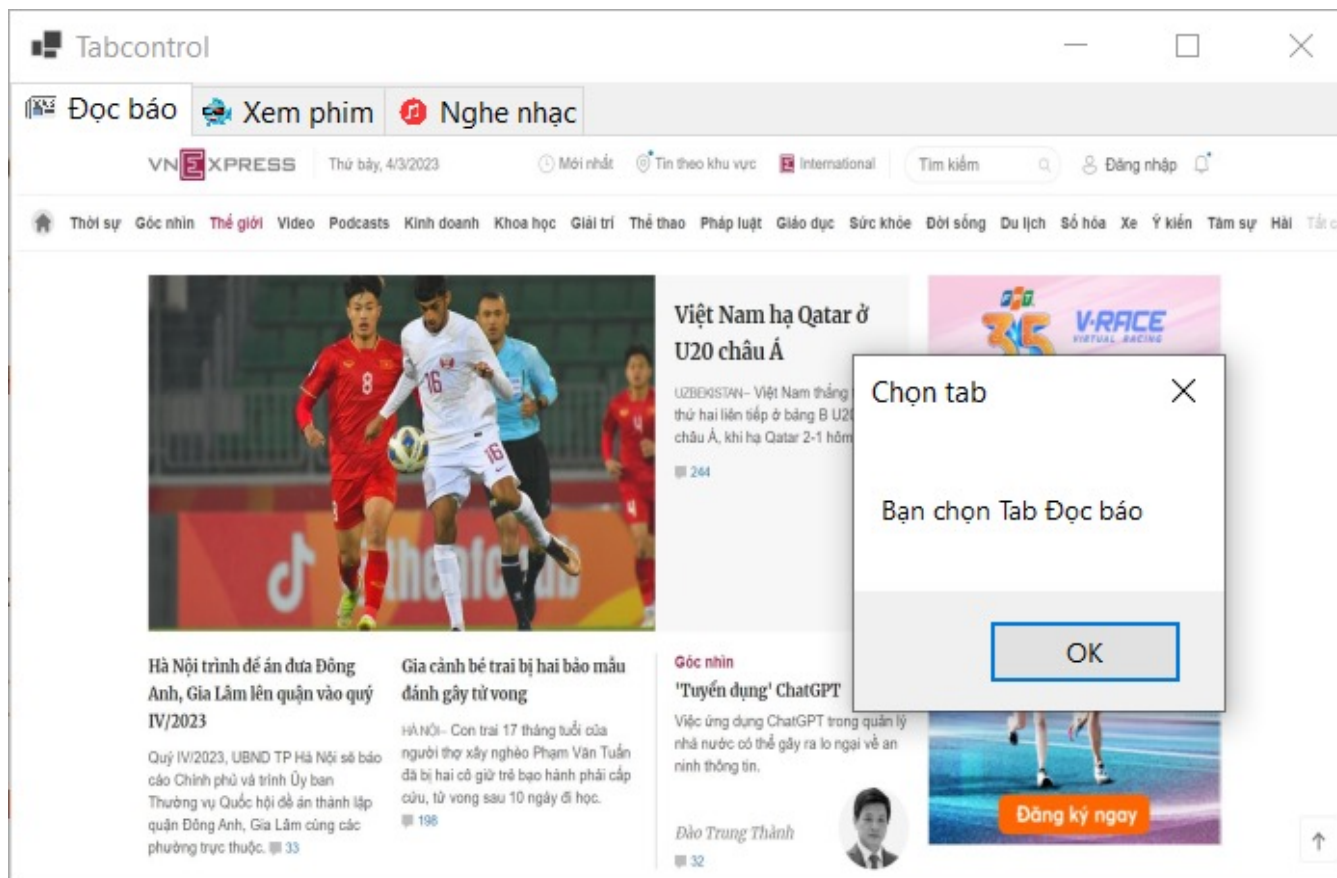
# + TabControl

- TabControl hiển thị một tập hợp các trang.
- Mỗi trang có chứa các control của chính nó.
- TabControl được sử dụng để sắp xếp một số lượng lớn các control vì chúng sẽ chiếm ít không gian hơn.
  - **Alignment**: là vùng mà các tab được căn lề.
  - **ImageList**: là ImageList control được sử dụng để hiển thị ảnh trên tab.
  - **MultiLine**: xác định liệu tab có được hiển thị trên nhiều dòng.
  - **SelectedIndex**: biểu diễn index của trang tab được chọn.
  - Sự kiện SelectedIndexChanged xảy ra khi thuộc tính SelectedIndex bị thay đổi giá trị.
  - **SelectedTab**: biểu diễn trang tab được chọn.
  - **TabCount**: cho biết số lượng trang tab
  - **TabPage**: xác định tập hợp các trang tab.

# + BÀI TẬP THỰC HÀNH

50

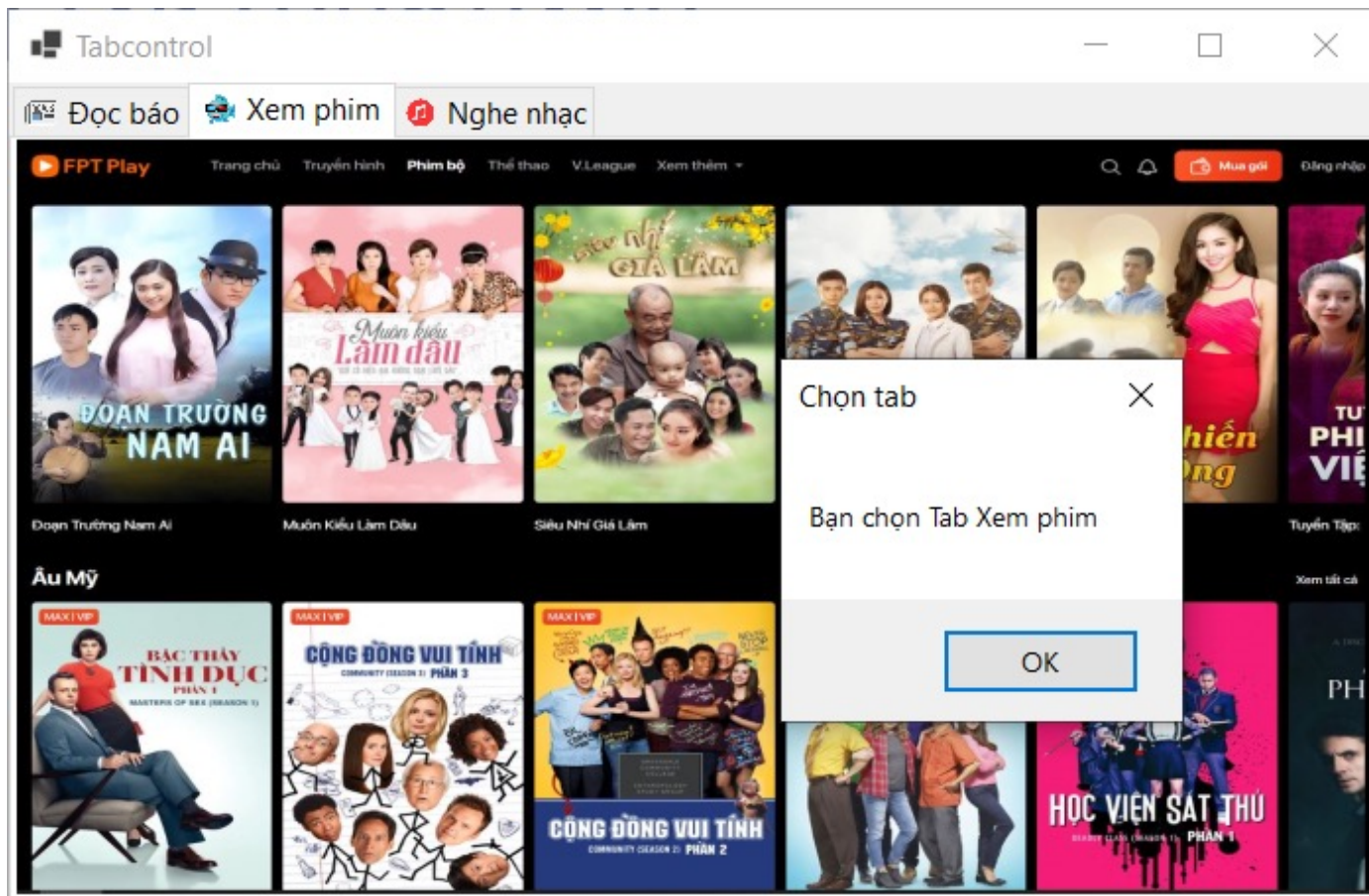
- **Bài 4.6:** Viết chương trình sử dụng Tabcontrol.
- Tạo 3 tab “Đọc báo”, “Xem phim” và “Nghe nhạc”. Thêm icon cho tab.
- Khi nhấn vào tab nào thì hiển thị nội dung tab đó và cửa sổ thông điệp.



# + BÀI TẬP THỰC HÀNH

51

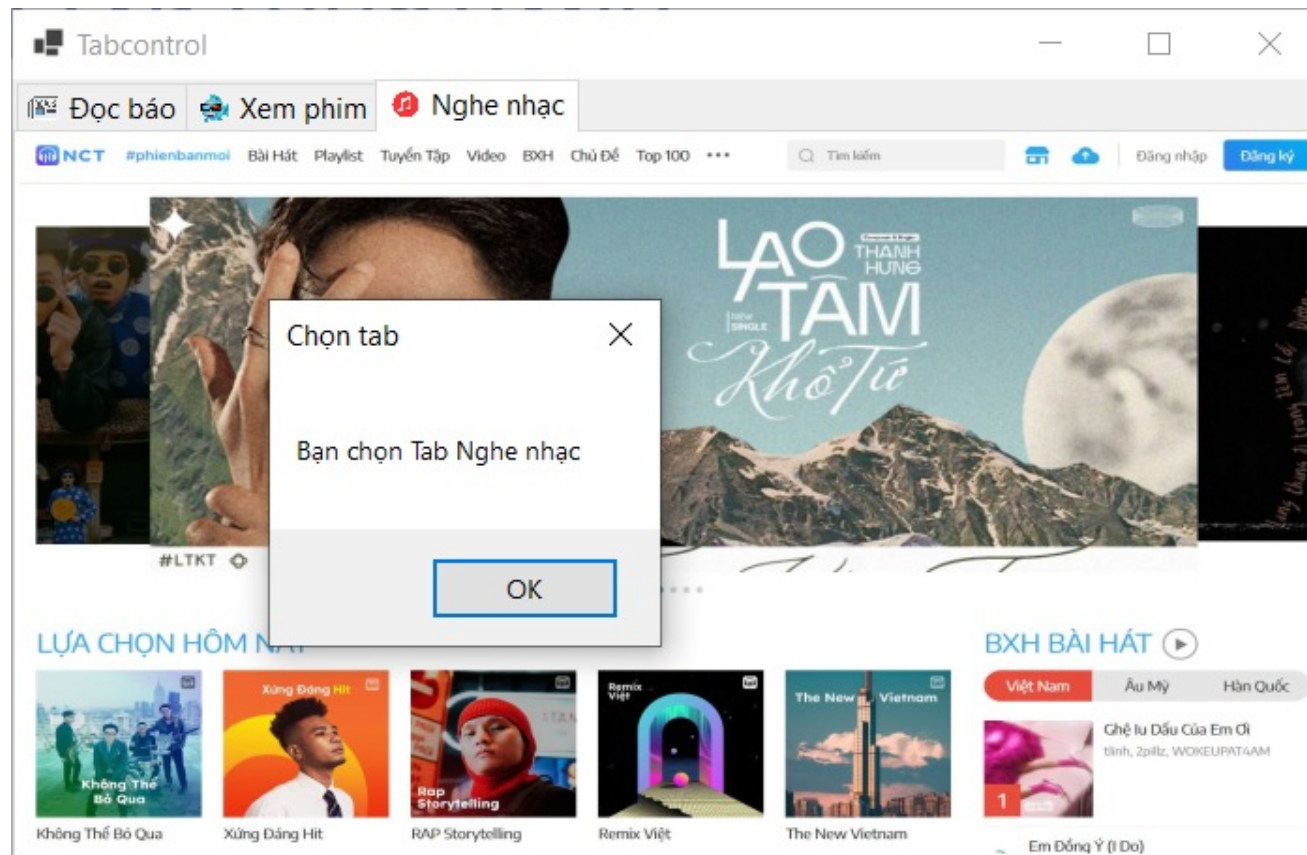
- **Bài 4.6:** Viết chương trình sử dụng Tabcontrol.
- Tạo 3 tab “Đọc báo”, “Xem phim” và “Nghe nhạc”. Thêm icon cho tab.
- Khi nhấn vào tab nào thì hiển thị nội dung tab đó và cửa sổ thông điệp.



# + BÀI TẬP THỰC HÀNH

52

- **Bài 4.6:** Viết chương trình sử dụng Tabcontrol.
- Tạo 3 tab “Đọc báo”, “Xem phim” và “Nghe nhạc”. Thêm icon cho tab.
- Khi nhấn vào tab nào thì hiển thị nội dung tab đó và cửa sổ thông điệp.



## + Tạo ra Menu

- Các lớp có liên quan tới menu: MainMenu, MenuItem và ContextMenu đều thừa kế từ lớp Menu.
- MainMenu và ContextMenu đều có thuộc tính MenuItem.
  - MenuItem là một tập hợp các đối tượng MenuItem
  - Mỗi đối tượng biểu diễn một item của menu.

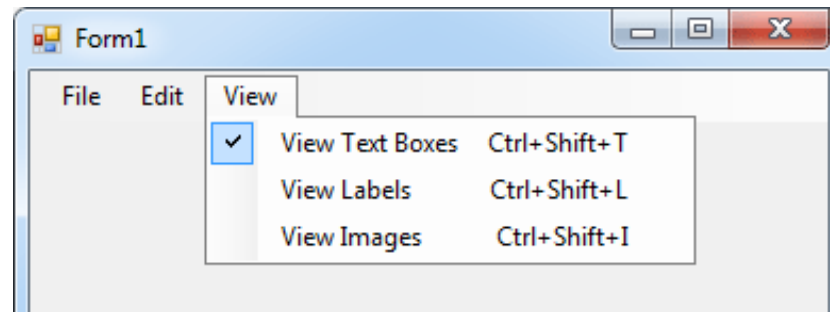
# + MỘT SỐ THÀNH PHẦN CHÍNH CỦA MenuItem

- **Checked:** xác định mỗi item có checkmark hoặc radiobutton ở bên cạnh hay không.
- **Enabled:** xác định menu item có được phép sử dụng hay không.
- **MenuItems:** xác định tập các đối tượng MenuItem có quan hệ với menu và được sử dụng để tạo ra cấp trúc menu phân cấp.
- **Parent:** cho biết menu cha của một menu item.
- **RadioCheck:** xác định menu item có radio button ở bên cạnh không.
- **Shortcut:** xác định phím tắt của menu item.
- **Text:** xác định text hiển thị trên menu item.
- Sự kiện Click xảy ra khi người sử dụng select menu item.
- Sự kiện Popup xảy ra trước khi submenu xuất hiện. Sự kiện này thường được sử dụng để add, remove, enable, disable, check, uncheck menu item trước khi hiển thị nó.
- Sự kiện Select xảy ra khi người sử dụng select menu item.
- Phương thức PerformClick tạo ra sự kiện Click cho menu item tương tự như người sử dụng click nó.

# + MainMenu

55

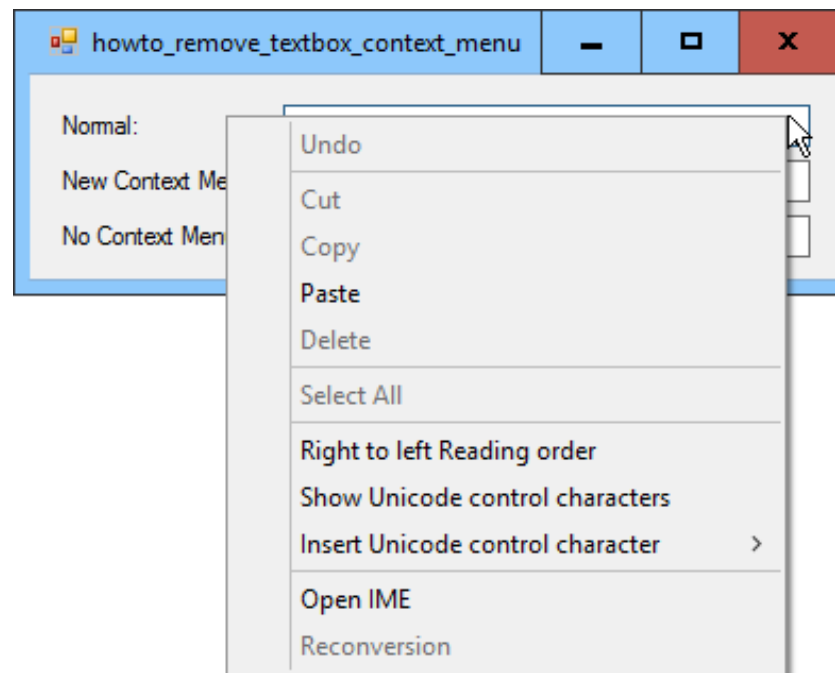
- MainMenu là container control.
- Mỗi Window Form chỉ có một đối tượng MainMenu, được xác định thông qua thuộc tính Menu của đối tượng Form.
- Thành phần quan trọng nhất của MainMenu là MenuItem





# + ContextMenu

- ContextMenu chỉ có một menu item ở mức đỉnh.
- Một ContextMenu thường được gắn với một Control hoặc một Form thông qua thuộc tính ContextMenu.





# + StatusBar

57

- StatusBar được sử dụng để hiển thị thông tin và thường được đặt ở phía dưới của Form.
- Khi bổ sung StatusBar lên Form ta phải gán z-order của nó để StatusBar không chồng lên các control khác, bằng cách click chuột phải trên StatusBar và chọn Send to Back.
  - **Panels** là một tập hợp các đối tượng StatusBarPanel. Panels chia StatusBar thành nhiều vùng
  - **Alignment**: xác định cách căn lề của text và icon trên panel.
  - **AutoSize**: xác định cách panel tự điều chỉnh kích thước. Gồm 3 giá trị: None, Contents và Spring.
  - **BorderStyle**: xác định kiểu biên. Gồm 3 giá trị: None, Raised, và Sunken.
  - **Icon**: icon hiển thị trên StatusBar.
  - **ToolTipText**: xác định ToolTip hiển thị trên StatusBar.

# + ToolBar

58

- ToolBar được sử dụng để tạo ra thanh công cụ trong Window và thường được gán ở phía dưới MainMenu. (chú ý gán lại z-order)
  - **Buttons**: xác định một tập các đối tượng ToolBarButton.
  - **ImageList**: xác định đối tượng ImageList lưu trữ trên icon.
  - **ShowToolTips**: xác định liệu toolbar có hiển thị ToolTips hay không.
- Có thể viện dẫn menu item bằng cách gọi phương thức PerformClick() trên đối tượng MenuItem.
  - Cần phải biết menu nào viện dẫn ToolBarButton.
  - ToolBarButton cung cấp thuộc tính Tag - để lưu bất kỳ đối tượng nào cần được gán với ToolBarButton.
- Sự kiện ButtonClick xảy ra khi toolbar button bị click.

# + ỨNG DỤNG MDI

59

- Ứng dụng MDI có một Form cha và các Form con.
  - Không thể di chuyển Form con ra ngoài Form cha.
  - Có thể mở nhiều Form con cùng một lúc.
  - Các Form con đóng, mở, max, min không phụ thuộc vào nhau. Nhưng nếu Form cha đóng thì tất cả các Form con đều đóng.
  - Form cha thường có menu chính.
- Để tạo ra MDI form, vẫn tạo ra Form như thông thường và đổi thuộc tính `IsMdiContainer` thành `true`.
- Để tạo ra Form con, vẫn tạo ra Form như thông thường và gán `MdiParent` là form MDI đã tạo ra.

## + Một số thành phần của lớp Form có liên quan đến MDI Form:

- `ActiveMdiChild`: xác định MDI child đang được active
- `IsMdiContainer`: xác định Form có phải là MDI Container hay không.
- `MdiChildren`: xác định mảng các MDI child Form của một Form
- `MdiParent`: xác định MDI parent form của Form hiện thời
- Phương thức `LayoutMdi()` sắp xếp các MDI child Form trong MDI parent Form
- Sự kiện `MdiChildActivate` xảy ra khi MDI child Form được active hoặc đóng ứng dụng MDI.

# + BÀI TẬP THỰC HÀNH

- Xây dựng ứng dụng hiển thị ngày giờ.
- Menu chức năng: có mục Thoát dạng MenuItem. Khi người dùng nhấn chuột trái vào menu Thoát hoặc nhấn tổ hợp phím Ctrl + T sẽ thoát chương trình.
- Menu hiển thị: có dạng ComboBox chứa hai mục chọn, hiển thị thời gian hiện hành và hiển thị ngày tháng năm hiện hành.

