

# CHỦ ĐỀ 7

# WINDOWS FORM

GVGD: PHẠM THỊ KIM NGOAN

Email: ptkngoan@gmail.com



# Nội dung



Giới thiệu Windows Form (WF)



Các điều khiển thông thường



Các điều khiển đặc biệt



Điều khiển Menu và Container



Sự kiện bàn phím, chuột và lớp MessageBox

# Nội dung



## Giới thiệu Windows Form (WF)



### Các điều khiển thông thường



### Các điều khiển đặc biệt



### Điều khiển Menu và Container



### Sự kiện bàn phím, chuột và lớp MessageBox

# Graphical User Interface (GUI)

## Command line interface: CLI

```
--- rr.cktpa.wikimedia.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 112.076/112.076/112.076/0.000 ms
bash-2.05b$ grep -i /dev/sda /etc/fstab | cut --fields=3
/dev/sda1          /mnt/usbkey
/dev/sda2          /mnt/ipod
bash-2.05b$ date
Wed May 25 11:36:56 PDT 2005
bash-2.05b$ lsmod
Module           Size Used by
joydev            8256  0
ipu2200          175112  0
ieee80211         44228  1 ipu2200
ieee80211_crypt   4872  2 ipu2200,ieee80211
e1000            84468  0
bash-2.05b$
```

Tương tác qua keyboard  
Thực thi tuần tự

## Text user interface: TUI



GUI dựa trên text  
Mức độ tương tác cao hơn



# Graphical User Interface (GUI)

- Tương tác qua giao diện đồ họa.
- Đa số các hệ OS hiện đại đều dùng GUI
- Cho phép user dễ dàng thao tác

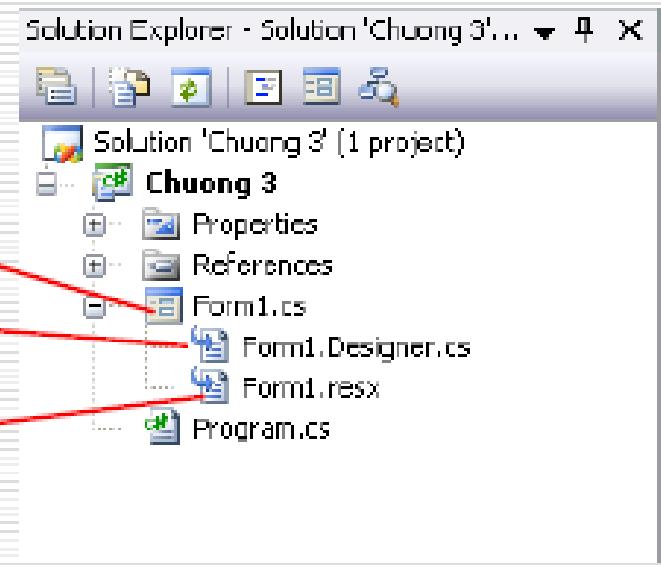


# Ứng dụng GUI

- Windows Form là nền tảng GUI cho ứng dụng desktop
- Các namespace chứa các lớp hỗ trợ GUI trong .NET
  - System.Windows.Forms:
    - Chứa GUI components/controls và form
  - System.Drawing:
    - Chức năng liên quan đến tô vẽ cho thành phần GUI
    - Cung cấp chức năng truy cập đến GDI+ cơ bản

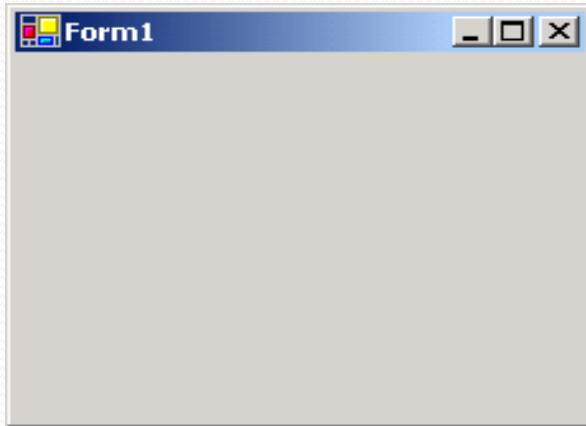
# WF trong C#

- ❖ **Ứng dụng WinForms gồm:**
  - Mã chương trình
  - Giao diện
  - Phương thức, biến cỗ của các Control

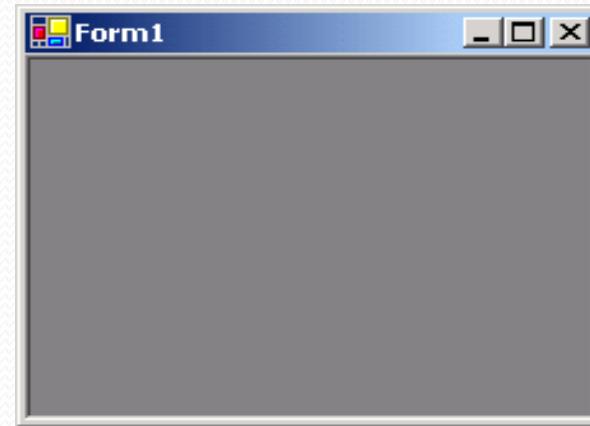


# Các loại Form

- ❖ Form dùng để tạo giao diện cho chương trình. Các loại Form:
  - Multiple Document Interface (MDI)
  - Single Document Interface (SDI)



**Single Document Interface (SDI)**



**Multiple Document Interface (MDI)**

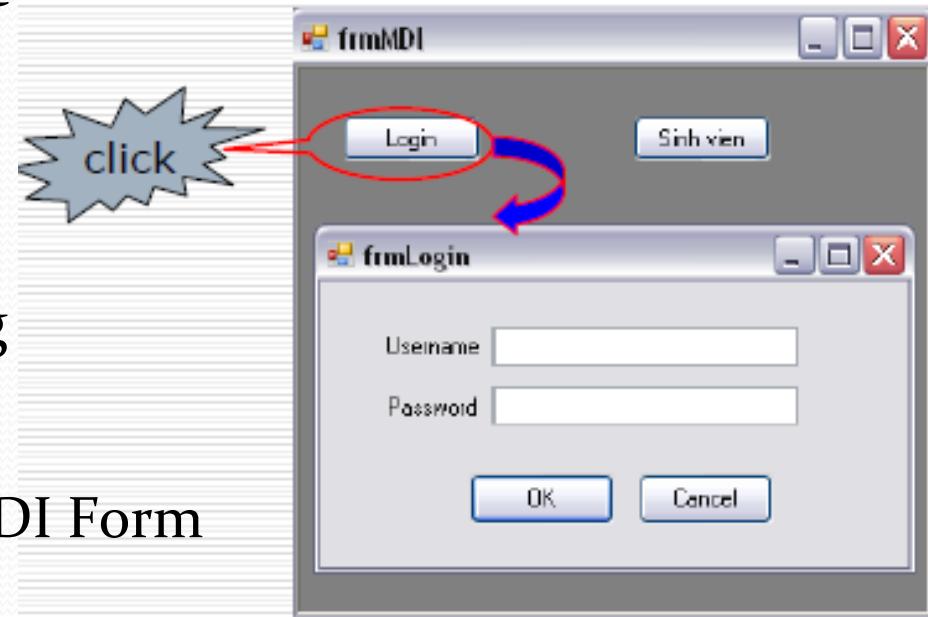
# Giao diện MDI

➤ Multiple Document Interface (MDI)

- Thuộc tính IsMdiContainer: true

➤ Child Form: nằm trong vùng làm việc của MDI Form

- Thuộc tính MdiParent: MDI Form



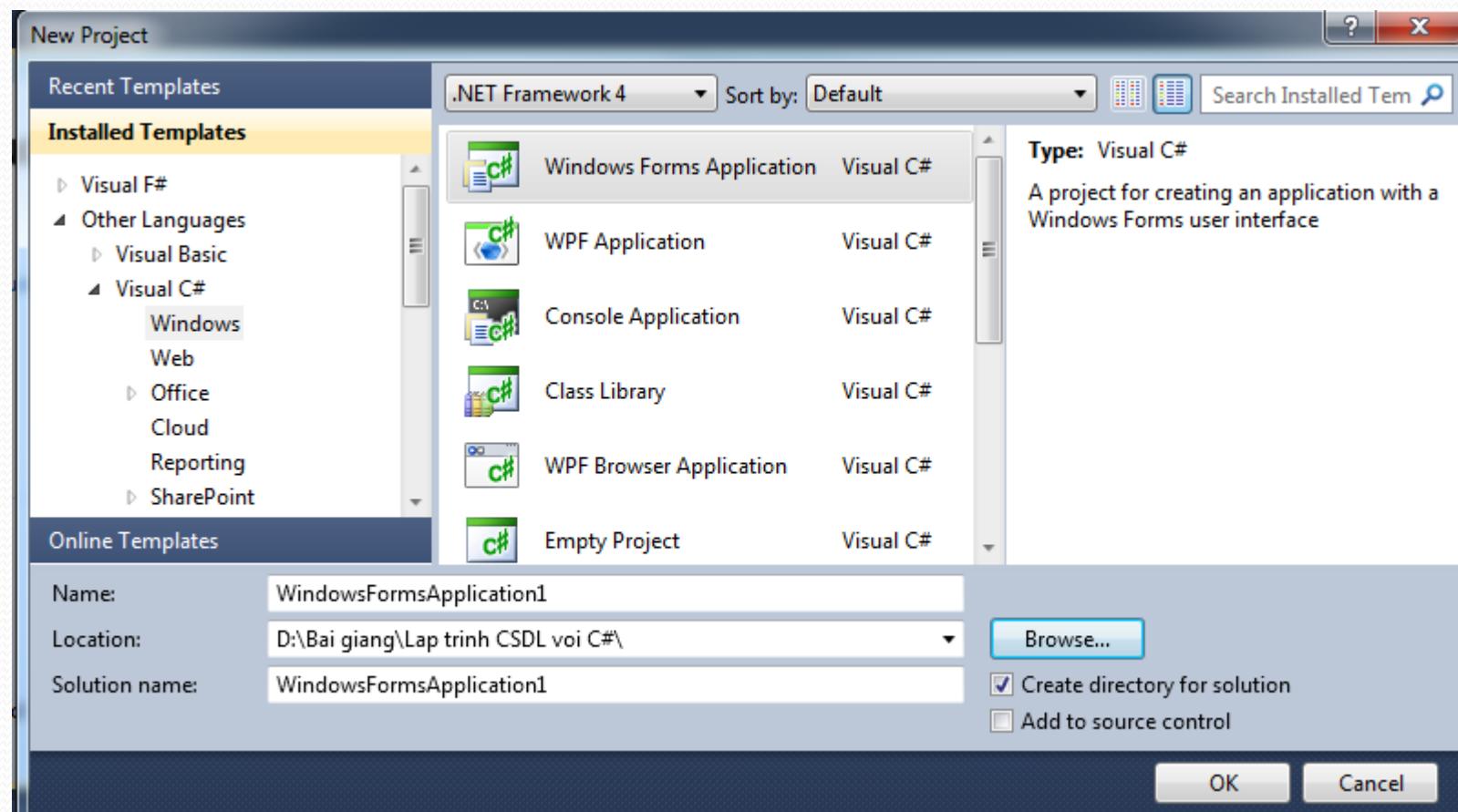
# Thêm Form mới

- Thêm Form từ màn hình thiết kế
  - Chọn **Project** → **Add Windows Form**
  - Chọn **Windows Form** → gõ tên **Form** → **Add**
- Sử dụng lệnh để tạo Form

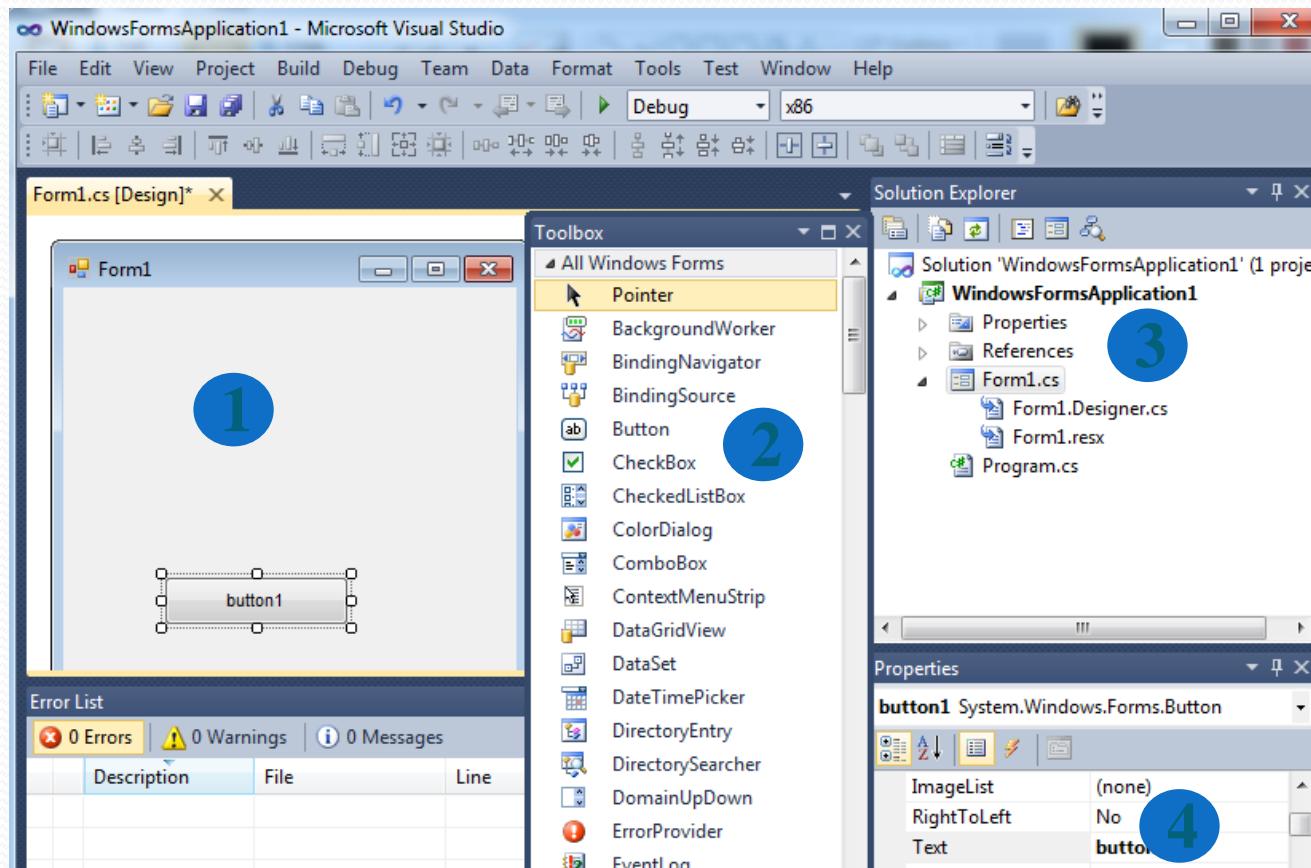
```
Form frm=new Form();
```

- Các kiểu **Form**
  - Dialog
  - Window

# Form từ màn hình thiết kế

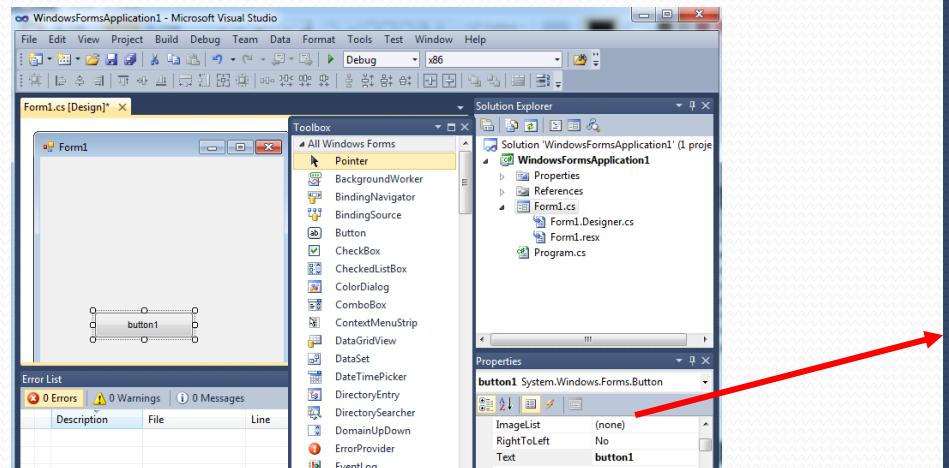


# Form từ màn hình thiết kế



- 1: Form ứng dụng
- 2: Control toolbox
- 3: Solution Explorer
- 4: Form properties

# Thuộc tính của Form



Properties	
Form1 System.Windows.Forms.Form	
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	Accessibility
AccessibleDescription	
AccessibleName	
AccessibleRole	Default
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	Appearance
BackColor	<input type="color"/> Control
BackgroundImage	<input type="file"/> (none)
BackgroundImageLayout	Tile
Cursor	Default
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	Font
Font	Arial, 8.25pt
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	ForeColor
ForeColor	<input type="color"/> ControlText
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	FormBorderStyle
FormBorderStyle	Sizable
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	RightToLeft
RightToLeft	No
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	RightToLeftLayout
RightToLeftLayout	False
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	Text
Text	Form1
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	UseWaitCursor
UseWaitCursor	False
<input type="button"/> <input type="checkbox"/> <input type="radio"/>	Behavior
AllowDrop	False
AutoValidate	EnablePreventFocusCh
ContextMenuStrip	(none)
DoubleBuffered	False
Enabled	True
<b>Text</b>	The text associated with the control.

# Thuộc tính của Form

## ➤ Thuộc tính nhận dạng:

- Name: mỗi Form có một giá trị duy nhất trong một Project.
- Text: chuỗi trên thanh tiêu đề
- Opacity: làm trong suốt bề mặt của Form.
- Icon: hình ảnh làm biểu tượng Form.
- ShowIcon: hiển thị|không hiển thị biểu tượng Form.

# Thuộc tính của Form

## ➤ Các thuộc tính thường dùng

- **AcceptButton**: Nút được click khi ấn phím *Enter*
- **CancelButton**: Nút được click khi ấn phím *Esc*
- **BackgroundImage**: Ảnh nền của **Form**
- **Font**: Font hiển thị của **Form** và **Font** ngầm định của các đối tượng của **Form**.
- **FormBorderStyle**: Kiểu đường viền của **Form**
  - **None**: **Form** không có đường viền
  - **Fix...**: Cố định kích thước khi chạy **Form**
  - **Sizeable**: Có thể thay đổi kích thước **Form**

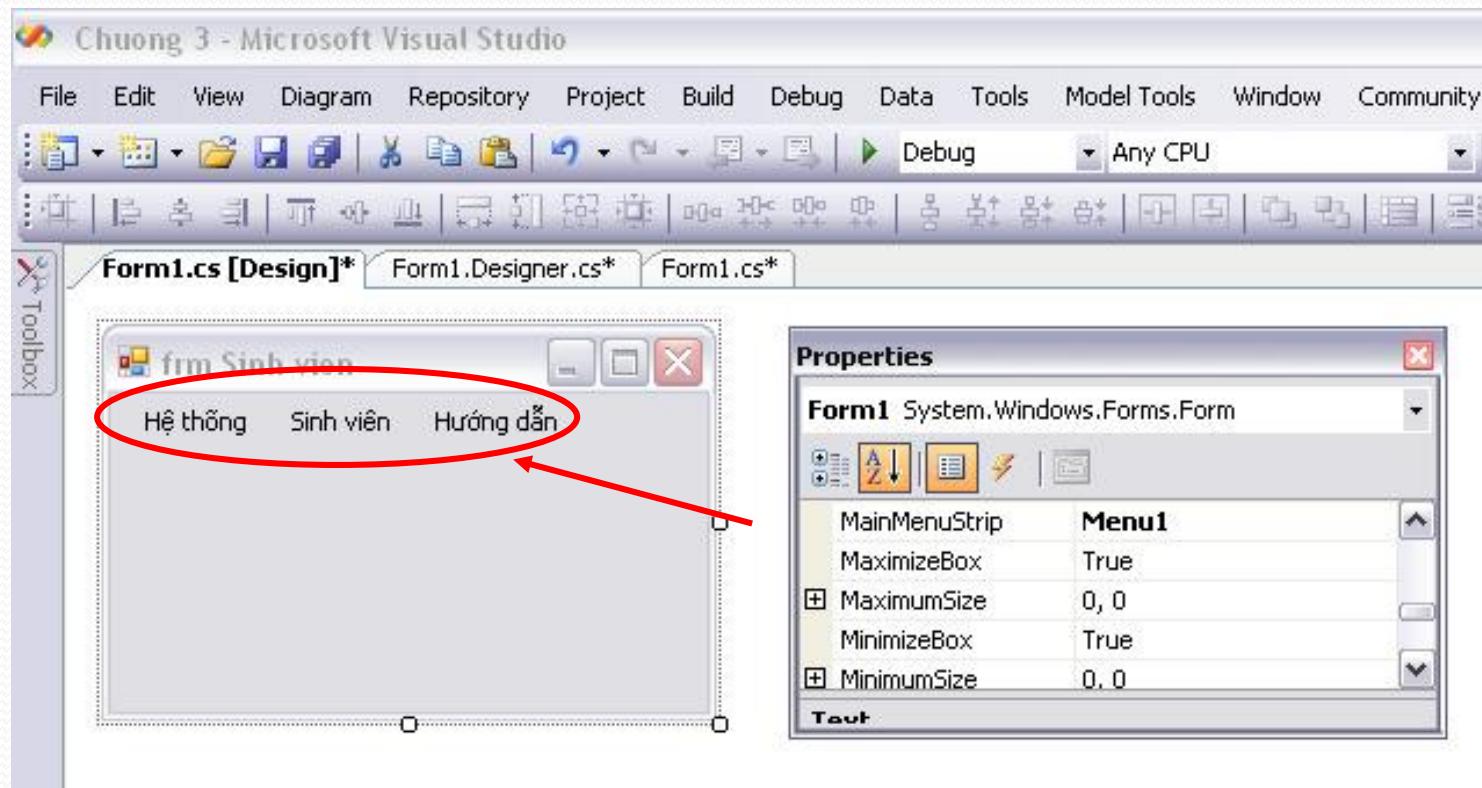
# Thuộc tính của Form

## ➤ Các thuộc tính thường dùng

- **ForeColor:** Màu chữ của Form và màu chữ của các đối tượng của Form.
- **MaximizeBox:** Có/không nút phóng to
- **MinimizeBox:** Có/không nút thu nhỏ
- **StartPosition:** Ví trí bắt đầu khi chạy Form
  - **CenterScreen:** Nằm giữa màn hình
- **WindowState:** Xác định trạng thái ban đầu Form

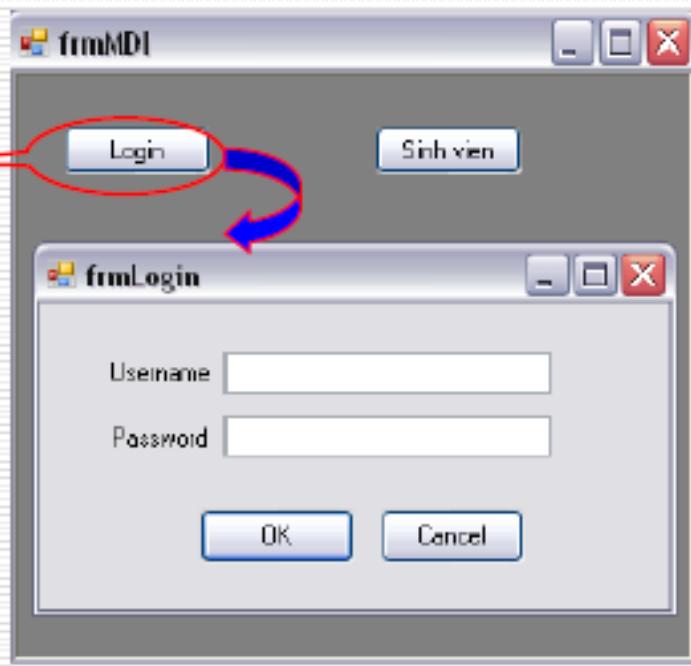
# Thuộc tính thực đơn

## ➤ MainMenuStrip



# Thuộc tính ứng với Child Form

- MdiParent: chỉ định MDI Form
- MdiChildren: danh sách Child Form



```
private void btnLogin_Click(object sender, EventArgs e)
{
    if (this.ActiveMdiChild != null)
        this.ActiveMdiChild.Close();
    frmLogin frm = new frmLogin();
    frm.MdiParent = this;
    frm.Show();
}
```

# Các phương thức của Form

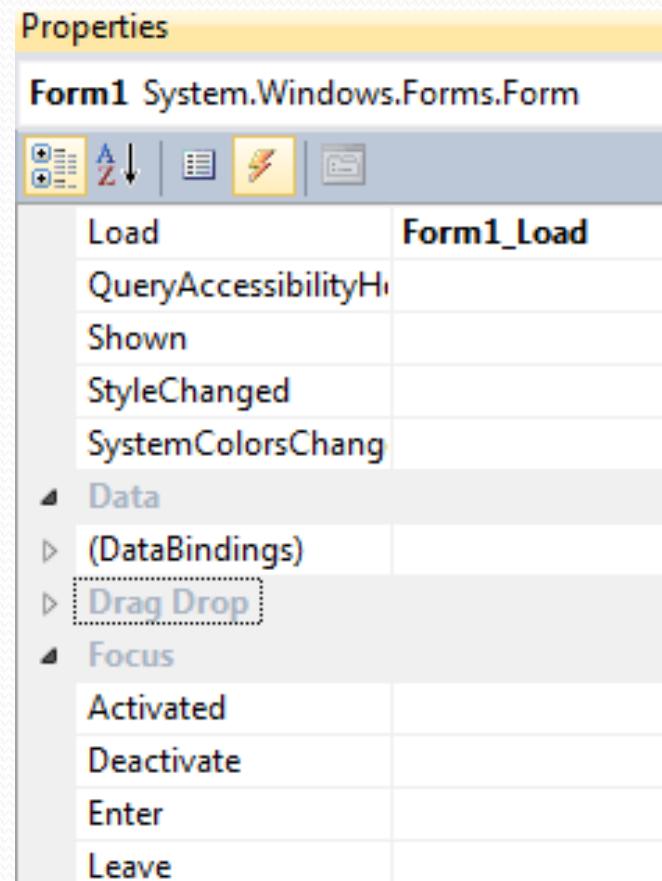
## ➤ Các phương thức thường dùng

- **Close:** Đóng Form và giải phóng các tài nguyên. Một Form đã đóng không thể mở lại.
- **Hide:** Ẩn Form và không giải phóng tài nguyên của Form.
- **Show:** Hiển thị một Form đã ẩn.

# Các sự kiện của Form

## ➤ Các sự kiện thường dùng

- **Load:** Xảy ra khi chạy Form (ngầm định khi nháy đúp chuột trong chế độ thiết kế).
- **FormClosing:** Xảy ra khi đóng Form.



# Nội dung



Giới thiệu Windows Form (WF)



Các điều khiển thông thường



Các điều khiển đặc biệt



Điều khiển Menu và Container

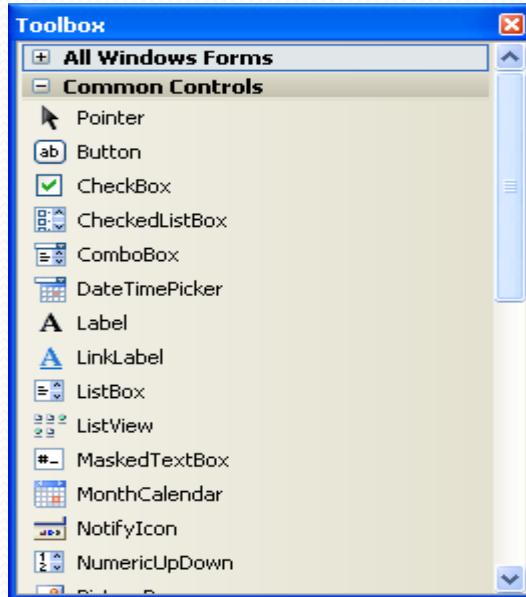


Sự kiện bàn phím, chuột và lớp MessageBox

## 3.2. Các điều khiển thông thường

### ➤ Các điều khiển (Control) của Form

- Là các thành phần đồ họa như Label, TextBox,...
- Namespace: System.Windows.Forms



## 3.2. Các điều khiển thông thường

- Mỗi điều khiển tạo ra các đối tượng cùng lớp
- Các đối tượng có:
  - **Properties:** Các thuộc tính mô tả đối tượng
  - **Methods:** Các phương thức thực hiện các chức năng của đối tượng.
  - **Events:** Các sự kiện sinh ra bởi sự chuyển động của bàn phím và con chuột, chi tiết do người lập trình viết.
- **Chú ý:** Các thuộc tính, sự kiện của các đối tượng có cùng tên → cùng ý nghĩa.

# Đặt tên theo tiền tố

- Mỗi form hay điều khiển có 1 tên
- Qui tắc đặt tên theo tiền tố

Điều khiển	Tiền tố	Ví dụ
Check box	chk	chkReadOnly
Combobox	cbo	cboEnglish
button	btn	btnExit
Form	frm	frmEntry

Tham khảo tại <https://msdn.microsoft.com/en-us/library/aa263493%28v=vs6.0%29.aspx>

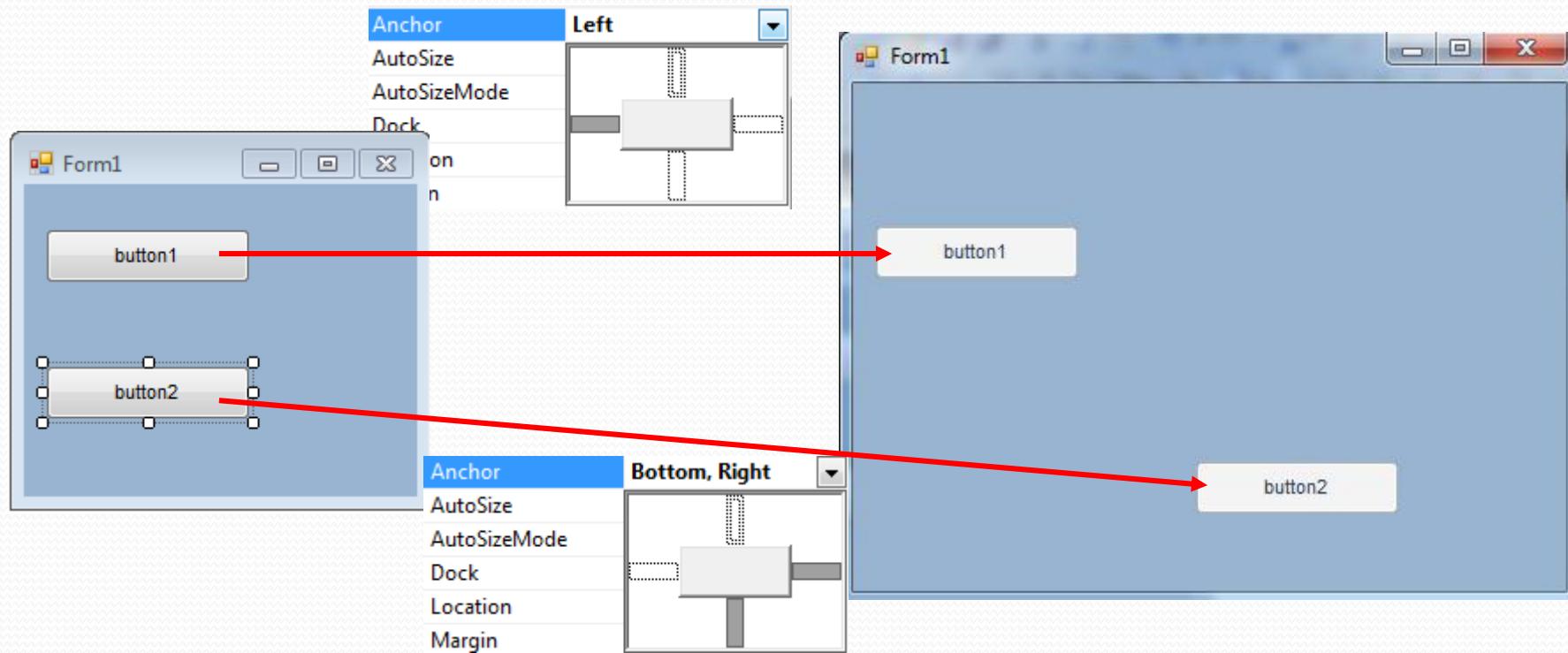
# Một số thuộc tính của control

Properties	Description
Text	Mô tả chuỗi xuất hiện trên control
Focus	phương thức chuyển focus vào control
TabIndex	Thứ tự tab của control (mặc định được VS.NET thiết lập)
Enabled	Thiết lập trạng thái của Control
Visible	Xác định hiển thị   ẩn Control
Anchor	Neo giữ control ở vị trí xác định, cho phép control di chuyển theo vị trí tương ứng với resize của Form.
Size	Xác nhận kích thước của control
Dock	Các control có thể gắn với một cạnh nào đó của Form, hoặc container của control.

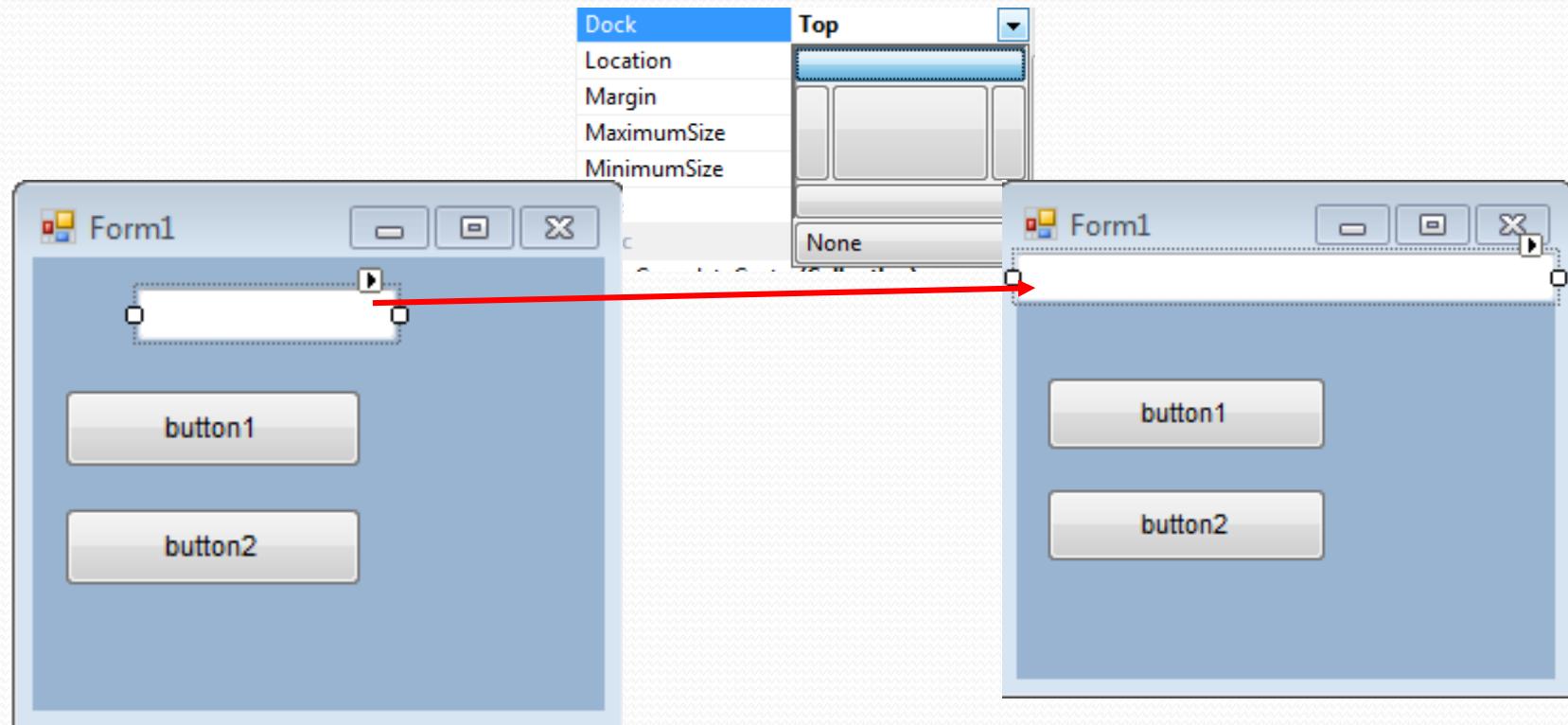
# Một số thuộc tính của control

Properties	Description
BackColor	Màu nền của control
BackgroundImage	Ảnh nền của control
ForeColor	Màu hiển thị text trên form
Enabled	Xác định khi control trạng thái enable
Focused	Xác định khi control nhận focus
Font	Font hiển thị text trên control
TabIndex	Thứ tự tab của control
TabStop	Nếu true, user có thể sử dụng tab để select control
Text	Text hiển thị trên form
TextAlign	Canh lề text trên control
Visible	Xác định hiển thị control

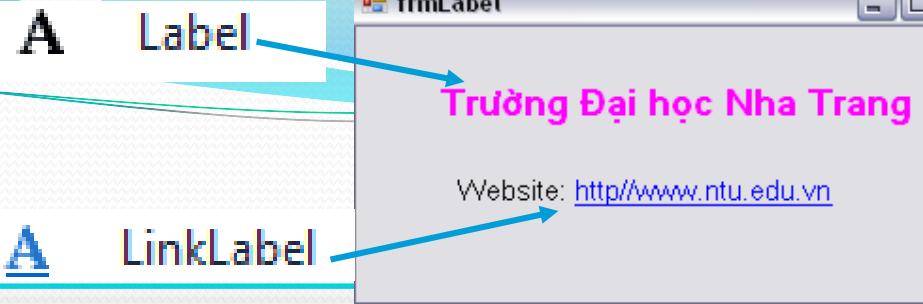
# Anchor



# Dock



# Label, LinkLabel

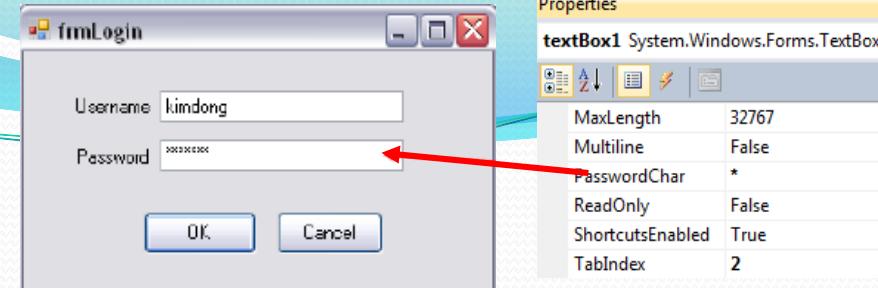


- Label: Đổi tượng hiển thị văn bản kết hợp hình ảnh
- Không sửa được văn bản hiển thị
- Các thuộc tính thường dùng
  - **AutoSize**: Tự thay đổi kích thước của đổi tượng
  - **Fo~~n~~ts**: Font chữ của đổi tượng **Label**
  - **ForeColor**: Màu chữ của đổi tượng
  - **Image**: Ảnh của đổi tượng
  - **Text**: Văn bản xuất hiện trên đổi tượng.
  - **TextAlign**: Lề của văn bản.

abl

## TextBox

# Textbox



- Đối tượng dùng để nhập dữ liệu từ bàn phím
- Các thuộc tính thường dùng
  - **Enabled:** Có/không cho phép thao tác đối tượng
  - **Multiline:** Có/không cho phép nhập dữ liệu nhiều dòng (ngầm định là không)
  - **PasswordChar:** Nhập ký tự làm mật khẩu
  - **ReadOnly:** Có/không cho phép sửa dữ liệu của đối tượng (ngầm định là có)
  - **Text:** Văn bản nhập (hiển thị) của đối tượng.

# Textbox

## ➤ Các sự kiện thường dùng

- **TextChanged:** Xảy ra khi nhập hoặc xoá các ký tự (ngầm định khi nháy đúp chuột trong chế độ thiết kế)
- **KeyDown:** Xảy ra khi ấn một phím bất kỳ trên đối tượng.
- **KeyUp:** Xảy ra khi thả một phím ấn trên đối tượng.

*Chú ý: Dữ liệu nhập vào TextBox là văn bản do đó nếu thực hiện các phép toán số học, logic thì cần chuyển sang kiểu số.*

# Button



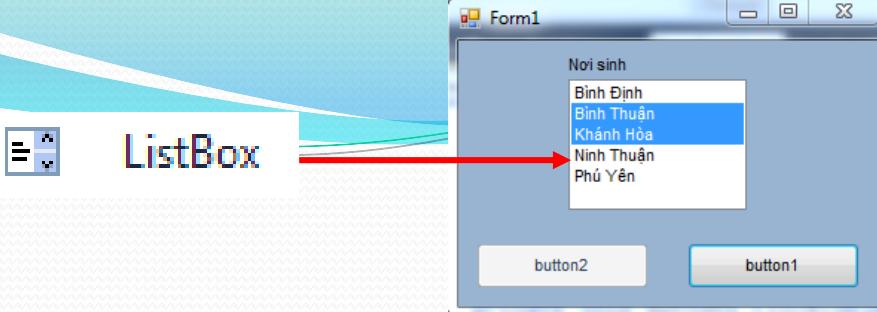
- Đối tượng nút ấn cho phép thực hiện một chức năng
- Có thể hiển thị hình ảnh kết hợp với văn bản
- Các thuộc tính thường dùng
  - **Text:** Văn bản hiển thị trên đối tượng
  - **Image:** Hình ảnh hiển thị trên đối tượng
- Các sự kiện thường dùng
  - **Click:** Xảy ra khi nhấn con trỏ chuột hoặc gõ Enter trên đối tượng (ngầm định khi nháy đúp chuột trong chế độ thiết kế).

# Ví dụ

## ➤ Giải phương trình bậc nhất

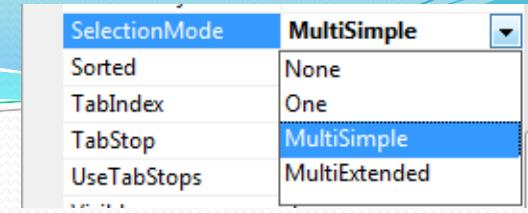
- Nhập hệ số a, hệ số b
- Chọn nút <Giải>
  - Kiểm tra dữ liệu nhập ?
  - Nếu không phải là số ?
  - Chuyển thành số a, b ?
  - Nếu  $a = 0$ 
    - Nếu  $b = 0 \rightarrow$  Vô số nghiệm
    - Nếu  $b \neq 0 \rightarrow$  Vô nghiệm
  - Nếu  $a \neq 0 \rightarrow x = -b/a$





## Listbox

- Trình bày danh sách phần tử dạng liệt kê và cho phép người sử dụng xem và chọn các phần tử.
- Các thuộc tính:
  - **MultiColumn:** Có/không chia **ListBox** thành nhiều cột.
  - **Items:** tập các phần tử trong **ListBox**
  - **SelectedIndex:** Trả về dòng hiện thời được chọn
    - Nếu chọn nhiều dòng: thuộc tính trả về 1 giá trị tùy ý của các dòng được chọn.
    - Nếu không chọn dòng nào: giá trị -1.



# Listbox

## ➤ Các thuộc tính:

- **SelectedIndices**: Trả về một mảng các chỉ số của các dòng được chọn.
- **SelectedItem**: Trả về giá trị dòng được chọn.
- **SelectedItems**: Trả về một mảng giá trị các dòng được chọn.
- **Sorted**: Có/Không sắp xếp dữ liệu trong **ListBox**, ngầm định là **False**.
- **SelectionMode**: Xác định số lượng dòng được chọn của **ListBox**.
  - one: Một dòng
  - Multi: Nhiều dòng

# Listbox

---

## ➤ Các phương thức thường dùng

- **GetSelected(index)** : Trả về giá trị **True** nếu dòng **Index** được chọn, ngược lại trả về **False**.
- **Add**: Thêm một dòng vào **ListBox**
  - `listBox1.Items.Add("Phú Yên");`
- **RemoveAt (row)** : Xoá dòng ở vị trí row
  - `listBox1.Items.RemoveAt(row);`
- **Clear**: Xoá tất cả các dòng
  - `listBox1.Items.Clear();`

# Listbox

---

## ➤ Sự kiện thường dùng

- **SelectedIndexChanged:** Xảy ra khi chọn một dòng.



ComboBox



# Combobox

- Là sự kết hợp của **TextBox** và **ListBox**. Dùng để trình bày danh sách các phần tử (item).
- Các thuộc tính thường dùng
  - **DataSource**: tập dữ liệu điền vào điều khiển
  - **DisplayMember**: Tên của trường tương ứng với chuỗi trình bày trên điều khiển
  - **DropDownStyle**: Xác định kiểu của ComboBox.
    - **Simple**: Chọn hoặc gõ giá trị
    - **DropDown** (ngầm định): Chọn hoặc gõ giá trị
    - **DropDownList**: Chỉ cho phép chọn giá trị.



## ComboBox



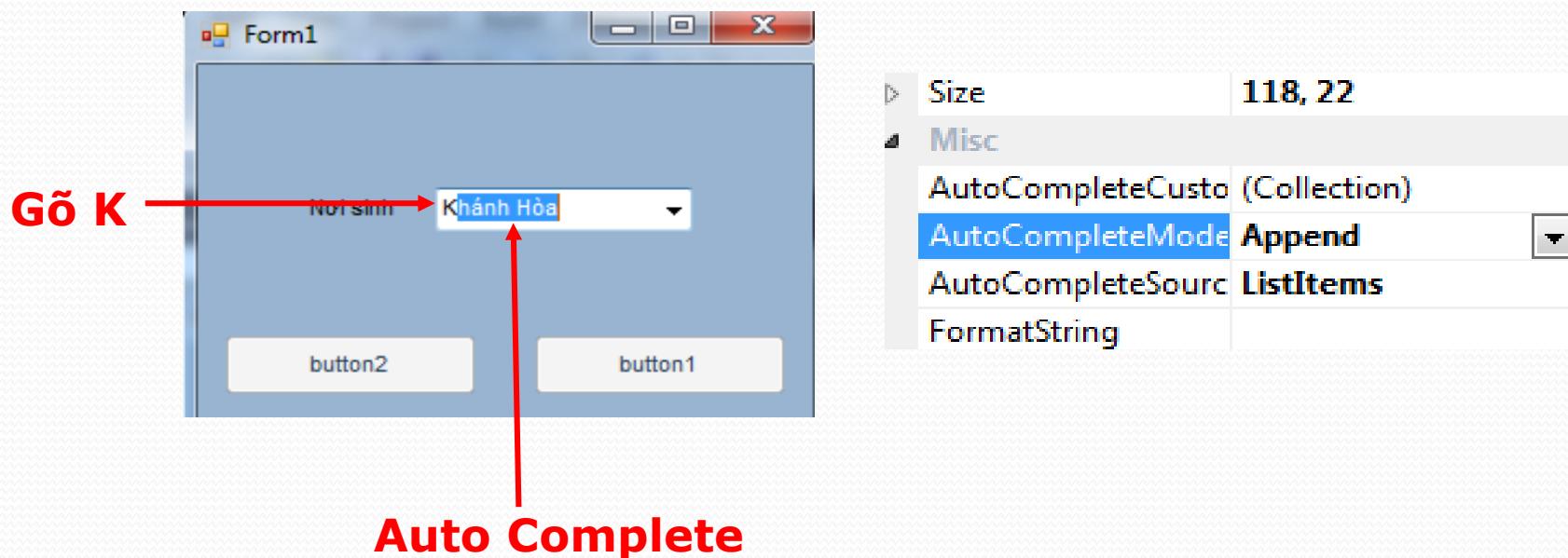
# Combobox

### ➤ Các thuộc tính thường dùng

- **Items**: tập các phần tử có trong điều khiển
- **Text**: giá trị chuỗi ứng với nhãn đang chọn
- **SelectedIndex**: Chỉ số dòng được chọn. Nếu không chọn có giá trị **-1**.
- **SelectedItem**: Giá trị dòng được chọn.
- **Sorted**: Có/Không sắp xếp dữ liệu trong **ComboBox**, ngầm định là **false**.

# Combobox

- Các thuộc tính thường dùng
  - **AutoCompleteMode**



# Combobox

---

## ➤ Các sự kiện:

- **SelectValueChanged:** xảy ra khi giá trị của phần tử được thay đổi.
- **SelectedIndexChanged:** Xảy ra khi chọn 1 dòng.

# Combobox

---

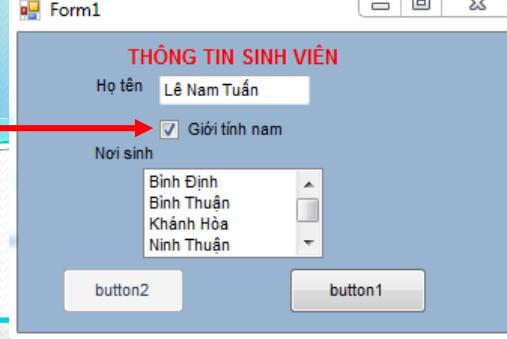
## ➤ Các Phương thức:

- **Add**: Thêm một dòng vào **ComboBox**
  - **comboBox1.Items.Add("Khánh Hòa");**
- **RemoveAt (row)** : Xoá dòng ở vị trí row
  - **comboBox1.Items.RemoveAt(row);**
- **Clear**: Xoá tất cả các dòng trong **ComboBox**
  - **comboBox1.Items.Clear();**

# Checkbox



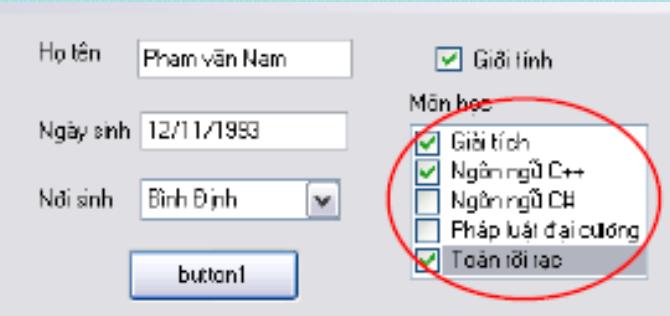
CheckBox



- Đối tượng cho phép chọn/không chọn giá trị
- Cho phép chọn đồng thời nhiều đối tượng
- Các thuộc tính thường dùng
  - **Checked**: Có/không đối tượng được chọn
  - **Text**: Văn bản hiển thị trên đối tượng
- Các sự kiện thường dùng
  - **CheckedChanged**: Xảy ra khi chọn/không chọn đối tượng (ngầm định khi nháy đúp chuột trong chế độ thiết kế)

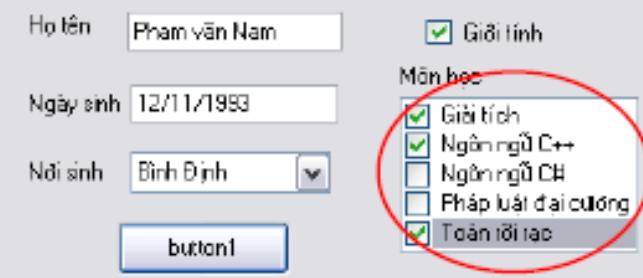


# CheckListBox

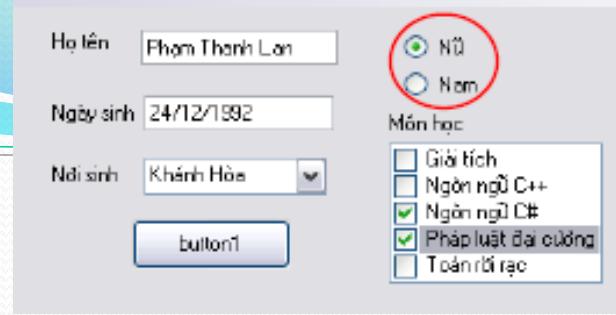


- **CheckedListBox** là sự mở rộng của **ListBox** bằng cách thêm **CheckBox** ở phía bên trái mỗi dòng
- Cho phép chọn một hay tập giá trị
- Các thuộc tính thường dùng
  - **CheckedItems**: Mảng các giá trị của dòng được đánh dấu **Check**.
  - **CheckedIndices**: Mảng các chỉ số dòng được đánh dấu **Check**.
  - **SelectionMode**: cho phép chọn nhiều phần tử.

# CheckListBox

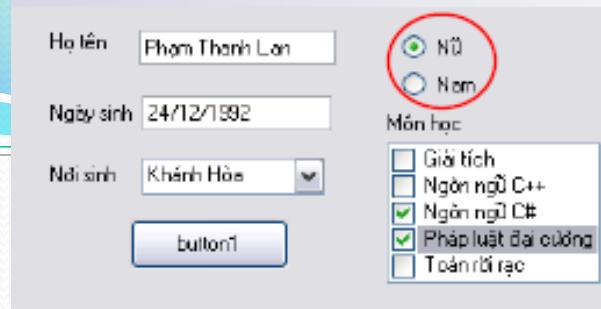


- Phương thức thường dùng
  - **GetItemChecked(index)**: Trả về **true** nếu dòng **index** được chọn.
- Sự kiện thường dùng
  - **ItemCheck**: Xảy ra khi dòng được **checked** hoặc **unchecked**.



# RadioButton

- Đổi tượng cho phép chọn/không chọn giá trị
- Cho phép chọn một đối tượng ở một thời điểm
  - Để chọn nhiều đối tượng phải đặt các điều khiển trong GroupBox hoặc Panel
- Các thuộc tính thường dùng
  - **Checked:** Có/không đối tượng được chọn
  - **Text:** Văn bản hiển thị trên đối tượng



# RadioButton

## ➤ Các sự kiện thường dùng

- **Click:** Xảy ra khi đối tượng được click.
- **CheckedChanged:** Xảy ra khi chọn/không chọn đối tượng (ngầm định khi nháy đúp chuột trong chế độ thiết kế)

# Nội dung



Giới thiệu Windows Form (WF)



Các điều khiển thông thường



Các điều khiển đặc biệt



Điều khiển Menu và Container



Sự kiện bàn phím, chuột và lớp MessageBox

# ListView

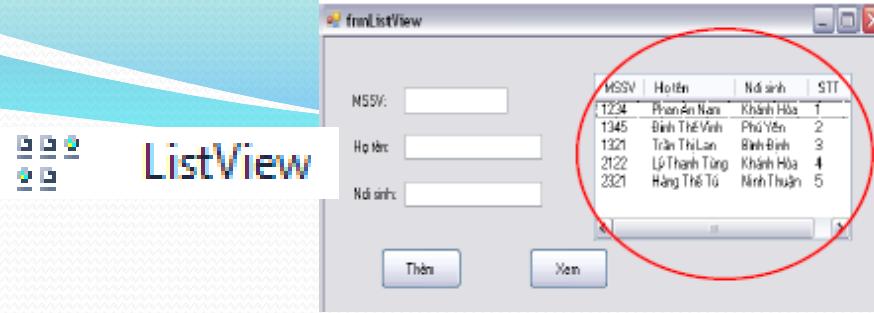


## ListView



- Dùng để hiển thị dữ liệu theo các dòng và các cột
  - Có thể chọn một hoặc nhiều dòng
  - Có thể hiển thị các biểu tượng theo các dòng
- Các thuộc tính thường dùng
  - **Columns:** Các cột hiển thị trong chế độ **Details**.
  - **GridLines:** Hiển thị lưới (chỉ hiển thị trong chế độ **Details**).
  - **Items:** Mảng các dòng (**ListViewItems**) trong ListView.

# ListView



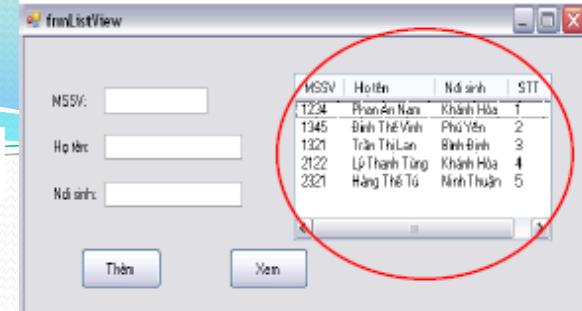
## ➤ Các thuộc tính thường dùng

- **LargeImageList**: Danh sách ảnh (**ImageList**) hiển thị trên ListView.
- **MultiSelect**: Có/Không cho phép chọn nhiều dòng (ngầm định là **True**).
- **SelectedItems**: Mảng các dòng được chọn.
- **View**: Kiểu hiển thị của ListView
  - **Icons**: Hiển thị danh sách theo các biểu tượng
  - **List**: Hiển thị danh sách theo một cột
  - **Details**: Hiển thị ListView theo danh sách nhiều cột

# ListView



## ListView



### ➤ Các phương thức thường dùng

- **Add:** Thêm một dòng vào ListView
- **Clear:** Xoá tất cả các dòng của ListView
- **Remove:** Xoá một dòng trong ListView
- **RemoveAt (index)** : Xoá một dòng ở vị trí index

### ➤ Sự kiện thường dùng

- **ItemSelectionChanged:** Xảy ra khi chọn một dòng.
- **SelectedIndexChanged**

# ListView



## ListView

frmListView

MSSV	Họ tên	Nơi sinh	STT
11234	Phan An Nan	Khánh Hòa	1
1345	Bình Thế Vinh	Phú Yên	2
1321	Trần Thị Lan	Bình Định	3
2122	Lý Thành Tùng	Khánh Hòa	4
2321	Hàng Thanh Tú	Ninh Thuận	5

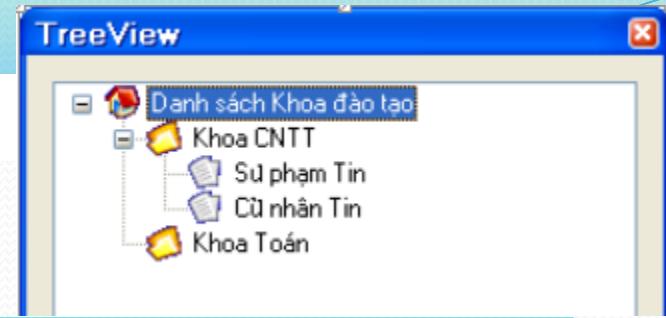
MSSV:   
Họ tên:   
Nơi sinh:

Thêm Xem

- Thêm các item vào ListView
  - Thêm item trong màn hình thiết kế form
  - Thêm item thông qua code
- Các lớp định nghĩa Item
  - **System.Windows.Forms.ListViewItem**
  - Mỗi item trong ListView có các item phụ gọi là subitem
    - Lớp **ListViewItem.ListViewSubItem** định nghĩa các subitem của ListView
    - Lớp **ListViewSubItem** là inner class của **ListViewItem**



## TreeView



# TreeView

- Trình bày danh sách phần tử phân cấp theo từng nút hình cây.
- Các thuộc tính:
  - **Nodes**: Mảng các **TreeNodes** trong TreeView.
    - **Nodes .Add**: Bổ sung một node vào cây.
    - **Nodes .Clear**: Xoá toàn bộ các node trên cây.
    - **Nodes .Remove**: Xoá một node trên cây và các node con của nó.
  - **SelectedNode**: Node hiện thời được chọn



# TreeView

## ➤ Các thuộc tính thường dùng

- **CheckBoxes**: Có/không xuất hiện các checkbox trên các node. Ngầm định là **False**.
- **Checked**: Có/không một **Node** được check (thuộc tính **CheckBoxes** phải được đặt là **True**)
- **ImageList**: Chỉ ra danh sách ảnh hiển thị trên các node.
  - **ImageList** là một mảng các đối tượng ảnh.
  - Tạo **ImageList** bằng cách kéo điều khiển vào Form, nháy chuột phải và chọn Choose Image để thêm các ảnh vào **ImageList**.



# TreeView

## ➤ Các thuộc tính thường dùng

- **SelectedImageIndex:** Chỉ ra chỉ số ảnh được hiển thị trên node khi node được chọn.
- **ImageIndex:** Chỉ ra chỉ số ảnh được hiển thị trên node khi node không được chọn (deselected).
- **Text:** Text hiển thị của Node.
- **FirstNode:** Node con đầu tiên của node.
- **LastNode:** Node con cuối cùng của node.
- **PrevNode:** Node con trước node con hiện thời.
- **NextNode:** Node con tiếp theo node hiện thời.



# TreeView

---

- Các phương thức thường dùng
  - **Collapse**: Thu nhỏ các node con của node.
  - **Expand**: Mở rộng các node con của node.
  - **ExpandAll**: Mở rộng tất cả các node con.
  - **GetNodeCount**: Trả về số lượng node con.
- Các sự kiện thường dùng
  - **AfterSelect**: Xảy ra khi một node được chọn (ngầm định khi nháy đúp chuột ở chế độ thiết kế).
  - **BeforeExpand**: Xảy ra khi mở rộng một node



# DateTimePicker / MonCalendar

---

- Cho phép người sử dụng chọn giá trị thời gian
- Các thuộc tính:
  - **Format:** định dạng hiển thị (long, short, time, custom)
  - **CustomFormat:**
    - dd: hiển thị 2 con số của ngày
    - MM: hiển thị 2 con số của tháng
    - yyyy: hiển thị 4 con số của năm



# DateTimePicker / MonCalendar

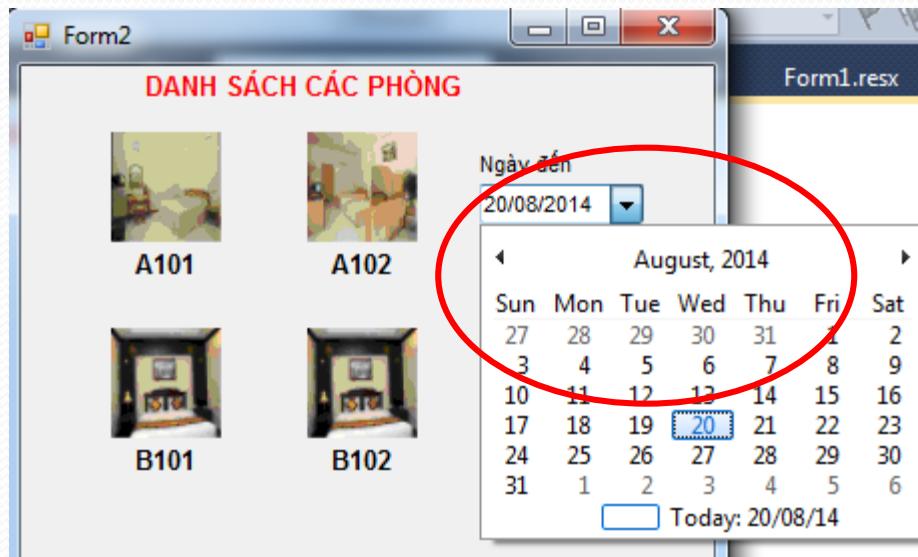
## ➤ Các thuộc tính:

- **CustomFormat, Format**: định dạng thời gian
- **MaxDate, MinDate**: ngày lớn nhất/ nhỏ nhất
- **Value**: giá trị thời gian trên điều khiển

## ➤ Các sự kiện:

- **ValueChanged**: xảy ra khi người dùng chọn giá trị khác với trước đó.

# DatePicker / MonCalendar



# PictureBox

- Sử dụng để hiển thị ảnh dạng bitmap, icon, JPEG, GIF, ...
- Sử dụng thuộc tính Image để thiết lập ảnh lúc design hoặc runtime.
- Các thuộc tính
  - Image: ảnh cần hiển thị
  - SizeMode: Normal, StretchImage, AutoSize, ...



PictureBox





A101



A102



B101



B102

# ImageList

- Cung cấp tập hợp những đối tượng image cho các control khác sử dụng
  - ListView
  - TreeView
- Các thuộc tính:
  - **ColorDepth**: độ sâu của màu
  - **Images**: trả về ImageList.ImageCollection
  - **ImageSize**: kích thước ảnh
  - **TransparentColor**: xác định màu là transparent



# ImageList

## ➤ Tạo ImageList

- Kéo control **ImageList** từ ToolBox thả vào Form
- Thiết lập kích thước của các ảnh: **ImageSize**
- Bổ sung các ảnh vào **ImageList** qua thuộc tính **Images**

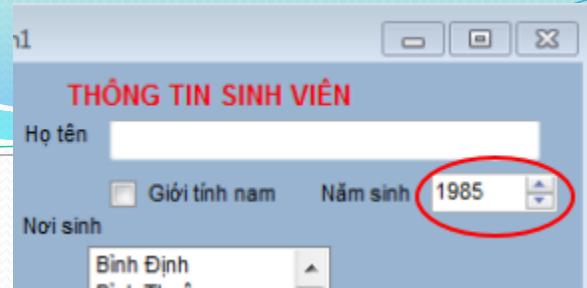
## ➤ Sử dụng **ImageList** cho các control

- Khai báo nguồn image là **imagelist** vừa tạo cho control (thuộc tính ImageList)
- Thiết lập các item/node với các **ImageIndex** tương ứng ( design view hoặc code view)



## NumericUpDown

# NumericUpDown

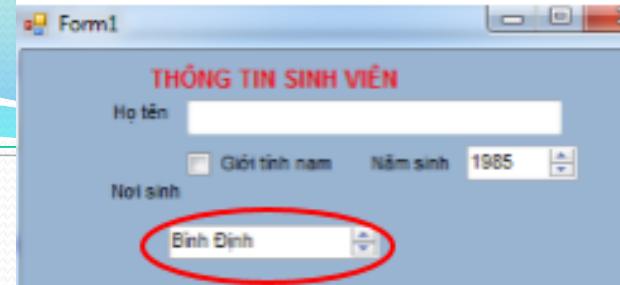


- Cho phép user chọn các giá trị trong khoảng xác định thông qua
  - Nút up & down
  - Nhập trực tiếp giá trị
- Các thuộc tính
  - Minimum
  - Maximum
  - Value
  - Increment
- Sự kiện: ValueChanged
- Phương thức: DownButton, UpButton



## DomainUpDown

# DomainUpDown



- Cho phép user chọn item trong sổ danh sách item thông qua
  - Button Up & Down
  - Nhập từ bàn phím
- Các thuộc tính:
  - **Items**: danh sách item
  - **ReadOnly**: true chỉ cho phép thay đổi giá trị qua Up & Down
  - **SelectedIndex**: chỉ mục của item đang chọn

# DomainUpDown

## ➤ Các thuộc tính:

- **SelectedItem**: item đang được chọn
- **Sorted**: sắp danh sách item
- **Text**: text đang hiển thị trên DomainUpDown.

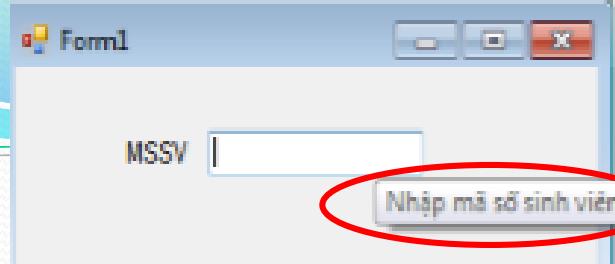
## ➤ Sự kiện

- **SelectedItemChanged**

# ToolTip



ToolTip



- Cung cấp chức năng hiển thị một khung text nhỏ khi user di chuyển chuột vào control bất kỳ
- Khung text chứa nội dung mô tả ý nghĩa của control
- Cách sử dụng
  - Từ ToolBox kéo **ToolTip** thả vào form
  - Kích chọn control muốn thêm **ToolTip**
  - Trong cửa sổ Properties của control sẽ có thuộc tính **ToolTip**. Thêm text vào thuộc tính này để hiển thị khi tooltip xuất hiện.

# Nội dung



Giới thiệu Windows Form (WF)



Các điều khiển thông thường



Các điều khiển đặc biệt



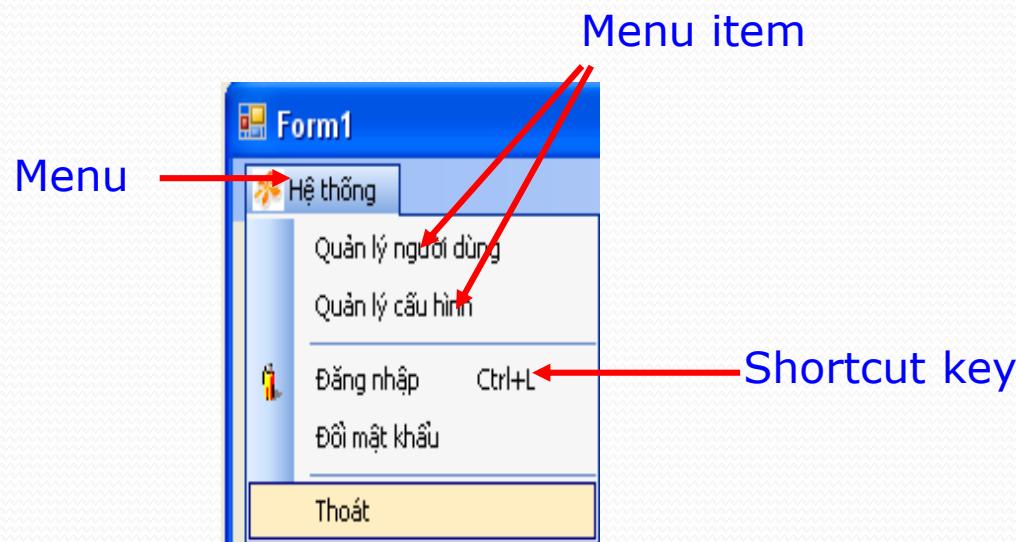
Điều khiển Menu và Container



Sự kiện bàn phím, chuột và lớp MessageBox

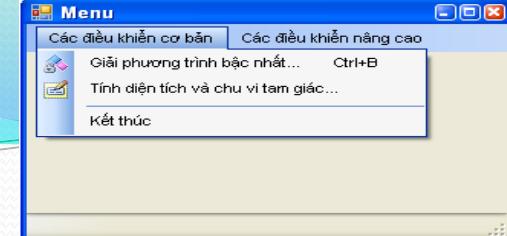
# Menu

- Menu cung cấp nhóm lệnh có quan hệ với nhau cho các ứng dụng Windows





## MenuStrip

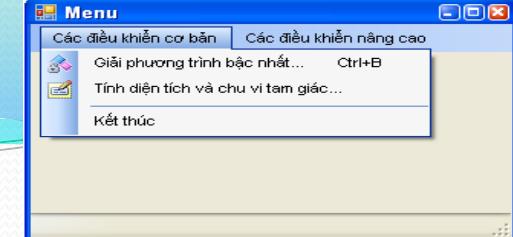


# MenuStrip

- Cho phép thiết kế menu trên Form
- Các thuộc tính thường dùng:
  - **Name**: Tên menu được dùng trong mã lệnh.
  - **ShortCutKey**: Đặt phím nóng cho menu
  - **ShowShortcut**: Có/Không phím nóng hiển thị trên dòng menu. Ngầm định là **true**.
  - **Text**: Xuất hiện trên dòng menu.
- Sự kiện thường dùng:
  - **Click**: Xảy ra khi một dòng của menu được click chuột hoặc ấn phím nóng.



# MenuStrip



## MenuStrip

### ➤ Tạo Menu:

- Kéo biểu tượng **MenuStrip** vào Form.
- Gõ nội dung và đặt tên cho các dòng của menu
  - Chọn **Properties** → **Name**
- Chèn hình ảnh cho các dòng của menu
  - R\_Click chọn **Set Image**, chọn **Local Resource** → **Import** → chọn hình ảnh
- Đặt phím nóng cho các dòng của menu
  - **Properties** → **ShortCutKey**



# ToolStrip

---

- ToolStrip là sự thay thế cho ToolBar trong các ứng dụng trước đây
- Vị trí thường xuất hiện là ngay bên dưới thanh menu
- Cung cấp các button cho phép thực hiện các chức năng thường dùng trong menu
- ToolStrip là dạng container cho phép chứa các control



# ToolStrip

## ➤ Xây dựng ToolBar

- Kéo điều khiển **ToolStrip** vào **Form**
- Nháy chuột vào biểu tượng phải và chọn đối tượng tạo **ToolBar**
  - **Button:** Nút ấn
  - **DropDownButton:** Nút sổ xuống
  - **Separator:** Đường phân cách



# ToolStrip

## ➤ Xây dựng ToolBar

- Đặt tên cho các nút của ToolBar
  - **Properties → Name**
  - Chèn hình ảnh cho đối tượng của ToolBar
    - R\_Click chọn Set Image, Chọn Local Resource → Import → chọn hình ảnh
  - Viết mã lệnh
    - Gọi từ menu: <đối tượng>.PerformClick()
    - Gọi trực tiếp đối tượng



# StatusStrip

---

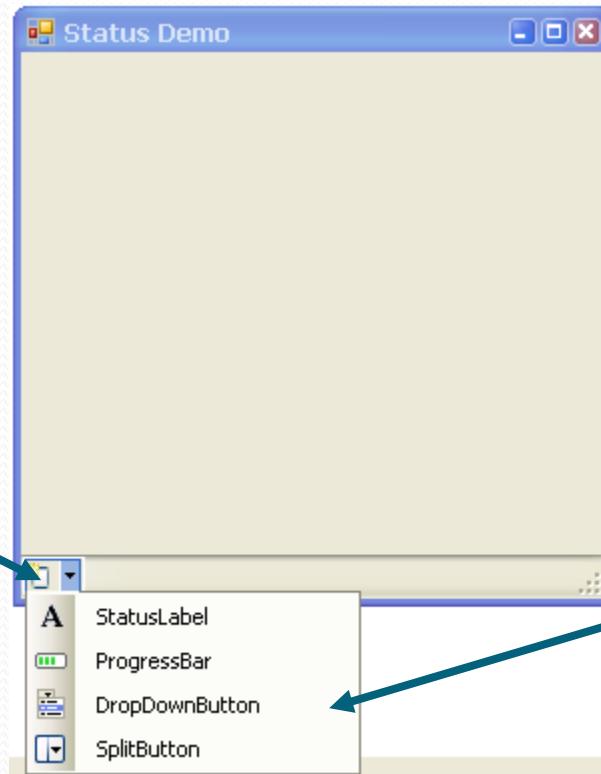
- Hiển thị thông tin trạng thái của ứng dụng
- Nằm bên dưới cùng của Form.
- Các lớp liên quan
  - StatusStrip: là container chứa control khác
  - ToolStripStatusLabel: control có thể add vào StatusStrip

# StatusStrip

- Tạo các item cho StatusStrip

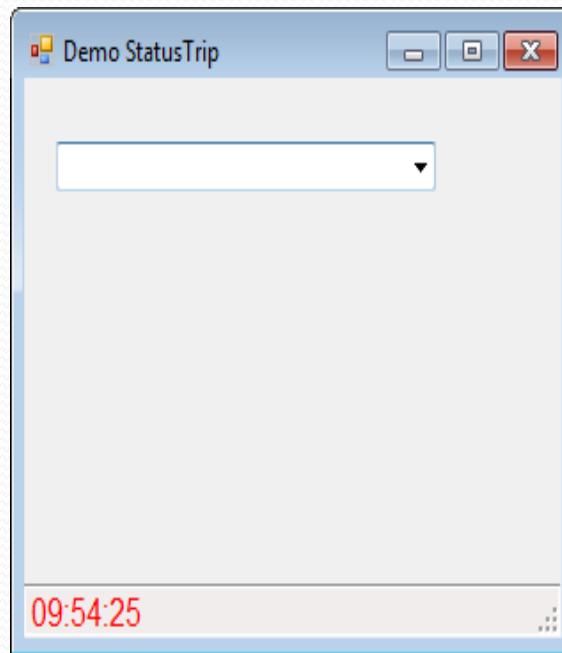
Tạo các item cho  
StatusStrip

Các kiểu control  
cho StatusStrip



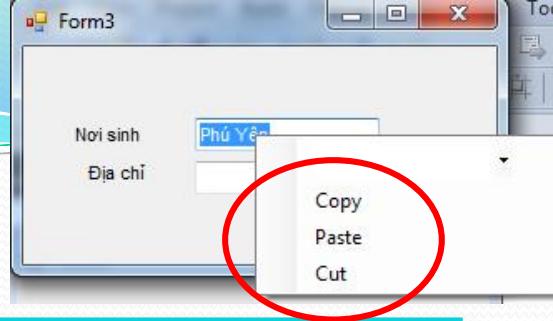
# Ví dụ

---

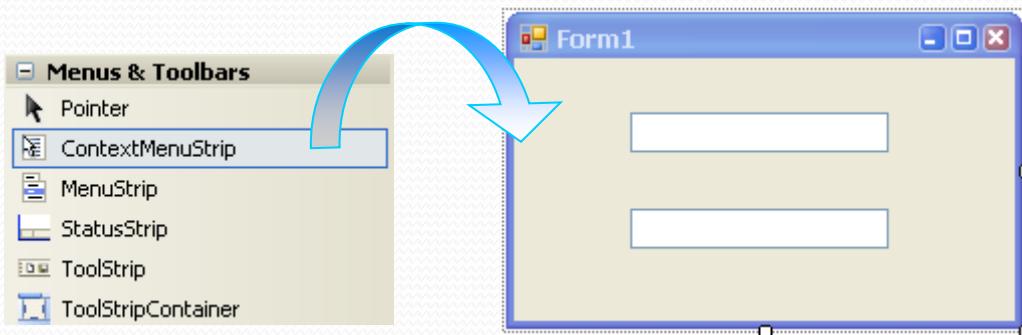


## ContextMenuStrip

# ContextMenuStrip



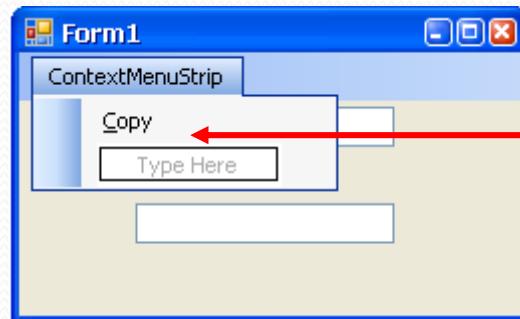
- Xuất hiện khi người dùng Click chuột phải
- Thông thường menu này xuất hiện tùy thuộc vào đối tượng trong vùng kích chuột phải.
- Trong ToolBox kéo **ContextMenuStrip** thả vào **Form**





# ContextMenuStrip

- Kích vào **ContextMenuStrip** để soạn thảo các menu item
- ContextMenuStrip tạm thời thể hiện trên cùng của form. Khi chạy thì sẽ không hiển thị cho đến khi được gọi



Soạn thảo Context  
Menu tương tự như  
Menu bình thường

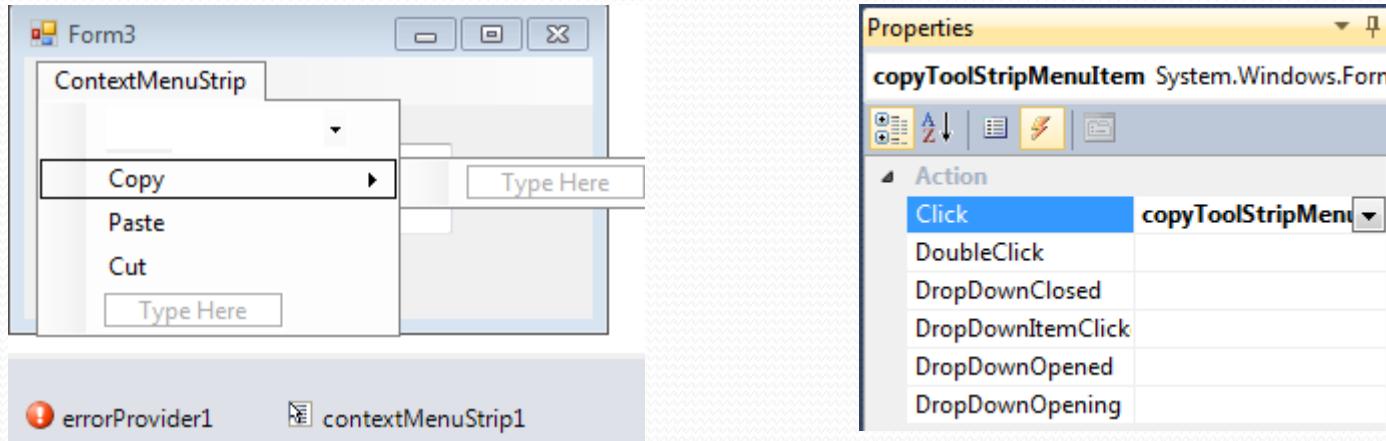




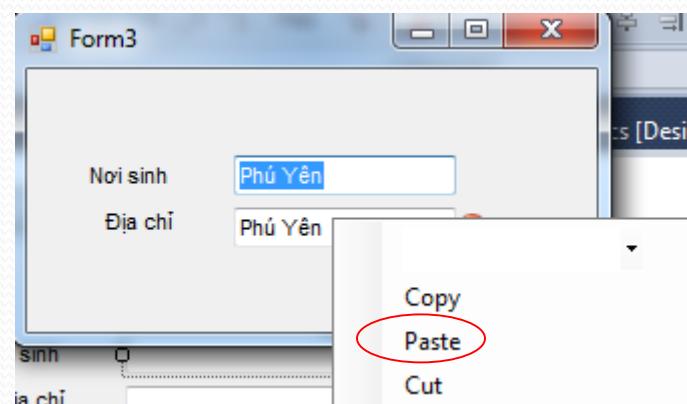
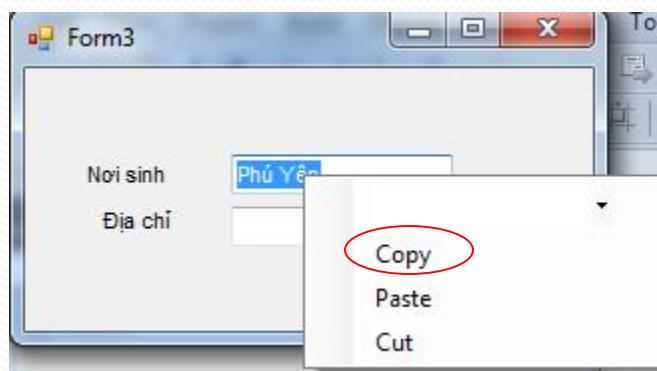
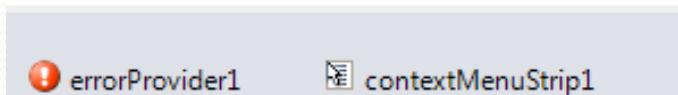
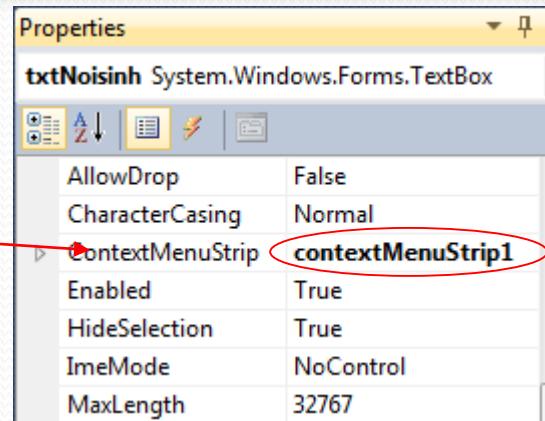
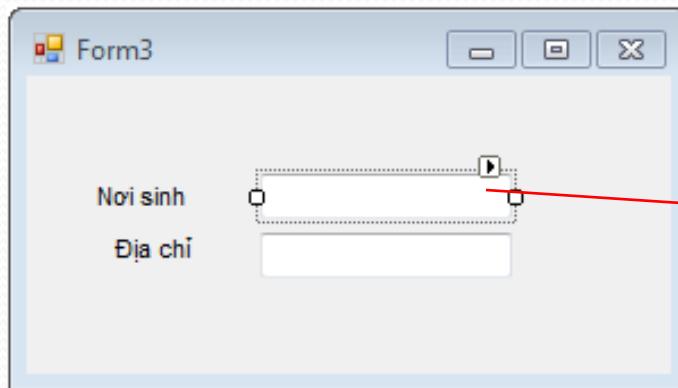
# ContextMenuStrip

- Mỗi control đều có property là: **ContextMenuStrip**
  - Khai báo thuộc tính này với ContextMenuStrip
    - Khi đó user kích chuột phải lên control thì sẽ hiển thị context Menu đã cài đặt sẵn
- Khai báo trình xử lý sự kiện Click cho ContextMenu
  - Kích đúp vào menu item của Context Menu để tạo
  - Hoặc trong cửa sổ Properties -> Event kích đúp vào sự kiện Click.

# ContextMenuStrip



# ContextMenuStrip



MSSV:	53161620
Họ tên:	Lê Thành Hưởng
Nơi sinh:	Khánh Hòa
Địa chỉ:	23 Lê Lợi - Khánh Hòa



# GroupBox

- Là khung dùng để nhóm các điều khiển khác.
- Các thuộc tính:
  - **Text:** tựa đề của điều khiển
  - **Controls:** danh sách control chứa trong Groupbox



## Panel

# Panel



- Chứa nhóm các control, không có tựa đề, có thanh cuộn (scrollbar)
- Các thuộc tính:
  - AutoScroll: Xuất hiện khi panel quá nhỏ để hiển thị hết các control, mặc định là false
  - BorderStyle: Biên của panel, mặc định là None, các tham số khác như Fixed3D, FixedSingle
  - Controls: danh sách control chứa trong Panel



## TabControl

# TabControl



- Cho phép thể hiện nhiều control trên một form
- Các control có cùng nhóm chức năng sẽ được tổ chức trong một tab (page)
- Các thuộc tính:
  - TabPages: chứa danh sách điều khiển TabPages
  - Appearance: dạng hiển thị
  - SelectedIndex: chỉ mục của Tab page được chọn
-

# TabControl

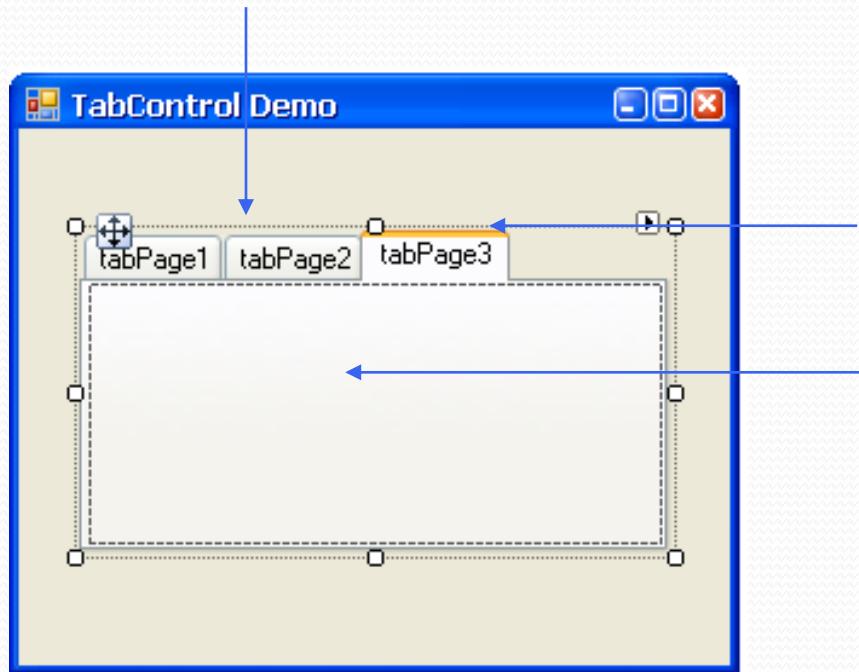
---

## ➤ Các sự kiện:

- SelectedIndexChanged: xảy ra khi giá trị của thuộc tính SelectIndex thay đổi

# TabControl

TabPage



TabControl

TabPage

# Nội dung



Giới thiệu Windows Form (WF)



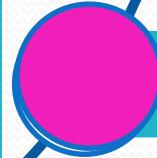
Các điều khiển thông thường



Các điều khiển đặc biệt



Điều khiển Menu và Container



Sự kiện bàn phím, chuột và lớp MessageBox

# Lập trình sự kiện

(Event- Driven Programming)

- Chương trình GUI thường dùng Event-Driven Programming
- Các đối tượng có thể kích hoạt sự kiện và các đối tượng khác phản ứng với những sự kiện đó
- Luồng chương trình được điều khiển bởi sự tương tác User-Computer

# Sự kiện (Event)



- Các sự kiện được sinh ra do điều khiển bàn phím hoặc con chuột.
- Các sự kiện của điều khiển đặt tên theo quy tắc:  
**ControlName\_EventName**.
- Firing an event: khi đối tượng khởi tạo sự kiện
- Listener: đối tượng chờ cho sự kiện xuất hiện
- Event handler: phương thức phản ứng lại sự kiện

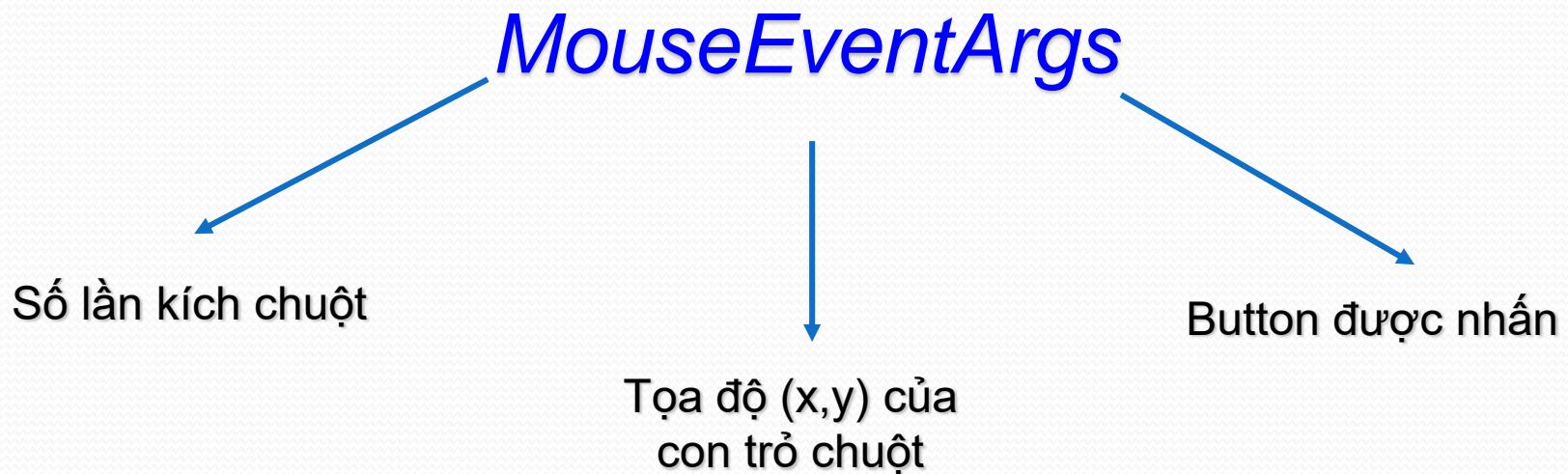


# Mouse Event

- Mouse là thiết bị tương tác thông dụng trên GUI
- Một số các thao tác phát sinh từ mouse
  - Di chuyển
  - Kích chuột
- Ứng dụng cần xử lý sự kiện chuột nào sẽ khai báo trình xử lý tương ứng
- Lớp **MouseEventArgs** được sử dụng để chứa thông tin truyền vào cho trình xử lý sự kiện mouse.
- Mỗi trình xử lý sự kiện sẽ có tham số là đối tượng object và đối tượng MouseEventArgs (hoặc EventArgs)

# Mouse Event

- Tham số cho sự kiện liên quan đến mouse



# Mouse Event

## ➤ Các sự kiện thường dùng

- **MouseEnter:** Xảy ra khi đưa con trỏ chuột vào vùng của đối tượng.
- **MouseLeave:** Xảy ra khi đưa con trỏ chuột ra khỏi vùng của đối tượng.
- **MouseDown:** Xảy ra khi ấn nút chuột trong khi con trỏ chuột đang nằm trong vùng của đối tượng.
- **MouseUp:** Xảy ra khi thả nút chuột trong khi con trỏ chuột đang nằm trong vùng của đối tượng.
- **MouseMove:** Xảy ra khi di chuyển con trỏ chuột trong vùng của đối tượng.

# Mouse Event

---

- *Sự kiện chuột với tham số kiểu EventArgs*
  - **MouseEnter**
  - **MouseLeave**
- *Sự kiện chuột với tham số kiểu MouseEventArgs*
  - **MouseDown**
  - **MouseUp**
  - **MouseMove**

# Mouse Event

## ➤ Thuộc tính của lớp MouseEventArgs

- Button: Button được nhấn {Left, Right, Middle, none} có kiểu là MouseButtons
- Clicks: Số lần button được nhấn
- X: Tọa độ x của con trỏ chuột trong control
- Y: Tọa độ y của con trỏ chuột trong control



# Keyboard Event

---

- Phát sinh khi một phím được nhấn hoặc thả
- Có 3 sự kiện
  - KeyPress
  - KeyUp
  - KeyDown
- KeyPress phát sinh kèm theo mã ASCII của phím được nhấn, không cho biết trạng thái các phím bổ sung {*Shift, Alt, Ctrl...*}
- Sử dụng KeyUp & KeyDown để xác định trạng thái các phím bổ sung.

# Keyboard Event

## ➤ Các sự kiện thường dùng

- **KeyDown:** Xảy ra khi một phím được ấn trên đối tượng.
- **KeyUp:** Xảy ra khi một phím được thả trên đối tượng
  - **KeyEventArgs:** Tham số cho sự kiện **KeyDown** và **KeyUp**.
- **KeyPress:** Xảy ra khi ấn và thả một phím trên đối tượng.
  - **KeyPressEventArgs:** Tham số cho sự kiện **KeyPress**

# Keyboard Event

## ➤ Tham số KeyPressEventArg

- Là tham số của sự kiện KeyPress
- **KeyChar**: Trả về ký tự của phím được ấn
- **Alt**: Có/không phím Alt đã được ấn
- **Control**: Có/không phím Control đã được ấn.

## ➤ Tham số KeyEventArgs

- Là tham số của các sự kiện **KeyDown** và **KeyUp**.
- **Shift**: Có hay không phím Shift đã được ấn.
- **KeyCode**: Trả về phím được ấn.
- **KeyValue**: Trả về mã của phím được ấn

# Lớp MessageBox

- Dùng để hiển thị hộp thoại với tiêu đề, các loại nút và biểu tượng khác nhau.
- Dùng để hiển thị một thông báo
- Dùng để xác nhận một hành động



# Lớp MessageBox

---

- Dùng phương thức **Show** để trình bày hộp thoại
- Có nhiều hình thức của phương thức Show tùy thuộc vào tham số: **title**, **message**, **buttons** (MessageBoxButtons Enum), **icons** (MessageBoxIcon Enum)

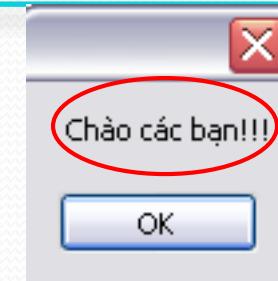
# Lớp MessageBox

- Hằng **MessageBoxButtons**: AbortRetryIgnore, OK, OKCancel,...
- Hằng **MessageBoxIcons**: Error, Information, Question, Warning,...
- Nút được chọn mặc định tùy vào **MessageBoxDefaultButton**: button1, button2, button3.
- Phương thức **Show** trả về giá trị là một trong các Enum của **DialogResult**: OK, Cancel, Yes,...

# Một số hình thức của Show

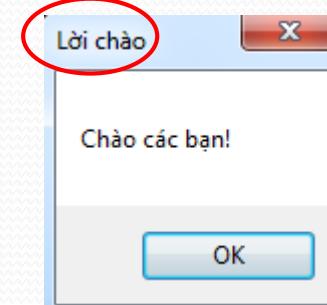
## ➤ MessageBox.Show(String)

```
MessageBox.Show("Chào các bạn!!!");
```



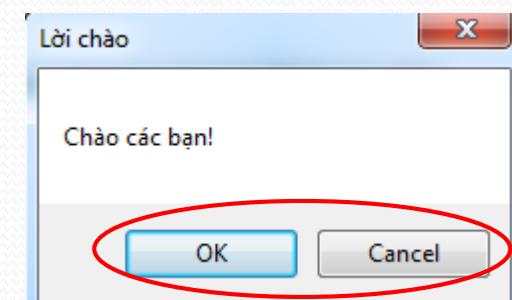
## ➤ MessageBox.Show(String, String)

```
MessageBox.Show("Chào các bạn!",  
    "Lời chào"));
```



## ➤ MessageBox.Show(String, String, MBB)

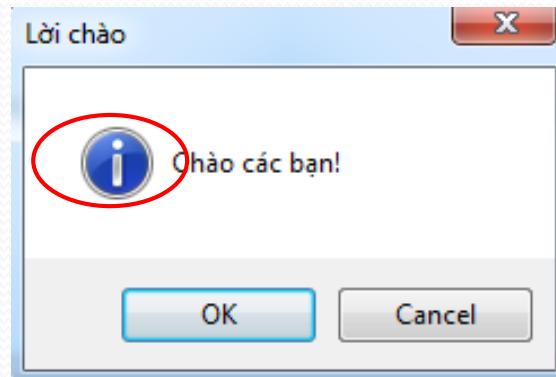
```
MessageBox.Show("Chào các bạn!", "Lời chào",  
    MessageBoxButtons.OKCancel);
```



# Một số hình thức của Show

- MessageBox.Show(String, String, MBB, MBI)

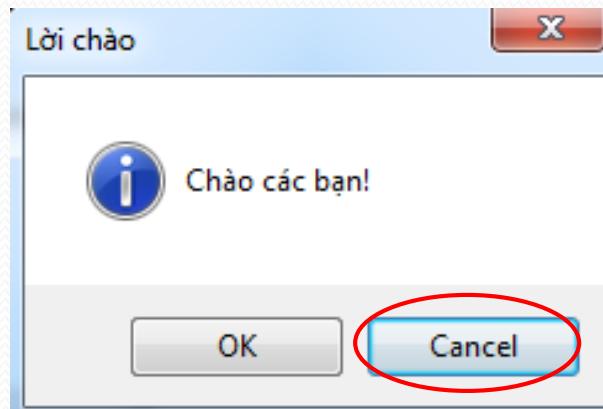
```
MessageBox.Show("Chào các bạn!","Lời chào",
MessageBoxButtons.OKCancel, MessageBoxIcon.Information);
```



# Một số hình thức của Show

- MessageBox.Show(String, String, MBB, MBI, DB)

MessageBox.Show("Chào các bạn!", "Lời chào",  
MessageBoxButtons.OKCancel, MessageBoxIcon.Information,  
MessageBoxDefaultButton.Button2);



# Giá trị trả về của Show



```
private void button1_Click(object sender, EventArgs e)
{
    if (MessageBox.Show("Bạn chắc chắn đóng Form?", "Thông báo",
        MessageBoxButtons.YesNo, MessageBoxIcon.Warning,
        MessageBoxDefaultButton.Button1) == DialogResult.Yes)
    { this.Close(); }
}
```

# Kết thúc chủ đề 7

---

