



Windows Form

Giảng viên: BÙI NGỌC LÊ

Nội Dung

- Graphical User Interface (GUI)
- Event Driven Programming
- Ứng dụng Windows Form dùng C#
- Khuôn mẫu của ứng dụng Windows Form chuẩn
- Cách tạo ứng dụng Windows Form trong VS 2010
 - Tạo ứng dụng Form
 - Chỉnh sửa form
 - Thêm component vào form
 - Viết phần xử lý cơ bản

GUI

Command line interface: CLI

```
--- rr.chtpa.wikimedia.org ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ns
rtt min/avg/max/ndev = 112.076/112.076/112.076/0.000 ns
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

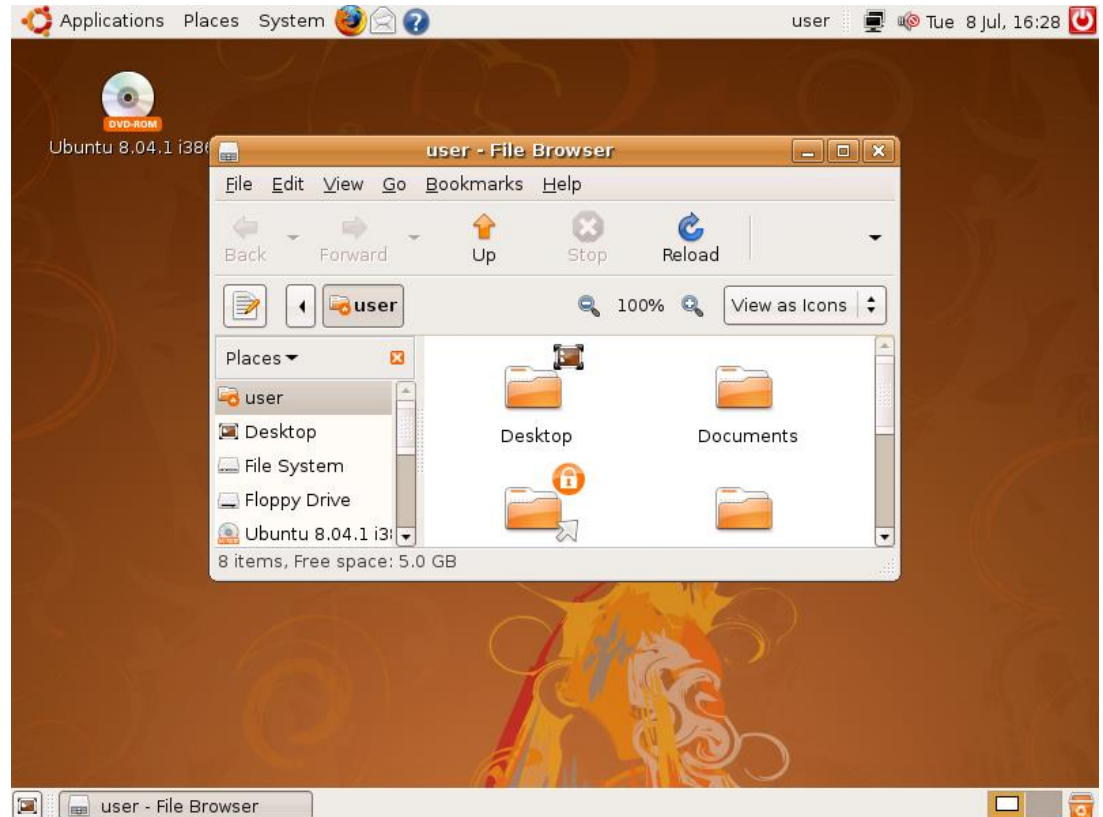
GUI

Graphical User Interface: GUI

**Tương tác qua giao
diện đồ họa độ
phân giải cao**

**Đa số các hệ OS hiện
đại đều dùng GUI**

**Cho phép user dễ dàng
thao tác**



GUIs

- Chương trình hiện đại đều dùng GUI
- Graphical: text, window, menu, button...
- User: người sử dụng chương trình
- Interface: cách tương tác chương trình

- Thành phần đồ họa điển hình
 - Window: một vùng bên trong màn hình chính
 - Menu: liệt kê những chức năng
 - Button: nút lệnh cho phép click vào
 - TextBox: cho phép user nhập dữ liệu text

GUI Application

- **Windows Form là nền tảng GUI cho ứng dụng desktop**
 - (Ngược với Web Form ứng dụng cho Web)
 - Single Document Interface (SDI)
 - Multiple Document Interface (MDI)
- **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

Event– Driven Programming

Cách truyền thống

Danh sách các lệnh thực thi tuần tự

Việc kế tiếp xảy ra chính là lệnh tiếp theo trong danh sách

Chương trình được thực thi bởi máy tính

Event-Driven Programming

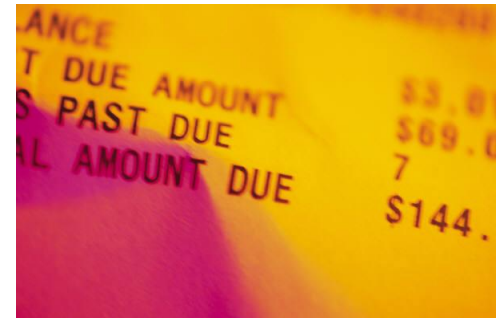
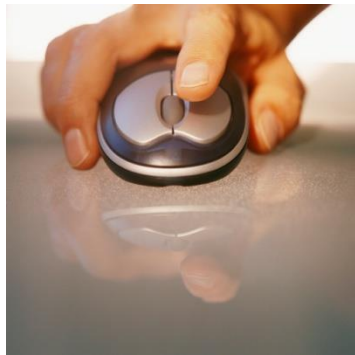
Các đối tượng có thể kích hoạt sự kiện và các đối tượng khác phản ứng với những sự kiện đó

Việc kế tiếp xảy ra phụ thuộc vào sự kiện kế tiếp

Luồng chương trình được điều khiển bởi sự tương tác User-Computer

Event-Driven Programming

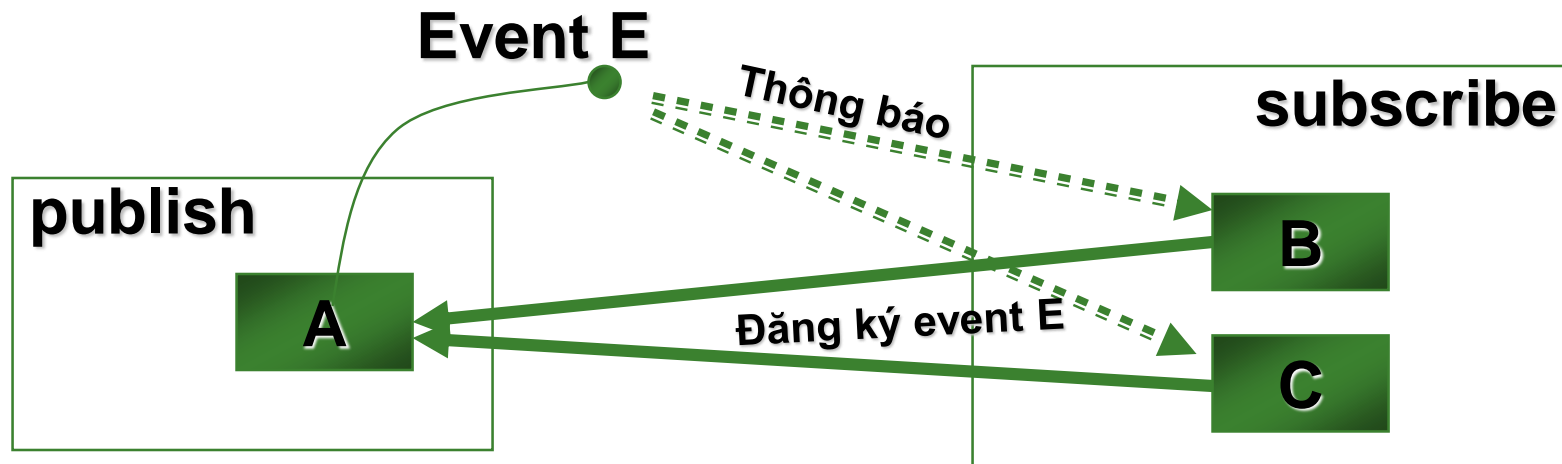
- Chương trình GUI thường dùng Event-Drive Programming
- Chương trình chờ cho event xuất hiện và xử lý
- Ví dụ sự kiện:



- 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

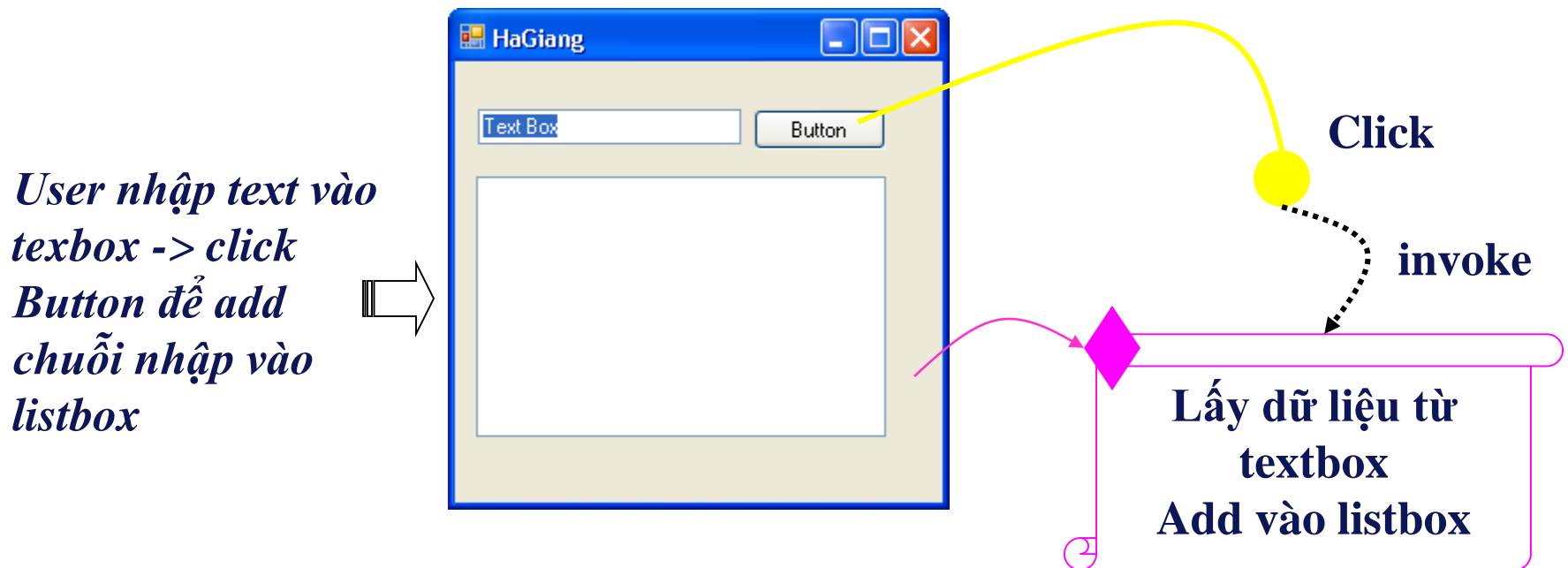
Event-Driven Programming

- Trong C#, Event-Driven Programming được thực thi bởi event (*xem slide Delegate & Event*)



Event-Driven Programming

■ Minh họa xử lý trong form

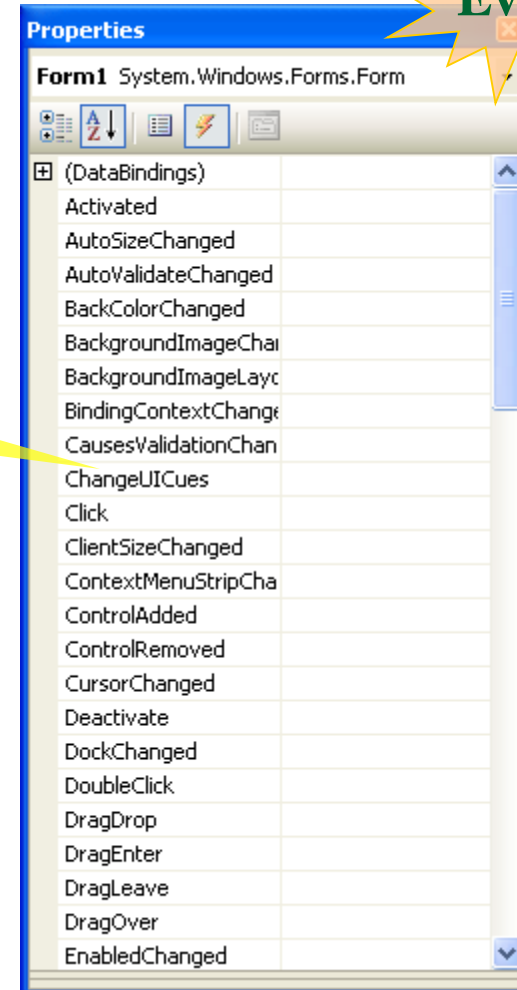


Button đưa ra sự kiện click
Form có event handler cho click của button

Event-Driven Programming

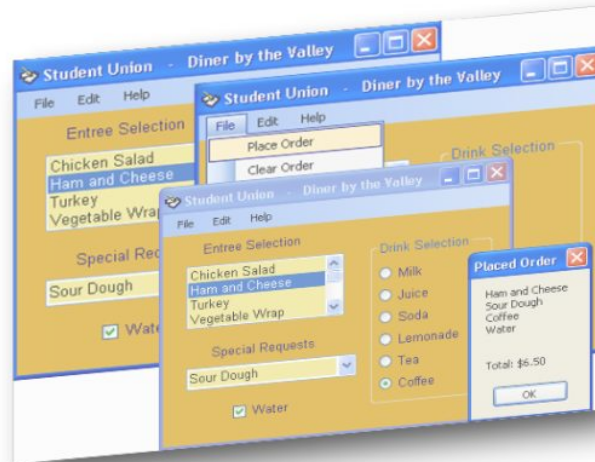
■ GUI-based events

- ❑ Mouse move
- ❑ Mouse click
- ❑ Mouse double-click
- ❑ Key press
- ❑ Button click
- ❑ Menu selection
- ❑ Change in focus
- ❑ Window activation
- ❑ ...



Event

Danh sách
event cho
Form



Windows Forms Application

Windows Form App

- Sử dụng GUI làm nền tảng
- Event-driven programming cho các đối tượng trên form
- Ứng dụng dựa trên một “form” chứa các thành phần
 - Menu
 - Toolbar
 - StatusBar
 - TextBox, Label, Button...
- Lớp cơ sở cho các form của ứng dụng là Form

System.Windows.Forms. Form

Namespace

Class

Minh họa WinForm App

The screenshot displays a Windows Forms application titled "Employees". The interface includes a top navigation bar with icons for "Manage Time Off", "Manage Wages", "Send Email", "Print Envelope", "Print Address Book", and "Print Employment". The main form is divided into several sections:

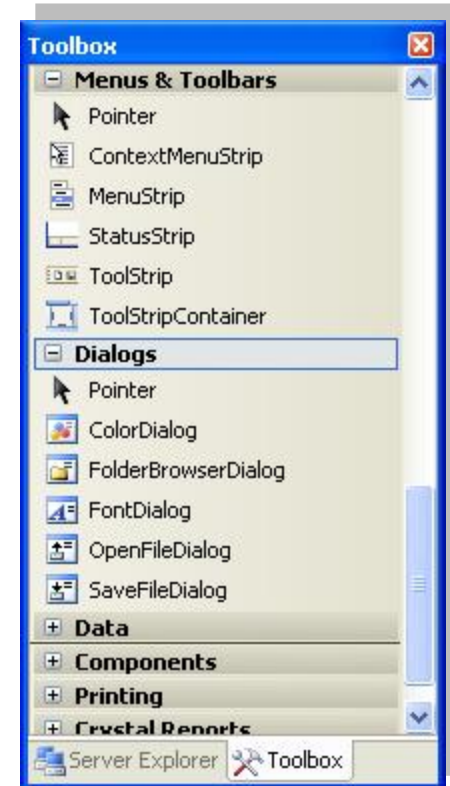
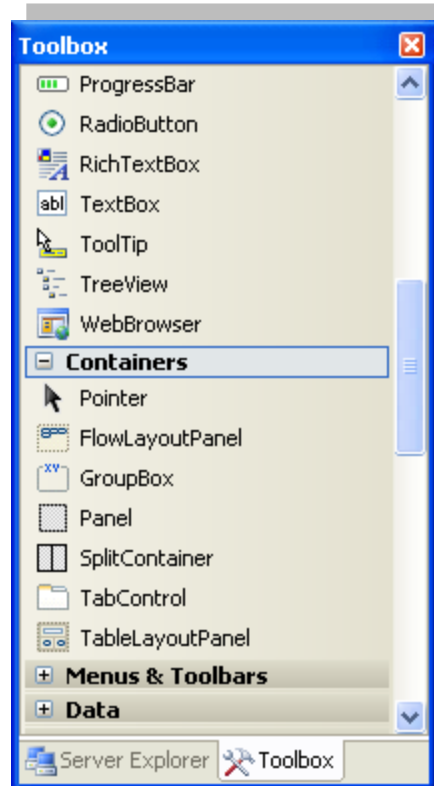
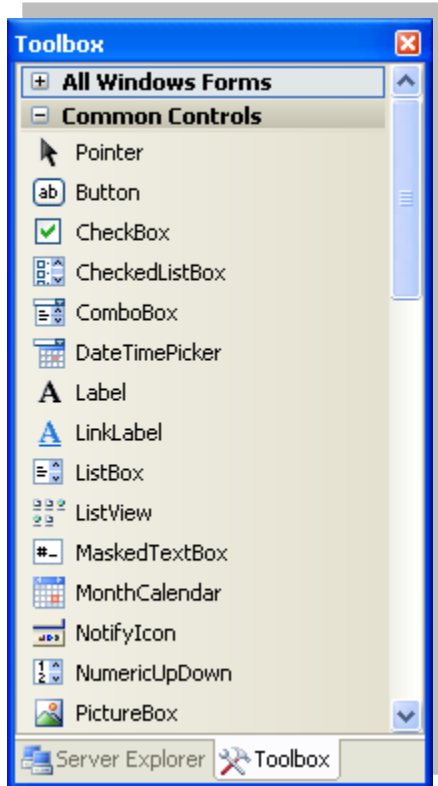
- Employee Info:** Fields for First Name (Shirley Ann), Last Name (Dawson), Gender (Female), Position (Secretary), DOB (18 March 1958), Age (51 yrs 10 months), Date Started (10 Apr 2008), Service (1 yrs 3 months), and Date Left.
- Address:** Fields for Address (1 Oak View), City (Basingstoke), Post Code (RG12 0JY), and County (East Yorkshire).
- Contact Info:** Fields for Phone ((01185) 188118) and Email (someone@winformapp.co.uk).
- Holiday Entitlement:** A field for Days Per Year (20).
- Notes:** A large text area for additional information.

At the bottom, there are navigation buttons: "New", "List All", "First", "Previous", "Next", "Last", "Help", "Print", "Delete", "Add", "New", and "Home".

GUI Components/Controls

- Components/controls được tổ chức vào các lớp thừa kế, cho phép dễ dàng chia sẻ các thuộc tính
- Mỗi component/control định nghĩa các
 - Thuộc tính
 - Phương thức
 - Sự kiện
- Cách dễ nhất là sử dụng VS .NET Toolbox để thêm control và component vào form

Components and Controls cho Windows Form



Toolbox của Visual Studio .NET 2005

Ứng Dụng WinForm đơn giản

Form1.cs

```
using System;
using System.Windows.Forms;
using System.Drawing;
```

Lớp Form cơ sở

```
namespace HaGiang
{
```

```
    public class Form1 : Form
    {
```

```
        Label title;
        public Form1()
        {
```

Control kiểu Label

```
            this.Text = "Hello World!";
            this.Size = new Size(400, 200);
            title = new Label();
            title.Text = "Hello World";
            title.Location = new Point(50, 50);
            this.Controls.Add(title);
```

Thiết kế form & control

Add control vào form

```
        public static void Main(string[] argv)
        {
            Application.Run(new Form1());
        }
    }
```

Chạy ứng dụng với
Form1 làm form chính

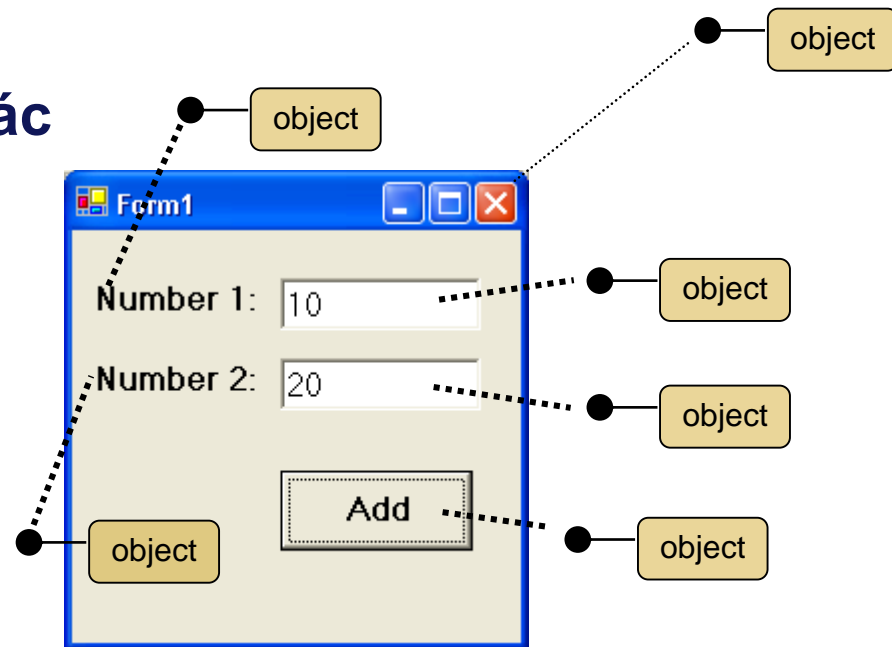
Các bước tạo ứng dụng WinForm cơ bản

- Tạo lớp kế thừa từ lớp Form cơ sở
- Bổ sung các control vào form
 - Thêm các label, menu, button, textbox...
- Thiết kế layout cho form (bố trí control)
 - Hiệu chỉnh kích thước, trình bày, giao diện cho
 - form
 - Control chứa trong form
- Viết các xử lý cho các control trên form và các xử lý khác
- Hiển thị Form
 - Thông qua lớp Application gọi phương thức Run

Nên sử dụng IDE hỗ trợ thiết kế GUI!

Form và control

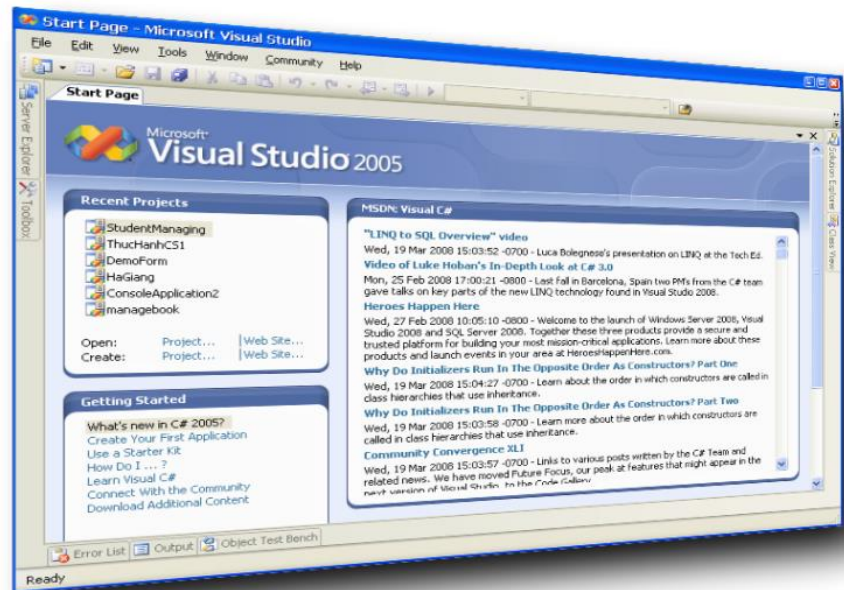
- Tất cả các thành phần trên form đều là đối tượng
- Các class control
 - `System.Windows.Forms.Label`
 - `System.Windows.Forms.TextBox`
 - `System.Windows.Forms.Button`
 - ...
- Các control là instance của các lớp trên.



Các thuộc tính của Form

Property	Description	Default
Name	Tên của form sử dụng trong project	Form1,Form2...
AcceptButton	Thiết lập button là click khi user nhấn Enter	
CancelButton	Thiết lập button là click khi user nhấn Esc	
ControlBox	Hiển thị control box trong caption bar	True
FormBorderStyle	Biên của form: none, single, 3D, sizable	Sizable
StartPosition	Xác định vị trí xuất hiện của form trên màn hình	WindowsDefaultLocation
Text	Nội dung hiển thị trên title bar	Form1, Form2, Form3
Font	Font cho form và mặc định cho các control	
Method	Description	
Close	Đóng form và free resource	
Hide	Ẩn form	
Show	Hiển thị form đang ẩn	
Event	Description	
Load	Xuất hiện trước khi form show	

Minh họa tạo ứng dụng Windows Form từ Visual Studio .NET



Tạo WinForm App từ VS. 2005

Cơ chế xử lý sự kiện code behind

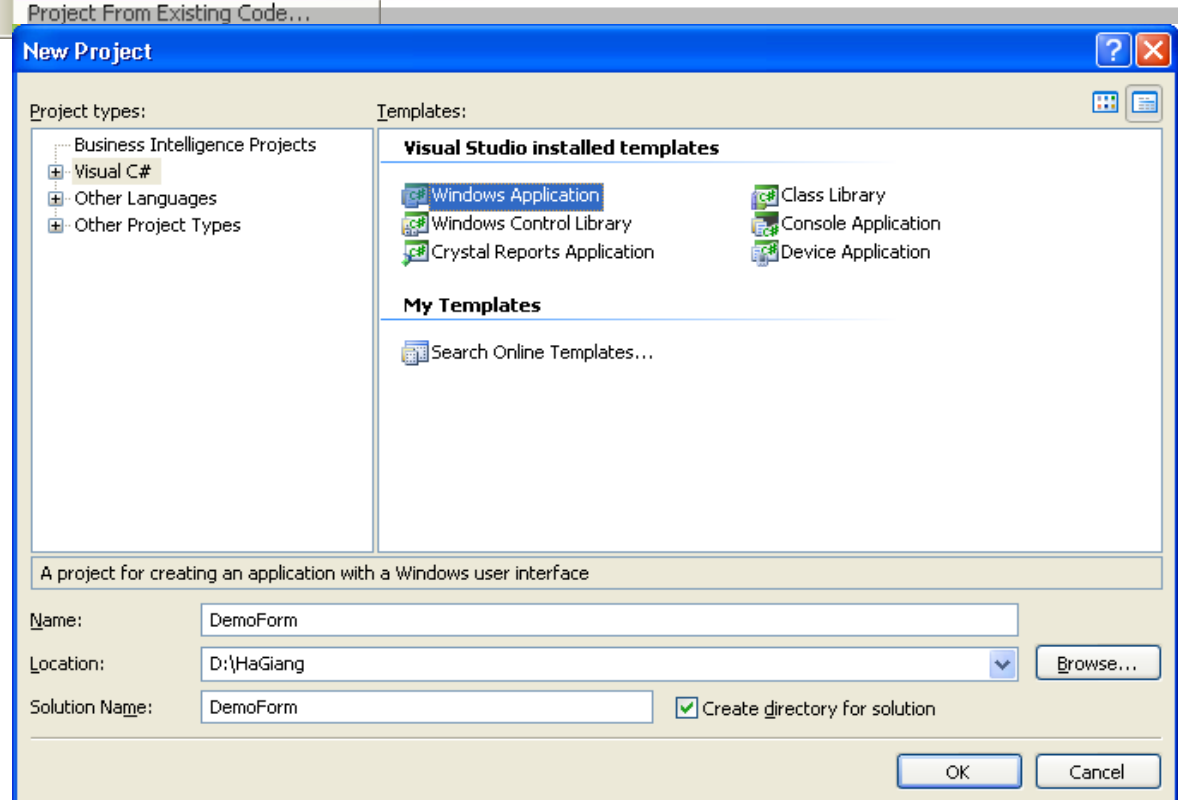
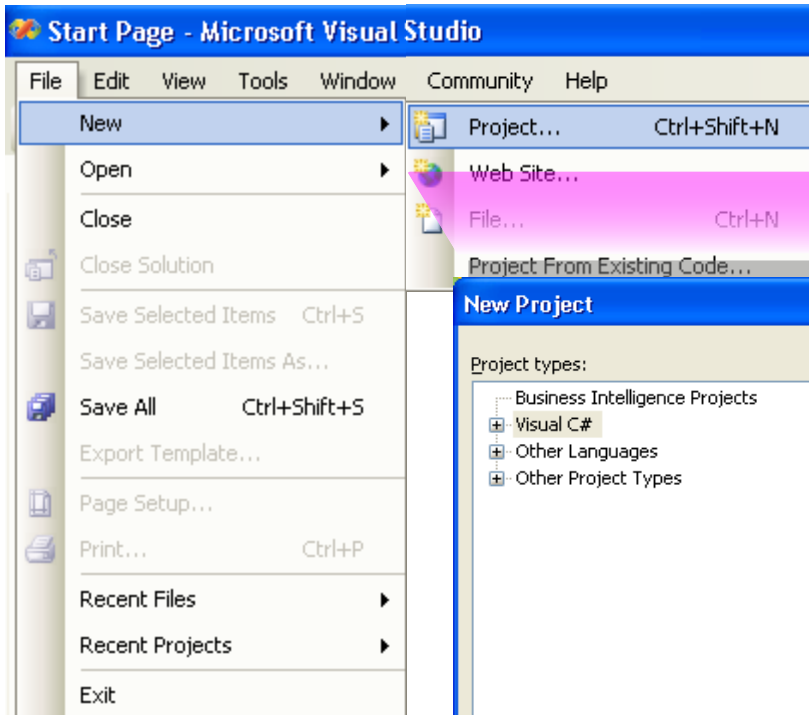
Hỗ trợ WYSISYG cho GUI design



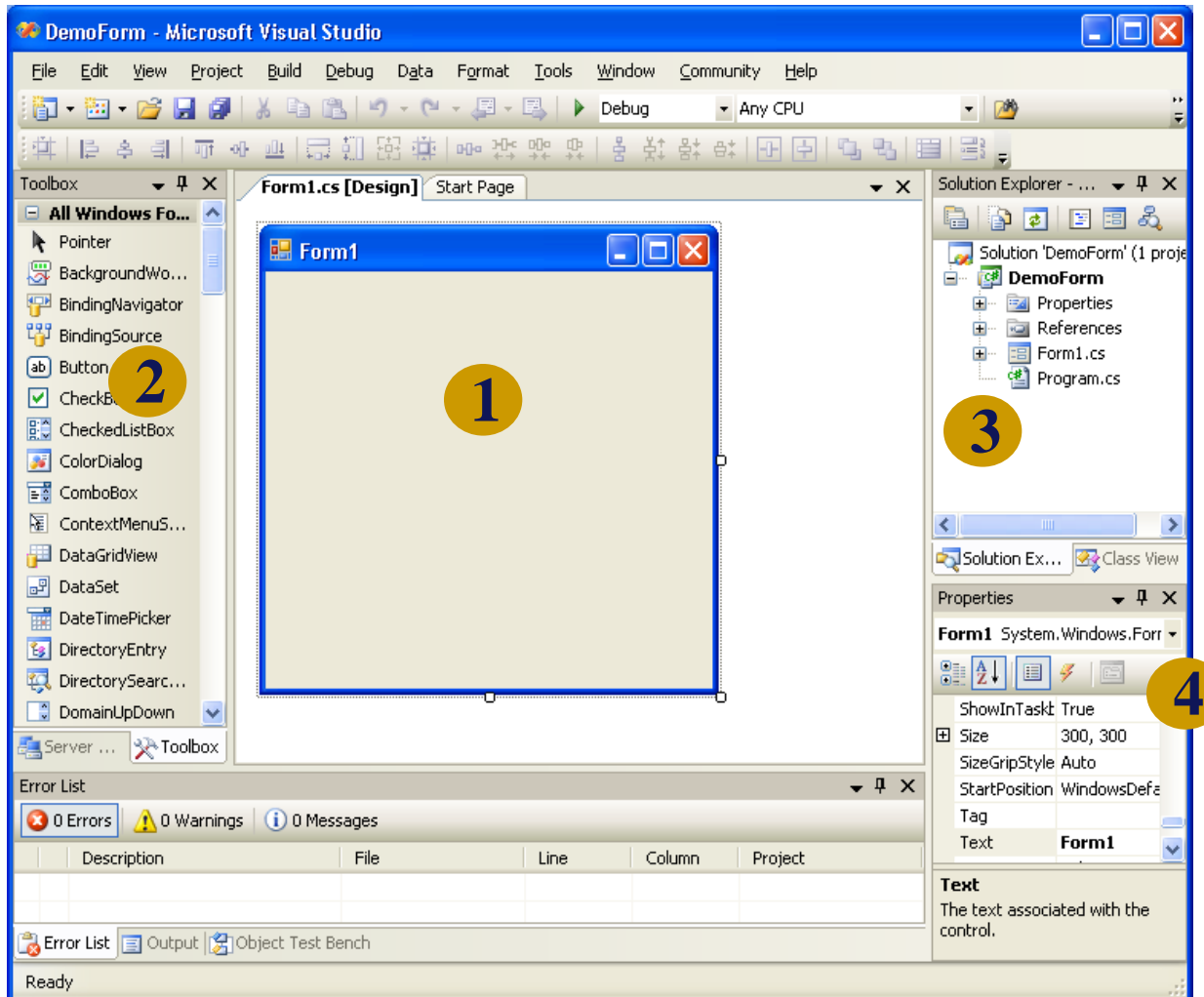
Nhanh chóng & dễ dàng tạo UD Windows Form

Tạo WinForm App từ VS. 2005 (2)

Tạo project: Windows App



Tạo WinForm App từ VS. 2005 (3)



**Windows App do
VS.2005 khởi tạo**

- 1: form ứng dụng**
- 2: control toolbox**
- 3: Solution Explorer**
- 4: Form properties**

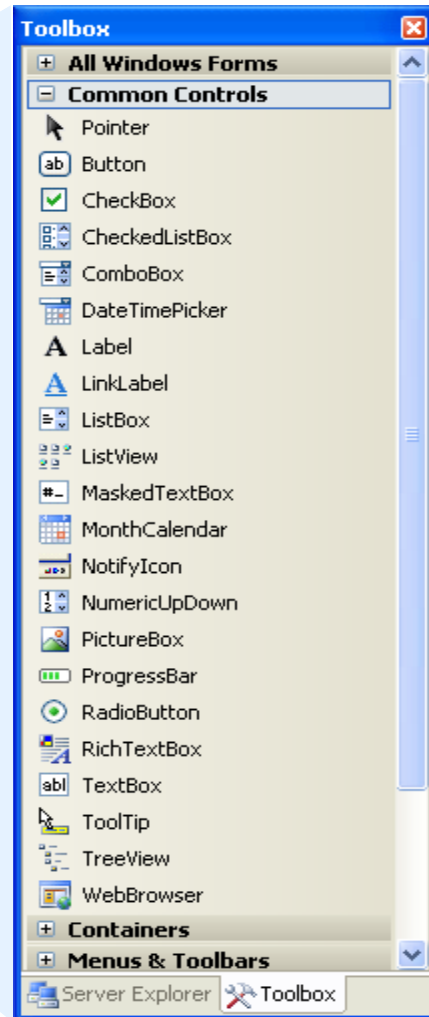
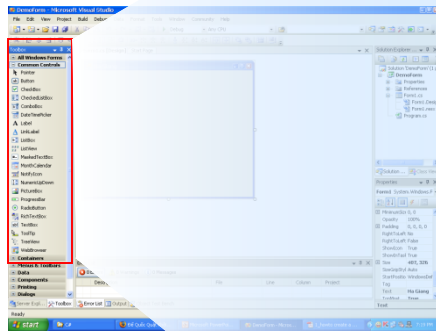
Tạo WinForm App từ VS. 2005 (4)

- Màn hình thiết kế Form, cho phép người lập trình kéo thả những control vào trong form
 - Tất cả những code được tạo tự động dựa trên sự thao tác thiết kế form của user
 - Rút ngắn nhiều thời gian cho việc thao tác giao diện form
 - Tính năng trực quan WYSIWYG



**Có được ứng dụng form
mặc dù chưa viết code!**

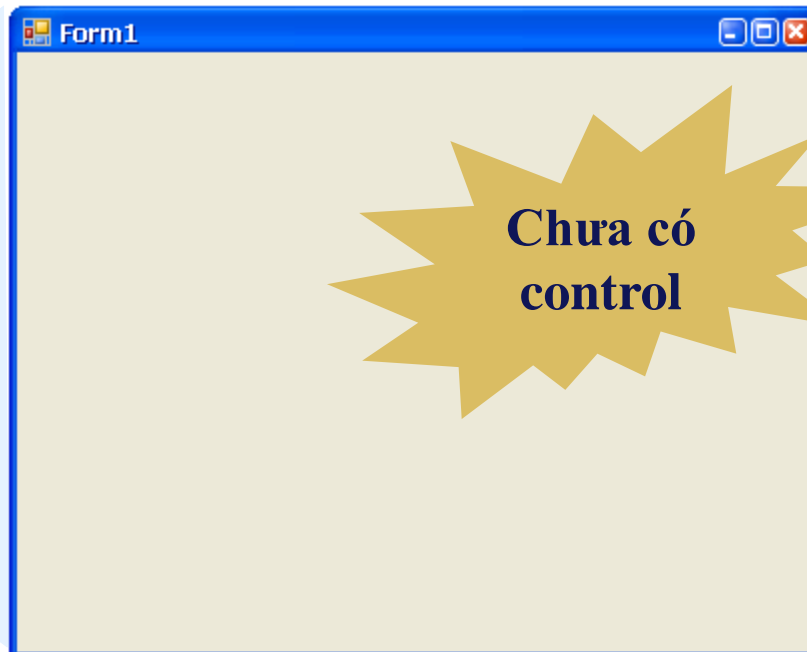
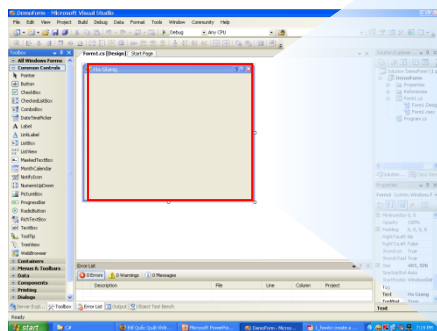
Toolbox



Toolbox

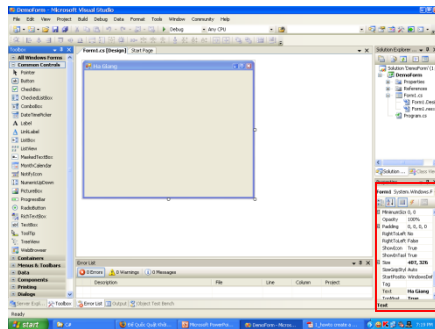
- Kéo thả control lên form
- Code được phát sinh tự động

Giao diện thiết kế form



Form chính của ứng dụng

Cửa sổ properties



Properties

Form1 System.Windows.Forms.Form

(ApplicationSetting)	
(DataBindings)	
(Name)	Form1
AcceptButton	(none)
AccessibleDescripti	
AccessibleName	
AccessibleRole	Default
AllowDrop	False
AutoScaleMode	Font
AutoScroll	False
AutoScrollMargin	0, 0
AutoScrollMinSize	0, 0
AutoSize	False
AutoSizeMode	GrowOnly
AutoValidate	EnablePreventFocu
BackColor	Control
BackgroundImage	(none)
BackgroundImageL	Tile
CancelButton	(none)
CausesValidation	True
ContextMenuStrip	(none)

Cửa sổ properties của form

Cửa sổ properties

Tên của form chính là tên lớp

Để dàng hiệu chỉnh form thông qua cửa sổ Properties

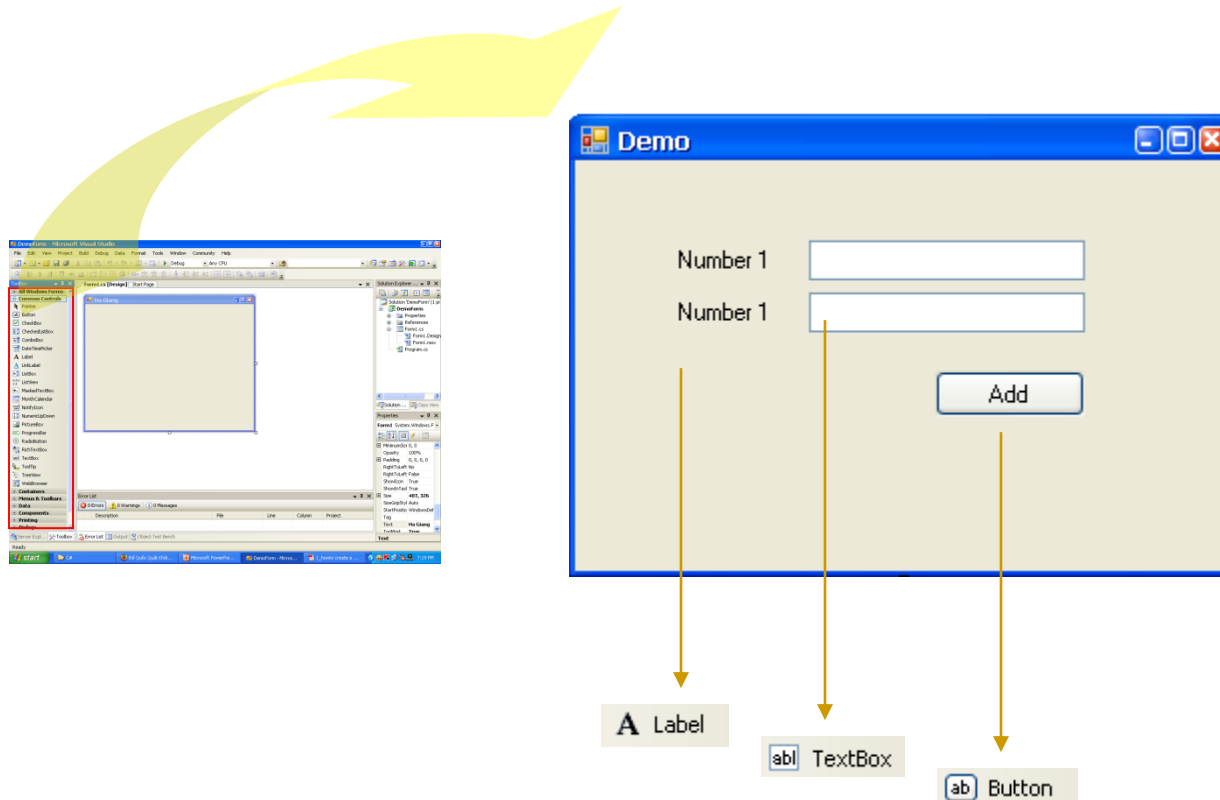
Thay đổi title

Form1 System.Windows.Forms.Form	
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	407, 333
SizeGripStyle	Auto
StartPosition	WindowsDefaultLoca
Tag	
Text	Ha Giang
TopMost	True
TransparencyKey	<input type="checkbox"/>
UseWaitCursor	False
WindowState	Normal

Text
The text associated with the control.

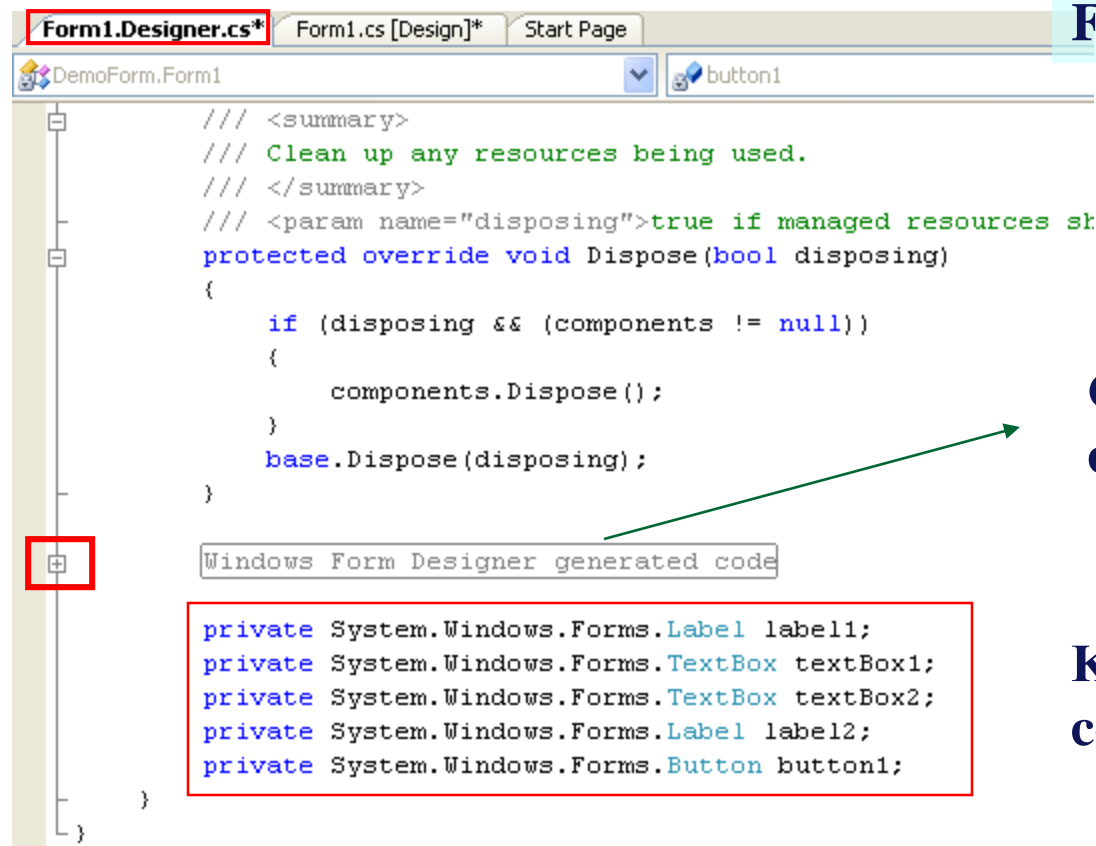
Thêm control vào form

- Kéo thả control vào form



Code của phần design

- Phần code thiết kế Form1 được tạo tự động

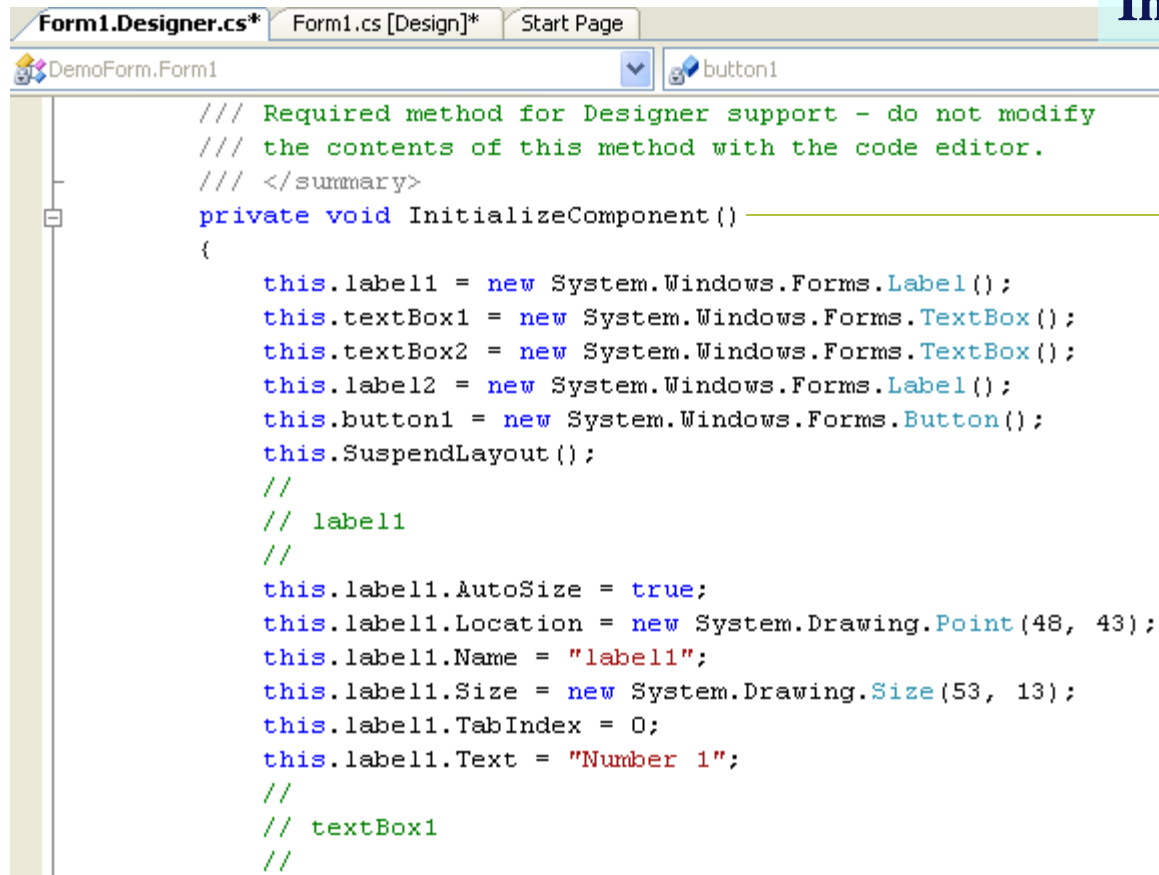


Form1.Designer.cs

Chứa code khởi tạo control

Khai báo các đối tượng control trên Form1

Code của phần design



```
Form1.Designer.cs*  Form1.cs [Design]*  Start Page
DemoForm.Form1
button1

/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.label1 = new System.Windows.Forms.Label();
    this.textBox1 = new System.Windows.Forms.TextBox();
    this.textBox2 = new System.Windows.Forms.TextBox();
    this.label2 = new System.Windows.Forms.Label();
    this.button1 = new System.Windows.Forms.Button();
    this.SuspendLayout();
    //
    // label1
    //
    this.label1.AutoSize = true;
    this.label1.Location = new System.Drawing.Point(48, 43);
    this.label1.Name = "label1";
    this.label1.Size = new System.Drawing.Size(53, 13);
    this.label1.TabIndex = 0;
    this.label1.Text = "Number 1";
    //
    // textBox1
    //
```

InitializeComponent

Tạo đối tượng

**Lần lượt khai
báo các thuộc
tính cho các
control**

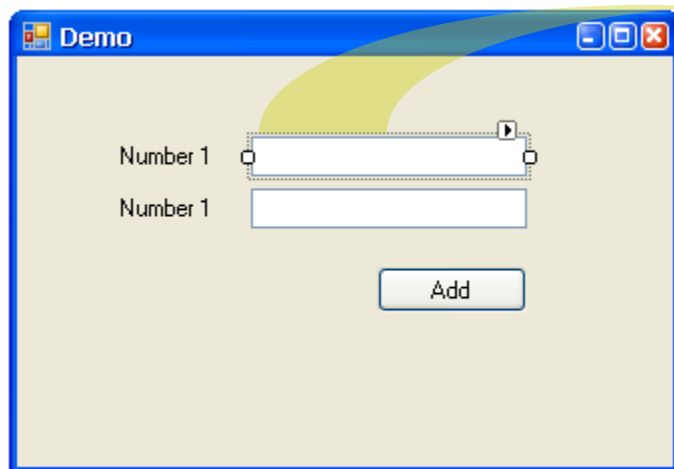
Code của phần design

InitializeComponent

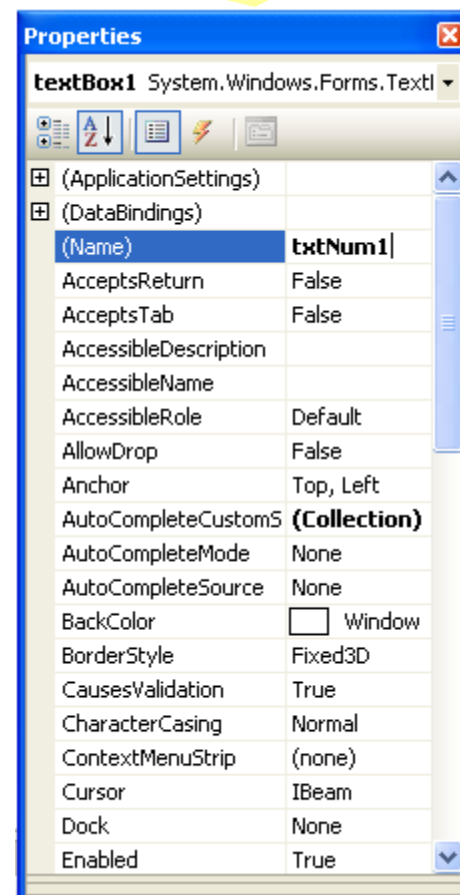
```
//  
// Form1  
//  
this.AutoScaleMode = new System.Drawing.SizeF(6F, 13F);  
this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;  
this.ClientSize = new System.Drawing.Size(328, 205);  
this.Controls.Add(this.BtnAdd);  
this.Controls.Add(this.txtNum2);  
this.Controls.Add(this.label2);  
this.Controls.Add(this.txtNum1);  
this.Controls.Add(this.label1);  
this.Name = "Form1";  
this.Text = "Demo";  
this.TopMost = true;  
this.ResumeLayout(false);  
this.PerformLayout();
```

Đưa các control vào danh sách control của Form1

Sửa thuộc tính của control



Thay đổi các giá trị qua cửa sổ
properties -> VS tự cập nhật
code



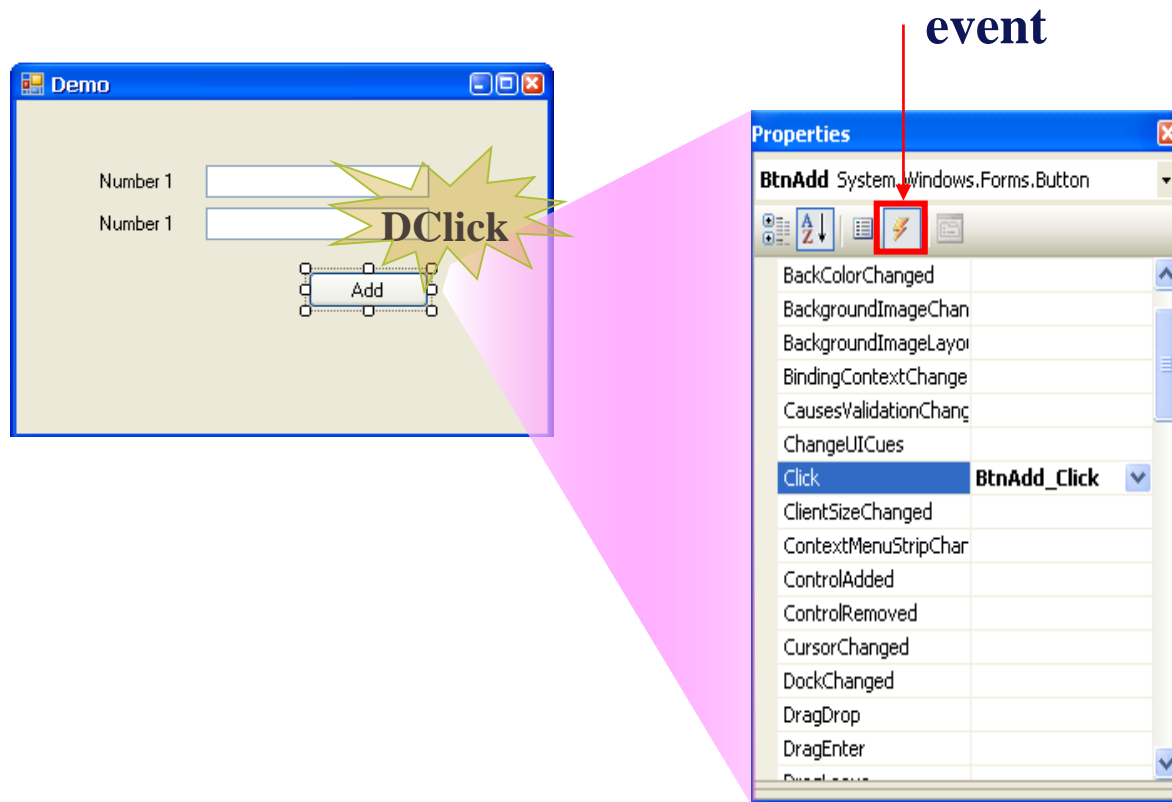
Đổi tên thành
`txtNum1`

Phần xử lý

- Khi click vào Add -> cộng 2 giá trị và xuất kết quả
- Thực hiện
 - Button Add cung cấp sự kiện click
 - Form sẽ được cảnh báo khi Add được click
 - Form sẽ lấy dữ liệu từ 2 textbox và cộng -> kết quả
- Cơ chế event
 - Button đưa ra sự kiện click: đối tượng publish
 - Form quan tâm đến sự kiện click của button, Form có sẽ phần xử lý ngay khi button click.
 - Phần xử lý của form gọi là Event Handler
 - Form đóng vai trò là lớp subscribe

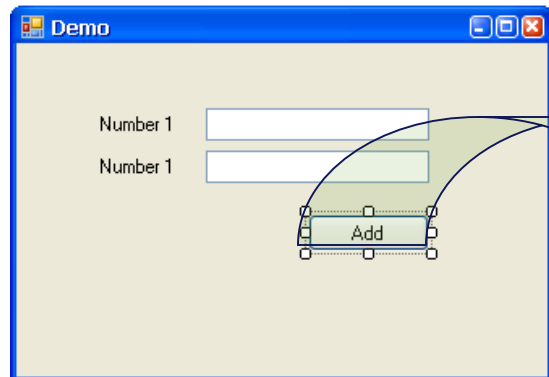
Khai báo event handler

- Kích đúp vào button Add trên màn hình thiết kế cho phép tạo event handler cho sự kiện này.



Cửa sổ quản lý
event của BtnAdd

Khai báo event handler



**Event handler cho
button Add**

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

    private void BtnAdd_Click(object sender, EventArgs e)
    {
        |
    }
}
```

Cùng signature method với System.EventHandler

Khai báo event handler

InitializeComponent

```
//  
// BtnAdd  
//  
this.BtnAdd.Location = new System.Drawing.Point(180, 105);  
this.BtnAdd.Name = "BtnAdd";  
this.BtnAdd.Size = new System.Drawing.Size(75, 23);  
this.BtnAdd.TabIndex = 4;  
this.BtnAdd.Text = "Add";  
this.BtnAdd.UseVisualStyleBackColor = true;  
this.BtnAdd.Click += new System.EventHandler(this.BtnAdd_Click);
```

Sự kiện click

Trình xử lý được gọi
khi event xảy ra

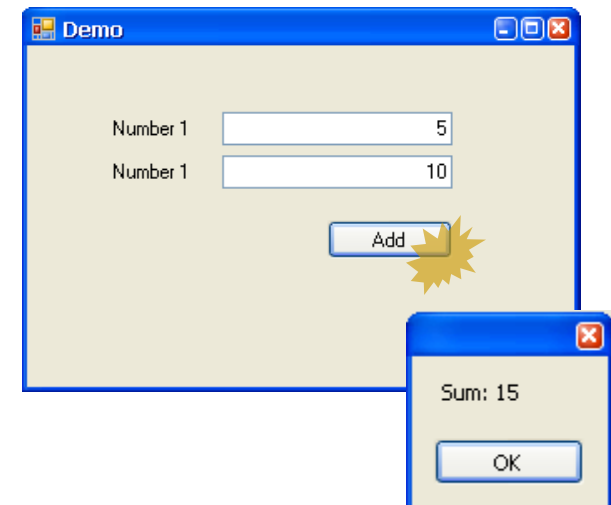
Delegate chuẩn cho event handler

Viết phần xử lý

- Phần xử lý của Form1 khi button click
 - Lấy giá trị của 2 textbox, cộng kết quả và xuất ra MeesageBox

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

    private void BtnAdd_Click(object sender, EventArgs e)
    {
        int sum;
        sum = int.Parse(txtNum1.Text) + int.Parse(txtNum2.Text);
        MessageBox.Show("Sum: " + sum.ToString());
    }
}
```



Phương thức của lớp Form

- **Các hành động có thể thực hiện trên form**
 - ❑ **Activate:** cho form nhận focus
 - ❑ **Close:** đóng và giải phóng resource
 - ❑ **Hide:** ẩn form
 - ❑ **Refresh:** tô vẽ lại
 - ❑ **Show:** cho form show ra màn hình (modeless) và activate
 - ❑ **ShowDialog:** hiển thị dạng modal
 - ❑ **Find Dialog** chính là dạng modeless
 - ❑ **Font dialog** dạng modal

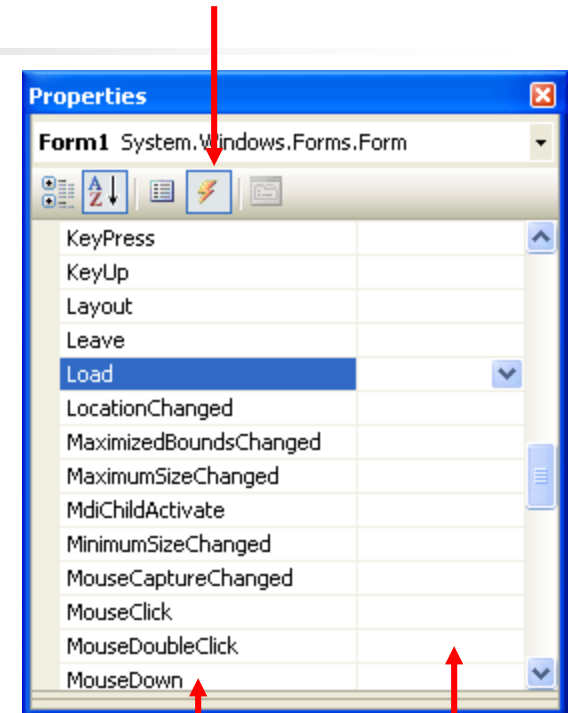
Event của Form

■ Tạo xử lý cho event

- ❑ Trong cửa sổ properties
- ❑ Chọn biểu tượng event
- ❑ Kích đúp vào tên event

■ Event thường dùng

- ❑ **Load**: xuất hiện trước khi form xuất hiện lần đầu tiên
- ❑ **Closing**: xuất hiện khi form đang chuẩn bị đóng
- ❑ **Closed**: xuất hiện khi form đã đóng
- ❑ **Resize**: xuất hiện sau khi user resize form
- ❑ **Click**: xuất hiện khi user click lên nền form
- ❑ **KeyPress**: xuất hiện khi form có focus và user nhấn phím



Tên event

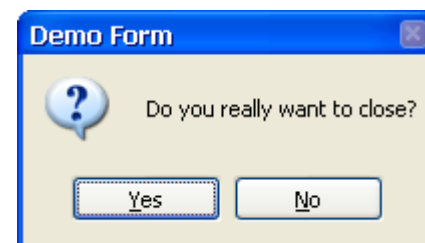
Trình xử lý
nếu có

Event của Form

- Ví dụ chương trình sẽ hỏi user xác nhận trước khi đóng ứng dụng.
 - Kích đúp vào item FormClosing trong cửa sổ event
 - Hàm Form1_FormClosing được tạo và gắn với sự kiện FormClosing
 - Viết code cho event handler Form1_FormClosing

this.FormClosing += new FormClosingEventHandler(this.Form1_FormClosing);

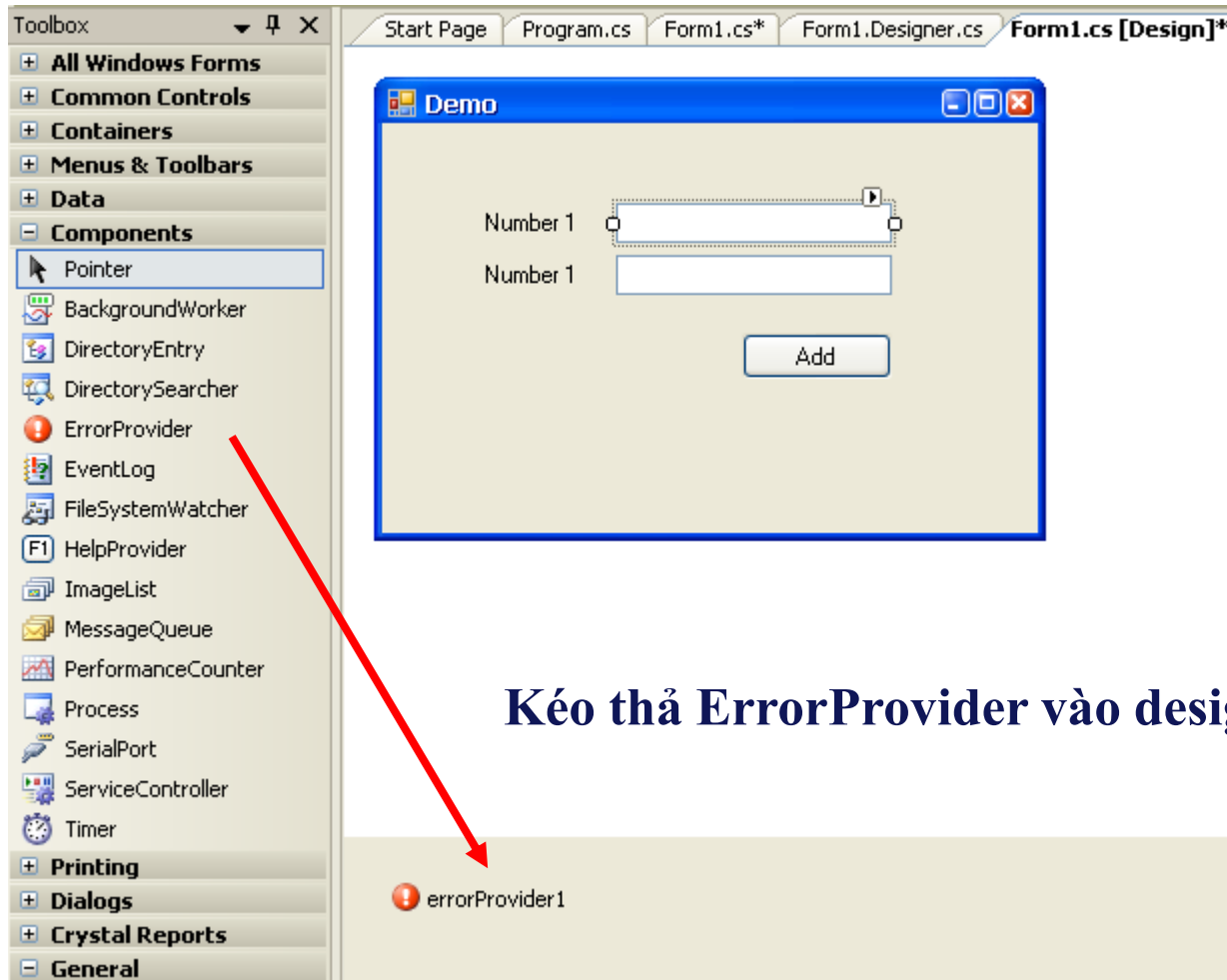
```
private void Form1_FormClosing(object sender, FormClosingEventArgs e)
{
    DialogResult r;
    r = MessageBox.Show("Do you really want to close?", "Demo Form",
        MessageBoxButtons.YesNo,
        MessageBoxIcon.Question,
        MessageBoxDefaultButton.Button1);
    if (r == DialogResult.No)
        e.Cancel = true;
}
```



Kiểm tra dữ liệu nhập

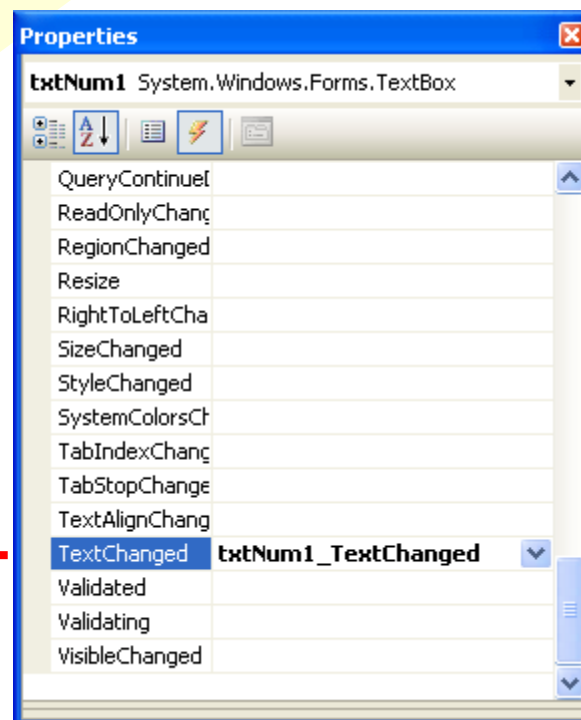
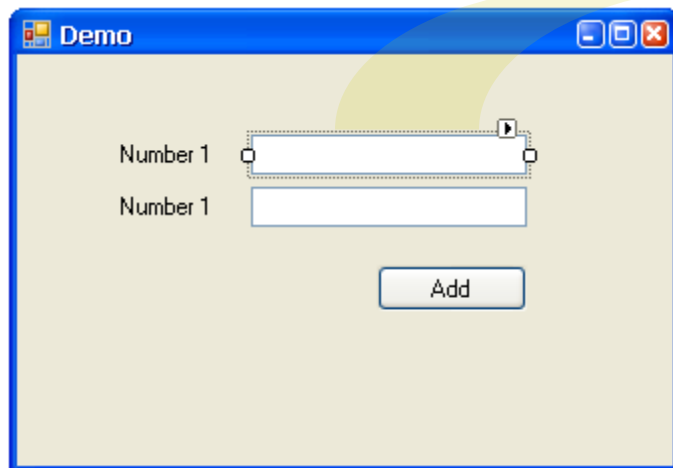
- Nếu user nhập vào chuỗi thì chương trình trên sẽ lỗi!
- Khắc phục:
 - Cảnh báo user nhập không đúng dạng
 - Xóa những ký tự không hợp lệ đó
- Sử dụng control `ErrorProvider` để cảnh báo lỗi khi user nhập không đúng
 - Trong Design View: kéo `ErrorProvider` từ `ToolBox/Component` vào form
 - Chặn xử lý sự kiện `TextChanged` khi user nhập liệu vào `textbox`
 - Nếu nhập sai thiết lập lỗi cho control `ErrorProvider` cảnh báo!

Bổ sung ErrorProvider



Kéo thả ErrorProvider vào design view

Xử lý sự kiện TextChanged của textBox

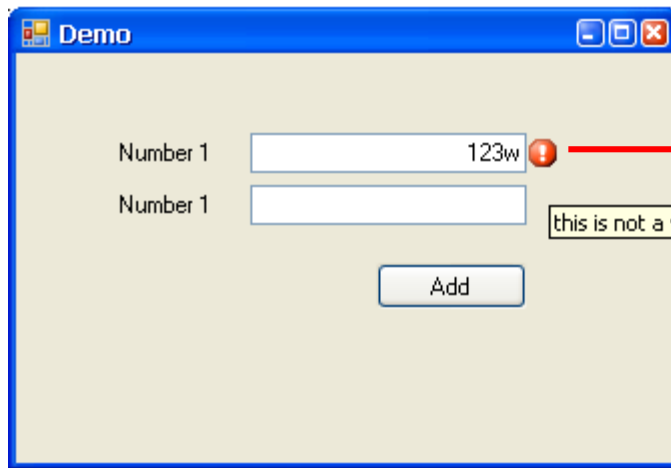


```
private void txtNum1_TextChanged(object sender, EventArgs e)
{
    Control control = (Control)sender;

    if (!Char.IsDigit(control.Text[control.Text.Length-1]))
        this.errorProvider1.SetError(control, "this is not a valid number");
    else
        this.errorProvider1.Clear();
}
```

Phần kiểm tra

ErrorProvider cảnh báo



Icon hiển thị lỗi

Di chuyển chuột vào icon, tooltip xuất hiện

Tóm tắt

- **Tổng quan lập trình GUI**
- **Cơ chế Event Driven Programming**
- **Ứng dụng Windows Form cơ bản**
- **Sử dụng Visual Studio .NET 2005 tạo ứng dụng WF**
 - **Windows Form Application**
 - **Sử dụng control: text, label, button**
 - **Xử lý sự kiện cho button, form**
 - **Sử dụng ErrorProvider**