



CHƯƠNG 6: SỬ DỤNG WINFORM

Phan Trọng Tiến

BM Công nghệ phần mềm

Khoa Công nghệ thông tin, VNUA

Email: phantien84@gmail.com

Website: <http://timoday.edu.vn>



Nội dung chính

1. Tổng quan
2. Tại sao sử dụng Windows Forms
3. Cấu trúc Windows Forms
4. Sử dụng Windows Forms
5. Demo: Thực hiện Windows Forms
6. Sử dụng các Control
7. Demo: Thực thi chức năng kéo và thả
8. Thừa kế của Windows Forms
9. Demo: Sử dụng thừa kế Windows Forms



Mục đích của chương

- ❑ Chương này cung cấp sinh viên kiến thức cần thiết để tạo các ứng dụng Winform.
- ❑ Sau bài này sinh viên có thể:
 - ❑ Mô tả các lợi ích của Windows Forms
 - ❑ Sử dụng các thuộc tính mới và các phương thức của Windows Forms
 - ❑ Viết code cho các sự kiện điều khiển
 - ❑ Sử dụng các control mới và các control cải tiến
 - ❑ Thêm và chỉnh sửa menu
 - ❑ Tạo một form được thừa kế từ form khác



1. Tổng quan

- ❑ Bạn sẽ nắm được các đặc điểm có sẵn trong Windows Forms, cách thay đổi các form và control, và các thuộc tính, các phương thức, các sự kiện.
- ❑ Bạn sẽ học cách tạo một vài dạng hộp thoại chuẩn của Windows.
- ❑ Nắm được tính thừa kế, cho phép bạn sử dụng kỹ thuật lập trình hướng đối tượng vào các form của bạn.



2. Tại sao sử dụng Windows Forms

- ❑ Có tập hợp các control phong phú
- ❑ Nhiều kiểu giao diện
- ❑ Hỗ trợ cải tiến việc in ấn
- ❑ Hỗ trợ cải tiến về đồ họa – GDI+
- ❑ Hỗ trợ khả năng truy cập qua các thuộc tính của control
- ❑ Hỗ trợ thừa kế
- ❑ Các đối tượng có thể cải tiến
- ❑ Thuận lợi cho thiết kế forms

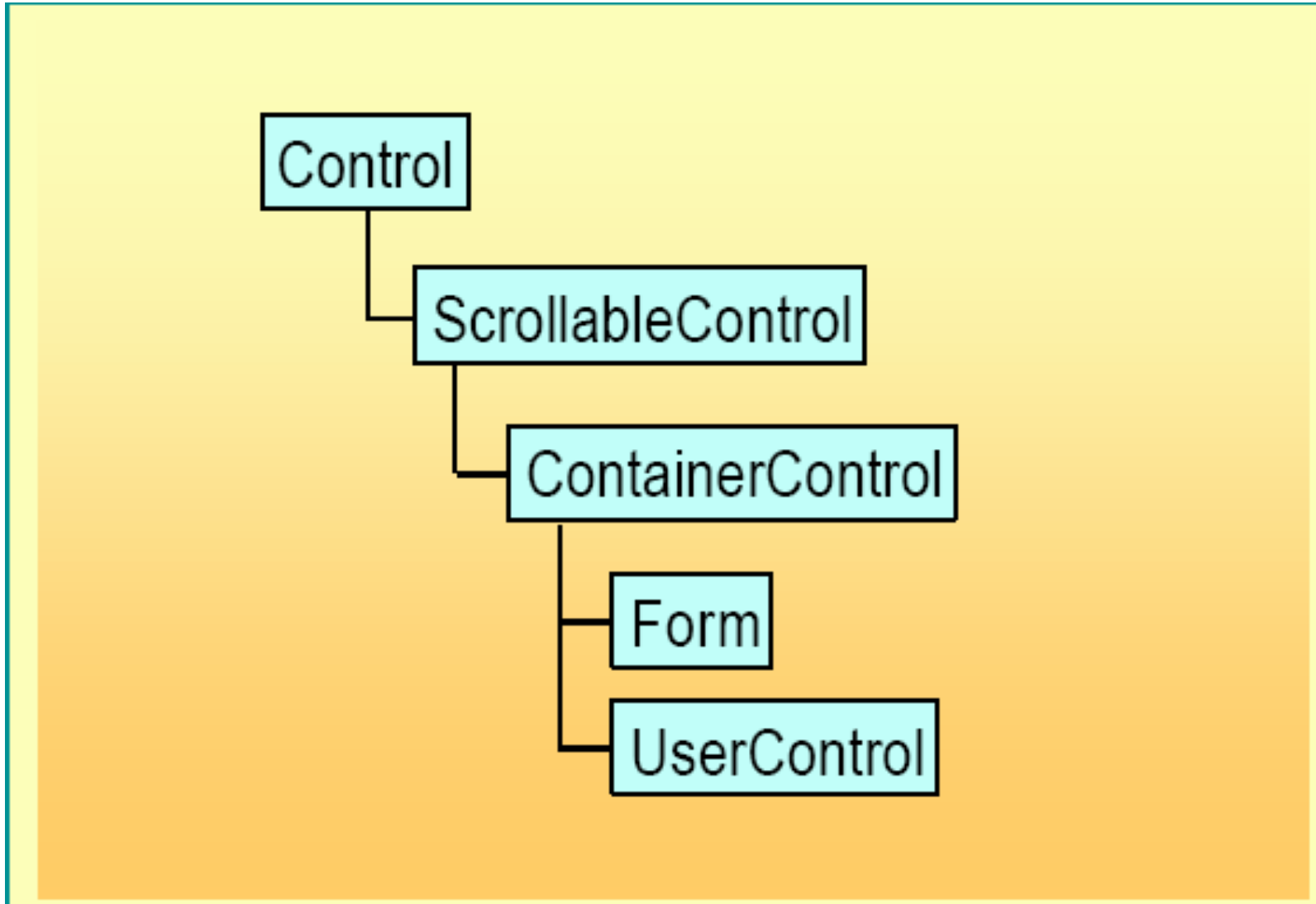


3. Cấu trúc của Windows Forms

- ❑ Phân cấp các lớp của Windows Forms
- ❑ Sử dụng lớp `Windows.Forms.Application`
- ❑ Nghiên cứu Code Behind của Windows Forms



Phân cấp các lớp của Windows Forms





Sử dụng lớp

Windows.Forms.Application

- ❑ Bắt đầu và kết thúc ứng dụng

Sub Main()

Dim frmFirst as New Form1()

frmFirst.Show() ‘Hiển thị form đầu tiên

Application.Run()

‘Cho phép ứng dụng tiếp tục sau khi form đóng

End Sub

- ❑ Dùng phương thức **DoEvents**
- ❑ Thiết lập thông tin và truy vấn thông tin ứng dụng

Dim strAppPath As String

strAppPath = Application.StartupPath

‘lấy đường dẫn chứa nơi cài file chạy



Nghiên cứu Code Behind của Windows Forms

❑ Imports

- ❑ Truy cập các chức năng trong namespace tham chiếu trong assemblies

Imports Winforms = System.Windows.Forms

❑ Class

- ❑ Thừa kế từ *System.Windows.Forms.Form*
- ❑ Constructor – Sub New()
- ❑ Initializer – Sub InitializeComponent()
- ❑ Destructor – Sub Dispose()



4. Sử dụng Windows Forms

- ❑ Sử dụng Form Properties
- ❑ Sử dụng Form Methods
- ❑ Sử dụng Form Events
- ❑ Điều khiển sự kiện
- ❑ Tạo Form MDI
- ❑ Sử dụng các dạng hộp thoại chuẩn



Sử dụng Form Properties

- ❑ Kích trên Form hoặc Control sẽ có hộp thoại Form Properties tương ứng

Property	Value
MainMenuStrip	(none)
MaximizeBox	True
MaximumSize	0, 0
MinimizeBox	True
MinimumSize	0, 0
Opacity	100%
Padding	0, 0, 0, 0
RightToLeft	No
RightToLeftLayout	False
ShowIcon	True
ShowInTaskbar	True
Size	300, 300
SizeGripStyle	Auto
StartPosition	WindowsDefaultLocation
Tag	
Text	Form1
TopMost	False
TransparencyKey	<input type="text"/>
UseWaitCursor	False
WindowState	Normal

Text
The text associated with the control.



Sử dụng Form Properties

- ❑ DialogResult
- ❑ Font
- ❑ Opacity
- ❑ MaximumSize và MinimumSize
- ❑ TopMost
- ❑ AcceptButton và CancelButton



Sử dụng Form Methods

- ❑ CenterToScreen và CenterToParent
- ❑ Close
- ❑ Show và ShowDialog



Sử dụng Form Events

- ❑ Activated và DeActivate
 - ❑ Activated là xảy ra khi Form được kích hoạt hoặc ngưng tương tác
 - ❑ DeActive là xảy ra khi Form mất focus.
- ❑ FormClosing
 - ❑ Xảy ra khi Form đang chuẩn bị đóng.
- ❑ FormClosed
 - ❑ Xảy ra sau sự kiện Closing và trước Dispose
- ❑ MenuStart và MenuComplete
 - ❑ Xảy ra khi menu nhận và mất focus.



Điều khiển sự kiện

- ❑ Điều khiển nhiều sự kiện với một thủ tục

```
Private Sub AddOrEditButtonClick(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnAdd.Click, btnEdit.Click
```

- ❑ Sử dụng AddHandler

- ❑ Là từ khóa dùng để add các sự kiện cho Form hoặc cho Control.

```
Private Sub NavigateBtnClick(ByVal sender As System.Object, ByVal e As System.EventArgs)
```

```
    MessageBox.Show("Moving record")
```

```
End Sub
```

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
```

```
    AddHandler btnNext.Click, AddressOf NavigateBtnClick
```

```
End Sub
```



Demo dùng sự kiện

- ❑ Tạo một ứng dụng dạng Winform hiển thị các thông tin trong cửa sổ
- ❑ Code cho các sự kiện
 - ❑ **Form1_Activated** Debug.WriteLine("Activated")
 - ❑ **Form1_Closed** Debug.WriteLine("Closing")
 - ❑ **Form1_Deactivate** Debug.WriteLine("Deactivated")
 - ❑ **Form1_SizeChanged** Debug.WriteLine("Size changed")



Tạo Form MDI

- ❑ Tạo Form cha
 - ❑ Bạn có thể đặt thuộc tính *IsMdiContainer*
 - ❑ Hoặc code ở sự kiện *Form_Load*
Me.IsMdiContainer = True
Me.WindowState = FormWindowState.Maximized
- ❑ Tạo các Form con
Dim doc As Form2 = New Form2()
doc.MdiParent = Me
doc.Show()
- ❑ Truy cập các Form con
- ❑ Sắp xếp các Form con



Sử dụng các dạng hộp thoại chuẩn

❑ MsgBox

```
If MsgBox("Continue?", MsgBoxStyle.YesNo + MsgBoxStyle.Question, "Question") _  
= MsgBoxResult.Yes Then
```

...

```
End If
```

❑ Lớp MessageBox

```
If MessageBox.Show("Continue?", "Question", _  
MessageBoxButtons.YesNo, MessageBoxIcon.Question) _  
= DialogResult.Yes Then
```

...

```
End If
```

❑ InputBox



5. Demo: Thực hiện Windows Forms

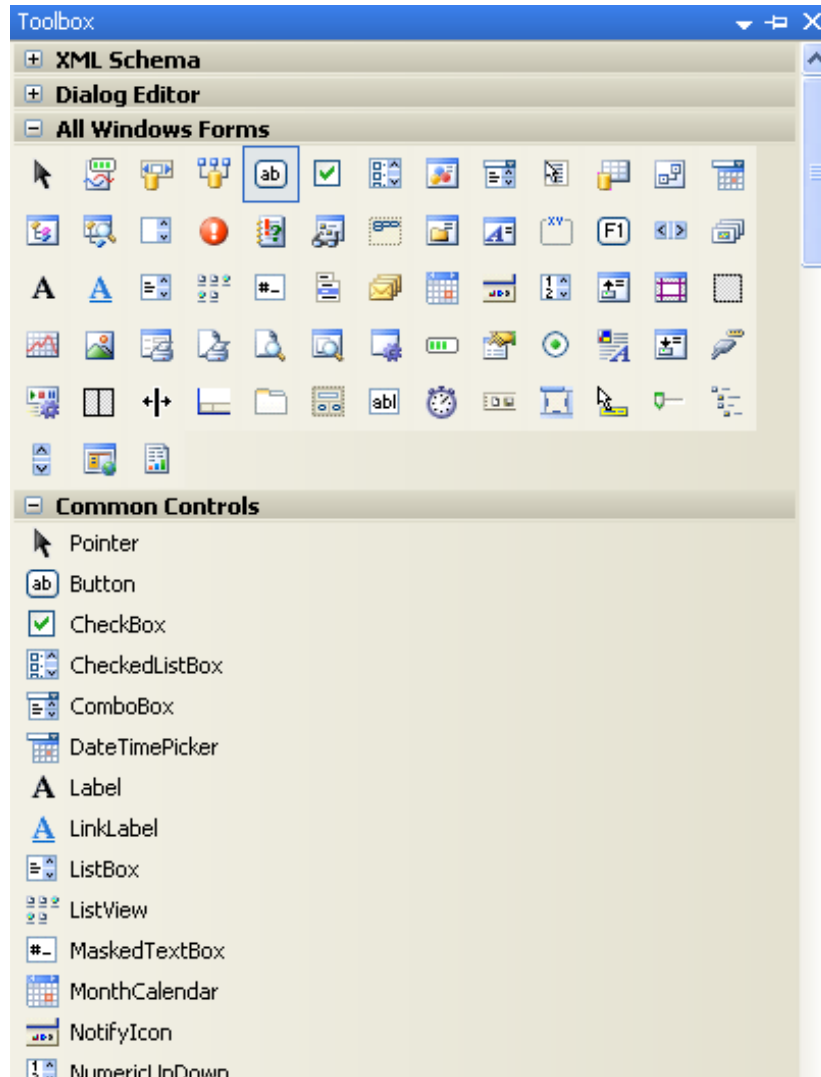
- ❑ Bạn sẽ học cách dùng *Properties* và *Methods* của Window Form, bao gồm tạo cả form riêng, độ trong suốt của form, và thanh cuộn tự động cho form.



6. Sử dụng các control

- ❑ Các control mới
- ❑ Sử dụng Properties của Control
- ❑ Sử dụng Methods của Control
- ❑ Tạo Menus
- ❑ Cung cấp Help cho người sử dụng
- ❑ Thực hiện chức năng Kéo – Thả

Các control mới



- CheckedListBox
- LinkLabel
- Splitter
- ToolTip
- NotifyIcon



Sử dụng Properties của Control

❑ Đặt vị trí Control

- ❑ Anchor

- ❑ Location

❑ Thuộc tính Text

Button1.Text = "Click Me"



Sử dụng Methods của Control

❑ BringToFront và SendToBack

Button1.BringToFront()

Button2.SendToBack()

❑ Focus

TextBox1.Focus()

TextBox1.SelectAll()



Tạo Menu

- ❑ Các lớp Menu
- ❑ Tạo Menu lúc thiết kế
 - ❑ Sử dụng Menu Designer
- ❑ Tạo Menu lúc chạy

```
Dim mnuMain As New MainMenu()
```

```
Dim mnuItem1 As New MenuItem, mnuItem2 As New MenuItem()
```

```
mnuItem1.Text = "File"
```

```
mnuMain.MenuItems.Add(mnuItem1)
```

```
mnuItem2.Text = "Exit"
```

```
mnuMain.MenuItems(0).MenuItems.Add(mnuItem2)
```

```
AddHandler mnuItem2.Click, AddressOf NewExitHandler
```

```
Menu = mnuMain
```




Cung cấp Help cho người sử dụng

❑ Control ErrorProvider

- ❑ Icon lỗi sẽ xuất hiện ở control kế tiếp, và message xuất hiện giống như ToolTip khi chuột di chuyển qua Icon.
- ❑ Được sử dụng kiểm tra dữ liệu đầu vào.

❑ Control HelpProvider

- ❑ Gắn các file trợ giúp dạng *.chm, *.hlp, *.html
- ❑ Control cung cấp thông tin trợ giúp qua thuộc tính **HelpString** hoặc **HelpTopic**



Demo: Sử dụng Controls

- ❑ Cách bố trí các thuộc tính của control Button.
- ❑ Thực thi các sự kiện cho control.
- ❑ Cuối cùng, cung cấp các trợ giúp người sử dụng qua control **HelpProvider** và **ToolTip** và cách lập trình tạo context menu.



Thực thi chức năng Kéo – Thả

- ❑ Sử lý đầu tiên
 - ❑ Dùng phương thức **DoDragDrop** trong sự kiện **MouseDown** của control kéo.
- ❑ Thay đổi Icon kéo
 - ❑ Thiết lập thuộc tính **AllowDrop** thành **True**
 - ❑ Thiết lập thuộc tính **Effect** của **DragEventArgs** trong sự kiện **DragOver** của control thả.
- ❑ Thả dữ liệu
 - ❑ Dùng phương thức **Data.GetData** để truy cập Data



Demo: Thực thi chức năng Kéo – Thả

- ❑ Thực thi chức năng kéo thả qua một ví dụ đơn giản.



8. Thừa kế của Windows Forms

- ❑ Tại sao thừa kế từ một Form
- ❑ Tạo một Form cơ sở (Form Base)
- ❑ Tạo một Form được thừa kế
- ❑ Thay đổi Form Base



Tại sao thừa kế từ một Form

- ❑ Một Form là một class, vì vậy nó có thể dùng thừa kế
- ❑ Các ứng dụng sẽ có một giao diện và cách sử dụng chuẩn.
- ❑ Các thay đổi ở form cơ sở sẽ tác động tới các form.
- ❑ Các ví dụ:
 - ❑ Các form Wizard
 - ❑ Các form Logon



Tạo một Form cơ sở (Form Base)

- ❑ Lên kế hoạch Form Base một cách cẩn thận
- ❑ Tạo form Base như form thông thường
- ❑ Thiết lập các thuộc tính truy cập cho các control
 - ❑ Private – Control chỉ truy cập trong form Base
 - ❑ Protected – control chỉ được truy cập trong các form kế thừa
 - ❑ Public – Control được truy cập trong bất kỳ module.
- ❑ Thêm từ khóa ***Overridable*** tới các Method một cách thích hợp
- ❑ Build Solution cho form Base



Tạo form được thừa kế

- ❑ Đảm bảo form base đã được hoàn thành
- ❑ Tham chiếu tới Assembly
- ❑ Tạo form mới được thừa kế
- ❑ Thay đổi thuộc tính khi cần thiết
- ❑ Viết chồng các Method hoặc Event khi có yêu cầu



Thay đổi Form Base

- ❑ Thay đổi Form Base
 - ❑ Các thay đổi sẽ tác động tới các form kế thừa khi Rebuilt
- ❑ Kiểm tra các form được kế thừa
 - ❑ Kiểm tra các thay đổi trước khi rebuilt lại ứng dụng
 - ❑ Kiểm tra lại sau khi rebuilt lại ứng dụng



9. Demo: Sử dụng thừa kế Windows Forms

- ❑ Tạo một lớp Base phục vụ cho mục đích thừa kế
- ❑ Viết chồng các Property, Method của các control form base
- ❑ Thay đổi form base sau khi nó đã được thừa kế.



Tổng kết

- Các lợi ích Windows Forms?
- Lớp **ContainerControl** là lớp cơ sở cho các control khác đúng hay sai?
- Viết code để truy cập tới đường dẫn file chạy ứng dụng
- Viết code để gọi btnOK khi người sử dụng ấn phím Enter.
- Liệt kê các control cung cấp trợ giúp cho người sử dụng
- Viết code để tạo một menu **Help** với một menu con **About** lúc chạy chương trình