CHƯƠNG 5: LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG TRONG VB.NET

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

I. Các đặc điểm lập trình hướng đối tượng
   1. Tính trừu tượng (Abstraction)
   2. Tính đóng gói (Encapsulation)
   3. Tính thừa kế (Inheritance)
   4. Tính đa hình (Polymorphism)

II. Thực hiện các đặc điểm lập trình hướng đối tượng trong VB.Net

III. Khai báo các thành viên Class
   1. Thủ tục khởi tạo(Constructors)
   2. Thủ tục khởi hủy(Destructors)
   3. Phương thức(Methods)
   4. Trường và Thuộc tính(Fields và Properties)

IV. Khai báo Namespaces

I. Các đặc điểm lập trình hướng đối tượng

1. Tính trừu tượng (Abstraction)
2. Tính đóng gói (Encapsulation)
3. Tính thừa kế (Inheritance)
4. Tính đa hình (Polymorphism)

So sánh class and object

- Class là một khuân mẫu hoặc một bản thiết kế mà định nghĩa các thuộc tính và các phương thức của đối tượng.
- Object là một bản sao chạy được của một class, sử dụng bộ nhớ và có hạn chế về thời gian.

1. Tính trừu tượng

- Khi bạn mua một tủ lạnh -> Quan tâm tới kích thước, độ bền và các đặc điểm của nó, chứ không quan tâm tới máy móc của nó được làm như thế nào -> gọi là sự trừu tượng.
- VB.Net cũng cung cấp tính trừu tượng qua class và objects
- Một class định nghĩa các thuộc tính và cách cư xử giống như các đối tượng
- Một object là bản sao của class

1. Tính trừu tượng

- Mỗi đối tượng có các đặc điểm hoặc thuộc tính -> gọi là thuộc tính (*property*) của đối tượng, và có thể thực hiện hành động -> gọi là phương thức (*method*).
- VB.Net cho phép bạn có khả năng tạo các thuộc tính và các phương thức cho các đối tượng khi tạo các class.
- Tính trừu tượng để giảm độ phức tạp của đối tượng, chỉ hiện ra các thuộc tính và các phương thức cần thiết cho đối tượng.
- Tính trừu tượng cho phép tổng quát hóa một đối tượng như một kiểu dữ liệu.

2. Tính đóng gói (Encapsulation)

- Được hiểu như việc ẩn thông tin. Nó ẩn những chi tiết không cần thiết của đối tượng.
- Ví dụ: Khi bạn bật tủ lạnh ->chức năng start bắt đầu nhưng bạn không thể nhìn thấy trong tủ hoạt động như thế nào.
- Tính đóng gói là một cách thi hành tính trừu tượng.

2. Tính đóng gói(Encapsulation)

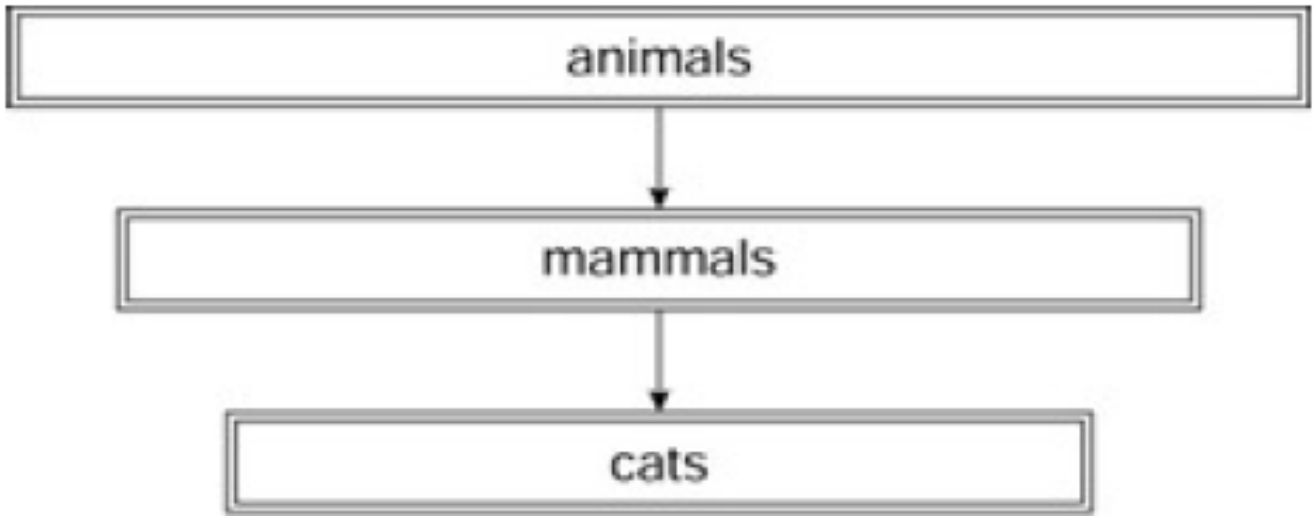
- Tính đóng gói ẩn việc thi hành của class đối với người sử dụng. Hay nói cách khác, nó chỉ hiển thị các thuộc tính và các phương thức của đối tượng.

3. Tính thừa kế

- Một class thừa kế từ một class tồn tại. Lớp thừa kế gọi là lớp con (subclass) và lớp class được thừa kế gọi là lớp cơ sở (base class).
- Tất cả các lớp trong VB.Net đều xuất phát từ lớp Object.
- Lớp con thừa kế các thuộc tính và các phương thức từ lớp cơ sở.
- Cũng có thể thêm các thuộc tính và phương thức cho lớp con. Bạn cũng có thể chồng các phương thức của lớp cơ sở.

3. Tính thừa kế

- Tính thừa kế cho phép bạn tạo phân cấp các đối tượng.
- Ví dụ: phân cấp class



3. Tính thừa kế

- Mặc định, tất các các class bạn tạo trong VB.Net có thể được thừa kế.
- Thừa kế cho phép bạn dùng lại code và tạo các đối tượng phức tạp hơn từ các đối tượng đơn giản.
- VB.Net cung cấp nhiều từ khóa cho phép bạn thi hành việc thừa kế

4. Tính đa hình

- Để chỉ một đối tượng tồn tại nhiều khuân dạng khác nhau.
- Ví dụ: Khi bạn mua tủ lạnh có 2 cách, bạn phải liên hệ với người bán hoặc nhà sản xuất.
  - Khi bạn liên hệ với ngừoi bán, người bán sẽ đặt hàng và liên hệ với công ty.
  - Khi bạn liên hệ với công ty, tuy nhiên công ty sẽ liên hệ với người bán ở vùng của bạn để sắp đặt việc phân phát tủ lạnh.
  - Như vậy, người bán và công ty là hai class khác nhau. Mỗi class đều có cách phản hồi khác nhau về cùng việc đặt hàng.->Hiểu như là tính đa hình trong lập trình hướng đối tượng

4. Tính đa hình

- Tính đa hình cho phép bạn tạo cùng phương thức nhưng thực thi các công việc khác nhau.
- Bạn cũng có thay đổi cách thực thi các phương thức của lớp cơ sở.

II. Thực thi OOP trong VB.Net

- Tính trừu tượng được thể hiện bằng việc dùng class
- Cú pháp tạo class:

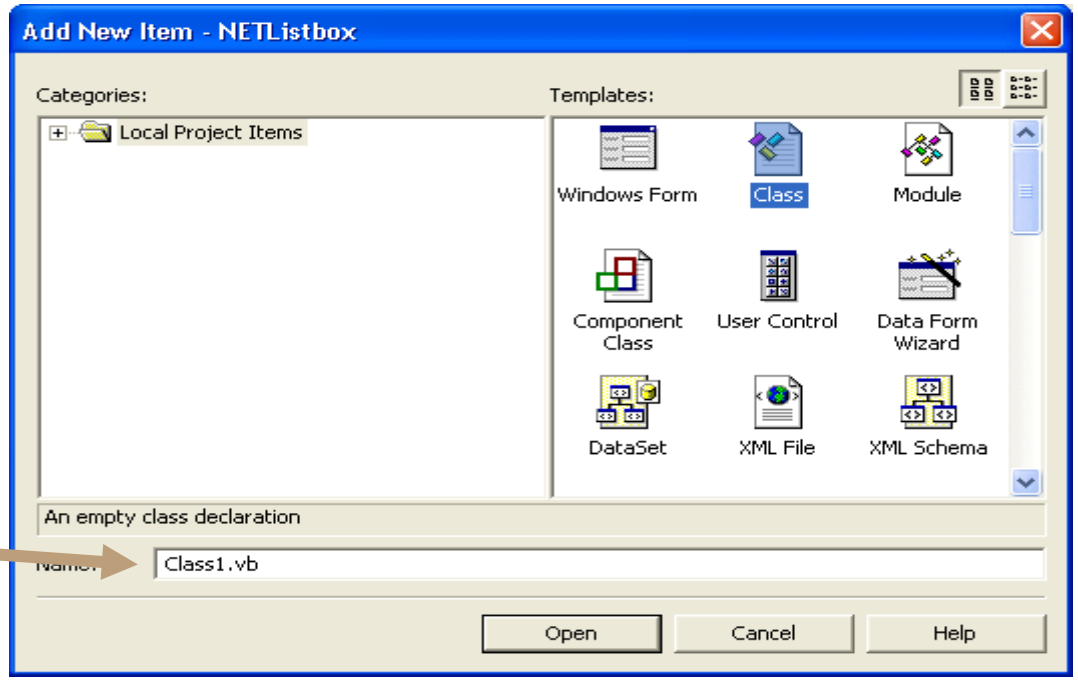
```
[AccessModifier][Keyword] Class _
  ClassName [Implements InterfaceName]

'Declare properties and methods

End Class
```

Tiếp

- AccessModifier định nghĩa khả năng truy cập của class, sử dụng một trong các từ khóa : Public, Private, Protected, Friend,Protected Friend.
- Keyword chỉ rõ các lớp có được thừa kế hay không, từ khóa Inherit, NotInheritable hoặc MustInherit.
- Class đánh dấu bắt đầu một class
- Classname: tên của một class
- Implements chỉ rõ class thực thi trên giao diện nào.
- InterfaceName miêu tả tên giao diện.Một class có thể thực thi trên một hoặc nhiều giao diện.
- End Class đánh dấu kết thúc khai báo của một class

Tạo class trong vb.net

Vào File \ Add New Item …



Nhập tên class

```
Public Class Communication
  'Declare properties and methods
End Class
```

Bảng AccessModifier

Access Modifier	Dùng trong	Mô tả
Public	module, class, structure	Được truy cập từ cùng project, từ project khác hoặc từ thành phần khác
Private	module, class, structure	Chỉ được truy cập trong cùng module, class , structure
Protected	Classes, class member	Được truy cập trong cùng class , hoặc class được kế thừa
Friend	module, class, structure	Truy cập được trong cùng project
Protected Friend	Classes, class member	Truy cập được trong cùng project Và từ các class được kế thừa

AccesModifier

- Một Module là một khối chứa được các class, thuộc tính, phương thức bạn định nghĩa.
- Một Structure được sử dụng để tạo ra kiểu dữ liệu người dùng tự định nghĩa
- Class member bao gồm các thủ tục, các trường, các phương thức định nghĩa trong class
- AccessModifier có khả năng cho bạn thực hiện tính trừu tượng và tính đóng gói

Tính thừa kế (Inherits)

- Cú pháp: Inherits
- Ví dụ:

```
Public Class ThisClass
        Inherits OtherClass
        'Property and method declarations
        'Other code
End Class
```

- Lớp *ThisClass* kế thừa từ lớp *OtherClass*
- VB.Net cung cấp các từ khóa khác nhau để thực hiện việc thừa kế

Bảng Keyword

Keyword	Được dùng với	Mục đính
Inherits	Classes	Thừa kế tất cả các thành viên của lớp thừa kế (trừ private)
MustInherit	Classes	Chỉ rõ lớp này chỉ sử dụng như lớp cơ sở
NotInheritable	Classes	Chỉ rõ lớp này không được sử dụng như lớp cơ sở
Overridable	Procedures	Chỉ rõ thủ tục có thể viết chồng trong class được thừa kế.
NotOverridable	Procedures	Chỉ rõ thủ tục không được viết chồng trong class được thừa kế.

Bảng Keyword (tiếp)

Keyword	Được dùng với	Mục đính
MustOverride	Procedures	Chỉ định các thủ tục phải viết chồng trong tất cả các lớp được kế thừa
Overrides	Procedures	Chỉ định một thủ tục được viết chồng từ lớp cơ sở
MyBase	Code	Gọi code của lớp cơ sở từ lớp được thừa kế
MyClass	Procedures	Gọi code của chính class đó
Protected	Procedures, fields	Chỉ định các thủ tục và các trường được truy cập trong cùng class và các class được thừa kế

Ví dụ

```
Public MustInherit Class Communication
        Public Sub New()
            MyBase.New()
            MsgBox("Constructor of Communication class", MsgBoxStyle.OKOnly)
        End Sub

            Public MustOverride Function Send() As Boolean
End Class

Public Class Email
     Inherits Communication
    Public Sub New()
        MyBase.New()
        MsgBox("Constructor of Email class", MsgBoxStyle.OKOnly)
    End Sub

     Overrides Function Send() As Boolean
      MsgBox("Send function of Email class", MsgBoxStyle.OKOnly)
      'Code specific to the Email class
      return True
     End Function
End Class
```

Ví dụ (tiếp)

```
Public Class Fax
    Inherits Communication
    Public Sub New()
      MyBase.New()
      MsgBox("Constructor of Fax class", MsgBoxStyle.OKOnly)
    End Sub

    Overrides Function Send() As Boolean
          MsgBox("Send function of Fax class", MsgBoxStyle.OKOnly)
      'Code specific to the Fax class
          return True
    End Function
End Class
```

Ví dụ (tiếp)

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
    System.EventArgs) Handles Button1.Click
    Dim int1 As Integer
    Dim communicate As Communication
    int1 = InputBox("Enter 1 to send an e-mail message and 2 to send a fax
    message.")
    Select Case (int1)
        Case "1"
        communicate = New Email()
        communicate.Send()
        Case "2"
        communicate = New Fax()
        communicate.Send()
    End Select
End Sub
```

Giải thích

- Lớp *Email* và *Fax* kế thừa từ lớp cơ sở *Communication*
- Tính đa hình thể hiện thể hiện ở chỗ bạn có thể chồng nhiều phương thức.
- Có thể ghi chồng phương thức của lớp cơ sở, nó có thể thực hiện các hành động khác.

III. Khai báo các thành viên Class

1. Constructor (Thủ tục khởi dựng)

- Được dùng để khởi tạo đối tượng
- VB.Net, Thủ tục Sub New thực hiện như một Contructor
- Thủ tục Sub New được thực hiện khi đối tượng của class đươc tạo, bạn có thể thực hiện các công việc cần thiết trước khi dùng đối tượng.
- Ví dụ khi bạn kết nối với Database, và các biến được khởi tạo trong Sub New

Contructors(tiếp)

- Tất cả các class trong VB.Net đều xuất phát từ class Object. Vì vậy khi tạo một class, bạn cần gọi Contructor của class Object. Để làm việc này bạn thêm câu lệnh MyBase.New() vào dòng đầu tiên của Contructor class của bạn.

```
Public Class MyNewClass
  Public Sub New()
    MyBase.New()
    'Code for Initializing objects and variables
  End Sub
 'Other class members
End Class
```

Contructors(tiếp)

- Contructor cũng có thể mang đối số.

```
Public class Employer
    Public Sub New(Optional ByVal iempcode As Integer = 0)
     ‘code initiation here
    End Sub
     ‘code here
End Class
```

Bạn có thể tạo các đối tượng Employer:

```
Dim Emp1 As New Employee(1001)
```

Hoặc

```
Dim Emp2 As Employee = New Employee(1001)
```

- Contructor là tùy chọn.

2. Destructors (Thủ tục khởi hủy)

- VB.Net cũng cung cấp thủ tục khởi hủy Sub Finalize.
- Thủ tục khởi hủy thực hiện ngược lại với hàm khởi tạo.
- Thủ tục khởi hủy sẽ giải phóng bộ nhớ và các tài nguyên đã được sử dụng của đối tượng.
- Thủ tục Sub Finalize là phương thức Protected

Ví dụ

```
Protected Overrides Sub Finalize()

    MyBase.Finalize()

    'Add code here

End Sub
```

- Từ khóa Overrides được sử dụng bởi vì Thủ tục Sub Finalize() kế thừa từ lớp Object

Destructors

- Thủ tục Sub Finalize() không được gọi khi chạy chương trình, .Net framework sẽ gọi thủ tục này khi đối tượng kết thúc để giải phóng bộ nhớ và tài nguyên đối tượng sử dụng.
- .Net Framework tự động thu thập rác -> Bạn không cần thêm công việc giải phóng bộ nhớ và tài nguyên.
- .Net Framework cung cấp giao diện IDisposable giúp bạn quản lý tài nguyên.

Destructors

- IDisposable cung cấp phương thức Dispose. Không giống thủ tục Sub Finalize(), bạn có thể gọi phương thức Dispose. Bạn có thể thêm code trong phương thức Dispose để giải phóng tài nguyên và các công việc như đóng kết nối tới đa
- Thủ tục Sub Finalize() đảm bảo rác được dọn dẹp nếu phương thưc Dispose không được gọi.

Methods

- Bao gồm thủ tục Sub và Function được khai báo trong class.
- VB.Net bao gồm các thủ tuc Sub, Function, và Property.
- Thủ tục Sub không trả về giá trị. Câu lệnh Sub .. End Sub. Khi thủ tục Sub được gọi, tất cả các câu lệnh trong thủ tục được thực hiện cho đến khi gặp các câu lệnh End Sub, Exit Sub hoặc Return.

Methods

- Thủ tục Function trả về giá trị khi gọi nó. Dùng câu lệnh Function và End Function để định nghĩa thủ tục Function. Thủ tục Function thực thi các câu lệnh phía trong cho đến khi gặp các câu lệnh End Function, Exit Function hoặc Return.
- Hai thủ tục Function và Sub có thể mang đối số như hằng, biến, biểu thức.
- Bạn có thể chồng các phương thức trong VB.Net.

Fields and Properties

- Fields là các biến được khai báo trong class mà có thể truy cập từ class khác
- Ví dụ:

```
Public Class MyClass1
    Public MyField As Integer 'Declaring a field
    'Other declarations and code
End Class

Dim MyObject As New MyClass1()
MyObject.MyField = 6
```

Properties

- Properties định nghĩa các thuộc tính của đối tượng.
- Cú pháp tạo Property:

```
Public Property NameProperty() As DataType
  Get
    Return PropertyValue
    'Where PropertyValue is the property's value
  End Get
  Set(ByVal value As DataType)
    PropertyValue = value
  'Where PropertyValue is the new value to be assigned
  End Set
End Property
```

Property

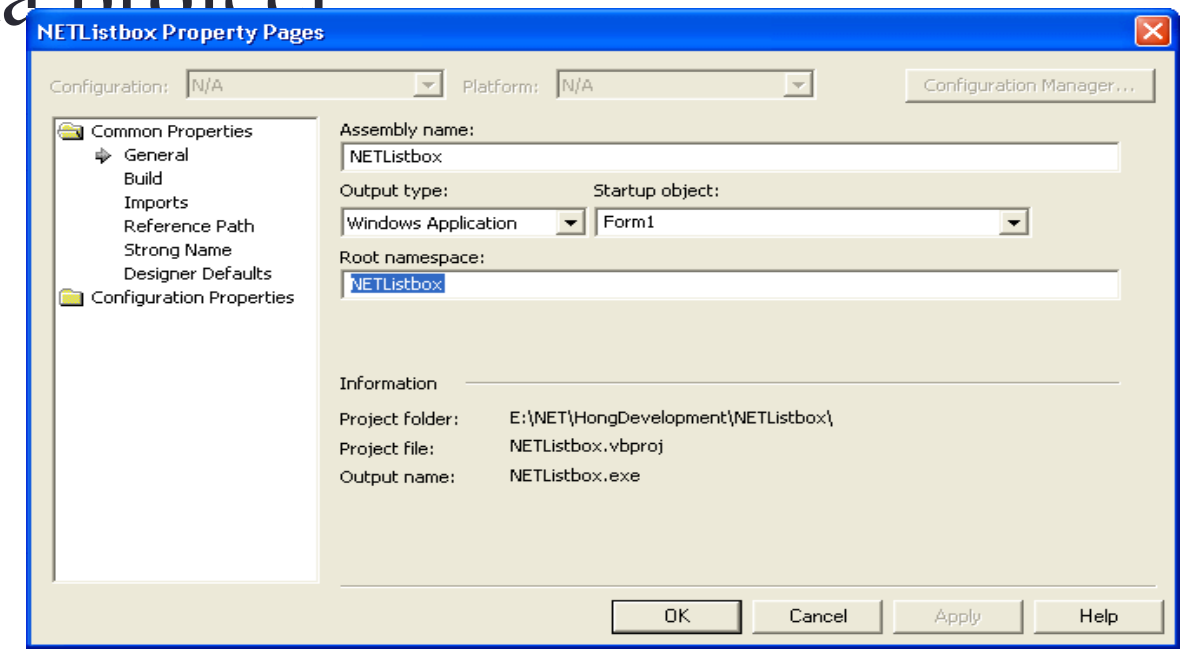
- Mặc định Property có cả hai thuộc tính đọc và ghi. VB.Net cũng định nghĩa thuộc tính chỉ đọc hoặc chỉ ghi.
- Thuộc tính chỉ đọc dùng từ khóa ReadOnly
- Thuộc tính chỉ ghi dùng từ khóa WriteOnly
- Các từ đặt trước từ khóa AcessModifier

IV. Khai báo Namespaces

- Namspaces được sử dụng để tổ chức các đối tượng được định nghĩa trong một assemby.
- Namspace cho phép bạn tổ chức phân cấp các đối tượng. Cấu trúc phân cấp là việc nhóm các đối tượng tương tự nhau, để dễ truy cập.
- Mặc định, tên Project được dùng làm namespace.

Khai báo Namespaces

- Nếu bạn click chuột phải lên tên project trong Solution Explorer rồi chọn **Properties** bạn sẽ xem được namespace của project



Khai báo Namespaces

- Bạn cũng có thể tạo Namespace riêng của bạn bằng cách dùng câu lệnh: Namespace and End Namespace

```
Namespace MyNamespace
    Public Class MyOwnClass1
    'Code for the MyOwnClass1
    End Class
    Public Class MyOwnClass2
    'Code for the MyOwnClass2
    End Class
End Namespace
```

Khai báo Namespaces

- Các namespace có thể lồng nhau

```
Namespace GroceryStore
    Namespace Edible
        Public Class Vegetables
        'Code for the class
        End Class
        Public Class Drinks
        'Code for the class
        End Class
    End Namespace
    Namespace Inedible
        Public Class Cosmetics
        'Code for the class
        End Class
        Public Class Toiletries
        'Code for the classx`
        End Class
    End Namespace
End Namespace
```

- Bạn có thể dùng từ câu lệnh Imports để truy cập vào các class trong namespaces