BÀI GIẢNG

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

CHƯƠNG 6

ĐA HÌNH & GIAO DIỆN

TRẦN THỊ THU THẢO

BỘ MÔN TIN HỌC QUẢN LÝ, KHOA THỐNG KÊ – TIN HỌC

TRƯỜNG ĐẠI HỌC KINH TẾ, ĐẠI HỌC ĐÀ NẴNG

THAOTRAN@DUE.EDU.VN

NỘI DUNG

1. Khái niệm đa hình
2. Lớp trừu tượng
3. Giao diện
4. So sánh giữa giao diện và lớp trừu tượng
5. Thực thi trên C#

KHÁI NIỆM ĐA HÌNH

❑ KHÁI NIỆM

Tính đa hình là hiện tượng các đối tượng thuộc các lớp khác nhau có thể hiểu cùng 1 thông điệp theo các cách khác nhau.

Ví dụ về đa hình trong thực tế. Ta có 3 con vật: chó, mèo, lợn. Cả 3 con vật này đều là động vật. Nhưng khi ta bảo cả 3 động vật kêu thì con chó sẽ kêu gâu gâu, con mèo sẽ kêu meo meo và con heo sẽ kêu ủn ỉn. Trong ví dụ trên 3 con vật: chó, mèo, lợn xem như là các đối tượng. Việc ta bảo 3 động vật kêu chính là thông điệp. Rõ ràng cả 3 con vật có thể hiểu cùng 1 thông điệp là kêu theo các cách khác nhau.

KHÁI NIỆM ĐA HÌNH

❑ CÁC BƯỚC THỰC HIỆN ĐA HÌNH

1. Lớp cơ sở đánh dấu phương thức ảo bằng từ khóa **virtual** hoặc **abstract**.
2. Các lớp dẫn xuất định nghĩa lại phương thức ảo này (đánh dấu bằng từ khóa **override**).
3. Vì tham chiếu thuộc lớp cơ sở có thể trỏ đến một đối tượng thuộc lớp dẫn xuất và có thể truy cập hàm ảo đã định nghĩa lại trong lớp dẫn xuất nên khi thực thi chương trình, tùy đối tượng được tham chiếu này trỏ tới mà phương thức tương ứng được gọi thực hiện. Nếu tham chiếu này trỏ tới đối tượng thuộc lớp cơ sở thì phương thức ảo của lớp cơ sở được thực hiện.

KHÁI NIỆM ĐA HÌNH

Ví dụ 1: Ta có 3 lớp Animal, Cat, Dog. Trong đó Cat và Dog kế thừa từ lớp Animal. Trong các lớp đều có phương thức **Speak()**.

Virtual là từ khoá dùng để khai báo 1 phương thức ảo (phương thức ảo là phương thức có thể ghi đè được)

```
class Animal
{
    public virtual void Speak()
    {
      Console.WriteLine("Animal is speaking. . .");
    }
}
```

KHÁI NIỆM ĐA HÌNH

Ví dụ 1:

override là từ khoá dùng để đánh dấu phương thức ghi đè lên phương thức của lớp cơ sở

```
class Cat : Animal
    {
        public override void Speak()
        {
            Console.WriteLine(" Cat is speaking Meo Meo");
        }
    }
    class Dog : Animal
    {
        public override void Speak()
        {
            Console.WriteLine(" Dog is speaking Gau Gau");
        }
    }
```

KHÁI NIỆM ĐA HÌNH

Ví dụ 1:

```
class Program
    {
        static void Main(string[] args)
        {
            Animal cat = new Cat();
            Animal dog = new Dog();
            cat.Speak();
            dog.Speak();
        }
    }
```

```
Cat is speaking Meo Meo
Dog is speaking Gau Gau
```

LỚP ĐỐI TƯỢNG

Tất cả các lớp trong C# đều được dẫn xuất từ lớp Object. Lớp Object là lớp gốc trên cây phân cấp kế thừa, nó cung cấp một số phương thức mà các lớp con có thể ghi đè (override) như:

Phương thức	Ý nghĩa
Equals()	Kiểm tra hai đối tượng có tương đương nhau không
GetHashCode()	Cho phép đối tượng cung cấp hàm Hash riêng để sử dụng trong kiểu tập hợp
GetType()	Trả về kiểu đối tượng
ToString()	Trả về chuỗi biểu diễn đối tượng
Finalize()	Xóa đối tượng trong bộ nhớ
MemberwiseClone()	Tạo copy của đối tượng

LỚP TRỪU TƯỢNG

Phương thức thuần ảo là 1 phương thức ảo và không có định nghĩa bên trong.

Lớp trừu tượng là lớp chứa phương thức thuần ảo.

Abstract là từ khoá dùng để khai báo 1 lớp trừu tượng hoặc 1 phương thức thuần ảo.

LỚP TRỪU TƯỢNG

Xét ví dụ 1: Ở đây phương thức Speak() của lớp Animal, phần định nghĩa của phương thức này chỉ là hình thức sau đó cũng sẽ bị các lớp kế thừa ghi đè lên.

Khi sử dụng abstract để nhấn mạnh 2 điều:

- Phương thức **Speak()** có thể ghi đè (override).
- Phương thức **Speak()** không có định nghĩa gì bên trong.

LỚP TRỪU TƯỢNG

Để khai báo lớp trừu tượng và phương thức thuần ảo ta chỉ cần thêm khoá abstract vào trước tên lớp và tên phương thức.

```
abstract class <Tên lớp cơ sở>
{

// Khai báo phương thức thuần ảo nên không cần định nghĩa nội dung
abstract public void <Tên phương thức>;

}
```

LỚP TRỪU TƯỢNG

Ví dụ 1: Thay đổi cách viết ở hàm cơ sở, kết quả ta nhận được vẫn như lúc đầu.

```
abstract class Animal
{
    abstract public void Speak();
}
```

```
Cat is speaking Meo Meo
Dog is speaking Gau Gau
```

GIAO DIỆN (interface)

❑ KHÁI NIỆM

Giao diện là một dạng của lớp trừu tượng được sử dụng với mục đích hỗ trợ tính đa hình. Trong giao diện không có bất cứ một cài đặt nào, chỉ có nguyên mẫu của các phương thức, chỉ mục, thuộc tính mà một lớp khác khi kế thừa nó thì phải có cài đặt cụ thể cho các thành phần này (tức là lớp kế thừa giao diện tuyên bố rằng nó hỗ trợ các phương thức, thuộc tính, chỉ mục được khai báo trong giao diện).
Khi một lớp kế thừa một giao diện ta nói rằng lớp đó thực thi (implement) giao diện.

GIAO DIỆN (interface)

❑ CÚ PHÁP

```
interface <tên interface>
{
    // Khai báo các thành phần bên trong interface
}
```

Trong đó:

•**Interface** là từ khoá dùng để khai báo 1 interface.

•**<tên interface>** là tên do người dùng đặt và tuân theo các quy tắc đặt tên

• **Lưu ý** là để tránh nhầm lẫn với lớp kế thừa thì khi đặt tên interface người ta thường thêm tiền tố "**I**" để nhận dạng.

ĐẶC ĐIỂM CỦA interface

- Chỉ chứa khai báo không chứa phần định nghĩa (giống phương thức thuần ảo). Mặc dù giống phương thức thuần ảo nhưng bạn không cần phải khai báo từ khoá abstract.
- Việc ghi đè 1 thành phần trong interface cũng không cần từ khoá override.
- Không thể khai báo phạm vi truy cập cho các thành phần bên trong interface. Các thành phần này sẽ mặc định là public.

ĐẶC ĐIỂM CỦA interface

- Interface không chứa các thuộc tính (các biến) dù là hằng số hay biến tĩnh vẫn không được.
- Interface không có constructor cũng không có destructor.
- Các lớp có thể thực thi nhiều interface cùng lúc (ở 1 góc độ nào đó có thể nó là phương án thay thế đa kế thừa).
- Một interface có thể kế thừa nhiều interface khác nhưng không thể kế thừa bất kỳ lớp nào.

ĐẶC ĐIỂM CỦA interface

Ví dụ 2: Mọi con vật đều có tiếng kêu. Ta có thể dùng định nghĩa một giao diện kèm thêm một thuộc tính để mô tả tiếng kêu của con vật.

```
interface ISpeak
{
    void Speak();
}
class Animal:ISpeak
{
    public void Speak()
    {
    Console.WriteLine("Animal is speaking...");
    }
}
```

ĐẶC ĐIỂM CỦA interface

Ví dụ 2:

```
class Cat : ISpeak
    {
        public void Speak()
        {
            Console.WriteLine("Cat is speaking Meo Meo");
        }
    }
    class Dog : ISpeak
    {
        public void Speak()
        {
            Console.WriteLine("Dog is speaking Gau Gau");
        }
    }
```

ĐẶC ĐIỂM CỦA interface

```
class Program
    {
        static void Main(string[] args)
        {
            Animal animal = new Animal();
            Dog dog = new Dog();
            Cat cat = new Cat();
            animal.Speak();
            dog.Speak();
            cat.Speak();
        }
    }
```

```
Animal is speaking. . .
Dog is speaking Gau Gau
Cat is speaking Meo Meo
```

SO SÁNH GIỮA interface & abstract

Những điểm giống nhau giữa **interface** và **abstract class**:

- Đều có thể chứa phương thức thuần ảo.
- Đều không thể khởi tạo đối tượng.

SO SÁNH GIỮA interface & abstract

Những điểm khác nhau giữa **interface** và **abstract class**:

Interface	Abstract class
Chỉ có thể kế thừa nhiều interface khác	Có thể kế thừa 1 lớp và nhiều interface
Chỉ chứa các khai báo và không có phần nội dung. Không thể chứa biến	Có thể chứa các thuộc tính (biến) và các phương thức bình thường bên trong
Không cần dùng từ khóa override để ghi đè	Sử dụng từ khóa override để ghi đè
Không có constructor và cũng không có destructor	Có constructor và cũng không có destructor
Không co phạm vi truy cập. Mặc định luôn là public	Có thể khai báo phạm vi truy cập

SO SÁNH GIỮA interface & abstract

Những điểm khác nhau giữa **interface** và **abstract class**:

Interface	Abstract class
Dùng để định nghĩa 1 khuôn mẫu hoặc quy tắc chung	Dùng để định nghĩa cốt lõi của lớp, thành phần chung của lớp và sử dụng cho nhiều đối tượng cùng kiểu
Cần thời gian để tìm phương thức thực tế tương ứng với lớp dẫn đến thời gian chậm hơn	Nhanh hơn so với interface
Khi cần thêm mới 1 khai báo. Cần phải tìm hết tất cả những lớp có thực thi interface này để định nghĩa nội dung cho phương thức mới	Đối với abstract class, khi định nghĩa một phương thức mới, hoàn toàn có thể định nghĩa nội dung phương thức là rỗng hoặc những thực thi mặc định nào đó. Vì thế toàn bộ hệ thống vẫn chạy bình thường

SO SÁNH GIỮA interface & abstract

Những điểm khác nhau giữa **interface** và **abstract class**:

Interface	Abstract class
Dùng để định nghĩa 1 khuôn mẫu hoặc quy tắc chung	Dùng để định nghĩa cốt lõi của lớp, thành phần chung của lớp và sử dụng cho nhiều đối tượng cùng kiểu
Cần thời gian để tìm phương thức thực tế tương ứng với lớp dẫn đến thời gian chậm hơn	Nhanh hơn so với interface

Nguồn: https://howkteam.vn/course/lap-trinh-oop-voi-c/interface-trong-lap-trinh-huong-doi-tuong-1396

BÀI TẬP ÔN TẬP CHƯƠNG

Bài 1. Xây dựng lớp **Hinhhoc** (Hình học) với phương thức tính **chu vi, diện tích** là phương thức trừu tượng hoặc phương thức ảo. Sau đó định nghĩa các lớp **HinhChuNhat** (Hình chữ nhật), **Hinhtron** (Hình tròn), **Hinhtamgiac** (Hình tam giác), **Hinhvuong** (Hình vuông) kế thừa từ lớp **Hinhhoc** với các thành phần dữ liệu và phương thức tính chu vi, diện tích cụ thể của từng loại đối tượng.

BÀI TẬP ÔN TẬP CHƯƠNG

Bài 2. Mọi thiết bị (**Máy quạt, Điều hòa, Tivi**) đều có tính năng **Mở (ON)** và tắt **(OFF).** Hãy dùng định nghĩa một giao diện kèm theo một thuộc tính để cho biết, máy quạt, điều hòa, tivi có đang mở hay không?

BÀI TẬP ÔN TẬP CHƯƠNG

Bài 3. Để quản lý danh mục sách, tạo một lớp **Edition** sử dụng phương thức trừu tượng hoặc phương thức ảo.

- Trong lớp **Edition**, tạo phương thức **CompareTo** để sắp xếp danh mục các ấn phẩm theo tên của Tác giả
- Tạo các lớp dẫn xuất gồm:

  Book (title, author, year, publisher)

  Article (title, author, journal, year)

  OnlineResoure (title, author, link, abstract)
- Trong phương thức **Main()**, tạo một mảng gồm n ấn phẩm, hiển thị thông tin đầy đủ từng danh mục, sắp xếp danh mục ấn phẩm theo tên ấn phẩm **(title)** hoặc theo tên tác giả **(author)** và thực hiện chức năng tìm kiếm ấn phẩm theo tên của tác giả **(author).**

BÀI TẬP ÔN TẬP CHƯƠNG

Bài 4. Một khách sạn 5 sao cần quản lý các thông tin cho thuê phòng (Phòng **Standard** và Phòng **VIP**). Với Phòng Standard cần lưu **tên khách hàng, số CMND, Số ngày thuê**. Tiền thuê phòng cho phòng Standard 500$/ ngày nếu ở dưới 5 ngày, 400$/ngày nếu ở trên 5 ngày. Với mỗi phòng VIP cần lưu **tên khách hàng, số CMND, Số ngày thuê, loại phòng** (**phòng Luxury, phòng President**). Nếu thuê dưới 5 ngày, phòng Luxury sẽ có giá thuê là 1100$/ngày và phòng President có giá thuê là 1300$/ngày. Thuê từ 6 ngày trở lên thì tính 1000$ cho cả hai loại phòng.
Viết chương trình xây dựng các lớp cần thiết, sau đó nhập danh sách thông tin thuê phòng (phòng Standard và phòng VIP), sau đó xuất ra các thông tin sau:

- Xuất ra tất cả các thông tin thuê phòng (kể cả doanh số tương ứng)
- Tính tổng số tiền cho thuê phòng Standard và phòng VIP
- Xuất tất cả các thông tin liên quan đến việc thuê phòng Standard
- Tính tổng số tiền cho thuê phòng Luxury