Bài 6.1: Tổng quan về lớp và đối tượng

- ✓ Khái niệm và đặc điểm
- ✓ Cú pháp khai báo lớp
- ✓ Tổng quan về access modifier
- ✓ Thao tác với đối tượng
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Khái niệm và đặc điểm

- ✓Một lớp là một kiểu dữ liệu tham chiếu do người dùng tự định nghĩa.
- ✓Hiểu đơn giản, một lớp là một bản thiết kế, một cái khuôn đúc mà từ đó các đối tượng được tạo ra.
- ✓Một lớp được sử dụng để gói gọn các mô tả thông tin và hành động của một họ các đối tượng tương tự nhau.
- ✓Một đối tượng là một sản phẩm cụ thể sinh ra từ một cái khuôn đúc nào đó(các kiểu hay các lớp).
- ✓Một lớp bản thân nó không phải là đối tượng. Các đối tượng sinh ra từ cùng một lớp sẽ có chung các đặc điểm, hành vi nhưng khác về trạng thái của chúng.
- ✓Để khai báo một lớp ta sử dụng keyword class. Để tạo đối tượng của lớp ta sử dụng keyword new.

Khái niệm và đặc điểm

- ✓Một biến của một class sẽ nhận giá trị null cho tới khi nó được gán cho một đối tượng cụ thể bằng cách tạo mới đối tượng với keyword new hoặc sử dụng các đối tượng của kiểu tương thích.
- ✓Khi đối tượng của lớp được tạo, nó sẽ được cấp phát đầy đủ bộ nhớ trong bộ nhớ heap.
- ✓Sau đó được giám sát và thu hồi khi không còn được sử dụng bởi trình dọn rác của hệ thống.

Cú pháp khai báo lớp

✓Sau đây là cú pháp khai báo lớp:

```
access_modifier class ClassName
{
    // Methods
    // Properties
    // Fields
    // Events
}
```

✓Trong đó:

- ✓ Phần access-modifier là cấp độ truy cập của lớp.
- ✓ Tiếp đó tới keyword class là bắt buộc
- ✓ Sau keyword class là ClassName – tên lớp. Tên lớp thường là danh từ, cụm danh từ mô tả ý nghĩa sử dụng của class.
- ✓ Thân class bao bởi cặp ngoặc {}.
- ✓ Bên trong định nghĩa lớp có thể có 1 hoặc nhiều thành phần: các methods-phương thức, properties-thuộc tính, fields-các trường, events-các sự kiện.

Ví dụ

```
public class Student
{
    // các phương thức khởi tạo
    0 references
    public Student(string id)
    {
        Id = id;
    }
    // các thuộc tính
    1 reference
    public string Id { get; set; }
    0 references
    public string FullName { get; set; }
    0 references
    public int Age { get; set; }
    0 references
    public string Major { get; set; }
    0 references
    public float Gpa { get; set; }
    // các trường
    public string rank;

    // các phương thức
    0 references
    public void CalculateRank()
    {
        // do something
    }
}
```

Các access modifier

Sau đây là các access modifier:

- ✓public: cấp độ truy cập cao nhất, cho phép một lớp, thành phần được nhìn thấy và truy cập từ bất kì đâu.
- ✓protected: một lớp hoặc thành phần ở cấp độ truy cập này chỉ được nhìn thấy và truy cập bởi lớp chứa nó, các lớp con kế thừa từ lớp hiện tại.
- ✓private: một lớp hoặc thành phần ở cấp độ truy cập này chỉ được nhìn thấy và truy cập từ bên trong lớp hoặc struct chứa nó.
- ✓internal: lớp hoặc thành phần ở cấp độ truy cập này có thể được nhìn thấy và truy cập từ bất kì nơi nào trong cùng file assembly.
- ✓protected internal: lớp hoặc thành phần ở cấp độ truy cập này có thể truy cập từ code cùng assembly và từ lớp con trong assembly khác.
- ✓private protected: lớp hoặc thành phần ở cập độ truy cập này có thể được truy cập từ các lớp con cùng assembly.

Sử dụng access modifiers

- ✓Các thành phần bên trong 1 lớp như các methods, fields, properties, events được gọi chung là thành phần của lớp(class members).
- ✓Các access modifier thường đứng trước khai báo class, struct, record và các thành phần của struct, class.
- ✓Vì sự phức tạp của việc sử dụng access modifier nên tại thời điểm hiện tại ta mặc định sử dụng access modifier public. Những access modifier khác sẽ lần lượt tìm hiểu sau.
- ✓Nếu một kiểu khai báo ngay trong namespace và không chỉ định access modifier thì mặc định nó có access modifier là internal.
- ✓Các thành phần của lớp có thể khai báo bất kì access modifier nào đã kể ở trên. Mặc định nếu không chỉ định, chúng sẽ có access modifier là private.
- ✓Thành phần bên trong struct không thể khai báo access modifier là protected, protected internal, private protected vì nó không hỗ trợ kế thừa.

Constructor

- ✓Constructors: phương thức khởi tạo là một phương thức đặc biệt sử dụng với mục đích duy nhất là khởi tạo thông tin cho đối tượng của lớp.
- ✓Constructor luôn cùng tên với tên lớp và không có kiểu trả về, thường có access modifier là public.
- ✓Mặc định mỗi lớp có sẵn một constructor mặc định không tham số. Tại đây nó sẽ khởi tạo các thuộc tính, trường dữ liệu bằng giá trị mặc định của kiểu tương ứng mà thành phần dữ liệu được khai báo.
- ✓Một lớp có thể có 1 hoặc nhiều constructor.

```
1 reference
public class Student
{
    // các phương thức khởi tạo
    0 references
    public Student(string id)
    {
        Id = id;
    }
```

Các trường dữ liệu, thuộc tính

- ✓Đây là phần dữ liệu của lớp.
- ✓Nó dùng để mô tả trạng thái của đối tượng.
- ✓Một trường dữ liệu(field) là một biến khai báo bên trong lớp.
- ✓Một thuộc tính(property) là một phương thức đặc biệt cho phép tạo lập trường dữ liệu private tương ứng(với auto implement properties) và cho phép trả về cũng như gán giá trị cho trường dữ liệu được chỉ định.

```
// các thuộc tính
1 reference
public string Id { get; set; }
0 references
public string FullName { get; set; }
0 references
public int Age { get; set; }
0 references
public string Major { get; set; }
0 references
public float Gpa { get; set; }
// các trường
public string rank;
```

Các phương thức khác

- ✓Các phương thức(methods) khác khai báo trong class thường là các hành động liên quan đến dữ liệu(properties or fields) của lớp.
- ✓Ví dụ các hành động của một tài khoản ngân hàng:

```
1 reference
class BankAccount
{
    0 references
    public int Id { get; set; }
    2 references
    public long Balance { get; set; }
    0 references
    public string Owner { get; set; }
    0 references
    public string BankName { get; set; }
    // các hành động (methods):
    // hành động rút tiền
    0 references
    public long Withdraw(long amount)...
    // hành động chuyển tiền
    0 references
    public long BankTransfer(BankAccount other, long amount)...
    // hành động nạp tiền vào tk
    0 references
    public long Deposit(long amount) ...
}
```

Truy cập vào các thành phần của đối tượng

- ✓Khi đã có định nghĩa lớp, ta tạo đối tượng bằng cách gọi new đến các phương thức khởi tạo phù hợp và truyền vào đó các đối số tương ứng.
- ✓Mọi đối tượng của lớp phải được tạo trước khi có thể sử dụng.
- ✓Khi ta gán biến x = new Type(); bản chất là ta gán địa chỉ của một đối tượng mới kiểu Type cho x. Không phải x là một đối tượng mới được tạo ra. Do đó x chỉ chứa tham chiếu tới một đối tượng.
- ✓Khi ta gán một biến x cho một biến y cùng kiểu, x và y sẽ cùng tham chiếu tới một đối tượng. Do đó mọi thay đổi thực hiện bởi x hoặc y đều cùng ảnh hưởng đến đối tượng đang được tham chiếu.
- ✓Để truy cập tới một thành phần của đối tượng ta sử dụng cú pháp: **object.Component;**
- ✓Trong đó:
  - ✓ object là đối tượng cần thao tác.
  - ✓ Component là thành phần cần gọi. Có thể là methods, properties, fields, events.

Ví dụ minh họa

```
2 references
public class Student
{
    // các phương thức khởi tạo
    0 references
    public Student(string id)
    {
        Id = id;
    }

    0 references
    public Student(string id, string fullName, int age)
    {
        Id = id;
        FullName = fullName;
        Age = age;
    }
    // các thuộc tính
    2 references
    public string Id { get; set; }
    1 reference
    public string FullName { get; set; }
    1 reference
    public int Age { get; set; }
    0 references
    public string Major { get; set; }
    0 references
    public float Gpa { get; set; }
    // các trường
    public string rank;

    // các phương thức
    0 references
    public void CalculateRank()...
}
```

Ví dụ minh họa

✓Lời gọi và kết quả:

```
0 references
static void Main()
{
    Student nam = new Student("SV1001", "Tran Van Nam", 20);
    // truy cập vào họ và tên:
    Console.WriteLine("Ho ten: " + nam.FullName);
    nam.Gpa = 3.25f; // gán điểm TB
    nam.Major = "CNTT"; // gán chuyên ngành
    Console.WriteLine("Diem TB: " + nam.Gpa);
    Console.WriteLine("Chuyen nganh: " + nam.Major);
    Student nam2 = nam; // gán nam cho nam2
    nam2.Gpa = 3.66f; // gán lại điểm cho đối tượng
    Console.WriteLine("Diem sau cap nhat: " + nam.Gpa);
}
```

```
C:\WINDOWS\system32\cmd.exe
Ho ten: Tran Van Nam
Diem TB: 3.25
Chuyen nganh: CNTT
Diem sau cap nhat: 3.66
Press any key to continue
```

Nội dung tiếp theo

Tìm hiểu về phương thức khởi tạo