Bài 6.3: Lớp static và các thành phần static

- ✓ Đặc điểm và mục đích sử dụng
- ✓ Các thành phần static
- ✓ Phương thức khởi tạo static
- ✓ Import static
- ✓ Ví dụ minh họa
- ✓ Bài tập thực hành

Đặc điểm và mục đích sử dụng

- ✓Một lớp static là lớp có keyword static trước keyword class trong định nghĩa lớp.
- ✓Lớp static giống với lớp thông thường ngoại trừ việc ta không thể tạo đối tượng từ lớp static.
- ✓Mục đích sử dụng chính của lớp static là để tạo các lớp tiện ích cho phép thực hiện các chức năng chỉ dựa vào tham số đầu vào.
- ✓Khi truy cập các thành phần của lớp static từ bên ngoài lớp, ta sử dụng tên lớp và tên thành phần cần sử dụng phân tách nhau bởi dấu chấm. Ví dụ *Math.Abs(100);*
- ✓Ví dụ lớp Math chẳng hạn. Ta sử dụng các chức năng của lớp mà không cần tạo đối tượng.
- ✓Các đặc trưng của lớp static:
  - ✓ Chỉ chứa các thành phần static.
  - ✓ Không thể khởi tạo đối tượng.
  - ✓ Không thể kế thừa, không chứa phương thức khởi tạo non-static.

Các thành phần static

- ✓Một lớp không static có thể chứa các thành phần static là các phương thức, các trường, thuộc tính, sự kiện.
- ✓Các thành phần static có thể được gọi mà không cần đối tượng của lớp.
- ✓Cho dù có bao nhiêu đối tượng cụ thể của lớp được tạo ra thì chỉ có duy nhất 1 bản sao của các thành phần static tồn tại.
- ✓Phương thức và thuộc tính static không thể truy cập vào các thành phần non-static của lớp.
- ✓Ta thường tạo lớp non-static chứa một vài thành phần static thay vì cả class là static.
- ✓Thường sử dụng các thành phần static để đếm số lượng các đối tượng của lớp đã được tạo ra hoặc lưu trữ các giá trị dùng chung giữa tất cả các đối tượng của lớp.
- ✓Phương thức static có thể nạp chồng nhưng không thể bị ghi đè.
- ✓Các thành phần static được khởi tạo trước khi một thành phần static nào đó bị truy cập.

Các thành phần static

- ✓C# không hỗ trợ các biến cục bộ static. Ví dụ các biến static khai báo bên trong 1 phương thức sẽ bị coi là không hợp lệ.
- ✓Để khai báo thành phần của lớp là static, ta để keyword static ngay trước kiểu trả về của thành phần đó.

```
class Lesson63
{
    // phương thức static hiển thị thông điệp nhận được
    0 references
    static void ShowMessage(string msg)
    {
        Console.WriteLine("Message is: " + msg);
    }

    // phương thức static tính tổng 3 số nguyên int
    0 references
    static int Add(int a, int b, int c)
    {
        return a + b + c;
    }
}
```

Phương thức khởi tạo static

- ✓Là phương thức khởi tạo có keyword static trước tên constructor.
- ✓Sử dụng để khởi tạo các thành phần static trong lớp đó hoặc thực hiện các hành động chỉ cần thực hiện duy nhất 1 lần.
- ✓Static constructor tự động được gọi trước khi đối tượng đầu tiên của lớp đó được tạo hoặc trước khi bất kì một thành phần static nào bị truy cập.
- ✓Nếu trong một lớp của bạn mà có các thành phần static, bạn hãy cung cấp một static constructor để nó khởi tạo các thành phần static khi lớp đó được nạp vào chương trình.

Vài đặc trưng của static constructor

- ✓Static constructor không chứa tham số hay access modifier.
- ✓Mỗi struct hay class chỉ có thể có duy nhất 1 static constructor.
- ✓Static constructor không thể kế thừa hay nạp chồng.
- ✓Static constructor không thể gọi trực tiếp mà nó được gọi tự động bởi CLR.
- ✓Ta không thể kiểm soát được khi nào thì static constructor được gọi.
- ✓Static constructor khởi tạo lớp trước khi đối tượng đầu tiên của lớp được tạo ra. Static constructor chạy trước instance constructor.
- ✓Nếu ta không cung cấp static constructor, các thành phần dữ liệu static sẽ được khởi tạo giá trị mặc định của kiểu mà nó đang mang.

Ví dụ

```
10 references
class Employee
{
    0 references
    static Employee()
    {
        fine = 100000; // 100k
        AutoIncrementId = 10000; // mã nhân viên bắt đầu tăng từ 10000
    }
    3 references
    public Employee(string fullName, long salary)
    {
        EmpId = AutoIncrementId++; // gán mã tự tăng cho đối tượng mới
        FullName = fullName;
        Salary = salary;
    }

    4 references
    public static int AutoIncrementId { get; set; } // mã nhân viên tự tăng
    public static long fine; // tiền phạt
    1 reference
    public int EmpId { get; set; } // Mã nhân viên
    1 reference
    public string FullName { get; set; } // họ và tên
    1 reference
    public long Salary { get; set; } // mức lương
}
```

Ví dụ

```
0 references
static void Main()
{
    Console.WriteLine("Original AutoIncrementId: " + Employee.AutoIncrementId);
    Employee emp1 = new Employee("Hoang Thanh Trung", 15000000);
    Employee emp2 = new Employee("Truong Viet Hoang", 17000000);
    Employee emp3 = new Employee("Le Quynh Trang", 18000000);
    Console.WriteLine("After create 3 instance: " + Employee.AutoIncrementId);
}
```

```
ences
ss Employee
```

```
C:\WINDOWS\system32\cmd.exe
Original AutoIncrementId: 10000
After create 3 instance: 10003
Press any key to continue . . .
```

Import static

- ✓Từ C# 6, ta có thể import một lớp static sử dụng keyword using: **using static ClassName;**
- ✓Sau khi import, các thành phần trong lớp static đó có thể được sử dụng trực tiếp.
- ✓Ví dụ lớp Math, Console của namespace System:

```
using static System.Console;
using static System.Math;

namespace CSharpCourse
{
    0 references
    class Lesson63
    {
        0 references
        static void Main()
        {
            // sử dụng trực tiếp WriteLine thay vì Console.WriteLine
            WriteLine("Original AutoIncrementId: " + Employee.AutoIncrementId);
            Employee emp1 = new Employee("Hoang Thanh Trung", 15000000);
            Employee emp2 = new Employee("Truong Viet Hoang", 17000000);
            Employee emp3 = new Employee("Le Quynh Trang", 18000000);
            WriteLine("After create 3 instance: " + Employee.AutoIncrementId);
            WriteLine("Can bac hai cua 100 = " + Sqrt(100)); // thay vì Math.Sqrt()
        }
```

Nội dung tiếp theo

Tính đóng gói dữ liệu - Encapsulation