ENUM

Enum là gì

Kiểu định nghĩa sẵn và kiểu do người dùng định nghĩa

Trong tất cả các bài học trước đây bạn đều sử dụng các kiểu dữ liệu cơ bản C# **định nghĩa sẵn** như int, bool, string. Nó có nghĩa là bạn chỉ cần khai báo và khởi tạo biến thuộc kiểu dữ liệu đó để sử dụng.

Sử dụng các kiểu đã được định nghĩa sẵn dĩ nhiên là tiện lợi hơn nhưng đồng thời bạn cũng bị giới hạn. Các kiểu cơ sở định nghĩa sẵn đó không phải lúc nào cũng đáp ứng được yêu cầu của bạn.

Do đó, các ngôn ngữ lập trình đều cung cấp cho người lập trình khả năng **tự định nghĩa kiểu dữ liệu** của riêng mình.

Trong C# bạn có thể tự định nghĩa các kiểu dữ liệu riêng thuộc về một trong các nhóm sau: enumeration, structure, class, interface, delegate. Tất cả các kiểu định nghĩa sẵn của C# cũng rơi vào một trong các nhóm này.

Trong các kiểu dữ liệu cơ sở bạn đã biết thì các kiểu số (byte, int, long, double, v.v.), bool, char thuộc về nhóm structure; string và object thuộc về nhóm class.

Như vậy về bản chất, kiểu định nghĩa sẵn và kiểu tự định nghĩa không có gì khác biệt nhau. Chẳng qua một bên do đội ngũ phát triển C# làm ra, một bên do bạn tự làm. Sau khi định nghĩa kiểu, bạn có thể sử dụng nó như bất kỳ kiểu dữ liệu có sẵn nào.

Tuy nhiên, để định nghĩa kiểu, bạn phải học cú pháp định nghĩa (khai báo) riêng của từng nhóm.

Bài học này sẽ giúp bạn làm quen với cách định nghĩa kiểu dữ liệu của riêng mình thuộc về nhóm enumeration.

Khái niệm enum

Trước khi sờ vào enumeration, hãy cùng xem ví dụ sau.

Tên các ngày trong tuần theo tiếng Anh (Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday) là các giá trị cố định, hữu hạn. Khi cần lưu trữ và xử lý trong lập trình C#, bạn có thể sử dụng một biến thuộc kiểu string như thế này:

```
string dayOfWeek = "Sunday";
dayOfWeek ="Monday";
```

Một giải pháp khác thường được lập trình viên C sử dụng là các “magic constant”:

```
const int SUNDAY = 0, MONDAY = 1, TUESDAY = 2;
```

Những tình huống như trên dẫn tới nhu cầu xây dựng một kiểu dữ liệu giúp lưu trữ một danh sách hữu hạn hằng số, sao cho tại mỗi thời điểm, biến tương ứng chỉ có thể nhận một giá trị trong số đó.

C# cho phép định nghĩa kiểu dữ liệu để lưu trữ danh sách hằng số như vậy. Nó được gọi là *kiểu liệt kê* (enumeration).

Kiểu liệt kê là loại kiểu dữ liệu do người dùng định nghĩa chứa một danh sách (hữu hạn) hằng số. Mỗi hằng số được đặt một tên gọi tương ứng. Giá trị của hằng một số nguyên. Mỗi biến thuộc kiểu liệt kê tại mỗi thời điểm chỉ có thể nhận một giá trị trong danh sách.

Một số kiểu enum đã định nghĩa sẵn

C# đã định nghĩa sẵn một số kiểu liệt kê. Một vài trong số đó bạn đã từng sử dụng.

ConsoleColor

Khi học làm việc với console trong C# bạn đã gặp các lệnh như:

```
Console.ForegroundColor = ConsoleColor.Red; // đổi màu chữ sang đỏ
Console.ForegroundColor = ConsoleColor.Green; // đổi màu chữ sang xanh
```

`ConsoleColor` ở trên là một kiểu liệt kê được C# định nghĩa sẵn với 16 hằng số là các tên màu mà chúng ta có thể sử dụng để quy định màu chữ hoặc màu nền trong giao diện dòng lệnh. Các giá trị của ConsoleColor như sau:

Tên	Giá trị	Ý nghĩa
Black	0	The color black.
Blue	9	The color blue.
Cyan	11	The color cyan (blue-green).
DarkBlue	1	The color dark blue.
DarkCyan	3	The color dark cyan (dark blue-green).
DarkGray	8	The color dark gray.
DarkGreen	2	The color dark green.
DarkMagenta	5	The color dark magenta (dark purplish-red).
DarkRed	4	The color dark red.
DarkYellow	6	The color dark yellow (ochre).
Gray	7	The color gray.

Tên	Giá trị	Ý nghĩa
Green	10	The color green.
Magenta	13	The color magenta (purplish-red).
Red	12	The color red.
White	15	The color white.
Yellow	14	The color yellow.

ConsoleKey

Cũng trong bài học về console trong C# bạn gặp một enum khác là ConsoleKey, chứa danh sách các phím chuẩn dùng được trên console. Dưới đây là một số tên và giá trị của enum này.

Tên hằng	Giá trị	Ý nghĩa
A	65	The A key.
Add	107	The Add key (the addition key on the numeric keypad).
Applications	93	The Application key (Microsoft Natural Keyboard).
Attention	246	The ATTN key.
B	66	The B key.
Backspace	8	The BACKSPACE key.
BrowserBack	166	The Browser Back key (Windows 2000 or later).
BrowserFavorites	171	The Browser Favorites key (Windows 2000 or later).

Làm việc với enum trong C

Khi bạn đã hiểu được khái niệm enum, giờ là lúc tự tạo cho mình các enum riêng.

Khai báo kiểu enum

Trong C# kiểu enum được định nghĩa với từ khóa `enum`:

```
enum DayOfWeek { Sunday, Monday, Tuesday, Wednesday, Thursday, Friday, Saturday }
```

Ví dụ trên **khai báo kiểu** dữ liệu thuộc nhóm enumeration có tên là `DayOfWeek` với các giá trị (hằng) tương ứng là `DayOfWeek.Sunday`, `DayOfWeek.Monday`, v.v..

Về mặt ý nghĩa, `DayOfWeek.Sunday` hoàn toàn tương đương với việc sử dụng "magic constant" `SUNDAY = 0`, nghĩa là `DayOfWeek.Sunday` thực chất là một hằng số có giá trị 0. Tuy nhiên, sử dụng `enum` giúp quản lý code đơn giản tiện lợi hơn.

Rất rõ ràng, enum trong C# chỉ là một tập hợp hữu hạn của các hằng số, với mỗi hằng số có giá trị thuộc kiểu số nguyên.

Giá trị số của mỗi hằng mặc định bắt đầu từ 0 và tăng dần theo thứ tự của hằng số trong danh sách. Nghĩa là, `Sunday = 0`, `Monday = 1`, `Tuesday = 2`, v.v.. Nếu Sunday được gán giá trị khác 0, ví dụ `Sunday = 1`, thì `Monday` sẽ nhận giá trị 2, `Tuesday` nhận giá trị 3, v.v..

Mặc định, giá trị của hằng trong enum sẽ thuộc kiểu **int**. Tuy nhiên bạn có thể thay đổi thành kiểu số nguyên khác. Ví dụ:

```
enum Color : byte
{
    Red = 1,
    Green = 2,
    Blue = 3
}
```

Trong định nghĩa kiểu Color này, giá trị của mỗi hằng số giờ thuộc kiểu byte, thay vì int. Và chúng ta cũng chủ động gán giá trị cho nó, thay vì để compiler làm tự động.

Các enum khai báo như trên chỉ có thể **sử dụng nội bộ** trong phạm vi project. Nếu bạn muốn tạo thư viện để người khác có thể sử dụng, enum cần được khai báo với từ khóa điều khiển truy cập `public`:

```
public enum Color : byte
{
    Red = 1,
    Green = 2,
    Blue = 3
}
```

Như bạn đã biết trong bài học về kiểu dữ liệu cơ sở, tất cả enum đều thuộc nhóm value type, kể cả các enum bạn tự định nghĩa.

Khai báo và khởi tạo biến enum

Với enum `DayOfWeek` được định nghĩa như trên chúng ta có thể dễ dàng **khai báo và khởi tạo biến** thuộc kiểu DayOfWeek:

```
DayOfWeek day1 = DayOfWeek.Sunday, day2 = DayOfWeek.Tuesday;
var day3 = DayOfWeek.Monday;
var weekend = DayOfWeek.Saturday;
```

Nếu dùng kiểu Color định nghĩa như ở trên thì bạn có thể khai báo biến như sau:

```
var red = Color.Red;
var green = Color.Green;
var blue = Color.Blue;
```

Nó không khác gì với việc sử dụng các enum "xịn" của C# như ConsoleColor:

```
var red = ConsoleColor.Red;
var green = ConsoleColor.Green;
var blue = ConsoleColor.Blue;
```

Sử dụng biến enum

Để minh họa cách sử dụng biến enum, chúng ta cùng làm một project nhỏ.

Tạo một blank solution đặt tên là S05_Enum. Thêm project mới thuộc kiểu ConsoleApp vào solution và đặt tên project là P01_EnumVar. Viết code cho file Program.cs như sau:

```
namespace P01_EnumVar
{
    using System;
    using static System.Console;
    class Program
    {
        static void Main(string[] args)
        {
            // In biến enum ra console
            var gender = Gender.Male;
            WriteLine($"My gender is {gender} ({(int) gender})");
            var day = DayOfWeek.Tuesday;
            WriteLine($"Today is {day} ({(int)day})");
            var color = Color.Blue;
            WriteLine($"My favorite color is {color} ({(int)color})");
            var month = Month.Aug;
            WriteLine($"My birth month is {month} ({(int)month})");
            WriteLine("--------------------------");
            // Chuyển đổi từ số sang enum
            gender = (Gender)1;
            WriteLine($"My gender is {gender} ({(int)gender})");
            month = (Month)8;
            WriteLine($"My birth month is {month} ({(int)month})");
            WriteLine("--------------------------");
            // Sử dụng các phương thức của lớp Enum
            // GetNames trả về danh sách tên hằng
            foreach (var d in Enum.GetNames(typeof(DayOfWeek)))
            {
                Write($"{d} ");
            }
            WriteLine();
            // GetValues trả về danh sách hằng (bao gồm cả tên và giá trị)
```

```
            foreach (var d in Enum.GetValues(typeof(DayOfWeek)))
            {
                Write($"{(int) d} ");
            }
            WriteLine();

            WriteLine("-------------------------");
            // Đọc giá trị số từ bàn phím và chuyển thành kiểu enum sử dụng lớp Enum
            Write("What is your gender? ");
            // đọc một số từ bàn phím (0, 1, 2) và chuyển thành kiểu Gender
            gender = (Gender) Enum.Parse(typeof(Gender), ReadLine());
            WriteLine($"Your gender is {gender}");
            if (gender == Gender.Unknown)
                WriteLine("Sorry!");
            Write("What is your favorite color? ");
            // đọc một số (1, 2 hoặc 3) và chuyển thành kiểu Color
            color = (Color)Enum.Parse(typeof(Color), ReadLine());
            WriteLine($"Your favorite color is {color}");
            Write("What is your birth month? ");
            // đọc một số (từ 1 đến 12) và chuyển thành kiểu Month
            month = (Month)Enum.Parse(typeof(Month), ReadLine());
            switch (month)
            {
                case Month.Feb:
                case Month.Mar:
                case Month.Apr:
                    WriteLine("You're born in Spring!");
                    break;
                case Month.May:
                case Month.Jun:
                case Month.Jul:
                    WriteLine("You're born in Summer!");
                    break;
                case Month.Aug:
                case Month.Sep:
                case Month.Oct:
                    WriteLine("You're born in Autumn!");
                    break;
                case Month.Nov:
                case Month.Dec:
                case Month.Jan:
                    WriteLine("You're born in Winter!");
                    break;
            }
            ReadKey();
        }
    }
    /// <summary>
    /// Enum chứa danh sách giới tính
    /// </summary>
    enum Gender
```

```
    {
        Male, Female, Unknown
    }
    /// <summary>
    /// Enum chứa danh sách ngày trong tuần
    /// </summary>
    enum DayOfWeek
    {
        Monday = 2, // hằng Monday có giá trị bằng 2
        Tuesday = 3,
        Wednesday = 4,
        Thursday = 5,
        Friday = 6,
        Saturday = 7,
        Sunday = 8
    }
    /// <summary>
    /// Enum chứa 3 hằng số màu kiểu byte
    /// </summary>
    enum Color : byte
    {
        Red = 1,
        Green = 2,
        Blue = 3
    }
    /// <summary>
    /// Enum chứa tên các tháng, giá trị bắt đầu từ 1
    /// </summary>
    enum Month
    {
        Jan = 1, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec
    }
}
```

Ví dụ trên đã trình bày hầu hết các vấn đề cơ bản khi sử dụng enum trong chương trình C#. Hãy tự code lại ví dụ để nắm chắc cách sử dụng enum. Nếu gặp khó khăn, bạn có thể tải mã nguồn trong link ở cuối bài hoặc hỏi trong comment.

`Enum` (chữ E viết hoa) là một class đặc biệt trong C#. Class này cung cấp một số phương thức để làm việc với bất kỳ kiểu enum nào. Trong ví dụ trên bạn đã nhìn thấy và sử dụng các phương thức tĩnh GetNames, GetValues, Parse.