View danh sách: kiểu mảng, cấu trúc lặp

Trong hai bài trước chúng ta đã xây dựng các lớp view để hiển thị một cuốn sách, nhập dữ liệu cho một cuốn sách, và cập nhật thông tin của một cuốn sách.

Bởi vì chúng ta phải quản lý nhiều cuốn sách điện tử, chúng ta sẽ phải tiếp tục xây dựng một class mới để hiển thị một danh sách các cuốn sách điện tử.

Qua bài này chúng ta sẽ làm việc với dữ liệu kiểu mảng, và các cấu trúc lặp (while, do-while, for, foreach).

NỘI DUNG CỦA BÀI [Ẩn]

Thực hành 1: xây dựng lớp view để hiển thị danh sách Book

Tạo file mã nguồn mới BookListView.cs cho lớp `BookListView` và viết code như sau:

```
using System;
using Framework;

namespace BookMan.ConsoleApp.Views
{
    using Models;

    /// <summary>
    /// class để hiển thị danh sách Book
    /// </summary>
    internal class BookListView
    {
        protected Book[] Model; // mảng của các object kiểu Book

        /// <summary>
        /// hàm tạo
        /// </summary>
        /// <param name="model">danh sách object kiểu Book</param>
        public BookListView(Book[] model)
        {
            Model = model;
        }

        /// <summary>
        /// in danh sách ra console
        /// </summary>
        public void Render()
        {
```

```
            if (Model.Length == 0)
            {
                ViewHelp.WriteLine("No book found!", ConsoleColor.Yellow);
                return;
            }
            ViewHelp.WriteLine("THE BOOK LIST", ConsoleColor.Green);
            int i = 0;
            while (i < Model.Length)
            {
                ViewHelp.Write($"[{Model[i].Id}]", ConsoleColor.Yellow);
                ViewHelp.WriteLine($" {Model[i].Title}", Model[i].Reading ? ConsoleColor
                i++;
            }
        }
    }
}
```

Ở đây bạn khai báo một biến mảng một chiều: Book[] model. Mỗi phần tử của mảng thuộc kiểu Book.

Bạn cũng dùng đến cấu trúc điều khiển lặp để duyệt mảng này.

Sử dụng các cấu trúc lặp

Trong phần thực hành 1 chúng ta thấy rằng để làm việc với dữ liệu mảng bắt buộc phải có một cấu trúc giúp chúng ta lần lượt làm việc với từng phần tử của mảng. Ứng với mỗi phần tử sẽ cùng áp dụng chung một nhóm lệnh. Để thực hiện yêu cầu đó cần sử dụng một trong các cấu trúc lặp.

C# cung cấp 4 cấu trúc lặp khác nhau: do-while, while, for, foreach.

Cấu trúc while

Trong phương thức Render ở trên chúng ta đang sử dụng cấu trúc **while**.

```
int i = 0;
while (i < Model.Length)
{
    ViewHelp.Write($"[{Model[i].Id}]" , ConsoleColor.Yellow);
    ViewHelp.WriteLine($" {Model[i].Title}", Model[i].Reading ? ConsoleColor.Cyan : Cons
    i++;
}
```

Trong cấu trúc *while*, danh sách lệnh có thể không được thực hiện lần nào. Tình huống này xảy ra khi biểu thức logic nhận giá trị false ngay từ đầu.

Cấu trúc do-while

Chúng ta có thể viết lại thân phương thức Render với vòng lặp *do-while* như sau:

```
int i = 0;
do
{
    ViewHelp.Write($"[{Model[i].Id}]" , ConsoleColor.Yellow);
    ViewHelp.WriteLine($" {Model[i].Title}", Model[i].Reading ? ConsoleColor.Cyan : Cons
    i++;
} while (i < Model.Length);
```

Cấu trúc *do-while* khác biệt với *while* ở chỗ, danh sách lệnh sẽ được thực hiện trước, sau đó mới kiểm tra giá trị của biểu thức logic.

Khi sử dụng cấu trúc *do-while*, danh sách lệnh luôn luôn thực hiện ít nhất một lần. Do đó, cần lưu ý trong trường hợp mảng rỗng (chưa có phần tử nào), truy cập vào phần tử của mảng rỗng sẽ gây ra lỗi.

Trong trường hợp phương thức Render sử dụng *do-while* như trên, nếu mảng Model rỗng thì sẽ gây ra lỗi.

Khi duyệt mảng với *while* hoặc *do-while*, chúng ta đều phải tự khai báo một *biến điều khiển* và gán cho nó giá trị 0, là giá trị chỉ số bắt đầu mặc định của mảng trong C#. Trong thân vòng lặp, chúng ta phải tự tăng giá trị của biến điều khiển thêm một đơn vị, và

1. Nếu dùng vòng lặp *while*, giá trị của biến điều khiển sẽ được so sánh với độ dài của mảng trước. Nếu biến điều khiển nhỏ hơn độ dài mảng, lệnh trong thân vòng lặp sẽ được thực hiện. Ngược lại, vòng lặp sẽ kết thúc.
2. Nếu dùng vòng lặp *do-while*, lệnh trong thân vòng lặp sẽ luôn thực hiện trước, sau đó mới kiểm tra xem biến điều khiển có nhỏ hơn độ dài mảng hay không. Nếu biến điều khiển vẫn nhỏ hơn độ dài mảng, một chu kỳ mới sẽ bắt đầu. Ngược lại, vòng lặp sẽ kết thúc. Vì lý do này, nếu mảng rỗng, trong thân vòng lặp vẫn cố gắng truy cập vào phần tử số 0, vốn không tồn tại, và gây lỗi.

Cấu trúc for

Thân phương thức `Render` có thể viết lại với vòng lặp for như sau:

```
for(int i = 0; i < Model.Length; i++)
{
    ViewHelp.WriteLine($"[{Model[i].Id}]" , ConsoleColor.Yellow);
    ViewHelp.WriteLine($" {Model[i].Title}", Model[i].Reading ? ConsoleColor.Cyan : Cons
}
```

Cấu trúc này sẽ thực hiện danh sách lệnh một số lần xác định (trong khi hai cấu trúc trên không xác định được số lần thực hiện).

Trong cấu trúc for, biến điều khiển, cách thay đổi giá trị của biến điều khiển cũng như điều kiện kiểm tra biến điều khiển đều viết chung trong khai báo. C# sẽ tự thay đổi giá trị biến điều khiển theo công thức chúng ta cung cấp.

Cấu trúc for đặc biệt phù hợp để duyệt các phần tử mảng.

Cấu trúc foreach

Thân phương thức `Render` có thể viết lại với vòng lặp `foreach` như sau:

```
foreach(Book b in Model)
{
    ViewHelp.Write($"[{b.Id}]", ConsoleColor.Yellow);
    ViewHelp.WriteLine($" {b.Title}", b.Reading ? ConsoleColor.Cyan : ConsoleColor.White
}
```

Đây là một cấu trúc riêng của C#, trong đó C# sẽ tự động duyệt qua danh sách phần tử của tập hợp. Giá trị của mỗi phần tử sẽ lần lượt đưa vào biến và bắt đầu một chu kỳ. Biến này có thể được sử dụng trong thân vòng lặp.

Trong cấu trúc này, nếu tập hợp có bao nhiêu phần tử thì sẽ danh sách lệnh sẽ được thực hiện chừng ấy lần.

Đây là loại cấu trúc lặp an toàn và ngắn gọn nhất trong C#.

Khi sử dụng cấu trúc này, C# sẽ tự duyệt qua mảng Model. Với mỗi phần tử có trong Model, thân vòng lặp (lệnh WriteLine) sẽ được thực hiện một lần. Giá trị của phần tử đó sẽ chuyển sang biến b và có thể được sử dụng trong thân vòng lặp.

Cấu trúc này loại bỏ việc sử dụng phép toán chỉ số, vốn nguy hiểm nếu như tập rỗng, cũng như giảm số lượng code cần viết.

Thực hành 2: khai báo và khởi tạo mảng của các object kiểu Book trong controller

Bước 1. Bổ sung phương thức vào lớp BookController

Bổ sung phương thức List như dưới đây vào lớp BookController:

```
/// <summary>
/// kích hoạt chức năng hiển thị danh sách
/// </summary>
public void List()
{
    /* khai báo và khởi tạo một mảng, mỗi phần tử thuộc kiểu Book.
    * Lệnh dưới dây khai báo và khởi tạo 1 mảng gồm 6 phần tử,
    * mỗi phần tử thuộc kiểu Book.
    * Do Book là class, mỗi phần tử của mảng cũng phải được khởi tạo
    * sử dụng từ khóa new, tương tự như khởi tạo một object bình thường
    */
    Book[] model = new Book[]
    {
        new Book{Id=1, Title = "A new book 1"},
        new Book{Id=2, Title = "A new book 2"},
        new Book{Id=3, Title = "A new book 3"},
        new Book{Id=4, Title = "A new book 4"},
        new Book{Id=5, Title = "A new book 5"},
        new Book{Id=6, Title = "A new book 6"},
    };

    BookListView view = new BookListView(model);
    view.Render();
}
```

Bước 2. Điều chỉnh phương thức Main của lớp Program

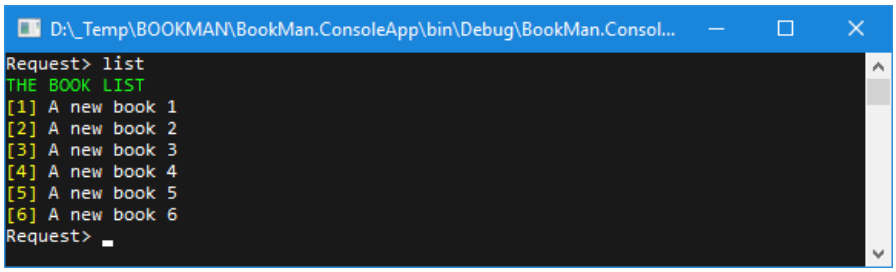
Bổ sung thêm một "case" nữa vào phương thức Main để chạy lệnh hiển thị danh sách:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```
namespace BookMan.ConsoleApp
{
    using Controllers;

    internal class Program
    {
        private static void Main(string[] args)
        {
            BookController controller = new BookController();

            while (true)
            {
                Console.Write("Request> ");
                string request = Console.ReadLine();

                switch (request.ToLower())
                {
                    case "single":
                        controller.Single(1);
                        break;

                    case "create":
                        controller.Create();
                        break;

                    case "update":
                        controller.Update(1);
                        break;

                    case "list":
                        controller.List();
                        break;

                    default:
                        Console.WriteLine("Unknown command");
                        break;
                }
            }
        }
    }
}
```

Bước 3. Dịch và chạy thử chương trình

Thử nghiệm lệnh list.



Kết quả thực hiện chương trình

Kết luận

Trong bài này chúng ta sử dụng dữ liệu kiểu mảng và các cấu trúc lặp dùng để duyệt các phần tử của mảng. Chúng ta cũng xây dựng được một class mới giúp hiển thị một danh sách các cuốn sách ra console.

+ Nếu bạn thấy site hữu ích, trước khi rời đi hãy **giúp đỡ** site bằng một hành động nhỏ để site có thể phát triển và phục vụ bạn tốt hơn.
+ Nếu bạn thấy bài viết hữu ích, hãy giúp **chia sẻ** tới mọi người.
+ Nếu có thắc mắc hoặc cần trao đổi thêm, mời bạn viết trong phần **thảo luận** cuối trang.
Cảm ơn bạn!