Controller, nối view – model: khởi tạo object, sử dụng object

Trong bài này, chúng ta sẽ xây dựng một class giúp ghép nối dữ liệu một cuốn sách điện tử (lớp `Book`) với lớp chuyên dùng để hiển thị một cuốn sách riêng rẽ (lớp `BookSingleView`). Qua đó chúng ta sẽ áp dụng cách khởi tạo object và truy xuất các thành viên của object.

Trong các bài trước, chúng ta đã lần lượt xây dựng hai class, một để mô tả dữ liệu (`Book`), một để mô tả cách hiển thị dữ liệu (`BookSingleView`), với mục đích tách rời dữ liệu và giao diện theo nguyên tắc của kiến trúc MVC.

Theo kiến trúc MVC, để ghép nối dữ liệu với giao diện, chúng ta cần đến một class trung gian: controller (lớp điều khiển).

Lớp controller có nhiệm vụ:

1. lấy dữ liệu và định hình dữ liệu cho phù hợp với yêu cầu của view;
2. khởi tạo view và cung cấp dữ liệu cho view.

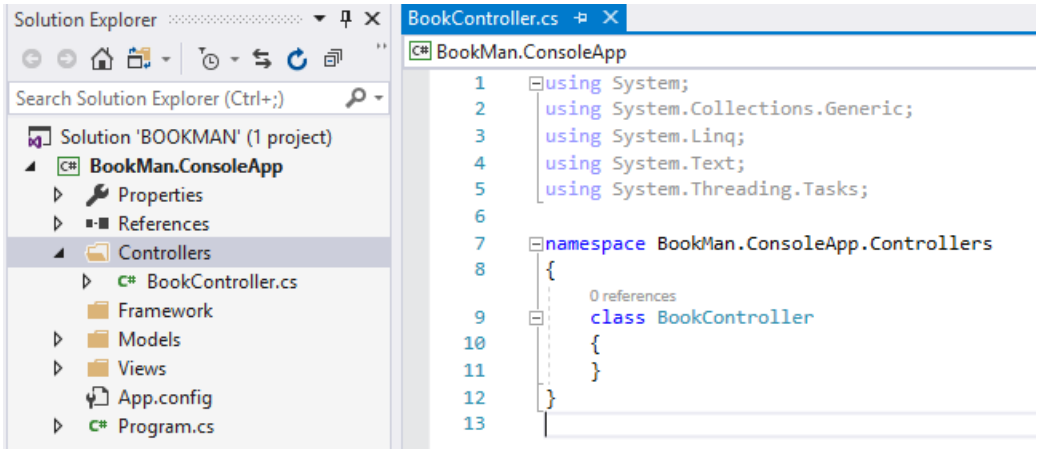
Như vậy controller phụ thuộc vào view và model, và do đó, được xây dựng sau hai loại class trên.

NỘI DUNG CỦA BÀI [Ẩn]

Thực hành 1: xây dựng lớp controller

Tạo lớp BookController

Tạo file mã nguồn mới BookController.cs để chứa lớp `BookController` trong thư mục Controllers.



Viết code cho lớp BookController

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

namespace BookMan.ConsoleApp.Controllers
{
    using Models; //luu ý cách dùng using với không gian tên con
    using Views;
    /// <summary>
    /// lớp điều khiển, giúp ghép nối dữ liệu sách với giao diện
    /// </summary>
    class BookController
    {
        /// <summary>
        /// ghép nối dữ liệu 1 cuốn sách với giao diện hiển thị 1 cuốn sách
        /// </summary>
        /// <param name="id">mã định danh của cuốn sách</param>
        public void Single(int id)
        {
            Book model = new Book();
            // khởi tạo view
            BookSingleView view = new BookSingleView(model);
            // gọi phương thức Render để thực sự hiển thị ra màn hình
            view.Render();
        }
    }
}
```

Viết code cho phương thức Main

Nhập code như sau cho lớp Program (file Program.cs):

```
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();
            controller.Single(0);

            Console.ReadKey();
        }
    }
```

```
21.  }
```

Dịch và chạy thử chương trình

Dùng phím F5 hoặc tổ hợp Ctrl+F5. Kết quả chạy chương trình như sau:

Kết quả chạy chương trình

Trong phần thực hành trên chúng ta gặp các lệnh sau:

```
1.  Book model = new Book();
2.  BookSingleView view = new BookSingleView(model);
3.  BookController controller = new BookController();
```

Đây là cấu trúc C# dùng để *khởi tạo object* của class.

Để *khởi tạo object* trong C# sử dụng từ khóa `new` và lời gọi tới một trong số các hàm tạo của class. Như trong đoạn code dưới đây,

```
1.  new Book();
2.  new BookSingleView(model);
3.  new BookController();
```

là các lệnh khởi tạo object.

> Hàm tạo là bắt buộc khi định nghĩa class. Tuy nhiên chương trình dịch của C# có khả năng tự thêm một hàm tạo cho class nếu nó không nhìn thấy định nghĩa hàm tạo nào trong class. Loại hàm tạo này có tên gọi là hàm tạo mặc định (default constructor).

Khai báo object thường đi cùng khởi tạo nhưng không bắt buộc. Ví dụ, các lệnh

```
Book model = new Book();
BookSingleView view = new BookSingleView(model);
BookController controller = new BookController();
```

vừa thực hiện khai báo, vừa khởi tạo object.

Thực hành 2: khởi tạo object của class Book sử dụng object initializer

Điều chỉnh code của lớp BookController

Điều chỉnh phương thức `Single` của lớp `BookController` như sau:

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

namespace BookMan.ConsoleApp.Controllers
{
    using Models; //lưu ý cách dùng using với không gian tên con
    using Views;

    /// <summary>
    /// lớp điều khiển, giúp ghép nối dữ liệu sách với giao diện
    /// </summary>
    internal class BookController
    {
        /// <summary>
        /// ghép nối dữ liệu 1 cuốn sách với giao diện hiển thị 1 cuốn sách
        /// </summary>
        /// <param name="id">mã định danh của cuốn sách</param>
        public void Single(int id)
        {
            // khởi tạo object với property
            Book model = new Book
            {
                Id = 1,
                Authors = "Adam Freeman",
                Title = "Expert ASP.NET Web API 2 for MVC Developers (The Expert's Voice
                Publisher = "Apress",
                Year = 2014,
                Tags = "c#, asp.net, mvc",
                Description = "Expert insight and understanding of how to create, custom
                Rating = 5,
                Reading = true
            };

            // khởi tạo view
            BookSingleView view = new BookSingleView(model);
            // gọi phương thức Render để thực sự hiển thị ra màn hình
            view.Render();
        }


    }
}
```

Dịch và chạy thử chương trình

Trong phần thực hành trên chúng ta gặp lại cách khởi tạo object sử dụng *bộ khởi tạo* (object initializer). Cú pháp khởi tạo này sử dụng property và được đưa vào từ C# 3 (.NET framework 3.5).

```
Book model = new Book
{
    Id = 1,
    Authors = "Adam Freeman",
    ...
}
```

Để truy xuất thành viên của object chúng ta sử dụng phép toán "." với tên object. Trên thực tế, từ bài trước trong lớp `BookSingleView` chúng ta đã sử dụng phép toán này để truy xuất các thuộc tính của object `Model` thuộc kiểu `Book`:

```
Console.WriteLine($"Authors:     {Model.Authors}");
Console.WriteLine($"Title:       {Model.Title}");
Console.WriteLine($"Publisher:   {Model.Publisher}");
Console.WriteLine($"Year:        {Model.Year}");
Console.WriteLine($"Edition:     {Model.Edition}");
Console.WriteLine($"Isbn:        {Model.Isbn}");
Console.WriteLine($"Tags:        {Model.Tags}");
Console.WriteLine($"Description: {Model.Description}");
Console.WriteLine($"Rating:      {Model.Rating}");
Console.WriteLine($"Reading:     {Model.Reading}");
Console.WriteLine($"File:        {Model.File}");
Console.WriteLine($"File Name:   {Model.FileName}");
```

Truy xuất phương thức thành viên đơn giản là một lời gọi phương thức từ một object nào đó. Việc truy xuất phương thức thành viên cũng sử dụng cấu trúc tương tự:

```
BookSingleView view = new BookSingleView(model);
view.Render();

BookController controller = new BookController();
controller.Single(0);
```

Kết luận

Trong bài này chúng ta đã áp dụng khai báo và khởi tạo object của class. Chúng ta cũng vận dụng kỹ thuật này để xây dựng lớp controller giúp ghép nối dữ liệu và giao diện theo mô hình MVC.