Tìm kiếm

Trang chủ

- PHP
- Liên hệ

- HTML
- CSS
- JavaScript
- PHP
- Server
- Java
- Tri thức
- SQL
- Lập trình C# (C Sharp)

- Liên hệ

C# cơ bản .NET Core

Phần 1 - Nhập môn C#

§ 1 Cài đặt, chương trình C# đầu tiên § 2 Lưu ý khi sử dụng ví dụ với .NET 6 § 3 Biến, kiểu dữ liệu và nhập/xuất § 4 Toán tử số học và gán § 5 So sánh, logic và lệnh if, switch § 6 Vòng lặp for, while § 7 Cấu trúc mảng, dữ liệu mảng § 8 Phương thức - Method § 9 Chuyển số thành chữ § 10 Lớp - Class § 11 Phương thức khởi tạo § 12 Kiểu giá trị, tham chiếu § 13 Chuỗi ký tự § 14 Struct và Enum § 15 Tính kế thừa § 16 Namespace § 17 Partial, Nested § 18 Generic § 19 Kiểu vô danh và dynamic § 20 null và nullable § 21 Tính đa hình - abstract - interface

Phần 2 - C# nâng cao

§ 22 Phương thức - Delegate § 23 Biểu thức lambda § 24 Event § 25 Phương thức mở rộng § 26 Hàm hủy - Quá tải toán tử - thành viên tĩnh - indexer § 27 Ngoại lệ Exeption § 28 IDisposable - using § 29 File cơ bản § 30 FileStream § 31 Collection - List § 32 SortedList § 33 Queue / Stack § 34 Linkedlist § 35 Dictionary - HashSet § 36 ObservableCollection § 37 LINQ § 38 (Multithreading) async - bất đồng bộ § 39 Type § 40 Attribute Annotation § 41 DI Dependency Injection § 42 (Multithreading) Parallel § 43 Thư viện lớp

Phần 3 - Networking

§ 44 **(Networking)** Uri, Dns, Ping § 45 **(Networking)** HttpClient § 46 **(Networking)** HttpMessageHandler § 47 **(Networking)** HttpListener § 48 **(Networking)** Tcp TcpListenerr/TcpClient

Phần 4 - Database - CSDL

(ASP.NET RAZOR) Khởi tạo và Route (Bài trước)
(Bài tiếp) **(ASP.NET RAZOR)** Layout trong ASP.NET Core

Cú pháp trong trang Razor Page ASP.NET Core

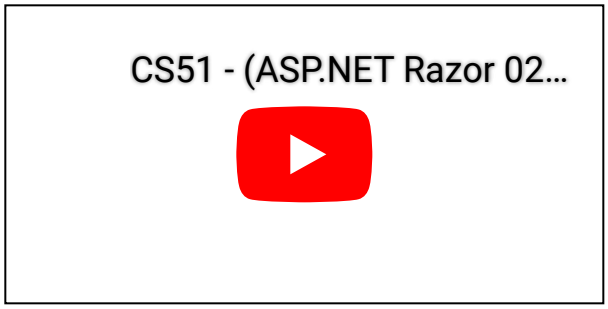
Viết các biểu thức cơ bản trong Razor, các khối lệnh trong Razor, sử dụng các cấu trúc điều khiển, vòng lặp và các chỉ thị trong Razor

Cú pháp trong Razor Page

Trong các bài trước bạn đã biết, Razor là một Template Engine - một cơ chế đọc nội dung trình bày trong văn bản ngôn ngữ đánh dấu .html, trong văn bản này có thể nhúng code C#. Mặc định thì nội dung HTML viết trong .cshtml không khác gì HTML trong các file .html thông thường. Tuy nhiên Razor hỗ trợ viết mã C# trong HTML, bằng cách sử dụng ký hiệu @ để chỉ thị chuyển đổi HTML thành C#, Razor sẽ kiểm tra biểu thức C# và dựng HTML tương ứng

Chỗ nào cần chỉ ra là code C# thì viết ký hiệu @ và tiếp đến là những từ khóa dành cho Razor như: page, namespace, functions, inherits, model, section hoặc các từ khóa C# như case, do, default, for, foreach, if, else, lock, switch, try, catch, finally, using, while

Để xuất ra ký tự HTML @ thì cần escape viết là @@.



Viết biểu thức Razor Page

Để viết biểu thức và xuất giá trị text của biểu thức (RENDER) bạn viết ngay sau ký hiệu @, biểu thức có trả về kết quả viết tường minh trong dấu () hoặc biểu một biểu thức ngầm định liên tục. Ví dụ:

```
<p class="text-danger">Sin góc @Math.PI radian là @Math.Sin(Math.PI/2)</p>
<p class="text-primary">Tổng quả 5 và 9 là @(5 + 9)</p>
```

Kết quả tương ứng là:

```
<p class="text-danger">Sin góc 3.141592653589793 radian là 1</p>
<p class="text-primary">Tổng quả 5 và 9 là 14</p>
```

Razor sinh HTML bằng cách thay thế các biểu thức sau @ thành giá trị

Lưu ý, các giá trị của biểu thức khi xuất ra đều được escaping

```
@("<p>Xin chào</p>")
```

Thì kết quả là:

```
&lt;p&gt;Xin chào&lt;/&gt;
```

Nghĩa là sẽ hiện thị ở trình duyệt: Xin chào

Nếu muốn xuất một giá trị mà không thực hiện escaping thì sử dụng hàm **Raw** trong **HtmlHelper**

```
@Html.Raw("<strong>Xin chào</strong>")
```

Khối lệnh Razor Page

Bạn có thể mở khối lệnh để viết các code C# trong khối lệnh này, khối lệnh viết trong {} sau ký hiệu @

```
@{
     .. Viết các lệnh C# ...
}
```

Lưu ý, khối lệnh thì không thực hiện Render (dựng HTML). Trong khối lệnh bạn có thể khai báo biến, thực hiện các câu lệnh, viết và gọi các hàm ... Bạn có thể mở ra nhiều khối lệnh, các biến, hàm từ khối lệnh trước vẫn tồn tại và có thể sử dụng lại bởi khối lệnh sau (cùng phạm vi)

```
@{
    int a = 10;
    int b = 11;

    int sum(int a, int b)
    {
```

```
        return a + b;
    }

}
<p>Tổng là: @sum(a,b)</p>
```

Xuất HTML trong khối code

Trong khối code, bạn có thẻ mở - đóng thẻ HTML thì Razor tự nhận biết đây bắt đầu nội dụng HTML, ví dụ

```
@{
    double sin45 = Math.Sin(Math.PI/4);
    <p>Sin góc 45 độ là @sin45 - Học C#</p>
}
```

Phần mở thẻ HTML ở trên là `<p>` sẽ tự động hiểu là HTML xuất ngay ra, vào trong nó lại có thể chèn biểu thức như phần trên (@sin45). Thậm chí việc trộn lẫn giữa khối lệnh và HTML lồng vào nhau phức tạp, ví dụ:

```
@{
    List<String> products = new List<string> {
        "Iphone",
        "Laptop",
        "CPU",
    };

    <h2>Danh sách shản phẩm</h2>
    <ul>
        @foreach (var product in products)
        {
            <li>Sản phẩm @product</li>
        }
    </ul>
}
```

Kết quả là:

```
<h2>Danh sách shản phẩm</h2>
<ul>
    <li>Sản phẩm Iphone</li>
    <li>Sản phẩm Laptop</li>
    <li>Sản phẩm CPU</li>
</ul>
```

Việc tự động nhận biết chuyển xuất HTML dễ dàng khi mở đóng thể HTML trong khối lệnh, tuy nhiên nếu muốn xuất ra một cách tường minh (kể cả không phải bắt đầu bằng thẻ HTML) thì trong khối lệnh dùng thẻ `<text>`

```
foreach (var product in products)
{
    <text> @product </text>
}
```

Có thể dùng ký hiệu `@:` cho biết phần sau của dòng là xuất HTML

```
foreach (var product in products)
{
   @:Sản phẩm @product
}
```

Sử dụng cấu trúc điều khiển, vòng lặp C# trong HTML Razor

Các cấu trúc điều khiển **if, switch**, các vòng lặp như **for, while ...** có thể bắt đầu sử dụng bất kỳ đâu trong nội dung HTML của .cshtml bằng cách sử dụng ký hiệu @ trước các từ khóa trên. Ví dụ điều khiển rẽ nhánh if

```
@{
    var age = 20;
}

@if (age >= 18) {
    @:Bạn được vào
}
else  {
    @:Bạn không được vào
}
```

Các chỉ thị directive trong Razor

Toàn bộ nội dung trang Razor (.cshtml) khi Razor Template Engine phân tích và dựng HTML thực chất nó tạo ra một lớp C# để biểu diễn trang này. Lớp này kế thừa từ lớp cơ sở **RazorPage<TModel>** . Do vậy bạn có thể sử dụng các chỉ thị để thêm vào thông tin của lớp C# sinh ra này.

Để xem kết quả lớp do Razor sinh ra, bạn build dự án và xem trong thư mục
`/obj/Debug/netcoreapp3.1/Razor/Pages`

@attribute

Để thêm thuộc tính mô tả (Xem attribute-annotation) vào lớp sinh ra bởi Razor, ví dụ:

```
@attribute [Authorize]
```

@functions

Dùng để thêm các trường, thuộc tính, phương thức vào lớp sinh ra bởi Razor

```
@functions{
    String name;
    int age {set; get;}

    String Welcome() {
        return $"Xin chào {this.name}, bạn {this.age} tuổi";
    }
}

@{
    this.name = "XuanThuLab";
    this.age = 20;
    <p>@Welcome()</p>
}
```

@implements và @inherits

Chỉ thị @implements dùng để khai báo lớp sinh ra bởi Razor triển khai một giao diện nào đó. Chỉ thị @inherits để cho biết kế thừa từ một lớp nào đó

```
@implements IDisposable
@inherits ClassAbc
```

@model

Chỉ thị @model để xác định kiểu dữ liệu được truyền đến Razor Page trong thuộc tính `Model` của lớp Razor sinh ra.

```
@model Kiểu_Model
```

@namespace

Chỉ thị @namespace để khai báo namespace cho lớp sinh ra bởi Razor.

```
@namespace Your.Namespace.Here
```

Lưu ý trong Razor Page mỗi Page đều nạp `Pages/_ViewImports.cshtml`, trong file này có khai báo namespace cho tất cả các Page.

@page

Chỉ thị này cho biết file .cshtml là một Razor Page (khác với View sau này trong MVC).

@section

Chỉ thị này kích hoạt một khối View, Page được được sử dụng để xuất HTML (render) bằng lệnh `@RenderSection(tên_section)` ở chỗ khác

Ví dụ định nghĩa .cshtml

```
@section MySection {
    <p>Xin Chào Các Bạn</p>
}
```

Trong `_Layout.cshtml` có thể gọi

```
@RenderSection("MySection", false)
```

@using

Chỉ thị @using để khai báo sử dụng namespace trong lớp sinh ra bởi Razor

```
@using System.Text
```

1 bình luận Sắp xếp theo **Mới nhất**

Viết bình luận...

Tat Tien

thanks

Thích · Phản hồi · 17 tuần

Plugin bình luận trên Facebook

Top-level statement trong lập trình C# .NET 6 Tạo các Requirement và Authorization handler chứng thực quyền truy cập Authorize trong ASP.NET Core (ASP.NET Core MVC) Triển khai ứng dụng ASP.NET trên Server Linux với Kestrel Apache Nginx (ASP.NET Core MVC) Giới thiệu một số admin template và tích hợp SB Admin (ASP.NET Core MVC) Tích hợp trình quản lý file vào website (ASP.NET Razor) Giới thiệu Razor Page và Route trong Razor Page Tạo thư viện C# NET Core và chia sẻ lên nuget.org Sử dụng Attribute Annotation trong lập trình C# CSharp (Networking) Sử dụng HttpMessageHandler cho HttpClient trong C# CSharp (Networking) Sử dụng HttpClient trong C# tạo các truy vấn HTTP
(ASP.NET RAZOR) Khởi tạo và Route (Bài trước)
(Bài tiếp) **(ASP.NET RAZOR)** Layout trong ASP.NET Core
Giới thiệuPrivacyTừ điển Anh - ViệtChạy SQLRegExpCubic-bezierUnix timestampKý tự HTMLcalories, chỉ số BMRchỉ số khối cơ thể BMITạo QR CodeLịch vạn niên Liên hệ RSS

Đây là blog cá nhân, tôi ghi chép và chia sẻ những gì tôi học được ở đây về kiến thức lập trình PHP, Java, JavaScript, Android, C# ... và các kiến thức công nghệ khác
Developed by XuanThuLab