Partial Page trong ASP.NET Core

Partial Page, tạm dịch là *trang riêng phần* hoặc *trang cục bộ*, là loại trang Razor đặc biệt có thể được "nhúng" vào một trang khác.

Hãy tưởng tượng một giao diện web thông thường với header, footer, sidebar. Bạn có thể muốn tạo ra ba bộ phận độc lập tương ứng và lắp ghép chúng lại trong một trang layout. Điều này càng cần thiết nếu từng cấu phần này có thiết kế phức tạp, hoặc làm việc theo nhóm.

Partial page tạo ra khả năng tái sử dụng một khối code giao diện, cũng như khả năng tách một giao diện phức tạp ra các bộ phận cấu thành để xây dựng độc lập.

> Các vấn đề trình bày trong bài học này có thể áp dụng chung cho cả Razor Pages và MVC. Để đơn giản, các ví dụ được thể hiện với Razor Pages. Bạn có thể áp dụng nguyên vẹn với MVC.

NỘI DUNG CỦA BÀI [Ẩn]

Thực hành: chia layout thành cấu phần

Chúng ta bắt đầu bằng một bài thực hành nhỏ.

Bước 1. Tạo một project mới theo template Web Application.

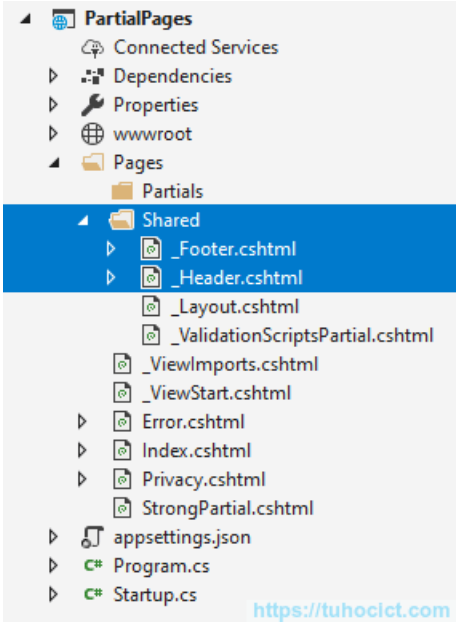
Mở file _Layout.cshtml (thư mục /Pages/Shared) bạn sẽ thấy code đại khái như sau:

Code này phân chia làm 3 phần rõ ràng: header, body và footer. Đây là một layout khá đơn giản. Các thành phần header và footer cũng rất đơn giản.

Bước 2. Tạo file _Header.cshtml trong thư mục Shared, sau đó cắt toàn bộ code phần header của _Layout.cshtml chuyển sang _Header.cshtml.

Bước 3. Tạo file _Footer.cshtml trong thư muc Shared và cũng di chuyển khối footer của _Layout.cshtml sang.

Chú ý, nếu trong _Header.cshtml và _Footer.cshtml có nôi dung gì thì xóa bỏ hết.



Bước 4. Ở vị trí của khối header và footer trong file layout thay thế bằng hai thẻ tương ứng:

```
<partial name="_Header" />
<div clas="container>
...
</div>
<partial name="_Footer" />
```

Giờ nếu chạy chương trình, layout và toàn site vẫn hoạt động như bình thường.

Sự khác biệt ở chỗ bạn đã tách phần header và footer của layout thành hai file riêng biệt, _Header.cshtml và _Footer.cshtml. Trong layout giờ bạn chỉ việc gọi thẻ <partial … /> để "ghép" hai phần riêng kia lại.

_Header.cshtml và _Footer.cshtml bạn vừa tạo ở trên được gọi là các **partial page** – trang cục bộ/ trang riêng phần.

Khái niệm Partial page

Partial page là một trang cshtml nhưng **không chứa** directive `@page` như thông thường. Trong partial page chỉ chứa markup và logic để hiển thị thông tin.

Do không chứa directive @page, partial page không tham gia vào cơ chế routing của Razor Pages. Nghĩa là không có URL nào tương ứng với một partial page. Và do vậy bạn cũng không thể trực tiếp truy xuất trang từ trình duyệt.

> Tuy nhiên bạn có thể truy xuất nó qua URL dựa trên truy vấn AJAX.

Chú ý, nếu bạn đặt directive @page vào đầu file partial page thì nó lại trở thành một trang thông thường.

Partial page chỉ có tác dụng khi được "nhúng" vào một trang khác. Rất dễ dàng hình dung partial page thực ra chỉ là một khối markup được tách ra đặt vào một file riêng để có thể xây dựng độc lập và tái sử dụng trong các trang.

Partial page cung cấp khả năng phân tách một giao diện web phức tạp ra các cấu phần đơn giản hơn để xây dựng độc lập và song song.

Partial cũng có thể có model class của riêng nó. Trong trương hợp này nó được gọi là strongly-typed partial page.

Lưu ý rằng, partial page chỉ nên được sử dụng cho các UI không có logic phức tạp. Nếu cần một khối UI độc lập có khả năng xử lý dữ liệu hay xử lý các logic phức tạp, bạn nên sử dụng view component.

Nói tóm lại, bạn nên nhìn nhận partial page là một khối HTML có thể tái sử dụng.

Sử dụng partial page

Có vài cách nhúng partial page vào trang Razor.

Phương pháp được khuyến khích sử dụng là sử dụng **partial tag helper** (như chúng ta đã từng sử dụng):

```
<partial name="_Header" />
<partial name="_Footer" />
```

Trong đó, giá trị của name là tên file tương ứng (bỏ đi phần mở rộng). Tên file *không phân biệt chữ hoa/thường*. Phương pháp này luôn hoạt động theo kiểu *bất đồng bộ*.

Phương pháp này chỉ dùng được trong ASP.NET Core 2.1 trở lên.

Cách thứ hai là sử dụng phương thức **Html.Partial**:

```
@Html.Partial("_Header")
@Html.Partial("_Footer")
```

Cách thứ ba là sử dụng phương thức Html.RenderPartial:

```
@{ Html.RenderPartial("_Header"); }
@{ Html.RenderPartial("_Footer"); }
```

Lưu ý sự khác biệt giữa Partial và RenderPartial là ở chỗ: Partial trả về chuỗi HTML, do đó nó được gọi như một *biểu thức Razor* `@Html.Partial("_Header")`; RenderPartial có kiểu trả về là void nên nó phải được gọi trong *khối code Razor* `@{ Html.RenderPartial("_Header"); }`.

Partial có phương án bất đồng bộ tương ứng là PartialAsync. RenderPartial có phương án bất đồng bộ tương ứng là RenderPartialAsync.

Trong tất cả các phương án trên, RenderPartialAsync và RenderPartial viết trực tiếp kết quả vào chuỗi phản hồi. Do vậy nó có hiệu suất cao nhất. Tuy nhiên sự khác biệt về hiệu suất thực tế là không đáng kể.

File và đường dẫn tới partial page

Razor Pages không đặt ra giới hạn gì với tên file partial page. Tuy nhiên người ta thường quy ước đặt tên có dấu gạch chân phía trước để dễ phân biệt nó với trang Razor thông thường.

Khi sử dụng partial page ở trên bạn đã thấy rằng không cần phải cung cấp đường dẫn đầy đủ tới file partial page. Chúng ta phân biệt hai tình huống.

Trường hợp 1. Nếu bạn cung cấp tên file partial page có phần mở rộng cshtml:

Razor Pages sẽ hiểu rằng bạn đang cung cấp đường dẫn tương đối tới file partial page tính từ thư mục chứa trang chính.

Ví dụ, nếu bạn gọi partial page "_LoginForm.cshtml" từ trang chính /Pages/Member/Register/Login.cshtml như sau: `<partial name="_LoginForm.cshtml" />`, Razor Pages sẽ hiểu rằng cần tìm file _LoginForm.cshtml trong thư mục /Pages/Member/Register. Nó không tìm ở bất kỳ chỗ nào khác nữa.

Nếu lời gọi của bạn là `<partial name="Form/_LoginForm.cshtml" />` thì Razor Pages chỉ tìm file _LoginForm.cshtml trong thư mục /Pages/Member/Register/Form.

Lưu ý, tuyệt đối không để ký tự / trước đường dẫn. Ví dụ, lời gọi `<partial name="/Form/_LoginForm.csthml" />` sẽ tìm kiếm file tương ứng trong thư mục /Form **trực thuộc dự án** (ngang cấp với /Pages).

Trường hợp 2. Nếu bạn cung cấp tên file partial page **không** có phần mở rộng cshtml:

Razor Pages tự động tìm kiếm file có tên tương ứng trong các thư mục mặc định theo trình tự sau:

(1) Tìm trong thư mục chứa page gọi.

Ví dụ, nếu trang chính của bạn là /Pages/Member/Register/Login.cshtml, và từ trang này bạn gọi partial page "LoginForm" (`<partial name="Login" />`), Razor Pages sẽ tìm kiếm file LoginForm.cshtml trong thư mục Register đầu tiên.

(2) Nếu không tìm thấy, Razor Pages bắt đầu tìm trong thư mục cha. Quá trình tiếp diễn cho đến thư mục Pages nếu vẫn không tìm thấy file tương ứng.

Tiếp ví dụ trên, nếu trong Register không có file cần tìm, Razor Pages sẽ tiếp tục tìm kiếm trong Member, rồi Pages.

(3) Nếu vẫn không tìm thấy file tương ứng, Razor Pages sẽ tìm kiếm tiếp trong thư mục /Pages/Shared.

/Pages/Shared là thư mục đặc biệt có tên gọi là "thư mục đã đăng ký" (registered location).

Bạn cũng có thể tự đăng ký thư mục có vai trò tương tự Shared qua phương thức ConfigureServices trong Startup như sau:

```
public void ConfigureServices(IServiceCollection services) {
    services
        .AddRazorPages()
        .AddRazorOptions(options => {
            options.PageViewLocationFormats.Add("/Pages/Partials/{0}.cshtml");
        });
}
```

Phương thức `PageViewLocationFormats.Add("/Pages/Partials/{0}.cshtml")` đăng ký thêm thư mục /Pages/Partials để Razor Pages tìm kiếm file cshtml.

Partial page và dữ liệu

Partial page cũng có thể có model class của riêng nó. Bạn dùng directive `@model` để chỉ định model class đứng sau partial page. Loại partial page có model class cũng được gọi là **strongly typed partial page**.

Model class cung cấp dữ liệu và khả năng xử lý logic cho partial page.

Hãy cùng thực hiện ví dụ sau:

Bước 1. Thêm trang **_UserList.cshtml** vào thư mục **/Pages/Shared** và viết code như sau:

```
@model List<(string firstName, string lastName)>

<div class="shadow p-3 mw-50">
    <p class="h3">US Presidents</p>
    <ul>
        @foreach (var p in Model) {
```

```
            <li>@p.firstName <b>@p.lastName</b></li>
        }
    </ul>
</div>
```

Directive @model chỉ định rằng model class của trang partial này có kiểu List<(string firstName, string lastName)>.

Đây là một danh sách của các tuple (string, string), trong đó phần tử đầu đặt tên là firstName, phần tử thứ hai đặt tên là lastName.

Như vậy bạn có cả một danh sách để xử lý (hiển thị) sử dụng thẻ <ul> (hoặc <ol>).

Bước 2. Thêm trang **StrongPartial.cshtml** vào thư mục **/Pages** và viết code như sau:

```
@page
@model IndexModel
@{
    var data = new List<(string, string)> {
        ("Donald", "Trump"),
        ("Barack", "Obama"),
        ("George W.", "Bush"),
        ("Bill", "Clinton"),
        ("George H. W.", "Bush"),
        ("Ronald", "Reagan"),
        ("Jimmy", "Carter"),
        ("Gerald", "Ford"),
        ("Richard", "Nixon"),
    };
}

<partial name="_UserList" model="data" />
```

Đây là chỗ bạn tạo dữ liệu, sử dụng trang partial _UserList và cung cấp dữ liệu cho nó.

Biến data là một danh sách các tuple (string, string).

Chúng ta sử dụng lối gọi trang partial thông qua markup <partial name=”…” />

Để ý rằng dữ liệu cung cấp cho trang partial thông qua attribute model=”…”. Giá trị của model có thể là một biểu thức.

Chạy ứng dụng và mở trang `/strongpartial` bạn thu được kết quả như sau:

Lưu ý, model class của partial page ***không thể xử lý truy vấn***. Tức là, các phương thức handler như OnGet, OnPost dù có thể viết trong model class của partial page nhưng chúng không có được vai trò của các handler.

Điều này liên quan đến việc partial page thực chất là một bộ phận của một trang (có directive @page). Truy vấn đi đến trang theo cơ chế routing và được quy định bởi @page. Partial page không tham gia vào cơ chế routing nên cũng không thể nhận truy vấn.

Bạn cũng không nên xây dựng các model class phức tạp cho partial page. Nhìn chung, partial page nên được sử dụng cho các khối UI không có logic phức tạp đứng sau. Do vậy, model class của partial page cũng chỉ nên đóng vai trò là dữ liệu sẽ được hiển thị.

> Nếu cần một khối UI để tái sử dụng có khả năng xử lý logic và dữ liệu (độc lập), bạn nên dùng view component.

Partial Page và AJAX

Partial page có ứng dụng đặc biệt khi kết hợp với AJAX và named handler.

> Nếu bạn chưa quen thuộc với AJAX:
> AJAX là viết tắt của Asynchronous JavaScript And XML. Đây là một loại kỹ thuật kết hợp XMLHttpRequest (của trình duyệt), JavaScript, và HTML DOM (Document Object Model) để tải một phần nội dung của trang mà không cần tải lại nguyên vẹn trang web.
> Ở đây chúng tôi sẽ không hướng dẫn cụ thể về AJAX mà chỉ đưa ra một ví dụ nhỏ.

Hãy cùng thực hiện một ví dụ nhỏ sau đây.

Bước 1. Tạo page mới AjaxPartial trong thư mục Pages.

Bước 2. Viết code cho model class AjaxPartialModel (file AjaxPartial.cshtml.cs) như sau:

```
using System.Collections.Generic;

using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;

namespace PartialPages.Pages {
    public class AjaxPartialModel : PageModel {
        public PartialViewResult OnGetPartial() {
            var data = new List<(string, string)> {
                ("Donald", "Trump"),
                ("Barack", "Obama"),
                ("George W.", "Bush"),
                ("Bill", "Clinton"),
                ("George H. W.", "Bush"),
                ("Ronald", "Reagan"),
                ("Jimmy", "Carter"),
                ("Gerald", "Ford"),
                ("Richard", "Nixon"),
            };
            return Partial("_UserList", data);
        }
    }
}
```

Như bạn có thể thấy, chúng ta đang tái sử dụng partial page _UserList đã tạo ra trong mục Partial Page và dữ liệu.

Trong class này chúng ta xây dựng named hander `OnGetPartial`. Để ý kiểu trả về của handler này là `PartialViewResult`. Lệnh return trả về chuỗi HTML tạo ra từ partial page _UserList và dữ liệu từ biến data.

Bước 3. Viết code cho AjaxPartial.cshtml như sau:

```
@page
@model PartialPages.Pages.AjaxPartialModel
@{
    ViewData["Title"] = "AjaxPartial";
}

<a id="btnLoad" class="btn btn-success mb-2">Press me to load the list of US presidents<
<div id="lstPresidents"></div>

@section Scripts{
    <script>
        $(function () {
            $('#btnLoad').on('click', function () {
                $('#lstPresidents').load('/ajaxpartial?handler=partial');
            });
        });
    </script>
}
```

Ở đây chúng ta sử dụng đoạn code JavaScript với hỗ trợ AJAX của thư viện JQuery. Có mấy lưu ý:

- JQuery được cài đặt mặc định cùng với template Web Application.
- Thư viện này đã được tham chiếu tới từ layout mặc định.
- Bạn phải đặt khối code sử dụng JQuery trong vùng @section Scripts { … }. Nếu không đặt trong vùng này, khối <script>…</script> của bạn sẽ xuất hiện trước đường link tới thư viện JQuery, vào do đó sẽ không thực hiện được.

Giờ khi chạy chương trình, nếu người dùng bấm vào nút link, danh sách tổng thống Mỹ (từ _UserList partial page) sẽ đổ vào vị trí thẻ mà trang sẽ không bị tải lại.

Kết luận

Qua bài học này bạn đã học được cơ chế tái sử dụng code sử dụng partial page. Partial page cho phép phân chia một giao diện phức tạp ra các cấu phần đơn giản hơn để xây dựng song song.

Lưu ý chỉ nên sử dụng partial page nếu khối giao diện không đòi hỏi logic và xử lý dữ liệu. Trong trường hợp ngược lại bạn nên sử dụng view component.

Tải mã nguồn Partial Page

1 file(s) 0.00 KB

TẢI MÃ NGUỒN

+ Nếu có thắc mắc hoặc cần trao đổi thêm, mời bạn viết trong phần **thảo luận** cuối trang.
Cảm ơn bạn!