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

§ 49 **(ADO.NET)** SqlConnection § 50 **(ADO.NET)** SqlCommand § 51 **(ADO.NET)** DataAdapter § 52 **(EF Core)** Tổng quan § 53 **(EF Core)** Tạo Model § 54 **(EF Core)** Fluent API § 55 **(EF Core)** Query § 56 **(EF Core)** Scaffold § 57 **(EF Core)** Migration

Phần 5 - ASP.NET CORE

§ 58 **(ASP.NET CORE)** Hello World! § 59 **(ASP.NET CORE)** Middleware § 60 **(ASP.NET CORE)** Map - Request - Response § 61 **(ASP.NET CORE)** IServiceCollection - MapWhen § 62 **(ASP.NET CORE)** Session - ISession § 63 **(ASP.NET CORE)** Configuration § 64 **(ASP.NET CORE)** Gửi Mail § 65 **(ASP.NET CORE)** SASS/SCSS § 66 **(ASP.NET CORE)** LibMan

Phần 6 - ASP.NET RAZOR

§ 67 **(ASP.NET RAZOR)** Khởi tạo và Route § 68 **(ASP.NET RAZOR)** Cú pháp Razor § 69 **(ASP.NET RAZOR)** Layout trong ASP.NET Core § 70 **(ASP.NET RAZOR)** Partial § 71 **(ASP.NET RAZOR)** ViewComponent § 72 **(ASP.NET RAZOR)** TagHelper § 73 **(ASP.NET RAZOR)** HtmlHelper § 74 **(ASP.NET RAZOR)** PageModel § 75 **(ASP.NET RAZOR)** Model Binding § 76 **(ASP.NET RAZOR)** HTML Form, Validation § 77 **(ASP.NET RAZOR)** Upload File § 78 **(ASP.NET RAZOR)** Entity Framework § 79 **(ASP.NET RAZOR)** Paging § 80 **(ASP.NET RAZOR)** Identity (1) - Register, Login, Logout § 81 **(ASP.NET RAZOR)** Identity (2) Lockout, Reset Password § 82 **(ASP.NET RAZOR)** Identity (3) Google Login § 83 **(ASP.NET RAZOR)** Identity (4) Facebook Login § 84 **(ASP.NET RAZOR)** Identity (5) profile, password, email ... § 85 **(ASP.NET RAZOR)** Identity (6) Role § 86 **(ASP.NET RAZOR)** Identity (7) Role-based Authorization § 87 **(ASP.NET RAZOR)** Identity (8) RoleClaim § 88 **(ASP.NET RAZOR)** Identity (9) Authorization Handler § 89 **(ASP.NET RAZOR)** IAuthorizationService

Phần 7 - ASP.NET MVC

§ 90 **(ASP.NET MVC)** Controller - View § 91 **(ASP.NET MVC)** Route § 92 **(ASP.NET MVC)** EF, Identity § 93 **(ASP.NET MVC)** Binding, Validation § 94 **(ASP.NET MVC)** Xây dựng Website(1) § 95 **(ASP.NET MVC)** Xây dựng Website(2) § 96 **(ASP.NET MVC)** Xây dựng Website(3) § 97 **(ASP.NET MVC)** Xây dựng Website(4) § 98 **(ASP.NET MVC)** Giỏ hàng - Cart (5) § 99 **(ASP.NET MVC)** elFinder (5) § 100 **(ASP.NET MVC)** SB Admin (6) § 101 **(ASP.NET MVC)** Kestrel, publish

- Lập trình PHP
  - PSR
  - Laminas
  - SPL
  - Xenforo

(Multithreading) Parallel (Bài trước)
(Bài tiếp) **(Networking)** Uri, Dns, Ping

Tạo thư viện C# NET Core và chia sẻ lên nuget.org

Cách tạo ra dự án thư viện lớp trong C# Net core, thiết lập một dự án khác tham chiếu đến dự án thư viện, build thư viện và chia sẻ lên nuget.org để có thể tích hợp vào bất kỳ lúc nào

- Dự án thư viện C# NET Core
- Tham chiếu sử dụng dự án thư viện
- Chia sẻ thư viện lên nuget.org

Trong phần này tìm hiểu cách tạo ra thư viên trong C# NET Core, tham chiếu một dự án đến thư viện, chia sẻ thư viện lên **nuget.org**

Tạo dự án thư viện NET CORE 3

Thư viện trong C# NET Core là một loại dự án code, được xây dựng và build ra để sau này có thể tích hợp vào các dự án khác. Trong phần này ta sẽ xây dựng một thư viện, trong thư viện chỉ có một lớp tĩnh cung cấp một vài phương thức để sinh mã HTML, nhằm sinh mã HTML nhanh hơn trong các ví dụ với ASP.NET. Lớp tĩnh này có tên `HtmlHelper`, các bước xây dựng như sau:

Tạo ra một thư mục đặt tên `htmlhelperlib`, trong nó tạo ra thư mục `htmlhelper` chứa code dự án, sau đó vào thư mục này thực hiện lệnh để tạo dự án thuộc loại thư viện NET Core

```
mkdir htmlhelperlib
cd htmlhelperlib
mkdir htmlhelper
```

Lệnh khởi tạo dự án, đang trong thư mục htmlhelper gõ lệnh

```
dotnet new classlib
```

Sau lệnh này dự án được tạo, mở file `.csproj` ra và cập nhật các thông tin tại mục `>PropertyGroup>` thêm vào gồm:


```
<TargetFramework>netcoreapp3.1</TargetFramework>
<PackageId>XTL.HtmlHelper</PackageId>
<Version>1.0.0</Version>
<Authors>xuanthulab</Authors>
<Company>XUANTHULAB.NET</Company>
<GenerateTargetFrameworkAttribute>false</GenerateTargetFrameworkAttribute>
```

Trong phần PackageID nếu có ý định chia sẻ lên **nuget.org** thì phải chọn tên duy nhất, không trung với tên các gói đã có trên nuget.org

Tạo ra file `HtmlHelper.cs` và xây dựng thư viên lớp như sau:

```
using System.Collections.Generic;
using System.Text;
using Microsoft.AspNetCore.Http;

namespace XTLAB.HtmlHelper {

    public static class HtmlHelper {
        /// <summary>
        /// Phát sinh trang HTML
        /// </summary>
        /// <param name="title">Tiêu đề trang</param>
        /// <param name="content">Nội dung trong thẻ body</param>
        /// <returns>Trang HTML</returns>
        public static string HtmlDocument (string title, string content) {
            return $@"
                    <!DOCTYPE html>
                    <html>
                        <head>
                            <meta charset=""UTF-8"">
                            <title>{title}</title>
                            <link rel=""stylesheet"" href=""/css/site.min.css"" />
                            <script src=""/js/jquery.min.js"">
                            </script><script src=""/js/popper.min.js"">
                            </script><script src=""/js/bootstrap.min.js""></script>
                        </head>
                        <body>
                            {content}
                        </body>
                    </html>";
        }

        /// <summary>
        /// Phát sinh HTML thanh menu trên, menu nào  active phụ thuộc vào URL mà request gửi đến
        /// </summary>
        /// <param name="menus">Mảng các menu item, mỗi item có cấu trúc {url, lable}</param>
        /// <param name="request">HttpRequest</param>
        /// <returns></returns>

        public static string MenuTop (object[] menus, HttpRequest request) {

            var menubuilder = new StringBuilder ();
            menubuilder.Append ("<ul class=\"navbar-nav\">");
            foreach (dynamic menu in menus) {
                string _class = "nav-item";
                // Active khi request.PathBase giống url của menu
                if (request.PathBase == menu.url) _class += " active";
                menubuilder.Append ($@"
                                <li class=""{_class}"">
                                    <a class=""nav-link"" href=""{menu.url}"">{menu.label}</a>
                                </li>
                                ");
            }
            menubuilder.Append ("</ul>
");

            string menuhtml = $@"
                    <div class=""container"">
                        <nav class=""navbar navbar-expand-lg navbar-dark mainbackground"">
                            <a class=""navbar-brand"" href=""/"">XTLAB</a>
                            <button class=""navbar-toggler"" type=""button""
                                data-toggle=""collapse"" data-target=""#my-nav-bar""
                                aria-controls=""my-nav-bar"" aria-expanded=""false"" aria-label=""Toggle navigation"">
                                <span class=""navbar-toggler-icon""></span>
                            </button>
                            <div class=""collapse navbar-collapse"" id=""my-nav-bar"">
                                {menubuilder}
                            </div>
                    </nav></div>";

            return menuhtml;
```

```
        }

        /// <summary>
        /// Những menu item mặc định cho trang
        /// </summary>
        /// <returns>Mảng các menuitem</returns>
        public static object[] DefaultMenuTopItems () {
            return new object[] {
                new {
                    url = "/RequestInfo",
                        label = "Request"
                },
                new {
                    url = "/Form",
                        label = "Form"
                },
                new {
                    url = "/Encoding",
                        label = "Encoding"
                },
                new {
                    url = "/Cookies",
                        label = "Cookies"
                },
                new {
                    url = "/Json",
                        label = "JSON"
                }
            };
        }

        public static string HtmlTrangchu () {
            return $@"
          <div class=""container"">
            <div class=""jumbotron"">
                <h1 class=""display-4"">Đây là một trang Web .NET Core</h1>
                <p class=""lead"">Trang Web này xây dựng trên nền tảng  <code>.NET Core</code>,
                chưa sử dụng kỹ thuật MVC - nhằm mục đích học tập.
                Mã nguồn trang này tại <a target=""_blank""
                    href=""https://github.com/xuanthulabnet/learn-cs-netcore/blob/master/ASP_NET_CORE/03.RequestResponse/"">
                    Mã nguồn Ví dụ</a>

                </p>


                <hr class=""my-4"">
                <p><code>.NET Core</code> là một hệ thống chạy đa nền tảng (Windows, Linux, macOS)</p>
                <a class=""btn btn-danger btn-lg"" href=""https://xuanthulab.net/lap-trinh-c-co-ban/"" role=""button"">Xem thêm</a>
            </div>
        </div>
         ";

        }

        // Mở rộng String, phát sinh thẻ HTML với nội dụng là String
        // Ví dụ:
        // "content".HtmlTag() => <p>content</p>
        // "content".HtmlTag("div", "text-danger") => <div class="text-danger">content</div>
        public static string HtmlTag (this string content, string tag = "p", string _class = null) {
            string cls = (_class != null) ? $" class=\"{_class}\"" : null;
            return $"<{tag + cls}>{content}</{tag}>";
        }
        public static string td (this string content, string _class = null) {
            return content.HtmlTag ("td", _class);
        }
        public static string tr (this string content, string _class = null) {
            return content.HtmlTag ("tr", _class);
        }
        public static string table (this string content, string _class = null) {
            return content.HtmlTag ("table", _class);
        }

    }
}
```

Để ý, file code có sử dụng namespace `Microsoft.AspNetCore.Http`, nên cũng cần tích hợp gói này vào dự án bằng lệnh

```
dotnet add package Microsoft.AspNetCore.Http
```

Như vậy bạn đã có một dự án thư viện ở thư mục `htmlhelperlib/htmlhelper`, bạn có thể build thử để xuất file thư viện .dll

Thư viện này lớp `HtmlHelper` ở namespace `XTLAB.HtmlHelper`

```
dotnet build
```

Dự án tham chiếu đến dự án thư viện NET Core

Dự án thư viên trên code đang ở thư mục `htmlhelperlib/htmlhelper`, ta sẽ tạo ra một dự án code khác có tham chiếu sử dụng thư viện trên. Giả sử trong thư mục `htmlhelperlib` tạo ra một thư mục `htmltestcode` để khởi tạo là một dự án console, vào thư mục `htmltestcode` và thự hiện lệnh:

```
dotnet new console
```

Sau đó, đứng ở thư mục `htmlhelperlib` gõ lệnh sau để tham chiếu sử dụng htmlhelper

```
dotnet add htmltestcode/htmltestcode.csproj reference htmlhelper/htmlhelper.csproj
```

Sau lệnh này thấy trong file `htmlhelper.csprgj` có thông tin:

```
<ItemGroup>
    <ProjectReference Include="..\htmlhelper\htmlhelper.csproj" />
</ItemGroup>
```

Sử dụng lớp từ thư viện, bạn mở program.cs ra và code thành

```
using System;
using XTLAB.HtmlHelper;    // Sử dụng được namespace từ thư viện HtmlHelper

namespace htmltestcode
{
    class Program
    {
        static void Main(string[] args)
        {
            // Gọi thử hàm từ HtmlHelper
            String html = "Ví dụ sử dụng HtmlHelper".HtmlTag("div", "text-danger");
            Console.WriteLine(html);
            // Kết quả: <div class="text-danger">Ví dụ sử dụng HtmlHelper</div>

        }
    }
}
```

Như vậy bạn đã tham chiếu đến một dự án thư viện, khi build dự án này thì thư viện cũng build theo nếu mã nguồn thư viện cập nhật và chưa build lại.

Chia sẻ thư viện lên Nuget.org

Bạn cần vào trang `https://www.nuget.org/` đăng ký một tài khoản và đăng nhập vào.

Truy cập địa chỉ `https://www.nuget.org/account/apikeys` để lấy API Key. Bấm vào **Create** điền các thông tin như key name điền tên do bạn đặt, `Glob Pattern` thì điến *, sau đó bấm vào Create.

Khi API Key được tạo, bấm vào copy để lấy, key có dạng như `oy2aebwkfhj3jcg5k7aarfahset74gsejnljx42pwabcde`

Để chi sẻ thư viện, nó cần biên dịch và đóng gói thành file `.nupkg`, tại thư mục dự án bạn gõ lệnh:

```
dotnet pack
```

Nó sinh ra file tại đường dẫn `htmlhelperlib/htmlhelper/bin/Debug/XTL.HtmlHelper.1.0.0.nupkg`

Nếu muốn mỗi lần lệnh dotnet build thì nó đóng gói nupkg luôn thì thêm vào mục `<PropertyGroup>`

```
<GeneratePackageOnBuild>true</GeneratePackageOnBuild>
```

Giờ ta tiến hành chia sẻ lên nuget.org,ở thư mục chứa file nupkg gõ lệnh sau:

```
dotnet nuget push XTL.HtmlHelper.1.0.0.nupkg --api-key qz2jga8pl3dvn2akksyquwcs9ygggg4exypy3bhxy6w6x6 --source https://api.nuget.org/v3/index.json
```

Nhớ thay chính xác API Key của bạn.

Sau lệnh này, quá trình đẩy lên nuget.org diễn ra, sau đó nuget.org sẽ kiểm tra gói của bạn, nếu nó không chứa mã độc thì sẽ được public (có thể mất hàng giờ kiểm tra) và có thể tải về.

Ví dụ này, sau khi đưa lên nuget.org, ai cũng có thể tích hợp vào code của họ bằng lệnh

```
dotnet add package XTL.HtmlHelper --version 1.0.0
```

XTL.HtmlHelper

Tham khảo mã nguồn ASP_NET_CORE/htmlhelperlib hoặc tải về ex054

Mục lục bài viết
Dự án thư viện C# NET CoreTham chiếu sử dụng dự án thư việnChia sẻ thư viện lên nuget.org

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 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 Lập trình multi thread C# sử dụng Parallel chạy song song các tác vụ Dependency injection (DI) trong C# với ServiceCollection
(Multithreading) Parallel (Bài trước)
(Bài tiếp) **(Networking)** Uri, Dns, Ping
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