



LẬP TRÌNH C# 4

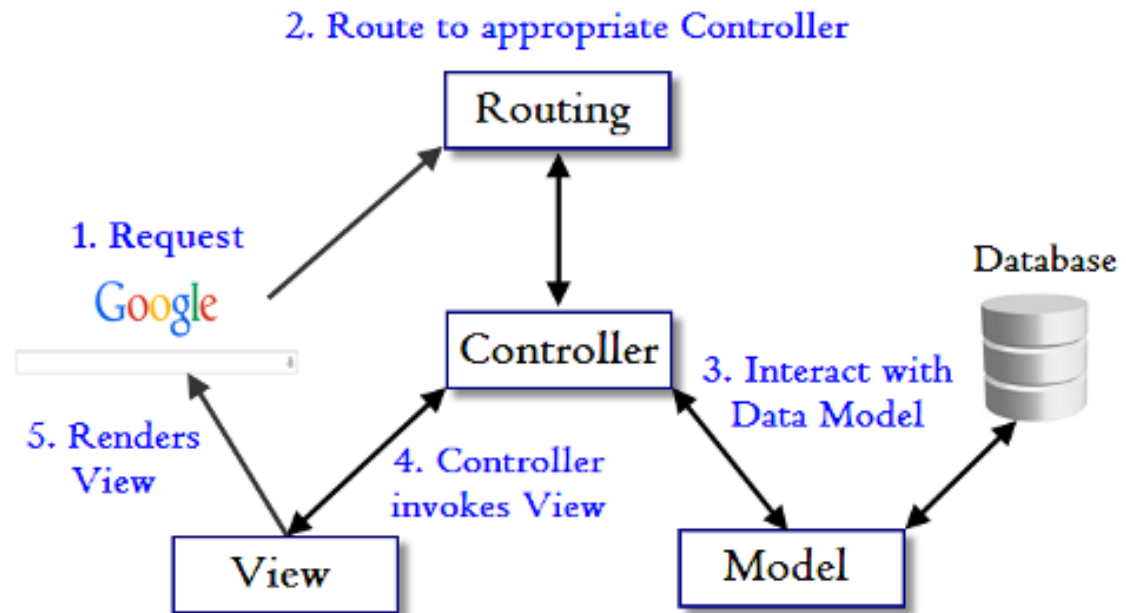
BÀI 4: ROUTING IN ASP.NET CORE MVC

- ① Conventional Routing
- ① Attribute Routing
- ① Route Constraint

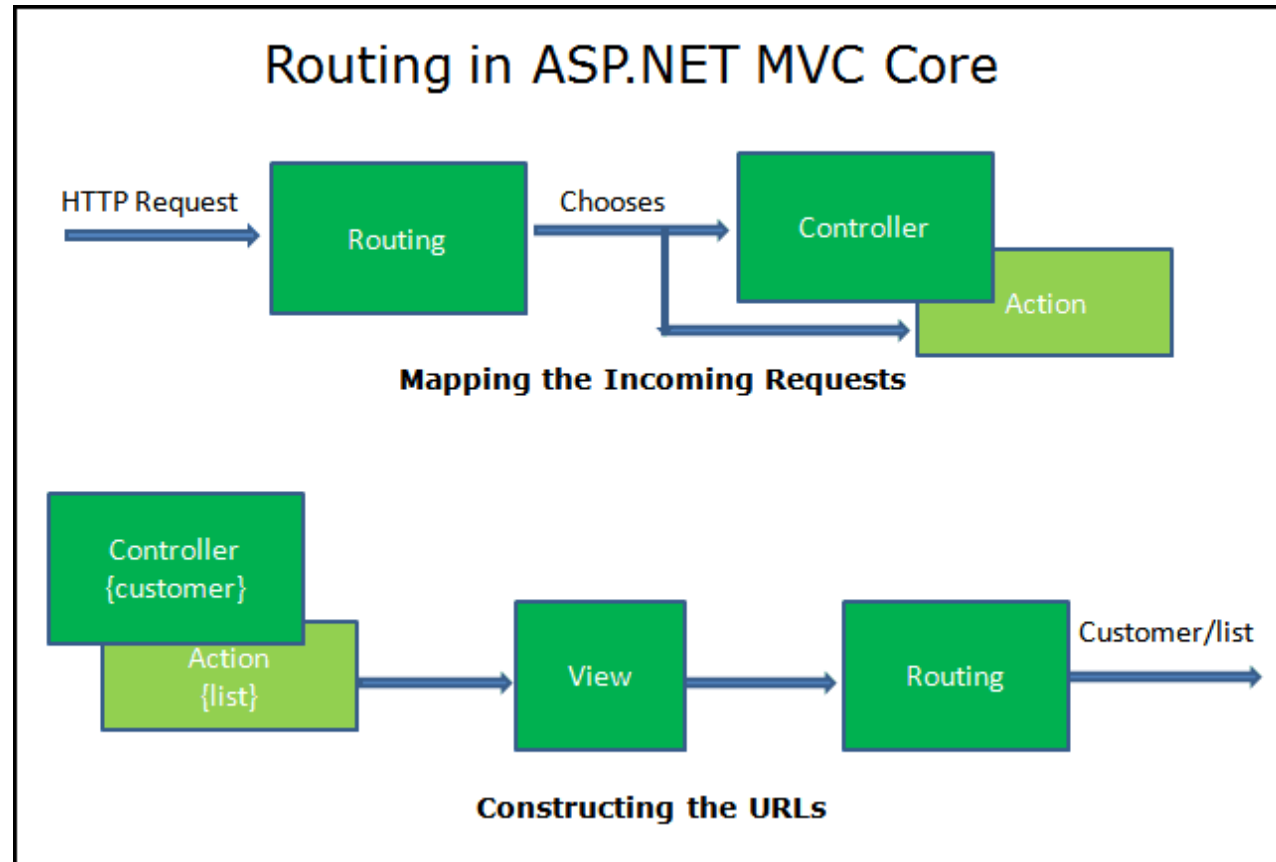


❑ Quá trình tiếp nhận và xử lý cơ bản ứng dụng mvc

❑ Routing là thành phần tiếp nhận và phân phối request đến controller

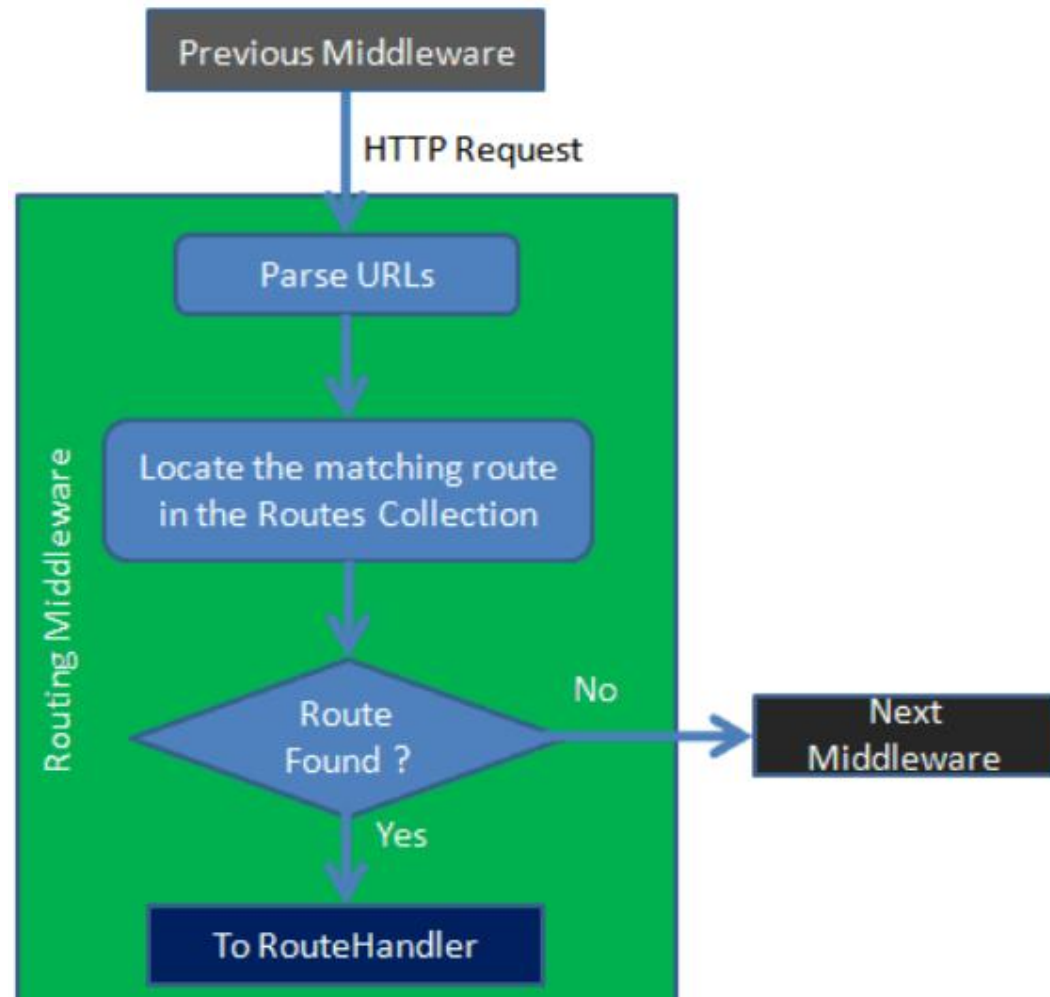


- Routing trong ASP.NET Core là quá trình xem xét các URL request gửi đến và "chỉ đường" cho nó đến Controller Actions, và được sử dụng để tạo ra URL đầu ra



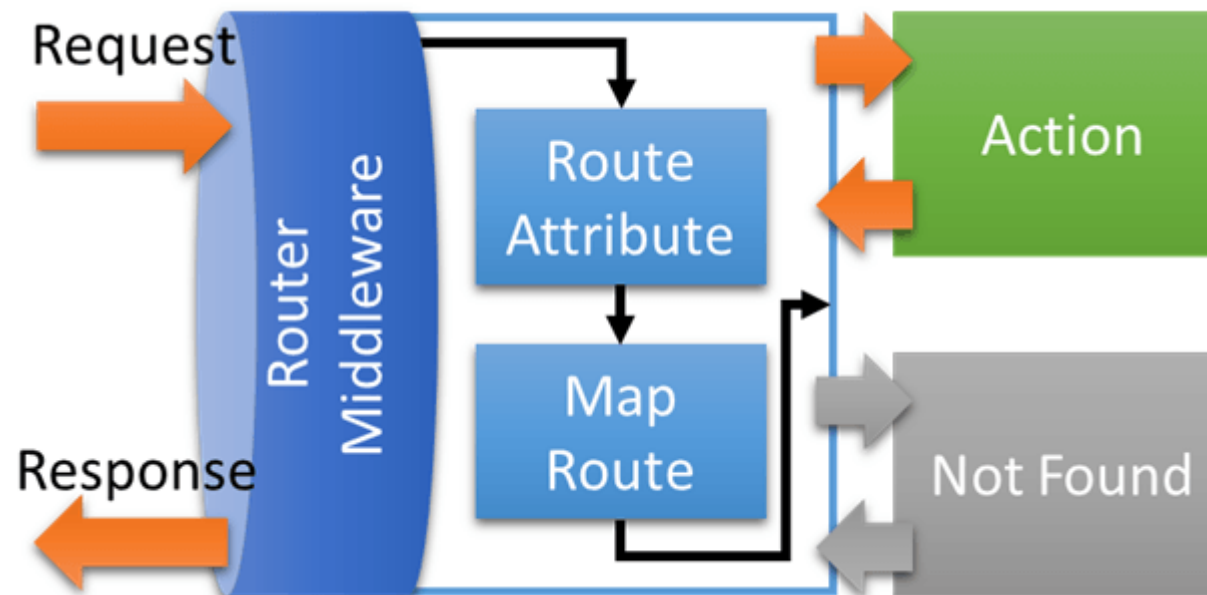
Khi request đến thì Routing Middleware sẽ

- ❑ Phân tích URL
- ❑ Tìm kiếm có Route nào match trong RouteCollection
- ❑ Nếu Route tìm thấy thì đẩy sang RouteHandler
- ❑ Nếu không tìm thấy Route thì bỏ qua, gọi middleware tiếp theo





- ❑ **RouteCollection**: là một tập hợp tất cả các Route trong ứng dụng, các Route sẽ thêm vào collection khi ứng dụng khởi động
- ❑ **RouteHandler** là một thành phần quyết định sẽ làm gì với Route. Khi cơ chế routing tìm được một Route thích hợp cho một request đến, nó sẽ gọi đến RouteHandler và gửi route đó cho RouteHandler xử lý.

- ❑ Trong ASP.NET Core MVC có thể định nghĩa routing theo 2 cách
 - ❖ Conventional Routing: tạo ra Route dựa trên một loạt các quy tắc được định nghĩa trong file Startup.cs
 - ❖ Attribute Routing: Tạo các Route dựa trên các attribute đặt trong Controller action

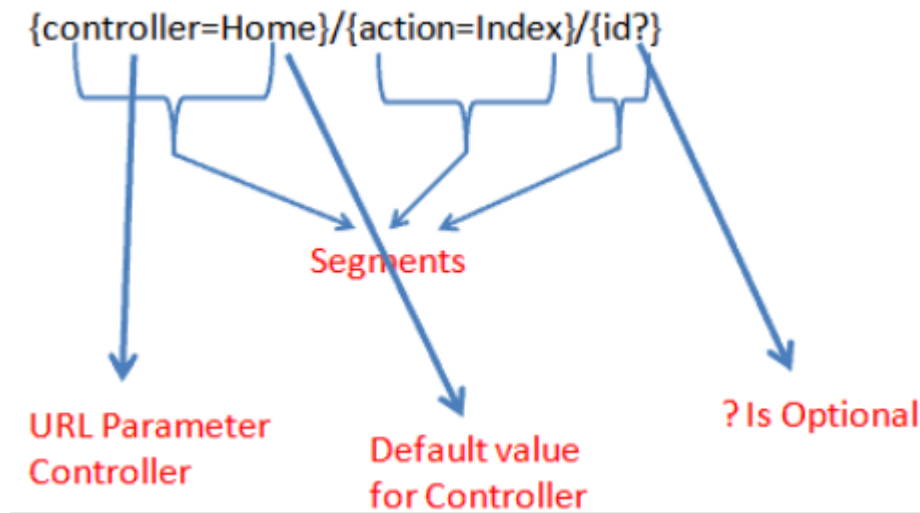


- ❑ Convention based Routes được cấu hình trong phương thức Configure của Startup.cs
- ❑ Thêm Routing Middleware vào request pipeline dùng `app.UseMvc()` hoặc `app.UseMvcWithDefaultRoute()`
- ❑ `app.UseMvc` tạo một thể hiện của class `RouteBuilder`. `RouteBuilder` có extension method là `MapRoute` thêm Route vào Route Collection.
- ❑ Ví dụ `MapRoute` tạo một route đơn lẻ nó có tên là default với URL Pattern của route là `{controller=Home}/{action=Index}/{id?}`

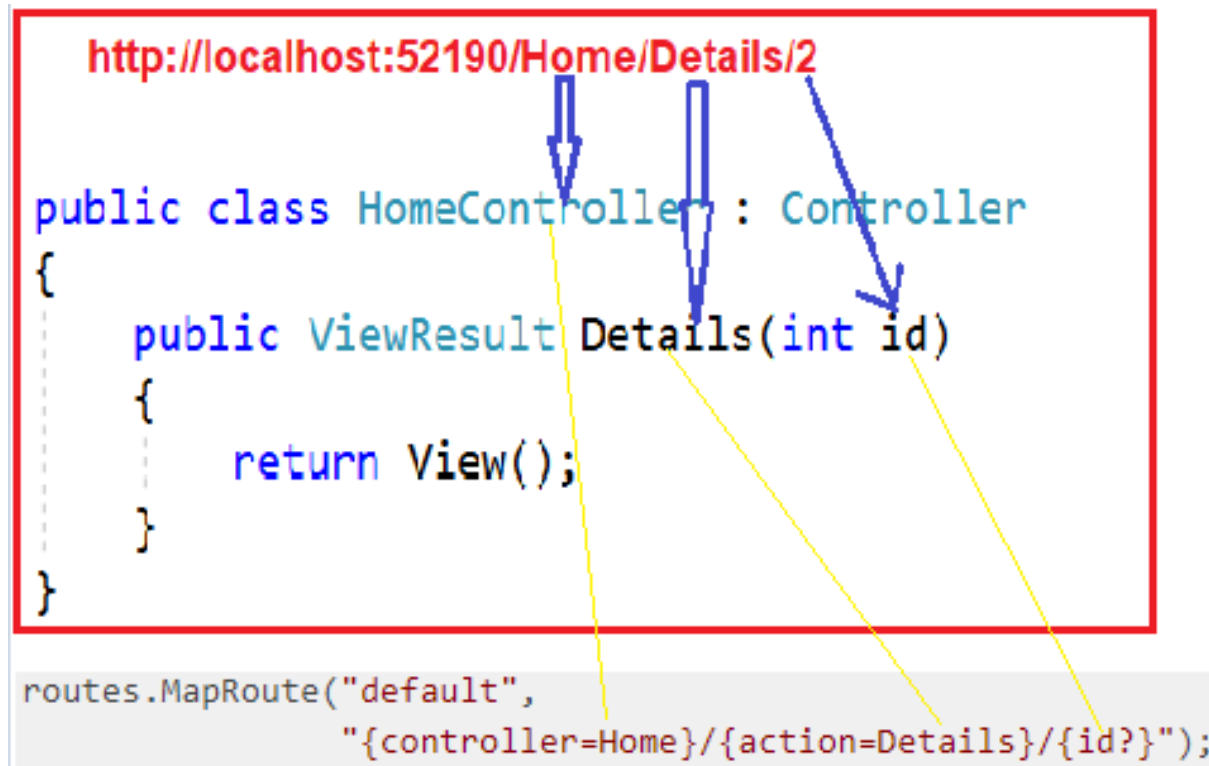
```
app.UseMvc(routes =>
{
    routes.MapRoute(
        "default",  "Name of the Route",
        "{controller=Home}/{action=Index}/{id?}");  "URL Pattern"
    });
```


- ❑ một URL Pattern bao gồm một hoặc nhiều phần chia tách bởi dấu gạch chéo.
- ❑ Route Parameter được bao bọc bởi một cặp dấu ngoặc nhọn ví dụ {controller}, {action}.
- ❑ Route Parameter có thể có giá trị mặc định như {controller=Home}
- ❑ Dấu ? sau tên tham số chỉ ra tham số đó không yêu cầu phải có giá trị.

- URL {controller=Home}/{action=Index}/{id?}:
Đăng ký một route có thành phần đầu tiên trên URL là một controller, phần thứ 2 là Action method trong controller đó. Và phần cuối là dữ liệu thêm vào tên là id



- ❑ URL Matching: Mỗi phần trong URL request đến sẽ match tương ứng với thành phần của URL Pattern



- ❑ Tạo controller Student với 2 action method

```
public string Index()
{
    return "Index() Action Method of StudentController";
}
0 references | 0 requests | 0 exceptions
public string Details(string id)
{
    return "Details() Action Method of StudentController and " + id.ToString();
}
```

- ❑ Khai báo dùng app.UseMvcWithDefaultRoute() trong Startup.cs

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    app.UseMvcWithDefaultRoute();
    app.Run(async (context) =>
    {
        await context.Response.WriteAsync("Hello World!");
    });
}
```

❑ Sinh viên cho biết khi run project trên trình duyệt <http://localhost:port> được kết quả nào? giải thích?

1. ~~Index() Action Method of StudentController~~
2. ~~Details() Action Method of StudentController~~
3. Hello World!

❑ `app.UseMvcWithDefaultRoute()` mặc định dùng controller Home, project chỉ mới tồn tại controller Student → ứng dụng chạy middleware `await context.Response.WriteAsync("Hello World!");`

❑ Sv đề xuất cách run ứng dụng cho kết quả

Index() Action Method of StudentController

- ❖ Chỉ định trực tiếp controller Student trên url <http://localhost:5000/student>
- ❖ Không dùng app.UseMvcWithDefaultRoute(), dùng app.UseMvc() với extension method MapRoute

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    //app.UseMvcWithDefaultRoute();
    app.UseMvc(routes =>
    {
        routes.MapRoute(
            name: "default",
            template: "{controller=Student}/{action=Index}/{id?}");
    });
}
```

❑ Sv đề xuất cách run ứng dụng cho kết quả

`Details()` Action Method of `StudentController`

❖ dùng `app.UseMvc()` với extension method `MapRoute`

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    //app.UseMvcWithDefaultRoute();
    app.UseMvc(routes =>
    {
        routes.MapRoute(
            name: "default",
            template: "{controller=Student}/{action=Details}/{id?}");
    });
}
```

❑ Sv đề xuất cách run ứng dụng cho kết quả

Details() Action Method of StudentController and id_string

localhost:5000/student/Details/id_string

Hoặc

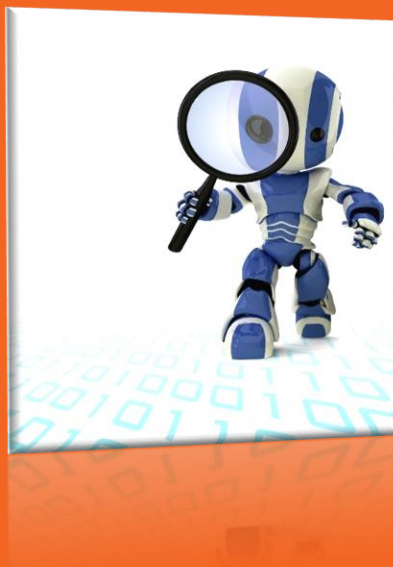
```
app.UseMvc(routes =>
{
    routes.MapRoute(
        name: "default",
        template: "{controller=Student}/{action=Details}/{id='id_string'}");
});
app.Run(async (context) =>
{
    await context.Response.WriteAsync("Hello World!");
});
```




DEMO

- Hiện thực các ví dụ





LẬP TRÌNH C# 4

BÀI 4: ROUTING IN ASP.NET CORE MVC (P2)

- ❑ Làm sao chạy ứng dụng với controller và không cấu hình routing trong Startup.cs

```
public class Startup
{
    public void ConfigureServices(IServiceCollection services)
    {
        services.AddMvc();
    }

    public void Configure(IApplicationBuilder app, IHostingEnvironment env)
    {
        app.UseMvc();
    }
}
```

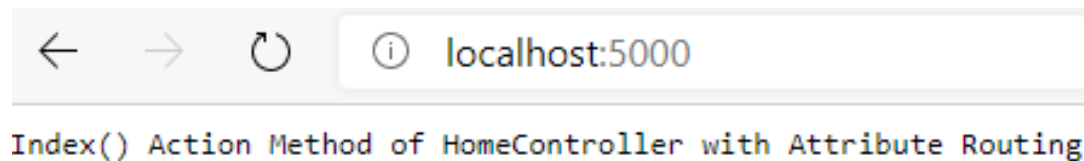
```
public class HomeController : Controller
{
    public string Index()
    {
        return "Index() Action Method of HomeController";
    }
}
```

- ❑ ➔ Attribute Routing

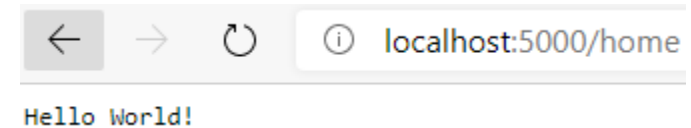
- ❑ Attribute routing sử dụng các attribute để định nghĩa trực tiếp các route trong controller action

```
public class HomeController : Controller
{
    [Route("")]
    0 references | 0 requests | 0 exceptions
    public string Index()
    {
        return "Index() Action Method of HomeController with Attribute Routing";
    }
}
```

- ❑ Run project:



- ❑ Nếu run project với localhost:5000/home hoặc localhost:5000/home/index:



- ❑ Để chạy được với các dạng Url pattern khác nhau cần thêm đầy đủ các pattern routing

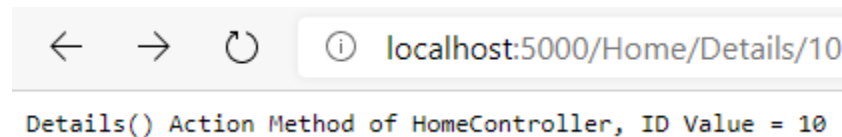
```
public class HomeController : Controller
{
    [Route("")]
    [Route("Home")]
    [Route("Home/Index")]
    public string Index()
    {
        return "Index() Action Method of HomeController with Attribute Routing";
    }
}
```

Chạy kiểm tra:

1. <http://localhost:52190>
2. <http://localhost:52190/home>
3. <http://localhost:52190/home/index>

- ❑ Có thể định nghĩa các URL Parameter tức là các tham số trên URL trong cặp ngoặc nhọn

```
public class HomeController : Controller
{
    [Route("Home/Details/{id}")]
    0 references | 0 requests | 0 exceptions
    public string Details(int id)
    {
        return "Details() Action Method of HomeController, ID Value = " + id;
    }
}
```



- ❑ Run project

- ❑ Tạo parameter là optional bằng cách thêm dấu: "?"

```
[Route("Home/Details/{id?}")] //id là optional
0 references | 0 requests | 0 exceptions
public string Details(int id)
{
    return "Details() Action Method of HomeController, ID Value = " + id;
}
```

- ❑ Có thể kết hợp attribute routing thay đổi tên Controller và tên Action method khi run project trên trình duyệt tạo Url thân thiện
- ❑ Vd đổi Home-MyHome và StartPage-> friendlyName

```
public class HomeController : Controller
{
    [Route("")]
    [Route("MyHome")]
    [Route("MyHome/friendlyName")]
    0 references | 0 requests | 0 exceptions
    public string StartPage()
    {
        return "StartPage() Action Method of HomeController with friendlyName";
    }
}
```

- ❑ Có thể run project với các url: localhost:5000 hoặc localhost:5000/MyHome hoặc localhost:5000/MyHome/friendlyName

- ❑ Thiết lập attribute routing cho controller thay vì cho action giúp ràng buộc controller tốt hơn, hạn chế sự bất động bộ khi code ứng dụng
- ❑ Giảm thiểu sự trùng lặp code

```
[Route("Home")]  
public class HomeController : Controller  
{  
    [Route("")]  
    [Route("Index")]  
    public string Index()  
    {  
        return "Index() Action Method of HomeController";  
    }  
  
    [Route("Details/{id?}")]  
    public string Details(int id)  
    {  
        return "Details() Action Method of HomeController, ID Value = " + id;  
    }  
}
```

/Home
/Home/Index
/Home/Details
/Home/Details/2

- ❑ Route Constraint được sử dụng trong 2 nhiệm vụ chính:
 - ❖ giúp lọc và giới hạn các giá trị không mong muốn truyền vào controller action
 - ❖ sử dụng để hỗ trợ Route Engine phân biệt giữa 2 route tương tự nhau về url pattern
- ❑ Các ràng buộc áp dụng cho giá trị truyền vào URL Parameter, có 2 cách để thêm Constrain vào URL Parameter:
 - ❖ Thêm trực tiếp vào URL Parameter (Inline)
 - ❖ Sử dụng tham số Constrain trong phương thức MapRoute

- ❑ Inline Constraint được thêm vào sau URL Parameter và được tách với dấu hai chấm
- ❑ Vd cần truyền Id kiểu int trong action Index và ràng buộc trong routing:

```
public class HomeController : Controller
{
    public string Index(int id)
    {
        return "I got " + id.ToString();
    }
}
```

```
routes.MapRoute("default", "{controller=Home}/{action=Index}/{id:int?}");
```

Kiểm tra với URL là `"/Home/Index/10"` và `/Home/Index/Test`

- ❑ Các constraints sử dụng các tham số trong phương thức MapRoute.

```
using Microsoft.AspNetCore.Routing.Constraints;

app.UseMvc(routes =>
{
    routes.MapRoute("default",
        "{controller}/{action}/{id}",
        new { controller = "Home", action = "Index" },
        new { id = new IntRouteConstraint() });
});
```

- ❑ Danh sách của Route Constraint

CONSTRAINT	CÚ PHÁP	CLASS	GHI CHÚ
int	{id:int}	IntRouteConstraint	Kiểm tra tham số chỉ được là số nguyên 32 bit.
alpha	{id:alpha}	AlphaRouteConstraint	Kiểm tra tham số chỉ được là ký tự thuộc bảng chữ cái tiếng anh A-Z
bool	{id:bool}	BoolRouteConstraint	Kiểm tra tham số chỉ được là giá trị logic true hoặc false.
datetime	{id:datetime}	DateTimeRouteConstraint	Kiểm tra tham số chỉ được là giá trị DateTime
decimal	{id:decimal}	DecimalRouteConstraint	Kiểm tra giá trị tham số chỉ được là kiểu thập phân
double	{id:double}	DoubleRouteConstraint	Cho phép giá trị tham số chỉ được là kiểu số thực 64 bit
float	{id:float}	FloatRouteConstraint	Cho phép tham số chỉ là kiểu số thực dấu chấm động
guid	{id:guid}	GuidRouteConstraint	Chỉ cho phép kiểu Guid

- ❑ Có thể áp dụng ràng buộc routing tại action method

```
[Route("Home/Index/{id:int}")]  
public string Index(int id)  
{  
    return "I got " + id.ToString();  
}
```

❑ Kết hợp các constraints:

- ❖ Nhiều constraints có thể được kết hợp bởi dấu hai chấm chia tách

```
"/{id:alpha:minlength(6)?}"
```

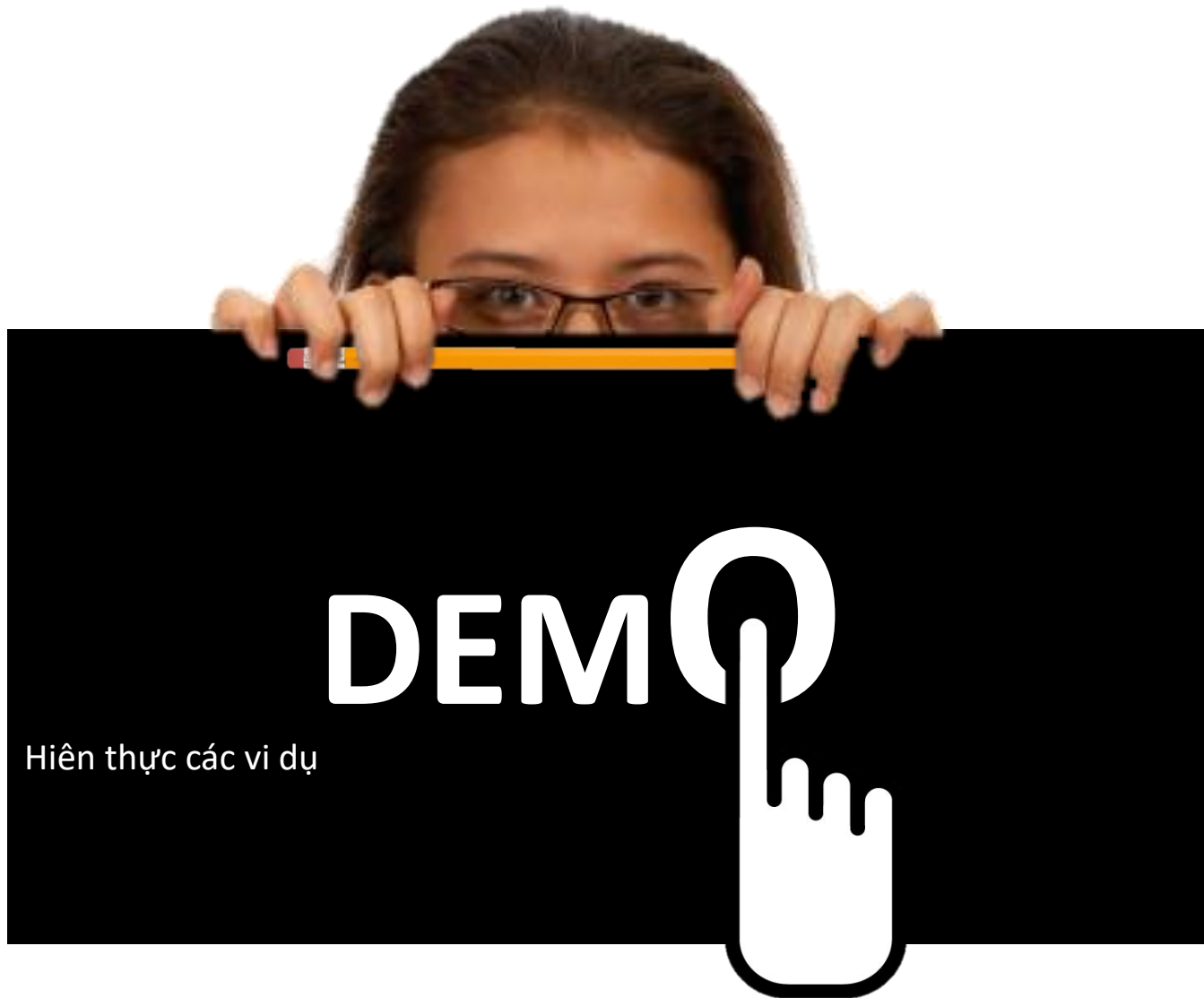
- ❖ sử dụng phương thức Constraints trong MapRoute

```
Using Microsoft.AspNetCore.Routing.CompositeRouteConstraint;  
  
constraints: new {  
    id = new CompositeRouteConstraint(  
        new IRouteConstraint[] {  
            new AlphaRouteConstraint(),  
            new MinLengthRouteConstraint(6)  
        })  
}
```

- ❑ Route Constraint được sử dụng để hỗ trợ Route Engine phân biệt giữa 2 route tương tự pattern.
- ❑ vd có 2 route đều là "post/{id:int}" và "post/{id:alpha}". Route Constraint có thể chỉ cho Routing Engine chọn PostByID Action method nếu giá trị là số và PostByPostName nếu giá trị id là kiểu chữ cái.

```
app.UseMvc(routes =>
{
    routes.MapRoute("default",
        "post/{id:int}",
        new { controller = "Post", action = "PostsByID" });

    routes.MapRoute("anotherRoute",
        "post/{id:alpha}",
        new { controller = "Post", action = "PostsByPostName" });
});
```



Tổng kết bài học

- ◎Conventional Routing
- ◎Attribute Routing
- ◎Route Constraint





KẾT THÚC