
📘 Routing in ASP.NET Core MVC

---

1. What is Routing?

Routing is the mechanism that maps an incoming **URL** to a **controller action**.

It answers the question:

“Which controller and action should handle this URL?”

Routing is the **first decision point** after a request enters the MVC pipeline.

---

2. Why Routing is Fundamental in MVC

Routing:

- Connects browser URLs to controller actions
- Determines how parameters reach actions
- Enables clean, readable URLs
- Works together with model binding

Without routing:

- Controllers cannot be reached
- URLs feel “hardcoded”
- MVC request flow is unclear

---

3. Where Routing is Configured

Routing is configured in **Program.cs**.

```
app.UseRouting();


app.UseAuthorization();


app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");
```

This is called **Conventional Routing**.

---

4. Conventional Routing (Default MVC Routing)

Default Route Pattern

{controller=Home}/{action=Index}/{id?}

Meaning:

- controller → Controller name
- action → Action method
- id? → Optional parameter

Example 1: Default URL

Browser:

https://localhost:5001/

Resolved as:

- Controller → HomeController
- Action → Index()

Example 2: Explicit Controller & Action

Browser:

https://localhost:5001/Home/About

Resolved as:

- Controller → HomeController
- Action → About()

Example 3: With Route Parameter

Browser:

https://localhost:5001/Students/Details/5

Resolved as:

- Controller → StudentsController
- Action → Details(int id)
- id = 5

---

5. Optional Route Parameters

The ? makes a parameter optional.

{id?}

Action:

public IActionResult Details(int? id)

Browser:

/Students/Details

/Students/Details/5

Both URLs are valid.

---

6. Routing vs Model Binding (Clear Separation)

- **Routing** decides **which action is executed**
- **Model Binding** decides **how data is passed to parameters**

Example:

/Students/Details/5?show=true

Routing:

- Chooses StudentsController.Details()

Model Binding:

- id = 5
- show = true

---

7. Attribute Routing (Route on Controller / Action)

Attribute routing defines routes **directly on controllers and actions**.

Controller-Level Route

[Route("students")]

public class StudentsController : Controller

{

}

Browser:

/students

Action-Level Route

```
[Route("students/details")]
public IActionResult Details()
```

Browser:

/students/details

Combined Controller + Action Routes

```
[Route("students")]
public class StudentsController : Controller
{
    [Route("list")]
    public IActionResult List()
    {
        return View();
    }
}
```

Browser:

/students/list

---

8. Route Parameters in Attribute Routing

```
[Route("students/details/{id}")]
public IActionResult Details(int id)
{
    return Content($"id={id}");
}
```

Browser:

/students/details/10

Result:

- id = 10

---

9. HTTP Verb-Based Routing

```
[HttpGet("students/create")]
public IActionResult Create()
{
    return View();
}
[HttpPost("students/create")]
public IActionResult Create(Student s)
{
    return RedirectToAction("Index");
}
```

Same URL, different HTTP methods.

---

10. Route Constraints

Route constraints restrict parameter values.

Integer Constraint

```
[Route("students/details/{id:int}")]
public IActionResult Details(int id)
```

Browser:

/students/details/5 ✓

/students/details/abc ❌

GUID Constraint

```
[Route("orders/{id:guid}")]
```

Browser:

/orders/3fa85f64-5717-4562-b3fc-2c963f66afa6

---

11. Route Name Usage

```
app.MapControllerRoute(
```

```
    name: "studentRoute",
    pattern: "students/{action}/{id?}");
```

Used for URL generation (advanced scenarios).

---

12. Generating URLs Using Tag Helpers

Razor View

```
<a asp-controller="Students" asp-action="Details" asp-route-id="5">
    View Student
</a>
```

Generated URL:

/Students/Details/5

No hardcoded URLs → route-safe.

---

13. asp-route-* for Query Strings

```
<a asp-controller="Students"
   asp-action="Search"
   asp-route-name="Ravi"
   asp-route-age="20">
   Search
</a>
```

Generated URL:

/Students/Search?name=Ravi&age=20

---

14. Routing Precedence Rules

Order of execution:

1. Attribute routes
2. Conventional routes
3. Default route

If attribute routing is used, it **overrides conventional routing** for that action.

---

15. Common Routing Mistakes

- Forgetting MapControllerRoute
- Wrong controller/action names
- Route parameter name mismatch
- Missing optional ?
- Mixing attribute and conventional routes incorrectly

---

16. Best Practices

- Teach conventional routing first
- Use attribute routing for APIs or special cases
- Keep URLs readable
- Avoid hardcoded links
- Use tag helpers for navigation

---

17. Summary

- Routing maps URLs to controller actions
- Configured in Program.cs
- Two types: Conventional & Attribute
- Supports parameters and constraints
- Works before model binding
- Essential for MVC request flow

Exercises for Students

1. Map /Students/List using conventional routing.
2. Create an attribute route /students/profile/{id}.
3. Add an integer constraint to a route.
4. Generate a URL using asp-controller and asp-action.
5. Test route precedence by mixing conventional and attribute routes.

.