Chapter 16:
Navigation and Routing

Gill Cleeren

@gillcleeren

Agenda

- Routing 101
- Implementing navigation with tag helpers

Routing 101

URLs today

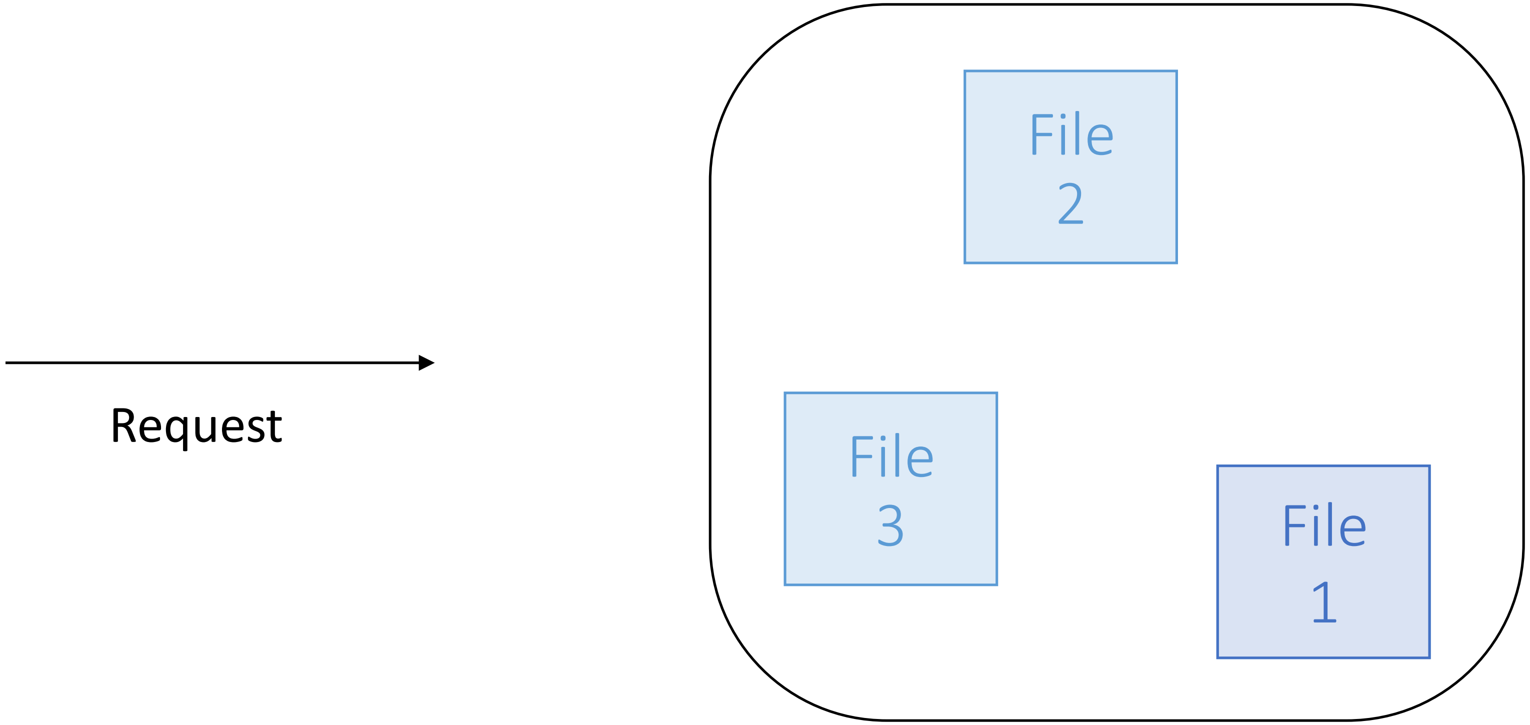
- Long
- Ugly
- Difficult for SEO
- Not easy to see what’s there

→ Routing engine in ASP.NET MVC is a big improvement!

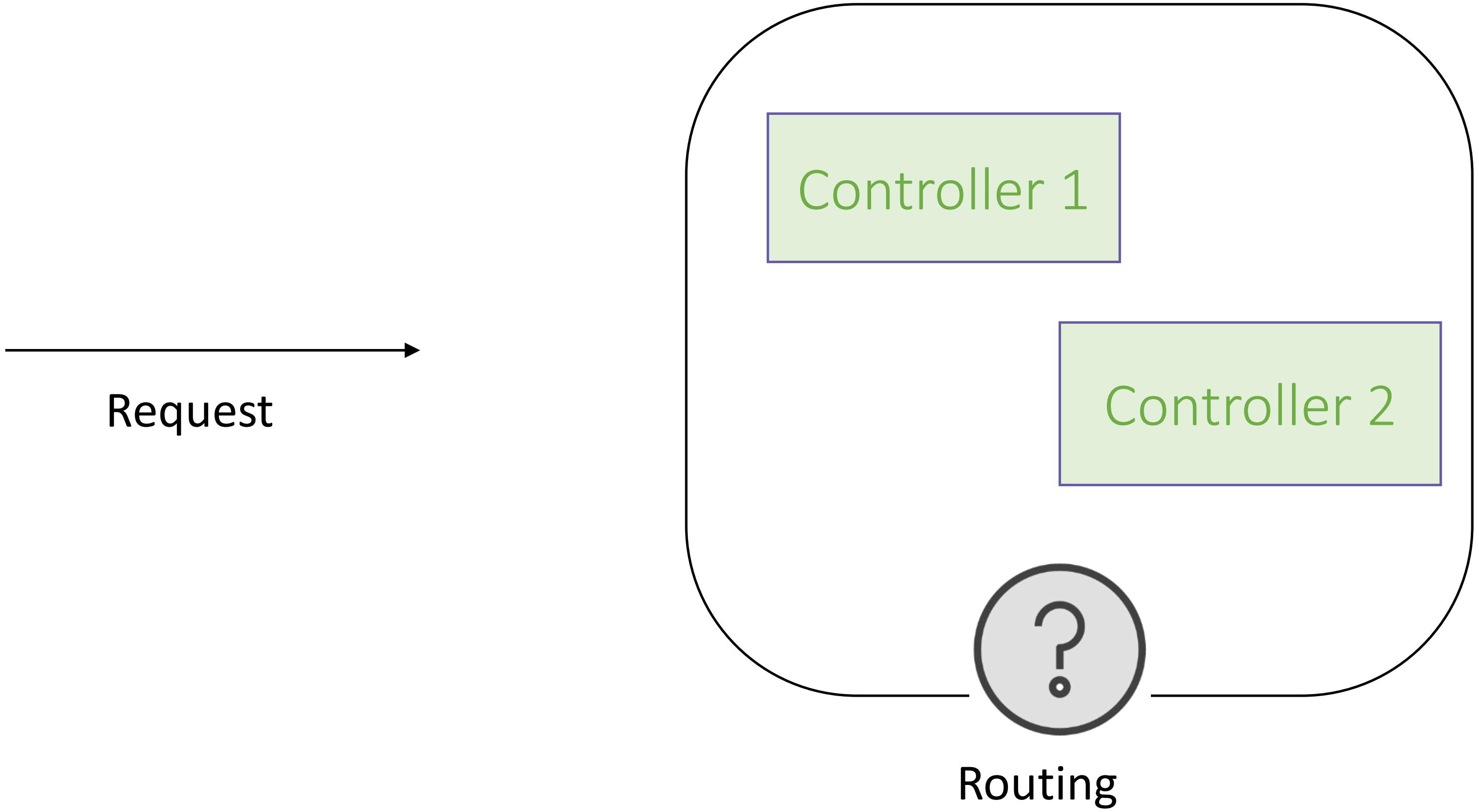
URLs today

- What a great URL should look like
  - A domain name that is easy to remember and easy to spell
  - Short URLs
  - Easy-to-type URLs
  - URLs that reflect the site structure
  - URLs that are *hackable* to allow users to move to higher levels of the information architecture by hacking off the end of the URL
  - Persistent URLs, which don’t change

Routing in ASP.NET WebForms



Routing in MVC applications



Routing

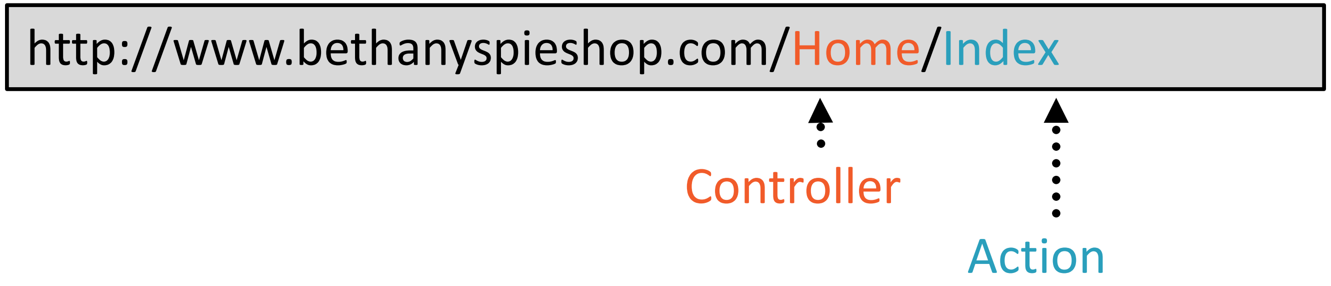
- Used for incoming and outgoing links
- Based on middleware that needs to be included in our Startup class

```
app.UseMvcWithDefaultRoute();
```

- MapRoute allows us to create routes
  - Default route is being created here
  - Uses patterns

```
app.UseRouting();
app.UseEndpoints(endpoints =>
{
        endpoints.MapControllerRoute(
                name: "default",
                pattern: "{controller}/{action}");
});
```

Patterns for routes



http://www.bethanyspieshop.com/Home/Index

Controller

Action

{Controller}/{Action}

```
http://www.bethanyspieshop.com/Home : no match

http://www.bethanyspieshop.com/Home/index/pies : no match
```

Working with defaults for the routes

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

Passing variable values

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

```
public ViewResult Index(string id)
{
}
```

Using optional segments

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

Using constraints on the route

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

Using static segments

```
routes.MapRoute(
  name: "categoryfilter",
  template: "Shop/{action}/{category?}",
  defaults: new { Controller = "Pie", action = "List" });
```

Attribute routing

```
[Route("[controller]/Details/{id}")]
public IActionResult Details(int id)
{
    var pie = _pieRepository.GetPieById(id);
    if (pie == null)
      return NotFound();
        return View(pie);
}
```

Adding navigation to the site

Tag helpers

- Most navigation happens because of clicking on a link or submitting a form
  - Used to be done using an HTML Helper

```
@Html.ActionLink ("View Pie List","List",  "Pie")
```

- Now possible using a Tag Helper

```
<a asp-controller="Pie" asp-action="List">
  View Pie List
</a>
```

DEMO

Adding navigation to the site

LAB

Exercise 11. Adding the detail page and navigation