Razor, Routing and Helpers

Razor

- A markup syntax to create dynamic web pages.
- Introduced with ASP.NET MVC 3.
- Embed with .NET languages C# and VB.

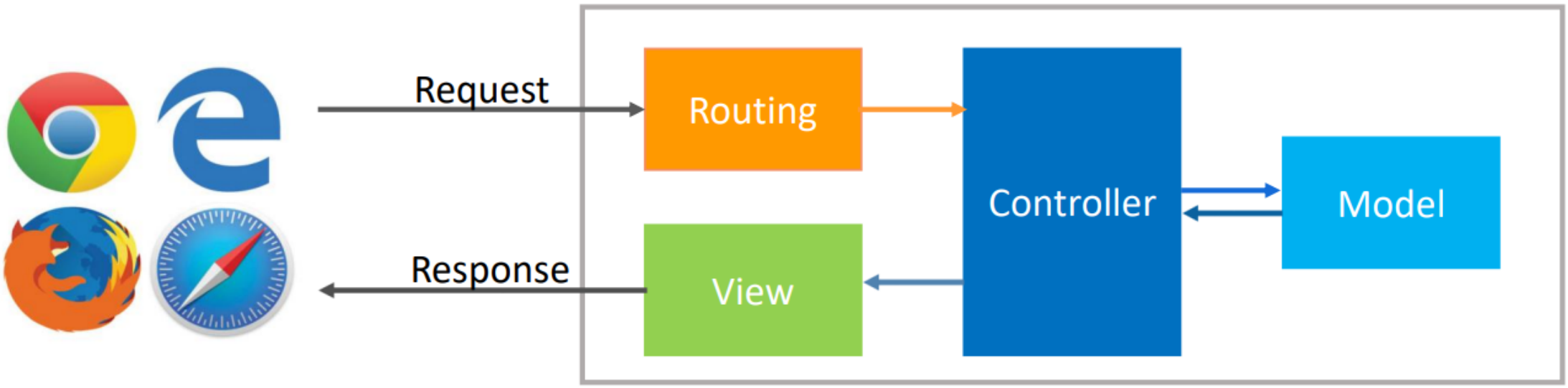
```
<!-- Single statement block -->
@{ var msg = "Hello World"; }
<!-- Inline expression or variable -->
<p>My message is: @msg</p>

<!– if Statement -->
@if(flag==true) {
      <p>Hi</p>
   }
```

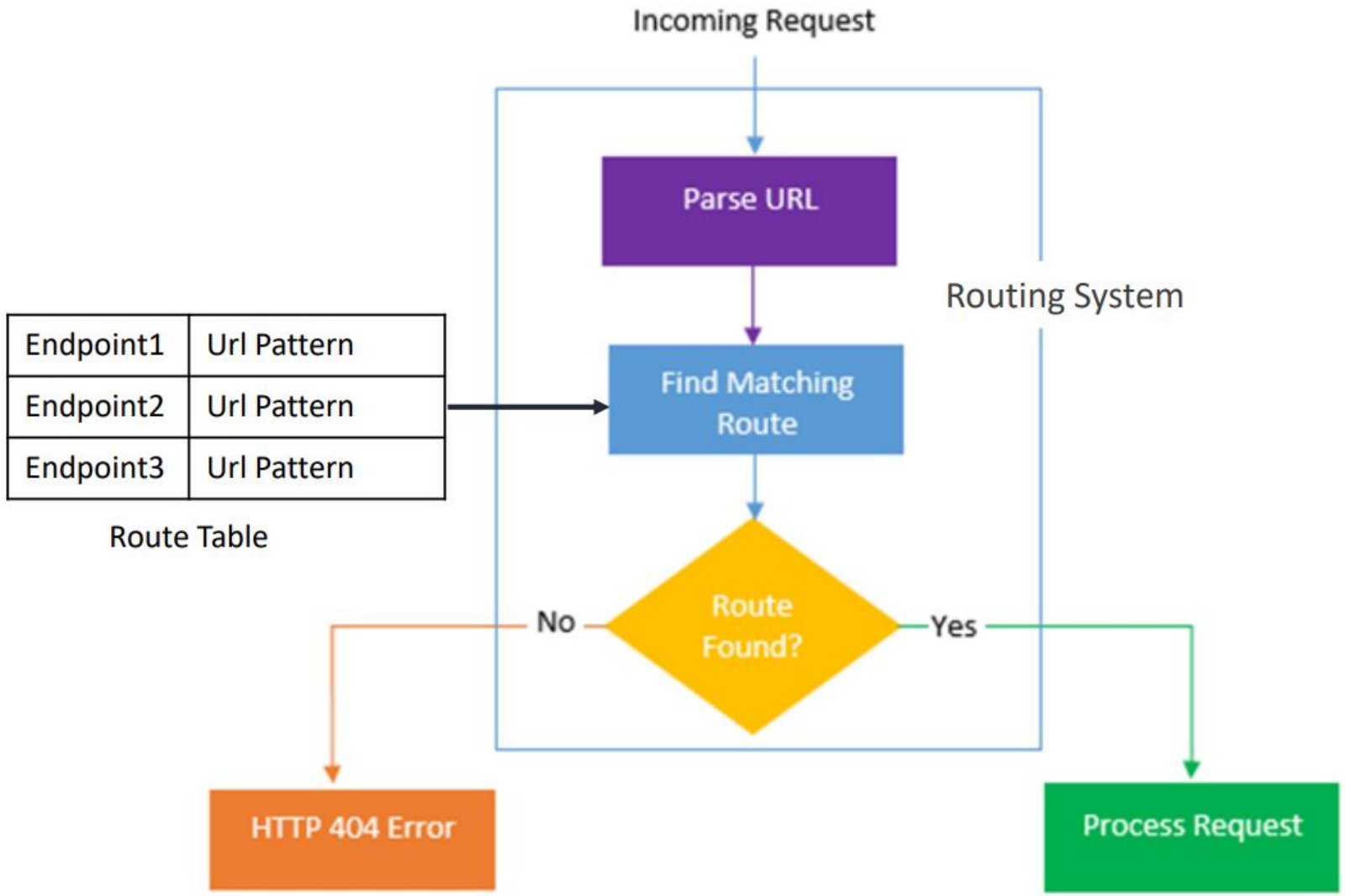
```
<!-- Loop -->
@for(int i = 1; i <= 20; i++)
   {
      <p>Line @i</p>
   }
```

Routing

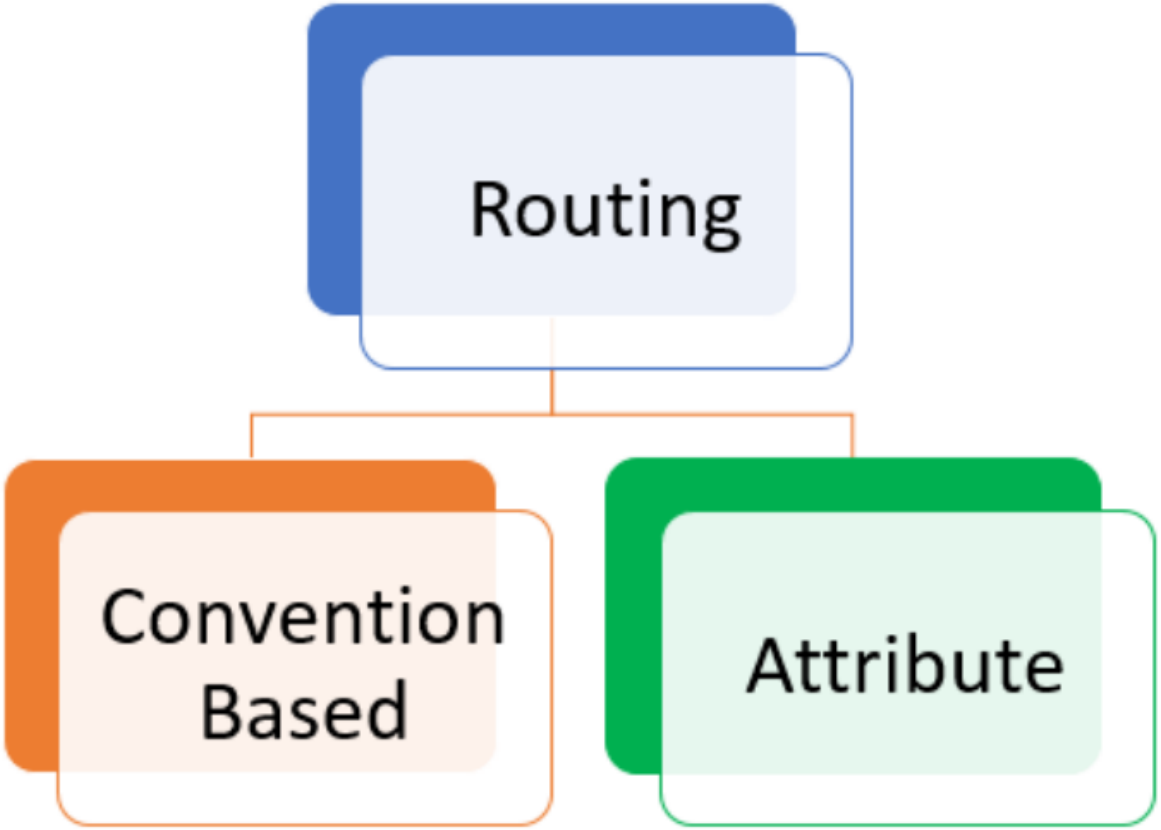
- A pattern matching system that monitor the incoming request and figure out what to do with that request.
- Matches the incoming request against a set of endpoints and their criteria. Typically, a way to serve user request.
- Supports Link/URL generation for registered routes.



Request Handling



Types of Routing



Convention Routing Configuration & Mapping

```
public class ProductController : Controller
{
    //Product
    0 references | 0 requests | 0 exceptions
    public IActionResult Index()
    {
        return View();
    }
    //Product/Create
    0 references | 0 requests | 0 exceptions
    public IActionResult Create()
    {
        return View();
    }
    //Product/Edit/1
    0 references | 0 requests | 0 exceptions
    public IActionResult Edit(int id)
    {
        return View();
    }
}
```

ASP.NET Core 1.x to 2.x

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

ASP.NET Core 3.x and above

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

Attribute Routing Configuration and Mapping

```
[Route("Home")]
0 references
public class HomeController : Controller
{
    [Route("")]      // "Home"
    [Route("Index")] // "Home/Index"
    [Route("/")]     //  ""
    0 references | 0 requests | 0 exceptions
    public IActionResult Index()
    {
        return View();
    }
    [Route("About")] // "Home/About"
    0 references | 0 requests | 0 exceptions
    public IActionResult About()
    {
        return View();
    }
}
```

```
[Route("[controller]/[action]")]
0 references
public class ProductsController : Controller
{
    [HttpGet] // Matches '/Products/List'
    0 references | 0 requests | 0 exceptions
    public IActionResult List()
    {
        // ...
        return View();
    }
    [HttpGet("{id}")] // Matches '/Products/Edit/{id}
    0 references | 0 requests | 0 exceptions
    public IActionResult Edit(int id)
    {
        // ...
        return View();
    }
}
```

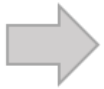
Html Helpers

- These are just like ASP.NET Server controls.
- Extension methods of HtmlHelper class, IHtmlHelper Interface. Return Html markup to create form controls.
- You can create your own helpers, or use the built in HTML helpers.

- Built-In Html Helper
  - Standard Html Helpers
  - Strongly Typed Html Helpers
  - Templated Html Helpers
- Custom Html Helpers

```
@Html.TextBox("Name")
```

```
@Html.TextBoxFor(m => m.Name)
```



```
@Html.EditorFor(m  => m.Name)
```

Url Helper

- Renders HTML links and raw URLs
- Generates links based on routing configuration
- Helps to avoid hard coded Urls

Tag Helpers

- Allows server-side code to embed within HTML elements in Razor files
- Written in C# and target HTML elements based on element name, attribute name, or parent tag
- Supports an HTML-friendly development experience

Tag Helpers

- Form Tag Helper
- Form Action Tag Helper
- Input Tag Helper
- Label Tag Helper
- Anchor Tag Helper
- Image Tag Helper
- Select Tag Helper
- Textarea Tag Helper
- Validation Message Tag Helper
- Validation Summary Tag Helper
- Partial Tag Helper
- Script Tag Helper
- Link Tag Helper
- Cache Tag Helper
- Component Tag Helper
- Distributed Cache Tag Helper
- Environment Tag Helper