Working with Routing
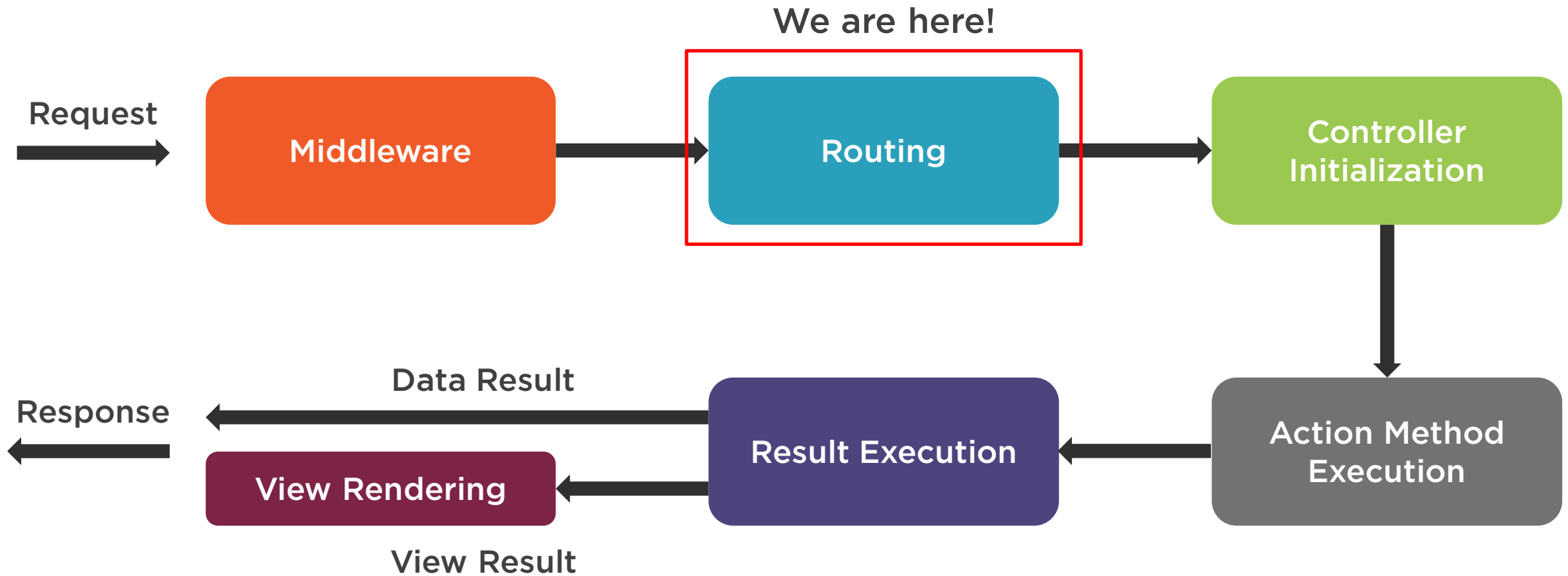
Alex Wolf

www.alexwolfthoughts.com

Sidebar!

This module assumes some familiarity with routing

We will review the basics to ensure common ground

The MVC Request Life Cycle

We are here!

Request → **Middleware** → **Routing** → **Controller Initialization**

Controller Initialization → **Action Method Execution**

Data Result

Response ← **Result Execution** ← **Action Method Execution**

Result Execution → **View Rendering**

View Result

To-Do List

What is Routing?

Demo- Routing Basics

Introducing Endpoint Routing

Endpoint Routing Internals

Demo - Touring the Routing Pipeline

Leveraging the Routing System

Demo - Working with Endpoint Routing

Demo - Routing with Razor Pages

What is Routing?

Routing

The process of mapping an incoming request to an endpoint.

Types of Routing

Conventional Routing

Uses application wide patterns to match a URL to a Controller Action Method

Attribute Routing

Implemented through Attributes applied directly to a Controller or Action Method

```
app.UseEndpoints(endpoints => {

    endpoints.MapControllerRoute(

        name: "default",

        pattern: "{controller=Home}/{action=Index}/{id?}"

    );

})
```

Working with Conventional Routes

Registered during startup and consumed by Routing Middleware

Routes use patterns that map to URL segments and values

Understanding Route Pattern Matching

Incoming URL: localhost/Task/Edit/23

Route Pattern: {controller}/{action}/{id}

```csharp
public class TaskController
{
        public ActionResult Edit(int id)
        {
                return View();
        }
}
```

```csharp
[Route("product")]
public class ProductController : Controller
{

    [Route("edit/{id}")]
    public IActionResult Edit(int id)
    {
        return View();
    }


    [Route("landing")]
    public IActionResult Index()
    {
        return View();
    }
}
```

◄ **Route Attribute to provide a base URL for the Controller**

◄ **Route Attribute with a dynamic template syntax for the product id**

◄ **Route Attribute with a static path**

Routing Configuration Options

Customizable segment patterns

Default values

Data type enforcement

Custom constraints

Understanding Route Evaluation Order

If the Endpoint has Route Attributes, evaluate them in parallel

Attribute Route #1	Attribute Route #2	Attribute Route #3

Otherwise evaluate any Conventional Routes in order

Conventional Route #1

Conventional Route #2
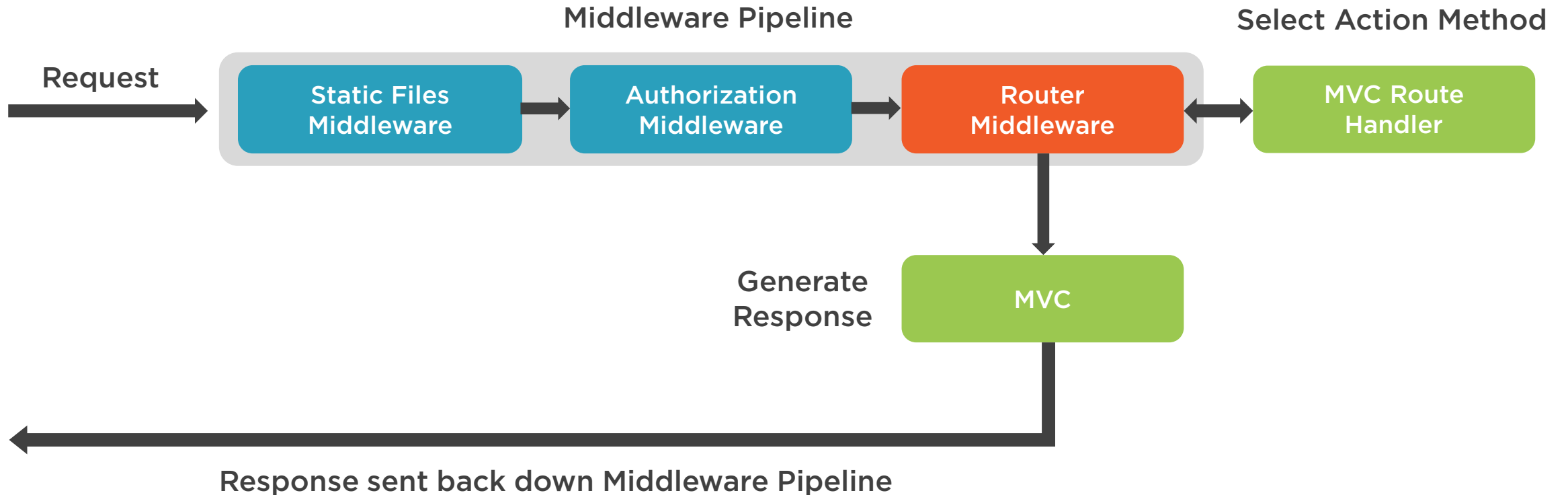
Conventional Route #3

Demo

Working with Basic Routing Concepts
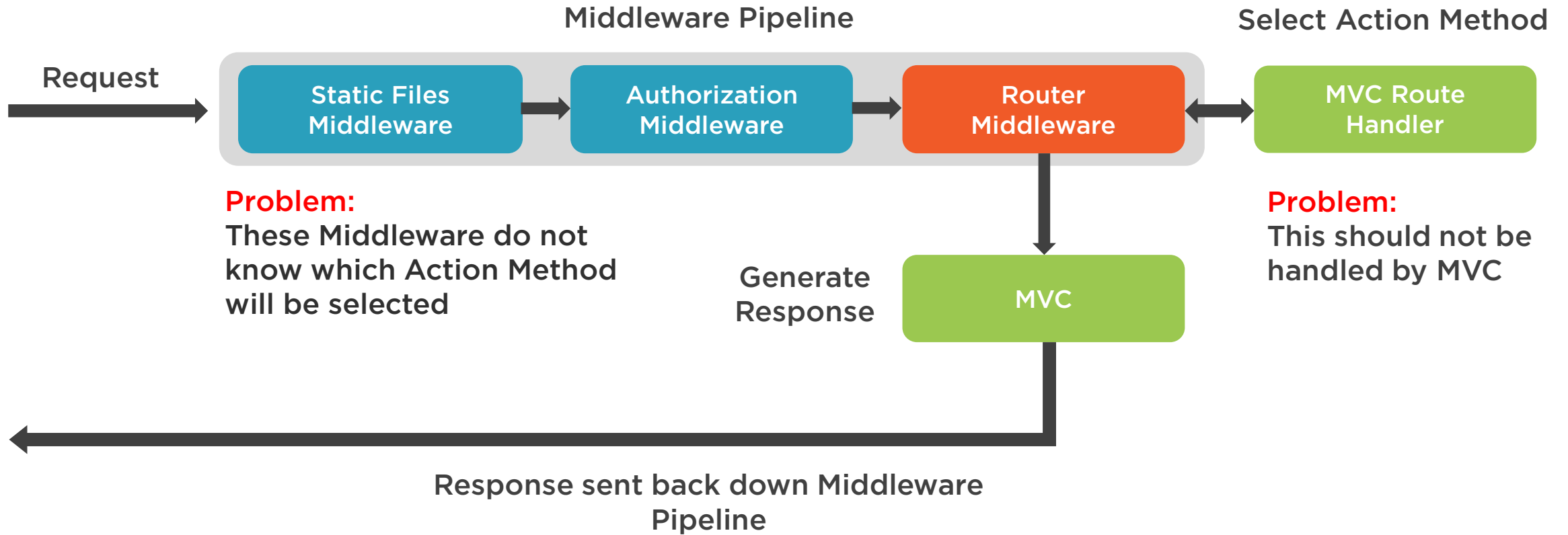
Introducing Endpoint Routing

Understanding the Legacy Routing Pipeline

Why is this a problem?

Problems with the Legacy Routing Pipeline

Middleware Pipeline

Select Action Method

Request

Static Files Middleware → **Authorization Middleware** → **Router Middleware** ↔ **MVC Route Handler**

Problem:
These Middleware do not know which Action Method will be selected

Problem:
This should not be handled by MVC

Generate Response

MVC

Response sent back down Middleware Pipeline

The Middleware Involved in Endpoint Routing
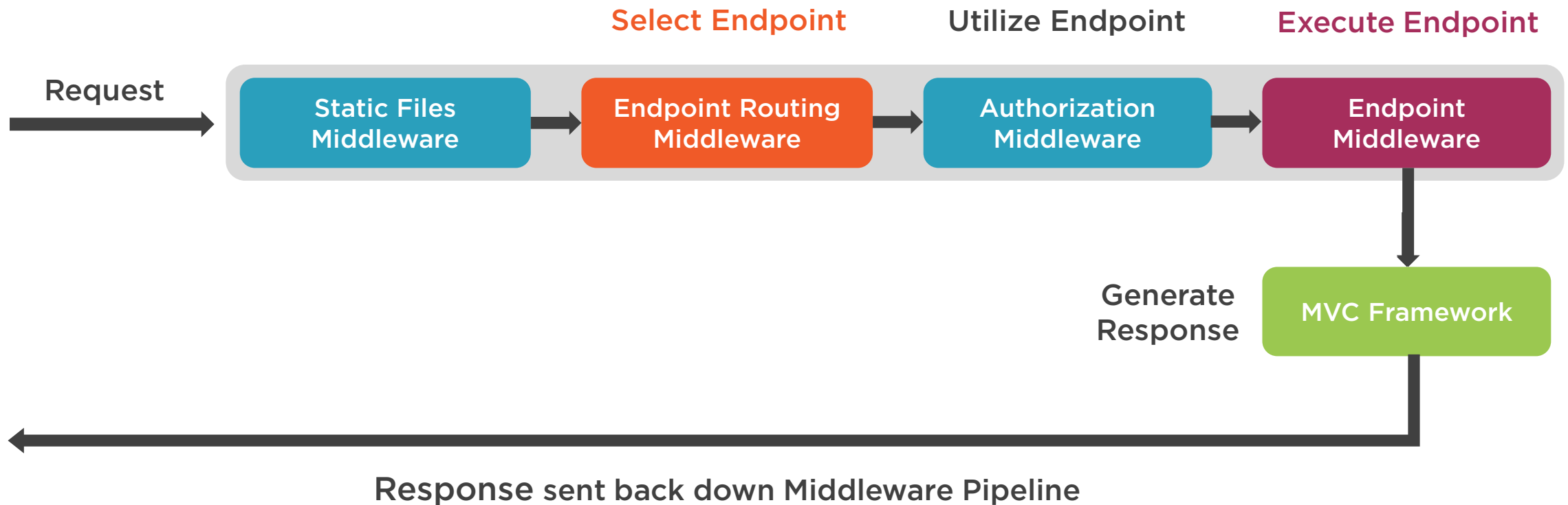
Endpoint Routing Middleware

Decides which Endpoint should handle the request

Endpoint Middleware

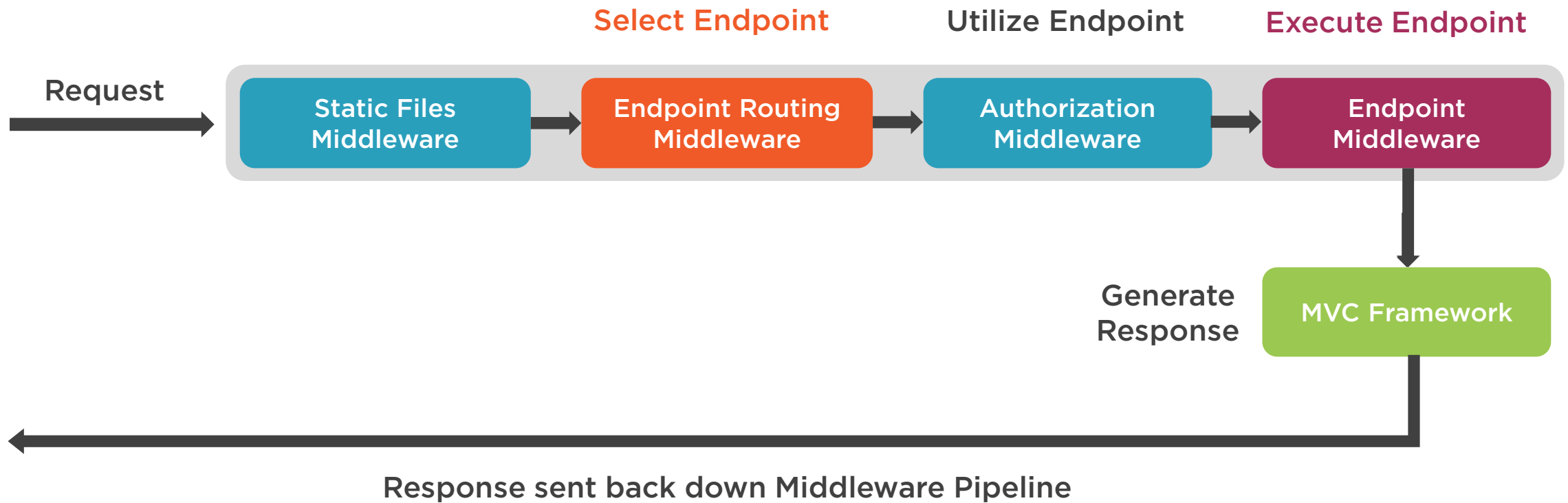**Executes the selected Endpoint to generate a response**

Understanding the Endpoint Routing Pipeline

Select Endpoint **Utilize Endpoint** **Execute Endpoint**

Request →

| Static Files Middleware | → | Endpoint Routing Middleware | → | Authorization Middleware | → | Endpoint Middleware |

Generate Response → MVC Framework

Response sent back down Middleware Pipeline

Exploring Routing Internals

Understanding the Endpoint Routing Pipeline

Select Endpoint

Utilize Endpoint

Execute Endpoint

Request

| Static Files Middleware | Endpoint Routing Middleware | Authorization Middleware | Endpoint Middleware |

Generate Response

MVC Framework

Response sent back down Middleware Pipeline

Endpoints

Classes that contain a Request Delegate and other metadata used to generate a response.

```csharp
public class RouteEndpoint : Endpoint
{

    public string DisplayName { get; set; }


    public string Pattern { get; set; }


    public EndpointMetadata Metadata { get; }


    public RequestDelegate RequestDelegate { get; }

}
```

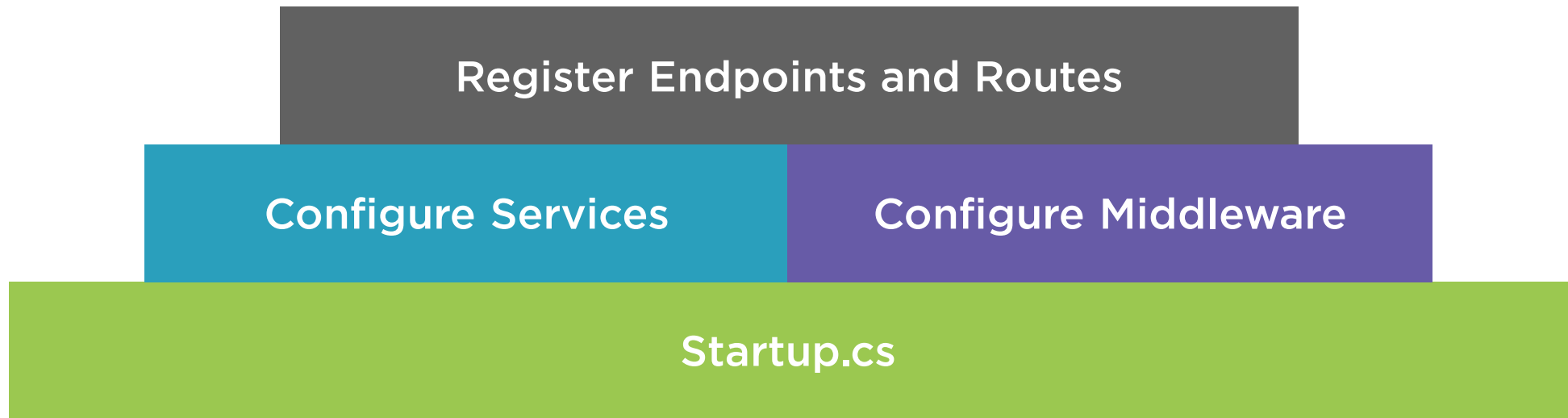◄ **The RouteEndpoint subclass**

◄ **A friendly display name**

◄ **The Route Pattern for matching**
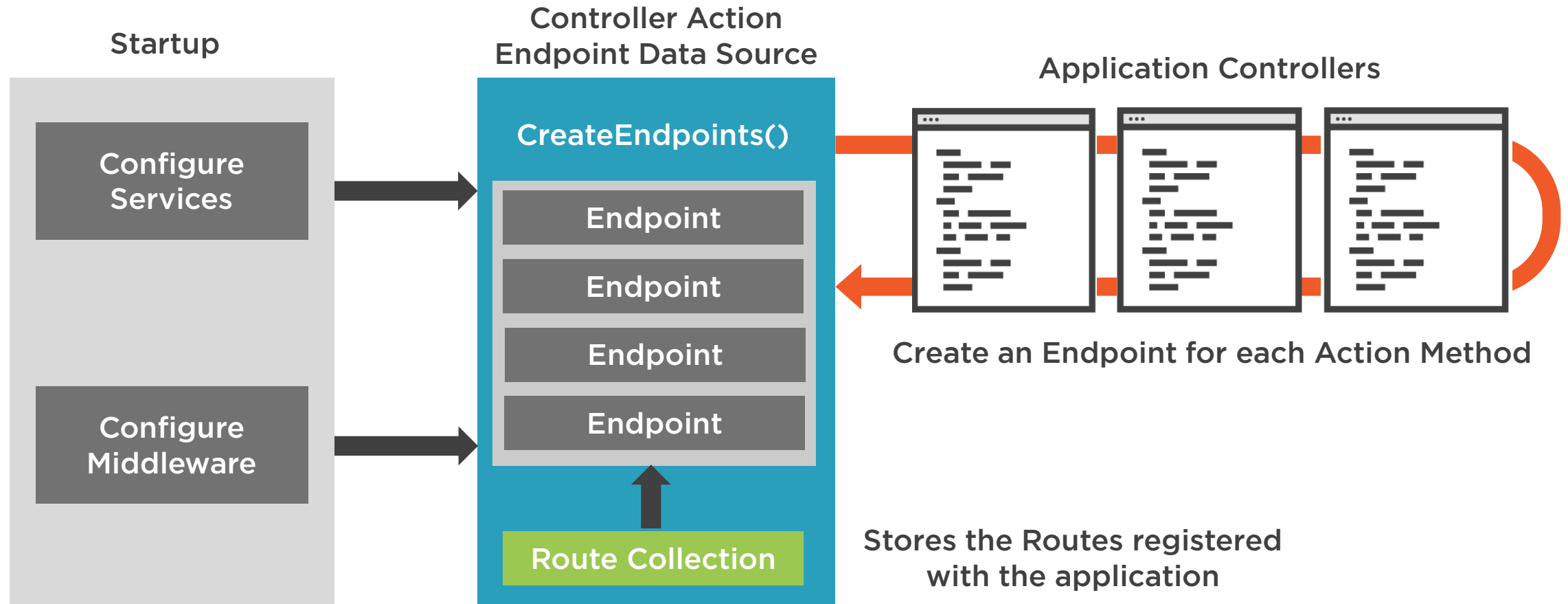
◄ **Miscellaneous Endpoint metadata**
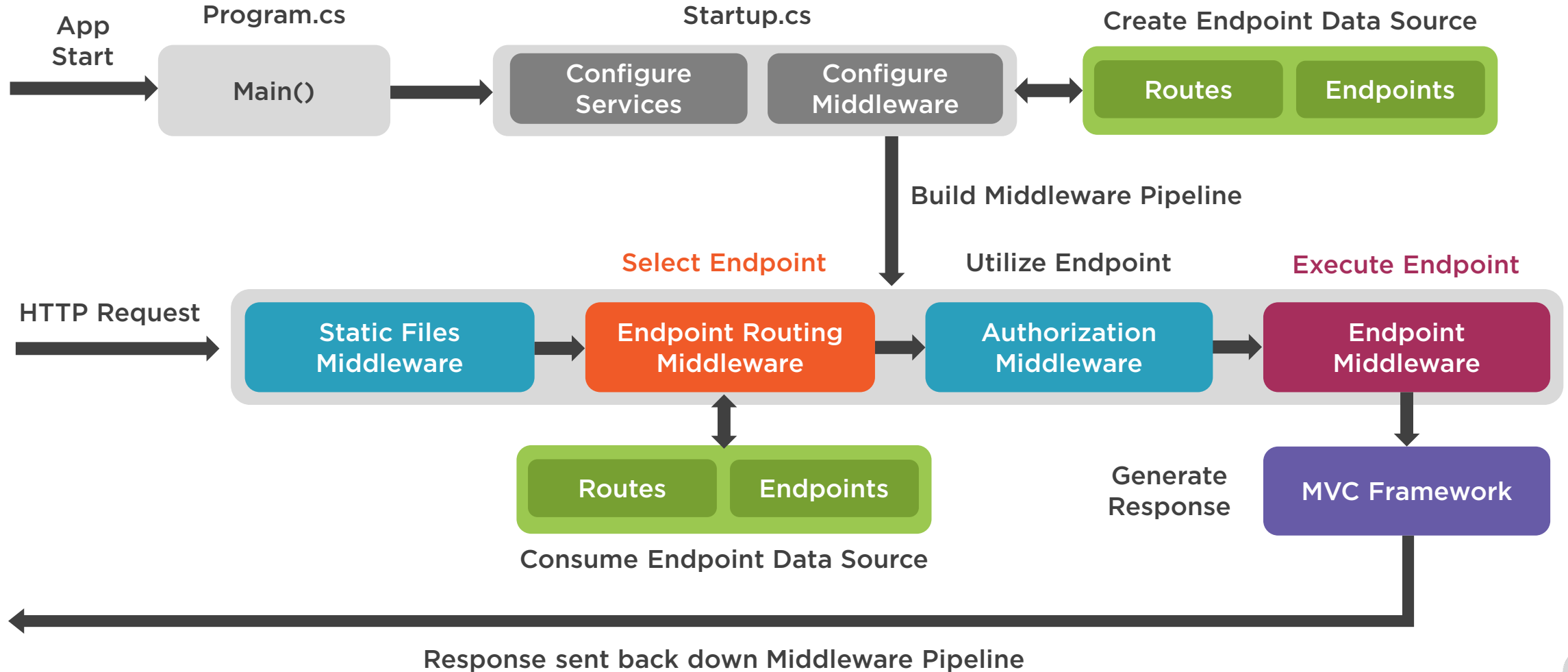
◄ **The Request Delegate to generate a response**

Building the Application Endpoints

Building the Application Endpoints (cont.)

Startup

Configure Services

Configure Middleware

Controller Action Endpoint Data Source

CreateEndpoints()

Endpoint

Endpoint

Endpoint

Endpoint

Route Collection

Application Controllers

Create an Endpoint for each Action Method

Stores the Routes registered with the application

Understanding the MVC Request Pipeline

App Start → **Program.cs** Main() → **Startup.cs** [Configure Services | Configure Middleware] → **Create Endpoint Data Source** [Routes | Endpoints]

Build Middleware Pipeline

HTTP Request → **Static Files Middleware** → **Select Endpoint** Endpoint Routing Middleware → **Utilize Endpoint** Authorization Middleware → **Execute Endpoint** Endpoint Middleware

Consume Endpoint Data Source [Routes | Endpoints]

Generate Response → **MVC Framework**

Response sent back down Middleware Pipeline

Demo

Stepping through Endpoint Routing

Expanding on Endpoint Middleware Concepts

Revisiting Application Feature Switching

Disabled

Enabled

Understanding the Endpoint Routing Pipeline

Response sent back down Middleware Pipeline
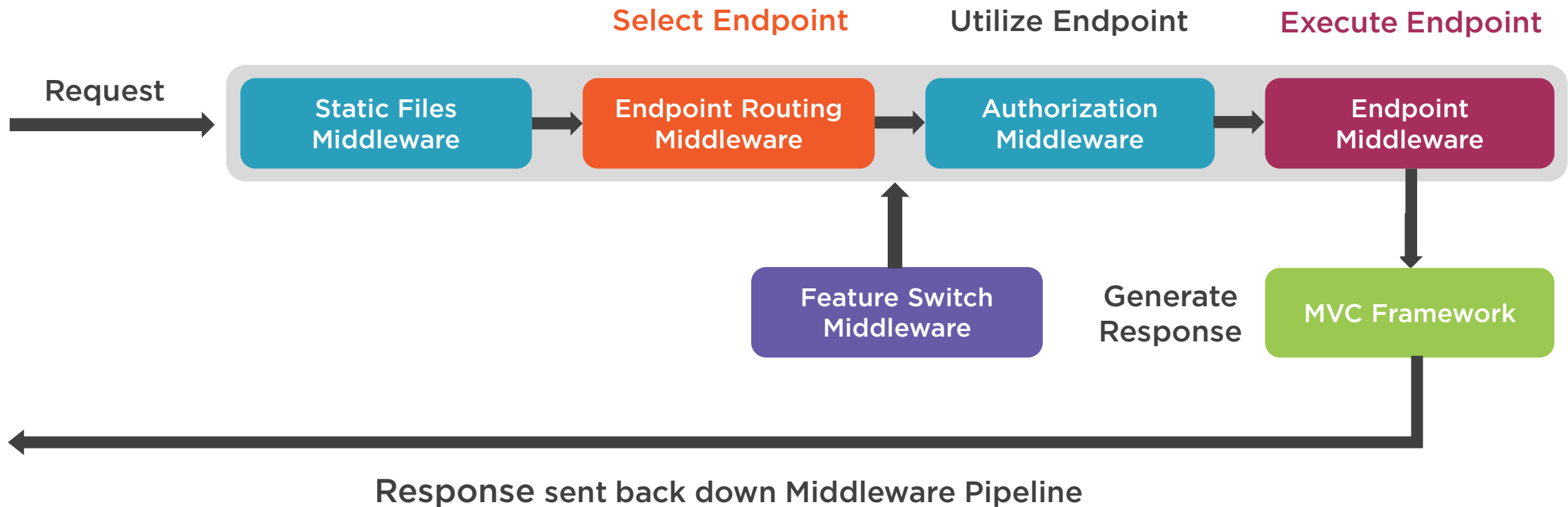
A note about feature switching.

Demo

Building Custom Endpoint Routing Components

Demo

Endpoint Routing with Razor Pages

Summary

Routing in .NET Core 3.0 maps an incoming request to an Endpoint

Endpoints are classes that represent Action Methods and Razor Pages

Two Middleware Components drive routing

- Endpoint Routing Middleware
- Endpoint Middleware

Endpoints include Route Patterns to match against an incoming URL

Endpoints also contain a Request Delegate to generate a response

The Request Delegate wraps around Controller and Action Method execution