.NET Framework: Developing Modern Web Apps with ASP.NET MVC – Workshop*PLUS*

Wael Kdouh

Senior Consultant

v1.0

Module 8: Routing

Module Overview

Module 8: Routing

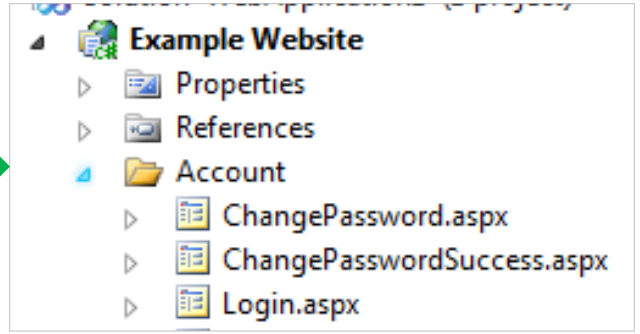
Section 1: Routing and URL Overview

Lesson: Routing and URL Overview

URL

- URL can represent physical files on disk in ASP, JSP, PHP, ASP.NET (without routing), etc.

http://example.com/
Account/Login.aspx



- ASP.NET Model-View-Controller (MVC) maps URL to action methods of Controller classes

http://example.com/
Account/Login

```
[Authorize]
public class AccountController : Controller
{
    [AllowAnonymous]
    public ActionResult Login(string returnUrl)
    {
        ViewBag.ReturnUrl = returnUrl;
        return View();
    }
```

ASP.NET MVC Routing

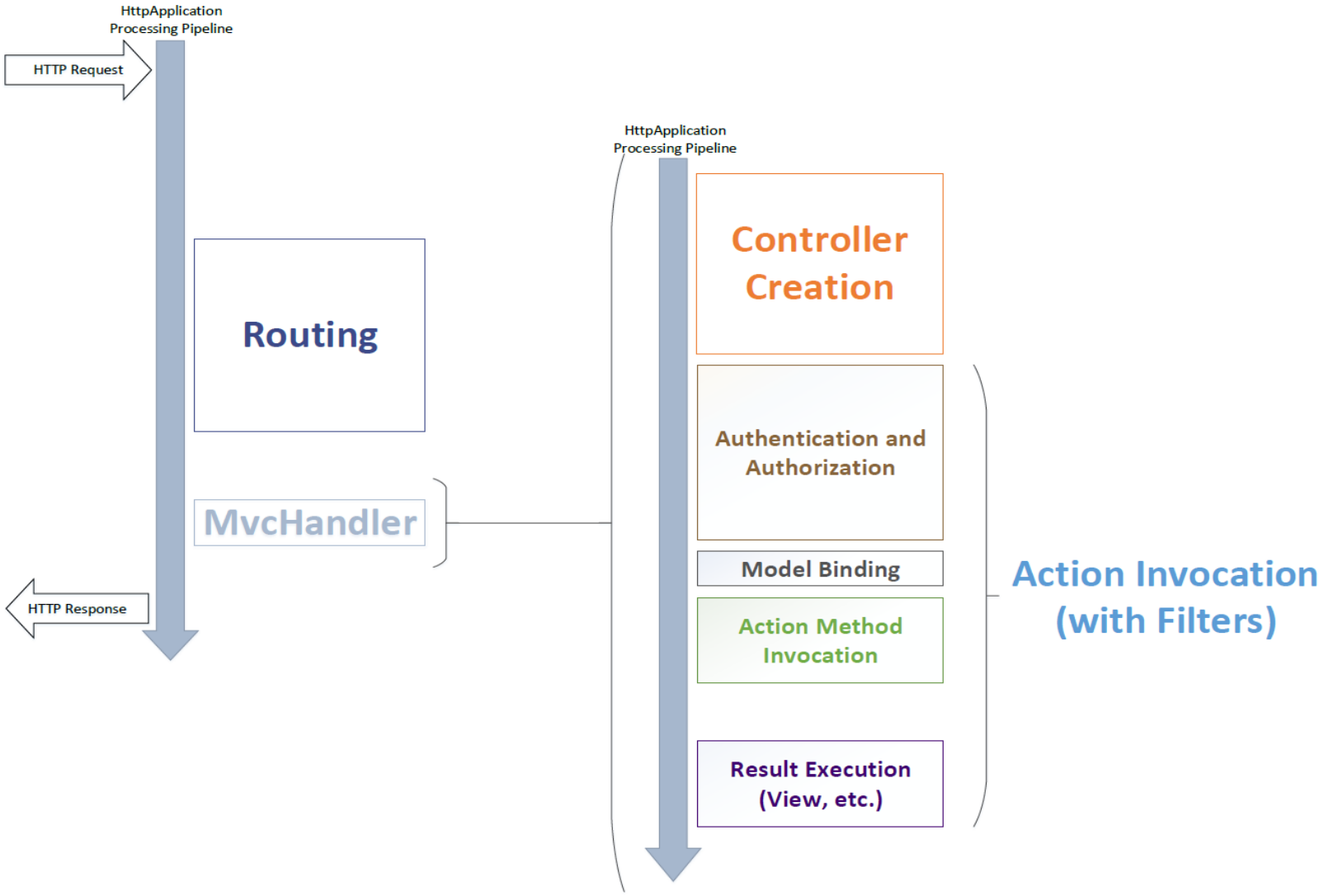
- A route is a URL pattern mapped to a handler
- Handler can be a physical file or action method in a controller
- Route instance specifies:
  - URL pattern
  - Route handler
  - Route name (optional)

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

    routes.MapRoute(
        name: "Default",
        url: "{controller}/{action}/{id}",
        defaults: new
        {
            controller = "Home",
            action = "Index",
            id = UrlParameter.Optional
        }
    );
}
```

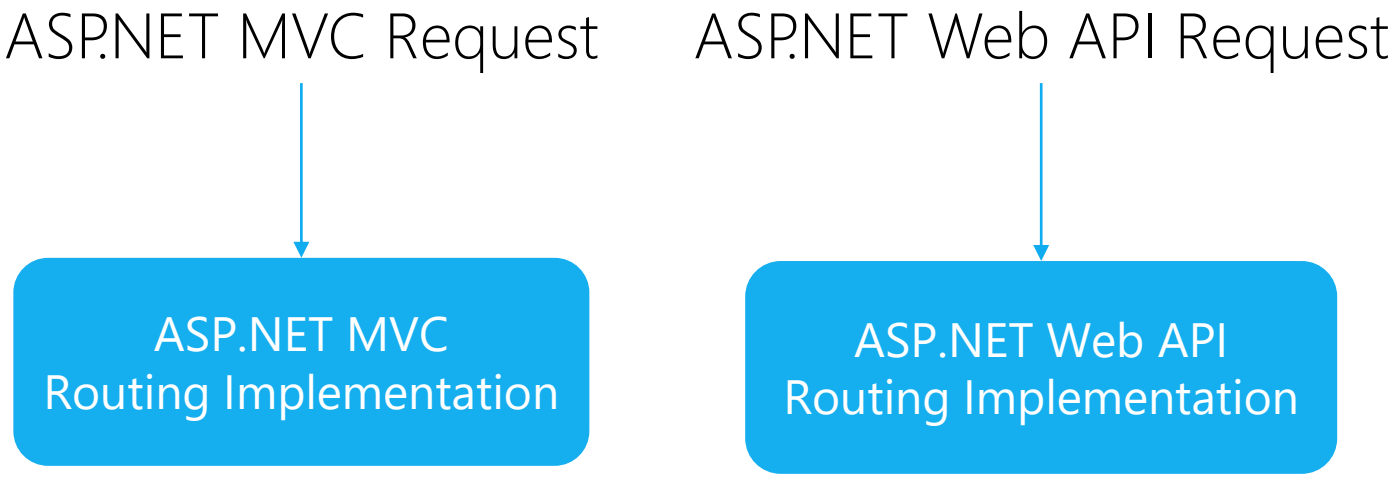
- ASP.NET MVC Routing also constructs outgoing URLs corresponding to controller actions

ASP.NET MVC: Routing in HTTP Application Processing Pipeline



Request Routing MVC5/Web API2

- Similar behavior
- Separate implementations
- Developed by two different teams in Microsoft



Module 8: Routing

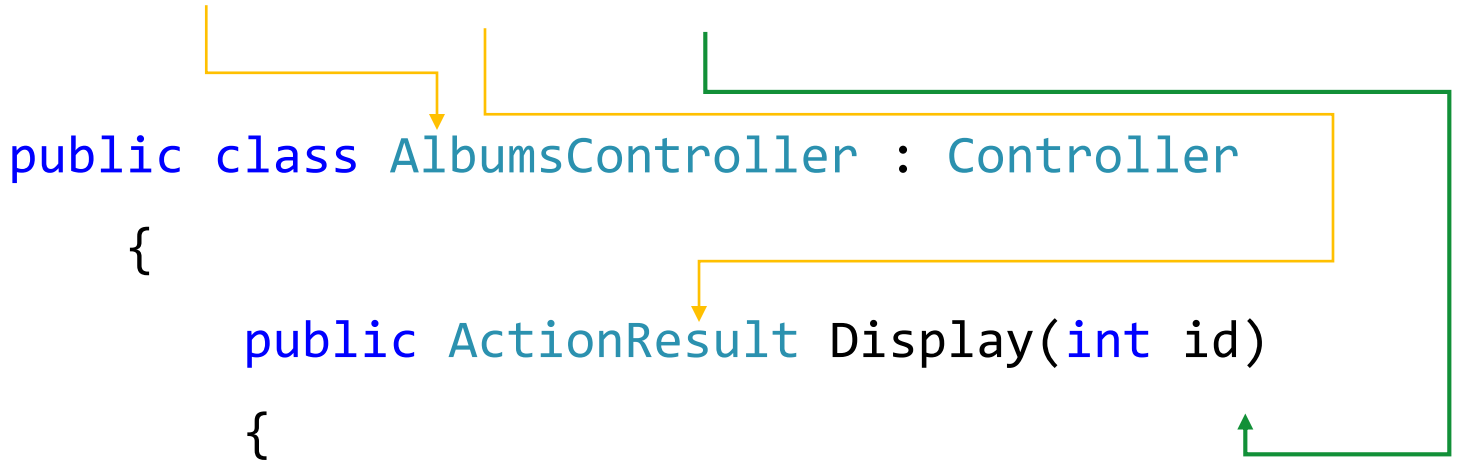
Section 3: ASP.NET MVC Routing Techniques

Lesson: Routing and MVC

Route Mapping to Controller Actions

URL Pattern: `{controller}/{action}/{id}`

URL: `~/albums/display/123`



```
public class AlbumsController : Controller
    {
        public ActionResult Display(int id)
        {
            // Do something
            return View();
        }
    }
```

Named Routes

- Always use route names to avoid ambiguities during route generation

```
@Html.RouteLink(linkText: "Test Route", routeName: "Test",
routeValues: new { controller = "Test", action = "Index", id = "123" })

@Html.RouteLink(linkText: "Default Route", routeName: "Default",
routeValues: new { controller = "Home", action = "Index", id = "123" })
```

Optional and Default Parameters

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

    routes.MapRoute(
        name: "Default",
        url: "{controller}/{action}/{id}",
        defaults: new { controller = "Home", action = "Index",
            id = UrlParameter.Optional }
    );
}
```

Route URL Pattern	Defaults	Examples of Matching URLs
{controller}/{action}/{id}	new {id = UrlParameter.Optional}	/albums/display/123 /albums/display
{controller}/{action}/{id}	new { controller="home", action="index", id = UrlParameter.Optional }	/albums/display/123 /albums/display /albums /

Route Constraints

- Constraints allow you to apply a regular expression to URL segments to restrict request matching

```
routes.MapRoute("blog", "{locale}/{year}/{month}/{day}",
    new { controller = "Blog", action = "Index" },
    new
    {
        locale = "[a-z]{2}-[A-Z]{2}",
        year = @"\d{4}",
        month = @"\d{2}",
        day = @"\d{2}"
    });
```

Example URL	Match/No-Match?
~/en-US/08	No match
~/en-US/08/05/25	No match
~/en-GB/2008/05/25	Match
~/fr-FR/2012/04/2	No match
~/fr-FR/2012/04/02	Match

Demo: MVC Routing

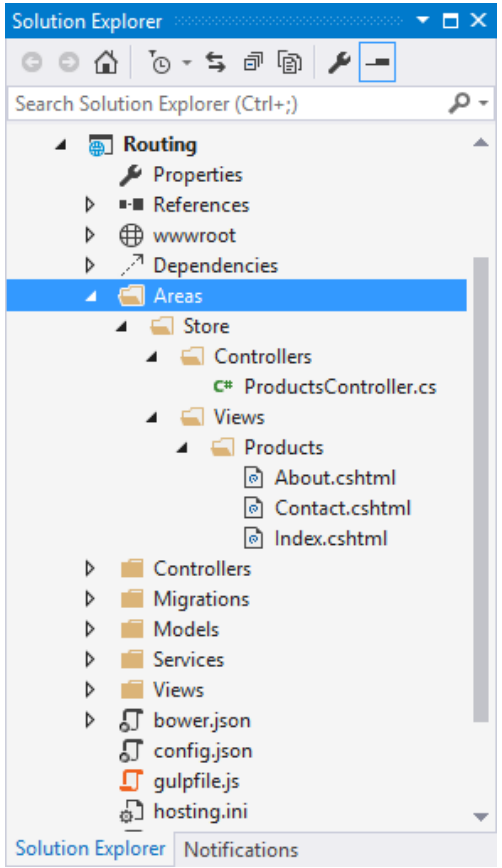
Module 8: Routing

Section 3: ASP.NET MVC Routing Techniques

Lesson: Areas

Areas

- MVC areas separate a large MVC application into smaller functional groups
  - For example, a large e-commerce site is divided into areas for storefront, product reviews, user accounts, etc.
- Area guidelines:
  - *Areas* directory must exist as project child directory
  - *Areas* contains subdirectory for each area
  - Controllers should be located at: */Areas/[area]/Controllers/[controller].cs*
  - Views should be located at: */Areas/[area]/Views/[controller]/[action].cshtml*



Area Registration

```
public class AdminAreaRegistration : AreaRegistration
{
    0 references | 0 exceptions
    public override string AreaName
    {
        get
        {
            return "Admin";
        }
    }

    0 references | 0 exceptions
    public override void RegisterArea(AreaRegistrationContext context)
    {
        context.MapRoute(
            "Admin_default",
            "Admin/{controller}/{action}/{id}",
            new { action = "Index", id = UrlParameter.Optional }
        );
    }
}
```

Area Linking from Views

```
@Html.ActionLink("See Products Home Page", "Index", "Home", new { area = "Products" }, null)
```

```
@Html.ActionLink("Go to Home Page", "Index", "Home", new { area = "" }, null)
```

Demo: Area

Module 8: Routing

Section 3: ASP.NET MVC Routing Techniques

Lesson: Attribute Routing

Convention-Based Routing vs. Attribute Routing

Convention-based Routing

```
config.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{id}",
    defaults: new { id = RouteParameter.Optional }
);
```

Attribute Routing

```
[Route("{productId:int}/{productTitle}")]
public IActionResult Show(int productId) { ... }
```

Optional and Default Parameters

```
public class BooksController : ApiController
{
    [Route("api/books/locale/{lcid:int?}")]
    public IEnumerable<Book> GetBooksByLocale(int lcid = 1033) { ... }
}
```

```
public class BooksController : ApiController
{
    [Route("api/books/locale/{lcid:int=1033}")]
    public IEnumerable<Book> GetBooksByLocale(int lcid) { ... }
}
```

Common Route Prefix

```
[RoutePrefix("api/books")]
public class BooksController : ApiController
{
    // GET /api/authors/1/books
    [Route("~/api/authors/{authorId:int}/books")]
    public IEnumerable<Book> GetByAuthor(int authorId)
```

```
[RoutePrefix("customers/{customerId}")]
public class OrdersController : ApiController
{
    // GET customers/1/orders
    [Route("orders")]
    public IEnumerable<Order> Get(int customerId)
}
```

Inline Constraints

Constraint	Description	Example Template
alpha	Matches uppercase or lowercase Latin alphabet characters (a-z, A-Z)	"Product/{ProductName:alpha}"
int	Matches a Signed 32-bit integer value	"Product/{ProductId:int}"
long	Matches a Signed 64-bit integer value	"Product/{ProductId:long}"
minlength	Matches a string with a minimum length	"Product/{ProductName:minlength(10)}"
regex	Matches a regular expression	"Product/{productId:regex(^\\d{4}$)}"

Route Constraints

```
// eg: /users/5
[Route("users/{id:int}"]
public ActionResult GetUserById(int id) { ... }
// eg: users/ken
[Route("users/{name}"]
public ActionResult GetUserByName(string name) { ... }

// eg: /users/5 but not /users/10000000000 because it is larger than int.MaxValue, and not /users/0 because of the min(1) constraint.
[Route("users/{id:int:min(1)}")]
public ActionResult GetUserById(int id) { ... }
// eg: /greetings/bye and /greetings because of the Optional modifier,
// but not /greetings/see-you-tomorrow because of the maxlength(3) constraint.
[Route("greetings/{message:maxlength(3)?}")]
public ActionResult Greet(string message) { ... }
```

Demo: Attribute Routing

Module 8: Routing

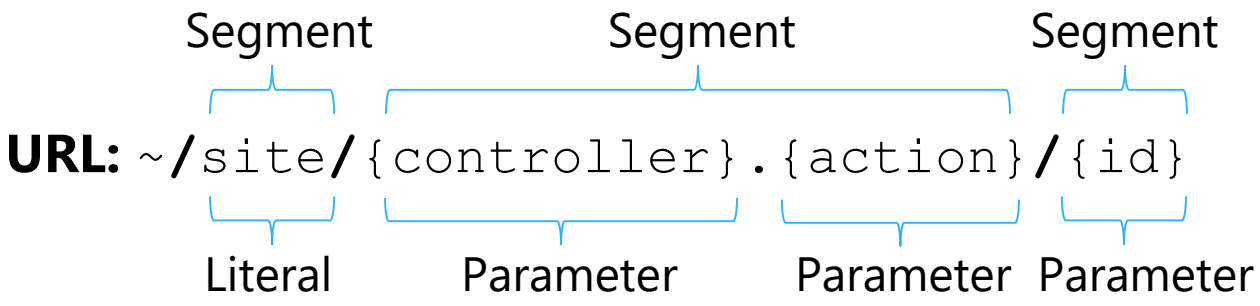
Section 3: ASP.NET MVC Routing Techniques

Lesson: Parameters

URL Parameter Value Mapping

URL Pattern: `{first}/{second}/{third}`

URL	URL Parameter Values
~/Products/Show/123	first = “Products”; second = “Show” third = “123”
~/electronics/pcs/baz	first = “electronics”; second = “pcs” third = “baz”
~/a.b/b-c	first = “a.b”; second = “b-c” third = “”



Overflow Parameters

- Overflow parameters are route values that are not specified in the route's definition
  - Appended to generated URL as query string parameters

```
routes.MapRoute("Reports", "reports/{year}/{month}/{day}", new { day = 1 });
```

Parameters	Resulting URL	Reason
year=2010, month=1, day=12	/reports/2010/1/12	Straightforward match
year=2010, month=1	/reports/2010/1/1	Default for day = 1
year=2010, month=1, day=12, category=64	/reports/2010/1/12?category=64	Overflow parameters go into query string
Year=2007	null	Required parameters not provided

URL Patterns

Route Definition	Example of Matching URL
{controller}/{action}/{id}	~/Products/show/beverages
{table}/Details.aspx	~/Products/Details.aspx
blog/{action}/{entry}	~/blog/show/123
{reporttype}/{year}/{month}/{day}	~/sales/2008/1/5
{locale}/{action}	~/US/show
{language}-{country}/{action}	~/en-US/show
{controller}.{action}.{id}	~/Products.Show.123

Multiple URL Parameters in a Segment

- Route URL may have multiple parameters in a segment
- Parameters cannot be adjacent to avoid ambiguity

Route URL	Request URL	Route Data Result
{filename}.{ext}	~/Foo.xml.aspx	filename="Foo.xml" ext="aspx"
My{title}-{cat}	~/MyHouse-dwelling	title="House" cat="dwelling"
{foo}xyz{bar}	~/xyzxyzxyzblah	foo="xyzxyz" bar="blah"
{title}{artist}	-	-
{Filename}{ext}	-	-

Catch-All Parameter

- Catch-All parameter allows for route to match arbitrary number of segments
- Catch-All parameter in URL Pattern:

```
"{controller}/{action}/{id}/{*ExtraParam}"
```

Example

- URL: `~/Home/Index/1234/523/89`
- RouteDebugger Output:

Matched Route: {controller}/{action}/{id}/{*ExtraParam}

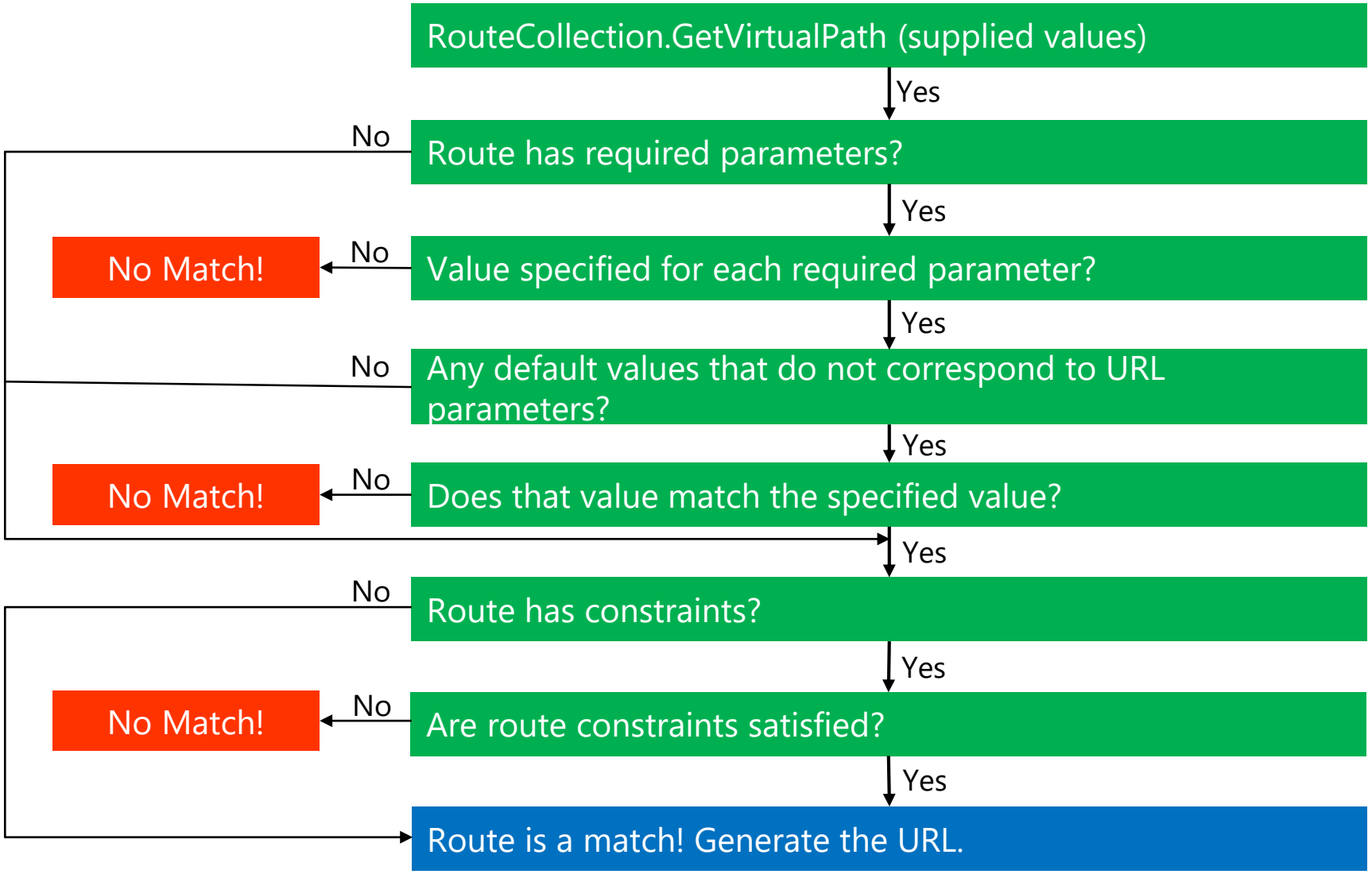
Route Data		Data Tokens	
Key	**Value**	**Key**	**Value**
controller	Home		
action	Index		
id	1234		
ExtraParam	523/89		

Module 8: Routing

Section 3: ASP.NET MVC Routing Techniques

Lesson: More Routing

URL Generation



Routing vs. URL Rewriting

Routing	URL Rewriting
Used for mapping a URL to a resource	Often used to map old URLs to a new set of URLs
Routing embodies resource-centric view; never rewrites URL	Rewrites URLs to correctly map to the resource
Routing helps generate URLs using the same routing rules	URL rewriting only applies to incoming requests
Performed at ASP.NET level	Besides ASP.NET, it can be implemented with Internet Server API (ISAPI) filters at Internet Information Services (IIS) level

```
routes.MapRoute(
    name: "Default",
    url: "{controller}/{action}/{id}",
    defaults: new {
        controller = "Home",
        action = "Index",
        id = UrlParameter.Optional });
```

```
<RewriterConfig>
   <Rules>
      <!-- Rules for Product Lister -->
      <RewriterRule>
         <LookFor>~/Products/Beverages\.aspx</LookFor>
         <SendTo>~/ListProductsByCategory.aspx?CategoryID=1</SendTo>
      </RewriterRule>
      <RewriterRule>
   </Rules>
</RewriterConfig>
```

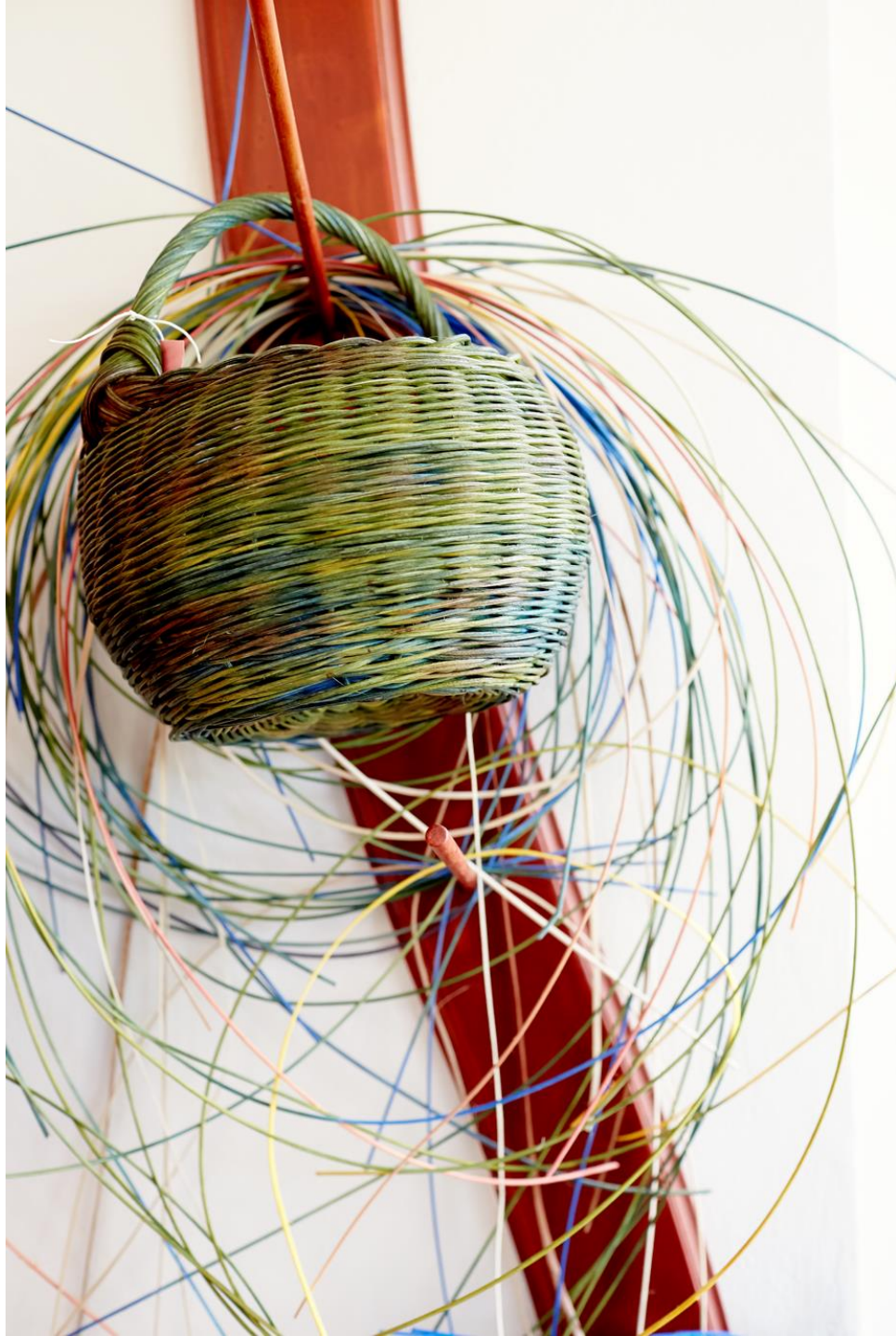
Routing Extension

Interface	Description
IHttpControllerSelector	Selects the controller.
IHttpControllerTypeResolver	Gets the list of controller types. The DefaultHttpControllerSelector chooses the controller type from this list.
IAssembliesResolver	Gets the list of project assemblies. The IHttpControllerTypeResolver interface uses this list to find the controller types.
IHttpControllerActivator	Creates new controller instances.
IHttpActionSelector	Selects the action.
IHttpActionInvoker	Invokes the action.

```
var config = GlobalConfiguration.Configuration;
config.Services.Replace(typeof(IHttpControllerSelector), new MyControllerSelector(config));
```

Module Summary

- In this you learned about:
  - Usability guidelines for URLs
  - ASP.NET MVC Routing
  - Conventional Routing
  - Attribute Routing
  - MVC areas and their route registration
  - URL generation through routing rules
  - Request routing pipeline



Lab: Routing


Microsoft