StrathWeb.

A free flowing web tech monologue.

(https://www.strathweb.com)

ABOUT ME

(http://twitter.com/filip_woj)

I'm Filip W. (http://twitter.com/filip_woj), a .NET developer. I like Toronto Maple Leafs (http://mapleleafs.com), Rancid (http://rancidrancid.com) and ASP.NET (http://www.asp.net). Oh and I love the Lowlands (http://en.wikipedia.org/wiki/Scottish_Lowlands).

Follow @filip_woj

CATEGORIES

.net (https://www.strathweb.com/category/net/) (135)

.net core (https://www.strathweb.com/category/net-core/) (38)

apache cordova (https://www.strathweb.com/category/apache-cordova/) (1)

asp.net 5 (https://www.strathweb.com/category/asp-net-5/) (17)

asp.net core (https://www.strathweb.com/category/asp-net-core/) (36)

asp.net mvc (https://www.strathweb.com/category/asp-

OCT 6, 2014 | .NET (HTTPS://WWW.STRATHWEB.COM/CATEGORY/NET/) ASP.NET WEB API (HTTPS://WWW.STRATHWEB.COM/CATEGORY/ASP-NET-WEB-API/) C# (HTTPS://WWW.STRATHWEB.COM/CATEGORY/C/)

Route matching and overriding 404 in ASP.NET Web API (https://www.strathweb.com/2014/10/route-matching-overriding-404-asp-net-web-api/)

In ASP.NET Web API, if the incoming does not match any route, the framework is simply hard wired to return 404 to the client (or possibly pass through to the next configured middleware, in case of an OWIN hosted Web API). This is done immediately, without entering anywhere further down the pipeline (i.e. message handlers would not be invoked).

However, an interesting question was posted recently at StackOverflow (http://stackoverflow.com/questions/26156750/how-to-set-asp-net-web-api-route-constraint-http-status-return-code/26201869) – what if you want to override that hard 404, and given your specific routing requirements, respond to the client with a different status code if a specific route condition fails?

I already answered at StackOverflow (http://stackoverflow.com/questions/26156750/how-to-set-asp-net-web-api-route-constraint-http-status-return-code/26201869#26201869), but decided this deserves a blog post regardless.

Default behavior of route matching

By default, all incoming HTTP requests are handled by an `HttpServer` instance, which uses `HttpRoutingDispatcher` (a specialized `HttpMessageHandler`) to determine if there is a Web API route that matches the request.

If no match is found, `HttpRoutingDispatcher` will arbitrarily short circuit a 404 response without letting anything else in the Web API pipeline touch the request anymore.

Contrary to most Web API services, `HttpRoutingDispatcher` is not resolved from anywhere, but rather newed up directly right in the heart of the framework, so swapping it with alternative implementation (to changed the 404 behaviour) is pretty much impossible.

The problem

net-mvc/) (35)

asp.net mvc 6 (https://www.strathweb.com/category/asp-net-mvc-6/) (7)

asp.net vnext (https://www.strathweb.com/category/asp-net-vnext/) (6)

asp.net web api (https://www.strathweb.com/category/asp-net-web-api/) (96)

azure (https://www.strathweb.com/category/azure/) (9)

azure service bus (https://www.strathweb.com/category/azure-service-bus/) (1)

benchmark dotnet (https://www.strathweb.com/category/benchmark-dotnet/) (1)

bing maps (https://www.strathweb.com/category/bing-maps/) (1)

blazor (https://www.strathweb.com/category/blazor/) (1)

c plus (https://www.strathweb.com/category/c-plus/) (2)

c# (https://www.strathweb.com/category/c/) (137)

dnx (https://www.strathweb.com/category/dnx/) (3)

dotnet-cli (https://www.strathweb.com/category/dotnet-cli/) (2)

dotnet-script (https://www.strathweb.com/category/dotnet-script/) (5)

entity framework (https://www.strathweb.com/category/entity-framework/) (1)

espn api (https://www.strathweb.com/category/espn-api/) (2)

events (https://www.strathweb.com/category/events/) (1)

fsharp (https://www.strathweb.com/category/fsharp/) (1)

So if you visit that StackOverflow question (http://stackoverflow.com/questions/26156750/how-to-set-asp-net-web-api-route-constraint-http-status-return-code/26201869), the OP wants to respond with 412 (Precondition Failed) instead of 404 if the route is matched, but his constraint is not fulfilled.

C#

```csharp
[Route("Number/{id:int:min(2):max(10)}")]
public HttpResponseMessage GetNumber([FromUri] int id)
{
    //return 200 from here
    //if the constraint in the route failed return 412 not 404
}
```

Implementing a smart IHttpRouteConstraint

To create custom constraining, you need to write a class that implements `IHttpRouteConstraint`. The interface is simple, as it only has a single method:

C#

```csharp
public interface IHttpRouteConstraint
{
    bool Match(HttpRequestMessage request, IHttpRoute route, string parameterName, IDictionary<string, object> values, HttpRouteDirection routeDirection);
}
```

The principle is very straight forward – you have access to all the necessary information such as the route parameter name, its values supplied form the client side and the request and route information. Based on that, if your route pre-condition is matched, return `true`, otherwise, return `false`.

The trick is, that you can also throw an exception – and if it's an `HttpResponseException`, the Web API infrastructure (the `HttpServer`) will catch it and convert it into an `HttpResponseMessage` (if the exception carries a relevant status code) or use the instance of the `HttpResponseMessage` attached to the exception itself.

The range constraint, with a custom status code, which addresses our problem, is shown below.

C#

git (https://www.strathweb.com/category/git/) (1)

glimpse (https://www.strathweb.com/category/glimpse/) (1)

html5 (https://www.strathweb.com/category/html-5/) (4)

identity server (https://www.strathweb.com/category/identity-server/) (2)

iis (https://www.strathweb.com/category/iis/) (2)

javascript (https://www.strathweb.com/category/javascript/) (9)

jquery (https://www.strathweb.com/category/jquery/) (4)

jquery mobile metro (https://www.strathweb.com/category/jquery-mobile-metro/) (1)

katana (https://www.strathweb.com/category/katana/) (2)

kindle (https://www.strathweb.com/category/kindle/) (1)

knockout.js (https://www.strathweb.com/category/knockout-js/) (8)

last.fm api (https://www.strathweb.com/category/lastfm-api/) (2)

linq (https://www.strathweb.com/category/linq/) (1)

mac (https://www.strathweb.com/category/mac/) (2)

msbuild (https://www.strathweb.com/category/msbuild/) (2)

mvc core (https://www.strathweb.com/category/mvc-core/) (2)

nancy (https://www.strathweb.com/category/nancy/) (2)

```
public class RangeWithStatusRouteConstraint : IHttpRouteConstraint
{
    private readonly int _from;
    private readonly int _to;
    private readonly HttpStatusCode _statusCode;

    public RangeWithStatusRouteConstraint(int from, int to, string statusCode)
    {
        _from = from;
        _to = to;
        if (!Enum.TryParse(statusCode, true, out _statusCode))
        {
            _statusCode = HttpStatusCode.NotFound;
        }
    }

    public RangeWithStatusRouteConstraint(int from, int to, HttpStatusCode statusCode)
    {
        _from = from;
        _to = to;
        _statusCode = statusCode;
    }

    public bool Match(HttpRequestMessage request, IHttpRoute route, string parameterName, IDictionary<string, object> values, HttpRouteDirection routeDirection)
    {
        object value;
        if (values.TryGetValue(parameterName, out value) &amp;&amp; value != null)
        {
            var stringValue = value as string;
            var intValue = 0;

            if (stringValue != null &amp;&amp; int.TryParse(stringValue, out intValue))
            {
                if (intValue >= _from &amp;&amp; intValue <= _to)
                {
                    return true;
                }

                //only throw if we had the expected type of value
                //but it fell out of range
                throw new HttpResponseException(_statusCode);
            }
        }

        return false;
    }
}
```

Note that in attribute routing, all constraint parameters have to be passed as primitives so the status code is passed as string and then cast to the `HttpStatusCode` enumeration. Alternatively, you can use an `int` there, since that would cast correctly too. The second constructor (with strongly typed status code) will be used in centralized routing.

If the route value falls outside of the expected range, we throw the Exception, instead of returning false and relying on `HttpRoutingDispatcher` to issue a 404. If the value is not an `int` then we do not throw, since the route is not matched correctly anyway. Thanks to this, we can drop the additional integer constraint – since our range will already require it to be an integer anyway.

Remember that this is a greedy constraint – if there is another route with similar structure it would not be reached anymore, as this constraint is going to force a specific response explicitly in case it’s not matched.

Applying the constraint to attribute routing

In order to use this with attribute routing you need to give the constraint a name in the `ConstraintMap`. This is that at the application startup, through `DefaultInlineConstraintResolver`.

native (https://www.strathweb.com/category/native/) (1)

ninject (https://www.strathweb.com/category/ninject/) (2)

odata (https://www.strathweb.com/category/odata/) (4)

omnisharp (https://www.strathweb.com/category/omnisharp/) (4)

osx (https://www.strathweb.com/category/osx/) (2)

owin (https://www.strathweb.com/category/owin/) (5)

php (https://www.strathweb.com/category/php/) (1)

roslyn (https://www.strathweb.com/category/roslyn/) (26)

scriptcs (https://www.strathweb.com/category/scriptcs/) (11)

scripting (https://www.strathweb.com/category/scripting/) (6)

servicestack (https://www.strathweb.com/category/servicestack/) (2)

signalr (https://www.strathweb.com/category/signalr/) (7)

swift (https://www.strathweb.com/category/swift/) (2)

testing (https://www.strathweb.com/category/testing/) (4)

twitter boostrap (https://www.strathweb.com/category/twitter-boostrap/) (1)

typescript (https://www.strathweb.com/category/typescript/) (1)

visual studio (https://www.strathweb.com/category/visual-studio/) (3)

Instead of simply saying:

```C#
config.MapHttpAttributeRoutes();
```

You have to say:

```C#
var constraintResolver = new DefaultInlineConstraintResolver();
constraintResolver.ConstraintMap.Add("rangeWithStatus", typeof(RangeWithStatusRouteConstraint));
config.MapHttpAttributeRoutes(constraintResolver);
```

With such setup, you can now modify the original route to:

```C#
[Route("Number/{id:rangeWithStatus(2, 10, PreconditionFailed)}")]
public HttpResponseMessage GetNumber([FromUri] int id)
{
    return Request.CreateResponse(HttpStatusCode.OK, id);
}
```

Such setup ensures a 412 response in case the routing constraint fails.

Applying the constraint to centralized routing

So far we dealt with direct routing (attribute routing), but the same constraint works equally well with centralized routing. You just need to use the appropriate overload of the `MapHttpRoute` method.

```C#
        config.Routes.MapHttpRoute(
                name: "MyRoute",
                routeTemplate: "Number/{id}",
                defaults: new { controller = "Test" },
                constraints: new { id = new RangeWithStatusRouteConstraint(2, 10, HttpStatusCode.PreconditionFailed) }
                );
```

In this case, we can use the other constructor of our `RangeWithStatusRouteConstraint` and pass the status code in the enum format directly visual studio code (https://www.strathweb.com/category/visual-studio-code/) (3)

windows phone 7 (https://www.strathweb.com/category/windows-phone-7/) (1)

wordpress (https://www.strathweb.com/category/wordpress/) (1)

WPF (https://www.strathweb.com/category/wpf/) (2)

ARCHIVES

June 2019 (https://www.strathweb.com/2019/06/) (1)

May 2019 (https://www.strathweb.com/2019/05/) (1)

April 2019 (https://www.strathweb.com/2019/04/) (1)

March 2019 (https://www.strathweb.com/2019/03/) (1)

February 2019 (https://www.strathweb.com/2019/02/) (1)

January 2019 (https://www.strathweb.com/2019/01/) (2)

December 2018 (https://www.strathweb.com/2018/12/) (1)

November 2018 (https://www.strathweb.com/2018/11/) (1)

October 2018 (https://www.strathweb.com/2018/10/) (2)

September 2018 (https://www.strathweb.com/2018/09/) (1)

August 2018 (https://www.strathweb.com/2018/08/) (2)

July 2018 (https://www.strathweb.com/2018/07/) (3)

June 2018 (https://www.strathweb.com/2018/06/) (1)

May 2018 (https://www.strathweb.com/2018/05/) (2)

April 2018 (https://www.strathweb.com/2018/04/) (2)

March 2018 (https://www.strathweb.com/2018/03/) (1)

February 2018 (https://www.strathweb.com/2018/02/) (1)

January 2018 (https://www.strathweb.com/2018/01/) (2)

December 2017 (https://www.strathweb.com/2017/12/) (2)

November 2017 (https://www.strathweb.com/2017/11/) (1)

October 2017 (https://www.strathweb.com/2017/10/) (1)

September 2017 (https://www.strathweb.com/2017/09/) (2)

August 2017 (https://www.strathweb.com/2017/08/) (1)

July 2017 (https://www.strathweb.com/2017/07/) (1)

June 2017 (https://www.strathweb.com/2017/06/) (3)

May 2017 (https://www.strathweb.com/2017/05/) (1)

April 2017 (https://www.strathweb.com/2017/04/) (1)

March 2017 (https://www.strathweb.com/2017/03/) (1)

January 2017 (https://www.strathweb.com/2017/01/) (2)

December 2016 (https://www.strathweb.com/2016/12/) (2)

November 2016 (https://www.strathweb.com/2016/11/) (2)

October 2016 (https://www.strathweb.com/2016/10/) (1)

September 2016 (https://www.strathweb.com/2016/09/) (4)

August 2016 (https://www.strathweb.com/2016/08/) (1)

July 2016 (https://www.strathweb.com/2016/07/) (2)

June 2016 (https://www.strathweb.com/2016/06/) (4)

May 2016 (https://www.strathweb.com/2016/05/) (3)

April 2016 (https://www.strathweb.com/2016/04/) (3)

March 2016 (https://www.strathweb.com/2016/03/) (3)

February 2016 (https://www.strathweb.com/2016/02/) (2)

January 2016 (https://www.strathweb.com/2016/01/) (1)

December 2015 (https://www.strathweb.com/2015/12/) (2)

November 2015 (https://www.strathweb.com/2015/11/) (2)

October 2015 (https://www.strathweb.com/2015/10/) (2)

September 2015 (https://www.strathweb.com/2015/09/) (3)

August 2015 (https://www.strathweb.com/2015/08/) (1)

July 2015 (https://www.strathweb.com/2015/07/) (2)

June 2015 (https://www.strathweb.com/2015/06/) (1)

May 2015 (https://www.strathweb.com/2015/05/) (1)

April 2015 (https://www.strathweb.com/2015/04/) (2)

March 2015 (https://www.strathweb.com/2015/03/) (1)

February 2015 (https://www.strathweb.com/2015/02/) (1)

January 2015 (https://www.strathweb.com/2015/01/) (2)

December 2014 (https://www.strathweb.com/2014/12/) (2)

November 2014 (https://www.strathweb.com/2014/11/) (2)

October 2014 (https://www.strathweb.com/2014/10/) (3)

September 2014 (https://www.strathweb.com/2014/09/) (2)

August 2014 (https://www.strathweb.com/2014/08/) (2)

July 2014 (https://www.strathweb.com/2014/07/) (2)

June 2014 (https://www.strathweb.com/2014/06/) (3)

May 2014 (https://www.strathweb.com/2014/05/) (1)

April 2014 (https://www.strathweb.com/2014/04/) (2)

March 2014 (https://www.strathweb.com/2014/03/) (2)

February 2014 (https://www.strathweb.com/2014/02/) (3)

January 2014 (https://www.strathweb.com/2014/01/) (1)

December 2013
(https://www.strathweb.com/2013/12/) (3)

November 2013
(https://www.strathweb.com/2013/11/) (2)

October 2013
(https://www.strathweb.com/2013/10/) (2)

September 2013
(https://www.strathweb.com/2013/09/) (2)

August 2013
(https://www.strathweb.com/2013/08/) (3)

July 2013
(https://www.strathweb.com/2013/07/) (3)

June 2013
(https://www.strathweb.com/2013/06/) (3)

May 2013
(https://www.strathweb.com/2013/05/) (3)

April 2013
(https://www.strathweb.com/2013/04/) (4)

March 2013
(https://www.strathweb.com/2013/03/) (5)

February 2013
(https://www.strathweb.com/2013/02/) (4)

January 2013
(https://www.strathweb.com/2013/01/) (7)

December 2012
(https://www.strathweb.com/2012/12/) (3)

November 2012
(https://www.strathweb.com/2012/11/) (5)

October 2012
(https://www.strathweb.com/2012/10/) (5)

September 2012
(https://www.strathweb.com/2012/09/) (5)

August 2012
(https://www.strathweb.com/2012/08/) (6)

July 2012
(https://www.strathweb.com/2012/07/) (5)

June 2012
(https://www.strathweb.com/2012/06/) (6)

May 2012
(https://www.strathweb.com/2012/05/) (6)

April 2012
(https://www.strathweb.com/2012/04/) (5)

March 2012
(https://www.strathweb.com/2012/03/) (6)

February 2012
(https://www.strathweb.com/2012/02/) (1)