Forwarding Requests in a REST Service

Article
Michael Smart · Oct 7, 2016 4m read

Forwarding Requests in a REST Service

One useful feature of our REST framework is the ability for a dispatch class to **identify request prefixes and forward them** to another dispatch class. This approach of modularizing your URL map will improve code readability, enable you to easily maintain separate versions of an interface, and provide a means to protect API calls that only certain users will be allowed to access.

Overview

To set up a REST Service on your Caché instance, you need to define a dedicated CSP application and create the associated **dispatch class** that handles incoming requests. The dispatch class extends %CSP.REST and will include an XData block that contains your URL map. This tells the system which method to call when a particular request is received.

For example:

```
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
  <Route Url="/orders" Method="GET" Call="GetOrders"/>
  <Route Url="/orders" Method="POST" Call="NewOrder"/>
</Routes>
}
```

The <Route> elements define the various requests that the service will handle. A GET request for the resource "/orders" will call into the classmethod "GetOrders". A POST request made for the same resource will instead call into the "NewOrder" method.

It's important to note that the CSP application name is not considered as part of the requested resource name in our URL map. Consider a request made to the address:

```
http://localhost:57772/csp/demo/orders
```

If our CSP application is called "/csp/demo", then the only segment of the request handled by the dispatch class is what comes after the application name. In this case, that is "/orders".

Request Forwarding

Rather than call a method inside the dispatch class, the other option for your URL map is to forward all requests matching a particular prefix to a different dispatch class.

This is done using the <Map> element in the UrlMap section. The element contains two attributes, **Prefix** and **Forward**. If the request URL matches one of the prefixes, we send the request to the specified dispatch class for further processing.

For example:

```
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
  <Map Prefix="/shipping" Forward="Demo.Service.Shipping"/>
  <Route Url="/orders" Method="GET" Call="GetOrders"/>
  <Route Url="/orders" Method="POST" Call="NewOrder"/>
</Routes>
}
```

A GET or POST request for "/orders" will be handled directly by this class. However, requests matching the prefix "/shipping" will be redirected to the dispatch class Demo.Service.Shipping, which has its own URL map:

```
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
  <Route Url="/track/:id" Method="GET" Call="TrackShipment"/>
</Routes>
}
```

URL Routing Breakdown

To demonstrate how each component of the requested URL factors into what method is finally called, we will break down a request for the following address:

```
http://localhost:57772/csp/demo/shipping/track/123
```

*/ **http://**	The protocol used for the request.
localhost:57772	The server that we connect to.
/csp/demo/shipping/track/123	The resource being requested.
/csp/demo	The CSP application name. A dispatch class is defined for the application, route the request there.
/shipping/track/123	The resource segment sent to the first dispatch class.
/shipping	The prefix that matches the <Map> element in the URL map. Redirect to the Demo.Service.Shipping class.
/track/123	The resource segment sent to the second dispatch class. Matches the route "/track/:id". Call the method TrackShipment(123).

Motivation

- ***Source control —*** Separating your REST API into multiple classes will reduce the overall size of each class, which in turn will help keep your source control history concise and readable.
- ***Versioning —*** A simple way to support multiple versions of an API simultaneously is to use forwarding. A single dispatch class could forward requests matching the /v1 or /v2 prefixes to a dispatch class implementing that version of the API. The REST API at the heart of Atelier, our new IDE, uses the same versioning scheme.
- ***Security —*** If your API needs to have routes that are restricted to certain users — for example, to allow only administrators to make certain types of requests — it makes sense to isolate these routes in their own class,

then forward requests using a particular prefix. If the second dispatch class defines an **OnPreDispatch** method, its code will be executed before processing each request. The service can use this to check a user's privileges and decide whether to continue processing or cancel their request.

#Best Practices #CSP #JSON #REST API #XML #Caché #InterSystems IRIS
70 3 0 1 2,460

Log in or sign up to continue
Add reply

Source URL: https://community.intersystems.com/post/forwarding-requests-rest-service