Article
Fabian Haupt · Sep 2, 2016 2m read

Advanced URL mapping for REST

By now it's a commonplace how to implement a basic REST API in Caché and there is good documentation about it here: REST in Caché

A question that comes up from time to time is:

How can I make a parameter in my REST url optional?

Simply put, is it possible to create a URL map in Caché that maps a URL like this:

```
http://server/API/object///old
```

While this might look odd, this is actually a valid URL. You can read the details in RFC3986 section 3.3.

The framework provided by %CSP.Rest actually allows us to create maps that match the above URL. In the above referenced documenation this is only mentioned in a side note explaining that the URL map is being translated into a regular expression.

To leverage this power, we need to understand regular expressions. There is a nice document by Michael exploring the basics of regular expressions within the context of Caché here. If you're not familiar with regular expressions I highly recommend taking a moment to read through it.

In %CSP.Rest the UrlMap block actually compiles the maps into regular expressions. For example, an URL map like Url="/class/:namespace/:classname" (taken from the Docserver example) is being compiled into:

```
/class/([^/]+)/([^/]+)
```

Recalling Michael's article you will see that this RegEx is matching a URL starting with /class/ and then throwing the next two pieces of the URL separated by / into two matching groups. Those will later be passed into the GetClass method as parameters.

We can pass in regular expressions into the Url parameter of the UrlMap in our class directly. This way we circumvent the limitations of the simplification. And it allows us to create advanced URL maps:

```
XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
{
<Routes>
<Route Url="//API/object/([^/]*)/([^/]*)/old" Method="GET" Call="GetClass"/>
</Routes>
}
```

In this case we just replaced the *1 or more* modifier + with *0 or more* *.

In combination with the regular expression intro you now have an idea of the power at your hands.

Please remember though:

```
with great power comes great responsibility
---(not Uncle Ben!)
```

Regular expressions are a very powerful tool, and as such regularly tempt people into implementing things that might really have been better off with other solutions.
Keep that in mind when creating your next regex masterpiece!

Happy coding!

#Mapping #REST API #Caché