Easy Caching in ColdFusion with MRECache

Posted At : September 16, 2020 9:00 AM | Posted By : Steve
Related Categories: com.sebtools

I avoided dealing seriously with caching for more of my programming career than I like to admit. Perhaps the old saw that "There are only two hard things in Computer Science: cache invalidation and naming things." (Phil Karlton) held me back.

Beyond that, though, I think I always had a sense that I didn't like the approaches that I found.

A few years ago, however, one of my clients started looking ahead to a major increase in traffic along with an increase in complexity. So, I needed to look into caching.

This caused me to think about what I wanted in a caching solution. I wanted something, first of all, where I could make a single call and get back a result without having to think about whether or not the result was cached when I made the call. I wanted to be able to tell my caching layer where I want to get the data from and it can decide whether to get me a cached version or get a live version first and cache it for later.

Next, I wanted something that would cache by time and/or changes to data. I really liked cachedwithin on cfquery, but I didn't like that there was no easy way to update all of the query caches for a table when I updated that table. So, I wanted a caching solution that allowed me the option of making data fresh when it changed.

Then I came up with a solution that really satisfied me (and allowed us to handle the traffic increase without any trouble). Mostly, it made the syntax simple and this addressed one of my primary complaints of other solutions that I've seen.

So, let's instantiate MRECache.cfc:

The "init" of MRECache.cfc has the following arguments (all optional):

- **id**: Basically a "scope" for the instance.
- **timespan**: Default timespan for how long data will stay cached.
- **idleTime**: Default timespan for how soon data should be uncached if it is idle (no requests having been made for that data in that timespan).
- **Observer**: My Observer service.

The "timeSpan" and "idleTime" arguments work just like the same attributes in the underlying ColdFusion caching on which MRECache is built.

If Observer is present, MRECache will call "MrECache:run" any time it retrieves uncached data. It will also call "MrECache:func" if it runs a function and "MrECache:method" if it runs a component method.

Here is an example of instantiated MRECache with an id of "myid" and a three hour default cache timespan:

```
<cfset Variables.MrECache = CreateObject("component","com.sebtools.MRECache").init(
   id="myid",
   timespan=CreateTimeSpan(0,3,0,0)
)>
```

When using MRECache, I mostly use the "meth" method (short for "method"). It also has a "method" method, which works the same except that you have to specify an "id" manually and a "func" method to call a UDF.

Let's say that I have a method called "getMyRecords" that I want to cache. Then I could rename it as "getMyRecords_Live" method and write a new "getMyRecords" method as:

```
<cffunction name="getMyRecords" access="public" returntype="query" output="no">
   <cfargument name="ExampleArg1" type="string" required="true">
   <cfargument name="ExampleArg2" type="numeric" required="false">

   <cfreturn Variables.MrECache.meth(
      Component=This,
      MethodName="getMyRecords_Live",
      Args=Arguments
   )>
</cffunction>
```

That method would return the results of "getMyRecords_Live" for those arguments, using cached data instead of calling the method, if available. This will use the timeSpan and idleTime of the instance of MrECache, but you can also pass those arguments in to the method to use different values.

I can also clear my caches with pretty good granularity. If I want to clear all caches for a given instance of "MrECache", I call:

```
Variables.MrECache.clearCaches()
```

Note that if two different instances use the same "id" argument in instantiation, then both will be cleared. The same cache names on instances with different "id" arguments in instantiation, however, will have different cache scopes.

If I want to clear for the "getMyRecords_Live" method, I call:

```
Variables.MrECache.clearCaches("getMyRecords_Live")
```

This would clear the cache for all calls to "getMyRecords_Live", regardless of what arguments were passed in.

I could even clear caches for all methods that start with "get" in this instance:

```
Variables.MrECache.clearCaches("get")
```

One extra bonus that I worked in to MRECache is that the intervals for timeSpan and idleTime arguments are very flexible. You can pass in a numeric TimeSpan value and it will work just like the same attributes in ColdFusion's built in caching functionality.

You also have the option of passing in English language intervals. Here are some examples:

- every other minute
- four hours
- daily
- weekly
- hourly
- monthly
- every twelve seconds
- every two minutes twelve seconds

So, from the previous example, if you wanted to cache the result of a call for two weeks, you could do the following:

```
<cfreturn Variables.MrECache.meth(
   Component=This,
   MethodName="getMyRecords_Live",
   Args=Arguments,
```

```
   interval="two weeks"
)>
```

You could still clear out the cache any time you wanted as mentioned above, of course.

That's the basics of MRECache. It has some other handy utility methods that I'll cover later, but this at least covers the basic of using it.

Enjoy!

MRECache is part of the **com.sebtools package**. It is a free ColdFusion package and you can read more about it in my **com.sebtools blog series**. Check it out!

If you need any help with com.sebtools, or ColdFusion in general, **let me know**.