Optimizing your CFML

Posted At : November 4, 2004 11:49 AM | Posted By : Mark Kruger
Related Categories: Coldfusion Optimization

There are many things you can do to optimize a web site before you ever examine the CFML. In fact, many poorly written sites run just fine because on big beefy servers and robust database servers. Sometimes, when troubleshooting a site's performance you start with those resources - is the web server too busy? Is the database indexed and optimized? Is there enough RAM? Is the CF server correctly configured? ... and so on. Once you have Exhausted these possibilities it may be time to look at the actual CF code itself. That's when your advanced knowledge and experience can begin to really pay off.

Where do I start?

I like to start with the framework. How is the site using Application and session variables. Are there locking issues? Is the code written appropriately for these scopes. For example, one site I was working with had the following code:

```
<Cfset application.var1 = 'blah'>
<Cfset application.var2 = 'blah'>
<Cfset application.var2 = 'blah'>
```

in the Application.cfm file. There was no *CFIF* clause surrounding it. The result of this code is to overwrite the application scope with every single request. That not only sets the site up for lock-ups due to race conditions, but it pretty much destroys any advantage of having variables in the application scope at all. You might as well use the variables scope. I change the code to:

```
<cfif NOT IsDefined('Application.var1') or IsDefined('url.refreshAp')>
<Cfset application.var1 = 'blah'>
<Cfset application.var2 = 'blah'>
<Cfset application.var2 = 'blah'>
</cfif>
```

This code will only run if there is no defined Application scope (like when the server is restarted) *or if* there is a URL variable called "refreshAp". This is useful for the times when I actually *do* want to refresh the application variables. I simply add "refreshAp" as a url var to any page inside the ap and the variables are refreshed.

Scope your variables

Because the order of precidence dictates that the *Variables* scope is always checked first, it is the only scope I ever allow to be defaulted. Every other scope must be carefully and accurately scoped (url, form, request, application, session etc.).

Slow Pages

Turn on the slow page logging on the CF server and examine pages taking longer than *n* seconds to load. Then, take each of these pages and examine the code for weaknesses or areas where they can be optimized.

Use the Debug Information

It's simply impossible to glean anything useful from browsing pages on your site without looking at the debug. The debug can show you:

- Query execution time - this is an awesome clue in optimization, don't ignore it (more on that tomorrow).
- Individual include page execution time - You can see how long individual CFC's and include pages take to run. NOTE: cf module and custom tags are not reported in this area. In the CFMX world, they should be converted to CFC's anyway.
- Lists of variables - This information is useful in debugging and scoping.

Query optimization

Frankly, optimizing your queries is the place where the most significant gains can always be made. The truth is that a great many very good developers are weak in the area of SQL and Database management and coding. They can write great CFML code, but they write lousy query code. They often fail to use data- binding, they often use the asterisk indiscriminently, and they often pull in far more columns and rows thane they need. If you are an intermedite web programmer and you want to be a guru, take time to become at least an intermediate SQL programmer before you go any further. You won't be sorry. In my next blog I will tackle some of the techniques used to optimize query code.