Application Variables Part II - Thread Safe Gotcha

Posted At : August 22, 2005 11:53 AM | Posted By : Mark Kruger
Related Categories: Coldfusion Tips and Techniques, Coldfusion Troubleshooting

Here's a tip on application variables from the inestimably knowledgeable **Sean Corfield**. In my previous post on **Application Variables** I used a pretty typical example of how they are initialized. I checked for the existence of a particular application variable. If it did not exist, I would lock and set all the variables. Sean pointed out that this approach is *still subject* to threading issues. To start with, here's the original example.

```
<cfif NOT isDefined('Application.dsn')OR IsDefined('url.forceAp')>
<cflock scope="APPLICATION" timeout="1">
<Cfset Application.dsn = 'myDatasource'>
<cfset Application.imgPath = expandPath('./images')>
</cflock>
</cfif>
```

Here's what could happen. Thread "A" and "B" hit the CFIF condition at the same time and both attempt the lock. One thread succeeds (let's say thread "A") and the other one waits for the lock to be free. As soon as it is free - meaning *after* thread "A" has already initialized the variable - thread "B" grabs the lock and initializes the variables again.

As Sean pointed out, this is irrelevant in my current example because I'm setting atomic (don't you love OO speak) variables that are static and it doesn't really matter if thread "A" overwrites thread "B". So let's use a better example. Let's say that you hold a table in memory that contains the IP addresses of the current users for security sake. Here's the code.

```
<cfif NOT isDefined('Application.curIps')OR IsDefined('url.forceAp')>
<cflock scope="APPLICATION" timeout="1">
<cfscript>
      curIps = queryNew("IP");
      queryAddRow(curIps);
      querySetCell(curIps,"IP",cgi.remote_addr);
      Application.curIps = curIps;
</cfscript>

</cflock>
</cfif>
```

Other code (presumably) checks the IP address and takes appropriate action. If thread "A" seizes the lock and writes it's IP into the query, but thread "B" comes along and overwrites it, you have the potential for the User of thread "A" to run amuck on your site without having his or her IP logged in the query. To avoid this, Sean suggested this:

```
<!--- check condition --->
<cfif NOT isDefined('Application.curIps') OR IsDefined('url.forceAp')>
   <!--- Use a named lock --->
<cflock name="ipInitialize" timeout="1" type="exclusive">
      <!--- check to make sure condition is still true --->
      <cfif NOT IsDefined('Application.curIps') OR IsDefined('url.forceAp')>
```

```
        <cfscript>
             curIps = queryNew("IP");
            queryAddRow(curIps);
            querySetCell(curIps,"IP",cgi.remote_addr);
            Application.curIps = curIps;
        </cfscript>
    </cfif>
</cflock>
</cfif>
```

In this case thread "A" will initialize the query, but Thread "B" will not overwrite it - because thread "B" will *check again* just to be sure that the variable doesn't exist.

A couple of notes. The code above is theoretical. Obviously other factors could make the overwrite issue a non-factor. It's only there to illustrate how merely checking and locking does not make your code thread safe. Secondly, I'm not at all sure if this code applies to the **onApplicationStart( )** method of an Application.cfc file. I suspect that the framework model sorts out the threads for you so that the onApplicationStart() method only runs once for 1 thread. If you have any insight on this - feel free to comment.