Components and CFTHREAD

Posted At : June 13, 2008 7:00 AM | Posted By : Steve
Related Categories: ColdFusion

If you are already using CFCs and you know about CFTHREAD then there is nothing for to see here, move along.

I was in a conversation with another developer recently about the advantages of CFCs. I felt that I did a poor job explaining the advantages.

Yesterday, I used CFTHREAD for the first time and in so doing ran across a great example of one of the advantages of CFCs.

One of my clients has online training that is that is assigned by rather complicated logic. Assignments can be changed based on several different types of data. If any one of them is changed then the relevant assignments must be recalculated. Very rarely, depending on the data that is changed, the recalculation may be very time-consuming.

This is sufficiently rare that it doesn't pose any problem for site performance, but it can cause the user (one of a very small number of high-level administrators) to wait for up to several minutes. This is obviously no good for usability.

I'm not sure if the assignments logic can be greatly optimized, but it wouldn't be worth much effort if I could just solve the wait for the user.

Fortunately, CFTHREAD solves just this problem.

My before code:

```
<cffunction name="saveAssignments" access="public" returntype="void" output="no" hint="I save lesson assignments.">
<!--- several arguments --->

<!--- complicate assignment logic --->

</cffunction>
```

My after code:

```
<cffunction name="saveAssignments" access="public" returntype="void" output="no" hint="I save lesson assignments.">
<!--- several arguments --->

<cfthread name="saveAssignments">
<cfset saveAssignments_Internal(argumentCollection=arguments)>
</cfthread>

</cffunction>

<cffunction name="saveAssignments_Internal" access="public" returntype="void" output="no" hint="I save lesson
assignments.">
<!--- several arguments --->

<!--- complicate assignment logic --->

</cffunction>
```

CFTHREAD causes the code within the CFTHREAD block to be run in a separate thread so that the user doesn't have to wait on the code to execute. This would cause problems if I needed to return data from the function call, but I don't.

The advantage the CFCs provide here is that I know that the saveAssignments method of my component is called from several places throughout the application (including other places in the same component), but I don't need to know where they are in order for my change to be effective.

I didn't need to create the saveAssignments_Internal method in order to accomplish this, but for readability I didn't want CFTHREAD wrapped around a long block of code.

These are the kinds of changes that are easy to do if your code is structured properly. If you can't make this kind of change in your code easily, consider using components to organize your code.