Configuring CF 7 Multiserver Instances - Including Custom JVM Settings

Posted At : April 17, 2006 10:31 PM | Posted By : Mark Kruger
Related Categories: Coldfusion MX 7

Multi-server Installation have some distinct advantages. Process isolation is the one that makes the most sense to me. Many, maybe most servers house more than 1 web site. Even Intranet servers seem to have more than 1 site configured. A "single site" server is pretty rare - at least outside of a cluster. Having a way to isolate 1 site from another on the server itself is an ideal way of keeping bad code or database connections on one site from dragging down another. It's not for everyone. It certainly adds another layer of complexity to your server. Still, it's an excellent solution in some cases. Consider this scenario.

You have a server with ten Coldfusion sites. Nine of the sites are lightly used and seem to experience no difficulty. The tenth site, however, is a pain in the Tukas. The traffic generates copious database calls and often drags all ten sites down. The only solution *seems* to be to restart Coldfusion as the threads start queuing up. What can you do about it? Create 2 instances of JRUN/Coldfusion. That in itself is actually pretty easy. Make sure that when you installed the Coldfusion server you chose "multiserver" as the type of install. If you did then you are ready for the following.

In the Coldfusion Administrator go to the "Enterprise Manager" and click on the "Instance Manager". Click "add new instance". Type in "badsite" and Coldfusion will copy files and configure stuff to create your new instance. Make sure and check the "Create Windows Service" option. Once Coldfusion is done you will see a new instance, "badsite" in the list. You can stop or start the service. It even has it's own admin where you can configure data sources and other settings -very cool. Use the "web server configuration tool" (in the macromedia program group) to map an instance to a web site. This process (of mapping site to instance) does little more than add a "wildcard mapping" to the site that points to JRun4\lib\wsconfig\1\jrun_iis6_wildcard.dll - where the "1" is the number of the instance. Your site *should* be ready to serve CFM files using your new instance.

If you take a look at your process viewer you might see something alarming depending on what your JVM.config file looks like. If, when you were serving all nine sites from the same instance, you configured your JVM to use 1.3 gigs of memory you will see your "jrun" instances are gloaming onto to 1.3 gigs virtual memory *each* (check the VM column). Unless you have 8 gigs on your server that could be a problem. In fact, when I was experimenting I added several instances and I began to see an error that said:

```
Error occurred during initialization of VM

Could not reserve enough space for object heap
```

The JVM needed to allocate appropriate space using both physical and virtual memory. I increased the size of the page file and the error went away, but that didn't solve my problem. Obviously I want to keep Jrun running within *physical memory* as much as possible.

The Problem

It seems that when creating a new instance Jrun was using the settings from my JVM.config file for each instance. The total memory usage would equal the allocation in the JVM times the number of instances. If I allocated a 512 Megs heap and 128 Megs of MaxPermSize, and I had 3 instances I would end up with 640 Megs times 3 or 1.9 gigs. On a 2 gig server that's too much. To fix this easily I just scale back my settings in my JVM. In the scenario above I could scale back to 768 megs times 2 instances for 1.8 gigs. That is marginal, but probably doable. But there's a catch. There is an "admin" server that runs the Coldfusion administrator in the multi-server install. That's why you always see 1 more Jrun than you anticipated. I have to add that in. So I might end up with 512 megs X 3 instances for that 1.9 gig number.

But that is not optimal. Why should the little used "admin" server get a third of the resources? In my above scenario I might want to devote a full half or two thirds of memory to the heavily trafficked site. I will rarely want an even distribution. What I really need is to be able to install each instance with it's own settings - right?

The solution

The solution is involved, but it's doable.

1. Create a JVM.config file for each of your instances. I used "jvm.config_small", "jvm.config_medium" and "jvm.config_large"; where small, medium and large represent how I was allocating the heap site.
2. Open your services control panel and bring up the list of "Macromedia ...." services - these are your instances.
3. From your list of instances in the services control panel, choose a service to configure for custom JVM. Let's say the name of your "badsite" instance is "Macromedia JRUN AS badsite". Open a command line and "CD" to the /Jrun4/bin directory. Type the following to remove the service. This does not remove any files - just "unregisters" the service.

   ```
   c:\Jrun4\bin>jrunsvc -remove "Macromedia JRUN AS badsite"
   ```

4. Now "reregister" the same instance with the following command:

   ```
   C:\JRun4\bin>jrunsvc -install badsite "Macromedia JRUN AS badsite" "Macromedia JRUN AS
   badsite" -config jvm.config_large
   ```

5. Hit F5 on the services control panel and you should see your service. Try to start it. If you get an error go look in the jrun4/logs directory for the badsite-out.log and the badsite-err.log.

You will want to do the steps above *after* adding instances in the CF Admin but *before* you use the web server configuration tool. Remember, I had already set services up using the CF Admin so all the files are in place. Doing it this way requires me to un-register and register the services, but I found that this is actually less painful than copying files and editing XML config files - or using the Jrun Console. It is possible to use the Jrun Console to create server instances, but then you have to copy in appropriate Coldfusion ear or war files to finish it off.

This is *how I discovered* to do it. As always I welcome input from folks who have a better mousetrap. Fire away and I'll give credit where credit is due. In fact, as I've been working through this (and I continue to learn all the ins and outs of configuring multiserver installs) Robi Sen has been a great help.