# Local Development Setup Pt 2: Multiple ColdFusion Instances

Posted At : July 23, 2007 10:01 AM | Posted By : Cutter
Related Categories: Apache, ColdFusion 8, Development, ColdFusion

Our last post discussed installing **Apache** and **ColdFusion**, as well as configuring your default instance for Apache access. Now it's time to create additional ColdFusion instances.

By default, ColdFusion (or, more appropriately, JRun) is only configured to utilize 512MB of RAM per instance, and is only capable of accessing 1024MB. This is due to a limitation of 32bit JVMs, and will someday be formerly addressed by Adobe. But that doesn't mean that you are necessarily restricted to only using 1GB of RAM for ColdFusion. You may define multiple instances of the server, each of which will address it's own memory space, it's own instance of the JVM, and it's own instance of ColdFusion (and JRun). Not only does this allow you to utilize more of the memory available to you today, in our high powered systems, but it will also sandbox applications that are separated into their own instances.

For instance: Let's say you have a dynamic template application. One that reads the requested URL and supplies customized content dependent upon the site identified. Any number of sites could be configured in a database, rendered by the same code, off of a single instance of ColdFusion (or a clustered set of instances, maybe). You could have a 'sites' instance of ColdFusion that served this content. Now, the same set of sites might require a backend administrator, or content management system, for the configuration of those sites. You might set this up on a single domain name, with users logging in to their specific set of tools and data. It would be it's own application, with dynamic options and data according to the user logging in. This might be placed in another 'control' instance of ColdFusion.

Setting up additional instances of ColdFusion is easy, but requires a small bit of manual effort when working with Apache. First of all, the connectors for JRun and Apache are not completed automatic, so you need to setup a few folders on the file system. Find the root directory for JRun. The default location is *C:\JRun4*. You are creating folder for the connectors, which will be located in the *{JRun Root}\lib\wsconfig*. Notice that there is already a subdirectory titled *1*. This is the connector for your default ColdFusion instance. You'll want to create an empty subdirectory for each instance you setup, named exactly as you will name your instances. According to the above example, you want to create a 'sites' directory, and a 'control' directory.

Your next step requires logging into the ColdFusion Administrator of your default ColdFusion instance. In our last post we setup a url for accessing this, *http://username.companyname.loc/CFIDE/Administrator*. Once you've logged in, in the default instance (and only the default instance) you will see an option at the bottom of the menu for **Enterprise Manager**. You'll want to select this, and it's sub-item, **Instance Manager**. Here you will see a *samples* instance that is already defined, though disabled. This is the instance for running the sample applications that ship with ColdFusion. To create a new instance you simply select **Add New Instance**. This will bring up the new instance dialog. In the server name type 'sites' (exactly as you named the folder, including case) and select the **Create Windows Service** option, then hit **Submit**. That's it! ColdFusion automatically goes through a four step process to create your new instance, giving you status updates along the way. Once it's complete, go back to the **Instance Manager** and do the same thing for your 'control' instance.

OK, so you have new instances, but Apache still can't talk to them yet. We need to do a little more work on the Apache config before we can really start to play. The first thing you'll need to do here is locate the *JRun Settings* block in your httpd.conf. It'll look very similar to this:

```
# JRun Settings
LoadModule jrun_module "C:/JRun4/lib/wsconfig/1/mod_jrun20.so"
<IfModule mod_jrun20.c>
 JRunConfig Verbose false
 JRunConfig Apialloc false
 JRunConfig Ssl false
 JRunConfig Ignoresuffixmap false
 JRunConfig Serverstore "C:/JRun4/lib/wsconfig/1/jrunserver.store"
 JRunConfig Bootstrap 127.0.0.1:51000
 #JRunConfig Errorurl <optionally redirect to this URL on errors>
 #JRunConfig ProxyRetryInterval <number of seconds to wait before trying to reconnect to unreachable clustered server>
 #JRunConfig ConnectTimeout 15
 #JRunConfig RecvTimeout 300
 #JRunConfig SendTimeout 15
 AddHandler jrun-handler .jsp .jws .cfm .cfml .cfc .cfr .cfswf
</IfModule>
```

Alright, some major points to notice here. Two big lines to look at for multiserver configuration stuff, the *Serverstore* and the *Bootstrap*. These will be different for each instance of ColdFusion. You probably already recognize most of the path in the *Serverstore* value. The 'control' and 'sites' instance folders that you had created will replace the **1** in your new definitions. The *Bootstrap* value comes from each instance's *port* setting in it's *JRunProxyService*. To get this value, go to that instance's *jrun.xml* file, located at *C:\JRun4\servers**[instance name]***\SERVER-INF\jrun.xml*. Open this file and find the following *service* definition block:

```
<service class="jrun.servlet.jrpp.JRunProxyService" name="ProxyService">
 <attribute name="activeHandlerThreads">25</attribute>
 <attribute name="backlog">500</attribute>
 <attribute name="deactivated">false</attribute>
 <attribute name="interface">*</attribute>
 <attribute name="maxHandlerThreads">1000</attribute>
 <attribute name="minHandlerThreads">1</attribute>
 <attribute name="port">51002</attribute>
 ....
```

Two things you need here. First, make sure that the *deactivated* attribute is set to **false**. Next, write down the *port* value. So, if you are in the *jrun.xml* of your 'control' instance, and the *port* is '51020', then write that down (control: 51020) and do the same for your 'sites' instance. Also remember that you will need to restart these instances after changing the *deactivated* attribute.

Next, let's break out the default ColdFusion instance specific information and place it inside it's own include config file. In your Apache *conf* directory, create a new file - *cf_defaultinstance.conf*. In this file we'll place those settings we want for our default instance:

```
<IfModule mod_jrun20.c>
 JRunConfig Verbose false
 JRunConfig Ignoresuffixmap false
 JRunConfig Serverstore "C:/JRun4/lib/wsconfig/1/jrunserver.store"
 JRunConfig Bootstrap 127.0.0.1:51000
</IfModule>
```

With these settings now within their own include, we can now remove them from the *httpd.conf* file:

```
# JRun Settings
LoadModule jrun_module "C:/JRun4/lib/wsconfig/1/mod_jrun20.so"
<IfModule mod_jrun20.c>
 JRunConfig Verbose false
 JRunConfig Apialloc false
 JRunConfig Ssl false
 JRunConfig Ignoresuffixmap false
 #JRunConfig Serverstore "C:/JRun4/lib/wsconfig/1/jrunserver.store"
 #JRunConfig Bootstrap 127.0.0.1:51000
 #JRunConfig Errorurl <optionally redirect to this URL on errors>
 #JRunConfig ProxyRetryInterval <number of seconds to wait before trying to reconnect to unreachable clustered server>
 #JRunConfig ConnectTimeout 15
 #JRunConfig RecvTimeout 300
 #JRunConfig SendTimeout 15
 AddHandler jrun-handler .jsp .jws .cfm .cfml .cfc .cfr .cfswf
</IfModule>
```

Notice that I just commented them out. You can remove the entirely if you like, but I'm gonna leave it. Next I'm going to adjust my *VirtualHost* for my default instance administrator access:

```
#
# Use name-based virtual hosting.
#
NameVirtualHost 127.0.0.1:80

 ....

<VirtualHost 127.0.0.1:80>
 ServerAdmin username@companyname.com
 # Root folder for my scratchpad stuff
 DocumentRoot "C:\Documents and Settings\username\My Documents\wwwroot"
 ServerName username.companyname.loc
 # Alias for /CFIDE, which the CF install placed in my Apache webroot.
 # This is solely for our dev environment, and would not be a good practice
 # within a production environment
 Alias /CFIDE "C:/Program Files/Apache Group/Apache2/htdocs/CFIDE"
 <Directory "C:/Program Files/Apache Group/Apache2/htdocs/CFIDE">
  AllowOverride All
  Order allow,deny
  Allow from all
 </Directory>
 ErrorLog logs/username.companyname.loc-error.log
 CustomLog logs/username.companyname.loc-access.log common
 # SGB [072007]: Add include for default ColdFusion instance
```

```
 Include conf/cf_defaultinstance.conf
</VirtualHost>
```

Now *username.companyname.loc* is setup to use the default ColdFusion instance. Next, setup an include for your 'control' instance. In Apache's *conf* directory, create another config file - *cf_controlinstance.conf*. Remember those *port* numbers you wrote down from the *jrun.xml* files? It's in the *Bootstrap*:

```
<IfModule mod_jrun20.c>
 JRunConfig Verbose false
 JRunConfig Ignoresuffixmap false
 JRunConfig Serverstore "C:/JRun4/lib/wsconfig/control/jrunserver.store"
 JRunConfig Bootstrap 127.0.0.1:51020
</IfModule>
```

Then you could define a special domain for accessing the 'control' instance's ColdFusion Administrator, by adding another *VirtualHost* directive to the Apache config:

```
<VirtualHost 127.0.0.1:80>
 ServerAdmin username@companyname.com
 # Root folder for a 'control' instance
 DocumentRoot "C:\Documents and Settings\username\My Documents\wwwroot\admin"
 ServerName control.companyname.loc
 # Alias for /CFIDE, each CF instance has it's own CFIDE.
 # This is solely for our dev environment, and would not be a good practice
 # within a production environment
 Alias /CFIDE "C:/JRun4/servers/control/cfusion.ear/cfusion.war/CFIDE"
 <Directory "C:/JRun4/servers/control/cfusion.ear/cfusion.war/CFIDE">
  AllowOverride All
  Order allow,deny
  Allow from all
 </Directory>
 ErrorLog logs/control.companyname.loc-error.log
 CustomLog logs/control.companyname.loc-access.log common
 # SGB [072007]: Add include for the 'control' ColdFusion instance
 Include conf/cf_controlinstance.conf
</VirtualHost>
```

Notice the different path for the CFIDE folder. Each created instance will have a unique CFIDE. Also notice that I changed the *DocumentRoot* path, to reflect the root of the application I'll use with the instance. Now that you've setup your 'control' instance, config, and VirtualHost, you can do the same thing for your 'sites' instance. Just watch your *port* value, *Serverstore* path, and *CFIDE* and *DocumentRoot* paths.