Using the CFC Proxy in a ColdFusion Cluster

Posted At : February 6, 2008 12:59 PM | Posted By : Steven Erat
Related Categories: Flex, Java, Adobe, ColdFusion, Macromedia

The CFC Proxy API was introduced as a supported feature in CFMX 7.01. It allows you to call ColdFusion Components (CFCs) from Java classes such as a standalone servlet running in the same JVM. In order for this to work, the Java class must be loaded by the ColdFusion classloader rather than a higher level classloader in the J2EE container. To load a Java class with the ColdFusion classloader, the class's jar file must be specified in ColdFusion's web.xml under the cf.class.path parameter. To avoid managing multiple copies of a custom jar file between ColdFusion instances clustered on JRun, you can put a single copy of the custom jar file under a central location outside the JRun root directory. Then modify the web.xml for each CF instance to point to that jar file in the cf.class.path entry. Surprisingly, there is no documentation on using CFCProxy on livedocs.adobe.com, but instead you can find this reference on Ben Forta's website. The reference describes the API and provides a brief example implementation. A few details are left out such as how to compile the custom Java class, so I'll provide a quick walk through of how I set all this up...

A Sample CFC saved as C:\TEMP\CFCProxy\Test.cfc

```
<cfcomponent>
  <cffunction name="getData" returntype="struct">
    <cfargument name="msg" required="Yes">
     <cfscript>
       st = StructNew();
       st.name = "michael";
       st.email = "remin@macromedia.com";
       st.date = "#now()#";
       st.message = msg;
       return st;
     </cfscript>
  </cffunction>
 </cfcomponent>
```

CFCInvoker Java source file saved as C:\TEMP\CFCProxy\CFCInvoker.java

```
import coldfusion.cfc.CFCProxy;
import coldfusion.bootstrap.BootstrapClassLoader;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpServletRequest;
   public class CFCInvoker
   {
     public coldfusion.runtime.Struct invoke()
     {
       coldfusion.runtime.Struct map = null;
       try {
         /*
          declare variable for storing reference to CFCProxy
         */
         CFCProxy myCFC;
         /*
          instantiate CFC
         */
         myCFC = new CFCProxy("C:\TEMP\CFCProxy\test.cfc");
         /*
          Build arguments array
         */
         Object<] myArgs = {"Java invocation working"};
         /*
          invoke method from CFC. It returns a structure.
         */
         map = (coldfusion.runtime.Struct) myCFC.invoke("getData", myArgs);
                } catch (Throwable ex) {
       ex.printStackTrace();       }
           return map;
   }
}
```

Batch File Compiler saved as C:\TEMP\CFCProxy\Compile_and_Jar_CFCInvoker.bat. In order to import all the required classes such as BootstrapClassLoader, HttpServletRequest, and CFCProxy the cfusion.jar, cfmx_bootstrap.jar, and jrun.jar need to be pulled into the classpath. Adjust the paths to match your system (or write a shell script analog) and double click to run.

```
@echo off
echo This will compile and jar up the CFCInvoker Java source file
echo Press Control+C to abort.
pause
SETLOCAL
set JRUN_HOME=C:\JRun4
set CF_HOME=%JRUN_HOME%\servers\cfusion\cfusion-ear\cfusion-war
set JAVA_HOME="C:\Program Files\Java\jdk1.6.0"
set CLASSPATH=.;%CF_HOME%\WEB-INF\cfusion\lib\cfusion.jar;%CF_HOME%\WEB-INF\lib\cfmx_bootstrap.jar;%JRUN_HOME%\lib\jrun.jar;
set PATH=%JAVA_HOME;.;%PATH%
javac -classpath %CLASSPATH% CFCInvoker.java
jar -cvf CFCInvoker.jar CFCInvoker.class
echo Done
ENDLOCAL
pause
```

Modify cf.class.path parameter for each ColdFusion instance's web.xml file. Add the path for the centrally located jar file:

```
<context-param id="coldfusion_context_88">
        <param-name>cf.class.path</param-name>
        <param-value>
            ./WEB-INF/cfusion/lib/updates,./WEB-INF/cfusion/lib,./WEB-INF/cfusion/gateway/lib,./WEB-INF/flex/jars,./WEB-INF/cfform/jars,C:/TEMP/CFCProxy/CFCInvoker.jar</param-value>
</context-param>
```

Restart All ColdFusion Instances

Test Calling The Java Class by putting a CFCInvoker_tester.cfm in the webroot.

```
<cfscript>
    CFCInvokerObj = createObject("java","CFCInvoker");
</cfscript>
<cfdump var="#CFCInvokerObj.invoke()#"/>
```

In this example, the CFCInvoker_tester.cfm just creates an instance of the Java class and calls invoke() on it, but a real life example might be that the Java class is a Servlet and is available via URL pattern to accept requests directly.

Since the custom CFCProxy Java class is centrally located in a single jar file outside the ColdFusion or JRun roots it can be shared across multiple instances without worrying about revision differences across the cluster members.