Secret Powers of Session Handling in CFML

Charlie Arehart, CTO
New Atlanta Communications
charlie@newatlanta.com

Last Updated: Aug 2004

Overview

- Introduction
  - Changes for CF4/5 versus CFMX (and BlueDragon)
  - Available new "J2EE Sessions" feature
- Common Challenges for Session Handling
  - Discussion of each challenge
  - Solutions for CF4/5, and CFMX and BlueDragon
- Leveraging Extended Features of J2EE and .NET
  - Enhancements for CFML on J2EE
  - Enhancements for CFML on .NET

About Your Speaker

- CTO of New Atlanta Communications since April '03
  - Company based in Alpharetta, GA (30 miles north of Atlanta)
- 7 yrs CF experience (21 yrs Enterprise IT)
- Co-author of ColdFusion MX Bible
- Frequent contributor to ColdFusion Dev Journal
- Past accomplishments of note
  - Tech Editor, CFDJ
  - Team Macromedia Member
  - Allaire/Macromedia Certified Instructor
  - Allaire/Macromedia Certified Adv CF Developer (4, 5, MX)
  - Macromedia Customer Advisory Board Member
  - Contributor to Macromedia Devnet , Dev Exchange
- Frequent speaker to user groups, conferences worldwide
- Also pursuing Masters at Dallas Theological Seminary
  - part-time via Atlanta extension campus

Understanding the Different CFML Engines

New Atlanta COMMUNICATIONS

- CF4 and 5
  - Run on underlying C++ engine created by Allaire/Macromedia
  - Many developers still using this edition
    - Will show some solutions suitable for them

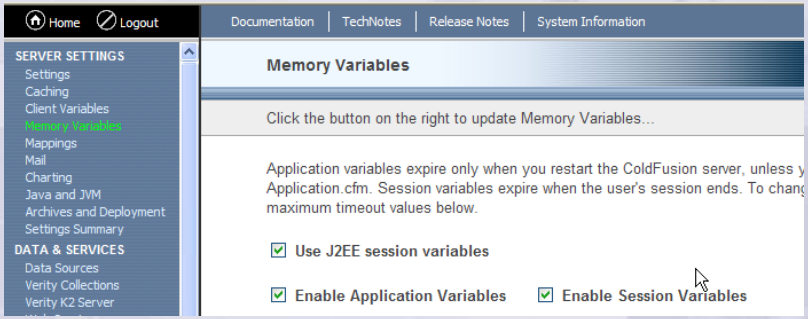
- CFMX 6.1
  - Runs on underlying J2EE engine created by Macromedia (JRun)
    - New architecture opens some new doors for session mgt
    - Optionally can be deployed on other J2EE servers
  - Most CFML developers have moved to CFMX
    - But may not know about some hidden features
    - Features discussed for CF4 and 5 still work in CFMX

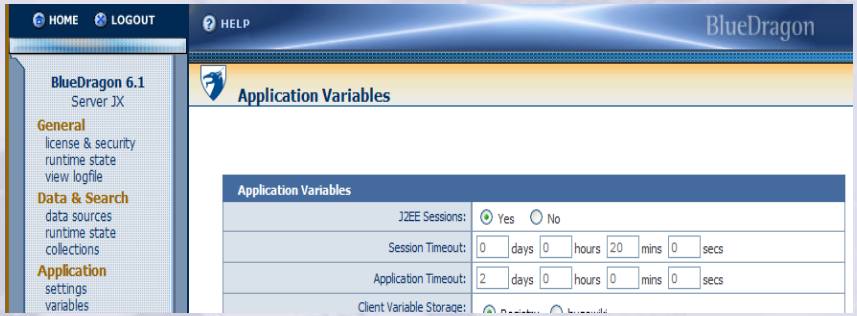
Understanding the Different CFML Engines (cont.)

- BlueDragon 6.1
  - An alternative CFML engine, can be used in place of CF
    - Shares functionality and architecture of CFMX 6.1
    - CF 4/5/MX applications should run without change
    - Offers many advantages not available in ColdFusion
  - Runs on underlying J2EE engine created by New Atlanta (ServletExec)
    - Optionally can be deployed on other J2EE servers
    - Also can optionally be deployed on .NET framework
  - More at www.newatlanta.com/bluedragon/

J2EE Sessions

- "J2EE Sessions" is an optional new feature
  - Enabled in both CFMX and BlueDragon 6.1





- Causes CFML engine to give up control of session handling to the underlying J2EE server
  - CFMX gives control to Jrun on standalone CFMX
  - BlueDragon gives control to ServletExec on Server editions of BlueDragon
  - CFMX or BlueDragon/J2EE give control to whatever J2EE server you may use
- You still use session variables the same way as before
  - But they're managed by the J2EE server instead of CF/BD engine
- In both engines, must restart after changing to take effect

J2EE Sessions (Cont.)

- One noticeable change when "J2EE Sessions" used
  - Uses new JSessionID cookie to associate a user to their session

CFID	33
CFTOKEN	98688059
SESSIONID	REGRESSION_33_98688059
URLTOKEN	CFID=33&CFTOKEN=98688059

struct	
sessionid	cc302244641088544899117
urltoken	CFID=33&CFTOKEN=98688059&jsessionid=cc302244641088544899117

  - This JSessionID cookie is stored in browser memory only
    - Often referred to as a "session" cookie, in that it is not persisted across browser restart
  - Different from CFID/CFTOKEN that are stored to disk by browser and have very long life
    - Primarily in support of client variables, which have long life
- Resources
  - "How to enable J2EE session management in ColdFusion MX"
    - http://www.macromedia.com/support/coldfusion/ts/documents/tn18232.htm
  - "New Possibilities for Session/Client Variable Handling in CFMX"
    - http://www.sys-con.com/story/?storyid=41646&de=1

Session Challenges

- Common Challenges for Session Handling
  - Terminate Session on Browser Close
  - Insecure SessionIDs
  - Unexpected Session Timeouts
  - Handling Sessions When Cookies Are Not Enabled
  - Terminate Session at Will
  - Locking Session Variable Access

Terminate Session on Browser Close

- Challenge: Terminate Session on Browser Close
  - User A creates session, closes browser and leaves
    - User B opens browser, still has access to user A's session
  - Cause: cookie used to track sessions is persistent
  - Solution: cause browser cookie to be non-persistent
- Solution in CF4 and Above
  - Using CFML on next slide, change CFID/CFTOKEN cookies to be non-persistent, stored as memory-only/"session" cookie on browser
  - Note: doing this precludes use of "client" variables in CFML
- Solution in CFMX/BlueDragon
  - Can either use solution above, or use "J2EE Sessions" instead
  - Enable "J2EE Sessions" in Admin Console
    - JSessionId automatically created as a memory-only cookie
- Note:
  - To experience this benefit, user must close all browser windows/instances that share a given SessionID

Terminate Session on Browser Close (Cont.)

- Solution in CF4 and Above
  - First, need to delete previously existing CFID and CFTOKEN cookies:

```
<CFCOOKIE NAME="CFID" VALUE="#CFID#" EXPIRES="NOW">
<CFCOOKIE NAME="CFTOKEN" VALUE="#CFTOKEN#" EXPIRES="NOW">
```

  - Then, in Application.cfm, set these to per-session cookies:

```
<CFAPPLICATION NAME="myCFApp" SESSIONMANAGEMENT="YES"
    SETCLIENTCOOKIES="NO">
 <CFIF not IsDefined("Cookie.CFID")>
    <CFLOCK SCOPE="SESSION" TYPE="READONLY" TIMEOUT="5">
        <CFCOOKIE NAME="CFID" VALUE="#SESSION.CFID#">
        <CFCOOKIE NAME="CFTOKEN" VALUE="#SESSION.CFTOKEN#">
    </CFLOCK>
</CFIF>
```

  - Resource: "How to write CFID and CFTOKEN as per-session cookies"
    - http://www.macromedia.com/support/coldfusion/ts/documents/tn17915.htm

Insecure SessionIDs

- Challenge: Insecure SessionIDs
  - CFID/CFTOKEN values used are small, simple numbers
    - As few as one digit for CFID, CFTOKEN is 8 digits
  - Number can be easily used (attacked) to gain access to session belonging to someone else on server
- Solution in CFMX
  - New option in Admin console, "Settings" page: "Use UUID for cftoken"
  - Creates the UUID CFTOKEN by prepending a random 16-digit hexadecimal number to a ColdFusion UUID value
    - 3ee6c307a7278c7b-5278BEA6-1030-C351-3E33390F2EAD02B9
- Solution in CF4.5/5
  - Can make registry entry change (in simulated registry on Linux) to effect similar change in behavior
- Note as well, using "J2EE sessions" uses Jsessionid instead
- Resource:
  - "How to guarantee unique CFToken values"
    - http://www.macromedia.com/support/coldfusion/ts/documents/tn18133.htm

Unexpected Session Timeouts

- Challenge: Unexpected Session Timeouts
  - Users report that their sessions are being lost sooner than they expect
    - Perhaps they're being kicked back to the app's login screen
- Solution
  - Could be that server is restarting frequently
    - Investigate is there's trouble, or server is being restarted intentionally
  - Could be that session timeout for application is too low
    - Can raise timeout time, but no higher than "max" set in Admin console
  - Be aware of mix of CFMX/J2EE timeouts ("session invalid" error)
    - http://www.macromedia.com/support/coldfusion/ts/documents/session_invalid_j2ee.htm
  - Could implement feature to keep sessions alive on browser
    - See Apr 2000 CFDJ article, "Avoiding Unwanted Session Timeouts"
      - http://www.sys-con.com/story/?storyid=41925&de=1
  - Be aware of resources used by keeping sessions alive longer

Tracking Sessions

- Challenge: Tracking Sessions
  - People often want a tool to report how many sessions are active
    - Security concerns preclude built-in mechanisms allowing one user to see the session data of another
- Solutions for CF4 and above
  - Still, developers have created mechanisms (custom tags, applications, code snippets) to track sessions
    - In database, in application scope, and more
    - CFDJ Article, "Live Monitoring of User Sessions"
      - http://www.sys-con.com/story/?storyid=41950&DE=1
    - http://www.cfhub.com/advanced/cfapplication/applicationexample.cfm
    - http://www.teratech.com/coldcuts/cutdetail.cfm?cutid=211
    - http://tech.badpen.com/index.cfm?mode=entry&entry=3
- Solution for CFMX
  - Macromedia has an undocumented library for tracking sessions
    - coldfusion.runtime.SessionTracker

Tracking Sessions (cont.)

- CFMX's "coldfusion.runtime.SessionTracker" Example

```
<cfset x = "">
<cfset sessionTracker =
    x.getClass().forName("coldfusion.runtime.SessionTracker").newInst
    ance()>
<cfset sessionKeys = sessionTracker.getSessionKeys()>

<cfloop condition="#sessionKeys.hasMoreElements()#">
    <cfdump
    var="#sessionTracker.getSession(sessionKeys.nextElement())#">
</cfloop>
```

- Beware
  - Undocumented, could change, may not behave as you'd expect
  - Also, security concerns:
    - http://tech.badpen.com/index.cfm?mode=entry&entry=4

Tracking Sessions (cont.)

- Others Solutions available when running CFML on J2EE (using CFMX or BlueDragon)
- Can leverage J2EE "listeners"
  - Resources:
    - "Making the Most of J2EE Event Listeners"
      - http://www.sys-con.com/story/?storyid=44774&DE=1
    - "More Servlets and JSP", Chapter 11 code for tracking sessions
      - http://archive.moreservlets.com/Chapter11.html
- J2EE server admin consoles often also offers session tracking/tools
  - Enabling "JRun Connection Monitoring"
    - http://livedocs.macromedia.com/jrun/4/JRun_Administrators_Guide/netmon.htm
- "Java Application Monitor (JAMON)" tool
  - http://www.javaperformancetuning.com/tools/jamon/index.shtml
  - Offers CFML code sample to integrate into CF apps

Handling Sessions When Cookies Are Not Enabled

- Challenge: Handling Sessions When Cookies Are Not Enabled
  - Some browser users will disable support for cookies
    - Also, some very old browsers (and some wireless phones) don't support them
  - Without cookies, a new session id (CFID/CFTOKEN/JSessionID) will be generated for each request from a user
    - Will seem that their session variables are never "set"
- Solution:
  - Must pass sessionid on each request from browser to server, using CFML to set the value on A HREF, FORM, CFFORM, and CFLOCATION
  - Must determine whether to send CFID/CFTOKEN/JSessionID depending on whether using client and/or session variables, and if J2EE sessions
  - In CFMX/BlueDragon, available new URLSessionFormat function helps
    - Wrapped around a URL, it determines whether (and which) id is needed
- Resource:
  - "Using client and session variables without cookies"
    - http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/shared11.htm

Terminate Session At Will

- Challenge: Terminate Session At Will
  - Perhaps on logout, want to force termination of session (or part of session)
  - Can both protect user and also preserve resources in high volume environment
- Solution in CF4 and Above, CFMX, and BlueDragon
  - StructDelete function to delete a single session variable
    - StructDelete(session,"keyname")
  - If using "J2EE Sessions", can use J2EE method to invalidate session
    - <cfset getPageContext().getSession().invalidate()>
    - In BlueDragon, has benefit of clearing session and causing new JSessionID
    - **Warning: in my tests in CFMX, using this feature causes "session is invalid"**
  - Tempting to use StructClear to clear entire session scope
    - Several challenges …

Terminate Session At Will (cont.)

- Problems with using StructClear on sessions
  - StructClear clears SessionID/CFID/CFtoken built-in variables as well as your data
  - Also, the user may legitimately be using another window to talk to another app on the same site.
- Solutions
  - Instead, clear the critical session variables individually
  - Or put your data in a structure in the Session scope, then clear that structure
    - For example, put all your application variables in Session.MyVars and then call StructClear(Session.MyVars) to clear the variables
- Resource: MM TechNote 14143
  - "ColdFusion 4.5 and the StructClear(Session) function"
  - Applies to CF4.5 through CFMX (and BlueDragon)

Locking Session Variable Access

- In CF4 and 5, developers were warned to use CFLOCK around all access (read and write) to sessions
  - Features were also added to the CF Admin Console to control server-wide locking
- In CFMX (and BlueDragon), need is greatly diminished
  - Locks needed only to prevent "race conditions", where logic might update a variable if run by two or more threads at once
  - Resource:
    - http://www.macromedia.com/support/coldfusion/ts/documents/tn18235.htm

Enhancements for CFML on J2EE and .NET

- Enhancements for CFML on J2EE and .NET
  - Sharing Sessions Between CFML and JSP/Servlets and ASP.NET
  - Persistence Over Restarts or for Failover
  - Replication Across Other Servers (Failover/Load Balancing)
- All require enabling of "Use J2EE Sessions"
  - Same name used for equivalent feature BlueDragon/.NET as well

Sharing Sessions Between CFML and JSP/Servlets

- Can now integrate CFML and JSP/servlets, including sharing Session/Application/Request scope variables
  - Sessions set in one are available in the other
  - Again, if "J2EE Sessions" are enabled in CF/BD Admin
- Available in the following deployments
  - CFMX Enterprise (standalone and J2EE)
  - BlueDragon Server JX and BlueDragon/J2EE
  - Not available in CFMX Standard or BlueDragon Server (free edition)
- Note
  - If CFAPPLICATION specifies a NAME attribute (as is typical), session variables in JSP/servlet will be stored within a "map" (structure) of that name
    - Otherwise accessible in JSP/servlet with same name as CFML
- Resource
  - MM Manual, "Developing ColdFusion MX Applications", Chap 33 "Integrating J2EE and Java Elements in CFML Applications"
    - http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/java.htm#wp1176234
  - "Together at last: Sharing session data between ColdFusion and J2EE components"
    - http://www-106.ibm.com/developerworks/ibm/library/i-coldstudio/

Sharing Sessions Between CFML and JSP/Servlets (cont.)

- Example of accessing session in JSP
  - If CFAPPLICATION has no NAME:
    - Getting: <%= session.getAttribute("***varname***") %>
    - Setting: <% session.setAttribute("***varname***","***somevalue***"); %>
  - If CFAPPLICATION has NAME ("test" in this example):
    - Getting

```
<%
java.util.Map map = null;
map = (java.util.Map) session.getAttribute("appname");
%>

<%= map.get("varname")%>
```

    - Or

```
<%@page import="java.util.*" %>

<% ((Map)application.getAttribute("appname")).get("varname");%>
```

    - Setting

```
<%@page import="java.util.*" %>

<% ((Map)application.getAttribute("appname")).put("varname","somevalue");%>
```

Sharing Sessions Between CFML and ASP.NET

New Atlanta COMMUNICATIONS

- .NET
  - Same feature of shared session/application/request scopes is available between CFML and ASP.NET
  - Available on BlueDragon/.NET only
- Examples (assuming application has no NAME)
  - Getting

```
<%@ Page language="c#" AutoEventWireup="false" %>
<% Response.Write(Session["varname"]); %>
```

  - Setting

```
<%@ Page language="c#" AutoEventWireup="false" %>
<% Session.Add( "varname", "somevalue" ); %>
```

Persistence Over Restarts or for Failover

- Overview
  - When server is restarted, what happens to session?
    - Recall problem of "Unexpected Session Timeouts"
      - Or if load balancing/failover forces user to new machine
    - Sessions are typically stored in server memory, so lost at restart
  - Solution: most J2EE servers offer option to persist sessions
    - Stored optionally to file system, database, state server, or other
    - Combines best of client and session variables
      - Sessions can last longer and are preserved over restarts
  - When does persistence take place?
    - Manually, after an interval, or on any update
- J2EE
  - In J2EE servers, may be enabled in admin console
    - In JRUN admin console, select web application and see "General settings", then "Enable File-based Session Persistence"
      - Couldn't get it to work
  - Most J2EE servers also enable this via a setting in an XML file

Persistence Over Restarts or for Failover (cont.)

- In WebLogic, for instance, edit/create weblogic.xml file (in WEB-INF) directory

```
<weblogic-web-app>
 <session-descriptor>
    <session-param>
     <param-name>PersistentStoreType</param-name>
     <param-value>file</param-value>
    </session-param>

    <session-param>
     <param-name>PersistentStoreDir</param-name>
     <param-value>PathToFileForStorage</param-value>
    </session-param>
 </session-descriptor>
</weblogic-web-app>
```

- To return to memory-based sessions, set param-value to **memory**

Persistence Over Restarts or for Failover (cont.)

New Atlanta COMMUNICATIONS

- .NET
  - Can set persistence via setting in web.config per application
    - (or machine.config for server-wide control)
- .NET also offers concept of a "state service", a Windows service to manage persistence of sessions
- XML settings to change in config file

```
<configuration>
    <system.web>
      <sessionState
          mode="StateServer"
          stateConnectionString="tcpip=127.0.0.1:42424"
          cookieless="false"
          timeout="5"/>
    </system.web>
</configuration>
```

- To return to memory-based sessions, set **mode="InProc"**
  - Other values are **"SQLServer"** and **"Off"**

Replication Across Other Servers (Failover/Load Balancing)

- Overview
  - When using clustering/load balancing/failover, when user is transferred to a new server, what happens to their session?
  - Simplistic solution: "sticky sessions"/affinity
    - Force user to remain on single server for life of session
      - If failover, session lost
  - Better solution: persistence to database/file system/cookies
    - As discussed in previous topic
  - Still another alternative: session replication
    - May be in-memory across servers, to database, and more

Replication Across Other Servers (Failover/Load Balancing) (Cont.)

- Capabilities vary by J2EE Server
  - Again, some may enable config in admin console
    - In JRUN admin console, select web application and see "General settings", then "Enable Session Replication"
  - Or may be enabled using a setting in an XML file
- WebSphere Network Deployment
  - CFMX is unable to work on WSND
    - BlueDragon works as expected
  - See technote from Macromedia
    - http://www.macromedia.com/support/coldfusion/ts/documents/was51_support.htm
- .NET also offers replication of sessions
  - And will therefore be enabled for CFML on BlueDragon/.NET

Replication Across Other Servers (Failover/Load Balancing) (Cont.)

- Resources:
  - "Developing Web Applications for WebLogic Server", "Using Sessions and Session Persistence in Web Applications"
    - http://edocs.beasys.co.jp/e-docs/wls/docs81/pdf/webapp.pdf
    - See as well
      - "Using WebLogic Server Clusters", "HTTP Session State Replication"
  - "Clustering and Load Balancing in Tomcat 5"
    - http://www.onjava.com/pub/a/onjava/2004/03/31/clustering.html
  - "Tomcat 5 Clustering/Session Replication"
    - http://jakarta.apache.org/tomcat/tomcat-5.0-doc/cluster-howto.html
  - "In Memory Session Replication In Tomcat 4"
    - http://www.theserverside.com/articles/article.tss?l=Tomcat

Enhancements for CFML on J2EE, Resources

New Atlanta COMMUNICATIONS

- Other Resources for CFML/J2EE Integration
  - "Making the Case for CFML on J2EE"
    - http://www.sys-con.com/story/?storyid=44481&DE=1
  - "CFML on J2EE: Easy as 1-2-3"
    - http://www.sys-con.com/story/?storyid=45338&DE=1
- Resources for J2EE Session Mgt
  - Sun Servlet API docs for Session Object
    - http://java.sun.com/j2ee/sdk_1.3/techdocs/api/javax/servlet/http/HttpSession.html
  - "Managing HttpSession Objects"
    - http://www.sys-con.com/story/?storyid=37330&DE=1

Summary

- Many frequent session handling challenges can be solved
  - Terminate Session on Browser Close
  - Insecure SessionIDs
  - Unexpected Session Timeouts
  - Handling Sessions When Cookies Are Not Enabled
  - Terminate Session at Will
  - Locking Session Variable Access
- These can be solved for both CF 4 and 5, as well as in CFMX and BlueDragon (whether on a J2EE server or not)
  - Simpler solutions work on CF4 and 5
  - "J2EE Sessions" feature adds more power, useful even on standalone versions of CFMX and BlueDragon
- Deploying CFML on J2EE servers adds still more features
  - Solve problems of integration, persistence, and replication
- Deploying CFML on .NET, with BlueDragon, opens still more doors
  - Integration, persistence, replication, and more

Contact Information

- Charlie Arehart
- CTO, New Atlanta Communications
- charlie@newatlanta.com
- newatlanta.com/bluedragon/
- (678) 256-5395