Marrying Java & ColdFusion

March 2002

Charles Arehart
Founder/CTO SysteManage
Our Practice Makes You Perfect ™
carehart@systemanage.com


Agenda

• Considering Java as an alternative
• Integrating CF and Java Today
  – CFOBJECT, CFX_ Custom tags, CFSERVLET
• Leveraging Different Approaches
  – Java Language features
  – Existing Java classes
  – Using Java API to 3rd party tools
  – Existing Custom tags to solve problems
  – Creating your own classes, custom tags
  – CFServlet (and servlets, JSPs and EJBs)
• Future integration possibilities for CF’ers
• Software Needed
• Learning More


Marrying Java & ColdFusion 1
Marrying Java & ColdFusion

Audience

- Intended primarily for those with:
  - ColdFusion experience
  - Aware of Java and its potential value to application developers
    • Or just curious
- No Java experience expected
- Anyone interested in:
  - Learning about Java integrating ColdFusion & Java
  - Extending ColdFusion capabilities via Java
  - Will likely come away capable of doing it immediately

Our Goal

- Goals
  - Put Java in perspective for CF developers
  - Get a flavor for opportunities, challenges
  - Suggest ways to integrate the two today
  - Muse on where the future may lead for us
  - Point to resources for learning more
Who Am I?

• A programmer by trade, trainer at heart
  – With nearly 20 years IT experience
    • 5 years ColdFusion, 1 year server-side Java
  – Monthly contributor to ColdFusion Developer Journal, bi-monthly columnist in Java Developer Journal
  – Speaker at user groups, conferences worldwide
  – Macromedia Certified Instructor & Developer
  – Course developer
• Best practices, architectural consultant
  – Contract mentor to development teams

The Java Siren Song

• Java: it’s here & it’s big
  – As a platform and language used by thousands of orgs
  – CFMX (aka Neo) will run on a Java platform
    • But you need not learn Java to use CFMX
• Still, may have been feeling pull
  – Some orgs, developers DO want to leverage Java
    • Now, as well as in CFMX
  – May have been able to “fight” getting into it
  – Time has come to face the truth
    • A large market is demanding Enterprise java solutions
    • Also, many tools coming out on the market, built in Java
Server vs. Client-side Java

- **Java’s been known by many for use in applets**
  - Some even worry about CFMX and “java client security”
- **Our focus is on server-side java**
  - Not client-side
  - As with CF, we can just be generating HTML
- **Microsoft’s pulling JVM out of Future Windows Operating Environments**
  - Is it a threat to Java’s future?
    - May have more effect on client side
    - Not that big a deal for server-side developers

Considering Java as an Alternative

- **Not really about caving in**
  - Learning how and when to best leverage Java as a CF developer
    - Not about leaving CF for Java
    - More about how to best marry the two
      - Getting easier all the time
- **Not even necessarily about learning java**
  - Just when and how to integrate java tools
    - Though certainly, to do much, will need to learn java
- **Free to get started**
  - Most platform tools are free, for developer level deployment
- **Much of what we cover should also apply in CFMX**
  - Because it’s based on Java platform, should integrate even better
Why Bother?

- **Benefits for current projects**
  - Leveraging java library functions for features
  - Accessing existing apps/modules within enterprise
    - Whether written as classes, EJB’s, or Servlets
  - Creating new classes to extend functionality
- **Benefits as a professional developer or development firm**
  - Many tools available, written in java
    - Saves you having to build yourself
    - May be packaged for use with JSP/Servlets
      - Can easily figure out how to convert for use in CF

Ways To Use Java

- **To call existing things**
  - Native Java language offering richer environment
  - Classes, Servlets, CFX_custom tags you or someone else has built
    - Can be a class that produces a result (like a subroutine)
      - Or perhaps a JavaBean
    - Or one whose methods, properties you manipulate within CF
  - Tools that have java API
    - Such as charting tools, PDF creation tools, etc.
      - may choose to use API rather than some custom tag
  - Or …
Ways To Use Java

• To call existing things (cont.)
  – EJB’s
    • Those built within company to support applications
    • Tools that have EJB interfaces
      – (NetCharts EJB, for example)
  • Or to create any of these new, for your apps

What Can Java Give Us?

• Among the things Java can offer CF developers include:
  – XML manipulation
  – Mail handling
  – Networking
  – Compression
  – Encryption
  – Graphics generation
  – Image creation and manipulation
  – PDF creation and formatting
  – Spell checking
  – And much much more
• Enterprise integration
Finding Existing Examples

- **Macromedia Tag Gallery** offer downloadable examples of CFX's, objects usable by CFOBJECT
  - See separate “java” link on left in taggallery
  - Also, dozens of things found based on search of “java” at taggallery
    - Search criteria used
      - java not applet not "java script" not "no java" not "java tree"
    - This ignores applets, other extraneous things not relevant to this topic
  - Examples:
    - CFX_zipbrowser, CFX_html2pdf, More later

- **Also third party sites, like CFDev.com**
  - Free and commercially available java custom tags

- **As well as javasoft.com (Sun Microsystems)**
  - for general interest Java resources, examples, etc

Going Outside of CF Entirely

- **Could consider working outside of CF entirely**
  - Java Servlets
  - JavaServer Pages
  - Enterprise JavaBeans
  - And more

- **Will discuss this more later**
  - Would be entire other discussion
  - Focus for now is on just integrating CF and Java
Integrating CF and Java Today

• **CFOBJECT for Java**
  – Used to call upon java classes available on the same server running CF
    • leverage java language
    • access existing tools, business objects, including EJB’s

• **CFX_ Java Custom tags**
  – Used to call upon Java classes on same server
    • leverage CF integration within the Java class (accessing/updating CF variables, queries, etc.)
    • Many existing CFX_ tags in Macromedia Taggallery, elsewhere

• **CFServlet**
  – Used to integrate with Servlets on same or other server
    • Either obtaining the output of the servlet
      – pass data to/from servlet

Installation Support for Java Integration

• **To leverage CFOBJECT and CFXs for Java**
  – Must have at least 4.5.1 of ColdFusion
  – Must have Java Runtime Environment 1.2+
  – Must configure CF server Admin to setup classpath and other parameters
    • Critical points are
      – JVM path: where JVM is installed (and jvm’s name)
        » Mine is: D:\jdk1.3\jre\bin\classic\jvm.dll
      – Classpath: where your classes will be stored
        » Must restart CF server after changing
      – CFX_jar path: where cfx.jar used by CFX’s will be stored

• **See later slide “Software Needed”**
  – how to obtain related software, versions, etc.

• **See “A Cold Cup O’ Joe” Part 1, by Guy Rish**
  – Jan 01 CFDJ (http://www.sys-con.com/coldfusion/article.cfm?id=202)
  – Great discussion of config, setup issues, challenges
CFOBJECT for Java

- New support as of 4.5 for CFOBJECT
  Type="Java"
  `<CFOBJECT ACTION="CREATE" TYPE="Java"
  CLASS="classname" NAME="classref">`
- Also createobject() function for use in CFSCRIPT
  `<CFSCRIPT>
  classref = createobject("java","classname");
  classref.methodname(parameter);
  </CFSCRIPT>`
- Allows call to java objects
  - Either native java language objects or custom ones
  - Just need to know methods and attributes

Calling Java for Extended Functionality

- If there’s something Java can do, call it
- Example:
  ```
  <!--- // Create a java.util.GregorianCalendar Object // --->
  <CFOBJECT ACTION="CREATE" TYPE="Java" CLASS="java.util.GregorianCalendar"
  NAME="myCalendar">```
  <!--- Default Constructor is being called implicitly --->
  <!--- Check if 2001 is a leap-year --->
  <CFSET is2001LeapYear = myCalendar.isLeapYear(2001)>```
  <!--- Retrieve current date --->
  <CFSET theYear = myCalendar.get(myCalendar.YEAR)>```
  <CFSET theMonth = myCalendar.get(myCalendar.MONTH) + 1>```
  <CFSET theDay = myCalendar.get(myCalendar.DATE)>```
  `<cfoutput>
  <html>
  <body>
  Is 2001 a leap-year? <b>#is2001LeapYear#</b>
  Today is the <b>#theMonth# / #theDay# / #theYear#</b>
  </body>
  </html>`
  ```
Extending with Java (cont)

- Notice that that involved no compilation of any java classes
  - Was leveraging Java libraries from within CF
    - Converted from code offered in a java book
    - Just need to see how to convert to CFOBJECT
      - Invoke methods like functions
      - Get/set properties like normal variable use
    - No need to even add any classpath to CF Admin
- Can only really benefit if you understand Java
  - More to the point, know the various capabilities of the language libraries
  - See http://java.sun.com/j2se/1.3/docs/API/index.html
    - For online documentation of libraries

Calling a Class to Execute Some Tool

- Many tools now provided as java classes
- May be able to call upon them from within CF
  - With no java coding, per se, required
  - See next slide’s example, calling
    - Popchart from corda.com
      - Java-based dynamic server-side charts
      - Relatively inexpensive
      - also supports CF w/out Java (using URL’s to pass data)
Using a Tool’s Java API

- If tools offers Java API, review docs and convert to calls within CF
  - Here is an example of calling PopChart from Corda

**Example:**

```cfscript
<cfscript>
  graphImage=createobject("java","com.corda.pcis.PCISEmbedder");
  graphImage.setServerInfo("http://localhost:81");
  graphImage.setWidth(540);
  graphImage.setHeight(330);
  graphImage.setAppearanceFile(“apfiles/bargraph1.bin”);
  graphImage.setPCScript("textbox.settext(Hello) graph.categories(Math, Science, English) graph.series(Fred, 16, 87, 76; Gus, 68, 77, 85)");
  eHTML = graphImage.getEmbeddingHTML();
  writeoutput(ehtml);
</cfscript>
```

Using a Tool’s Java API (cont)

- Converted from code offered in their docs
  - Again, just need to see how to convert to CFOBJECT
    - invoke methods like functions
    - Get/set properties like normal variable use

- Must set classpath in CF admin to wherever that code (like the PopChart product code library) is stored
  - For me, I added:
    - D:\ProgramFiles\Corda\image_server\Java_API_Framework\PCISEmbedder.jar;D:\ProgramFiles\Corda\image_server\pcis_classes\PopChartServer.jar
  - Then restart CF server
Calling a Java Class You Build

- Can treat the opportunity like a subroutine, calling a java class you’ve created
  - May provide business rule functionality

- Examples
  - See next slide for example

- Need to build and compile the java class
  - Put it somewhere in the defined classpath, or modify the classpath, as defined in the CF admin
  - Need to know how to write java
    - More in a moment

- When creating own Java classes, beware:
  - Must restart CF server after recompile of any class instantiated via CFOBJECT (or change of classpath)
    - Fixed in CF5 with new dynamic class load path

Calling a Java Class You Build (cont.)

- Java program, grades.java

```java
public class grades {
    public static char letterGrade(int numGrade) {
        // Returns the letter grade corresponding to numerical grade, numGrade
        if (numGrade >= 90) return 'A';   // 90 or above gets an A
        else if (numGrade >= 80) return 'B';   // 80 to 89 gets a B
        else if (numGrade >= 65) return 'C';   // 65 to 79 gets a C
        else if (numGrade >= 50) return 'D';   // 50 to 64 gets a D
        else return 'F';   // anything else gets an F
    } // end of function letterGrade()
}
```

- Compiled as grades.class, in classpath

- CF Code to call that:

```cfc
<cfscript>
<cfobject action="CREATE" type="JAVA" name="x" class="grades">
<cfset ret = x.letterGrade(40)>
<cfoutput>#chr(ret)#</cfoutput>
</cfscript>
```
More Elaborate Example

- See Allaire article, “Leveraging Java Classes from ColdFusion”
- Shows how to use a java class to read in a text file
- Better than using CFFILE for very large files
  - extends Java library BufferedReader class
  - Handles problem with CF not being able to detect null at end of file
- Let’s see it run!

Building Java Classes

- To build your own classes, need to know how to write java
  - More than just learning language
    - On simple level, must also learn how to create java code
      - Various issues such as class name, case sensitivity
      - Compilation: how, where
    - On larger level, should learn java language concepts
      - Object-oriented programming, inheritance, polymorphism
      - Constructors, packages, interfaces, inner classes, etc.
      - Many concepts foreign to procedural programmers
  - Plenty of good books on Java
    - More on “reading list” later
Compiling Java Classes

- **Need to know how to execute compiler**
  - Need Java SDK, not just Java Run-time
  - Need to find javac.exe
    - Mine is in: D:\jdk1.3.1_02\bin
  - Need to execute the compiler
    - Either from command line
      - Start>Programs>Accessories>Command prompt in Win2k
    - Or from within Java Editor
      - such as Jrun Studio, discussed later
  - Likely need to define location of compiler on system path
    - So that you can execute it from any directory on workstation
  - In win2k, use Start>Settings>Control Panel>System>Advanced>Environment Variables>System Variables
    - Add compiler location to PATH variable
  - Typically run compiler from location of .java files
    - Typically will just want the resulting .class files placed there as well

Some Challenges

- **For more complex uses**
  - Need to understand calling an object’s default constructor or overloaded constructors
    - Does not currently support overload by attribute datatype
    - Available javacast function to control type passed to java
  - “Cold Cup O’ Joe” Part 3 discusses much of this
    - April 01 CFDJ
  - Release 5 adds new GetException() function
    - To manage java exceptions from CFOBJECT more effectively
Calling EJB’s

- CFOBJECT can also be used to call Enterprise JavaBeans
  - Demonstrated in Macromedia “Developing Applications with JRun”, chapter 42
  - EJB’s run in an EJB container (JRun is one such)
  - Some tools now available as EJB’s
    - NetCharts EJB from visualmining.com
  - Currently, it may be better to call EJB’s via Custom Tags
    - Current approach in CFOBJECT calling EJB’s has high overhead
    - KB 21977 explains how use of “custom class loader” is the solution

CFOBJECT Summary

- 4 approaches to using CFOBJECT
  - Call on core java language APIs
  - Call on a 3rd party tool’s API
  - Create your own classes
    - Or perhaps call upon your own organizational objects
  - Create or call EJBs
    - Though better to call these (for now) in custom tags
CFX_ Java Custom Tags

- Similar to calling upon objects, but ...
  - Class can read/manipulate CF elements
    - Variables, queries
  - Can also pass it attributes
    - Processed with request.getAttribute in class
  - Can also generate output that will be shown within stream of output from the calling CF template
    - Just like regular custom tags
  - Will see simple example

- Must be registered in the CF administrator
  - Then called as <cfx_objectname>

CFX Example

- HelloColdFusion.java

```java
import com.allaire.cfx.*;
public class HelloColdFusion implements CustomTag {
    public void processRequest( Request request, Response
        response )
        throws Exception
    {
        String strName = request.getAttribute( "NAME" ) ;
        response.write( "Hello " + strName ) ;
    }
}
```

- Compiled as HelloColdFusion.class, in CF Admin classpath
- Registered in Admin as CFX_HelloColdFusion

- CF Code to call that:

```
<CFX_HelloColdFusion NAME="Bob"/>
```
Compiling CFX’s

- 3 steps: compile, define in admin, call
- In compiling CFX’s
  - Make sure system classpath includes full path to `\cfusion\java\classes\cfx.jar`
    - Not just the path, but the jar itself
    - Note this is for compiling
      - refers to classpath set in OS, not in CF Admin
  - Or, specify in javac command, as in:
    ```java
    javac -classpath c:\Cfusion\Java\classes\cfx.jar compileme.java
    ```
  - Prior to CF5
    - Must restart CF server after any recompile of CFX
      - If it’s been executed since start of CF server
    - CF5’s new dynamic Java classpath feature does not seem to apply

Defining CFX’s in CF Admin

- In server>java section
  - Provide name, choose type, enter classname
    - Do not provide “.class” extension
  - If misspelled, or not found on classpath
    - Error will be ClassNotFoundException
    - Case is sensitive
      - Error will be NoClassDefFoundError
  - No need to restart based on these changes
    - But will need to restart if class itself is changed after being called
      - Except if new CF5 dynamic classpath feature is set
Calling CFX’s

- In calling `<cfx_>` tag
  - Portion of name after `cfx_` is not case sensitive
    - `<cfx_test>` is same as `<CFX_TEST>` or `<cfx_Test>`
      - At least on Windows platform

Admin Classpath Traps

- May sometimes have difficulty due to failure in classpath set in CF
  - May not be what you think it should be
  - Create simple custom tag to report it
    ```java
    import com.allaire.cfx.*;
    public class show_classpath_via_cfx implements CustomTag
    {
        public void processRequest( Request request, Response response )
            throws Exception
        {
            response.write(System.getProperty("java.class.path"));
        }
    }
    ```
  - Register it as a CFX, call it
    - Will display classpath
    - May not be what you think it should be
      - CF seems to manipulate what’s placed in the admin
      - Automatically places cfx.jar path there
Admin Classpath Traps (cont)

- Be sure to restart CF server after any recompile of CFX or CFOBJECT called class
  - And be sure to restart CF server (not exec or RDS)
- Also restart after changing Classpath in admin
  - Be sure to use correct slashes in paths in classpath
- Be aware that jar files in a given directory are not searched unless on path by name (.jar)
  - When errors occur, can use file text search feature to find jar that may contain needed class
- CF Admin screen has 500 char limit on width of Classpath input field

More about Java CFXs

- Several available methods, attributes for developing java Custom Tags
  - Query interface
  - Request Interface
  - Response Interface
- See “Developing Web Apps” for details
  - Chapter 18
- May hear of old way of doing Java CFX’s, CFX_J
- Open debate about performance of CFX versus CFOBJECT
### Summary of CFOBJECT/CFX

- **4 approaches to using CFOBJECT**
  - Call on core java language APIs
  - Call on a 3rd party tool’s API
  - Create your own classes
    - Or perhaps call upon your own organizational objects
  - Create or call EJBs
    - though better for now to do in custom tags
- **2 approaches to using CFX Java Custom tags**
  - Call on 3rd party CFX’s to add application functionality
  - Create/call on own CFX’s to perform some processing in Java
    - Remember that main point of CFX’s is to read/write CF variables/queries and/or generate response output

### JRun and Java Application Servers

- **JRun is a J2EE Java Application Server**
  - And a very well respected one, at that
  - Reasonably priced, free developer edition
- **Others include**
  - IBM Websphere
  - BEA Weblogic
  - Apache TomCat
- **Enables execution of JavaServer Pages, Servlets**
  - JRUN also includes features to run EJBs, and more
About J2EE

- Alternative for web application development
  - JSPs
  - Servlets
  - EJBs
  - And more
- JSP’s ease entry into Java world
  - Work similarly to CF, scripted web app development
- See “Learning more”

Why Consider JSP/Servlets?

- Lots more clients clamoring for JSP/Servlets than CF
  - Still plenty of market for CF, just larger for JSP/Servlets
- “Learn once/Run Anywhere”
  - Skills learned in working with JRun migrate for the most part to other java application servers
    - IBM Websphere, BEA WebLogic, Apache TomCat, etc
- What can JSP/Servlets do that CF can’t?
  - On a simple-to-intermediate level, not much
    - More the integration possibilities
    - May promote better design, separation of implementation/interface
  - But now, knowing how CF can integrate Java, maybe not as critical
    - Choose the best platform for each task
Integrating CF and JRUN: CFServlet

• Allows call to servlet running under JRun
  – Also allows sharing of data between them
  – Output of servlet can be shown within calling template
  – Only works with JRun
• Need only know
  1. Server’s IP address, if not same as CF server
  2. JRun JCP Port (JRun and other app servers use unique ports to distinguish from web server)
     • Use value of jcp.endpoint.main.port in server’s local.properties file
  3. Servlet name
     • <CFSERVLET code="SimpleServlet" jrunproxy="127.0.0.1:51001”>
• Let’s see an example
  – See “CFServlet Issues” for problems calling “demo app” servlets

Passing Data to/from CFServlet

• Allows passing data by value or reference
  – By value:
    • <cfservletparam name="servletparm" value="value">  
    • Available to servlet as “parameter”
    • Changing parameter in servlet has no effect on CF variable
  – By reference:
    • <cfservletparam name="servletattr" variable="CFVarname”>
    • Optional “type” attribute to help CF tell java the datatype
    • Available to servlet as “attribute”
    • Changing attribute value in servlet will change CF variable
• Optional WriteOutput="yes/no” attribute
  – If no, output of servlet is written to CF output stream
  – If yes, written to Cfservlet.output variable
Message Passing

- One last thought on integration issues
- When considering communications between CF and java components, consider XML
  - XML provides powerful means to pass structured data between disparate systems
  - Need not be Macromedia’s WDDX format
    • If you control the two systems talking to each other, you can create your own meaning in the XML
  - See “Online Ticket Store” in Jan 2000 CFDJ

CFSERVLET Issues

- May hear of old way of doing this: CF_Servlet
- Always passes any value of CGI.Query_String
- Remember, can read/write CF variables/queries
  - Can optionally show servlet output within calling CF template’s output
- Could also just use CFHTTP
  - If no need to read/write CF variables/queries from within servlet
  - And if wanting to include output from non-JRun servlet
CFSERVLET Issues

• By default, can only run servlets in Default-App
  – Means it won’t work with JRun demo apps
  – Unless you add the path for that class directory (for me, {jrun.rootdir}/servers/default/demo-app/WEB-INF/classes) to the other locations listed in JMC under JRun Default Server>Java Settings>ClassPath
    • Restart Jrun default server
• May need to modify Jrun Server’s local.properties file
  – May need to add “.jcp” to end of servlet.services, as in:
    • servlet.services=jndi,mail,url,(servlet.webapps),web,jcp
  – If “web connector” is run, it may change the value of the jcp.endpoint.main.port
  – Some changes in JMC may also cause loss of jcp on list of services
    • If attempt to use CFServlet fails, look into that

Simpler Forms of Integration

• Keep in mind that Servlets (and JSPs) can also be called from within CF templates just as regular HTML pages could:
  – <A href>
  – <cflocation>
  – <cfhttp>
  – <form action=>
  – <img src>
  – Etc.
• So can “integrate” them into your application without need to use CFSERVLET
Future integration possibilities for CF’ers

- **ColdFusion MX, a.k.a Neo**
  - Changing CF from running on a C++ platform to running on a Java platform
  - Does NOT propose to change CF developers into Java developers
    - CF templates will run unchanged
    - Under the cover, will be turned into Java Servlets (it’s said)
  - Will open further integration possibilities
  - More important for Macromedia
    - Allows them to leverage the Java platform in building CF components (underlying CF tags)
      - they currently must build many things from scratch that are available as Java libraries
    - Will allow them to focus energy on creating new features

Software Needed for Java

- **To do any java integration, need Java Runtime Engine on server**
  - Can obtain free from javasoft.com
- **To run JRun or do JSP/servlet integration, will need a Java application server**
  - Macromedia offers free 3 person developer edition
  - Will need Java 2 EE (J2EE) SDK on top of that
    - Available on install, also free from javasoft.com
  - Other java app servers from IBM, BEA (beasys.com), Apache Tomcat (apache.org), and more
Software Needed (cont.)

- **To do EJB integration, need EJB server**
  - JRun Enterprise provides that, as do other products
- **ColdFusion 4.5.1 +**
  - added java custom tags and CFObj ect support for java
- **Pre 4.5.1**
  - cfx_j custom tag
  - cf servlet custom tag

Java Editors

- **Editing JSP is like editing CF (mix HTML and java)**
  - Java classes, CFX Custom tags, Servlet are pure java
  - CF Studio can act as JSP editor
    - Knows about JSP tags, etc.
    - Kawa is a real Java editor that Macromedia acquired
      - Can compile things, debug, etc.
      - Was dropped recently
    - JRun Studio is a hybrid
      - Adds ability to compile to CF Studio interface
      - Not really a true Java IDE
  - Dreamweaver Ultradev
    - Includes features to build/edit CF, JSP, ASP code
  - Other Java editors have support JSP’s/Servlets
    - IDEs: JBuilder (Borland), Forte (Sun), Visual Age (IBM)
    - Editors: JPadPro, UltraEdit-32, many others are shareware
More on Things at TagGallery

- CFX_HTML2PDF
  - Interesting idea, messy implementation, hard to use as-is
- CFX_Query2PDF
  - Better (but narrower focus). Many formatting options
- Internet Source Control (remote SCC)
- Java2html (display java source nicely)
- CFX_ZipBrowser (zip files on server)
- CFX_JAVA (?)
- XS (XML/XSL transformations)
- Simple JavaBean Wizard
- KavaChart
- Other sources:
  - CFDev.com, various java-oriented component repositories

A Reading List

- Macromedia CF documentation
  - CFObject, CFX_custom tags, CFServlet
  - Language Reference and “Developing Web Apps”
  - Administering CF (for admin setup)
- Macromedia JRUN documentation
  - Even without java experience, they make using understanding JSPs/servlets rather easy
A Reading List

- **JavaServer Pages Application Development**
  - Ben Forta, Scott Stirling, and company
- **JavaServer Pages (O’Reilly, Hans Bergsten)**
- **Beginning Java Objects**
  - Great book for fundamentals of java/objects/design
- **Other good J2EE Books**
  - Core Servlets and JSP
  - Professional Java Server Programming (J2EE Ed.)
  - Enterprise JavaBeans
- **Other good foundational Java books**
  - Core Java2; Beginning Java2; Thinking in Java

Beware of Some Books

- **Servlet/JSP books often teach to java developers**
  - Focus more on showing them how to do web application development (stateless programming, passing data from URL’s and forms, etc.)
  - Presume that reader already knows Java
    - Core Servlets and JSP (Hall), Professional Java Server Programming (Patzer)
  - Some presume no Java experience
    - JavaServer Pages (Bergsten), Java Server Pages App Dev (Forta)
- **Java books often focus too much on client-side Java (applets, Swing)**
  - Not as relevant to us interested in server-side java
  - But important to consider for their core java content
Learning More

• Many CFDJ articles
  – Especially current 8 part “Cold Cup O’ Joe” series by Guy Rish
  – Matching ColdFusion with Server-side Java, by Christian Schneider
  – 3 part “Java for CF’ers” series by Ben Forta
  – 6 part “Online Ticket Store” series by Ajit Sagar
    • 4 parts of which were in Java Dev. Journal

• Java Developers Journal
  – Many interesting ones
  – My bi-monthly Journeyman J2EE series
    • September 01, November 01, January 02

Learning More

• Excellent Macromedia JRun documentation
  – Download the Macromedia JRun 3-user demo
  – No time expiration

• Macromedia now offering several Java classes
  – Java for Web Developers (3 day)
  – FastTrack to JSP (2 day)
  – Building J2EE Apps
  – These all replace former ‘Servlets, Java and JSP’ class

• I’m planning a class on Integrating CF & Java
  – Will expand on many of today’s topics, walkthroughs, tips & tricks, more substantial applications
  – See last slide for contact info
Topics Covered Further in My Forthcoming Class

- My class would cover more, and w/ walkthroughs
  - Actual tools for integrating CF/Java
  - Setting up the environment/challenges
  - Solving nasty classpath problems, compile tricks
  - Some Java programming fundamentals
  - Finding, leveraging other libraries, classes
  - Trying different database integration alternatives
  - Integration with more products/services
    - XML/XSL, PDF, Graphing engines, Servlets etc.
  - Converting Java, Servlet code to COBJECT/CFX
  - Performance implications of calling objects
  - Solving nasty classpath problems, compile tricks
  - Introduction to Java programming language
  - Comparing Java to other object-oriented CF approaches (Spectra, FuseBox, CObjects)
  - More about Neo, tagfusion
  - And more

* Topics are tentative and subject to change

For Further Questions

- For follow-up questions
  - Or more information on Java or ColdFusion consulting, training, or presentations

- Please contact:
  Charlie Arehart, CTO, Systemanage
carehart@systemanage.com
(voice) 301 604 8399
http://www.systemanage.com

- Available for training, mentoring, short-term consulting (1-5 days):
  - All manner of CF topics, beginner-adv
    - Maximizing developer productivity
    - Best practices, performance tuning
  - J2EE (JSP/Servers/EJBs), Integrating CF/Java
  - Database design, Int/Adv SQL topics
  - Testing, Graphing, Source Code Control, and more
  - Wireless applications