

COLD FUSION Developer's Journal

ColdFusionJournal.com

2007 Volume:9 Issue: 5

CFImage

PART 1



Scorpio Becomes Beta 5

It Is Possible! 7

ColdFusion 8 Beta Is
Released by Adobe 26

A Complete Application
with RPC Communications
and JMS 28

Presorted
Standard
US Postage
PAID
St. Croix Press

REALWORLD
JAVA SEMINAR
THE LARGEST JAVA DEVELOPER
EVENT ON THE EAST COAST
REALWORLDJAVA.COM

Register Today!
Early Bird: Save \$100
www.realworldjava.com

A Roadmap
for Java Professionals

SEE PAGE 13





Say hello to the next generation.

It's found in the world's most inspirational places. It's pushing the boundaries of art direction and design. Introducing Adobe® Creative Suite® 2.3. Today's most talented designers and art directors are working smarter and faster because of it. Just ask the masterminds behind INTERspectacular—they rely on the Creative Suite to help bring their ideas into the world. See how they do it at adobe.com/creativemind. It's everything but the idea. Better by Adobe.™

Luis Blanco and Michael Uman,
INTERspectacular

adobe.com/creativemind

©2006 Adobe Systems Incorporated. All rights reserved. Adobe, the Adobe logo, Creative Suite and Better by Adobe are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries.

Your Web site ...



Your Web site Powered by Hot Banana



Hot Banana Web Sites Are Built Fully Loaded!

Why settle for building and managing a vanilla Web site when you can savor a full featured **Hot Banana** Web site?

Hot Banana is a fully-loaded Web content Management System with all the Web site optimization and eMarketing campaign tools your Marketing department craves. And, it's easy-to-use, search engine friendly and fine tuned for peak performance.

So what's the cherry on top? It's ColdFusion of course - you'll love how easy **Hot Banana** is to implement, integrate and customize.

Schedule your **Free** Taste Trial Today!

Contact Us Now:
hotbanana.com/cfdj-demo
1.866.296.1803
sales@hotbanana.com

Features...

- 100% browser-based
- Updated RTE
- Powerful DAM
- XHTML & CSS compliant
- Multilingual support
- Granular security
- Flexible workflow
- Press release manager
- Keyword analysis
- Web analytics center
- A/B Testing
- Powerful DAM
- Form builder
- RSS & blogs
- SEO tools
- Content reuse & scheduling
- Lead source logging & tracking
- Custom metadata
- Event registration
- Email manager



Web Content Management For Marketing

**editorial advisory board**

Jeremy Allaire, *founder emeritus, macromedia, inc.*
Charlie Arehart, *CTO, new atlanta communications*
Michael Dinowitz, *house of fusion, fusion authority*
Steve Drucker, *CEO, fig leaf software*
Ben Forta, *products, macromedia*
Hal Helms, *training, team macromedia*
Kevin Lynch, *chief software architect, macromedia*
Karl Moss, *principal software developer, macromedia*
Michael Smith, *president, teratech*
Bruce Van Horn, *president, netsite dynamics, LLC*

editorial**editor-in-chief**

Simon Horwith simon@sys-con.com

executive editor

Nancy Valentine nancy@sys-con.com

research editor

Bahadır Karuv, PhD bahadir@sys-con.com

production**lead designer**

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com
Tami Beatty tami@sys-con.com

editorial offices**SYS-CON MEDIA**

577 Chestnut Ridge Rd., Woodcliff Lake, NJ 07677
Telephone: 201 802-3000 Fax: 201 782-9638
COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)
is published monthly (12 times a year)
by SYS-CON Publications, Inc.

postmaster: send address changes to:

COLDFUSION DEVELOPER'S JOURNAL
SYS-CON MEDIA
577 Chestnut Ridge Rd., Woodcliff Lake, NJ 07677

©copyright

Copyright © 2007 by SYS-CON Publications, Inc.
All rights reserved. No part of this publication may
be reproduced or transmitted in any form or by any means,
electronic or mechanical, including photocopy
or any information, storage and retrieval system,
without written permission.

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ
FOR LIST RENTAL INFORMATION:
Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

For promotional reprints, contact reprint
coordinator Megan Mussa, megan@sys-con.com.
SYS-CON Publications, Inc., reserves the right to
revise, republish and authorize its readers to use
the articles submitted for publication.
All brand and product names used on these pages
are trade names, service marks, or trademarks
of their respective companies.

Scorpio Becomes Beta



By Roger Strukhoff

Roman
astrology
turned into
the Greek alphabet
recently, with the an-
nouncement that the

previously named "Scorpio" has been released
as the ColdFusion 8 Beta. Initial reaction from
the most sophisticated members of the CF
community was positive.

Gartner analyst Ray Valdes said, "the ColdFusion upgrade puts to rest questions about Adobe's commitment to the technology after acquiring Macromedia. People may have wondered if Adobe appreciated ColdFusion, and I think they do. I think it's part of their enterprise software strategy."

CF8 is, of course, the first version of ColdFusion to be branded with the Adobe label. Its general release is set for later this year. CF8, a server-side Java app, also has .NET support (something that's already been written about online at SYS-CON Media). This aspect of it means simply that "customers don't need to select one technology to the exclusion of the other," according to Adobe senior product marketing manager Tim Buntel. JBoss app server support is now provided as well for those who work within this open source technology acquired last year by Linux vendor Red Hat. CF8 can also generate and manipulate PDF docs.


Well aware of the growing client-side AJAX phenomenon, Adobe has also "leveraged" its own client-side Flex rich Internet application (RIA) technology into CF8 "for integration of complex environments with intuitive interfaces," according to Adobe.

ColdFusion guru Ben Forta noted the pres-

ence of an Eclipse-based debugger (the first interactive debugger in seven years), and has also blogged about this version's other Eclipse extensions, including RDS panels, a Services Browser panel, a CF Log Viewer, RDS support, help, and wizards. ColdFusion Developer's Journal editor-in-chief Simon Horwith has also noted all of the "cool" new features, and has said that there is "a nice performance boost over previous versions as well."

Simon also held forth recently about CF8 in an exclusive interview with SYS-CON.TV, which can be found online at www.sys-con.tv. He expressed real astonishment at how much faster CF8 was. He said that although he expected most of the new features, and had also expected some performance increase, he was really bowled over by how much faster CF8 was than previous versions. He also traced a brief history of CF, noting that there was trepidation among developers when the Allaire-born technology was first acquired by the much-larger company Macromedia, and how that trepidation only increased over the past year when the even-larger Adobe acquired Macromedia.

Simon also said any such fears were allayed, and that he and other developers to whom he has spoken were very happy with how Adobe has been treating these developers' favorite technology.

Enjoy! 

About the Author

Roger Strukhoff is Group Publisher and Editorial Director of SYS-CON Media. He spent 15 years with Miller Freeman Publications and The International Data Group (IDG), then co-founded CoverOne Media, a custom publishing agency that he sold in 2004. His work has won awards from the American Business Media, Western Press Association, Illinois Press Association, and the Magazine Publishers Association. You can read his blog at "rssblog.linux.SYS-CON.com" and contact him at [roger\(at\)sys-con.com](mailto:roger(at)sys-con.com).

roger@sys-con.com



Solution
PARTNER

INTERGRAL
information solutions

<cf_essentials>



Indispensable CFMX tools.

Meet us at cf^{fusion} united 07

June 27th - June 30th



FusionReactor

CFMX / J2EE Server, database
and application monitoring

from **\$249**

FusionDebug



Interactive Debugging
for ColdFusion MX

from **\$105**



www.fusion-reactor.com

Trademarks and Registered Trademarks are the property of their respective owners.

founder & ceo

Fuat Kircaali fuat@sys-con.com

group publisher

Roger Strukhoff roger@sys-con.com

advertising

senior vp, sales & marketing

Carmen Gonzalez carmen@sys-con.com

advertising sales director

Megan Mussa megan@sys-con.com

associate sales manager

Corinna Melcon corinna@sys-con.com

sys-con events

president, events

Carmen Gonzalez carmen@sys-con.com

events manager

Lauren Orsi lauren@sys-con.com

events associate

Sharmonique Shade sharmonique@sys-con.com

customer relations

circulation service coordinator

Edna Earle Russell edna@sys-con.com

Alicia Nolan alicia@sys-con.com

sys-con.com

vp, information systems

Bruno Decaudin bruno@sys-con.com

information systems consultant

Robert Diamond robert@sys-con.com

web designers

Stephen Kilmurray stephen@sys-con.com

Richard Walter richard@sys-con.com

accounting

financial analyst

Joan LaRose joan@sys-con.com

accounts payable

Betty White betty@sys-con.com

subscriptions

Subscribe@sys-con.com

Call 1-888-303-5282

For subscriptions and requests for bulk orders, please send your letters to Subscription Department

Cover Price: \$8.99/issue

Domestic: \$89.99/yr (12 issues)

Canada/Mexico: \$99.99/yr

All other countries \$129.99/yr

(U.S. Banks or Money Orders)

Back issues: \$12 U.S. \$15 all others

It Is Possible!

Simultaneous development with CF7 and CF8



By Jeff Chastain

Now that Scorpio is here (at least in beta), it is time to start figuring out all the new tricks

of the trade, right? However, as most of your clients will not be switching immediately, you will still have to be doing ColdFusion 7 work for some time.

How do you run both ColdFusion 7 to support your current clients and ColdFusion 8 to keep your development skills sharp? You could run one using the built-in Web server, but that's not really the same as your production systems. Instead, let's take a look at using the multiserver configuration option to run both CF7 and CF8 instances simultaneously and under your choice of Web server. The problem is, the system requirements for CF7 and CF8 are not compatible with each other, right? Yes and no...it's not quite out of the box, but it is possible to have complete working copies of each, running in complete parallel. The key is using the multiserver configuration and the JVM config.

ColdFusion Multiserver Configuration

For those who have never worked with the multiserver configuration (myself included until a few months ago), what is it? During the installation process for ColdFusion,

the installer prompts you to select a server configuration option (see Figure 1). There are three options: single server (default), multi-server, and J2EE deployment. The single server option allows you to install a single copy of ColdFusion on your development workstation and connect it to your Web server of choice. This was always the route I took and I expect many did as well.

The J2EE deployment option gives you the ability to set up ColdFusion as an EAR or WAR file that can then be deployed to a Java application server such as JRun or Tomcat. If this went straight over your head, you probably need to know more about Java servers and this option won't be a lot of use to you – who needs a Java server if you can use the single self-contained server option, right?

The second option – multiserver deployment – is the one we are going to look at here. This option installs a full copy of Adobe's JRun 4 application server and then installs an instance of ColdFusion as a Web application on that Java application server. Adobe has set this up so that you can manage the different server instances deployed (including ColdFusion) through the ColdFusion administrator and never have to log into the JRun administrator. This is very nice for those of us who don't want to or don't have the time to become Java application server experts.

What Multiple Instances Give You

There are many benefits to setting up ColdFusion in a multiserver configuration. By going this route, you can deploy multiple instances of ColdFusion on the same server – hence the “multi-server” configuration. There are many benefits from a production standpoint including adding additional security between applications in the form of

TABLE OF CONTENTS

CFImage

The new image functionality is just awesome!

Ben Nadel...14



5

editorial

Scorpio Becomes Beta

By Roger Strukhoff

7

feature

It Is Possible!

Simultaneous development with CF7 and CF8

By Jeff Chastain

24

cfugs

ColdFusion User Groups

26

cf8

ColdFusion 8 Beta Is Released by Adobe

The wait is over

By Derek Vaughan

28

book excerpt

A Complete Application with RPC Communications and JMS

Multi-Tier Applications Development with Flex

By Yakov Fain, Victor Rasputnis and Anatole Tartakovsky

sandboxes and fine-tuning the JVM settings to boost the performance of different applications.

From a developer's standpoint, especially on our own development workstations, a multi-instance deployment of ColdFusion also provides several benefits. First, we can configure different settings including ColdFusion mappings and data sources for different applications. Have you ever had two applications that use the same framework but different versions? Both require the same mapping to be set up, but the mapping points to different file system locations. With a multi-instance deployment of ColdFusion you can put each application in its own ColdFusion instance, as you can configure the mappings and any other settings uniquely for each application.

The second and more important benefit in this case is that you can set different instances of ColdFusion to run using different JVMs. This is critical when trying to set up both ColdFusion 7, which only supports up to JVM 1.4.2, and ColdFusion 8, which will support JVM 1.4.2 but is better suited and performs better on the latest JVM (1.6).

The Install

Let's walk through the process of setting up a ColdFusion development workstation with both ColdFusion 7 and ColdFusion 8 deployed side-by-side. First a bit of background as to how I am putting together these instructions: my workstation does not have ColdFusion installed at all, so I am starting clean. Also,

my workstation is running Windows Vista, so it already prefers the latest JVM, and, last, I am using IIS 7 for my Web server. This is not a big deal, although there are still a few issues regarding the JVM that we will walk through at that point.

Since I am running Windows Vista, I'm going to start with the ColdFusion 8 installer. You could easily do this process in reverse, but the ColdFusion 7 installer and Windows Vista don't play as nicely together as it is an "unsupported operating system." The first step is to run the ColdFusion 8 installer, agree to the license, and select the Developer Install (in my case). At this point, you will be prompted to select the installer configuration (see Figure 1).

For this install, select the multiserver configuration option. As mentioned previously, this will install a copy of Adobe's JRun 4 server, then install and deploy ColdFusion as a Web application on that server. By going this route, we can install an instance of ColdFusion 8 side-by-side with an instance of ColdFusion 7, which is our end goal.

At this point, the installation goes pretty much the same way as a single server configuration would. There are two things to take note of, however. When the installer asks for different path names (i.e., the CF Administrator location), remember that we will be installing both ColdFusion 7 and ColdFusion 8, which will utilize different CFIDE folders. My recommendation, and what has worked for me thus far, is instead of putting the CFIDE folder in c:\inetpub\wwwroot, put it in c:\inetpub\cfmx8root or c:\inetpub\cfmx7root. This way it is obvious later on which CFIDE folder belongs to which version.

The other thing to take note of is when the installer asks which Web site(s) to configure for use with ColdFusion. Again, in my case I am using IIS, so this may look a bit different based on your choice of Web server. With IIS, the installer gives you the option of installing to all Web sites, selecting one or more Web sites, or using the built-in Web server (see Figure 2).

On my first attempt, I selected "All IIS Websites" and this caused me quite a few headaches down the road. On my second attempt, I wised up and specified to only install and configure ColdFusion for the Default Web Site. By selecting this option, it's much easier to configure new Web sites down the road to communicate different instances of ColdFusion.

At this point, you should be good to go with the rest of the ColdFusion installation.

Your First ColdFusion Instance

When you log into the ColdFusion administrator after the installation, you should see a new menu item on the left-hand navigation menu called "Enterprise Manager" with a sub-link of

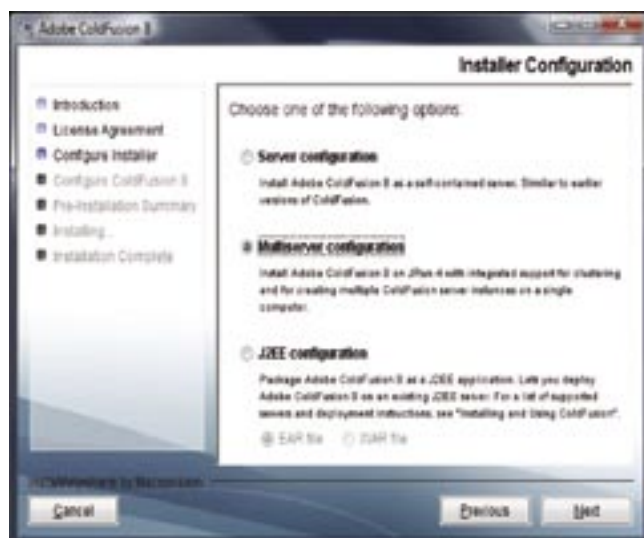


Figure 1

“Instance Manager” (see Figure 3).

This screen is where you will add and remove instances of ColdFusion without having to use the JRun management console. Let's take a quick moment to look at the process of adding a new ColdFusion instance. First, click on the “Instance Manager” link under “Enterprise Manager.” Then, click the “Add New Instance” button. You should see a screen similar to Figure 4 prompting you for information about the new instance/server.



Figure 2

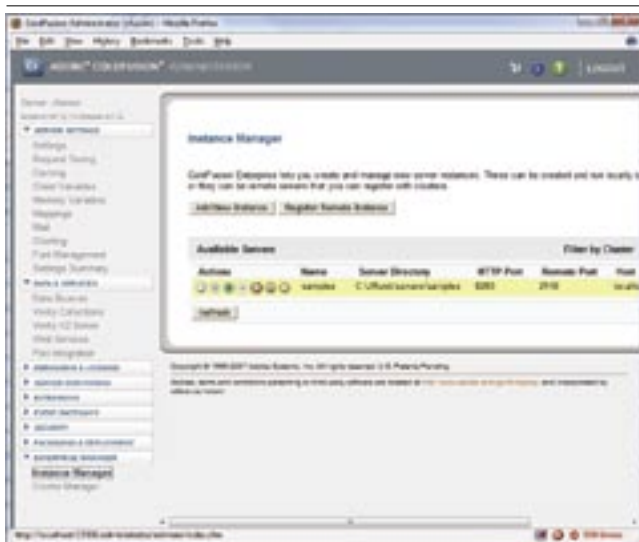


Figure 3

In this form, you will need to specify a name for the server – this is the JRun server and will only be needed for recognition purposes so that you know which instance you are working with later. The server directory should be auto-populated as you enter a name for the server and my recommendation is to leave this as is.

At this point, we have no EAR or WAR file to use to create this instance, so leave that field blank. Since I am installing ColdFusion 8 on Windows Vista, I can go ahead and check the two check boxes to have a Windows Service set up to automatically start the ColdFusion instance upon boot. If you are installing ColdFusion 7 first on Windows XP, this should work as well. Click the Submit button and after a brief wait you should have a new ColdFusion instance ready to go. The acknowledgment screen should list some details about your new instance and give you a link to the ColdFusion administrator for that instance – remember, this is a completely separate instance with its own ColdFusion administrator. The next step is to connect this instance up to our Web server of choice – IIS 7 in my case. Adobe has helped us out here by including the “Web Server Configuration Tool.” A shortcut to this tool should have been installed in your start menu. Selecting this option will show a command prompt window and a little Java window as seen in Figure 5.

To configure your Web server to work with your new ColdFusion instance, click the Add button. A new window will appear with a configuration form based upon your Web server setup. Using IIS, mine looks like Figure 6.

In the first section, you will want to select the JRun Server instance with the name you provided in the ColdFusion administrator when you created the new ColdFusion instance. In the second section, you'll want to select the Web site to which you wish to connect this JRun instance and then select the “Configure Web server...” check box in order to have all of the mappings, etc., set up. Last, click OK and the script should display some information in a command-prompt window before returning to the original Web server configuration window and showing the new ColdFusion instance. Just browse to the Web address associated with the selected Web site and you should be good to go. You have just created your first (officially it is your second instance as the default CF Administrator is running on an instance called “cfusion”) ColdFusion multi-instance server configuration.

Your Second ColdFusion Instance

Now that we have ColdFusion up and running in a multi-instance server configuration, it is time (in my case) to get ColdFusion 7 set up. With Windows Vista, it can be a bit tricky to get the installer working, but what you need to do is run it with full administrative rights; to do this, right-click on the installer icon

“At this point, you should have a working ColdFusion 8 instance side-by-side with a working ColdFusion 7 instance”

and select “Run as Administrator.” Give it a few seconds and the installer should start up. As with the ColdFusion 8 install, the portions of the installer to focus on are the installation configuration and the default path names.

As before, agree to the license and select the developer edition for the ColdFusion 7 installer. This time, since we already have a Java Web server installed as part of the ColdFusion 8 install, we want to select the J2EE installation configuration option as shown in Figure 7.

What this option does is have the installer “deploy” ColdFusion as an EAR file that the JRun server, which was installed with the previous ColdFusion 8 install, will accept and turn into a working ColdFusion 7 server.

Like the ColdFusion 8 install, I chose to set the base installation path for ColdFusion and any associated add-ons (search, etc.) to C:\CFusionMX7 to make sure there is a difference between the ColdFusion 8 and ColdFusion 7 installs. During this installation, a new screen will present itself requesting the “Context Root.” Since we are deploying this EAR to our local JRun server and wish it to behave as much as possible like the “standard” installation of ColdFusion, make sure this value is set to “/”. This will set up the ColdFusion instance to respond to the root localhost address for your development machine. From this point on, the installer should look pretty familiar.

Deploying ColdFusion 7 to JRun

After the ColdFusion 7 installer completes, you should find a set of files at the installation path you provided (C:\CFusionMX7\ in the example). In this folder, you should find a file named “cfusion.ear”. This is your ColdFusion 7 installation that now needs to be deployed to the previously installed JRun application server. To do this, we can utilize the same interface in the default ColdFusion administrator that we used to create a new instance of ColdFusion 8. Simply select the “Instance Manager” option under the “Enterprise Manager” option on the ColdFusion Administrator page as shown in Figure 8.

Specify a server name and path as you did before, but when you get to the EAR/WAR file field, insert the path to the “cfusion.ear” file that was generated during the ColdFusion 7 installation. By doing this, the instance manager will deploy the ColdFusion 7 instance to the JRun server instead of the default ColdFusion 8 instance.

Last, for a ColdFusion 7 instance, you do not want to select the check box to have the instance manager create an NT service for this instance. We will do that manually as there are a few things that need changing in order to get ColdFusion 7 to start properly. Now, click the Submit button to add the new ColdFusion instance and we’ll almost be finished. This will be the same process that is followed every time a new instance of ColdFusion 7 is deployed, using the same cfusion.ear file each time.

Last, the JVM

JVM is short for Java Virtual Machine and is one of those little Java components that is necessary for ColdFusion or most any Java application to run. The problem we have is that by installing ColdFusion 8 first, any other application set up on that same JRun server will by default use JVM version 1.6. While this is perfect for ColdFusion 8, ColdFusion 7 does not support a JVM version above 1.4.2.

To get around this, first download and install JVM 1.4.2 separately from the Java Web site (<http://java.sun.com/j2se/1.4.2/download.html>). In this case, you’ll need to install the full SDK and not just the JRE as ColdFusion relies on server components of the library that are not part of the JRE. Once the 1.4.2 version of the JVM is installed, we can use it to run the ColdFusion 7 instances.

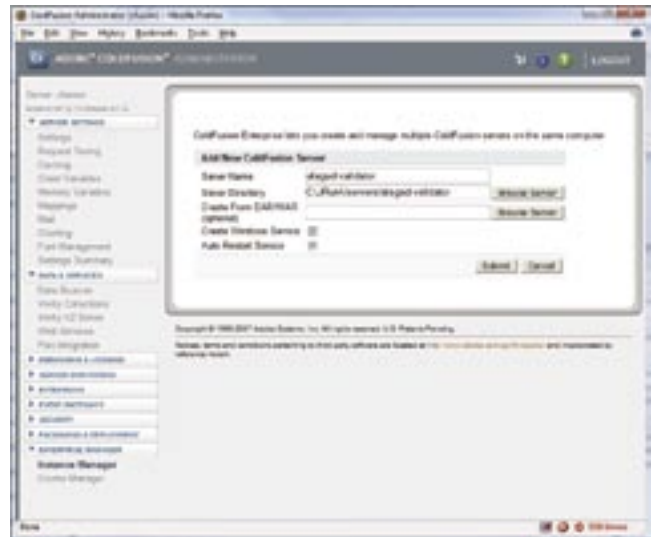


Figure 4

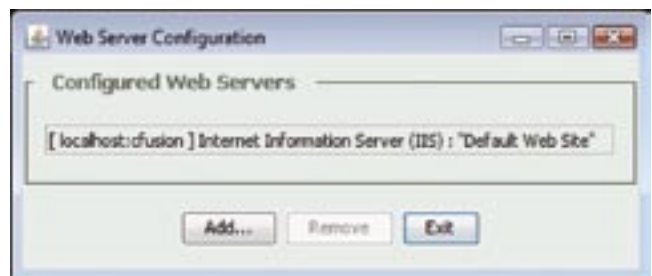


Figure 5



Figure 6

To set up JRun to utilize the different JVM, navigate to `c:\jrun4\bin` and find the `jvm.config` file. Make a copy of this file (`jvm_cfm7.config`) and then open it in your favorite text editor. At the top of the file, there should be a line that starts as “java. home”. This line should be changed to read “java. home=`c:/j2sdk1.4.2_14/jre`” with the path pointing to the location where the 1.4.2 version of the JVM was installed. Save the file and then we need to use this file to configure the new ColdFusion 7 instance.

Remember, during the process of adding the new instance, the check box to create the Windows Service was left blank. This is intentional as by default each instance will use the version 1.6 JVM. However, Adobe has included a command-line utility to create a Windows



Service and, using that utility, we can instruct the service to use the new `jvm_cfm7.config` file and settings that we just created. The command-line utility string should look something like this.

```
c:\jrun4\bin\jrunsvc.exe -install [JRun Server Name] "Adobe ColdFusion 7 AS [JRun Server Name]" "Adobe ColdFusion 7 AS [JRun Server Name]" -config [JVM Config File Name]
```

This command will create a new Windows Service that upon system boot will start the JRun server specified and use as its configuration the customized JVM config (`jvm_cfm7.config`) file created earlier.


The last step to getting the ColdFusion 7 instance working is to set it up with your chosen Web server. There is a problem here because the Web Site Configuration Tool may work or may not again because of the JVM version discrepancy between ColdFusion 7 and ColdFusion 8. There is another command-line script that will take care of the configuration automatically and could easily be bundled with the above script for setting up the Windows service.

```
c:\j2sdk1.4.2_14\jre\bin\java -jar c:\jrun4\lib\wsconfig.jar -server "[JRun Server Name]" -ws iis -site "[IIS Web Site Name]" -coldfusion -v
```

This command, launched using the 1.4.2 JVM version, will add the Web site connectors to the IIS Web site with the specified name, attach it to the JRun server with the specified name, and set it up for ColdFusion 7.

The Ultimate Development Workstation

At this point, you should have a working ColdFusion 8 instance side-by-side with a working ColdFusion 7 instance. If you need additional instances, they should be very simple to set up. Additional ColdFusion 8 instances are created using the default ColdFusion administrator and the Web Site Configuration Tool. Additional ColdFusion 7 instances are created using the default ColdFusion administrator, the `cfusion.ear` file, and the two command-line scripts mentioned earlier.

Hopefully, this will allow you to benefit from some of my hours of frustration and allow you to have a dual-CF development box that is easy to use and learn on at the same time. 

About the Author

Jeff Chastain has been developing software applications using object-oriented programming for over 12 years and has been developing Web applications in ColdFusion for over 8 years. He has experience in a variety of industries, from Fortune 500 companies to his own consulting practice. Currently, Jeff is an applications architect and systems developer for Alagad, Inc., and contributes to the blog at <http://www.doughughes.net>.

jchastain@alagad.com



Figure 7

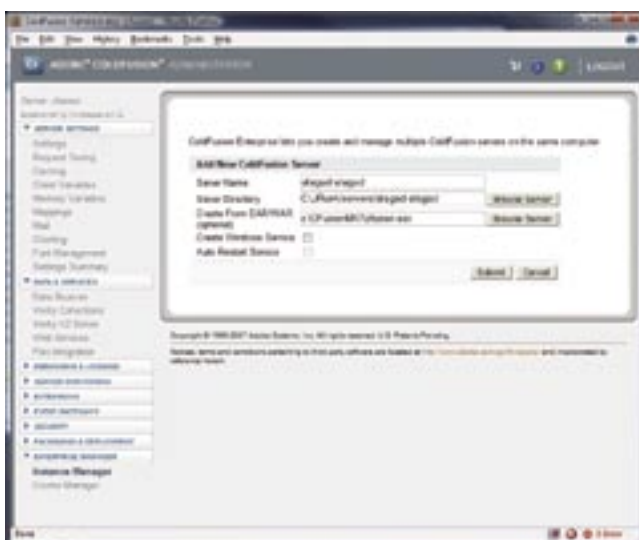


Figure 8

REALWORLD
JAVA
THE LARGEST JAVA DEVELOPER
EVENT ON THE EAST COAST

SEMINAR
REALWORLDJAVA.COM

Register Today!
Early Bird: Save \$100
www.realworldjava.com

A Roadmap for Java Professionals



August 13, 2007


The Roosevelt Hotel
The Roosevelt Hotel
New York City

Hands-On Education: The Largest Java Developer Event on the East Coast

A seasoned Java professional has to know more than just the syntax of the Java language. Java EE offers a set of standardized technologies for enterprise development. A number of open-source frameworks such as Spring or Hibernate are widely used in a variety of Java applications. Familiarity with new "beyond-Java" languages and technologies will widen your horizons and make you a more valuable Java professional.

— Sponsored by —

No. 1 Technology Magazine in the World



— Produced by —



For this and other great events
visit www.EVENTS.SYS-CON.com

LEARN FROM THE MASTERS:

- | | | |
|---|---|--|
| > Java 6.0 - New Features | > Code Quality:
Pay Now or Pay Later | > XML Processing in Java |
| > EJB 3, Spring and Hibernate:
A Comparative Analysis and
Recommendations | > EJB 3. and
Java Persistence API | > Programming with
Spring Framework |
| > Adobe Flex for Java Developers | > Concurrent Programming
in Java | > AJAX for Java Developers |
| > Enterprise Service Bus as
a Centerpiece of SOA
Implemented with Java | | > Ruby on Rails for
Java Developers |

CFImage

**The new image functionality
is just awesome!**

Part I



By Ben Nadel

Finally, ColdFusion 8 has incorporated image manipulation directly into the ColdFusion tag and scripting language. No more are we, as developers, chained to third-party products.

These products, while excellent in quality, just mean adding one more level of complexity to any application that requires server-side image manipulation. Now, with CFImage and an abundance of image-related functions, ColdFusion has done to image manipulation what it has done to almost every other aspect of Web applications development – it has made it simple.

Due to the wide array of image functionality in Cold-

Fusion 8, I am going to try and break this tutorial up into several small and manageable parts. The first part will cover basic reading and writing of images using CFImage and the related image manipulation functions. But, before we get into that, let's just quickly touch upon these new features as a whole.

ColdFusion 8 has given us CFImage. CFImage provides us with tag-based access to only a small subset of the image functionality including:

- Retrieving information about an image
- Reading an image into memory
- Resizing an image
- Rotating an image
- Adding a border to an image
- Converting an image from one file format to another
- Creating a CAPTCHA image
- Writing an image to a file
- Writing an image to the browser

In addition to the CFImage tag, ColdFusion has introduced

dozens of new image manipulation functions including:

- `ImageAddBorder()`
- `ImageBlur()`
- `ImageClearRect()`
- `ImageCopy()`
- `ImageCrop()`
- `ImageDrawArc()`
- `ImageDrawBeveledRect()`
- `ImageDrawCubicCurve()`
- `ImageDrawLine()`
- `ImageDrawLines()`
- `ImageDrawOval()`
- `ImageDrawPoint()`
- `ImageDrawQuadraticCurve()`
- `ImageDrawRect()`
- `ImageGetEXIFTag()`
- `ImageGetHeight()`
- `ImageGetIPTCTag()`
- `ImageGetWidth()`
- `ImageGrayscale()`
- `ImageInfo()`
- `ImageNegative()`
- `ImageNew()`
- `ImageOverlay()`
- `ImagePaste()`
- `ImageRead()`
- `ImageReadBase64()`
- `ImageResize()`
- `ImageRotate()`
- `ImageRotateDrawingAxis()`
- `ImageScaleToFit()`
- `ImageSetAntialiasing()`

The ColdFusion 8 CFImage tag and the image-related functions all deal with a new ColdFusion object:

`coldfusion.image.Image`

Most of you are not going to care about this one all that much, but for those of you who are interested in the underlying Java methods of the `coldfusion.image.Image` object, I have listed them below (skip past this if you have no idea what I'm talking about). Please note that any underlying methods of the actual Java/ColdFusion objects are found by gathering object metadata and through reflection; none of these methods are documented or officially supported. If you choose to use them, you do so at your own risk and discretion.

- **`addBorder`**(`int`, `java.lang.String`, `java.lang.String`) – returns: `void`
- **`blur`**(`int`) – returns: `void`
- **`brighten`**() – returns: `void`
- **`clearRect`**(`int`, `int`, `int`, `int`) – returns: `void`
- **`copyArea`**(`int`, `int`, `int`, `int`) – returns: `coldfusion.image.Image`
- **`copyArea`**(`int`, `int`, `int`, `int`, `int`, `int`) – returns: `coldfusion.image.Image`
- **`crop`**(`float`, `float`, `float`, `float`) – returns: `void`
- **`draw3DRect`**(`int`, `int`, `int`, `int`, `boolean`, `boolean`) – returns: `void`
- **`drawArc`**(`int`, `int`, `int`, `int`, `int`, `int`, `boolean`) – returns: `void`
- **`drawCubicCurve`**(`double`, `double`, `double`, `double`, `double`, `double`, `double`, `double`, `double`, `double`) – returns: `void`
- **`drawLine`**(`int`, `int`, `int`, `int`) – returns: `void`
- **`drawLines`**([`I`], [`I`], `boolean`, `boolean`) – returns: `void`
- **`drawOval`**(`int`, `int`, `int`, `int`, `boolean`) – returns: `void`
- **`drawPoint`**(`int`, `int`) – returns: `void`
- **`drawQuadraticCurve`**(`double`, `double`, `double`, `double`, `double`, `double`) – returns: `void`
- **`drawRect`**(`int`, `int`, `int`, `int`, `boolean`) – returns: `void`
- **`drawRoundRect`**(`int`, `int`, `int`, `int`, `int`, `int`, `boolean`) – returns: `void`
- **`drawString`**(`java.lang.String`, `int`, `int`, `coldfusion.runtime.Struct`) – returns: `void`
- **`flip`**(`java.lang.String`) – returns: `void`
- **`getBase64String`**(`java.lang.String`) – returns: `java.lang.String`
- **`getClass`**() – returns: `java.lang.Class`
- **`getColor`**(`java.lang.String`) – returns: `java.awt.Color`
- **`getCurrentGraphics`**() – returns: `java.awt.Graphics2D`
- **`getCurrentImage`**() – returns: `java.awt.image.BufferedImage`
- **`getExifMetadata`**(`javax.servlet.jsp.PageContext`) – returns: `coldfusion.runtime.Struct`
- **`getExifTag`**(`java.lang.String`, `javax.servlet.jsp.PageContext`) – returns: `java.lang.String`
- **`getHeight`**() – returns: `int`
- **`getImageBytes`**(`java.lang.String`) – returns: [`B`]
- **`getIptcMetadata`**(`javax.servlet.jsp.PageContext`) – returns: `coldfusion.runtime.Struct`
- **`getIptcTag`**(`java.lang.String`, `javax.servlet.jsp.PageContext`) – returns: `java.lang.String`
- **`getSource`**() – returns: `java.lang.String`
- **`getWidth`**() – returns: `int`
- **`grayscale`**() – returns: `void`
- **`info`**() – returns: `coldfusion.runtime.Struct`
- **`initializeMetadata`**(`javax.servlet.jsp.PageContext`) – returns: `void`
- **`invert`**() – returns: `void`
- **`overlay`**(`coldfusion.image.Image`) – returns: `void`
- **`paste`**(`coldfusion.image.Image`, `int`, `int`) – returns: `void`
- **`readBase64`**(`java.lang.String`) – returns: `void`
- **`resize`**(`java.lang.String`, `java.lang.String`, `java.lang.String`) – returns: `void`
- **`resize`**(`java.lang.String`, `java.lang.String`, `java.lang.String`, `double`) – returns: `void`
- **`rotate`**(`float`, `float`, `float`, `java.lang.String`) – returns: `void`
- **`rotateAxis`**(`double`) – returns: `void`
- **`rotateAxis`**(`double`, `double`, `double`) – returns: `void`
- **`scaleToFit`**(`int`) – returns: `void`
- **`scaleToFit`**(`java.lang.String`, `java.lang.String`, `java.lang.String`) – returns: `void`
- **`scaleToFit`**(`java.lang.String`, `java.lang.String`, `java.lang.String`, `double`) – returns: `void`
- **`setAntiAliasing`**(`java.lang.String`) – returns: `void`
- **`setBackground`**(`java.lang.String`) – returns: `void`
- **`setColor`**(`java.lang.String`) – returns: `void`
- **`setDrawingStroke`**(`coldfusion.runtime.Struct`) – returns: `void`
- **`setDrawingStroke`**(`float`, `int`, `int`, `float`, [`F`], `float`) – returns: `void`

- **setRenderingHint**(java.awt.RenderingHints\$Key, java.lang.Object) – returns: void
- **setTransparency**(double) – returns: void
- **setXorMode**(java.lang.String) – returns: void
- **sharpen**(float) – returns: void
- **sharpenEdge**() – returns: void
- **shear**(float, java.lang.String, java.lang.String) – returns: void
- **shearAxis**(double, double) – returns: void
- **translate**(int, int, java.lang.String) – returns: void
- **translateAxis**(int, int) – returns: void
- **write**(java.lang.String, float) – returns: void
- **writeBase64**(java.lang.String, java.lang.String, boolean) – returns: void

For those of you who took a look at the underlying Java methods, you can see that there's a lot of stuff wrapped up in these image objects. The ColdFusion 8 CFImage tag and the related image manipulation functions probably provide some sort of facade that wraps around the underlying methods. For more information about what these do (as they are undocumented), try exploring the java.awt library. Ben Forta stated at the New York ColdFusion Users group that Adobe has built the ColdFusion image manipulation around this library, but that they have also greatly extended it to work around the limits that this library holds.

One of the most exciting features of the new ColdFusion 8 image manipulation is the large number of file formats that can be read in and written out. To give you insight into what is available, ColdFusion 8 has provided two methods: `GetReadableImageFormats()` and `GetWriteableImageFormats()`. These return comma-delimited lists of the file types that ColdFusion can deal with.

Calling `GetReadableImageFormats()` returns:

BMP, GIF, JFIF, JPEG, JPEG 2000, JPEG-LOSSLESS, JPEG-LS, JPEG2000, JPG, PNG, PNM, RAW, TIF, TIFF, WBMP

CFImage : Info Read - struct	
colormodel	CFImage : Info Read - struct
alpha_channel_support	NO
alpha_pre-multiplied	NO
bits_component_1	8
bits_component_2	8
bits_component_3	8
colormodel_type	ComponentColorModel
colorspace	Any of the family of RGB color spaces
num_color_components	3
num_components	3
pixel_size	24
transparency	OPAQUE
height	500
source	G:\Server\kinky_solutions\site_v1\testing\cf8\lady.jpg
width	236

Figure 1

Calling `GetWriteableImageFormats()` returns:

BMP, GIF, JFIF, JPEG, JPEG 2000, JPEG-LOSSLESS, JPEG-LS, JPEG2000, JPG, PNG, PNM, RAW, TIF, TIFF, WBMP

Notice that ColdFusion 8 can handle GIF images! Super sweet! I don't know how they handled that one (since I thought Prodigy owned the patent on that or something silly), but it's good to see that we have a huge variety of image formats that we can read and write to.

Now that we have a general overview of what kind of new and exciting image manipulation features are available in ColdFusion 8 and how many image types can be utilized, let's get into the meat of this part of the tutorial: reading and writing images. To start with, let's examine reading images using the CFImage tag. When reading in an image, the source of the image can be any one of the following:

- Absolute path name
- Path name relative to the Web root
- URL
- CFImage variable
- BLOB / Byte array
- Base64 encoded image data
- Binary object

Did anyone else see "URL" and get mentally turned on? 'Cause I did...but, let's not get ahead of ourselves. I want to quickly explore these different source type values and, to do so, I am going to use the INFO action of the CFImage tag. In the first example we are using the CFImage tag in conjunction with an absolute path name to the image.

```
<!--
Read in the image file. This will read in the binary
image into an object of coldfusion.image.Image.
-->
<cfimage
action="INFO"
source="#ExpandPath( './lady.jpg' )#"
structname="objImageInfo"
/>

<!--
The CFImage tag above stores the image information
into a struct, objImageInfo. Dump out this struct.
-->
<cfdump
var="#objImageInfo#"
label="CFImage : Info Read"
/>
```

This reads color information, source value, and dimensions into a struct. Running the above code, we get the following CFDump output shown in Figure 1.

The other input types give a similar output, so I won't show the CFDump, however, I will walk through the code. The next one on the list is the Web-relative path:



Visit the *New* **www.SYS-CON.com** Website Today!

The World's Leading *i*-Technology
News and Information Source

24/7

FREE NEWSLETTERS

Stay ahead of the i-Technology curve with
E-mail updates on what's happening in your industry

SYS-CON.TV

Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!

Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST

Streaming video on today's i-Technology news, events, and webinars

EDUCATION

The world's leading online i-Technology university

RESEARCH

i-Technology data "and" analysis for business decision-makers

MAGAZINES

View the current issue and past archives of your favorite i-Technology journal

INTERNATIONAL SITES

Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

<i>IT Solutions Guide</i>	<i>MX Developer's Journal</i>
<i>Information Storage+Security Journal</i>	<i>ColdFusion Developer's Journal</i>
<i>JDJ</i>	<i>XML Journal</i>
<i>Web Services Journal</i>	<i>Wireless Business & Technology</i>
<i>.NET Developer's Journal</i>	<i>Symbian Developer's Journal</i>
<i>LinuxWorld Magazine</i>	<i>WebSphere Journal</i>
<i>Linux Business News</i>	<i>WLDJ</i>
<i>Eclipse Developer's Journal</i>	<i>PowerBuilder Developer's Journal</i>


```
<!-- Read in the image using a web-relative path. -->
<cfimage
action="INFO"
source="/lady.jpg"
structname="objImageInfo"
/>
```

This gives us the same output, but the source attribute is `"/lady.jpg"`.

The next input type is a URL (sha-wing!):

```
<!-- Read in the image using a URL. -->
<cfimage
action="INFO"
source="http://localhost/testing/cf8/lady.jpg"
structname="objImageInfo"
/>
```

This gives us the same output, but the source attribute is `"http://localhost/testing/cf8/lady.jpg"`. When reading from a URL-based image, you have to be careful about the URL format. It's not just important that the resultant data is image data, it's also important that the last thing in the URL be the file name. For example, using a URL with this query string:

```
<!-- Read in the image using a URL. -->
<cfimage
action="INFO"
source="http://localhost/testing/cf8/lady.jpg?referrer=bennadel"
structname="objImageInfo"
/>
```

...will throw the ColdFusion error:

The com image format is not supported on this operating system. Use GetReadableImageFormats() and GetWriteableImageFormats() to see which image formats are supported.

The problem here is that ColdFusion is not recognizing the query string of the URL. This also means that you can't grab URLs that are served by ColdFusion using CFContent. This will only work to direct image URL access. This seems like a bug to me as I would have assumed it was just doing a binary grab via CFHttp (or something like that), but I guess there is more to it than I can understand. I would rather it try to grab the resultant data and throw an exception if the read fails.

Of course, there is a hack to get around this: you can add the image format to the query string:

<http://localhost/testing/cf8/lady.jpg?referrer=bennadel&type=.jpg>

This will work quite nicely. You get the idea here that ColdFusion is just doing some sort of ListLast() type functionality on the URL to get the file extension. A bit of a hack, but it works if you need it to.

The next input type is a ColdFusion image object. We don't have to perform actions just on new images; we can perform them on existing image objects:

```
<!-- Read the image into ColdFusion memory. -->
<cfimage
action="READ"
source="#ExpandPath( './lady.jpg' )#"
name="objImage"
/>
```

```
<!-- Get the image info from the existing image. -->
<cfimage
action="INFO"
source="#objImage#"
structname="objImageInfo"
/>
```

The next input type is the BLOB (Binary Large Object Bitmap) and the byte array. I am grouping these two together because I am sure what the difference really is. To me, they are both binary objects (I think). I don't generally deal with these sorts of objects:

```
<!-- Read in the binary image data. -->
<cffile
action="READBINARY"
file="#ExpandPath( './lady.jpg' )#"
variable="binImage"
/>
```

```
<!-- Read the binary directly into an image object. -->
<cfimage
action="INFO"
source="#binImage#"
structname="objImageInfo"
/>
```

When you read in an image in this fashion, the source attribute is empty.

The next input type is a Base64 encoded string. The CFImage tag has the boolean attribute, `IsBase64`, which flags the input as being Base64 data. This doesn't mean the target file is a Base64 data text file – this means the actual input value is Base64 image data:

```
<!-- Read in the binary image data. -->
<cffile
action="READBINARY"
file="#ExpandPath( './lady.jpg' )#"
variable="binImage"
/>
```

```
<!--
Read in the Base64 image data. To get Base64 image
data, we can convert the binary read we did above.
-->
<cfimage
action="INFO"
source="#ToBase64( binImage )#"
structname="objImageInfo"
isbase64="true"
/>
```

Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



- Exclusive feature articles
- Latest CFDJ product reviews
- Interviews with the hottest names in ColdFusion
- Code examples you can use in your applications
- CFDJ tips and techniques

That's a savings of \$29.89 off the annual newsstand rate. Visit our site at www.sys-con.com/coldfusion or call 1-800-303-5282 and subscribe today!

ColdFusion Developer's Journal





I don't deal with Base64 encoded images all that often, but this seems like something that could be very useful when put into clever hands. When you read in an image in this fashion, the source attribute is empty. The Base64 data does not require headers to be used.

For all the above "Read" demos, we were really just getting the info about the image. If you want to actually read the image into a ColdFusion variable (in the form of the coldfusion.image.Image data type), use the action "READ":

```
<!--- Read image into memory. --->
<cfimage
action="READ"
source="#ExpandPath( './lady.jpg' )#"
name="objImage"
/>
```

While this reads the image into memory, if you CFDump it out, you'll get the same output as if you were dumping out the INFO action result.

Now that we have covered the tag-based reading, let's examine reading in images using the new ColdFusion 8 image functions. To start with, let's look at ImageRead(). Like the CFImage tag, the ImageRead() function can take a variety of source types:

- Absolute path name
- Web relative path name
- URL

The ImageRead() function takes this as the only argument:

```
<!--- Read in the image using an absolute path. --->
<cfset objImage = ImageRead(
ExpandPath( "./lady.jpg" )
) >

<!--- Read in the image using a web-relative path. --->
<cfset objImage = ImageRead(
"./lady.jpg"
) >

<!--- Read in the image using a URL. --->
<cfset objImage = ImageRead(
"http://localhost/testing/cf8/lady.jpg"
) />
```

Unlike the CFImage tag, ImageRead() does not have a boolean flag for reading in Base64 data. When it comes to functions, if you want to read in Base64 image data, you have to use the ImageReadBase64() function:

```
<!---
Read in the binary image data. This will read
the data into a binary object, NOT the ColdFusion
image object.
--->
<cfset
action="READBINARY"
file="#ExpandPath( 'lady.jpg' )#"
variable="binImage"
```

```
/>

<!---
Read in the image by converting the binary image
data to a Base64 encoding.
--->
<cfset objImage = ImageReadBase64(
ToBase64( binImage )
) />
```

In addition to these straightforward methods, ColdFusion 8 can also use the ImageNew() function to read in images. ImageNew() has other features (optional arguments), but for our read/write tutorial, let's just concentrate on the first argument – the image source. Like the methods and tags above, ImageNew() can also take a variety of source types:

- Absolute path name.
- Web relative path name.
- URL.
- ColdFusion image variable.
- A BLOB / byte array.
- A Java buffered images.

We have looked at most of these types above. The only one here that stands out is the Java buffered image. This is the object that the ColdFusion image is wrapped around and is exposed through the function ImageGetBufferedImage(). To demonstrate this, we can grab the buffered image of one image object and use it to instantiate a new image:

```
<!--- Read in the image. --->
<cfset objImage = ImageNew(
"./lady.jpg"
) />

<!---
Using the buffered image data from the
first image, create a new image.
--->
<cfset objImage2 = ImageNew(
ImageGetBufferedImage( objImage )
) />
```

This kind of functionality can be super useful if you are interfacing with another Java component that handles image manipulation using the java.awt library and all you have access to is the underlying buffered image.

That just about wraps up reading in image files. Let's quickly cover writing images. As with reading in images, writing them can be done by using the CFImage tag as well as the image functions. The CFImage tag makes writing files especially easy since the source attribute is so flexible. Here, we are going to read an image from a URL and save it to disk:

```
<!---
Grab the image from the give source URL and save
it to disk (at the relative web path).
--->
<cfimage
```




Register Now!
**EARLY-BIRD
SAVINGS!**
\$200

ROOSEVELT HOTEL
NEWYORK CITY

June 25-27, 2007

Virtualization: Solutions for the Enterprise.

Delivering The #1 i-Technology Educational and Networking Event of the Year!

As today's CIOs and IT managers rise to the challenge of using their enterprise computing infrastructure as cost-effectively as possible while remaining responsive in supporting new business initiatives and flexible in adapting to organizational changes, Virtualization has become more and more important.

A fundamental technological innovation that allows skilled IT managers to deploy creative solutions to such business challenges, Virtualization is a methodology whose time has come. The fast-emerging age of Grid Computing is enabling the virtualization of distributed computing, of IT resources such as storage, bandwidth, and CPU cycles.

But Virtualization can also apply to a range of system layers, including hardware-level virtualization, operating system level virtualization, and high-level language virtual machines.

Register Online!

www.VirtualizationConference.com



THE FIRST MAJOR VIRTUALIZATION EVENT IN THE WORLD!

**HURRY, FOR EXHIBITOR AND
SPONSORSHIP OPPORTUNITIES
CALL 201-802-3020**

*Learn from
the Experts...*

— BROUGHT TO YOU BY —



```

action="WRITE"
source="http://farm1.static.flickr.com/240/458712628_2d6eff71e2.jpg"
destination="funny.gif"
overwrite="true"
/>

```

Notice that the destination path is not an absolute server path – it's a path relative to the current Web page. Also notice that the destination image was a GIF file format. This will automatically convert the image from the JPG that was served at the URL to the GIF image format that we save onto the disk. By default ColdFusion will throw an error if there are naming conflicts. To overcome this, you can set the Overwrite attribute to true (defaults to false) – this will overwrite any existing files of the same name. Now, not only is it super easy to grab images, it's super easy to convert image types.

We used the WRITE action here to grab the image and convert it to a new file format. You can also use the CONVERT action to accomplish just about the same thing. The only real difference (that I can see) between using a straight up WRITE versus CONVERT is that the CONVERT action allows you to also save the image data into a ColdFusion image object:

```

<!--
Grab the image from the give source URL and save
it to disk (at the relative web path).
-->
<cfimage
action="CONVERT"
source="http://farm1.static.flickr.com/240/458712628_
2d6eff71e2.jpg"
destination="funny.gif"
overwrite="true"
name="objImage"
/>

<!-- Write the image to the browser. -->
<cfimage
action="WRITETOBROWSER"
source="#objImage#"
format="jpg"
/>

```

Here, not only are we converting the image from a JPG format to a GIF image format, we are also storing the image data into the ColdFusion 8 image object, objImage. This also demonstrates our next mode of image writing: writing the image directly to the browser.

Notice that to do this, I am specifying the source of the image (the image object we created) and the output format of JPG. The format attribute can be PNG, JPG, or JPEG but it defaults to PNG. Now, when I first saw this action, I had assumed it was working the same way CFContent worked – by streaming the file to the browser as the only returned content. However, WriteToBrowser actually returns the image inline to the page.

If you look at the source of the page with the rendered inline image, you will see something like this:

```

```

If I run that page a few times, I get a variety of different source values:

```






```

ColdFusion is actually writing your file to some sort of temporary image storage and then serving it up the way any other image or file would be served up. But what is CFFileServlet? If I look in the root of my ColdFusion 8 test account, there is no such directory. This is some sort of public mapping, but realize this – this is not a ColdFusion request; this is an image file request. Image file requests go the Web server, not the ColdFusion application server. I assume this means that in order for this to work, IIS (or which ever Web server you use) must have a mapping for CFFileServlet to some ColdFusion directory. This makes me nervous. I think it's an awesome feature, but I'm not sure I like having to rely on mappings and tying in with settings external to the ColdFusion application server.

On the flip side, however, I do like this for the very reason that the image request is not going to the ColdFusion server. Serving up images via ColdFusion's CFContent tag is relatively slow and puts a drain on the ColdFusion resources. Storing an image to a temp directly and then serving it using the Web server is going to be 10 times more efficient.

I would like to know more about how CFFileServlet works and specifically how often that directory is cleaned out. I don't want to start writing a lot of images to the browser only to find out that it is clogging up this temp directory. Unfortunately, this directory is not discussed in the ColdFusion 8 CFML Reference manual.

Images can also be written to disk after a variety of CFImage image manipulation actions, but for now, we are trying to stick to just straight up read/write, not manipulation.

When it comes to writing images using

ColdFusion 8 image functions, there are basically two options: ImageWrite() and ImageWriteBase64(). ImageWrite() take three arguments (potentially):

- **Name:** The name of the ColdFusion image object you are going to write.
- **Destination:** The absolute or relative path name to which you are going to write the image. This argument is optional if the Source attribute of your image is available. For instance, if you read in the image locally via CFImage and then called ImageWrite() without a destination, it would overwrite the original image based on the internal Source. However, if you grabbed the image from a URL via CFImage, you must include a destination value with ImageWrite() as you cannot use the Source value (external URL) to store the image.
- **Quality:** A value between 0 and 1 that can be used to set the JPG quality.



Reading an image from a URL and writing it to disk could be done this way:

```
<!-- Grab the image from the give source URL. --->
<cfimage
action="READ"
source="http://farm1.static.flickr.com/183/392739661_70619076b7.jpg"
name="objImage"
/>

<!-- Write the image to the disk. --->
<cfset ImageWrite(
objImage,
"./waterfall.jpg",
.75
) />
```

As you can see, since we are grabbing the image via a URL, we must include the destination value. I have also included the JPG quality value, but this was not necessary as that value defaults to .75.

ImageWriteBase64() writes ColdFusion images to text files using a Base64 encoding. It takes four arguments:

- **Name:** The name of the ColdFusion image object that we are writing to disk.
- **Destination:** Since the destination here is not an image but a text file, the destination is required (the internal Source of the image will not help us).
- **Format:** This is the format of the image that we are going to encode. This is required, otherwise ColdFusion won't know which headers and conversion algorithm to include.
- **InHTMLFormat:** This specifies how the data is going to be written to the text file. If you specify false (default) or omit the argument, the Base64 is written to the file without any headers. If you specify true, the Base64 encoding is written to the disk with HTML-style headers.

Here, we can modify our previous example to read from a URL and store to disk as a Base64-encoded JPG image:

```
<!-- Grab the image from the give source URL. --->
<cfimage
action="READ"
source="http://farm1.static.flickr.com/183/392739661_70619076b7.jpg"
name="objImage"
/>

<!-- Write the image to the browser. --->
<cfset ImageWriteBase64(
objImage,
"./waterfall.txt",
"JPG"
) />
```

Just a final note on paths; the Web-relative path (e.g., /lady.jpg) is relative to the currently executing Web page, not to the currently executing ColdFusion template. Therefore, if you are in an included template, you might get files stored in unexpected places if you don't fully grasp this concept.

Well, that's it. That's the quick(ish) overview on how you can read and write image files using ColdFusion 8's new CFImage tag and accompanying image functions. It's awesome that there is such a variety of input and output methodologies. Sorry that this went longer than intended, but this introduction only scratched the surface. ColdFusion's new image functionality is just plain awesome.

• • •

This article was reprinted with permission from Ben Nadel's blog: <http://www.bennadel.com/index.cfm?dax=blog:766.view>.

About the Author

Ben Nadel has worked with ColdFusion for eight years and is a super ColdFusion enthusiast. He blogs regularly about all aspects of Web development on his personal site, <http://www.bennadel.com>, and does his best to give back to the ColdFusion community through online code demos and his "Ask Ben" blog posts. He is also a Certified Advanced ColdFusion MX7 developer and is one of the lead programmers at Nylon Technology.
ben@bennadel.com

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
Adobe	www.adobe.com/products/coldfusion/flex2		2,3
CFDynamics	www.cfdynamics.com	866-233-9626	36
CFDJ	http://coldfusion.sys-con.com/	201-802-3000	19
Hotbanana	hotbana.com/cfdj	866-296-1803	4
Intergral	www.fusiondebug.com		6
Paperthin	www.paperthin.com	1-800-940-3087	31
Virtualization Con+Expo	virtualizationconference.com	201-802-3020	21
RealWorld JAVA Seminar	realworldjava.com	201-802-3020	13
Sys-Con Website	www.sys-con.com	201-802-3020	17
WebApper	www.seefusion.com		35

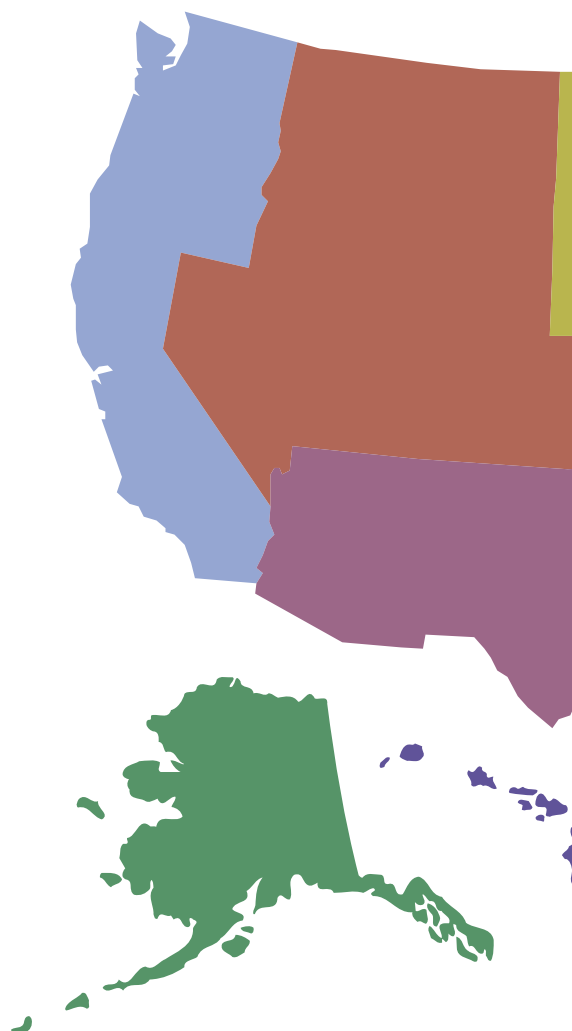
General Conditions: The Publisher reserves the right to refuse any advertising not meeting the standards that are set to protect the high editorial quality of. All advertising is subject to approval by the Publisher. The Publisher assumes no liability for any costs or damages incurred if for any reason the Publisher fails to publish an advertisement. In no event shall the Publisher be liable for any costs or damages in excess of the cost of the advertisement as a result of a mistake in the advertisement or for any other reason. The Advertiser is fully responsible for all financial liability and terms of the contract executed by the agents or agencies who are acting on behalf of the Advertiser. Conditions set in this document (except the rates) are subject to change by the Publisher without notice. No conditions other than those set forth in this "General Conditions Document" shall be binding upon the Publisher. Advertisers (and their agencies) are fully responsible for the content of their advertisements printed in ColdFusion Developer's Journal. Advertisements are to be printed at the discretion of the Publisher. This document includes the positioning of the advertisement, except for "preferred positions" described in the rate table. Cancellations and changes to advertisements must be made in writing before the closing date. "Publisher" in this "General Conditions Document" refers to SYS-CON Publications, Inc. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions. This index is provided as an additional service to our readers. The publisher does not assume any liability for errors or omissions.

ColdFusion

For more information go to...

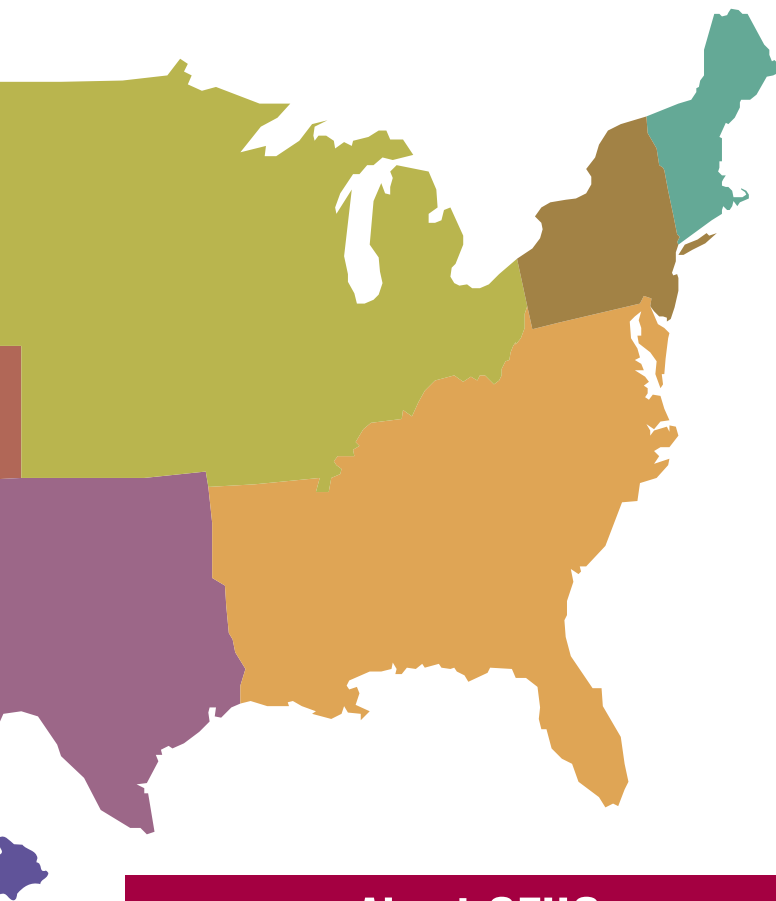
U.S.

Alabama Huntsville, AL CFUG www.nacug.com	Louisiana Lafayette, LA MMUG http://www.acadianammug.org/	New York Albany, NY CFUG www.anyfcug.org
Arizona Phoenix, AZ CFUG www.azcfug.com	Maryland California, MD CFUG http://www.smdcfug.org	New York New York, NY CFUG www.nycfug.org
California Bay Area CFUG www.bacug.net	Maryland Maryland CFUG www.cfug-md.org	New York Syracuse, NY CFUG www.cfugcny.org
California Sacramento, CA CFUG http://www.saccfug.com/	Massachusetts Boston CFUG http://bostoncfug.org/	North Carolina Raleigh, NC CFUG http://tacfug.org/
California San Diego, CA CFUG www.sdcfug.org/	Massachusetts Online CFUG http://coldfusion.meetup.com/17/	Ohio Cleveland CFUG http://www.clevelandcfug.org
Colorado Denver CFUG http://www.denvercfug.org/	Michigan Detroit CFUG http://www.detcfug.org/	Oregon Portland, OR CFUG www.pdxcfug.org
Connecticut SW CT CFUG http://www.cfugitives.com/	Michigan Mid Michigan CFUG www.coldfusion.org/pages/index.cfm	Pennsylvania Central Penn CFUG www.centralpenncfug.org
Connecticut Hartford CFUG http://www.ctmug.com/	Minnesota Southeastern MN CFUG http://www.bittercoldfusion.com	Pennsylvania Philadelphia, PA CFUG http://www.phillycfug.org/
Delaware Wilmington CFUG http://www.bvfcug.org/	Minnesota Twin Cities CFUG www.colderfusion.com	Pennsylvania State College, PA CFUG www.mmug-sc.org/
Florida Jacksonville CFUG http://www.jaxcfug.org/	Missouri Kansas City, MO CFUG www.kcfusion.org	Tennessee Nashville, TN CFUG http://www.ncfug.com
Florida South Florida CFUG www.cfug-sfl.org	Nebraska Omaha, NE CFUG www.necfug.com	Tennessee Memphis, TN CFUG http://mmug.mind-over-data.com
Georgia Atlanta, GA CFUG www.acfug.org	New Jersey Central New Jersey CFUG http://www.cjcfug.us	Texas Austin, TX CFUG http://cftexas.net/
Illinois Chicago CFUG http://www.cccfug.org	New Hampshire UNH CFUG http://unhce.unh.edu/blogs/mmug/	Texas Dallas, TX CFUG www.dfwcfug.org/
Indiana Indianapolis, IN CFUG www.hoosierfusion.com	New York Rochester, NY CFUG http://rcfug.org/	Texas Houston Area CFUG http://www.houcfug.org



User Groups

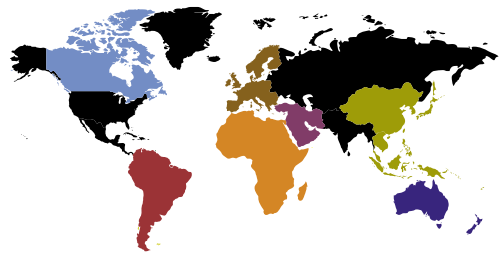
<http://www.macromedia.com/cfusion/usergroups>



About CFUGs

ColdFusion User Groups provide a forum of support and technology to Web professionals of all levels and professions. Whether you're a designer, seasoned developer, or just starting out – ColdFusion User Groups strengthen community, increase networking, unveil the latest technology innovations, and reveal the techniques that turn novices into experts, and experts into gurus.

INTERNATIONAL



Australia
ACT CFUG
<http://www.actcfug.com>

Australia
Queensland CFUG
<http://qld.cfug.org.au/>

Australia
Victoria CFUG
<http://www.cfcentral.com.au>

Australia
Western Australia CFUG
<http://www.cfugwa.com/>

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Germany
Central Europe CFUG
www.cfug.de

Italy
Italy CFUG
<http://www.cfmentor.com>

New Zealand
Auckland CFUG
<http://www.cfug.co.nz/>

Poland
Polish CFUG
<http://www.cfml.pl>

Scotland
Scottish CFUG
www.scottishcfug.com

South Africa
Joe-Burg, South Africa CFUG
www.mmug.co.za

South Africa
Cape Town, South Africa CFUG
www.mmug.co.za

Spain
Spanish CFUG
<http://www.cfugspain.org>

Sweden
Gothenburg, Sweden CFUG
www.cfug-se.org

Switzerland
Swiss CFUG
<http://www.swisscfug.org>

Turkey
Turkey CFUG
www.cftr.net

United Kingdom
UK CFUG
www.ukcfug.org



ColdFusion 8 Beta Is Released by Adobe

The wait is over

By Derek Vaughan

The wait is over and the latest version of ColdFusion has just been released in beta.

Code named "Scorpio" (the eighth sign of the Zodiac) during the development stages, the new release is officially named ColdFusion 8 and has just recently been made available for free trial beta accounts through select ColdFusion Web hosting providers.

The Hosting News caught up with Mr. Tim Buntel, senior product marketing manager, ColdFusion at Adobe, and with Mr. Monish Sood, director of marketing with ColdFusion Webhost, HostMySite.com, to learn more about the new release, its features, and how to get started with a free beta account. First the basics – the new beta release has been available since May 30, 2007. At this point the full commercial release of ColdFusion 8 is anticipated to be in a couple of months. Adobe hasn't set a specific date yet, as the company is leaving a bit of time to react should they discover anything through the public beta. According to the company, that puts the estimated delivery of the full release at pretty much dead center of the calendar year.

When asked to distill the key features of the new ColdFusion 8 into a brief statement, Mr. Buntel stated, "The tip of the iceberg on new features is already a pretty long list. I think if I had to say one thing about the release, this is Adobe ColdFusion. ColdFusion 8 shows what happens when you inject ColdFusion with Adobe's DNA.

It's really about making this great experience for the application user, with a whole slew of new features: PDFs, rich Internet applications, and multimedia presentations – simply a much more meaningful experience for users. There is also a lot more integration in the enterprise. We're doing for .NET what we've done for Java in the past. Plus Exchange integration; then also enhancing the developer's experience."

One feature that's getting considerable attention is the ability for ColdFusion 8 to invoke .NET components directly from

ColdFusion Markup Language (CFML). This is true for local or remote .NET components. According to Adobe, the new feature works much like `cfoject/CreateObject` for Java. Here's how Tim Buntel describes it: "There are organizations that have both Java and .NET development work going on, whether through acquisition or size or different divisions or different areas of the world. ColdFusion has been in a great position over the years to build an application with ColdFusion, if you were doing Java development as well – no problem. ColdFusion can take advantage of those Java assets within the ColdFusion language for its own application. Now we're able to do the same thing for .NET. If some group in your organization is writing a specific piece of business logic in .NET – up until now the ColdFusion application couldn't take advantage of the development work that the .NET team was doing – except maybe with SOAP services, and there were some issues with that. Now your ColdFusion team can leverage the business logic that the .NET team developed within ColdFusion in a much easier and efficient way than you could have with Web services.

For the .NET guys the important thing to them is that they don't have to change their code. The primary way today that you would bridge Java and .NET would be with .NET remoting, but that would require the .NET team to change their .NET. They would actually have to compile .NET assemblies to use remoting, but they don't want to have to do that. ColdFusion 8 will allow you to access .NET directly from ColdFusion without the .NET team having to do any additional work. We're unique as a Java into .NET bridge without using either Web services or .NET remoting.

Other new features of note: the ability to monitor the server applications in production. According to information supplied by Adobe, the server monitoring will permit developers to diagnose slow page queries and threads, track memory usage, manage active threads, and monitor database usage. Added in this version as well is the ability to create on-demand presentations with a customizable look and feel. ColdFusion 8 will also further im-

prove the Flex/ColdFusion connectivity through data exchange simplification and a simpler architecture.

The new features look good, but what if you're satisfied with the current ColdFusion version that you're running today? Mr. Buntel explained the benefits of upgrading to ColdFusion 8: "First



and foremost, in terms of people who are running ColdFusion today – there are compelling reasons to move to ColdFusion 8 even if you don't rewrite a single line of code that you're currently running in ColdFusion version 7 or 6.1 or 6. Simply moving your applications to ColdFusion 8 will yield a whole range of benefits. It's going to be faster, it's going to be more stable, and you will have the ability to monitor those applications in production.


Existing ColdFusion developers will be able to take advantage of all of these features quickly and easily, as well as well people who are getting into rich Internet application development and want to find the easiest and most productive server-side complement for a rich Internet app. If you're doing AJAX or you're doing Flex, or Flash for rich clients, ColdFusion 8 will allow those applications to talk to databases, object services, and enterprise infrastructure very easily and make you productive to hook that rich Internet app up to all of those back-end services." To get started, if you feel comfortable installing ColdFusion on your own computer or workstation – all you have to do is visit Adobe Labs and they will provide you with the full product along with all kinds of documentation and helpful articles to get you started. You can also access multimedia demonstrations. That's all free. According to Tim Buntel, "A developer can just grab that and go to town."

He continued, "If you don't want to install or are unable to install the product on your machine for any reason, then with a ColdFusion hosting partner such as HostMySite.com, you'll be able to sign up and have a small account that is running on ColdFusion 8 that will allow you to start writing code and use the features from that environment. Those accounts are free as well."

With regard to the beta accounts mentioned earlier, Monish Sood, director of marketing at HostMySite.com, explained the details, "The ColdFusion 8 Beta comes fully loaded with features including a 600 MB MS SQL Server database, 4 GB of disk space, and ASP.NET 3.0 framework access. We want developers to be able to test the newest release of ColdFusion in real-world situations with features that are available with our ColdFusion developer edition plans."

Mr. Sood continued, "We have worked closely with Adobe throughout the beta process to transfer all feedback from participating developers. The developers get access to production-level features to test in a beta environment. HostMySite.com provides the same level of support and service for the beta as any production plan."

For further details on the new beta release of ColdFusion 8, please visit Adobe Labs at: <http://labs.adobe.com/technologies/coldfusion8>.

To sign up for the free ColdFusion 8 beta account with HostMySite.com, please visit: <http://www.hostmysite.com/CF8>. 

About the Author

Derek Vaughan is chief marketing officer with TechPad Agency, LLC, a full resource advertising and marketing agency, specializing in products and services for the Web hosting industry. Derek's writing appears courtesy of the dedicated server hosting experts at www.thehostingnews.com.

derek@techpadagency.com

REPRINT IT!

Once you're in it...



...reprint it!

- ColdFusion Developer's Journal
- Java Developer's Journal
- Web Services Developer's Journal
- Wireless Business & Technology
- XML Journal
- PowerBuilder Developer's Journal
- .NET Developer's Journal

Contact Megan Mussa
201 802-3024
megan@sys-con.com

REprints

 SYS-CON
MEDIA

A Complete Application with RPC Communications and JMS

Multi-Tier Application Development with Flex

By Yakov Fain,
Dr. Victor Rasputnis,
& Anatole Tartakovsky

This excerpt describes the process of creating a complete Flex-Java distributed application. Upgrading Flex applications to Java Enterprise Edition applications is done with Flex Data Services. FDS provides transparent access to POJO, EJBs, and JMS and comes with adapters for frameworks like Spring and Hibernate.

These powerful capabilities come free for a single-CPU server, otherwise see your local Adobe dealer. Flex can also invoke any SOAP Web Service or send an HTTP request to any URL via Web Services and HTTPService components. Here we will illustrate Flex controls, HTTPService, RemoteObject, and Consumer via two versions of a stock portfolio application. The first version will show communications between Flash and a plain old Java object (POJO) using Flex remoting. We'll also explain how to use the HTTPService to read the RSS news feed. In the other version we'll add the stock (aka security) price feed using Flex Messaging and the Java Messaging Service (JMS). While explaining the FDS capabilities, we will also walk you through a typical design with Flex containers.

Designing a Stock Portfolio Application

Our goal is to create a Web application that will receive and display a feed containing security prices and the latest financial news as in Figures 1 and 2. This section contains a sort of functional specification of such an application. We'll populate the top portion of the screen with the stocks included in the user's portfolio. For simplicity's sake, we'll store the user's portfolio in the XML file as in Listing 1.

When the user clicks on a row with a particular stock (i.e., ADBE as in Figure 1), it will populate the lower data grid with the headlines related to the selected stock. The news should be coming from <http://finance.yahoo.com/rss/headline>. The column Link will contain the URL of the news, and when the user clicks on the link, a new browser window pops up, displaying the selected news article.

The top of the screen contains the toggle buttons Show Grid/Show Chart. When the Show Grid option is selected, the user will see the screen as in Figure 1, and when Show Chart is selected, the top data grid will be replaced with the pie chart (see Figure 2), preserving the same functionality (clicking on the pie slice repopulates the news headlines according to the selected security). The market data is refreshed on the screen every second or so.

The first version of the application will query the server POJO that generates random numbers. Later in this excerpt, we'll subscribe to a JMS topic and consume a real-time data feed from an external Java application.

In this application we'll use the basic MXML and ActionScript from the first chapters of this book. We assume that the reader has a basic knowledge of Java syntax. We've included mini-references on the Java Naming and Directory Interface and JMS. So let's roll up our sleeves...

Adding a Data Grid

We'll start by adding the top grid displaying the stock portfolio. The Flex dataGrid component is a natural choice for data that can be presented as rows and columns. MXML is a great tool for modularizing development. The name of the new .mxml



Figure 1: Stock portfolio screen - the show grid view

file automatically becomes the name of a new tag that can be reused in other files. For example, the code in Listing 2 assumes that there is an MXML file named PortfolioView1.mxml (in reality it can also be an ActionScript file named PortfolioView1.as or any other file that is assigned to this tag via the component library manifest).

In case of a default namespace (`xmlns=""`), our PortfolioView1.mxml from Listing 3 will be co-located in the same directory with the application file portfolio.mxml. Let's discuss the design of PortfolioView1. It contains DataGrid portfolioGrid in a Flex Panel with the title "Portfolio." XML from Listing 1 is loaded into the portfolioModel e4x object. The list of securities from that file is fed into portfolioGrid via a binding expression `{portfolioModel.security}`. This expression returns an XMLList of all nodes named "security" that are direct children of the root node (see Listing 3).

Even if we won't add any more code, isn't it impressive that it takes only a dozen lines of code to read the XML file, parse it, and display it in a grid shown in Figure 3? But this application has a static nature: it does not connect to any price quote feed.

In other words, you would always see \$33.38 for Microsoft, and \$82.15 for IBM.

In Listing 3, the curly braces surrounding the expression indicate that this expression is being used as a source in data binding. It is crucial for Flex programming to fully understand the strengths and weaknesses of binding. Binding is based on

event listeners automatically generated by the Flex compiler as a response to declarative binding annotations. To initiate binding generation, developers use a combination of curly braces, `mx:` Binding tags, and `[Bindable]` metadata directives. (Refer to the Adobe Flex manual for more detail.)

Next, we need to periodically connect to the server for new prices and update the Price and Value columns. So let's use a special Flex component called RemoteObject:

```
<mx:RemoteObject id="freshQuotes" destination="Portfolio"...>
```

The RemoteObject component allows calling methods of a specified remote POJO, which is configured on the server as a destination Portfolio in a special XML file. We'd like to emphasize that Flex transparently calls Java from ActionScript. The client needs to know the name of the destination and the method to call, for example, `getQuotes()`. All the dirty work of data marshaling between ActionScript and Java is done for you by the Flex framework. If, for example, a Java method returns an object of the StockQuoteDTO.java type, Flex de-serializes the Java object and builds its ActionScript peer on the client. However, for performance reasons, it's recommended that you create the peer ActionScript class and register it with the framework. Please note that all RPC communications, including RemoteObject, are asynchronous. In other words, we don't exactly call a remote method, but rather send a message to the server, requesting a call of the specific Java method. Not only is the client's request(s) executed asynchronously, but even sending to the server is done asynchronously. If you need to do multiple RemoteObject invocations in your script, Flex will batch them together and send in the end of the script execution.

The results of remote invocations are returned via events. RemoteObject provides the result event for success or fault for failures. You should write the corresponding handler functions. Flex will call these methods, supplying an Event object as a parameter. It's your responsibility to get the information from the event and act accordingly. Friendly advice: you will save yourself hours of time if you supply a fault handler.

In the next code snippet we set concurrency to last, because if Flex decides to batch the outgoing requests, we do not want to send out more than one request in a batch; if a user clicks on the screen sending more than one request in quick succession, the last request will suppress all previous ones. Similarly, when the results are coming back, we are interested only in the one we sent last (see Listing 4).

The tag `<mx:RemoteObject>` allows using result and fault handling on both the object and method levels. The method settings will take precedence over the RemoteObject's ones. For server-side support, you have to download and install Flex Data Services 2 Express Edition (<http://www.adobe.com/products/flex/>), and deploy it as a Web application in the J2EE server of your choice, for example, in Tomcat. FDS comes with a set of XML files, which you will use to configure your server-side objects.

To register a POJO class with a Flex client we need to update the configuration file on the server side. This lets you hide details of the service provider (i.e., actual Java class names) by specifying so-called destinations where you specify access constraints, etc. The following section in Listing 5 has to be added to



Figure 2: Stock portfolio screen – the show chart view



Figure 3: A data grid populated from Portfolio.xml

the `remoting-config.xml` file.

Clients won't know that the actual name of our POJO is `com.theriabook.ro.Portfolio`, but they'll be able to refer to it by the nickname `Portfolio`. Flex looks for classes specified in destination mappings on the Web Application classpath including JARs inside `WEB-INF/lib` and classes under `WEB-INF/classes`. The Java class `Portfolio.java` (see Listing 6) is a simple random number generator simulating market-like real-time price changes for several hard-coded securities.

The `StockQuoteDTO.java` (see Listing 7) contains the last price of a particular stock. However, the client can really benefit from knowledge of the structure and the datatypes of the returned DTOs. Listing 8 shows the `ActionScript`'s counterpart for the `StockQuoteDTO.java` object. While Flex does not need this definition in order to deserialize the Java object that includes member variables of standard types (by default it creates a dynamic object and adds the required properties of the Java object that's being deserialized), it does help performance (since the deserialized object is immediately allocated in memory), ensures the output datatypes, and enforces the type conversion.

The `[RemoteClass...]` metadata tag above the class definition tells the Flex de-serialization method to use this particular class for de-serialization whenever the server sends `com.theriabook.jms.dto.StockQuoteDTO` object down.

When the client successfully gets the quotes, it processes them and asks for new ones, as shown in Listing 9.

The E4X provides a very elegant solution for navigating an XML object here. The `applyQuotes` function iterates through the quotes from our portfolio and gets the `XMLList` based on the matching `Symbol` attribute via evaluation of the `portfolioModel` security expression. (`Symbol==quote.symbol`.) This looks similar to XPath, but it's easier, isn't it?

Since there's a chance that the E4X expression above will return an empty `XMLList`, it's better to check for zero-length to avoid exceptions.

We are modifying the same `XMLList` that has been set as the data provider of our grid. In fact for data grids, Flex maintains an internal

`XMLListCollection` with this `XMLList` as a source. When the program changes the data in the `XMLList`, these changes are automatically reflected on the screen. Error reporting is often done by calling `Alert()`, which brings up a pop-up window. But we suggest a less obtrusive way, whereby as an error condition disappears, the normal display restores without user interaction. Let's put a simple red `Label` control right above the data grid. Later we'll embed it in the `Panel`'s title. An error, if any, will be displayed in this `Label` control, and to make it a bit fancier, the detailed error description will be displayed as a tooltip whenever the user moves the mouse over this field:

```
<mx:Label color="red" toolTip="{errorTip}" text="{errorText}"
width="100%"/>
```

To implement this functionality, in the scripting section we will create two bindable variables:

```
[Bindable]
private var errorText:String;
[Bindable]
private var errorTip:String;
```

When an error occurs, the `onFault` function sets the values of the variables `errorText` and `errorTip`, and their bindable nature will immediately display these values in the `<mx:Label>`. But most important, we will attempt to recover by pulling the new quotes set. (see Listing 10). Do not forget to clean the `errorText` and `errorTip` variables in the function `onResult`, if the next attempt to pull the quotes will be successful.

Let's spend some time discussing the process of initiating the quote request. The function `pullQuotes()` gets initially invoked upon creation of the `Panel`:

```
<mx:Panel xmlns:mx="http://www.adobe.com/2006/mxml" title="Portfolio" .
. .
creationComplete="pullQuotes();">
```

To generate a remote call of some anonymous function every second, we'll use the `setTimeout` mechanism. The anonymous function initiates the call of the Java object proxied by the remoting destination `freshQuotes`:

```
private function pullQuotes():void{
setTimeout(function () :void {freshQuotes.getQuotes();},1000);
}
```

Listing 11 has the complete code of the first version of `PortfolioView1.mxml`, which contains just a data grid.

Adding the Charting Component

The population of the data grid is complete, and we are ready to work on adding a Flex charting component (see Figure 4).

Let's create a simple application that adds the pie chart below the data grid and gives the grid and the chart 50% of the screen height each:

```
<?xml version="1.0" encoding="utf-8"?>
<!--portfolio2.mxml -->
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="*" >
```

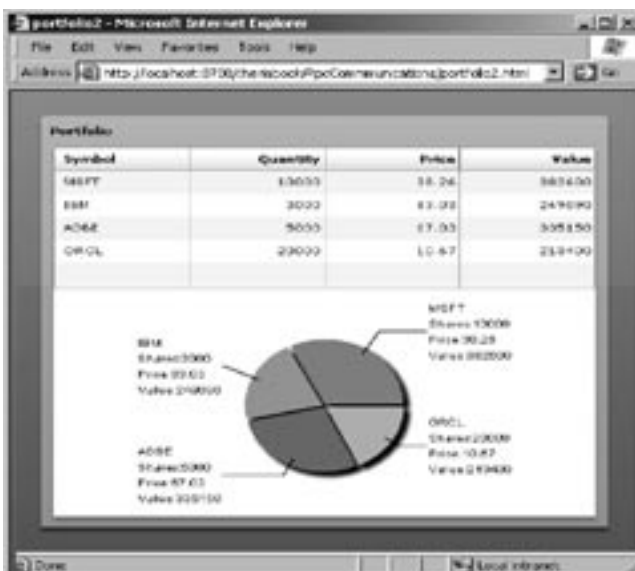


Figure 4: Adding a charting component

```
<PortfolioView2 id="pv"/>
</mx:Application>
```

The fragment of the PortfolioView2.mxml is shown in Listing 12. (Listings 12-16 can be downloaded from the online version of this article at <http://coldfusion.sys-con.com>.)

Flex Panel containers have a layout property (horizontal, vertical, and absolute). Since the vertical layout is the default, our chart is positioned right under the grid. Please note that chart's dataProvider is based on the same XMLList portfolioModel.security as is the dataProvider of the portfolioGrid. In other words, we have two views of the same data model. The data binding feature results in immediate updates of both controls on each change of the model. Java Swing developers will appreciate the benefits of this feature as opposed to the JavaBean and property listeners hassle (see Listing 13).

For the pie chart, we've selected a callout type of label positioning, specified so the size of the pie is proportional to the attribute Value, and we've added a 3D depth to the chart by setting explode Radius to 0.02. Note how we've assigned the function name showPosition to the labelFunction attribute of the pie chart. The signature of the labelFunction assumes that the first argument brings the element of the dataProvider corresponding to the current wedge in the series.

Chart/DataGrid Toggling

Imagine a deck of playing cards: only the top card is visible.

Hide the top card and you'll see the next one. We'll use the deck of two "cards": one card will display the grid, and another one – the chart. To Java Swing developers this should look like the CardLayout. In Flex jargon it's called <mx:ViewStack>.

A fragment of PortfolioView3.mxml introduces MXML components <mx:ViewStack>, <mx:ToggleBarButton>, and <mx:Canvas> (see Listing 14).

The most suitable Flex containers for toggling the views are ViewStack or its direct descendant TabNavigator. The latter uses more screen real estate to paint the tabs. So we'd rather put the view toggling controls on the unused area of the Panel's title bar. The ViewStack component provides programmatic indexed access to the child containers and shows them one at a time.

The simplest Flex container is called Canvas, so we wrap up the DataGrid and the PieChart separately inside it. A Canvas is a descendant of the Container ActionScript class and has the properties label and icon. There are two "non-programmatic" ways to use these properties. First, certain containers use them implicitly, for instance, TabNavigator automatically arranges its tabs to display labels and show icons from the nested child containers. The second way is to explicitly use the ViewStack as a data provider for the descendants of the NavBar control such as ButtonBar, LinkBar, and, in our case, ToggleBarButton. When you use a ViewStack as a data provider, the label and icon properties of the ViewStack container's children are used to populate the navigation items. The ViewStack feeds ToggleButtonbar and ToggleButtonbar controls the ViewStack in return (see Listing 15).

Efficient Web Content Management


CommonSpot™ Content Server is the ColdFusion developer's leading choice for efficient Web content management.

Our rich Web publishing framework empowers developers with out-of-the-box features such as template-driven pages, custom content objects, granular security, and powerful ColdFusion integration hooks (just to name a few), allowing you to rapidly build and efficiently deploy dynamic, high performance Web sites.

CommonSpot's open architecture and extensive APIs enable easy customization and integration of external applications. With CommonSpot, you can design and build exactly what you need. Best of all, CommonSpot puts content management in the hands of content owners. With non-technical users responsible for creating and managing Web content, developers are freed to focus on strategic application development.

Evaluate CommonSpot today.

To see your site *running* under CommonSpot, call us at 1.800.940.3087.



fast
easy
affordable

features.

- 100 % browser-based
- Content object architecture
- Template driven pages
- 50+ standard elements
- Content reuse
- Content scheduling
- Personalization
- Flexible workflow
- Granular security
- Mac-based authoring
- 508 compliance
- Full CSS support
- Custom metadata
- Taxonomy module
- Extensible via ColdFusion
- Web Services content API
- Custom authentication
- Replication
- Static site generation
- Multilanguage support

1.800.940.3087
www.paperthin.com

Paper | Thin

© Copyright 2005 PaperThin, Inc. All rights reserved.

The toggle buttons Show Grid/Show Chart use images, and it's a good idea to embed these resources right into the SWF file. This way the browser can download the entire client in one HTTP request (but the size of the SWF file becomes larger). For the same reason, the multi-file Java applets are packaged into a single JAR.

The next step is to move the toggle buttons up, as shown in Figure 7.

In Listing 15, we use canvases to wrap, but now we're going wrap up the entire Portfolio Panel. Yes, the Panel will become a child of the Canvas. Here is why we need it. The Canvas is the simplest container, and it's also the absolute positioning container. In other words, if you place A and B as children of the Canvas, they overlap unless each of them has specific x and y coordinate settings. So we're planning on overlapping the ToggleButtonBar and the PortfolioPanel in a single Canvas. As an additional measure, to make the ToggleButtonBar appear on the far right, we put the ToggleButtonBar inside the HBox container. Make sure that the RemoteObject and the XML are moved out to become children of the Canvas; they have to be at the outermost level – this is an MXML requirement (see Listing 16).

Dealing with Financial News

Let's set up the scene for the financial news first, and then

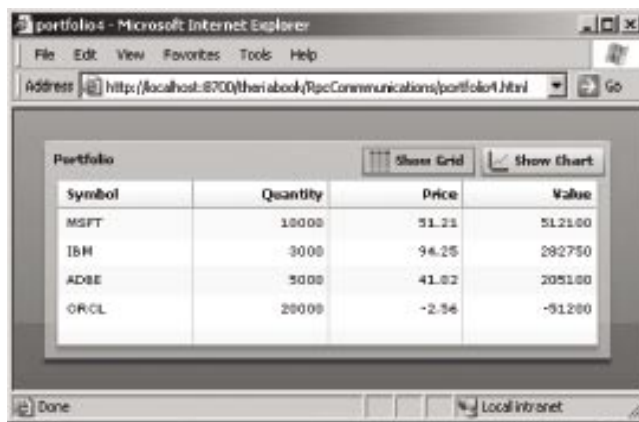


Figure 5: Using ViewStack for toggling

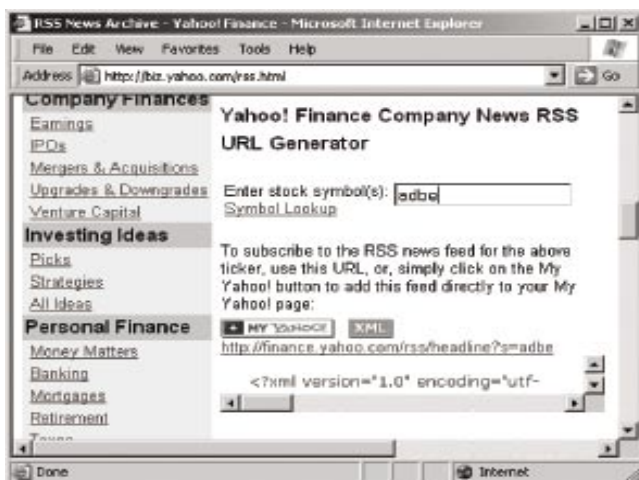


Figure 6: RSS news archive from Yahoo! Finance


we'll add them to the screen that we've developed so far. The first cut of our financial news screen will look like Figure 8. The data you see here are provided by an RSS feed. RSS stands for Real Simple Syndication and is used for presenting such data as news, blogs, or other Web content in a form of XML that contains summaries of the articles as well as links to the full version of the content. We use it simply to illustrate the client/server communication via the Flex component called `<mx:HTTPService>`, which facilitates HTTP POST and GET requests from the Flash player to a remote URL with automatic embedding of the parameters. In particular, we'll be using the RSS news generator offered by Yahoo! Finance. Just enter the following URL in your browser: <http://biz.yahoo.com/rss.html>.

You should see a screen similar to the one depicted in Figure 6. This Web site lets you enter a stock symbol, for example, ADBE.

You'll see a little orange XML button, with a new URL that looks like <http://finance.yahoo.com/rss/headline?s=adbe>. Just follow this link, which will bring you to the RSS XML feed with the latest news about the symbol ADBE.

In Part 2 of this excerpt we'll be done with programming master-detail relationships; our users will click on the row in the portfolio grid (Figure 7) and see the news as in Figure 9, then click on one of the lines in the column Link, which opens a new Web browser's window with the full news content.

...

This article is an excerpt from Chapter 5 of *Rich Internet Applications with Adobe Flex & Java* by Yakov Fain, Dr. Victor Rasputnis, and Anatole Tartakovsky, published by SYS-CON Books/2007. 

About the Authors

Yakov Fain is a managing principal of Farata Systems, consulting, training and product company. He's authored several Java books, dozens of technical articles. SYS-CON Books will be releasing his latest book, "Rich Internet Applications with Adobe Flex and Java: Secrets of the Masters" this Fall. Sun Microsystems has nominated and awarded Yakov with the title Java Champion. He leads the Princeton Java Users Group. Yakov teaches Java and Flex 2 part time at New York University. He is an Adobe Certified Flex Instructor.

Dr. Victor Rasputnis is a Managing Principal of Farata Systems. He's responsible for providing architectural design, implementation management and mentoring to companies migrating to XML Internet technologies. He holds a PhD in computer science from the Moscow Institute of Robotics. You can reach him at vrasputnis@faratasystems.com

Anatole Tartakovsky is a Managing Principal of Farata Systems. He's responsible for creation of frameworks and reusable components. Anatole authored number of books and articles on AJAX, XML, Internet and client-server technologies. He holds an MS in mathematics. You can reach him at atartakovsky@faratasystems.com

yfain@faratasystems.com
vrasputnis@faratasystems.com
atartakovsky@faratasystems.com

Listing 1 A fragment of a portfolio.xml

```
<portfolio>
  <security>
    <Symbol>MSFT</Symbol>
    <Quantity>10000</Quantity>
    <Price>33.38</Price>
    <Value>1</Value>
  </security>
  <security>
    <Symbol>IBM</Symbol>
    <Quantity>3000</Quantity>
    <Price>82.15</Price>
    <Value>1</Value>
  </security>
  ...
</portfolio>
```

Listing 2 The first version – portfolio1.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<!--portfolio1.mxml -->
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" xmlns="*"
>
  <PortfolioView1 id="pv"/>
</mx:Application>
```

Listing 3 The first version of PortfolioView.mxml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- PortfolioView1.mxml -->
<mx:Panel xmlns:mx="http://www.adobe.com/2006/mxml"
  title="Portfolio" width="100%" height="100%" ...>
  <mx:XML format="e4x" id="portfolioModel" source="portfolio.xml" />
  <mx:DataGrid id="portfolioGrid" width="100%"
    dataProvider="{portfolioModel.security}">
    <mx:columns>
      <mx:DataGridColumn dataField="Symbol"/>
      <mx:DataGridColumn dataField="Quantity" textAlign="right"/>
      <mx:DataGridColumn dataField="Price" textAlign="right"/>
      <mx:DataGridColumn dataField="Value" textAlign="right"/>
    </mx:columns>
  </mx:DataGrid>
  ...
</mx:Panel>
```

Listing 4 RemoteObject Tag

```
<mx:RemoteObject id="freshQuotes" destination="portfolio"
  fault="onFault(event);">
  <mx:method name="getQuotes" concurrency="last" result="onResult(event);"/>
</mx:RemoteObject>
```

Listing 5 Configuring a Flex remoting service

```
<destination id="Portfolio">
  <properties>
    <source>com.theriabook.ro.Portfolio</source>
```

```
</properties>
</destination>
```

Listing 6 A simple stock quote generator – Portfolio.java

```
package com.theriabook.ro;
import java.util.Random;
import com.theriabook.jms.dto.StockQuotedTO;

public class Portfolio {
  static Random random = new Random();
  static StockQuotedTO[] quotes = {
    new StockQuotedTO("IBM", 82.0),
    new StockQuotedTO("MSFT", 27.0),
    new StockQuotedTO("ADBE", 38.0),
    new StockQuotedTO("ORCL", 13.0)};
  double volatility=.05;
  public StockQuotedTO[] getQuotes() {
    for (int i = 0; i < quotes.length;i++){
      quotes[i].last += random.nextGaussian()* volatility;
    }
    return quotes;
  }
}
```

Listing 7 The Java DTO – StockQuoteDTO.java

```
package com.theriabook.jms.dto;
import java.io.Serializable;
public class StockQuoteDTO implements Serializable {
  private static final long serialVersionUID = 4672447577075475117L;
  public String symbol;
  public double last;
  public StockQuoteDTO(String sym, double newPrice){
    symbol = sym;
    last = newPrice;
  }
}
```

Listing 8 The ActionScript on the client: StockQuoteDTO.as

```
package com.theriabook.jms.dto {
[RemoteClass(alias="com.theriabook.jms.dto.StockQuoteDTO")]
public dynamic class StockQuoteDTO
{
  public var symbol:String;
  public var last:Number;
}
```

Listing 9 Processing received quotes

```
private function onResult(event:ResultEvent):void {
  ...
  var quotes:Array = event.result as Array;
  applyQuotes(quotes);
  // Pull the next set of quotes
  pullQuotes();
}
```



```

private function applyQuotes(quotes: Array):void {
    for (var i:int=0; i<quotes.length; i++) {
        var quote:StockQuotedTO = StockQuotedTO(quotes[i]);
        var list: XMLList = portfolioModel.security.(Symbol==quote.
symbol);
        if (list.length()!=0) {
            var row:XML = XML(list);
            row.Price = Math.round(100*quote.last)/100;
            row.Value = Math.round(row.Price * row.Quantity);
        }
    }
}

```

Listing 10 Error processing

```

private function onFault(event:FaultEvent):void {
    errorText = "Portfolio feed failure...";
    errorTip = "Destination:" + event.currentTarget.destination + "\n"
+
    "Fault code:" + event.fault.faultCode + "\n" +
    "Detail:" + event.fault.faultDetail;
    // Try to pull the new quotes
    pullQuotes();
}

```

Listing 11 PortfolioView1.mxml

```

<?xml version="1.0" encoding="utf-8"?>
<!-- PortfolioView1.mxml -->
<mx:Panel xmlns:mx="http://www.adobe.com/2006/mxml"
    title="Portfolio"
    width="100%" height="100%"
    creationComplete="pullQuotes();"
>
    <mx:XML format="e4x" id="portfolioModel" source="portfolio.xml" />
    <mx:Label color="red" toolTip="{errorTip}" text="{errorText}"
width="100%" />
    <mx:DataGrid id="portfolioGrid" width="100%" dataProvider="{portfoli
oModel.security}" >
        <mx:columns><mx:Array>
            <mx:DataGridColumn dataField="Symbol"/>
            <mx:DataGridColumn dataField="Quantity"
textAlign="right"/>
            <mx:DataGridColumn dataField="Price" textAlign="right"/>
            <mx:DataGridColumn dataField="Value" textAlign="right"/>
        </mx:Array></mx:columns>
    </mx:DataGrid>
    <mx:RemoteObject id="freshQuotes" destination="portfolio"
        fault="onFault(event);" >
        <mx:method name="getQuotes" concurrency="last"
result="onResult(event);"
        />
</mx:RemoteObject>
<mx:Script><![CDATA[
    import mx.rpc.events.*;
    import com.theriabook.jms.dto.StockQuotedTO;

```

```

[Bindable]
private var errorText:String;
[Bindable]
private var errorTip:String;

private function onResult(event:ResultEvent):void {
    errorText = "";
    errorTip = "";
    var quotes:Array = event.result as Array;
    applyQuotes(quotes);
    // Pull new quotes set
    pullQuotes();
}

private function onFault(event:FaultEvent):void {
    errorText = "Portfolio feed failure...";
    errorTip = "Destination:" + event.currentTarget.destination +
"\n" +
    "Fault code:" + event.fault.faultCode + "\n" +
    "Detail:" + event.fault.faultDetail;
    // Pull new quotes set
    pullQuotes();
}

private function applyQuotes(quotes: Array):void {
    for (var i:int=0; i<quotes.length; i++) {
        var quote:StockQuotedTO = StockQuotedTO(quotes[i]);
        var list: XMLList = portfolioModel.
security.(Symbol==quote.symbol);
        if (list.length()!=0) {
            var row:XML = XML(list);
            row.Price = Math.round(100*quote.last)/100;
            row.Value = Math.round(row.Price * row.Quantity);
        }
    }
}

private function pullQuotes():void{
    setTimeout(    function ():void {freshQuotes.getQuotes();
},1000);
}
]></mx:Script>
</mx:Panel>

```

Download the Code...
Go to <http://coldfusion.sys-con.com>

Have you ever
wanted to "see"
what's going on
inside your
ColdFusion servers
and applications?

Now you can.



SeeFusion is a unique, powerful tool for real-time monitoring of your ColdFusion applications and servers, providing a wide variety of metrics on request times, database queries, memory utilization, code performance and more!

 **www.seefusion.com**

SeeFusion is a product of Webapper Services, LLC
Development, Consulting, Products, Training
www.webapper.com



Relax.

You're hosting with CFDynamics

We at CFDynamics work hard to have the most reliable servers, 99.9% uptime and outstanding customer service! We hope our customers enjoy just a little more "Relax" time because of our efforts.

Try us out today and see why our customers are kicking back a little more often.

- **99.9% Average Uptime!**
- **State-of-the- Art Data Center including HVAC, fire suppression, bandwidth & security**
- **24/7 Network Monitoring**

CFDynamics

The web host that helps you maximize your website



1.866.233.9626



customerservice@cfodynamics.com



cfodynamics.com