

COLD FUSION Developer's Journal

ColdFusionJournal.com

April 2004 Volume: 6 Issue: 4

SAVE UP TO \$400



Subscribe today
and get up to
3 FREE CDs!
SEE PAGE 35 FOR DETAILS

Editorial

Dreamweaver MX 2004, Take 2

Robert Diamond page 5

User Conference

CFUN-04: Who, What, When, Why?

Charlie Arehart page 7

RETAILERS PLEASE DISPLAY
UNTIL JUNE 30, 2004

\$9.99US \$9.99CAN



**SYS-CON
MEDIA**

Performance

Load Testing on a Budget

By Kelly Tetterton

page 8

**Frameworks: Features Without
Fixtures** *onTap framework can save huge
amounts of development time*

Isaac Dealey

16

**Foundations: Adventures in
Encapsulation** **PART 3 Security in Fusebox 4**

Hal Helms

22

**G11N: Do You Want Coffee with That
Mojibake?** *Character encodings and CFMX*

Paul Hastings

28

**Journeyman CF: Making the Case
for CFML on J2EE** *There are some
compelling reasons to go this route*

Charlie Arehart

36

<BF> on <CF>: The DRK Treasure Trove

Ben Forta

*DevNet Resource Kit may have
just what you're looking for*

48



For the greatest hits
of the 70's, 80's and 90's
call your web host's
tech support.

For answers call us at 1-866-EDGEWEB
3 3 4 3 9 3 2

When calling your web host for support you want answers, not an annoying song stuck in your head from spending all day on hold. At EdgeWebHosting.net, we'll answer your call in two rings or less. There's no annoying on-hold music, no recorded messages or confusing menu merry-go-rounds. And when you call, one of our qualified experts will have the answers you're looking for. Not that you'll need to call us often since our self-healing servers virtually eliminate the potential for problems and automatically resolve most CF, IIS and ASP problems in 60 seconds or less with no human interaction. Plus, our multi-user support system allows you to track support requests for each of your engineers individually, lookup server availability, receive a copy of all errors on your site in real time, and even monitor intrusion attempts on your site in real time. **For a new kind of easy listening, talk to EdgeWebHosting.net**

By the Numbers:

- 2 Rings or less, live support
- 100% Guarantee
- 99.998% Uptime
- 150 MBPS Fiber Connectivity
- 24 x 7 Emergency support
- 24 Hour free backup



EDGE
WEB HOSTING

What are you WAITING for?

www.edgewebhosting.net

Shared Hosting ¥ Managed Dedicated Servers ¥ Semi-Private Servers
ColdFusion ¥ SQL Server ¥ .NET ¥ Self-Healing Servers ¥ Value Priced

© 2003 Edge Web Hosting. All rights reserved. Edgewebhosting.net and Edge Web Hosting logos are trademarks of ACS Edgewebhosting.net. ColdFusion is a trademark of Macromedia. ASP, SQL Server, .NET are registered trademarks of Microsoft Corp.

Win
a year
of free
hosting*



*On Shared Hosting or the equivalent value
See <http://edgewebhosting.net/cfdj> for details

THERE ARE SHORTCUTS TO SUCCESS.

AND OUR CERTIFIED COLDFUSION INSTRUCTORS
WILL SHOW YOU THEM.

When you take a ColdFusion class with WinMill Software, you're learning the tools of the trade from the most talented and respected experts in the country. Our vendor-certified instructors are experienced professionals who are well-versed in the latest techniques and spend 75% of their time in the field working on real-world projects. Through an intensive hands-on lesson plan and expert knowledge transfer, you'll learn to harness the true power of ColdFusion.

Every student receives complimentary access to WinMill's On-Line Education Forum (OLEF), which allows students to ask questions and receive answers from our technical experts and instructors, quickly and accurately.

420 Lexington Avenue, Suite 444, New York, NY 10170

We are currently offering the following
three-day courses:

Fast Track to ColdFusion MX
Advanced ColdFusion MX Development

**Call 1-888-711-MILL (6455) or email
us at inquiry@winmill.com for our latest
price specials and offers.**

**Go to www.winmill.com to view
our Course Schedule.**

Courses are held at our state of the art facility
in New York City or on site per request.

We also offer vendor-certified training in
IBM, Check Point and Mercury Interactive.





It's everybody's PDF™

Finally, a software company that offers affordable yet flexible PDF solutions to meet every customer's needs. Using activePDF™ to automate the PDF creation process eliminates the need for end-user intervention so your employees can concentrate on what they do best.

Licensed per server, activePDF solutions include a COM interface for easy integration with ColdFusion. Dynamically populate PDF forms with information from a database, convert ColdFusion web pages to PDF on the fly, dynamically print reports to PDF using CF and much more. Users can also merge, stitch, stamp, secure and linearize PDF, all at a fraction of the cost of comparable solutions. Download your free trial version today!



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

Robert Diamond robert@sys-con.com

technical editors

Raymond Camden raymond@sys-con.com

Simon Horwith simon@sys-con.com

executive editor

Jamie Matusow jamie@sys-con.com

editor

Nancy Valentine nancy@sys-con.com

associate editors

Gail Schultz gail@sys-con.com

Jean Cassidy jean@sys-con.com

Jennifer Van Winckel jennifer@sys-con.com

research editor

Bahadir Karov, PhD bahadir@sys-con.com

production

production consultant

Jim Morgan jim@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com

Richard Silverberg richards@sys-con.com

Tami Beatty tami@sys-con.com

contributors to this issue

Charlie Arehart, Isaac Dealey, Robert Diamond,
Ben Forta, Paul Hastings, Hal Helms,
Jeffrey Houser, Kelly Tetterton

editorial offices

SYS-CON MEDIA

135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

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

135 Chestnut Ridge Rd., Montvale, NJ 07645

©copyright

Copyright © 2004 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.

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.

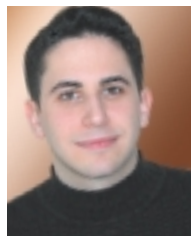
Dreamweaver MX 2004, Take 2

As many of you know from my editorial a few months back, I made the official switch from Homesite+ to Dreamweaver MX 2004 shortly after it was first released. The new product took some getting used to and the transition stretched out; it wasn't until I uninstalled Homesite+ entirely from my machine (and was no longer able to sneak back in when no one was looking) that I finally got myself into the habit of using Dreamweaver, and really began to explore its plethora of features. It took that extra push to fully delve into the ins and outs, and once I did, I was surprised and pleased with the switch.

Just last week, I was over at a Dreamweaver-less friend's house doing some collaborative development. While helping him with some programming in Homesite+ I really began to miss my new IDE of choice. It's amazing how quickly I'd developed my little habits and quirks for programming in Dreamweaver. I'm sure many of you are the same in that while you can pick up and learn most software, your productivity is certainly the highest with what you know best. Avoid separating a developer from his home environment – or else!

For those who dabble in Dreamweaver – and those who have made the full leap – things recently got even better with the release of the 7.0.1 product updater. The biggest “fix” is that Macromedia boasts performance improvements of 50% on Windows machines, and up to 70% on Macs. I don't know that I'd put the number that high myself, but it's visibly snappier – and who knows what I can do with all those newfound minutes that I'm saving up by day's end.

In addition to improving the performance of Dreamweaver 2004, they've improved the stability as well, which so far has resulted in 100% uptime on my laptop with it open an average of 10 hours a day. Both of these are very welcome additions for software that many of us spend our days, nights, weekends, and holidays developing in. In addition to these major updates, there's a slew of smaller ones,



By Robert Diamond

all of which you can read about on Macromedia's Web site while waiting for your updater to download.


An update to the ColdFusion Reference help was released as a separate DW 2004 download. “The new reference fixes a problem with section introductions that contained only a heading and no links to the section topics. It also fixes a problem where the standard refer-

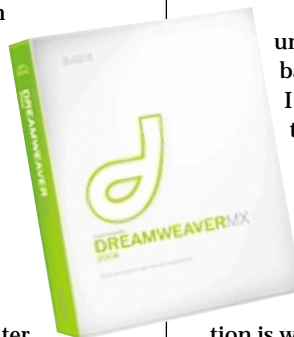
ence subheadings (Description, Category, Syntax, and so on) were not displayed.” This isn't bundled into the 7.0.1 updater because of its file size, and the low percentage of Dreamweaver users that are CFers as well, I assume, but it's available separately, and worth plunking in while you're there.

So now Dreamweaver can do nearly everything for you, except design all of your pages while you sit back and relax with a cup of coffee, but perhaps they'll include that in an upcoming version.

I'm going to end this month's column with a short rant of caution, based on a general observation that I've made of many sites I frequent that have recently undergone redesigns. It's important to remember that just because most of us are on broadband now, it doesn't mean that each and every bit of the pipe can or should be clogged up by your site. Easy access to informa-

tion is what users need. That's something to remember – no matter which IDE you use.

</rant> 



About the Author

Robert Diamond is vice president of information systems for SYS-CON Media, and editor-in-chief of ColdFusion Developer's Journal.

Named one of the “Top thirty magazine industry executives under the age of 30” in Folio magazine's November 2000 issue, Robert holds a BS degree in information management and technology from the School of Information Studies at Syracuse University. Visit his blog at www.robertdiamond.com.

robert@sys-con.com

BlueDragon [6.1]

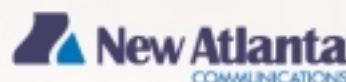
RELEASED!



RUN YOUR CFML:

Within BlueDragon Server,
or as a native .NET or J2EE
web application.

On the platform, web server,
and operating system of
your choice.



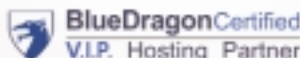
Get BlueDragon hosting:

CFDynamics is now offering BlueDragon hosting!
Find out more by visiting: www.cfdynamics.com/bluedragon

As one of the ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to deliver new and cutting edge technologies to the ColdFusion community. We are excited to announce our premiere partnership with New Atlanta by offering BlueDragon hosting. We look forward to seeing how BlueDragon expands the possibilities of ColdFusion. Come join CFDynamics and New Atlanta in expanding the ColdFusion horizon!

CFDynamics

A Division of Konnections Inc.



POWERFUL HOSTING PLANS

- FREE SQL server access
- FREE account setup
- UNLIMITED email accounts
- GENEROUS disk space / data transfer
- 30 day MONEY-BACK GUARANTEE
- GREAT VALUE!

RELIABLE NETWORK

- 99.99% average uptime!
- State-of-the-art data center has complete redundancy in power, HVAC, fire suppression, bandwidth, and security
- 24 / 7 network monitoring

FANTASTIC SUPPORT SERVICES

- Comprehensive online support
- Knowledgeable phone support
- We focus on your individual needs

WWW.CFDYNAMICS.COM

866 - CFDYNAMICS

866-233-962-6427

CFUN-04: Who, What, When, Why?

Something for everyone

There are many excellent conferences and user groups for CFML developers, and one that consistently gets great reviews is Michael Smith's CFUN event, which will be held in the Washington, DC, area June 26–27. This preview gives you the 4-1-1 on the event.

CFUN is the national ColdFusion and Web programming conference that Rockville, MD–based IT firm TeraTech (www.teratech.com) hosts each June in the DC area. CFUN stands for ColdFusion User Network and, based on my experiences at CFUNs I have attended (and spoken at) in past years, there's plenty of opportunity for users to learn and network – and have fun doing it!

Conferences like this are important not just for what you learn but also for who you meet. There's a tendency for many developers to find themselves working in isolation. CFUN and conferences like it offer a chance to immerse yourself for a few days in best practices and fresh ideas while also connecting or reconnecting with friends and mentors in the community.

Many conference-goers find that when they meet the speakers (many of them nationally known) they're far more approachable than expected – and usually willing to answer questions even after the event.

The tutorials, keynotes, and lectures offer great insights and techniques, while the informal interactions that take place are of greater value for many. As one attendee last year said, "Without events like CFUN we become stale. With it, we're fresh!"

Plenty of Topics for Everyone

There are five tracks this year, with topics ranging from introductory to advanced, covering ColdFusion and Flash/Flex programming and more, as well as general topics applying to all, such as project management and accessibility. Here's the track rundown:

- **Bootcamp:** Basic ColdFusion and Flash topics
- **Advanced:** Advanced ColdFusion topics
- **Empowered:** Fusebox and Project management topics
- **Accessibility:** Creating Web sites that disabled people can use, Section 508



By Charlie Arehart

- **Integration:** Flash, Flex, and other technologies integrated with CF topics

Some of the 42 topics include:

- Oh Grow Up! "Kiddie Scripter" to "Software Architect"
- Working with Remote Data
- Application Blueprinting
- Deploying CFML on J2EE: Opportunities and Challenges
- Component Development in Flash MX 2004
- How NOT to Fusebox – Lessons from the Trenches
- Integrating ColdFusion with Microsoft Office
- How to be a Guru Coder
- HTML Markup for Accessibility You Never Knew About
- Leveraging Macromedia Flex and ColdFusion MX
- Integrating Flash with Fusebox
- Creating Accessible Web Forms
- CFMX XML Tricks and Traps
- XSLT for Data Manipulation
- CFMX – Strange But Not True
- Managing ColdFusion Projects
- User-Defined Functions
- Advanced Topics in Custom Tags, UDFs, and CFCs
- **CFDJ** Panel
- Practice Safe Stress

—continued on page 46

About the Author

Charlie Arehart is CTO of New Atlanta Communications, makers of BlueDragon. A Macromedia Certified Advanced ColdFusion developer and trainer, he continues to support the CFML community, contributing to several CF resources, and speaking frequently to user groups throughout the country.

charlie@newatlanta.com

The Art & Science of Load Testing on a Budget

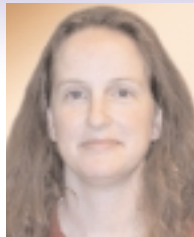
It's worthwhile to load-test before a crisis

Load testing your applications is much like flossing your teeth or taking out the garbage: you know you should do it; you know (vaguely, perhaps) that there will be consequences if you don't do it – and yet somehow it always seems to slip to the last item on the to-do list.

Until, of course, there's a crisis of some kind.

It's useful to reiterate why it's worthwhile to load-test before a crisis. In the first place, you want to be able to give yourself and/or your client confidence that the application you're building can handle the expected traffic – and the only way to do that is to actually simulate that traffic. It's a little too easy to assume that if an application performs well under "normal" testing conditions that it will behave the same way (if only, perhaps, a little slower) under higher-traffic conditions. In many cases, however, that's just not so.

Second, as a developer it's important to have confidence in your application. Are there any hidden problems in your code? Load testing will often find anything lurking in the corners that hasn't been brought to light by functional testing. This is obviously important for intangible reasons (who wouldn't like to brag about their application?), but it's also important for internal and external business reasons:



By Kelly Tetterton

- Does the application need to be re-designed in any way?
- Can the current database schema support the application under heavy load?
- Do the queries need to be optimized?
- Can this code base be reused for other high-traffic applications?
- Does the application meet current expectations? Will this application scale well for future growth?
- If it will scale, then what are the budget considerations for you and/or your client? Do you need additional hardware? Load-balancing software?
- Will this application make you and/or your client happy?

This last item may sound like an intangible, but it's really not; a happy client is often a source of recurring revenue.

And yet, despite all those very good reasons in favor of load-testing applications prior to launch, my own company, Duo Consulting, didn't do it. Budgets were always too tight, it seemed, to make this part of the project plan – and besides, doesn't everyone know that you have to be a very large corporation to be able to afford any load testing? As a small company, we felt that we just weren't equipped to do it.

The Crisis

The crisis inevitably came. One of our clients is a local park district that offers online registration four times a year. The number of online registrations had been steadily increasing each season, and in particular, the first few hours of each registration period were becoming increasingly problematic. At

9:00 a.m. on the first day of registration, parents all around the greater Chicago area were poised over their keyboards to try to get their children into a limited number of slots in the park district programs. Finally, this past year, the intense traffic became too much and our production server hung right in the middle of the heaviest registration period.

Load testing was now no longer optional; it was paramount. Could our application handle any kind of serious traffic, or was the recent registration experience somehow exceptional? Did we need to rewrite any major portions of the code? Did we need to throw more hardware at the problem? What were the limits of the application, and could we scale up in time for the next registration period? And how could we afford the actual load testing itself?

In desperation, we turned to Microsoft's Web Application Stress (WAS) Tool. It's a free tool that we could download immediately, and although it's not a high-end tool like SilkPerformer, it turned out to be just right for our needs. This shouldn't be taken as a final recommendation – there are many options out there, so there's likely to be something suited to your budget and/or platform. One quick place to look for a rundown of tool options is the Software QA/Test Resource Center at www.softwareqatest.com/qatweb1.html#LOAD. There are also services that will perform load testing for you, and we even briefly considered bringing in Macromedia to help us with analysis and load testing. The cost for such a service can be prohibitive (the Macromedia consult, for instance, would have cost almost \$15,000), however, and we felt that ultimately we were in the best position to analyze the site in depth. We already knew the code and the database architecture; we just needed the tools to get started.

That didn't mean we could start immediately. Setting up the environment and conditions for load testing – especially if it's the first time you've done it – does take a certain amount of time. First, we needed to set up a Web server that could approximate the production environment as closely as possible.

It's not necessary to seek perfection in this regard – it may be difficult to find a machine to dedicate to testing that has comparable hardware and memory to what's in production – but it is important to get as close to the production environment as possible. This means, at a minimum, making sure the code base is the same, the version of ColdFusion is the same (including any updates and/or hotfixes), the IIS settings are the same, the OS is the same (including any patches), the ODBC drivers are the same, and the ColdFusion administrator settings are the same.

If you do have hardware, memory, etc., that mimics your production site, that's even better – the closer you can get to your production environment, the more accurate and useful your tests will be. But don't despair if you can't create an exact duplicate; for instance, we didn't have an available machine that was the exact hardware and memory equivalent of our production server, yet our testing was still very, very useful.

You will also need to be able to point the site to a test database server (i.e., repurpose your development database). As far as your test clients are concerned, you should plan to set up multiple clients with WAS – one client machine will probably not approximate the kind of traffic you'll want to test against,

and beyond a certain point, you're testing load on the client machine rather than the Web server.

We initially tried setting up a single designated client machine that would run multiple clients, but we found that, at least with the WAS tool, we repeatedly risked hanging the client machine rather than the Web server we were trying to test against, which meant that we weren't really load-testing at all because the "load" wasn't always getting from the client machine to the Web server. Once we moved to a scenario where we had multiple machines running the WAS clients, we could see from the Web server activity that we were finally getting the kind of load testing we wanted.

Here are the general steps we followed after setting up the Web server itself and determining which machines would act as WAS client machines. Your mileage will almost certainly vary, but these are good starting points:

First, make sure your test site is set up so that it will be accessible by all your designated clients. What exactly this means will be largely determined by what you want to test. If you want to be able to load-test from both inside and outside your organization, then your setup will obviously be different than if you want to test solely from within your organization.

I'd suggest strictly internal testing at first; our experience shows that once you move to external testing, you're simultaneously testing your application and your clients' bandwidth, which makes it harder to narrow down what your actual problems might be. If you're testing internally – that is, from machines on your local network to machines on your local network – all the test machines are on a level playing field as far as bandwidth goes. Once you move to external testing you have additional variables to contend with, many of which may be outside your control: connection speed (dial-up versus broadband), service providers, etc. External testing – is certainly quite useful; it's just that it's probably not the first approach you should use.

It's also important to make sure that the URL for the test site is unique and doesn't conflict with any other versions of the site that you may have running, as you want to be certain you get clean data from your tests. In our case, we set up a domain that follows this convention: `preprod.[site_name].duoconsulting.com`.

Second, make sure all the machines involved in the testing process are time-synced. You need to be able to compare apples to apples, using the cleanest data possible – and that means getting log files from all the machines involved in which the timing of particular events and/or errors can be matched up easily.

Third, make sure the WAS clients are configured and set up properly on each machine. This is not necessarily as straightforward as it sounds. Although the WAS tool is very useful, the setup instructions are, as a colleague of mine put it, "written by developers, for developers."

In particular, I found two documents very helpful: Microsoft's "HOW TO: Install and Use the Web Application Stress (WAS) Tool" (<http://support.microsoft.com/default.aspx?scid=kb:en-us:313559>) and "HOW TO: Measure ASP.NET Responsiveness with the Web Application Stress Tool" (<http://support.microsoft.com/default.aspx?scid=kb:en-us:815161>). The installation article will walk you through the

IE configuration (don't ignore the proxy setting information – and know that for these purposes, “localhost” works where “127.0.0.1” does not). The second article provides tips on script configuration – in particular, why it's important to build in a warmup period and enable random delay. None of these items are intuitively obvious from the WAS tool itself, so be sure to read these articles, both of which are available on the Microsoft Web site.

WAS saves its scripting and report data in an Access database, so we found it useful to designate one client machine as the “parent.” The parent client was then used to create the scripts needed for testing, and that database was then copied to the other client machines. WAS runs as a Windows Service, so be sure to choose the File > Exit & Stop Service command from WAS after you're done constructing your scripts and before you attempt to copy the database.

Set Aside Some Time

Next, you need to reserve the time to do the testing. Again, this may not be as straightforward as it sounds. You first need to determine who will be involved, and if there are multiple people, make sure that they're available for the duration of your testing plan.

In our case, both I, as lead programmer (and the person most familiar with the application), and our systems administrator (who monitored the machines during the tests) really needed to be present. You should plan on the testing process itself taking longer than you really expect (especially on your first try). There will be stops and starts, unexpected results and delays, not to mention environment bugginess, so don't expect the process to be completed in a single day.

Another equally important time consideration is the availability of equipment and network resources – and the question of when you can abuse them. If the point of the testing exercise is to find the limits of your application, you'll need to be able to hang the machines involved in the testing (probably multiple times) without any major repercussions.

If you're sharing a development server with someone who has a project deadline to meet, don't plan on testing when it will interfere with that deadline. If you're using a staging server that other clients have access to, then make sure that they aren't caught off guard by your testing. Ideally, you'll have other options so you won't have to work around either of the scenarios outlined above, but you may not have that luxury. If you must work around other people using your test machines, be extraordinarily conservative about timing on these machines – don't schedule the usage back-to-back, because if a machine goes down because of testing it may take some time to get the machine back in shape for its other purposes.

You then need to prepare to capture data from your WAS client testing. This will depend on what you're looking to find, of course, but in our case we threw our net as wide as possible precisely because we weren't sure what we were looking for. So we gathered reports from the WAS tool, database traces (you can reach SQL Profiler from the Tools menu in SQL Enterprise Manager; see “HOW TO: Troubleshoot Application Performance with SQL Server,” <http://support.microsoft.com/default.aspx?scid=kb;en-us;224587>, for instructions on how to

enable this), and the Web server performance monitor. We found that gathering data from all three of these sources really gave us a full picture of what was going on with the application: the WAS reports from the client provided information on timeouts, socket errors, and hits/requests per second; the database traces allowed us to track longer-running queries; and the reports from the Web server performance monitor gave us insight on simultaneous users, queue request times, and average page response times.

Using these sources in conjunction with each other, we could really narrow down what was happening and when. You should plan on running far more tests than you might initially expect, so it's important to organize the data as well. We numbered each test, and all data captured corresponded to those numbered tests; we ran more than 60 tests over the course of several weeks, so this was crucial when it came time to prepare comparative reports.

It is equally crucial to prepare to record your anecdotal observations of each test as well. You may think that the data you capture will speak for itself, but that may not be true (or at least obvious).

For instance, we at first thought that our main data point would be the number of simultaneous users that the Web server could support, but it turned out that that particular statistic didn't really speak for itself; although according to the Web server performance monitor logs our early tests showed a high number of simultaneous users on the site (good), those same tests produced very high numbers of timeouts and socket errors from the WAS reports (bad), as well as very slow page response times from other portions of the performance monitor logs (very bad).

We would have had a far more difficult time figuring out what data we should be focusing on if we hadn't kept our own personal notes as well. We made notes as to whether the site was fast or slow or erratic, and at roughly what points during the test these things were happening. And again, this is especially important if you're doing lots of testing over a period of time: if you don't have your own notes about Test 4, you'll have a very difficult time comparing it to Test 61 three weeks later – or even finding good starting points for comparison.

Creating a Script

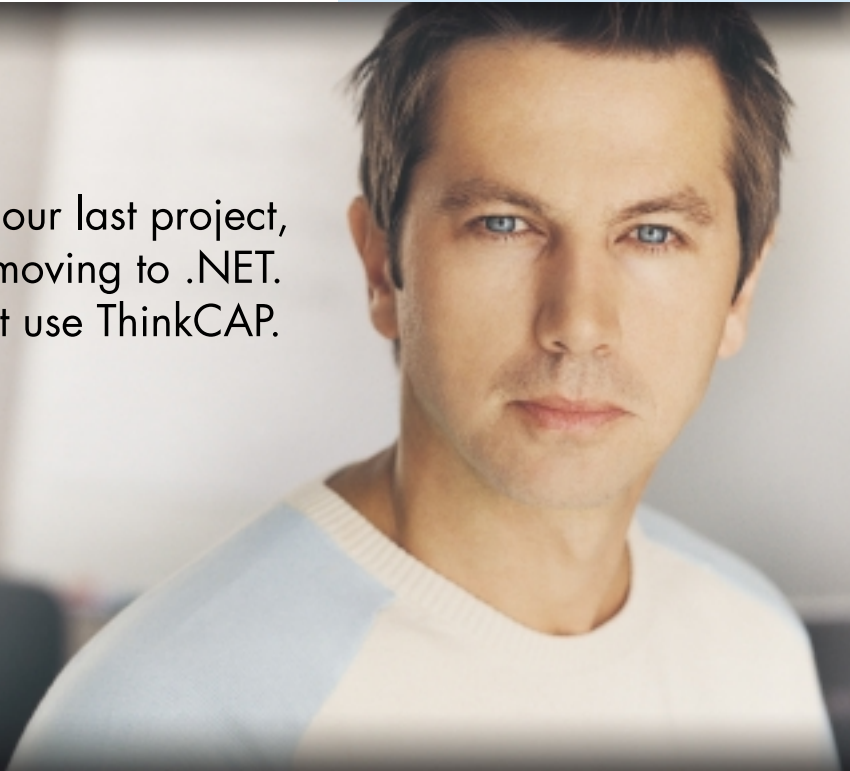
Finally, with the testing environment in place it's time to create a script. Again, what this means will vary depending on your needs. In our case, we recorded what we felt would be a fairly typical user session over the course of a couple of minutes, including what we thought might be the problem areas. Don't get too attached to the idea of the perfect script – it may be that you'll need several different scripts over the course of your testing process as you narrow down your problem areas.

You will probably also want to keep your initial scripts fairly short, and expand them only later. Our first scripts were only 10 minutes long (in other words, we were looping over our recorded script several times) – which was certainly more than enough to see where the weak points were, especially as we added more users to the mix. Longer, endurance scripts (ones you might run overnight or over the course of a weekend), should probably be employed only after you've squashed all

Think .NET development is more productive than J2EE?

Think **again.**

Due to delivery pressures on our last project,
we thought about moving to .NET.
I suggested we stick with Java, but use ThinkCAP.



Think **better.**

ThinkCAP[™]

ClearNova's ThinkCAP is a comprehensive application platform that simplifies and accelerates the development and maintenance of J2EE-based business applications by 50 to 70%.

ThinkCAP's visual & intuitive designers bring high productivity to business developers (those with VB or PowerBuilder-like skills), content owners, and administrators while allowing J2EE architects & programmers to leverage its component infrastructure and build business logic using the tools and approaches they prefer. ThinkCAP utilizes existing infrastructure, web services, legacy systems, and business applications.

ThinkCAP saves organizations time and money—and lowers project risks. Applications are written faster and require less maintenance. Project teams utilize in-house skills and require less training. Existing infrastructure and application servers are leveraged.

With ThinkCAP you can build quality applications faster.

Learn more about ThinkCAP at www.clearnova.com/thinkcap

CLEARNOVA
APPLICATION DEVELOPMENT • SIMPLIFIED • ACCELERATED

Highly Visual Development Environment

MVC Framework with Page Flow & Actions

Advanced Data Aware Controls:
Forms, DataViews, Queries, Navigations
Workflows, Graphs, Treeviews, Grids, Tabs

Smart Data Binding[™] to data, objects, XML,
sessions, or requests

Browser & server-side validation

Visual unit testing with RapidTest[™]

Service Flow Designer aggregates Web
Services, EJBs, XML, and POJOs

Content Management engine & tools

Supports .NET clients

Integrated, seamless security

Use any app server or 3rd party tool

the obvious bugs you've found in your short scripts; ideally, longer scripts can provide another, more realistic, benchmark for you, but only if you can run them without quickly hanging the servers involved.

After you've created the scripts you think you'll need, copy the Access database that holds those scripts from the parent client to all the child clients – that way, you're sure that everyone has the same script data, and that only the generated reports will be unique.

Testing

Once the clients are set and time is reserved for people and machines, coordinate your time and set the scripts running. You should plan on scaling your tests by adding increasing numbers of clients rather than heavier scripts. In other words, we found that it's better to progress your tests as follows:

- **Test 1:** 1 script on 1 client machine, 100 users x 1 thread
- **Test 2:** 1 script on 2 client machines, 100 users x 1 thread
- **Test 3:** 1 script on 3 client machines, 100 users x 1 thread
- **Test 4:** 1 script on 4 client machines, 100 users x 1 thread
- **Test 5:** 1 script on 5 client machines, 100 users x 1 thread

rather than trying to scale up testing with:

- **Test 1:** 1 script on 1 client machine, 100 users x 1 thread
- **Test 2:** 1 script on 1 client machine, 200 users x 1 thread (or worse, 100 users x 2 threads)
- **Test 3:** 1 script on 1 client machine, 300 users x 1 thread (or worse, 100 users x 3 threads)
- **Test 4:** 1 script on 1 client machine, 400 users x 1 thread (or worse, 100 users x 4 threads)
- **Test 5:** 1 script on 1 client machine, 500 users x 1 thread (or worse, 100 users x 5 threads)

Both scenarios look like they're testing 100–500 users, but if you follow the second scenario rather than the first you're very quickly going to be testing the limits of your client machine (CPU, in particular) rather than the application on your Web server – and your results will be skewed accordingly.

The number of users multiplied by the number of threads equals the number of sockets being created, and we found that creating 500 sockets on a single client machine just bogged down that machine; even the WAS Help notes that you should “be careful not to increase the stress level on the clients such that these boxes spend more time context switching between threads than doing actual work.” And the more threads you have, the more work your client machines will be doing simply switching between them. Obviously, if you have only a single client machine available to you, then your options are limited; just be aware that this will then be an additional factor in your testing.

With your first series of tests, you're really looking to get some initial benchmark scripts, conditions, and results for comparison purposes later. Those may come with the very first scripts you try, or it may take, as in our case, several attempts to get something useable for a baseline. When we began testing, for instance, we had lockups and crashes at alarmingly low user levels. We had to iteratively tweak the script until we eliminated some of our longer-running queries

from the script. We were not ignoring those problematic queries – we returned to tune them as soon as we could eliminate some of the other underlying problems we were seeing – but it wasn't useful in the beginning to try to slay all our dragons all at once.

Refining the Testing Process

You will also, inevitably, be refining the set of data you're going to focus on as the most important. As I noted above, when we began our testing process we assumed that we would be using the “average number of users on site” as our first and most important measure of comparison, because we knew (roughly) the number of users we needed to be able to support. But as it turns out, that particular set of data was far less helpful in measuring the user experience we were after than “average page response time.” Be flexible here: this is when you should carefully compare your results data with your anecdotal team notes.

So what exactly was our specific experience? As I mentioned above, we first spent some time tweaking our baseline test scripts, and we got some pretty horrible (if revealing) numbers. At 100 simultaneous users, the site performed just as expected – fairly fast page loads of just a few seconds. This was the “normal” mode for the site. However, at 500 simultaneous users from five separate client machines going against a single dedicated MX6.1 Enterprise server, we had:


- Average page response times well over 1 minute
- Average queue request times well over 1 minute
- Hundreds of timeouts and socket errors

This meant that if users were actually lucky enough to get in the queue to reach the site, they might be waiting a couple of minutes before getting any response. Obviously, we had to get those numbers down.

The first thing we did was to try to track down the worst offender – and that was clearly the database. We found that even with our short, basic scripts, eventually we would get database locks because we were using database-stored client variables. Because we had separated out our client variable storage for this application into a discrete database, we could easily see that there was far more activity there than we would have expected. Even though we had disabled the global updates to our client variable storage for the site, the application was still making unnecessary trips to the database server with each page hit.

Further research showed that in our particular instance, we could very easily switch from database-stored client variables to cookie-only client variables. This may or may not be true for others: if you are storing a great deal of information in your client variables, then database storage is probably most appropriate. If you're not storing very much information (less than 4K) and cookies won't be a problem for the site – and you're prepared with a P3P policy – then using cookie storage for your client variables may be the way to go. Once we made the change to cookie storage, site performance increased considerably.

We could then revert our scripts to include some of the earlier problematic long-running queries; we had first excluded



Consumer
Products



Music



Fighting AIDS



Water

INTO

WHAT ARE YOU INTO?

At Macromedia, we're continually inspired by the passion of our customers. Visit "Into" to see their stories, or share your own. www.macromedia.com/into



Announcing Macromedia MX 2004:
New versions of Macromedia® Flash™, Dreamweaver®,
Fireworks® and Studio. [Run with it.](#)

Copyright © 2003 Macromedia, Inc. and its licensors. All rights reserved. Macromedia, the Macromedia logo, Dreamweaver, Flash, Fireworks, and Macromedia Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.

them by looking at our initial scripts and comparing the database traces with the lines in the script that seemed to correspond to those queries, and then simply deleting those page calls from the script. We reran the tests with the modified scripts (that is, with the page calls added back in), capturing the database trace as we did so. We could then easily identify those queries that ran most often, as well as those that grabbed the most database CPU.

This tracking really gave us bang for our buck – we were able to identify just a few problem queries and concentrate our efforts on those. We optimized those queries as much as we could, and even devised a new caching strategy to eke out more performance gains. By this time, we could see the following numbers for 500 simultaneous users on the same machine:

- Average page response times under 20 seconds
- Average queue request times under 20 seconds
- No timeouts, and only a few socket errors

Although this was a significant improvement over where we had started our testing, it still wasn't going to meet the needs of our client, so we then set up a load-balanced environment and reran our tests. The load-balancing environment we set up was a combination software-hardware solution: we used

had eased significantly, but that the Web server CPU usage then rose to over 70% during certain portions of the scripts. Fix one problem, and the application bottlenecks somewhere else.

Since the process is so iterative, you'll have to clarify with your team fairly quickly what your specific endpoint will be. Of course, it has to be realistic – our client initially wanted to be able to support an entire season's possible registrants all at once, potentially 75,000 simultaneous users, which, given the budget and the actual needs of the site, didn't make sense (it had never experienced more than 1500 simultaneous users). It should be noted that upon reflection our client agreed to more realistic goals.

Even with realistic goals, however, it would be very easy to load-test yourself out of existence if they're not specific enough, because there's always more testing and tweaking that you could do. At some point, you and your team will need to decide something along the lines of, "we will tune the application so that all pages respond within 2 seconds when there are 500 simultaneous users on the site." In our case, we ultimately wanted to reduce the average page-response time and reduce the average queue time so that we could reach that 2-second goal. But whatever your particular goal is, once you get there, stop the testing.

"Load testing will often find anything lurking in the corners that hasn't been brought to light by functional testing"

additional machines controlled by load-balancing software from Coyote Point. Again, there are many other options possible here, including setting up multiple instances of ColdFusion and load-balancing between those instances. Not surprisingly, load balancing brought us significant gains as well. And because we had run the earlier tests, we also got a fairly good sense of how much gain we would get with each additional machine (Web server and ColdFusion server) – and we could then project how many additional servers we would need to add to reach the goals that we and the client had set together.

The Nature of the Beast

As you can see from just the short summary above, our testing was a highly iterative process, run by art at least as much as by science. In part, this is the nature of the beast – it takes a certain amount of trial and error before you hit upon the right problems and their corresponding solutions. But this also happens in part because as you refine your application environment, the source of your problems will change.

For instance, in our first tests the database CPU was maxing out during most of the script, but the Web server CPU would hardly ever rise above 10%. Why? Because of the client variable problem – it was overloading the database so much (as well as frequently locking it up) that the Web server didn't have that much to work with. Once we eliminated the client variable problem we could see from the traces that the database usage

Recode, Retest, Relaunch

The testing, after all, is just the first part of what you need to do. Now you have a game plan for refining the application or the database, or both, but you still need to recode, retest, and relaunch (or launch) the site. And that, obviously, takes time. So again, be cognizant of any looming deadlines so that the initial load-testing phase doesn't take up so much time that you won't be able to improve your production application. Once we got reasonably close to our 2-second page-load goal with our internal testing, we stopped our testing and did the actual recoding and regression testing we needed to do before relaunching the application.

Once we had recoded and relaunched our application, we did one final set of load tests – first, to verify our expectations; and second, to allow the client to experience the site while we load-tested. This second reason may seem like an after-thought, but it's not.

Remember that one of the main goals of load testing is to establish client confidence. Although we had been reporting our progress to the client throughout the process, this would be the first time for them to actually experience the faster version of the site. There's nothing that will establish confidence like setting up a test scenario and having your client experience the site at the same time. Having said that, be prepared for slightly different results than you may have had with strictly internal testing – because again, you'll also be testing band-

width limitations, which throws another set of variables into the mix.

We set up specific times for our external, preproduction load tests, and let our client know ahead of time when those would be. As a result, many members of the organization were able to use the site while we were load-testing. They knew what to expect, they could see where the weak points were, and they could clearly see that the site performed better. We got client buy-in – and that's invaluable.

Going Live


The day of reckoning finally arrived – the next registration period. But this time things went smoothly. In fact, things went even better in production than in some of our final load tests, partly because we had constructed our tests so conservatively, and partly because of the low latency over the network during our internal tests (which created many more requests per unit of time). Not only were there no crashes in production, but the site performed without any slowdowns, even when we were processing nearly 300 orders a minute with well over 500 simultaneous users:

- The Web servers' CPU usage was consistently 10% or less.
- The database server used 15% or less of its CPU.
- Pages responded in well under 2 seconds, on average.

- There were 0 queued requests (and therefore, the average queue request time was 0!).

It was a completely different user experience, and both the client and the end users were very pleased with the results.

Conclusion

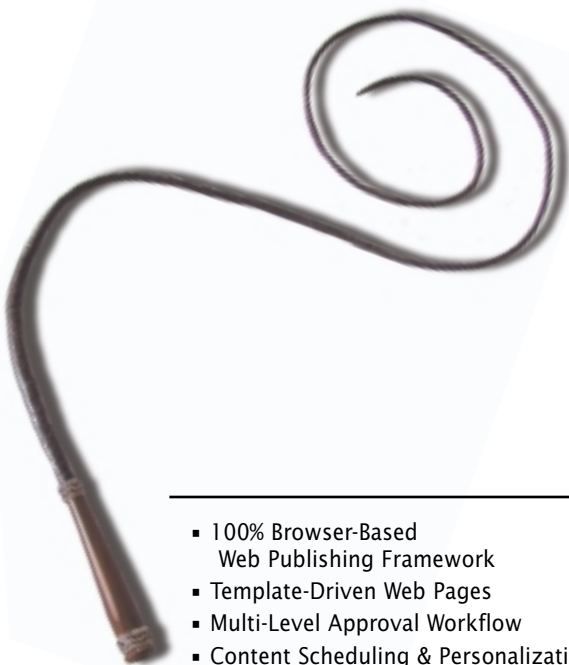
In the end, the load testing wasn't free, but the expense we incurred was worth it. There are many different options for testing, and I've discussed only a small number of the available tools and approaches here. You should review the tools and/or service options that seem best for your organization's needs and budget. For Duo Consulting, pursuing the load testing in-house with the necessary time, patience, and resources gave us client confidence, developer confidence, and a roadmap for scaling the application as usage increased. 

About the Author

Kelly Tetterton is the technical lead at Duo Consulting (www.duoconsulting.com) in Chicago and has been designing and programming for the Web since 1993. She is a Certified Advanced ColdFusion MX Developer with expertise in content management systems and Fusebox methodology.

ktetterton@duoconsulting.com

Content can be unruly. Make it behave with **CommonSpot™**



CommonSpot™ Content Server.

Put content management in the hands of content owners.

To tame your unruly web site, whip your content into shape with CommonSpot. With a rich, out-of-the-box feature set, you'll be up and running in weeks – not months. Built in ColdFusion, it's easy to integrate and customize CommonSpot to meet your requirements. CommonSpot's intuitive, browser-based interface makes authoring a snap, while its template and data-driven architecture and granular permissions allow for flexible yet powerful control.

Find out why CommonSpot was voted Best Web Content Management Tool in the CFDJ Reader's Choice Awards. Visit www.paperthin.com, or call today to schedule an online demonstration.

With CommonSpot, you'll have content tamed in no time.

- 100% Browser-Based Web Publishing Framework
- Template-Driven Web Pages
- Multi-Level Approval Workflow
- Content Scheduling & Personalization

- Rich Custom Metadata Support
- 508 Accessibility Compliance
- Seamless ColdFusion Integration
- More than 50 powerful, out-of-the box features

Paper Thin

800.940.3087
www.paperthin.com



Features Without Fixtures

onTap framework can save huge amounts of development time

If you've ever worked on a licensed application you know how prepackaged applications rarely meet enough of the client's needs to be implemented without modification. Whether the application is intended for your own use or someone else's, there's usually a feature or two missing, keeping the tools from being exactly what you need. Most scripted applications can't even be configured without manually editing an existing configuration file.

Although intolerable to desktop customers, this has become the de facto standard with Web applications. So you trudge along, laboriously configuring the application and making your modifications by manually editing another developer's original templates. Several months pass, and the original software author publishes a new version. You're excited to see new features that will make your life easier, and you're even entitled to a free upgrade as a part of your licensing. You want to upgrade, and you would... if only you had the time.

In the intervening months you've edited quite a few of the author's original templates and as a result, upgrading to the latest release will involve days or weeks of tedious examination to determine what you've

changed, how you've changed it, and how to ensure that your changes will integrate with the new version without causing problems. Although this may seem unavoidable, there is another way.

In August 2003 I published a new open source ColdFusion project called the onTap framework. Rather than belabor the merits of frameworks, I'm going to show you what makes the onTap framework unique. Specifically, I'm going to show you how to use the onTap framework to save hours, weeks, even months of development time by automating the integration of separate, often unrelated applications and components.

To use a publishing analogy, most scripted applications are like hardback books: if you want to add pages you have to strip the binding and rebind the book. That's a lot of work. In this article I show you how to design applications that resemble three-ring binders, wherein adding pages in the middle is the assumed norm.

A Brief Demonstration

To illustrate how the framework supports this idea of discrete, decoupled modification, I'm going to begin with a demonstration. I'll take a hands-on approach, showing you how to design and implement a site for a fictional magazine using the framework's HTML function library for modular display. Start with a copy of the onTap framework, which you can download from www.turnkey.to/ontap. Extract the onTap framework core archive into a directory where it will be accessible from the Web. In the framework's root directory, add these folders:



By Isaac Dealey

```

/_components/cfdj/_htmlhead/
/_components/cfdj/_local/
/_components/cfdj/index/_local/
/cfdj/

```

In the cfdj directory, add a template named index.cfm with only the following line of code:

```
<cfinclude template="#request.tapi.process()#">
```

Don't worry if this looks strange; it will all make sense very shortly. Open a browser and navigate to [http://\[your domain\]/ontap/cfdj/](http://[your domain]/ontap/cfdj/), where you just placed the index template. You'll see a generic message from the framework indicating that it couldn't find any content for the index page in this directory and providing a link to the framework documentation. Now, to create content for this page we're going to create two new templates. The first template, `/_components/cfdj/_local/100_defaults.cfm`, will contain the code shown in Listing 1.

What these lines do is to create several ColdFusion native structure variables that represent certain HTML elements on a Web page. What they *haven't* done is to display any HTML content. To display the content, create another template called `/_components/cfdj/index/_process.cfm`. In this template add only the following two lines of code:

```

<cfoutput>#htlib.show(view.main)#</cfoutput>
<cfdump var="#view.main#">

```

Refresh the browser and you'll see that now the page displays a menu and a few lines of text. Below the text you'll also see a ColdFusion dump of your content for debugging. You'll notice how all the content is sorted into structures and arrays. At first glance this may seem rather complicated. However, I'll show you how the onTap framework's architecture and the HTML library make using these structures easy and alleviate the problems with customization I mentioned earlier.

Unfortunately, the menu isn't very pretty (all the links run together) and the title text for the magazine doesn't look any different from the content. Let's fix that. Create a style sheet template in `/_components/cfdj/_htmlhead/100_default.css` with the following code:

```

#title { font-size: 20pt; margin: 10px; }
#menu a { margin: 5px; text-decoration: none; }

```

Refresh your browser. You'll notice that while you needed to create the style template, you didn't write any HTML to apply the style sheet to your document; the framework did this for you. Here's where it starts to get sticky. Our imaginary client sees this draft and decides they like how it's looking, but they don't want to display the home link on the home page. So now you need another page for testing purposes, to confirm that the menu is different on the home page.

Copy the template `/cfdj/index.cfm` to `/cfdj/members/index.cfm` and duplicate the directory `/_components/cfdj/index/` to the location `/_components/cfdj/members/index/`. Now you can view the page at [http://\[your domain\]/ontap/cfdj/members](http://[your domain]/ontap/cfdj/members) and you can see that it has the same content. To remove the home

link from the home page, create a new template named `/_components/cfdj/index/_local/100_menu.cfm` and include this code:

```
<cfset htlib.removeChild(view.menu,1)>
```

Refresh the home page, and you'll see that the home link is gone. To ensure this meets the client's requirements, however, we have to be sure that the link remains on other pages, so refresh the members page as well. If the home link is still visible on the members page, congratulations! You've just made a discrete and complex modification to a page with one line of code in a completely separate template.

The members link appears on the members page, however, and since the client didn't want the home link on the home page, they probably don't want the members link on the members page either. Add another template in `/_components/cfdj/members/_local/100_display.cfm` with the following code:

```

<cfset htlib.removeChild(view.menu,2)>
<cfset htlib.replace(view.main,
    "CF Dev Journal","CFDJ Members")>

```

Refresh the members page. You'll notice that in addition to removing the members link you've also changed the title of this page. Looking at the ColdFusion dump below the content, you'll notice that the function `htlib.replace()` takes the outermost structure as its first argument, followed by the string to replace and the new string. In this case I know the string "CF Dev Journal" appears only once in the content, although if I wanted to be more efficient and precise, I could instead pass only the title object to the first argument of the `htlib.replace()` function.

Establishing a Brand

There are myriad other ways in which we could modify the display in this example, such as highlighting the active page in the menu instead of removing it. Right now I'd like to demonstrate one more way in which this display can be discretely modified, which is to brand it. Imagine that our client (who shall remain nameless) has decided to market their site as an ASP application to other clients. One of the clients comes to you and says they don't like the layout. Instead of having the menu centered across the top, they want it vertical, along the right side of the page.

In most cases a client with such a request would be told either that it's simply not possible or that the change will require a lot of time and money. I'm going

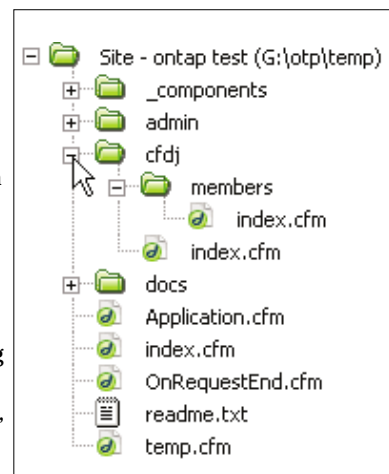


Figure 1: A site containing a members directory logically separates content available only to members from content available to the general public

to show you how to make this change quickly and discretely, using all the same code as the rest of the clients, without editing any existing templates and without affecting any of the other ASP clients.

Using much the same architecture, which allows individual pages to be discretely modified, the framework allows sites to be branded by domain. A branded domain may display different style sheets, images, and even content for each client. The framework maps each domain to a directory, where it looks to find custom code for branding. Custom templates for the `www.widgets.com` domain, for instance, would be stored in `/_components/_brand/www_widgets_com`. If your domain name includes a hyphen it will also be converted to an underscore in addition to the dots. Using this convention, create a new template in `/_components/_brand/[your domain]/cfdj/_local/100_menu.cfm` with the following code:

```
<cfscript>
    htlb.style(view.menu,"float","right");
    htlb.style(view.menu,"text-align","left");
    htlb.addChildContainers(view.menu,"div",true);
</cfscript>
```

Refresh the page again. You'll see now that the menu is almost magically transported to the right side of the page. Again, to ensure the clients' requirements are met, you need to view the same page from a different domain (or IP address). Declare this project a success if the menu remains centered across the top for all other domains.

Especially as the size of a site or application grows, the ability to make wide-sweeping, yet discrete changes to the content and layout in this manner can save your sanity – as well as allow you to accommodate a lot of client requests that otherwise would not be reasonable. If your clients are anything like mine, I recommend becoming as familiar as possible with the Cascading Style Sheets (CSS) standard as well. Using style sheets for as much of the display as possible will not only reduce the load on the ColdFusion Server but also greatly reduce the amount of code required to achieve clients' desired look and feel.

Under the Hood

You've probably already guessed that the onTap framework is automatically including files in different directories, since you didn't have to type any `cfinclude` tags to benefit from code placed in new templates in different directories. You might wonder why these templates have names like `100_default.cfm` or `100_default.css`.

The onTap framework's principles were generated from the simple idea that most of us weren't getting as much out of the `cfinclude` tag as we could. Usually it's just a static relative path to a template containing static HTML for a header or a footer template for layout. Though this approach is a bit better than having all static templates, it doesn't take advantage of all the power a dynamic content server like ColdFusion has to offer.

Knowing that variables can be used in the template attribute of a `cfinclude` tag, I then connected this idea of dynamic inclusion of templates with the query returned from the `cfdj`-directory tag's list action. If you could get a list of all the templates in a given directory, then you could include all those templates without necessarily worrying what might be in the directory. There would have to be a pattern for determining which files should be included and in what order.

Obviously, you don't want to attempt to include a PDF document in the middle of a ColdFusion page if someone happened to put one in the same directory, so by default the onTap framework includes only files beginning with two or more digits and an underscore and ending with the `.cfm` extension. The files are included in alphabetic order so that the number at the beginning of the file works like an old-fashioned line number.

Two additional breakthroughs turned this quirky idea into a real framework. First is the idea that any given page request can be divided into several logical stages such as beginning, middle, and end. The onTap framework divides each request into seven stages, including `_application`, `_htmlhead`, `_local`,

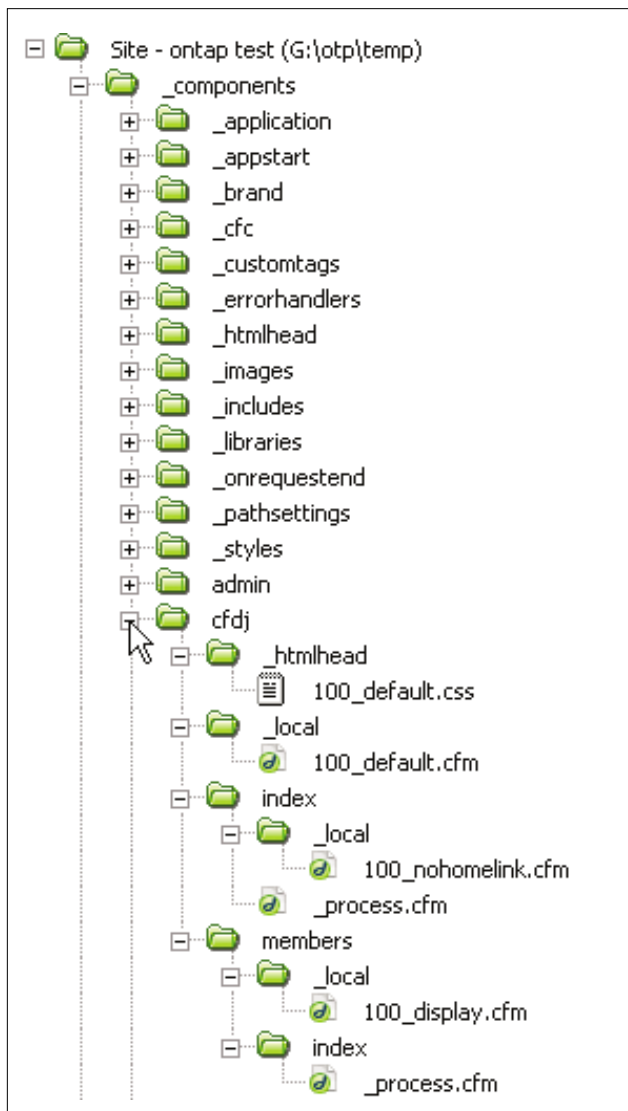


Figure 2: The onTap framework is designed to use your existing site or application structure

NORM MEYROWITZ
PRESIDENT OF PRODUCTS



WEB DEVELOPERS ARE USING OUR MX PRODUCTS
IN WAYS WE NEVER DREAMED. JUST IMAGINE
WHAT THEY'LL DO WITH THE NEW MX 2004.

I've been endlessly amazed by the things our customers have done using our MX generation of products. And with all the new features in Studio MX 2004, it's going to be even easier and faster for them to realize their visions.

Dreamweaver MX 2004 helps you get that picture in your head turned into a web site faster than ever. That should be welcome news to the millions of web professionals who use Dreamweaver to create sites and applications. We've added things like CSS support, target browser check and improved code hinting to help you get through projects in far less time. And with Fireworks MX 2004 you can optimize web graphics up to 85% faster.

The new Flash MX Professional 2004 takes our industry standard tool for building rich content and applications to a whole new level. It really helps stretch your abilities—no matter where you fall on the designer-developer continuum. For development, we've added things like data-aware components and an extensibility layer so you can add your own features. For design, we've added high-quality, long-format video that's really impressive.

These are just a few of the new features. The products are available individually, but they really work well together in Studio MX 2004. Don't take my word for it. Download a free trial and read more at our web site.

Let me know what you dream up. norm_01@macromedia.com

Copyright © 2003 Macromedia, Inc. and its licensors. All rights reserved. Macromedia, the Macromedia logo, Dreamweaver, Flash and Macromedia Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.
*Offer valid for English, French and German products only.



Introducing
Macromedia MX 2004.

www.macromedia.com/go/2004



_layout (header), [content], _layout (footer), and _onrequestend. The templates for any given stage are placed in a separate subdirectory, and the names of the directories begin with an underscore to separate framework-stage directories from other subdirectories.

The remaining breakthrough involved the realization that most developers already handle most of the work of structuring an application. They do this by creating logical directory structures. For example, a site containing a members directory logically separates content available only to members from content available to the general public (see Figure 1). A members/forum directory then separates forum content from other members-only content. Each subdirectory is a logical child of its parent directories, so with this in mind the onTap framework is designed to use your existing site or application structure instead of forcing you into a new set of conventions regarding site structure (see Figure 2). This is accomplished by placing the set of related stage directories within the directories for your logical site or application areas. So, for instance, if your members forum needs its own style sheet, that template would be located in the directory /members/forum/_htmlhead.

Because it's theoretically possible for a malicious visitor to

execute an individual template within a directory, there must be a way to protect your code. For instance, if any visitor can execute the template /members/forum/post/100_db_insert.cfm without first executing /members/_htmlhead/100_authenticateuser.cfm, a malicious user might be able to insert content into your forum without signing in or even registering as a member. This is the reason for the _components directory.

All of your site's or application's publicly accessible templates and directories are placed in the application root directory, and all of your code is placed in mirrored templates and directories in the _components directory. Code in the _components directory is protected by an Application.cfm template that prevents any template in these directories from being accessed directly within a browser. Templates in the _components directory are then accessed by the include tag:

```
<cfinclude template="#request.tapi.process()#">
```

The function request.tapi.process() returns a relative path to a custom tag that handles all the particulars of determining which content is included and in what order (with the exception of _application and _onrequestend stage code, which are executed in the application's corresponding Application.cfm or OnRequestEnd.cfm template). For example, when visiting the

LISTING 1

```
<cfscript>
    htlib = request.tapi.html;
    view = structnew();
    view.main = htlib.new("div");
    htlib.style(view.main, "text-align", "center");
    view.menu = htlib.new("div", "", "menu");
    view.title = htlib.new("div", "", "title");
    view.content = htlib.new("div", "", "content");

    htlib.childAdd(view.main, view.menu);
    htlib.childAdd(view.main, view.title);
    htlib.childAdd(view.main, view.content);
    htlib.childAdd(view.title, "CF Dev Journal");
    htlib.childAdd(view.content, "Hello World");

    mnu = structnew();
    mnu.home = htlib.linkNew("Home", "cfdj/", "T");
    mnu.members = htlib.linkNew("Members", "members/");
    mnu.history = htlib.linkNew("History", "history/");
    mnu.contact = htlib.linkNew("Contact Us", "talk/");

    htlib.childAdd(view.menu, mnu.home);
    htlib.childAdd(view.menu, mnu.members);
    htlib.childAdd(view.menu, mnu.history);
    htlib.childAdd(view.menu, mnu.contact);
</cfscript>
```

Listing 2

```
/_application/
/members/_application/
```

```
/members/forum/_application/
/members/forum/index/_application/

/_htmlhead/
/members/_htmlhead/
/members/forum/_htmlhead/
/members/forum/index/_htmlhead/

/_local/
/members/_local/
/members/forum/_local/
/members/forum/index/_local/

/members/forum/index/

/members/forum/index/_onrequestend/
/members/forum/_onrequestend/
/members/_onrequestend/
/_onrequestend/

Listing 3
/_application/
/_brand/domain/_application/
/members/_application/
/_brand/domain/members/_application/
/members/forum/_application/
/_brand/domain/members/forum/_application/
/members/forum/index/_application/
/_brand/domain/members/forum/index/_application/
```

Download the Code...

Go to www.coldfusionjournal.com

page [http://\[your domain\]/ontap/members/forum/index.cfm](http://[your domain]/ontap/members/forum/index.cfm), the onTap framework will automatically include all the serialized templates in these _component subdirectories in the order shown in Listing 2.

Branded subdirectories for a specific domain are interleaved with these directories, so to take the _application stage, for example, if you've added templates and directories to brand this stage, they will execute in the order shown in Listing 3.


This structure provides many advantages over writing contiguous ColdFusion base templates. When using a code versioning system you're much less likely to experience conflicts or bottlenecks involving the checking in or checking out of templates because each template is so small and controls a discrete portion of the application. By using the branding framework it's possible to simultaneously develop and test multiple displays or multiple approaches to the same technical problem using a consolidated code base (no more copy-and-paste development). Most important, if you develop large or "enterprise" applications, other developers or other companies can develop extensions for your applications that can be installed without editing any of your templates.

Currently a Plugin Manager module and a Login module are available for download from the onTap framework site. By the time this article is published I hope to see an additional security module on the site. These are the beginnings of a col-

lection of common tools that can be easily combined in new projects without the need for any programming to install and configure them.

Final Thoughts

That's all there is to it. Now you know how to build basic applications using the onTap framework. You know how to structure these applications so that they can be customized and extended without being edited. You know how to customize someone else's application without editing or overwriting their templates. You know how to brand applications for clients without impacting their neighbors. Only one question remains: Which features will you set free?

For more information about the onTap framework, visit the Web site at www.turnkey.to/ontap and join the onTap framework mailing list at www.houseoffusion.com/cf_lists. 

About the Author

Isaac Dealey has worked with ColdFusion since 1997 (version 3) for clients from small businesses to large enterprises, including MCI and AT&T Wireless. He evangelizes ColdFusion as a volunteer member of Team Macromedia, is working toward becoming a technical instructor, and is available for speaking engagements.

info@turnkey.to



HostMySite.com

Built for ColdFusion Pros by ColdFusion Pros

- 24 / 7 / 365 Phone Support
- 99.9+% Uptime
- Macromedia Alliance Partner
- "Full Control" Panel
- CFMX 6.1 or CF 5.0
- SQL Server 2000 or 7.0
- Custom Tags Welcome

plans from

\$8.95 / mo.

FREE Domain Name*
FREE Setup
FREE 2 Months

Visit www.HostMySite.com/cfdj for:

2 Months Free

FREE Setup and FREE Domain Name

*on any annual shared hosting plan

call
today

877•248•HOST
(4678)

Adventures in Encapsulation

PART III

Security in Fusebox 4

I recently gave a class in Fusebox 4 during which the issue of security came up. This issue seems to cause a good deal of trouble for programmers – so much so that when we wrote Fusebox 4 we provided some built-in abilities to help programmers with this. In this article I'll discuss the way security is handled in Fusebox 4 and note the difference between permissions and roles.

I find that a lot of the trouble people have with security comes from a misunderstanding of the difference between permissions and roles. An analogy may help. A permission is like the key to a door: either you have one or you don't. At the level at which the key is tried in the lock, the role the user is in is immaterial; what counts is whether he or she has the proper key.

A role (or a member in a user group) is used to facilitate the management of users and permissions. Since most of us use a database to store security information, let's look at one possible schema (see Figure 1).

While admirable for its simplicity, such a design will make for a lot of work for administrators: each user must have permissions assigned to him or her individually. A better plan is to assign permissions to user groups and then to assign individuals to user groups. The schema for this design is shown in Figure 2.

While this schema makes life for administrators much easier, it fails to capture individual permissions. For example, an individual might have, in addition to permissions gained from membership in a user group, other permissions specific to him or her. To provide for both ease of use and flexibility, we need a third schema (see Figure 3).

Now the user can be assigned to a user group and gain permissions by virtue of membership in a group and/or be assigned individual permissions. The total "permissions profile" of an individual user comes from the sum of his or her user group permissions and individual permissions.

Wait – I see a hand in the back. What's that? What happens if we want to deny an individual specific permissions? For that, we introduce our final data schema (see Figure 4).

With this plan, the permissions profile of an individual is the sum of his or her user group permissions and individual permissions minus any anti-permissions.



By Hal Helms

While databases are good for storing information, we want a more immediate way to associate a permission profile with a user within our application. There are many ways to do this. One popular means is by having a list of permission IDs attached to the current user, sometimes called "list-based permissions." This string can get quite long. If you dislike such verbosity (as I do), you might want to provide a way of encoding this information in a more succinct format. Following are two such plans.

In "bit-based permissions," permission IDs are assigned from the set of numbers generated by the powers of two. Here is an example in code:

```
<cfset deleteDoc = 2^0 />
<cfset editDoc = 2^1 />
<cfset createDoc = 2^2 />
<cfset readDoc = 2^3 />
```

These permissions are not hierarchical; having a permission to delete a document does not give you permission to read it.

To add a permission to a user, start the user out with a permission of 0:

```
<cfset user.permissions = 0 />
```

Then, add other permissions to the user:

```
<cfset user.permissions = deleteDoc + readDoc />
```

The variable, *user.permissions*, has a value of 9. To check whether the user has the required permission, use the *BitAnd()* function in ColdFusion:

```
<cfif BitAnd(user.permissions, deleteDoc) GT 0>
    Come right in!
<cfelse>
    No way, pal!
</cfif>
```

If bit functions are new to you, they might need a word of explanation. The *BitAnd()* function works by comparing the bits that make up binary numbers. For example, the decimal number, 13, is represented as a binary 1101. This can be thought of as having the "one," "four," and "eight" bits turned "on," while the "two" bit remains "off." *BitAnd()* checks to see if both corresponding bits in the two numbers are nonzero values, so comparing decimal 13 to decimal 4 is just a matter of checking the value of the

"four" bits for both numbers. Here, they are both off, so BitAnd() returns 0. BitOr(), another ColdFusion function, checks to see whether either of the bits is on.

To integrate user groups into this, add permissions to a user group:

```
<cfset registeredUsers = readDoc + editDoc />
```

Then assign the user to a group:

```
<cfset user.permissions = registeredUsers />
```

What's that? Another hand? Ah, you want to know what happens if a user belongs to several user groups. Good question. Here we can't just add the user groups together. As an example, let's create another user group and give it some permissions:

```
<cfset superUsers = deleteDoc + editDoc + createDoc />
```

The variable, *superUsers*, has a value of 7. If you attempt to add the user to these groups by simple addition (*user.permissions = registeredUsers + superUsers*), things go wrong. Your user will have a new permission (16) that doesn't exist, while losing all previous permissions!

Another ColdFusion function can help us:

```
<cfset user.permissions = 0 />
<cfset user.permissions = BitOr(user.permissions, registeredUsers) />
<cfset user.permissions = BitOr(user.permissions, superUsers) />
```

The value of *user.permissions* will be set to 15, meaning that they have permission to delete, edit, create, and read a user. The BitAnd() test will now work properly.

There is one possible problem with using this bit-based security: you are restricted to 32 permissions (since ColdFusion works with 32-bit numbers). One simple way to solve this is to assign permission values based on their position in a string. The code begins like this:

```
<cfset deleteDoc = 1 />
<cfset editDoc = 2 />
<cfset createDoc = 3 />
<cfset readDoc = 4 />
```

The "permission string" for a user with no permissions would look like this: 0000. Adding a permission to read a document would turn the fourth position on: 0001. Having permission to read a document and delete a document would give us 1001. One caveat: the permission string must be stored as a string. If it's saved as a number, the leading zeros will be lost. This is the code to test whether a user has a permission to read a document:

```
Mid(user.permissions, readDoc, 1)
```

These are three ways to build permission profiles. Regardless of how you create yours, Fusebox can help you in testing for permission. This is accomplished by a *permissions* attribute that can be set in the *circuit.xml.cfm* file. Here is an example:

```
<circuit access="public">
  <fuseaction name="showAdminMenu" permissions="basicAdmin" />
</circuit>
```

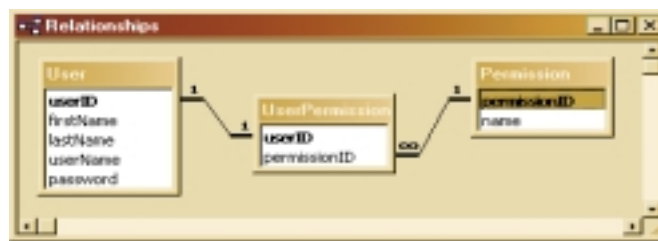


Figure 1: This schema will work but doesn't make for easy administration of permissions

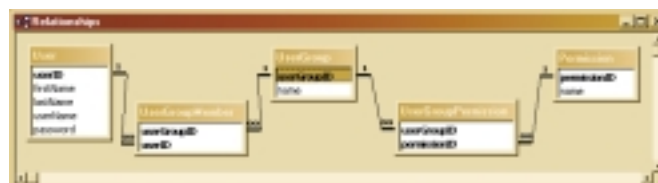


Figure 2: This schema makes for easier administration but fails to capture individual permissions



Figure 3: This schema is almost right



Figure 4: Now we've got it!

The permissions attribute defines the required permission to execute a fuseaction. Here, it is set to the string, "basicAdmin". By itself, setting the permissions attribute does nothing to implement security, but it can be used by a Fusebox plug-in to easily implement security.

Plug-ins are pieces of code that execute at various plug-in points within the Fusebox core code. Plug-ins provide a way to extend the core functionality of Fusebox without altering the core code. One of these plug-in points is *prefuseaction*. Plug-in code registered at this point will execute prior to every fuseaction – ideal for our use. Here is a simple *permissionsChecker* plug-in that will work with list-based permission profiles.

```
<!-- Permissions checker -->
<cfset fuseactionPermissions =
Application.fusebox.circuits[thisCircuit].fuseactions[myFusebox.thisFuseaction].permissions />
```

—continued on page 39

Making Decisions in Your Code with cfif

Using conditional logic in Web development

This month I will examine the cfif tag and discuss how we can use that tag to make decisions in our code. Along the way I'll talk about Boolean logic, decision operators, and Boolean operators in CFML.



By Jeffrey Houser

Understanding Conditional Logic

Before you can implement conditional logic in your code you have to understand what conditional logic is. I have no doubt that every one of you knows what conditional logic is, although you may never have heard it referred to as such. Conditional logic is just a fancy way of saying that you are going to make a decision. Usually the decision is based upon some sort of condition or the result of an operation.

You and I make decisions every day, so let's delve into some real-world examples. Suppose I'm driving a car and come to an intersection with another road. I need to make a decision whether to turn left, turn right, or continue straight on the main road. Each decision may have different consequences. If I turn right, I go to the bank. If I turn left, I go to the grocery store. Going straight might take me somewhere else, home, for example. Since it has been a long day, I want to go home. I'm going to drive straight. This is an example of conditional logic in practice. I made a choice to go straight, based on a condition, I want to go home, and something happened as a result of that condition.

Making decisions about groceries and the bank is all well and good, but how does this apply to Web development? I'm glad you asked. Suppose you have a sign-up form on your Web site. Your users enter some account information, such as an e-mail address, their name, a username, and password. You take that information and create a user account in your database.

What happens if the user does not enter a username? Can you still create the account? How do you handle that situation? You can use conditional logic to make sure that the username is not left blank. Suppose a user looks at the shopping cart before adding items to it? You can use conditional logic to display an "empty cart" message. Suppose a user enters a birth date as November 15, 2050? You get the idea – we can use conditional logic in our code to solve these types of problems.

The Format of the cfif Tag

In CFML, you can use the cfif tag to perform conditional logic statements. It comes in this format:

```
<cfif expression>
  Perform Actions
</cfif>
```

The cfif tag is an oddball tag in the CFML language. Unlike most tags, it doesn't take any explicit parameter name and value arguments; it is simply followed by an expression after the tag name. A review of expressions can be found in last month's article; or you can go straight to the source: http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/cfml_b14.htm. The CFML interpreter uses an expression to determine whether or not to process data. Expressions are literal values, variables, or functions (which return a Boolean value).

The expression inside the cfif tag must evaluate to a Boolean value (a true/false). Boolean values are represented as either "true" or "false", "yes" or "no", or "0" or any number. If the expression evaluates to true, then the action immediately following the <cfif> is performed. If the expression evaluates to false, then the action is not performed and template execution will continue after the end cfif tag. There are two types of operators that are used with cfif tags: decision operators and Boolean operators. Both evaluate to Boolean values.

Decision and Boolean Operators

The two types of operators used in a cfif expression are decision operators and Boolean operators. Boolean operators refer to specific operators from Boolean algebra. Decision operators are used for comparisons. This is a list of the Boolean and decision operators supported in CFML:

- **NOT:** A Boolean operator that returns the opposite of the specified expressions. If you want the NOT of true, then you get false. The NOT of false is true.
- **AND:** A Boolean operator that returns true if both operands are true, but false otherwise.
- **OR:** A Boolean operator that returns true if at least one of the operands is true, but returns false otherwise.
- **XOR:** A Boolean operator that performs an exclusive or. If only one of the values is true, then the result is true. Otherwise the result is false.
- **EQV:** A Boolean operator that performs equivalence.

Returns true if both operands are true or both operands are false. Returns false if the operands are different.

- **IMP:** A Boolean operator that performs implication. It is akin to the logical statement "If condition 1, then condition 2." This returns false if condition 1 is true and condition 2 is false. True is returned in all other cases.
- **IS, EQUAL, EQ:** Decision operators that test for two values being equal. If the first value is equal to the second value, the final result is true.
- **IS NOT, NOT EQUAL, NEQ:** Decision operators that test for two values not being equal. If the two values are equal, then the final result will be false.
- **GREATER THAN, GT:** Decision operators that are used to check if the first value is greater than the second value.
- **GREATER THAN OR EQUAL TO, GTE, GE:** Decision operators that check to see if the first value is larger than or equal to the second value.
- **LESS THAN, LT:** Decision operators that are used to check if the first value is smaller than the second value.
- **LESS THAN OR EQUAL TO, LTE, LE:** Decision operators that check to see if the first value is smaller than or equal to the second value.
- **CONTAINS:** A decision operator that returns true if the value on the left contains the value on the right.
- **DOES NOT CONTAIN:** A decision operator that returns true if the value on the left does not contain the value on the right.

You can read more about decision operators in the Macromedia Livedocs at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/expresa6.htm> and about Boolean operators at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/expresa5.htm>.

It is worth noting the order in which these operations are applied. This builds right off the order of operations I discussed in last month's article. First, the arithmetic operators are applied, and then the string operator is applied. Following that, the decision operators are applied: EQ, NEQ, LT, LTE, GT, GTE, CONTAINS, DOES NOT CONTAIN from left to right in the order that they appear. Then come the Boolean operators. First the NOT operator is applied, then the AND operator. Then the OR operator, followed by the XOR operator. Next comes EQV, and finally, IMP. You can always use parentheses to change the order of operations. In fact, when in doubt, use parentheses to force the order of operations! In most cases you won't be mixing arithmetic or string operands with decision or Boolean ones; however, decision and Boolean operands are often used together. Let me show you how to put this into practice.

Take Some Code for a Test Drive

Suppose you want to make a simple interface for users to choose which way to turn their car. This sample uses an HTML form to collect the user input, as shown below. If you are unfamiliar with HTML forms, I found some great Web-based tutorials located at www.weballey.net/forms and www.wdvl.com/Authoring/Scripting/Tutorial/html_forms_intro.html.

**PREVIEWING MAY 11, 2004,
AT NETWORK+INTEROP, LAS VEGAS**

INFORMATION STORAGE + SECURITY JOURNAL!



**FOR MORE INFORMATION VISIT
WWW.ISSJOURNAL.COM**



From the World's Leading i-Technology Publisher

©COPYRIGHT 2004 SYS-CON MEDIA. ALL BRAND AND PRODUCT NAMES ARE TRADE NAMES. SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.

```
<form action="Go.cfm" method="post">
  <input type="Radio" name="Route" value="Right">Right
  <input type="Radio" name="Route" value="Left">Left
  <input type="Radio" name="Route" value="Straight">Straight<br>
  <input type="submit">
</form>
```

The form uses radio buttons to let the user select a direction and then click Submit to see the result page:

```
<cfif Route is "Right">
  Go to Bank<br>
</cfif>
<cfif Route is "Left">
  Go to Store<br>
</cfif>
<cfif Route is "Straight">
  Go Home<br>
</cfif>
```

If you copy this code to see the example in action, make sure that you put it in a file named go.cfm and put both files in the same directory. This code uses the decision operator IS to determine the value of Route. It uses multiple cfif tags to figure out the value of the Route variable and what it should be.

Specifying Additional Conditions with cfelse and cfelseif

Up to this point the cfif tags have performed actions only if an expression resulted to true. What happens when you need to perform some actions when a condition is false? Well, you could use the NOT operator to create a second cfif statement, but there is a better way. CFML offers us the <cfelse> tag. This is the syntax:

```
<cfif expression>
  Perform Actions
<cfelse>
  Perform Other Actions
</cfif>
```

The cfif tag remains the same, as do the actions that we perform if the condition is true. What changes is that we have a cfelse tag. The cfelse tag does not take any parameters or expressions – it simply offers the alternative code to run in case a cfif evaluates to false.

If you go back to the driving example, you will see that we have three separate conditions. Turning left directs us to the grocery store, turning right takes us to the bank, and going straight sends us home. In this case, it won't be possible to perform all of the actions at once. Our first example is inefficient because we evaluate all conditions even after finding the correct one.

There is a better way to handle this situation – through the use of a tag called cfelseif. The cfelseif tag allows us to add multiple conditions to our cfif statement. If the first condition does not evaluate to true, then the second cfelseif con-

dition is evaluated. If the second is not true, then the third is evaluated. Use of the cfelseif is optional, and there is no limit to the number of cfelseif conditions you can have. By contrast, the cfelse tag can be used only once. This is the syntax for cfelseif:

```
<cfif expression>
  Perform Actions
<cfelseif expression2>
  Perform Other Actions
</cfif>
```


The code is very similar to the previous sample. The cfelseif tag contains an additional expression, whereas the cfelse did not. It will be evaluated only if the cfif expression is false. When there are cfelseif tags and a cfelse tag, the cfelse tag must always come last (after all cfelseif tags). The previous example used multiple cfif tags. The cfelseif tag lets us combine the separate cfif tags into a single login. This is the code:

```
<cfif Route is "Right">
  Go to Bank<br>
<cfelseif Route is "Left">
  Go to Store<br>
<cfelse>
  Go Home<br>
</cfif>
```

If the route chosen is right, then we go to the bank. If the route is left we go to the store. If nothing else, we continue to go straight on home.

Where Do I Go from Here?

The cfif statement can get very complicated if you start using one cfif statement inside another cfif statement. If you want to do some independent study, I would suggest reading up on cfswitch at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-c10.htm#wp1103819> and cfcase at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-pa9.htm#wp2664410>. They are used together to form case statements, which are a different form of conditionals that you can use in your code. The use of cfswitch and cfcase statements is often a great way to simplify complex cfif.

It has been a pleasure writing these articles for you, and I'm pleased to announce that I've received my first topic request. Next month I will be writing about structures and arrays. Keep the ideas coming! 

About the Author

Jeffrey Houser has been working with computers for over 20 years, and in Web development for over 8 years. He owns a consulting company, and has authored three separate books on ColdFusion, most recently ColdFusion MX: The Complete Reference (McGraw-Hill Osborne Media).

jeff@instantcoldfusion.com

LOOK FOR YOUR **FREE...**





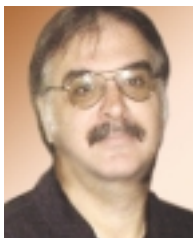
Do You Want Coffee with That Mojibake?

Character encodings and CFMX

This is the second in a series of articles on globalizing ColdFusion MX (CFMX) applications. This article examines character encodings and CFMX, BIDI (bidirectional text), the use of Cascading Style Sheets (CSS) in application globalization (G11N), and why we should all just use Unicode.

Space is limited so I'm going to assume that you've read the first article, which covered globalization concepts and terminology.

No, *mojibake* isn't a new kind of Krispy Kreme donut. Mojibake, or “文字化け” in Japanese, literally meaning “ghost characters” or “disguised characters,” is a term that has crept into the G11N field and is often used to indicate gibberish text that has become corrupted because of bad or missing character encoding. For instance, “文字化け” becomes the mojibake “☐\$BJ8;z2=\$1☐(J)” when the character encoding is messed up (I



By Paul Hastings

plucked this example from some e-mail correspondence). Yes, this issue arises often enough that somebody coined a term for it.

Why do we have to worry about these sorts of things? Pornography aside, text is by far the most commonly used data type in Web applications. It should be obvious then that it's critical that people are able to understand the content your Web application is delivering (otherwise what's the point?). The key to this is making sure Web applications, Web servers, database back ends, and users' browsers are all in agreement regarding character

encoding. With that in mind, the purpose of this article is to:

- Explain what character encodings are
- Provide some background to the more common character-encoding issues
- Explain the sometimes tricky business of BIDI text
- Indicate how CSS can help you develop G11N Web applications
- To convince you, come hell or high water, to just use Unicode rather than try to deal with all the various character encodings on a case-by-case basis.

One thing that makes understanding these issues difficult is the plethora of oftentimes conflicting terminology in use today,

even within some of the “standards” bodies. While I’m reasonably certain that someone, somewhere, will object, I think the terms I chose to use here are common and plausible enough for the purposes of this article.

What Are Character Encodings?

Let’s begin by dissecting, in a simple-minded way, human language into its component parts, beginning with the simplest, characters.

Characters, Glyphs, and Other Sea Creatures

I suppose it might be useful to think of a character as an “atom” within a “molecule” of text content like a word. But you really have to think of a character in the abstract, as an entity without regard to its appearance (“a”, “a”, “a”, or “a” – it’s still an “a”). The Unicode Consortium (www.unicode.org) defines an abstract character as a unit of information used for the organization, control, or representation of textual data. The Unicode Consortium’s “character encoding model” (Unicode Technical Report 17, www.unicode.org/unicode/reports/tr17) defines three basic concepts:

A *character repertoire* is simply a set of distinct abstract characters; some folks refer to this as a “character set.” In practice, a character repertoire usually corresponds to an alphabet (your ABCs) or a symbol set (musical notation, for instance). Note that a character repertoire can contain characters that look the same in some presentations, such as Latin uppercase A and Cyrillic uppercase А, but which are in fact logically distinct. Once again, you need to separate the way a character looks from what it actually represents.

A *character code* is a mapping from a set of abstract characters to a set of nonnegative (but not necessarily consecutive) integers – the abstract character made real to computers, if you will. Each abstract character’s mapped integer is called its “code point.” For example, in Unicode the code point for “A” is 65; the code point for the first letter of the Thai alphabet is 3585.

A *character encoding* is a method or algorithm for presenting characters in a digital form by mapping sequences of code numbers of characters to sequences of bytes. For example, in the MS-874 (Thai) encoding “เ” has a code point of 161; that same code point is assigned to “ı” in the Latin-1 encoding (that’s an inverted exclamation point by the way) and “У” (Cyrillic capital letter short U) in the Windows Cyrillic encoding.

The visual representations of characters are called glyphs. You need to understand that text presentations, such as fonts, are applied to glyphs and not to the abstract characters. A font is a collection of glyphs. In practical terms, a font is a numbered set of glyphs, the numbers corresponding to code positions of the characters (represented by the glyphs). A font, at least in this sense, is entirely dependent on character code. It’s this dependence that often causes the appearance of boxes (□) or other strange characters in text streams – a browser’s fonts simply can’t render the requested character code because it’s not in that font, or more rarely, it violates some display rule. It’s therefore important to fully understand which character encodings are contained in which fonts. Test, don’t simply assume.

A script is a collection of related characters required to represent text in a particular language, for instance, Latin, Greek, Thai, Japanese, or Arabic. Note that one script might also be used in several languages. For example, Arabic is used in Pashto,

Urdu, Farsi (Persian), and of course Arabic. A writing system is composed of a set of characters from one or more scripts that are used to write a particular language. A writing system also includes the rules that govern character presentation. For exam-

LANGUAGE	CHARACTER ENCODING(CODE PAGE)
Afrikaans (af)	iso-8859-1, windows-1252
Albanian (sq)	iso-8859-1, windows-1252
Arabic (ar)	iso-8859-6
Basque (eu)	iso-8859-1, windows-1252
Bulgarian (bg)	iso-8859-5
Byelorussian (be)	iso-8859-5
Catalan (ca)	iso-8859-1, windows-1252
Croatian (hr)	iso-8859-2, windows-1250
Czech (cs)	iso-8859-2
Danish (da)	iso-8859-1, windows-1252
Dutch (nl)	iso-8859-1, windows-1252
English (en)	iso-8859-1, windows-1252
Esperanto (eo)	iso-8859-3 (good luck finding browser support)
Estonian (et)	iso-8859-15
Faroese (fo)	iso-8859-1, windows-1252
Finnish (fi)	iso-8859-1, windows-1252
French (fr)	iso-8859-1, windows-1252
Galician (gl)	iso-8859-1, windows-1252
German (de)	iso-8859-1, windows-1252
Greek (el)	iso-8859-7
Hebrew (iw)	iso-8859-8
Hungarian (hu)	iso-8859-2
Icelandic (is)	iso-8859-1, windows-1252
Inuit (Eskimo) languages	iso-8859-10
Irish (ga)	iso-8859-1, windows-1252
Italian (it)	iso-8859-1, windows-1252
Japanese (ja)	shift_jis, iso-2022-jp, euc-jp
Korean (kr)	euc-kr
Latvian (lv)	iso-8859-13, windows-1257
Lithuanian (lt)	iso-8859-13, windows-1257
Macedonian (mk)	iso-8859-5, windows-1251
Maltese (mt)	iso-8859-3 (good luck finding browser support)
Norwegian (no)	iso-8859-1, windows-1252
Polish (pl)	iso-8859-2
Portuguese (pt)	iso-8859-1, windows-1252
Romanian (ro)	iso-8859-2
Russian (ru)	koi8-r, iso-8859-5
Scottish (gd)	iso-8859-1, windows-1252
Serbian (sr) cyrillic	windows-1251, iso-8859-5
Serbian (sr) latin	iso-8859-2, windows-1250
Slovak (sk)	iso-8859-2
Slovenian (sl)	iso-8859-2, windows-1250
Spanish (es)	iso-8859-1, windows-1252
Swedish (sv)	iso-8859-1, windows-1252
Thai (th)	tis-622, iso-8859-11, windows-874
Turkish (tr)	iso-8859-9, windows-1254
Ukrainian (uk)	iso-8859-5

Notes: ISO stands for the International Standards Organization, which among other things produces standards for character encodings. Windows refers to the character encodings produced for the MS Windows operating system by Microsoft. Source: W3C www.w3.org/International/O-charset-lang.html

Table 1: Typical character sets in various languages

ple, Thai has what are affectionately referred to as “jumping vowels” such as “เ” (*sala a*) as used in the Thai word for nothing “อะไร” (transliterated into English as *plao*), which jumps in front of the consonant “ป” (*por pla*) but is pronounced as if it didn’t (that is “*plao*” instead of “*aopl*”). Another example is the writing direction (left-to-right or right-to-left, for instance) a particular script uses – languages, don’t have a direction; only the scripts used in their writing systems do.

Quite often the choice of a character repertoire, code, or encoding is presented as the choice of a language, even though a language setting is quite distinct from character issues. There are, however, some more or less “natural” relationships between languages and character encodings. Table 1 shows a partial list of these.

There are several things to note from Table 1:

- The sheer number of character encodings
- The fact that the same character encoding is used in several languages.
- Many languages, Japanese for example, might be referenced by more than one character encoding – which can help compound the confusion provided by the previous point. I especially enjoy this one on projects with tight deadlines.

To me these all just spell trouble in a G11N application. It’s this kind of exuberant variety that causes mojibake and other headaches.

Common Character Encoding Issues

The large variety of character encodings means globalized applications based on them need to go the extra mile in order to implement them. This management effort must by

necessity extend from the back-end database through to the pages delivered to the client’s browser. This is quite a daunting task that could become expensive as well. In many instances character encoding-based applications deny the possibility of back-end database consolidation, that is to say that rather than a small number of databases to manage you could well end up with one database per character encoding. Depending on the database technology used it could also mean rolling out one Web server per character encoding (a common occurrence with desktop databases such as MS Access, for example). Obviously, economics eventually forces a database change, but often too late.

Variety also means choice. Languages with more than one character encoding are especially troublesome, as it’s generally impossible to forensically determine which encoding was originally used. You can end up with text data encoded in one character encoding but displayed in another. This happens quite often with text data that has passed through many sets of hands and the original character encoding metadata has been lost along the way. It can also occur when no character encoding “hint” is included in a Web page and a browser’s default doesn’t match the original character set. In HTML the hint is normally provided by the `charset` property in HTTP Content-Type header:

```
<META http-equiv=Content-Type
content="text/html;
charset=caveDwellingCodePage">
```

where `caveDwellingCodePage` is the character encoding you require. This should be declared as early as possible in the header section of your Web page. You should also note that the W3C has chosen to use `charset` as a synonym for

character encoding. For XHTML compliance you would simply add a slash to the end of that tag:

```
<META http-equiv=Content-Type
content="text/html;
charset=caveDwellingCodePage" />
```

While CFMX will happily ignore this meta-header, I would still urge you to include it for the sake of spidering robots and other content-indexing programs, as well as accessibility software. It also provides a hardcoded artifact as to what the original character encoding intentions were. In CFMX the `CFPRO-CESSEINGDIRECTIVE`, `CFCONTENT`, and `CFHEADER` tags (and the `SETENCODING` function) provide this hinting. XML hinting is usually done with an encoding pseudo-attribute in the XML declaration at the start of a document:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

There is also another, more subtle, character encoding pitfall. Some character encodings masquerade as related but when examined in detail are in fact not related. For example, the Windows Latin-1 character encoding is quite often mislabeled by Web developers as ISO-8859-1 on the Internet, but in actual fact it is a superset of ISO-8859-1. The extra characters provided by the Windows superset will confuse browsers that actually treat it as ISO-8859-1, whether you told them via `charset` hinting or it is simply handled as a default character encoding. It’s not just the Windows OS; the Mac OS also has a few similar issues. Its Roman character encoding is quite often labeled as ISO-8859-1 even though it predates that ISO encoding by several years. It does not have exactly the same character repertoire, and many of the characters it does share with ISO-8859-1 actually have different code points. Even the Mac Latin-1 or Mac Mail character encoding, an attempt at aligning the Mac OS Roman repertoire with ISO-8859-1, is not quite equivalent but it is very often labeled as if it were.

Finally, as most of the character encodings listed in Table 1 are codepage encodings and can contain only 256 code points, you cannot mix languages within the same text stream, as these

“After a particularly frustrating week dealing with character encoding issues, I was going have... ‘Just Use Unicode’ tattooed on my forehead”

encodings overlap (commonly in the last 128 code points). If you think this isn't a common occurrence, just look at this article; so far it has mixed Japanese, Thai, and English. Another point to consider, from an I18N perspective, is the good practice of allowing users to manually swap languages and to show their language choice (Thai) in that language (ไทย). It's the little things that count, after all.

BIDI Concepts

BIDI, or bidirectional text, can be somewhat difficult to understand, especially for folks used to text in one, usually left-to-right (LTR), direction. I can only skim the surface of this complex subject here (for instance, I'm going to skip clean over Arabic script's special ligature and shaping features, the so-called "national" or "Hindi" digit shapes, Hebrew's five "Final Form" consonants, and directionally neutral characters such as spaces and punctuation, among other things), but hopefully it will be enough for a basic grasp of BIDI issues. Why bother if it's so complicated? Because more than 500 million people in the Middle East, Central/South Asia, and Africa use languages with bidirectional scripts. These languages include Arabic, Farsi (Persian), Azerbaijani, Urdu, Punjabi, Pushto, Hebrew, and Yiddish. If you recall from the first article, the Middle East region is also experiencing more than 100% growth in Internet usage.

First off, why is it bidirectional? Aren't Arabic and Hebrew scripts written in just the one direction? No, actually they're not. Numbers embedded in these scripts are in fact written LTR just as in Western European text (the most significant digit is first or left-most; 100 is not written as 001 in BIDI scripts) even though the remainder of the text is written RTL. You will very often also find languages written in LTR scripts mixed in with RTL scripts (transliteration of proper or place names can be confusing and sometimes impossible; more often than not these aren't localized and are simply dumped "as is" into the RTL text stream). This is what makes the whole page BIDI. For further spice, note that in Arabic mathematical expressions are written from RTL, even though numbers within the

equations are still written from LTR. As you can see, BIDI text handling is quite complicated, so much so that Flash and several other products still don't properly handle BIDI text at all or need to handle it as a special case (PDF document creation, for example with the excellent iText Java PDF library www.lowagie.com/iText).

Next, in what scripts (recall that languages don't have direction) would you normally write BIDI text? Table 2 shows a few examples, though you will most likely encounter only Arabic and/or Hebrew (the rest are included mainly to show off how well-read the author is). Table 3 shows a larger list of commonly localized languages, their scripts, and the script's direction.

Ideograph languages (Chinese, Japanese, Korean, or CJK, for instance) are often quite "flexible" in their script's direction. For the most part these are written LTR or TTB (top-to-bottom); you might also find them written RTL (and very often when TTB). Chinese-language newspapers are classic examples of this kind of directional elasticity, one page may combine LTR, TTB (with the vertical columns RTL), and RTL text. Makes my head spin.

In Arabic and Hebrew scripts there are three conventions for the order in which text is encoded, two of which are most commonly encountered:

- **Logical order:** Text is stored in memory in the same order it would be spoken or typed. Characters have an inherent direction attribute that is used by a display algorithm to determine the most likely display order for the corresponding glyphs.
- **Visual order:** Text is stored line-by-line in left-to-right display order (that is, the Arabic and Hebrew nonnumeric text is encoded in reverse order). This is characteristically found in text data created by older systems.

In HTML the DIR attribute specifies the base direction (LTR, RTL) of directionally neutral text (which Unicode defines as text not having an inherent directionality). For example <HTML DIR="RTL">. You can also specify direction for several other HTML elements, including <TABLE>, <BODY>, <P>, etc. Text Texin's Web site

Breakthrough Technology
for the 21st Century



Dynamic Interactive Charting

Bringing Java 2D and 3D
to Cold Fusion Developers.
Visually create Java
chart servlets without writing
a single line of code.



Dynamic Model

JDBC database connection

Dynamic Updates

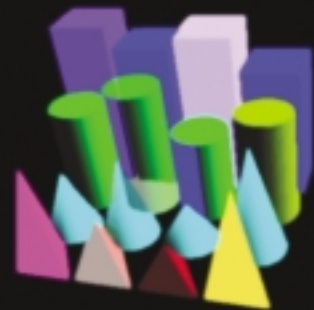
Pure Java Servlets allow
live updates to charts
from your browser.

Create or Edit

With our WYSIWYG
Editor and List boxes.

Flexible Licensing

Have it your way!
\$10 & \$20 Chart Servlets
Licensing, Consulting Services.



Simple to complex, charts that work.
We Guarantee it!

Mention this ad to qualify for your introductory offer

jvisgenwizards.org

A Java™ Sun Microsystems
iForce Development Partner.

RIGHT-TO-LEFT SCRIPT	LANGUAGES
Arabic	Arabic, Azeri/Azerbaijani ¹ , Bakhtiari, Balochi, Farsi/Persian, Gilaki, Javanese ³ , Kashmiri, Kazakh ³ , Kurdish (Sorani), Malay ³ , Malayalam ³ , Pashto, Punjabi, Qashqai, Sindhi, Somali ² , Sulu, Takestani, Turkmen, Uighur, Western Cham, Urdu
Hebrew	Hebrew, Ladino/Judezmo ² , Yiddish
N'ko	Mandekan
Syriac	Assyrian, Modern Aramaic Koine, Syriac
Thaana/Thāna	Dhivehi/Maldivian
Tifinar	Tamashek

Notes:

1. Azeri/Azerbaijani is written in Latin, Cyrillic, or Arabic scripts.
2. Ladino, Judezmo, and Somali are typically written in the Latin script today.
3. Javanese, Kazakh, Malay, and Malayalam were historically written in the listed script, but use another script in modern practice.

Source: W3C FAQ: Script direction & languages www.w3.org/International/questions/qa-scripts.html

Table 2: Some examples of right-to-left script and language

(www.i18nGuy.com/markup/right-to-left.html) has an excellent set of tips for writing RTL text in markup which can be summarized as:

- Use the HTML element, not the BODY element, to set the overall document direction.
- Use character encodings that employ logical, not visual, ordering, such as Unicode, Windows-1255, Windows-1256, ISO-8859-6-i, ISO 8859-8-i. Don't use the visually ordered ISO 8859-6, ISO 8859-8, ISO-8859-6-e, and ISO-8859-8-e. See RFC 1555 for more information.
- Use markup (the dir option such as in `<div dir="ltr" lang="th"> </div>`) instead of the Unicode bidirectional control characters (LRE, RLE, etc.), which need to be embedded in the text stream and are somewhat harder to use.

While we're on the BIDI topic, it's also important to understand that the BIDI concept applies to the whole Web page layout, not just the text content. Visual page flow will also need to be RTL to convey the same meaning to BIDI users. In LTR languages, the most important information is usually placed in the upper-left corner of the screen/page, in RTL it would be the upper-right that's most important. Perhaps more important and very often overlooked, graphics – especially navigation graphics – will be understood by these same users to have an RTL meaning. Figure 1 provides a simple example of this. In RTL languages, which button in Figure 1 do you think will skip to the end? Remember that RTL graphics should be mirror images of their LTR counterparts.

How CSS Can Help

I'm assuming you have a basic understanding of CSS mechanics (because that's about what I have). While CSS is something of a hot issue these days, the G11N world has long looked to CSS in developing global Web applications. Take for example the HTML `` element – knowing now what you know about character encoding and fonts, doesn't it make

COUNTRY/REGION	SCRIPT	DIRECTION	LANGUAGE
Afghanistan	Arabic	RTL	Pashto
Armenia	Armenian	LTR	Armenian
Austria	Latin	LTR	German
Belgium	Latin	LTR	Dutch, French
Brazil	Latin	LTR	Portuguese (Brazilian)
Bulgaria	Cyrillic	LTR	Bulgarian
China, except Hong Kong	Simplified Chinese	LTR or TTB	Mandarin
Croatia	Latin	LTR	Croatian
Czech Republic	Latin	LTR	Czech
Denmark	Latin	LTR	Danish
Estonia	Latin	LTR	Estonian
Finland	Latin	LTR	Finnish
France	Latin	LTR	French
Georgia	Georgian	LTR	Georgian
German	Latin	LTR	German
Greece	Greek	LTR	Greek
Hong Kong	Traditional Chinese ²	LTR or TTB	Cantonese
Hungary	Latin	LTR	Hungarian
India	Devanagari	LTR	Hindi ³
Israel	Hebrew	RTL	Hebrew
Italy	Latin	LTR	Italian
Japan	Kanji + Hiragana + Katakana	LTR or TTB	Japanese
Korea	Hangul, Hanja	LTR or TTB	Korean
Latin America, except Brazil	Latin	LTR	Spanish
Latvia	Latin	LTR	Latvian
Lithuania	Latin	LTR	Lithuanian
Middle East	Arabic	RTL	Arabic
Netherlands	Latin	LTR	Dutch
North America	Latin	LTR	English, French, Spanish
Norway	Latin	LTR	Norwegian
Pakistan	Arabic	RTL	Urdu
Poland	Latin	LTR	Polish
Portugal	Latin	LTR	Portuguese (Portugal)
Romania	Latin	LTR	Romanian
Russia	Cyrillic	LTR	Russian
Serbia and Montenegro	Cyrillic	LTR	Serbian
Slovakia	Latin	LTR	Slovak
Slovenia	Latin	LTR	Slovenian
Spain	Latin	LTR	Catalan, Spanish
Sweden	Latin	LTR	Swedish
Switzerland	Latin	LTR	French, German, Italian
Taiwan	Traditional Chinese	LTR or TTB	Mandarin
Thailand	Thai	LTR	Thai
Turkey	Latin	LTR	Turkish
United Kingdom	Latin	LTR	English

Notes:

1. "TTB" is top-to-bottom, "LTR" is left-to-right, "RTL" is right-to-left.
2. Hong Kong script includes characters of the Hong Kong Supplementary Character Set.
3. English language software is often used in India

Source: W3C FAQ: Script direction & languages www.w3.org/International/questions/qa-scripts.html

Table 3: Directions of commonly localized languages

sense to use a few semantically appropriate CSS selectors instead of a wheelbarrow full of elements?

CSS is often used to control changes in fonts, font sizes, and line heights when language changes in a G11N application. As a real-world example of this, consider Simplified versus Traditional Chinese. Users tend to prefer different fonts for each character encoding, even though they may be using many of the same characters. In theory there are four ways of accomplishing this (see WC3 FAQ: Styling using the lang attribute: www.w3.org/International/questions/qa-css-lang.html):

1. the :lang() pseudo-class selector (XHTML)
2. a [lang|= "..."] selector that matches the beginning of the value of a language attribute
3. a [lang = "..."] selector that exactly matches the value of a language attribute
4. a generic class or id selector

I'm going to ignore the first three methods, simply because most browsers currently do not fully support them. For future reference, the W3C recommends the first method, CSS2 language pseudo-class selector :lang() method. Since we have to make do with the world as we find it, let's examine an example of the generic class or id selector approach shown in Figure 2.

Use the following styling:

```
body      {font-family: "Times New Roman", serif; }
.ar       {font-family: "Traditional Arabic", serif; font-size: 12px;}
.zht      {font-family: PMingLiU,MingLiU, serif;}
.zhs      {font-family: SimSum-18030;SimHei, serif;}
.din      {font-family: "Doulos SIL", serif;}
.th       {font-family: " "Angsana New""; font-size: 14px;}
```

Note: The xml:lang and lang are added to allow for expected future support.

The concept is simple. We add a generic class for each language we want to support. We can then easily "tune up" or "skin" the text presentation per language using fonts, sizes, etc. Besides the extra code required this method also has the disadvantage of having to explicitly define each and every possible language/locale we wish to support. If we wanted to supply larger-size fonts for Australians (en-AU) and Canadians (en-CA) we would have to exactly define two classes for that, otherwise they would inherit text properties from the BODY selector. If the lang|= "..." selector, which matches the beginning value for an attribute, actually worked in all browsers, we could simply define a class, en-BadEyesight, which would then match en-AU and en-CA.

As with any application of CSS, you might encounter issues with browser versions, but I count these as trivial compared to trying to handle this using normal HTML formatting elements.

Just Use Unicode

At one time, after a particularly frustrating week dealing with character encoding issues, I was going to have this section's title, "Just Use Unicode" tattooed on my forehead. My wife and kids couldn't quite see the sense in that so I had to forgo the

experience. Nonetheless, I can't put this any straighter: *Just Use Unicode.*

The preceding section on character encoding should have put you off your G11N feed. The only surefire cure for that is of course Unicode. Using Unicode simplifies things tremendously. You only have to deal with one encoding (UTF-8) on the front end and back end. Mojibake might then indeed become another type of Krispy Kreme donut.

Even BIDI issues become simpler, as Hebrew and Arabic characters have a direction and it becomes fairly straightforward to embed LTR text in RTL text streams, though directionally neutral characters between so-called "direction runs" (such as Bahrain بحرين Kuwait, which is LTR RTL LTR) still require some inline markup to make clear.

Furthermore, standards bodies like the World Wide Web Consortium (W3C) now expect all new RFCs to use Unicode for text encoding. National governments, for example India's, also back Unicode.

I think it's also important to point out how short this section on using Unicode is in comparison with all the "stuff and nonsense" dealing with code page encodings. Just using Unicode does actually simplify things a great deal, and I for one could do with a lot more simplification.

Not that everything's beer and pretzels with Unicode; there is some controversy surrounding it. At one time, Unicode was branded as a Western imperialist cultural plot because of its attempts to consolidate CJK characters, the so-called "Han consolidation." (In all fairness nobody was suggesting consolidating all the "A"s spread across various languages; I think perhaps because they were spread across various languages might be one reason not to, though the rule "Thou shall not disturb existing encodings" might also be at work). This has lately shifted to the idea that Unicode is some sort of Microsoft world-domination conspiracy, though these people should then find it curious that companies like Sun and Oracle con-

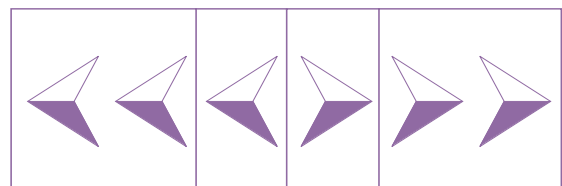


Figure 1: VCR style navigation


```
<p>Yes it is polite to welcome each user in their own language:</p>
<ul>
<li class="cho"><xml:lang="zh-CN">lang="zh-CN">欢迎</li>
<li class="cht"><xml:lang="zh-TW">lang="zh-TW">歡迎</li>
<li class="af"><xml:lang="af">lang="af">Kokooxopler</li>
<li class="ar"><xml:lang="ar">lang="ar">وبهلا</li>
<li class="ru"><xml:lang="ru">lang="ru">Добро пожаловать</li>
<li class="din"><xml:lang="din">lang="din">Kuduak</li>
<li class="th"><xml:lang="th">lang="th">สวัสดี</li>
</ul>
```

Figure 2: The generic class approach

sidered Microsoft's "mortal enemies," are also Unicode Consortium members. I've always liked to think of Unicode like the Borg: "resistance *is* futile," so why bother?

Conclusion

You can find a simple example of some the things discussed in the article in Listing 1. There is a bewildering variety of character encodings in use today, which can often lead to gibberish text due to bad or missing encodings information. This situation can be greatly simplified by using Unicode.

The rubber meets the road in the next article, entitled "In the Year 2525: Cultural Aspects of G11N." This article will deal with handling date, currency, numeric formatting, calendars, collation (sorting), and so on. If you've been looking for some CF code, you'll see it in this article. 

About the Author

Paul Hastings, who after nearly 20 years of IT work is now a perfectly fossilized geologist, is CTO at Sustainable GIS, an agile consulting firm specializing in Geographic Information Systems (GIS) technology, ColdFusion Internet and intranet applications for the environment and natural resource markets, and of course globalization. Paul is based in Bangkok, Thailand, but says that's not nearly as exciting as it sounds.

paul@sustainableGIS.com

Listing 1

```
<cfscript>

    // determines locale from IP
    geoLocator=createObject("component","cfc.geoLocator");
    // handles explicitly located java style resource bundles
    rb=createObject("component","cfc.javaRB");
    // guard against localhost, it can't be in our DB
    if (cgi.REMOTE_ADDR EQ "127.0.0.1")
        ipAddress="147.66.10.158"; //somewhere down under
    else
        ipAddress=cgi.REMOTE_ADDR;
    // get locale, combines IP & user chosen language from browser
    thisLocale=geoLocator.findLocale(ipAddress,cgi.HTTP_ACCEPT_LANGUAGE);

    // in this case, we're also looking for language for CSS class, so
    strip off locale
    thisLang=left(thisLocale,2); //locale in java form, en_US, th_TH,
    etc.
    // resource bundles are stored under current app dir in resources
    sub-dir
    thisDir=GetDirectoryFromPath(expandpath("*."));
    rbFile=thisDir & "resources\welcome.properties";//some sort of wel-
    come blurb
    welcomeRB=rb.getResourceBundle(rbFile,thisLocale); //move rb
    key/values to cf struct
</cfscript>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
    <title><cfoutput>#welcomeRB.title#</cfoutput></title>
    <!--
        in this case, we assume all locales stuffed into one CSS, a
        better
        method would be to have a base CSS & import locale specific
        formatting
        on a per locale basis.
    --->
    <link rel="stylesheet" type="text/css" href="css/g11n.css">
</head>
<cfoutput>
<body>
    <p class="#thisLang#">#welcomeRB.introTxt#</p>
    <p class="#thisLang#">#welcomeRB.giveUsYourMoneyTxt#</p>
</body>
</cfoutput>
</html>
```

Don't Miss CFDJ's Next Issue!



ColdFusion MX and .NET 101:

How CF can be coupled with .NET solutions to allow for interoperability throughout your IT universe.

Making the Most of J2EE Event Listeners:

An introduction and basic explanation of these event listeners examines some ways to listen for session events and respond with custom code.

Creating PDFs from ColdFusion MX Doesn't Have to Cost a Dime: Learn how to create PDFs from CF for free without using a CFX tag.

Ensuring CF as an E-commerce Platform:

ColdFusion is more than capable of meeting all our needs.

CF101: Structures and Arrays: An introduction to structures and arrays and details on how to make use of arrays in ColdFusion.

Download the Code...

Go to www.coldfusionjournal.com

A LIMITED TIME SAVINGS OFFER FROM SYS-CON MEDIA

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

AND SAVE UP TO \$400 AND RECEIVE UP TO 3 FREE CDs!*

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS

3-Pack

Pick any 3 of our magazines and save up to **\$275⁰⁰**
Pay only \$175 for a 1 year subscription plus a **FREE CD**

- 2 Year – \$299.00
- Canada/Mexico – \$245.00
- International – \$315.00

6-Pack

Pick any 6 of our magazines and save up to **\$350⁰⁰**
Pay only \$395 for a 1 year subscription plus **2 FREE CDs**

- 2 Year – \$669.00
- Canada/Mexico – \$555.00
- International – \$710.00

9-Pack

Pick 9 of our magazines and save up to **\$400⁰⁰**
Pay only \$495 for a 1 year subscription plus **3 FREE CDs**

- 2 Year – \$839.00
- Canada/Mexico – \$695.00
- International – \$890.00



CALL TODAY! 888-303-5282

MX Developer's Journal			
U.S. - Two Years (24) Cover: \$143	You Pay: \$49.99 /	Save: \$167 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$29.99 /	Save: \$60	
Can/Mex - Two Years (24) \$168	You Pay: \$79.99 /	Save: \$137 + FREE \$198 CD	
Can/Mex - One Year (12) \$84	You Pay: \$49.99 /	Save: \$40	
Int'l - Two Years (24) \$216	You Pay: \$89.99 /	Save: \$127 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$59.99 /	Save: \$30	
Digital Edition - One Year (12)	You Pay: \$19.99		

Linux World Magazine			
U.S. - Two Years (24) Cover: \$143	You Pay: \$79.99 /	Save: \$63 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 /	Save: \$32	
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD	
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4	
Int'l - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

Java Developer's Journal			
U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22	
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 /	Save: \$48 + FREE \$198 CD	
Can/Mex - One Year (12) \$84	You Pay: \$79.99 /	Save: \$4	
Int'l - Two Years (24) \$216	You Pay: \$176 /	Save: \$40 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

Web Services Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Int'l - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

.NET Developer's Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Int'l - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

XML-Journal			
U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 /	Save: \$68 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 /	Save: \$14	
Can/Mex - Two Years (24) \$192	You Pay: \$129 /	Save: \$63 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$89.99 /	Save: \$6	
Int'l - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

Pick a 3-Pack, a 6-Pack or a 9-Pack					
<input type="checkbox"/> 3-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 6-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 9-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.

TO ORDER • Choose the Multi-Pack you want to order by checking next to it below. • Check the number of years you want to order. • Indicate your location by checking either U.S., Canada/Mexico or International. • Then choose which magazines you want to include with your Multi-Pack order.

WebLogic Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Int'l - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Int'l - One Year (12) \$180	You Pay: \$179 /	Save: \$1	

ColdFusion Developer's Journal			
U.S. - Two Years (24) Cover: \$216	You Pay: \$129 /	Save: \$87 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$108	You Pay: \$89.99 /	Save: \$18	
Can/Mex - Two Years (24) \$240	You Pay: \$159.99 /	Save: \$80 + FREE \$198 CD	
Can/Mex - One Year (12) \$120	You Pay: \$99.99 /	Save: \$20	
Int'l - Two Years (24) \$264	You Pay: \$189 /	Save: \$75 + FREE \$198 CD	
Int'l - One Year (12) \$132	You Pay: \$129.99 /	Save: \$2	

Wireless Business & Technology			
U.S. - Two Years (24) Cover: \$144	You Pay: \$89 /	Save: \$55 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$72	You Pay: \$49.99 /	Save: \$22	
Can/Mex - Two Years (24) \$192	You Pay: \$139 /	Save: \$53 + FREE \$198 CD	
Can/Mex - One Year (12) \$96	You Pay: \$79.99 /	Save: \$16	
Int'l - Two Years (24) \$216	You Pay: \$170 /	Save: \$46 + FREE \$198 CD	
Int'l - One Year (12) \$108	You Pay: \$99.99 /	Save: \$8	

WebSphere Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Int'l - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Int'l - One Year (12) \$180	You Pay: \$179 /	Save: \$1	

PowerBuilder Developer's Journal			
U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 /	Save: \$190 + FREE \$198 CD	
U.S. - One Year (12) Cover: \$180	You Pay: \$149 /	Save: \$31	
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 /	Save: \$180 + FREE \$198 CD	
Can/Mex - One Year (12) \$180	You Pay: \$169 /	Save: \$11	
Int'l - Two Years (24) \$360	You Pay: \$189.99 /	Save: \$170 + FREE \$198 CD	
Int'l - One Year (12) \$180	You Pay: \$179 /	Save: \$1	

*WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Online Today www.sys-con.com/2001/sub.cfm

**SYS-CON
MEDIA**

Making the Case for

There are some compelling reasons

Do you really know what it means to run CFML applications on a J2EE server? How does it work, and why would you bother? There are many benefits you may never have considered. In this article, the first of a series, I'll answer these questions.



By Charlie Arehart

You may have heard that you can deploy your CFML on a J2EE server such as WebLogic, WebSphere, JRun, or JBoss – or on a servlet engine such as ServletExec or Tomcat. But what does this mean to a CFML developer, especially one who's not familiar with (or even interested in) J2EE?

Whether you're an organization with multiple applications running on a single server, or a hosting company, or even if you have just one application on your current CF server, I hope this article will help you see when and where CFML on J2EE can make sense. I'll conclude by pointing you to resources for learning more about this topic.

The Two Main Reasons to Run CFML on J2EE

Running CFML on J2EE really isn't all that different from running CFML the way you're already used to, as I'll show in the next article. For now, just take my word for it that as a CFML developer, you really don't need to worry about most of the details of the J2EE platform, nor do you need to understand anything about Java to appreciate the most significant reasons for considering this alternative. It's just your CFML running on a J2EE server.

The bigger question, of course is simply why a CFML developer would bother – or why a shop moving to J2EE would care – about preserving their CFML investment. They're two sides of the same coin, but very different perspectives, and they hint at the two compelling reasons for running CFML on J2EE.

I'll start with the more obvious reason: you're in a shop moving to J2EE anyway. But I'll conclude (and spend more time) with an entirely different motivation: even if you don't know or care about J2EE, there are many benefits to deploying

CFML on a J2EE server that we just don't get on a stand-alone CFML server.

The thing is, you won't hear any of these reasons explained to you this way by the typical advocates for J2EE.

Reason 1: You're Facing an Organizational Move to J2EE

The first and most obvious reason to consider this alternative is simply if you work in a shop where you're being told that the organization is making a strategic move to J2EE. They're bringing in BEA WebLogic or IBM WebSphere (or choosing any of a number of alternatives like ServletExec, Sun ONE, Oracle, Tomcat, JRun, etc.), and word has come down that it's time to either rewrite or retire the CFML apps.

What does this mean to you as a CFML developer? And what about all that CFML you've written?

Some organizations have developed very complex, enterprise-class CFML applications that they rely on – or perhaps a small part of the organization relies on them, even if the larger IT organization does not. Simply rewriting those as JSPs and servlets (the lingua franca of J2EE servers) is not going to be easy, nor will it happen quickly.

Such organizations may face expenses of millions of dollars and several man-years of effort to do such a conversion. While there are tools that purport to do such conversions automatically, many have found them wanting and inadequate for complex applications. Then, too, what happens to the ever-growing requests for new functionality during this downtime?

CFML on to go this route J2EE

Moving CFML Developers to J2EE?

What happens to CFML developers in such a move? They may hold a tremendous repository of domain understanding. If an organization shifts to J2EE, what do they plan to do with the CFML developers? Let them go? Retrain them?

Becoming a professional-class J2EE developer isn't something you can learn in a couple of weeks, or even several months. Beyond learning the core Java language (and its libraries), you also need to learn JSPs and servlets (and their APIs), and typically also EJBs (Enterprise JavaBeans). Then there are the debates about which of those to use and when, how to apply design patterns, use of MVC and/or frameworks like Struts, etc.

I discussed this challenge in more detail in a *Java Developer's Journal* article entitled, "The Many Sides of J2EE Development," (*JDJ*, Vol. 6, issue 11) available at www.sys-con.com/story/?storyid=36739. The complexity of J2EE development using pure Java is proving to be its Achilles heel.

Many organizations find that their Web app backlog grows significantly when they go the pure J2EE route – this despite its fundamental focus on reuse. But its adherents simply don't think there's any viable alternative, and few of them have heard of the possibility of deploying CFML on J2EE – if indeed their purist intentions would let them even consider it.

As a consultant you can find lots to learn and keep yourself ever-growing and ever in business serving a J2EE environment, but if you've got a job to do (and/or have apps that need to be built) you won't be doing it quickly in a pure J2EE development mode.

Solving the Architecture Dilemma with CFML on J2EE

Here's the key: What if you could keep the CFML you've created and deploy it on a J2EE server? That's the focus of this series, and it at least saves you rewriting your CFML to get it to run in the J2EE environment. If your organization is moving to J2EE you can deploy your CFML as a J2EE Web application, which simply means running the CFML on the J2EE server. After that, you can slowly integrate your application with other J2EE applications and components. Heck, you might even be able to pare down that application backlog by doing some of the apps in CFML instead of pure J2EE.

One of the arguments you may face is that your IT management will not install CFMX or any other stand-alone CFML server (or they want to remove the existing one), but that doesn't mean you can't run CFML anymore.

Most CFML developers run their applications on a stand-alone CFML server. Whether that's an older 4.x/5 release of ColdFusion Professional or Enterprise; or CFMX Standard or Enterprise; or BlueDragon Server or Server JX – these run as a service on Windows or as a daemon on Linux or Unix.

However, what I'm talking about here is running your CFML on a J2EE server, which means you have no CF server (and no BlueDragon server). You just run your CFML as a J2EE Web application on a J2EE server such as WebLogic, Websphere, ServletExec, JRun, Tomcat, etc., using either the second or third installation options of CFMX Enterprise or BlueDragon/J2EE edition.

I'll explain in the next article what all this means, but for now just know that to the J2EE folks (including the server administrators), it really can be transparent to them that

you're even running CFML. All you give them is a J2EE Web application, and they deploy it like any other J2EE application. (Actually, that's how it is with BlueDragon/J2EE. With CFMX running on J2EE there's more involved both in building the Web app and after deploying it, so it's not quite as transparent. But each achieves the end result of running CFML on a J2EE server.)

KEEPING THE CFML DEVELOPERS DOING CFML

An important point to keep in mind as a CFML developer is that if you have to (or want to) run your CFML on a J2EE server, you don't need to learn anything about Java, JSPs, servlets, EJBs, or any other J2EE-based features. You can continue doing development in CFML (and using your preferred editor). It's just that your application is "deployed" on a J2EE server.

I'm not saying that pure J2EE development is bad – indeed, for a Java developer it's the most natural Web application development platform possible. You may even want to learn more about it just to add to your toolkit. In fact, there are great things you can do with CFML to integrate with JSPs, servlets, EJBs (and even JMS and other APIs) if you like, whether you write them or someone else in your shop does. It's just that you don't *need* to understand those things to deploy your CFML on a J2EE server.

Reason 2: To Leverage Benefits of Running CFML on J2EE

Perhaps you're reading this and thinking, "What's all this got to do with me? My shop isn't moving to J2EE." I'd like to show you the benefits you can obtain by running your CFML on J2EE – benefits that you just don't get running CFML as you do on stand-alone servers.

For now, just accept that running CFML on J2EE is easy. I will demonstrate the benefits more clearly in the future article(s) as well. In the meantime, note that you can get started with the documentation from Macromedia and New Atlanta Communications, as well as in the resources I list at the end of this article.

With your CFML running as a J2EE Web application it can leverage all the same benefits any J2EE Web application would reap from running in a J2EE server environment. Some of those benefits are unique and not available when running CFML on a stand-alone server.

Application Management Features

J2EE servers are designed and built to support enterprise-class application processing. As such, they typically include features that would be useful to CFML developers deploying their applications on such a J2EE server. These include the following.

CLUSTERING, LOAD BALANCING, AND FAILOVER

Most J2EE servers offer mechanisms for defining a set of physical servers as a cluster such that they can operate in conjunction with one another, cooperatively handling requests (typically in a high-volume environment) so that no single server becomes overwhelmed (load balancing) and taking over requests being handled by a server that goes down (failover).

Some J2EE servers offer only one or another of these features (if any), perhaps leaving such things as load balancing to external devices, protocols, round-robin DNS servers, or even external Web servers. This brings up another point: some J2EE servers contain a built-in Web server, which itself may or may not offer such features as load balancing and failover.

The stand-alone versions of ColdFusion MX and BlueDragon each offer their own built-in Web server, but both describe it as being intended for development and testing only. As for clustering, ColdFusion MX Enterprise does offer that capability as well. See the Macromedia documentation for more details on when and how this works.

SESSION REPLICATION

Still another benefit offered by enterprise-class J2EE servers when a number of server instances are grouped together, as in the discussion of clusters above, is the ability to replicate sessions among the servers. In other words, the servers are declared to share their sessions with each other (or with one or more of the others backing it up), so that as changes are made to a session on one server, those changes are replicated to the others that back it up.

The goal, of course, is to permit one or more servers to serve as a backup or failover server to handle requests for a server should it crash or otherwise become unavailable. With simpler failover systems it's not possible to replicate sessions, so you must instead declare that the servers in the cluster use "sticky sessions" (as with the clustering in CFMX Enterprise). In that approach, the clustering system simply ensures that each user remains associated with just a single server, and if that server fails there is no way to preserve the user's session via failover.

Clearly, session replication offers a substantial improvement in application reliability and availability. Like many such benefits, it comes at a cost: the overhead of keeping sessions replicated among the participating servers. Some J2EE servers offer a means to temper this cost (perhaps varying how often sessions are replicated or what session data is replicated).

SESSION PERSISTENCE

A subtle twist on the previous discussion – but an important one, is the notion that active sessions in a server instance can be preserved over server restarts. It's not about replication to another server. Indeed, it's a feature that's particularly useful if you're *not* using clusters and session replication. Session persistence could be useful for any application that uses sessions. This way, if your server goes down, when it comes back up the active sessions are restored to memory and still available. If that restart happened quickly enough it's possible that an end user might never experience a problem (if the user doesn't make a request while the server is down).

The sessions may be persisted to a file system, a database, or something else. Again, there's certainly a cost to enable this feature, since the persistence operation must again take place either on every session change or at some interval. The question is simply whether the benefit is worth the price. But it's another benefit available only when running CFML on a J2EE server.

—continued on page 40

```
<cfset circuitPermissions =
Application.fusebox.circuits[myFusebox.thisCircuit].permissions />

<cfif NOT ListFindNoCase(fuseactionPermissions, Session.user.permis-
sions)>

    You do not have the permissions needed for this action.
    Click <a href="index.cfm?fuseaction=sm.login">here</a> to
    log in again.

</cfif>
<cfabort />
```

This plug-in can be altered to work with whatever permissions scheme you use and will ensure that the permissions specified in the fuseaction tag match up with the user's permissions.

While this level of fuseaction granularity provides a great deal of flexibility, you may find that you wish to restrict access only at the circuit level. In that case, you can set the permissions attribute of the circuit tag and leave off any permissions for individual fuseactions.

```
<circuit access="public" permissions="basicAdmin">
    <fuseaction name="showAdminMenu" />
</circuit>
```

In the plug-in example, you would need to change the conditional code to this:


```
<cfif NOT ListFindNoCase(circuitPermissions, Session.user.permis-
sions)>

    You do not have the permissions needed for this circuit.
    Click <a href="index.cfm?fuseaction=sm.login">here</a> to
    log in again.

</cfif>
```

To make things even more flexible, you can provide permissions attributes for both circuit and fuseaction. By doing this, you can first test whether the user has permissions to use the circuit as well as permissions to use a specific fuseaction.

If you'd like to see a sample Fusebox application that uses a plug-in to implement security, download the files from www.sys-con.com/coldfusion/sourcec.cfm or from www.halhelms.com. To learn more about Fusebox 4, read *Discovering Fusebox 4*, available at www.techspedition.com or from Amazon.

Next month, I'll continue with our explorations into encapsulation. 

About the Author

Hal Helms (www.halhelms.com) is a Team Macromedia member who provides both on-site and remote training in ColdFusion, Java, and Fusebox. Hal is cofounder of the Mach-II project.

hal.helms@teamallaire.com

SAVE

16% OFF

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 \$17⁸⁹ 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



One more point: the ability to leverage both session replication and persistence is something that comes only when you enable “J2EE sessions” in the administration console of either CFMX or BlueDragon when deploying a CFML Web application on a J2EE server. When you do this, you’re in effect telling the J2EE server to take over control of session management, rather than having the CF or BlueDragon CFML engine handling sessions.

This reflects one of many kinds of mechanisms that previously we’d look to the CFML engine to provide, but the J2EE server may provide it as well – and perhaps do it better – or at least handle it better in a clustered environment. The next feature is another example.

DATASOURCES DEFINED AT THE J2EE SERVER LEVEL

Just as we could give up control in our applications to have the J2EE server manage our sessions, we could also give up control to the J2EE server to define and manage our datasources. This can have many benefits. It helps first, though, to clarify that what this means is simply that rather than define a datasource in the CF or BlueDragon admin console, you would instead define it in the J2EE server’s admin console. The features and interface are generally very similar.

And you would still refer to the datasource name in CFQUERY (and related tags) just as you do now, but instead of being a datasource name defined in the CF or BlueDragon admin, you would refer to the datasource name defined in the J2EE server.

The difference is that the J2EE server may offer some features that you don’t get in either CF or BlueDragon’s admin console or datasource definition. For instance, the J2EE server may perform connection pooling better, it may offer more updated database drivers or support additional databases, or it may offer additional benefits in a clustered environment.

And, as I will detail later here, you can arrange to run several applications on a single server. If you define the datasource at the J2EE server level, then that one configuration can be leveraged by any of the several applications.

One last thought on this topic is that since the J2EE servers may have far more customers (there are a lot more Java developers in the world than CFML developers), you may also see improvements coming about more quickly in some areas (in the area of J2EE datasources, for instance) than you would in the CFML world.

BUILT-IN APPLICATION TRACKING

J2EE servers often offer mechanisms to track and manage applications as they run. This can include reporting how many requests have been made, what kind of processing they’ve performed, how long they’ve been running, how many sessions have been started and their runtimes, what sort of datasource processing has occurred, etc. (I’ll add that BlueDragon offers this information about CFML requests in all editions, including the stand-alone server ones. While ColdFusion Enterprise used to offer a Server Reports feature with some of this information, this is no longer available in CFMX.)

Running Several Applications at Once

The advanced application management features described

above are both needed and enhanced by the fact that when a J2EE Web application is deployed on a J2EE server, that application is in many ways segregated from other Web applications deployed on the same server.

What this means is that unlike different applications being separated from each other only because they’re in different subdirectories, as happens on stand-alone versions of ColdFusion and BlueDragon, when you deploy CFML as a Web application on a J2EE server, it’s segregated from other Web applications on that J2EE server.

There are a couple of levels of segregation. If you’re familiar with the concept of “independent instances,” I’m not even talking about that in this section. I’ll discuss that concept later in this article.

SEPARATE APPLICATION MANAGEMENT

When CFML is deployed as a Web application on a J2EE server, that Web application can be stopped and started independently of any others. When an application is stopped, requests sent to the server are rejected, indicating that the application is unavailable. This could be useful during maintenance operations.

This is a level of granular control over applications that we don’t experience in either ColdFusion or BlueDragon stand-alone servers, where you can only stop and start the entire server, thus affecting all applications running there. (Note that this isn’t really stopping the application, but instead just stopping requests to it.)

This capability is extended with even more power when running as multiple independent instances, which is again a different and valuable feature explained later.

Some J2EE servers also offer a means to “reload” the Web application, which can be useful when configuration changes have been made to underlying XML and Java files.

SEPARATE ADMINISTRATION CONSOLES AND CONFIGURATION

Extending on the fact that each Web application is deployed separately from one another, an additional benefit of deploying CFML on a J2EE server is that each application can have its own administration console. I’m referring here to ColdFusion’s or BlueDragon’s console.

This is a powerful benefit that means you could give the owners of a single application the ability to both administer their application environment and set up various configuration aspects of their CFML environment. Imagine enabling debugging for one application but not for another, or enabling a datasource in one and not another, or turning on trusted cache for one but not another.

Still another benefit comes in the area of security: you can configure the Web application (using the Web application’s web.xml file and/or features unique to each J2EE server) to configure different Web applications to have different security settings. These could influence how Web application requests are authenticated and authorized (if you choose to use that) as well as controlling what directories can be accessed by code within the Web application.

These are clear benefits for hosting environments, but even an intranet with multiple unrelated applications, or a compa-

CFUN4



301.424.3903
info@teratech.com

Two whole days of...
Networking! Learning! Fun!

Sixth Annual ColdFusion Conference

June 26th & 27th, 2004

\$199

Per Person

Special Price until 3/31/04

Charlie Arehart	Simon Horwith
Jo Belyea-Doerrman	Larry Hull
Raymond Camden	Chafic Kazoun
Christian Cantrell	Matt Liotta
Sandra Clark	Tom Muck
Sean Corfield	Rey Muradaz
Robert Diamond	Nate Nelson
Michael Dinowitz	Samuel Neff
Steve Drucker	Jeff Peters
David Epler	John Quarto
April Fleming	Neil Ross
Ben Forta *tentative	Stephen Shapiro
Shlomy Gantz	Michael Smith
Critter Gewlas	Geoff Snowman
Mark Gorkin	Jeff Tapper
Hal Helms	Dave Watts

5 TRACKS!

Bootcamp - Basic ColdFusion and Flash topics

Advanced - Advanced ColdFusion topics

Empowered - Fusebox & Project management topics

Accessibility - making sites that disabled people can use, section 508

Integration - Flash, Flex and other technologies integrated with CF topics

www.cfconf.org/cfun-04/

"I learned numerous techniques last year that have helped myself and our development team to better deliver quality products to our customers. CFUN is also a great venue to meet some of the names you see in CFDJ and DevNet and talk with them one on one."

-Phillip D



ny offering a service as an ASP (application service provider), could benefit.

Other features you might prefer to control in this granular manner include virtual directory mappings, custom tag paths, CFX tag registration, etc. (as well as things like whitespace compression, global error handlers, application timeouts, and client variable storage, though these can also be controlled at an application level via `application.cfm`.)

I will point out that one challenge with running multiple applications, each having their own configuration settings and CF/BD administration console, is that there's no current mechanism in either ColdFusion or BlueDragon to propagate changes in one application so that it they're applied/synchronized with other CFML-based Web applications on the J2EE server. For now, this is something you need to manage yourself. (On the other hand, many J2EE servers will automatically deploy a given Web application across all servers in a cluster so that if the first one is configured correctly before deployment, its configuration information will also be deployed with the app across all the servers.)

RUNNING MULTIPLE VERSIONS OF APPLICATION, CFML ENGINE ON ONE SERVER

Continuing the idea enabled by the previous feature, since each application is effectively segregated from all others, and since the configuration info is contained within the Web application along with your CFML code, it's also possible then for each Web application to be created using different versions of the CFML engine.

I'm going to discuss only BlueDragon here, since I'm not as clear about how CFMX on J2EE works in this regard. In the case of BlueDragon, under the covers the configuration of the Web app as deployed on the J2EE server tells the J2EE server that requests for CFML pages should be handed to the BlueDragon engine. The guts of that engine – how to run CFML – are included within the Web application itself (the directory or WAR file, for those who understand such things).

So you could deploy some CFML within a Web application running one version of BlueDragon (say, BlueDragon 3.0) and then another running the CFML as BlueDragon 6.1 (our latest release). You could use this to test different betas or upgrades within a release as well.

ent from those I've described so far, though it's easy to confuse them. I'm referring to the concept of running multiple independent instances on the J2EE server, a phrase you might have heard quite a bit about if you've been following CFMX closely.

There have been articles both from Macromedia and in *CFDJ* (the latter listed at the end of this article) describing this with respect to CFMX. The thing is, there's nothing special about CFMX that enables it. It's something that you can leverage with BlueDragon deployment on J2EE as well. In a future article I'll discuss deployment of CFML on J2EE as independent instances using BlueDragon.

With this feature, some J2EE servers offer the means to create separate virtual servers (sometimes called "instances"). In stand-alone implementations of CF and BlueDragon, all applications run under a single server instance. With a J2EE server offering this feature, each virtual server is independent of the others in terms of the underlying JVM that controls the server.

APPLICATION ISOLATION

The biggest advantage to each being controlled by its own JVM is that if the application(s) in one virtual server/instance fails or slows to a crawl, the application(s) running in another virtual server/instance should be able to continue to run unaffected.

Also, each JVM can be configured independently, allowing the developer to change JVM settings such as the amount of memory enabled, the version of the JVM, the classpath for the JVM, etc.

This is different from the application segregation features I described earlier, in which each deployed application can be controlled separately and has its own admin configuration. If they're not running in separate virtual servers/instances, then you get the benefits of application configuration segregation but not of process isolation.

By the same token, even if you don't (or can't) choose to enable independent instances, there are still those benefits that come from the segregation of J2EE Web applications, even in a single virtual server/instance.

PERFORMANCE/THROUGHPUT IMPROVEMENT

One thing that is indeed unique to using multiple independent instances is a possible performance/throughput improvement by using more than one instance on a single

**"It's perhaps a dirty secret (in the opinion of J2EE folks),
but you can do pretty much anything in CFML that you
can do in JSP or in a servlet"**

You could even extend this to creating different Web apps for different stages of development, perhaps one Web app for the development stage and another for QA testing, etc. It's really powerful to be able to run multiple versions of the application or different CFML engine versions on a single server.

Leveraging Multiple Independent Instances

The final aspect of application segregation is one that's differ-

server. This may seem illogical; if you run more than one instance, you're clearly spreading out the available resources of the machine. How can that improve performance?

The secret is in the fact that while each JVM has less memory to work with (since it's sharing a fixed amount of available memory), it also is therefore responsible for managing less memory when it comes time to do garbage collection, a JVM feature used to manage unused memory when processes leave

objects orphaned. (It's an inherent aspect of how Java-based applications work – and while we're programming in CFML, under the covers, everything is really happening in Java.)

ADDITIONAL CLUSTERING BENEFITS

One other cool thing about multiple independent instances is that by enabling them on a single machine, you can in effect “cluster” these multiple instances and then leverage the features discussed earlier in the Application Management Features section. These include clustering, load balancing, failover, starting/stopping applications independently, session replication, and session persistence.

And just as having segregated application configuration among multiple Web applications (even in a single instance) offers advantages, the same is obviously true of Web apps deployed on independent instances.

Of course, another benefit of the instances being separate from each other – and perhaps the primary reason for creating such virtual servers – is that you can generally configure Web servers to support multihoming, in which a single Web server supports multiple IP addresses or domain names, such as www.mycompany.com and services.anothercompany.com, each running out of the Web root of the different instances.

Reducing Licensing Costs by Reducing Machine Quantity

There's another major benefit that can be derived from

the combined features described in the previous two sections on supporting multiple applications. Whether deploying them on a single instance or on multiple instances, there's a real chance that you could save money using these features.

Consider that each offers the means to create multiple versions of your applications, and possibly even to cluster them, etc. You can clearly gain benefits that would otherwise require you to have separate physical machines (and each would of course need its own license for ColdFusion or BlueDragon).

Whether for clustering, or for running different versions for testing/QA, etc., each of those separate Web applications (or instances) runs under a single license. Both products are licensed on a per CPU basis when deployed on a J2EE server, not per instance or application. You could actually save money by deploying on J2EE.

Integrating CFML with J2EE

Another major benefit of deploying CFML on J2EE is that you can integrate your CFML with other J2EE components such as JSPs and servlets, EJBs, JMS queues, and more.

This is powerful. If your company is moving to J2EE, you may well feel a pinch from trying to do things in CFML, even once you've convinced management that you're really running on J2EE. They may know that other applications or components have been built in pure J2EE.

WebServices

.NET J2EE XML JOURNAL

LEARN WEB SERVICES. GET A NEW JOB !

SUBSCRIBE TODAY TO THE WORLD'S LEADING WEB SERVICES RESOURCE

Get Up to Speed with the Fourth Wave in Software Development

- Real-World Web Services: XML's Killer App!
- How to Use SOAP in the Enterprise
- Demystifying ebXML for success
- Authentication, Authorization, and Auditing
- BPM - Business Process Management
- Latest Information on Evolving Standards
- Vital technology insights from the nation's leading Technologists
- Industry Case Studies and Success Stories
- Making the Most of .NET
- Web Services Security
- How to Develop and Market Your Web Services
- EAI and Application Integration Tips
- The Marketplace: Tools, Engines, and Servers
- Integrating XML in a Web Services Environment
- Wireless: Enable Your WAP Projects and Build Wireless Applications with Web Services!
- Real-World UDDI
- Swing-Compliant Web Services
- and much, much more!

Only \$69.99 for 1 year (12 issues)*
* Newsstand price \$83.88 for 1 year
Subscribe online at www.wsj2.com or call 888 303-5252
*Offer subject to change without notice



No worries. It's perhaps a dirty secret (in the opinion of J2EE folks), but you can do pretty much anything in CFML that you can do in JSP or in a servlet. Want to call an EJB? You can. Want to include a header that's already been written in JSP? Go for it. Need to call a Java library method that does something you can't do in CFML? It will work. Are you curious to see if a CFML page can be viewed in a Struts application, or be rendered as a portlet? People have done it.

Your CFML running as a J2EE Web application can benefit from other aspects that apply to pure J2EE Web applications, such as integration with servlet filters and listeners. I already mentioned how you might also use J2EE Web application security authentication and authorization. Each of these is a feature that applies to any template requested in a J2EE Web application that's been configured to use the feature; they're not unique to JSPs and servlets.

Supporting Various Platforms

Finally, one aspect of deploying CFML on J2EE is something that may not seem to be an obvious "benefit" if you look at J2EE servers and wonder what you can do with them using CFML.

With stand-alone versions of BlueDragon and CFMX, the vendors (New Atlanta and Macromedia, respectively) have had to build separate installers to be executable on various operating systems.

With deployment on J2EE, however, you're not installing a stand-alone CFML server. You're just deploying the CFML in (or as) a J2EE Web application. That Web application can be created and implemented in a platform-independent way.

Again, I'll speak more for BlueDragon here, as I have more experience with it. The BlueDragon/J2EE edition is available as a zip file. When you unzip it, it's just a directory containing the documentation and a subdirectory that is itself the J2EE Web application. You can make a copy of that subdirectory, name it whatever you like, copy your CFML into it, and deploy it onto any J2EE server.

The point is that there is no platform-specific installation routine to be executed to create the skeletal J2EE Web application into which you place your CFML to be deployed onto a J2EE server. That's any J2EE server, on any platform. There's nothing OS-specific about the J2EE Web application, at least with BlueDragon/J2EE.

Bottom line: you can run your CFML on any platform for which there's an available J2EE server. I've been asked if BlueDragon would allow a CFML developer to run their application on such things as a mainframe or an iSeries/AS 400, and such operating environments as FreeBSD, OpenBSD, and even NetWare. My answer is the same: if there's a J2EE server for them (and there is), then you can deploy BlueDragon/J2EE there and run your CFML application. It's truly CFML everywhere!

Summary

I hope I've helped persuade you that running CFML on J2EE can not only be a great solution if your organization is moving in that direction, but even if you've never considered J2EE before.

If You're Facing a Mandated Move to J2EE

If you're facing an inevitable, inexorable push in your environment toward J2EE, then running your CFML on J2EE gives you several benefits:

- You can just park the CFML on the J2EE server and keep it running.
- You can start integrating with Java, if you'd like, at your own pace.
- You can start converting CFML apps to JSP/servlets – again, at your own pace.

In shops where a move to J2EE is being mandated from the top, this is a fight for the very survival of CFML. You've invested far too much in your CFML apps (and talent) to just throw them away.

You'll be arguing against architecture gurus who see CFML as a toy, if indeed they've even heard of it. Their experience may be tainted by the CF servers of old. You'll need to help them understand how all this works. I hope this article serves as a starting point for your discussions.

You'll also benefit from understanding more about how these J2EE architects and developers see things, and the environment in which they work. I will discuss these issues further in an upcoming article.

If You're Motivated by the Benefits of CFML on J2EE

Even if you weren't interested in deploying on J2EE before, I hope you can see now why it's so exciting. There are just so many benefits that you can't get by running your CFML on a stand-alone server:

- Application management
 - Clustering, load balancing, and failover
 - Session replication and persistence
 - J2EE datasources
 - Application tracking
- Running several applications at once
 - Separate application management
 - Separate administration consoles and config settings
 - Running multiple application versions/engines at once
- Leveraging multiple independent instances
 - Application isolation
 - Performance/throughput improvement
 - Additional clustering benefits
- Reducing licensing costs by reducing machine quantity
- Integrating CFML with J2EE
- Supporting various platforms

As you can tell, I'm excited about this prospect. Others have been too, and the resources that follow describe many of these features in more detail.

Resources

I'll be giving more details about working with CFML on J2EE in a future article, but if I've really done a good job of motivating you, you may want to get started right away.

Installing CFMX for J2EE Deployment

The ColdFusion manuals explain their approach to deploy-

ing CFML on J2EE. There are actually several resources, and to be honest, the approach has changed three times since CFMX was first released. (There was "phase 1" and "phase 2" in the CFMX timeframe, when the capability was called CFMX for J2EE, then another change in the CFMX 6.1 release when it was bundled into the CFMX Enterprise product as an additional installation option.)

The best place to start may be the sections of the Macromedia site devoted to the subject, including:

- *ColdFusion MX and the Java Platform:* www.macromedia.com/software/coldfusion/j2ee
- *Deploying ColdFusion MX 6.1 on J2EE Application Servers:* www.macromedia.com/go/cfmj2ee-cert
- *Installing the J2EE Configuration:* <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/installj.htm>

Installing BlueDragon/J2EE

- *BlueDragon Documentation:* www.newatlanta.com/products/bluedragon/self_help/docs/index.cfm

Both CFMX and BlueDragon are available for free trial and as free development editions.

Integrating CFML with Java/J2EE

- *Integrating J2EE and Java Elements in CFML Applications:* <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/java.htm>
- *ColdFusion MX/J2EE Hybrid Applications:* <http://sys-con.com/coldfusion/article.cfm?id=636>
- *Fun with Filters:* <http://www.sys-con.com/coldfusion/article.cfm?id=573>

Still another resource for learning more about J2EE integration with CFML is *Reality ColdFusion MX: J2EE Integration* (Macromedia Press) by Ben Forta, et al. Like other books in the "Reality" series, it's got a different style and isn't really geared toward teaching you how to do the integration. You're more like a fly on the wall as the book goes through several different projects, but you'll surely pick up more information along the way. (I'll add that Amazon and other sites list me as a coauthor. Actually, I pulled out of the book over editorial differences. This series of articles is more the style of presentation I'd wanted to make to CFML developers.)

Multiple Independent Instances

- *When One ColdFusion Is Not Enough:* <http://sys-con.com/coldfusion/article.cfm?id=626>
- *Using Multiple Server Instances:* <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/clusteri.htm>
- *Advantages of using multiple instances for ColdFusion MX for J2EE:* www.macromedia.com/devnet/mx/coldfusion/j2ee/articles/multiple.html
- *Installing and Configuring ColdFusion MX 6.1 Multiple Instances with IIS and Apache Virtual Hosts:* www.macromedia.com/devnet/mx/coldfusion/articles/multi_instance.s.html

- *Introducing Multiple Server Instances in ColdFusion MX Enterprise 6.1:* www.macromedia.com/devnet/mx/coldfusion/articles/multiple_61.html

Making the Case to J2EE Folks/Management

Finally, I'll point out a couple of other previous *CFDJ* articles written by my colleague (and boss), Vince Bonfanti. These offer still more discussion on the general benefits of running CFML on J2EE, but they're more from the standpoint of convincing J2EE developers and management why CFML on J2EE is different from what they might expect:

- *It's Not ColdFusion - It's J2EE!:* <http://sys-con.com/coldfusion/article.cfm?id=588>
- *Making the Case for CFML:* <http://sys-con.com/coldfusion/article.cfm?id=618>

About the Author

Charlie Arehart is CTO of New Atlanta Communications, makers of BlueDragon. A Macromedia Certified Advanced ColdFusion developer and trainer, he continues to support the CFML community, contributing to several CF resources, and speaking frequently to user groups throughout the country.

charlie@newatlanta.com

"I was totally intimidated by Java, but I knew I had to learn it. Your class taught me what I honestly thought I couldn't be taught." - Sharon T

Java for ColdFusion Programmers?

Java for ColdFusion Programmers, the five-day Hal Helms, Inc. training class is designed for ColdFusion programmers; no previous Java experience is needed. You'll learn how to think in objects and program in Java.

For class information and registration, come to halhelms.com.

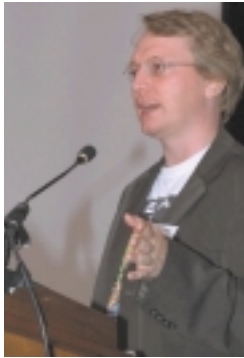


Figure 1: Michael Smith opens CFUN-2K

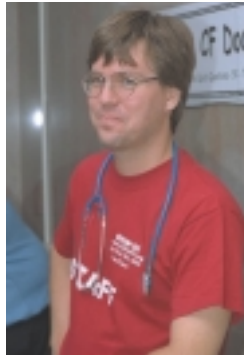


Figure 2: The CF Doctor is in

- Java for ColdFusion Programmers
- SQL Server Reporting Services
- Introduction to SQL Server 2000 Security
- Mach II at Macromedia
- CFC Best Practices, Tips, and Tricks
- CSS for Better Sites
- Variables and Conditions
- And many more!

Networking Opportunities Galore

If you're looking for work, CFUN is a fantastic place to market your skills and connect with consultants and development firms. Discuss the market, where it's going, and what you can offer in it. Sure, the job market's tight and most everyone's been affected, but there are companies looking for the right people. Investing in a conference like CFUN demonstrates commitment to your future and can give you an edge in getting a job. (I recommend you get your résumé and portfolio together before the event so you're able to follow up fast on Monday morning with any leads you get!)

CFUN isn't a "party," but I expect you'll have fun learning and meeting people. It's an opportunity to integrate into (and ingratiate yourself with) the great ColdFusion community while learning new stuff too. If you're a consultant, or looking for one, or just want to network with others in your position, it's one of the best ways to meet a lot of folks. At the end of the conference you'll be exhausted. You'll have learned a lot, made new friends and contacts, and have a pocket full of business cards!

In addition to meeting the local folks, you'll find guru programmers, authors, and other important players in CF who

come to the DC area for CFUN. People who can't afford the big conferences can still share in the experience and reap the rewards of spirit, motivation, education and, most important, bringing the local community together for a weekend to connect in person.

Then again, people have come to attend from far and wide (even from Canada) because of what they've heard or experienced.

Serving the Community Since 1999

CFUN started in 1999 when TeraTech and the Capital PC User Group (CPCUG) organized the first CFUN conference at the NIH headquarters in Bethesda, MD. Over 900 people registered or walked in, filling the Masur Auditorium to overflowing. A second room (the Lipsett Amphitheatre) was set up with a live video link so extra people could view the proceedings! Since then there has been a CFUN conference each June.

Since moving the conference to new hotel facilities the number of speakers has grown from 12 to 30, with a total of 42 talks, now offered in 5 tracks in concurrent sessions so you can select the topics that interest you the most.

The CF Doctor Is In

One of the staples of the conference is the CF Doctor, Douglas Smith (no relation to Michael), who has been at all of the conferences answering questions and giving advice.

Additionally, the conference has also had many of the leading vendors in the CFML space offering their wares and demos, as well as advice and often giveaways. And to help break up the educational intensity, each year there have been programming "fun and games" too (CF Jeopardy, CF Millionaire, and more). And there are always lots of door prizes, too!

Speakers by the Dozens

Speakers typically include many of the famous authors, list members, and other

nationally known names in the CFML programming community. This year, Macromedia guru Ben Forta, who's written more books than many can count, will be giving a keynote on Sunday. Local authors Jeff Peters (*Fusebox: Developing ColdFusion Applications* and other books) and Steve Drucker (contributor to the first editions of Ben's seminal CF book) are speaking. And Microsoft .NET evangelist Geoff Snowman will be talking about new SQL Server features.

CFDJ magazine authors Hal Helms, Ray Camden, Simon Horwith, and Charlie Arehart are coming as well, and of course TeraTech's own Michael Smith will be speaking too. This is just a sampling of the many great speakers – 30 in all – giving 42 talks over 2 days. See the site (URL at the end of the article) for all the speakers and topics.

CFDJ and Other Sponsors

CFDJ will be hosting a panel where you can put your questions to the experts. Plus you can talk with *CFDJ* editor-in-chief Robert Diamond. Other sponsors with booths will include HostMySite, New Atlanta Communications (BlueDragon), activePDF, House of Fusion, and O'Reilly.

The local PC user group, CPCUG, will be staffing a membership information booth together with a local nonprofit, Byte Back (www.byteback.org), that provides training for people who can't




afford regular computer classes so they can learn how to program and get a better job.

Conclusion

Again, there are many CFML conferences held each year, and each has its strengths. I personally love speaking at all of them. But here's what one attendee said about CFUN last year:

"The CFUN series of conferences is by far the best ColdFusion conference of the year. Each year, Michael puts together the most advanced and sophisticated set of ColdFusion topics and speakers to be found – anywhere – for any price. Nowhere else do you get to network with the real CF gurus and peers alike in a fun, friendly, casual atmosphere. The timing of the meeting couldn't be better, either. The early summer dates for CFUN provide an excellent 'opposite end of the year' opportunity to see what's going on with ColdFusion and the Macromedia world. Year after year, the meeting provides a first look opportunity with CF Server and Fusebox product releases. Best courses, best learning, best timing, best networking, best price – best meeting. Period."

The CFUN-04 ColdFusion user conference is being held Saturday, June 26, through Sunday, June 27, from 8am–5pm in Rockville, MD. The cost is \$269. Full details and registration information are at www.cfconf.org/cfun-04. 

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
21C JVISGENWIZARDS	WWW.JVISGENWIZARDS.ORG	650-641-1744	31
ACTIVEPDF	WWW.ACTIVEPDF.COM	866.GoToPDF	4
CFDYNAMICS	WWW.CFDYNAMICS.COM	866.233.9626	6
CLEARNOVA	WWW.CLEARNOVA.COM/THINKCAP		11
EDGE WEB HOSTING	WWW.EDGEWEBHOSTING.NET	1.866.EDGEWEB	COVER II
HAL HELMS, INC	WWW.HALHELMS.COM		45
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM/CFDJ	877.248.HOST	21
INTERAKT ONLINE	WWW.INTERAKT.RO	301-424-3903	COVER III
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800.379.7729	IV
MACROMEDIA	WWW.MACROMEDIA.COM/INTO	415-252-2000	13
MACROMEDIA	WWW.MACROMEDIA.COM/GO/2004	415-252-2000	19
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	15
TERATECH	WWW.CFCONF.ORG/CFUN-04/	301-424-3903	41
WINMILL SOFTWARE	WWW.WINMILL.COM	888-711-6455	3

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 discretion 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.

THE INSIDER INTELLIGENCE YOU NEED...

TO KEEP AHEAD OF THE CURVE



SELECT THE INDUSTRY NEWSLETTERS THAT MATCH YOUR NEEDS!
CHOOSE ONE – OR TRY THEM ALL!

-
-
-
-
-
-
-
-

FREE

E-Newsletters

SIGN UP TODAY!

Go to www.SYS-CON.com

The most innovative products, new releases, interviews, industry developments, and plenty of solid i-technology news can be found in SYS-CON Media's Industry Newsletters. Targeted to meet your professional needs, each e-mail is informative, insightful, and to the point. They're free, and your subscription is just a mouse-click away at www.sys-con.com.

Exclusively from the World's Leading
i-Technology Publisher



Don't Delay!

Subscribe for FREE!

at www.sys-con.com

The DRK Treasure Trove

DevNet Resource Kit may have just what you're looking for

Way back in the early Allaire days, registered ColdFusion users were given access to "fuel packs", product add-ons (in the form of custom tags) that may or may not have become part of the core product later.

In fact, tags like <CFSEARCH>, <CFPOP>, and <CFLDAP> all started life as fuel packs, and later became part of the core ColdFusion product. Fuel packs allowed us to slip-stream functionality into the product without actually requiring a product revision.

Fuel packs no longer exist, but their spirit lives on in the form of the DRK, the DevNet Resource Kit. Released quarterly, each DRK features content for ColdFusion, Dreamweaver, Flash, and more. And not just new tags; DRKs also contain technical documents, sample book chapters, and even complete applications. With the sixth DRK now available, I thought I'd take the opportunity to call out some of the content that I think will be of particular interest to ColdFusion developers.

CFNETTOOLS

Have you ever needed to do a DNS or reverse DNS lookup? I wrote UDFs way back when to do this, but here is a better way. CFNETTOOLS (in DRK6) is a collection of tags that provide DNS lookup functionality and more, even allowing you to perform WHOIS lookups and determine the mail server (MX record) for a specified domain.

CFNETTOOLS is made up of four custom tags and supporting Java code. Here are some simple usage examples:

```
<!-- Get current time -->
<CF_TIME NAME="time"
          SERVER="time.nist.gov">
<!-- Display it -->
<CFOUTPUT>#time#</CFOUTPUT>
```

This example makes a SNTP (Simple Network Time



By Ben Forta

Protocol) call to get the current time from the time server at www.nist.gov (the National Institute of Standards and Technology Web site), and then display that time.

This next example does a simple DNS lookup:

```
<!-- Get IP address -->
<CF_DNSLOOKUP NAME="ip"
               ACTION="getip"
               ADDRESS="www.forta.com">
<!-- Display it -->
<CFOUTPUT>#ip#</CFOUTPUT>
```

And there's more. Very useful, indeed.

JIMG

JIMG (in DRK4) is a set of graphics manipulation tools, exposed via custom tag and ColdFusion Components interfaces. JIMG is built on top of JAI (the Java Advanced Imaging API) and exposes all sorts of functionality, including image cropping and resizing, adding borders, placing text on top of an image, obtaining image height and width, image rotation, and more.

The following code snippet demonstrates basic JIMG usage:

```
<CFIMPORT TAGLIB="c:\cfusionmx\CustomTags\jimg"
          PREFIX="img">
```

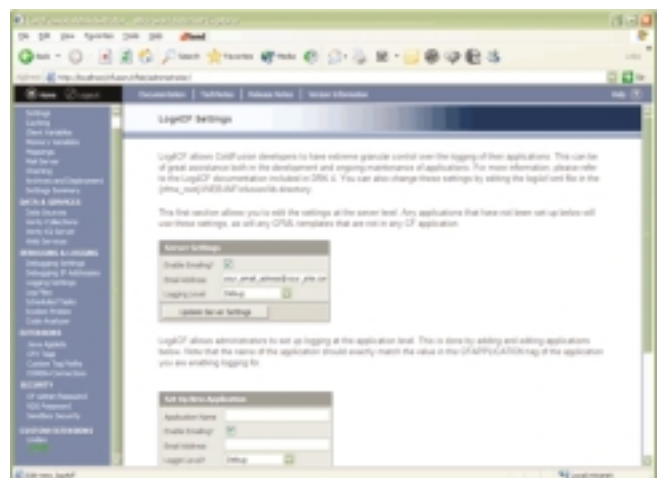


Figure 1: Log4CF Administrator

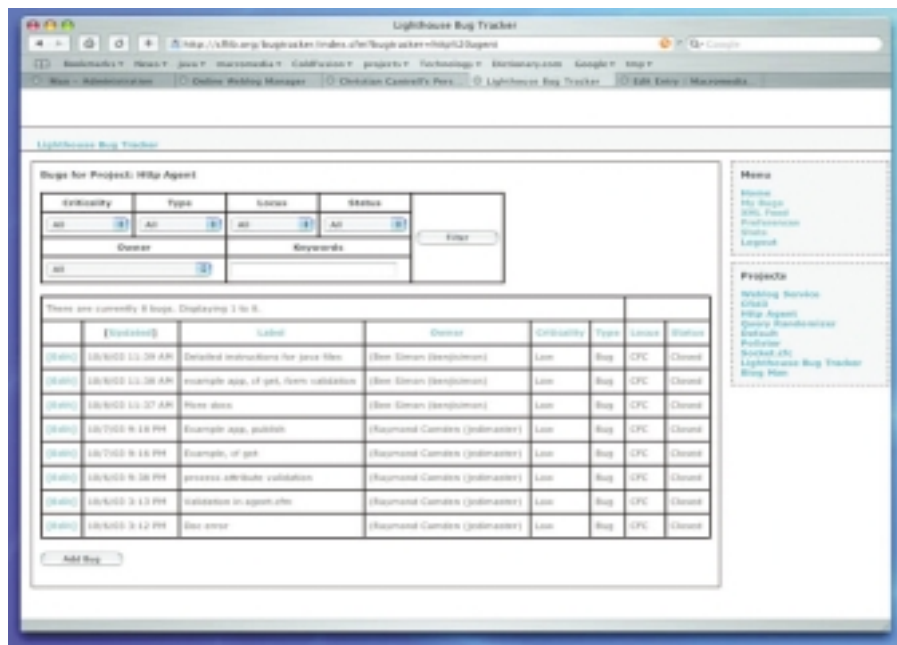


Figure 2: Lighthouse Bug Tracker

```
<img:SEQUENCE>
  <img:LOAD FILENAME="#imageSource#"
  <img:SCALETOATLEAST WIDTH="75"
    HEIGHT="75">
  <img:SAVE FILENAME="#imageDest#">
</img:SEQUENCE>
```

This code creates a thumbnail from a specified image. <CFIMPORT> imports the custom tags, SEQUENCE begins a sequence of operations, LOAD loads a specified image, SCALETOATLEAST scales the image to thumbnail size, and SAVE then saves that image.

Internally, the JIMG code makes a series of rather complex Java calls, but that is all hidden from you. Just call the custom tags or CFC methods, and graphics manipulation is a snap.

Lindex

Lindex (in DRK3) is a Verity replacement built using the Lucene open-source full-text search engine. The Lindex interface is modeled on that of Verity; the tags and attributes are similar enough that users familiar with ColdFusion's Verity support should have no problem using Lindex.

Lindex is distributed as a JAR file containing the core Lucene library, Java wrapper classes, and three ColdFusion custom tags that expose Lindex to your CFML code. <CF_LINDEXSEARCH> is used to perform searches (similar to

<CFSEARCH>); <CF_LINDEXINDEX> is used to update indexes (similar to <CFINDEX>); and <CF_LINDEXCOLLECTION> is used to manage collections (similar to <CFCOLLECTION>).

And the best part is that as Lucene is pure Java, Lindex is supported on all platforms supported by ColdFusion (including those not supported by Verity).

Lindex is not as feature rich as Verity, and does not support as many languages and file types, but if it fits your needs it may well be worth a look.

Log4CF

Logging is an important part of application debugging, troubleshooting, and fine-tuning. ColdFusion's built-in logging support is powerful but lacks much of the granular control that developers need. This is where Log4CF (in DRK4) becomes useful. Using Log4CF you can create your own log entries with varying severity levels, have logs automatically sent to you via e-mail, manage and browse log file contents, and even leave logging inside of your application (so that it may be enabled if needed later on).

Log4CF is distributed as a custom tag that does the actual logging and an Administrator interface that plugs into the standard ColdFusion Administrator (see Figure 1), which can be used to con-

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 Carrie Gebert
201 802-3026
carrieg@sys-con.com

REprints

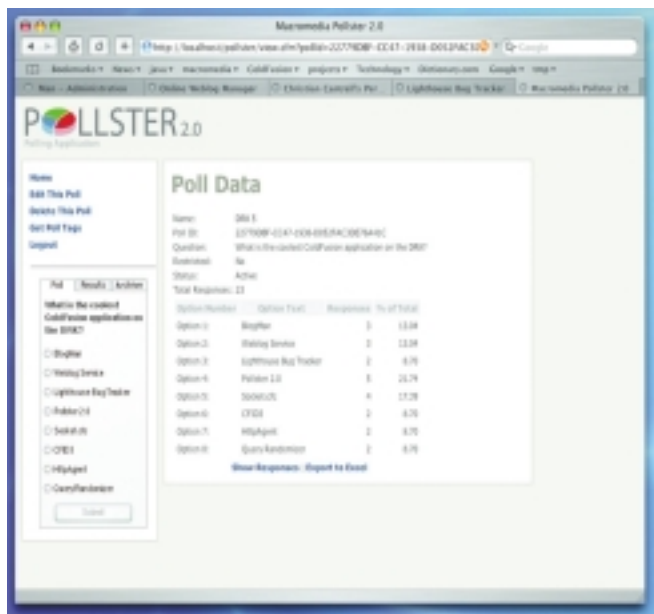


Figure 3: Pollster administration screen

figure Log4CF settings at the server or application level.

Log4CF improves a developer's or administrator's ability to monitor applications granularly. The simple API of this tag integrates with the existing ColdFusion logging paradigm, making it easy to adopt.

Blog Man

Do you have a blog yet? Need one? Blog Man (first introduced in DRK5 and then updated for DRK6) is a complete blog application written entirely in ColdFusion. The current version supports:

- All the blogging features you'd expect
- Configurable UI
- Support for Access, SQL Server, and MySQL
- Automatic archiving
- Blog posting via e-mail
- And more

This is another great ColdFusion-based blog.

Lighthouse Bug Tracker

If you write code, then you need to track bugs. It's an unfortunate fact of development life, and one we all need to deal with. Lighthouse Bug Tracker (in DRK5) is a complete bug-tracking application featuring:

- Multiple tiers, with presentation entirely separated from underlying content (allowing developers to create their own front ends, if needed)
- Simple data entry, search, and tracking screens (see Figure 2)
- XML data storage

Sure, you could write your own bug tracker, or keep using yellow sticky notes, but this app could make your life a whole lot easier.

Pollster

Pollster is a polling application (first introduced in DRK4 and then updated for DRK5), a ColdFusion back end with an interactive Flash front end. Pollster allows you to easily embed polls and surveys into your sites. An administration screen (see Figure 3) is used to define surveys and even generate the needed Flash embedding code.

You can see an example of Pollster on my blog page at www.forta.com/blog. Many others are running this useful utility, too.

Socket.cfc

ColdFusion provides tags that are used to access common Internet protocols like POP, SMTP, LDAP, HTTP, and FTP. But what if you need lower-level connections? Socket.cfc (in DRK5) provides a mechanism by which to make low-level socket calls as needed.

Here is a simple example:

```
<!--- Get instance of socket object --->
<CFOBJECT COMPONENT="socket"
    NAME="sockObj">
<!--- Connect to port 25 on mail server --->
<CFSET sockObj.Init("mail.mydomain.com", 25, false)>
<!--- Execute VRFY command --->
<CFSET response=sockObj.WriteText("VRFY #email#")>
<!--- Close connection --->
<CFSET sockObj.Close()>
```

This code uses <CFOBJECT> to obtain an instance of the socket component. The Init() method is used to open a connection to a specified host and port. The WriteText() method is used to send a command to the remote host (here an SMTP VRFY command is being sent), and any response will be stored in a variable named "response". And finally, the Close() method is used to close the connection.

Socket.cfc makes Java calls internally, but these are all hidden from you. Just instantiate the component and make method calls; it's as simple as that.

Summary

I've listed just a few of the DRK tags and applications, the ones that I use myself. There are a lot more, too, and not just for ColdFusion. If you have not taken a look at the DRKs yet, visit www.macromedia.com/software/drk/. You may just find exactly what you were looking for (or exactly what you didn't know you needed).



About the Author

Ben Forta is Macromedia's senior product evangelist and the author of numerous books, including ColdFusion MX Web Application Construction Kit and its sequel, Advanced ColdFusion MX Application Development, and is the series editor for the new "Reality ColdFusion" series. For more information visit www.forta.com.

ben@forta.com

Dreamweaver Website Development

MANY needs - ONE solution



MX Kollection for ColdFusion and PHP

Create powerful dynamic lists

- Sort, filter and page result sets
- Perform multiple deletes
- Easily create Master/Detail lists

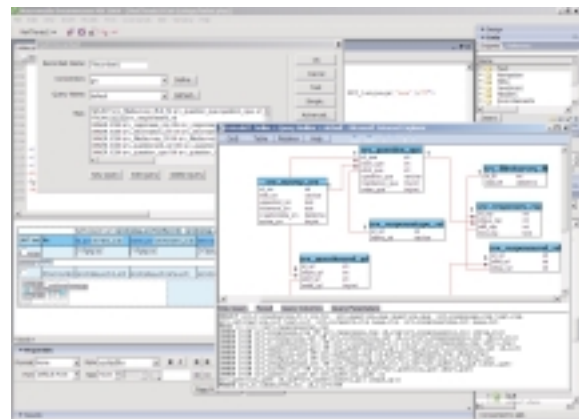
Build unified add/modify forms

- Client side and server side validation
- Insert into two tables
- Create form actions such as send mail or delete related record
- File/Image upload and resize

Create recordsets visually

- Build complex queries across multiple tables quickly

... and many more



The MX Kollection is a **suite of extensions** designed to **change the way you create dynamic web applications** with Dreamweaver MX.

ColdFusion and **PHP** developers will find our product enabling them to visually develop **e-Commerce, CMS, CRM, Backend** and other web solutions with increased efficiency, quality and capability.

Our customers think of it as **the next level in Dreamweaver MX visual software development.**

Download the demo and see the features and benefits of our extensions:

<http://www.interaktonline.com/>

MX Kollection - Professional web tools



be the pilot!

FREE SETUP on Shared
Hosting Accounts With
ColdFusion MX Support
Use Promo Code CFDJ2004



INTERMEDIA.NET

WE DARE YOU TO TAKE A FREE TEST FLIGHT!

Managing technology that runs your business is a matter of trust and control. INTERMEDIA.NET gives you both.

TRUST. Since 1995 we have been providing outstanding hosting service and technology to our clients. Don't take our word for it... take theirs.

"The support and service that you offer are nothing short of golden. The high quality of your system and service for CF customers is something one could only ever dream of." – Claude Raiola, Director, AustralianAccommodation.com Pty. Ltd.

CONTROL. We give you instant control over your site, server and account configuration changes. No more submitting requests and waiting for someone else to take action. You are in control to pilot your business through its daily needs.

BE THE PILOT. Take a free test flight and see what our HostPilot™ Control Panel offers you beyond all others. Check out our SLA guarantees. To see more testimonials and to find out about our competitive advantages, visit our Web site at www.Intermedia.NET.

Managed Hosting • Shared Hosting • Microsoft Exchange Hosting

Call us at: 1.800.379.7729 • Visit us at: WWW.INTERMEDIA.NET



HostPilot™