

# ColdFusion Developer's Journal

ColdFusionJournal.com

August 2004 Volume: 6 Issue: 8

**web services EDGE**  
conference & expo

**Web Services Edge 2005 East**  
February 15-17, 2005  
Hynes Convention Center  
Boston, MA

See page 35 for details



## Editorial

### **In with the New...**

Simon Horwith page 5

## CF Community

### **Tales from the List**

Simon Horwith page 7

## News

page 27

## CFCS

### **Caching ColdFusion Components in Shared Memory**

Jesse Skinner page 28

## CFUGs

### **ColdFusion User Groups**

page 40

RETAILERS PLEASE DISPLAY  
UNTIL OCTOBER 31, 2004

\$9.99US \$9.99CAN



**SYS-CON MEDIA**

# XSLT and ColdFusion

By Matt Woodward

page 20

## **Conferences:** Lots to Learn at CFUN

**2004** So many great sessions, it was difficult to choose

8

## **Site Work:** Using XML to Share Performing Arts Schedules A practical application of the XML features in CFMX

James Edmunds

12

## **CF101:** Creating a Remember Me

**Login** Implementing a login script on your site

Jeffrey Houser

32

## **Foundations:** Fusebox or Mach-II? A look at the strengths and weaknesses of both frameworks

Hal Helms

36

## **Load Balancing:** Web Server Load-Balancing Options Making the transition to hardware-based

Frank S. DeRienzo

38

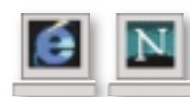
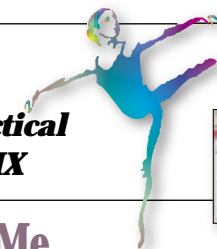
## **Vector Space:** Building a Keyword Vector Space Engine in ColdFusion Adding an extra dimension to your keyword searches

Matt Perdeaux

42



Sean Corfield





## edgmodern.com

"Macromedia® Dreamweaver® MX 2004 has helped us design and build a fully functional site which helps us immediately respond to customer needs—all without a Ph.D. in web development."

Drew Sanocki, EDGE\*MODERN, Co-founder.



## gulfstream.com

"With Dreamweaver, we'll be able to take even greater advantage of CSS, which will add huge efficiencies to how we develop and maintain our website."

Will Dent, Gulfstream Aerospace, Interactive Marketing.

 **Dreamweaver.** The highest common denominator.

Copyright © 2004 Macromedia, Inc. and its licensors. All rights reserved. Macromedia, the Macromedia logo, and Dreamweaver are trademarks or registered trademarks of Macromedia, Inc. in the U.S. and other countries. Other marks are the properties of their respective owners. Reference to any specific commercial products, processes, or services, or the use of any trade, firm, corporation name, or product depiction is for the information and convenience of the public, and does not constitute endorsement, recommendation, or favoring by Apple; Gulfstream Aerospace Corporation, a General Dynamics Company; EDGE\*MODERN; or S2D2, LLC.



Now updated and  
up to 70% faster.

See more at: [macromedia.com/go/dwupdated](http://macromedia.com/go/dwupdated)





# 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!



[www.activePDF.com](http://www.activePDF.com)

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

## editorial

## editor-in-chief

Simon Horwith [simon@sys-con.com](mailto:simon@sys-con.com)

## technical editor

Raymond Camden [raymond@sys-con.com](mailto:raymond@sys-con.com)

## executive editor

Jamie Matusow [jamie@sys-con.com](mailto:jamie@sys-con.com)

## editor

Nancy Valentine [nancy@sys-con.com](mailto:nancy@sys-con.com)

## associate editors

Gail Schultz [gail@sys-con.com](mailto:gail@sys-con.com)

Torrey Gaver [torrey@sys-con.com](mailto:torrey@sys-con.com)

## editorial assistant

Rachel Matusow [rachel@sys-con.com](mailto:rachel@sys-con.com)

## research editor

Bahadır Karuv, PhD [bahadir@sys-con.com](mailto:bahadir@sys-con.com)

## production

## production consultant

Jim Morgan [jim@sys-con.com](mailto:jim@sys-con.com)

## lead designer

Abraham Addo [abraham@sys-con.com](mailto:abraham@sys-con.com)

## art director

Alex Botero [alex@sys-con.com](mailto:alex@sys-con.com)

## associate art directors

Louis F. Cuffari [louis@sys-con.com](mailto:louis@sys-con.com)

Richard Silverberg [richards@sys-con.com](mailto:richards@sys-con.com)

Tami Beatty [tami@sys-con.com](mailto:tami@sys-con.com)

## contributors to this issue

Sean Corfield, Frank S. DeRienzo, James Edmunds,  
Hal Helms, Simon Horwith, Jeffery Houser, Matt Perdeaux,  
Jesse Skinner, Matt Woodward

## 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](mailto:kevin.collopy@edithroman.com)  
Frank Cipolla: 845 731-3832, [frank.cipolla@epostdirect.com](mailto:frank.cipolla@epostdirect.com)

For promotional reprints, contact reprint  
coordinator Kristin Kuhnle, [kristin@sys-con.com](mailto:kristin@sys-con.com).  
SYS-CON Publications, Inc., reserves the right to  
revise, republish and authorize its readers to use  
the articles submitted for publication.

All brand and product names used on these pages  
are trade names, service marks, or trademarks  
of their respective companies.

# In with the New...

When Robert Diamond asked me to take over his duties as editor-in-chief of

**ColdFusion Developer's Journal**, my initial reaction was shock. Not so much because he'd asked me to do the job, but because his editorial would not be the first thing I'd see the next time I open a copy of *CFDJ*. He's been the editor-in-chief for as long as I can remember... and a damn good one at that.

I asked Rob why he was handing over the reins, and his response was that he wanted to shift his focus towards other things at SYS-CON and that he thought it'd be good to jump-start the magazine and give it a kick in the pants. I don't want to think of it as "out with the old..." but rather simply "... in with the new." Those of you who know me know that I'm currently living in London so perhaps a more appropriate title for this editorial would be (in the words of Monty Python) "and now for something completely different...". That said, I will do my best to meet your and Rob's expectations, and fill the void that he leaves behind.

At the time of this writing, Macromedia has released one new product (Flash Lite) and two new versions of products (Contribute 3 and Flash 7.2) all within the last three weeks. They also have a new product launch and an updater to an existing product slated for release in the near future (due to NDA I can't say anything more than that). Blackstone, the next version of ColdFusion, is supposedly going to be released somewhere between the end of Q4 this year and Q1 next year – one can only assume a JRun version or updater will also be released. Although no release date has been formally announced, it's also safe to assume that we'll most likely see the official release of FlashCast in the very near future as well.

Who knows what other new products and product versions might be released before the year is through? I mention all of this because I'm excited about the new products that Macromedia has released and I'm even more excited about the impending releases.




By Simon Horwith

Blackstone, in particular, looks very promising. With more and more new products on the market, many developers are finding it difficult to keep up. Many of the new products are not only powerful in their own right, but can also be integrated with ColdFusion to offer CF developers something new and beneficial.

So what can ColdFusion developers do to keep up to date with the

latest Macromedia offerings? Helping developers to overcome this challenge is one of the main services I believe this magazine can offer and I believe *CFDJ* needs to remain sensitive to this fact and reflect it in the articles it runs.

Over the past year or two there has been a significant increase in the number of *CFDJ* articles that focus on Java/J2EE integration with CFML applications. My hope is that we can continue this trend to not only include Java integration but integration of ColdFusion with other Macromedia products as well; this, of course, while maintaining the commitment that *CFDJ* has always had to offering expert advice on "pure" CFML development topics.

In addition to integration with other technologies/solutions, I have also been considering several other column and article ideas that will present readers with information that's been scarce (or lacking) in the community. This includes giving a voice to the User Groups and possibly even to Macromedia itself in a more official capacity. If and how this happens remains to be seen, as do many other possible content offering and format evolutions. The best news of all though is that I'm open to any and all suggestions. If you have article or column ideas for the magazine, please don't hesitate to send them my way ([simon@horwith.com](mailto:simon@horwith.com)). 

## About the Author

*Simon Horwith is the new editor-in-chief of ColdFusion Developer's Journal.*

[simon@horwith.com](mailto:simon@horwith.com)

# CommonSpot™

## Efficient Content Management

fast. easy. affordable.



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

**With CommonSpot Content Server you get it all.** CommonSpot's exceptional blend of rapid deployment, ease of use, customization and scalability make it the leading ColdFusion content management solution.

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

For larger implementations, CommonSpot scales efficiently, delivering enterprise-level capabilities like replication, static content generation, multi-site clustering, personalization and custom authentication, across a diverse set of platforms.

For the past six years, PaperThin has been a leader in the ColdFusion community, and CommonSpot has been the solution of choice for organizations of all sizes, including AFL-CIO, Boeing, Kaiser Family Foundation, Ohio University, PGA.com and hundreds of others. CommonSpot's sophisticated feature set and affordable pricing are an unbeatable combination.

Call us today at **800.940.3087** to schedule a live demonstration of your site running under CommonSpot, or visit **[www.paperthin.com](http://www.paperthin.com)** to learn more.

Paper | Thin



# Tales from the List

## Everyone's gone loopy!

**F**or whatever reason, there has been a recent increase in posts on the *CFDJ* List, dealing with looping in CFML. Throughout the month *CFDJ* members wrote inquiring about how to loop over variable scopes and other structures, lists, Excel spreadsheet data,

XML data, and more!

This month I thought I'd talk about a thread that began with a post from Jean-Marc Bottin about looping over lists in CFSCRIPT. Jean-Marc wrote the list because after attending the MacroChat on Advanced ColdFusion Techniques that I gave during Macromedia Community Week, he decided to begin using CFSCRIPT more. Unfortunately, Jean-Marc couldn't find any documentation for how to do a "<CFLOOP list='>" within a CFSCRIPT block. So, he turned to the List for answers.

The obvious response that Jean-Marc received, which any of you who are familiar with CFSCRIPT would most likely also suggest, is to do something like the following (first posted by Stephen Moretti in response):

```
<CFSCRIPT>
    for (i=1; i lte listlen(theList,theDelimiter);
    i=i+1) {
        thisItem = ListGetAt(theList,i,theDelim-
        iter);
        other stuff.....
    }
</CFSCRIPT>
```

The purpose of this article is not to discuss CFSCRIPT syntax, but rather the ideas behind three discussions that ensued from responses to the thread. I've chosen to discuss these topics because they do a good job of illustrating that any task in CFML, no matter how trivial it may seem, is worth thinking about.

The first response worth discussing was a response he received that stated that you should not loop in CFSCRIPT because it's much slower than looping with tags. This is simply not true, as was pointed out by several members of the List. However, it did bring attention to several other points worth keeping

in mind. First is the fact that prior to ColdFusion MX, most series of operations executed in a CFSCRIPT block would execute much faster than their tag counterparts. This is because only one call to the parser was required to execute the commands, as opposed to one call per tag. As of ColdFusion MX, the performance differences are, for the most part, negligible. This is due to

the fact that CFML is now compiled to Java bytecode.

That said, there are two other reasons to use CFSCRIPT, which may or may not be compelling for developers. The second reason to use CFSCRIPT is that it suppresses whitespace. This can mean a significantly quicker page load time for some site visitors, and is also much more friendly for people who wish to view or parse the page source (including screen reader applications). The third reason to use CFSCRIPT is readability. CFSCRIPT code tends to be easy to read – easier than tag syntax for many people.

— continued on page 19

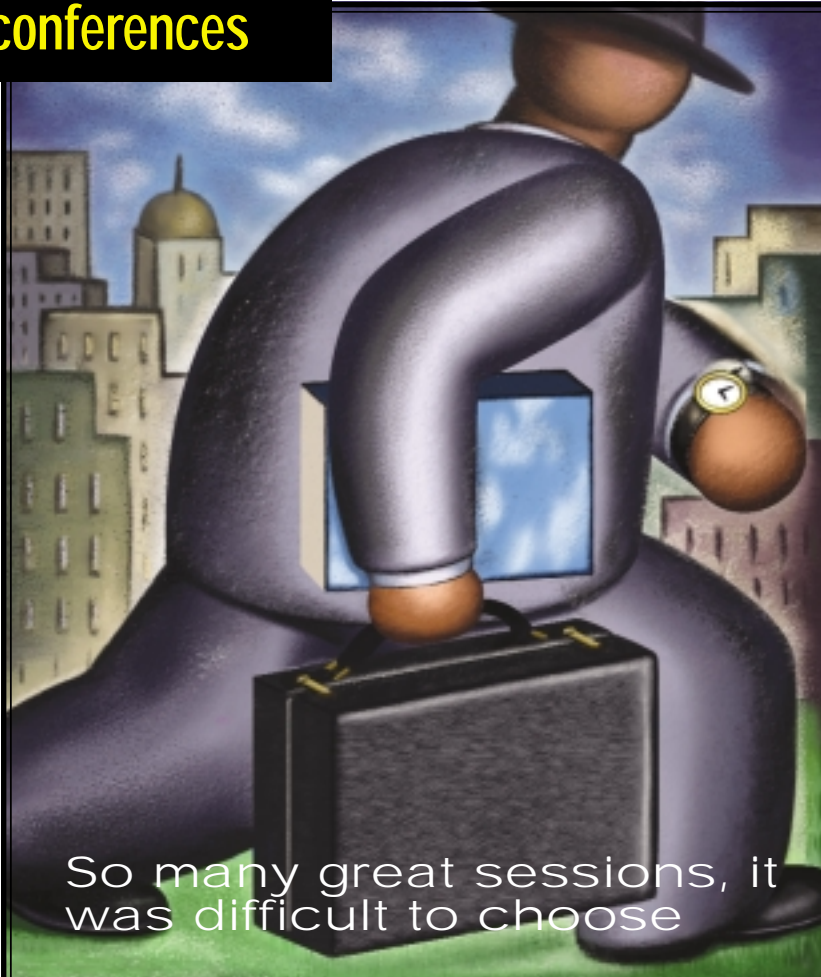
### About the Author

*Simon Horwith is co-technical editor of CFDJ, and chief technology officer of eTRILOGY Ltd., a software development company based in London, England. Simon has been using ColdFusion since version 1.5 and is a member of Team Macromedia. He is a Macromedia Certified Advanced ColdFusion and Flash developer and is a Macromedia Certified Master Instructor. In addition to administering the CFDJ List mail list and presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers.*

[simon@horwith.com](mailto:simon@horwith.com)



By Simon Horwith



# Lots to Learn at CFUN 2004

**C** FUN is the national ColdFusion and Web programming conference that Rockville, Maryland-based IT firm TeraTech

([www.teratech.com](http://www.teratech.com)) hosts each June in the DC area.

CFUN ([www.cfconf.org/cfun-04/](http://www.cfconf.org/cfun-04/)) stands for

ColdFusion User Network, and based on my trip to

CFUN-04 there were plenty of people learning, networking, and having fun doing it!

## CFUN Day Zero and Some Helium

While the conference proper didn't start until June 26, I flew in a day early to catch up with some of the other speakers at the MMUG manager meeting day and the speaker dinner. I spent a lot of the day talking to Matt Liotta about He3 – his company's new ColdFusion editor, based on Eclipse, that debuted in beta form at CFUN. The CFUN “party pack” contained the He3 beta on CD along with many other goodies. So what does He3 offer? A color-coding ColdFusion editor with tag completion – a little rough round the edges but with great things promised – and built-in Regexp and XPath panels that let you build and test your



By Sean Corfield

regular expressions and XML queries in “real time” using arbitrary snippets of text and XML (highlighting matches as you type). He3 also recognizes Mach II applications and provides an intelligent XML editor for the mach-ii.xml configuration file, showing both a source view and a tabular view of each of the sections of the file, with the ability to add and delete entries using the table view – very useful for building out the skeleton of the application. I haven't tested it yet but I understand He3 also supports Fusebox 4 and has a similarly intelligent editor for FB4 XML files.

Based on Eclipse, He3 has a variety of cool editing tricks up its sleeve, including auto-updates from RichPalette's Web site (so you'll get new features as they're made available), “quick diff” against all previously saved versions of a file (very useful to keep track of what you've been doing to a file!), integration with CVS, and so on.

Other than He3, I caught the tail end of the Macromedia User Group Managers' sessions with Ed Sullivan talking about the history and future of MMUGs, which was interesting. That was followed by the speakers' dinner at a Brazilian BBQ restaurant (where I spent more time chatting with Matt) and then the obligatory evening in the bar discussing everything CF-related. This tailed off into sessions in various rooms, with more beer, talking about Mach II, and then a late-night bitch-fest about the good, the bad, and the ugly in ColdFusion and the developer community.



## Change of Plans

I was planning to start the day with more Matt Liotta (his "Utilizing Web Services" session), followed by Ray Camden's "CFC Best Practices, Tips, and Tricks," then, after lunch, Hal Helms' BOF on methodologies, Michael Smith (standing in for Shlomy Gantz) on "Managing CF Projects," Simon Horwith on evolving from a scripter to an architect, April Fleming's "XSLT for Data Manipulation," and finally, the **CFDJ** panel that included Macromedia's Tim Buntel. At least that was my plan...



Ray Camden overcomes projector problems

## The Keynote – Stephen Shapiro of 24/7

This was kind of fun but not CF-specific. A creativity guru, Stephen Shapiro talked about some techniques for generating new solutions to problems and how to think outside the box. I'd like to have heard more about his forthcoming book, *Goal-Free Living – Passion-Filled Life*, but I guess that will have to wait for another time.

By the time I got to Matt Liotta's talk on Web services, it was completely full so I hung out and chatted with folks in the hallway. Next up was Ray Camden's session about best practices for using CFCs. He had a lot of technical problems with the projector, which unfortunately cut short his talk somewhat, but he went through some good basic tips for folks coming to CFCs. Personally I had hoped for a bit more technical depth but I think it was appropriate for the audience overall. And Ray more than maintained his sense of humor through the traumas of the projector problems!

## BOF Lunch and XSLT

Lunchtime meant a Birds Of a Feather session with Hal Helms, Ben Edwards, Jeffrey Houser, Joseph Flannigan – and me (co-opted by Hal). We talked about the

good and the bad in frameworks and methodologies. It was an interesting and lively discussion but hard to summarize any particular sentiment from the group as a whole. After lunch, Michael Smith ran Shlomy's session on managing CF projects. The Standish Group was quoted as saying that the key factors in successful projects are: user involvement, executive management support, and a clear statement of requirements. These three factors accounted for 50% of the influences!

The final session of the day was April Fleming on using XSLT for data transformation. I've never used XSLT so it was a good introduction for me. She showed code using an MS-specific XML parser (a COM object), which made the examples look more complex than they needed to be – she'd done that so the code would run on CF5 as well as CFMX (but wouldn't run on non-Windows platforms!). Her primary example was cool though, taking a single XML packet and transforming it into both HTML for display and SQL to create and populate database tables. This certainly showed the power of XSLT!

## CFDJ Panel and CF Chat

Then it was time for the *CFDJ* Panel (chaired by Jeff Peters since Robert Diamond was delayed). Charlie Arehart (New Atlanta), Tim Buntel (Macromedia), Michael Smith (TeraTech), Simon Horwith, and Hal Helms took questions from the floor. Tim said that Macromedia is looking to raise the profile of CFCs, and drive more folks to use them, by making them more accessible to beginners (e.g., through Dreamweaver behaviors that produce clean code with logic in CFCs, separated from presentation code). I would have liked to see the panel go on much longer but all good things come to an end. Stan Cox made a traditional appearance, muttering about problems with his `<blink>` tag but was, mercifully, removed



Speaker Sean Corfield, David Epler, and Sandra Clark

by security before he could disrupt the panel too much!

After the panel, Michael ran a ColdFusion version of Who Wants to Be a Millionaire but I retired to my room with Chris Philips to look at a problem he was having with SES URLs. Then it was time for the networking social event in the hotel's nightclub. A brief visit to Hal's and Ben's suite (where a poker tournament seemed to be in full swing) was followed by a fairly brief visit to the bar, followed by a not-so-brief visit to Steve Nelson's and

**"For every single session slot, there were actually two sessions I wanted to attend (sometimes three!)"**

Rey Muradaz's rooms to be entertained by Bogdan Ripa's beer bottle-opening talents (how to be creative in the absence of a proper bottle opener). Bogdan and his Interaktcrew were visiting from Romania to promote their sophisticated suite of Dreamweaver MX extensions!

## Reaching Mach II

On day 2, I had planned to attend two accessibility talks first thing, but lack of sleep got the better of me and I had to skip them, catching another two hours of much needed rest so that I could function during the rest of the day. Apologies to John Hamman and Larry Hull for missing their talks. Before my talk, I met with Daniel Dougherty, who'd been picked to interview me for five minutes, and he had some great questions. I believe TeraTech will post the various attendee/speaker interviews at some point so I'll keep y'all in suspense!

Then it was time for my Mach II talk. This is the ColdFusion OO methodology, not twice the speed of sound! A good percentage of folks in the audience were on CFMX 6.1 and were already using either Fusebox or Mach II so that was quite a change from some of my gigs. The presentation seemed to go over well and there

were some good questions from the floor – thanks to everyone who attended (and special thanks to those folks who gave me good evaluations – I got a bottle of wine at the wrap-up session for tying as “best speaker” with Charlie Arehart from New Atlanta! I’m honored!)

### Tools BOF and Blackstone Secrets

Sunday lunchtime saw a new Birds Of A Feather session added, for IDE and tool support for frameworks, led by Matt



Ben Forta talks with CFUN attendees

Liotta. He demo’d some of He3’s support for Mach II (table-based editing of the XML file). I showed a utility that renders Mach II’s event handlers as hyperlinked pages (so you can click on filter, listener, view, and event names and jump to their definitions; I haven’t made this public yet!). Jeff Peters showed a couple of tools relating to Fusebox (MindMapper and FuseMinder) and then Steve Nelson showed his test harness generation tools. The aim was to raise awareness and to get feedback about what sort of tools people wanted. One thing that wasn’t demo’d but seemed to generate interest was Dave Ross’s tool for converting XMI (the XML output from several UML modeling tools) to CFCs.

Next up was Ben Forta’s keynote on Blackstone! He raced through some of the things he’s been showing at CFUG presentations (Flash forms, PDF generation, report generation, sourceless deployment and EAR/WAR file packaging) and then gave a CFUN exclusive sneak peak: the event gateway! This is probably the most exciting and radical addition ever to ColdFusion: by writing a small amount of Java, it allows you to connect pretty much anything to ColdFusion and have external, asyn-

chronous events trigger method calls on a CFC. The example Ben showed was an agent that watched a folder for new, changed, or deleted files and automatically called the appropriate method on a CFC to populate/update a database based on the contents of the file. While this generated a lot of “ooohs” from the audience, I suspect that the real impact of this feature will take awhile to sink in – it opens up a whole new field of use for ColdFusion since this lets it process requests that are not Web-based.

### The Final Lap

One of the great things about CFUN for me was the wealth of really good sessions. For every single session slot, there were actually two sessions I wanted to attend (sometimes three!). Of course, the downside is that you just can’t get to see everything. (Apparently CFUN-05 will repeat popular sessions). I’d already had to make several hard choices and, following the keynote, I had to make another one. I decided to go hear Jeff Peters talk about “Fusebox 4 in 40 or Fewer” (instead of David Epler’s session about HTML markup for accessibility, which was my other choice). Sorry David. Jeff went through the entire life cycle process he uses, starting with wireframing and prototyping, followed by fuse analysis and circuit architecture,




The crowd at CFUN’s Sunday keynote



Sean Corfield celebrates after the last day of CFUN

and finally code generation, all supported by tools. Because Ben’s talk ran over (understandable!), Jeff’s session was a bit compressed but, although he clearly felt the time pressure, he managed to cover everything in a fairly comprehensive manner. It was very interesting to see someone else’s process, especially one so different from mine.

The final session I went to was Sandra Clark’s on “Accessible Web Forms.” She started out by demo’ing a screen reader trying to read a fairly typical Web form. The result was incomprehensible! Then she went through a long list of stuff you can do to help make forms more accessible (using fieldset, legend, label, etc.) as well as what not to do. (Don’t use accesskey – it conflicts with screen readers’ keyboard shortcuts and they don’t provide a “reset” button on the form). She got into a lot of depth and it made me realize what a complex subject this is, but I sure learned a lot from her!

There was a final general session with prizes and thank-yous and then folks began to drift off toward home. I spent most of the evening in the bar with Michael, Sandra, the guys from Interakt, Nate Nelson, and many others, discussing everything CF-related (and many things that weren’t). And then goodbyes... until next time! All in all, it was a terrific conference with some awesome material. I enjoyed myself immensely, especially talking to so many CFers! I also learned a bunch of stuff (especially from April’s “XSLT for Data Manipulation” talk and Sandra’s “Accessible Web Forms” talk!). Sandra got a well-deserved “second,” behind Charlie and me – she really is a very good speaker and loves her subject matter! See y’all at CFUN-05? 

### About the Author

*Sean Corfield is director of architecture at Macromedia and has worked in IT for over 20 years. Sean is a staunch advocate of software standards and best practices and maintains the Macromedia ColdFusion MX Coding Guidelines. Recently Sean has become a staunch advocate of Mach II. You can reach him at [www.corfield.org](http://www.corfield.org).*

[sean@corfield.org](mailto:sean@corfield.org)

# CFDynamics

A Division of Konnections Inc.

WHERE COLDFUSION EXPERTS HOST.



## RUN YOUR CFML:

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

**NOW AVAILABLE!**

# BlueDragon [6.1]



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.

## QUALITY CF 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

**SIGN UP FOR A HOSTING  
ACCOUNT TODAY!**

Use promo code: **CFDJ04H**  
and receive a **FREE T-SHIRT!**



**BlueDragon**Certified  
V.I.P. Hosting Partner



macromedia  
**ALLIANCE PARTNER**

**WWW.CFDYNAMICS.COM**  
**866 - CFDYNAMICS**  
866-233-9626





# Using XML to Share Performing Arts Schedules

A practical application of the XML features in CFMX

**D**on't you have it set up so you can just automatically pull our listings from our Web sites?" the e-mail asked.

"Not yet," I typed in my response. "But it's a great idea, and I'm working on it now."

My correspondent was the head of an organization that is a member of a five-state performing arts presenters association, a 50-member group whose Web site I have managed for the past five years. The association is made up of community and university organizations that bring touring opera, dance, theater, and music to their respective locales.

Sharing information about one another's offerings has always been an important component of the group's activities, not only to keep abreast of what's going on in the region, but also to be alert to possible cooperative routing of traveling attractions.



By James Edmunds

As any of you who work with such groups know, a consortium of nonprofits is likely to include members with a diverse range of monetary resources, staff, and technological sophistication. So any technical solutions introduced to the group are likely to be adapted in steps or stages, and must be inviting even to organizations with small resources.

There had been some attempts to create a group calendar that included events from the various member organizations, including – in the early days – having a volunteer hand type from member brochures into a spreadsheet. Even that low-tech approach was hit or miss, as some organizations neglected to mail their brochures to the poor soul doing the compiling!

About three years ago, when the last of the member organizations finally added their own local Web sites, I instituted a grouped search feature for the members, which used the free search facility offered by Atomz ([www.atomz.com](http://www.atomz.com)) to crawl the calendar listing pages of the various member organizations'

Web sites. This didn't create a single set of compiled listings, but it was at least beneficial to members wishing to see which of their colleagues might be presenting an attraction in their area of interest. I also have Atomz crawl an archive of the group's list serve e-mail so members can track mentions of artists and attractions that are still prospective and for which tours might still be building.

It was probably the Atomz search my correspondent meant when he expressed a recollection that we were "automatically" pulling listings from his Web site. But, as valuable as that service is to us, it is not creating an aggregated calendar of event listings.

The means to create an aggregated calendar are readily at hand, though. If each organization could output commonly formatted XML, these outputs could easily be combined into a single group calendar. And, if a group like ours developed a protocol for sharing this information, it could likely be useful to others.

I proposed to my group that we undertake to create what we're calling performing arts event syndication (PAES). The leaders of the group told me to set up a framework to do it.

As luck would have it, I also have as a client a consortium of performing arts presenters in one of the states that is part of the big group. This client has a couple of members large enough to also be members of the multi-state group. The single-state group already has a combined calendar of their events, entered through a simple back end. I realized that if I could generate XML from the single-state Web site for each of its members, I could turn around and capture the XML of the two members of the multistate group, for the multistate group's aggregate calendar. In other words, I could provide my own early adopters!

## RSS: Weblogs, News, and Why Not Performing Arts Events?

As a blogger (<http://poorclio.com>), I was aware of the growing use of RSS for Web log feeds as well as news feeds. I reasoned that there would be an advantage to making the first version of PAES conform to an RSS standard, so I chose RSS 2.0. In fact, the project was begun simply by using the RSS 2.0 protocol, relying at this stage on participants observing a couple of simple conventions in their RSS in order to comply with the formatting of the aggregate calendar.

In fact, one of the benefits I planned to tout to all parties involved was the value of having an RSS output that end users might subscribe to in their own news readers. In the future, our project may call upon us to create a more refined or specific protocol for PAES, in which case our members are likely to simply output both; but for now, in these early stages, we can demonstrate and build the project with RSS.

Because my background was in the newspaper business, I also envision that a publisher creating a community arts or events calendar might one day compile such listings by crawling the XML URLs of the organizations in the coverage area. Once again, though such an application might evolve its own specialized protocol, RSS seemed as good a starting point as any to demonstrate the concept.

Additionally, there is a wealth of resources available online to assist someone wishing to output RSS, in most cases specif-

ic to any of the various application servers. And, of course, there are RSS validators easily accessible on the Web.

So I was ready for the first step. Within my own client base I had a single-state group that had a couple of the organizations that would feed the multistate calendar. Why not create an RSS feed for all of the organizations in the single-state group, and then feed the two who were members of the multistate group into the aggregate calendar? This would have the side benefit of providing an RSS feed for all of the members of the single-state group, most of which aren't members of the multistate group – and which tend to be smaller organizations unlikely to have the technological resources to be out front on an issue like developing RSS feeds on their own. I could feed my own project, and also do a good turn for the members of my single-state group.

## Step One: Creating Some XML That Will Later Be Aggregated

The single-state group's Web site features a calendar of events displayed in a monthly grid calendar format, identifying events by type, city, and date. A link under each event leads to a detail page with information about the specific event, as well as the presenting organization. The database is MySQL, and the site is written in ColdFusion. The information about the events is in one table, and, for the detail pages, is joined to a table of information about the member organizations.

The table for the events is sufficient in this case to generate the information needed for our RSS output. After finding some good resources on the Web, I created a template (see Listing 1) that loops over each organization, checks to see if it has any upcoming events, and then loops through those events and writes them to an XML file.

While these XML files are valid RSS, you will note a couple of interesting things about the way dates are used. The date is actually found twice, once as part of "description" and once in the "pubDate". Also, you will note that in this case "pubDate" –

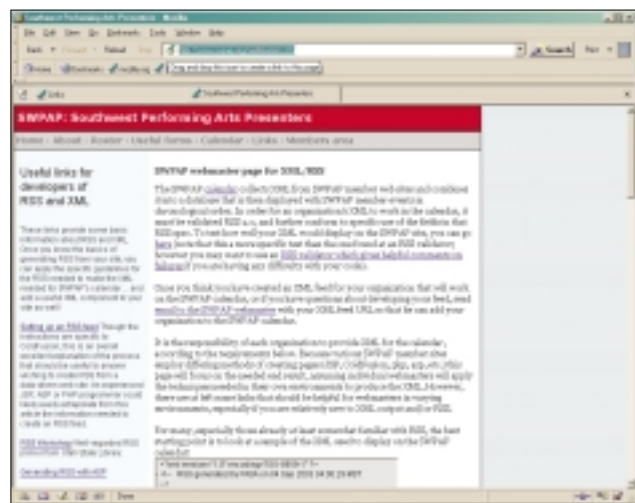


Figure 1: A page with links to information about how to generate RSS from various application servers, as well as other helpful links and information, is provided for webmasters

unlike most RSS – is not the date the information was generated, but rather the date that the event will occur. This allows the “items,” once passed into the collective database, to be sorted on the “pubDate” date. Also, if the XML is displayed in aggregators (which sometimes do not show the pubDate), readers will see the dates.

It is also worth noting that the event time is not necessarily carried by the date/time object that becomes the pubDate. My observation of the manner in which many of the member organizations of the multistate group set up their own databases led me to avoid relying on that date/time object to carry the event time. Many organizations use only the date portion for parsing event order and use a text field for the time, possibly to allow for non-time designations such as “immediately after the performance” or even “TBD.” In any case, practical experience indicated that I might have better success if I planned to rely on the event date and time to be passed through as text in the description field.

## Eating What I Cooked...

Now that I had a couple of RSS feeds of calendar information tailor-made to my needs, it was time to capture them and aggregate them into a single calendar on the multistate organization's Web site. The template I wrote for doing that was made easy by ColdFusion MX's XMLParse() function (see Listing 2).

The first thing I do is remove all earlier entries into my aggregated listings, which are in a table called XMLEvents. Then I poll the table that has information about the members of the organization, and select those that have an XML output (and also have paid their dues!). Using CFHTTP, I loop through each of them and collect the wanted XML from the designated URLs, and distribute the parsed XML into a CFQUERY insert that populates the events into the XMLEvents table. If there are errors, I generate mail, and also show error messages on

screen for any time that I call the template from the browser.

The template is scheduled to run every day, and the result is a set of similarly formatted event listings that can be sorted on date and displayed in any variety of chronological renderings.

## ... and Getting Someone Else in the Kitchen

Now that I had demonstrated that I could complete the circle on my own, it was time to start to make it work with the participation of the member organizations of the multistate group. Since I knew that roughly half the members had Web sites employing application servers, and that several different application servers were represented, I created a “helper” page for webmasters with links to sources of information about creating RSS with ColdFusion, ASP, JSP, PHP, etc. (see Figure 1). I also provided a link to an RSS feed validator, and a feed test that lets a webmaster see how his output compares to a model output.

I sent an e-mail to the multistate group's list serve, which goes to the executive directors, asking them to have their webmasters take a look at the project and contact me with any questions they might have. The first response came from a member organization whose Web site employs ASP; it took the webmaster there about 45 minutes to create the template for outputting the XML, and it's been humming along since (see Figure 2).

A few weeks ago, at the most recent meeting of the multistate group, it was my pleasure to award a prize (a macadamia nut pie from Hawaii, no less!) to the executive director of that early-adopting member organization. Now, I'm looking forward to having others join in.

## The Reality of Small Nonprofits

In my next appeal to members of the multistate group, I'll point out how relatively painless it was for the pie winners to incorporate the XML output into the function of their Web site. The technology side of creating this kind of project is often far less challenging than the organizational behavior, even in a group like the one I am working with, which is relatively progressive in its outlook. (When I first started working with the multistate group, one of the initial emphases was on getting members to use a list serve rather than calling around or sending one another batches of faxes; one of the officers told me proudly, “Honey, I check my e-mail once a month, whether it needs it or not.”)

The reality is that the kinds of nonprofit organizations that present performing arts are often lacking in staff and resources, and in many cases are removed from the source of their technological services. In cases like this, I sometimes have greener-on-the-other-side fantasies about being a corporate IT mogul, who I am tempted to imagine simply decides on what shall happen in his empire, and thereby his will is done. Come to think of it, maybe corporate IT guys have that same fantasy!

In any case, the most-likely scenario for my multistate group is that a few more early adopters will join in with their XML at my personal urging, until there is enough traction and momentum that those who aren't in the calendar feel like they

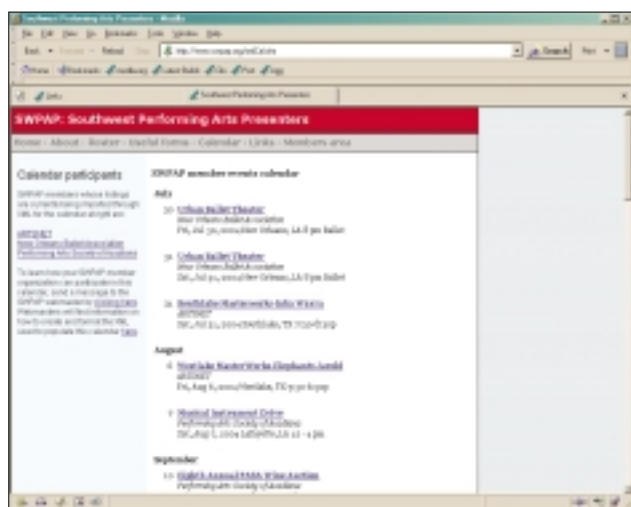


Figure 2: The finished calendar output displaying aggregated events in chronological order. As participation increases, calendar display will be altered to monthly grid format with detailed navigation and a search feature



# 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

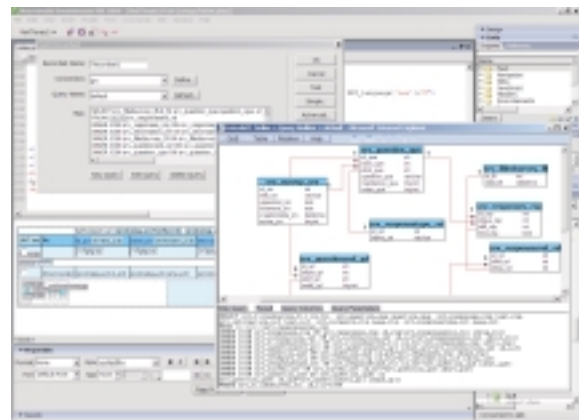
### Build unified add/modify forms

- 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




are missing an opportunity, and want to join in – even to the point of switching to an application server approach if they're not using it already, in order to be able to participate.

## Where From Here?

The benefits of publishing and sharing calendar information through XML are so great that it seems inevitable that arts organizations will all want to do this at some point. The more places an event is listed, the more people will see it, and the more tickets will be sold. Aggregation of events has the additional advantage of placing your events next to other similar events. Search engine visitors and others who find a calendar while on the hunt for say, a dance performance in one city, are good prospects to buy tickets to a dance performance in a second city, even a couple of hundred miles away. Performing arts calendar aggregation groups information by affinity as much as by location.

Additionally, middle users or republishers (perhaps in print, in the electronic media, or on the Web) could be served by having a reliable source of commonly formatted events listings, which could be aggregated and/or transformed through XSLT, etc., as needed or useful.

If a new protocol is developed for these events, either as a stand-alone PAES or as an extension of RSS, it might be able to

serve newspapers, chambers of commerce, or other organizations whose mission includes creating aggregate calendars. These calendars may include not only arts events, of course, but other community functions as well. In fact, maybe instead of standing for Performing Arts Event Syndication, one day PAES might be understood to mean Publicly Attended Event Syndication. What greater honor can a technology acronym garner than to have more than one antecedent? 

## About the Author

*James Edmunds is a freelance Internet developer and arts administration consultant living in New Iberia, Louisiana. After a career in journalism that included writing for national publications such as Newsweek and serving as editor for an alternative weekly newspaper he founded in southern Louisiana, James began to pursue a second career working with arts groups. Though he had no technology background, his interest in harnessing the power of the Internet to serve the interests of the arts led him into Internet development, an arena in which he has now gone beyond the arts to serve a general business clientele.*

[jamesedmunds@jamesedmunds.com](mailto:jamesedmunds@jamesedmunds.com)

## Listing 1:

```
<CFQUERY name="getOrgs" datasource="#variables.thedsn#" username=
"#variables.thedsnusername#" password="#variables.thedsnpassword#">
SELECT distinct
organization
FROM Events
</CFQUERY>
<CFOUTPUT>
<CFSET NumberOfOrgs = getOrgs.RecordCount>
</CFOUTPUT>

<cfloop from="1" to = "#NumberOfOrgs#" index="odx">

<CFSET TheLPNORG = getOrgs.organization[odx]>
<CFTRY>
<cfquery name="getCount" datasource="#variables.thedsn#"
username="#variables.thedsnusername#" password="#variables.
thedsnpassword#">
SELECT *
FROM Events
WHERE eventdate >= #CreateODBCDate(Now())#
AND organization = '#TheLPNORG#'
</cfquery>
<CFSET numberOfFeedItems = getCount.RecordCount>
<CFIF #variables.numberOfFeedItems# gt 10>
<cfset NumberOfFeedItems = 10>
</CFIF>

<CFIF #variables.NumberOfFeedItems# gt 0>
<cfquery name="getContent" datasource="#variables.thedsn#"
username="#variables.thedsnusername#" password="#variables.
thedsnpassword#">
SELECT *
FROM Events
WHERE eventdate >= #CreateODBCDate(Now())#
AND organization = '#TheLPNORG#'
ORDER BY eventdate
LIMIT #NumberOfFeedItems#
</cfquery>
<cfset theDatetime = "#dateFormat(now(),
```

```
"ddd, dd mmm yyyy")# #timeformat(now(),
"HH:mm:ss")# MST">
```

```
<cfsetting enablecfoutputonly="yes">
<cfsavecontent variable="theXML">
```

```
<cfoutput><?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- RSS generated by LPN on #theDatetime# -->
<rss version="2.0">
<channel>
<title>#TheLPNORG# Events</title>
<link>http://www.lapresenters.org</link>
<description>Output of event calendar of the Louisiana
Presenters Network</description>
<language>en-us</language>
<copyright>Copyright Louisiana Presenters Network</copyright>
<docs>http://www.lapresenters.org/rss/docs>
<lastBuildDate>#theDatetime#</lastBuildDate>
<image>
<title>LPN</title>
<url>http://www.lapresenters.org/images/LPNLogoTop2.gif</url>
<link>http://www.lapresenters.org</link>
</image>
</cfoutput>
```

```
<cfloop from="1" to = "#numberOfFeedItems#" index="ctr">
<cfscript>
title = replace(getContent.title[ctr],
"&", "&amp;", "ALL");
EventType = replace(getContent.EventType[ctr],
"<", "&lt;", "ALL");
EventCity = replace(getContent.EventCity[ctr],
"<", "&lt;", "ALL");
title = REReplaceNoCase(title, "<[^>]*>-/", "", "ALL");
date = dateFormat(getContent.eventdate[ctr],
"ddd, dd mmm yyyy");
time = timeformat(getContent.eventdate[ctr],
"HH:mm:ss") & " CDT";
pubDate = date & " " & time;
showdate = dateFormat(getContent.eventdate[ctr],
"ddd, mmm d, yyyy");
```



Experience

Connect

Learn

Develop

Exchange

## Get the power of experience (with the benefits of early registration).

Waves are the powerful result of critical mass, various elements coming together at a specific place and time. The experience of people coming together to shape ideas and share information is equally powerful. That's MAX: a powerful experience.

Learn the best techniques from industry experts; get current on new products; and share ideas in a forum dedicated to delivering the next wave of great websites and applications.

### Experience

Choose from over 80 different hands-on and workshop sessions, in seven tracks, to create a schedule to meet your specific needs.

### Connect

Share ideas with leading developers and designers at networking receptions, Birds-of-a-Feather sessions, and a special event.

Retain your competitive edge at general sessions where you'll hear directly from Macromedia® and other industry leaders on what's next.

**It all happens November 1-4  
in New Orleans. Be a part of it.  
Register today.**

### Early Bird Registration

Register before August 31st to save \$200 and get the best session selection.

Time is running out, register now at  
[www.macromedia.com/go/max](http://www.macromedia.com/go/max)

# MAX

The 2004 Macromedia® Conference

© 2004 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo 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.



```

showtime = getcontent.eventtime[ctr];
</cfscript>
<cfoutput>
<item>
<title>#XMLFORMAT(title)#</title>
<description>#showdate# #XMLFormat(EventCity)#, LA #showtime#
#XMLFormat(EventType)#</description>

<link>http://www.lapresenters.org/oneevent.cfm?event_id=#getContent.
event_id[ctr]</link>
<author>webmaster@lapresenters.org</author>
<pubDate>#pubDate#</pubDate>
</item>
</cfoutput>
</cfloop>
<cfoutput>
</channel>
</rss>
</cfoutput>
</cfsavecontent>
<cfscript>
theLPNorg = replace(#variables.thelpnorg#,
" ", "", "ALL");
theLPNorg = replace(#variables.thelpnorg#,
" ", "", "ALL");
</CFSCRIPT>
<cffile action="write" nameconflict="overwrite"
file="c:\inetpub\lapresenters\rss\#THELPNORG#.xml" output="#theXml#">
</CFIF>
<CFCATCH>
<CFOUTPUT>
XML failure for #THELPNORG#<br>
</CFOUTPUT>
</CFCATCH>
</CFTRY>
</cfloop>

```

## Listing 2:

```

<CFTRY>
<CFQUERY datasource="#variables.thedsn#" username="#variables.
thedsnusername#" password="#variables.thedsnpassword#">
DELETE FROM XMLEvents
WHERE XMLEVENT_ID > 0
</CFQUERY>

<CFQUERY name="getTheXMLs" datasource="#variables.thedsn#"
username="#variables.thedsnusername#" password="#variables.
thedsnpassword#">
SELECT member_id,organization,thexml
FROM members
WHERE thexml <> '' AND renewaldue > #variables.renewalthreshold#
</CFQUERY>

<CFSET HowManyXMLs = getTheXMLs.RecordCount>

<cfloop from="1" to="#variables.HowManyXMLs#" index="xx">
<CFOUTPUT query="getTheXMLs" startrow="#xx#" maxrows="1">

<CFTRY>
<cfhttp url="#thexml#" method="get">
<cfset myDoc=XMLParse(CFHTTP.FileContent)>
<cfset Items=myDoc.rss.channel.XMLChildren>
<cfset myItemsArrayLength=ArrayLen(Items)-8>

<cfloop from="1" to="#myItemsArrayLength#" index="idx">
<CFSET theTitle = #myDoc.rss.channel.item[idx].Title.XMLText#>
<CFSET theDescription =

```

```

#myDoc.rss.channel.item[idx].Description.XMLText#>
<CFSET theLink = #myDoc.rss.channel.item[idx].Link.XMLText#>

<CFSET thepubDate = #ParseDateTime(myDoc.rss.channel.item[idx].
pubDate.XMLText)#>
<CFSET theOrganization = #organization#>
<CFQUERY name="addItem" datasource="#variables.thedsn#"
username="#variables.thedsnusername#" password="#variables.
thedsnpassword#">
INSERT INTO XMLEvents (Title,Description,Link,pubDate,organization)
VALUES

('#TheTitle#','#TheDescription#','#TheLink#','#ThePubDate#','#
TheOrganization#')
</CFQUERY>
</cfloop>
#xx# #thexml#<br>
<br>

<CFCATCH>
<CFMAIL to="#variables.erroremail#"
from="#variables.erroremail#"
subject="XMLCAL within loop"
server="#variables.themailserver#"
type="HTML">

URL was: #thexml#<br>

Loop number #xx#<br>
<br>

Didn't load. A #cfcatch.type# exception occurred.
#cfcatch.message# #cfcatch.Detail#<br>
<br>

<CFDUMP var="#cfcatch.tagcontext#">

</CFMAIL>
Failure for URL: #thexml#<br>
Loop number: #xx#<br>
<br>
</CFCATCH>
</CFTRY>

</CFOUTPUT>
</cfloop>

<CFCATCH type="any">
<CFMAIL to="#variables.erroremail#"
from="#variables.erroremail#"
subject="CombineXML"
server="#variables.themailserver#"
type="HTML">
Didn't load. A #cfcatch.type# exception occurred.
#cfcatch.message# #cfcatch.Detail#<br><br>
<br>
This message occurs if any of the XML parse loops fails.
<br>

<CFDUMP var="#cfcatch.tagcontext#">

</CFMAIL>
Some sort of failure. Possibly a bad XML URL or site down when polled
for XML.
</CFCATCH>

</CFTRY>

```

Download the Code...  
Go to [www.coldfusionjournal.com](http://www.coldfusionjournal.com)

Obviously, there are other reasons you may want to use CFSCRIPT, such as already being familiar with ECMA Script syntax.

The second response that sparked discussion regarded the evaluation of the list length in the “for” clause. One of the things I mentioned in a post was that the second condition in the “for” statement is evaluated on every pass. To illustrate this, run the following code, which loops from 1 to the length of a list and deletes the last entry in the list on each pass:

```
<cfscript>
foo =
"a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z";
for (i = 1; i lte listLen(variables.foo); i = i + 1){
    writeOutput("Loop " & variables.i & "
of " & listLen(variables.foo) & "<br>");
    foo = listDeleteAt(variables.foo,
listLen(variables.foo), ",");
}
</cfscript>
```

Because the listLen() is evaluated on every pass, the output would look like the following:

```
Loop 1 of 26
Loop 2 of 25
Loop 3 of 24
Loop 4 of 23
Loop 5 of 22
Loop 6 of 21
Loop 7 of 20
Loop 8 of 19
Loop 9 of 18
```

```
Loop 10 of 17
Loop 11 of 16
Loop 12 of 15
Loop 13 of 14
```

While this is not a massive amount of overhead, it is also unnecessary and could easily be avoided. If the length of the list was guaranteed not to change within the loop, a variable could be set to the length of the list prior to looping and the “for” condition would loop as long as “i” is less than or equal to that variable. It would look like so:


```
foo =
"a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s,t,u,v,w,x,y,z";
bar = listLen(variables.foo);
for (i = 1; i lte variables.bar; i = i + 1){...}
```

The third response to the thread that I wanted to discuss was an oversimplified statement I made that, “In general, you shouldn’t loop... PERIOD.” Making such a simple yet bold statement raised a few eyebrows (notably Ray Camden’s). It may sound odd, and there is an immediate urge to say “but what about...”, but it’s still a perfectly valid statement. Of course, the

**“You should always stop and ask yourself whether or not it’s absolutely necessary to loop in your code”**

statement should be qualified with “Not if you don’t have to” (which the post did state as well).

Obviously, there is a time and a place to loop. Every programming construct, every tag and function in CFML, is the correct tool to use in some scenario. Obviously some are used more than others, and looping is one of those things. The fact remains that you should always stop and ask yourself whether or not it’s absolutely necessary to loop in your code. There are even several functions that help you avoid looping (Query functions such as valueList() and quotedValueList(); Array functions such as arrayAvg() and arrayToList(); and Structure functions such as structKeyList() and structCount(), etc.). If looping is unavoidable (and there are often times when nothing’s wrong with that) then pay close attention to the code inside the loop. After all, this is code that will be executed repeatedly per user request so make it as efficient as possible.

In addition to these valuable lessons about looping, I’d like to reiterate a statement I made at the beginning of the article. Any task in CFML, no matter how trivial it may seem, is worth thinking about. This isn’t really specific to CFML and it might sound like common sense, but I am always surprised how few people pay little attention to small details in their code. Every line of code is worth evaluating not only syntactically but for causality as well. Too many developers take “simple tasks” and “best practices” for granted. To these people I say, “Open your eyes and question everything.” 

**FREE\*CD!**  
(\$198.00  
VALUE!)

*Secrets of the Java Masters*  
Every **JDJ** Article on One CD!



## — The Complete Works —

CD is edited by **JDJ** Editor-in-Chief Alan Williamson and organized into 33 chapters containing more than 1500 exclusive **JDJ** articles!

All in an easy-to-navigate HTML format! **BONUS: Full source code included!**

ORDER AT [WWW.SYSCON.COM/FREECD](http://WWW.SYSCON.COM/FREECD)

\*PLUS \$9.95 SHIPPING AND PROCESSING (U.S. ONLY)



Only from the World's Leading i-Technology Publisher

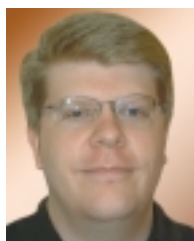
# XSLT and ColdFusion

Whipping your XML data into shape

**C**oldFusion MX offers a simple and easy way to unleash the power of XSLT for manipulating your XML data. Here's how.

From Web services to news and blog data feeds to configuration files, XML is everywhere these days. Far from the buzzword it was when the W3C approved the standard in 1998, XML is now the primary means of data exchange for many organizations and has become the *lingua franca* of text data in Web application development. Although most of us are aware of how important and ubiquitous XML has become, effective methods for using and manipulating XML data may still be a mystery.

The notion of dealing with text data may conjure up nightmarish visions of parsing comma-delimited text files, but, thankfully the days of hunting for line breaks and counting characters are long gone. The hierarchical nature of XML data makes it easy to read for both humans and computers. What XML lacks is the means to manipulate and search itself. Enter XSLT, which provides a powerful way to search (using XPath statements) and to transform XML data from one form into another and – true to form – ColdFusion MX makes using XSLT extremely simple and accessible.



By Matt Woodward

In this article I'll describe how XSLT can be used to transform raw XML data into HTML. We'll start with a basic example that uses simple XSLT pattern matching to display XML data as HTML. Then, we'll move on to a slightly more advanced example that includes conditional logic and sorting to make our XML data even more useful.

## XML: Why Use It?

If you haven't worked with XML data yet, you may be wondering why anyone would use XML, as opposed to a database, for storing and retrieving data. One reason might be for data exchange. Because XML is platform-neutral and highly structured, it's a great way to exchange data between applications, especially when accessing a database directly isn't an option. Web services are a great example of this. With the proliferation of Web services you may find one that's perfect for your needs, but because Web services return XML data, you'll need to transform it for use in your applications.

You may also find that even when you don't need to share data, the use of XML may simplify data storage and retrieval. I recently developed an application for a large paint supply company. Each of their product types had widely differing attributes. Based on the data itself, and how it would be used, it made sense to store the product details as XML rather than as separate fields



in the database. I kept the high-level attributes (name, catalog number, etc.) as database fields but decided XML was the perfect way to store the more unwieldy product details. Once this XML data was created and stored, however, I needed a way to manipulate and present it to different types of users.

## XML, Meet XSLT

You probably noticed a recurring theme in the previous two paragraphs. More often than not, XML data will have to be manipulated to suit your purposes, and, if you want to display XML data to your users, chances are they won't appreciate a simple dump to their browsers. Because XML is just text, you could parse the data the old-fashioned way, but with XSLT we can manipulate and transform XML data in far more powerful ways.

XSLT stands for "Extensible Stylesheet Language Transformations," and, as you may have already surmised, XSLT's job is to transform XML data from one form into another. This might mean taking one XML format and converting it to another (the purpose XSLT was originally designed to fulfill), but XSLT is powerful and flexible enough to transform XML into practically any format you may need.

Now you know XSLT's purpose and potential, but you may still be wondering exactly what it is. At its most fundamental level, XSLT is a "flavor" of XML, meaning that XSLT stylesheets are written in XML and must meet all of the requirements of the XML standard. XSLT is also a language, so it has many familiar programming language constructs, such as conditional statements and loops.

XSLT's core purpose is to modify XML documents based on patterns (defined using XPath syntax) matched in the XML data. XSLT stylesheets are typically nothing more than a set of rules that tell the XSL processor to match a pattern in an XML document and transform the data within the matching section using the instructions in the XSLT stylesheet. In a sense, XSLT does for XML what regular expressions do for plain text, only much more powerfully and elegantly.

Don't be concerned if this seems complicated; the interaction between XML and XSLT will become quite clear through a few simple examples. The beauty of working with XML and XSLT in ColdFusion MX is that all of the complexity of XML and XSL processors is handled by the ColdFusion server. In fact, aside from writing XSLT stylesheets, we need to concern ourselves only with a single ColdFusion function to unleash the power of XSLT. (For the remainder of the article I'm assuming you have some familiarity with XML concepts; if you need a refresher, please see [www.macromedia.com/devnet/topics/xml.html](http://www.macromedia.com/devnet/topics/xml.html) or one of the resources listed at the end of this article.)

## A Simple Transformation

For our first foray into the world of XSLT, let's consider a very simple example. Imagine that

you're the Web developer for your local zoo. Although your workplace may seem like a zoo on occasion, I'm using the word "zoo" literally in this case, so we'll be dealing with animals. (No, I'm not referring to anyone you work with!) Your task is to display an HTML list of animals at the zoo, but the zoo's database administrator guards her database the way a mother tiger guards her young, so the only way she'll provide you with data is as XML. Listing 1 shows the XML data you receive from your DBA. Although you could take the advice of Lazy (the zoo's resident sloth) and send the XML data directly to the user's browser, the end result isn't particularly pretty (see Figure 1).

Lazy has gotten you into trouble before, so you're going to ignore him this time and use XSLT to transform the XML data into HTML. This is a very common use of XSLT. Most modern browsers support this type of transformation directly within the browser, but because older browsers don't support XSLT and the syntax and available functionality may vary from browser to browser, we're going to use ColdFusion's built-in XSLT processor.

## Adding Style to Substance

One of XSLT's main strengths is pattern matching, so our first task is to create a match pattern in our XSLT stylesheet and give it instructions to execute when it finds the matching XML data. In order to keep this example simple and focus on the basic template matching capabilities of XSLT, we'll present the data in the order in which we receive it. (We'll investigate some other possibilities later.) Listing 2 shows an XSLT stylesheet that transforms the XML data into a simple HTML table.

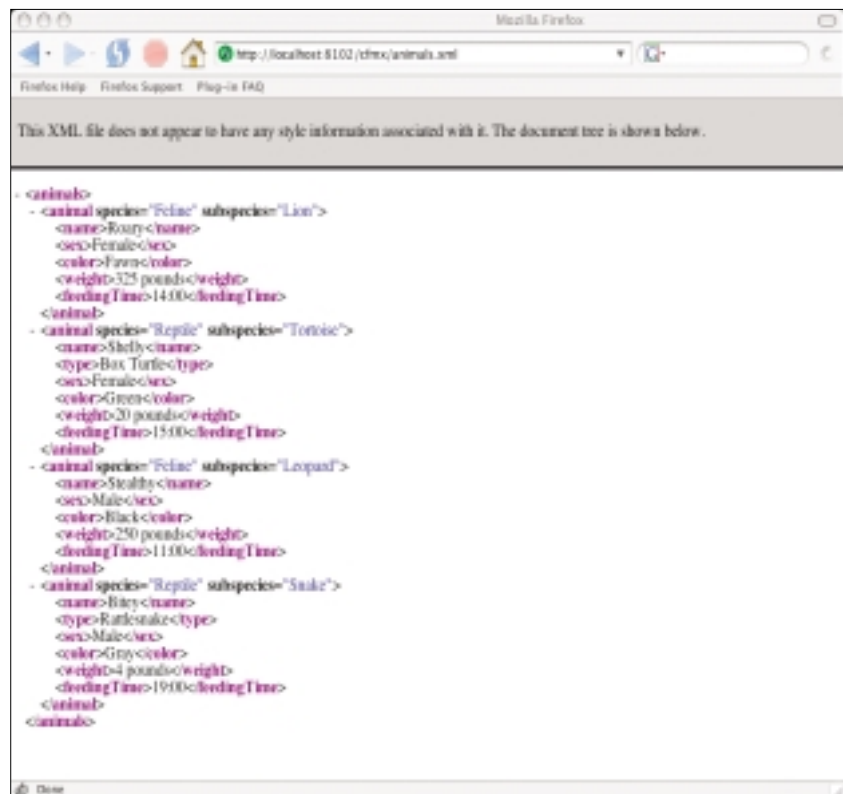


Figure 1: Browser display of raw XML data

If you haven't worked with XSLT before, this may seem a bit foreign, so let's walk through it. At the top of the document is an `<xsl:stylesheet>` element that contains a couple of attributes. For our purposes you don't need to know anything about this element except that it has to be present in exactly this format in order for some XSLT processors (including the Apache Xalan processor that's built into ColdFusion) to work correctly.

Following the first line is an `<xsl:output>` element that tells the XSL processor what to expect within the document. The W3C's XSLT specification defines xml, html, and text as valid output methods, so in our case we use html.

Next, we get to the heart of XSLT: template matching. The `<xsl:template match="/animals">` instruction tells the XSLT processor to start at the top of our XML document and find the `<animals>` element. Conceptually, the use of "/" in XSLT is similar to referencing a Web server's document root by using "/", so this tells the XSLT processor to start at the top (the "root" node) of the document. The code following the `<xsl:template>` tag is a series of output directives that are processed once a match is found, so this is where we place the HTML code that will begin to build our page.

Match patterns in XSLT are defined using XPath. According to the W3C, XPath is "a language for addressing parts of an XML document." Another language? Technically, yes, XPath is a separate language. Luckily we don't have to know much about it to use it effectively, so I'm going to keep the dive into XPath relatively shallow for the purposes of this article.

## Retrieving the Details

Following the basic HTML code is another XSLT instruction, `<xsl:for-each select="animal">`. If you think this might be a looping instruction, you're right! (Reward yourself with a trip to your local zoo, but please don't feed the animals.) Because our `<xsl:template match="/animals">` instruction put us immediately inside the `<animals>` element (this is also known as a "node"), `<xsl:for-each select="animal">` tells the XSLT processor to find each `<animal>` element nested within the `<animals>` node and output the HTML within the loop for each animal. The lack of a "/" in this select is conceptually similar to a relative file path; since we're already inside `<animals>`, our match pattern is simply "animal."

Note that there are numerous ways to achieve the same result. One method is to use an `<xsl:apply-templates>` instruction that corresponds to a separate `<xsl:template match="something">` instruction within the same stylesheet. Both because I wanted to introduce `<xsl:for-each>` and also



Species	Sub-Species	Name	Type	Sex	Color	Weight	Feeding Time
Polar	Bear	Barry	Female	Female	Grey	325 pounds	14:00
Koala	Tristan	Stacy	Male	Female	Grey	20 pounds	15:00
Polar	Kangaroo	Stacy	Male	Black	250 pounds	11:00	
Koala	Snake	Eric	Male	Grey	4 pounds	19:00	

Figure 2: Using XSLT and ColdFusion to transform the data produces a somewhat more user-friendly output

due to some changes we're going to make to our stylesheet in a moment, I opted for the loop here as opposed to another template match.

Inside the `<xsl:for-each>` loop we see the last of our new XSLT instructions, `<xsl:value-of>`, which tells the XSLT processor to retrieve particular pieces of data from the XML. Data in XML can be stored in two basic ways: as an attribute or as an element. Attributes are name/value pairs that are within an XML tag, whereas elements are separate tag pairs. The value of an element is the text between the element's opening and closing tags. This is admittedly simplified, but for the purposes of this article further distinctions aren't necessary.

To retrieve the value of an attribute (a name/value pair that's within an opening tag), simply prefix the name of the attribute with an "@" symbol in the select portion of the `<xsl:value-of>` instruction. To retrieve the value of an animal's "species" attribute for example, we use the following:

```
<xsl:value-of select="@species" />
```

Retrieving the value of elements is quite similar. Omit the "@" symbol from the select instruction, use the name of the element as the select value, and XSLT retrieves all of the text between the element's tag pair:

```
<xsl:value-of select="name" />
```

Before moving on, let's reinforce our budding XSLT knowledge by comparing the `<xsl:for-each>` loop and XSLT data retrieval to something more familiar to ColdFusion programmers. If we had retrieved this data from a database using cfquery, we would output our table rows like so:

```
<cfoutput query="animals">
  <tr>
    <td>#species#</td>
    <td>#subspecies#</td>
    <td>#name#</td>
    <!-- etc. -->
  </tr>
</cfoutput>
```

This is functionally equivalent to our XSLT `<for-each>` statement:

```
<xsl:for-each select="animal">
  <tr>
    <td><xsl:value-of select="@species" /></td>
    <td><xsl:value-of select="@subspecies" /></td>
    <td><xsl:value-of select="name" /></td>
    <!-- etc. -->
  </tr>
</xsl:for-each>
```

## Outputting the Results

Now for the easy part: using ColdFusion to apply our XSLT stylesheet to our XML data and output the results. XSLT isn't terribly complex but it may be unfamiliar to many of you, so

# Complete source code and asset management in Dreamweaver MX—now possible with Surround SCM.

Dreamweaver users know a beautiful Web-based product is only skin deep. Underneath, it's a tangle of hundreds or thousands of ever changing source files. Without a good development process and strong tools, bad things happen. Surround SCM can help.

Surround SCM lets you...

*Track multiple versions of your source files and easily compare and merge source code changes.*

*Check out files for exclusive use or work in private workspaces when collaborating on a team.*

*Automatically notify team members of changes to source files—push changes through your organization.*

*View complete audit trails of which files changed, why, and by whom.*

*Associate source code changes with feature requests, defects or change requests (requires additional purchase of TestTrack Pro).*

*Remotely access your source code repository from Dreamweaver MX.*

Surround SCM adds flexible source code and digital asset control, powerful version control, and secure remote file access to Dreamweaver MX. Whether you are a team of one or one hundred, Surround SCM makes it easier to manage your source files, letting you focus on creating beautiful Web-based products.

## Features:

Complete source code and digital asset control with private workspaces, automatic merging, role-based security and more.

IDE integration with Dreamweaver MX, JBuilder, Visual Studio, and other leading Web development tools.

Fast and secure remote access to your source files—work from anywhere.

Advanced branching and email notifications put you in complete control of your process.

External application triggers let you integrate Surround SCM into your Web site and product development processes.

Support for comprehensive issue management with TestTrack Pro—link changes to change requests, bug reports, feature requests and more.

Scalable and reliable cross-platform, client/server solution supports Windows, Linux, Solaris, and Mac OS X.

## Want to learn more?



Download Seapine's just-released white paper, **Change Management and Dreamweaver**, to discover tips, tricks and best practices for achieving full software configuration functionality. Visit [www.seapine.com/whitepaper.php](http://www.seapine.com/whitepaper.php) and enter code WM0604 to receive your copy today.

[www.seapine.com](http://www.seapine.com)  
1-888-683-6456





thankfully ColdFusion does the rest of the work for us in three easy steps (see Listing 3 for the entire file). First, we read the XML data:

```
<cffile action="read" file="#ExpandPath('.')#/animals.xml"
variable="animalsXml" />
```

Next, we read the XSLT stylesheet:

```
<cffile action="read" file="#ExpandPath('.')#/animalsHtml.xsl"
variable="animalsXsl" />
```

Finally, we use the `XmlTransform()` function to transform the XML data and output the results:

```
<cfoutput>#XmlTransform(animalsXml, animalsXsl)#</cfoutput>
```

Voila! You've just magically transformed XML data into HTML, with a little help from ColdFusion (see Figure 2).

This example assumes the XML and XSLT documents are retrieved using `cffile`, but this data can be retrieved other ways, such as from a database or with `cfhttp`. As long as the first variable passed to `XmlTransform()` is XML text or a ColdFusion XML variable, and the second variable is XSLT, ColdFusion handles the rest.

## Felines and Reptiles Don't Mix: Another Transformation

So far, so good. We're outputting XML data as HTML. But the animals are getting restless. Felines and reptiles are comingling in our output, and when it comes down to it, this simple list isn't particularly helpful. It's more or less an XML data dump in sheep's clothing (a.k.a. HTML). Fortunately, we can use XSLT to make this data more useful.

Let's imagine that the zookeepers for the felines want a feline-only listing and – as an additional unreasonable demand on you – they want the felines listed in order of feeding time so they can better manage their duties. With traditional text manipulation this would be quite a chore, but with XSLT this task is rather trivial. You don't even have to ask your DBA for a different data feed.

Let's extend our recently acquired XSLT pattern-matching skills and instead of outputting all of the animals, we'll output only `<animal>` elements for which the species attribute is "Feline". Then we'll sort the felines by the `<feedingTime>` ele-



Species	Sub-Species	Name	Sex	Color	Weight	Feeding Time
Feline	Leopard	Snoddy	Male	Black	250 pounds	1100
Feline	Lion	Roupy	Female	Fawn	325 pounds	1400

Figure 3: The final version satisfies even the "unreasonable demands"

ment and we'll have our feline keepers purring. We'll also update the HTML header information so our feline keepers know that this is their list. Listing 4 shows the updated XSLT stylesheet.

Most of Listing 4 should look familiar. The first addition is our sort tag, which is simple yet extremely powerful. `<xsl:sort select="feedingTime" />` tells the XSLT processor to perform an ascending sort on the elements within the for-each loop, based on the value of the `<feedingTime>` element. If you've ever dealt with writing your own sorting functionality, you'll appreciate the power of this simple XSLT tag.

The other addition is `<xsl:if>`, which as you might guess is a conditional instruction. `<xsl:if test="@species='Feline'">` tells the XSLT processor, "If the species attribute of this animal is Feline, output the following." If the test fails, the XSLT processor skips the output within the `<xsl:if>` tag for the current loop iteration. XSLT doesn't have a corresponding `<xsl:else>` instruction, although `<xsl:choose>`, `<xsl:when>`, and `<xsl:otherwise>` can be used to create a switch-like statement, offering additional power for conditional processing.

To use ColdFusion to output our newly transformed data, we simply follow the steps outlined above and replace the original XSLT stylesheet with the new one (see Listing 5). Yes, it's really that simple! (See Figure 3.)

## Conclusion

I hope this brief introduction to XSLT has at least piqued your interest and taught you a little about this powerful partner to XML. XSLT extends well beyond what I could cover here, so I encourage you to investigate further. If you're working with XML data, XSLT can make your life far easier by opening up possibilities for XML data transformation that would otherwise be difficult or impossible to achieve. (See Figure 3.)

## Resources

- Tidwell, D. (2001). *XSLT: Mastering XML Transformations*. O'Reilly.
- Mangano, S. (2003). *XSLT Cookbook*. O'Reilly.
- Horwith, S. (2004). Working With XML in ColdFusion: [www.how2cf.com/files/papers/cfxml.pdf](http://www.how2cf.com/files/papers/cfxml.pdf)
- XSLT Tutorial: [www.w3schools.com/xsl/default.asp](http://www.w3schools.com/xsl/default.asp)
- W3C XSLT Recommendation: [www.w3.org/TR/xslt](http://www.w3.org/TR/xslt)
- "What is XSLT?" <http://xml.com/pub/a/2000/08/holman/index.html>
- XSLT Recipe of the Day: [www.xml.com/cookbooks/xsltckbk/solution.csp?day=1](http://www.xml.com/cookbooks/xsltckbk/solution.csp?day=1)
- Macromedia DevNet XML Topic Center: [www.macromedia.com/devnet/topics/xml.html](http://www.macromedia.com/devnet/topics/xml.html)

## About the Author

Matt Woodward is a Web application developer for i2 Technologies in Dallas, and also works as a consultant through his company, Sixth Floor Software. He is a Macromedia Certified ColdFusion developer, a member of Team Macromedia, and has been working with ColdFusion since 1996. In addition to his ColdFusion work, Matt also develops in Java and PHP.

[matt@sixthfloorsoftware.com](mailto:matt@sixthfloorsoftware.com)

# September 18th & 19th 2004 Washington DC

Come to the fifth annual Fusebox conference. Learn from the experts, network with your peers and see the latest Fusebox technology. Both Beginner and Advanced sessions. Just \$199.

## Fusebox2004



**Hal Helms,  
Jeff Peters,  
Sandra Clark,  
Michael Smith,  
and more...**

[www.cfconf.org/fusebox2004/](http://www.cfconf.org/fusebox2004/)



**TeraTech Inc**

[www.teratech.com](http://www.teratech.com)  
301.424.3903

**COLD FUSION** Developer's  
Journal  
halhelms 

**Listing 1: animals.xml**

```
<?xml version="1.0" encoding="utf-8"?>
<animals>
  <animal species="Feline" subspecies="Lion">
    <name>Roary</name>
    <sex>Female</sex>
    <color>Fawn</color>
    <weight>325 pounds</weight>
    <feedingTime>14:00</feedingTime>
  </animal>
  <animal species="Reptile" subspecies="Tortoise">
    <name>Shelly</name>
    <type>Box Turtle</type>
    <sex>Female</sex>
    <color>Green</color>
    <weight>20 pounds</weight>
    <feedingTime>15:00</feedingTime>
  </animal>
  <animal species="Feline" subspecies="Leopard">
    <name>Stealthy</name>
    <sex>Male</sex>
    <color>Black</color>
    <weight>250 pounds</weight>
    <feedingTime>11:00</feedingTime>
  </animal>
  <animal species="Reptile" subspecies="Snake">
    <name>Bitey</name>
    <type>Rattlesnake</type>
    <sex>Male</sex>
    <color>Gray</color>
    <weight>4 pounds</weight>
    <feedingTime>19:00</feedingTime>
  </animal>
</animals>
```

**Listing 2: animalsHtml.xml**

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="html" />
  <xsl:template match="/animals">
    <html>
      <head>
        <title>Our Zoo Animals</title>
      </head>
      <body>
        <h3>Our Zoo Animals</h3>
        <table border="0" width="80%" cellpadding="2" cellspacing="1" bgcolor="#999999">
          <tr bgcolor="#dedede">
            <th>Species</th>
            <th>Sub-Species</th>
            <th>Name</th>
            <th>Type</th>
            <th>Sex</th>
            <th>Color</th>
            <th>Weight</th>
            <th>Feeding Time</th>
          </tr>
          <xsl:for-each select="animal">
            <tr bgcolor="#ffffff">
              <td><xsl:value-of select="@species" /></td>
              <td><xsl:value-of select="@subspecies" /></td>
              <td><xsl:value-of select="name" /></td>
              <td><xsl:value-of select="type" /></td>
              <td><xsl:value-of select="sex" /></td>
              <td><xsl:value-of select="color" /></td>
              <td><xsl:value-of select="weight" /></td>
              <td><xsl:value-of select="feedingTime" /></td>
            </tr>
          </xsl:for-each>
        </table>
      </body>
    </html>
```

```
</html>
</xsl:template>
</xsl:stylesheet>
```

**Listing 3: displayAnimals.cfm**

```
<!-- read in animals.xml -->
<cffile action="read" file="#ExpandPath('.')#/animals.xml"
variable="animalsXml" />

<!-- read in animalsHtml.xml -->
<cffile action="read" file="#ExpandPath('.')#/animalsHtml.xml"
variable="animalsXsl" />

<!-- transform and output -->
<cfoutput>#XmlTransform(animalsXml, animalsXsl)#</cfoutput>
```

**Listing 4: felinesHtml.xml**

```
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
  <xsl:output method="html" />
  <xsl:template match="/animals">
    <html>
      <head>
        <title>Feline Feeding Schedule</title>
      </head>
      <body>
        <h3>Feline Feeding Schedule</h3>
        <table border="0" width="80%" cellpadding="2" cellspacing="1" bgcolor="#999999">
          <tr bgcolor="#dedede">
            <th>Species</th>
            <th>Sub-Species</th>
            <th>Name</th>
            <th>Sex</th>
            <th>Color</th>
            <th>Weight</th>
            <th>Feeding Time</th>
          </tr>
          <xsl:for-each select="animal">
            <xsl:sort select="feedingTime" />
            <xsl:if test="@species='Feline'">
              <tr bgcolor="#ffffff">
                <td><xsl:value-of select="@species" /></td>
                <td><xsl:value-of select="@subspecies" /></td>
                <td><xsl:value-of select="name" /></td>
                <td><xsl:value-of select="sex" /></td>
                <td><xsl:value-of select="color" /></td>
                <td><xsl:value-of select="weight" /></td>
                <td><xsl:value-of select="feedingTime" /></td>
              </tr>
            </xsl:if>
          </xsl:for-each>
        </table>
      </body>
    </html>
  </xsl:template>
</xsl:stylesheet>
```

**Listing 5: displayFelines.cfm**

```
<!-- read in animals.xml -->
<cffile action="read" file="#ExpandPath('.')#/animals.xml"
variable="animalsXml" />

<!-- read in felinesHtml.xml -->
<cffile action="read" file="#ExpandPath('.')#/felinesHtml.xml"
variable="felinesXsl" />

<!-- transform and output -->
<cfoutput>#XmlTransform(animalsXml, felinesXsl)#</cfoutput>
```

**Download the Code...**  
Go to [www.coldfusionjournal.com](http://www.coldfusionjournal.com)



## Fifth Annual Fusebox Conference

(Washington, DC) – The fifth annual Fusebox conference will be held September 18–19, 2004, in the Washington, DC area. Fusebox is a free programming methodology for ColdFusion that improves program maintenance and makes for more satisfied clients. Speakers include Hal Helms, Jeff Peters, Sandra Clark, and Michael Smith. The conference costs \$199. [www.cfconf.org/fusebox2004](http://www.cfconf.org/fusebox2004).

## Macromedia Delivers Update of Flash MX 2004

(San Francisco) – Macromedia has announced the immediate availability of an update for Macromedia Flash MX 2004 and Flash MX Professional 2004 that offers improved stability and more comprehensive documentation. The update (version 7.2) is available to current Flash MX 2004 and Flash MX Professional 2004 customers for no additional charge at [www.macromedia.com/go/updates](http://www.macromedia.com/go/updates).

This update to Flash MX 2004 increases developer productivity by reducing initial launch and compile times and making the application lighter and more efficient to use. Significant improvements to documentation consist of more than 400 additional code examples and new chapters on component creation and customization.

"The Flash community has been very engaged in helping us deliver this update," said Mike Chambers, product manager for developer relations, Macromedia. "The result is a significant collection of changes that substantially improve the developer and designer experience."

## Macromedia Fulfills Promise of Web Content Management

(San Francisco) – Macromedia has announced the Macromedia Web Publishing System, which provides everything an organization needs to affordably build and manage Web and intranet sites. The Macromedia Web Publishing System enables tens, hundreds, or thousands of content contributors to publish Web sites for internal and external communications. The WPS consists of Macromedia Studio MX 2004, Macromedia Contribute 3, Macromedia FlashPaper 2, and new Contribute Publishing Services. [www.macromedia.com/go/wps/](http://www.macromedia.com/go/wps/).

## Macromedia Announces Contribute 3

(San Francisco) – Macromedia Contribute has received a major upgrade. Contribute redefines Web publishing by enabling nontechnical users to update pages on Web sites or intranets as easily as they would edit a Microsoft Word document. Macromedia Contribute 3 adds granular administrator control, flexible approval workflow, editing enhancements, and Dreamweaver MX 2004 integration to its award-winning ease-of-use and browse-edit-publish workflow. Contribute 3 is a key element of the new, enterprise-ready Macromedia Web Publishing System for organizational deployments to hundreds or thousands of business users. For more information, or to download a preview release of Macromedia Contribute 3, visit [www.macromedia.com/go/contribute3/](http://www.macromedia.com/go/contribute3/).



# HostMySite.com

Built for ColdFusion Pros by ColdFusion Pros

plans from

**\$8.95** / mo.

**FREE** Domain Name\*  
**FREE** Setup  
**FREE** 2 Months

- 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

Visit [www.HostMySite.com/cfdj](http://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)

# Caching ColdFusion Components in Shared Memory

Caching CFCs can pay off in improved performance and scalability

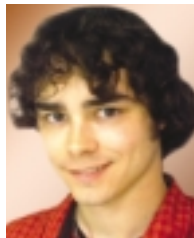
A big challenge when building Web applications is dealing with scalability and performance. Sometimes these issues are forgotten until after the application is being used by the public and under heavy load. Performance tuning usually happens as an afterthought when all the code is already completed.

While I was developing the latest version of Hot Banana, I faced the following challenge. Hot Banana is a ColdFusion Web content management system that uses ColdFusion Components (CFCs) extensively. It queries a database, processes the data, and serves up dynamic Web pages every time a user visits a Web site powered by Hot Banana. There is a lot of data to process in order to generate a single Web page. The dynamic navigation structure needs to be generated from data in the database. All the dynamic content needs to be pulled from the database and processed. Dynamic images and links need to be processed as well.

As we added more and more functionality and control to our Web pages I was faced with the challenge of speeding up the time it takes to generate a Web page. It was taking more than a full second to generate a page on our development server, which was completely unacceptable. I was forced to take a look at the different pieces that make up a Web page and find ways to cut corners.

First I looked at the entire tree structure of all the Web pages on the Web site, what we call the navigation structure. The navigation structure is used to build dynamic menus on the Web page, the site map, and more. Even though every page on the Web site uses the same navigation, every page was building it from scratch every time. We were already using a ColdFusion Component to manage the navigation structure. This CFC was being created by every Web page plus the administration area. This CFC provides a number of functions for interacting with the navigation. Most of the time was being spent searching through the navigation to return results.

To solve this, I decided to keep the navigation CFC stored in a shared scope. I could have stored only the data in shared memory, but by keeping the CFC itself in memory, the Web pages wouldn't need to create and initialize it every time. Then it would be able to use its internal indexes to search through the navigation and return information quickly. Sometimes you



By Jesse Skinner

might just want to keep data in shared memory with a CFC, but since our navigation CFC is the only CFC accessing and managing the data, it made sense to keep the data inside the CFC and keep the CFC in shared memory. I could have put the navigation CFC in any ColdFusion scope, and there are a number to choose from.

Sometimes, the SESSION scope would make sense for storing data and objects. For example, this would allow each visitor to the Web site to potentially have a different navigation structure.

This could have made sense if visitors to the Web site were able to customize the navigation of the Web site. Since all visitors would be seeing the same navigation, the SESSION scope wasn't the right decision.

Originally, I put the navigation CFC into the APPLICATION scope. Hot Banana can support more than one Web site on a single server. Each Web site is running under its own application, so there can be separate SESSION management across different Web sites. Since each Web site would have its own instance of the navigation CFC and its own application, I thought the APPLICATION scope would be a good choice.

But there was something I hadn't considered: our administration area is kept under a different application than the public Web site. When administrators logged in and added a Web page, I needed to refresh the navigation CFC being used by the Web site. Unfortunately, since this CFC was kept in the other APPLICATION scope of the Web site, there was no simple way to access it.

In the end, I decided to put everything related to Hot Banana into the SERVER scope. Since Hot Banana uses a number of different APPLICATION scopes at once, I wanted the convenience of being able to access any CFCs and data from any Web site. The SERVER scope is the most broadly shared scope available. It's shared across all applications that run on the Web server. By putting data into this scope, you allow all ColdFusion applications on the server to have access to these CFCs and data. This isn't always appropriate. If you can guarantee you have exclusive use of the server, you won't have any problems. Otherwise, you'll need to carefully think about the implication of having these CFCs available to the whole server.

I created a single ColdFusion struct variable within the SERVER scope, SERVER.HOTBANANA. I then stored all the cached CFCs and data in this struct. This prevents my SERVER variables from interfering with other ColdFusion applications that want to use the SERVER scope. Inside this structure, I have a struct for each Web site on the server. This lets each site have its own instance of the navigation CFC. Now, the administration area knows where to look to find a navigation CFC for a particular Web site.

You can see there's no limit to the ways you can organize your data, and each decision has its own pros and cons. It's important to make these decisions carefully. It helps to centralize the management of cached CFCs so that you can easily change your decision in the future.

Originally, to create an instance of the navigation CFC, I was using the CreateObject function, along with an initialization call:

```
objNavigation = CreateObject("component", "hotbanana.navigation");  
objNavigation.init();
```

Instead, I made a global custom GetObject function that would handle the job of caching and managing CFCs (see Listing 1). We use an initialization function, so the GetObject function calls this function for us so that it only happens once.

By passing "true" as the second parameter in this function, you can specify whether or not to use a cached version of the CFC. This function assumes that you will need only one instance of each CFC. You may not want to use this style of caching all the time. For example, you may want to have many different instances of a single CFC to manage different data. You should utilize different strategies and methods of caching CFCs to take advantage of the different ways your application uses CFCs.

You might feel inclined to start caching every CFC you can. This is great – until the Web server starts running out of memory. If the amount of data being cached is great, and there are many applications using cached data, this can easily become a problem. It's important to think about how often the data is being used, and to try to balance the trade-off between per-

formance and memory usage. If some data is rarely accessed, caching it won't have a significant impact on performance.

You may also want to design an algorithm that limits the number of CFCs cached in memory. For example, there may be a CFC for every Web page on a Web site. Your algorithm could cache only the top 10 requested Web pages. This would allow a Web site to get very large without having as much impact on the server's memory.

## Caching Different Types of Data

There's no limit to the types of data you can cache within a CFC. Data that takes time to generate or retrieve is a good candidate for caching. Queries, XML documents, structs, and even other objects are worth caching. You may already be using the CACHEDWITHIN attribute of CFQUERY to cache queries in memory. This method is easy to use but lacks control. You can clear all queries at once only by using CFOBJECTCACHE. There is also a limit to the number of queries you can cache.

By caching queries as instance variables within CFCs you can have a lot more control over when to refresh the data. You also have no limit to the number of queries you cache. Parsing, outputting, and manipulating ColdFusion XML documents can be heavy on the server. By keeping an XML document in memory, you have the option of manipulating the document directly without needing to parse in or write out an XML document every time. You can improve performance by accessing such documents directly, the same way you would access structs.

Using cached CFCs as instance variables within other cached CFCs is a very efficient way of architecting an applica-

## Web based collaboration made to order.

Collaboration

Flexible Security, 508 Compliance,  
Offline Capabilities, Reporting, Easy  
Integration and much more...

**1-866-477-7542**



2003 CFDJ Readers' Choice Award  
Winner for Best eBusiness Software  
& Best Web Application

Discussion Forums

[www.fusetalk.com](http://www.fusetalk.com)



Discussion forum solutions that make web-based collaboration risk-free and easy.



tion. With Hot Banana, there are dozens of CFCs that interact with each other. This could create a tree of CFCs, with many being created unnecessarily. By reusing cached CFCs we create more of an interacting web of CFCs. This actually reduces the amount of memory taken up by the application because no unnecessary CFCs will be created.

You can even use shared CFCs to cache data updates as well. We were keeping track of visitors by updating the database every time someone visited, but I found the hit to the database was becoming too much. To solve this, I put a shared CFC in memory for each Web site. This CFC was in charge of remembering log updates (in an array), then writing all the updates to the database once an hour.

If you're generating dynamic HTML output, or any string output, you can use CFSAVECONTENT to cache the output within the CFC. This could even be the only data that you need to cache in your application, especially when it takes the greatest amount of time to generate the string.

Caching the results of CFHTTP can definitely improve performance. Sometimes you may be required to cache these results. If your application is pulling in an RSS feed, often you will be allowed to request the feed only once an hour. If this is the case, you will certainly need to cache the results. You can just look at any function and decide if the result of that function can be cached. It can be easy to implement this style of caching. For example, let's say you have a function that returns a query (see Listing 2). This function needs to hit the database only once. The result will be available in memory as long as the CFC exists. You can apply this strategy to any function or block of code you want to cache.

## Refreshing the Cache

Caching data is great, just as long as the data doesn't change. In any application, though, data is going to be changing at different times, sometimes constantly. We need to con-

sider when it is appropriate to cache data, when we will want to refresh the cache, and how we want to manage refreshing cached data and CFCs.

One extreme strategy is to wipe the whole cache every time any of the data changes. This can be unnecessarily excessive. The benefit is that it's a very easy strategy to maintain. If all the cached data is in a single struct in memory, performing a single StructClear() can wipe the entire cache at once. It's the same strategy ColdFusion uses with cached queries, by letting us use only CFOBJECTCACHE to clear out all cached queries. This strategy can be appropriate if data doesn't change very often – and if a performance hit is acceptable when changes do occur.

At the other extreme, you may want to refresh only the exact piece of data that has changed. You can achieve this if you centralize all the actions on a piece of data within a single CFC, as you may very well be doing anyway. Then, when the data gets updated, you can clear out and regenerate it.

There are other strategies between these two extremes. You may want to clear out the cache periodically, say, once an hour. Or you may want to clear the cache based on certain events that happen within the system. For example, a cached Web page could be refreshed any time an administrator loads the administration area for that page.

During development you may want to disable caching of CFCs. Otherwise, as you're making changes, your code will still be accessing the old, cached versions. It's important to note that if you do this you will definitely need to test your code with caching turned back on. Bugs relating to caching issues are always the most confusing, frustrating, and difficult bugs to narrow down and fix – especially if users of the system aren't aware of the caching going on behind the scenes.

## Concurrency Issues

If only one person uses your application at a time, you like-

### Listing 1

```
function GetObject(txtCFC, blnCached) {
    var scope = SERVER.EXAMPLE;
    var myObject = 0;

    if (blnCached) {
        if (not StructKeyExists(scope, txtCFC)) {
            myObject = CreateObject("component", txtCFC);
            myObject.init();
        }
        scope[txtCFC] = myObject;
        return scope[txtCFC];
    } else {
        myObject = CreateObject("component", txtCFC);
        myObject.init();
        return myObject;
    }
}

objNavigation = GetObject("hotbanana.navigation", true);
```

### Listing 2

```
<cffunction name="getQuery">
    <cfif not StructKeyExists(this, "myQuery")>
        <cfquery name="this.myQuery" datasource="myDSN">
            SELECT *
            FROM table
        </cfquery>
    </cfif>
    <cfreturn this.myQuery />
</cffunction>
```

### Listing 3

```
<cfcomponent displayName="example">
```

```
<cffunction name="resetArray">
    <cfset this.array = ArrayNew(1)>
    <cfset this.array[1] = "First Element">
    <cfset this.array[2] = "Second Element">
    <cfset this.array[3] = "Third Element">
</cffunction>

<cffunction name="getItemThree">
    <cfset resetArray()>
    <!-- Hope nothing happens to this.array -->
    <cfreturn this.array[3]>
</cffunction>
</cfcomponent>
```

### Listing 4

```
<cfcomponent displayName="example">
    <cffunction name="resetArray">
        <cflock name="exampleLock" timeout="10" type="exclusive">
            <cfset this.array = ArrayNew(1)>
            <cfset this.array[1] = "First Element">
            <cfset this.array[2] = "Second Element">
            <cfset this.array[3] = "Third Element">
        </cflock>
    </cffunction>

    <cffunction name="getItemThree">
        <cfset resetArray()>

        <cflock name="exampleLock" timeout="10" type="readonly">
            <cfreturn this.array[3]>
        </cflock>
    </cffunction>
</cfcomponent>
```

Download the Code...

Go to [www.coldfusionjournal.com](http://www.coldfusionjournal.com)

ly won't run into any problems. Unfortunately for us, ColdFusion can use more than one thread at a time. This means that while one Web page is trying to update cached data, another Web page could be trying to retrieve this data at the same time. There are absolutely no guarantees about the order in which things will happen in this scenario.

Say you have a cached CFC that contains an array of data with three items in it. The CFC has two functions, `resetArray()` and `getItemThree()`. When the function `resetArray()` is called, it first sets `this.array = ArrayNew(1)`, then it fills the array with data. Later, in `getItemThree()`, there is code that first calls `resetArray()`, then returns the third element in `this.array` (see Listing 3).

What would happen if `getItemThree()` were called by two different users at the same time? The answer is, we don't know. It's possible that everything will work fine. However, it's also possible that immediately after `this.array = ArrayNew(1)` is called, and before `this.array[3]` is set, the other `getItemThree()` function will try to access `this.array[3]` and find that this array element does not exist.

This is a very difficult situation to test. You won't realize there are problems with your code until it's being used by a large number of people. Even when you do know there is a problem, it's difficult to be able to re-create the problem, let alone prove that the problem has been resolved. How do we prevent problems like this? Luckily, ColdFusion provides us with CFLOCK. What we can do is put CFLOCK tags throughout our code (see Listing 4).

This will ensure that only one user can call this function at a time. It will guarantee that when we access `this.array[3]`, it will still be initialized from the `resetArray()` function. When one user is trying to call `resetArray()` and the other is trying to return `this.array[3]`, one will have to wait for the other to finish.

We might be tempted to put CFLOCK tags around all the code in our cached CFCs. This might allow us to be certain things will be kept consistent, but it may have an impact on the performance of the server. Most of the time you won't have problems with two users accessing a function at the same time. It's only when the cached data is being manipulated or accessed that you will need to be concerned. You should try to put as little code as possible inside CFLOCK tags.


A nice way to achieve this is through the use of *getter* and *setter* functions. These are special types of functions whose only role is to set and return variables in a CFC. Anytime you want to access these variables, you call these functions. Then you can put CFLOCK tags inside these two functions. In the getter function, we can use the CFLOCK attribute `type="read-only"`. This way, more than one getter function can run simultaneously with no problems and no performance implications. Only the setter function will need `type="exclusive"`.

It's not just instance variables that we need to be concerned about. Local variables in our CFC's functions can be a problem too. If you don't declare your local variables at the top of the function with the `var` keyword, they actually become instance variables of the CFC. This means that when someone is in the middle of a function, all of a sudden all the variables being used by that function could suddenly change. The function could be in the middle of a loop, when all of a sudden it finds itself at the start of the loop. The only way to solve this is to be sure you declare all your variables at the top of every

function using the `var` keyword. This is generally a good idea anyway, as it ensures that your variables won't be overwriting variables from other functions.

## Conclusion

After putting these practices into play, putting the core structure of Hot Banana in shared memory, and caching as much data as possible, I was able to reduce the load time of a Web page substantially. The time was reduced from over 1,000 milliseconds to as low as 10 milliseconds for a single Web page. This yielded a 99% improvement in performance, resulting in a very stable and scalable system.

Caching CFCs in shared scopes is a good way to improve the performance and scalability of your ColdFusion applications. There are definitely a lot of things to watch out for, but if you do it carefully and diligently, you will be rewarded with a big performance payoff. 

## About the Author

Jesse Skinner currently works as the senior architect of Hot Banana ([www.hotbanana.com](http://www.hotbanana.com)). He's excited about the future of Web applications and is always striving to push the envelope. He is a computer science graduate from the University of Waterloo and is certified in Java and ColdFusion.

[jesse@hotbanana.com](mailto:jesse@hotbanana.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](http://halhelms.com).

# Creating a Remember Me Login

## Implementing a login script on your site

**M**any of my articles in this column have dealt with theoretical concepts and syntax of implementing those concepts in ColdFusion. In this article, I want to concentrate on the implementation steps you might take when building something.

Most Web applications have a “sign me up” feature that allows users to register. Registered users often have access to additional information or features that anonymous users don’t. I’m going to walk you through the process of creating a simple login form, including database authentication and a “remember me” checkbox.

### The Database

Before you start coding this application, you’ll need to create a database. Most login schemes include a username and password. You can put that information in a table called Users. The table will also need a unique identifier, called a primary key in database terms. Here is a sample of data from the table:

UserID	Username	Password
1	Jeff	Houser
2	CFDJ	Author
3	ColdFusion	Macromedia

Normally, a users table would have much more information than just a username and password. A name, address, and e-mail address are common additions. For the purposes of this example, we’ll keep it simple. In a real-world application, you should consider encrypting the passwords in your database for security purposes. You can do this using ColdFusion’s hash function. More information about the hash function is located at <http://livedocs.macromedia.com/coldfusion/6.1/html-docs/func113.htm#wp1105551>. You would use the hash function before saving the user’s password into the database. For the purposes of this example, the passwords will remain in plain text.

You could implement a login script using just the users table we defined, but you’ll find it limiting. The user is either logged in, or not logged in. There is no distinction between



By Jeffery Houser

different levels of access. Suppose, for example, that anonymous users can look at products, registered users can post reviews of products, and admin users can change product information. You’ll need something more than an “Authenticated” or “Not Authenticated” security structure. To do this we’ll need to be able to group users into security groups. The SecurityGroups table will need a primary key and a group name. This is an example of some possible groups:

GroupID	GroupName
1	Admin
2	Anonymous
3	Registered

You’ll also need an intersection or linking table to associate a user with his or her groups. This table will contain the GroupID and the UserID. The reason for creating this as a separate table is so that a single user can be in multiple groups, and a single group can have multiple users inside of it. This is known as a many-to-many relationship in the world of database design.

Here is an example of the SecurityUserGroups intersection table:

GroupID	UserID
1 (Admin)	1 (Jeff / Houser)
3 (Registered)	1 (Jeff / Houser)
3 (Registered)	3 (ColdFusion / Macromedia)

I added the group and user names next to the ID in parentheses to more easily show the relations. In a relational database, you would store only the ID. Our finished database structure is seen in Figure 1.

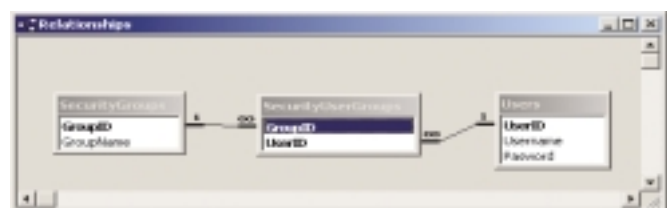


Figure 1: Database diagram



Before accessing this database from your ColdFusion code, you'll have to create a datasource in the ColdFusion administrator. Setting up the datasource is beyond the scope of this article, but it's a pretty straightforward task and you can read about setting up datasources for any database at

<http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/datasou4.htm#wp1277125>.

## The Login Form

With our database ready for use, we can start examining the login form. Most forms have two parts: an input page and a processing page. The input page asks for the username and password. It also has a checkbox for a "Remember Me" functionality. The code behind the form is shown in Listing 1.

The login form can be seen here:

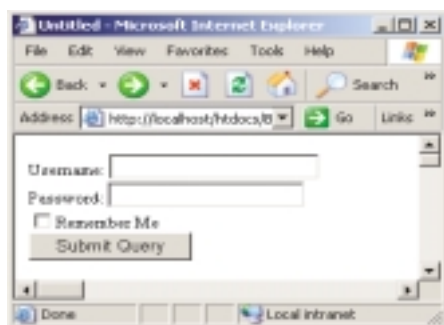


Figure 2: Login form

Enter Jeff in the username field and Houser in the password field. Check the Remember Me box and click submit. This brings us to the processing page (see Listing 2).

The RememberMe form variable value comes from a checkbox. If not checked when the form is submitted, the variable will not exist on the form processing page. You can use the cfparam tag to default if this situation occurs. The second step in the process is to validate the user's login information against the database. The cfquery tag is used to run the database query. The query joins the users table and the SecurityUserGroups table, where the username and password fields are equal to the input of the form. The query retrieves all information from the users table and the list of GroupIDs from the intersection table. If the database stored hashed passwords, we would change the query comparison to:

```
Users.password = '#hash(form.password)#'
```

This allows us to correctly compare two hashed values, not plain text passwords. This way we are keeping the user's password information secure.

We can check the RecordCount variable of our query to see if the query returned any rows. If the query did not return any rows, then the user did not enter a valid username and password combination. The login should fail. If rows are returned, then the login was a success. The code creates two session variables to process the login. To make use of session variables, you'll have to use the cfapplication tag. ColdFusion's application framework is beyond the scope of this article; however, you can read about it at

<http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-pa3.htm#wp1097308>.

The first value, LoggedIn, is a Boolean value that specifies that the user has logged in. I would default this value to false when the session is initialized. The second variable, *groups*, contains a list of all the groups that the user is in. The ValueList will give us a list from the column in the query.

Later in your application, when you have to decide whether a user should have access to a resource or not, you can use ListFind against the groups variable to see if the user is allowed. Here's an example:

```
<cfif ListFind(session.Groups, 1)>
    allow Access
<cfelse>
    No Access
</cfif>
```

If the user is in the admin group he or she can see the resulting HTML code, or access the corresponding resource. If not, then he or she will not be given access.

In this code, we are rolling our own security scheme. Many applications will use this approach, due to the complexity of security functions in the pre-CF MX days. However ColdFusion MX introduced a much improved security scheme using some new tags: cflogin, cfloginuser, and cflogout. They allow you to log in a user and set up a list of roles that ColdFusion will handle internally. The roles tie in with the role attributes of functions inside a ColdFusion component. These new tags are not in wide use yet, but they are definitely worth checking out if you are build-

# Once you're in it...



## ...reprint it!

- ColdFusion Developer's Journal
- Java Developer's Journal
- Web Services Journal
- Wireless Business & Technology
- MX Developer's Journal
- PowerBuilder Developer's Journal
- .NET Developer's Journal
- LinuxWorld Magazine
- WebSphere Journal
- WLDJ

Contact Kristin Kuhnle  
201 802-3026  
kristin@sys-con.com

REprints

SYSCON  
MEDIA

ing an application from the ground up (<http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/appsecu6.htm>). The reason I don't use them in my development is because my biggest project of the moment is being built to run off of BlueDragon, which does not yet support the functions.

## Remembering the User

The one portion of code that I haven't explained yet is the "remember me" portion, so let's look at it in detail. There are many different ways you could implement the "remember me" portion of code. In most methods, you'll set a cookie on the user's machine, and store the same value in the database.

When the user returns to the site, you can check to see if the cookie exists. If it does, you can retrieve the user's information from the database based on the cookie value. Simpler systems where security is not an issue may store the user's primary key ID. More complex systems with heavier security requirements may assign a UniqueID value, created with the CreateUUID function. Some systems I've worked with will store the CFID and CFTOKEN values generated by ColdFusion and used for session management, and use those to remember the user.


For the purposes of our sample, we are going to store the user's primary key as a cookie, but remember that in applications where security is a priority, this is probably not your best move. If form.RememberMe is set to true, then we use the cfcookie tag to create a cookie on the user's browser. We name the cookie UserID. The value is set to the UserID value returned from the query. It is set to never expire.

Setting the cookie is only the first step. When a user comes to the site, something will have to be implemented to check to see if we know who they are, or not. We are going to put this

code in the Application.cfm. The code in the Application.cfm will look like an abbreviated version of the code in the login-processing page (see Listing 3).

First the code checks whether the IsLoggedIn session variable is defined. If it isn't, then this is the first time a user has come to the site. Next, we check if the UserID cookie variable exists. If it does, Next we run a query to get the user data based on the UserID. If the query finds the user, the code sets the two session variables. If not, it defaults them to the value. If the cookie doesn't exist at all, it defaults the session values.

## Conclusion

This article demonstrated a simple method for implementing a login script on your site. It incorporated many common security elements and used many common ColdFusion tags. The approach I took in this article is not the only approach that could be used, but it is simple yet elegant. For those who want more, you can check out the authenticationAPI included in Macromedia's DRK 7. In my next column I'll talk more in depth about ColdFusion's application framework and the cfapplication tag, including setting up the session and application scopes. 

## 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 CF, most recently ColdFusion MX: The Complete Reference (McGraw-Hill Osborne Media).*

[jeff@instantcoldfusion.com](mailto:jeff@instantcoldfusion.com)

### Listing 1

```
<form action="loginip.cfm" method="post">
  Username: <input type="Text" name="Username"> <br>
  Password: <input type="Text" name="Password"> <br>
  <input type="Checkbox" name="RememberMe" value="True"> Remember Me<br>
  <input type="Submit">
</form>
```

### Listing 2

```
<!-- Get User's info based on db -->
<cfquery name="GetUser" datasource="CFDJCF101August">
  select users.*, SecurityUserGroups.GroupID
  from users, SecurityUserGroups
  where users.UserID = SecurityUserGroups.userID and
  users.username = '#form.username#' and
  users.password = '#form.password#'
</cfquery>

<!-- If the recordcount is 0, the login info is invalid --->
<cfif GetUser.recordcount GTE 1>
  <cfset session.loggedin = true>
  <cfset session.Groups = ValueList(GetUser.GroupID)>

  <!-- set the RememberMe cookie -->
  <cfif form.RememberMe is true>
    <cfcookie name="UserID" value="#GetUser.UserID#" expires="never" >
  </cfif>

<cfelse>
```

```
You didn't log in<br>
</cfif>
```

### Listing 3

```
<cfif not IsDefined("session.loggedin")>
  <cfif IsDefined("cookie.UserID")>
    <!-- Get User's info based on db -->
    <cfquery name="GetUser" datasource="CFDJCF101August">
      select users.*, SecurityUserGroups.GroupID
      from users, SecurityUserGroups
      where users.UserID = SecurityUserGroups.userID and
      users.userID = #Cookie.UserID#
    </cfquery>

    <!-- If the recordcount is 0, the login info is invalid --->
    <cfif GetUser.recordcount GTE 1>
      <cfset session.loggedin = true>
      <cfset session.Groups = ValueList(GetUser.GroupID)>
      <cfdump var="#session#">
    <cfelse>
      <cfset session.loggedin = false>
      <cfset session.Groups = "2">
    </cfif>
  <cfelse>
    <cfset session.loggedin = false>
    <cfset session.Groups = "2">
  </cfif>
</cfif>
```

Download the Code...

Go to [www.coldfusionjournal.com](http://www.coldfusionjournal.com)





**web services**  
conference & expo

**EDGE**

**Web Services Edge  
2005 East**

**International Web Services Conference & Expo**

**Announcing Web Services Edge 2005  
Extending Call for Papers to August 30th**  
Submit your proposal today!

**The Largest  
i-Technology  
Event of  
the Year!**



**Guaranteed  
Minimum  
Attendance  
3,000  
Delegates**



Tuesday, 2/15:  
Conference & Expo

Wednesday, 2/16:  
Conference & Expo

Thursday, 2/17:  
Conference & Expo

**Hynes Convention Center, Boston, MA  
February 15-17, 2005**

Join us in delivering the latest, freshest, and most proven Web Service solutions... at the *Fifth Annual Web Services Edge 2005 East-International Conference & Expo* as we bring together IT professionals, developers, policy makers, industry leaders and academics to share information and exchange ideas on technology trends and best practices in secure Web services and related topics including:

- Transitioning Successfully to SOA
- Federated Web services
- ebXML
- Orchestration
- Discovery
- The Business Case for SOA
- Interop & Standards
- Web Services Management
- Messaging Buses and SOA
- Enterprise Service Buses
- SOBAs (Service-Oriented Business Apps)
- Delivering ROI with SOA
- Java Web Services
- XML Web Services
- Security
- Professional Open Source
- Systems Integration
- Sarbanes-Oxley
- Grid Computing
- Business Process Management
- Web Services Choreography

### 3-Day Conference & Education Program features:

- Daily Keynotes from companies building successful and secure Web Services
- Daily Keynote Panels from each Technology Track
- Over 60 sessions and seminars to choose from.
- Daily Training Programs that will cover Web Service Security, J2EE, and .asp.NET
- FREE Full-Day Tutorials on .NET, J2EE, MX, and WebSphere Tutorials
- Opening Night Reception

### Interested in Exhibiting, Sponsoring or Partnering?

Becoming a Web Services Edge Exhibitor, Sponsor or Partner offers you a unique opportunity to best position your organization's message and visibility to a targeted audience of Web Services Professionals. Make your plans now to reach the most qualified software developers, engineers, system architects, analysts, consultants, group leaders, and C-level management responsible for Web Services, initiatives, deployment, development and management at the regions best known IT business address, The Hynes Convention Center in Boston.

For exhibit and sponsorship information please contact Jim Hanchrow at 201.802.3066, or email at [jimh@sys-con.com](mailto:jimh@sys-con.com).

Sponsored by:



WebServices JOURNAL



SDTimes



asp.netPRO

NET JOURNAL



WebSphere JOURNAL



All brand, and product names mentioned above are trade names, service marks or trademarks of their respective companies.

Contact for Conference Information: Jim Hanchrow, 201-802-3066, [jimh@sys-con.com](mailto:jimh@sys-con.com)



**[www.sys-con.com/edge](http://www.sys-con.com/edge)**



# Fusebox or Mach-II?

A look at the strengths and weaknesses of both frameworks

For more than seven years, Fusebox ([www.fusebox.org](http://www.fusebox.org)), now in its fourth version, has been the dominant framework for building ColdFusion applications. During that time, Fusebox has evolved from a set of best practices into a mature framework capable of tackling very large jobs while remaining easy enough to use for everyday small tasks.

As one of the contributors to Fusebox, I've been very gratified by the tremendous response from developers who regularly e-mail me, relating stories of their success using Fusebox. Here's a sample e-mail I received recently: "I'm writing to thank you for your efforts on Fusebox. I went to one of your classes in DC three years ago, and Fusebox, along with FLiP, has let me turn out one success after another. My boss started out skeptical, but he's seen the results and now Fusebox is a requirement for all new hires. Fusebox rocks!"

Of course, there are a few who e-mail me to tell me how much they hate Fusebox. Here's one such e-mail: "Fusebox, what exactly is it? A methodology? A framework? I think it's a cult, turning out little Fusebox disciples that spread the stupid Fusebox message. I've gone on too many consultant gigs where they required Fusebox, and I spend half my time explaining how lame Fusebox is." I was tempted to respond to my correspondent that perhaps he would have had better success if he had spent all of his time solving their problem instead of engaging in religious wars, but I simply thanked him for his post.

Over the last year, I've worked with Ben Edwards on a new framework, called Mach-II ([www.mach-ii.com](http://www.mach-ii.com)). Since I'm associated with both Fusebox and Mach-II, I receive e-mails asking me which one I really like. Well, I like both of them – for different reasons. In this article, let's take a look at what Web frameworks are meant to do and examine the similarities and differences between Fusebox and Mach-II.

What is a framework? The definition I find the most helpful comes from the Earth System Modeling Framework, a collaboration of universities and corporations concerned with earth science: "We use the term framework to refer to a structured



By Hal Helms

collection of software building blocks that can be used and customized to develop components, assemble them into an application, and run the application."

A framework provides prebuilt and pretested code that can be used as a skeleton on which to build applications. Web frameworks may be lightweight or heavyweight. Heavyweight frameworks typically apply themselves to a particular domain where they offer greater help in building domain-specific applications and require a greater degree of conformity as to the code being written. Lightweight frameworks are usually more generic (meant to deal with many different domains); they offer basic "plumbing" assistance and provide greater latitude in the way that developers write code.

Both Fusebox and Mach-II are lightweight frameworks. They offer help connecting the presentation tier of an application with the business logic and data persistence tiers.

Why bother using a framework? Here are a number of reasons:

- **Improved reliability:** Working with the prebuilt components of both Fusebox and Mach-II means working with pretested components. This reduces the bugs to which "one-off" code is prone.
- **Developer productivity:** Having prebuilt code assets means that a substantial percentage of the code you'll need to make your application run has already been written. And though all frameworks have a learning curve, the price to learn the framework is paid once while the benefits in terms of faster delivery of software continue to pay off.
- **Better, more granular security:** A robust security model is not a trivial thing to implement. It just makes sense to have base security capabilities built into a framework rather than having to develop one for each new application.
- **Easier team development:** It may be, as the old song says, that you say toe-MAY-toe and I say toe-MAH-toe, but by agreeing on a common framework, we can work on different aspects of an application without fears that our two efforts will be incompatible.
- **Faster delivery:** Given coders with the same skills, a development team that starts off with 30% of its code written for it will have a substantial advantage when working to a tight deadline.
- **Reusable code assets:** Code written for one specific application within a given framework context can often be reused for another application.

- **Less expensive development:** All of the above benefits boil down to this: it costs less to build the same application with a framework than without it. In the competitive environments in which most of us work, the cost differential can be a very important factor.

Every framework, like every application, has design tradeoffs. We start off on one path – taking this road rather than that other one – and those choices have consequences. Many of the religious wars fought in the tech sector are brought about by one partisan concentrating on the strengths of his/her particular choice, ignoring the tradeoffs involved in that choice. And, of course, the disputant on the other side does the same. These “great debates” are usually framed in the most simplistic possible way: Which is better, A or B? But, without specifying what they mean by “better,” both sides argue at cross purposes.

The framework we choose should be determined by our own circumstances and needs. Fusebox and Mach-II fit the same architectural role, but approach their jobs from different perspectives and offer different benefits. Both Fusebox and Mach-II rely on an XML configuration file whose chief job is to determine what actions shall be taken in

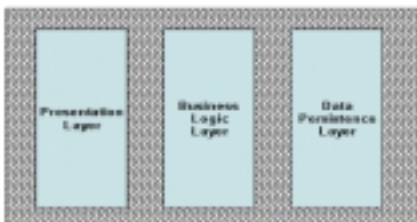


Figure 1: Web framework

response to a specific request. Both frameworks allow developers to extend the basic framework by means of “plugin points” – specific times in the execution of a request where developer-provided code is run. Both frameworks allow for separation of presentation code from business logic and data persistence.

Those are their similarities. Their differences lie chiefly in the way that both frameworks solve the same problem. Mach-II takes an unabashedly object oriented approach to application development. Developers make use of the framework by using and extending prewritten classes.

Mach-II was conceived as an implementation of the Model-View-Controller (MVC) design pattern, adapted for the Web. This imposes certain restrictions on developers. Done correctly, the model part of MVC is separate from any single application. When creating the model layer, developers make a scale model of the domain in which they are interested. Model components have no presentation aspect. Instead they serve to capture business logic in the context of objects.

Let's take an example. Here is a CFC that models an Employee for the ABC company.

```
<cfcomponent displayname="Employee">
  <cfset variables.firstName = "" />
  <cfset variables.lastName = "" />
  <cfset variables.dateOfHire = "" />
  <cfset variables.payStrategy = "" />

  <!--- init function, a pseudo-constructor,
        intentionally omitted from code printout --->

  <!--- getters/setters for instance variables
        intentionally omitted from code printout --->

  <cffunction name="getYearsInService"
    access="public" returntype="numeric"
    output="false">
    <cfreturn DateDiff('yyyy', now(),
      getDateOfHire()) />
  </cffunction>

  <cffunction name="isVested" access="public"
    returntype="boolean" output="false">
    <cfif getYearsInService() GT 5>
      <cfreturn true />
    </cfif>
    <cfreturn false />
  </cffunction>

  <cffunction name="getPayDue" access="public"
    returntype="numeric" output="false">
    <cfreturn getPayStrategy().getPayDue(this)
  />
  </cffunction>
</cfcomponent>
```

Notice that there is no presentation code. There is also no data persistence code: no queries to get information from—or write information to—a database. The reason is that domain models confine themselves to modeling the business itself. They don't concern themselves with how users might view the business or with what interface will be used to interact with the model; nor are they

concerned with their own persistence.

For many ColdFusion programmers, that just seems wrong. How can you get information without a database? What good is a model if users can't interact with it? From the object oriented (OO) viewpoint, though, what we have is a good example of highly cohesive encapsulation. An Employee object should only be concerned with its own, narrow sphere of interest. It's not that OO programmers aren't concerned with issues of presentation and data persistence. There will be other objects, specialized for those jobs, to handle these aspects.

This “object think” gives rise to many highly specialized components that are interdependent on other components to accomplish their work. At runtime, objects send messages back and forth to other objects in response to a particular request. In fact, a running OO application greatly resembles a complex conversation among many participants. The “smarts” of the model lie not in any one component, but in the interaction of all of the components. For an analogy, you might think of an anthill.

As someone who teaches OO to ColdFusion programmers, I can attest to how utterly foreign this type of development initially seems to programmers reared on the procedural model. But talk to these same procedural programmers by the end of the class, and an overwhelming majority, having become accustomed to this strange, new way of thinking, see the greater maintainability and reusability of individualized components. A well-constructed model can work with many different applications and we get real code reuse, not simply “copy and paste” repurposing of code.

Mach-II is ideal for this type of development. It provides a way to integrate individual model, data persistence, and view components into an application, while keeping the separate parts separate. If you are very comfortable working with object orientation, you'll likely find Mach-II a natural fit.

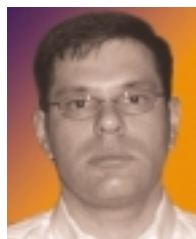
Fusebox is a procedural framework that can work with object models (just as Mach-II does) but can also be used by procedural programmers writing procedural code. Fusebox 4 also encourages the use of the MVC design pattern (adapted for the Web), but does not require developers to adopt MVC.

—continued on page 39

# Web Server Load-Balancing Options

## Making the transition to hardware-based

There are two basic Web server load balancing options: hardware-based and software-based. The latter have been slowly disappearing from the enterprise while the former have been gaining a larger presence.



By Frank S. DeRienzo

Many software-based options have reached the end of their product life cycle. Macromedia ClusterCATS is among those end-of-life products; it has not been ported to IIS6. The primary software-based Web server load-balancing (actually failover rather than load-balancing) option with IIS6 is MS NLB.

ColdFusion Web sites on IIS5 running multiple servers clustered with ClusterCATS will soon need to transition to either MS NLB or a hardware-based solution. While clustering CF on JRun is always an option for the application servers, this does not provide a means to cluster Web servers. Since session traffic must pass through a Web server, something more robust than DNS round-robin and more application-oriented than MS NLB is desirable in the enterprise. ClusterCATS has an acronym suffix that stands for Content, Application & Transaction Smart. It is possible to replace ClusterCATS with a hardware-based solution that offers similar features, but with much greater load capacity and robustness.

Hardware-based options have been steadily improving over the past eight years. Most high-end hardware load balancing devices (HLDs) are now application-aware; they not only monitor the health of a Web server, but also the health of an application server. If a Web server is running, but an application server is stalled, most HLDs will redirect traffic away from the stalled server even though the Web server is alive. Only five years ago, in order to get a LocalDirector to react to monitor the health of a CF application, you had to use the ClusterCATS dynamic feedback protocol (dfp) agent to communicate the state of the application to the LocalDirector. Virtually all newer devices can monitor a content string produced by an application server and react to a problem faster than the dfp agent could register the status of an application.

Many HLDs have algorithms that can tell when there is a degradation in the performance of a server. Sometimes, a Web or application server in a pool may be handling sessions in a

decreased capacity; most high-end HLDs are quite adept at ensuring that the best performing server is the one to which the inbound traffic is directed.

Some HLDs are also very adaptable to unique infrastructure requirements. They can operate at many layers of the OSI model. Figure 1 shows an HLD sitting on a network without operating as a bridge or a router; it is connected to a flat network by a single network interface card and consequently is about as easy to integrate as a PC. In this example, an F5 BIG-IP is managing session traffic both in front of the Web servers and between distributed Web servers and application servers.

Competition in the HLD space is very aggressive; manufacturers are offering trade-in options and competitive upgrades. There is even a large used hardware market where many a frugal network engineer and Webmaster have found a high-end HLD at a low-end price. Be careful though – caveat emptor – there's a lot of junk out there on the used market as well. And if you are running an enterprise operation, there is no replacement for the professional support team and the software

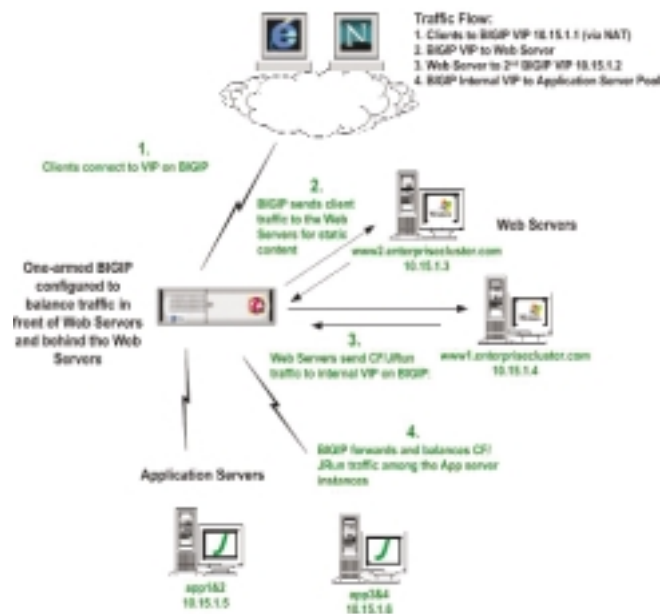


Figure 1: ColdFusion form for creating new Java pets




upgrades that come with a new purchase.

Here are some tips to help you make the transition on your multi-server CF site from a software-based Web server load balancing solution to a hardware-based option.

1. Carefully choose an HLD with the feature set that best matches your needs; some HLDs work better as routers than they do as bridges, for example. At least one high-end option can fit into your network with as little complication as adding a server. Look at your network infrastructure and decide what will fit best.
2. Decide whether you need SSL termination at the HLD. Some HLDs can accelerate SSL traffic from a single virtual IP address (VIP) and direct traffic back to multiple servers in a pool, while some require that SSL traffic be terminated at the Web server. The former option requires only a single certificate for multiple servers, while the latter requires a certificate for each Web server.
3. Keep the initial configuration simple. If you wish to integrate JRun connector-based application-level load balancing or session replication, you may wish to begin with a simple one-to-one Web server to CF server correlation and get that operational behind your new HLD before moving on to more complex configurations.
4. If CF is distributed onto separate platforms from your Web servers, you may wish to use your new HLD to balance traffic in front of the Web server and also between the Web servers and the CF servers. See the clustering section of the Macromedia CF DevCenter for details on this option.

## "Hardware ...is quickly becoming the **only viable option in the enterprise**"

Hardware has always been a more robust Web server load balancing option; it is quickly becoming the only viable option in the enterprise. Mission-critical, multi-server CF sites with heavy traffic should include hardware-based load balancing. 

### About the Author

Frank S. DeRienzo is part of the Macromedia MX Professional Services team. At Macromedia, he has focused on high availability and scalability through Web site clustering and Web server integration with various hardware load balancing and content management platforms. Prior to joining BrightTiger/Allaire/Macromedia in June 1997, he had a distinguished military career with the U.S. Army Rangers and Special Forces.

[fderienzo@macromedia.com](mailto:fderienzo@macromedia.com)


## Fusebox or Mach-II? —continued from page 37

So, which is better? I'll give you a breakdown of what I consider to be the strengths and weaknesses of both frameworks, but first, I'd like to encourage you to find out more about both frameworks by attending the annual Fusebox conference on September 18–19 (Saturday–Sunday) in Rockville, Maryland (DC area).

This year, we'll be releasing version 4.1 of Fusebox at the conference. This newest version includes native XML support for object invocation and a new assertion mechanism, as well as other improvements. There will be talks and hands-on workshops. Whether you're an old Fusebox pro, or someone who wants to evaluate Fusebox, I highly recommend coming to the conference. For more info and registration, go to [www.cfconf.org/fusebox2004](http://www.cfconf.org/fusebox2004).

	Strengths	Weaknesses
Fusebox	<ul style="list-style-type: none"><li>• excellent for team development</li><li>• mature</li><li>• has supporting methodology (FLIP)</li><li>• relatively easy to learn</li><li>• good documentation with Fusedocs</li><li>• strong community support</li><li>• excellent performance due to parsing cycle</li></ul>	<ul style="list-style-type: none"><li>• encapsulation model not as strong</li><li>• code reuse through copy/paste</li></ul>
Mach-II	<ul style="list-style-type: none"><li>• excellent encapsulation model</li><li>• reuse of code rather than repurposing of code</li><li>• better support for framework "plug-ins"</li><li>• helps developers fully embrace the OO model</li><li>• can be used with CFCs or with Java</li></ul>	<ul style="list-style-type: none"><li>• works better with small teams of experienced OO programmers</li><li>• harder to learn</li><li>• no accompanying methodology</li></ul>

In short, there is no single answer to the question, "Which is better?" It depends on your background, on the makeup of your development environment, and on what you're trying to accomplish. With both Mach-II and Fusebox, we have a well-thought-out and implemented framework for helping us write applications better and faster. By any standards, the choice of either framework is a winning one.

See you at the Fusebox conference! 

### About the Author

Hal Helms ([www.halhelms.com](http://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](mailto:hal.helms@teamallaire.com)



# ColdFusion

For more information go to...

## U.S.

**Alabama**  
Huntsville  
Huntsville, AL CFUG  
[www.nacflug.com](http://www.nacflug.com)

**Alaska**  
Anchorage  
Alaska Macromedia User Group  
[www.akmmug.org](http://www.akmmug.org)

**Arizona**  
Phoenix  
[www.azcfug.org](http://www.azcfug.org)

**Arizona**  
Tucson  
[www.tucsoncfug.org](http://www.tucsoncfug.org)

**California**  
San Francisco  
Bay Area CFUG  
[www.bacflug.net](http://www.bacflug.net)

**California**  
Riverside  
Inland Empire CFUG  
[www.sccflug.org](http://www.sccflug.org)

**California**  
EL Segundo  
Los Angeles CFUG  
[www.sccflug.org](http://www.sccflug.org)

**California**  
Irvine  
Orange County CFUG  
[www.sccflug.org](http://www.sccflug.org)

**California**  
Davis  
Sacramento, CA CFUG  
[www.saccflug.org](http://www.saccflug.org)

**California**  
San Jose (temporary)  
Silicon Valley CFUG  
[www.siliconvalleycfug.com](http://www.siliconvalleycfug.com)

**California**  
San Diego  
San Diego, CA CFUG  
[www.sdcflug.org/](http://www.sdcflug.org/)

**California**  
Long Beach  
Southern California CFUG  
[www.sccflug.org](http://www.sccflug.org)

**Colorado**  
Denver  
Denver CFUG  
[www.denvercfug.org/](http://www.denvercfug.org/)

**Delaware**  
Kennett Square  
Wilmington CFUG  
[www.bvcfug.org/](http://www.bvcfug.org/)

**Delaware**  
Laurel  
Delmarva CFUG  
[www.delmarva-cfug.org](http://www.delmarva-cfug.org)

**Florida**  
Jacksonville  
Jacksonville, FL CFUG  
[www.jaxcfusion.org/](http://www.jaxcfusion.org/)

**Florida**  
Winter Springs  
Gainesville, FL CFUG  
[www.gisfusion.com/](http://www.gisfusion.com/)

**Florida**  
Plantation  
South Florida CFUG  
[www.cfug-sfl.org](http://www.cfug-sfl.org)

**Florida**  
Tallahassee  
Tallahassee, FL CFUG  
[www.tcfug.com/](http://www.tcfug.com/)

**Florida**  
Palm Harbor  
Tampa, FL CFUG  
[www.tbmmug.org](http://www.tbmmug.org)

**Georgia**  
Atlanta  
Atlanta, GA CFUG  
[www.acfug.org](http://www.acfug.org)

**Illinois**  
East Central  
East Central Illinois CFUG  
[www.ecicfug.org/](http://www.ecicfug.org/)

**Indiana**  
Avon  
Indianapolis, IN CFUG  
[www.hoosierfusion.com](http://www.hoosierfusion.com)

**Indiana**  
Mishawaka  
Northern Indiana CFUG  
[www.ninmug.org](http://www.ninmug.org)

**Iowa**  
Johnston  
Des Moines, IA CFUG  
[www.hungrycow.com/cfug/](http://www.hungrycow.com/cfug/)

**Kentucky**  
Louisville  
Louisville, KY CFUG  
[www.kymug.com/](http://www.kymug.com/)

**Louisiana**  
Lafayette  
Lafayette, LA MMUG  
[www.cflib.org/acadiana/](http://www.cflib.org/acadiana/)

**Maryland**  
Lexington Park  
California, MD CFUG  
<http://www.smdcfug.org>

**Maryland**  
Rockville  
Maryland CFUG  
[www.cfug-md.org](http://www.cfug-md.org)

**Massachusetts**  
Quincy  
Boston, MA CFUG  
[www.bostoncfug.com](http://www.bostoncfug.com)

**Michigan**  
East Lansing  
Mid Michigan CFUG  
[www.coldfusion.org/pages/index.cfm](http://www.coldfusion.org/pages/index.cfm)

**Minnesota**  
Brooklyn Park  
Twin Cities CFUG  
[www.colderfusion.com](http://www.colderfusion.com)

**Missouri**  
Overland Park  
Kansas City, MO CFUG  
[www.kcfusion.org](http://www.kcfusion.org)

**Missouri**  
O'Fallon  
St. Louis, MO CFUG  
[www.stlmug.com/](http://www.stlmug.com/)

**New Jersey**  
Princeton  
Central New Jersey CFUG  
<http://www.cjcfug.us/>

**Nevada**  
Las Vegas  
Las Vegas CFUG  
[www.snfcug.com/](http://www.snfcug.com/)

**New York**  
Albany  
Albany, NY CFUG  
[www.anycfug.org](http://www.anycfug.org)

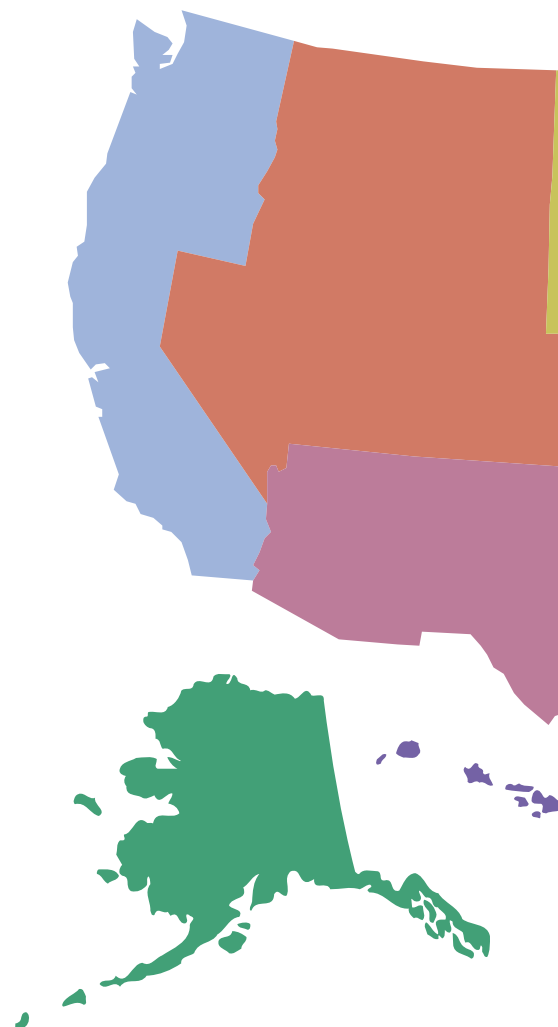
**New York**  
Brooklyn  
New York, NY CFUG  
[www.nycfug.org](http://www.nycfug.org)

**New York**  
Syracuse  
Syracuse, NY CFUG  
[www.cfugcny.org](http://www.cfugcny.org)

**North Carolina**  
Raleigh  
Raleigh, NC CFUG  
[www.ccfug.org](http://www.ccfug.org)

**Ohio**  
Dayton  
Greater Dayton CFUG  
[www.cfd Dayton.com](http://www.cfd Dayton.com)

**Oregon**  
Portland  
Portland, OR CFUG  
[www.pdxcfug.org](http://www.pdxcfug.org)



# User Groups

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



**Pennsylvania**  
Carlisle  
Central Penn CFUG  
[www.centralpenncfug.org](http://www.centralpenncfug.org)

**Pennsylvania**  
Exton  
Philadelphia, PA CFUG  
[www.phillycfug.org/](http://www.phillycfug.org/)

**Pennsylvania**  
State College  
State College, PA CFUG  
[www.mmug-sc.org/](http://www.mmug-sc.org/)

**Rhode Island**  
Providence  
Providence, RI CFUG  
[www.ricfug.com/www/meetings.cfm](http://www.ricfug.com/www/meetings.cfm)

**Tennessee**  
LaVergne  
Nashville, TN CFUG  
[www.ncfug.com](http://www.ncfug.com)

**Tennessee**  
Germantown  
Memphis, TN CFUG  
[www.mmug.mind-over-data.com](http://www.mmug.mind-over-data.com)

**Texas**  
Austin  
Austin, TX CFUG  
[www.ctexas.net/](http://www.ctexas.net/)

**Texas**  
Corinth  
Dallas, TX CFUG  
[www.dfwcfug.org/](http://www.dfwcfug.org/)

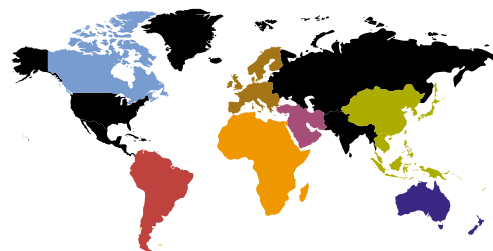
**Texas**  
Houston  
Houston Area CFUG  
[www.houcfug.org](http://www.houcfug.org)

**Utah**  
North Salt Lake  
Salt Lake City, UT CFUG  
[www.slcfug.org](http://www.slcfug.org)

## About CFUGs

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

## INTERNATIONAL



**Australia**  
ACT CFUG  
[www.actcfug.com](http://www.actcfug.com)

**Australia**  
Queensland CFUG  
[www.qld.cfug.org.au/](http://www.qld.cfug.org.au/)

**Australia**  
Southern Australia CFUG  
[www.cfug.org.au/](http://www.cfug.org.au/)

**Australia**  
Victoria CFUG  
[www.cfcentral.com.au](http://www.cfcentral.com.au)

**Australia**  
Western Australia CFUG  
[www.cfugwa.com/](http://www.cfugwa.com/)

**Brazil**  
Brasilia CFUG  
[www.cfugdf.com.br](http://www.cfugdf.com.br)

**Brazil**  
Rio de Janeiro CFUG  
[www.cfugrio.com.br/](http://www.cfugrio.com.br/)

**Brazil**  
Sao Paulo CFUG  
[www.cfugsp.com.br](http://www.cfugsp.com.br)

**Canada**  
Kingston, ON CFUG  
[www.kcfug.org](http://www.kcfug.org)

**Canada**  
Toronto, ON CFUG  
[www.cfugtoronto.org](http://www.cfugtoronto.org)

**Ireland**  
Dublin, Ireland CFUG  
[www.mmug-dublin.com/](http://www.mmug-dublin.com/)

**Italy**  
Italy CFUG  
[www.cfmentor.com](http://www.cfmentor.com)

**Japan**  
Japan CFUG  
[cfusion.itfrontier.co.jp/jcfug/jcfug.cfm](http://cfusion.itfrontier.co.jp/jcfug/jcfug.cfm)

**Scotland**  
Scottish CFUG  
[www.scottishcfug.com](http://www.scottishcfug.com)

**South Africa**  
Joe-Burg, South Africa CFUG  
[www.mmug.co.za](http://www.mmug.co.za)

**South Africa**  
Cape Town, South Africa CFUG  
[www.mmug.co.za](http://www.mmug.co.za)

**Spain**  
Spanish CFUG  
[www.cfugspain.org](http://www.cfugspain.org)

**Switzerland**  
Swiss CFUG  
[www.swisscfug.org](http://www.swisscfug.org)

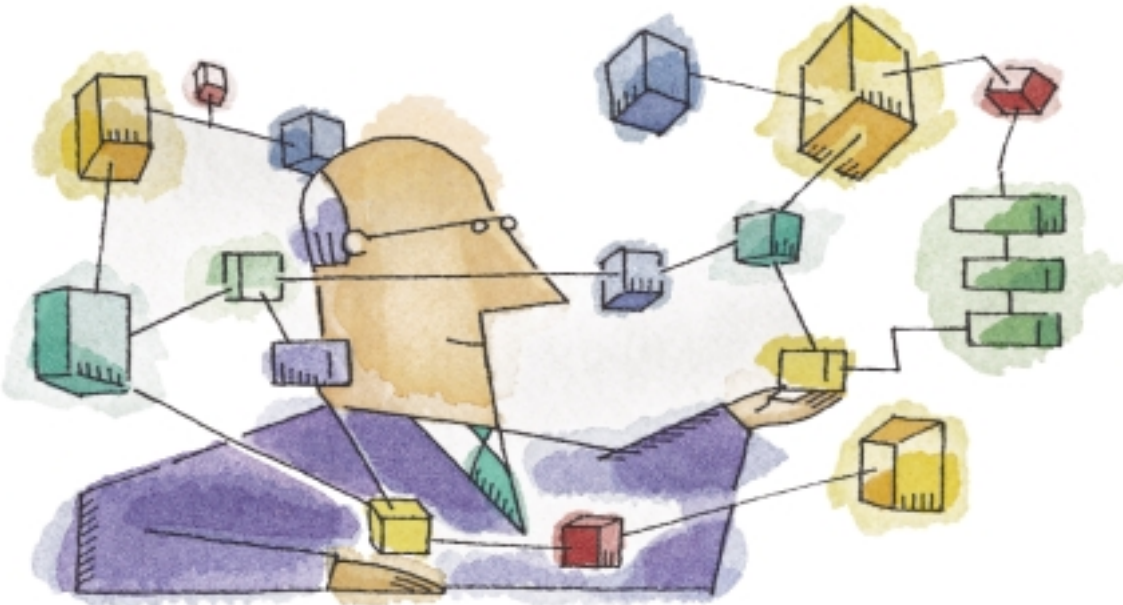
**Thailand**  
Bangkok, Thailand CFUG  
[thaicfug.tei.or.th/](http://thaicfug.tei.or.th/)

**Turkey**  
Turkey CFUG  
[www.cftr.net](http://www.cftr.net)

**United Kingdom**  
UK CFUG  
[www.ukcfug.org](http://www.ukcfug.org)







## Building a Keyword Vector Space Engine in ColdFusion

Adding an extra dimension to your keyword searches

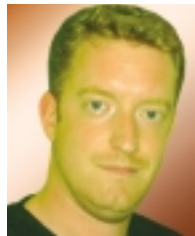
**D**o you want to extend your searching options beyond the basics? Would automated “Related Items” functionality have clients beating a path to your door? Here’s one way to go about it.

### The Problem

Keyword search is one of the most frequently requested features of modern Web development and we all have our own preferred solutions. Approaches based on Verity, Lucene, full-text indexes, or plain old SQL are all popular, but, while they provide basic search functionality, their limitations are well documented. What if you want a bit more than just matching occurrences of a couple of keywords? What if you want to relate entire content items on your site to each other, or to items on other Web sites? Just how do you power that “Related Items...” section that your client has seen on Amazon and just has to have?

There are two fundamental approaches to tackling this:

1. Human intervention: You could build a system that allows site editors to manually link different content items together or to set up a classification scheme to link items; or you



By Matt Perdeaux

could take the Amazon approach, recording user interactions and using collaborative filtering techniques.

2. A more sophisticated approach to searching: Techniques such as latent semantic indexing (LSI) and Bayesian analysis are effective but involve costly technologies and time-consuming integration.

Let’s assume we don’t want to increase the editorial overhead on our client, that our site will not have the level of user interaction required for collaborative filtering, and that we certainly don’t have the budget to buy Autonomy. This article will take a few introductory steps into the LSI jungle and explain the basics of building and using a keyword vector space.

### The Theory

The *vector space* is the underpinning of LSI; it is the mathematical representation of all the content items in your repository, based upon the concept of the *term space*. The term space is built by taking each individual keyword in the content repository and assigning it a geometrical axis. Individual items can be plotted as vectors in this high-dimensional term space, based on the keywords they contain and their frequency.

The vectors of items that share multiple keywords will be

close together in the vector space, whereas items that share few keywords will be farther apart. Once this mathematical model is set up, calculating their degree of similarity is simply a matter of finding the angle between vectors.

Confused? Let's look at a basic example. We have two articles on our Web site. The first article contains the words "web" twice and "blog" once. Of course, proper Web site content will contain much more than two keywords, but it is easier for now to visualize the idea with just two words.

Our term space will therefore consist of two orthogonal axes – one representing the keyword "web" and the other "blog". We can plot a vector for the article onto this term space – two units along the "web" axis and one along the "blog" axis (see Figure 1).

Now that the vector space has been set up, additional articles can be plotted as vectors in the same space. The second article contains the word "blog" twice and "web" once, so its vector will appear alongside the first vector (see Figure 2).

The angle between these vectors is the measure of the similarity of the two articles. The angle between two vectors A and B can be found using the cosine measure – a rearrangement of the scalar (or dot) product equation:

$$\cos \theta = \frac{A \cdot B}{|A| \cdot |B|}$$

which for our example vector space looks like:

$$A = 1i + 2j,$$

$$B = 2i + 1j$$

$$\cos \theta = \frac{(1i + 2j) \cdot (2i + 1j)}{\sqrt{1^2 + 2^2} \cdot \sqrt{2^2 + 1^2}} = \frac{(1)(2) + (2)(1)}{\sqrt{5} \cdot \sqrt{5}} = \frac{4}{5} = 0.8$$

If two vectors are identical, the angle between them will be 0°, the cosine of which is 1. Two perpendicular vectors (no shared keywords) will have an angle of 90° and a cosine of 0. So the above result multiplied by 100 will return the match measurement; the two articles show 80% similarity based on the vector space analysis.

That's the theory behind the vector space approach. The following sections will walk through a basic example of its implementation.

## The Solution

The example consists of two templates: default.cfm (see Listing 1) sets up some sample content and calls methods in vector.cfc (see Listing 2) which build the term space, build the item vectors, and calculate the cosine measures. The template displays one full content item and lists its matches by relevance.

### Step 1: Set Up Sample Data

We start by setting up some sample content items to fire at the vector space engine. The sample content in

# Subscribe Today!

— INCLUDES —  
**FREE**  
DIGITAL EDITION!  
(WITH PAID SUBSCRIPTION)  
GET YOUR ACCESS CODE  
INSTANTLY!



*The major infosecurity issues of the day... identity theft, cyber-terrorism, encryption, perimeter defense, and more come to the forefront in ISSJ the storage and security magazine targeted at IT professionals, managers, and decision makers*

# SAVE 50% OFF!

(REGULAR NEWSSTAND PRICE)

**Only \$39<sup>99</sup>** ONE YEAR 12 ISSUES

**www.ISSJournal.com**  
**or 1-888-303-5282**

**SYS-CON MEDIA**

*The World's Leading i-Technology Publisher*

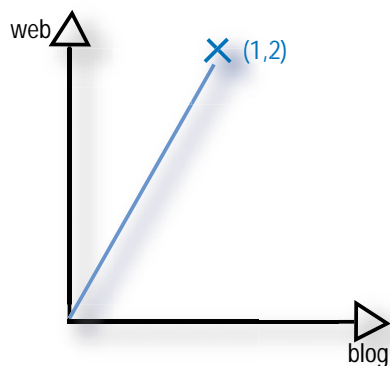


Figure 1: The vector representing the first article

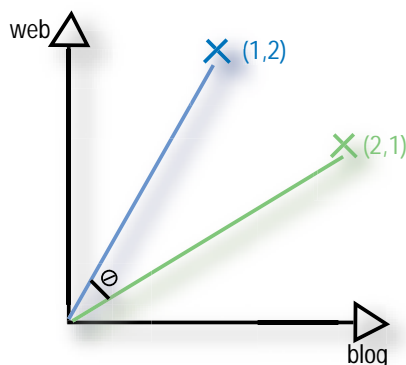


Figure 2: The angle formed between the vectors representing the two articles is the measure of their similarity.

Listing 1 is stored as an array of structures and consists of five short items about two distinct subjects, pasted from Google News (UK).

## Step 2: Set Up the Term Space

The term space is effectively a list of the unique keywords in content. Each keyword will be represented by an axis in the term space. The content array (arItems) is sent to the prepareTermSpace public method in vector.cfc (see Listing 2). This method loops through the array and concatenates all the titles and bodies of the items into one long string. This aggregate string is then sent to the getUniqueKeywords private method (see Listing 2). This method's job is to return an array containing all keywords, but it goes through a number of rationalizing steps before producing the final list.

1. First, all HTML and punctuation is removed by calling another private method, removePunctuation (see Listing 2), which runs through a series of regular expressions to strip the string down to raw text.
2. The next step is to get rid of all the "stop" words in the string. Stop words are common articles, pronouns, and so forth that do not convey meaning on their own and are of little use in the search engine context. The removeStopWords private

method (see Listing 2) loops through an example stop-word list and removes any matches. The compilation of stop-word lists is an art; depending on your content, there may be additional words you can add to the list. For instance, if your site is about ColdFusion, you can probably add the words "ColdFusion," "cfm," and "Macromedia" to your stop list. The more omnipresent keywords filtered out at this stage, the less processing needed to calculate the item vectors.

3. The next step we should be thinking about is called stemming. The list of keywords will contain the same words in different forms: singular and plural, different verb tenses, and so forth. Stemming is a technique that removes suffixes from words, leaving a common root. A popular algorithm used for this purpose is the Porter Stemming Algorithm, which is explained in great detail at [www.tartarus.org/~martin/PorterStemmer](http://www.tartarus.org/~martin/PorterStemmer). For brevity, we will turn a blind eye to stemming in this example, but we are missing out on a further reduction in the term space dimensions and the associated performance and usability benefits.
4. The list of keywords is sorted alphabetically (after converting the list to an array, for speed) and duplicates are removed. The getUniqueKeywords method can return either a basic keyword list or a list with a record of keyword frequency. The term space does not need to record frequency, so we need only the straight list for now. We will use the frequency option later, when we call the same method while calculating item vectors.

The final result is an array (arTermSpace) listing the unique keywords contained in the sample content. Each keyword represents an axis in the term space.

## Step 3: Build the Item Vectors

With the term space set up, the next step is to calculate the vectors for each item. We do this in default.cfm (see Listing 1) by calling the buildItemVectors public method in vector.cfc:

```
arItemVectors = objVector.buildItemVectors(arItems=arItems,
arTermSpace=arTermSpace, intTitleWeightFactor=3);
```

This method loops through the content array and performs the following steps for each item:

1. A concatenated string containing the title and body is sent to the getUniqueKeywords method, which this time returns a list of the keyword frequencies. A title weighting factor (in this case, intTitleWeightFactor=3) is introduced by repeating the title string multiple times in the concatenated string.
2. Armed with the list of keyword frequencies (arItemKeywords), the item vector is built by looping through each keyword in the term space and recording the number of times that the keyword appears in the item. Term space keywords that do not appear in the item have a zero value recorded in their vector. Each item vector is stored in arItemVectors[x].arVector, with the corresponding item ID in arItemVectors[x].intItemID.



It is worth noting at this point that this will probably not be the optimum way to store the term space and vectors. If your application stores content in a database, you can build/increment the term space and item vectors when a new item is added and store them in the database.

#### Step 4: Return List of Item-Relevance Matches

With the term space and item vectors calculated, the preparation work is over with. We can now get on with calculating some cosine measures and returning relevance rankings.

In default.cfm (see Listing 1), we build an arguments collection to send to the getItemMatches public method (see Listing 2). The arguments collection consists of the vector of the current item being viewed (arCurrentItemVector), the vectors of the other items in the content collection (arItemVectors), the maximum number of matches to be returned (iMaxRows), and a lower-bound value for relevance (iThreshold). The example is set to return all example items – you will need to tweak the arguments based on your own content. A large value for iThreshold will return fewer, but more relevant, matches.

The getItemMatches method (see Listing 2) loops through the other item vectors and calculates the cosine measure of each one compared with the current item by calling the calculateCosineMeasure private method. If a cosine measure value is greater than or equal to the lower-bound value, it is added to a result array (arItemMatches). Once the full array is built, it is sorted using the arrayOfStructsSort UDF by Nathan Dintenfass, available from cflib.org at [www.cflib.org/udf.cfm?ID=359](http://www.cflib.org/udf.cfm?ID=359). The array is then truncated to contain iMaxRows items.


#### Step 5: Displaying the Results

The template outputs the title and body of the current item and a list of item matches (by looping through arItemMatches). The match titles are looked up from the main arItems content array, the cosine measure is recorded, and links allow the user to reload the default.cfm page to focus on a different item.

### Conclusion and Further Development

So there we have it – a vector space engine that provides “related items” functionality without the need for additional editorial overhead or collaborative filtering based on complicated user interaction.

**“The vectors of items that share multiple keywords will be close together in the vector space, whereas items that share few keywords will be farther apart”**

These techniques are only the first steps into latent semantic indexing. Further processing would apply some factors to the item vectors to reduce the effect of terms that appear in a large number of items and normalize item lengths. The “semantic” part of the technique comes when the entire model is put through a singular value decomposition algorithm (see <http://mathworld.wolfram.com/SingularValueDecomposition.html>). This reduces the number of axes in the term space and compresses adjacent vectors together. In effect, keywords are superimposed, meaning that semantically similar words such as “car” and “automobile” become mathematically identical, making search results much more useful. A good introduction to the world of LSI can be found at: [http://javelina.cet.middlebury.edu/lisa/out/cover\\_page.htm](http://javelina.cet.middlebury.edu/lisa/out/cover_page.htm). 

#### About the Author

Matt Perdeaux is technical architect at Headshift, an Internet consulting firm specializing in online social interaction. Matt has been designing and developing ColdFusion-based applications for five years in both corporate and agency environments. Matt's Web applications have received a number of awards, including 2003 CRM Industry Innovation of the Year and the 2003 Platinum Internet Marketing Attorney award.

[matt@perdeaux.co.uk](mailto:matt@perdeaux.co.uk)

**FREE\*CD! (\$198.00 VALUE!)**

*Secrets of the ColdFusion Masters*  
Every **CFDJ** Article on One CD!



**— The Complete Works —**

CD is edited by **CFDJ** Editor-in-Chief Robert Diamond and organized into 23 chapters containing more than 450 exclusive **CFDJ** articles!

All in an easy-to-navigate HTML format! **BONUS: Full source code included!**

ORDER AT [WWW.SYS-CON.COM/FREECD](http://WWW.SYS-CON.COM/FREECD)

\*PLUS \$9.95 SHIPPING AND PROCESSING (U.S. ONLY)

©COPYRIGHT 2004 SYS-CON MEDIA. WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE. ALL BRAND AND PRODUCT NAMES ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS OF THEIR RESPECTIVE COMPANIES.



Only from the World's Leading i-Technology Publisher

## vector space

### Listing 1

```
<cfparam name="intItemID" default="1"> <!-- Default item ID to
display if no URL variable present -->
<cfscript>
    objVector = CreateObject("component", "vector");

    // STEP 1 - Set up a holding array containing test items
    arItems = ArrayNew(1);
    arItems[1] = StructNew();
    arItems[1].intItemID = 1;
    arItems[1].vcTitle = "Blair defends his Iraq strategy";
    arItems[1].txtBody = "Tony Blair told MPs he accepted full
        responsibility for mistakes highlighted by the Butler Report,
        which he described as made 'in good faith'. ...";

    arItems[2] = StructNew();
    arItems[2].intItemID = 2;
    arItems[2].vcTitle = "MoD denies iPod ban";
    arItems[2].txtBody = "The Ministry of Defence (MoD) has denied
Reuters reports claiming it has banned high-capacity personal storage
devices, such as iPods. ...";

    arItems[3] = StructNew();
    arItems[3].intItemID = 3;
    arItems[3].vcTitle = "Blair accepts Butler report findings";
    arItems[3].txtBody = "TONY Blair today welcomed the Butler report
findings, saying it showed the government and intelligence services
acted in 'good faith'. ...";

    arItems[4] = StructNew();
    arItems[4].intItemID = 4;
    arItems[4].vcTitle = "UK military denies ban on iPod";
    arItems[4].txtBody = "The Ministry of Defence has denied reports that
it has banned Apple's iPod due to fears it could be used to steal
sensitive files. ...";

    arItems[5] = StructNew();
    arItems[5].intItemID = 5;
    arItems[5].vcTitle = "Butler 'A by-Election Boost for Blair'";
    arItems[5].txtBody = "Lord Butler's report into the intelligence on
Iraq could help Tony Blair in tomorrow's by-elections, a political
expert said today. ... ";

    // STEP 2 - Set up term space (find unique keywords in all items)
    arTermSpace = objVector.prepareTermSpace(arItems=arItems);

    // STEP 3 - Build Vector for each item
    arItemVectors = objVector.buildItemVectors(arItems=arItems,
        arTermSpace=arTermSpace, intTitleWeightFactor=3);

    // STEP 4 - Find item matches for current item being viewed
    args = StructNew();
    args.arCurrentItemVector = arItemVectors[intItemID].arVector;
    arItemVectorsWithoutCurrent = arItemVectors;
    temp = ArrayDeleteAt(arItemVectorsWithoutCurrent, intItemID);
    args.arItemVectors = arItemVectorsWithoutCurrent;
    args.iMaxRows = 5;
    args.iThreshold = 0;
```

```
    arMatches = objVector.getItemMatches(argumentCollection=args);
</cfscript>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head><title>Vector Space Example</title></head>
<body>
<!-- STEP 5 - Display the results -->
<cfoutput>
<h1>#arItems[intItemID].vcTitle#</h1>
<p>#arItems[intItemID].txtBody#</p>
<ul>
    <cfloop index="cLoop" from="1" to="#ArrayLen(arMatches)#">
        <li>
            <a
href="default.cfm?intItemID=#arMatches[cLoop].intItemID#">#arItems
[arMatches[cLoop].intItemID].vcTitle#</a>
            (#arMatches[cLoop].intCosineMeasure#% match)

        </li>
    </cfloop>
</ul>
</cfoutput>
</body>
</html>
```

### Listing 2

```
<cfcomponent hint="Vector Space routines Component" displayName="vector">
<!-- Include arrayOfStructsSort UDF by Nathan Dintenfass, available
at http://www.cflib.org/udf.cfm?ID=359 -->
<cfinclude template="arrayOfStructsSort.udf" />

<!-- PRIVATE METHODS -->
<cffunction name="removePunctuation" access="private" returnType=
"string" output="false" hint="Remove punctuation and HTML from
input string">
    <cfargument name="sString" type="string" required="true" />
    <cfset var sWords = "" />
    <cfscript>
        sWords = REReplaceNoCase(arguments.sString,
            "<(.|\\n)+?>", " ", "ALL");
        sWords = REReplace(sWords, "&(.+?);", " ", "ALL");
        sWords = REReplace(sWords, "[[:punct:]]", " ", "ALL");
        sWords = REReplace(sWords, "[[:cntrl:]]", " ", "ALL");
        sWords = Replace(sWords, "£", " ", "ALL");
        sWords = REReplace(Trim(sWords), "\\s{1,}", " ", "ALL");
    </cfscript>
    <cfreturn sWords />
</cffunction>

    <cffunction name="removeStopWords" access="private"
returnType="string" output="false" hint="Removes stop words from input
string">
    <cfargument name="sString" type="string" required="true" />
    <cfset var arStopWords =
ListToArray("$.0,1,2,3,4,5,6,7,8,9,a,able,about,after,again,all,almost,alr
eady,also,although,am,an,and,another,any,are,are,around,as,at,b,based,be,b
ecause,been,before,being,between,both,bring,but,by,c,came,can,com,come,com
```

# Reach Over 100,000 Enterprise Development Managers & Decision Makers with...



*Offering leading software, services, and hardware vendors an opportunity to speak to over 100,000 purchasing decision makers about their products, the enterprise IT marketplace, and emerging trends critical to developers, programmers, and IT management*

Don't Miss Your Opportunity  
to Be a Part of the Next Issue!

## Get Listed as a Top 20\* Solutions Provider

**For Advertising Details  
Call 201 802-3021 Today!**

\*ONLY 20 ADVERTISERS WILL BE DISPLAYED. FIRST COME FIRST SERVE.



*The World's Leading i-Technology Publisher*

A new tool for MX professional  
developers and designers...

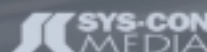


### ADVERTISE

Contact: Robyn Forma  
robyn@sys-con.com  
(201) 802-3022  
for details on rates  
and programs

### SUBSCRIBE

[www.sys-con.com/  
mx/subscription.cfm](http://www.sys-con.com/mx/subscription.cfm)  
1 (888) 303-5282



# MX

developer's journal



## vector space

```

es,could,d,did,do,does,doing,done,e,each,eight,else,etc,even,every,f,fiv
e,for,four,from,g,get,gets,getting,go,going,got,h,had,has,have,he,he,her
,here,him,himself,his,how,however,href,http,i,if,in,including,into,is,it
,it,its,j,just,k,kb,know,l,like,looks,m,mailto,make,making,many,may,mb,m
e,means,might,more,more,most,move,mr,much,must,my,n,need,needs,never,nic
e,nine,no,not,now,o,of,often,oh,ok,on,on,one,only,or,org,other,our,out,o
ver,own,p,piece,q,r,rather,re,really,s,said,same,say,says,see,seven,sev-
eral,she,should,since,single,six,so,so,some,something,still,stuff,such,t
,take,ten,than,that,the,their,them,them,then,there,there,these,they,they
,thing,things,this,those,three,through,to,too,took,two,u,under,up,us,use
,used,using,usual,v,ve,very,via,w,want,was,way,we,we,well,were,what,when
,where,whether,which,while,whilst,who,why,will,with,within,would,x,y,yes
,yet,you,your,z") />
    <cfset var cStopWord = 0 />
    <cfset var sWords = arguments.sString />
    <cfscript>
        for (cStopWord=1; cStopWord LTE ArrayLen
            (arStopWords); cStopWord = cStopWord + 1) {
            sWords = ReplaceNoCase(sWords, " " &
                arStopWords[cStopWord] & " ", " ", "ALL");
        }
    </cfscript>
    <cfreturn sWords />
</cffunction>

<cffunction name="getUniqueKeywords" access="private"
returntype="Array" output="False" hint="Returns a list of unique
keywords in a string.">
    <cfargument name="txtContent" type="string" required="true" />
    <cfargument name="bRecordFrequency" type="boolean"
        required="false" default=false />
    <cfset var lstKeyWords = "" />
    <cfset var arKeywords = ArrayNew(1) />
    <cfset var arKeywordsOutput = ArrayNew(1) />
    <cfset var sPrev = "" />
    <cfset var cWord = 0 />
    <cfset var temp = "" />
    <cfscript>
        // Remove punctuation & HTML
        lstKeyWords = removePunctuation(" " &
arguments.txtContent & " ");

        // Remove stop words
        lstKeyWords = removeStopWords(lstKeyWords);

        // Convert string to list of single words
        lstKeyWords = REReplace(Trim(lstKeyWords),
"\s{1,}", " ", "ALL");

        // Convert list to array and sort into
        alphabetical order
        arKeywords = ListToArray(lstKeyWords);
        temp = ArraySort(arKeywords, "textnocase");

        // Run through words list, removing duplicates and
        recording the frequency if applicable
        sPrev = "";
        for (cWord=1; cWord LTE ArrayLen(arKeywords);

```

```

        cWord = cWord + 1) {
            if ((Not IsNumeric(arKeywords[cWord]))
                AND (arKeywords[cWord] NEQ sPrev)) {
                if (arguments.bRecordFrequency) {
                    arKeywordsOutput[ArrayLen
                        (arKeywordsOutput)+1] = StructNew();
                    arKeywordsOutput[ArrayLen
                        (arKeywordsOutput)].vcKeyWord
                        = arKeywords[cWord];
                    arKeywordsOutput[ArrayLen
                        (arKeywordsOutput)].intFrequency = ListValue
                        CountNoCase(lstKeyWords, arKeywords[cWord]);
                } else {
                    arKeywordsOutput[ArrayLen(arKeywordsOutput)+1] =
                        arKeywords[cWord];
                }
                sPrev = arKeywords[cWord];
            }
        }
    </cfscript>
    <cfreturn arKeywordsOutput />
</cffunction>

<cffunction name="calculateCosineMeasure" access="private"
returntype="numeric" output="False" hint="Returns cosine measure
(0 to 100) for two supplied vectors">
    <cfargument name="arVector1" type="array" required="true" />
    <cfargument name="arVector2" type="array" required="true" />
    <cfset var cAxis = 0 />
    <cfset var iNumerator = 0 />
    <cfset var iSumSquares1 = 0 />
    <cfset var iSumSquares2 = 0 />
    <cfset var iCosineMeasure = 0 />
    <cfscript>
        // loop through each axis and keep running totals
        for (cAxis=1; cAxis LTE ArrayLen(arguments.arVector1);
            cAxis=cAxis+1) {
            iNumerator = iNumerator + (arVector1[cAxis]*arVector2
                [cAxis]);
            iSumSquares1 = iSumSquares1 + arVector1[cAxis]^2;
            iSumSquares2 = iSumSquares2 + arVector2[cAxis]^2;
        }
        iCosineMeasure =
Round(100*(iNumerator/(sqr(iSumSquares1)*sqr(iSumSquares2))));
    </cfscript>
    <cfreturn iCosineMeasure />
</cffunction>

<!-- PUBLIC METHODS -->
<cffunction name="prepareTermSpace" access="public" returntype=
"Array" output="False" hint="Takes items from array and returns
the term space.">
    <cfargument name="arItems" type="array" required="true" />
    <cfset var txtContent = "" />
    <cfset var cItem = 0 />
    <cfset var arTermSpace = ArrayNew(1) />
    <cfscript>

```

# Subscribe Today!

## SAVE 16% OFF

### 12 Issues for \$89<sup>99</sup>

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/](http://www.sys-con.com/coldfusion/) or call 1-800-303-5282 and subscribe today!

**ColdFusion** Developer's Journal



## CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
2004 RCAs	<a href="http://WWW.SYS-CON.COM">WWW.SYS-CON.COM</a>	888-303-5282	39
ACTIVEPDF	<a href="http://WWW.ACTIVEPDF.COM">WWW.ACTIVEPDF.COM</a>		4
CFDYNAMICS	<a href="http://WWW.CFDYNAMICS.COM">WWW.CFDYNAMICS.COM</a>	866-233-9626	11
CFDJ	<a href="http://WWW.SYS-CON.COM/COLDFUSION/">WWW.SYS-CON.COM/COLDFUSION/</a>	888-303-5282	49
COLDFUSION RESOURCE CD	<a href="http://WWW.SYS-CON.COM/FREECD">WWW.SYS-CON.COM/FREECD</a>	888-303-5282	45
FUSEBOX 2004	<a href="http://WWW.CFCONF.ORG/FUSEBOX2004/">WWW.CFCONF.ORG/FUSEBOX2004/</a>	301-424-3903	25
FUSETALK	<a href="http://WWW.FUSETALK.COM">WWW.FUSETALK.COM</a>	866-477-7542	29
HAL HELMS, INC	<a href="http://WWW.HALHELMS.COM">WWW.HALHELMS.COM</a>		31
HOSTMYSITE.COM	<a href="http://WWW.HOSTMYSITE.COM/CFDJ">WWW.HOSTMYSITE.COM/CFDJ</a>	877-248-4678	27
IS+S	<a href="http://WWW.ISSJOURNAL.COM">WWW.ISSJOURNAL.COM</a>	888-303-5282	43
INTERAKT ONLINE	<a href="http://WWW.INTERAKTONLINE.COM">WWW.INTERAKTONLINE.COM</a>		15
INTERMEDIA.NET	<a href="http://WWW.INTERMEDIA.NET">WWW.INTERMEDIA.NET</a>	800-379-7729	COVER IV
IT SOLUTIONS GUIDE	<a href="http://WWW.SYS-CON.COM/IT">WWW.SYS-CON.COM/IT</a>	201-802-3021	47
JAVA RESOURCE CD	<a href="http://WWW.SYS-CON.COM/FREECD">WWW.SYS-CON.COM/FREECD</a>	888-303-5282	19
MACROMEDIA	<a href="http://WWW.MACROMEDIA.COM/GO/DWUPDATED">WWW.MACROMEDIA.COM/GO/DWUPDATED</a>		COVER II & PAGE 3
MACROMEDIA MAX	<a href="http://WWW.MACROMEDIA.COM/GO/MAX">WWW.MACROMEDIA.COM/GO/MAX</a>		17
MX DEVELOPER'S JOURNAL	<a href="http://WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM">WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM</a>	888-303-5282	47
PAPERTHIN	<a href="http://WWW.PAPERTHIN.COM">WWW.PAPERTHIN.COM</a>	800-940-3087	6
SEAPINE	<a href="http://WWW.SEAPINE.COM">WWW.SEAPINE.COM</a>	888-683-6456	23
SYS-CON REPRINTS	<a href="mailto:KRISTIN@SYS-CON.COM">KRISTIN@SYS-CON.COM</a>	201-802-3026	33
WEB SERVICES EDGE EAST	<a href="http://WWW.SYS-CON.COM/EDGE">WWW.SYS-CON.COM/EDGE</a>	201-802-3045	35
WEBCORE TECH	<a href="http://WWW.WEBCORETECH.COM">WWW.WEBCORETECH.COM</a>	877-WCT-HOST	COVER III

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.

# Don't Miss

CFDJ's

# Next Issue!



**ColdFusion Security Best Practices:** A thorough overview of ColdFusion security coding practices.

**Java Wrapper for GNU Privacy Guard:** A look at GnuPG, an open source replacement of PGP.

**Top Ten Web Security Tips:** A few simple steps can make your site much more secure.

**Using ColdFusion for Network Monitoring:** ColdFusion can now access a wide array of network information via SNMP.

## vector space

```

        // Add content of title and body field to holding string
        for each item in content array
            for (cItem=1; cItem LTE
ArrayLen(arguments.arItems); cItem=cItem+1) {
                txtContent = txtContent &
arguments.arItems[cItem].vcTitle & " " &
arguments.arItems[cItem].txtBody & " ";
            }
        // Get list of unique keywords
        arTermSpace =
getUniqueKeywords(txtContent=txtContent, bRecordFrequency=false);
    </cfscript>
    <cfreturn arTermSpace />
</cffunction>

<cffunction name="buildItemVectors" access="public"
    returnType="Array" output="False" hint="Builds vectors for each
    item in provided term space.">
    <cfargument name="arItems" type="array" required="true" />
    <cfargument name="arTermSpace" type="array" required="true" />
    <cfargument name="intTitleWeightFactor" type="numeric"
        required="false" default=1 />
    <cfset var arItemVectors = ArrayNew(1) />
    <cfset var cItem = 0 />
    <cfset var cTerm = 0 />
    <cfset var cItemTerm = 0 />
    <cfset var txtContent = "" />
    <cfset var temp = "" />
    <cfset var arItemKeywords = ArrayNew(1) />
    <cfscript>
        for (cItem=1; cItem LTE ArrayLen(arguments.arItems);
            cItem=cItem+1) {
            arItemVectors[ArrayLen(arItemVectors)+1] = StructNew();
            arItemVectors[ArrayLen(arItemVectors)].intItemID =
                arguments.arItems[cItem].intItemID;

            // Get unique keywords and frequencies in item content
            txtContent = RepeatString(arguments.arItems[cItem].vcTitle & "
", arguments.intTitleWeightFactor) & " " &
                arguments.arItems[cItem].txtBody & " ";
            arItemKeywords = ArrayNew(1);
            arItemKeywords = getUniqueKeywords(txtContent=txtContent,
                bRecordFrequency=true);

            // Set up empty array to hold distance along each axis in the
            term space
            arItemVectors[ArrayLen(arItemVectors)].arVector = ArrayNew(1);

            // Loop through vector array and record frequency of keyword
            in item vector
            cItemTerm = 1;
            for (cTerm=1; cTerm LTE ArrayLen(arguments.arTermSpace);
                cTerm=cTerm+1) {
                if ((cItemTerm LTE ArrayLen(arItemKeywords)) AND
                    (arguments.arTermSpace[cTerm] EQ arItemKeywords
                    [cItemTerm].vcKeyword)) {
                    arItemVectors[ArrayLen(arItemVectors)].arVector[cTerm] =
                        arItemKeywords[cItemTerm].intFrequency;
                    cItemTerm = cItemTerm + 1;
                }
            }
        }
    </cfscript>
    <cfreturn arItemVectors />
</cffunction>

```

```

    } else {
        arItemVectors[ArrayLen(arItemVectors)].arVector
            [cTerm] = 0;
    }
}
</cfscript>
<cfreturn arItemVectors />
</cffunction>

<cffunction name="getItemMatches" access="public" returnType="array"
    output="False" hint="Return list of item matches">
    <cfargument name="arCurrentItemVector" type="array" required="true"
        hint="Current item vector"/>
    <cfargument name="arItemVectors" type="array" required="true"
        hint="Full set of item vectors"/>
    <cfargument name="iMaxRows" type="numeric" required="false"
        default="10" />
    <cfargument name="iThreshold" type="numeric" required="false"
        default="10" />
    <cfset var arItemMatches = ArrayNew(1) />
    <cfset var cItem = 0 />
    <cfset var intHoldingCosMeasure = 0 />
    <cfscript>
        // Calculate cosine measure for each item array and record if
        greater than provided threshold value
        for (cItem=1; cItem LTE ArrayLen(arguments.arItemVectors);
            cItem=cItem+1) {
            intHoldingCosMeasure = calculateCosineMeasure(arVector1=
                arguments.arCurrentItemVector, arVector2=arItemVectors
                [cItem].arVector);
            if (intHoldingCosMeasure GTE arguments.iThreshold) {
                arItemMatches[ArrayLen(arItemMatches)+1] = StructNew();
                arItemMatches[ArrayLen(arItemMatches)].intItemID =
                    arItemVectors[cItem].intItemID;
                arItemMatches[ArrayLen(arItemMatches)].intCosineMeasure =
                    intHoldingCosMeasure;
            }
        }

        // Sort array by cosine measure
        arItemMatches = arrayOfStructsSort(arItemMatches,
            "intCosineMeasure", "desc", "numeric");

        // Truncate array if longer than maximum number of rows
        if (ArrayLen(arItemMatches) GT arguments.iMaxRows) {
            for (cItem=ArrayLen(arItemMatches); cItem GT
                arguments.iMaxRows; cItem=cItem-1) {
                temp = ArrayDeleteAt(arItemMatches, cItem);
            }
        }
    </cfscript>
    <cfreturn arItemMatches />
</cffunction>
</cfcomponent>

```

Download the Code...  
Go to [www.coldfusionjournal.com](http://www.coldfusionjournal.com)



# One damn good Cold Fusion hosting firm!

Webcore Technologies specializes in ColdFusion, ASP and SQL hosting. We offer these solutions in both dedicated server and shared virtual environments.

Webcore can design and implement clustered or load-balanced solutions, managed firewalls, VPN's, multi-site failover, and more. In addition, we also offer corporate Internet access, web site design, and technology consulting services.

Our in-house tier 1 level data center enables us to provide the most reliable hosting in the industry. Our Microsoft and Cisco certified team ensures that all support issues are resolved promptly and professionally. Unlike other hosting companies that answer with a phone attendant, you will receive a live person when you call Webcore. No phone attendant, no frustrations, just world class customer service and support the first time you call.



Webcore Technologies, Inc.  
877.WCT.HOST  
[www.webcoretech.com](http://www.webcoretech.com)

# be the pilot!

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



## 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](http://www.Intermedia.NET).

Managed Hosting • Shared Hosting • Microsoft Exchange Hosting

Call us at: 1.800.379.7729 • Visit us at: [WWW.INTERMEDIA.NET](http://WWW.INTERMEDIA.NET)

