



COLD FUSION Developer's Journal

ColdFusionJournal.com

July 2001 Volume: 3 Issue: 7

Sept. 24-25, 2001 New York, NY



12

Oct. 22-25, 2001 Santa Clara, CA



55

Sept. 24-25, 2001 New York, NY



53

Editorial

It's Time for Teams!

Robert Diamond page 5

Foundations

A Script for Teamwork

Hal Helms page 14

Journeyman ColdFusion
Charles Arehart page 48

Q&A

Bruce Van Horn page 50

First Look

Eron Cohen page 52



by Sarge Sargent page 6

CFDJ Feature: Maintaining Live Verity Collections in a Clustered Environment

Keeping live Verity data indexed and accessible to multiple Web servers

Jeremy Petersen & Dan Kison



18

Software Systems: Tracking Software Issues

Using the Web to collaborate on tracking issues

David Keener

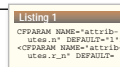


22

Custom Tags: User-Defined Functions in CF 5.0

Deciding between custom tags and user-defined functions

Mike George

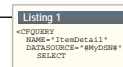


26

Data-Driven Pages: 'Disappear' from the Invisible Web

Your page hits are about to go way up

Matt Robertson



32

CFDJ Feature: VTML by Example Part 2

How to successfully extend the CF Studio IDE

Christian Schneider



36

Practical CF: CFContent with Images

to secretly track readers of your e-mails

Eron Cohen & Michael Smith

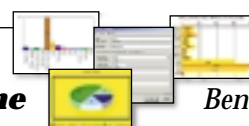


42

<BF> on <CF>: Be Extremely Graphic

ColdFusion 5.0 introduces a true graphing engine

Ben Forta



44

ABLECOMMERCE
www.ablecommerce.com

ABLECOMMERCE
www.ablecommerce.com

CORDA TECHNOLOGIES

www.corda.com

international advisory board

Jeremy Allaire, *CTO, macromedia, inc.*
 Charles Arehart, *CTO, systemanage*
 Michael Dinowitz, *house of fusion, fusion authority*
 Steve Drucker, *CEO, fig leaf software*
 Ben Forta, *products, macromedia*
 Hal Helms, *training, team allaire*
 Kevin Lynch, *president, macromedia products*
 Karl Moss, *principal software developer, macromedia*
 Ajit Sagar, *editor-in-chief, XML journal*
 Michael Smith, *president, terratech*
 Bruce Van Horn, *president, netsite dynamics, LLC*

department editors

editor-in-chief
 Robert Diamond robert@sys-con.com
vice president, production
 Jim Morgan jim@sys-con.com
executive editor
 M'lou Pinkham mpinkham@sys-con.com
managing editor
 Cheryl Van Sise cheryl@sys-con.com
editor
 Nancy Valentine nancy@sys-con.com
associate editor
 Jamie Matusow jamie@sys-con.com
associate editor
 Gail Schultz gail@sys-con.com
associate editor
 Brenda Greene brenda@sys-con.com
assistant editor
 Gregory Ludwig greg@sys-con.com
product review editor
 Tom Taulli
tips & techniques editor
 Matt Newberry

writers in this issue

Charles Arehart, Eron Cohen, Robert Diamond, Ben Forta, Mike George, Hal Helms, David Keener, Dan Kison, Jeremy Peterson, Matt Robertson, Sarge Sargent, Christian Schneider, Michael Smith, Bruce Van Horn

subscriptions:

For subscription requests please call
 1 800 513-7111 or go to: www.sys-con.com
 domestic \$89.99/yr. (12 issues)
 canada/mexico \$99/yr
 overseas \$129/yr
 back issues \$12 U.S. \$15 all other

editorial offices: SYS-CON MEDIA, INC.

135 Chestnut Ridge Rd., Montvale, NJ 07645
 Telephone: 201 802-3000 Fax: 201 782-9600
 COLD FUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)
 is published monthly (12 times a year)
 for \$89.99 by SYS-CON Publications, Inc.,
 135 Chestnut Ridge Rd., Montvale, NJ 07645

postmaster: send address changes to:

COLD FUSION DEVELOPER'S JOURNAL
 SYS-CON MEDIA
 135 Chestnut Ridge Rd., Montvale, NJ 07645

copyright © 2001 by SYS-CON MEDIA

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.

For promotional reprints, contact reprint coordinator:
 Christine Russell christine@sys-con.com

SYS-CON PUBLICATIONS, INC., reserves the right to revise, republish, and authorize its readers to use the articles submitted for publication.

worldwide distribution:

by Curtis Circulation Company
 730 River Road, New Milford, NJ 07646-3048

distribution in USA:

by International Periodical Distributors
 674 Via De La Valle, Suite 204, Solana Beach, CA 92075
 Phone: 619 481-5928

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



It's Time for Teams!

BY ROBERT DIAMOND



I'm writing this month's editorial myself, but if I wanted to stay true to the focus of the issue, I'd be writing it with a partner, or several. *Collaboration* is the name of the game in the world of ColdFusion, and in the magazine world as well. It's a team that produces **CFDJ** and without them, you wouldn't have a magazine. If it wasn't for our fantastic team of writers, editorial staff, production designers, and advertising sales folks, we'd be nowhere. Without them, you'd be lucky to receive a blank pile of pages each month. With them all, however, we put together a fantastic resource each month, one that we're quite proud of, and one that I hope you enjoy.

If you haven't guessed yet, this month's focus is on collaboration. The thought of collaboration seems a simple one, but in this not-utopian world we live in, it's not always as easy as it seems. What messes up the whole process? Why, people do, of course.

Collaboration is a hot topic these days; as the scale and diversity of projects increase, the need for more knowledge does as well. This need exists on several levels, ranging from small projects to the enterprise. As demands for new site features increase, from Flash animations to mobile access, many of us are using the resources and knowledge of others to help put it all together. That's by no means a knock on those who can do it all themselves. Definitely the more you know, the better off you are, but no one knows it all, and that's why we're here. It's hard to specialize in everything these days, and tapping the experience of someone else can often be an easy way to reduce development headaches. I'm proud to say that I've taken advantage of the assistance offered by others, and my development life has been better and easier because of it. I've also gotten more sleep at night.

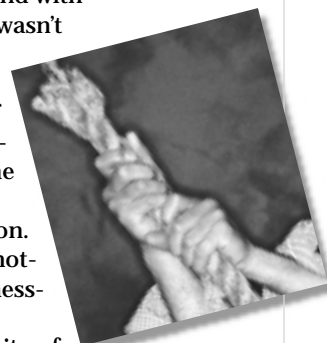
Now....onto some highlights of what the team here at **CFDJ** has in store for you this month.

Hal Helms takes the bird's-eye view of working together in the modern world. E-mail, instant messenger, and the like are all fantastic tools with one problem...they're being used by people! He's got some great tips, tricks, and stories about creating a "Program Definition Language" to get you out of the bind. It's definitely worth a read and, of course, is Fusebox-related as well. If you aren't using Fusebox yet - you should be.

David Keener has written on tracking software issues - using the Web to collaborate on the always tricky art of gathering and fixing bugs in your projects. Along with source code, he provides the schema for making a SQL server-based front-end for tracking issues with your masterpiece in progress. Moving down from a top-level page to projects and issues, with forms for entering all the necessary information, it's an end-to-end solution. If you're building software or working on large projects, especially in a team, you won't regret setting one of these up. It's the easiest way to make sure that everyone is on the same page.

Sarge Sargent, who has conveniently (for us at least) traveled near and far to ColdFusion developers around the globe, hits on all the major issues. Experienced from his days at WashingtonPost.com, and several other major projects, he discusses a fantastic development method that I highly recommend.

Also, we've got lots of other articles, from Master Guru Ben Forta and others, that look at utilizing the new features in CF 5.0. Enjoy!



ABOUT THE AUTHOR

Robert Diamond is editor-in-chief of ColdFusion Developer's Journal as well as SYS-CON's newest magazine, *Wireless Business & Technology*. Named one of the "Top thirty magazine industry executives under the age of 30" in Folio magazine's November 2000 issue, Robert recently graduated from the School of Information Studies at Syracuse University.

Robert Diamond

ROBERT@SYS-CON.COM

I've been out of the hard-core development team dynamic for more than a year now, so I was a little hesitant when **CFDJ** approached me about writing this article. Nevertheless, I have traveled to various ColdFusion shops around the globe and worked with numerous development teams.

The majority of these shops have a tight-knit group of developers, who eat and sleep CF and Web development. Others have one or two developers who happen to have heard about a cool technology that allows integration with powerful back-end systems with an easy-to-use, tag-based language,

ment, but it is important to stress the staging – which can be broken down further into quality assurance (QA) and user acceptance testing (UAT) – and production areas.

You should stage all good production code before going live. I learned this fundamental precept in the early days at WashingtonPost.com, where our access actually included storyboarding, development, staging, and production.

The staging/testing/QA server provides a platform to make necessary code revisions without touching “live” code. Following good principles, the staging server platform mirrors the production

OPTIMAL DEVELOPMENT ENVIRONMENT

Working together in CF

and have let CF lead them to the mountaintop. And then there are those shops filled with developers who swear they are top-notch but don't really have a clue as to when not to use pound signs.

But even those shops at the high end of the CF development spectrum can have environments that are not conducive to good coding practices. Managers, as well as developers, need to understand the crucial aspects of the development environment: team, architecture, and methodology. This article is an attempt to level the playing field, so that even the worst coders are in an environment that will help elevate their game.

Architecture

Although an understanding of the development process is fundamental to successful Web development, that discussion is beyond this article's scope. Rather, my focus is on facilitating this process. Rudimentary to this facilitation are the provision and protection of three distinct environmental areas: development, testing/staging, and production. Obviously, this article focuses on develop-

ment, and contains a frozen copy of the production code and database. This provides backup systems for disaster recovery, allowing full code migration to production, and extra resources to scale the site during unexpected load hikes.

It seems common sense to divide your development life cycle – at a minimum – into the development and production phases. However, you would be surprised – or maybe you wouldn't – at how many sites develop and modify code on production servers, data, and source code.

Again, good practice is to code and test functionality in development, and then migrate this functionality to a staging area for QA and/or UAT. Once approved, this code base should be frozen – then pushed to production. We never, ever want to modify production code! And code should never move from production to development – code should be copied from the staging server to development for code revisions. Let's not continue to bemoan such sites but rather proceed in breaking down the ideal development environment architecture.



“What’s more debilitating and frustrating to a developer or graphic artist than having to be creative on a slow, antiquated machine?”

The Development Server

The first step in achieving the optimal development environment is dedication: management must make the necessary expenditures for dedicated resources – human, hardware, and software. The development team and their IDEs are only part of this environment. (We will cover those later in the article.) For now, let’s focus on the development server.

The ideal development platform will completely mirror the production architecture but usually on a single box. Say your

production environment includes three systems: two Compaq 1850R servers running Windows 2000 Server, IIS 5, and CF 5, and a Compaq 3000 running Windows 2000 Server and SQL Server 2000. Then your development server should be a Compaq 1850R running Windows 2000 Server, IIS 5, CF 5, SQL Server 2000, and some source control software. (We will cover source control in a minute.) Some small shops can only budget for a small development server, but even a desktop HP system should be adequate for these shops.

Nevertheless, if budget permits, align the development systems parallel to production. This provides your developers with a simulated production playground, and provides extra machines for swapping in the event of catastrophic hardware or system failures. Mirroring the production architecture in development also helps eliminate hardware as a possible source for bottlenecks – and we all debug our code and eliminate bottlenecks, don’t we?

Please refrain from implementing a Windows-based development environ-



team, each developer had his or her own copy of CF Studio, a CF server, a Web server, and a local copy of the database. The single-user edition of CF Enterprise that's bundled on the CF Studio CD is provided for development purposes. Developers can integrate this with Peer Web Services on their desktops and code against Microsoft SQL Server 2000 Personal Edition. The Personal Edition comes with both the Standard and Enterprise editions and is not sold separately. It is powerful enough for desktop development.

Obviously, if you have an Oracle shop, you'll want Oracle's developer editions – you want to keep your development, staging, and production environments as homogeneous as possible.

This configuration is ideal because the developers can work independently on local copies of source code and data instead of trampling one another on the development server, or worse, contaminating your production database. In order to do this properly, protect the

code on the development server with enterprise-level version control software.

Most small shops use freeware versions, such as GNU SCCS (www.gnu.org) and ComponentSoftware's CVS (www.componentsoftware.com). Larger players in the change management or version control space are Perforce P4 (www.perforce.com), Merant PVCS (www.merant.com/products/pvcs/), Starbase StarTeam (www.starbase.com/products/starteam/), and Microsoft Visual SourceSafe (<http://msdn.microsoft.com/ssafe/>). Although I have encountered Visual SourceSafe at most of the shops I have visited, more CF shops are turning to PVCS solutions.

The key features for change management are versioning, document comparison and merge, rollback, and deployment. Integration with your IDE, or integrated development environment, is a plus that will help win over your developers – and winning them over is just what you'll have to do if you have not already implemented a version control solution.

Whichever change management system you deploy, the client software must make its use relatively easy in order for developers to consistently use it.

Frameworks and Methodology

Application frameworks and coding methodology coincides with change management and source control. Developers need good application frameworks to form a solid structure within which to properly code cohesive sites. CF has its own Application Framework instantiated with `Application.cfm` and `OnRequestEnd.cfm`; but this framework is really a building block for focusing on Session and Client variable management and some rudimentary application-level security. It does not address the larger, more prevalent issues of code reuse and symmetry, teamwork efficiency and productivity, commenting and documentation, and so on.

Furthermore, a sound application framework provides some base functionality fundamental to the site. This fosters code reuse, as developers can focus on retooling or customizing that piece of functionality instead of reinventing the wheel each time.

Methodologies provide guidelines for coding the application structure. A methodology is composed of standards. It dictates how to code the site or application so that your code is legible, manageable, and reusable. A good methodology reads easily and is documented thoroughly. It is stern enough to keep developers focused, yet flexible enough to allow developers to grow as they learn.

Application framework and methodology are essential to good rapid application development. However, most shops are guilty of coding without one or both.

A Familiar Scenario

You're an IT or Web shop manager of a team of three Web developers. Your team is pretty strong in fundamental CFML and quickly grasping the advanced concepts. Now you have the opportunity to hire what you perceive is a hotshot. You've heard this developer's name all over the CFUG meetings and you leap at the opportunity to enhance your staff.

This developer comes on board and right away begins to affect the team. However, instead of leaving his bags (bad coding habits) at the door, he starts littering your source code with his own unique indentation style, his own custom tags, and so forth. The code he writes is advanced stuff, but he neglected to comment the code, so your other team members cannot learn by his examples, debug his code, or make any editions without his hand-holding. What should you do?

ment for a Solaris production environment. If you realize most Web development tools are Windows-based, these only need to reside on your developers' desktops. So if your production system is a Solaris or Linux solution, your development system(s) should be also. Your developers' desktops should be the only Windows systems in the environment.

The Developer Desktop

What's more debilitating and frustrating to a developer or graphic artist than having to be creative on a slow, antiquated machine? In a perfect world, each developer and/or artist would have a fast, powerful desktop machine with the fastest available processor, ample RAM, and hard-drive space. For CF developers, this machine should not only include a licensed copy of CF Studio, but it should also be a miniversion of the production system.

This means each developer should be working with a local copy of the production source and data. In my last development

"In a perfect world, each developer and/or artist would have a fast, powerful desktop machine with the fastest available processor, ample RAM, and hard-drive space. For CF developers, this machine should not only include a licensed copy of CF Studio, but it should also be a miniversion of the production system"

"This developer comes on board and right away begins to affect the team. However, instead of leaving his bags (bad coding habits) at the door, he starts littering your source code with his own unique indentation style, his own custom tags, and so forth. The code he writes is advanced stuff, but he neglected to comment the code, so your other team members cannot learn by his examples, debug his code, or make any editions without his hand-holding. What should you do?"

The Other Side of the Story

You are the hired gun, and, upon walking into the place, you notice their staff is still coding at the CF 3.11 level – their code is full of pound signs, needlessly complex syntax, and deprecated tags. You remain humble, sit in your cubicle, and begin to do your thing. Your tasks are simple and the coding is basic – to you. You are pleased with your work so far, but at the next monthly meeting, you are reamed for your nonstandard methods and lack of comments, and accused of changing the entire application. Do you leave or stay?

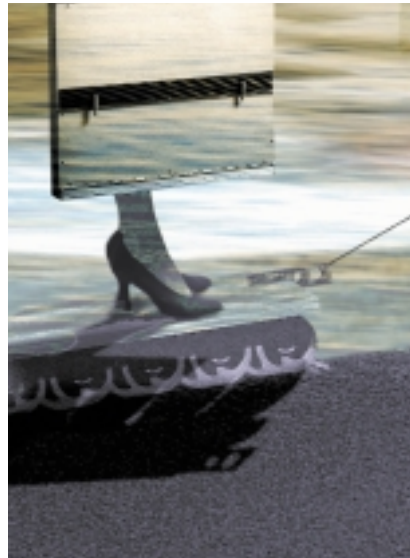
A Slightly Different Scenario

Again, same setup as the first scenario, you're this same IT or Web shop manager of three. You decide to send your hungriest developer to a local Advanced ColdFusion Development course, figuring she will be able to pass the lessons on to the rest of the staff. Instead, she takes what she has learned and finishes the next major application revision with advanced methods. However, she now realizes her increased marketability, and serves you notice in lieu of a more lucrative position. She leaves without sharing her onion or documenting the code enhancements she just made. Now what do you do?

There are numerous scenarios to spin through, but I think the point is now obvious. A good application framework and methodology could have remedied all three scenarios. A sound application framework and clear, well-documented methodology would have facilitated the hired gun's adjustment to the team dynamic. Certainly some of his coding style could have enhanced the framework, providing impetus for upgrading the current methodology.

In the case of the hungry developer, providing good documentation for every code revision or even mandating a comment section at the head of each template as part of the coding methodology would allay this situation. If the developer jumps ship, then the rest of the team doesn't have to remain in the dark. The knowledge transfer has already taken place.

Where should shops turn for an application framework and/or methodology? iiFramework (www.iiframework.com) is a popular CF application framework. Other object-oriented CF application frameworks include SmartObjects (www-smartobjects.com) and cfObjects (www-cfobjects.com). Fusebox (www.fusebox.org) is the most popular CF methodology. Spectra is Macromedia's answer to the CF application framework and methodology paradigm. (Many readers are aware that version 1.5.1 is the last feature-additive release of Spectra as the core components are being melded into the next major release of the CF server.) Each of these has its share of strong points and shortcomings.



Patrick Steil does a great job comparing Fusebox, Allaire Spectra, and iiFramework in his article, "Rad+++ Propel Your ColdFusion Project with Application Frameworks" in *CFDJ* (Vol. 2, issue 11). See Benjamin Pate's "Introducing SmartObjects: Build Extendable, Reusable, Object-Based Components Using CFML," article in *CFDJ* (Vol. 2, issue 8) for an object-oriented CF framework primer.

The Weakest Link

The most important part of the development environment is the team dynamic.

Like the proverbial chain, a development team is only as strong as its weakest developer. That said, the importance of a Web development team composed of technically sound individuals cannot be over-emphasized. There are three key members of the Web development team: Web/CF developer, graphic, and database.

At a minimum, a good Web developer today is well versed in HTML 4, Dynamic HTML (DHTML), Cascading Style Sheets (CSS), and ANSI-SQL. It is not enough to simply know about deprecated tags, such as the FONT tag, but in knowing how to correctly code cross-browser, inline style syntax is fundamental.

A strong Web developer has mastered complex DHTML and JavaScript routines – having built a vast library of eye-catching mouseovers, flying layers, and so on – and possesses some fundamentals of server-side languages such as CFML, ASP, JSP, and Java.

The complete Web developer is one who not only has mastered all of the above prerequisites, but whose lunch pail also includes the full Web development smorgasbord and Internet alphabet soup: XML\XHTML, WAP\CHTML\I-Mode, LDAP, SSL, and so forth.

Building upon that weak link, the consummate CF developer has mastered all of Ben Forta's tips and tricks, understands the complexities of structures and WDDX (Web Distributed Data Exchange) packets, and understands the intricacies of advanced caching techniques, load distribution, and session management. The well-rounded CF developer contributes to the community by actively participating in local ColdFusion Users Groups (CFUG).

The CF developer is the MVP of the development team, so it would behoove the good IT manager to have several MVP candidates on the team. To ensure MVP status, Macromedia has developed two certification offerings as part of its Macromedia Certified Professional Program (MMCP): Certified ColdFusion Developer and Certified Web Site Developer.

For a complete listing of CFUGs, visit the Allaire Web site at <http://devex.allaire.com/developer/usergroups/>; for more informa-

tion on WDDX see www.openwddx.org/; for more information on ACP, see www.allaire.com/services/training/certification/index.cfm.

The final team members are the graphic artist and database administrator. Typically, the CF developer fills one or both of these roles as well, however, the team will function more productively if each member focuses on the area of his or her specialty in designing/developing the site or application. A great number of CF developers can animate images in tools like Macromedia Fireworks or crop photos with Adobe Photoshop. However, there is no substitute for a true graphic designer who can take an idea from charcoal and sketchpad to full-length Flash animation in a few hours.

Likewise, there is no substitute for a good DBA to manage your database. For dynamic Web applications, nothing is as

Tools of the Trade

The key to rapid application development is accessibility to tools that help you effectively build and deploy powerful, high-end applications quickly. There are a myriad of Web development tools, or integrated development environments (IDE), and graphic suites out there to help attain this goal. I started my Web development career with Notepad, then moved on to Kenn Nesbitt's WebEdit, and Nick Bradbury's HomeSite – acquired by Allaire in 1996.

Today, the industry is crowded with good WYSIWYG editors, such as Microsoft's FrontPage 2000 and Adobe's GoLive. I won't even go into the numerous Java IDEs. (Sorry, I did not intend this to be a biased product review, but there are really only two IDE choices for serious CF development: UltraDev and Studio.)

When it comes to graphics, artists usually pick their poison and stick with it.

Dreamweaver UltraDev. UltraDev takes Dreamweaver to the edge by complementing the robust Web design engine with support for such data-driven languages as CF, ASP, and JSP. UltraDev integrates with ColdFusion Studio to provide an unparalleled combination of database and debugging tools – and visual development environment to help rapidly design and deliver powerful applications.

For traditional CF development, ColdFusion Studio is the best – hands down! ColdFusion Studio is the de facto CF IDE – pardon the pun. Allaire built Studio on its award-winning HomeSite HTML editor and added features such as full CFML (ColdFusion Markup Language) support, Remote Data Services, a CSS editor, and Source Code Control support. Studio's Project metaphor integrates with any Microsoft Source Code Control (SCC) API-compliant versioning software to help facilitate team coordination on all applications: simple or complex. Again, Macromedia has done a great job of integrating its leading tools – Studio for CF development, UltraDev for visual development – into one sweet package, ColdFusion 4.5 UltraDev 4 Studio.

As I said, CF Studio is my tool of choice for CF development. However, I am increasingly finding Dreamweaver UltraDev and Fireworks on more developer's desktops, and Photoshop on graphic artists' Macs, at each shop I visit.

Conclusion

The optimum CF development environment has dedicated human, hardware, and software resources. Developers use industry-leading IDEs to develop against local resources and within a versioning system on the dedicated development server. The staging server gets a copy of this code base for testing, where it is frozen upon acceptance for migration to production.

Your job as an IT or Web shop manager is to make this development environment conducive to your developers' efforts. Your job as CF developers is to use standards-based methodologies within application frameworks to help facilitate the rapid application development process.



About the Author

Sarge is a senior consultant and practice manager with CF Services with Macromedia Consulting Services. He has been coding advanced applications in CF since version 2.0.

ssargent@macromedia.com

"The key to rapid application development is accessibility to tools that help you effectively build and deploy powerful, high-end applications quickly"



vital and volatile as the data in the database. It is indeed the backbone and life of the application. Take it from a former CF Developer/SQL Server DBA, you don't want your CF developer writing code and worrying about Database Maintenance Plans, truncating logs, proper replication scenarios, or any other DBA functions.

We have a saying in the industry: "Let the Database and Web Server do their jobs, and let ColdFusion do its." The same goes for the development team – let each member do his or her job and focus on his or her specialty. However, separation of power does not equal segregation of force. Dividing the Web team along duty lines is not a call to build up walls, put on blinders, or break the lines of communication.

Communication is the key to success in any relationship – Web development is no different. Part of letting the database do its job is the CF developer asking the DBA to create a stored procedure on the database to replace a long-running query.

Adobe Photoshop (www.adobe.com/products/photoshop/main.html) continues to be the industry standard for traditional graphics and photo retouching, shadowed by CorelDraw 10 Graphics Suite (www3.corel.com/cgi-bin/gx.cgi/App-Logic+FTContentServer?pagename=Corel/Product/Details&id=CC110Y1YKCC), Ulead PhotoImpact (www.ulead.com/pi/-runme.htm), and Macromedia Freehand (www.macromedia.com/software/freehand/). Macromedia's Fireworks 4 (www.macromedia.com/software/fireworks/) has become the favorite among Web developers for Web graphics creation, animation, optimization, and integration.

Macromedia solidified its spot atop the IDE market with the release of its

CFDYNAMICS
www.cfdynamics.com

COLDFUSION EDGE 2001, FAST TRACK

The ONLY ColdFusion Event Backed by
the Power of **SYS-CON MEDIA** and **COLDFUSION Developer's Journal**

Cf COLD FUSION
fasttrack

REGISTER
ONLINE
TODAY!

Save
\$300

Save the Dates

Prepare for ColdFusion Certification

Kevin Lynch



Kevin Lynch is president of Macromedia Products. He joined Macromedia in 1996 and has been instrumental in forming its Web strategy. As president of products, Kevin is responsible for developing Macromedia's award-winning family of software and solutions.

Jeremy Allaire



As Chief Technology Officer, Jeremy is instrumental in guiding Macromedia's product direction and is the company's primary technology evangelist, responsible for establishing key strategic partnerships within the Internet industry. Jeremy was the founder and former chief technology officer for Allaire Corporation, which merged with Macromedia in March 2001.

Ben Forta



Ben Forta is Allaire Corporation's product evangelist for the ColdFusion product line. He is the author of the best-selling ColdFusion 4.0 Web Application Construction Kit and its sequel, Advanced ColdFusion 4.0 Development, as well as Allaire Spectra E-Business Construction Kit and Sams Teach Yourself SQL in 10 Minutes. He recently released WAP Development with WML and WMLScript.

Plan to Attend

Prepare for ColdFusion Certification Exam with
Comprehensive Technical Sessions designed for
Beginner and Advanced ColdFusion Developers

Session Topics include:

- Web Fundamentals
- Application Development
- Database Concepts
- Client State Management
- Troubleshooting, and a special focus on ColdFusion 5.0.

webServices

JAVA DEVELOPERS JOURNAL

XML JOURNAL

wireless

UNIX BUSINESS WEEK

COLDFUSION Developer's Journal

WebSphere

Dr. Dobb's

webtechniques

CK FOR COLDFUSION CERTIFICATION



Hilton New York, New York City

***fasttrack* Program**

**SEPTEMBER
24-25 2001**

A Preliminary Glance at Topics

- | | | |
|--|--|--|
| <ul style="list-style-type: none">• ColdFusion 5.0• Cross browser support• Browser/server interactions• HTML standards related to ColdFusion• Variables and scoping• Looping• Functions• <CFTAGS>• Conditional processing• Custom Tags• Arrays and their usage• Dimensions• CFSRIPT possibilities and limitations• Structures• Passing structures into custom tags• Variable scopes | <ul style="list-style-type: none">• Combining complex variables• Upload files• MIME types• Append/Retrieve/Download• <CFCCONTENT>• Creating agents• Tags and parameters• CF HTTP• Syndication• WDDX• Parameterizing• Data-parsing techniques• HTTP header information• Exception handling• Database interactions• Stored procedures• CF transaction management | <ul style="list-style-type: none">• Bind parameters• SQL queries• Joining tables• Grouping data output• Handling nulls• Database manipulation• Caching queries• Application.CFM• Cookies• Client variables• Session variables• Application variables• Server variables• Storage locations• Debugging |
|--|--|--|



Call
201 802-3004
to reserve
exhibit space
today!

Owned and Produced by



135 Chestnut Ridge Road
Montvale, NJ 07645
(201) 802-3069
Fax: (201) 782-9051
visit

WWW.SYS-CON.COM/COLDFUSIONEDGE
for more information

NEITHER SYS-CON MEDIA NOR SYS-CON EVENTS, INC., ARE SPONSORS OF, OR ARE AFFILIATED IN ANY WAY WITH, CAMELOT COMMUNICATIONS, INC., OR ANY EVENTS PRODUCED BY CAMELOT COMMUNICATIONS, INC. JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS AND SYS-CON EVENTS ARE INDEPENDENT OF SUN MICROSYSTEMS.

A Script for Teamwork

BY
HAL
HELMS



We need a better plan for developing software remotely

In this issue of *CFDJ*, the editors are looking at the subject of collaboration and ColdFusion.

There are many types of collaboration, of course, and this month I'd like to talk about distributed team development with ColdFusion.

I was first introduced to this idea by a former boss of mine, who, under cost-cutting pressures, decided that working remotely had the decided benefit of him not paying for office space and, with no more effort than pushing the "Send" button on his e-mail, remote working was the new "new thing."

Soon we had people in Georgia, Florida, California, and Missouri all (apparently) working together seamlessly! I say "apparently" because to the nonmanagerial eye, it looked like a complete mess, but my boss sent glowing reports along to the company president touting our success as remote workers.

Much had been said about teleconferencing, high-speed Internet access and the like, but somehow those niceties were never implemented. No matter, we just kept on remotely working – right up until the point that the company president decided to speak with some of his remote workers and heard a decidedly different story.

Before long, we had a new boss, many of the remote workers were let go, and the idea of remote working was now the "old thing." It was back to fighting traffic to get to the office on time.

It seemed to me this was a missed opportunity. Since that experience, I've tried to see how distributed team development could be made a reality. In speaking with others, it turned out that the situation at my old company was not such an uncommon scenario. From all corners, good intentions were being negated by bad communication.

Whatever else can be said about it, warm bodies congregating together in a small space does promote communication. Naturally, much of it is not germane to the work to be done, but we are all used to this. For the most part, we are willing to accept that admitted inefficiency of office chatter. It's a small price to pay for the informal communication network that forms the backbone by which information is shared in most corporations.

Most often, the attempts to improve communication center on trying to reestablish the informal communication network to function remotely. This takes the shape of companies paying for extra phone lines, DSL access, telephone headsets, and fax machines – as well as pushing remote workers to pay more attention to their e-mail and/or instant messaging. Just last week I heard about a particularly valuable employee who wanted to work remotely; his company responded by having a T-1 line run to his house!

Sometimes this works. Still, an awful lot of time is wasted waiting for a colleague to respond via instant messenger (you don't know it, but she just got up to make some tea) – and we're all familiar with how much time e-mail can waste. I think we make a fundamental mistake when we try to "virtualize" the experience of bumping into someone on the way to the watercooler or overhearing a colleague in a cubicle next to ours.

Albert Einstein once tried explaining the idea of a wireless telegraph:

"The wireless telegraph is not difficult to understand. The ordinary telegraph is like a very long cat. You pull the tail in New York, and it meows in Los Angeles. The wireless is the same, only without the cat."

Something similar might be said

about virtual watercooler conversations. We need a better plan for developing software remotely – without the watercooler – one that recognizes that the communication medium appropriate to physical proximity may not be the same for remote teaming.

I think we have good examples of people having solved this problem all around us. When you plug your laptop into the wall, you take it for granted that the plug will fit the socket. This happens because a specification was established on the interface between a 110-volt electrical appliance and the electricity grid that powers our homes and buildings. Such a specification lets manufacturers of different products communicate – without the watercooler.

Manufacturers may have specifications that allow them to turn out huge volumes of their product with confidence that it will interoperate with others. In software development, however, we have no such thing. The situation is much more analogous to musket making prior to the invention by Eli Whitney of standardized parts – each product is a unique work of its craftsman-creator; any interoperability between parts is coincidental.

In software this situation arises because we often design as we code. We don't lay down a specification because we don't know what that specification is beforehand. Putting the best face on it, it "evolves" as the project moves along.

I find this to be a singularly bad idea – one whose only virtue is that we're all too familiar with this "methodology." Because I wanted to be able to work remotely, I have worked hard to build capabilities for remote teaming into Fusebox. Along the way, I've tried to assemble some of

CODECHARGE
www.codecharge.com

the best practices of developers to create a system that would allow me to have consistently successful projects.

The result of that work, which I've called Extended Fusebox (or XFB for short), lays out how an integrated application methodology that begins with wireframes ends with acceptance testing. Over the last two years of using XFB, I've done a number of development jobs remotely. I have one client for whom I've done quite a bit of work (including a fully functional e-commerce site) who I've never met in person. I've found that success in remote working is achieved not by trying to reproduce a physical environment remotely but in having clear, understandable, simple specifications.

Knowing what I'm going to build before I build just seems to work better! And this happens only when I use things like wireframes and prototypes to help the client see what it is that they want. As I've said many times, it came as a revelation to me that clients weren't telling me what they wanted in requirements-gathering meetings because they didn't know what they wanted – until they saw it.

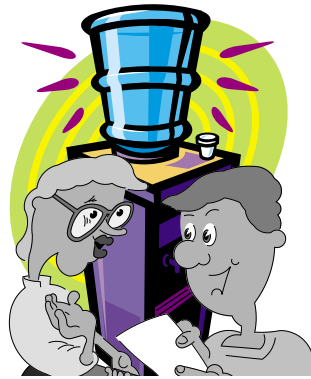
I know of no better way than Web prototyping to let clients "see it" before the code is written. This is particularly effective as, to most clients, the interface – the prototype – is the application. With a prototype, you'll finally get the feedback that usually doesn't surface until you deliver the application. If you use my DevNotes code, you'll be able to communicate with clients and developers remotely while having a central repository for these communications.

So far, so good. Clients and project managers can work remotely using the Web to determine what will be built. But it's developers who will write the code to turn the prototype into reality. We won't be successful in remote teaming unless we can find a way to incorporate remote developers into our process.

I've found the first step in achieving this is architecting the program. Since I use XFB as my methodology, my architecture will reflect this Fusebox-centric view of the world. You may use some other methodology – or you may have your own, unstated one. Whichever method works for you, I find archi-

tecting the program is essential.

In my experience with large-scale, complex Web applications, decisions need to be made at a global level about modules, variables, and responsibilities – yet I know many developers whose idea of architecture is designing the database. The rest of the architecture is left to the coders to figure out. But from my observations, that laissez-faire approach sets coders – and the project – up for failure.



I would certainly not recommend that approach for a remote teaming environment. Instead I have found success in giving coders clear, explicit guidance in the form of what is sometimes called a program definition language (PDL), which forms a sort of contract (not the legal kind, thankfully) between the architect and the coder. The architect, in effect, says to the coder, "You fulfill this contract for this specific bit of code and I'll worry about the integration with the rest of the program."

Of course, you would only take such a position if you had confidence in your architecture, but if you do, creating a PDL for coders to deliver on is remarkably effective. Coders, free of the nagging questions about how this particular line of code will affect the others in the program, are able to turn out clean, precise code. After all, it's seldom the problem of finding the correct syntax that slows down programming. Rather it's all the "what if" questions that are (to coders at least) like weights to a runner.

As the project becomes more involved, there are more of these what-if questions – more weights – and the project bogs down. This leads to the almost universal experi-

ence of writing software: the first 90% takes three months and the final 10% takes an additional three months.

Having a PDL (I use Fusedoc, which I created just for this purpose) can transform the experience for both coder and manager. The manager, having created an application architecture and translated this into PDLs for individual files, can give these to any number of programmers to complete. My design goal in creating Fusedoc was that any competent ColdFusion programmer should be able to complete the code file without knowing anything more about either the application or the underlying database.

This may seem to border on insanity. I recently came across a quote by Neils Bohr, the Danish physicist who was at the heart of so many discoveries in that wacky world of quantum physics, that seemed to put it best: "We all agree that your theory is crazy. The question is, 'Is it crazy enough?'" Let me relate an experience of remote teaming and you be the judge of whether the theory is crazy enough.

The job was an e-commerce site for a large technology company in California. My client was an independent project manager in Boston. I was working in Atlanta. When my client was awarded the job, it was with the proviso that the job be done in one month.

"You what?" I sputtered. "How are you going to do that?"

"Well, actually, Hal, that's why I'm calling you..."

It was clear to me that we had only one chance for success – there just wasn't any time for recoding because we hadn't quite understood the client – or because the client had never made clear what they wanted.

"I want you to spend three weeks on the prototype," I told my client. You can probably imagine his reaction, but he was intrigued enough to hear more. "This project won't fail because we can't figure out how to implement security, or get credit card approvals, or keep customers in our database," I told him. "If it fails, it will be because we didn't deliver what the client wanted – and the only way we can find out what they want is to deliver it to them: thus, the prototype."

"But here's the thing: the prototype has to look exactly like the fin-

We all agree that your theory is crazy.
The question is, 'Is it crazy enough?'

—Neils Bohr, Danish physicist

ished product – no variations at all. If there are, you've given your client wiggle room and we'll never get agreement on acceptance testing. The prototype you're doing has to form the basis for acceptance testing – so you've got to get it right."

He did a tremendous job. When I got the prototype from him, it looked exactly like a real application. (In fact, he had some trouble convincing his client that it wasn't the real application!) My job was then to turn that prototype into an application architecture that would hold up under stress – both the stress of many simultaneous users and the stress that would surely come later when the client needed to maintain it.

Once this was done, I wrote Fusedocs for the individual code files that the architecture portion had identified. I had just over four days left to deliver a working application. I had already lined up about 20 coders (literally across the world) who (1) knew the XFB methodology and (2) were waiting for the code.

They had no idea what the application was; all they got were individual files. They never saw the database; it had been very recently designed by my friend and colleague, Jeff Bane, and we had used a technique I developed called *query sims* to provide query data to the developers without actually hooking their code up to a database.

Forty-eight hours later, I got all the code back. There were more than 200 separate fuses. Part of my agreement with the developers was that each one would provide test harnesses for their code so that I knew the files had been unit-tested.

I will confess that I was more than a little nervous as I began to "stitch" the files together. You know the old saying that in theory there's no difference between practice and

theory – but in practice, there is! A little over four hours later I was ready to do what one developer calls a "smoke test," where you run the code and look for smoke pouring out of your server.

Now, I'd like to tell you that everything went perfectly well; all files integrated seamlessly. If this had been a movie, that's just how it would have been written, but it didn't turn out quite this way. As a matter of fact, it turned out that, in my haste, I had left out two files that needed to be written – and in seven or eight others I had neglected to tell the coder about a variable that had to be passed along.

I wrote the Fusedocs for the missing files, wrote the code for those files, and then added in the pass-thru variable in the others. This took me about two-and-a-half hours. The next time I turned the machine on, no smoke: the application worked. Wacky as it sounds, we delivered the project early. More important, we discovered that a remote development team can work if we understand the dynamics behind it and create a system that accommodates these.

So the story has a happy ending: my client in Boston got the job done on time; his client in California was thrilled; working in Atlanta, I learned a whole lot about remote teaming; a bunch of coders from several different countries were paid well to write clean code without having to try to read the client's mind – all in all, a Hollywood sort of ending.

Of course, every movie has a curmudgeon – and mine was Jeff Bane, who was

a little worried that we might have created sky-high expectations for the sequel to our "movie." Well, I appreciated his concern, but I assured him that I had duly impressed on my client that this was an awfully high-risk project; that things could have happened that could have sabotaged our efforts; that next time we really should ...

Cell phone rings.

"Hello? Oh, hello, David! How's the weather in Boston? You know, I was just writing an article about the experience with remote development of the e-commerce site. I was just thinking how crazy we must have been to take on...what's that? A full company intranet? And you told them what?!! Listen, David..."

Fade to black.

For more information on the tools and methodologies for remote teaming discussed in this article, visit www.halhelms.com.



ABOUT THE AUTHOR

Hal Helms
(www.halhelms.com)
is a Team Allaire member who provides both on-site and remote training in ColdFusion and Fusebox.


HAL.HELMS@TEAMALLAIRE.COM

PACIFIC ONLINE

www.paconline.net



Maintaining **Live** Verity Collections in a Clustered Environment



“...The Verity 97 engine was designed to handle ongoing ‘file transactions’ while live searches are being performed”

The task seemed monumental: “Come up with a way to keep live Verity data indexed and accessible to multiple Web servers in a clustered environment.” The scope of the data was huge – hundreds of thousands of pieces of content – with new additions made 24 hours a day, every day of the year. All known methods to accomplish this task just did not seem to do the job adequately. It was time to get creative.

While analyzing the inner workings of the stock Verity 97 engine that ships with ColdFusion 4.5, the following keys to success were identified:

1. Verity searches are CPU-intensive; thus it would be best not to force all the servers in the cluster to search on one server, but rather make each server responsible for its own searches to best spread out the workload.
2. Verity indexing is even more CPU-intensive than the searching; thus any box stuck with the task of continual indexing would spend most of its time pegged at 100% CPU usage and would not be desirable for any other task.

The theory started to clarify. We would dedicate one server to continually update the indexes. This would be called the *Utility Server*. As soon as this server finished a round of updates, it would start the next. As mentioned above, this Utility Server would spend most of its time at 100% CPU usage, so it would be used only for the index updates.

The next piece of the puzzle was to determine the best way to share file access to these Verity collections. With ever-changing data and a CPU operating at 100%, the Utility Server would not be an option. So we decided to dedicate another box to simply hold the most current and searchable version of the indexes. This would be called the *Index Server*.

The final component was the Web servers. This was easy, as all they had to do was map Verity collections to the *Index Server*.

The basic setup made sense, but some of the details were still missing:

1. If the Utility Server was copying updated index files to the Index Server, what would happen to any ongoing searches that were currently using the collections on the Index Server?
2. How would we clean old index data from the Index Server without impacting ongoing searches?

One idea was to work with multiple collection groups on the Index Server, say collection “A” and collection “B.” While collection “A” was being indexed and copied over, collection “B” would handle all of the searches from the Web servers. This sounded like a possible solution but added an extra degree of complexity. Was the extra complication necessary?

What if we could simply use CFFILE to copy new index data and delete old index data directly to the live set of collections on the Index Server? But that would go directly against what we have been told is a best practice (locking all Verity collections during update transactions). Before we gave up on the idea, we contacted Verity representatives to see if they could offer any advice. We were pleasantly surprised when they informed us that the Verity 97 engine was designed to handle ongoing “file transactions” while live searches are being performed. Time to run some tests and see what we could find out.

For our test environment we set up a small clustered environment of our own (see Figure 1). On another set of eight machines (two machines per Web server), we opened a total of 20 browser instances on each of the four Web servers and ran test code (see Listing 1) that was looping over a basic CFSEARCH on a Verity collection.

At the same time the searches were being performed, we were updating the Verity collection on the Utility Server, copying the collection to the Index Server, and deleting old Verity files from the Index Server. Building the indexes using CFINDEX and copying the files with the use of CFDIRECTORY and CFFILE was very straightforward (see Listing 2) but deleting the files required a little more thought. Because Verity collections store many files in pairs, we would need some logic that deleted all but the latest pair of files in each Verity subfolder. So we came up with a custom tag called `tag_delete` (see Listing 3).

Deleting Old Verity Files

It’s important to understand the structure of the Verity files. This helps to better understand why we delete what we do with the Delete tag. To learn the file structure of Verity collections and why the deletion of the old Verity files is important, see www.allaire.com/handlers/index.cfm?id=18429&method=full (“Understanding Verity Collections in ColdFusion”).

```
<!-- call to the tag -->
<cf_tag_Delete
    source=""
    fileDateTime="">
```

The tag is called by passing two parameters. The source parameter is the directory path in which to begin deletion, and fileDateTime is a time stamp used to mark how far in the past the tag can delete (see Listing 3).

As the code in Listing 1 was executing, the output would begin by returning the old record count. As we updated the indexes, the output would change to reflect the new data:

103
103
103
103
127
127

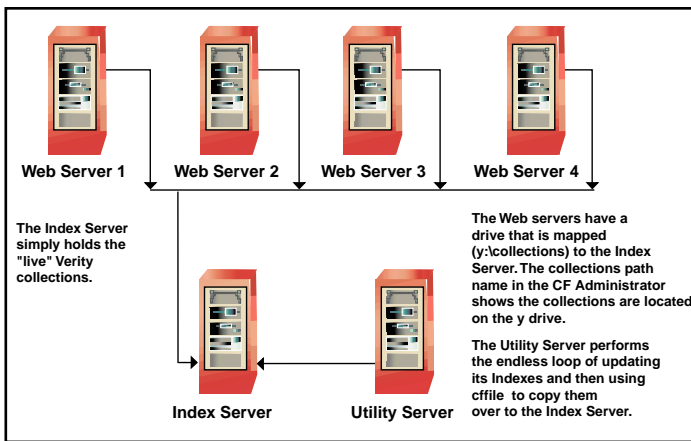


FIGURE 1: Live Verity collection server environment

Over the course of many such tests, we did not experience any collection corruption or errors of any kind.

Interacting with Copies of Index Files

To clarify a major point in regard to not locking our Verity index transactions, we're talking only about interacting with copies of the index files themselves and CFSEARCH, *not* the actual index creation and modification process of CFINDEX – as that code is always run with single thread access on the Utility Server.

Live for Six Months

Our tests were successful. We moved from our test data to an actual clustered environment and started using real data. The final product has been live for six months (at the time of this writing) with no ill effects. We are confident in this theory, but, as always, make sure to run tests in your own environment to guarantee that it will work for you.

The following is a more detailed look at the machines we used and a basic set-up procedure for each machine.

Machines Used

- One Utility Server
- One Index Server
- Four Web servers (multiple)

Utility Server

The Utility Server has a scheduled task set to run every minute. This scheduled task sets a lock file (lock_verityCollectionUpdate.txt). When this scheduled task begins, it checks the existence of this lock file. If the file exists, the task will end without attempting to do an update. If the lock file does not exist, the task will run. It will write the lock_verityCollectionUpdate.txt file, retrieve all new data since the last time it ran, and then update and optimize all Verity collections. Next the task will copy all new index files to the Index Server. It then reads over the Index Server's collection files and removes all but the latest files. Finally, the task will delete the lock_verityCollectionUpdate.txt file.

Index Server

The Index Server simply holds the “live” Verity collections.

Web Servers

The Web servers all map their collections to the Index Server. There are no local collections on these machines; thus, as per load balancing setup, each Web server will supply the CPU usage for any searches its visitors require. As Verity searches can become CPU-intense, this gives your searches the most bang for your buck.

Setup Procedure

- Ensure that all ColdFusion tasks are running under accounts that have security privileges to work with all servers. (In Windows NT 4.0 and 2000, go to Services, then check the properties of the CF Services to see what account they're running under.)
- Make sure to use realistic request timeouts, so that your indexing code will not timeout before it's complete.

Utility Server

- Map a drive to the location in which you'll store the files on the Index Server.
- Create the collections on this machine. Depending upon how long it takes for your collections to build, it's advisable to build your collections using Netscape 4.X – as IE has been known to time out.
- Set up a scheduled task to run task_verityCollection-Update.cfm (see Listing 4) to run every minute. Enable this task as soon as the Index Server is set up.

Index Server

Copy the collections from the Utility Server to the Index Server. This will prime the pump and needs to be done by hand only once, for initial setup. This step is also important to set up the proper directory structure for the files on the Index Server, so that the file transactions can take place.

Web Servers

- Map drive to the Index Server.
- In the CF Administrator, set up a new mapped Verity collection pointing the map drive to the Index Server you just set up. *Note:* You will need to get the collection name exactly as it's shown on the Utility Server.

About the Authors

Jeremy Petersen is certified in ColdFusion and has been using it since version 1.5. He is the manager of Web application engineering for TeachStream, Inc., in Salt Lake City, Utah.

Dan Kison has worked with ColdFusion since version 3.0. He created Web sites for the Air Force for four years, and continues to build Web applications.

jeremy.petersen@teachstream.com

dkison@yahoo.com

Listing 1

```
<!-- test search code -->

<cfloop index="loopOn" from="1" to="50">
  <cfsearch name="s1"
    collection="test"
    criteria="*">
    <cfoutput> #s1.recordCount#<br></cfoutput>
</cfloop>
```

Listing 2

```
<!-- tag_Copy.cfm full code listing -->
<!-- This custom tag copies a complete folder
structure (including sub folders) -->
<cfdirectory action="LIST"
  directory="#attributes.source#"
  name="dirlist">
<cfloop query="dirlist">
  <!--if we are on a sub directory then
call another instance of this tag for the sub
directory. Also, ignore the . and .. directory
entries. -->
  <cfif dirlist.type is "Dir"
    AND dirlist.name is not "."
    AND dirlist.name is not "..">
    <cf_tag_Copy
```


Listing 3

Listing 4

```

set to #now()#
<!-- write new lock file-->
<cffile action="write"
file=
"#utilityServerPath#lock_verityCollectionUpdate.txt"
output="#Now()#">

<!-- update index --->
<cfindex collection="#collectionName#"
action="Update"
type="path"
key="C:\Inetpub\wwwroot\work"
urlpath="127.0.0.1/work/"
extensions=".html,.htm,.cfm,.txt"
language="English"
recurse="yes">
<P>The collection has been Indexed

<!-- optimize index --->
<CFCOLLECTION
Action="Optimize"
Collection="#collectionName#">

<P>The collection has been optimized

<!-- copy optimized index files to "live"
location on index server-->
<cf_tag_Copy
source="#utilityServerPath##collectionName#\
destination="#indexServerPath##collectionName#\">
<P>The Collection has been copied to
#indexServerPath##collectionName\

<!-- remove old index files from
"live" location --->
<!-- read in date/time of last lock
from file --->
<cffile action="READ"
file="#utilityServerPath#
lock_verityCollectionUpdate.txt"
variable="lockDate">

<cf_tag_Delete
source="#indexServerPath##collectionName#\
fileDateTime="#lockDate#">
<P>#indexServerPath##collectionName\
has been cleaned of old index data files

<!-- output results --->
<CFSET eTime = #Now()#>
<p>Start: <b>#sTime#</b> <br>
End: <b>#eTime#</b> <br>
Total time (seconds):
<b>#DateDiff("s", sTime, eTime)#</b>

<!-- remove lock file-->
<cffile action="delete"
file="#utilityServerPath#
lock_verityCollectionUpdate.txt">
<p> Deleting Lock File.

<!-- lock file exists --->
<cfelse>
<font size="+5" color="Red">LOCKED!</font>
<!-- read in date/time of last
lock from file --->
<cffile action="READ"
file=
"#utilityServerPath#lock_verityCollectionUpdate.txt"
variable="lockDate">
<p>LockDate: #lockDate#
<p>File: #utilityServerPath#
lock_verityCollectionUpdate.txt

<!-- lock file is too old (over 45 minutes)-
lets override/remove it --->
<cfif DateDiff("n", lockDate, now()) GT 45>
<!-- remove lock file-->
<cffile action="delete"
file=
"#utilityServerPath#lock_verityCollectionUpdate.txt">
</cfif>

</cfif>
</cfoutput>
</body>
</html>

```

DDDDDDDDDD

Tracking Software Issues

Using the Web to collaborate on tracking and resolving software issues

BY
DAVID
KEENER



Building software systems is a lengthy and complicated process that may involve anywhere from a single developer to multiple teams of developers, who may all be working in different locations."

Many projects include complicating factors, such as the integration of diverse technologies, interaction with other software systems, or adherence to a defined set of requirements.

Is it any surprise that the software development process is prone to errors? Communication errors. Software defects (bugs). Or maybe even just ideas for features to be implemented in the next phase of a project (that tend to get lost or forgotten if they're not recorded somehow). These types of issues need to be tracked and resolved for any project of significant size.

The Web provides an excellent platform in which to field an application that will allow teams of developers to collaborate on tracking and resolving software issues. I will provide a design for a rudimentary Web-based issue-tracking system, as well as suggestions on potential enhancements to the system.

Goals of the System

The goals of this issue-tracking system are the following:

1. To serve as a central location where all project issues are recorded.
2. To allow developers, testers, and other personnel to collaborate in an essential development task, the resolution of project issues.
3. To serve as a record of issues that have been resolved.

To meet these goals, the system will record issues in a database, which will be Microsoft SQL Server in this case (the design could easily be adapted for other databases). The system design will support multiple projects, each of which may have numerous issues associated with it.

The system will provide a top-level Project page, which will display a list of projects to which a user has been assigned. Each project can have

issues associated with it. The system will allow users to view, create, edit, and delete issues as needed.

Database Design

The foundation for this issue-tracking system is the database design, shown in Figure 1. As a convention, each table name begins with a prefix of "pt_", which means that these tables can generally be dropped into an existing database without producing a name conflict with existing tables.

The PT_PROJECT table provides a list of projects, each with a unique ID and a project name. This table also has a column called ACTIVE_IND (with possible values of Y or N) that indicates whether a project is active or not. This allows completed projects to be marked as inactive, so they don't appear on a user's list of projects.

Each project can have multiple issues associated with it. These issues are stored in the PT_ISSUES table, which contains the following columns:

- **issue_id**: A unique ID for the issue.
- **change_date**: The date and time that the issue is updated.
- **entry_date**: The date and time that the issue is first entered.
- **issue_desc**: A description of the problem. This can be of any length.
- **issue_status_id**: A foreign key to the PT_ISSUE_STATUS_LOOKUP table. An issue can have an issue status ID of "Open," "Test," or "Closed."
- **issue_title**: A short title to describe the issue.
- **issue_type_id**: A foreign key to the PT_ISSUE_TYPE_LOOKUP table.

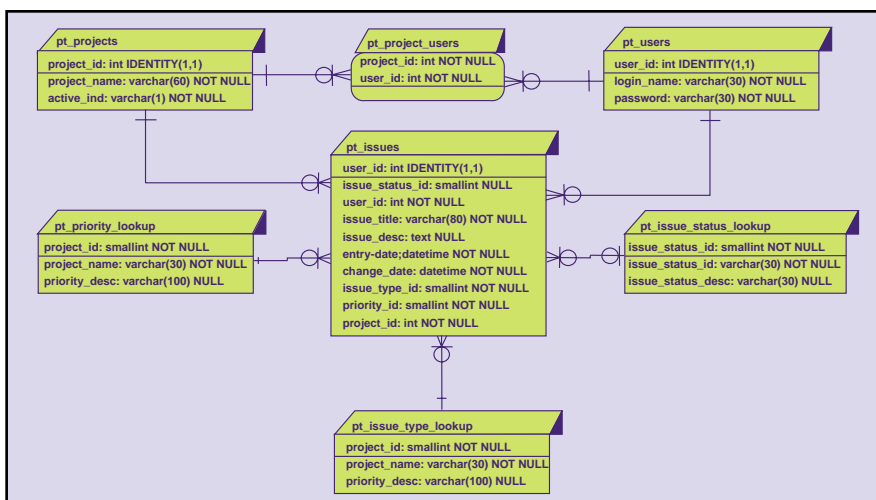


FIGURE 1: Issue-tracking system database design

CONCEPTWARE AG
www.conceptware.com/eye

Issue types may include "Defect," "Idea," "Database Change," or any other types that may be appropriate.

- **project_id:** A foreign key to the PT_PROJECTS table. Stores the project ID of the project to which the issue belongs.
- **priority_id:** A foreign key to the PT_PRIORITY_LOOKUP table. Some issues are just more important than others. This column allows the importance of an issue to be defined, such as "Critical," "High," "Normal," "Low," and "Minor."

The PT_USERS table defines the users for the system. Then, the PT_PROJECT_USERS table defines the projects to which a user has access.

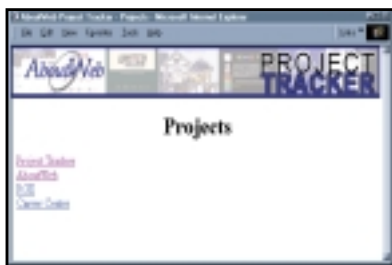


FIGURE 2: Projects page

Let the Tracking Begin

The top-level Web page for the system is the Projects page, which displays a list of active projects to which a user has access. For this example system, the user ID is defined in the session variable, session.user_id. Typically, you would implement a login scheme for the system. Upon successfully logging in, the user ID would be set in a session variable, and would thus be available for the Projects page. Since we're focusing on the issue-tracking aspects of the system, I've provided a snippet of test code at the beginning of the Projects page that hard-codes the number 1 for the user ID. (The code for this article can be found at www.ColdFusionJournal.com.)

The Projects page is displayed in Figure 2, and the corresponding code is shown in Listing 1. The name of each project is set up as a hyperlink to the Issues page, with a URL that includes the project ID as a parameter.

The Issues page (see Figure 3) displays a concise and ordered list of issues for the specified project. Each row shows the title of an issue, the issue ID, the issue status, the priority

and the last change date for the issue. The title of the issue is a link to the Issue View page, which displays all of the details concerning the issue. Each row also includes links to the Edit Issue and Delete Issue pages. Finally, near the top of the page, there's a link to the New Issue page.

Modifying Issue Information

These last three Web pages, the New Issue, Edit Issue, and Delete Issue pages, allow users to change information in the database after responding to a form. In each case, the form posts the information from the form to the same Web page. Thus, each Web page works in two modes.

In the first mode, no form information has been posted to the page. Therefore, the code displays the Web page with the form to collect information from the user. Upon submission, the information from the form is posted to that Web page, where ColdFusion makes them available as form variables.

The code checks to see if one of the form variables is defined. If it is, then the page goes into the second mode, handling the posted information. In the case of the New Issue page, for example, this means inserting information into the database. It uses the CFTRANSACTION tag to ensure that the operation, if it consisted of multiple steps, does not get interrupted. It then redirects the user back to the Issues page, properly specifying the URL parameter for the project.

This technique allows all code associated with a defined task, such as editing issue information, to be contained in a single source file. Some developers prefer not to handle forms this way, but it eases maintenance on those large ColdFusion projects.

Just a Starting Point

While useful, the issue-tracking system described in this article is a fairly basic example of the breed. Here are examples of enhancements that can be added to the system to create a more full-featured software development tool:

- **Login System.** Mentioned previously, the issue-tracking system almost demands a login mechanism.
- **Privileges.** A system for privileges could be created, so that not all users would have access to fea-



FIGURE 3: Issues page

tures, such as creating, editing, or deleting issues. For example, a consulting company might want to make the issue-tracking system available for a client, but set privileges so that only the development team could modify the issues.

- **Security.** Currently, a user could look at issues from a project they're not associated with simply by altering the project ID parameter in the URL for the Issues page. A more robust system would have some security in place to prevent this.
- **Issue Notes.** The system allows an issue to be updated with additional text to describe the resolution process. Many systems store updates as time-stamped notes that are associated with an issue. So an issue might consist of the original issue as entered, plus a series of entries in a PT_NOTES table that would represent a history of the issue and the steps taken to resolve it.

This is just a starting point. There are numerous features that could be added to expand the capabilities of this system.

Universal Accessibility

In only the past few years, the Web has expanded to the point where almost anybody can access the Web no matter where they are. Its near universal accessibility makes it the ideal platform for fielding collaborative Web applications, of which the issue-tracking system described in this article is only one example.

ABOUT THE AUTHOR
David Keener is the chief information officer for AboutWeb (www.aboutweb.com), a Web solutions company located in the Washington, DC, area. He has a BS in computer science and is a specialist in information publishing.

MACROMEDIA

www.macromedia.com/downloads

User-Defined Functions in ColdFusion 5.0

Deciding between custom tags
and user-defined functions:
Which should you be using?

BY
MIKE
GEORGE



Before the release of ColdFusion 5.0, developers had to add user-defined functionality to their applications through custom tags. Now they have the ability to create user-defined functions.

In this article I'll demonstrate some of the similarities and differences between custom tags and user-defined functions in ColdFusion 5.0.

Before we discuss user-defined functions, though, let's review what custom tags are. A ColdFusion custom tag is just a ColdFusion template that can be called within an application. Custom tags are useful for hiding code complexity and allowing other developers to use added functionality in their applications without having to understand the custom tag code.

What Are User-Defined Functions?

User-defined functions are similar to custom tags, in that they hide code complexity; however, user-defined functions are written between `<CFSCRIPT>` blocks. The `<CFSCRIPT>` tag allows developers to write JavaScript-like code in CFML pages. One advantage of using the `<CFSCRIPT>` tag is that functions that often return useless values (e.g., `ArrayAppend()`, `ArraySort()`, `ArrayClear()`, etc.) can be written without being assigned to a variable. For example, the following "traditional" ColdFusion

```
<CFSET tmp =  
ArrayAppend(aMyArray, "3")>
```

can be rewritten in a `<CFSCRIPT>` block as follows:

```
<CFSCRIPT>  
    ArrayAppend(aMyArray, "3");  
</CFSCRIPT>
```

The `<CFSCRIPT>` tag can improve code readability in a num-

ber of situations by sometimes allowing the developer to write code with less awkward syntax. Developers who have experience with JavaScript or C may find writing user-defined functions less clumsy than writing custom tags. For a complete review of `<CFSCRIPT>` syntax, see Ben Forta's article, "Stick to the Script" (*CFDJ*, Vol. 2, issue 7).

Custom tags have a protected variable scope, meaning that any variable used on the calling page must be explicitly passed into the custom tag for it to be accessible. Likewise, any variable used within a custom tag must be passed out for it to be available on the calling page.

Custom tags also provide an advantage in that they can be called by any page within the application. This can be done by placing the tag in the ColdFusion custom tags installation directory (c:\cfusion\CustomTags) or by using the `<CFMODULE>` tag.



Like custom tags, user-defined functions can be written to protect variables used within the function. This is done with the `var` keyword, which has been added to the `<CFSCRIPT>` syntax in ColdFusion 5.0. However, unlike custom tags, user-defined functions can access variables that are available on the calling page.

What this means is that it's possible for the function to overwrite existing variables on the page. Unlike custom tags, user-defined functions can only be called from a page that physically contains the code for the function.

The fact that a developer can not call a user-defined function from any page within the application may seem troubling, but it's easily remedied. If the developer has a library of user-defined functions that he or she would like to access throughout an application, a file containing the `<CFSCRIPT>` code can be saved and simply included through a `<CFINCLUDE>` tag in either the calling document or the Application.cfm file.

Calculating a Factorial

I would like to demonstrate the difference between custom tags and user-defined functions through the calculation of a factorial. A factorial is a simple mathematical computation, which is often depicted as $n!$, where n is the number for which you want to calculate the factorial. A factorial of a positive integer is equal to the number multiplied by all integers less than it. For example, $5!$ is equal to $5 \times 4 \times 3 \times 2 \times 1 = 120$.

MACROMEDIA

www.macromedia.com/go/devcon01

JDJEDGE

conference & expo

2001

September 23-26, 2001
is your most valuable educational
opportunity of the year.
Don't Miss It!

With

Cf **COLD FUSION**
fasttrack

Register
NOW
for best rates!

The Depth and Breadth of Education to Advance Your Professional Skills

The JDJEdge 2001 Conference & Expo provides your best opportunity to understand how Java technologies can solve enterprise challenges.

Four information-packed JDJEdge Tracks featuring leaders in Java Technologies

Track 1 J2ME – Micro Edition & Wireless

Cutting-edge sessions for software engineers and hardware specialists working on wireless solutions.

Track 2 J2SE – Standard Edition

General Java programming for corporate programmers developing full Java applications, including several introductory sessions.

Track 3 J2EE – Enterprise Edition

Advanced sessions for software architects, Web programmers, corporate developers and consultants developing server-based applications.

Track 4 Working with I-Technology

Technical and management sessions for business analysts, corporate systems managers, architects, project managers and CIOs.



JDJ Readers' Choice Awards, the Oscars of the Software Industry, is the world's most widely participated industry award program. Winners will be recognized at JDJEdge 2001 International Java Developer Conference & Expo in New York.

Who Should Attend

- Developers, Programmers, Engineers
- i-Technology Professionals
- Senior Business Management
- Senior IT/IS Management
- Analysts, Consultants

The only event backed by
SYS-CON Media and
Java™ Developer's Journal

JDJEDGE 2001 Features

- Unmatched Keynoters and Faculty
- Over 150 Intensive Sessions and Tutorials
- The Largest Java Expo on the East Coast
Participation by Invitation Only
- BEA WebLogic™ FastTrack to Certification
- IBM WebSphere™ FastTrack to Certification
- Macromedia ColdFusion™ FastTrack to Certification
- Sun Java™ FastTrack to Certification

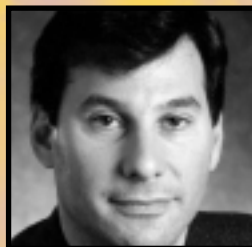
www.sys-con.com/JDJEdge/

Keynote Panel

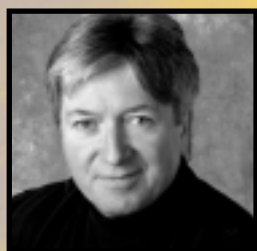
LYNCH
President
Macromedia



BARATZ
CEO, Zaplet, Inc.
Former President, JavaSoft



ROSS
Founder, JavaLobby
Panel Moderator



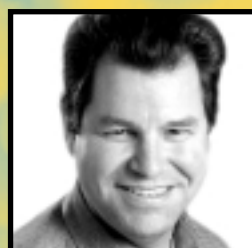
SMITH
Chief Java Strategist
IBM



DIETZEN
CTO
BEA



SCOTT
CEO, PointBase, Inc.
Cofounder, Oracle



GOSLING
Creator of Java
VP & Fellow, Sun Microsystems

ADVISORY BOARD



J. Milbery



S. Phipps



A. Sagar



J. Westra



S. Rhody



A. Williamson

KIESSLING
CEO
Sitraka



Media Sponsors:

Java Lobby	XML Journal	Java Developer's Journal
WebServices Journal	Wireless Business & Technology	Linux Business Week
WebTechniques	Programmer's Paradise	
ColdFusion Developer's Journal	WebSphere Developer's Journal	

Owned & produced by



135 Chestnut Ridge Road,
Montvale, NJ 07645
201 802-3069
Fax: 201 782-9651

In the world before ColdFusion 5.0, we would calculate factorials through the use of a custom tag. To demonstrate, I have created a custom tag (see Listing 1) called `<cf_GetFact>`. My custom tag takes two attributes: `N`, which is the number I want to calculate the factorial for; and `r_N`, which is the name of a variable that I want to contain the returned result. For simplicity, I have not added any error checking to the custom tag.

To calculate $5!$, I call the custom tag as follows:

```
<CF_GetFact N="5" r_N="n">
```

To optionally print out the result of $5!$, I would add the following code after the call to the custom tag:

```
<CFOUTPUT>#n#</CFOUTPUT>
```

A local variable will then be available on the calling page called `n`. In looking at the code for the custom tag, you may notice that while the tag gets the job done, it is not very elegant. In addition, I have to pass two parameters into my tag to get results (the beginning number and the factorial result).

What if I just want to print the value of $5!$ on a page? After the call to `<cf_GetFact>`, I would then need to place the variable `n` within a `<CFOUTPUT>` tag. This whole process may not seem like a big deal, since this is what ColdFusion developers have been doing for some time, but it's better suited to a user-defined function.

If you're familiar with scripting syntax (e.g., the `<CFSCRIPT>` tag), then there's an easier (and much more refined way) to compute a factorial. Since ColdFusion 5.0 allows developers to write user-defined functions, I can simply create a function on the page (using JavaScript-like syntax) within a `<CFSCRIPT>` block (see Listing 2). Again, for simplicity, I have omitted any error checking from the function.

I can now call the function exactly as I would any other ColdFusion function (e.g., `Now()`, `Len()`, `Abs()`, etc.). For example, if I'm only interested in printing the value of $5!$ to the page, I can simply add the following code anywhere

I use both custom tags and user-defined functions in development. Both have their place. Both allow us features and solutions that the exclusive use of one or the other does not. Be flexible, so that you can use them both to their best advantage"

on the page:

```
<CFOUTPUT>#GetFact(5)#</CFOUTPUT>
```

Note that the call to this user-defined function can be either before or after the function is actually defined in the `<CFSCRIPT>` block. ColdFusion 5.0 is smart enough to figure out where the function is defined. However, if you include your user-defined function through a `<CFINCLUDE>` tag, Allaire recommends that you make calls to the user-defined function only after the `<CFINCLUDE>` tag on the page.

Now allow me to explain how the `GetFact()` function works. When the `GetFact()` function is called, the argument of the function (in the above example, the argument is "5"), gets passed into the function as the variable `n`. The function checks to see if the variable `n` is less than or equal to 1. If so, the function returns the value of 1. Otherwise, the function returns the current number (5) multiplied by another call to the function, passing the attribute of `n-1` (4 in this case).

This process continues until the value of the variable `n` is equal to 1. The final result (in the above example) turns out to be $5 \times 4 \times 3 \times 2 \times 1$, or 120. This is a simple example of a recursive function. This function will continue to call itself until a specific condition is met. In this case, the function stops when `n` is equal to 1. User-defined functions in ColdFusion 5.0 do not need to be recursive; however, a recursive solution is better for some types of mathematical computation – such as this factorial example.

Recursive functions are very powerful tools, which, in some instances, are the only viable way to solve some mathematical problems. However, be careful when dealing

with recursion. Recursive functions gobble up memory quickly and you can imagine how an inexperienced programmer could easily write an infinite loop that could use up the system resources.

The `GetFact()` function looks and behaves exactly like a built-in ColdFusion function. An added benefit to using a user-defined function over a custom tag (in this case at least) is the fact that I don't need to worry about passing a variable out of the function, whereas with a custom tag the developer must worry about both the attributes and caller scopes within the custom tag. If I want to assign the results of `GetFact()` to a variable, I can assign a variable to the function within a `<CFSET>` or within a `<CFSCRIPT>` block.

For example:

```
<CFSET n = GetFact(5)>
```

and

```
<CFSCRIPT>
n = GetFact(5);
</CFSCRIPT>
```

are both legal references to the user-defined function. The calls to this user-defined function are both more elegant and easier to understand than the calls to the `<CF_GetFact>` custom tag. While a user-defined function may not always be the best solution to a particular problem, I believe it is the appropriate solution for this example.

Which Should You Be Using?

I've discussed the similarities and differences between custom tags and user-defined functions. Which should you be using? User-defined functions are great if you're accustomed to writing C or

'Disappear' from the Invisible Web

BY
MATT
ROBERTSON



Your page hits are about to go way up

Data-driven sites are a great way to manage and deliver content. But at a price. You may as well kiss deep linking and extensive search-engine indexing goodbye. Here's a solution to this problem, but it's not pretty.

Recently, in an online forum, a new developer asked how to submit data-driven Web pages to search engines. I'm sure the answer was unexpected: pages requiring parameters to display properly can't be submitted at all.

Data-driven pages make up what is commonly known as the invisible Web, whose size has been estimated at hundreds of times the size of the static Web that search engines see.

Engines that can overcome the static limitation – like invisibleweb.com and beta.profusion.com – are still in their infancy, so don't expect them or similar technology to come into widespread use anytime soon.

This article describes a simple method that allows you to produce submit-friendly, seemingly static pages from your dynamic content. The dynamic nature of your content will be preserved, and this method will not require any special sort of administrative access to your server. This will allow search engines to see your pages while you retain the ability to manage them. The number of pages on your site visible to a search engine is about to go way up.

Prepare Yourself

Everyone knows successful projects are built on elegant code written with surgical skill and finesse. Ah, but there's something to be said for putting on a hockey mask and running around with an ax. That's a good characterization of the method I'll discuss here, and in fact the whole idea seems a bit crazy at first glance.

It's All in the Details

Dynamic pages typically rely on some sort of passed variable – often a URL variable – to display data-driven content. Thus to display the details

on three products the links to those pages look something like this:

```
item.cfm?IID=1
item.cfm?IID=23
item.cfm?IID=456
```

The parameter IID is the key value used to pull the requested content from your database (to keep things simple, my examples assume the unique identifier is a numeric variable along the lines of a SQL Identity or Access AutoNumber field).

When indexing the above links, search engine robots will see that question mark and stop dead, thereby ignoring that portion of your site. To solve the problem, we have to do two things: first we must eliminate the need for a passed variable; next we must have one discrete, physical HTML/CFML "entrance" page for each dynamic item in the database.

Listing 1 contains dynamic.cfm: a simple dynamic page showing a store product. The oversimplified design contains only the basics of a data-driven page: a query and its output. There are no meta tags. There's no point in including them, since the page can't be crawled or submitted in the first place.

Listing 2 contains maker.cfm. This template will produce the individual entrance pages. Lines 1–7 select the key value field from all records in the Items table.

The Ax Man Cometh

This is where things get hairy. Lines 8–15 execute a CFLOOP over those query results. This CFLOOP accomplishes two tasks. First, by using query output on Lines 9–10, it creates a variable, used on Line 14 by CFFILE. The CFFILE operation on Lines 11–14 takes static.cfm (see Listing 3) and lit-

erally copies it, as many times as there are records to loop over.

As a result of the above, each newly created copy of static.cfm has a key value embedded in its filename. You'll wind up creating a gaggle of pages named like this:

```
static_1.cfm
static_23.cfm
static_456.cfm
```

What Have I Done?

If you have 200 products in your store, you just created 200 duplicate ColdFusion templates with slightly different names. Let's assess the consequences of this bizarre move.

Listing 3, static.cfm5, serves the same purpose as Listing 1. Added to it is code that extracts the identity number from the filename and makes it more attractive to a search engine.

The CFSET in Lines 1–2 uses the REFind function and the variable CGI.PATH_INFO to hunt down the starting position of the filename-embedded ID number: 1 position is subtracted from the value returned by REFind so the next CFSET on Lines 3–4 only removes the text up to the first position of the identity number, not including it.

The third and final step in stripping out the ID number is on Lines 5–6. It uses REReplaceNoCase to remove everything after (and including) the ".cfm" in the template name. By using a regular expression rather than just looking for ".cfm" we strip out any parameters that may be attached to the end of the URL for some reason.

These three operations create #variables.IID#, which is plugged into the CFQUERY on lines 7–18. *Note:* This query has the text field Items.Keywds in it. This contains item-specific search keywords for

MACROMEDIA

www.allaire.com/usergroups

use later in the template. If you already have a keywords field to aid your visitors in searching for products, then it may fit in nicely right here with no modification.

Lines 19–21 perform an important maintenance function. What happens if you delete a database record? Should you delete the corresponding entrance page? As these pages are now visible to search engines, they're also susceptible to linkrot (search engine-indexed pages that no longer exist).

If you remove an item from your database, Lines 19–21 sense this by evaluating the `RecordCount` and providing a referring link if necessary. I'm keeping things simple here. You can get as fancy as you like.

Lines 22–31 are necessary for my style of page content, but maybe not yours. This example uses the database

record's Items. Description field to provide the META Description content on Line 38. Since on my sites this field contains fully formatted HTML, it must be stripped to an unbroken string of bare text. The first step in this process uses the custom tag `CF_RENO`Tags on line 22 to remove HTML tags. The remainder of this code block uses the `Replace` function to remove carriage returns and line feeds.

Lines 34–38 add a title (important to search engines), which consists of the product name, plus the META Keywords and META Description tags. The remainder of Listing 3 is simple query output.

How Does It Affect the Existing Site?

It won't. These apparently static entrance pages work outside your normal site structure. Build the master template (i.e., `static.cfm`)

using your regular data-driven template as the model – menu links included – adding in only the search engine optimizations and the key field extraction code. Visitors may arrive via the entrance page, but as soon as they hit a link they'll be right back in your site's normal structure.

If you don't want your Web directory to contain hundreds of entrance pages, an easy solution presents itself: create the files in a subfolder off the regular template folder and adjust your links accordingly (using a BASEHREF statement, perhaps).

Will it be as simple as that? Maybe, but if you're like me, you pass more than one parameter via the URL. Your design will have to account for this. In my case, it is usually fairly simple: all I often need is to run a couple of extra key-lookup queries and create a session identifier.

Listing 1

```
<CFQUERY
  NAME="ItemDetail"
  DATASOURCE="#MyDSN#">
    SELECT
      Items.ItemID,
      Items.Name,
      Items.Description,
      Items.Price
    FROM Items
    WHERE Items.ItemID = (#url.IID#)
</CFQUERY>
<html><head><title>Product Item</title></head>
<body>
<CFOUTPUT QUERY="ItemDetail">
#ItemDetail.ItemID#, #ItemDetail.Name#<br>
#ItemDetail.Description#<p>
#ItemDetail.Price#
</CFOUTPUT>
</body></html>
```

Listing 2

```
<CFQUERY
  NAME="Maker"
  DATASOURCE="#MyDSN#">
  SELECT Items.ItemID
  FROM Items WHERE 0=0
  ORDER BY Items.ItemID ASC
</CFQUERY>

<CFLOOP QUERY="Maker">
  <CFSET DestFile="c:\myfolder\item_
    & Maker.ItemID & ".cfm">
  <CFFILE
    ACTION="Copy"
    SOURCE="c:\myfolder\static.cfm"
    DESTINATION="#DestFile#">
</CFLOOP>

<html><body>
<p>You just created
<CFOUTPUT>#Maker.RecordCount</CFOUTPUT> files:
<ol>
<CFLOOP QUERY="Maker">
<CFSET TheFile="item_" & Maker.ItemID & ".cfm">
<CFOUTPUT>
<li><A HREF="#TheFile#">#TheFile#</A></li>
</CFOUTPUT>
</CFLOOP>
</ol>
</body></html>
```

Listing 3

```
<CFSET variables.IDNum =
#REFind("[[:digit:]]", CGI.PATH_INFO)# - 1>
<CFSET variables.IID =
RemoveChars(CGI.PATH_INFO,1,variables.IDNum)>
<CFSET variables.IID =
ReplaceNoCase(variables.IID,".cfm.*","")>
<CFQUERY
NAME="ItemDetail"
DATASOURCE="#MyDSN#">
    SELECT
        Items.ItemID,
        Items.Name,
        Items.Description,
        Items.Price,
        Items.Keywds
    FROM Items
    WHERE Items.ItemID = (#variables.IID#)
</CFQUERY>
<CFIF ItemDetail.RecordCount LT 1>
    <CFLOCATION URL="index.cfm" ADDTOKEN="No">
</CFIF>
<CF_RENOtags INPUT="#ItemDetail.Description#">
<CFSET CRLF = Chr(13) & Chr(10)>
<CFSET LF = Chr(10)>
<CFSET CR = Chr(13)>
<CFSET variables.stripped =
    Replace(stripped_text, CRLF, " ", "ALL")>
<CFSET variables.stripped =
    Replace(variables.stripped, CR, " ", "ALL")>
<CFSET variables.stripped =
    Replace(variables.stripped, LF, " ", "ALL")>
<html><head>
<CFOUTPUT>
<title>#ItemDetail.Name#</title>
<meta name="keywords"
    content="#ItemDetail.Keywds#">
<meta name="description"
    content="#variables.stripped#">
</CFOUTPUT>
<body>
<CFOUTPUT QUERY="ItemDetail">
#ItemDetail.ItemID#, #ItemDetail.Name#<br>
#ItemDetail.Description#<p>
#ItemDetail.Price#
</CFOUTPUT>
</body></html>
```

CODE
LISTINGS

The code listings for this article can also be located at www.ColdFusionJournal.com

Note that creating entrance pages for every data-driven page on your site isn't necessarily the best idea. I have clients with hundreds of products in their stores, which translates to hundreds of submittable pages. Enough is enough. Besides, it's the product pages that contain the content visitors want in the first place. You'll have to assess each project individually with these issues in mind.

How Does It Change My Traffic?

Your pattern of site entry should change completely in a few months. You'll start to see visitors coming in from entrance pages with regularity – as people search for things and find your new engine-indexed pages in the results. You should also see more visitors – as you now have a much “bigger” site with more relevant, indexable content.

Text Quantity Range

Will this affect how you produce content? Possibly, depending on what your site is about. Just popping in META tags will give only limited results. Search engines generally

need more to work with on a page. Typically the text quantity should aim for the range of 200–600 words (even though the rules – such as they now exist – vary by engine and seem to change weekly).

A product detail page with just a photo and a product title isn't likely to do much for you. Put in a paragraph or two describing the product, its benefits, related products for sale, and so on.

Red-Flagged as an Engine-Spammer

A comprehensive discussion of best practices on this subject is beyond the scope of this article. One thing I will say: don't create 200 pages and immediately submit them. You'll get red-flagged as an engine-spammer if you try that. An excellent resource for learning more is at www.searchenginewatch.com.

Once you know how to submit within the rules, there remains the repetitive process of doing so. Here it makes the most sense to use a quality robot submitter. Personally I use WebPosition Gold's Scheduler and

Submission features. Particularly nice is the fact that I can add hundreds of pages to a submission task in a single step with just a few mouse clicks.

Consider creating a ColdFusion-powered robots.txt routine if you're serious about keeping yourself crawler-friendly. While the creation of referrer pages to replace obsolete content is pretty standard advice, and the method discussed here neatly handles this, it's a good idea to update robots.txt to exclude the obsolete copies of static.cfm, which now are merely linkrot-preventing referrers. Without too much extra effort you can now data-drive what was formerly a manual process and further enhance your page-indexing methodology.

Throws the Doors Wide Open

It's big, it's bad, and, as promised, it's just plain ugly. Nonetheless, it completely eliminates a major limitation found in your typical data-driven site, and throws your doors fully open to the “visible” World Wide Web.



ABOUT THE AUTHOR
Matt Robertson is president and chief designer at MSB Designs, Inc., a Web development firm specializing in ColdFusion-driven applications.

MATT@MYSECRETBASE.COM

DTN FINANCIAL SERVICES

www.finwin.com

VTML

by *Example*

How to successfully
extend the
ColdFusion Studio IDE

Part 2

In Part 1 of this series (**CFDJ**, Vol. 3, issue 6), I explained how to easily generate the necessary VTML code for providing Tag Inspector, Tag Help, and Tag Insight features inside CF Studio for your custom tags. (Part 1 is available online at www.sys-con.com/coldfusion/archives/.)

Part 2 focuses on building Tag Editor dialogs using VTML to provide real user interface-based dialogs for your custom tags in CF Studio. If you haven't read Part 1, take a quick look at it before continuing Part 2.

Building Tag Editor Dialogs with VTML

To build Tag Editor dialogs you'll have to leave the guided paths of visual VTML editing and go deep into the hand-coding of VTML. But it's not as complex as it may seem at first glance, so let's begin by analyzing what we already have in code by using a fictional custom tag called `<cf_myOwnCustomTag>`, which can take the following attributes: `<cf_myOwnCustomTag headline="Just a headline" status="Active" titleFont="Verdana" color="Red">`.

Listing 1 contains the completely finished VTML code for our fictional custom tag.

Named by `<TAG NAME="CF_MYOWNCUSTOMTAG"></TAG>`, the VTML file contains five sections (as outlined in Part 1), while the `<ATTRIBUTES>` section contains a list of `<ATTRIB>` tags that represent the attributes the custom tag can take and their types. (Part 1 provides a list of these types.)

The most interesting attribute type is Enumeration, which is a predefined list of certain values an attribute can take:

```
<ATTRIB NAME="STATUS" TYPE="ENUMERATED">
  <ATTRIBOPTION VALUE="Active"/>
  <ATTRIBOPTION VALUE="Inactive"/>
  <ATTRIBOPTION VALUE="Pending"/>
</ATTRIB>
```

Aside from the value, it's also possible to set a caption for each `<ATTRIBOPTION>` tag:



```
<ATTRIB NAME="STATUS" TYPE="ENUMERATED">
  <ATTRIBOPTION VALUE="Active" CAPTION="Active
connection"/>
  <ATTRIBOPTION VALUE="Inactive"
CAPTION="Inactive connection"/>
  <ATTRIBOPTION VALUE="Pending" CAPTION="Pending
request"/>
</ATTRIB>
```

This `<ATTRIBUTES>` section of the VTML file is responsible for the Tag Insight feature. The `<ATTRIBCATEGORIES>` section

logically groups attributes into different categories to better organize them in the Tag Inspector. By default, all attributes are in a single category named Misc, but you can group them as you like by simply adding further <ATTRIBGROUP> tags here:

```
<ATTRIBCATEGORIES>
  <ATTRIBGROUP NAME="Misc"
ELEMENTS="COLOR,TITLEFONT,HEADLINE,STATUS"/>
</ATTRIBCATEGORIES>
```

The last section generated by the Tag Definitions Editor is <TAGDESCRIPTION>, which provides your custom tag with a help file (this is an important and handy feature when you're distributing your custom tags). As mentioned in Part 1, by placing the HTML file inside the C:\Program Files\Allaire\ColdFusionStudio\Extensions\Docs\CFMLTags\directory with a proper name, you can activate the help file in Studio by pressing F1 over your tag.

```
<TAGDESCRIPTION HELPFILE="../../Docs/CFMLTags/cf_my-
OwnCustomTag.htm"/>
```

To build a tag editing dialog the Tag Definitions Editor needs the code for the user interface and how each control corresponds to an attribute. The <EDITORLAYOUT> and <TAGLAYOUT> sections are responsible for this. Inside the <EDITORLAYOUT> section you'll have to code the user interface by placing controls such as text boxes, drop-down lists, color pickers, and so on, onto the dialog. The <TAGLAYOUT> section describes how the Studio IDE should generate the tag code after you press "OK."

You should be concerned about how the user interface looks. I usually scribble a draft of how the dialog should look and what controls to use that correspond to the attributes. Only after finishing this do I start coding the dialog with VTML.

For our sample tag called <cf_myOwnCustomTag>, we have the following list of attributes: Color (a color value), Titlefont (a font name), Headline (just plain text), and Status (an enumeration of Active, Inactive, and Pending). This list suggests that our Tag Editor dialog should have a color picker, a font picker, a text box, and a drop-down list of the three predefined values for the Status attribute.

The VTML tags you'll need to know for designing the user interface of the dialog are <CONTAINER> and <CONTROL>. As their name suggests, a <CONTAINER> tag (like a panel) can contain <CONTROL> tags corresponding to input controls such as a text box or color picker. Let's start filling the <EDITORLAYOUT> section of our VTML file with a basic dialog for <cf_myOwnCustomTag>:

```
<EDITORLAYOUT HEIGHT="250" WIDTH="500">
  <CONTAINER NAME="Panel1" TYPE="Panel" WIDTH="480"
HEIGHT="200" CAPTION="Basic Information">
    <CONTROL NAME="lblHeadline" TYPE="label" CAP-
TION="Headline" DOWN="30" RIGHT="20"
        WIDTH="60" ALIGN="Right"/>
    <CONTROL NAME="txtHeadline" TYPE="TextBox"
ANCHOR="lblHeadline" CORNER="NE" WIDTH="350"/>
  </CONTAINER>
</EDITORLAYOUT>
```

This code generates a 500x250 pixel dialog containing one panel (titled Basic Information) that acts as a container for a label control (displaying the text "Headline") and a text-box

control named txtHeadline. As the attributes of the <CONTAINER> tag are more or less self-explaining (name, type, width, height, and caption), I'll take a closer look at the <CONTROL> tag. Think of the layout as a relative layout where you can position certain controls with hard-coded coordinates inside a panel (the attributes down and right starting at the upper-left corner of the container panel).

However, you can also position controls relative to each other by setting their anchor attributes to the name of the control to anchor at, as well as specifying the corner (here, NE for north-east, NW for northwest, SE for southeast, and SW for southwest) where the control should be placed, along with the attributes DOWN and RIGHT to assign an offset.

This relative positioning is helpful when you're moving certain controls on the dialog, because you have to move only one control and all anchored controls move too, retaining their alignment and relative position. See the Tag Inspector of the <CONTROL> tag for all possible values of its attributes.

By extending the above dialog to show all input controls of our <cf_myOwnCustomTag> sample, which should be a drop-down list, a font chooser, and a color picker, the result looks like that shown in Listing 2 and in Figure 1; see also Listing 1 for its VTML source.

The following types of input controls are possible with VTML:

- Label
- TextBox
- CheckBox
- RadioGroup
- DropDown
- ListBox
- TextArea
- FontPicker
- ColorPicker
- Image
- FileBrowser
- StyleTextBox
- StyleTextArea
- SQLTextArea
- ActiveX

For a complete reference description on how to use these control types, see the ColdFusion Studio help (the section titled "Customizing the Development Environment"). Describing

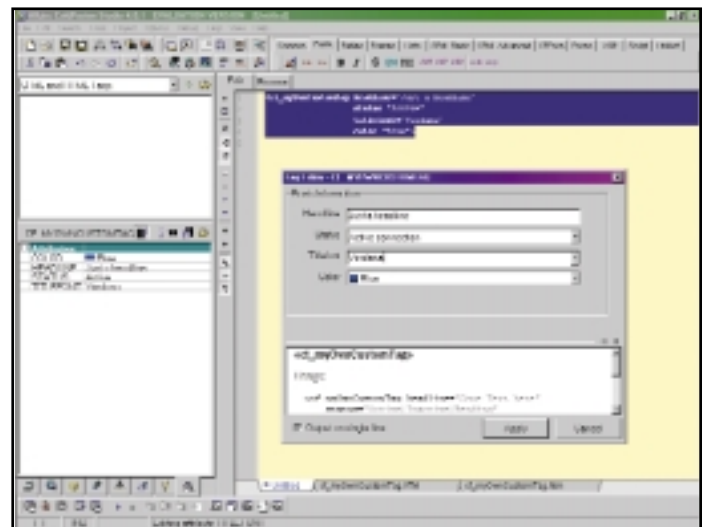


FIGURE 1: Tag Editor in action

how to use each type would be out of this article's scope; it's meant to be a tutorial, not a reference.

Having noticed that the <CONTROL> tags reside in a <CONTAINER> tag acting as a panel, you're surely interested in what other types of containers are possible:

- **Panel:** A general purpose panel container; can contain any controls
- **TabDialog:** A tab dialog container capable of containing one or more TabPage containers
- **TabPage:** A tab page that can hold a panel; only used inside a TabDialog container

As you see from this list, it's possible to design a tabbed dialog to split and organize many input controls into small chunks accessible via tabs. Listing 3 provides an example of a tabbed dialog.

Making the Dialogs Work

So far you've seen how to design the user interface of a Tag Editor inside the <EDITORLAYOUT> section. Now it's time to make your dialogs work; that is, how to assign certain input controls to certain attributes.

At first the Tag Editor must be aware of what attribute data is currently present in your template to prefill the

input controls. For this only a small change must be made in the <ATTRIBUTES> section: add the attribute CONTROL="theControlName" to each <ATTRIB> tag, while the names of each control should match the names given in the <CONTROL> tags:

```
<ATTRIB NAME="COLOR" TYPE="COLOR"
CONTROL="txtColor"/>
<ATTRIB NAME="TITLEFONT" TYPE="FONT"
CONTROL="txtTitlefont"/>
...
```

Now, when you're clicking on a <cf_myOwnCustomTag> tag that's holding attributes, you'll see their values successfully being populated into the dialog. But there's still one special situation: imagine that your custom tag is a paired custom tag like <cf_someTag>Some content here</cf_someTag> that's holding content, and you wish to populate this content into your editor dialog. No problem; just assign the content to a control of your dialog using the following special <ATTRIB>-tag: <ATTRIB NAME="\$\$TAG-BODY" CONTROL="txtNameOfControl"/>.

The final step in building a Tag Editor is to assign how the data entered into the dialog's input controls is populated back into the template. To do this you'll have to touch the final <TAGLAYOT> section of our VTML file, which defines a tem-

plate of how the finished tag (here <cf_myOwnCustomTag>) will be filled with content from the dialog.

Here we'll cover a little bit of WIZML code to define some basic logic of how to generate the tag. For a deeper look into WIZML, I recommend Part 3, which will cover WIZML in greater detail with regard to building custom wizards.

To define what your Tag Editor dialog will output, fill some content inside the <TAGLAYOUT> section. Whatever content there is between <TAGLAYOUT> and its closing counterpart </TAGLAYOUT> will be put back into the template. To output dynamic data depending on what the user has entered in the dialog, you can use WIZML variables, which are written using the following convention: \$\$ {name-OfVariable}.

A simple version of our <TAGLAYOUT> section would look like this:

```
<TAGLAYOUT>
  <cf_myOwnCustomTag
    headline="$$ {txtHeadline}" sta-
    tus="$$ {txtStatus}"
    titleFont="$$ {txtTitlefont}"
    color="$$ {txtColor}">
  </TAGLAYOUT>
```

Try it out and you'll see your Tag Editor dialog working (don't forget to

CFXHOSTING
www.cfxhosting.com

restart CF Studio to let all changes take effect). But you can go even further by using additional logic inside the <TAG-LAYOUT> section.

For example, you can determine whether the user has checked the “Output on single line” checkbox inside the Tag Dialog, and whether the user has set the option of lowercase or uppercase tag code. You can even check for any unknown attributes assigned to the tag code and keep the attributes that are not covered by your dialog untouched.

For these three cases the following special WIZML variables are passed to the dialog:

- **OPTIONLinearLayout**. Returns “true” or “false.” Specifies whether the tag should be generated with its attributes in a single line or indented vertically.
- **OPTIONLowerCaseTags**. Returns “true” or “false.” Specifies whether the tag should be generated using lowercase.
- **TAGDATAUnknownAttributes**. A string containing all attributes that were contained in the edited tag string, but not recognized by the editor dialog.

You can check for these variables with an easy WIZML tag: just ask if they’re true using <WIZIF><WIZELSE></WIZIF>, much

like the <CFIF><CFELSE></CFIF> construct that you know (see Listing 1).

Here we simply check for a linear layout and assign either a zero-length string to the variable Spacer or a line break and some spaces. The Spacer is then used behind every attribute, which is output either as lowercase or uppercase depending on the OPTIONLowerCaseTags variable. The template would yield the following Tag Layout:

- **Linear Layout:**

```
<cf_myOwnCustomTag headline="Just a
headline" status="Active"
titleFont="Verdana" color="Red">
```

- **Nonlinear Layout:**

```
<cf_myOwnCustomTag headline="Just a
headline"
    status="Active"
    titleFont="Verdana"
    color="Red">
```

Finally, you may want to incorporate any unknown attributes of the tag such as <cf_myOwnCustomTag headline="Just a headline" status="Active" titleFont="Verdana" color="Red" someUnknownThing=42 backgroundColor="Blue">. To do this, the editor should check for unknown attributes and retain them instead of losing them during the editing process. All unhandled attributes, if there are any, are available in

the variable TAGDATAUnknownAttributes, which should simply be appended at the end of the tag:

```
<cf_myOwnCustomTag
headline="$$ {txtHeadline} "$$ {Spacer}
status="$$ {txtStatus} "$$ {Spacer}
titleFont="$$ {txtTitleFont} "$$ {Spacer}
} color="$$ {txtColor} "<WIZIF
TAGDATAUnknownAttributes NEQ
">$$ {Spacer}
$$ {TAGDATAUnknownAttributes}</WIZIF>>
```

For a complete listing of the VTML file see Listing 1.

Finally, here’s a tip you’re surely looking for: instead of restarting CF Studio every time to see your changes, you can press CTRL-SHIFT-ALT-C to reload all VTML files into memory. This saves you a lot of time, and you can blame me for telling you the tip at the end of this article instead of the beginning.

I’d like to sum up that coding VTML is easier than many developers might expect, and doing this little extra work when distributing custom tags pays off in a great deal of added convenience. Macromedia has built well-developed extension possibilities inside the Studio IDE that should be used wherever possible.



mail@christian-schneider.de

PAPERTHIN

www.paperthin.com

```
<CONTROL NAME="lblTitlefont" TYPE="label"
```

```
<TAGDESCRIPTION  
HELPPFILE=" ../.. /Docs/CFMLTags/cf_myOwnCustomTag.htm" />
```

```
Listing 4: Sample help file
<!--- /// cf_myOwnCustomTag.htm /// --->
<!-- Place into
#StudioDirectory#/Extensions/Docs/CFMLTags/ --->
```

```
<html>
<head>
  <title>cf_myOwnCustomTag</title>
</head>
```

```
<body>

<font face="Arial, Helvetica">
<b>&lt;cf myOwnCustomTag&gt;</b>
```

<p/>
Usage:


```
<pre>
    &lt;tcf_myOwnCustomTag headline=<font color="Gray">"Some
Text here"</font>
    status=<font
color="Gray">"Active|Inactive|Pending"</font>
    titleFont=<font color="Gray">"Verdana, Arial"</font>
color=<font color="Gray">"Blue"</font>&gt;
</pre>
```

<p/>
Description:

Some sample Custom Tag that does exactly nothing...

```
<p/>
Author:<br>
<a href="mailto:mail@Christian-Schneider.de">Christian
Schneider</a>
```

```
</font>

</body>
</html>
```

```
<EDITORLAYOUT HEIGHT="220" WIDTH="500">
```

```
<EDITORLAYOUT HEIGHT="220" WIDTH="500">
  <CONTAINER NAME="Panel1" TYPE="Panel"
    WIDTH="480" HEIGHT="190"
    CAPTION="Basic Information">
```

```
<CONTROL NAME="lblHeadline" TYPE="label"
CAPTION="Headline" DOWN="30" RIGHT="20"
WIDTH="60" ALIGN="Right"/>
```

```
<CONTROL NAME="txtHeadline" TYPE="TextBox"
ANCHOR="lblHeadline" CORNER="NE"
RIGHT="10" WIDTH="350"/>
```

```
<CONTROL NAME="lblStatus" TYPE="label"
CAPTION="Status" ANCHOR="lblHeadline"
CORNER="SW" DOWN="15"
WIDTH="lblHeadline" ALIGN="Right"/>
```

```
<CONTROL NAME="txtStatus" TYPE="DropDown"
ANCHOR="lblStatus" CORNER="NE" RIGHT="10"
WIDTH="txtHeadline">
  <ITEM VALUE="Active"
  CAPTION="Active connection"
  SELECTED="Yes" />
```


CFCCONTENT with Images – from Web Bugs to Banner Servers

Use CFCCONTENT to secretly track readers of your e-mails

BY ERON COHEN AND
MICHAEL SMITH



Many of us have used the CFCCONTENT tag that comes with ColdFusion to serve up files to browsers, but very few ColdFusion developers are aware that the CFCCONTENT tag can be used in conjunction with the HTML tag to serve up graphics, such as JPEGs and GIFs.

In this case, the why of doing this is perhaps just as interesting as the how.

It turns out that using this technique is perfect for use with creating an advertising banner server, controlling access to graphic files, or – on the more sinister side – creating “Web bugs.”

If you don’t recall, a Web bug is a graphic (usually an invisible 1 pixel shim) that is embedded in an HTML e-mail message or Word document that tips off its creator when and who is reading without readers even knowing their access was logged.

If you’ve never used the CFCCONTENT tag before, it’s an excellent tool to become familiar with. In layman’s terms, CFCCONTENT tells a Web browser that it’s about to receive a non-HTML file, and then sends it to the browser. It does this by allowing you to specify a MIME type and a filename to send to the browser. So a ColdFusion template name can be put in place of a JPEG or GIF file, like so:

```
<IMG  
SRC="http://www.myserver.com/  
images/send_graphic.cfm">
```

The ColdFusion template “send_graphic.cfm” will contain a CFCCONTENT tag that specifies “image/gif” for the MIME type and is pointed at the name of an actual .GIF file. The kicker is that you can also include code that logs the access to the file to

a database table or does just about anything else ColdFusion can do. This is where privacy advocates get upset.

If logging access isn’t bad enough, your send_graphic.cfm file could also use CFCOOKIE to set a cookie on the viewer’s machine. In turn you could later check for the cookie when the user visits your Web site. If the cookie is there, then you could infer that the person viewed the e-mail, and then decided to visit the Web site. And that’s just the beginning of the worst of the possibilities.

A more common use of CFCCONTENT in this way is to serve graphics for a banner server-type application. The logistics and possibilities are about the same as for a Web bug. The only major difference is that even less savvy Internet users

are aware that someone is probably logging each and every time the graphic is viewed. In the same spirit as with security flaws in applications, the authors of this article feel that it’s better to make as many people as possible aware of these techniques and then let them decide how to use the information. This is, after all, real-world stuff that is regularly used by Web programmers at Microsoft, Barnes & Noble, and other major direct e-mailers. So in that spirit, let’s look at some example code in Listing 1.

This simple example uses a custom tag called <CFX_NSlookup> free from Lewis Sellar’s Intrafoundation (www.intrafoundation.com/freeware.html), and is used to get the user’s domain name from the IP Address. We will use CFTRY tags to catch any possible logging failures and just send the image anyway. Finally, we use CFSETTING to suppress any extra white space that might be generated by our code formatting. To avoid problems with Web browsers, the only output we need or want comes from CFCCONTENT.

So there you have it. When the Web browser or e-mail client loads the HTML containing , their IP address, the date, and possibly their domain name are logged in a database and the graphic sent, and the uneducated viewer is none the wiser.

MIME Type

A MIME type is a description of what kind of data the browser is receiving. It helps the browser decide how to display the data. For example, a type image/gif will be displayed as an image while type text/HTML is a regular HTML page. MIME was originally intended for sending binary files via e-mail, thus the abbreviation stands for Multipurpose Internet Mail Extension.

Tip: In Netscape the menu option View, Page Info will display all parts of a Web page together with their MIME types.

ABOUT THE AUTHORS

Eron Cohen has worked with ColdFusion since 1996. Currently he is a freelance Web developer and trainer.

Michael Smith is president of TeraTech (www.teratech.com/). He runs the MDCFUG and recently organized the two-day, Washington, DC-based CFUN-2k conference that attracted more than 750 participants.

1. The actual location of the file does not have to be in the accessible Web path. (This is good if your users are paying for the files, such as graphics libraries, PDF reports, or install EXEs.)
2. You can include code in the template that can log access to the file.
3. The file to be displayed can be dynamically selected based on other criteria; for instance, random image display, and graphic size based on connection speed.

Resources

1. *Web bug FAQ:* www.eff.org/pub/Privacy/Profiling_cookies_webbugs/web_bug.html
2. *General privacy site:* www.privacyfoundation.org/



MICHAEL@TERATECH.COM

```

send_graphic.cfm:
<CFTRY>
<CFSETTING enablecfoutputonly="yes">

<CFPARAM name="nslookup" default="unknown">

<CFLOCK NAME="NSLOOKUP" TIMEOUT="30">
<CFX_NSlookup IHOST="#CGI.remote_addr#">
</CFLOCK>

<CFQUERY NAME="Log_Image_Views"
DATASOURCE="#application.dsn#">

    INSERT INTO log_image_views
    (logo_view_IP,logo_view_date,logo_view_domain )
    values
    (' #CGI.remote_addr#',#createodbcdatetime( "now()" )#),
    '#NSLookup#')

</CFQUERY>

<!---
////////////////////////////////////////
<!--- // Force the browser to download the image file.
//---->
<!---
////////////////////////////////////////

<CFCONTENT TYPE="image/gif"
FILE="c:\images\invisible_pixel.gif">

<CFSETTING enablecfoutputonly="no">

<CFCATCH TYPE="any">
<CFCONTENT TYPE="image/gif" FILE="
c:\images\invisible_pixel.gif">
</CFCATCH>

</CFTRY>

```

COD
LISTING

DDDDDDDDDDDDDDDDDDDDDDDDDDDDDD

The code listing for
this article can also be located at
www.ColdFusionJournal.com

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



BY
BEN
FORTA

Be Extremely Graphic

CF5 introduces a true graphing engine

More ColdFusion applications are running on intranets and extranets than just about anywhere else. This is not surprising – as ColdFusion makes data access and reporting so simple it is a natural fit for applications in these environments. And, as such, data reporting is one area that ColdFusion 5 addresses head-on with the introduction of a true graphing engine.

ColdFusion Graphing

ColdFusion developers have always wanted a way to graph data. And so back in ColdFusion 2 days, a series of Java applets were introduced to simplify the creation of common business graphs (pie charts, bar charts, etc.). These worked, but they had some serious limitations:

- They were Java applets, and as such were not supported by all browsers.
- They suffered from long download times and poor performance.
- They could not be printed easily.

ColdFusion 5 solves the graphing problem in a very different way. Included with ColdFusion 5 is a version of Macromedia Generator – a high-performance, scalable, and proven graphics generation engine. Generator is designed to create dynamic images in a variety of formats using templates that are processed in real time.

And Generator does not need any client-side software because the file formats it creates are standard formats like JPEG (which are supported by all browsers automatically) – the only exception to this is Flash, the use of which is optional (although with over 96% of all browsers supporting Flash it is pretty standard too).

Now before you ask, no, you cannot (yet) extend or enhance the graphing features exposed to ColdFusion. Yes, Generator can do much more than has been made available at this time, and, at some point in the future, there likely will be a way to better utilize more of Generator within ColdFusion. But for now you have to use the charts and graphs explicitly defined in ColdFusion – and, fortunately, they'll do most of what you need, as you are about to see.

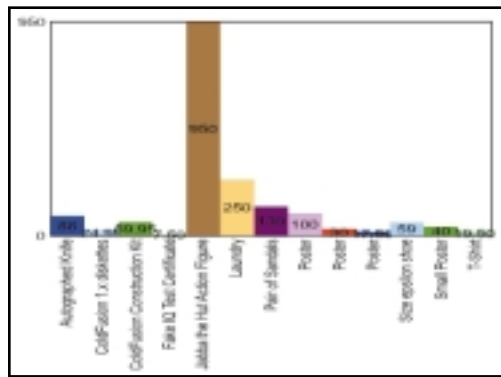


FIGURE 1: Minimal options and attributes

The <CFGRAPH> Tag

As you have come to expect of ColdFusion, the way you create graphs is by using a tag. And as you've also come to expect, that tag is intuitively named <CFGRAPH>. This seemingly simple tag hides all the complexity of creating all sorts of business graphs and charts with all sorts of options and features.

But instead of telling you about <CFGRAPH>, let's look at an example. Figure 1 shows a simple bar chart showing the relative prices of items for sale. (All of the data in these examples are taken from the databases and example applications in my new ColdFusion 5 Web Application Construction Kit.)

The code I used to create the graph is in Listing 1.

First I retrieved data with a basic <CFQUERY> – the query name is Merchandise, and the two columns retrieved are MerchName and MerchPrice. To create the graph, all I had to do was pass that data to <CFGRAPH> – QUERY takes the name of the query to be used, TYPE specifies the graph type (BAR, PIE, AREA, etc.), ITEMCOLUMN takes the name of the query column to use as the item name, and VAL-UECOLUMN takes the name of the column containing the value to use.

Simple as that, and ColdFusion does the rest.

Note: ITEMCOLUMN and VAL-UECOLUMN take column names, not values, so don't place pound signs around the column names.

So what did that tag actually do? In Listing 2 is the code that was embedded in the generated output. I know it looks a little complex, but that is because it is invoking the Flash player so as to display the above graph in Flash format. In other words, the four lines of ColdFusion code in Listing 1 generated and embedded Flash content – without you having to learn anything about Flash.

And, no, you are not limited to generating Flash-based graphs. <CFGRAPH> can also generate JPEG and PNG images (not GIF though, sorry). But as I did not specify a format in my <CFGRAPH> tag, the default format was used, and the default is Flash.

Lots of Options

The previous example used a minimal set of options and attributes. This next example, in Listing 3, is quite the other extreme, using all sorts of attributes (far more than you'd typically use, but it helps make a point).

INTERNATIONAL WIRELESS BUSINESS & TECHNOLOGY CONFERENCE & EXPO

SHAPING WIRELESS STRATEGY FOR THE ENTERPRISE

wireless **EDGE**
conference & expo

Santa Clara Convention Center, Santa Clara, CA

CONFERENCE & EXPO

May 7-9, 2002

The Only Wireless Event
Backed by
The Power of

**SYS-CON
MEDIA**

wireless



Plan to Attend the 4-Day Conference **SHAPE YOUR WIRELESS STRATEGY SAVE THE DATES!**

WirelessEdge will provide the depth and breadth of education and product resources to allow companies to shape and implement their wireless strategy. Developers, i-technology professionals and IT/IS management will eagerly attend the WirelessEdge program to plan short and long-term wireless strategies to build competitive advantage. You will network with peers and explore cross-industry applications.

Plan to Exhibit

PROVIDE THE RESOURCES TO IMPLEMENT WIRELESS STRATEGY RESERVE YOUR EXHIBIT SPACE NOW!

The Expo will provide access to the most qualified audience of potential buyers and specifiers to over 200 exhibitors.

The Conference will motivate and educate. The Expo is where attendees will want to turn ideas into reality. Be ready to offer solutions.

Plan to Sponsor

THE LARGEST WEST COAST WIRELESS CONFERENCE & EXPO IN 2002 MANY SPONSORSHIPS ARE EXCLUSIVE

Rise above the noise. Establish your company as a market leader. Deliver your message with the marketing support of SYS-CON Events and SYS-CON Media.

Conference Tracks

Track One: Development

WAP
i-Mode
Bluetooth / 802.11
Short Messaging
Interactive Gaming
GPS / Location-Based
Wireless Java
XML & Wireless Technologies

Track Two: Connectivity

Smart Cards
Wireless LANs
incl. Bluetooth
UMTS/3G Networks
Satellite Broadband

Track Three: Wireless Apps

Education
Health Care
Entertainment
Transport
Financial Services
Supply Chain Management

Track Four: Hardware

Cell Phones/
WorldPhones
PDAs
Headphones/
Keyboards /
Peripherals
Transmitters/
Base Stations
Tablets

Track Five: Business Futures

Wireless in
Vertical Industries
The WWW
Unwired
Management
From 3W to 4W:
Issues and Trends
"Always-On"
Management
Exploiting the
Bandwidth Edge
Unplugged
Valueware
Wireless Sales &
Marketing

Call
201 802-3069
to reserve
your space
today!

Owned and Produced by

**SYS-CON
EVENTS**

135 Chestnut Ridge Road
Montvale, NJ 07645
(201) 802-3069

Fax: (201) 782-9051
visit

WWW.WIRELESSEGE2002.COM
for more information

NEITHER SYS-CON MEDIA NOR SYS-CON EVENTS, INC., ARE SPONSORS OF, OR ARE AFFILIATED IN ANY WAY WITH, CAMELOT COMMUNICATIONS, INC., OR ANY EVENTS PRODUCED BY CAMELOT COMMUNICATIONS, INC.

JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON PUBLICATIONS AND SYS-CON EVENTS ARE INDEPENDENT OF SUN MICROSYSTEMS.

MEDIA SPONSORS:

**SYS-CON
MEDIA**

**JAVA
DEVELOPERS
JOURNAL**

**XML
JOURNAL**

wireless

COLD FUSION

**UNIX
WEEK**

WebSphere

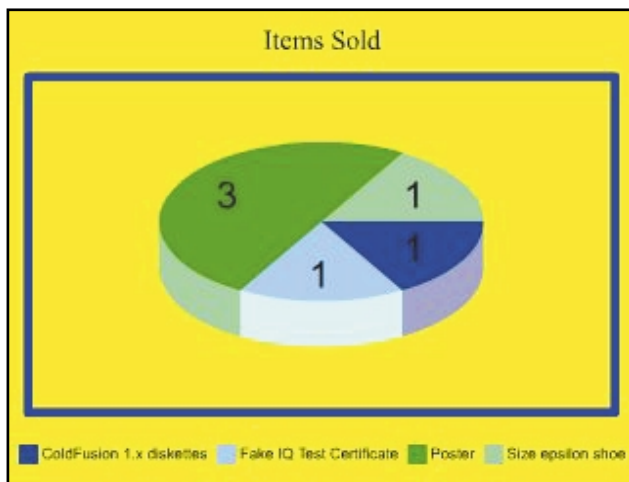
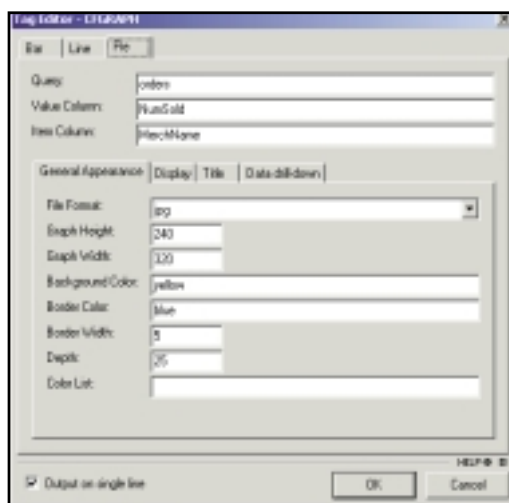


FIGURE 2: Explicit attributes

Again, I started with a `<CF-QUERY>`, this time retrieving recent items sold and a count of each. Then comes the `<CFGRAPH>` tag. The type is specified as `PIE`, the query name is provided; the format is set to `"jpg"` (otherwise the default of Flash would have been used); and explicit height and widths are specified along with background and border colors, border size, and 3D depth. And finally, label font and size, title text and font, and legend position and font are specified. The end result is seen in Figure 2.

FIGURE 3: `<CFGRAPH>` Tag Editor

As you can see, `<CFGRAPH>` is very flexible and highly configurable. And the Tag Editor seen in Figure 3 (available for ColdFusion Studio 4.5x as well as in ColdFusion Studio 5 when it is available) makes using these attributes a breeze.

Data Drill-Down

One other very important feature that I must mention is the ability to generate graphs that support data drill-down – that is, click on a pie slice or chart bar and go to a URL that provides additional (or more detailed) information.

`<CFGRAPH>` does not do this automatically, but it does provide a simple mechanism for associating URLs with graph components. Using this you can create your own drill-down interfaces by simply passing the URL of the next chart or graph to go to (see Listing 4).

This time the query retrieves a list of directors, and what they have been paid (converted into thousands by a division in the `SELECT` statement itself). The `<CFGRAPH>` tag is similar to the ones seen previously, although this one provides explicit colors (instead of using the defaults).

The important changes here are the last two attributes. `URL` takes the URL to go to if you click on a pie slice. But if the same URL is used for all slices, how would you know which slice was clicked? That's where the `URL-COLUMN` attribute comes in to play. It takes the name of a column, the value of which is appended to the URL specified in URL.

Here `"details.cfm?- Name="` is the URL, and `Name` (whatever value is in column `Name`) is the `URL-COLUMN`. So, if the name were "Ben Forta" the generated URL for that slice would be `"details.cfm?- Name=Ben+Forta"`. In other words, a unique URL is created for each slice by combining the fixed URL

and a dynamic query column (see Figure 4).

Data drill-down is a valuable `<CFGRAPH>` feature, but it is important to note that it's available only when using Flash as the `FORMAT` (and not with `JPG` or `PNG`).

For Even Greater Control

You might have been wondering why `<CFGRAPH>` has a matching `</CFGRAPH>` tag. Well, this is why – `<CFGRAPH>` has a child tag named `<CFGRAPHDATA>` that can be used to explicitly populate graphs with data (without using the `QUERY` attribute). For example, the following creates a pie chart containing four pie slices:

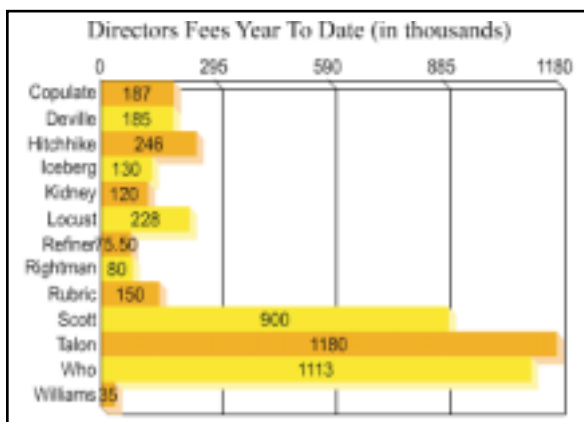
```
<!-- Create graph -->
<CFGRAPH TYPE="pie">
  <!-- Pie slices -->
  <CFGRAPHDATA ITEM="G"
    VALUE="7">
  <CFGRAPHDATA ITEM="PG"
    VALUE="28">
  <CFGRAPHDATA ITEM="PG13"
    VALUE="16">
  <CFGRAPHDATA ITEM="R"
    VALUE="31">
</CFGRAPH>
```

Of course, you can also use `<CFGRAPHDATA>` to populate a graph with query data – to do this simply use a `<CFLOOP>` to loop through the query results and then call `<CFGRAPHDATA>` for each row like this:

```
<!-- Create graph -->
<CFGRAPH TYPE="bar">
  <!-- Loop through data -->
  <CFLOOP QUERY="Directors">
    <!-- Add item to graph -->
    <CFGRAPHDATA
      ITEM="#Name#" VALUE="#Paid#">
  </CFLOOP>
</CFGRAPH>
```

Why would you ever want to do this? There are three primary reasons:

1. `<CFGRAPHDATA>` takes an optional `URL` attribute that you can use to specify a totally unique URL for each data point (as opposed to using a single URL with just a different query string).
2. `<CFGRAPHDATA>` takes an optional `COLOR` attribute that you can use for greater color control.
3. By passing data manually to graphs you can use CFML formatting functions and other programmatic functions to manipulate the data as needed.



You don't have to use <CFGRAPHDATA> if you don't want to, but it's nice to know it's there when you need it.

Summary

<CFGRAPH> is one of the most exciting new features in ColdFusion 5, and one that is built on top of a proven and scalable product - Macromedia Generator. With multiple graph types, three output formats, and dozens of configuration options, <CFGRAPH> proves once again that Macromedia and the ColdFusion team know what developers want, and are deeply committed to providing it.

ABOUT THE AUTHOR
Ben Forta is Macromedia's evangelist for the ColdFusion product line. His new book, ColdFusion 5 Web Application Construction Kit, is now shipping.

BEN@FORTA.COM

Listing 1

```
<!-- Retrieve merchandise list -->
<CFQUERY DATASOURCE="ows" NAME="Merchandise">
    SELECT MerchName, MerchPrice
    FROM Merchandise
    ORDER BY MerchName
</CFQUERY>

<!-- Graph prices -->
<CFGRAPH QUERY="Merchandise"
    TYPE="BAR"
    ITEMCOLUMN="MerchName"
    VALUECOLUMN="MerchPrice">
</CFGRAPH>
```

Listing 2

```
BACKGROUNDcolor="yellow"
bordercolor="blue"
borderwidth=5"
depth=25"
valuecolumn="NumSold"
itemcolumn="MerchName"
showvalueLabel="Yes"
valueLabelFont="Arial"
valueLabelSize="20"
valueLocation="INSIDE"
title="Items Sold"
titleFont="Times"
showLegend="below"
legendFont="Arial">
```

Listing 4

```
<!--- Get directors and amount paid to each --->
<CFQUERY DATASOURCE="ows" NAME="Directors">
SELECT LastName+', '+FirstName AS Name,
        SUM(Salary)/1000 AS Paid
FROM Directors, FilmsDirectors
WHERE Directors.DirectorID=FilmsDirectors.DirectorID
GROUP BY LastName+', '+FirstName
ORDER BY LastName+', '+FirstName
</CFQUERY>

<CFGRAPH TYPE="HORIZONTALBAR"
        QUERY="Directors"
        VALUECOLUMN="Paid"
        ITEMCOLUMN="Name"
        TITLE="Directors Fees Year To Date (in thou-
sands)"
        TITLEFONT="Times"
        FILEFORMAT="Flash"
        DEPTH="5"
        COLORLIST="orange,yellow"
        GRIDLINES="3"
        URL="details.cfm?Name="
        URLCOLUMN="Name">

</CFGRAPH>
```

The code listing for
this article can also be located at
www.ColdFusionJournal.com

Why It's Wrong to Use Application.dsn in Your Templates

...and what to do about it

BY
CHARLES
AREHART



You've probably seen the use of a variable called "application.dsn" (or "application.datasource") in code. Perhaps you've even been taught to use the method in a class.

Maybe you've even been doing it in your own code. I'm talking about setting this variable in the application.cfm file to hold the name of your data source for a given application.

It seems so innocuous, and it seems to provide ways to make your code easier to maintain (change the dsn variable once in the application.cfm, and all the templates that use it under control of that application.cfm get the benefit of the change).

The Problem

However, there's a problem and it can be a nasty one. This issue has to do with the locking of (or failure to lock) shared-scope variables such as this one, and the fact that rarely does any discussion of this approach include the consequences of using shared variable scopes.



There have been other articles in **CFDJ** on the subject of shared variable locking, as well as Macromedia Knowledge Base articles.

Maybe you didn't make the connection between them and this issue. This article puts the problem in perspective and offers some explanations of how to understand and resolve it. The ultimate solution involves using a "request"-scoped variable instead of an "application"-scoped one.

If you're not familiar with shared variable scopes, or are fuzzy about locking issues, or perhaps have never understood what "request" scope variables are about, this article should help you.

If you do understand these things, change your references to application.dsn to request.dsn, and if you find any locks around code that was using such an application.dsn variable, you need to consider whether those should stay as well. I'll also offer insights into how to find and fix such locking references.

Some Background

You may see code doing this in application.cfm:

```
<CFSET application.dsn = "whatever">
```

which declares the variable as "global," in essence, and can therefore be used later in all other templates as in:

```
<CFQUERY datasource="#application.dsn#" ... >
```

to refer to that data source name. The upside to this is that if the data source name needs to be changed (from "whatever" to "whatever_test", for instance), you can simply modify the application.cfm to point to the new name, and all templates under its control get the benefit of the change.

It's a good plan, but the use of an application-scoped variable is flawed. It opens you to potentially troublesome locking issues (for more on that, including a good explanation of why it's a problem, see "ColdFusion Locking Best Practices" at www.allaire.com/Handlers/index.cfm?ID=20370&Method=Full). More important, you can have the intended benefit with an equally useful and less troublesome way.

The Solution

For reasons I'll explain in a moment if it's still unclear, I'm suggesting that you stop using the application scope to hold the data source name. However, instead of dropping "application." from the variable name, I'm suggesting that you use the "request" scope. In other words, do the following (in your application.cfm):

```
<CFSET request.dsn = "whatever">
```

and then in all your templates do:

```
<CFQUERY  
datasource="#request.dsn#" ... >
```

If it's not clear why this is useful, or how it works, then there may be confusion about:

- How the application.cfm works like a CFINCLUDE (and how we could, but won't, use a local variable called "dsn")
- What the request scope is about (and why it's better to use "request.dsn")

Let me explain both. Though it would seem that many understand the first point, I find they often don't and it's fundamental to the rest of the discussion. The use of request scoped variables is also poorly understood by many. The end result is that you'll no longer have to deal with locking issues with regard to this variable.

How Application.cfm Works Like a CFINCLUDE

And How We Could, but Won't, Use a Local Variable Called "dsn"

Nearly every CF developer knows that whenever a CF template is run, CF first tries to execute any application.cfm that exists in the same directory (or its parent directory, or its grandparent, and so on).

What may not be obvious is that CF actually runs the application.cfm like a CFINCLUDE, which means that any variables set there, including "local" variables (such as `<CFSET firstname="bob">`), are then available to the template that was originally being run.

So let's say we have a template that does the following:

```
<CFOUTPUT>Hello
#firstname#</CFOUTPUT>
```

If it did this and nothing else, you might expect it to fail since you can't refer to variables that don't exist, and it's only outputting the variable, not creating it. But if `firstname` was set in application.cfm (assuming this template is in a directory controlled by that application.cfm), it can indeed refer to the variable.

Knowing that, you may wonder why the folks who promoted this solution of setting the dsn variable to the application scope even bothered. They could just as easily have said:

```
<CFSET dsn = "whatever">
```

and then in all their templates do:

```
<CFQUERY datasource="#dsn#" ...>
```

It would work. In fact, there's no need to be using the application scope to pass the variable to all templates (make it global), because any variables set in the application.cfm "trickle down" to all templates, in effect making them global. There are certainly good uses of application scoped variables, but this isn't one of them.

The simple example of setting a variable called "dsn" (or what could be formally specified as `variables.dsn`, which is the same thing) to hold the name of the data source would work, and would trickle down to all the templates. It's effectively "global," at least for the life of that template, and it's reset at the execution of each template by being executed in the application.cfm each time.

I've recommended that you use "request.dsn" rather than "dsn" or "variables.dsn". Why? And what is the request scope, anyway?

What the Request Scope Is About And Why It's Better to Use "Request.dsn"

If you understand that setting a local variable called "dsn" will work, can you think of any situation in which your code may expect to have access to that variable but won't? Think hard. Okay, custom tags. Yep, most know that custom tags have their own local variable scope. So a local variable set in the caller (or in the application.cfm) won't be available to the custom tag.

When we were setting the dsn variable to an application scope, it was available in the custom tag (as are all shared scopes and also form, url, cgi, and other variables that are passed to the calling template).

By changing the application.dsn variable to dsn (or `variables.dsn`, same thing) while it's available in all templates under control of the application.cfm, it's not available to any custom tags called by those templates. That's why we need to use the request scope instead.

Its sole purpose, poorly understood though it is, is to create local variables in a program that are available within custom tags called by that program (and vice versa).

That's it. Nothing more.

Many confuse the request scope with some sort of persistence or shared nature (and the fact that there's a different kind of "request" scope in other languages like ASP and JSP only confuses matters further). It's best to think of it as nothing more than a scope that allows local variables to be seen in a custom tag called by the template setting the request scope variable. And since our request.dsn variable is set in the application.cfm, it trickles down to the template being executed and is therefore also available to any custom tags we call as well.

That's why you should use the request scope rather than a local scope.

Remediation of Application.dsn Misuse

One last thought: you may not be able to blindly do a global search and replace application.dsn with request.dsn wherever it occurs. Depending on the savvy of the coder, the use of application.dsn may have at least two wrinkles that require more care than just replacing application.dsn with request.dsn:

- You may be testing for the existence of application.dsn before setting it in the application.cfm, in which case you need to set the request.dsn outside that test since it will never "already exist."
- You may have CFLOCKS surrounding CFQUERYS, or CFLOCKS that are moving the application.dsn to a local variable. You wouldn't want to just rename the scope in those instances.

Each of these cases requires a little more care to resolve. I provide further insights into solving these problems on my Web site, www.systemmanage.com/cff/application_dsn_bad.cfm.

I hope this article not only helps prevent problems with application.dsn, specifically, but also increases your understanding of request scope variables and application.cfm processing in general.



ABOUT THE AUTHOR

Charles Arehart is an Allaire certified trainer/developer and CTO of SystemeManage, an Allaire partner. He contributes to several CF resources, provides on-site coaching and consultation, and is a frequent speaker at user groups throughout the country.

CAREHART@SYSTEMMANAGE.COM

Ask the Training Staff

BY
BRUCE
VAN HORN



A source for your CF-related questions

We have some more good questions to tackle this month. We'll look at Client variables, a pesky CFMAIL problem, and ways to delete Session variables.

Q: *Can you give me a brief explanation of the "Client" variable scope?*

A: The Client variable scope is very similar to the Session scope in that you use it to keep track of certain pieces of data for individual users of your site ("clients"). The biggest difference is that Client variables are not stored in memory as are Session variables.

Client variables are stored physically, either on your server in a database, on your server in the registry, or back in the client's browser as cookies. Because they're stored physically, they don't expire as quickly or get lost if the server is restarted (as Session variables do). This allows you to keep information for individual users and have it available to them when they come back.

A good example would be preference settings for each user. Rather than have users log back in, or reset their preferences every time they come back to your site, you could store them in client variables so that information is available when the user returns.

Like Session variables, Client variables are stored and retrieved using two cookies (CFID and CFTOKEN) as identification. Also like Session variables, you can't use Client variables unless you enable CLIENTMANAGEMENT in your CFAPPLICATION tag (see Listing 1).

Before using Client variables, decide where you want to store them. By default, CF stores all Client variables in the Registry on your server. Never do this! Another choice is to have CF send them back to the user as cookies. This also isn't great because your user can delete or modify them, plus they all get re-sent with every page request (a performance issue for people with modems!). The best solution is to store them in a database.

To have CF store Client variables in a database, go to the Variables section of the CF Administrator. Select the

data source you want to use for Client variable storage (create a new data source pointing to an empty database before doing this step). Click "Add" to add this data source to the list of available storage options. You'll need to answer three questions about the data source and then click "Create."

Now your data source will appear in the list of available storage options. You can set it as the default storage option (recommended) or you can have each application use a different storage data source by specifying it in the CFAPPLICATION tag as shown in Listing 1.

Q: *Can I use the Client variable scope to store complex data types (like structures or query result sets)?*

A: No and yes (isn't that helpful?). As for the no part, the Client scope can only be used for storing simple values (strings or numbers). However, there is a way around this. You can take a complex data type and turn it into a string using WDDX. To demonstrate this idea, Listing 2 shows three separate CF pages. The full details of WDDX are too great to expound in this column, but the general concept is fairly easy.

First, you must enable client management in your Application.cfm page. Note that CLIENTSTORAGE points to a data source (don't store Client variables in the Registry or as cookies!). The next page, SetClientVar.cfm, runs a query, turns that query into a WDDX packet, and creates a Client variable with the WDDX packet as its value. The third page, ReadClientVar.cfm, takes the value of the Client variable (which is a WDDX packet), turns that packet back into a query record set, and loops over that query to display some results.

Q: *After upgrading to CF 4.5, when I send an e-mail with CFMAIL (see Listing 3), the*

formatting is not retained. Even though I enter empty lines in between two sentences, those empty lines are removed when the mail is sent. Do you know why this happens, and is there any way to avoid it? I didn't have this problem before we upgraded to CF 4.5.

A: Yes! I hit my head against the wall many times before someone, I forget who, helped out. Go to your CF Admin, and check if you have this turned on: Suppress White Space. That's your culprit. You'll have to turn off this feature and use CFSETTING, or CFSILENT, instead. (This answer was provided by Raymond Camden, principal Spectra compliance engineer for Macromedia.)

Q: *What's the best way to get rid of Session variables I created? Currently, I'm just overwriting them with an empty string (i.e., <CFSET Session.LoggedIn = ""), but they still exist. Is there a better way to do this?*

A: Yes, you can actually delete variables out of the Session scope using the StructDelete() function. The Session scope is actually a complex variable type called a Structure (a single variable capable of holding multiple values, each identified by a unique name). Therefore, when you "set" a Session variable, you are actually just creating a "key" or an entry in a structure named "Session." You can also delete all the values stored in the Session scope (or any structure) using the StructClear() function. Listing 4 shows some examples.

• • •

Please send your questions about ColdFusion (CFML, CF Server, or CF Studio) to AskCFDJ@sys-con.com. Visit our archive site at www.Netsite-Dynamics.com/AskCFDJ.



BRUCE@NETSITEDYNAMICS.COM

ABOUT THE AUTHOR

Bruce Van Horn is president of Netsite Dynamics, LLC, an Allaire certified instructor, and a member of the CFDJ International Advisory Board.

Listing 1

```
Application.cfm
<CFAPPLICATION NAME="Test"
                CLIENTMANAGEMENT="Yes"
                SETCLIENTCOOKIES="Yes"
                CLIENTSTORAGE="cfsnippets">
```

Listing 2

```
Application.cfm
<CFAPPLICATION NAME="Test"
                CLIENTMANAGEMENT="Yes"
                SETCLIENTCOOKIES="Yes"
                CLIENTSTORAGE="cfsnippets">

SetClientVar.cfm

<CFQUERY NAME="GetCenters" DATASOURCE="cfsnippets">
    SELECT * FROM Centers
</CFQUERY>

<CFWDDX ACTION="CFML2WDDX" INPUT="#GetCenters#"
OUTPUT="CentersWDDX">

<CFSET Client.Centers = Variables.CentersWDDX>

ReadClientVar.cfm

<CFWDDX ACTION="WDDX2CFML" INPUT="#Client.Centers#" OUT-
PUT="GetCenters2">

<CFOUTPUT QUERY="GetCenters2">
```

```
#GetCenters2.Name#, #GetCenters2.City#<BR>
</CFOUTPUT>
```

Listing 3

```
<CFMAIL...>
#DateFormat(Now(),'mm/dd/yy')#

Dear #GetClients.FirstName# #GetClients.LastName#:

Thank you for your recent...

Sincerely,
...
</CFMAIL>
```

Listing 4

```
<!-- to delete a single session variable -->
<CFSET tmp = StructDelete(Session,"LoggedIn">

<!-- to delete all session variables -->
<CFSET tmp = StructClear(Session)>
```

CODE
LISTING

The code listing for
this article can also be located at
www.ColdFusionJournal.com

CFEXTRAS
www.cfxtras.com

CFDYNAMICS
www.cfdynamics.com

Ektron Turns Up the Heat

eWebEditPro eases writing HTML

BY
ERON
COHEN



If you have used any content management systems or any of the popular Web-based e-mail portals, you've probably had an opportunity to use something like eWebEditPro.

The idea behind this product is to make it easy for anyone to write HTML-enhanced content right on a Web site.

When the user goes to type text, instead of a regular bland HTML `<TEXTAREA>`, he or she is greeted by a wealth of menus and buttons, similar to what you would find in Microsoft Word 2000. With clicks of the mouse, a user can choose fonts, insert images and tables, and even format the document.

This adds power and flair to any sort of Web-based material and takes the pressure off Web developers to add and change Web content.

Improvements in 2.0

Ektron, the creators of eWebEditPro, has been making the product for a few years, and frankly they've gotten it down. The newest version of eWebEditPro is a real treat for ColdFusion developers, and in the wake of a whole generation of new competitors, it brings the battle of inline WYSIWIG HTML editors to a new level.

eWebEditPro was one of the first of its kind, so it's no wonder that Ektron has been able to come up with such useful and needed innovations with eWebEditPro 2.0.

The list of improvements in this version is long:

1. **A much better installation program:** As a seasoned installer of software, I've had my share of setup difficulties. Ektron has dramatically improved the installation process by creating a well-planned windows installer. It makes it much easier to get everything going with ColdFusion.
2. **A built-in FTP engine:** Instead of a user choosing from a library of prepared images, he or she can upload them to a specified FTP location and they'll be ready to go. This is a semiautomated process, so it should be no problem for inexperienced users to grasp. Once uploaded, eWebEditPro 2.0 automatically generates the correct "src=" link inside of an image tag to embed the image into the document.
3. **Improved customization:** As before, the Web developer can set up which options and buttons are available to the end user. This is done by editing an XML file that gets installed by default with eWebEditPro. It's also possible to designate a cascading stylesheet to be applied to content that's created in the editor. But it doesn't stop there. You can enable the end user to set up his or her own customizations. He or she can choose which buttons to display on the toolbar and turn on and off one of the five peel-off menu palettes.



Other Enriching Features

Other features really make it a gem. For instance, Ektron has done a fantastic job with "cut and paste" from other Windows

programs. If the end user has a document that was created in Microsoft Office (or some other compliant Windows program), he or she can simply copy and paste it into eWebEditPro.

Once it's pasted, the material is automatically scanned and cleaned of the superfluous XML junk that Microsoft wants us to think is necessary. And with the addition of the stylesheet override feature in 2.0, the Webmaster will have extra control over the elements of the document – above and beyond just restricting fonts and styles that are available in the toolbar.

Other perks also come with this product. For instance, it's one of the few in its class that supports Netscape. It also has a spell checker, although this feature is relatively slow and depends on having Office 97 or higher installed for use.

It also comes with what is often overlooked as a feature, a well-written developer's guide and, equally important, a very good end-user's guide.

It bears mentioning that eWebEditPro also works with PHP, ASP, Perl, and JSP. It also works with some of the more popular content management systems such as those from Interwoven and Vignette. In fact Macromedia thought it was so good that it now includes eWebEditPro with every copy of Spectra.

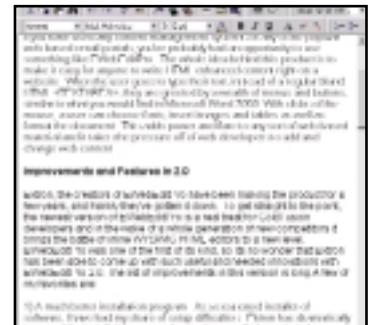


FIGURE 1: The eWebEditPro 2.0 interface

Getting eWebEditPro 2.0

The base price for the software is \$299 for 10 non-concurrent users. This means that you designate 10 specific people as end users. So to stick to the license, only those 10 people may use it.

If you opt for this license, it means you can't put this baby on your Internet Web site for the general public to use. If you have this in mind, you'll have to move up to the enterprise edition, which is \$5,999 and requires you to sign a contract with Ektron. If your organization doesn't need the unlimited license, but needs more than the initial 10, it can add on the number of users it needs by spending an additional \$299 for every 10 users.

eWebEditPro 2.0 has a free 30-day trial that you can download from its Web site. Visit www.Ektron.com for information about eWebEditPro and other Ektron offerings.

Also Ektron has recently introduced eMPower Express, a low-cost Web content management application that ships with a fully integrated copy of eWebEditPro. Priced at \$499, this is an alternative for Web professionals who do not want to custom-build an application.

About the Author

Eron Cohen has been working with ColdFusion since 1996. Currently he is a freelance Web developer and trainer in Maryland.

eron_cohen@yahoo.com

WEB SERVICES INTERNATIONAL DEVELOPER CONFERENCE & EXPO

Hilton New York, NYC

Santa Clara Convention Center, CA

web services **EDGE**
conference & expo

EAST COAST



web services **EDGE**
conference & expo

WEST COAST



Get Up to Speed with the Fourth Wave in Software Development

Real-World Web Services:
Is It Really XML's Killer App?

Demystifying ebXML:
A Plain-English Introduction

Authentication, Authorization
and Auditing:
Securing Web Services

Wireless: Enable Your
WAP Projects for Web Services

The Web Services Marketplace:
An Overview of Tools,
Engines and Servers

Building Wireless Applications
with Web Services

How to Develop and Market
Your Web Services

Integrating XML in a
Web Services Environment

Real-World UDDI

WSDL: Definitions and
Network Endpoints

Implementing
SOAP-Compliant Apps

Deploying EJBs in a
Web Services Environment

Swing-Compliant Web Services

The First Major Conference & Expo Focusing Exclusively on Web Services

Integrating applications with Web Services

Web services, in its infancy, promises to be the
next major impact on the business model of the
new millennium.

The Web Services Edge Conferences in New York
and California provide the venue to explore the
implications of Web Services and their impact on
business process.

The Expo will bring together leading technology
vendors and their business partners to showcase
the products and services necessary to integrate
applications and create e-business applications.

Now is the time for companies to become
educated on the opportunities Web Services
create for more efficient enterprises.

Web Services Edge is the venue to get up
to speed.

Plan to Sponsor

Many sponsorships are exclusive.
Rise above the noise and let SYS-CON Media's
integrated marketing work for you.

webServices

Plan to Exhibit

Call 201 802-3004 to reserve space

Steve Benfield, a leading industry expert
and frequent speaker at business and
technology events, is also cofounder and
CEO of Bondi Software.



Steve Benfield
Conference Tech Chair

EAST
Sept 24-25, 2001

WEST
Oct 24-25, 2001

Owned and
Produced by

**SYS-CON
EVENTS**

135 Chestnut Ridge Road
Montvale, NJ 07645
201 802-3069
Fax: 201 782-9601
visit

WWW.SYS-CON.COM/WEBSERVICEEDGE
for more information

Call
201 802-3004
to reserve
your space
today!

NEITHER SYS-CON MEDIA NOR SYS-CON EVENTS, INC., ARE SPONSORS OF OR ARE AFFILIATED IN ANY WAY WITH, CAMELOT COMMUNICATIONS, INC., OR ANY EVENTS PRODUCED BY CAMELOT COMMUNICATIONS, INC.

MEDIA SPONSORS:

**SYS-CON
MEDIA**

JAVA DEVELOPERS
JOURNAL

XML JOURNAL

wireless

ColdFusion JOURNAL

UNIX JOURNAL

WebSphere

www.javadevelopersjournal.com

www.wirelessjournal.com

www.xml-journal.com

www.powerbuilderjournal.com

www.coldfusionjournal.com

www.linuxbusinessweek.com

subscribe online
www.sys-con.com
or call
800 513-7111

wireless | java | xml | linux | coldfusion | powerbuilder

SYS-CON MEDIA

SYS-CON MEDIA

www.WebServicesDevelopersJournal.com

The World's Leading Independent
Web Services Journal Resource

PREMIER ISSUE

Web Services JOURNAL

Volume 1 Issue 1

The Web Services Marketplace

written by Jack Morin

Feature: Is It Real?
Real World web services

Security: Audit
and Auditing Sys

Wireless: Ex
for Web Services

Tutorial: How
Web Services A

WSDL: Defin
Open standards-based

Feature: Deploying
Web Services Environments

WebServices & Sizing: Ant

Feature: Implementing
SOAP-compliant Applications

Introductory
Charter
Subscription
SAVE 20%

SUBSCRIBE NOW

ONLY \$79.99
for 1 year
12 issues

regular rate \$89.99 for 12 issues
Canada/Mexico \$99
All other countries \$129

AND SAVE \$20

Off the Annual Subscription Rate

SYS-CON Media, the world's leading publisher of technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of Web Services

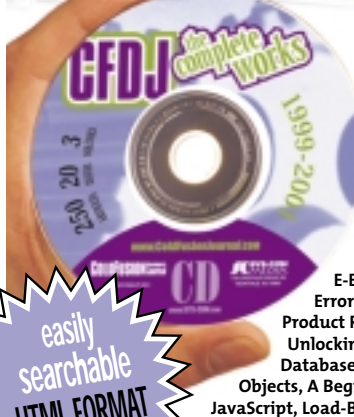
web Services JOURNAL

ORDER ONLINE AND GET **10% DISCOUNT**

THE COMPLETE WORKS



only
\$79.99



Check out over 250 articles covering topics such as...

Custom Tags, ColdFusion and Java, Finding a Web Host, Conference Reports, Server Stability, Site Performance, SYS-CON Radio Interviews, ColdFusion Tips and Techniques, Using XML and XSLT with ColdFusion, Fusebox, Building E-Business Apps, Application Frameworks, Error Handling, Wireless ColdFusion, Product Reviews, Ask the Training Staff, Unlocking Verity's Potential, Authentication, Database Tips and Tricks, Monitoring, Smart Objects, A Beginner's Guide to CF, Safe Scripting, JavaScript, Load-Balanced Servers and more...

Questions? E-mail CFDJCD@SYS-CON.COM

GO TO **WWW.JDJSTORE.COM** TO ORDER

easily
searchable
HTML FORMAT

web services **EDGE** conference & expo

October 24-25, 2001

XML **EDGE** conference & expo 2001

October 22-25, 2001

Santa Clara
Convention
Center
Santa Clara, CA

FREE
Registration
Register for one...attend both

Fundamentally
Improving the
Speed, Cost &
Flexibility of
Business
Applications

Many New
XML Products
Will Hit the
Market
in Q3
of 2001

See them First at
XMLEdge2001

**Register
Online!**

www.SYS-CON.com

or Call

201 802-3069

Opening Keynoter

Charles Goldfarb
The Father of XML

**SYS-CON
MEDIA**

Owned & produced by

**SYS-CON
EVENTS**

WEB SERVICES CEO PANEL



Litwack
CEO
SilverStream



O'Connor
President
Sonic Software



Kutay
CEO
AltoWeb



O'Toole
Chairman
Cape Clear



Slama
CEO
Shinka Technologies



Morris
CEO
IONA Technologies

W W W . S Y S - C O N . c o m

XML JOURNAL

JAVA DEVELOPER JOURNAL

webServices

wireless

WebSphere

CONFUSION

WebLogic

webtechniques

SDTimes

Programmer's Paradise

bea

xml expo

XMLCities

shinka

CAPE CLEAR

sonic

SilverStream

PointBase

Your Own Magazine

Do you need to differentiate yourself from your competitors?
Do you need to get closer to your customers and top prospects?
Could your customer database stand a bit of improvement?
Could your company brand and product brands benefit from a higher profile?
Would you like to work more closely with your third-party marketing partners?
Or, would you simply like to be a magazine publisher?

SYS-CON Custom Media is a new division of SYS-CON, the world's leading publisher of Internet technology Web sites, print magazines, and journals.

SYS-CON was named America's fastest-growing, privately held publishing company by *Inc. 500* in 1999.

SYS-CON Custom Media can produce inserts, supplements, or full-scale turnkey print magazines for your company. Nothing beats your own print magazine for sheer impact on your customers' desks... and a print publication can also drive new prospects and business to your Web site.

Talk to us!

We work closely with your marketing department to produce targeted, top-notch editorial and design. We can handle your distribution and database requirements, take care of all production demands, and work with your marketing partners to develop advertising revenue that can subsidize your magazine.



So contact us today!

East of the Rockies,
Robyn Forma,
robyn@sys-con.com,
Tel: 201-802-3022

West of the Rockies,
Roger Strukhoff,
roger@sys-con.com,
Tel: 925-244-9109

Once you're in it...

- Wireless Business & Technology
- Java Developer's Journal
- XML Journal
- ColdFusion Developer's Journal
- PowerBuilder Developer's Journal

...reprint it!



Contact Christine Russell
201 802-3024
christine@sys-con.com



Reprints

www.WebSphereDevelopersJournal.com

The World's Leading Independent WebSphere Developer Resource

PREMIER ISSUE

WebSphere

DEVELOPER'S JOURNAL

U.S. \$15.00 (CANADA \$18.00)

Consolidating Legacy Data with WebSphere

written by Brady Flowers

Feature: Programming IBM WebSphere Applications

EJB: IBM Advanced WebSphere Program

Feature: IBM WebSphere Network Design Implications

Tutorial: Introduction to Application Server EJB

XML: WebSphere XML Open standards-based technology

Feature: How to Install and Configure WebSphere Performance Pack

WebSphere & EJB Corner: An Introduction to High performance applications

Feature: How to Configure WebSphere in Multiprocessor Environments

Introductory Charter Subscription

SAVE 20%

SUBSCRIBE NOW

ONLY \$144 for 1 year 12 issues

regular rate \$180 for 12 issues

AND SAVE \$35

Off the Newsstand Cover Price

SYS-CON Media, the world's leading publisher of technology magazines for developers, software architects, and e-commerce professionals, brings you the most comprehensive coverage of WebSphere

WebSphere

WebSphere and WebSphere-based logos are trademarks or registered trademarks of IBM in the United States and other countries.

What's Online

www.sys-con.com/coldfusion

CFDJ Online

Check in everyday for up-to-the-minute news, events, and developments in the ColdFusion industry. Visit www.sys-con.com/coldfusion and be the first to know what's happening.

ColdFusion Edge 2001, New York, September 23-26, 2001 Register now and save \$600

The ColdFusion FastTrack at **JavaEdge 2001** offers ColdFusion developers, both beginner and advanced, comprehensive technical sessions designed to prepare them for the ColdFusion 4.5 Certified Developer Exam. Topics for the conference include Web fundamentals, application development, database concepts, client state management, and troubleshooting. Sessions are designed for beginner, intermediate, and advanced developers.

Go to www.sys-con.com/coldfusionedge for registration, conference schedule, keynotes, sponsors, travel information, and a list of exhibitors.

Search ColdFusion Jobs

ColdFusion Developer's Journal is proud to offer our employment portal for IT professionals. Get direct access to the best companies in the nation. Find out about the "hidden job market" and how you can find it. As an IT professional curious about the market, this is the site you've been looking for.

Simply type in the keyword, job title, and location and get instant results. You can search by salary, company, or industry.

Need more help? Our experts can assist you with retirement planning, putting together a résumé, immigration issues, and more.

JDJStore.com

You're just clicks away from what's new in the market. There are new tools and programs to help you get where you need to be. From the latest version of ColdFusion Studio to Allaire's Spectra, it's easy to get them right on your doorstep. They're all here – just visit the **CFDJ** Web site.

Ben Forta's ColdFusion Tip of the Day

Click here for ColdFusion tips, links, tags, and resources from Allaire's CF evangelist. A new tip every day!

Product Reviews

Considering a product upgrade? Want to know the ins and outs of a new product before you purchase it? Make sure you read our in-depth product reviews.

Our writers get behind the hype and give you the facts. All products are tested by experts and leaders in the information technology industry.



<dot.com>

- buyer's guide
- coldfusion forums
- mailing list
- coldfusion jobs
- coldfusion store

<magazine>

- advertise
- authors
- customer service
- editorial board
- subscribe

<content>

- archives
- digital edition
- editorial
- features
- interviews
- product reviews

<conferences>

- coldfusion edge
- fast track
- new york, ny
- sept 23-26

<search cfdj>

Search

<cfdj specials>



JDJStore.com
\$1,295.00

ColdFusion Server 5
Professional Edition

QUICK POLL

- What is your favorite new feature of ColdFusion 5.0?
- ☐ User-Defined Functions
 - ☐ Query of Queries
 - ☐ Application Deployment
 - ☐ Integrated Clustering
 - ☐ Performance Optimizations
- Vote

Click for a FREE 30-day trial



Click here to DOWNLOAD



empirix

INFORMATION

NT

web

hosting

FREE

LIST

code

CFWEB

AbleCommerce

shopping

cart

FREE

TRIAL

version

3.0

download

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

CFML

Macromedia Releases ColdFusion Server 5

(San Francisco, CA) – Macromedia, Inc., announced the availability of ColdFusion Server 5, the latest version of its Web application server.

ColdFusion Server 5 offers major enhancements in development, administration, and performance. A new integrated charting engine, advanced full text

searching, and new language features enable faster development of business

applications. Extensive new administrative tools reduce the time and costs for deploying and managing ColdFusion applications. ColdFusion 5 also delivers a significant performance increase – running some

applications as much as three times faster than ColdFusion 4.5. A free 30-day evaluation version is available for download at www.macromedia.com/software/coldfusion/resources/

CFXGraphicsServer Gets New Home

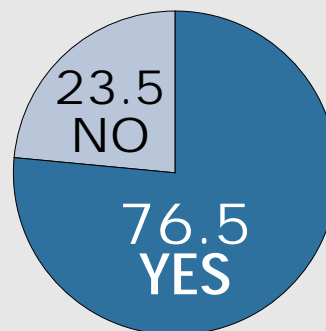
(Rockville, MD) – TeraTech has acquired ownership of the rights to CFXGraphicsServer, the server-side graphing and charting solutions for ColdFusion.

CFXGraphicsServer incorporates over 30 graph types



and styles and over 100 possible graph attributes. It also includes a full VTML "Visual Inference" that speeds up development. www.cfxgraphicsserver.com www.teratech.com

ColdFusionJournal.com Live Poll



Will You Upgrade to ColdFusion 5.0?

The first of our live polls at ColdFusionJournal.com. Please check the site regularly for new polls, online content, and more!

Activedit v2.5 Now Available

(Utica, NY) – CFDev.com introduces a new version of Activedit with spell checker, customizable tool bars, tab view, and the addition of special characters.

Activedit provides developers with the ability to embed an HTML editor into a Web page and save the



generated HTML into a database, file, e-mail, and more. It was awarded "Best Custom Tag" in last year's **ColdFusion Developers Journal** Readers' Choice Awards. A free trial of the product is available at www.cfdev.com.

ADVERTISER INDEX

ADVERTISER	URL	PH	PG
ABLECOMMERCE	WWW.ABLECOMMERCE.COM	888.801.1333	2-3
ACTIVEPDF	WWW.ACTIVEPDF.COM	888.389.1653	11
CFDYNAMICS	WWW.CFDYNAMICS.COM	800.422.7957	59
CFXHOSTING	WWW.CFXHOSTING.COM	866.CFX-HOST	53
CFXTRAS	WWW.CFXTRAS.COM	704.408.6186	59
CODECHARGE	WWW.CODECHARGE.COM	650.754.9810	21
COLDFUSION EDGE 2001	WWW.SYS-CON.COM/COLDFUSIONEDGE	201.802.3069	8-9
CONCEPTWARE AG	WWW.CONCEPTWARE.COM/EYE	011.781.275.6171	17
CORDA TECHNOLOGIES	WWW.CORDA.COM	801.805.9400	19
EMPIRIX	WWW.EMPIRIX.COM	781.993.8500	4
EVOLUTIONB	WWW.EVOLUTIONB.COM/WDJA	877.622.7551	63
HOSTCENTRIC	WWW.HOSTCENTRIC.COM/CFDJ	888.932.9376	27
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM	877.215.HOST	47
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800.379.7729	64
JDJEDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/JDJEDGE	201.802.3069	54-55
JDJ STORE	WWW.JDJSTORE.COM	888.303.JAVA	39, 60
MACROMEDIA	WWW.MACROMEDIA.COM/DOWNLOADS	888.939.2545	23
MACROMEDIA	WWW.MACROMEDIA.COM/TRAINING	888.939.2545	29
MACROMEDIA	WWW.VUE.COM/ALLAIRE	877.460.8679	31
PACIFIC ONLINE	WWW.PACONLINE.NET	877.503.9870	41
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	37
SYS-CON CUSTOM MEDIA	WWW.SYS-CON.COM	925.244.9109	35
SYS-CON MEDIA REPRINTS	WWW.SYS-CON.COM	201.802.3024	35
WEB SERVICES EDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/WEBSERVICESEDGE	201.802.3004	51
WEB SERVICES JOURNAL	WWW.SYS-CON.COM/WEBSERVICES	201.802.3000	39
WEBSHEDDEVELOPERS JOURNAL	WWW.SYS-CON.COM/WEBSHERE	800.513.7111	35
WIRELESS EDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/WIRELESS	201.802.3069	49
XMLEDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/XMLEDGE	201.802.3069	12-13

Next Month...

Here's a sneak peek...

Optimal Development Environment: Working Together in CF by Sarge Sargent

A Script for Teamwork: Distributed, team development with ColdFusion by Hal Helms

Tracking Software Issues: Using the Web to collaborate on tracking and resolving software issues by David Keener

CFCONTENT with Images: From Web Bugs to Banner Servers: Use CFCONTENT to secretly track readers of your e-mail with ColdFusion by Eron Cohen and Michael Smith

COLDFUSION Developer's Journal

Don't miss the August issue!

EVOLUTIONB

www.evolutionb.com/casestudies

INTERMEDIA.NET
www.intermedia.net