



COLD FUSION Developer's Journal

ColdFusionJournal.com

November 2003 Volume: 5 Issue: 11

ANNOUNCING

SEE PAGE 35

**EDGE
2004 (EAST)**

**FEB 24-26
2004
BOSTON**

Development Technologies Exchange

Editorial

CF_Defense

Robert Diamond page 5

CF Community

Tales from the List

Simon Horwith page 7

Tips & Tricks

Enhancing Verity Search Results

Randy L. Smith page 8

Absolute Beginners

Persistence: Creating State

Derrick Rapley page 42

ColdFusion News

page 50

RETAILERS PLEASE DISPLAY
UNTIL JANUARY 31, 2004

\$8.99US \$9.99CAN



**SYS-CON
MEDIA**

Joseph Flanigan
page 16

CONFIRMING THE NAMESPACE

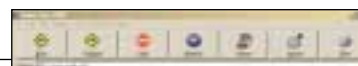
UDF: isDefinedValue

Web Sites: Accessibility the Easy Way
Web standards help ease the pain PART 3



Sandra Clark
20

<BF on CF>: The Ten Commandments 2004
A new, improved version

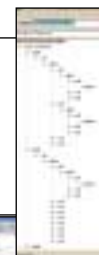


Ben Forta
26

Tags: Harnessing the Magic of the <gleep> Tag
Do some powerful things with very simple code

Bob Siegel
30

**Project Management: Working in a
Distributed Development Team**
Good written communication is paramount



Gavin Brook



32

CFCs: component.cfc
The mother of all components



Simon Horwith
36

Foundations: Unnatural Acts
Seven common practices that lead to failure...and their remedies

Hal Helms
38

Blueprints: CFDEBUGGER
Tracing code execution in BlueDragon



Charlie Arehart
44



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

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

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

By the Numbers:

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



EDGE
WEB HOSTING

What are you WAITING for?

www.edgewebhosting.net

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

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

Win
a year
of free
hosting*



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

Liberate your ColdFusion Talents

NQcontent v2
WEB CONTENT MANAGEMENT

**The only CMS
with the developer in mind**

NQcontent is the ColdFusion based Content Management platform that eliminates repetitive development work and extends your talents to give you room to deliver robust, dynamic, websites that work all the time!



- Best Content Management Tool
2002 WINNER
- Most Innovative Application
2002 WINNER
- Best e-Business Software
2002 WINNER
- Best Web Application
2002 WINNER



Visit www.nqcontent.com for more information



It's everybody's PDF™

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

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



www.activePDF.com

Visit our booth #217 at Macromedia MAX 2003

editorial advisory board

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

editorial**editor-in-chief**

Robert Diamond robert@sys-con.com

technical editors

Raymond Camden raymond@sys-con.com

Simon Horwith simon@sys-con.com

executive editor

Jamie Matusow jamie@sys-con.com

editor

Nancy Valentine nancy@sys-con.com

associate editors

Gail Schultz gail@sys-con.com

Jean Cassidy jean@sys-con.com

Jennifer Van Winckel jennifer@sys-con.com

production**production consultant**

Jim Morgan jim@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com

Richard Silverberg richards@sys-con.com

Tami Beatty tami@sys-con.com

contributors to this issue

Charlie Arehart, Gavin Brook, Sandra Clark,
Joseph Flanigan, Ben Forta, Hal Helms,
Simon Horwith, Derrick Rapley,
Bob Siegel, Randy L. Smith

editorial offices**SYS-CON MEDIA**

135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

TELEPHONE: 201 802-3000 FAX: 201 782-9638

COLD FUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)

is published monthly (12 times a year)

by **SYS-CON Publications, Inc.**

postmaster: send address changes to:

COLD FUSION DEVELOPER'S JOURNAL

SYS-CON MEDIA

135 Chestnut Ridge Rd., Montvale, NJ 07645

©copyright

Copyright © 2003 by SYS-CON Publications, Inc.
All rights reserved. No part of this publication may
be reproduced or transmitted in any form or by any means,
electronic or mechanical, including photocopy
or any information, storage and retrieval system,
without written permission.

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

For promotional reprints, contact reprint coordinator.
SYS-CON Publications, Inc., reserves the right to
revise, republish and authorize its readers to use
the articles submitted for publication.

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

CF_Defense

Every month when it comes time to write this editorial, I look around at a variety of sources for inspiration. It's not as easy writing each month as most of us on the magazine (hopefully) make it look. In order to cover the CF universe with the bird's-eye view that we've got here on a monthly basis, it requires keeping up on all the various goings-on throughout the real-time Net, and then selecting those things that can be covered each month in the space given.

Looking for inspiration this month was one of the easier tasks because there was some good news to report, and who doesn't like good news? I came across a proud blog entry by Dominic Plouffe, cofounder of FuseTalk (www.fusetalk.com). He talks about AnandTech Forums, a client of theirs whose site is devoted to the discussion of computer hardware (<http://forums.anandtech.com>). They've recently been named the second largest forum on the Web today, by Big Boards (www.bigboards.com), a site that tracks online message boards. This is all quite relevant here because not only does AnandTech run off of ColdFusion, it's running off of it quite well.

We tout CF success stories in the magazine from time to time, and a couple of times a year cover what major sites out there are running off of CFML. Ben Forta has an excellent ongoing list up on his site at <http://forta.com/cf/using>. The feedback that we've received from readers on this is that aside from being an interesting break from the technical content, this data helps sell clients on using one of the CFML servers on the market today.

As a whole, the language, because of the easy learning curve that attracted many of us to it, has often been unfairly dismissed as a tool for beginners, not given the credit that it deserves for what it can do when used right. It's sometimes a challenge to both those in the consulting space, and those of us in internal IT departments to present CF-based solutions and to defend them among all the other solutions out there.

Getting back to the AnandTech forums, they are currently serving 120,000+ users off of a



By Robert Diamond

FuseTalk ColdFusion solution pulling from a back-end database of over 9.4 million messages. The forums see 14–20 million page views, and about 2 million unique users per month.

We checked with the folks over at FuseTalk about the hardware configuration, and here's what they're running off of:

Database Server:


Microsoft SQL Server 2000
3GB memory, RAID 10
Dual 1900 AMD processors

Web servers (3 of them):

ColdFusion 5
512MB memory • 1 IDE hard drive
Dual 1900 AMD processors

Because of the sheer technical power that they're throwing at it, as well as the high amount of traffic, they're using hardly any caching of the forums – it's hitting that SQL server constantly. Additionally, the three Web servers provide a redundant system to serve it all up.

It's important to tout the CF_Success stories out there, and if you've got one of your own running a top-tier site, let us know for future coverage!

One other bit of recommended reading this month is a great blog entry by Ben entitled, "The Changing Case Against ColdFusion," which is another must-read (<http://forta.com/blog/index.cfm?mode=e&entry=935>). 

About the Author

Robert Diamond is vice president of information systems for SYS-CON Media, and editor-in-chief of ColdFusion Developer's Journal. Named one of the "Top thirty magazine industry executives under the age of 30" in Folio magazine's November 2000 issue, Robert holds a BS degree in information management and technology from the School of Information Studies at Syracuse University. Visit his blog at www.robertdiamond.com.

robert@sys-con.com

Ektron Success Story #1096

Enterprise Web Content Management without the enterprise integration.



Beth knows that effective, timely communication is key in today's competitive market. By choosing Ektron's enterprise Web content management solution with Web Services and RSS Syndication, her partners and distributors are receiving accurate, up-to-date information. As a result the Web site has become a powerful, strategic, business tool.

Let us show you how an Ektron XML Web Content Management Solution can enhance your Web site.

Learn More at:
www.ektron.com/cfdj

Ektron - Redefining Web Content Management

president & ceo

Fuat Kircaali fuat@sys-con.com

vp, business development

Grisha Davida grisha@sys-con.com

group publisher

Jeremy Geelan jeremy@sys-con.com

technical director

Alan Williamson alan@sys-con.com

advertising**senior vp, sales & marketing**

Carmen Gonzalez carmen@sys-con.com

vp, sales & marketing

Miles Silverman miles@sys-con.com

advertising director

Robyn Forma robyn@sys-con.com

director of sales & marketing

Megan Mussa megan@sys-con.com

advertising sales manager

Alisa Catalano alisa@sys-con.com

associate sales managers

Carrie Gebert carrieg@sys-con.com

Kristin Kuhnle kristin@sys-con.com

sys-con events**president, events**

Grisha Davida grisha@sys-con.com

conference manager

Michael Lynch mike@sys-con.com

regional sales manager

James Donovan james@sys-con.com

customer relations**circulation service coordinators**

Niki Panagopoulos niki@sys-con.com

Shelia Dickerson shelia@sys-con.com

Edna Earle Russell edna@sys-con.com

manager, jdj store

Rachel McGouran rachel@sys-con.com

sys-con.com**vp, information systems**

Robert Diamond robert@sys-con.com

web designers

Stephen Kilmurray stephen@sys-con.com

Christopher Croce chris@sys-con.com

online editor

Lin Goetz lin@sys-con.com

accounting**financial analyst**

Joan LaRose joan@sys-con.com

accounts receivable

Kerri Von Achen kerri@sys-con.com

accounts payable

Betty White betty@sys-con.com

subscriptions**Subscribe@sys-con.com**

1 888 303-5282

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

Cover Price: \$8.99/issue

Domestic: \$89.99/yr (12 issues)

Canada/Mexico: \$99.99/yr

all other countries \$129.99/yr

(U.S. Banks or Money Orders)

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

Tales from the List

Whole lotta shaking going on!

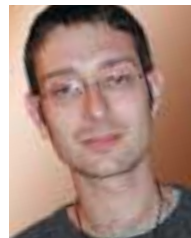
That's right, there's a whole lotta shaking going on – in the IT world, that is. Macromedia's Studio MX 2004 suite of products has now begun shipping, and the reviews are in! In addition to the suite, Macromedia Central has entered the public limelight with the public beta release having been launched. Not all of the news comes from Macromedia. SYS-CON Media, the folks who bring you *ColdFusion Developer's Journal* each month, have now launched *MX Developer's Journal* – a magazine aimed at covering the entire range of MX products.

Rather than examining a specific *CFDJ* List thread this month, I'm going to examine the overall discussion topic trends and buzz that's a result of the new suite of products.

First of all, I'll begin by saying, "shame on you" to anyone out there who has not yet picked up a copy of Studio MX 2004. Without a doubt, this is the finest suite of development products on the market. Each individual product is worlds superior to its predecessors. Macromedia has also given us material proof that their vision of how development tools will look, feel, and integrate with each other really works. Hats off to the Macromedia product development teams – they've outdone themselves!

There have been many reviews of the suite – all of which are good. Those of you who are looking for reasons to purchase or upgrade to Studio MX 2004 or who are trying to convince your boss to do so, should look to these articles and reviews for compelling reasons to do so. To name a few, *PC Magazine* gave a stellar review (www.pcmag.com/art cle2/0,4149,1275209,00.asp) and Forrester Research gave the new suite very good marks as well (www.macromedia.com/homepage/forrester.pdf). Flash MX 2004 also won a CNET Editor's Choice Award in August 2003.

Macromedia also just released the public beta of Central, the new offline/online Flash Application-based browser. For those of you who haven't yet seen it, Central is a very slick new product release from Macromedia, aimed



By Simon Horwith

at testing experimental waters. Essentially Central is a browser. Rather than browsing the Web though, Central runs Flash applets (mini applications) within its window. These Flash applications communicate with the Web via Web services in order to retrieve data in real time and for offline viewing, and to allow end users to interact with other server-side functionality.

The mini Flash apps not only have the ability to store data for offline viewing, but also have the ability to exchange data with each other. This allows you to do things like export an address book entry from a Flash e-mail inbox to a calendar application, or download the morning news for offline viewing later.

The tighter integration of the MX suite of

—continued on page 47

About the Author

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

simon@horwith.com

Enhancing Verity Search Results

Let Verity do the indexing work

Your customer says, "I want my site search to include all of my regular site pages and my data-based items as well. When my customer clicks on the link, I want it to show the correct page. Oh, and I want all occurrences of the searched phrase to be highlighted in yellow."

Back at your desk, you follow the book and plug in the parameters for each page. The data-based entries are a little more difficult, but you figure it out and run the program to populate the Verity search engine.

The items in the static pages are found okay, but none of the data-based content appears correctly – although sometimes you see your ColdFusion code show up in the results page (Oy!). You mess around a little more and get the data-based items into Verity, but clicking on the link keeps taking you back to the home page or to the root of the data-based page, ignoring your URL options.

And what's this about a yellow highlight?

Review of the Basics

This article assumes you know the basics of ColdFusion programming and have at least tried to get the Verity search engine working. I am not going to delve into an in-depth explanation of what Verity is, but I will start with a very quick overview of how to get rolling. I suggest you pick up a copy of Ben Forta's ColdFusion Web Database Construction Kit books if you haven't yet attempted Verity.

To further illustrate this story, I will refer you to a client site, www.pcamn.org (Prevent Child Abuse Minnesota), which I have received permission to use. I recently upgraded their site search to do every-



By Randy L. Smith

thing that will be discussed in this article, and it will serve as my working example.

To begin, we'll first add our static pages to the Verity search engine. Every time I update a Verity search engine collection, I like to wipe it clean to make sure nothing old hangs around to haunt me. To do this, I use the action "refresh" when I add

the first page:

```
<CFINDEX collection="PCAMN" action="REFRESH" type="FILE"
```

```
key="d:\webclients\www.pcamn.org\www\about.cfm"
language="English"
urlPath="http://www.pcamn.org">about *
```

There is a server-oriented path to the file and a Web-oriented URL to the root location of that page (Verity handles adding the "about.cfm"). I also added the name of the file with an asterisk separator after the CFINDEX operation because I have the Verity search engine update script automated (more on that later) with the "Write Results to a File" option selected. I send myself a daily report that includes all of the text files from all of my automated systems in one combined daily report. This allows me to check every morning for any problems with my automated systems.

Next, we want to add the rest of our static pages to the Verity index using the "action=update" option. You can copy and

paste to add them all, or you can make life simpler for yourself by putting the names of the files in a list and looping through that list to add the files to the index:

```
<cfloop index="ii" list="abuse.cfm,
advocacy.cfm, radiothon.cfm, campaign.cfm,
involved.cfm, resources.cfm, give.cfm,
links.cfm, marketplace.cfm, programs.cfm, mem-
bers.cfm, involved.cfm, walk.cfm,
prevention.cfm">
  <CFINDEX collection="PCAMN" action="UPDATE"
type="FILE"

key="d:\webclients\www.pcamn.org\www\#TRIM(ii)#"
language="English"
urlPath="http://www.pcamn.org">
<cfoutput>#TRIM(ii)</cfoutput> *
</cfloop>
```

Using a list will keep your script shorter and will make it much easier on the next person to update the list of indexed static pages when the clients' needs change. You could set type="path" and index the entire directory, but I usually have a few files in the directory that I don't want to index and this method allows me to be more selective.

Make sure you do not include your all-code-only pages in the list or you'll end up with code examples in the search output. Verity seems to be unhappy with pages that do not have output, and appears to revert to indexing your source code in those cases. At least that's what I've experienced. Perhaps a Verity expert could explain to us why code sometimes gets indexed and ends up in our search results.

Tricking Verity for the Data-Based Entries

We have taken care of the static pages, but how do we merge in the data-based entries and tie them to the appropriate

page when the site visitor clicks on the search results link?

I use a little trick for this because Verity seems to ignore my pleadings. Now, this is going to get just a little complicated because we use a common table for this particular client. A common table allows us to plug in a custom tag for both display and administration of their content for multiple pages containing different topics. Modifying the attributes that the tag sends makes it easy to add additional functionality on short notice. Perhaps I'll write an article on that at a later date, but suffice it to say for now that data is displayed based on a key field, and the table also has an "active" column (triggering whether the data can be displayed on the public site) as well as "start showing" and "stop showing" date fields (meaning that if today's date is outside of those dates, the data is not to be made available to the public).

In operation, we refer to a common keyword in the column "itemtext1" of this table for different pages of the site. For example, county_resources.cfm only needs records from the table where itemtext1 is "resources". Additionally, some of our data-based pages employ an additional keyword to narrow down the results better, and those are in a field named "freeform2". For instance, if we want to display all county resources for Becker County, we would call: www.pcamn.org/county_resources.cfm?itemtext=resources&freeform2=Becker.

Please refer now to Listing 1.

Similar to the static pages, we use the names of the items to loop through the CFLOOP list. Additionally, we have the names of the page URLs in the list URLLIST and a Y/N indicator in CUSTOM1LIST to tell us whether or not the page needs the FREEFORM2 parameter added. We keep track of which item in the list to use via the variable THECOUNTER.

Now, as we loop through this script we increment the counter and collect the appropriate items from these two lists into the ADDCUSTOM1 and THEURL variables. We determine what the CFINDEX parameter "title" will contain depending on the contents of ADDCUSTOM1. Study the setting of the variable THETITLE for a moment. Notice the use of the piping character "|". We will use this to our advantage when we display the search results.

Continuing, we collect the "searchresults" recordset by querying the database using ITEMTEXT1 and checking to make sure the record is active and within the date range to make sure we only index those database items that are relevant. Notice in the query how we are combining all of the usable fields from each record by putting them into a comma-delimited list in the BODY tag. This allows us to let site visitors search on all of the database fields that contain usable data. Your search will need to be refined to your clients' needs.

You could create a customized search field to combine data in the search result if you needed to. For example, my query in Listing 1 could have combined all of my search fields into one like this:

```
SELECT itemid, freeform2, itemtitle,
       RTRIM(ISNULL(itemtitle, ' ')) + ' ' +
       RTRIM(ISNULL(CAST(itemdesc AS var-
char(8000)), ' ')) + ' ' +
       RTRIM(ISNULL(freeform2, ' ')) + ' ' +
       RTRIM(ISNULL(itemtext1, ' ')) + ' ' +
       RTRIM(ISNULL(itemtext2, ' ')) + ' ' +
       RTRIM(ISNULL(freeform1, ' ')) AS search-
field
FROM tblFSN_dynamic_new ...
```

and then I would have used BODY="searchfield" in CFINDEX. The problem here is that the "itemdesc" field is a large text field and not all of the data will be indexed. It also makes for more complex code to maintain than is needed.

Moving forward to the CFINDEX function in Listing 1, notice that we are using the "custom1" and "custom2" parameters. These will come into play in the search results page.

'Custom' Tricks

Please now refer to Listing 2. This is the script for processing the search results. When the search form calls this script, one of the first things we do is remove all angle brackets and pound signs from the searcher's phrase in an attempt to keep hackers at bay. If there is nothing left in the search phrase (or if it was empty to begin with), we simply return the site visitor unceremoniously to the home page.

Next, we conduct the Verity search. This script should be old hat to you until you reach the output area where we are setting the variable THETITLE. If the title has "PUT-CUSTOM2-HERE," we replace it

with the contents of the CUSTOM2 variable from the Verity search.

Then, proceeding down a little further, if there is more than one list item in the title, using the piping character as the delimiter we set up our HREF variable to use everything after the first piping character as our URL, and if we run into any "PUT-CUSTOM1-HERE" strings we replace those with the contents of the CUSTOM1 variable from the Verity search.

This has allowed us to customize the results of the Verity search on the fly without having to custom code and index every single variation of the database. We are letting Verity do the work of indexing, and we are forcing it to output the page references that we want our program to call based on the data retrieved.

Yellow Highlights of Search Listing Results

After getting the above indexing working, highlighting the search results in yellow is actually the easy part. By performing a ReplaceNoCase on the search results for the searched-for phrase (we do this on both title and summary information, but that's up to you) and replacing that phrase with one wrapped with a standard HTML call (remember the "all" parameter on ReplaceNoCase!), you have effectively highlighted the search phrase in your results.

Post Page Yellow

After this worked properly, we thought it fell a little short because sometimes the summary returned by Verity did not contain the actual searched-for phrase. We wanted the referenced page to have the search phrase highlighted in yellow as well, but how, short of rewriting all of our pages to accept a highlight-in-yellow parameter?

If you were observant in reviewing Listing 2, you would have noticed near the middle that the URL that is actually called is "showme.cfm". Notice, also, the URL parameters "SP" and "PAGE".

Please refer now to Listing 3.

Listing 3 accepts as parameters all of the possible fields we may search on, as well as the SP and PAGE parameters. It first checks to see in what format the URL was sent (via the PAGE parameter) and if it needs to prepend "http://" -related information.

Why? Because we're going to make use

of the CFHTTP function, and CFHTTP is picky about the URL format.

After setting up the variables for CFHTTP, we call for the page using the parameters passed. When the results return, we simply do the same ReplaceNoCase call that wraps the search phrase ("SP") with the `` code. Output the resulting variable and voila! your page now has the search phrase highlighted everywhere it was found.

Keeping It All Up to Date

Alas, having a super search engine won't help much if you don't keep the data fresh, and if you have written administrative tools for your clients to manage content and/or you allow for active/not and start/stop showing dates,

you need to refresh the Verity index at least daily.

Two methods come to mind for keeping the clients' Verity index fresh. You could provide them with an administrative link to the script that populates the collection. This would let them update it whenever they updated the data. Because we all know of the human tendency to either forget or put it off, this is not the best solution.

Your best solution is to set up your ColdFusion Administrator's Scheduler to automatically run the update in the wee hours every day or week (depending on how frequently your client updates their content). Make sure you test it first by running it directly from the Administrator, and make sure your time-out value is high enough to cover an ever-expanding database of content.

Limitations

I am expecting e-mail from those of you with better ideas or criticism. Great! If they contribute to the improvement of this method, I'll submit a follow-up article. 

About the Author

Randy L. Smith is president/CEO of Midwest Computer Programming and Internet (www.mcpi.com), an Internet/intranet database solution provider based in Hudson, Wisconsin. He has been developing large-scale, Web-based applications for businesses and non-profits of all sizes, as well as state and federal entities, since 1993. Randy has been working in the computer industry since 1978, and with ColdFusion since 1996.

randy@mcpi.com

LISTING 1: Adding a Common Table to the Verity Search Engine

```
<cfset URLList="county_resources.cfm, list2.cfm, list.cfm, news.cfm,
index.cfm">
<cfset Custom1List="Y,Y,Y,N,N">
<cfset TheCounter=0>
<cfloop index="ii" list="resources,PUBLICATIONS,NEWSLETTER,News and
Events,Main Article">
    <cfset TheCounter=TheCounter+1>
    <cfset AddCustom1=ListGetAt(Custom1List,TheCounter)>
    <cfset TheURL=ListGetAt(URLList,TheCounter)>
    <cfoutput>
        <cfif AddCustom1 IS "Y">
            <cfset TheTitle="PUT-CUSTOM2-HERE|#TRIM(TheURL)|#?itemtext=#TRIM
(ii)|#&freeform2=PUT-CUSTOM1-HERE">
        <cfelse>
            <cfset TheTitle="PUT-CUSTOM2-HERE|#TRIM(TheURL)|#?itemtext=#TRIM(ii)|#">
        </cfif>
    </cfoutput>

    <cfquery name="searchresults" datasource="datasourcename"
        username="abc" password="123">
        SELECT itemid, freeform2, itemtitle, itemdesc, itemtext1, itemtext2, freeform1
        FROM tblFSN_dynamic_new
        WHERE itemtext1='#TRIM(ii)|#'
        AND active='Y'
        AND (getdate() BETWEEN startshowing and stopshowing
        OR startshowing IS NULL OR stopshowing IS NULL)
        ORDER BY freeform2
    </cfquery>

    <CFINDEX collection="FSN" action="UPDATE" type="CUSTOM" key="ITEMID"
        title="#TheTitle#">
```

```
body="itemtitle,itemdesc,freeform2,itemtext1,itemtext2,freeform1"
custom1="freeform2" custom2="itemtitle" language="English"
query="searchresults" external=NO>

</cfoutput>
<br>Found #searchresults.recordcount# records for #TRIM(ii)|# - #TRIM(TheTitle)|# *
</cfoutput>
</cfloop>
```

LISTING 2: SEARCH_RESULTS.CFM

```
<cfparam name="TheSearchPhrase" default="abuse">
<cfparam name="FORM.startrow" default=1>
<cfset TheReplaceList="<,>,&#;">
<cfset TheSearchPhrase=REPLACELIST(FORM.searchphrase,TheReplaceList,"")>
<cfset TheSearchPhrase=REPLACE(TheSearchPhrase,"","","ALL")>
<cfif LEN(TRIM(TheSearchPhrase)) LT 1</cflocation url="index.cfm"></cfif>
<html>
<head><title>PCAMN - Index</title></head>
<cfinclude template="header.cfm">
<br>
<h3>Search Results</h3>
<CFSEARCH name = "GetResults" collection = "PCAMN" criteria =
    "#TRIM(TheSearchPhrase)#"
    maxRows = "100" external = "No" startRow = "#FORM.startrow#">
    <TABLE cellSpacing="0" cellPadding="1" width="500" border="0">
        <TBODY><TR><TD width="100%">
            <CFIF GetResults.RecordCount LT 1>
                <br><br>
                <B>Nothing found for "<cfoutput>#TRIM(TheSearchPhrase)|#</cfoutput>".
            <br><br>
                Please try again with a different search phrase.</B>
            <br><br>
        </CFIF>
```

Announcing **BlueDragon 6.1!**

BlueDragon [6.1]



NOW INCLUDES:

- CFCs
- XML & Web Services
- Full Text Searching
- Internationalization
- CFML Source Encryption (BDA)
- C++ and Java CFXs
- Enhanced CF5 Compatibility
- Improved Performance and much more...

RUN YOUR CFML :

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

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

Go to www.newatlanta.com/bluedragon for details and downloads!



With over 10,000 corporate and individual customers in over 65 countries around the world, New Atlanta has provided reliable server-side products since 1997.

This is for you.



This is for you. We put in the features you told us were must-haves, plus a few we just thought you'd like.

Flash MX 2004

Flash MX Professional 2004

- › Improved runtime performance
- › Professional video support
- › Forms-based development

"It's a must-have upgrade for current users ..."

-CNET.com, August 2003



Dreamweaver® MX 2004

- › True WYSIWYG CSS
- › In-line cross-browser validation
- › Native graphics editing

"... a big step forward."

-Forrester Research

Fireworks® MX 2004

- › Faster performance
- › New drawing tools
- › New live effects and photo-editing tools

“... when it comes to web graphics creation, Macromedia's Fireworks is still the undisputed king.”

-CNET.com, August 2003



Studio MX 2004

- › All new versions of Dreamweaver, Flash, and Fireworks
- › Updated user interface
- › Deeper cross-product integration



The new versions of Dreamweaver, Flash, and Fireworks in Studio MX 2004 have been awarded the CNET Editors' Choice Award, August 2003.

Macromedia MX 2004.

Upgrade, try, or buy at www.mx2004.com

Run with it.



CONFIRMING THE NAMESPACE

UDF: isDefinedValue

● sDefinedValue extends the CF built-in function isDefined by checking for the existence of a value in the variable space. isDefined checks for the existence of a variable's named space and not for a value in the named space.

isDefinedValue improves isDefined logic scope by the proposition, "Is the variable defined and does it have some value?" An optional second parameter extends the scope further by asking, "Is the variable defined and does it have this value?"

The isDefined function is one of the most used functions in CF applications as a logic switch that enables follow-on application code. Since isDefined does not check for values in the variable space, most application programs require additional coding following the isDefined statement to verify values. As an alternative, programmers can use isDefinedValue in the same application logic and save the additional coding.

For example, the data missing in a URL query string like `?&name=` would pass `<cfif isDefined("url.name")>`,

but would fail `<cfif isDefinedValue("url.name")>` because `?&name=` has no value.

In Listing 1, isDefinedValue, the function requires one parameter (varname) [1] that "evaluates as a string value to determine whether the variable named by the string exists." This parameter is just like the isDefined function.

Statements like, `<cfif isDefined("form.submit") >` and `<cfif isDefinedValue("form.submit") >` have the same syntax. One benefit of using isDefinedValue is that it allows an optional parameter, value [10], that will be compared [11] with an existing variable's value. For the comparison, both parameters must be a ColdFusion simple value.

For example, `<cfif isDefinedValue("form.submit", "Enter")>` would test the value of form.submit for the value "Enter". Table 1 shows a complete listing of evaluations of input conditions.

ColdFusion evaluates variable values as simple values. Array index values and structure key values resolve to simple values. For example, a defined array index, `A[1] = b`; can be checked as `isDefinedValue("A[1]", "b")` to confirm that the array A exists and that it has an index of value [1] of the value "b". One test



By Joseph Flanigan

that may not be so obvious is, `x=""`; `isDefinedValue("x", "")`, which means to check that `x` exists and that its value is empty. If the test is true, `isDefinedValue` will return a 1 even though the value is empty, which may seem contradictory in that generally the function returns 0 when the value is empty.

`isDefinedValue` can check for 12 conditions that apply to ColdFusion simple and complex variable data types. The first check [5] looks at the variable name string to extract the name handle from an array variable with an index value. This allows `isDefinedValue` to check an array with an index. The built-in `isDefined` throws an error with a parameter like `isDefined("MyArray[1]")` whereas `isDefinedValue("MyArray[1]")` can check the array index location as a simple value.

CFMX introduced the try/catch feature for cfscript functions. `isDefinedValue` uses this feature to return 0 whenever an exception occurs. Within the function, exception situations can happen when `isDefined` fails [5], evaluation of the variable [7] fails, or the EQ [11] operator fails. If any of these situations occurs, it means there is no defined value, and returning a 0 [36] is consistent with the function's purpose.

Defined value testing for variables using complex data types (arrays, structures, and queries) uses the CF built-in functions to determine the data type [17, 22, 27] and then check for value. CF provides a built-in function for structures and for arrays [19, 24] that tests if the namespace is empty. Query structures require additional programming [29] to examine the record count returned by `<cfquery>`.

When a CFQUERY returns 0 records (see Table 2, note 3), the `isDefined("qGetEmployees.RecordCount")` function returns YES. Providing the query does not throw an exception, ColdFusion always returns a record count key. Inexperienced pro-

Variable to check	Parameter	Returns*
not defined	varname	0 [32]
not defined, compare value	varname, value	0 [32]
defined, empty value	varname	0 [14,19,24,30]
defined, has value	varname	1 [15,20,25,29]
defined, does not equal value	varname, value	0 [12]
defined, equals value	varname, value	1 [11]
bogus or error	varname	0 [36]

Table 1: isDefinedValue Input Evaluation Conditions and Results

* *isDefinedValue* returns a Boolean value rather than logic constants, true or false. CF treats both formats as logically the same, but Boolean values require no conversion when used with SQL server.

Test	isDefined	isDefinedValue	isEmpty
a. Simple Variable Value Equal { <code>x=1</code> ; <code>isDefinedValue("x","1");</code> }	NA	1	NA
b. Simple Variable Value Not Equal { <code>x=1</code> ; <code>isDefinedValue("x","2");</code> }	NA	0	NA
c. Simple Variable Value Is Equal Empty { <code>x=""</code> ; <code>isDefinedValue("x","");</code> }	NA	1	NA
d. Array Index Unknown { <code>a[1];</code> }	ERROR	0	ERROR
e. Array Index Assigned Value { <code>a[1] = "1";</code> }	ERROR	1	ERROR
f..Array Index Assigned Empty { <code>a[1] = "";</code> }	ERROR	0	ERROR
g. Structure Key Index { <code>request.box.leg[1] = 1;</code> }	ERROR	1	ERROR
h. Confirm Query Returns 0 Records { <code>isDefinedValue("qGet2.recordcount",0);</code> }	NA	1	NA

ERROR – ColdFusion throws error
NA – Not applicable to function

Table 2: isDefinedValue Comparison Table

Terms
<ul style="list-style-type: none"> • Variable: In programming languages, the name of a namespace member and the explicit reference to the value represented by data that the variable contains. • Variable Space: In programs, the amount of program memory allowed for the data a variable contains. • Simple Value: In ColdFusion, a simple value represents one value. The ColdFusion simple data types are strings, integers, real numbers, Boolean values, and date-time values. • Empty Value: In ColdFusion, a simple variable whose value is an empty string ("") or a string of zero length. • Assigned Value: The result of a program operation that creates the data value a variable contains. In ColdFusion, variables are generally declared when assigning the variable a value. • Declared Variable: A name that is a proper member of its namespace set used to refer to the variable's value. Some program environments have additional conditions that are not part of the namespace like data type and possible operation constraints. In program languages, a declaration can be done by explicit program statements or derived by statement construction context as is the case in ColdFusion. • Namespace: A set of unique names where each name is an alphanumeric term consisting of both letters and numbers and often other symbols that identify a data set, a statement, a program, or a catalogued procedure (i.e., in programs, the identity of a variable, an array, a function, a subroutine, a common block, or a namelist). In addition to a program environment, the notion of namespace occurs in other abstract universes like the domain name system, XML elements and entities, and dictionaries.

cf functions


grammers may test for a RecordCount namespace to determine if there is query data. And, of course, the code fails because there is no data. isDefinedValue takes the extra step [29] to ensure that the value of RecordCount is not zero.

In www.cflib.org, there's another function similar to isDefinedValue called isEmpty (see Listing 2, by Fabio Serra). If you compare the code for isDefinedValue and isEmpty, the first observation is that isEmpty is tight, clean, and efficient. isEmpty returns logic constants TRUE/FALSE and isDefinedValue returns bit value 1/0. ColdFusion treats both returns as logically the same. (I like to use bit values returns especially where the function's return will be used in binding to other services like SQL server.)

In a comparison test program where the test conditions provide "NOT isEmpty is equal to isDefinedValue" cases, both functions return the same results. In test conditions where isDefinedValue features, value checking, and array checking exceeded isEmpty, the logical inverse did not apply. Table 2 shows these cases.

A comparison of the two functions reveals a distinction in the conceptual purpose and in the practical implementation. The isEmpty logical premise is "not empty until proven empty." This premise is supported by a default return of false; that is, an unknown variable is not empty. The isDefinedValue logical premise is that "variables exist only with value." By extension, then, any unknown variable would be false. So, isEmpty(unknown) returns false and isDefinedValue(unknown) returns false. This means isEmpty does not equal isDefinedValue as a conceptual inverse.

In application programs, however, the conceptual situation does not occur unless an exception occurs because both functions verify that the variable exists before any further function processing. Once past this verification of successful determination of a variable's condition of empty or of some value, the two functions can conclude correct results and will be the logical inverse of each other. So, in practice, when determining the condition of a variable's value, NOT isEmpty is equal to isDefinedValue.

ColdFusion programmers have three excellent functions, isDefined, isEmpty, and isDefinedValue, to use for checking variables and variable conditions. isDefined confirms the namespace. isEmpty confirms the variable exists and has an empty condition. isDefinedValue confirms the variable exists with a value and, optionally, can test the value of a simple variable. 

About the Author

Joseph Flanigan's computer career spans 35 years. In 1995, he switched from Perl to ColdFusion for Web applications. By late 1999, his first ideas for the Switchbox framework started and were published as Switch-box.org in August 2000. Joseph is a member of the IEEE and ACM.

joseph@switch-box.org

Listing 1: UDF: isDefinedValue — CFMX version by Joseph Flanigan

```
1 function isDefinedValue(varname)
2 {
3     var varvalue = "";
4     try{
5         if (IsDefined(listfirst(Arguments[1],"(")))
6         {
7             varvalue = evaluate(Arguments[1]);
8             if (IsSimpleValue(varvalue))
9             {
10                 if (ArrayLen(Arguments) EQ 2 )
11                     { if ( varvalue EQ Arguments[2]){return 1;}
12                     else return 0;
13                 }
14                 else if ( find(varvalue,"(") ) {return 0;}
15                 else return 1; // something is there
16             }
17             else if (IsStruct(varvalue))
18             {
19                 if (StructIsEmpty(varvalue)) { return 0;}
20                 else {return 1;}
21             }
22             else if (IsArray(varvalue))
23             {
24                 if (ArrayIsEmpty(varvalue)) {return 0;}
25                 else {return 1;}
26             }
27             else if (IsQuery(varvalue))
28             {
29                 if (YesNoFormat(varvalue.recordcount)) {return 1;}
30                 else {return 0;}
31             }
32             return 0; // not defined
33         }
34     } //try
35     catch(Any excpt)
36     { return 0; } // return excpt.Message;
37     return 0;
38 }
```

References: www.cflib.org/udf.cfm?id=990 CFMX;
www.cflib.org/udf.cfm?ID=968 CF5

Listing 2: UDF: isEmpty – CF5 version by Fabio Serra

```
1 function isEmpty(varName) {
2     var ptr = "";
3
4     if(not isDefined(varName)) return true;
5     ptr = evaluate(varName);
6
7     if(isSimpleValue(ptr)) {
8         if(not len(ptr)) return true;
9     } else if(isArray(ptr)) {
10         if(arrayIsEmpty(ptr)) return true;
11     } else if(isStruct(ptr)) {
12         if(structIsEmpty(ptr)) return true;
13     } else if(isQuery(ptr)) {
14         if(not ptr.recordCount) return true;
15     }
16
17     return false;
18 }
```

Reference: www.cflib.org/udf.cfm?id=420

Download the Code...
Go to www.coldfusionjournal.com

One damn good Cold Fusion hosting firm!

Webcore Technologies specializes in ColdFusion, ASP and SQL hosting. We offer these solutions in both dedicated server and shared virtual environments. Webcore can design and implement clustered or load-balanced solutions, managed firewalls, VPN's, multi-site failover, and more. In addition, we also offer corporate Internet access, web site design, and technology consulting services.

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



Webcore Technologies, Inc.
877.WCT.HOST
www.webcoretech.com

Accessibility the Easy Way

Web standards help ease the pain PART 3

Last month's article covered the laws of accessibility and the means of validating accessible Web content to Section 508 and the Web Accessibility Initiative (WAI). However, making sure your Web site conforms to the law is not enough. There are plenty of Web sites out there that may comply with the WAI or Section 508, but still aren't readable in certain speech browsers.

Just as you can't assume (except in some specific intranet environments) which operating system, browser, and browser version your users will be viewing your site from, you can't specify the exact way a visually impaired user will hear your site. So, the question is, how do you make sure your site is fully accessible in the spirit of the law as well as the letter?

Letter of the law means that your site has passed a validation tool and can specifically show (in court) how it conforms to each of the 508 paragraphs or WAI guidelines. Spirit of the law means that no matter how someone enters your site, the content is available to them. We could probably do this the same way most of us develop our sites – grab every browser we know of and test, test, test. The problem though is the same, we cannot with any reasonable level of certainty declare that our site will run in every browser, every operating system, every version, and look (or sound) the same way. So the question becomes, how can we make sure that our site is usable without both breaking the bank and making our timeline unreasonable?

Has your Web site ever broken when a new browser version accesses it? Have you ever had the code in your page be almost impossible to maintain because there are so many browser-specific branchings? Ever tell yourself there has to be a better way? Well there is, and it's called Web standards.



By Sandra Clark

A History Lesson

Back in the dark ages of Web programming we had the "browser wars." Both of the major browsers at the time (Netscape Navigator and Internet Explorer) were trying to be the preeminent browser and one of the ways they did it was by introducing their own tags that were not part of the HTML standard.

(Remember Netscape's <BLINK> and Microsoft's <MARQUEE>?) In 1997, Ziff-Davis launched a petition drive for Web designers to make known their wishes to both Microsoft and Netscape to support the standards in Web development. Over 35,000 electronic signatures were collected. In 1999, the Web standards organization ([webstandards.org](http://www.webstandards.org)) launched another petition drive for Microsoft to support the Web standards. Guess what? It worked. Today, every modern browser supports Web standards (some better than others, but they all are on board).

Web Standards

So what are Web standards? Standards are those items defined by the World Wide Web Consortium (W3C) and are broken into several areas:

- Structural Languages (markup)
 - HTML 1.0 and 1.1
 - XML 1.0
- Presentation Languages (presentation)
 - Cascading Style Sheets (CSS) 1.0 and 2.0

Web standards promote the separation of presentation from content. When you get

right down to it, the most important part of your Web site is the content you provide to your customers. The way it looks is (in the grand order of things) much less important.

"Designing and building with these standards simplifies and lowers the cost of production, while delivering sites that are accessible to more people and more types of Internet devices. Sites developed along these lines will continue to function correctly as traditional desktop browsers evolve, and as new Internet devices come to market."

– www.webstandards.org

When we design our pages using Web standards, there is also a positive impact on our businesses as well. Developing with Web standards speeds development, simplifies maintenance, and reduces bandwidth costs. However there is one caveat to designing with Web standards: we must give up the fiction that we can make all our pages look exactly the same in all browsers.

Somewhere in our minds is the idea that our Web pages should resemble printed magazines, with our layout pixel perfect and with everyone seeing our page exactly in the same manner. This fictitious Web nirvana is an unattainable goal and the sooner we can divorce ourselves from this assumption, the better off both we and all of our users will be.

In reality, Web pages offer all of us so much more than printed pages that if you really think about it, the fact that we restrict ourselves to mimicking a printed page is foolish (see Table 1 for the differences between print and Web).

Stop Using Tables for Layouts

One of the most limiting ways we design for the Web is to use tables for layouts. While tables do give us control over how our Web pages look, they cause many problems, including:

- Not all speech browsers can handle layout tables in a way that is not frus-

Printed Page	Web Page
One Medium – easy to control	Rendered differently depending on operating system, browser and version
Text sizes stay the same	Users should have control over the size
Nonusable by visually impaired	Availability of screen readers and speech browsers makes content available to a wide variety of people
Static – Never changes	Dynamic – Can be personalized to the user
Limiting	Limitless

Table 1: Print and Web page differences

trating for a user.

- Nesting tables complicates development and hinders maintenance.
- Nesting tables contributes to bandwidth bloat and causes browsers to load pages slowly since a browser cannot render a table until it is closed, thus slowing down the experience for the user.

To show the frustrations of a noncompliant table layout, consider Figures 1–3, showing how the sony.com Web site looks in a visual browser versus two speech browsers, pwWebSpeak and Simply Web 2000.

Web standards dictate that content be separated from presentation. Presentation means specifying how the content should be styled or placed. So while HTML (or preferably XHTML) is used for marking up the content, the actual presentation code should be in a cascading style sheet (CSS). (Note that in its strict format, XHTML requires CSS for all presentations.) To see how using this concept simplifies accessibility, let's look at Wired, a standards-based Web site, through the same three browsers in Figures 4–6.

The CSS 2.0 specification supports positioning and all the major, modern browsers do support it. (Microsoft's Internet Explorer has some peculiarities in its box model.) This is done using `<div id=""></div>` layers to position portions of code. Many users have given up on using this positioning in frustration since they could not get differ-



Figure 1: Sony.com's Web site in Internet Explorer looks really nice and is easy to navigate around.

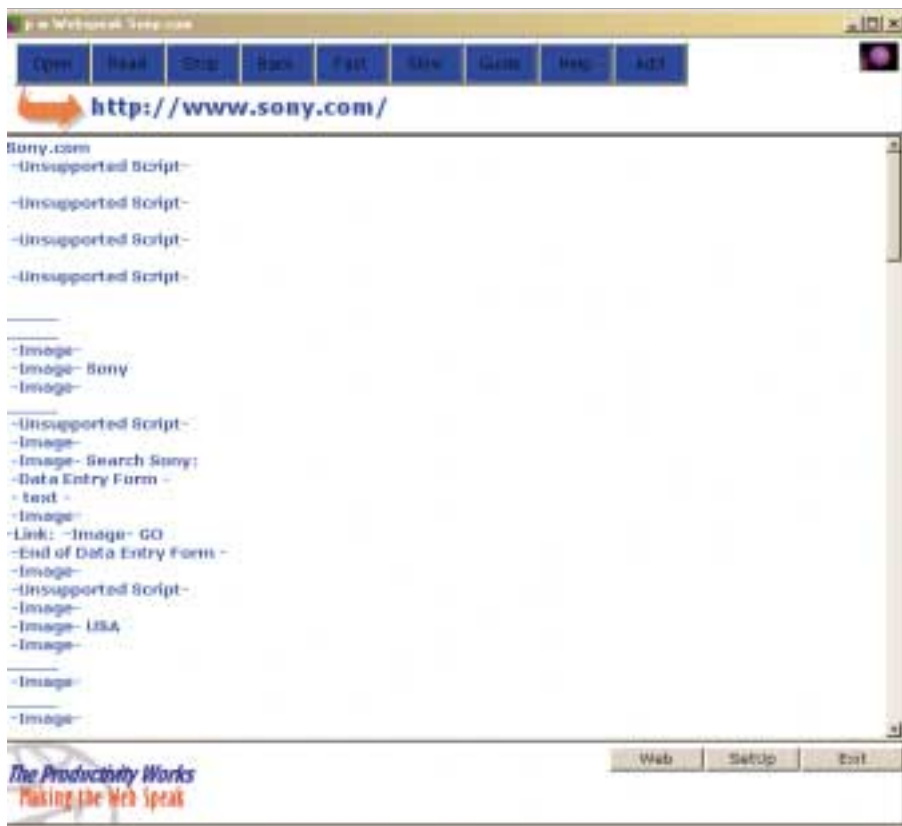


Figure 2: This shows the same Web site in pwWebSpeak. Notice how all the images have no words. It's harder to navigate through this Web site without the visual characteristics, but the site does flow in a logical manner through the tables.

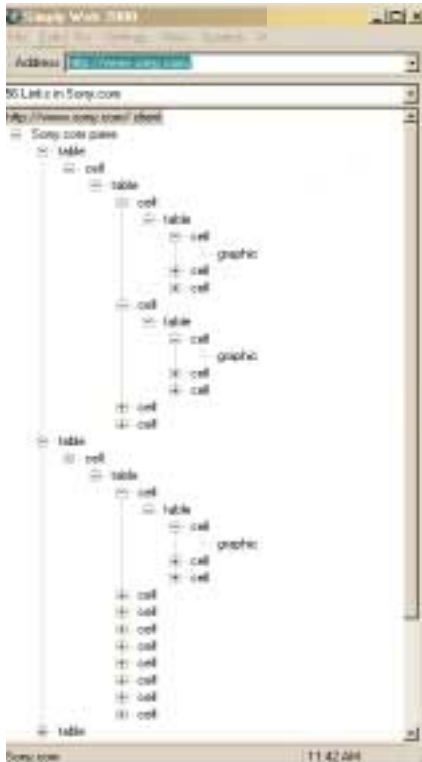


Figure 3: Here's the same Web site in Simply Web 2000. The table layout is literally rendered, making it almost impossible for someone who is blind to access this site.

ent browsers to render the same code in approximately the same way. The reason for this is many developers do not understand about DocType sniffing.

In order to deal with both Web standards compliance and with compliance for older code that relies on the quirks of individual browsers to render, the major browsers actually have two different rendering engines in them. The choice of rendering engine used is reliant upon the <doctype> that is chosen. Use of a specific <doctype> can force a browser to render the document in "quirks" or "standards" mode. No <doctype> in a document will automatically render the document in "quirks" mode. Although having a <doctype> has not been a requirement for valid Web sites, having a valid <doctype> at the beginning of your page does two things:

1. It allows a validator to make sure that your HTML markup is correct and valid for the type of document you are trying to create.
2. It allows a browser to decide specifically how to render your document.



Figure 4: wired.com's Web site in Internet Explorer is nice to look at and easy to navigate.



Figure 5: The same Web site using pwWebSpeak. Notice that due to the lack of images used in the HTML for spacing and backgrounds, as well as properly giving images a name indicative of their function, the page is much more readable than its sony.com counterpart.

CFUN4



301.424.3903
info@teratech.com

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

Sixth Annual **ColdFusion** Conference

June 26th & 27th, 2004

\$189

Per Person
Special Price until 12/31/03

Charlie Arehart
Jo Belyea-Doerrman
Vince Bonfanti
Ray Camden
Christian Cantrell
Sandra Clark
Robert Diamond
Michael Dinowitz
Steve Drucker
Matteo Foschetti
Shlomy Gantz
Mark Gorkin
Hal Helms
Simon Horwith
Jeff Houser

Chafic Kazoun
Matt Liotta
Tom Muck
Rey Muradaz
Samuel Neff
Jeff Peters
John Quarto
Derrick Rapley
Neil Ross
Kevin Schmidt
Michael Smith
Geoff Snowman
Jeff Tapper
Kevin Towes

Speakers from CFUN-03

www.cfconf.org/cfun-04/

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

-Phillip D



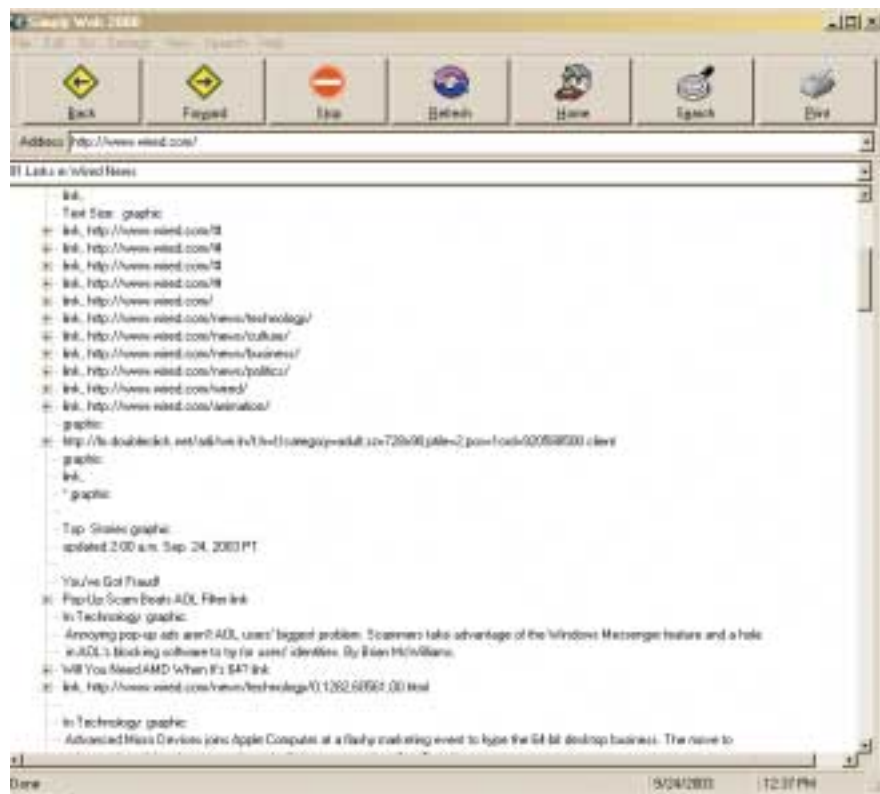


Figure 6: wired.com through the Simply Web interface. Notice the extreme difference. With the substitution of `<div>` tags for layout instead of nested tables, Simply Web is able to read the content to the user in a much more agreeable fashion.

I usually try for an XHTML transitional <doctype> so I have typically used the following <doctype> on most of my documents. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">. Note that <xml version="1.0" encoding="UTF-8"?>, which is required for validation, is missing, since putting that in there will force Internet Explorer 6 into quirks mode. This <doc type> will implement standards mode for all browsers except Safari. For a full listing of the <doctype>s and their associated browser modes, check out www.hut.fi/~hsivonen/doctype.html.

Using a layout without tables simplifies accessibility on many levels. By reducing the complexity of the page code itself, the ability to write code to the guidelines specified in Section 508 or the WAI is much more easily realized. To demonstrate, consider the following two code snippets.

Listing 1 shows a number of links (menuing system) in a typical tabled lay-

out. Each menu link has its own row and cell within a table (which is part of another table not shown here). Listing 2 shows the same code, but styled, using `<div>` tags and depending on CSS for positioning as well as styling (the CSS is not reproduced here for space purposes. The link listing uses the positioning capabilities (CSS-P) offered in the CSS 2.0 specification solution.

As you can see, the CSS-P solution requires much less HTML coding, is much more readable, and is easier to maintain. As a side benefit, browser rendering of the CSS-P is faster for two reasons: (1) there is less code to render, and (2) rendering of CSS-P can happen while the browser is receiving the information since it doesn't need to wait to have all the information downloaded before rendering, as is the case for nested table layouts. Although there is a learning curve associated with CSS, the benefits are fully worth the time spent learning it.

Keep in mind that tableless layouts use CSS-P which is a part of the CSS-2


specification. This means that only browsers that support CSS-2 will work with this type of layout. Supporting browsers include Mozilla, Netscape 7, Internet Explorer 6, Opera 7, and Safari. Netscape Navigator 4 will not support tableless layouts. However, at the time of this writing Netscape 4 has a less than 2% market penetration overall, so the decision to support a nonstandards-based Web browser is up to you.

Conclusion

This month we discussed using Web standards as a way of making sure our Web applications are readable in all types of speech browsers, without having to test each and every one of them. By using Web Standards and then applying the guidelines of Section 508 and/or the WAI to our pages, we ensure access on a wide level.

For those of you who are not convinced that CSS can render beautiful Web pages or standard layout schemes, here are a few links to take a look at:

- CSS Zen Garden: www.csszengarden.com/ (demonstrates what can be accomplished visually through CSS)
- Eric Meyers css/edge: www.meyerweb.com/eric/css/edge/ (demonstrates cutting-edge CSS layouts and effects)
- www.imaginaryworld.net/alt.html: Some CSS positioning designs, free for the taking

Next month we'll explore Fusebox 4 and how to use the new contentvariables attribute to capture content and stream to a CSS-P layout. For those of you who still need to support table layouts for Netscape 4, I'll show you a way to have your cake and display it in tables as well. 



About the Author

Sandra Clark, a Macromedia Certified Advanced ColdFusion developer, is a senior software developer with the Constella Group in Bethesda, Maryland. She has contributed material to the ColdFusion 5.0 Certified Developers Study Guide published by Syngress Media/Osborne McGraw Hill and is an author on Discovering Fusebox 4, by Techspedition, Inc. She has also spoken at various User Groups and ColdFusion user conferences around the country.

Listing 1

```
<table height="287" width="100%" border="0">
<tbody>
<tr>
<td width="100%" height="18">
Menu</td></tr>
<tr>
<td width="100%" height="18">
<a href="calendar.html">Training</a></td></tr>
<tr>
<td width="100%" height="34">
<a href="http://forms.html">
Forms & Procedures</a></td></tr>
<tr>
<td width="100%" height="50">
<a href="http://www.accommodation.html">
Accessibility Drawings and Plans</a></td></tr>
<tr>
<td width="100%" height="21">
</td></tr>
<tr>
<td width="100%" height="18">
Links</td></tr>
<tr>
<td width="100%" height="50">
<a href="acomlinks.htm">
Reasonable Accommodation Links</a></td></tr>
<tr>
<td width="100%" height="34">
<a href="access.html">
Accessibility Links</a></td></tr>
<tr>
<td width="100%" height="34">
<a href="employment.html">
Employment Links</a></td></tr>
<tr>
<td width="100%" height="34">
<a href="recreational.html">
Recreational Links</a></td></tr>
```

```
<tr>
<td width="100%" height="19">&nbsp;</td></tr>
<tr>
<td width="100%" height="17">
Recent News</td></tr>
<tr>
<td width="100%" height="28">
<a href="ZS.html">News Title</a></td></tr>
</tbody></table>
```

Listing 2

```
<div id="navcol">
<div id="nav">
<p class="menuheading">Menu</p>
<a class="menu" href="calendar.html">
Training</a>
<a class="menu"
href="forms.html">Forms & Procedures</a>
<a class="menu" href="accommodation.html">
Accessibility Drawings and Plans</a>
<p class="menuheading">Links</p>
<a class="menu" href="acomlinks.htm">
Reasonable Accommodation Links</a>
<a class="menu" href="access.html">
Accessibility Links</a>
<a class="menu" href="employment.html">
Employment Links</a>
<a class="menu" href="recreational.html">
Recreational Links</a>
</div>
<div id="news">
<p class="menuheading">Recent News</p>
<a class="menu" href="ZS.html">
News Story</a>
<p>&nbsp;</p>
<p>&nbsp;</p>
<p>&nbsp;</p>
</div>
</div>
```

Download the Code...

Go to www.coldfusionjournal.com

WHERE COLD FUSION EXPERTS HOST

Are you going to the Macromedia MAX conference?

Take the CFDynamics

MAX
Challenge

Every attendee who registers will receive
a FREE CFDynamics T-shirt!

REGISTER TODAY: www.cfdynamics.com/maxchallenge

REAL SATISFACTION. REAL SERVICE.

At CFDynamics, ColdFusion web hosting is our complete focus. You'll speak with a real CF system administrator every time you call. You'll also enjoy real benefits, not just promotional tidbits. With accounts starting at \$16.95/month*, our clients enjoy free SQL Server Access, free account setup, unlimited email accounts and a 30-day money-back guarantee. Real satisfaction. Real service. Forget about cold computer voices and confusing phone loops. Contact us today to find out why CFDynamics is the experts choice for ColdFusion hosting.

*see website for details

866-233-9626 | WWW.CFDYNAMICS.COM

CFDynamics

A Division of Connections, Inc.

The Ten Commandments 2004

A new, improved version

It's been about seven years since I first inscribed my "10 Commandments of ColdFusion Development" for my first ColdFusion book, and four years since they were last revised (yes, my Ten Commandments are not as omnipresent as their more famous namesake). ColdFusion has changed much over this time, as have the applications we're building and how we build them. And so this month, in honor of MAX, I present to you "The Newer and Even More Improved Ten Commandments of ColdFusion Development," or to adhere to Macromedia nomenclature, "The Ten Commandments 2004."

I

Have a Plan

We've all done it, and probably more than once. ColdFusion makes it so easy to start coding that there is often the temptation to start projects by firing up an editor and creating CFM files. That's a bad thing indeed. Nothing is more harmful to your development efforts than failing to plan properly, and you should be spending more time planning than coding, not less.

Planning involves thinking through every aspect of your application, from database design to UI considerations, from resource management to schedules and deliverables, and from feature lists with implementation details to language and presentation. You'd never build a house without detailed blueprints (well, you might try, but you'd never get the necessary permission to begin work), and building an application is no different.

I am constantly amazed by the number of applications that I am asked to look at that have no supporting documentation. And not just small development shops – I'm talking about some of the largest and most respected corporations too. Scalability problems? I would-



By Ben Forta

n't doubt it. I'd actually be amazed if such an app ever *did* scale. You cannot expect an application that grew to scale in spite of its developers. Nor can you expect it to be bug-free, manageable, or delivered on time.

Of course, some clients just want the job done.

Competition may require that you cut corners or lose the deal, and planning always seems like the logical place to cut. So what to do? You may opt to do as requested, recognizing that you'll probably end up with a dissatisfied client, and will thus be creating more work for yourself (or someone else) in the long term. Or you may opt to walk away (idealistic, and perhaps not always realistic). It's a tough call, but whatever you do, make sure that the client is fully aware of the implications of the demands and decisions made (and be sure to get clients to sign off on exactly what they want so as to protect yourself in the future).

II

Organize Your Application

An extension of planning your application is organizing it (along with any other applications). Applications are

made up of lots of little bits and pieces, and keeping them organized is imperative.

This includes directory structures and determining where common files should go, moving images to their own directory (or server), breaking long files into smaller, more manageable (and more reusable) ones, and even ensuring consistent organization among different applications.

The intent here is to make it easier for you (and the lucky soul who inherits the project) to locate code, isolate problems, reuse development, be able to replace or update components without risking breaking things...you get the idea. Understanding application design should not require perusing extensive code listings to understand code blocks and their relationship to each other; application organization must be logical and clear and ideally intuitive too. Going back to the prior Commandment, "Have a Plan," all organization should be documented in detail as part of that plan.

III

Set Coding Standards

This is an interesting one, and one I get asked about often. Macromedia has not published formal recommendations on coding standards, nor in my opinion should they. Macromedia's job is to create killer tools and products for us developers; our job is to use them as works best for us. I don't believe that a single set of coding standards would work for all developers, but at the same time, I don't believe any developer should be writing code that does not adhere to a standard – *any* standard.

Coding standards include everything from file- and directory-naming conventions, to variable naming conventions, to code organization and ordering within your source code, to error-handling, to componentization, and much more. (For example, if all variables that con-

tain dates begin with “dt”, then references to a variable named “dtOrderDate” become very self-explanatory.)

The purpose of coding standards is to ensure some level of consistency in your code. Whether it is to allow other developers to be able to understand and work with your code, or whether it is simply so that you’ll know what the heck you did (and why) six months down the line, coding standards provide a mechanism to create code that describes and explains itself.

There is no right or wrong coding standard as long as it is used. The only thing wrong about coding standards is not using one. (*Note:* To get you started, you may want to take a look at Sean Corfield’s Coding Guidelines at www.corfield.org/coldfusion/coding_standards/.)

An extension of this is the use of application methodologies and architectures. The purpose of these is to encourage good design and the use of best practices. I am not going to comment on the virtues or failings of specific methodologies or architectures; there is no right or wrong here either.

Don’t be swayed by hype and trends and acronyms. What is right for someone else may not be right for you, and what is right for one of your applications may not be right for all your applications. Any architecture should help you write better code. If you find one that works for you, great; if not, come up with something yourself, or adapt something you find. Either way, just do it.

IV

Comment Your Code

This is an obvious one, but apparently few of us have the time to pay attention to the obvious. So, I’ll say it once again, *all code must be commented*. (For the record, I’d fire an employee on the spot for turning in code that is not commented, that’s how serious a sin I believe this one to be.)

Every source code file needs a descriptive header containing a description, the author info, creation date, chronological list of changes, any dependencies and assumptions, and any other relevant information. In addition,

every conditional statement, every loop, every set of variable assignments, and every include or component reference must be commented with a simple statement explaining what is being done, and why.

It’s a pain, I know. But the next time you (or anyone else) has to work with the code you’ll appreciate the effort immeasurably. And you might even be able to safely make code changes without breaking things in the process.

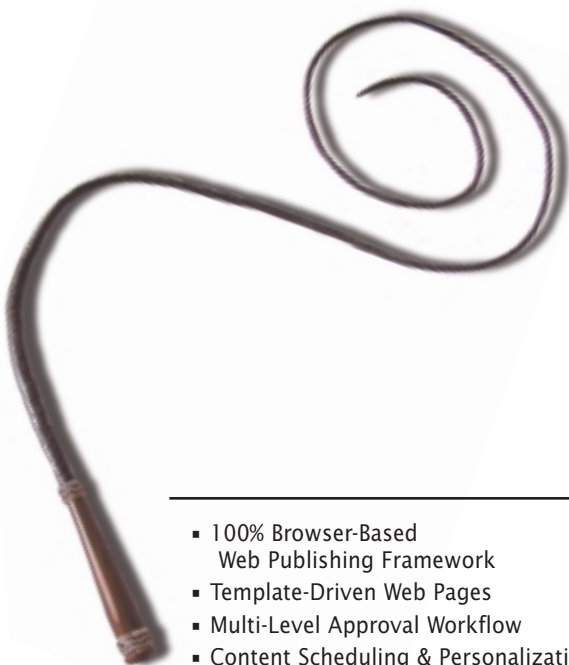
V

Functionality First, Then Features

Yet another obvious one, and a common beginner’s mistake. Yes, creating whiz-bang UI effects may be far more fun than writing business rules and data-entry validation routines, but the latter are far more important to the success of your application. Concentrate on creating a complete working application, then pretty it up as needed.

But at the same time, don’t neglect the user experience. Some of the best applications ever created failed (and

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



CommonSpot™ Content Server.

Put content management in the hands of content owners.

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

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

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

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

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

Paper Thin

800.940.3087
www.paperthin.com

continue to fail) because they were a pain to use. While it is a mistake to start with menus and color choices and application screen flow, it is an even bigger mistake to fail to leave adequate time for these.

It's about sequencing and balancing. Do so and increase the chance that you'll finish on schedule for a change. The final result might not be as cool as you'd like, but there is something to be said for an application that actually works, even an uncool looking one. Furthermore, (as explained in the next Commandment) it is very difficult to debug logic problems when the code is cluttered with fancy formatting and features.

VI

Build and Test Incrementally

You'd be amazed (or maybe you wouldn't be) by the number of e-mail messages I get asking me to help debug attached files – attached files with hundreds of lines of code, often more, and often multiple files all needed to make the application work. My standard response to these messages is “Yes, I'll help you debug your code, but first narrow it down to just the few lines in question.” Not surprisingly, many developers find that the process of narrowing down the problems allows them to diagnose their code themselves.

Being sent the messages and requests isn't what bothers me (I know I am going to regret saying this, but I really do not mind those at all). What really bothers me is that what becomes apparent is that core code was never tested in isolation. This goes back to the prior Commandment, “Functionality First, Then Features.” And the same is true for testing.

When you develop core components of your application, test them. Write little test routines, hard code, or smoke and mirrors as necessary, but however you do it, do it. Obviously you'll have to test your complete application when done, and some problems won't come to light until then, but the more you can test code blocks in isolation, the better.

VII

Never Reinvent the Wheel, and Plan Not To

This is one I have written about extensively, especially in this column. Write code with reuse in mind, and reuse code whenever possible. When designing your code, put the extra time in up front to make sure it is not hard-coded or highly task-specific unless it absolutely has to be. Make sure code can stand on its own two feet whenever possible, never make unnecessary assumptions about where code is being used and how, and never reference explicit scopes or variables outside of your own code.

The benefits? Being able to reuse existing code will shorten your development time. You'll also stand a far greater chance of creating bug-free code when you use components that have already been used and tested. Plus, if you do make subsequent fixes and corrections, all code that uses the improved components benefit.

ColdFusion developers have lots of reuse options, and should pay particular attention to ColdFusion Components (introduced in detail in a two-part column in *CFDJ*, Volume 4, issues 6 and 7), which both help and encourage tiered application design (see *CFDJ*, Volume 3, issue 8, and Volume 4, issue 10), code reuse, and all-around better coding.

Lots of benefits, and no downside whatsoever. Should be a no-brainer. Enough said.

VIII

Use All the Tools at Your Disposal, Not Just ColdFusion

This is an extension of the previous commandment, and another one I have written about before (see *CFDJ*, Volume 1, issue 3, “Take Your Database Out of Retirement,” and *CFDJ*, Volume 2, issue 3, “When NOT to Use ColdFusion”). Unlike all the other Commandments in this list, this one is more ColdFusion specific.

ColdFusion applications are usually not standalone entities. They rely on database servers, mail servers, and much more. In addition, ColdFusion can leverage Java, Web services, COM, CORBA, and

C/C++ code. Use these tools, as many as needed, and always attempt to pick the best one for a specific job. The best ColdFusion applications are not the ones written purely in ColdFusion, but the ones that leverage the best technologies for the job, all held together by ColdFusion. And the worst ColdFusion applications are the ones that try to go it solo.

The truth is, even beyond your ColdFusion application development, if you are serious about application development and a future in this industry, then it's in your best interest to not be a one-trick pony. Diversifying your skills will make you a better and more valuable developer, and will also improve your ColdFusion applications in the short term.

IX

Respect (and Fear) Production Servers

There are two very different aspects to this one.

The first is obvious, or so you'd think. But any time I bring this up in front of a group of CF developers, the grins, sheepish looks, and knowing glances convince me that there are transgressors in our midst.

All development and testing *must* occur on servers established for just that purpose. Yes, this means you'll need additional hardware or the installation of ColdFusion Enterprise (so as to have multiple instances of ColdFusion to work with), but the extra cost is nothing compared to the cost of bringing down your application because that little change was not as little as you expected.

Write your code, test it, debug it as needed, deploy it to a testing server, test it some more and some more, and then finally, deploy it to your live production server. And don't repeat this process too often. Instead of uploading slightly changed versions of your application every day, collect the changes, test them some more, and deploy them monthly or weekly, or whenever works best for you.

That's the respect part. As for fear, production servers are vulnerable, they are necessarily publicly visible, and they are necessarily used and accessed by all

sorts of people (including many of whom you may otherwise choose to have nothing to do with).

A little paranoia is healthy when it comes to public-facing servers. Assume that your server will be compromised, and that whatever is on it (and whatever it has access to) will be stolen or tampered with at some point. This has very practical implications – from never embedding passwords in source code, to not keeping databases on your Web server, to not using accounts with access to lots of other resources.

I'll state it again, just to be perfectly clear about this: anything that is on your server, and anything that your server has access to, will be stolen or compromised at some point. Be frightened, be very frightened. And once you have come to grips with your fear, be mindful and cautious.

The key here is that your production server is sacred. Don't touch it at all unless you have to (and the less frequently the better), and never, ever, make changes on it, even minor ones. In addition, don't leave your valuables exposed, or put differently, put nothing on your server that you'd not want in the hands of others.

X

Keep Things Simple

Simplicity is a good thing, complexity usually isn't. This affects everything from user interfaces to database schemas to documentation to application architecture. Keep it simple. This has obvious implications for user interfaces, application flow, data-entry screens, and error messages. But it goes way beyond these too.

At the risk of upsetting hardcore extremists, this is even true of application architecture and methodologies. I have seen relatively simple apps (apps that should have taken days to develop) turn into behemoth projects because of an insistence that everything be an MVC application. Not that I have anything against MVC, but we don't live in a one-size-fits-all world. If a simpler design is effective, that may be good enough.

Similarly, while it would be wonderful to be able to clearly articulate a black-and-white distinction of what should be written in ColdFusion and what should be written in Java, the reality is that there

are shades of gray. Sometimes the less pure solution may be the simpler one. There is no right or wrong here, but there is a lot to be said for simplicity.

Conclusion

There you have it – my new, and greatly improved, Ten Commandments of ColdFusion Development. Of course, you may completely disagree with my list, in which case I'd love to hear what you'd change, and why.

Good luck with your coding, and I look forward to meeting you in person at MAX in Salt Lake City, UT (November 18–21).



About the Author

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

ben@forta.com



HostMySite.com

Built for ColdFusion Pros by ColdFusion Pros

plans from
\$8.95 / mo.
and ~~FREE~~
Domain Name

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

Visit www.HostMySite.com/cfdj for:

3 Months Free
and FREE Setup on Any Hosting Plan

call
today

877•248•HOST
(4678)

Harnessing the Magic of the `<gleep>` Tag

Do some powerful things with very simple code

What the bleep is a “gleep” tag? Now I know you’ve never heard of it, but then neither have any browsers. That’s where the magic comes in.



By Bob Siegel

From the dawn of the Internet, it has been a prime directive of all browsers to ignore any tags they do not recognize, and that is what the `<gleep>` tag depends on. You can have all sorts of “tags” on your HTML pages to use for all sorts of things, and they will never be visible in the browser window. And simple ColdFusion functions allow you to easily do some powerful things.

Case in Point

One of your sites has a page with a certain section that must be periodically updated with new or revised text. Your boss or client wants a clerical person (who is paid far less than the regal hourly rate you make for programming) to be responsible for doing this work. Now, before you scour the Web for some CFX or package that can make this happen without the nontechie screwing it up, please realize that as programmers, we can “invent” a solution rather than buy one. (That’s what we do!)

All we need to do is put up a form for someone to enter the new content. Using the code below will simply and efficiently replace the old version of the text with the new text from a form where the user inputs the new text or edits the existing text. All that’s needed is that the page being updated has the editable text area bracketed with `<gleep>` tags.

The code immediately below would appear in the form that shows the present content of the editable area for possible revision:

```
<CFFILE Action = "READ" FILE = "#ExpandPath('mypage.htm')#" VARIABLE = "pagecontent">
```

```
<cfset pagecontent = Replacenocase(pagecontent, "<gleep>", chr(7), "ALL")>
<cfset editablearea = ListGetAt(pagecontent, 2, chr(7))>
<TEXTAREA NAME="editablearea">#editablearea#</TEXTAREA>
```

This code (in the form action page) writes the new version back to the edited page:

```
<CFFILE Action = "READ" FILE = "#ExpandPath('mypage.htm')#" VARIABLE = "pagecontent">
<cfset pagecontent = Replacenocase(pagecontent, "<gleep>", chr(7), "ALL")>
<cfset pagecontent = ListSetAt(pagecontent, 2, form.editablearea, chr(7))>
<cfset pagecontent = Replace(pagecontent, chr(7), "<gleep>", "ALL")>
<CFFILE Action = "WRITE" FILE = "#ExpandPath('mypage.htm')#" OUTPUT = "#pagecontent#">
```

Now, by adding only five gleep-related statements, you’re all done! Is that not magic?

What’s happening is that the replacement of the gleep tags with any convenient character will never appear in the actual page text. (I use `chr(7)` because in ancient times, `chr(7)` was what made your computer emit a beep and “beep” rhymes with “gleep”.) This creates a CF list with three elements, one before the `<gleep>`, one between the two `<gleep>`s, and one after the second `<gleep>`. You can then conveniently use all the nice CF list functions.

Then, this `chr(7)` is used as the list-delimiter for a `ListSetAt()`. We then put the `<gleep>`s back so they will be there for next time, and so they will be ignored by browsers. This also allows your users to do it again and again until they get it the way they like.

You might also thoughtfully add

```
<CFLOCATION URL="mypage.htm" ADDTOKEN="No">
```

to shoot the user over to the page for viewing after the change is made.

Cascading <gleep> Tags

No rocket science here, but if you have a series of things that must be added to or edited separately, you can just have a series of <gleep>s and replace them as needed with a series like:

```
<cfset pagecontent = ListSetAt(pagecontent,2,form.newtext,chr(7))>
<cfset pagecontent = ListSetAt(pagecontent,3,form.newtext,chr(7))>
<cfset pagecontent = ListSetAt(pagecontent,4,form.newtext,chr(7))>
```

and so forth.

You can see an example of this at www.AccessNewAge.com/BassStation/. Of course you need to do a "View Source" to see them.


Note that there is no such thing as a </gleep> tag, as it would make the coding more complicated than we want to bother with – and we certainly want to keep everything simple!

Multiple <gleep> Tags

If the "cascading" form is not powerful enough for you, the "multiple" form can be used. On one site I did, every month the resident astrologer has to create a new horoscope page with a blurb for each of the 12 astrological signs. The strategy is almost the same, with the difference being that there are 12 pairs of <gleep> tags: <gleepAries>, <gleepTaurus>, etc. There's also a "general" blurb for each month.

The code looks like this :

```
<cfset signs =
"General,Aries,Taurus,Gemini,Cancer,Leo,Virgo,Libra,Scorpio,Sagittarius
,Capricorn,Aquarius,Pisces">
...
<cfloop index="sign" list="#signs#">
  <cfset PageContent =
    replacenocase(PageContent,"<gleep#sign#>",chr(7),"ALL")>
  <cfset SignContent = trim(evaluate("form.gleep#sign#"))>
  <cfset PageContent = listsetat(PageContent,2,SignContent,chr(7))>
  <cfset PageContent =
    replace(PageContent,chr(7),"<gleep#sign#>","ALL")>
</cfloop>
```

Now that you understand the principles of <gleep>-ing I'm sure that you clever readers will think of things to do with them that I haven't. Please let me know what you come up with! We can do some powerful things with very simple code. I like that. 

About the Author

Bob Siegel has been doing full-time computing work since 1969 and Web work since 1995. He founded the NYPC CFSIG and ran it for several years. Bob is a frequent speaker at ColdFusion meetings and conferences.

Bob.Siegel@verizon.net



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

See what's new at FuseTalk.

Visit us at booth 303 at "MAX 03" in Salt Lake City, November 18-21



Or to learn more! www.fusetalk.com 1.866.477.7542

Working in a Distributed Development Team

Good written communication is paramount

Think of the problems you encountered during your last project, then imagine the team spread across different floors or buildings, or even continents! The problems and pitfalls are greatly magnified to a point where development is severely hindered.

This is a typical distributed development team. The personnel involved in the project are unable to have direct face-to-face communication. I will discuss common problems that can be encountered when working in this environment and talk about solutions that will greatly improve project development.

First, let's start by defining an example team. A global project has been proposed and the only problem is that some members of the team are based in the UK, and some in the U.S. This is not only multi-site it is also intercontinental, where time of day becomes a big issue. Figure 1 shows the organizational structure of our development team.

The distribution of the staff means it is both difficult and expensive to all meet face-to-face, so solutions need to be found to facilitate a good working environment.

Definition and Design

The key to any project is good communication. It is absolutely vital that everybody knows what's going on, and that people can find out what they need to know. There are two key areas where communication is vital – within the company and between development teams.



By Gavin Brook

Business is the driving force behind any project. A project is not even started unless there is benefit to the company, so it is up to the business to define what it actually wants and expects. Documentation is the core of a successful project. Without clearly defined goals and expectations, a project can begin to shift scope and floun-

der very early on in the development cycle.

For development to begin, a functional specification must be written. This defines the scope of a project – its goals and expectations. By writing this document in a nontechnical language, everybody involved in the project knows what is required. A poorly written specification means that there is a chance that somebody will misinterpret the requirements and unexpected results are assured. This not only causes friction between the business and development teams, but it can also jeopardize the project time line, as changes will inevitably need to be made.

The next document is the technical specification. This takes the functional specification and builds upon it, defining how the goals and expectations will be

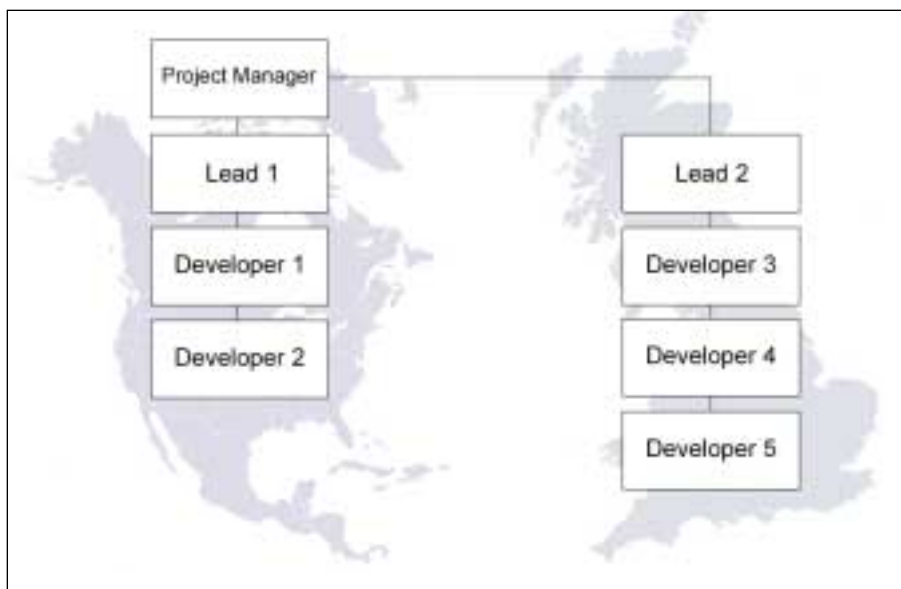


Figure 1: The organizational structure of the development team

met. This is written in a language that clearly defines how the application will work, and is in enough detail for the development team to understand its implementation. By sitting down and thinking the requirements through, the development cycle can be defined and documented.

An important step to take after the creation of these two documents is sign off. The appropriate staff from key parties involved will review these documents and agree if they are a solid definition of the project. Without this sign-off step, when the project is further down the line and people's expectations have changed, chaos can ensue.

Let's take our global development team we talked about. Without solid documentation there is no standard, and each team will take a slightly different course. This can mean that the final outcome will be vastly different than what was initially envisaged. With good documentation, the two teams can work in unison and meet the project's requirements.

Communication

Communication becomes even more important when different facets of the project are being worked on independently. We have already discussed specifications and documentation, and these go a long way to facilitating communication. But there should always be an open forum for the exchange of ideas.

One of the most common things that has to be communicated is change. During the life cycle of the project, technical issues and changes in expectations arise that need to be accurately documented and communicated to everyone. There are many tools available that facilitate this. Personally, I use a very simple application that was developed in-house to track changes and issues. These applications can track the life cycle of all issues and distribute details to the appropriate people. This provides full auditing of the documentation and a history of changes.

I feel it's important to mention that no matter how helpful these tools are, without good data, they can become a hindrance. For the most part, changes should be written in a high-level language so that everyone can easily understand what has changed. With technical reasons it is not always possible to break this down to easily understandable terms. Even so, I find it is good practice to always try to make it easy to read, as change affects everyone, including nontechnical staff.

The second line of communication is interdeveloper. In the case of our example team, the developers are split over two continents, in two different time zones. This means that the two teams may not be working at the same time, so good written communication is paramount. Not all projects are spread so far apart, but the principles apply equally to both.

A development plan is a definition of each part of an application, and a breakdown of how it is achieved. For instance, in the case of a FuseBox project, the application is broken down into a series of files containing displays, queries, and actions. A development plan for these types of projects will consist of a list of the application sections and their component files. Another benefit of producing a well-defined project plan is that it also gives a very good indication of the size of a project, and hence the development time.

From a clear development plan it is easy to define tasks and set priorities. With two or more separate teams, it is important that everyone knows what they need to do and when they need to have it done by. Collectively deciding task workloads and

completion dates help plan out the whole application ahead of time. It is quite common for one team's work to depend on the other's. I find that leaving some less critical tasks unassigned allows developers to continue working in the event of any unexpected deviations from the plan. A complete plan allows the project to continue moving forward in the event of deviations or technical issues. I will discuss the former in the next section.

Time to Start

When the development process begins, there are many things developers can do to help the project along, other than writing code. One of the most important is code commenting. By accurately commenting the code, any other developer currently working on the project (or in the future) can work out what different parts of the code do. A useful standard to use is FuseDocs. Although designed specifically to work with FuseBox applications, the principles can be used in others.

A FuseDoc is basically structured text, based on XML, that fully describes the file. The packet used in FuseDocs starts by defining the name of the file and what version of FuseDocs is being used (see Listing 1). Full details of XML are beyond the scope of this article, but there is a wealth of information available in books and on the Internet (for example, www.fusedoc.org). The next few lines in the packet describe the purpose of the file. It is written as though you are the page and you are describing your function. It has been designed this way so that everyone easily understands even though they are not the original developer.

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

Java for ColdFusion Programmers?

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

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

The final section describes the inputs and outputs of the file. These are what are required to make that file function correctly within the application. By defining these, anyone can find out what is going into the file, and what is expected to be left out. A typical FuseBox application development cycle usually includes a cycle of just writing the FuseDocs for the application so that all the developers know how the application will flow. This is only a brief overview of FuseDocs, but there are many articles available about this and FuseBox.

Version Control

Version control is a system whereby all changes to files are tracked and stored as different versions. These may be as unsophisticated as making several copies of the file, or automatically tracked by a dedicated application. I prefer the latter as it saves disk space and keeps an audit trail of what was changed and by whom. There are many different packages that perform this function, and these are available as either open source or commercial.

For the open source route, a popular version control system is CVS. This can be downloaded freely (www.cvshome.org), and is a very comprehensive system. It's flexible and has more features than most people will need. The only real downside is that it is driven by a command line. This makes it quite cumbersome to use and unfamiliar to people used to graphical interfaces. There are other open source projects that provide graphical interfaces, and these can be quite varied in their operation. If you have the time, download a copy and try it out.

Commercial packages tend to have richer graphical interfaces, and can be

much easier to use. PCVS is one such package that is commonly used. This provides client and Web interfaces, and has the added benefit of support. It is similar in functionality to CVS, and as such is quite comprehensive. Another package is SourceSafe. This is produced by Microsoft and is usually included in all of its development environment distributions.


All these packages have several things in common. They all support multiple users, archive changes to each version, and have the ability to mark a file "is locked". This means someone is working with the file and the software will not allow you to enter changes unless you are the person who locked it. This is particularly important when several people are working on a project. Once the correct changes have been made, the file is checked back and unlocked. Another major benefit of these systems is that they allow you to roll back to previous versions. This means if a mistake is made, the previous version from the archive can be retrieved. These benefits alone mean that version control should be a key part of any project.

Working Practices

At the end of the day, all the technology in the world is not going to help if it's never used. That's why establishing good working practices is important. Just as I have explained earlier, communication is the basis of good practice. Imagine our example team again – on two sides of the planet, meaning as one team is almost finishing for the day, the other is beginning. What if one team didn't report their progress or problems they'd found? The team relying on that report to continue work is left not knowing where to start, and cannot continue with their own

work. It's not a showstopper; they could probably work out where the other team got to, but it would cost valuable time.

There are several methods of communication available that enhance these best practices and allow a feeling of solidarity. E-mail is probably the most important, as it's available to almost everyone these days. Instant messaging can be extremely beneficial, especially if the teams are working at almost the same time. When used for business purposes it can save considerable amounts of time, as most people will answer an instant message more quickly than an e-mail. Conferencing is another key practice that should be established, and is easily done through a variety of systems. The ability to talk to many people remotely is invaluable, allowing a "virtual" meeting. These systems can also come with whiteboard facilities, as well as application sharing. The latter allows you to demonstrate applications collectively, whereby everyone sees what you are doing with an application.

Whatever the structure of your team, these concepts apply to all. The key to success is communication. Without it, these projects are going to be severely hindered, or even doomed to failure. 

About the Author

Gavin Brook works as a technical lead for Xerox Corporation. He has worked on many large-scale projects in ColdFusion and other languages, and has been designing and building enterprise applications in ColdFusion for many years. Gavin is currently working on a distributed development team.

gavin@gavinbrook.com

Listing 1: An example of a FuseDoc XML packet

```
<!--
?xml version="1.0" encoding="UTF-8" ?>
"http://www.officeprinting.xerox.com/smart/dtd/fusedoc.dtd"
<fusedoc fuse="qry_updateDatabase.cfm" language="ColdFusion" specification="2.0">
  <responsibilities>
    I update the database with either the edited details or the new data.
  </responsibilities>

  <properties>
    <history author="Gavin Brook" email="gavin@gavinbrook.com"
date="20030812 11:10 AM" role="architect" type="create"/>
  </properties>
```

```
</io>
  <in>
    <number name="data_id" format="CFML" Scope="attributes"/>
    <boolean name="enabled" format="CFML" Scope="attributes"/>
    <String name="some_text" format="CFML" Scope="attributes"/>
    <String name="more_text" format="CFML" Scope="attributes"/>
  </in>

  <out>
    <String name="DB_Status_Code" format="CFML" Scope="variables"/>
    <String name="DB_Message" format="CFML" Scope="variables"/>
  </out>
</io>
</fusedoc>
-->
```

Download the Code...
Go to www.coldfusionjournal.com

International Conference & Expo

Edge 2004 EAST

Development Technologies Exchange

February 24-26, 2004

Hynes Convention Center, Boston, MA

**ARCHITECTING JAVA, .NET, WEB SERVICES,
OPEN SOURCE, AND XML**

Addressing:

- ▶ Application Integration
- ▶ Desktop Java
- ▶ Mobility
- ▶ ASP.NET
- ▶ VS.NET
- ▶ JavaServer Faces
- ▶ SOA
- ▶ Interoperability
- ▶ Java Gaming

Register By
November 21, 2003

Up To
SAVE \$400



Hynes Convention Center
Boston, MA



For more information visit
www.sys-con.com
or call
201 802-3069

Over 200 participating companies will display and demonstrate over 500 developer products and solutions.

Over 3,000 Systems Integrators, System Architects, Developers, and Project Managers will attend the conference expo.

Over 100 of the latest sessions on training, certifications, seminars, case-studies, and panel discussions promise to deliver real world benefits, the industry pulse and proven strategies.

Full Day Tutorials Targeting

- Java • .NET • XML
- Web Services • Open Source

Conference program available online!
www.sys-con.com/edgeeast2004

Contact information: 201 802-3069 • events@sys-con.com

component.cfc

The mother of all components

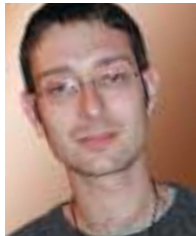
Yes, that's right, components have mommies and daddies. Well, not really, but that statement isn't too far from the truth. This article explains "component.cfc", a hidden gem in the ColdFusion Component architecture that is part of ColdFusion MX.

By now, everybody knows that the introduction of the component framework in ColdFusion MX was one of the most radical changes to CFML since its inception. Their introduction was significant for three primary reasons. First, they allow developers to expose functionality to the Flash Gateway and to any other Web service-enabled environment. Second, they offer a level of abstraction far superior to that of custom tags and user-defined functions. The third reason CFCs have made such an impact on CFML is that they offer ColdFusion developers the ability to implement object-oriented features.

These object-oriented features include new variable scopes with varying degrees of exposure outside the component; the "packaging" of functionality, introspection, and the generation of meta data; object-instance-based programming; and inheritance. In traditional object-oriented programming languages, all objects inherit from a parent "Object" class. What many developers don't realize is that components, too, have a parent "class". Its name is "component.cfc" and it can be a very valuable tool when developing applications.

Where Can I Find component.cfc?

If you create a CFC that does not inherit from any other CFC and you view its meta data, you'll notice that the value for the "hierarchy" meta information defaults to "WEB-INF.cftags.component". You will find this file in the "CfusionMX\WWWRoot\WEB-



By Simon Horwith

INF\cftags" directory if you're running the standalone version of ColdFusion, and in "JRun4\servers\{your_server_name}\cfusion-ear\cfusion-war\WEB-INF\cftags" if you're running the J2EE version (on top of JRun4). You can view the meta data for component.cfc on a server by browsing [http://\[server_domain\]/CFIDE/componentutils/cfcexplorer.cfc?method=getcfcinhtml&name=WEB-INF.cftags.component](http://[server_domain]/CFIDE/componentutils/cfcexplorer.cfc?method=getcfcinhtml&name=WEB-INF.cftags.component), replacing "{server_domain}" with "localhost:8500" or whatever other URL you use to browse to that machine.

What Does component.cfc Do?

Well, nothing really. Not out of the box, anyway. The component.cfc file is empty by default in ColdFusion MX 6.1. In ColdFusion MX it had a tiny bit of encrypted data at the top of the file. As already discussed, all components inherit from component.cfc by default. Think of it as the Application.cfm of components – a place to put code that is implicitly included. The difference is that the code outside of any method in component.cfc is executed only when a ColdFusion Component is first initialized, and that its methods and meta data are inherited by other components. It gives you a single place to put code that must be common to all CFCs on your server.

Why Would I Want to Do That?

Having one place to put functionality and initialization code that will be available to all server components has two important uses. As already mentioned,

you can think of it as similar to Application.cfm, which means that if you're running a server that's dedicated to hosting one application and the application is using CFCs for its business logic (as every application should be), it's a very good place to put functionality that all components need to have.

For example, if your server is hosting your company's public Web site, all of your components might need to have access to the current company catalog of products. Creating a method in component.cfc that retrieves the current catalog information from a database and returns it to the caller would ensure that all components have the ability to retrieve the catalog by calling that method. While this is a neat trick, it's not always terribly useful or desirable, especially when your application is running on a server that hosts other completely separate ColdFusion applications.

There's another use for component.cfc though – when wielded properly, it can be a powerful development tool. More than anything else, I like to put debugging methods in my component.cfc file so that they are available in any application I happen to be working on. Extending the component framework with development tools can make a developer's life a little bit easier, as we will soon see.

Coding with component.cfc

It's a little known fact that ColdFusion Components don't technically require a start and end <cfcomponent> tag... they will work without them. That said, without the <cfcomponent> tag, a CFC will not properly inherit from the base component.cfc file nor will it allow the use of <cfproperty> tags if you are defining meta information, so as a best practice, follow the standard recommended practices when creating components.

By default, component.cfc is an empty file. Because it does not inherit from any other component (after all, it is the mother of all components), you can code methods in the file without adding any <cfcomponent> tags. You can optionally add <cfcomponent> tags to component.cfc in order to generate meta data about the component or to enable the use of <cfproperty> tags before your methods. Like inheritance with other CFCs, any meta information, methods, and variables in the component are inherited by all other components on the server.

As you may have guessed, creating a method in one component with the same name as a method in component.cfc will override the method. The original method

defined in component.cfc can be accessed with the “super” keyword as “super.methodName()”, provided that the component you are in does not inherit from any other component. Let’s examine one very practical use for this master component file.

Component.cfc Uses

If you’re anything like me, you use <cfscript> a lot. As wonderful as <cfscript> is, it is not without limitations. In particular, the use of tags is strictly forbidden within a <cfscript> block. It is often necessary during the development process to view data in order to determine what’s happening in your code. The <cfdump> tag is perfect for this, but wait – you cannot <cfdump> within a <cfscript> block. What to do?

Developers like me who use <cfscript> often, have made it a standard practice to create a function using <cffunction> that accepts a single argument of any data type, <cfdump> that variable, then <cfabort> the page. That function can then be called from <cfscript>, thus giving developers the ability to “see what’s happening” within their <cfscript> code.

Here’s the code I wrote inside of my component.cfc file so that I can dump a value from any ColdFusion Component <cfscript> block:

```
<cfcomponent displayname="Root" hint="Root CFC">
    <!--::: function to dump then abort ::-->
    <cffunction name="dumpIt" output="true" displayname="Dump It" hint="A
debugging tool" access="private" returntype="void">
        <cfargument name="it" displayname="It" hint="The Thing To Dump"
required="true" type="any">
            <cfdump var="#arguments.it#">
            <cfabort>
        </cffunction>
</cfcomponent>
```

You’ll notice that the method is private because I do not need nor want to expose it to cfm pages that are instantiating my components. It also means that the method does not display along with the other methods of a component that’s inherited it when that component’s meta data is browsed.


The “dumpIt()” method is useful for developers who do not use <cfscript> as well. It is generally considered a best practice to encapsulate business logic with components. This means passing a value to a component method, operating on that value, and returning a value to the caller page or application. With this in mind, it’s a safe assumption that the components in your application(s) aren’t directly outputting anything to the HTTP output stream.

Pretty much every component I create not only uses <cfscript> heavily, but never outputs anything. To make sure nothing is output, I always set my <cffunction> tag “output” attributes to “false”. This is definitely a best practice in most scenarios, but not being able to output directly from a component method also means you cannot <cfdump> the value of variables within the component when you’re debugging your application. Once again, the “dumpIt()” method is a very handy tool, as from within a “silent” component, you can pass the suspicious variable to this method (which has the “output” attribute set to “true”) in order to see what’s going on in your code.

Where Do We Go from Here?

Component.cfc is a very practical debugging tool for developers and can also be a powerful part of your applications as well. We have looked at just one way that it can be used as an

effective development tool. You could create variations of the “dumpIt()” function so that you have one method to dump and abort and another to dump and allow processing to continue. It wouldn’t be difficult to add methods to your component.cfc that allow you to do other useful things such as time blocks of code using the getTickCount() function, for example.

Be forewarned that there are two major drawbacks to placing code, especially application rather than debugging/development code, in component.cfc. One is that from an architectural point of view, you are physically separating your code from the rest of your application business logic. The other is that you are making your code less portable – it’s no longer a simple matter of copying a directory structure from one server to another in order to move your application when some of the logic is housed in component.cfc. Other than that, have fun and take advantage of it! The sky’s the limit. 

About the Author

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

simon@horwith.com

Don't Miss CFDJ's Next Issue!



Flash Application Development Guidelines

Here's how to exploit the power of Flash MX and ColdFusion to build the most professional, usable, efficient, powerful, and quality solution on the Web.

MX to iSeries Demystified

How to connect ColdFusion MX to an IBM iSeries (formerly AS/400) DB2 database.

The Trouble with Macs

A custom tag helps tackle problems with comparing strings and uploading files.

Web Site Accessibility

An exploration of Fusebox 4 and how to use the new contentvariables attribute.

Design Patterns in ColdFusion: Creational Patterns

Part 2 expands on the topic of object creation and offers several Creational pattern examples.

Unnatural Acts

Seven common practices that lead to failure... and their remedies

Ah, television! Where else can you see so much action? Murders are routinely committed and solved, dastardly plots are hatched and foiled, and characters learn some deep life-truth – all within a single hour.

This makes the runaway success of ESPN's "World Series of Poker" and the Travel Channel's "World Poker Tour" all the more mysterious. What is it about a group of middle-aged men, 11.5-gram clay chips, and 52 brightly colored cards gathered about a large felt-covered table that's so appealing to so many?

When I was in a bookstore recently looking for books on poker, the sales clerk looked at me knowingly as I made my purchase. "We've been selling a lot of those," she told me. "Do you watch the games on TV?" I admitted I had.

While many people have played "Wednesday night poker" with friends for years, I've been a latecomer to it. Feeling slightly guilty for the years of poker I had missed out on, I decided to plunge into it. The rules are simple enough; surely a bit of poker knowledge would lead me to success.

I was fortunate enough to have two students who are excellent poker players and who recently attended classes I held in Las Vegas. During the day, I taught them about successful programming; during the night, they taught me about successful card playing.

One of the lessons I found hardest to



By Hal Helms

learn was that successful poker players don't passively accept ("call") other people's bets; they take the aggressive act of initiating and raising bets. "If a [card] hand is good enough to call," one player told me, "it's good enough to raise."

"But," I protested, "it just doesn't feel natural."

My student-cum-teacher looked at me curiously. "Of course it's not natural. What's natural is losing."

This truth must have been slowly seeping in, for I found myself thinking about programming in light of this.



Albert Einstein once remarked that what passed for "common sense" is really the collection of prejudices acquired by the age of 18. According to a 2002 Standish Group report, corporate software projects fail at an astonishing 70% rate. I began to ask myself what prejudices are common to us programmers, and what *unnatural* acts are called for to avoid the common experience of failure.

Here's a list of seven common practices that lead to failure – along with their remedies.

1. Undervaluing encapsulation: This practice leads to large files with a great many lines of code. A single file then has many responsibilities. I once worked with a programmer who showed me (proudly) an application that consisted of a single file with 12,000 lines of code. When we practice encapsulation, we break the application into many small, discrete pieces, each of which is responsible for doing only a small bit of work. A well-defined interface specifies what the code requires and what it will provide. The greatest single reason for object-oriented programming's success is the powerful abilities it provides programmers to practice encapsulation. ColdFusion components (CFCs) allow us to build highly encapsulated components whose interface we can examine with the excellent component browser provided with ColdFusion MX. A well-encapsulated application then resembles nothing so much as a conversation between components.

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



ADVERTISE

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

SUBSCRIBE

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



MX
developer's journal



2. Testing later: Here, I'm being charitable. Often, of course, no testing is done; we rely on our users to do the testing for us. The reason for this practice is, I think, that while we know we should test our software, we may not know exactly how to go about this. Testing is not something that should be done later. We should treat writing code and writing tests for that code as two parts of a whole: one isn't done without the other.

Kent Beck, of Extreme Programming fame, has written an excellent book entitled *Test-Driven Development*, which provides clear guidance on how to approach software by writing the tests *before* the code is written.

3. Cowboy coding: In this practice, the myth of "rugged individualism" is applied to the practice of writing code. Each programmer heads out into the sunset on his or her own horse, determining an individual course of action. Pair programming? Code reviews? Not out here in the Wild West where such sissified acts are objects of scorn to real men (and, sadly, women). Put several cowboys together and you don't have a team – you have the potential for a shoot-out. It's ironic that the same personality that often leads to excellent acts of individual programming spells disaster when a project is too large or its lifespan is too long for a single individual. The remedy for this malady is stiff medicine: the programmer must actively work to create an environment in which others are involved and consulted. Talk about an unnatural act!

4. Omitting prototypes: In his terrific book, *Code Complete*, Steve McConnell provides clear statistics that show that the greatest single action we can take to guarantee a successful software project is prototyping. Prototyping goes far beyond simple mock-ups; a complete prototype should make the user think that he or she is seeing the actual software. If we create the "front end" of our software first, we have the greatest chance of success with actual users – the only valid judges of our

success or failure. Why is this so important? Simply because to the users, the front end is the application; they take it for granted that "all that back-end stuff" will work. Seem odd? Well, as Alan Cooper points out, when was the last time you bought a phone and demanded to hear the dial tone before purchasing it? Didn't you take it for granted that "all that phone stuff" would work correctly?


5. Not defining acceptance tests: By what metric do we judge success? If we have no predefined acceptance tests, how can we know that we've succeeded, or that we're even done? For too many, an "acceptance test" consists of asking the client, "Do you like it?" That question – do you like it? – is enormously important. It's so important that it needs to be placed in the context of the prototype where clients can go through the iterative process of finding out exactly what they want in a safe environment. Acceptance tests should be a formal, measurable way of simply ensuring that the approved prototype runs. Acceptance tests should be established prior to the start of coding and should define both the procedures and the hardware on which tests will be run.

6. Not using a framework: The history of programming can be seen as a series of greater and greater abstractions. ColdFusion's popularity is due in large part to its ability to abstract lower-level functionality into tags. Oddly, though, some programmers stop here: each application is built as if it were the first and only application. As is the next application and the next. Frameworks provide a foundation on which you can build your own applications. One of the great benefits of both abstraction and frameworks is that they require less low-level work, giving you the time and freedom to work on making your app truly great. The downside of frameworks is that to become proficient, you must learn the framework. Luckily, popular frameworks such as Fusebox 4 and Mach-II have active communities that are quick to offer

help, and books and training exist to advance your expertise.

7. Shallow learning: Rene Magritte, the Belgian surrealist painter, once painted a large pipe on a canvas, with words underneath proclaiming, "This is not a pipe." Magritte's point was that an image is not the thing itself. I sometimes think that all buzzwords should be accompanied by a sign proclaiming, "This is not an idea." The culture of programming is "dumbed down" when we rely on TLAs (three-letter acronyms!) and buzzwords to take the place of a deep understanding. There is indeed an overwhelming amount to learn in our business, and each day our present knowledge becomes worth less as it edges toward obsolescence. I wish I had a great remedy for this, but I know of none.

Instead, we must have the courage to admit that continuous learning is an absolute requisite for us and do all in our power to deepen our understanding by reading, experimenting, and attending training classes and conferences. The problem with buzzwords is that they short-circuit this admittedly difficult process, tricking users into thinking that they have already attained knowledge. This illusion of knowledge is intellectual quicksand that will undermine our efforts far more profoundly than simple ignorance.

Some unknown wit once remarked that insanity is defined as "doing the same thing over and over – while expecting different results." We all want to excel at what we do, but unless we've arrived at the pinnacle of our craft, we're going to have to step out of our comfort zone and perform a few unnatural acts to get there. 

About the Author

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

hal.helms@teamallaire.com

A LIMITED TIME SAVINGS OFFER FROM SYS-CON MEDIA

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

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

RECEIVE
YOUR DIGITAL
EDITION
ACCESS CODE
INSTANTLY
WITH YOUR PAID
SUBSCRIPTIONS

3-Pack

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

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

6-Pack

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

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

9-Pack

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

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



CALL TODAY! 888-303-5282

Pick a 3-Pack, a 6-Pack or a 9-Pack

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

**TO
ORDER**

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

☐ MX Developer's Journal

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

☐ Linux World Magazine

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

☐ Java Developer's Journal

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

☐ Web Services Journal

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

☐ .NET Developer's Journal

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

☐ XML-Journal

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

☐ WebLogic Developer's Journal

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

☐ ColdFusion Developer's Journal

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

☐ Wireless Business & Technology

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

☐ WebSphere Developer's Journal

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

☐ PowerBuilder Developer's Journal

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

*WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

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

**SYS-CON
MEDIA**

Persistence: Creating State

The differences between using persistent and nonpersistent cookies

What is state? You may have heard the popular phrase, "The Web is a stateless environment." Simply put, data cannot persist across multiple page requests to the server. By nature, each request is independent of the others.

When building Web applications, we often have a need for persistent data, data that is available across multiple page requests while being unique to each user. There are many uses for persistent data, from checking to see if a user's session has timed out, to storing information about a user for use later in the application.

How Is State Created?

For data to persist, state must be created between the client and the application. In ColdFusion, state is created with two identifiers/variables that are unique to each user, CFID and CFTOKEN. For a state to persist, these two variables must be stored in one of two places: either in a cookie on the client's browser, or passed as variables in the URL with each page request. ColdFusion then uses the CFID and CFTOKEN to associate persistent data with each user by associating the values of these variables with CFID and CFTOKEN variables that are set in the client and session scopes. In ColdFusion, creating state is a fairly simple process.

Although persistent data can be stored in multiple places (in cookie, session, and client variables), that is not the focus of this article. Persistent variables will be covered in Part 2. This article will address the differences between using persistent and non-persistent (session) cookies, and how to maintain state when cookies aren't allowed or turned off.

Using Persistent Cookies for State

What is a "persistent cookie?" A persistent cookie is written to the client's hard drive as opposed to being stored in the client's memory. The cookie is so named because it will remain on the user's hard drive (persist) either until its specified expiration date or until it is removed by the user.

The easiest and simplest way to create state using persistent cookies is to use the SETCLIENTCOOKIES attribute of the tag <CFAPPLICATION> (see Listing 1). SETCLIENTCOOKIES is an optional attribute that defaults to "Yes" if not specified. Upon execution, two variables, CFID and CFTOKEN, are stored in a cookie on the client's browser if the client has enabled cookies.

Two other attributes are also included in <CFAPPLICATION>: CLIENTMANAGEMENT and SESSIONMANAGEMENT. These two attributes are optional for <CFAPPLICATION>, but at least one needs to be set to "Yes" in order for ColdFusion to write the cookie storing CFID and CFTOKEN to the client's browser. Although you may never need to reference these two variables, you can output their values with #cookie.cfid# and #cookie.cftoken# respectively. As stated above, ColdFusion uses CFID and CFTOKEN to associate persistent data with each unique user.

What is the advantage of using persistent cookies if session variables time out? The use of persistent cookies is advantageous when using client variables. Session variables typically time out after a period of inactivity. However, client variables can persist across multiple days, weeks, and even months (this setting is controlled in the ColdFusion Administrator). By using persistent cookies, client variables that were set weeks ago can still be accessed by a previous user. This could be a great way to store a user's preferences.

Seems too easy, what's the catch? Although creating state with persistent cookies is very easy, it does have some shortcomings. I'll give a scenario with two users, Joe and Adam. Joe logs on to an application at a public workstation. Some information about Joe is pulled from a database and stored in session variables (i.e., credit card, SSN, phone number, etc.). Joe does some work and closes all browser windows when he finishes.

Adam comes along a few minutes later and types in the URL for the same application, but the auto fill from the browser's history offers suggestions for other pages in the same application. Adam then selects one of the suggested URLs. Since it was only a few minutes between Joe and Adam, Joe's session hadn't expired and all of Joe's information is now available to Adam. Adam has the ability to navigate the application as Joe. Whether intentional or not, it could cause some serious damage.

Using persistent cookies in Web applications may also be frowned upon by some government agencies since persistent cookies are written to the client's hard drive. There may be requirements to use nonpersistent cookies.

By Derrick Rapley

Using Nonpersistent (Session) Cookies

As opposed to their counterpart, nonpersistent cookies are not written to the client's hard drive, but to the client's memory. Once all instances/windows of a browser are closed, then the nonpersistent cookie is erased from the client's memory.

Nonpersistent cookies can be set using the CFCOOKIE tag (see Listing 2). The key to creating a session cookie is to set the EXPIRES attribute in <CFCOOKIE> to "Now". EXPIRES is an optional attribute and defaults to "Now" if it is not specified. By specifying EXPIRES="Now", a cookie is written to the client's memory. If the cookie you are setting is already a persistent cookie on the client, setting EXPIRES="Now" removes the cookie from the cookie.txt file, leaving the cookie set in the client's memory. Setting session cookies can be created regardless of the value of SETCLIENTCOOKIES (Yes or No) in <CFAPPLICATION>.

In the example in Listing 2, I used the client-scope counterparts to set the cookie variables. You may use either the client- or session-scope counterparts of CFID and CFTOKEN to set the cookies, depending on if you have the CLIENTMANAGEMENT and SESSIONMANAGEMENT attributes enabled in <CFAPPLICATION>.


If Cookies Are Disabled

It is not uncommon to find users who may have cookies disabled for one reason or another. New to ColdFusion MX is a function, URLSessionFormat(), that appends the client information to the URL if the client has cookies disabled. If cookies are enabled, then no information is appended to the URL (see Listing 3).

Unfortunately, for developers using previous versions of ColdFusion, URLSessionFormat() is not an available function. If CLIENTMANAGEMENT is enabled in <CFAPPLICATION> the user may append the variable CLIENT.URLTOKEN to a link. Likewise, if SESSIONMANAGEMENT is enabled in <CFAPPLICATION>, the user may append SESSION.URLTOKEN to a link (see Listing 4). URLTOKEN is a concatenation of the CFID and CFTOKEN.

Using either method to append the CFID and CFTOKEN to a URL can become cumbersome since it has to be performed with every single link and form posted within the application. The one exception is that neither method is necessary when using <CFLOCATION> when setting ADDTOKEN="Yes". By setting the attribute ADDTOKEN="Yes", <CFLOCATION> will automatically append the URLTOKEN to the URL.

Conclusion

There is a myriad of possibilities to consider when creating state for an application. Although there is no wrong way to configure state management, the best solution is the one that fits your organization's or client's needs. 

About the Author

Derrick Rapley is a Web applications developer with QSS Group, Inc., and also maintains the ColdFusion community portal www.cfbookmark.com.

arapley@qssgroupinc.com

Listing 1

```
<!-- Include this snippet in Application.cfm -->
<cfapplication name="myApplication"
  setclientcookies="Yes"
  clientManagement="Yes"
  sessionManagement="Yes">
```

Listing 2

```
<!-- Include this snippet in Application.cfm -->
<cfapplication name="myApplication"
  setclientcookies="yes"
  clientmanagement="yes"
  sessionmanagement="yes">

<cfcookie name="CFID"
  value="#client.CFID#" expires="now">
<cfcookie name="CFTOKEN"
  value="#client.CFTOKEN#" expires="now">
```

Listing 3

```
<!-- Use URLSessionFormat for each link within your application -->
<cfoutput>
  <a href="#URLSessionFormat("nextPage.cfm")#">Next Page</a>
</cfoutput>
```

Listing 4

```
<!-- If client management is enabled -->
<cfoutput>
  <a href="nextPage.cfm?#client.urltoken#">Next Page</a>
</cfoutput>

<!-- If session management is enabled -->
<cfoutput>
  <a href="nextPage.cfm?#session.urltoken#">Next Page</a>
</cfoutput>
```



ATTN: Developers

STEP UP
to the mike
and be... **HEARD!**

Go to
<http://developer.sys-con.com>

Calling Sleek Geeks Everywhere!

Make sure you have your finger on the pulse of i-Technology...bookmark <http://developer.sys-con.com> today.

i-Technology
News
i-Technology
Views
i-Technology
Comment
i-Technology
Debate

© COPYRIGHT 2003, SYS-CON MEDIA
WWW.SYS-CON.COM

SYS-CON MEDIA

CFDEBUGGER

Tracing code execution in BlueDragon

Have you ever wanted to see what lines of code are being executed in your template? If you looked for CFTRACE to do that, you were likely disappointed. The CFDEBUGGER tag in BlueDragon, however, can be a great tool for debugging.

When CFML developers need to figure out what their code is doing, in terms of what lines of code their template is executing, they have surprisingly few options. There are other debugging tools, but they address different needs.

There's the server debugging info, but that's all after the fact, simply showing the state of variables at the end of the program (and/or in some cases their values when passed into it). And many rely on a trusty combination of CFABORT and CFDUMP to help identify control flow issues and other debugging challenges.

There's also the new CFTRACE tag introduced in CFMX and the older CFLOG tag from CF 4; each offers interesting variants on enabling you to track the value of some variables or log some string at a point in time. But neither of these can be used to simply show what lines of code have been executed in a run of a template. And the Java-based stack trace doesn't solve this problem either.

The closest we had was the interactive debugging tool of CF Studio that worked with CF 5, but is no longer supported in CFMX. Even then, many found the interactive debugger to be lacking and often not usable.

If you simply wanted to get some listing of all the lines of code that were executed in a template (what some really would call a "trace"), there's currently no solution in CFML unless you use BlueDragon. In this month's **Blueprints** column I'd like to introduce this feature



By Charlie Arehart

that could be useful to all CFML developers.

CFDEBUGGER: It's There If You Need It

<CFDEBUGGER> is a tag that works in BlueDragon only, and it simply writes an indication of each CFML line of code that's been executed to a log file. That's something many

have long wished for. There are times when this is just the ticket to solve a challenging debugging problem. Here's a simple example of its use. Let's say you have a template that just does this:

```
<CFDEBUGGER LOGFILE="trace.log">
<cfset name="bob">
```

Note that no closing CFDEBUGGER tag is needed (as will be explained later). This will create an entry in a file named trace.log (as indicated in the LOGFILE attribute) with the following info:

```
#0: CFDEBUGGER trace started @ 19/Aug/2003
15:03.19
#1: active.file=C:/inetpub/wwwroot/regression/
cfdebugger.cfm
#2: tag.end=CFDEBUGGER; L/C=(1,1); File=C:/
inetpub/wwwroot/regression/cfdebugger.cfm
#3: tag.start=CFSET; L/C=(2,1); File=C:/inetpub/
wwwroot/regression/cfdebugger.cfm
#4: tag.end=CFSET; L/C=(2,1); File=C:/inetpub/
wwwroot/regression/cfdebugger.cfm
#5: file.end=C:/inetpub/wwwroot/regression/cfde-
bugger.cfm
#6: Session Ended
```

Note that it indicates the time the template was run and the template's name, which is useful because the log could hold lots of such "trace sessions," for reasons I'll explain in a moment. More important, for each CFML tag it encounters, the trace shows its start and end lines in the given template. Unlike looking in an editor, this log shows only lines with CFML tags that were executed, not all lines in the entire file.

And while the CFML error message (in both BlueDragon and CF) reports the line of code on which an error occurs, what if the problem is a logic error that's not generating any CFML errors?

This capability could be useful for debugging a knotty problem where you can't tell which lines of code are being executed. If this was a several-hundred-line program doing loops and ifs, it could help you quickly narrow down just where the flow of control was going if you were having trouble figuring it out. Sometimes even well-placed CFWRITEs, CFDUMPs, and CFABORTs just don't do the trick.

I'll grant that it could produce a lot of data to analyze (and sadly it's not in a format that's easily conducive to performing automated analysis), but there will be times when it's better than nothing.

And it's something that CF has never had. Did you perhaps think that this was what CFTRACE would or should have done? It doesn't. CFTRACE (introduced in CFMX) just writes the value of a given variable or string to either a log file, the debugging output, or the screen. It's just a little more useful than CFLOG or CFDUMP. (BlueDragon currently supports CFLOG and will eventually support CFTRACE.)

So CFDEBUGGER could be very handy when you need it. Give it a try. Just be careful not to leave it on lest it create humongous log files.

Here are some additional notes about the use of CFDEBUGGER.

Where Is the Trace File Stored?

In the example I showed, I didn't name the path for the location to the log file. Where does it go then?

In the Server editions of BlueDragon, it's stored in the BlueDragon install directory. For instance, if you're using BlueDragon Server JX and it's installed in C:\Program Files\New Atlanta\BlueDragon_Server_JX_3.1, that's where you'll find the log file.

In the J2EE edition, the actual location will depend on the J2EE app server you're using. I've not found any logic to explain where the server decides to put it. It's neither the application's context root nor the install directory for the app server. For instance, in my WebLogic 8.1 test, it was found in C:\bea\user_projects\domains\mydomain. In JBoss, I found it in C:\jboss-3.2.1_tomcat-4.1.24\bin. In JRun, it was in my C:\JRun4\bin. If you have another server, just do a search within that server's directory for whatever you named the file (you're free to call it what you want), and make note of that location for future reference.

Can I Provide a Full Path for the Log File?

Rather than wondering where it may be stored, you can also just name the full path to the intended directory for placing the file. For example, using LOGFILE="c:\trace.log" works as expected, and in both the Server and J2EE edition.

What Happens If You Do a CFINCLUDE or Custom Tag Call While Tracing?

No problem. The trace log will continue to log all the activity (hopefully that's what you wanted). The log file does indicate when you switch to running a new template. At the point of the cfinclude, you'd see an entry like this:

```
#6: file.start=C:/Inetpub/wwwroot/includedfile.cfm
```

where in this example my calling template included a file called includedfile.cfm. That's useful. The same is true when calling a custom tag.

When Does Tracing Begin?

The tracing begins at the point that the CFDEBUGGER tag appears. Therefore, it will *not* trace activity in the application.cfm that occurs before you start tracing. Naturally, if you place the CFDEBUGGER tag in the application.cfm, then indeed it will do the tracing from that point forward (as I just said above, the tracing does remain in effect across includes and custom tags, and application.cfm is executed like an implicit cfinclude before any template runs, so the same logic applies).

When Does the Tracing Stop?

Tracing ends at request termination (of course, if you have an onrequestend.cfm, it will trace that as well). As will be discussed, while tracing stops at the end of request execution, the log file is not emptied upon each new request.

Is There a Way to Turn Off Tracing?

In other words, if a calling/including template or the application.cfm starts the tracing, is there a way to stop it in the called/included/requested program? No, currently there is not. And again, there is currently no available closing CFDEBUGGER tag to surround code to be traced. If you don't want to do the tracing, you should just remove the CFDEBUGGER tag.

THE INSIDER INTELLIGENCE YOU NEED...

TO KEEP AHEAD OF THE CURVE

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

- JAVA > Newsletter
- WebServices > Newsletter
- JOURNAL > Newsletter
- wireless > Newsletter
- WebLogic > Newsletter
- WebSphere > Newsletter
- ColdFusion > Newsletter
- .NETjournal > Newsletter

FREE

**E-Newsletters
SIGN UP TODAY!**

Go to www.SYS-CON.com

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

Exclusively from the World's Leading
i-Technology Publisher

**SYS-CON
MEDIA**

**Don't Delay!
Subscribe
for FREE!
at www.sys-con.com**

If a template is running under the influence of another that turns tracing on within the current request, you'll just need to remove the CFDEBUGGER from that controlling template if you don't want it to trace any that it includes/calls/requests.

You might wonder if you could solve the problem another way – perhaps by leaving the LOGFILE attribute value empty or leaving it off entirely, hoping it will turn the logging off. That doesn't work and will instead get an error. I suppose in Linux boxes you could redirect the tracing to the null device, though.

Again there's no way to stop tracing within a request once it's started for a given request, short of removing the CFDEBUGGER tag.

Is a New Trace File Created on Each Request?

No, and this is very important. The trace files are cumulative. They are appended to by each request that writes to them. Beware, therefore, as they can grow quite large with successive executions. Don't forget to remove the CFDEBUGGER when no longer doing tracing. There is currently no attribute to cause the tag to erase existing log entries or otherwise manage the log file size. (It does create one if it doesn't exist the first time you call it.)

What Happens When a New CFDEBUGGER Is Encountered Naming a New Log File?

In other words, what happens if one template (such as application.cfm or a calling/including template) has turned on the tracing to one file, and another template (or even later code in the same template) offers a new CFDEBUGGER naming a new log file? Tracing simply stops on the first log file and continues with the newly named one. That could be useful (or it could be a hassle in some instances, but that's the way it currently works).

Does CFDEBUGGER Trace Functions As Well? Or Just Tags?

It currently traces just tags, not references to expressions within tags. For instance, consider the following code:

```
<CFDEBUGGER LOGFILE="trace.log">
<cfoutput
this is a test
#cgi.request_method#
another line
#now()#
</cfoutput>
```

It might be nice if it reported encountering the variable and function references on the 4th and 6th lines. It does not, reporting only on the opening and closing CFWRITE tag. The same would apply to expression references within CFMAIL, etc., as well as statements within CFSWIFT. These are not individually traced.


(There's a TagsOnly attribute available for the tag, designed to take a yes/no value, which was perhaps intended to solve this problem, but it's not currently implemented.) It's not really a big deal if you think about it, at least with respect to CFWRITE and CFMAIL. The value of detecting the flow of control within these tags would seem somewhat diminished.

One last question that some may wonder about: Is it risky for BlueDragon to add a tag like CFDEBUGGER to CFML? Does it make your code incompatible? Look at it this way: you wouldn't leave the CFDEBUGGER tag in a production CFML application. You'll only use it for debugging purposes, so there's really no risk to your using it and therefore our adding it.

Some Possible Improvements

I have alluded to some challenges of using the CFDEBUGGER tag. It's not perfect. While it's useful now, perhaps as more people take advantage of this nifty tag, New Atlanta can consider improving it. Here are some possible improvements I'd look forward to:

- Change the log output format into a CSV file or the like (at least, give us the option) so that New Atlanta could then perform analysis on the output. For instance, it would only be a short step to then be able to produce a report indicating which lines of code were executed the most.
- Write out the time taken to execute each tag. Again, this could then feed into an analysis reporting not by frequency of execution but by total time per line of code. That could be really cool.
- Offer a means to turn off tracing, in case it was enabled in a calling/including template or application.cfm and you don't want it continuing in an included/called/requested template.
- Cause it to create a new log file on each new request (rather than growing unchecked), or a way to indicate a maximum number of requests to keep or a maximum size the log file should grow, or perhaps an attribute to indicate that the log should be wiped clean before being written to for the current request.
- Have the tag only work as it would seem it should, letting it trace the lines containing variable and function references.
- Add an attribute that indicates if it should only create its tracing output if the debugging option is turned on in the Admin. That way it could be left in place in production and not have any effect unless debugging is on, or you run it while debugging is enabled for your IP. CFTRACE works this way in CFMX, and our new CFASSERT tag that we've added to the latest release of BlueDragon works similarly (more on that in a later article). Supporting debugging tags to run only when debugging/testing is on is a good model. Then again, some may especially want it tracing while folks are running the template in production. That's why I think it should be controllable as an attribute.

But, for now, enjoy what it does offer, which is a unique way to trace the lines of code being executed in a CFML template. 

About the Author

Charlie Arehart is a Macromedia Certified Advanced ColdFusion developer and trainer. He has recently become CTO of New Atlanta Communications, makers of BlueDragon. In his new role, he will continue to support the CFML community, contributing to several CF resources, and speaking frequently at user groups throughout the country.

charlie@newatlanta.com

Tales from the List —continued from page 7

products and the new product features that together represent Macromedia's vision of how Web developers will work to "get things done," has resulted in an increase of threads on *CFDJ* List and other list servers, centered around use of non-CF products. No surprise, these threads usually relate to the new features in Dreamweaver MX 2004.

ColdFusion MX 6.1 has also made a huge impact on the discussion topics across the board. This release, which was not officially part of the MX 2004 Studio release but is included in Studio MX 2004, addressed many of the concerns and shortcomings of the original MX release. Among many other enhancements, the installation process and server performance is now superb, and many of the ColdFusion Component Architecture shortcomings have been addressed.

This is not recent news, but a noticeable trend is that developers are beginning to take advantage of the underlying J2EE architecture. Never before have there been so many posts asking questions about Java development topics. There has also been a large number of threads about the ability to run ColdFusion as a J2EE application and about how to configure multiple J2EE server instances.

In my article last month, I explained that ColdFusion should be seen more as an ideal way to develop Java applications, and the trend of J2EE topics on *CFDJ* List indicates that this is indeed the attitude developers are beginning to adopt.

MAX, Macromedia's conference in Salt Lake City, Utah (November 18–21) – where you may be reading this – replaces the conferences formerly known as UCON and DevCon, and is creating a lot of buzz. The session topics scheduled for MAX also reflect both the thinking of ColdFusion as a J2EE development environment and, very much, the integration of Studio MX 2004 products. Macromedia has categorized the sessions by experience level (www.macromedia.com/macromedia/conference/schedule/by_level/) and by track (www.macromedia.com/macromedia/conference/schedule/by_track/).

Taking a look at the tracks and the sessions that fall within, there's "Client Side Development," which focuses mostly on Flash but also on Central and Dreamweaver development topics; "Design," which focuses primarily on Dreamweaver but also includes Central, FreeHand, and Contribute topics; "Experience Matters," which focuses on RIA development; and "Server Side Development," which is overwhelmingly ColdFusion topics but also includes PHP and .NET topics.

You will also find several topics that cover OOP, architecture, Java/CF hybrid applications, and other "software architecture" topics. All of the tracks include sessions that focus on the integration of two or more Macromedia technologies. You'll find that, in particular, the "design" track focuses often on product integration, and that (obviously) the "Experience Matters" sessions pretty much all go into discussions about multiple products (building rich front ends to robust server-side logic is a fundamental part of RIA, after all).

More and more, I believe we will be seeing cross-product discussion topics on the List and at our conferences. Whether you are a designer or a hardcore server-side developer, these discussions are worth paying attention to. As Macromedia makes Dreamweaver more and more the ideal development environment for CFML, and as rich Internet applications become a more common occurrence on the Web, you'll find that keeping up with the trends makes you that much more marketable and productive a developer.

I strongly encourage those of you at MAX to attend some of the sessions focused on cross-product use, as well as some of the more conceptual CF development sessions. I also encourage you to stop by the SYS-CON Media booth to say "hi" to our *CFDJ* editor-in-chief, Robert Diamond. Let him know what you think about the magazine, and also, pick up your free copy of the premier issue of *MX Developer's Journal*!

On a final note – I sincerely hope that you will also track me down before or after one of my sessions, in the vendors' arena, or at one of the social events. These conferences are ultimately all about the community. Every year I look forward to getting to meet *CFDJ* List members and *CFDJ* readers, to hear your opinions about the magazine and about this column, as well as for the fun of it!



CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
ACTIVEPDF	WWW.ACTIVEPDF.COM	866.GO.TOPDF	4
CFDYNAMICS	WWW.CFDYNAMICS.COM/MAXCHALLENGE	866.233.9626	25
EDGE EAST 2004	WWW.SYS-CON.COM	201-802-3069	35
EDGE WEB HOSTING	WWW.EDGEWEBHOSTING.NET	1.866.EDGEWEB	Cover II
EKTRON	WWW.EKTRON.COM/CFDJ		6
FUSETALK	WWW.FUSETALK.COM	866.477.7542	31
HAL HELMS, INC	WWW.HALHELMS.COM		33
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM/CFDJ	877.248.HOST	29
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800.379.7729	Cover IV
MACROMEDIA	WWW.MX2004.COM		13-15
MACROMEDIA	WWW.MACROMEDIA.COM/GO/CFMXAD		Cover III
MX DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM	888-303-5282	39
NETQUEST	WWW.NQCONTENT.COM		3
NEW ATLANTA COMMUNICATIONS	WWW.NEWATLANTA.COM/BLUEDRAGON		11
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	27
TERATECH	WWW.CFCONF.ORG/CFUN-04/	301.424.3903	23
WEBCORE TECHNOLOGIES	WWW.WEBCORETECH.COM	877.WCT.HOST	19

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

ColdFusion User Groups

For more information go to...

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

New England

New Hampshire
Northern N.E. MMUG
www.mmug.info

Massachusetts
Boston, MA CFUG
www.cfugboston.org

Rhode Island
Providence, RI CFUG
www.ricfug.com

Vermont, Montpelier
Vermont CFUG
www.mtbytes.com/dfug/index.htm

Midatlantic

New Jersey, Raritan
Central New Jersey CFUG
www.freecfm.com/cjcfug/index.cfm

New York
Albany, NY CFUG
www.anycfug.org

New York
Long Island, NY CFUG
www.licfug.org

New York
New York, NY CFUG
www.nycfug.org

New York
Rochester, NY CFUG
www.roch-cfug.org

New York
Syracuse, NY CFUG
www.cfugcny.org

Pennsylvania, Harrisburg
Central Penn CFUG
www.centralpenncfug.org

Pennsylvania
Philadelphia, PA CFUG
www.pacfug.org

Pennsylvania
Pittsburgh, PA CFUG
www.orbwave.com/pgchcfug

Pennsylvania
State College, PA CFUG
www.cfug-sc.org

Southern

Alabama
Birmingham, AL CFUG
www.bcfug.org

Alabama
Huntsville, AL CFUG
www.nacfug.com

Delaware, Dover
Delaware CFUG
www.decfug.org

Delmarva, Dover
Delmarva CFUG
www.delmarva-cfug.org

Florida
Gainesville, FL CFUG
<http://plaza.ufl.edu/aktas>

Florida
Orlando, FL CFUG
www.cforlando.com

Florida
Tallahassee, FL CFUG
www.tcfug.com

Florida
Tampa, FL CFUG
www.tbcfug.org

South Florida
South Florida CFUG
www.cfug-sfl.org

Georgia, Atlanta
Atlanta, GA CFUG
www.acfug.org

Georgia, Atlanta
Georgia CFUG
www.cfugorama.com

Georgia
Columbus, GA CFUG
www.vcfug.org

Kentucky
Louisville, KY CFUG
www.loulexcfug.com

Louisiana
New Orleans, LA CFUG
www.nocfug.org

Maryland
Annapolis, MD CFUG
www.ancfug.com

Maryland
Baltimore, MD CFUG
www.cfugorama.com

Maryland
Broadneck H. S. CFUG
www.cfug.broadneck.org

Maryland, Bethesda
Maryland CFUG
www.cfug-md.org

Maryland
California, MD CFUG
<http://smdcfug.org>

North Carolina
Charlotte, NC CFUG
www.charlotte-cfug.org

North Carolina
Fayetteville, NC CFUG
www.schoolink.net/fcfug/

North Carolina
Raleigh, NC CFUG
www.ccfug.org

Oklahoma
Oklahoma City, OK CFUG
<http://idgweb4.ouhsc.edu/cfug>

Oklahoma
Tulsa, OK MMUG
www.tulsacfug.org

Tennessee
Memphis, TN CFUG
<http://cfug.dotlogix.com>

Tennessee
Nashville, TN CFUG
www.ncfug.org

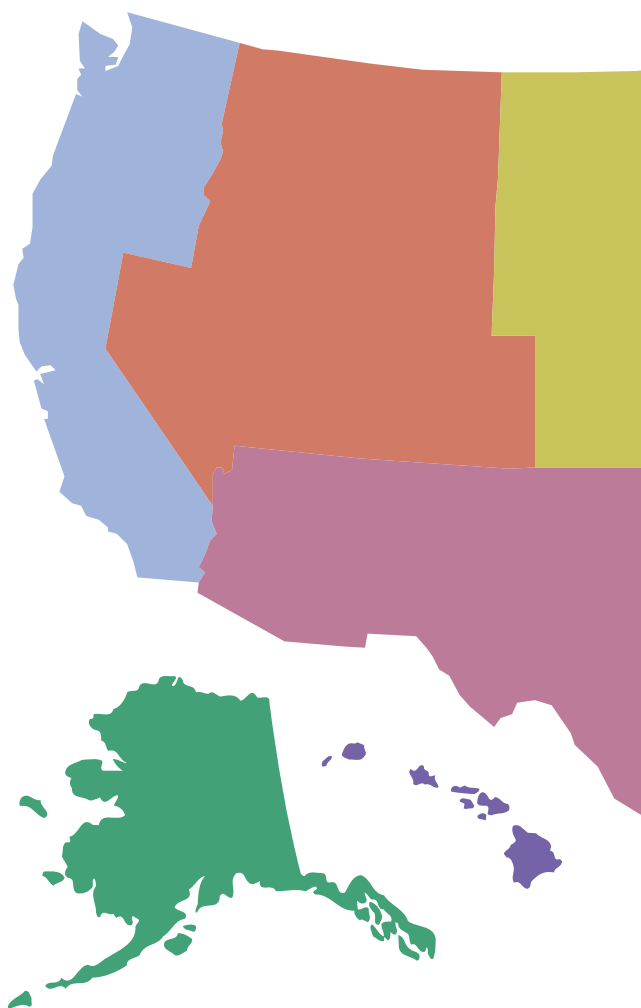
Virginia
Hampton Roads CFUG
www.hrcfug.org

Virginia
Northern Virginia CFUG
www.cfugorama.com

Virginia
Richmond, VA CFUG
<http://richmond-cfug.btgi.net>

Virginia, Roanoke
Blue Ridge MMUG
www.brmug.com

Washington D.C.
Washington, D.C. CFUG
www.cfugorama.com



Midwest

Northern Colorado
Northern Colorado CFUG
www.nccfug.com

Colorado Hamilton M.S. CFUG
<http://hamilton.dpsk12.org/teachers/team-c/intro.html>

Illinois, Champaign
East Central Illinois CFUG
www.ecicfug.org

Illinois, Chicago
Chicago, IL CFUG
www.cfugorama.com

Indiana
Indianapolis, IN CFUG
www.hoosierfusion.com

Indiana, South Bend
Northern Indiana CFUG
www.ninmug.org

Iowa
Des Moines, IA CFUG
www.hungrycow.com/cfug/

Michigan, Dearborn Mmaniacs
CFUG <http://ciwebstudio.com/mmania/mmania.htm>

Michigan, East-Lansing
Mid Michigan CFUG
www.coldfusion.org/pages/index.cfm

Minnesota, Minneapolis
Twin Cities CFUG
www.colderfusion.com

Rockies

Montana, Helena
Montana CFUG
<http://montanacug.site-o-matic.com>

Nevada
Las Vegas, NV CFUG
www.snfcug.com

Utah
Salt Lake City, UT CFUG
www.slcfug.org

Wyoming, Jackson
Wyoming MMUG
www.wyfcug.org

West Coast

California
Bay Area CFUG
www.bacug.net

California, Fresno
Central California CFUG
www.cccug.com

California, Inland Empire
Inland Empire CFUG
www.sccug.org

California
Los Angeles CFUG
www.sccug.org

California
Orange County CFUG
www.sccug.org

California
Sacramento, CA CFUG
www.saccug.org

California
San Diego, CA CFUG
www.sdcug.org

Southern California
Southern California CFUG
www.sccug.org

Oregon
Eugene, OR CFUG
www.EugeneCFUG.org

Oregon
Portland, OR CFUG
www.pdxcfug.org

Alaska

Alaska
Anchorage, AK CFUG
www.akcfug.org

Hawaii

Hawaii, Honolulu
Hawaii CFUG
<http://cfhawaii.org>

Southwest

Missouri
Kansas City, MO CFUG
www.kcfusion.org

Missouri
St. Louis, MO CFUG
www.psiwebstudio.com/cfug

Nebraska
Omaha, NE CFUG
www.necug.com

Ohio, Columbus
Ohio Area CFUG
www.oacug.org

Ohio
Mid Ohio Valley MMUG
www.movcfug.org

Wisconsin
Milwaukee, WI MMUG
www.metromilwaukee.com/usr/cfug

Arizona
Phoenix, AZ CFUG
www.azcfug.com

Arizona
Tucson, AZ CFUG
www.tucsoncfug.org

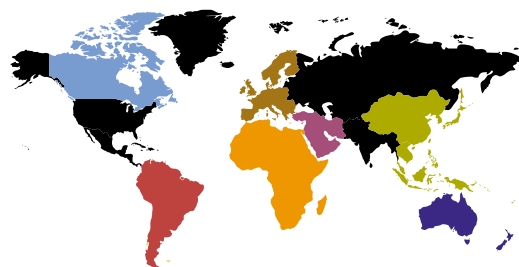
Texas
Austin, TX CFUG
<http://cftexas.net>

Texas
Dallas, TX CFUG
www.dfwcfug.org

Texas
San Antonio, TX CFUG
<http://samcfug.org>

About CFUGs

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



Africa

South Africa, Cape Town
Cape Town, South Africa CFUG
www.coldfusion.org.za

South Africa, Johannesburg
Johannesburg, South Africa CFUG
www.coldfusion.org.za

Central Europe, Munich
Central Europe CFUG
www.cfug.de

Finland, Helsinki
Finland CFUG
www.cfug-fi.org

France, Valbonne
France CFUG
www.cfug.fr.st

Germany, Frankfurt
Frankfurt CFUG
www.cfug-frankfurt.de

Ireland
Belfast, Ireland CFUG
www.cfug.ie

Ireland
Cork, Ireland CFUG
<http://viewmylist.com/cork>

Italy, Bologna
Italy CFUG
www.ingenium-mmug.org

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

Switzerland
Martin Bülmann, Switzerland CFUG
www.cfug.ch

United Kingdom, London
UK CFUG
www.ukcfug.org

United Kingdom
Northern England CFUG
www.cfug.org.uk

Asia

Malaysia, Kuala Lumpur
Malaysia CFUG
www.coldfusioner.com

Thailand
Bangkok, Thailand CFUG
<http://thaicug.tei.or.th>

Japan, Urayasu-city Japan CFUG
<http://cfusion.itfrontier.co.jp/jcfug/jcfug.cfm>

Australia

Australia-Melbourne
Australian CFUGs
www.cfug.org.au

Canada

Canada
Edmonton, AB CFUG
<http://edmonton.cfug.ca>

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Montreal, QC CFUG
www.kopanas.com/cfugmontreal

Canada
Ottawa, ON CFUG
www.cfugottawa.com

Canada, Ottawa (HS group)
Ecole Secondary CFUG
www.escgarneau.com/gug

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Canada
Vancouver, BC CFUG
www.cfug-vancouver.com

Canada
Vancouver Island CFUG
www.cfug-vancouverisland.com

Middle East

Pakistan Edu, Lahore Cantt
Pakistan Educational CFUG
www.cfeugpakistan.org

Pakistan, Lahore
Pakistan CFUG
www.cfugpakistan.org

Saudi Arabia, Riyadh
CFUG Saudia
www.cfugsaudia.org

Turkey, Izmer
Turkey CFUG
www.cftr.net

Europe

Belgium, Brussels
Belgium CFUG
www.cfug.be

South America

Brazil
Rio de Janeiro CFUG
www.cfugrio.com.br

SYS-CON Media Announces MX Developer's Journal

(Montvale, NJ) – MX Developer's Journal, SYS-CON Media's latest highly anticipated new title, launched on September 10, 2003. MXDJ will reach more than 2 million Macromedia MX developers using Flash, Dreamweaver, Fireworks, FreeHand, ColdFusion, and Director. Targeted at the professional developers and designers who use the award-winning Macromedia MX product family to build Web sites and applications, MXDJ is available now on newsstands and in specialty bookstores such as Barnes & Noble and Borders.



The publication debuted in conjunction with Macromedia's release of their MX 2004 product line, and the premier issue will be available at Macromedia MAX 2003, Macromedia's user conference.

Each issue of MXDJ will include:

- Hands-on tutorials covering every aspect of Flash MX, Dreamweaver MX, Fireworks MX, FreeHand MX, ColdFusion MX, and Director MX
- Full and constantly updated information on the overall Macromedia MX architecture
- Experience Reports on leading sites that already use rich Internet apps (RIA) to engage customers
- Insightful technical articles and feature stories on how best to harness the MX family of technologies
- Success stories – showcasing Fortune 500 companies already using Macromedia MX to deliver RIA
- Product reviews and sneak previews of upcoming releases within the Macromedia family
- How-to programming tips/code listings

A special, U.S. charter subscription rate is being offered to CFDJ readers, with substantial savings off newsstand prices. For more information, see www.sys-con.com/mx/charter.cfm, or call 1 (888) 303-5282.

Macromedia Breeze Live Now Available

(San Francisco) – Macromedia, Inc. has announced the availability of Macromedia Breeze Live, which reinvents the user experience of online meetings and takes the pain out of planning and running those meetings. Breeze Live is available as a standalone product or as an add-on module for the Macromedia Breeze platform.

"Macromedia Breeze Live breaks down the barriers of delivering effective online meetings by offering a simpler and better user experience before, during, and after the live session," said Keith Kitani, vice president of product management, Macromedia. "Online meetings have become a popular way for organizations to reduce the expense of meetings while retaining the interactivity and productivity. Breeze Live eliminates both the technical barriers and learning curve that have previously limited widespread adoption."

Breeze Live offers unique features not available in other online meeting solutions. Users can join a meeting by simply clicking a Web link, without any complicated downloads or setup. The solution builds on the power and broad reach of Macromedia Flash Player, which is already installed on more than 98% of online desktops, ensuring that most meeting participants won't have to download any software to participate. The application can also be customized to suit individual presentation styles, content, or the audience by having multiple customizable layouts within the meeting room.

For users hosting the meeting, Breeze Live enables content such as PowerPoint presentations to be set up in advance, preloading both content and the presentation layout. Meeting content, such as sharing spreadsheets or giving product demonstrations, can also be delivered live and changed in real-time during a meeting without having to interrupt the flow of the presentations. The ability to deliver recorded presentations and use persistent meeting rooms that retain the state in which they were

left make Breeze Live suitable for recurring meetings or presenting the same content to multiple groups.

"As Arizona is a large state and our hospital members are often hundreds of miles from our central office in Phoenix, we use Macromedia Breeze to enable member hospital executives to meet with staff and colleagues without spending hours traveling to and from our office," said Steve Nelson, director of Information Technology at the Arizona Hospital and Healthcare Association. "The Breeze interface makes it easy for executives to view presentations and discuss the material presented, saving us thousands of dollars on what we would have spent collectively on travel for an in-person meeting."

Macromedia Breeze enables companies to quickly and efficiently transfer knowledge, improve revenue generation, leverage their existing knowledge base, and reduce costs across the enterprise. The Macromedia Breeze platform enables businesses to quickly create and deliver on-demand learning and profes-



Integrated content authoring within PowerPoint

sional development courses, product introductions and presentations, sales force training, customer interactions, Web seminars, and online meetings.

Macromedia Breeze Live is available as either a hosted ASP or enterprise licensed solution. ASP annual packages start at \$83/month per user for a complete Web conferencing solution, including video and archiving. For more information, visit www.macromedia.com/go/breezelive/.

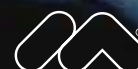


how many people does it take to change the web?

macromedia®
COLDFUSION®
MX

See for yourself. Upgrading to ColdFusion MX gives you amazing new features and a powerful Java™ architecture. Incorporate XML and web services with ease. Build flexible and maintainable applications with ColdFusion Components. Integrate with J2EE™ and .NET to deliver rich user interfaces with native connectivity to Macromedia Flash.™ Easily migrate existing applications and save time with the powerful new Macromedia MX development tools. Try ColdFusion MX today and see what you can do now.

Download the free 30-day trial at
www.macromedia.com/go/cfmxd



Copyright © 2002 Macromedia, Inc. All rights reserved. Macromedia, the Macromedia logo, ColdFusion, Flash, and Macromedia Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.

be the pilot!

Intermedia.NET is the #1
ColdFusion Hosting Provider*

FREE SETUP on Shared
Hosting Accounts With
ColdFusion MX Support

Use Promo Code CFDJ2003



INTERMEDIA.NET

WE DARE YOU TO TAKE A FREE TEST FLIGHT!

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

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

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

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

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

Managed Hosting • Shared Hosting • Microsoft Exchange Hosting

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



HostPilot™

*CFHub.com