

# ADVERTISERS INDEX

ADVERTISER	URL	PH	PG
ABLECOMMERCE	WWW.ABLECOMMERCE.COM	360.253.4142	2
ABLECOMMERCE	WWW.AUCTIONBUILDER.COM	360.253.4142	4
AFFINITY	WWW.AFFINITY.COM		49
ALLAIRE	WWW.VUE.COM/ALLAIRE	877.460.8679	39
ALLAIRE	WWW.ALLAIRE.COM	888.939.2545	61
ALLAIRE DEVELOPER CONFERENCE	WWW.ALLAIRE.COM/CONFERENCE		27
ANDREWS TECHNOLOGY	WWW.ANDREWSTECHNOLOGY.COM	888.393.0159	17
BIZNIZ WEB	WWW.BIZNIZWEB.COM	623.915.1661	64
CAREER OPPORTUNITIES		800.582.3089	57
CATOUZER	WWW.CATOUZER.COM		67
CFXHOSTING	WWW.CFXHOSTING.COM	800.939.8188	25, 31
CF DEVELOPER'S JOURNAL	WWW.COLDFUSIONJOURNAL.COM		31
CONCEPTWARE AG	WWW.CONCEPTWARE.COM		11
CORDA TECHNOLOGIES	WWW.POPCHART.COM	888.763.0517	3
CTIA	WWW.CTIA.COM		59
CYSCAPE	WWW.CYSCAPE.COM/BHFREE	800.932.6869	66
CYBERSMARTS	WWW.CYBERSMARTS.COM		66
DEVELOPERSNETWORK	WWW.DEVELOPERSNETWORK.COM	416.203.3690	23
DIGITALNATON	WWW.DEDICATEDSERVER.NET	877.624.7897	15
FALL INTERNET WORLD	WWW.PENTONEVENTS.COM		45
INFEA	WWW.INFEA.COM		63
INTELIANT	WWW.INTELIANT.COM	800.815.5541	43
INTELLIGENT ENVIRONMENTS	WWW.SCREENSURFER.COM		33
INTERLAND	WWW.INTERLAND.COM	800.419.1714	9
INTERMEDIA	WWW.INTERMEDIA.NET	800.379.7729	68
JAVA DEVELOPER'S JOURNAL	WWW.JAVADEVELOPERSJOURNAL.COM	201.802.3021	53
JDJ STORE	WWW.JDJSTORE.COM	888.303.JAVA	61
MISTRAL DESIGN GROUP	WWW.PURPLEHOSTING.COM		37
NETDIVE	WWW.NETDIVE.COM		41
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	21
SITEHOSTING.NET	WWW.SITEHOSTING.NET	877-NTHOSTING	57
SYS-CON MEDIA, INC.	WWW.SYS-CON.COM	800.513.7111	46
THE SHORTLIST.COM	WWW.SHORTLIST.COM		32, 47
VIRTUALSCAPE	WWW.VIRTUALSCAPE.COM	212.460.8406	19
WIRELESS DEVCON	WWW.WIRELESSDEVCON2000.COM		53
WOODBOURNE SOLUTIONS	WWW.WOODBOURNESOLUTIONS.COM	301.428.7620	63
XML DEVCON 2000-2001	WWW.XMLDEVCON2000.COM	800.513.7111	55
XML BOOTCAMP	WWW.SDEXPO.COM/BOOTCAMP	415.905.2702	13

Please visit  
the Web sites  
of our  
advertising  
partners  
who make it  
possible for us  
to bring you this  
Digital Edition  
(PDF) of *CFDJ*

# COLDFUSION Developer's Journal

ColdFusionJournal.com

October 2000 Volume: 2 Issue: 10



## Editorial

### **The Wonderful World of Wireless**

Robert Diamond page 5

## Foundations

### **Making Assertions**

Hal Helms page 14

## Product Reviews

### **AuctionBuilder Pro! 1.0 from AbleCommerce**

Carey Lilly page 22

### **CommonSpot from PaperThin**

Dave Horan page 56

## Guest Editorial

### **Great Philosophers of Software Development**

Steven D. Drucker page 26

## CF Conference

### **CFUN-2k = CF Party**

Charles Arehart page 36



page 42

## **CFDJ Feature: Recursive Custom Tags**

6

**A study of algorithms leads to superior application design**

Mark Cyzyk

## **<BF> on <CF>: 'The Ten Commandments' - Revisited**

18

**Even the holiest rules need an update**

Ben Forta

## **CFDJ Feature: A ColdFusion Based Oracle Database Monitor**

28

**Limitless possibilities with Web-enabled client/server applications**

Kailasnath Awati & Mario Techera

## **CF Tools: Use WDDX to Store Complex Variables for Clustered Web Servers**

36

**Powerful new tool helps CF developers meet Web needs**

Alan McCullough

## **CFDJ Feature: Toward Better Error Handling**

42

**Tips and Techniques**

Charles Arehart

## **Q&A: Ask the Training Staff**

48

**A new, interactive column for ColdFusion users**

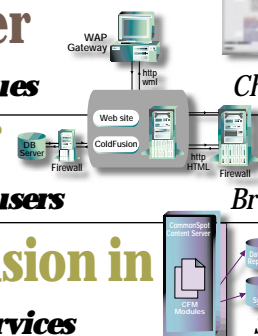
Bruce Van Horn

## **WAP & ColdFusion: ColdFusion in WAP Architecture**

52

**Create new services**

Sergey Ruseev



STEVEN D. DRUCKER, JIM ESTEN, BEN FORTA,  
STEVE NELSON, RICHARD SCHULZE, PAUL UNDERWOOD

**EDITOR-IN-CHIEF** ROBERT DIAMOND  
**ART DIRECTOR** JIM MORGAN  
**EXECUTIVE EDITOR** M'LOU PINKHAM  
**MANAGING EDITOR** CHERYL VAN SISE  
**EDITOR/COPY CHIEF** NANCY VALENTINE  
**ASSOCIATE EDITOR** JAMIE MATUSOW  
**PRODUCT REVIEW EDITOR** TOM TAULLI  
**TIPS & TECHNIQUES EDITOR** MATT NEWBERRY

### WRITERS IN THIS ISSUE

CHARLES AREHART, KAILASNATH AWATI, MARK CYZYK,  
ROBERT DIAMOND, STEVEN D. DRUCKER, BEN FORTA,  
HAL HELMS, DAVE HORAN, CAREY LILLY,  
ALAN MCCULLOUGH, SERGEY RUSEEV, BRUCE VAN HORN

### SUBSCRIPTIONS

SUBSCRIBE@SYS-CON.COM

FOR SUBSCRIPTIONS AND REQUESTS FOR BULK ORDERS,  
PLEASE SEND YOUR LETTERS TO  
SUBSCRIPTION DEPARTMENT.

SUBSCRIPTION HOTLINE 800 513-7111

COVER PRICE \$8.99/ISSUE

DOMESTIC \$79/YR. (12 ISSUES)

CANADA/MEXICO \$99/YR

OVERSEAS \$129/YR

BACK ISSUES \$12 EACH

**PRESIDENT AND CEO** FUAT A. KIRCAALI  
**VICE PRESIDENT, PRODUCTION** JIM MORGAN  
**VICE PRESIDENT, MARKETING** CARMEN GONZALEZ  
**GROUP PUBLISHER** LISE ST. AMANT  
**COMPTROLLER** BRUCE MILLER  
**ADVERTISING ACCOUNT MANAGER** ROBYN FORMA  
**ADVERTISING ACCOUNT MANAGER** MEGAN RING  
**ASSOCIATE SALES MANAGER** CARRIE GEBERT  
**ADVERTISING ASSISTANT** CHRISTINE RUSSELL  
**GRAPHIC DESIGNER** ALEX BOTERO  
**GRAPHIC DESIGNER** ABRAHAM ADDO  
**GRAPHIC DESIGNER** CATHRYN BURAK  
**GRAPHIC DESIGNER** LOUIS F. CUFFARI  
**GRAPHIC DESIGN INTERN** AARATHI VENKATARAMAN  
**WEB DESIGNER** STEPHEN KILMURRAY  
**WEB DESIGNER** GINA ALAYAN  
**SYS-CON EVENTS MANAGER** ANTHONY D. SPITZER  
**JDJ STORE.COM** AMANDA MOSKOWITZ

### EDITORIAL OFFICES

SYS-CON MEDIA, INC. 135 CHESTNUT RIDGE RD.,  
MONTVALE, NJ 07645  
TELEPHONE: 201 802-3000 FAX: 201 782-9600

COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)  
IS PUBLISHED MONTHLY (12 TIMES A YEAR)  
FOR \$79 BY SYS-CON PUBLICATIONS, INC.,  
135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

### POSTMASTER

SEND ADDRESS CHANGES TO:

COLDFUSION DEVELOPER'S JOURNAL

SYS-CON MEDIA, INC.

135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

### © COPYRIGHT

COPYRIGHT © 2000 BY SYS-CON MEDIA, 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.

FOR PROMOTIONAL REPRINTS, CONTACT REPRINT COORDINATOR.

SYS-CON PUBLICATIONS, INC., RESERVES THE RIGHT TO REVISE,  
REUBLISH AND AUTHORIZE ITS READERS TO USE  
THE ARTICLES SUBMITTED FOR PUBLICATION.

### WORLDWIDE DISTRIBUTION

BY CURTIS CIRCULATION COMPANY 730 RIVER ROAD,  
NEW MILFORD, NJ 07646-3048 PHONE: 201 634-7400

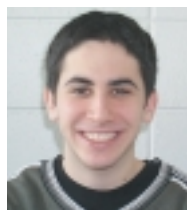
### DISTRIBUTED IN USA

BY INTERNATIONAL PERIODICAL DISTRIBUTORS

674 VIA DE LA VALLE, SUITE 204, SOLANA BEACH, CA 92075  
619 481-5928

ALL BRAND AND PRODUCT NAMES USED ON THESE PAGES  
ARE TRADE NAMES, SERVICE MARKS OR TRADEMARKS  
OF THEIR RESPECTIVE COMPANIES.

# The Wonderful World of Wireless

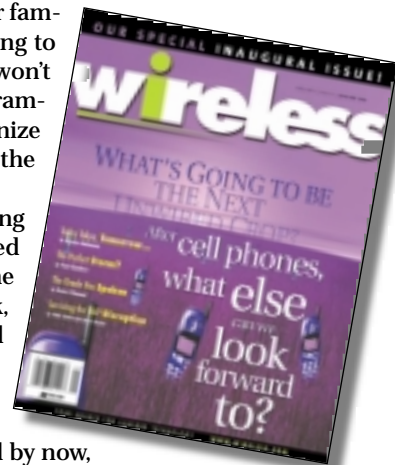


BY ROBERT DIAMOND

Wireless technologies are everywhere these days. I can't walk down the street for more than five minutes without seeing a cell phone, pager or Palm Pilot in use. And I'm seeing more of them every day. Wireless technologies are also being seen in the world of CF, which is being used more often each day to serve up data to WAP devices. Several articles about wireless technology have appeared in recent issues of **CFDJ**, and the feedback has been quite positive. Everyone who's written in has clamored for more coverage of this emerging market, and as we've been planning the content for the next several issues, we've taken that into consideration. I think you'll be pleased.

In fact, **SYS-CON Media** ([www.SYS-CON.com](http://www.SYS-CON.com)) - known to you as the publishers of **ColdFusion Developer's Journal** as well as **Java Developer's Journal**, **XML-Journal** and other such quality mags - has been receiving such a demand for wireless content that we've announced a whole new magazine... **Wireless** ([www.wireless.com](http://www.wireless.com)). Coming this fall, and available on newsstands everywhere, this will be the latest in our family of quality technical publications. **Wireless** is going to differ slightly from our other magazines in that we won't be covering just technical content, though the programming material will be in there. (In fact, you'll recognize a few of the regular **CFDJ** writers like Ben Forta in the premier issue.)

In addition to the developer content, we're putting together an amazing thing that can only be described as a "technical soup." And besides reporting on the technology that makes the unwired world work, we're going to focus on the people who make it all happen. In today's world the people driving the technology are just as important to follow as the technology itself - a vital way to stay on top of the fast-moving wireless market. If you haven't guessed by now, I'm quite involved with the magazine myself - as its founding editor and editor-in-chief. Don't worry (or get excited), I won't be leaving **CFDJ** anytime soon. I'll just be swapping hats often to cover both areas.



### On the Cold Front...

Back to ColdFusion, since there's been some interesting news lately. FAO Schwartz has implemented the Allaire Business Platform to relaunch its online toy store, and Pottery Barn, the popular home-furnishings store, is now the latest of several large and high-profile Web sites to be running on ColdFusion. Early results and feedback about the sites say that both the speed and the functionality are all quite impressive - demonstrating once again the power, robustness and, of course, scalability of CF. That and Allaire's latest financial announcement of total revenue increases of 155% for the three-month period ending June 30, 2000, in comparison to the three-month period ending June 30, 1999, put all of us in a very good position. It's a great time to be a CF developer!

ABOUT THE  
AUTHOR  
*Robert Diamond is  
editor-in-chief of  
ColdFusion Developer's  
Journal.*

*Robert Diamond*

ROBERT@SYS-CON.COM



# A study of algorithms leads to superior application design

## Recursive Custom Tags

BY MARK CYZYK

I have a confession to make: I wasn't a computer science major in college – I was a philosophy major. While the two disciplines have much in common (conceptual acrobatics, a high degree of abstraction, logical and analytical rigor, obtuse and convoluted texts), I'm finding now that, as a Web applications developer, a study of the basic tenets of computer science can help me create more sophisticated applications.

Case in point: the study of data structures and algorithms.

I recently had a programming problem to solve, one that lent itself perfectly to a solution offered in any data structures textbook. My problem was this: I was working with librarians on our campus on a project, part of which entailed the creation of a subject terms list. Once compiled, this hierarchical list of subject terms would be used as a controlled vocabulary to index items in our database. Although the list of terms the librarians came up with consisted of two levels (a general term and a specific term), there was no guarantee that in the future there wouldn't be a need to further subdivide the second level into a third level, and that level into a fourth and so on. In short, the number of levels in the hierarchy could be variable. So how should the data for this project be stored, and how should I then programmatically extract it as needed?

After pondering several scenarios involving multiple tables and confusing looping mechanisms, I decided to consult a colleague for advice. He took one look at the situation and, having had solid training in computer science theory, planted two buzzwords in my head: *tree* and *recursion*. Armed with this information, I scoured the Web and the library in search of information about trees as data structures and how recursive functions can be used to navigate them.

### Storing a Tree Structure

Insofar as the list of subject terms is in a hierarchy, with a term serving as root and subterms serving as branches, it is properly described as a tree data structure. My solution with respect to how it should be stored in the back-end database is to store the entire tree in a single table consisting of three fields. The first field, "SUBJECTID", serves as the primary key, uniquely identifying each row. The second field, "SUBJECT", holds the text of the subject term itself, for example, "Italian Landscapes." The third field, "PARENTID", serves as a pointer to the next item up the hierarchy from the current term. Thus the parent of the term "Italian Landscapes" might be something like "Italian Art," and the parent of the term "Italian Art" might be something like "Art History" and so forth. In this manner, each term in the table is linked to its conceptual parent all the way to the top of the tree where there are base terms that themselves don't have parents. These terms represent the starting points in the tree structure and have, by default, PARENTIDs of 0. Figure 1 represents how this tree structure is stored in a table (the "lookup-Subjects" table) in the back-end database.

### Recursion

But now that the data is properly stored, the question remains: How do we programmatically access it? For instance, chances are at some point we'd want to print out the entire table in outline form, with those subject terms lacking parents at the base and other items down the branches printing in indented form. How to do this? There must be some sort of loop involved here, but what is the nature of this loop? How does it know, in a structure where the length of each branch is variable, when to stop? The conceptual key to doing this is known as *recursion*.

A recursive function is a function (or method, custom tag or other chunk of code) that calls itself. It is essentially a loop, executing over and over again until some sort of “base condition” is met. In fact, the tasks that many recursive functions are used to accomplish can instead be carried out using various FOR and WHILE loops, but using a recursive function is often thought to be a more elegant solution.

To print out the contents of the lookupSubject table in outline form, I created a custom tag called “simpleBuildTree”. This tag results in the output represented by Figure 2.

Listing 1 is the code for the cf\_simpleBuildTree custom tag. It is worth going over line by line.

```
cf_simpleBuildTree
```

First, this custom tag is called in the usual way:

```
<cf_simpleBuildTree>
```

Line 1 of the tag sets a default value of 0 for a variable, #theID#, which is used to retrieve records from the database later on. If, however, #theID# is being sent as an attribute to the tag, it is rescope in lines 3–5 so that it can simply be referred to as #theID# instead of #attributes.theID#.

Lines 7–12 represent the “getCurrentItems” query. This query selects all records from the table whose PARENTIDs match the value of the current #PARENTID# variable. This value by default is set to 0, so the first time this query is run it will return all records appearing at the very top of the tree structure, that is, all records that are themselves not children of any other parent record. Referring back to Figure 1 for a minute, only two records fit this criterion – the record for “Arts and Humanities” and the record for “General and Reference.” Thus these two entries will serve as roots in the tree structure – and from these two roots all branches will grow.

Line 14 begins an unordered list, and line 16 begins a loop through the records retrieved by the getCurrentItems query (our two records).

Lines 18–20 output the current value of #getCurrentItems.subject# as a list item within the unordered list.

Line 22, however, is where the fun begins.

It’s possible that the current value of #getCurrentItems.subject# does not have any children; that is, it’s not the conceptual parent of any other subject term in the table. But it’s also possible that it has conceptual children, in which case we need to traverse down the tree, finding and outputting them all. The “checkForChild” query on lines 22–26 lets us know whether or not the current item has children attached to it. It does this by selecting records whose PARENTIDs match the current #SUBJECTID#.

On line 28 the recordcount for the checkForChild query is compared to 0. If it’s greater than 0 we know there must be children. If there are children, we need to loop back through the whole process, finding and outputting each one along the way. This is what the recursive call on lines 29–31 does: it calls the cf\_simpleBuildTree tag itself, this time passing a new value for #theID# that is equal to the current value of #getCurrentItems.subjectid#. At this point execution within the tag occurs within a completely new context – the context of the next item down the hierarchy.

Within this context line 14 begins a new unordered list nested within the unordered list a level up the hierarchy. Because HTML deals with nested lists properly, we’re presented with the correctly indented outline illustrated in Figure 2.

To back up for a minute to our first iteration through the getCurrentItems recordset, if the checkForChild query is run on the current value and comes up with nothing, nothing else

subjectid	subject	parentid
1	Arts and Humanities	0
2	General and Reference	0
3	Art History	1
4	Classics	1
5	English and American Literature	1
6	French	1
7	Film and Media Studies	1
8	Biography	2
9	Italian Art	3
10	Italian Portraiture	9
11	Italian Landscapes	9
12	Russian Art	3
13	Ancient Greek Literature	4
14	Ancient Roman Literature	4
(AutoNumber)		0

FIGURE 1: The lookupSubjects table

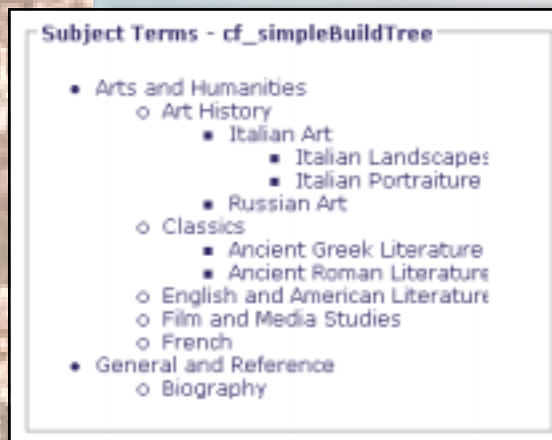


FIGURE 2: Output of cf\_simpleBuildTree



FIGURE 3: Calling cf\_reverseTree by itself

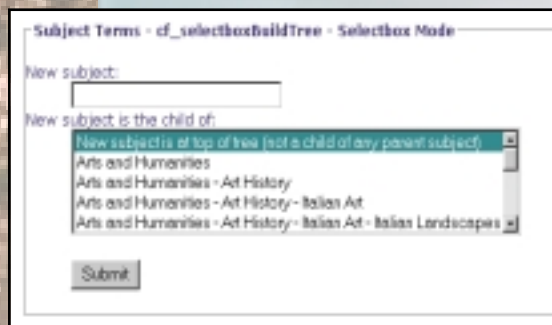


FIGURE 4: cf\_selectboxBuildTree on an HTML form

occurs; the code simply outputs the current value of #SUBJECT# as a root entry in the tree and moves on to the next record.

In this manner recursion can be used to process data contained in a tree structure. The important thing to remember here is that (1) a recursive call is being used to create a loop, and (2) there is a base condition that can be tested against to end the loop; in this case the base condition is reached when the checkForChild query returns a recordcount of 0, indicating that traversal down that particular branch has reached an end.



## cf\_buildTree and cf\_reverseTree

cf\_simpleBuildTree works great and outputs the data in a nice outline form. But suppose we wanted to additionally output each branch as one long string? Suppose we wanted output to look like:

```
Arts and Humanities
Arts and Humanities – Art History
Arts and Humanities – Art History – Italian Art
```

The first thing we'd need to do is put some switches in to let the program know whether we're in "outline" or "string" mode, then pass in an appropriate value indicating which mode we want. So if we just copy cf\_simpleBuildTree to a new file, say, cf\_buildTree, and make the edits, it can be called by:

```
<cf_buildTree
mode="outline"
>
```

The output in this mode is exactly the same as that in Figure 2.

Then it's a matter of editing it and putting in the aforementioned switches. Listing 2 contains the final code for cf\_buildTree, with any code referring to outline mode wrapped in the appropriate conditional code.

The major change required to output subject terms in string mode begins on line 36, within the checkForChild.recordcount conditional. Notice that, within string mode, this tag never actually outputs anything. Rather, beginning on line 37, it calls another custom tag, "cf\_reverseTree", passing it a value for the #theID# variable.

Here's what's going on: to output the subject terms in a string format such as "Arts and Humanities – Art History – Italian Art," the cf\_buildTree tag must somehow keep track of the entire list of these terms as it traverses the tree. But the recursive function it employs really doesn't do that; once it has traversed a branch to its end, it simply backs up one step and processes any other children left in the previous loop. If none are found, it keeps sequentially backing up the hierarchy, processing each branch, until it hits the root. Then it begins processing the next root term and all of its branches. The key thing to note here is that it's not backing all the way to the root each time – just one or two (or three or four...) levels until it finds a child in need of processing. Because this particular tree is an "unbalanced tree," that is, a tree whose respective branches may be of differing lengths, one cannot simply push and pop items onto a stack and then expect that stack to accurately represent a particular branch in string format.

The recursion employed by this tag and the cf\_simpleBuildTree tag determines the end nodes or leaves of each tree – it's what's known as a "depth first" search. Once we know this, we must do a reverse lookup of the parents of the leaf all the way back to the root. This is what the cf\_reverseTree tag does on lines 37–39. Code for this tag is shown in Listing 3.

First, notice that this tag can be called directly with the following call:

```
<cf_reverseTree
theID = 9
>
```

This results in the output shown on Figure 3.

How does cf\_reverseTree work? Pretty much the same way that cf\_buildTree does, only in reverse order. It takes as an attribute a SUBJECTID and, based on that SUBJECTID, it traverses up the tree to the root. As it does this it prepends the value of the #SUBJECT# variable to a list that is then passed to any other recursive iteration of itself. When the top of the tree is reached, it outputs the contents of the list. This output represents the entire branch in string format, from root to leaf.



One last thing to note about this tag is that, on line 25, a <CFEXIT> tag is called. This is done so that, if there is a parent to the current child, the contents of #theList# are not output; it ensures that #theList# is output only once, when the top of the tree is reached.

Returning to cf\_buildTree, we see that, on lines 35–45, if checkForChild.recordcount is not 0, and if the mode of operation is "string" the cf\_reverseTree tag is called, which outputs the branch in string mode. However, if there are no children to the current node, cf\_reverseTree is likewise called. This occurs on lines 45–51.

## Further Modification: cf\_selectboxBuildTree and cf\_selectboxReverseTree

How might the string mode of this tag be used? One use for it would be to dynamically populate an HTML selectbox.

Listings 4 and 5 illustrate the edited cf\_buildTree and cf\_reverseTree (saved as cf\_selectboxBuildTree and cf\_selectboxReverseTree, respectively) needed to do this. Listing 6 illustrates how this new tag is called. Its output is represented in Figure 4.

cf\_selectboxBuildTree is roughly the same as cf\_buildTree; the main difference is the new attribute passed to the cf\_selectboxReverseTree tag on lines 40 and 52. This attribute, named "initialID," passes in the value of the current SUBJECTID and will be used later by the cf\_selectboxReverseTree tag to output an HTML <OPTION> tag with the proper SUBJECTID value.

Taking a look at the code for cf\_selectboxReverseTree (see Listing 5), we see that the value of #INITIALID# is passed to every recursive iteration of the tag, on line 34, without its value ever being changed. Hence the value of the original SUBJECTID is retained and finally output as the value of the HTML <OPTION> tag on line 43.

In this manner a tree structure can be processed and output in "string" mode, and such string output can be used to dynamically populate an HTML selectbox.

## A Note on Performance

It is important to note that recursive functions are extremely resource intensive. This isn't surprising; each time a recursive call is initiated, the tag itself must be reexecuted while simultaneously retaining the state of previous iterations. Further, in the examples offered in this article there is the additional overhead of having the data stored in a back-end database. Processing a medium-sized tree structure with a recursive custom tag could result in literally hundreds of hits on the database server, which could be very time-consuming. Every effort should be made to speed up or eliminate these transactions. A significant performance gain can be made, in the cases illustrated above, by simply caching the queries for a brief period – especially the queries in the reverse lookup tags – so they are not duplicated. Then, if a query was previously run from within a recursive iteration, it need not be run again; rather, its results will simply and quickly be read from cache.

Even so, careful attention must be taken when deciding to use a recursive call to process data – and the decision to use a recursive call should be relative to the amount of data and the resources you have available for processing.


## Conclusion

Robert Lafore, in his lucid and readable book *Data Structures and Algorithms in Java*, points out that it's not enough to learn the syntax of a programming language – one must also learn how best to use that language to manipulate data in an efficient manner. And the study of data structures and the algorithms appropriate for processing them is the first step in that direction, a path that results

ultimately in superior application design. For someone who wasn't a computer science major in college, that's solid advice.

## Acknowledgments

The author wishes to thank his colleagues for their special contributions to his understanding of data structure and algorithms: Ian Goh for pointing the way and Craig Turkington for his insightful

comments on this article in draft. Both were computer science majors. 

### About the Author

Mark Cyzyk is an Allaire certified ColdFusion developer and Web developer at Johns Hopkins University and Nine Web, LLC, where he uses ColdFusion to create dynamic database-driven applications for his clients.

MCZYK@JHU.EDU

#### Listing 1: cf\_simpleBuildTree

```
1 <cfparam name="theID" default="0">
2
3 <cfif IsDefined("attributes.theID")>
4 <cfset theID = attributes.theID>
5 </cfif>
6
7 <cfquery datasource=#application.datasource#
  password=#application.password# name="getCurrentItems">
8 SELECT *
9 FROM LOOKUPSUBJECTS
10 WHERE PARENTID = #THEID#
11 ORDER BY SUBJECT
12 </cfquery>
13
14 <ul>
15
16 <cfloop query="GetCurrentItems">
17
18 <cfoutput>
19 <li>#subject#
20 </cfoutput>
21
22 <cfquery datasource=#application.datasource#
  password=#application.password# cachedwithin=#Create-
  TimeSpan(0,0,0,2)# name="checkForChild">
23 SELECT SUBJECTID
24 FROM LOOKUPSUBJECTS
25 WHERE PARENTID = #GETCURRENTITEMS.SUBJECTID#
26 </cfquery>
27
28 <cfif checkForChild.RecordCount gt 0>
29 <cf_simpleBuildTree
30 theID = "#GETCURRENTITEMS.SUBJECTID#"
31 >
32 </cfif>
33
34 </cfloop>
35
36 </ul>
```

#### Listing 2: cf\_buildTree

```
1 <cfparam name="theID" default="0">
2 <cfparam name="mode" default="outline">
3
4 <cfif IsDefined("attributes.theID")>
5 <cfset theID = attributes.theID>
6 </cfif>
7
8 <cfif IsDefined("attributes.mode")>
9 <cfset mode = attributes.mode>
10 </cfif>
11
12 <cfquery datasource=#application.datasource#
  password=#application.password# name="getCurrentItems">
13 SELECT *
14 FROM LOOKUPSUBJECTS
15 WHERE PARENTID = #THEID#
16 ORDER BY SUBJECT
17 </cfquery>
18
19 <cfif mode IS "outline"><ul></cfif>
20
21 <cfloop query="getCurrentItems">
```

```
22
23 <cfif mode IS "outline">
24 <cfoutput>
25 <li>#subject#
26 </cfoutput>
27 </cfif>
28
29 <cfquery datasource=#application.datasource#
  password=#application.password# cachedwithin=#Create-
  TimeSpan(0,0,0,2)# name="checkForChild">
30 SELECT SUBJECTID
31 FROM LOOKUPSUBJECTS
32 WHERE PARENTID = #GETCURRENTITEMS.SUBJECTID#
33 </cfquery>
34
35 <cfif checkForChild.RecordCount gt 0>
36 <cfif mode IS "string">
37 <cf_reverseTree
38 theID = "#GETCURRENTITEMS.SUBJECTID#"
39 >
40 </cfif>
41 <cf_buildTree
42 theID = "#GETCURRENTITEMS.SUBJECTID#"
43 mode = "#MODE#"
44 >
45 <cfelse>
46 <cfif mode IS "string">
47 <cf_reverseTree
48 theID = "#GETCURRENTITEMS.SUBJECTID#"
49 >
50 </cfif>
51 </cfif>
52
53 </cfloop>
54
55 <cfif mode IS "outline"></ul></cfif>
```

#### Listing 3: cf\_reverseTree

```
1 <cfparam name="theList" default="">
2
3 <cfif IsDefined("attributes.theID")>
4 <cfset theID = attributes.theID>
5 </cfif>
6
7 <cfif IsDefined("attributes.theList")>
8 <cfset theList = attributes.theList>
9 </cfif>
10
11 <cfquery datasource=#application.datasource#
  password=#application.password# cachedwithin=#Create-
  TimeSpan(0,0,0,2)# name="getCurrentItems">
12 SELECT *
13 FROM LOOKUPSUBJECTS
14 WHERE SUBJECTID = #THEID#
15 </cfquery>
16
17 <cfset theItem = #getcurrentitems.subject#>
18 <cfset theList = listPrepend(theList, "#theItem#")>
19
20 <cfif #getcurrentitems.parentID# IS NOT 0>
21 <cf_reverseTree
22 theID = "#GETCURRENTITEMS.PARENTID#"
23 theList = "#theList#"
24 >
25 <cfexit>
26 </cfif>
```

#### Listing 4: cf\_selectboxBuildTree

### Listing 5: cf\_selectboxReverseTree

### Listing 6: A form calling cf\_selectboxBuildTree

CODE  
LISTING

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



# Making Assertions

Produce code that's robust and scalable



BY  
HAL  
HELMS

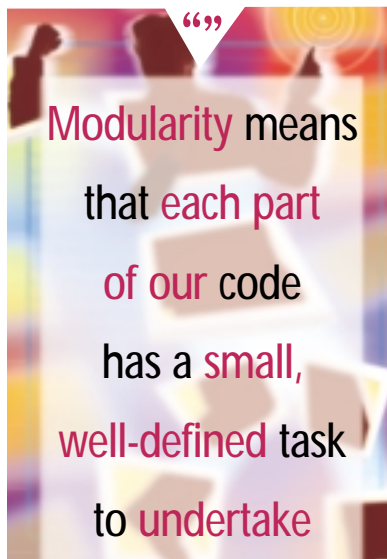
I'm a great believer in modularity of code. Modularity – breaking code into distinct pieces with a well-defined responsibility – is crucial if large-scale development projects are to be on time, on budget, scalable, maintainable and robust.

Modularity alone is no talisman. There's nothing magical about breaking code into pieces; a poorly organized set of a thousand lines of code can be just as disorganized when turned into 10 sets of a hundred lines. But often, long code sections indicate a lack of organization when the code grows topsy-turvy, with new functionality heaped onto old. Keeping modularity in mind when architecting a project can prevent run-away code. In programming, run-away anything isn't desirable.

Modularity means that each part of our code has a small, well-defined task to undertake. Fusebox ([www.fusebox.org](http://www.fusebox.org)) is a popular development methodology that's built to take advantage of modularity. In my work with Fusebox I put together a documentation standard called "Fusedoc." You can read more about this in previous issues of **CFDJ** (Vol. 2, issues 1, 2 and 3). The point I want to make is that having a documentation standard that tells the coder exactly what variables will be passed into/out of the code is a tremendous help in cutting down on runtime errors.

Modules typically don't have control over the variables passed into them. Other coders may call your module, or you may receive information from a credit card authorizer or some other third-party software. Even if you're writing all the code, you're encouraged to think of each module as a separate, stand-alone black box (the computer term is *encapsulation*). Your module should:

- Assume as little as possible about the world outside its scope.
- Make clear what its assumptions are.
- Protect itself (and its users) when those assumptions prove to be false.



Over time, programmers have developed the notion of "assertions" to protect code against false assumptions. Assertions are statements that the programmer believes to be true at runtime. They're particularly valuable when dealing with variables that will be present when the code executes. For example, if you're expecting to be passed a variable called `itemPrice`, you might assert that `itemPrice` should be a numeric variable, and it shouldn't be a negative number or have a value of zero. In case an assertion

fails, we want some sort of error-handling routine to engage.

Some languages have built-in support for assertions. ColdFusion doesn't, but I'm going to demonstrate how to create your own assertion facility that you can reuse for all your fuses (or whatever type of modules you use). Begin by declaring your assertions in a custom tag called `assert.cfm`.

```
<cf_assert
  assertion = "
    myAge as a: IsNumeric( |a| );
    |a| GT 0; |a| LT 100
    myName as b: IsDefined( '|b|'
    ); Len( '|b|' ) GT 4">
```

The syntax may look a little odd, but it's really quite simple. Begin with the name of the variable that will be live when `assert.cfm` is run, aliasing it the way you would within a SQL statement: `myAge as a`. The alias is used purely as a shortcut (laziness being one of the great virtues of all successful programmers). When you need to refer to the variable in the assertion, just wrap the alias in pipe symbols, as shown above.

In the foregoing example the call to the custom tag makes assertions about two variables. Use as many as you wish, separating each variable/assertion pair with a carriage return. For each pair a colon separates the variable section from the assertion section. For example, the assertion section for the variable `myAge` states that the variable is numeric and its value should be greater than 0 and less than 100.

Assertions can be any valid ColdFusion statement that's evaluated to True or False. These are all valid assertions as ColdFusion can evaluate them all to a Boolean value:

```
myVar as a: IsQuery( |a| )
myVar as a: ListFirst( |a|, '.'
) IS 'fnc'
myVar as a: DatePart( 'yyyy',
|a| ) GTE '2000'
```

*Note:* When the variable name is to be evaluated (rather than its value), place the alias (including its pipes) in single quotes. An example of this is the assertion that `myName` is, in fact, defined.

What if an assertion fails at runtime? The custom tag `assert.cfm` then throws an error that you pick up with

<CFTRY>/<CFCATCH> blocks that surround the call to assert.cfm.

```
<cftry>

<cf_assert
  assertion = "
    myAge as a: IsNumeric( |a| );
|a| GT 0; |a| LT 100
    myName as b: IsDefined( '|b|'
); Len( |b| ) GT 4">
<cfcatch type="FailedAssertion">
  #cfcatch.message#
</cfcatch>
</cftry>
```

If you look at the code for `assert.cfm` (see Listing 1), you'll see that I use `<CFTHROW>` to generate an error of a custom type: `FailedAssertion`. My `<CFCATCH>` tag looks for such an error type and dis-

plays the assertion that failed. Of course, in production code I'd have some actual error-handling deal with the error. Finally, `assert.cfm` looks for the presence and value of a global Boolean variable, `request-testAssertions`. If this value is `FALSE`, the tag will return without doing anything. This lets you turn off assertion checking during development.

If assertions are new to you, I encourage you to try them. They can be key allies in producing code that's robust and scalable. And while assertions should be used for protecting code against anomalies, they shouldn't be used instead of client-side validation. Validation is better done with JavaScript in the client's browser. 🍷

HAL.HELMS@TEAMALLAIRE.COM

### Listing 1: assert.cfm

```
<!-- assert.cfm -->
<!-- hal.helms@TeamAllaire.com -->

<!---
|| Responsibilities: First, I make sure that I'm
supposed to be checking assertions. If so, then I
check the assertions sent to me to ensure that all
assertions are TRUE. If any assertions fail, I
CFTHROW an error back to the calling page.
||
-->
||
END FUSEDOTC--->

<cfoutput>

<!---@COMMENT: If testAssertions was turned off,
then return without doing anything.--->
<cfparam name="request.testAssertions"
default="TRUE">
<cfif NOT request.testAssertions>
  <cfexit method="EXITTAG">
</cfif>

<!---@COMMENT: Make sure an assertion was sent to
me.--->
<cftry>
<cfparam name="attributes.assertion">

<cfcatch>
<cfset caller.assert = "FALSE: No assertion provid-
ed">
</cfcatch>
</cftry>

<!---@COMMENT: Loop over the assertions, getting a
```

```

handle on each variable/assertion pair--->
<cflowop list="#attributes.assertion#" index="variableAssertion" delimiters="#"chr( 10 )#">

    <!----...then separate the pair into various components--->
        <cfset varSection = Trim( GetToken( variableAssertion, 1, ':' ) )>
        <cfset varName = Trim( ListFirst( varSection, ' ' ) )>
        <cfset varAlias = Trim( ListLast( varSection, ' ' ) )>
        <cfset assertionSection = Trim( GetToken( variableAssertion, 2, ':' ) )>

    <!----...substitute the variable names for the aliases--->
    <cflowop list="#assertionSection#" index="anAssertion" delimiters=";">
        <cfset assertion = ReplaceNoCase( anAssertion, '|#varAlias#|', 'caller.#varName#', 'ALL' )>

    <!----...and if an assertion fails, throw the bum out.--->
        <cfif NOT ( Evaluate( Trim ( assertion ) ) )>
            <cfthrow type="FailedAssertion" message="The assertion #assertion# failed">
        </cfif>
    </cflowop>
</cflowop>
</cfoutput>
```

CODE  
LISTING  
■■■■■■■■■■■■■■■■

CODE  
LISTING

The code listing for  
this article can also be located at  
[www.ColdFusionJournal.com](http://www.ColdFusionJournal.com)

# 'The Ten Commandments' – Revisited



BY  
BEN  
FORTA

Even the holiest rules need a little update now and then

It's been about five years since I inscribed my "Ten Commandments of ColdFusion Development" for my first ColdFusion book, and as Commandments should, they've remained the same (more or less) with each subsequent revision.

Well, it had to happen. Unlike the other (and far more famous) Ten Commandments, mine were starting to stale and show signs of aging. So this month, in honor of the Second Annual Worldwide Allaire Developer Conference, I present to you "The New and Improved Ten Commandments of ColdFusion Development."

Nothing here is unique to CF development. These Commandments are just common sense, no matter what development you're doing. In addition, the Commandments are intended as starting points, not the final word. I strongly urge you to create a list (or adapt mine) of rules to govern your own development. Do so and be blessed with peace, prosperity...okay, I won't go there.

## I: Plan Before You Code

We've all done it, and probably more than once. ColdFusion makes it so easy to start coding that often there's the temptation to start projects by firing up ColdFusion Studio 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. And I don't mean planning your IPO.

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 (you might try, but you'd never get the necessary permission to begin work); building an application is no different.

I'm constantly amazed by the number of applications I'm 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. Scalability problems? I wouldn't doubt it. I'd actually be amazed if such an app ever did scale. You can't expect scalability from an application that grew in spite of its developers. Nor can you expect it to be bugfree, manageable or delivered on time.

Yes, I know that detailed planning takes time, time none of us have. But in the long run you'll come out ahead.

## 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.

Going back to the prior Commandment, "Plan Before You Code," 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. Allaire

hasn't published formal recommendations on coding standards, nor in my opinion should they. Allaire's job is to create killer tools and products for us developers; our job is to use them in the way that works best for us. I don't believe a single set of coding standards would work for all developers, but I also don't believe any developer should be writing code that doesn't adhere to a standard — any standard.

Coding standards include everything from file and directory naming conventions to variable naming conventions, code organization and ordering within your source, error-handling, componentization and much more. For example, if all variables that contain dates begin with "dt", then references to a variable named "dt\_orderDate" become self-explanatory.

The purpose of coding standards is to ensure some level of consistency in your code. Whether it's to allow other developers to understand and work with your code or simply so 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's no right or wrong coding standard as long as it's used. The only thing wrong with coding standards is not using them.

(If any of you have coding standards that you've developed and are willing to share, please e-mail them to me. I'll try to compile them for a future column, and if I use yours I'll credit you.)

## 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's not commented, that's how serious an offense I believe this to be).

Every source code file needs a header listing a description, 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's 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. You might even be able to make code changes safely without breaking things in the process.

### V: Never Make Changes on a Live Server

Another obvious one, 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 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, but the cost of a new box is nothing compared to the cost of bringing down your application because that little change wasn't 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, 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, weekly or whenever works best for you.

The key is that your production server is sacred. Don't touch it at all unless you have to...the less frequently the better. And never, ever, make changes on them, even minor ones.

### VI: Functionality First, Then Features

Another obvious one, and a common beginner's mistake. Yes, writing fancy DHTML menu-generation code is far more fun than writing data-entry validation routines, but the latter is far more important to the success of your application. Concentrate on creating a complete working application, then pretty it up as needed.

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

### VII: Build and Test Incrementally

You'd be amazed (or maybe you wouldn't) 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."

What bothers me isn't being sent the messages and requests (I know I'm going to regret saying this, but I really don't mind those at all). What really bothers me is that core code was never tested in isolation. This goes back to the prior Commandment, "Functionality First, Then Features." 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-mirror 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.

### VIII: Never Reinvent the Wheel, and Plan Not To

I've written about this one extensively, especially in this column (see

**CFDJ**, Vol. 2, issue 2, "Preserve Precious Resources – Recycle"). 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's not hard-coded or highly task specific unless it absolutely has to be.

The benefits? Being able to reuse existing code will shorten your development time. You'll also stand a far greater chance of creating bugfree 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 benefits. Lots of benefits and no downside whatsoever. Should be a no-brainer.

### IX: Use All the Tools at Your Disposal, Not Just ColdFusion

Another one I've written about before (see **CFDJ**, Vol. 1, issue 3, "Take Your Database Out of Retirement" and **CFDJ**, Vol. 2, issue 3, "When Not to Use ColdFusion"). Unlike the other Commandments, this one is more ColdFusion specific.

CF applications aren't usually stand-alone entities. They rely on database server, mail servers and much more. In addition, ColdFusion can leverage COM, CORBA, C/C++ code, and all sorts of Java-based bits and pieces. Use these tools, as many as needed, and always attempt to pick the best one for a specific job. The best CF applications aren't the ones written purely in ColdFusion; they're the ones that leverage the best technologies for the job, all held together by ColdFusion.

### X: Implement Version Control and Source Code Tracking


Source code changes. And changes are dangerous. As your applications grow, so does the need for tracking changes and source code control. Pick a version control package that works for you, and use it. Key features to look for are:

- The ability to lock files (so no one else edits a file while you edit it; if that happens someone's changes will be lost)
- The ability to view change history (what changed, by whom, when)

- The ability to roll back complete applications (so when the latest upgrade bombs you can easily roll back an entire application to a prior known state)
- The ability to create and check file dependencies (so you'll know what other code is affected by changes you make)
- Reporting

If you can integrate the product with ColdFusion Studio, that's even better. The bottom line – I don't care what product you use, just use one.

### Summary

There you have it – my new and greatly improved “Ten Commandments of ColdFusion Development.” Movie rights haven't been sold yet, and the lead role hasn't been cast (the obvious choice is busy with the NRA right now). If any progress is made in that department I'll let you know. In the meantime, good luck with your coding, and I look forward to meeting you in November at the Developer Conference in Washington, DC. 

## BEN FORTA RESPONDS...

To: ben@forta.com

Re: “Access Denied” (*CFDJ*, Vol. 2, issue 6, see below), by Ben Forta

*File-based databases are cheap and easy to use, and for smaller applications they usually perform quite well, sometimes even outperforming client/server databases. But they're also highly susceptible to data corruption and are terribly insecure. For most organizations there's nothing more precious than data and if this is true of your organization, then file-based databases should never be used with ColdFusion. This alone should be enough to push you toward client/server databases.*

Would MySQL have the same problems as Access with ColdFusion?

Paul Berenson  
paul@paulb.com

MySQL is a true client/server database, and if used properly will not suffer those risks. However, many users put MySQL on the same box as the Web server, which effectively introduces all the risks.

Ben Forta

### ABOUT THE AUTHOR

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

BEN@FORTA.COM

# AuctionBuilder Pro! 1.0

## by AbleCommerce

REVIEWED BY  
CAREY  
LILLY



Slick and intuitive, this app makes auction commerce worth bidding on

Name your price seems to be the catch-phrase in e-commerce these days. The success of sites like eBay and Priceline testify to that.

Not long ago retailers were scrambling to set up online stores. Now they're scrambling to set up online auctions. If you're champing at the auction bit for your own site, or are facing the task of building an auction application for your clients, AuctionBuilder Pro! from AbleCommerce may be just what you need.

Having had the experience of writing a custom auction application for one of my own clients, I had some insight into the complexity of the task that faced developers. My foray into this area had introduced a number of very involved problems, and that was a much simpler app than would be needed for a large auction site. AuctionBuilder Pro! provides not one, but three different auction environments, and does so in a manner that allows for quick customization, altering the environment to specific markets, and a surprisingly easy-to-use administration interface, considering the difficulties involved.

### What You Get

I tested AuctionBuilder Pro! version 1.0, which requires ColdFusion Professional 4.5 to run. Installation was quite painless and included no surprises. In fact, it took only about 20 minutes to install, set up and begin running my initial tests.

Installation on an NT Workstation with IIS was no problem, but I decided to run it on my creaky old Pentium system running Win98 and Personal Web Server. It was my thought that this would give me an idea of what the system would be like running over the Net and/or on a heavily loaded server. I will say that it performed very well under these circumstances.

The package includes three licenses, plus source code. Also provided is

a complete context-sensitive help and a programmer's reference for tailoring the product to your client. AuctionBuilder Pro! comes with three predefined auction environments:

- **AbleAuction:** This template is primarily for business-to-consumer sites that seem to be the primary targets for this product.
- **BidKingdom:** This template is probably the model you're most familiar with from other auction sites on the Web. It allows both businesses and consumers to buy and sell in an open-market atmosphere.
- **BidMadness:** This model is a reverse auction concept, where buyers post items they want to buy. Sellers then compete to provide the lowest price, and the low bidder "wins" by making the sale.

### Setting Up an Auction Site

Since I'm a little more concerned with how the site will be managed rather than used, I dove right into the auction administrator. The first thing to note is that all password requests are confirmed. In other words, administrators have to enter their password twice. This ensures that an unauthorized user can't just use the "back" button and repost the password to gain access. And that brings up another thought: the administration environment has a very complete navigation system, but since the application is so complex, it's easy to get a little lost at first. That being the case, there is a temptation to bypass the navigation and just use the "back" and "forward" buttons on your browser. The problem is, using the buttons may create duplicate database entries in some cases (as it did

VITALS

**AuctionBuilder Pro!, v. 1.0**  
AbleCommerce USA

**Address:** 5139 NE 94th Avenue  
Vancouver, WA 98662

**Web:** [www.auctionbuilder.com](http://www.auctionbuilder.com)

**E-mail:** [info@ablecommerce.com](mailto:info@ablecommerce.com)

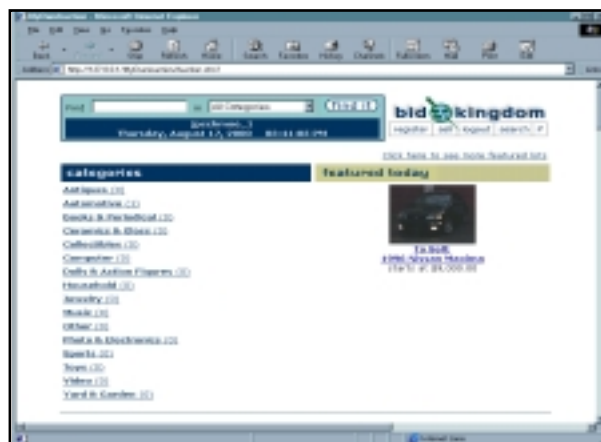
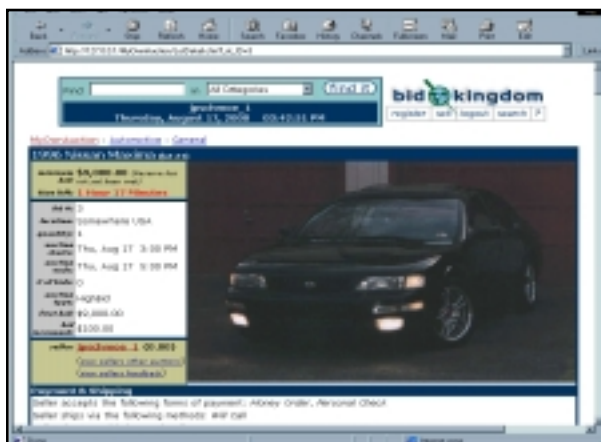
**Test Environment:**  
Requires ColdFusion Professional Suite 4.5. Installed on NT Workstation with IIS. Pentium system running Win98 and Personal Web Server.

**Pricing:** \$5,995  
Licensed for installation on one server and 3 auction halls with source code. Includes CF Professional 4.5

for me). Once you're familiar with the menus, it'll be a piece of cake, but you may have a shaky start to begin with. One nice touch that I appreciated was a menu icon that popped up a child window with the complete admin menu. If I got a little lost, this always helped me find my way back.

Your first admin page is where you set up "auction halls" for your site. Three licenses allowed me to create three auction halls based on each of the models mentioned above. An auction hall is essentially the auction environment in which site visitors will interact. Creating an auction hall is simple. Give it a name, and AuctionBuilder will create the appropriate data sources, server mappings and settings, all based on the model you select. In my testing I focused a bit more on the model that seemed to be what most site owners would want: the BidKingdom auction, where all users can browse listings and make purchases, and where pre-





mium (paid) members may create their own auctions.

Once the hall has been created, you select that hall to administer. The administration menu allows you to edit pretty much any data on the site, including auctions and lots up for bid. Activity logs and reports can be viewed from here as well. This is also where you'll set up some of the custom options, like item categories, fees for auction services, shipping methods and payment processors (AuctionBuilder Pro! includes support for CyberCash, Authorize.NET, and ICVerify).

## The User's End of Things

Now that I had an auction hall set up, it was just a matter of registering users and getting started with the selling and bidding. AuctionBuilder Pro! administrator gave me the URL to the auction site. One note: no "default" or "index" file is provided, so my Web server returned a 404 Error message when I went to the URL. Add the filename "auction.cfm" to the URL and you'll see the goods. Click the "register" icon and fill in the form provided. The registration process is fairly straightforward, but I did run into something that stopped me for a moment: entering an address requires an alias so that later users can simply select "home" from a drop-down list instead of retyping their address information. The same is true of users' payment information. A user can define one or more credit cards to pay with, and later just select one from a drop-down list using the alias. Your typical Web site visitors may not understand this right away, and since it's a required field they can't register without the alias defined. Be prepared to explain what

the alias is. The help explains this of course, but who actually reads that?

With all of the vital data added, I was now a premium member. Because premium members incur a fee in the default setting, they must be approved by an online credit processor or by the administrator. Presumably, members will be approved if their credit card payment is successfully processed online, but since I couldn't test that, I had to manually approve premium users. Once registered and approved, I was able to create auctions, bid on lots, win bids, receive outbid notices and generally do all the things I'm familiar with from other auction Web sites. In fact, having had some experience with eBay, I have to say that I like AuctionBuilder's out-of-the-box interface just a little better; it's slick and intuitive.

## Customizing

Included with the AuctionBuilder Pro! package is a 200-page programmer's reference in PDF format. After an exhaustive review of file structures, security features and database framework, there's a compact but complete set of directions for customizing most of the auction site. The critical question – How do I customize the "look and feel" of the site? – is answered in this section. A reference page guides you through global settings that allow you to make your AuctionBuilder Pro! site truly a part of your existing Web page. Other settings can be changed to allow you to customize how the site operates, and examples are given to guide you through these changes. It's my feeling that even a CF newbie could customize AuctionBuilder Pro! in a fairly short period of time. Also included in the programmer's refer-

ence are instructions for adding a payment processor, allowing you to use almost any method of payment that you or your clients demand.

AuctionBuilder Pro! is modular, which makes it easy to modify files for a particular activity. But its modular design also means you have to spend time tracing out exactly which files do what, and when. That being said, filenames are meaningful (e.g., search.cfm and sell.cfm are fairly descriptive), which makes it a great deal easier to locate what you're doing and where it needs to be changed. Much of the program uses CFMODULE to call additional files, and parameters are descriptive as well. In all, modifying the program's functions should involve a fairly short learning curve for most CF programmers.

## Conclusion

I stepped into this product somewhat gingerly, knowing from experience that the process of creating and running an auction Web site included some horribly complicated issues. This solution from AbleCommerce not only handles the complex tasks, but also breaks them down in a way that makes it quite easy. No matter what type of auction model your client needs, AuctionBuilder Pro! can handle it, and it can be modified to accommodate virtually any business. The program with source code and three licenses will run you \$5,995. This is not an application for your local mom and pop establishment. However, for most e-tail clients that want to move into the auction world, AuctionBuilder Pro! has everything you'll need.



## ABOUT THE AUTHOR

Carey Lilly is an associate with a Web site development firm based in the New York Metro area. He has been developing with ColdFusion since 1997 and has 10 years of experience with relational databases.

# Great Philosophers of Software Development

BY  
STEVEN D.  
DRUCKER

**M**en at some times are masters of their fates:  
The fault, dear Brutus, is not in our stars,  
But in ourselves, that we are underlings.  
—Julius Caesar, Act 1: Scene 2

While sometimes we're loath to admit it, as consultants we control our destiny. Despite applying the latest technologies, working long hours and starting initiatives with the best of intentions, software projects fail. These breakdowns take many forms. Some projects run over budget. Many initiatives fail to meet time-sensitive deadlines; others may fall prey to unexpected technical glitches or a genuine lack of development expertise. Developers criticize their clients for changing specifications. Clients criticize developers for missing deadlines. High-level executives are often left in the unenviable position of playing referee. Most often, however, projects are doomed from the start due to a failure by everyone involved to understand the true scope and nature of "the big picture."

*Alright brain, you don't like me and I don't like you but let's just do this and I can get back to killing you with beer.*

—Homer Simpson

Understanding requirements takes patience, commitment and an acknowledgment by all parties involved that the problem that needs solving is often completely different from the one that initially presents itself.

Frequently we encounter customers who initially chafe at the notion of paying for a detailed requirements analysis. They feel like they've thought through the problem and can't understand why we can't translate a 10-point bulleted list of requirements into a flat-fee price quote. To help them understand the challenges, I compare the software development effort to

building a bridge. The client has identified the need for crossing a body of water. Their functional spec contains the requirements for using steel (choosing CF as a platform): it must carry a certain number of cars per hour (scalability) and it should not under any circumstances fall down (failover, robustness). All other details are subject to interpretation. As application architects, it's our fiduciary duty to translate these broad requirements into finely detailed blueprints that can be handed off to the construction team. We must perform an environmental impact study to determine how the custom software will fit into a customer's business landscape. How will the site affect different business units? Will supplemental staff be required to maintain content or administer the application? Can we leverage other resources on the Web? Should we interface with databases both internal and external to the organization? Where will the company recoup their investment? What level of end-user or administrator training will be required? How will we measure success? Unfortunately, too many software initiatives charge ahead with arbitrary deadlines, inadequate resources and unclear goals because questions like these were neither asked nor answered. For our part, as developers, we're not immune to this criticism either. Frequently we're too willing to start coding before the appropriate level of specifications has been developed. Many of us lack the patience to sit through a facilitated requirements gathering process, and feel uncomfortable about voicing our reservations concerning designs we feel

don't make sense. The involvement of requirements facilitators such as project managers, business process modelers and management consultants proves invaluable.

*There's never time to do things right...but there's always time to do things over.*  
—Unknown

When specification phases are ignored, the inevitable result is that our development process degenerates into a multiple prototype approach. I've seen entire applications rewritten two, three, even four times before the final result is deployed. While not every contingency may be accounted for in the design phase, investing in a rigorous discovery and design phase certainly yields significant efficiencies over the brute force approach from every perspective. We can develop better products in less time for less expense. The process, however, may neither be shortchanged nor underappreciated.

*I have fought the fight, and now it's up to others to take the responsibility of leadership.*  
—Richard M. Nixon

This month we'll witness the introduction of new and exciting technologies at the Allaire Developer Conference. Don't let the challenges of mastering new techniques blind you to the real issue at hand – understanding your client's true requirements. Keep the big picture in focus and you'll never be led far astray. 

SDRUCKER@FIGLEAF.COM

## ABOUT THE AUTHOR

Steve Drucker is CEO of Fig Leaf Software, an Allaire Premier Partner, focused on giving clients a high return on their investment.



# A ColdFusion Based Oracle Database Monitor

BY  
KAILASNATH AWATI  
AND MARIO TECHERA

**Limitless possibilities  
with Web-enabled  
client/server applications**

*ColdFusion offers developers an easy way to Web-enable client/server applications. This fact has been noted and written about quite often in this journal (see, for instance, Jerry Bradenbaugh's article in the January 1999 issue of CFDJ [Vol.1, Issue 1]). Putting an application on the Web saves you the pain of client-side software installation and maintenance. Besides, the application then becomes truly portable, as it can be accessed from any browser anywhere in the world (at least in theory).*

There are several excellent client-server database administration (DBA) tools available on the market – DesktopDBA by the erstwhile Platinum Technologies and DBAr-tisan by Embarcadero Technologies are just two examples that come to mind. We would like to show you how to build a simple Oracle database monitor (hereby dubbed Ora-Fusion), armed with only a basic knowledge of ColdFusion and a casual acquaintance with the Oracle data dictionary.

Commercially available DBA tools go well beyond providing system and session-monitoring functionality. Most important, such tools have features that allow you to manipulate all kinds of database objects – tables, procedures, indexes. For simplicity, the scaled-down version of



OraFusion that we discuss does not include the ability to view and manipulate all possible database objects. In addition to a system and session monitor, our miniversion will include only the ability to extract table/view information from the database and to write and execute your own SQL. However, you will see that it is possible to extend the application to include whatever functionality you may require. For us it is an ongoing project – we add features to it as and when we find the time!

## Oracle-Related Issues

Oracle is a popular database of choice for mission-critical applications. Judging from the ever-increasing number of Oracle-related posts on the ColdFusion Forum (<http://forums.allaire.com>), it is evident that several Oracle installations are using the power of ColdFusion to Web-enable their applications. Now, unlike ColdFusion, it takes a while to become familiar with using the advanced features of the Oracle database server. Fortunately, the Oracle features we need to use can be conveniently discussed in a few paragraphs. We do this below.

## Connectivity with ColdFusion

We won't go into the details of how to set up and test the connectivity between ColdFusion and Oracle, as this has been amply discussed in the documentation (see the Allaire Web site for details). However, a few remarks regarding connectivity are perhaps in order. A ColdFusion server can establish a connection to an Oracle database via ODBC or a native driver. The latter is available only with the enterprise edition of ColdFusion. An obvious requirement is that our database monitor should work with either type of driver. The difference between the two is generally an issue only when you invoke stored procedures or use Oracle-specific features within your templates. We will avoid using such features, so our database monitor should work with either type of driver. We have successfully tested the code provided with both types of drivers. (The code listings for this article can be found on the **CFDJ** Web site, [www.coldfusionjournal.com](http://www.coldfusionjournal.com).)

## Oracle Data Dictionary and Dynamic Views

Oracle stores information about the data and structures in the database (metadata) in a set of special tables collectively known as the *data dictionary*. The information in the data dictionary can be accessed through a set of views that organize the information in useful

VIEW	DESCRIPTION OF CONTENTS
ALL CONSTRAINTS	Constraints defined on all tables accessible to the user
ALL CONS COLUMNS	Columns in constraints on all tables accessible to the user
ALL TABLES	Description of tables accessible to the user
ALL TAB COLUMNS	Descriptions of columns in all tables accessible to the user
ALL VIEWS	Description of all views accessible to the user

TABLE 1: List of ALL\_ views used in OraFusion

ways (note that these views may vary from one version of Oracle to the next). These views are broadly classified into three categories: USER\_ views, ALL\_ views and DBA\_ views. An example of each of these views is: USER\_TABLES, ALL\_TABLES and DBA\_TABLES. Any user who is allowed to create an Oracle session (i.e., log on to the database) has access to the USER\_ and ALL\_ views (see Table 1). As the name suggests, the DBA\_ views are accessible only to users with DBA privileges. A user querying the USER\_ views gets information about objects owned by him or her. The ALL\_ views return information on all objects accessible to the user – that is, the objects owned by the user and those objects to which the user has been granted access. The DBA\_ views contain information on all database objects. Additionally, some DBA\_ views provide more detailed information than the corresponding USER\_ and ALL\_ views.

The views described in the previous paragraph contain information about database objects. There is another extremely useful set of database instance level views that allow you to monitor database activity and performance. These views, which usually start with the characters VS (VSPARAMETER, VSSYSSTAT for example), can only be accessed by users with the system privilege “select any table” or with DBA privileges (which would generally include the “select any

table” privilege). These views are appropriately termed *dynamic views*, since they reflect the state of the database up to and at a given time. We will query some of these views to obtain system and session information in OraFusion. See Table 2 for a list of VS\_ views that we will use in our application.

## Application Security

Not all users should have access to all features of OraFusion. Clearly we would want only users with DBA privileges to view system and session information. In contrast, we would want ordinary users to see all the tables and views that they have access to. Fortunately we don't really need to do anything special to implement security, as Oracle ensures that a user will get to see only the data that he or she is authorized to.

All we have to do is find out if the user has permission to connect to the database. To do this we create a session, and then set appropriate ColdFusion session variables with the user identification information. Thereafter Oracle will check the user's data access permissions prior to executing subsequent queries. The only thing we need to remember is to trap any errors returned by Oracle. This is conveniently done via ColdFusion's exception handling tags – CFTRY and CFCATCH.

## Brief Overview of OraFusion

Before we dive into details it is a good idea to lay down the scope of our applica-

VIEW	DESCRIPTION OF CONTENTS
V\$ACCESS	Currently locked database objects and users accessing them
V\$DATABASE	Database information from the control file
V\$OPTION	Database options installed
V\$PARAMETER	Initialization parameters
V\$SESSION	Information on all current sessions
V\$SESSION_EVENT	Information on waits for events by all current sessions
V\$SESSTAT	Statistics for all current sessions
V\$SESS_IO	I/O statistics for all current sessions
V\$SGA	Summary SGA information
V\$SGASTAT	Detailed SGA information
V#\$STATNAME	Names of all statistics
V\$SYSSTAT	Cumulative statistics for the system
V\$SYSTEM_EVENT	Information on total waits for the system

TABLE 2: List of VS\_ views used in OraFusion





Each of the queries in Listings 5 through 8 go against VS views. We do not know whether a user has permissions to query these views, as our login procedure does not check for that. However, this doesn't matter because Oracle will throw us an error if the user doesn't have the appropriate permissions, and we already have a mechanism to pass the bad news back to the user. If we want to, we can even trap specific Oracle errors such as ORA-00942: Table or view does not exist (usually indicating that the user does not have appropriate privileges).

The session monitor consists of two templates – one showing an overview of all current sessions on the database with a drilldown to individual session details (see Listing 9 and Figure 5) and the other showing session details (see Listing 10). Again, the information displayed in these templates is extracted by querying the appropriate VS views – the relevant ones begin with the characters V\$SES. The drilldown to session detail is implemented as a hyperlink; the session ID is passed as a URL parameter.

Our scaled-down version of OraFusion allows users to view table and view data and information. Tables and views are quite similar so we can treat them using the same templates, using a session variable to keep track of which of the two we are dealing with.

Listing 11 displays a menu of available options. Here we implement only two options – table/view data and table/view information. The first option allows users to see all the data in the table and the second displays a list of table columns (with data types and other information) and con-



On making a selection from the table/view menu, the user is taken to a page that displays a complete list of tables/views that are accessible to the user (see Listing 12). This list is obtained by querying the `ALL_TABLES` view (remember that the `ALL_` views contain information of all objects accessible by the user). A short JavaScript function ensures that the user does select a table or view before proceeding to the next template.

Listing 13 is the template for table/view data display. There is one point here that's perhaps worth mentioning. We don't know



the column names until the user selects a table. So the display template has to evaluate column names and data values on the fly. This is conveniently done using the variable `queryname.columnlist` that ColdFusion creates with every query. A note of caution: our implementation will display all data in the table, which can be a dangerous proposition when a large number of records are returned. There are several ways to deal with this but we won't pursue them here.

Finally, Listing 14 is the template for table/view information display. Column information for tables and views is obtained by querying the view `ALL_TAB_COLUMNS`. For tables, constraint information is obtained from the view `ALL_CONSTRAINTS` and `ALL_CONS_COLUMNS` (see Figure 6). For views, the SQL text is obtained by querying `ALL_VIEWS`.

### Custom SQL

The final piece of our application is a feature that allows users to write and execute their own SQL on the Oracle server. The first template (see Listing 15) is a simple form with a text area in which users can compose their SQL. The action template (Listing 16) simply executes the SQL, and then displays the data (using the same technique as in the table data display template) if the statement was a `SELECT`, or an informational message otherwise. We could check for various types

The screenshot shows a web browser window with the title "SCOTT.EMP Columns and Constraints". It contains two main sections: "Columns" and "Constraints".

**Columns Section:**

Column Name	Datatype	Column Length	Data Grade	Column Permissions	Nullable
EMPID	NUMBER	4	1	SELECT, INSERT, UPDATE, DELETE	NO
ENAME	VARCHAR2	10	1	SELECT, INSERT, UPDATE, DELETE	NO
JOB	VARCHAR2	9	1	SELECT, INSERT, UPDATE, DELETE	NO
MGR	NUMBER	4	1	SELECT, INSERT, UPDATE, DELETE	NO
HIREDATE	DATE	7	1	SELECT, INSERT, UPDATE, DELETE	NO
SAL	NUMBER	7	1	SELECT, INSERT, UPDATE, DELETE	NO
COMM	NUMBER	7	1	SELECT, INSERT, UPDATE, DELETE	YES
DEPTNO	NUMBER	2	1	SELECT, INSERT, UPDATE, DELETE	NO

**Constraints Section:**

Constraint Name	Type	Referenced Table	Status
EMP_PK	PRIMARY KEY	SCOTT.EMP	Valid
EMP_FK	FOREIGN KEY	SCOTT.EMP	Valid

Below these tables, there is a section for "Constraint Columns" with a table showing the columns involved in the constraints.

FIGURE 6: Table Information display

of DDL/ DML statements using ColdFusion's rich set of string search functions, and then display a more informative message. Notice, again, that we do not bother with any error checking within ColdFusion, since the database will return an error if the statement does not make sense, or violates any access permissions.

### Wrapping Up

We hope we have shown you that building a basic database monitor using ColdFusion is really quite easy. It is also a good way to learn a bit about the Oracle data dictionary. This article discusses a simple implementation, leaving open several avenues for extending the application. Some of the more obvious extensions as:

1. Add more object management features such as Users, Roles, Procedures and Functions.
2. Allow the user to manipulate (ALTER, DROP, for example) objects in addition to displaying object information.
3. Add performance monitoring features.

In closing, we cannot resist making a final remark on the more general theme of this article. We have described a concrete example of how developers can Web-enable client/server applications using ColdFusion – further possibilities are indeed limitless. However, the lack of a runtime version (or pricing) for the ColdFusion server makes it difficult for developers to shrink-wrap their applications. Hopefully Allaire will address this issue in the near future.



### About the Authors

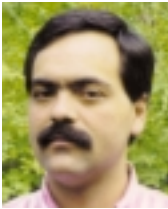
Kailasnath Awati develops Web-enabled database applications using ColdFusion and Oracle. He works for Williams & Partner Management Consulting, a Munich-based firm.

Mario Techera specializes in Oracle database design and tuning for Web applications. He also works for Williams & Partner, Management Consulting, where he is one of the founding partners.

kawati@orafusion.com

mtechera@wpmc.com

# Use WDDX to Store Complex Variables for Clustered Web Servers



BY  
ALAN  
McCOLLOUGH

Powerful new tool helps CF developers meet Web needs

In the beginning there was a Web site. Over time, the humble site matured into an e-commerce application. Hundreds of hits per day turned into thousands, then tens of thousands...until the server crashed and all the e-customers and all the e-money went elsewhere.

Innovation comes to the rescue, and somebody decides that servers can be lined up like phonebooths at a busy airport. If one server fails, the others will take over. If one server's too busy, others will take on the new demand. This concept is called *clustering*.

Today, a variety of schemes are available for clustering servers to form a single virtual server. Allaire provides Cluster Cats with ColdFusion Enterprise Edition. Microsoft provides Windows Load Balancing Service (WLBS) and Microsoft Cluster Services (MSCS). Others exist for platforms like Linux.

Not all clustering systems do the same thing or provide the same features, but they do share a common purpose – to unite multiple servers to appear as a single virtual server to users who access the server's resources. (Listings 1–4 can be found on the **CFDJ** Web site [www.coldfusion.com](http://www.coldfusion.com).)

## Problem 1: Loss of Persistence

Clustered servers work great for keeping a Web site online when a server (called a *node* in clustertalk) dies. However, any clients on the failed Web server might be in for a rude surprise when they hit the "Submit" button after 10 minutes of online shopping. HTTP 404 error! Or they get bounced back to the homepage of the Web site with a brand-new, totally empty shopping cart. What happened? What went wrong?

Let's say you have a CF application you wrote, and you use an array to hold a customer's online shopping basket. You might also use a structure to hold the same information. To keep this information available on the server side across multiple CF

templates, you use session variables. Session variables are stored in RAM on the Web server. In a clustered environment, however, the session variables stored on one node in a cluster won't be transferred to other nodes in the cluster in the event of a node failure (called a *failover* in clusterspeak). There goes the shopping cart, and your frustrated Web customer goes to the other guy's Web site to make that huge purchase. This annoying phenomenon of "having it and losing it" is what I call "loss of persistence."

## Solution 1: Database-Driven Client Variables

Allaire has provided a solution – database-driven client variables. If you snoop around inside ColdFusion Administrator on your Web site, you'll see an item in the Administrator's menu called *variables*. ColdFusion lets you store what are called *client variables*, which are indexed to a cookie that your client keeps on his or her system. (For information on using client variables, refer to "Creating and Manipulating Variables" and "Using the Application Framework"; both are in "Developing Web Applications with ColdFusion" which is part of your ColdFusion documentation.)

If client variables are enabled for your particular CF application, when a Web user visits your site the CF server checks for a particular cookie on the client side. If the cookie exists, the CF server then knows to associate the correct client variables with this user. Client variables work just like session variables; that is, you prefix the name of your variable with "client" instead of "session."

As an example:

```
<CFSET session.username = "Fred">
would be
<CFSET client.username = "Fred">
if you were using client variables.
```

In ColdFusion Administrator you'll see that client variables are normally stored in the Registry, which works great on a stand-alone Web server but doesn't do well in a clustered environment unless the clustering software specifically communicates Registry information between servers.

The solution is simple: you opt to store client variables in a database instead of the Registry. When client variables are stored in a database that's made available to all CF servers in a cluster, they'll survive a failover and your Web customers will never know that one of your servers just turned its power supply into a fried pork rind. For instructions on enabling database-driven client variables within a CF Application Server, read "So You Want to Manage a Session on Load-Balanced Servers," by Marc Furnaro (*CFDJ*, vol. 2, issue 6) available at [www.coldfusionjournal.com](http://www.coldfusionjournal.com). Do a search for cluster.

## Problem 2: Complex Variables Aren't Supported

When you use a database to store your client variables, you need to take into account one important detail: complex variables such as structures and arrays aren't supported, see Listing 1.

With database-driven client variables, when the last CFSET in this example attempts to execute, you'll receive an error indicating that complex datatypes aren't supported. What can you do? What about your existing applications

that make generous use of session variables that you want to convert to database-driven client variables?

## Solution 2: WDDX

Quoting from the [www.wddx.org](http://www.wddx.org) Web site FAQ, “WDDX (Web Distributed Data Exchange) is an XML-based technology that enables the exchange of complex data between Web programming languages...”

Starting with version 4.0, ColdFusion Server has WDDX capability built in. Listing 2 shows how to use this capability to solve our problem. WDDX will allow the expression of complex datatypes by using tags similar to HTML tags.

This would yield the variable “wddx\_MyArray”, which now contains:

```
<wddxPacket  
version='1.0'><header></header><data>  
<array  
length='1'><string>foo</string></  
array></data></wddxPacket>
```

This plain-text packet is then stored in the client variable “client.-MyArray”.

Listing 3 shows how to re-create the original array from the WDDX packet stored in the client variable.

The first line creates the empty array; the second line populates it with the stored data retrieved from the client variable.

## Implementing the Solution – A Fusebox App

Now you see how to move complex variables like structures and arrays in and out of database-driven client variables to survive a cluster failover. How do you integrate this solution into your programming?

In general, you need to re-create any complex variables before calling any ColdFusion templates. Then you need to serialize the variables into WDDX packets and store them in client variables after calling your CF templates. Listing 4 is an example application, written in the Fusebox style, that demonstrates how to implement the WDDX serializing and deserializing solution.

This app has been tested successfully with ColdFusion Enterprise Edition 4.5 and Microsoft SQL Server 7.0. You may need to make minor modifi-

cations to accommodate your particular site's configuration.

## Conclusion

By using the techniques provided in the sample application, ColdFusion developers working in a clustered server environment can now store all types of client information across all the nodes in their cluster. You no longer need to worry about your inability to store complex objects in client variables or sacrificing the flexibility of persistent variables.

WDDX is a powerful new tool available to the ColdFusion developer, and clustered server sites are becoming increasingly common as the demand for Web services grows beyond what a single server can supply. Database-driven client variables allow the CF developer to store persistent client information. By bringing these technologies together, you can create powerful, productive applications that meet the needs of today's demanding Web environment.



## ABOUT THE AUTHOR

*Alan McCollough is a Web programmer at the Alaska Native Medical center in Anchorage and a recently certified ColdFusion (4.5) developer.*

AMCCOLLOUGH@ANTHC.ORG



# CFUN-2k = CF Party

BY  
CHARLES  
AREHART



## Two days of sessions topped it off

**L**ate in July ColdFusion developers who simply couldn't wait for the November Allaire Developer conference had the opportunity to gather with compatriots and spend time learning CF tricks and techniques from some of the most popular CF speakers.

You may find this announcement only mildly interesting if you're fortunate enough to have a CF user group in your area. These groups are popular and meet all over the country, many of them monthly.

But this was no ordinary "CF user group." It was CFUN-2k.

### If You Organize It...

When Michael Smith, president of TeraTech, Inc., began planning and then put out the call for this meeting, billed as CFUN-2k, he had big plans. He arranged for a huge auditorium at the National Institutes of Health in Bethesda, Maryland.

He invited some of the most popular CF speakers nationwide, and decided not to charge a fee for attendees. Speakers responded from across the country and agreed to speak at no charge.

Then he sent out invitations to thousands of CF developers worldwide. They answered by the hundreds, spread the word and invited others.

In the end more than a thousand people registered, and Michael, his staff and a group of dedicated cosponsors began the arduous task



FIGURE 1: Michael Smith opens CFUN-2k.

of organizing, registering and pulling off a tremendous example of CF community building. They'd done it the year before, so they knew what they were getting themselves into.

CFUN-2k was on, and it was going to be bigger than any CF gathering to date. Bigger than the previous year and, yes, even bigger than last year's CF Developer Conference.

### ...They Will Come

Did I mention this was a weekend event? And not just one but two days? During one of the loveliest

summers on record in Washington?

Yet still they came. Many of the participants and several of the speakers traveled long distances (in a few cases several hundred miles). Seven hundred CF developers attended, hungry to learn and willing to sit for several hours each day just watching and listening as speakers covered subjects from basics to business issues to advanced techniques. An overflow room with video feed had to be set up to accommodate them.

It wasn't all work, though – there were breaks for refreshments and lunch, and constant giveaways of books, software, tee shirts and more. To break the ice even further, there was the clever and well-received CF "Family Feud" as well as "Who Wants to Be a CF Millionaire?"

### The Sessions

The meeting was a single-track format with 40-minute presentations. The roster of 16 speakers included many of the most popular local and national CF personalities. In order of appearance they were:

- **Adam Churvis**, president, Productivity Enhancement, purveyor of CommerceBlocks and a series of CF e-commerce seminars, spoke on "Using CF, Stored Procedures and Triggers," and offered lots of useful coding tips.
- **Christine Pascarella**, VP, sales, VirtualScape, shared insights into the challenges and opportunities as well as security issues involving CF hosting.
- I had the next slot, and showed how to build WML applications using ColdFusion.
- **John Paul Ashenfelter**, president, TransitionPoint, and coauthor of *CF for Dummies*, helped simplify



FIGURE 2: Crowds of CF programmers overflow NIH.

"complex" data types in ColdFusion (arrays, structures and WDDX).

- **Michael Smith** spoke on the dynamics of change in the Web development industry and on the value of peer-to-peer networking (among computers and ourselves!).
- **Howie Hamlin**, project manager, CoolFusion.com, purveyors of the InFusion Mail Server, spoke on their tool's many features and benefits.
- **Robi Sen**, CIO of Granularity.com and organizer of the first national CF conference in Fort Collins, Colorado, in 1998, spoke on business-to-business commerce with a focus on new forms of electronic data interchange and application syndication.
- **Steve Nelson** of SecretAgents.com, a noted Fusebox.org ambassador (and soon to be author of a Fusebox book), spoke on the Fusebox methodology and its significance as an alternative for creating highly reusable CF applications. Steve did double duty by delivering **Hal Helms's** talk on "Beginning Fusebox." Hal, who was unable to attend, is a well-known **CFDJ** writer, Fusebox guru and consultant.
- **Bill Rogers**, CEO of Ektron.com, purveyors of the eMPower content management solution, demonstrated their tool and how it enables content providers on your Web site team to offer content in an easy and inexpensive manner.

And that was just the first day! After a fun party Saturday evening, the following morning started right in with the next round of speakers:

- **Leon Chalnack**, president, Advanta Solutions, was unable to attend, so the next morning, at his request, I gave his presentation on the features and benefits of using ColdFusion custom tags.
- **April Fleming**, Web software developer for Federal Data Corp. and organizer of the Orlando CFUG, continued the WDDX theme started by John Paul Ashenfelter and showed a working example of using WDDX to syndicate data among CF servers.
- **Shlomy Gantz** of CoreActive ACG, the most highly rated speaker at last year's CFUN event, demonstrated the power and capability of integrating Macromedia Flash



FIGURE 3: "CF Doctor" answers CF questions.

with ColdFusion to create powerfully interactive applications.

- **Steve Drucker**, CEO of Fig Leaf Software and one of the most popular CFUG speakers nationwide, carried forward Shlomy's Flash ideas and added COM and Java for further open integration.
- **Michael Dinowitz**, master of the CF-Talk list, spoke on programming philosophy, both general and CF-specific, and on the value of the interconnected CF developer community.
- **Dave Watts**, CTO of Fig Leaf Software and the most prolific contributor to the CF-Talk mailing list, shared his wisdom on "extreme debugging" and identified several tools to help solve system problems that can affect a CF application.
- **Michael Imhoff**, CTO of OmniCypher, spoke on the pluses and minuses of using Microsoft Access as a database for CF applications in development and production.
- **Dave Aden**, VP, technology services, WWStudios.com and contributor to the *Allaire Spectra E-Business Construction Kit*, gave an introduction to Spectra as well as tips and tricks for those already familiar with it.

Almost all the presentations were made available on the CD and still more are available online at the conference Web site at [www.cfconf.org/cfspeakers.cfm](http://www.cfconf.org/cfspeakers.cfm).

### All Work and No Play? No Way!

As I mentioned before, the weekend was much more than just sessions and speakers. Besides the giveaways

and prizes (many organized by Amy Brooks, who as user group coordinator for Allaire deserves a big thanks from all of us all over the country), there were some elaborate and clever contests.

In the "CF Family Feud" Michael Smith and company had polled registrants in the weeks before to answer several questions (technical and non-work related) and had gathered the most popular answers for each. As in the TV game show, game organizer and emcee Chris Mosier quizzed two "families" composed of audience members and speakers. And in "Who Wants to Be a CF Millionaire," host and organizer Adam Churvis led two contestants through their paces on several challenging CF questions.

### You Asked? They Answered

Questions and answers were the rule throughout the conference. There was time after each talk for them. There was also a vendor area outside the hall with several CF companies and conference cosponsors demonstrating their products and, of course, ready and willing to answer any product questions.

The "CF Doctor" (Douglas Smith of TeraTech) was "in," trying to answer random questions throughout the conference. When he was stumped, he brought questions to a panel on Sunday composed of most of the speakers who also addressed questions from the audience.

### A Yeoman Effort

The folks at TeraTech as well as volunteers from Capital PC User Group (CPCUG) and the Maryland CFUG did a yeomanlike job, staffing the entrance area and performing all manner of support tasks, including planning, organizing, marketing, registration and coordination of the CD, not only during the weekend, but in the many days leading up to the event.

The speakers and cosponsors, along with Michael Smith and his company, deserve a round of applause (and received several) for their tremendous efforts. It was a worthy precursor of the upcoming second annual worldwide Allaire Developer Conference, November 5-8, in Washington, DC.



### ABOUT THE AUTHOR

Charles Arehart is a certified Allaire trainer and CTO of SysManage, an Allaire partner. He contributes to several CF resources and is a frequent speaker at user groups throughout the country.

CAREHART@SYSTEMANAGE.COM



**W**e've all seen it: the dreaded "Error Occurred While Processing Request," which is the headline above a normal CF error message. As developers, of course, we relish the detail it offers: the CF error message, the line number and actual line of code in error, the name and path of the template in which it occurred, and so on. But is this the best thing to show our end users?

Probably not. What should they do with that information? Do they care about these details? Could a hacker use them for unintended purposes? And perhaps most important, how do you even know the error has occurred?

Several features – some old, some new in 4.5, but in either case often missed and misunderstood – can dramatically improve error handling in ColdFusion. In this series of articles I'll cover the basic solutions, CFTRY/CFCATCH, CFERROR (three forms of it) and Site Wide Error Handling, which may be new to you.

In this issue, though, we'll focus on some administrator settings that you may not have considered and that may be quite important to any effective error-handling strategy, depending on the other error-handling choices you make. I'll show how you can reduce the level of detail offered in error messages, as well as improve the ease with which errors may be reported to you when they occur. Along the way I'm confident even the most experienced CF coder will learn a thing or two as some aspects of these features are poorly documented and not obvious.

I'll discuss those other error-handling choices (CFTRY/CFCATCH, CFERROR, and Site Wide Error Handling) in forthcoming issues of **CFDJ**, and at the end I'll offer a clear list of things you should be doing to better handle errors in your applications.

By applying even one of these solutions, with just a few minutes work (or seconds, in some cases) you could dramatically improve not only your end users' experience, but also the security of your site and your awareness of errors that occur in your applications.

### That Lovable, If Pesky, Error Message

The standard CF error message is a delight to the CF developer, with all that great detail. And if it's a SQL error, we even see the database error message, the name of the datasource in error and the SQL statement that was attempted. This is just great for debugging. An example is shown in Figure 1.

Clearly, the creators of CF were developers at heart and realized we'd need to see that info to solve problems.

**Quick Tip:** When you get an error, do you jump back into the code and try to find and fix it? Most developers seem to...and fail to take full advantage of the text of the message. In some cases, though (see Figure 1), what's wrong isn't always obvious.

Even then, if you hop back into the code to scroll around looking for the line of code in error, you're missing a useful shortcut. Notice the line number that's offered (line 79 in this example). In Studio, type CTRL-G (or Edit>GoTo Line) to jump to that line of code. It's not a perfect solution, however; if you have any CFINCLUDEs above the line in error, the numbers in the Studio won't be in sync with the one reported in the error.

# TOWARD BETTER ERROR HANDLING

BY CHARLES AREHART

This detail is great for developers, but when you move your code to a production environment or, more simply, when an end user visits the site, is that error message an appropriate thing to offer them?

### Is It the Right Thing for End Users?

There are several problems with letting users see an error message. First, of course, there's the compromise such errors make to your otherwise elegant user interface. How does it look when they go from your site's consistently blue background to the plain white CF error message page? And what are they going to do with this information? And again, how do you know the error has occurred for your users? Do you simply wait for a call and report it? And have you considered the security risks of showing the physical path of your templates and the SQL datasource name and statements of failed queries in this error message to users?

If you're like many CF developers, these are things you probably haven't thought about, or simply didn't know how to solve. As of Release 4.5, there are a host of options you can consider to limit, modify or hide the error message users see. You can even





arrange to log the errors in a database and/or send your developers an e-mail indicating that an error has occurred.

But these enhancements generally require programming changes. We want to start off with some quick changes that can be made to assist the user in informing you that an error has occurred, as well as limiting what information is displayed.

### **Administrator Configuration Changes**

In this article we'll consider three changes that can be made in the CF Administrator. They'll have an important impact on the information presented (or hidden) in the traditional CF error message.

Subsequent issues will cover modifying or completely hiding the display of the message as well as logging/e-mailing it to you automatically, with new "error-handling templates."

But even with those in place if you have them (or until we cover them next month), the changes discussed below have benefits. They'll improve both the ease of users reporting an error as well as the security of your site – especially if you won't be implementing error-handling templates.

The issues we'll cover are setting the administrator e-mail, hiding the SQL and datasource name for query errors, and hiding the template path to the template in error. Even if you think you understand these options, there's more to them than is documented. Let's look at them in detail.

### **Defining the Administrator E-Mail Error-Handling Address**

Some experienced developers may be surprised by my pointing this out, but it's one of the simplest improvements you



**FIGURE 1:** Error diagnostic information

can make, in the right circumstances, and it's easily implemented.

In the CF Administrator there's a place where you can define the administrator e-mail address for the server:

- Take the "settings" link under the Logging area on the left nav bar.
- Enter an e-mail address that should receive e-mails sent by users choosing an option that will be presented to them to send such messages.

Entering this address will from then on cause an additional line of display to be shown on the standard CF error page (the traditional unformatted one), at the bottom of the screen:

Please inform the [site administrator](#) that this error has occurred (be sure to include the contents of this page in your message to the administrator).

The "site administrator" link will now hold a hyperlink that when clicked will indicate to the browser that a mail message should be created to be sent to the e-mail address specified in that Administrator setting. (If the user's browser and e-mail client are properly configured to respond to such e-mail hyperlinks, the client's e-mail program will launch an e-mail message to the specified address – and the user can choose to fill in the subject and body of the message.)

Of course, we still have to hope they pay attention to the error message that states they should "include the contents of this page" in the message they send. And it's important to note that this doesn't mean that the "administrator e-mail" address will receive notification of every error that occurs – only when end users click the link offered in that message above. That's something we'll enable in the next issue.

### **Not Enough, But a Good First Step**

This still doesn't solve the other problems of the less-than-attractive interface of the error message or getting an automatic notification. But it's a step in the right direction and, unless you're in a shared server environment where no one person is a suitable recipient of all notifications about error messages, it's probably the first thing you ought to do. (Just be sure to change that administrator setting should that person ever change or leave, lest messages be sent to someone who'll never get them.)

It's also useful when you *do* use site-wide error handling, as it becomes a variable you can refer to in the site-wide error template, covered next month.

### **Hiding the SQL and Datasource Name, and Template Path**

I've already mentioned the potential security risk of showing the end user such details as the path to the template in error and the SQL code and datasource name in failed queries. In the next article I'll show how you can hide the entire error message from the user, so this point about hiding certain details of the message may become moot if you apply the other techniques. But it's important to understand, especially if you're not currently aware of the issue and may not soon implement those other techniques.

### **The Risk**

Let's look closely at the information shown in the normal CF error message, focusing just on the SQL and Datasource name of a failed query, and the path to the template in error (see Figure 1).

Could someone take advantage of knowing your Datasource name (SomeDSN, in the example) and the SQL code of the failed queries? Or the template path (C:\INETPUB\WWWROOT\SOMEDIR\ to the page in error?

Well, perhaps not if they're just a user of your Web site. Of course, if such a user could break in, they could certainly take advantage of the information. But such break-ins are unlikely, or at least well beyond the scope of this article.

But what if you're on a shared Web server with others who aren't related to your application but can also put CF code on the server in other directories? This is certainly an issue in a commercially hosted Web site on a server shared by many users. It's also a possible concern for users on a corporate Web site having multiple unrelated applications. (One solution to this problem is to use CF's Advanced Security, but this isn't a trivial exercise and is beyond the scope of this article.)

As for the SQL and Datasource Name, if someone can place CF code on the same server as you, have you considered the risk to your data if he or she knew the datasource name for your database? What's to stop someone from running a query against your database? If you're not password protecting your database, any CF template that's running on that same CF server can write a query against your datasource. Password protecting the file may be trivial or challenging in your environment, but explaining it is a subject for another article.

There is yet another risk, as described in Allaire Security Bulletins ASB99-04 and ASB99-09. It's possible in some situations (if not otherwise protected using the suggestions in the bulletins) for a Web visitor to append to the end of a URL (in certain rather unusual situations) extra SQL commands to be executed against your database. This isn't really a CF bug but rather a database driver issue.

In both these cases, if someone can see the SQL statements (including table and column names) shown in an error message, that makes it all the easier to take advantage of any security hole or weaknesses in your environment. So let's hide that information.

### **Hiding the SQL and Datasource Name in Query Errors**

The good news is that it's easy to stop that information from showing in the standard error message (or the one available for display as a variable in one of the error-handling template approaches, discussed in the next issue). It's just a simple tick of a checkbox in the CF Administrator.

It may not be so easy to find, however. It's presented along with server-side debugging control options:

- Take the "debugging" link under the Miscellaneous area on the left nav bar.
- Make sure "Show SQL and data source name" is unchecked.



Doing this will reduce the amount of detail in the message above by removing the lines:

```
SQL = "select * from jobs where jobs.com-  
panyid = 499 and jobs.composite_ad_yn  
<> 1 and jobs.jobtitleid = jobtitles.jobtitleid  
and jobs.cityid = val_cities.cityid order by  
stateid,city, jobtitle, date_posted desc"
```

Data Source = "SomeDSN"

That's another step in the right direction. Now you might be worried about the loss of this useful information as a developer, but there's good news.

### **Interesting Behavior That May Confuse You**

The text explaining this option in the Administrator screen suggests that it controls display, not only of the Datasource name but the SQL statement in error as well. You may find, however, that after doing this you'll still see the SQL statement in error. Is the feature broken? Will end users still see the SQL statement in error? That depends.

Although it's not documented anywhere, this option behaves in an interesting way, and it explains why this option is on the debugging options page. A user who can see the server-side debugging

information will always see the SQL statement in error, regardless of whether the "show SQL and datasource name" is turned off.

This is a nice feature for developers as it gives them the detail they need. It's likely that you don't show end users the debugging information, so this makes it really safe to turn off the display of the SQL statement. Those who shouldn't see it won't, while those who should, will.

### **Hiding the Template Path**

The security concerns don't stop there. Remember that display in the error message of the physical path to the template in error? That can also be used by hackers, or simply by others not associated with your project but located on the same server. A few tags available in CF, such as CFFILE and CFCONTENT, allow access to any file in the CF server.

The error occurred while processing an element with a general identifier of (CFQUERY), occupying document position (26:5) to (26:59).

If someone running code on your server (again, someone who has access to the server or, less likely, someone who's broken in) knows where your code resides,

that person can use those tags to grab your source code...or possibly your database.

There are two broader solutions to that problem: one is to restrict access to those tags in the Administrator, which is an option under "Basic Security." Or you can configure Advanced Security, mentioned previously, to protect directories from access by other developers on the same server.

Beyond those security approaches, you can also lessen the risk caused by error messages showing the template path by simply preventing its display in the standard CF error message. Like the "SQL and Datasource Name" option, it too is embedded, curiously, among the server-side debugging control options:

- Take the "debugging" link under the Miscellaneous area on the left nav bar.
- Make sure "Display the template path in error messages" is unchecked.

Doing this, along with the SQL and Datasource Name option, will reduce the message detail by changing the last line in the message to:

The specific sequence of files included or processed is:  
C:\inetpub\wwwroot\somedir\some-template.cfm

**GET YOUR OWN!**

**PowerBuilder**

**JBuilder**

**Developer's Journal**

**BUILDING ENTERPRISE PORTALS**

**GOLD FUSION**

**XML: IT'S THE FUTURE**

**Call and Subscribe Today!**

**1-800-513-7111**

**www.sys-con.com**



### Additional Observations

Sadly, this option doesn't respond the way the SQL and Datasource Name option does: even if you're authorized to see server-side debugging information, once this option is turned on, the template name and path will no longer be displayed in error messages. This will make it a little harder for a developer to determine a template in error.

If you have development and production environments, it's certainly worth considering whether you really want to use this option to hide the template path in the development environment.

Some errors (unless otherwise handled by error-handling strategies to be discussed in the next issue) will still show the template path anyway. Compilation errors are an example. Consider this example: turn off the display of the template path and run a template with the code:

```
<cfif></cfif>
```

This compilation error will display the complete path in the error message, after the details about the compilation error itself, as in:

This doesn't happen in all classes of errors, but it's something to keep in mind.

### Conclusion

We've covered a lot of ground, and hopefully you've learned some things about configuring the Administrator to improve error handling.

Next month I'll explain further the opportunity to have even greater control over the display of errors, or hide them entirely and simply e-mail them automatically to a support person with no interaction by the user, or perhaps even log them to a database. I'll provide two of the three alternatives: Site Wide Error Handling (new to 4.5) and application-level error handling (which has changed significantly in 4.5). In the final part I'll cover CFTRY and CFCATCH, which offer even finer levels of control over error handling within your code.

As for the recommendations to take from this article, consider the following as a summary of the Administrator changes available to you:

- Set the Administrator e-mail address (if you're on a server where all the applications are related and one person is appropriately ready to handle all such messages).
- Turn off the display of the Datasource Name and SQL for queries in error, and the template path to the template in error, in anything other than a pure testing/development environment.

As for some of the security best practices we alluded to:

- Password protect your databases (even in Access) so users running code on a shared server with you need more than just the datasource name and knowledge of your tables and column names to access your data.
- Read the two Allaire security bulletins about further protecting your data from end users who might be able to take advantage of certain security issues in some database drivers and with some templates.
- Consider the "Tag Restriction" options in the Administrator to further prevent the likelihood of abuse among unrelated developers on a shared server.
- Set the Administrator debugging options to prevent server-side debug information from showing to your users, which allows the hide "SQL and Datasource Name" option described above to indeed hide the SQL in error.

For all these options, see the Allaire documentation for more information. All the administrator configuration options presented in this article are discussed in the manual *Administering ColdFusion Server*.



CAREHART@SYSTEMANAGE.COM

# Ask the Training Staff

BY  
BRUCE  
VAN HORN



## A new, interactive column for ColdFusion users

It's been a long time since we announced this column back in May, but it's finally here! The purpose of the column is to expose you to some of the questions that Allaire's trainers are asked in the classroom and, hopefully, to give you the right answer.

Another goal is to give you, the readers of **CFDJ**, access to the same resources as our classroom students. You can e-mail your questions to us at AskCFDJ@syscon.com. We'll circulate them among the Allaire Certified Trainers to get you the best answer. Don't be shy. If you have a question about using ColdFusion, let us know!

Since this is the first column, we haven't built up a backlog of questions, so I'd like to share a few I've encountered from students over the last few months.

**Q:** Is there a way to get CF Studio to save a file automatically when I browse using the internal browser?

**A:** Yes. This is a common problem. You're working on a page that you've previously saved; you make a few changes and browse again, but you don't see the changes because you forgot to save the file again before using Studio's internal browser. You can solve this problem by going to the Options menu and choosing "Configure External Browsers...". From that dialog box select the option to "Automatically save changes to the current document," then click OK. Next, return to the Options menu and choose Settings (F8). Select the Browse option on the left. Check the box labeled "Use External Browser Configuration for Internal Browser" and click OK. No more unsaved pages when you browse!

**Q:** I've set a local variable in Application.cfm, but it doesn't work in any of my custom tags. Why?

**A:** This can be frustrating, but it starts to make sense if you think about how CF processes custom tags. While the code Application.cfm is automatically included in the page that calls the custom tag, the CF server calls the custom tag itself as a separate request without including any of the code from Application.cfm. This explains why you get an error in your custom tag – that variable doesn't exist.

You can get around this problem in several ways. The first way would be to reference that variable from within the custom tag using the "caller" scope instead of the "variables" scope. For example, if you create a local variable in Application.cfm called "MyDSN" (<CFSET MyDSN = "MyDataSource">), you'd reference that variable in a regular CF page using the "variables" scope (<CFQUERY DATASOURCE="#Variables.DSN#">). If that same page calls a custom tag, the code inside the tag has access to local variables that existed on the page that called it using the "caller" scope. For example, the code inside the custom tag page would look like this:

```
<CFQUERY DATASOURCE="#Caller.DSN#">
```

Another way to get around this problem would be to create the variable as an Application or Session variable instead of a local vari-

able. For example, you could change the code in Application.cfm to <CFSET Application.MyDSN = "MyDataSource"> (assuming you've used the <CFAPPLICATION> tag to initialize the application framework). Once the variable is an application-level variable, it'll be available to your custom tags simply by referencing it as "Application.MyDSN" in your code.

**Q:** I have a CFQUERY that uses fully qualified column names:

```
<CFQUERY NAME="qGetDist" DATA-SOURCE="DistributorData">
SELECT
Distributor.Distributor_Name, Country.Country_Name
FROM Distributor, Country
WHERE Distributor.Country_ID = Country.Country_ID
</CFQUERY>
```

When I try to output the value of Country.Country\_Name, I get an error. How do I reference those columns in my CF page?

**A:** I remember making the same kind of mistake when I first started writing CF code. The answer is easy. You don't reference the column variable using the table name as the prefix, you use the name of the query as the prefix. In other words, it should be qGetDist.Country\_Name, since qGetDist is the name you gave this query in the CFQUERY tag. Even though you prefix (or qualify) the database

ABOUT THE  
AUTHOR  
Bruce Van Horn,  
a regular contributor  
to CFDJ, is president  
of Netsite Dynamics,  
LLC, and an Allaire  
certified instructor.

column with the name of the table in your SQL statement, the database gives back to CF just the name of the column (e.g., Country\_Name). You can see this if you run the query using CF Studio's query builder tool. Only the column names come back to CF (which means you can't have duplicate column names in your query). When you reference query result columns as variables in your code, you need to prefix them with the name you specified in the NAME attribute of the CFQUERY tag.

**Q:** How can I find out what Session variables exist for a user?

**A:** The Session scope is actually a structure (a single variable that can hold many different values). The individual variables you set into the session scope are actually just "keys" within the Session structure. You can get a list of all the keys in a structure by using the StructKeyList() function (e.g., <CFSET SessionVars = StructKeyList(Session)>). This will give you a comma-separated list of all the keys for the session scope. You could also loop over the structure to get the same information:

```
<CFLOOP COLLECTION="#Session#" ITEM="Key-
Name">
<CFOUTPUT>
Session.#KeyName# = #Evaluate("Session."&
KeyName)#<BR>
</CFOUTPUT>
</CFLOOP>
```

**Q:** How much memory should I allocate in the ColdFusion Administrator for template caching?

**A:** The goal behind this setting is to allocate enough memory so all your CF pages get cached. Setting this number too small creates a "rolling cache" situation in which new requests are added to the cache and older ones are rolled out. The guideline is basically this: calculate the total size of your CF pages (your CFML pages only, not image files, HTML pages, etc.) in kilobytes and multiply that number by five. The reason for doing this is that the generated p-code (the code that ColdFusion generates for your server to actually execute) is substantially larger than the saved CFML code.

• • •

Please send your questions about ColdFusion (CFML, CF Server or CF Studio) to [AskCFDJ@sys-con.com](mailto:AskCFDJ@sys-con.com).



BRUCE@NETSITEDYNAMICS.COM



# The Role and Place of ColdFusion in WAP Architecture

BY  
SERGEY  
RUSEEV



Create new services for cellular communications operators and corporate customers

The rapid expansion of wireless access technologies, the implantation of the Internet into all spheres of human activity, and the pressure of mobile communication markets – these factors represent the development of wireless services.

Standards for mobile communications have been developing rapidly in the last few years. The sequential appearance of analog (AMPS, NMT-450) and digital standards (GSM, D-AMPS, DCS), and their gradual movement to a universal standard, resulted in the formation of mobile communications companies all over the world. A new community of wireless subscribers is being created.

technologies have progressed greatly. Now there are several typical solutions in the fields of access authorization, database publication, remote monitoring and Internet services integration to Web servers.

To save investments, developers want to get maximum usage and create new usage for existing technologies.

Market pressure is also a major influence. For example, there are more than 100 million Internet users, but the number of cellular network subscribers is more than double that number. Another example, the coefficient of penetration into the PC market (principal users of the Internet) is much lower than the one for cellular set owners – even in the U.S., a very computer-oriented country. So the Internet services and features that are required beyond the bounds of the Internet community already have a huge audience.

the HTTP server, it adapts the data that's received by HTTP queries from the WAP server to specialties of transmission environment within the cellular communication network. Common restrictions to the environment are:

- Connection instability
- Considerable service activation time
- Limited rate of data transmission to the cellular terminal
- Limited memory of the cellular terminal
- Limited calculating capacity of the cellular terminal
- Limited possibilities of information and graphical representation on the cellular terminal
- Truncated capacities of information input
- **WAP Server:** Includes the HTTP server of the WML content, the database server, and the application server as a binder element.

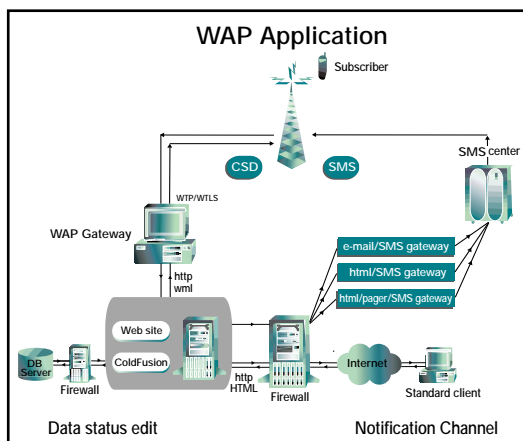


FIGURE 1: WAP application

New services, like roaming and data transfer, formed the basis for the globalization of services provided. Mobile subscribers need data availability. This is now possible due to wireless applications that have been developed based on standards that are applicable for any type of wireless carrier.

It's impossible to overestimate the penetration of the Internet into all areas of human activity; it's become the environment of information distribution and storage. The Internet is also a fast-growing business media. During the past 10 years Internet

## Components

The wireless application protocol (WAP) specification is one of the main technologies in wireless applications – the software that's based on the specification. The main technology components are the WAP browser, WAP gateway and WAP server (in its turn, it may consist of HTTP, and application and database servers). A detailed description of all WAP-specification peculiarities is beyond the limits of the article.

- **WAP Browser:** Usually it's a built-in wireless terminal, but in certain cases it may be executed as loaded software (Palm Pilot).
- **WAP Gateway:** Used as an intermediary between the WAP client and

Today, application servers are the standard solution in the development of Internet/intranet applications that provide basic functions of scalability, data protection and transparent access to databases. The products from Oracle, Sun and Allaire are the leaders in the market.

## ColdFusion as a WAP Application Server

The basic requirements for the development of wireless application software are:

- **Development simplicity:** The high dynamics of new services imply a simple method for their creation and modification. A program interface that allows developers to use



FIGURE 2: WAP demo subscribers interface

their acquired skills when developing applications is also required.

- **Wide integration:** Databases, Internet services, Web servers on the Net, and the phone services of cellular companies are indispensable for their dynamic content. The means for wide integration are contained in ColdFusion at the necessary level of tag development. A number of tags have already been created – some available free of charge. It's also possible to create tags independently. For example, to select and publish a recordset of two linked tables from the database, write some statements in CFML. To send this set as an e-mail, you need three more lines; to design the result as an XML set – one more line.
- **Interoperability:** Usually reached by using a WAP compatible browser that can interpret WML v.1.1. With ColdFusion it's possible to adapt WML contents to a specific cellular terminal type. To carry this out, a user variable that stores data on a terminal and a microbrowser type are analyzed.
- **Independence on the cellular communication standard:** Usually obtained by the WAP interface adjustment, it's an essential property of the WAP application.
- **Personalized user interface:** The CF developer may personalize the data published on request. The various possibilities of customizing WML page generation relies on client variables.
- **Strong security:** There are several levels of data protection. The basic ones are encoding and both

database and application access control.

- **Portability:** CF is available on all widespread hardware platforms. The software operates under Windows 95/98, Windows NT, Sun Solaris, HP UX, Linux and BSDI. There are also free versions of CF Express with limited functionality.
- **Flexibility and scalability:** It should be easily scalable with many program interfaces such as CORBA, Java, JavaScript, Perl, VBScript, COM and DCOM. Existing methods of document generation allow you to use packages as an HTML contents source, as well as a WML contents source.
- **Price:** CF version 4.5 costs \$1,500, and doesn't depend on a concurrent number of users.

### Wireless Application Sample Pilot by Peter Service, Ltd.

The scheme of a wireless application that demonstrates the main features of the ColdFusion application server is shown in Figure 1. This approach is realized on <http://wap.billing.ru> and <http://wap.nwgsn.com>. We used WAP-LITE gateway from Infinity Technologies ([www.waplite.com](http://www.waplite.com)) in the production mode for all our wireless solutions. Full-range WTLS-enabled applications for the corporate market are supported. In addition, WAPLITE gives the best solution for middle-sized wireless carriers worldwide.

A sample staff list from a wireless carrier involved in WAP is described below:

- The manager marks data on the current number of cellular company subscribers using a WAP interface. This data modification should always be controlled.
- The customer service employee enters the data into a specified DB table. After data has been marked, the session is terminated and the subscriber is transferred to waiting mode.
- The CF application dispatcher starts up the tracking application (frequency is set by the user) that checks the parameter that indicates changes in the DB table. If the variance value exceeds the quantum specified, then a request-on-notification will be sent via three gateways – e-mail, HTML/SMS and HTML/pager.

After notification of receipt, the cellular terminal owner may reactivate the session and execute operations specified in the fixed list stored in the database: publish the parameter that was tracked on a public server, call up customer service, technical service, and more. Here the caller doesn't need to know the called numbers.

The application server may also appear as a mediation device that realizes the initial processing of Call Data Record (CDR) from the WAP gateway.


At last, all features that are related to displaying database contents on a cellular terminal may be published via the same Web server using practically the same application server modules.

Figure 2 shows a demo variant of a WAP interface for a cellular network subscriber that allows (after authorization) modification of the service status. Next to this one, a variant of a similar interface on a cellular communications operator's Web server is shown for comparison.

The CF application server realizes several integration levels:

- Interaction with a WAP gateway on receipt of the CDR acting as a mediation device
- Interaction with active regional wireless networks
- Interaction with Internet services
- Application for protection from unauthorized access
- Establishing a session with a WAP terminal and access authorization
- Establishing a session with a visitor who orders services in standard HTML using a standard Web browser.

### Conclusion

Allaire's CF application server allows you to create new services for cellular communications operators and corporate customers, to scale these applications as well as adapt them to new conditions. It enables the wide integration that's required by wireless applications for cellular operators, public Internet services and service providers, and corporations. 

### ABOUT THE AUTHOR

Sergey Ruseev is the Web project manager for Peter Service Ltd., a telecommunications software development company in St. Petersburg, Russia. He has a BS in computer science from the Leningrad Electrical Engineering Institute.

# CommonSpot Content Server v.2.0

## by PaperThin, Inc.



REVIEWED BY  
DAVE HORAN

With personalization and customization becoming a mainstay of Web sites, the burden of adding and managing this functionality falls on Web developers. In addition, many organizations want to put the responsibility of content management and updates into the hands of nontechnical staff. Sounds great, until you need to provide code for this site, then the next and the next...

There are several solutions to assist with this problem, ranging from BroadVision and Spectra to small home-grown apps. CommonSpot Content Server from PaperThin, Inc., enters as a middleweight contender to solve the content management issues.

At Fusion Productions we were looking for a robust, ColdFusion-based content management system that provided flexible security, authentication, content scheduling and a personalization framework. We also wanted a cost-effective system that could be extended and easily integrated with the client's current legacy and Web-based application. CommonSpot fit the bill.

This ColdFusion-based product provides an application framework that offers user authentication, template-based page creation, Web-based administration and more. These features enable developers to concentrate on integration issues, special functionality and template

design. As the content is stored in the database, disk usage varies with the site, which I'll go into later. A dual-processor machine makes a significant impact on the content creation, indexing and serving process.

### System Requirements

As with most products, there are minimum and "suggested" base hardware requirements. For Windows NT, the recommended RAM as per PaperThin is 256Mb. This seems to work well, but if you have more services and sites running off the machine, the more memory the better. Four GB is the stated minimum disk space. This is definitely a minimum; you'll need more if you want some breathing room after loading the OS (NT), Web services, ColdFusion and more.

If you haven't caught on, CommonSpot is a ColdFusion application. As such, it can run in the same environments as ColdFusion; however, the Web-based administration

### VITALS

**CommonSpot Content Server v.2.0**  
PaperThin, Inc.

**Address:** 267 King Caesar Rd  
Duxbury, MA 02332

**Phone:** 800 940-3087

**Web:** [www.paperthin.com](http://www.paperthin.com)

**E-mail:** [info@paperthin.com](mailto:info@paperthin.com)

#### Test Environment:

Single and dual PIII 500MHz machines with 256–512Mb of RAM, running Microsoft Windows NT 4.0, SP5, IIS 4 and ColdFusion 4.0.1.

and content management is IE 4.0+ specific. Pages created with the system are viewable from almost all 3.x+ browsers.

### Product Features

CommonSpot boasts a rich feature set including:

- Hierarchical template-driven Web pages
- Thirty or more elements (controls) for presentation of content
- Rich text editor
- Specialized elements for the inclusion of ColdFusion, Perl, CGI and other applications
- Access based on roles and privileges
- Approval workflow
- Content scheduling and personalization
- Tailored and targeted views
- Usage tracking
- Metadata collection
- Full text indexing using Verity Search engine
- Keywords and dynamic views
- Content change subscription
- Centralized database repository
- Distributed administration
- Multilevel security privileges

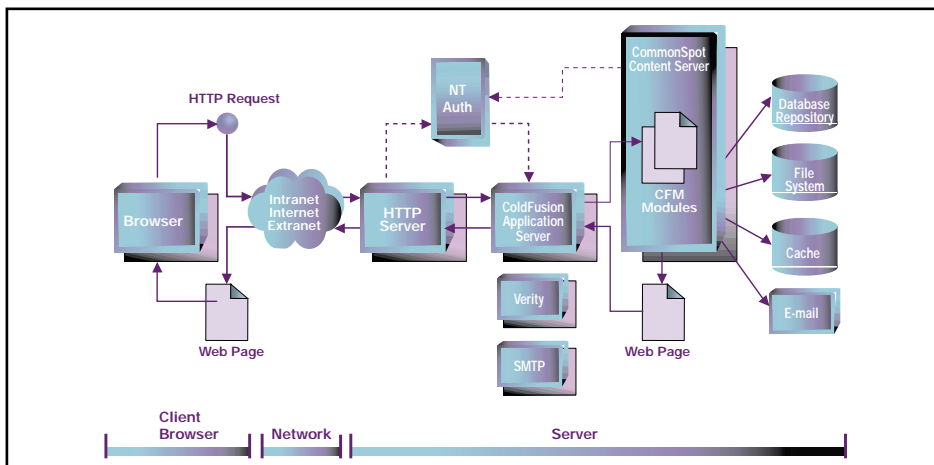


FIGURE 1: Basic functional architecture



- Dynamic page generation and static page caching
- Version history

Details on these functions can be found on the PaperThin Web site. I'll focus on a couple of the features pertinent to the developer: templates, security and personalization/customization. But first, let's look at the system architecture.

### Product Architecture

Figure 1 shows the basic configuration of the system, which should be familiar since it follows the same process of most ColdFusion applications. CommonSpot supports SQL Server, Oracle and Microsoft Access for the system databases. As you may have discovered with applications you've written, Access isn't always the appropriate choice for production systems. An Access database housed on the Web server may work well for prototyping and testing, but to ensure the scalability and performance of the system, PaperThin recommends using SQL Server or Oracle. For optimal performance you may want to run SQL Server or Oracle on a separate server.

The system contains three primary databases: sites, users and a site-specific database. Since CommonSpot accommodates multiple sites per server, the sites database contains tables that coordinate the details of each site hosted on that system instance. The users database contains data regarding the system's users and groups. Each site created within CommonSpot on the server has its own database to store content.

The content for each site is broken down into two parts: "live" content and "work-in-progress." When appropriate, the system tracks the content changes made by approved content providers and funnels these changes through approval channels. These control tables are hooked to the users database through the security system.

### Security

CommonSpot Content Server v.2.0 contains a well-thought-out security system that provides granular control over administration, con-

tent access and publishing approval. The system uses the basic idea of users and groups (e.g., several operating systems and a content management system), and allows the nesting of groups. There are no predefined roles per se. Rather, you build up access to site pages and elements using the users and groups you've defined. These are also used as the basis for site personalization.

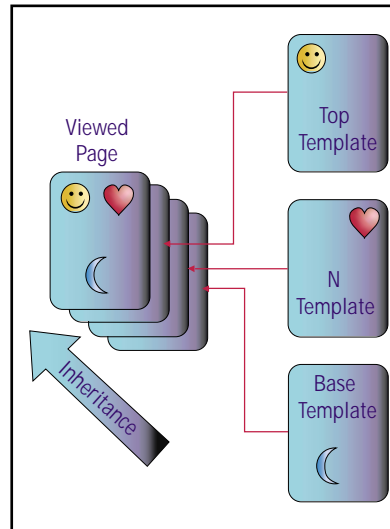


FIGURE 2: Page template inheritance

Security rights can be inherited from site-wide settings and parent templates (we'll discuss this in detail later). Security permissions can also be enforced and restricted. Enforced permissions ensure that content owners can't override permission previously assigned. Thus you may want to always enforce that Sally Sue is the approver of all pages created from the press release template. Restrictions allow certain permissions to be managed only in the template, and prevent authors from changing the font or layout properties outside the template (can't change at the page level).

The flexibility that the security scheme provides is great for setting up complex sites. However, the flexibility comes at the cost of more planning. Charging into a site without planning the users, groups and site security beforehand can cause major headaches. Of course, this is true for any Web design.

### Sites, Templates and Page Elements

CommonSpot builds on some common ideas in dynamic Web

design. All pages and templates derive from a template. The base template contains the "look and feel" and may contain navigational components. Content templates, built on top of a base template, contain one or more elements depending on the structure and content requirements of the page. Elements manage both layout properties (including font and color properties) as well as content. Layout and content, however, are managed separately. This allows you to restrict access to layout properties and only allow the submission of content, thus ensuring that design integrity is maintained. You can determine the hierarchy of a page and from which templates it's based (see Figure 2).

Like layered transparencies in an anatomy textbook, the pages are built up from the base template to the top-level page. One template can be the "parent" of many subordinate pages. For large-scale sites, this provides tremendous timesavings. By changing the format or content in one base template, you automatically update the content of all pages based on that template.

One question that occurs to any Web developer is performance. Does CommonSpot need to process all the templates for each page call? No. The system uses a proprietary caching mechanism to reduce load processing on the server and improve the delivery time of pages.

A page's structure is defined by the elements included on the page. Elements are essentially CF tags that render and manage content at an object level. They include bulleted lists, text blocks, images, Flash and page index.

### Publishing Process and Content Providers

Authorized end users can create pages via a browser interface (currently authoring is available only through Microsoft IE v.4.0+ browsers). Indicators that are displayed through the browser provide access to functionality specific to the element. Figure 3 shows an example of a page in author mode. Icons are displayed for the various elements for which the user has access. If you click on an icon, a DHTML menu for ele-

ment properties and controls pops up. Each menu is customized for the element type, although some menu items are common to all, for example, the user's role and the security of the page or element.

Authorized users can add new elements to a page or modify the content in an existing page element. Once changes are complete, users can submit them for publication. The system provides for a full rollback of changes in case there's a problem and maintains a history of them. The administration tool tracks the number of changes. When users submit changes, they're presented with a dialog to comment on the changes; these notes appear in the change history log.

One option for the publishing workflow process is content-change approval. Approvers can be assigned at the page or element level. When authenticated users log into the system, they receive a *pending actions* list that details the content that requires approval, or they can be notified via e-mail (SMTP mail gateway). In the current 2.0 version, CommonSpot supports multiple approvers for a given page or element. There are plans to support "routing" or multilevel approvals in version 2.1. That is, once Joe reviews and approves the content change, it's then routed to Sue for final approval. At any time those involved with content approval can see where the change stands and who has approved the content. Approvals can be through a single approver or a group.

### Personalization and Content Scheduling

Elements on a page need not be static. CommonSpot allows you to customize content based on the interests of a particular user, the audience and/or a scheduled time and date.

At the simplest level, CommonSpot displays content based on the assignment of "read" rights. By denying read access to an element, content can easily be restricted or hidden from unauthorized users.

Content may be personalized based on the audience or a user's affiliation with a group, or scheduled to render based on a future



FIGURE 3: Page editing in author mode

date, a period of time, a page category, site or subsite. The schedule/personalize element acts as a container in which any element can be placed and scheduled. Each item within the schedule element is managed independently and has its own set of rules that govern its display.

CommonSpot also allows for the creation of different versions or customized views of the same page (URL), each tailored to the distinct interests of different target audiences. When an authenticated user navigates to the URL of a targeted audience page, CommonSpot dynamically serves the right version of the page for the specific audience without redirection.

Keep in mind that everything I've described is browser-based. Sitting on your deck with your laptop, you can manage a site as easily as from your office PC!

### Extending the System

CommonSpot lets you extend functionality and integrate such features as JavaScript menus, DHTML and image maps in the base template. While all other templates store the properties and content within the database repository, the base templates define their characteristics in code (the base template being a ColdFusion module). Creating a custom base template isn't difficult

if you look at the example .cfm base templates provided. To "CommonSpot-ize" a standard .cfm page, insert the custom tag calls where you want the CommonSpot content tables to be.

As with many parts of the system, you can define your own category for templates and elements. When end users add new elements to the page, they simply select the item from the Element Gallery – a categorized list of predefined and available elements. A suggested first step is to create a new category for your own site-specific elements.

Creating your own elements is a fairly simple endeavor for the ColdFusion developer. A predefined CommonSpot element called *Custom ColdFusion Script* allows you to integrate custom ColdFusion code into any page. It's the glue that holds the CommonSpot-managed HTML content and custom ColdFusion applications together.

Listing 1 shows the code for a simple custom ColdFusion element that displays the current date and time. You'll notice that there are no references to <html> or <body> tags and such. The <cfoutput> tags around the entire element code are required as CommonSpot sets the <CFSETTING ENABLECFOUTPUTONLY="YES"> flag to minimize white space. Thus you'll need this to view any output from your element.

## Pros

### **Flexibility**

The system has a flexible template creation system that can be extended to meet your needs. You can create templates and prevent end users from changing the design. You can build up templates in the system or create your own. The open database architecture also lets you integrate entire custom applications, but you should use their authentication framework to validate users.

### **Distributed Content Management**

The changes to a Web site no longer need to be funneled through a small group (perhaps one – you!) of administrators. You can configure the site security to allow the marketing people to update their own pages, the documentation team to manage their downloads section, and tech support to update their FAQs.

### **Security and Authentication Framework**

The security and authentication framework is a tremendous aid in building medium- to large-scale sites. Integrated applications supported by the ColdFusion Script element are like any other element that can be scheduled and personalized, and carry its own element security, thus freeing you from writing yet another security scheme. The user groups are also useful for internal message lists and legacy system integration. The open architecture allows you to tie custom attribute tables to a user record. This is useful for syncing data with other user/member data-sources (e.g., billing systems, membership applications).

### **Scheduling and Personalization Framework**

I've written some small-scale personalization systems, and I didn't relish the idea of creating one for medium- and large-scale sites. The scheduling and personalization framework is a tremendous time-saver and a client-pleaser. Even better, you can train a set of key content managers in various departments to schedule and create their own content. The personalization features also let you easily hide entire application modules from the site for particular users or groups. This makes

the out-of-site, out-of-mind security safeguard easy to implement.

### **DB Driven and Scalable**

Since the system can run on any database system ColdFusion can handle, utilizing existing database resources is a snap. You can start out small by prototyping with Microsoft Access, then use a migration tool provided by PaperThin to move up to a higher performance RDBMS. The HTML content and metadata are stored in the database. This makes it easy to encapsulate repeated content into a CommonSpot element and reuse it in other parts of the site. Simple changes to the one element will automatically be replicated for every instance in the site.

*Note:* Although the bulk of the content is stored in the database, plan for additional space on the Web server itself for items stored in the local file system. These files include any uploaded documents such as PDF files, sound clips, videos, presentations and all site images. Pointers to these files are stored in the database but managed locally in the Web server.

## Cons

### **Authoring Refresh Speed**

During the authoring process each change requires a screen refresh. When in authoring mode not only is the content itself displayed, so are all the appropriate editing, element properties and work-in-progress icons, DHTML menus and dialog wizards. If you have any custom ColdFusion elements that perform processing (I had a few news feed elements), they also get processed for each refresh. All this takes time since the page in author mode is dynamically generated until it's published, and at that time, the page results are cached. This constant refreshing can get tedious, but only when putting the page together the first time. In maintenance mode, the page reload is bearable. PaperThin told me they're aware of the issue and are working on reducing the number of page reloads required during this process.

### **Planning and Setup**

This isn't so much a negative fac-

tor in the product, but rather a warning to new users of the system. All the flexibility the product provides is great, but controlling it can be a challenge if you don't properly plan your site, users and security in advance. There's a tremendous amount of configuration options and site design models; wading through these takes time and thought, but the benefits are well worth the effort.

The documentation is divided into two guides: Users and Administration. Although both are good for their intended audience, there's little information that addresses issues regarding overall site configuration, architecture and design as it relates to CommonSpot. PaperThin provides professional services to assist with this process.

### **End User Training**

As with the planning process, this is more a reflection of the product's complexity than of any flaw with the tool. One of the basic tenets of the product is putting content-management tasks in the hands of the content owners and freeing the Web developers. However, unless the end user is trained properly in the use of CommonSpot, the developer will spend a great deal of time answering questions and hand holding. The Users Guide does an excellent job of providing functional instructions on element use, how to manage images and more, but lacks a step-by-step workbook-style guide for end users to run through as a self-paced course. (PaperThin has plans to offer an interactive CD-ROM for end users that will guide them through the use of most of CommonSpot's functionality.) If you're planning a CommonSpot implementation, I'd suggest a train-the-trainer scenario for your shop. Having several key subject matter experts train a content contributor in each department will promote knowledge sharing and reduce the "just a quick question" syndrome.

### **Forms Support**

This is one area that's missing from the product. As of this writing there's no predefined forms element. It's easy to create your own forms element: cre-




ate a custom ColdFusion or HTML element containing the form code and insert it into the page, then create the form- processing page independently and have it call another page (which can be a CommonSpot page) when done. The lack of forms support is disappointing, but PaperThin is working on it. In the field I've found that most forms we create interface with outside databases or systems, so I'd have to custom create them anyway.

## Summary

CommonSpot Content Server v.2.0 met our needs as a site creation tool and custom application framework. Out of the box, it's fantastic for content-heavy sites that want to migrate away from the manual HTML editing scheme toward a more automated browser-based tool. The template model enables us to be more efficient with last-minute client requests, espe-

cially those involving changes to the basic layout or design of the site. We simply modify the base template and all pages based on that template are automatically updated.

It also serves well as a custom application framework. We can create custom forms and system interfaces, and hook into the frameworks authentication and user database to customize and automate the applications easily. This has dramatically reduced the time required to create secure personalized apps. Although it may not meet everyone's needs, it'll certainly suit any shop looking for a base on which to build apps that require end users to take a major role in their own content management. 

## ABOUT THE AUTHOR

*Dave Horan is the senior Web developer and Web technical lead for Fusion Productions, LLC. Fusion specializes in Web services for professional associations and meeting productions.*

## Web Links

1. *Fusion Productions:* [www.fusionproductions.com](http://www.fusionproductions.com)
2. *PaperThin:* [www.paperthin.com](http://www.paperthin.com)

### Listing 1: Sample custom CommonSpot element code

```
<cfset day_name = "Sunday,Monday,Tuesday,Wednesday,
Thursday,Friday,Saturday">
<cfoutput>
    <font face="Verdana,Arial" size="1">
        Today is <b>#ListGetAt(day_name,DayOfWeek(Now()))#,
#DateFormat(Now(), "mmm ddyyyy")#</b><br>
        Local time is <b>#TimeFormat(Now(),"h:mm tt")# EST</b>
    </font>
</cfoutput>
```

CODE  
LISTING



The code listing for  
this article can also be located at  
[www.ColdFusionJournal.com](http://www.ColdFusionJournal.com)

DHORAN@FUSIONPRODUCTIONS.COM

## ADVERTISERS INDEX

ADVERTISER	URL	PH	PG
ABLECOMMERCE	WWW.ABLECOMMERCE.COM	360.253.4142	2
ABLECOMMERCE	WWW.AUCTIONBUILDER.COM	360.253.4142	4
AFFINITY	WWW.AFFINITY.COM		49
ALLAIRE	WWW.VUE.COM/ALLAIRE	877.460.8679	39
ALLAIRE	WWW.ALLAIRE.COM	888.939.2545	61
ALLAIRE DEVELOPER CONFERENCE	WWW.ALLAIRE.COM/CONFERENCE		27
ANDREWS TECHNOLOGY	WWW.ANDREWSTECHNOLOGY.COM	888.393.0159	17
BIZNIZ WEB	WWW.BIZNIZWEB.COM	623.915.1661	64
CAREER OPPORTUNITIES		800.582.3089	57
CATOUZER	WWW.CATOUZER.COM		67
CFXHOSTING	WWW.CFXHOSTING.COM	800.939.8188	25, 31
CF DEVELOPER'S JOURNAL	WWW.COLDFUSIONJOURNAL.COM		31
CONCEPTWARE AG	WWW.CONCEPTWARE.COM		11
CORDA TECHNOLOGIES	WWW.POPCHART.COM	888.763.0517	3
CTIA	WWW.CTIA.COM		59
CYSCAPE	WWW.CYSCAPE.COM/BHFREE	800.932.6869	66
CYBERSMARTS	WWW.CYBERSMARTS.COM		66
DEVELOPERSNETWORK	WWW.DEVELOPERSNETWORK.COM	416.203.3690	23
DIGITALNATION	WWW.DEDICATEDSERVER.NET	877.624.7897	15
FALL INTERNET WORLD	WWW.PENTONEVENTS.COM		45
INFEA	WWW.INFEA.COM		63
INTELIANT	WWW.INTELIANT.COM	800.815.5541	43
INTELLIGENT ENVIRONMENTS	WWW.SCREENSURFER.COM		33
INTERLAND	WWW.INTERLAND.COM	800.419.1714	9
INTERMEDIA	WWW.INTERMEDIA.NET	800.379.7729	68
JAVA DEVELOPER'S JOURNAL	WWW.JAVADEVELOPERSJOURNAL.COM	201.802.3021	53
JDJ STORE	WWW.JDJSTORE.COM	888.303.JAVA	61
MISTRAL DESIGN GROUP	WWW.PURPLEHOSTING.COM		37
NETDIVE	WWW.NETDIVE.COM		41
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	21
SITEHOSTING.NET	WWW.SITEHOSTING.NET	877-NTHOSTING	57
SYS-CON MEDIA, INC.	WWW.SYS-CON.COM	800.513.7111	46
THE SHORTLIST.COM	WWW.SHORTLIST.COM		32, 47
VIRTUALScape	WWW.VIRTUALScape.COM	212.460.8406	19
WIRELESS DEVCON	WWW.WIRELESSDEVCON2000.COM		53
WOODBOURNE SOLUTIONS	WWW.WOODBOURNESOLUTIONS.COM	301.428.7620	63
XML DEVCON 2000-2001	WWW.XMLDEVCON2000.COM	800.513.7111	55
XML BOOTCAMP	WWW.SDEXPO.COM/BOOTCAMP	415.905.2702	13

### Able Solutions

Enter the realm of browsable store building and administration – from your browser. Build “your\_site.com” with secure Merchant Credit Card Processing. Maintain inventory, add discounts and specials to keep your customers coming back. Increase sales with cross selling and membership pricing.

**11700 NE 95th Street, Suite 100, Vancouver, WA**

**[www.ablecommerce.com](http://www.ablecommerce.com) • 360 253-4142**

### Catouzer

Catouzer develops web-based intranet and Customer Relationship Management software solutions. With Synergy 2.0, Catouzer continues its lead in providing secure web-based work environments. ColdFusion developers now have the most advanced framework to develop secure web-based projects.

**[www.catouzer.com](http://www.catouzer.com) • 604 662-7551**

### ComputerWork.com

ComputerWork.com is a premiere technical job site for computer professionals seeking employment in the IT/IS industry. ComputerWork.com will match your technical skills and career ambitions to our many employers looking to fill their jobs with specialists in computer related fields. You can submit your resume to a specific position on our job board or you can choose to submit your resume to our resume bank, which is accessed by nearly 400 companies nationwide. ComputerWork.com is the FASTEST way to your ideal career!

**6620 Southpoint Drive South, Suite 600 Jacksonville, FL 32216**

**[www.computerwork.com](http://www.computerwork.com) • 904-296-1993**

### Corda Technologies

Corda Technologies offers tools to build your own charts and graphs for internal reports, reports on your intranet and Internet sites and for many other applications where fast, high-quality graphs and charts are desirable. Corda also offers an Application Service Provider through PopChart.com which works with high-volume sports web sites to display sports statistics with exciting, interactive charts and graphs. PopChart!... an EXPLOSION of Possibilities!

**1425 S. 550 East Orem, UT 84097**

**[www.corda.com](http://www.corda.com) • 801-802-0800**

### DevelopersNetwork.com

Developers Network is the essential online business-to-business resource for new media technology and Internet business solutions. Our Resource, Business and Product channels combine elements of helpware and community in a business setting, successfully reaching those buyers developing and managing Internet strategies.

**3007 Kingston Road Toronto, Ontario CANADA M1M 1P1**

**[www.developersnetwork.com](http://www.developersnetwork.com) • 416-203-3610**

### digitalNATION - a VERIO company

digitalNATION, VERIO's Advanced Hosting Division, is the world's leading provider of dedicated server hosting, with over 1,650 servers online today. dN's superior connected network and service abilities have earned dN a solid reputation as a first-choice provider of dedicated server solutions (Sun, Windows NT, Linux and Cobalt). digitalNATION has been providing online and network services for over six years. One of the first ISPs to provide dedicated servers running Microsoft Windows NT, the dN staff has unparalleled experience in this industry.

**5515 Cherokee Ave, Alexandria, VA 22312-2309**

**[www.dedicatedserver.com](http://www.dedicatedserver.com) • 703 642-2800**

### Ektron

Ektron supports the next-generation needs of e-businesses by providing dynamic Web infrastructure solution tools designed for use by nontechnical staff. Ektron's flagship offering, eContentManager, gives staff members across an organization the hands-on ability to make real-time additions and updates to Web content without requiring knowledge of a programming language --

while still allowing for centralized administrative control and security. With competitive advantages such as ease-of-integration and drag & drop everything, Ektron is looking to provide these empowering products to customers, resellers and integrators.

*5 Northern Blvd., Suite 6, Amherst, NH 03031*

*[www.ektron.com](http://www.ektron.com) • 603-594-0249*

## Eprise Corporation

At Eprise Corporation, we're dedicated to providing software, professional services and partnerships that make it easy to leverage the Web for more profitable and effective business operations. Our flagship product, Eprise Participant Server, incorporates leading technology to transform the dated, one-size-fits-all Web site into a strategic business asset that delivers timely and targeted communications. Simply put, Eprise and Eprise Participant Server empower business professionals to create, update, and target Web-based communications, regardless of their technical knowledge or skill. Contact us today to find out more about our products.

*1671 Worcester Road, Framingham, MA 01701*

*[www.eprise.com](http://www.eprise.com) • 508-661-5200*

## FigLeaf Software

Fig Leaf Software specializes in developing turnkey web database applications and dynamic, data-driven websites. Our goal is to develop web-based client-server applications with functionality and interface design that are nearly indistinguishable from desktop software developed using traditional tools such as Visual Basic, Visual FoxPro, Delphi, or C. Above all, we want to bring maximum value to our clients at the minimum cost. The key to fulfilling this is ensuring our staff members are experts in their particular field. Our clients expect excellence, and we demand it of ourselves.

*1400 16th St. NW, Suite 220, Washington, DC 20036*

*[www.figleaf.com](http://www.figleaf.com) • 877.344.5323*

## Inteliant

Inteliant Corporation, a leading ColdFusion consulting firm, has an outstanding reputation for providing highly skilled developers for Internet, Intranet, Extranet, Software Development, or any ColdFusion application. Our national practice has emerged to meet the evolving needs of our clients by providing resources onsite or developing remotely. Our company provides the most cost effective service in the industry and we strive to add value to your projects by minimizing expenses whenever possible. Inteliant... "Delivering Intelligent Solutions

*1150 Hancock Street, Suite 4, Quincy, MA 02169*

*[www.inteliant.com](http://www.inteliant.com) • 800-815-5541*

## Interland

Interland, Inc., ranked the No. 1 Web Hosting Provider for small- to medium-sized businesses by Windows NT Magazine, Network Computing and PC Magazine, is a global leader in Web hosting solutions ranging from a basic Web site to a sophisticated e-commerce storefront. Interland proudly features 24-hour, 7-day toll-free technical support and an advanced Administration Page. By deploying the best products, services, and support in the industry, Interland can build a Web presence that works for you. Speed. Reliability. Support. - Guaranteed.

*101 Marietta Street, Second Floor, Atlanta, GA 30303*

*[www.interland.com](http://www.interland.com) • 800-214-1460*

## Intermedia, Inc.

Our advanced virtual hosting packages (powered by Microsoft Windows NT and Internet Information Server 4.0) offer an environment supporting everything today's advanced Web developer or sophisticat-

ed client could ask for. Complete ODBC support is available on plans B and C. We support Microsoft Index Server on all hosting plans.

*953 Industrial Avenue, Suite 121, Palo Alto, CA 94303*

*[www.intermedia.net](http://www.intermedia.net) • 650 424-9935*

## Macromedia

New Macromedia UltraDev lets you create database-driven Web applications faster than ever before. It also allows you to create ASP, JavaServer Pages, and CFML applications in a single design environment. So whether you love morking directly with source code, or prefer to work visually, cut the time it takes to create employee directories, product catalogs, database search pages and more.

*600 Townsend Street, San Francisco, CA 94103*

*[www.macromedia.com](http://www.macromedia.com) • 415 252-2000*

## SaiSoft

As a recognized Allaire, Microsoft and IBM Solutions Provider, SaiSoft's Strategic focus is to become the most definitive Internet Architect, by building long lasting e-business development partnerships. With development operations in India & the UK, SaiSoft also undertakes off-shore consultancy projects where a 'four-step implementation' model is adopted to meet client needs satisfactorily.

*446 East Street, Plainville, CT 06062*

*[www.saisoftonline.com](http://www.saisoftonline.com) • 860-793-6681*

## Sitehosting.NET

Successful electronic commerce starts at SiteHosting.net; a division of Dynatek Infoworld, Inc., which provides total Web development services. We offer personal and efficient customer service with reliability at value prices. All our plans include access to SSL (Secure Socket Layer). We support ColdFusion, Active Server Pages, Real Audio/Video, Netshow Server, and more. Our hosting price starts at \$14.95/month.

*13200 Crossroads Parkway North, Suite 360,*

*City of Industry, CA 91746*

*[www.sitehosting.net](http://www.sitehosting.net) • 877 684-6784*

## Virtualscape

Why host with Virtualscape? Nobody else on the Internet understands what it takes to host ColdFusion like we do. Virtualscape is the leader in advanced Web site hosting. From Fortune 500 extranets to e-commerce sites and more, developers recognize our speed, stability, reliability and technical support.

*215 Park Avenue South, Suite 1905, New York, NY 10003*

*[www.virtualscape.com](http://www.virtualscape.com) • 212 460-8406*

To place an ad in the  
**ColdFusion Marketplace**  
contact Robyn Forma at 201 802-3022

COLD FUSION MARKETPLACE



## iE Strengthens Partnership with Allaire

(Woburn, MA) – iE, a partner of Allaire Corporation, has a new advertisement that highlights iE ScreenSurfer's pivotal role in bringing IBM and Allaire environments together. This integration of iE ScreenSurfer and Allaire ColdFusion enables Web developers to easily incorporate information from screen-based mainframe and AS/400 applications directly into ColdFusion.



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

## RSW Software Named Web Application Testing Market Mover

(Waltham, MA) – Newport Groups' 1999 annual Load Test Market Summary and Analysis names RSW Software as the market mover in the Web application testing space.

The study reveals that RSW Software outpaces industry market growth for Web testing solutions, growing 381% versus the industry average of 190%, making it the leading market mover in 1999.

## AbleCommerce Launches AuctionBuilder for Linux and Solaris

(Vancouver, WA) – AbleCommerce, a Division of AbleSolutions Corporation, announced the release of AuctionBuilder 1.0. This browser-based application enables companies and developers to quickly create, customize and administer auctions and exchanges catering to the B2B, B2C and C2C dynamic commerce markets.



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

## Allaire Furnishes Pottery Barn with Internet Technology Solutions

(Newton, MA) – Pottery Barn, Williams-Sonoma's home furnishings division, has implemented Allaire ColdFusion to extend its brick-and-mortar presence to the Web.



The Web site includes the Home Tour where visitors can view some of the product and design ideas for the season, and Design Studio, a consumer resource for inspiration and information.

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

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

## WebPerfect Solutions Announces Agreement with Excalibur Technologies

(Vienna, VA) – WebPerfect Solutions announced a cooperative sales and marketing agreement with Excalibur Technologies. The companies will now leverage key synergies in their respective products, services and technology focus with joint sales and marketing activities, including events/seminars, direct marketing, advertising,

communications and public relations projects.

WebPerfect will offer Excalibur's leading search and retrieval technology – including Excalibur RetrievalWare, FileRoom and WebExpress – with its Internet business offerings. Excalibur will offer WebPerfect's turnkey Internet solutions.

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

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



## ColdFusion Developer's Journal

### Look What's Coming!

Attention Advertisers:

Don't Miss your chance to advertise in the Allaire Developer Conference

Special Issue! Call Robyn Forma  
201-802-3022

or email [robyn@sys-con.com](mailto:robyn@sys-con.com)



[www.cybersmarts.net](http://www.cybersmarts.net)

ColdFusion • ASP • ActiveState PERL

100% Browser Based Admin • Mail Domains • FREE SSL

ODBC • Logfiles • e-Commerce Solutions

Includes 10 Domains

cyberSmarts

dot net



## Concerned About Browser Compatibility?

Introducing...

### <CF\_BrowserHawk™>

Detect your visitor's...

- Browser type and version
- Flash™ and other plug-ins
- Disabled cookies, Java™, and JavaScript
- Reverse DNS lookups
- Connection speed
- Screen resolution
- WAP & XML

Discover for yourself why thousands of companies in over 35 countries rely on cyScape® to solve their browser compatibility needs. Call now

1-800-932-6869  
or +1-703-734-1000

And much more!



Download your  
**FREE trial!**

[www.cyscape.com/bhfree](http://www.cyscape.com/bhfree)

cyScape and BrowserHawk are trademarks of cyScape, Inc. and is a trademark of Macromedia, Inc. Java is a trademark of Sun Microsystems, Inc. All other marks are trademarks of their respective companies.

"A HOT component no developer should be without."

