ColdFusion fasttrack page 8

U.S. $8.99 (Canada $9.99)

ColdFusion Developer's Journal

ColdFusionJournal.com

June 2001 Volume: 3 Issue: 6

Editorial
Living on the Edge
Robert Diamond page 5

Interview
Shawn "Hanzo" Holmes
Creator of Unweb page 32

Product Review
Carey Lilly page 44

Journeyman ColdFusion
Charles Arehart page 48

Q&A
Bruce Van Horn page 58

SYS-CON MEDIA

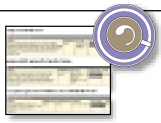
E-Commerce

with Verisign's Payflow Link

by Kelly Brown page 6

ColdFusion & Oracle: Putting Oracle SQL to Work in Your ColdFusion Applications Part 1 *Using Oracle's feature set within your Internet applications* — *Kailasnath Awati & Mario Techera* 14

ColdFusion & Java: A Cold Cup o'Joe Part 4 of 8 *Using Java objects in CFML* — *Guy Rish* 24



Custom Tags: Caching Redundant Dynamic Content with Custom Tags *A flexible static page generation method* — *Jon Block* 34

CyberCash Tag: Handling Credit Card Transactions Using CyberCash *Create a profitable online storefront* — *David Keener* 38

Custom Tags: VTML by Example Part 1 of 2 *How to successfully extend the CF Studio IDE* — *Christian Schneider* 42

<BF> on <CF>: Querying Queries *New coding opportunities without new tags or technologies* — *Ben Forta* 46

<!--Get all movies and expenses

Show Report: Macromedia User Conference 2001 *What's new and exciting in the Web-based software world* — *David Schwartz* 56

COLDFUSION Developer's Journal

department editors

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

writers in this issue

Charles Arehart, Kailasnath Awati, Jon Block, Kelly Brown, Robert Diamond, Ben Forta, David Keener, Carey Lilly, John Morgan, Guy Rish, Christian Schneider, David Schwartz, Mario Techera, Bruce Van Horn

subscriptions:
For subscription requests please call
1 800 513-7111 or go to: **www.sys-con.com**
domestic $89.99/yr. (12 issues)
canada/mexico $99/yr
overseas $129/yr
back issues $10 U.S. $15 all other

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

postmaster: send address changes to:
COLDFUSION DEVELOPER'S JOURNAL
SYS-CON MEDIA
135 Chestnut Ridge Rd.,Montvale, NJ 07645

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

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

worldwide distribution:
by **Curtis Circulation Company**
730 River Road,
New Milford, NJ 07646-3048

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

Living on the Edge

BY **ROBERT DIAMOND**

A few months ago I wrote an editorial in ***CFDJ*** (Vol. 3, issue 1) about conferences in general, and what fantastic opportunities they were for learning and networking. I spoke with many readers after that piece (at conferences of course!) and all agreed with that sentiment. Many of our readers regularly attend *i*-technology conferences to stay on top of their game, but if you don't – worry not. Now is as good a time as any to start!

I've spent the past month working with a great team on a very exciting project that I want to share with you all. On September 23–26, at the Hilton in midtown Manhattan, a colocated trade show of enormous proportions will be taking place – JDJEdge 2001 and Web Services Edge 2001. ColdFusion Edge 2001 FastTrack will also be held at the same time and location.

Twenty of the supertracks highlighting key Internet technologies at JDJEdge/Web Services Edge 2001 will be intensive learning sessions for CF developers – ranging from beginner to intermediate to advanced. These learning sessions are in quite good company with the other FastTracks covering Oracle 9*i*, BEA WebLogic, Java FastTrack to Certification, and IBM WebSphere. All of this, in addition to an enormous conference program covering the already established world of Java and the newly emerging world of Web Services.

At the conference look for a major keynote by Kevin Lynch, Macromedia's president of products. Later, on the first day, ColdFusion Fast Track kicks off with a preview of ColdFusion 6, known affectionately as Neo, provided by Mr. ColdFusion himself – Jeremy Allaire. Following these opening keynotes will be classes taught by some of the masters of the ColdFusion industry, including many names and faces you'll recognize from the pages of this magazine.

A ColdFusion conference wouldn't be a *ColdFusion conference* without Ben Forta, and he'll be presenting one of our key sessions. Also ***CFDJ*** regulars Charles Arehart, Hal Helms, Steve Drucker, Michael Dinowitz, Christian Schneider, and Kevin Schmidt will be teaching on such topics as **certification**, **FuseBox**, **wireless CF**, **Web Services**, **CF and JavaScript**, **ColdFusion 5**, **using ColdFusion and XML together**, and more....

Daily updates on conference activities, schedules, and registration can be found at www.sys-con.com/coldfusionedge. Check it out; I hope to see you there!

Now onto this issue... We've got the fantastic final piece from Charles Arehart on error handling, Guy Rish on CF and Java, and the article by Kailasnath Awati and Mario Techera on ColdFusion and Oracle is a good introduction to using Oracle 9*i* with ColdFusion on your Web sites.

Due to reader demand, we have two articles this month on integrating payment systems with CF. Both cover third-party solutions, a sometimes welcome alternative to in-house processing. One is by Kelly Brown about Verisign Payflow, and the other by Dave Keenan is on using CyberCash. Both are worth a read if you aren't doing any online sales at this point – two good examples of the technology that's "out there." We've got all that plus the usual articles from Ben Forta, Bruce Van Horn, and others. Enjoy!

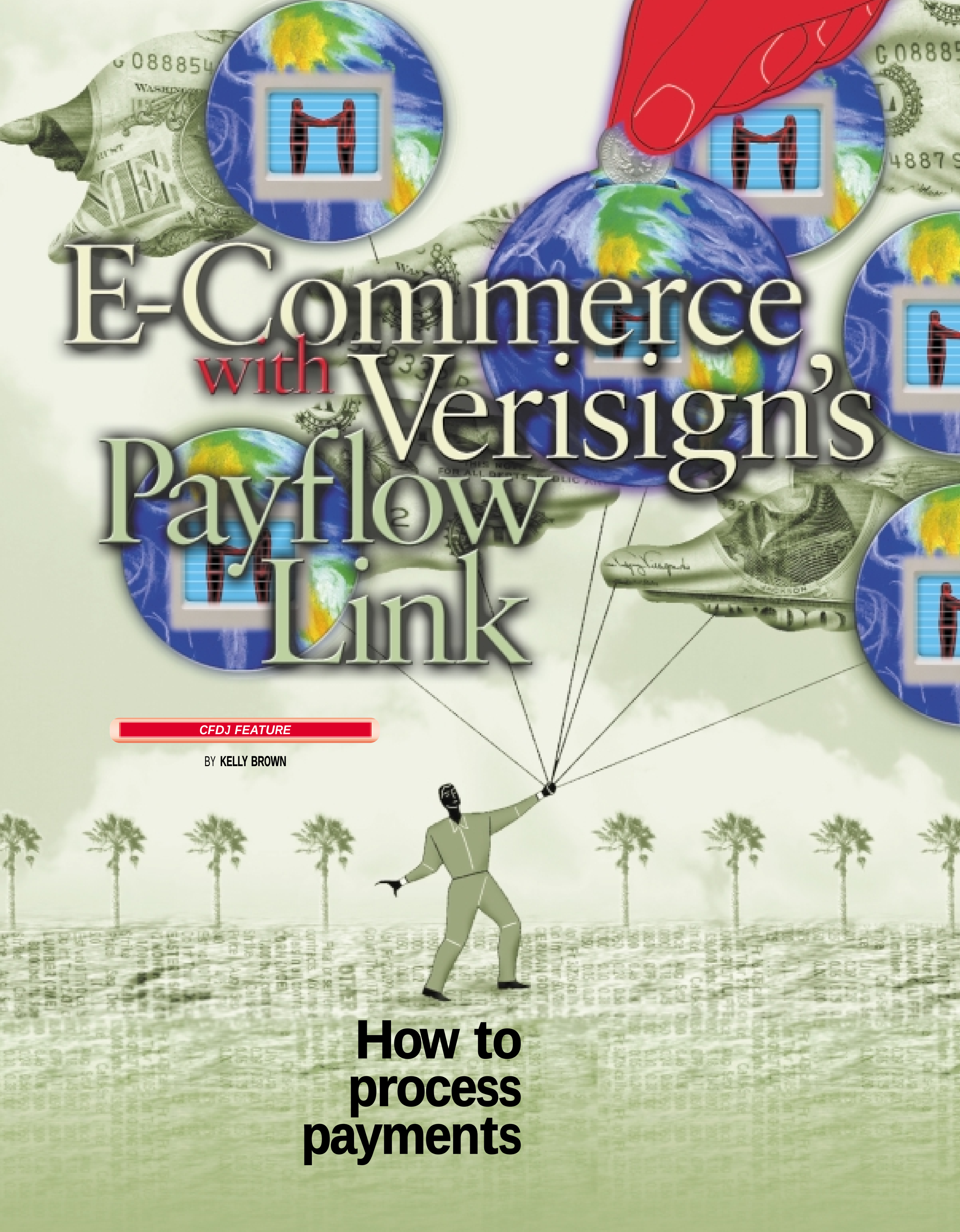
Robert Diamond

ROBERT@SYS-CON.COM

ABOUT THE AUTHOR

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

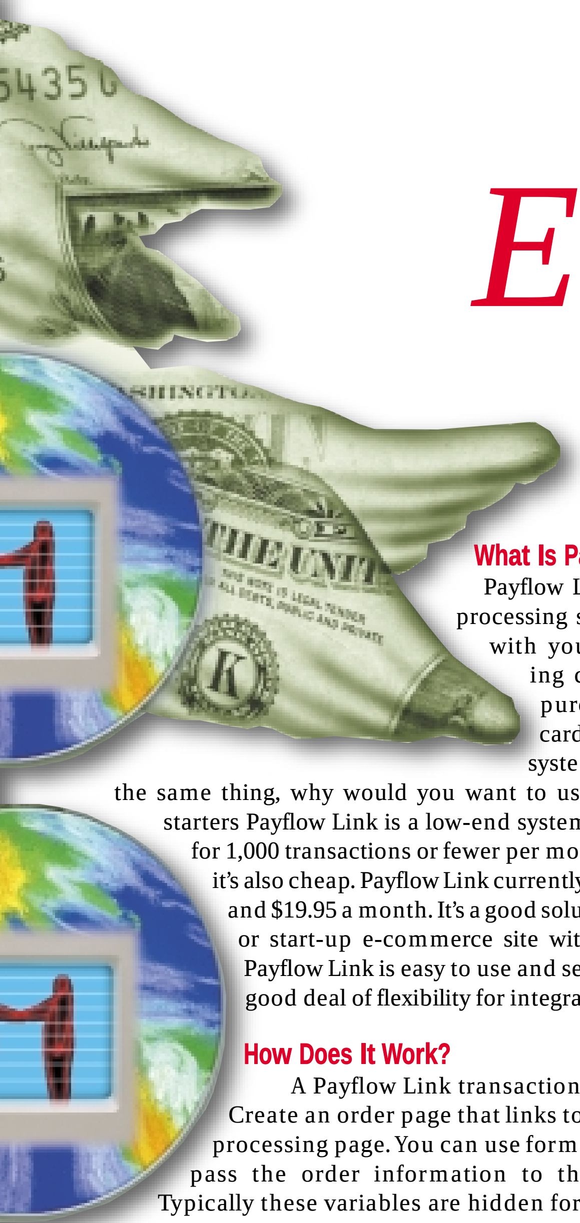
E-Commerce with Verisign's Payflow Link



CFDJ FEATURE

BY **KELLY BROWN**

How to process payments



E-commerce, for most people, means purchasing an item on the Web. This multistep process involves product selection, payment, and product delivery. This article focuses on the payment process.

What Is Payflow Link?

Payflow Link is a credit card processing system. It integrates with your Web site, allowing customers to make purchases using credit cards. As there are other systems available that do the same thing, why would you want to use Payflow Link? For starters Payflow Link is a low-end system. It's recommended for 1,000 transactions or fewer per month. That being said, it's also cheap. Payflow Link currently costs $179 for setup and $19.95 a month. It's a good solution for a small scale or start-up e-commerce site with a limited budget. Payflow Link is easy to use and secure, and provides a good deal of flexibility for integrating into your site.

How Does It Work?

A Payflow Link transaction starts on your site. Create an order page that links to the secure Payflow processing page. You can use form or URL variables to pass the order information to the Payflow system. Typically these variables are hidden form fields. They contain information that identifies your account and information about the purchase. Once the user is on the secure Payflow page, he or she is asked for personal and credit card information. The credit card transaction is processed and the user is returned to your site. When Payflow returns the user to your site, it passes along information about the transaction, such as whether the credit card transaction succeeded or not.

In addition to processing credit cards, the Payflow system allows you to log in and view information about your account. You can enter your merchant account information, customize your Payflow options, manually process credit card orders, and view credit card transaction reports. Figure 1 shows a sample of orders made through Payflow Link.

Payflow Settings

Integrating Payflow Link with ColdFusion is straightforward since the Payflow system is written in ColdFusion. You can sign up for a free Payflow Link test account on Verisign's Web site, www.verisign.com.

First we need to configure the Payflow Link, which is under the Account Info menu, using the Payflow Link Info option. Payflow Link is very flexible and has a lot of options; I'll only cover some of the more important ones.

The first configuration area is the Form Configuration. This controls which fields the purchase form on the Payflow site will contain and where the user will return to when the purchase is complete.

The first field is the Return Method. Your options are POST and LINK. Select link if you just want to link back to a thank you page. We want to process the data from the transaction so we're going to select post. The next option is the Return URL. This is where the user will be sent when the transaction is complete. I'll be using a thank you page that does some additional processing so I'm going to link back to my site at http://mysite/thankyou.cfm.

Next are the silent post options; however, to keep this example simple, I won't use them. With these options you can post data back to one page on your site while directing users to another page. This is a security option that can help prevent users from trying to manipulate form data to create fake orders, or to bypass the order process. Once you become familiar with the Payflow Link system you should look into these options.

The next fields are for gathering user information. The user has the option of entering his or her name, address, city, state, zip, counter, phone number, and e-mail. We're going to select these options and the editable options for our sample so the user can enter these values. In some cases you may not want these fields editable; for instance, you may want to pass these variables in through the initial form without the user changing them.

The next option is the Transaction Process Mode. If you're in test mode, test is your only option. Once you activate your site for real transactions, you still have the option of setting it to test if you're working on your code.

The next area of customization is the General Display options. I won't go into details, but you have lots of color options and can upload your site's logo. The Receipt Display options area allows you to set information that people will see when their credit card is processed. The E-Mail options area allows you to configure credit receipts to customers and send yourself e-mail alerts when orders are processed.

The final customization area is Security options. We have two options to configure in this area. The first is the Address Verification Service (AVS) provided by credit card services. This service tries to prevent fraud by ensuring that a customer's street address and zip code match the billing address for the credit card. You have the option of using AVS at various levels or turning it off. There's a lot of debate about the value of this system. There are problems with matching addresses exactly and many times it's too slow, which means it times out and accepts a transaction even if the address and zip don't match.

As the merchant you are responsible for credit card transactions, so it's probably a good idea to turn AVS on in at least the lowest level, so you can justify your security to your bank. If you make many purchases online, you'll notice that a lot of sites ask you to enter your zip code along with your credit card number so they can do AVS checking. It's easier to get a match on a zip code than an address.

The second security option is Accepted URLs. You have five fields to enter URLs. These are the only pages that the process form will accept for processing. This protects your purchase system

COLDFUSION EDGE 2001, FAST TRAC

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

ColdFusion fasttrack

REGISTER ONLINE TODAY!

Save $300

Save the Dates

Prepare for ColdFusion Certification

Kevin Lynch

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

Jeremy Allaire

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

Ben Forta

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

Plan to Attend

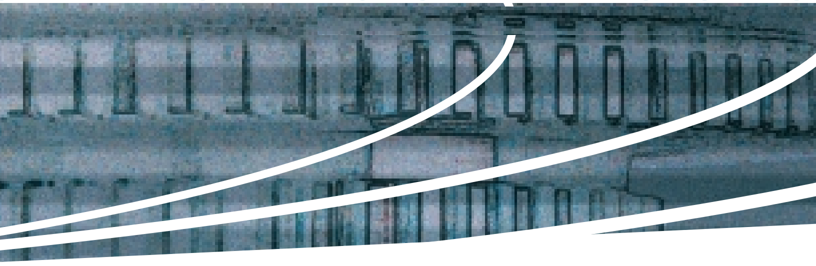
Prepare for ColdFusion Certification Exam with Comprehensive Technical Sessions designed for Beginner and Advanced ColdFusion Developers

Session Topics include:

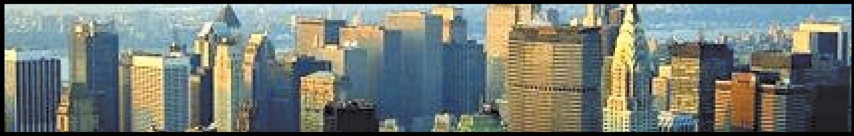
- Web Fundamentals
- Application Development
- Database Concepts
- Client State Management
- Troubleshooting, and a special focus on ColdFusion 5.0.

webServices JOURNAL · JAVA DEVELOPER'S JOURNAL · XML JOURNAL · wireless · LINUX BUSINESS WEEK · COLDFUSION Developer's Journal · WebSphere DEVELOPER'S JOURNAL · Dr. Dobb's · webtechniques

CK FOR COLDFUSION CERTIFICATION



ANNOUNCING WEBSERVICESEDGE 2001
COLDFUSION Developer's Journal
macromedia
Q&A
<allaire>
CFDJ Feature: Logging a SQL Using
Nested Tags
Custom Tags: Rolling Your Own...
<CF> on <CF>: Building a Bet
Custom Tags: The Flexibility o



Hilton New York, New York City

fasttrack Program

SEPTEMBER 24-25 2001

Call 201 802-3004 to reserve exhibit space today!

A Preliminary Glance at Topics

- ColdFusion 5.0
- Cross browser support
- Browser/server interactions
- HTML standards related to ColdFusion
- Variables and scoping
- Looping
- Functions
- <CFTAGS>
- Conditional processing
- Custom Tags
- Arrays and their usage
- Dimensions
- CFSCRIPT possibilities and limitations
- Structures
- Passing structures into custom tags
- Variable scopes

- Combining complex variables
- Upload files
- MIME types
- Append/Retrieve/Download
- <CFCONTENT>
- Creating agents
- Tags and parameters
- CF HTTP
- Syndication
- WDDX
- Parameterizing
- Data-parsing techniques
- HTTP header information
- Exception handling
  Database interactions
- Stored procedures
- CF transaction management

- Bind parameters
- SQL queries
- Joining tables
- Grouping data output
- Handling nulls
- Database manipulation
- Caching queries
- Application.CFM
- Cookies
- Client variables
- Session variables
- Application variables
- Server variables
- Storage locations
- Debugging

Owned and Produced by

SYS-CON EVENTS

135 Chestnut Ridge Road
Montvale, NJ 07645
(201) 802-3069
Fax: (201) 782-9051
visit
WWW.SYS-CON.COM/COLDFUSIONEDGE
for more information

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

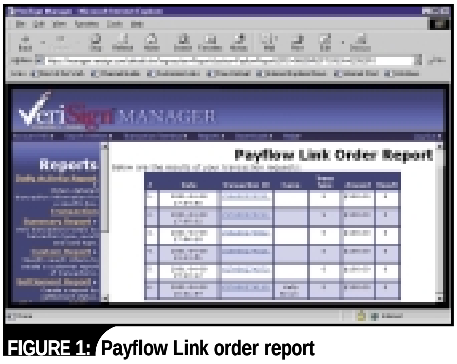


FIGURE 1: Payflow Link order report

from people creating their own forms that submit to your system. For instance, someone can't create a form to purchase a laptop computer for a dollar. Our sample site is going to use http://mysite/purchase.cfm as the only page to accept links from.

Figure 2 shows the sample processing page I created including a logo that I uploaded.

Purchase Page

Listing 1 shows a sample purchase page named purchase.cfm. If this was a real page, the fields would be dynamically generated, but I'm keeping things simple for this example. This page displays a simple table with a list of items purchased and a form with several hidden fields. The form action is https://payflowlink.verisign.com/payflowlink.cfm, which is the Payflow Link processing page. Note that this is a secure page and uses https instead of http.

There are a lot of fields you can use, but we're going to use only six: LOGIN, PARTNER, AMOUNT, TYPE, DESCRIPTION, and USER1. The first four fields are required for all transactions. The LOGIN is your Payflow LOGIN, which is also your vendor ID. The PARTNER is the company that set up your account; if you got it directly from Verisign, it will be Verisign. AMOUNT is the amount of purchase, a number. TYPE is the transaction type.

There are two types of transactions you will generally use – S for authorize and capture and A for authorize only. *Authorize* marks the customer's card for a deduction of the specified amount. *Capture* actually transfers the money from the customer's account to yours. Why wouldn't you always use the authorize and capture option? Legally you can't deduct money from a person's account until the order is shipped. If you're selling something that's delivered immediately, such as a subscription to a Web site, you can use the authorize and capture option. However, if you're shipping physical goods you should wait until the item is shipped to capture the charge. Notice I said *should*. Nothing prevents you from always using the authorize and capture, but you do so at your own risk.

The next two fields are not required, but are nice to have. The first is DESCRIPTION, which describes the purchase. You have a limit of 255 characters for this field, so use it carefully. The other field is USER1, which is a custom field. We can put anything we want in this field. The customer never sees this information, but it's good for passing along data we want to process later. I'm going to put the item purchased in this field.

Thank You Page

Listing 2 contains my sample thank you page named thankyou.cfm. This page simply displays a "Thank you for your purchase" to the user. It also inserts the transaction into our database. We grab the user's information, including name, address, city, state, zip, e-mail, and phone number. In addition, we gather some information about the transaction. The amount of the transaction is stored. Our customer field USER1 contains the item purchased so we store it in the item field.

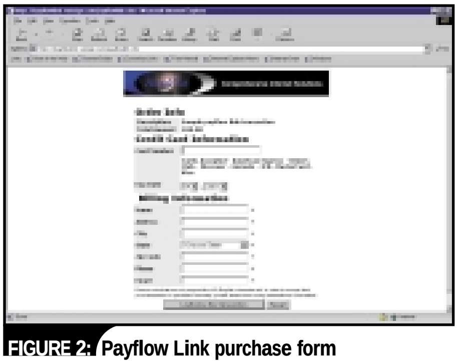


FIGURE 2: Payflow Link purchase form

Another field that's useful to have is the Payflow Link reference number. This is returned as the PNREF variable and is placed into the payflow_id field in the database. You can use this field to cross reference orders in your database with the transactions in the Payflow Link reports.

Conclusion

I've covered just the bare minimum of integrating with Verisign's Payflow Link system, but it should be enough to get you started. Once you get the basics down you can try some of the more advanced features. If you're looking for a cheap and easy way to get your e-commerce site to accept credit cards, Payflow Link is a good solution.

About the Author

Kelly Brown is the CTO of AboutWeb (www.aboutweb.com), an Internet solutions provider in the Washington, DC, area. He has a BS and MS in computer science and is a Microsoft-certified systems engineer.

kbrown@aboutweb.com

Listing 1

```
<html>
<head>
 <title>Purchase</title>
</head>
<body bgcolor="white">
<div align="center">
<h2>Purchase Item</h2>
</div>
<P>
<div align="center">
<TABLE border=1>
<TR>
<TH>Item</th>
<TH>Price</th>
</tr>
<TD>Item 1</td>
<TD>$100</td>
</table>
<form method="POST"
 action="https://payflowlink.verisign.com/payflowlink.cfm">
<input type="hidden" name="LOGIN" value="Your Login">
<input type="hidden" name="PARTNER" value="verisign">
<input type="hidden" name="AMOUNT" value="100">
<input type="hidden" name="TYPE" value="S">
<input type="hidden" name="DESCRIPTION"
   value="Sample payflow link transaction">
<input type="hidden" name="USER1" value="Item 1">
<input type="submit" value="Click here to Purchase">
</form>
</div>
</body>
</html>
```

Listing 2

```
<CFQUERY Name="insertorder" Datasource="Payflow">
   insert into purchase
      (name,address,city,state,zip,email,phone,
       amount,item,payflow_id)
   values
   ('#Form.name#','#Form.address#','#Form.city#',
    '#Form.state#','#Form.zip#','#Form.email#',
    '#Form.phone#',#Form.amount#,'#Form.user1#',
    '#Form.pnref#')
</CFQUERY>
<html>
<head>
 <title>Thank You</title>
</head>
<body bgcolor="white">
Thank you for your purchase.
</body>
</html>
```

CODE LISTINGS

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

XMLEDGE2001 INTERNATIO

MEE

XMLEDGE conference&expo 2001

The ONLY XML Event
Backed by the Power of
SYS-CON MEDIA and XML JOURNAL

Plan to Attend...This 4-Day Cutting-Edge Conference

Save the Dates!

XML...Dramatically Changing the Substance of Corporate Systems

- XML will fundamentally improve the speed, cost and flexibility of business applications.
- IT Managers expect XML Technology to enable performance of new operations and shorten application development time by 30% to 50%
- E-commerce transactions using XML are predicted to rise from .5% in 2000 to more than 40% by the end of 2003.
- XML products now under development will hit the market in the second half of 2001. *See them first at XMLEdge!*

www.XMLEdge2001.com

Conference Tracks

Track 1: PRESENTING XML

The latest tips, tricks and strategies for presenting XML in various front-end presentation environments

XSL / XSLT / CSS / XLINKS / XPOINTERS

XHTML

BROWSER SUPPORT FOR XML APPLICATIONS

WML AND WAP

VOICEXML

SCRIPTING AND XML (PERL / PHP / PYTHON / JAVASCRIPT)

WEB SERVICES

WIRELESS XML: WAP / WML, XML AND MOBILE COMMUNICATIONS

Track 2: REAL-WORLD XML

Actual Case Studies using XML and related technologies in various industry sectors

XML TOOLS / UTILITIES APPLIED TO ACTUAL APPLICATIONS

DOCUMENT AND CONTENT MANAGEMENT (EDITORS / PARSERS / SERVERS / TRANSLATORS)

REAL-WORLD APPLICATIONS

XML ISSUES IN THE REAL WORLD (PERFORMANCE / SECURITY / COMPRESSION / SERIALIZATION)

USE OF XML IN INDUSTRY SECTORS (MANUFACTURING / HIGH TECH, ETC.)

Track 3: XML & JAVA

The latest developments in XML and Java environments as well as insights into developing solutions using XML, Java and related technologies

DOM/SAX / JDOM

JAVA AND XML PROGRAMMING (JSP, JMS, ETC.)

PARSING XML IN JAVA

XML AND DATABASE PROGRAMMING (XQL)

Track 4: EAI with XML

Comprehensive coverage on developing Enterprise Application Integration solutions with XML

XML AND APPLICATION SERVERS

B2B SERVERS

PROCESS AND WORKFLOW INTEGRATION

XML-BASED INTEGRATION USING MESSAGING SYSTEMS

ERP INTEGRATION

DATABASE INTEGRATION

Track 5: XML STANDARDS

The latest news and updates on emerging XML standards from the industry gurus

W3C STANDARDS

XML DESIGN PATTERNS

DOM/SAX / JDOM

ROSSETTANET

SOAP / BIZTALK

DTDS AND XML SCHEMAS

XML REPOSITORIES (BIZTALK / OASIS)

NAL CONFERENCE & EXPO

ETING BUSINESS CHALLENGES WITH XML

CONFERENCE
October 22–25, 2001

EXPO
October 24–25, 2001

SANTA CLARA CONVENTION CENTER
Santa Clara, CA

The Largest International XML Conference of the Year! Reserve Your Exhibit Space Now!

Get the Competitive Edge...

- Over 100 exhibitors will have access to the most qualified audience of potential buyers and specifiers.
- They will demonstrate, meet the movers and shakers, solve their challenges and sell!

Plan to Sponsor – Many Sponsorships Are Exclusive!

- Demonstrate your commitment and capabilities to the largest XML audience to assemble on the West Coast in 2001!
- Rise above the noise. Deliver your message with the marketing support of SYS-CON Events and SYS-CON Media.

Who Should Attend XMLEdge2001

i-Technology Professionals
Developers, Programmers, Engineers

Technical Architects
Senior IT / IS Management

Analysts, Consultants
Product & Program Managers

Call 201 802-3069 to reserve your space today!

The only XML event backed by the full marketing power of **SYS-CON Media** and ***XML-Journal*** is your best opportunity to reach the largest, most qualified audience of software developers and *i*-technology professionals in 2001.

Owned and Produced by
SYS-CON EVENTS
135 Chestnut Ridge Road, Montvale, NJ 07645
201 802-3069 • Fax: 201 782-9601
Visit
www.XMLEDGE2001.com
for more information

Last Year's Exhibitors Include...

Baltimore Technologies
DNU Magazine
EarthWeb
Eliad Technologies, Inc.
E-Numerate Solutions, Inc.
E-Security Journal
Fawcette Technical Publications
GeekCruises.com
Hit Software, Inc.
IAM Consulting
IBM
Infoshark, Inc.
Infoteria Corporation
Ingeniux Corporation
Insight Software, Inc.
Interknack Corporation
IXIASOFT, Inc.
JavaPro

LearningPatterns.com
Mailers Software
Merant
Microsoft
Mindgap Systems Ltd.
MSDN Magazine
Nanobiz
Neocore, LLC
Netfish Technologies, Inc.
Oasis
Object Management Group
O'Reilly Network
Planet 7 Technologies Corp.
Progress Software Corporation
Pybiz
Radiant Logic, Inc.
Rogue Wave Software, Inc.
SD Times
Seagull

Search XML Resources.com
Sequoia Software Corporation
Skill Server
SoftQuad Software, LTD.
Sun Microsystems
Talva Corporation
The Gilbane Report
Tibco Software, Inc.
Whitehill Technologies, Inc.
X-Aware
XML Global Technologies, Inc.
XML-Journal
XML Magazine
XML Times
XML.com
XML.org
XML/EDI Group
XMLResources.com
XYZ Find

XML-JOURNAL XML-J READERS' CHOICE AWARD

XML–JOURNAL READERS' CHOICE AWARDS – THE "OSCARS OF THE SOFTWARE INDUSTRY" – RECOGNIZE 13 WINNERS AND 39 FINALISTS

AWARDS WILL BE PRESENTED AT XMLEDGE, THE LARGEST INTERNATIONAL GATHERING OF XML PROFESSIONALS ON THE WEST COAST OF THE U.S. IN 2001

NEITHER SYS-CON MEDIA NOR SYS-CON EVENTS, INC., IS A SPONSOR OF, OR IS AFFILIATED IN ANY WAY WITH, CAMELOT COMMUNICATIONS, INC., OR ANY EVENTS PRODUCED BY CAMELOT COMMUNICATIONS, INC.
JAVA AND JAVA-BASED MARKS ARE TRADEMARKS OR REGISTERED TRADEMARKS OF SUN MICROSYSTEMS, INC., IN THE UNITED STATES AND OTHER COUNTRIES. SYS-CON MEDIA AND SYS-CON EVENTS ARE INDEPENDENT OF SUN MICROSYSTEMS.
THE XMLEDGE2001 INTERNATIONAL CONFERENCE & EXPO IS IN NO WAY RELATED TO XMLEDGE BY MOBILEQ, A PLATFORM FOR A COMPLETE WIRELESS INTERNET INFRASTRUCTURE FOR DEVELOPING, INTEGRATING, MANAGING AND EVOLVING APPLICATIONS FOR THE BROADENING SPECTRUM OF WIRELESS DEVICES.

BY **KAILASNATH AWATI** AND **MARIO TECHERA**

ColdFusion *and* Oracle:

Putting Oracle SQL to Work in Your ColdFusion Applications

Part 1

Using Oracle's feature set within your Internet applications

SQL is the bread-and-butter language of relational databases. Although the language has been standardized (SQL-92 and SQL:99), virtually no vendor's implementation strictly conforms to the standards. Most database products use their own syntax (in joins, for example) and, more important, implement several language extensions.

For the developer it's necessary to know about product-specific extensions in order to leverage the database optimally. This is true whatever the nature of the application – Web or client/server. In addition, for the Web programmer knowledge of these extensions can help shift a lot of the processing load off the Web/application server to the database where it belongs (see Ben Forta's comments on this in "Take Your Database Out of Retirement," ***CFDJ***, Vol. 1, issue 3). In this article we present some features of Oracle SQL that we've found useful when developing ColdFusion applications.

This article is separated into two parts. Part 1 explores some of the built-in functions that you have at your disposal when working with Oracle and compares them to the corresponding ColdFusion functions. Part 2 will show how some basic application problems can be solved with SQL, with particular reference to Oracle-specific syntax. This overview is not comprehensive, but it should help the Oracle novice find his or her way around the database by pointing out features that work particularly well in ColdFusion application development.

Leveraging Oracle Functions in ColdFusion

This section presents functions that Oracle delivers "out of the box." Oracle also lets you define your own functions (analogous to custom tags in CF), but this process will be explained in a later article. For the moment, keep in mind that this discussion is just the tip of the iceberg.

For quick reference, we've listed ColdFusion string, numeric, and date functions with their Oracle counterparts in Tables 1, 2 and 3, respectively. Many ColdFusion functions have no direct counterpart in Oracle and vice versa. However, as you'll see in many of the following examples, much of what can be achieved with ColdFusion functions can also be achieved by using one or more Oracle functions in tandem.

Many examples use an Oracle table called DUAL. This is a single row table that contains a dummy entry (as can be seen by selecting * from DUAL). The table is owned by SYS (which is the user with the most privileges in Oracle installations), and most Oracle installations make it available to all users through a public synonym. Selecting from DUAL is a useful technique when you want to have Oracle do a computation and return the result to you.

Invoking Oracle Functions in ColdFusion

As you may be aware, ColdFusion doesn't allow you to invoke Oracle functions directly through its CFSTOREDPROC tag. However, you can make function calls through CFQUERY by simply SELECTing from DUAL. For example:

```
<cfquery name="yourFunction"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      yourFunction(arguments) return_value
   from
      dual

</cfquery>
```

The variable yourFunction.return_value[1] contains the returned value.

A Simple Audit Trail Using USER and SYSDATE

It's extremely useful to keep track of modifications (inserts or updates) to tables. One simple way to do this is to add four audit columns to any table whose data movements you wish to keep track of. CREATION_USER keeps track of the user that created the record; CREATION_DATE, the date the record was created; MODIFICATION_USER, the last user who modified the record; and MODIFICATION_DATE, which stores the date the record was last modified. Then, when carrying out an insert or an update, you merely insert (or update) these four (or two) columns using the Oracle functions USER and SYSDATE, respectively.

The insert in Listing 1 enters the current account user and system date (down to the second) into the four columns. These columns can then be used at any time to keep track of who made modifications to your data and when.

It's also worthwhile noting here that if you need extensive auditing of your data, that is, keeping exact track of any and all modifications to your data, Oracle has an advanced auditing system built into the RDBMS. The audit trail produced by the built-in system not only allows you to audit data, but also user sessions. A detailed discussion of the Oracle audit trail is beyond the scope of this article, but it's important for you as a CF developer to know that it exists in case you need advanced auditing capabilities.

Inserting Dates into Oracle

As we're sure many readers have experienced, inserting dates into an Oracle database from ColdFusion can be a bit trying. One simple way to do this is to pass date values as strings and convert the string to a date data type using the Oracle TO_DATE function (see Listing 2).

Truncating Dates

An Oracle date data type stores the date and a timestamp. This can get annoying sometimes, since you may want to filter out data based on date only (disregarding the time portion). One way of achieving this is to use the TRUNC function when comparing dates. This function sets the timestamp to midnight, as is made clear when executing the following query:

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      to_char(trunc(sysdate),'DD-MM-YYYY
      HH24:MI:SS') truncated_date
   from
      dual

</cfquery>
```

It's important to note that the second parameter of both TO_DATE and TO_CHAR functions is extremely flexible and easy to use. This parameter, formerly called a date mask, allows you to deal with just about any date/time format you may want to use.

COLDFUSION STRING FUNCTION	ORACLE CHARACTER FUNCTION
ASC	ASCII
CHR	CHR
FIND	INSTR
LCASE	LOWER
LEN	LENGTH
LTRIM	LTRIM
MID	SUBSTR
REPLACE	REPLACE
RTRIM	RTRIM
TRIM	LTRIM, RTRIM
VAL	TO NUMBER

TABLE 1 ColdFusion and Oracle character functions

Using DECODE

Did you know that Oracle provides a function that behaves similarly to the CFIF tag? If you didn't, look closely at the DECODE function. What makes DECODE so appealing is that you can use it just about anywhere within a SQL statement. Its syntax is:

```
DECODE(expression,
   evaluated_value_1, value_1,
   evaluated_value_2, value_2,
   evaluated_value_N,value_N,
   default_value)
```

where DECODE returns one of value_1, value_2,..,value_N, or default_value depending on whether *expression* evaluates to evaluated_value_1,evaluated_value_2,..,evaluated_value_N, or some other value (treated as the default).

This function is extremely useful in making dynamic assignments in SQL statements. As an example, assume that we pass a numeric month value (1 through 12) via a ColdFusion variable (form.month) to a select statement and, depending on the value of form.month, we want to select the appropriate month from a table containing monthly values. DECODE sets the column name on the fly (see Listing 3).

With the select statement in Listing 3 you've effectively turned the month into a parameter that causes the database to return data from different columns depending on its value.

The other way to achieve the same result would be to use a CFCASE or CFIF – a tedious and expensive option. DECODE is much more elegant, and it also pushes the task on to the database, thereby conserving your ColdFusion resources. Also note that Listing 3 also shows that the default_value is optional.

Ensuring Consistent Capitalization of Returned Values

Sometimes data is inserted into tables with inconsistent capitalization. For instance, some values in a column may have been entered in uppercase whereas others may be in lower. This can look quite ugly when displayed on a Web page. You can ensure consistent capitalization by using the INITCAP and LOWER functions in tandem.

```
<cfquery name="yourQuery"
  datasource="yourDSN"
  username="yourUsername"
  password="yourPwd">

  select
     initcap(lower(columnName)) initial_cap
  from
     yourTable

</cfquery>
```

Simple String Parsing Using INSTR and SUBSTR

At times you may want to extract a substring from a character string; however, you only know the characters delimiting the substring, not the numeric start and end positions of the substring. In this case, you can't use the SUBSTR (substring) function directly. However, you can use the INSTR function to get the positions of the delimiting characters and pass these on to the SUBSTR function. An (artificial) example should help make the preceding prose a bit clearer. Here we want to extract the word "whom" from the substring "for.whom.the.bells.toll".

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      substr('for.whom.the.bells.toll',
instr('for.whom.the.bells.toll','.',1,1)+1,

instr('for.whom.the.bells.toll','.',1,2)-
instr('for.whom.the.bells.toll','.',1,1)-1)
      second_string
   from
      dual

</cfquery>
```

Yes, we know you can achieve the same thing using the ColdFusion string function GetToken, but, remember, the idea is to stress your database and save your ColdFusion server.

Concatenating Strings

Sometimes it's necessary to concatenate two or more columns from a table into a single quantity for display purposes. One example of this would be concatenating the employee first name, last name, and ID into a single entity for display on a Web page. The easiest way to do this is via the concatenation operator ||, like so:

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      employee_ID|| ' '
||Initcap(lower(first_name)) || ' ' ||
Initcap(lower(last_name))
      emp_name
   from
      employees

</cfquery>
```

The query column alias emp_name contains the concatenated information. By the way, the concatenation operator is useful if you want to generate a comma-delimited file for import into Excel, for example. We'll leave this as an exercise for our readers.

Trimming Strings

The Oracle functions LTRIM (left trim) and RTRIM (right trim) are useful in trimming strings. These functions allow you to trim arbitrary characters from the ends of a string, as well as the usual whitespace characters.

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      ltrim(rtrim(‘abc012cbb’,’abc’),’abc’)
trimmed_string
   from
      dual

</cfquery>
```

The first argument of these functions is the string to be trimmed and the second is the trim string, which contains the characters to be trimmed out. Thus, the above query returns the string "012" – the characters "a", "b", and "c" are trimmed out from the left and right ends of the string. If you don't specify the second string, the functions assume you want to trim white-space characters – equivalent to the ColdFusion Ltrim and Rtrim functions.

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      ltrim(rtrim(‘   012     ’)) trimmed_string
   from
      dual

</cfquery>
```

This returns the string "012". Again this can be done (but shouldn't be) on the ColdFusion server.

Replacing Characters in a String

Trimming strings is fine, but what if you want to remove characters from the middle of a string, or if you want to replace one substring with another. No problem, Oracle offers you the REPLACE function. Here's an example:

COLDFUSION MATHEMATICAL FUNCTION	ORACLE NUMERIC FUNCTION
ABS	ABS
ATN,	ATAN
CEILING	CEIL
COS, SIN, TAN	COS, SIN, TAN
EXP	EXP
INT	FLOOR
LOG	LOG
MAX	GREATEST
MIN	LEAST
ROUND	ROUND
SGN	SIGN
SQR	SQRT

TABLE 2 ColdFusion and Oracle mathematical functions

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">
   select
      replace(‘01abc2’,’abc’) processed_string
   from
      dual

</cfquery>
```

This removes the "abc" in the middle of the first string. Here's an example in which one string is replaced by another:

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">
   select
      replace(‘01abc2’,’abc’,’def’)
processed_string
   from
      dual

</cfquery>
```

This replaces "abc" with "def".

Translating Characters in a String

The REPLACE function looks for an exact match of the replace string. What if you want to replace strings based on a character-by-character translation? For this, use the TRANSLATE function, see the following:

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">
   select
      translate(‘01abc2’,’abc’,’def’)
processed_string
   from
      dual

</cfquery>
```

What would happen if you change the string to "01acc2", keeping the replace and translate strings the same? Now that you've tried it, you know what TRANSLATE can do for you.

Arithmetic with Nulls

Database practitioners know that nulls in arithmetic expressions yield null results. However, sometimes we'd like to pretend that null values contribute a known value (usually zero) to a sum. For example, when summing a salary and commission column, where the commission column is nullable and the salary is not, we would like the sum to yield the salary in case the commission is null; that is, we'd like to treat a null commission as a zero value. This can be handled using the NVL function, as seen in this query:

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

  select
     2+nvl(null,0)
   from
     dual

</cfquery>
```

Warning the User About Data Volume

Assume your application has access to some tables with a large number of records, and the specs of the application require that users be permitted to drill down into the data as deep as they want (yes, it's crazy, but unfortunately some clients do demand such functionality). Before executing the (potentially fatal) queries, you may want to give the end user some feedback regarding the damage he or she is about to cause. The most straightforward way of doing this is by counting the number of records before you submit the query:

```
<cfquery name="countRecs"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

  select
     count(*) total_recs
   from
     …
</cfquery>
```

However, the number may not be very significant if you're dealing with wide tables with lots of columns or where the data types of the columns are very large, for example, VARCHAR2(2000). You can actually provide users with a good esti-

mate of the volume of data they're going to slosh about over the intranet/Internet by using the VSIZE function as follows:

```
<cfquery name="dataSize"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

  select
      NVL(VSIZE(column_1),0) +
NVL(VSIZE(column_2),0) + …
size_in_bytes
   from
      …
</cfquery>
```

VSIZE returns the number of bytes in the internal Oracle representation. The above query sums the total value of bytes in the internal representation for all the columns you've included and the records you've selected. Although this won't be exactly the volume of data to be transferred, it will be extremely close (for example, one character is one byte in Oracle internal representation). You can also use the same technique to prevent users from accessing more than a set volume of data.

Oracle SQL Extensions and Idiosyncrasies

Now that you've seen how Oracle SQL functions can be useful in your work, we'll show how you can avoid some complex ColdFusion coding by using some Oracle-specific SQL extensions and concepts.

Pseudo Columns

Pseudo columns behave identically to normal table columns, however, they're not stored anywhere and can't be updated or inserted. You're probably wondering what the last sentence means, so it's probably best to give an example.

One of the pseudo columns that Oracle makes available to every result set is the ROWNUM. For each row returned by a query, the ROWNUM pseudo column returns a number indicating the order in which Oracle selects the row from a table or a set of joined rows. The first row selected has a ROWNUM of 1, the second has 2, and so on. A frequent requirement is to get the top *N* records in a table. If you combine the ORDER BY clause in a query with a WHERE clause on the ROWNUM pseudo column, you can easily generate top *N* hit lists. For example:

```
<cfquery name="topN"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">
select  *  from
   (select employee_id, salary
     from employees order by salary desc)
where
   ROWNUM < 11;

</cfquery>
```

This query returns, very simply, the top 10 salaried employees and their salaries. No additional ColdFusion code is necessary.

This example also shows that the FROM clause doesn't necessarily need to contain a table name; it can even refer to a set of records returned by a subquery.

There's another Oracle pseudo column – ROWID – that's essentially the physical storage address of the row. This can be useful in constructing a SQL statement to detect duplicate rows in a table. We'll leave you to mull over how this can be done.

COLDFUSION DATE FUNCTION	ORACLE DATE FUNCTION
CREATEDATE	TO DATE
CREATEDATETIME	TO DATE
DATEADD	ADD MONTHS*
DATEDIFF	MONTHS BETWEEN*

TABLE 3 ColdFusion and Oracle date functions

Outer Joins

Outer joins are useful when you want to display the entire primary key of a reference table, even when the fact table (or referring table) doesn't contain data for all values of the foreign key. A simple example may be useful: consider a PRODUCT table with columns PRODUCT_ID and DESCRIPTION, and a SALES table with columns PRODUCT_ID and VALUE_SOLD. You may want to create a report that displays sales for all available products, including those for which no sales have been made. (Listing 4 shows an outer join.)

The first thing to notice in Listing 4 is the syntax for the outer join, which could be quite different from what you're used to in other database products. The principle is the same, though: the (+) on the SALES side of the join tells Oracle that you want all products returned, including those for which no sales have been made.

Bill of Materials (Hierarchy) Problems

In relational databases, hierarchies are usually represented through self-referential tables – the foreign key refers to the primary key of the same table. An example would be an EMPLOYEES table with EMP_ID as the primary key and MANAGER_ID as the foreign key (referring to EMP_ID). To decompose the hierarchy, join the table to itself as many times as there are levels in the hierarchy. However, the number of levels is usually not known beforehand. Oracle provides an easy way to deal with such a situation through its nonstandard START WITH...CONNECT BY PRIOR clauses (see Listing 5).

The query assumes that the top person in the hierarchy has no manager (manager_id is NULL). Note the use of the LPAD function to offset the records depending on the hierarchy level. The pseudo column LEVEL is always returned by such a query, and, as the name suggests, denotes the hierarchy level of the record.

Locking Records

A problem often encountered when developing a system that allows a user to update or delete data in database tables via the browser is that you want to make sure the user is the only one working on the records. Usually a CF developer will call up the records in question with a SELECT statement and then, based on user input, these records will be modified with a DELETE or UPDATE statement. However, if the records are not locked, another user may modify the data through another session and thus the current user is modifying "old" data.

There are many ways of dealing with such problems, often depending on the context of the transactions in question. One of the simplest ways is to use the SELECT…FOR UPDATE command. When you write your CFQUERY tag to call up the required records for display, add the FOR UPDATE command at the end of the select statement (see Listing 6). The result is that other database sessions will only be able to view the records in question, not modify them. The FOR UPDATE clause has placed a lock on the complete record set. Only the current user's session will be able to process an UPDATE or DELETE statement on the records. Any other session that attempts to modify the same data will automatically be forced to wait by Oracle. To release the locks and make the data available to other database sessions for modification, you need to send a COMMIT or close the current user's database session. Be careful when using such statements: make sure you lock only the necessary records using the WHERE clause. If not, you may accidentally cause other users to wait for COMMITs on data they're not interested in.

Conclusion

At this point we'd like to wrap up our introductory article in this series about using features of the Oracle RDBMS in your

ColdFusion applications. If you haven't yet looked at Oracle in detail, we hope we've managed to arouse your curiosity. The Oracle RDBMS is an extremely complex and efficient engine for storing and manipulating data, and it goes a long way in bringing about the principles set out by Dr. Codd, the founder of relational database theory.

As a ColdFusion developer you're in the privileged position of being able to pick and choose how best to utilize Oracle's rich feature set within your Internet applications. If you're clever about your choices, your applications will go into production quicker and run faster with fewer resources. Learn as much as you can about the database you're using – your users will thank you for your efforts.

By the way, one of the best sources for Oracle information is the Oracle developer's Web site at http://technet.oracle.com, and the free membership provides access to countless resources. Among the most important is the complete documentation for the Oracle RDBMS. You can view it online or download it in PDF format. Like any technical documentation it isn't exactly an inspiring read. However, it's comprehensive, well organized, and chock-full of examples, and is definitely your best bet for information on Oracle.

In Part 2 we'll delve into the procedural extensions that Oracle provides to SQL within the database: PL/SQL and Java.

About the Authors

Kailasnath Awati is a senior consultant at Williams & Partner, Management Consulting, where he manages the U.S. operations. He's an experienced Internet/database application developer and architect as well as an Allaire-certified ColdFusion developer. His personal Web site is www.orafusion.com.

Mario Techera specializes in Internet application design and development with relational databases. A founding partner at Williams & Partner, Management Consulting, he works in Munich, Germany.

kawati@wpmc.com

mtechera@wpmc.com

Listing 1

```
<cfquery name="yourInsert"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   insert into
      yourTable
      (column_1,
      ....
      creation_user,
      creation_date,
      modification_user,
      modification_date)
   values
      (column_1_value,
      ...
      USER,
      SYSDATE,
      USER,
      SYSDATE)

</cfquery>
```

Listing 2

```
<cfquery name="yourQuery"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   insert into
      yourTable
      (col_1,
      date_col)
   values
      (col_1_value,
      to_date(date_col_value,'DD-MM-YYYY')

</cfquery>
```

Listing 3

```
<cfquery name="monthlyValue"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

   select
      decode(#form.month#,
          1,january_value,
          2, february_value,
          3, march_value,
          4, april_value,
          5, may_value,
         6, june_value,
         7, july_value,
         8,august_value,
         9,september_value,
         10, october_value,
          11, november_value
          12,december_value) monthly_value
      from
          monthly_values
      where
          entry_id=#form.id#

</cfquery>
```

Listing 4

```
<cfquery name="sales"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">
select
   t1.product_id "product number",
   t1.description "description",
   t2.value_sold "sales"
from
   product t1,
   sales t2
 where
  t1.product_id=t2.product_id (+)

</cfquery>
```

Listing 5

```
<cfquery name="employeeHierarchy"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

  select
     lpad(' ',2*(level-1)) | |name
  from
     employees
  start with
     manager_id is null
  connect by prior
     emp_id=manager_id

</cfquery>
```

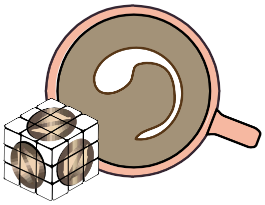
Listing 6

```
<cfquery name="Lock Records"
   datasource="yourDSN"
   username="yourUsername"
   password="yourPwd">

 select
employee_id,
salary
from
employees
order by
salary desc
for update

</cfquery>
```

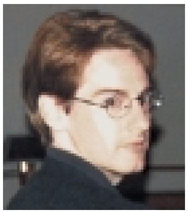
CODE LISTING ▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶▶

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

A Cold Cup o'Joe Part 4 of 8



Using Java objects



BY
GUY
RISH

In Part 3 of this series (***CFDJ***, Vol. 3, issue 4) we looked at the basics of creating and using Java objects in ColdFusion templates.

In Part 4, I discuss a number of other aspects of using Java objects in CFML as well as introduce some tools that I hope will be useful to you in debugging the Java components.

Bonus

While working on my own projects as well as this series I started to piece together a handy kit of Java classes for working with ColdFusion. The real payoff comes in the form of extending tools, two of which I'll introduce in this article: JVMLog and DebuggingFactory. These are fairly simple tools; however, they use a number of auxiliary classes that are beyond the scope of this article. I've packaged them all, in typical fashion, into a JAR that can be found on the ***CFDJ*** Web site. I'll release updates and additions to this JAR with future articles in this series. Each release is identified by month and year; this one is ccoj_042001.jar, for the April 2001 release.

Amend the CLASSPATH setting on the Java Settings panel to include this JAR.

A Confession

I lied in Part 3 of this series. It was a little white lie, but a lie nonetheless. Writing a Java class that works with ColdFusion requires some special attention. Allow me to plead my case anecdotally. One of the things I'm frequently asked is: Why doesn't my "Hello, World" program work when I call it from a ColdFusion template? Almost invariably I find that inquiring minds have constructed a valid "Hello, World" program just as they've been instructed to do in all their beginning Java books.

```
public class HelloWorld
{
 public static void main(String[]
 args)
 {
  System.out.println("Hello,
 World");
 }
}
```

Unfortunately, this doesn't work in ColdFusion. Remember, as I discussed in Part 3, a <CFOBJECT> or CreateObject call loads only the class and doesn't execute it until use. One of the things the Java command line interpreter will look for is a main method in the class file being executed. To add to the inquirer's frustration, the ColdFusion interpreter doesn't capture the standard output or standard error output of executing Java objects. Getting data or feedback from a Java object should happen though returned values from method calls. Designing a Java object that interacts with ColdFusion should reflect that ideal. So printing anything to System.out or System.err is verboten, strictly speaking. Thus a "Hello, World" in Java, written to be well behaved for ColdFusion, would look something like this:

```
public class HelloWorld
{
 public String sayHello()
 {
  return(new String("Hello,
 World"));
 }
}
```

and could be used in a ColdFusion template with a few lines:

```
<CFSCRIPT>
 obj = CreateObject("java",
 "HelloWorld");
 WriteOutput(obj.sayHello());
</CFSCRIPT>
```

Log It

While I steadfastly agree with the whole concept of well-designed classes with queryable state information, as opposed to unsolicited feedback, I felt this was a drawback. There are plenty of times when the ability to capture both the standard and error output is valuable, even if only for logging purposes. To that end I created a class that allows you to tee tap everything written to System.out and System.err and have it directed to a log.

In the JVMLog class I created you can replace the PrintStream instances used by the JVM's system object. The new PrintStream subclass that I created will allow for any data directed to it to be logged (through an Observer design pattern) before being sent to its final destinations, shown in this short <CFSCRIPT> snippet from Listing 1.

```
<CFSCRIPT>
 log = CreateObject("java", "JVM Log");
log.init("C:\CFUSION\LOG\java.log");
</CFSCRIPT>
```

Naturally this is something you'll have to manually toggle on and off yourself each time the ColdFusion Server is restarted. Toggling the logging mechanism off can also be done with an equally short <CFSCRIPT> snippet from Listing 2.

```
<CFSCRIPT>
 log = CreateObject("java", "JVM Log");
 log.restore();
</CFSCRIPT>
```

This is really no great feat. I merely made the PrintStream subclass static members of JVMLog. The class file caching feature of the JVM did much of the work for me.

JavaCast – Overloading Revisited

In Part 3 I talked about some of the idiosyncrasies surrounding method overloading. In an effort to provide you with more help I spent a little time investigating JavaCast. Unfortunately, I didn't get the results I was expecting. According to the online documentation (*CFML Language Reference*, Chapter 2: "ColdFusion Functions, JavaCast") this function is supposed to allow the developer to overcome the problems with overloaded methods by casting variables into specific and distinguishable data types. Unfortunately, I couldn't get the example in the documentation to work without modification (for starters <CFOBJECT> doesn't appear as a two-part tag anywhere else in the CFML language documentation I've read except in this example, but I digress).

The JavaCast function takes two arguments: Type and Variable. The Type argument specifies the Java data type the CFML variable will be converted into (Boolean, int, long, double, or String); the Variable argument is the CFML variable that's to be converted.

I created a simple class called SignatureCheck (see Listing 3) that contains a method called print with several different signatures, one for each of the accepted types. Then with a template (see Listing 4) I cast a numeric CFML variable to those types. In all cases the calls to the print method of the SignatureCheck succeeded, and in all cases they called the exact same print method, the one with the primitive int as its argument.

I scratched my head for a long time on this one. Reviewing the broken sample in the JavaCast documentation, I noted the order that it listed the signatures. Interestingly enough, when I shuffled the order of the print methods in SignatureCheck, I received some different results. As long as the method that took the String appeared in the source file before the others, my example partially worked. A little more shuffling, recompiling, restarting, and head scratching and I found that getting any two of the numerical signatures to work was beyond me. Whichever one appeared after the String signature in the source file was the one that would get called in all cases. In frustration I even created a signature with a char (which is not one of the listed castable types) only to have a truncated number begin to appear in my browser.

It would seem that despite the documentation to the contrary, JavaCast doesn't work as advertised.

Nested Objects

A common enough practice in OO projects is to have hierarchies of nested objects. Without getting into complicated code listings, we can imagine such a hierarchy as a shelf object that contains zero to many book objects, each containing zero to many pages. It's conceivable then to obtain the name of a book with the following code snippet:

```
shelf.getBook(2).getName();
```

Or the contents of a page with the following:

```
shelf.getBook(2).getPage(1).getText();
```

It could then be expected that to get the page content, a statement like this could be used in ColdFusion:

```
 <CFSET pagecontent =
shelf.getBook(2).getPage(1).getText()>
```

Using this syntax raises an exception. It seems that the current versions of ColdFusion can't navigate nested hierarchies at all. A series of <CFSET> statements is need to drill down, such as:

```
<CFSET book = shelf.getBook(2)>
<CFSET page = book.getPage(1)>
<CFSET pagecontent = page.getText()>
```

This can get rather tedious in deeply nested sets (though using <CFSCRIPT> blocks will cut the typing a little) and, unfortunately, there doesn't seem to be a way around this.

Catching Errors

It's always advisable to use all the error-handling capabilities available to you. The possibility that your application will do something you didn't intend under unforeseeable conditions is high, even with applications that have been thoroughly tested. To this end, Allaire has provided us with some very useful facilities. One of the more powerful ones is the structured exception handling in CFML, which will seem familiar to the Java developer. With <CFTRY> and <CFCATCH> you can intercept troublesome conditions with your Java objects.

Assuming a properly configured JVM, there are essentially two different classifications of errors that can be raised using Java. The first type are errors thrown by the CFML interpreter trying to load Java classes, create objects, or locate methods or properties in a class. The second type consists of exceptions thrown during the execution of a properly loaded and called Java class.

Both kinds of errors are demonstrated in the template (see Listing 5) using a slightly modified version of the Hello class, from Part 3 (see Listing 6). In this template I try to create an object with a misspelling in the Class parameter of <CFOBJECT>, make a method call (with a misspelling), and finally I correctly call a method whose only purpose is to raise an exception. In all three cases I display the entire CFCATCH structure (using an exceptionally useful custom tag by Nathan Dintenfass, <CF_OBJECTDUMP>, which can be found at Allaire's TagGallery), shown in Figure 1.

As you can see, the first two errors look fairly common for CFML errors, though the value of the Type key being "UNKNOWN" is troublesome. It's the third error that you're more likely to encounter. There's actually quite a bit of information to be gleaned here. The Type key has a listed value, "OBJECT"; the Message key holds the fully qualified Java exception class (in this case java.lang.Exception); the Detail key also contains the Java exception class, a colon (if there is a message), the message sent with the raise exception, an additional period, and some additional text informing you that a Java exception occurred. I'm not overly fond of the noise in the Detail key, so within my Java code I put the message text of my exception in square brackets. This tactic allows me to dissect the string and display only the error message I really want to be displayed, as seen in this snippet:

```
#ListGetAt(ListGetAt(CFCATCH.Detail,
1, "]"), 2, "[")#
```

If you're using <CFSCRIPT> for your Java work you'll have to wrap that block inside your <CFTRY> block as <CFSCRIPT> doesn't yet support try/catch blocks.

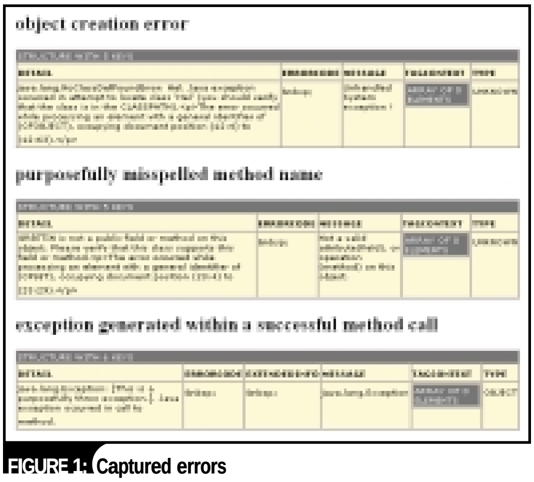
Class Caching

One of the powerful things the ColdFusion Server does for the Java developer is to load the JVM only once. Since the JVM caches a class's bytecode, once the class is loaded all calls to it are relatively quick, thus your templates that are using Java objects should remain responsive.

The downside to this wonderful feature comes in the development phase of your Java components. Every time you update and recompile the source, you have to restart the ColdFusion Server to flush the JVM's class cache from its memory. As tedious as this is, it could almost be considered acceptable in small development efforts. However, providing hot fixes in a testing or staging environment is a completely different matter. Restarting the ColdFusion Server just to flush the JVM could be devastating to days' worth of functional or performance testing.

Keeping it simple, an exemplification of this would be to execute the following:

```
<CFSCRIPT>
  hw = CreateObject("java", "Hello");
  WriteOutput(hw.greeting());
</CFSCRIPT>
```



object creation error

DETAIL	ERRORCODE	MESSAGE	TAGCONTEXT	TYPE
[illegible]	[illegible]	Unhandled System exception !	[illegible]	[illegible]

purposefully misspelled method name

DETAIL	ERRORCODE	MESSAGE	TAGCONTEXT	TYPE
[illegible]	[illegible]	Not a valid [illegible] operation (method) on the object	[illegible]	[illegible]

exception generated within a successful method call

DETAIL	ERRORCODE	EXTENDEDINFO	MESSAGE	TAGCONTEXT	TYPE
[illegible]	[illegible]	[illegible]	java.lang.Exception	[illegible]	OBJECT

FIGURE 1: Captured errors

This will cause the JVM to cache the Hello class shown in Listing 6. First open the source for Hello class in your favored editor, then do a search and replace on the word "Hello," changing it to "Hi." Save, recompile, and execute the snippet again. You should see the same results. Your change won't be picked up until you restart your ColdFusion Server.

I've trolled through the documentation and the forums looking for some hint of a secret Allaire setting that would allow me to restart the JVM or flush its class loader, to no avail. I had some difficulty interpreting the documentation about how to force Java CFX tags to reload (which I'll discuss in a later article in this series), but nothing for regular Java classes. After a little thinking I came up with a simple-to-use solution – I created my own custom class loader wrapped with a class factory design pattern. This class factory (the Debugging-Factory class) stays loaded, as does the custom loader (which I made static within the factory), so a fair measure of performance is still maintained. The benefit is that the factory has a method for flushing the custom loader. What's more, the custom loader filters incoming requests so that the "primordial" class loader still handles all the classes from the default Java packages.

You can load the Debugging-Factory with the following:

```
 <CFSCRIPT>
factory = CreateObject("java",
 "net.rish.coldfusion.
 util.DebuggingFactory");
</CFSCRIPT>
```

Using the factory to load and flush classes from its own internal class loader is simple. Review a snippet from Listing 7 below:

```
<CFSCRIPT>
factory = CreateObject("java",>
"net.rish.coldfusion.
 util.DebuggingFactory");
hw = factory.create("Hello");
WriteOutput(hw.greeting());
</CFSCRIPT>
```

We see the creation of the factory, a call to the factory to create the Hello class (from Listing 6), and the use of the instantiated Hello object. Executing Listing 7 should show you a "Hello, World" in the browser. Pretending that this is unacceptable you can now go to Hello's class source and change the greeting method to "Hi." Recompile the Hello and hit the Refresh button on your browser. What you see is pretty much what you'd expect under normal operation – the new greeting is not displayed. But if you execute the template shown in Listing 8, the DebuggingFactory's flush method is invoked. This will clear the factory's internal class cache. Now for the real test: if you return to Listing 7 and hit the Refresh button, the new greeting should be displayed.

The class factory is generic. One of its shortcomings is that it actually instantiates the requested class. This means you can't use alternate constructors for the objects you're creating. In a real-world scenario, you'd have to implement your own or subclass this one.

Wrapping It Up

Saying that most Java developers would be comfortable with CFML would be a stretch. However, with a little understanding and a few tools, a reasonable degree of functionality can be gotten.

Standing Upon the Shoulders of Others

The JVM logging and object-caching solutions I implemented are based upon ideas from a few different places, such as the Observer and Factory design patterns. I eagerly recommend, as a good investment of time, researching these and other design patterns, which might help you construct better software or divine better solutions.

A good place to start would be with one of the most recognized book on the subject: *Design Patterns: Elements of Reusable Object-Oriented Software*, by Erich Gamma et al (Addison-Wesley). It's not a quick read, nor is it a onetime read. It's the jumping-off point for years of research and application. While it's targeted for folks doing object-oriented programming, I've found it's still applicable to any kind of software development effort.

ABOUT THE AUTHOR

Guy Rish teaches and mentors ColdFusion and object-oriented analysis and design for Andrews Technology, a consulting firm in San Francisco. He holds instructor certifications from Rational Software and Allaire and teaches ColdFusion for SFSU in the Multimedia Studies Program.

GRISH@CNCDSL.COM

Listing 1

```
<html>
<head>
 <title>JVMLog</title>
</head>

<body>

 <cfparam name="logname" default="C:\CFUSION\LOG\jvm.log">

 <cftry>
  <cfscript>
   if(IsDefined("URL.logname"))
   {
    logname = URL.logname;
   }

   if(IsDefined("Form.logname"))
   {
    logname = Form.logname;
   }

   log = CreateObject("java",
    "net.rish.coldfusion.util.JVMLog");
   log.init(logname);
  </cfscript>

  <cfcatch type="Any">
   <p>
   Installation of the JVM log failed.
   <br>
   <cfoutput>
    #CFCATCH.Detail#
   </cfoutput>
   </p>
   <cfabort>
  </cfcatch>
 </cftry>

 <h2>JVMLog Installed</h2>
</body>
</html>
```

Listing 2

```
<html>
 <head>
  <title>JVMLog</title>
 </head>

 <body>

 <cftry>
  <cfscript>
   log = CreateObject("java",
   "net.rish.coldfusion.util.JVMLog");
   log.restore();
  </cfscript>

  <cfcatch type="Any">
  <p>
  Restoration of the JVM standard PrintStreams failed.
  <br>
  <cfoutput>
  #CFCATCH.Details#
  </cfoutput>
  </p>
  <cfabort>
  </cfcatch>
 </cftry>

 <h2>JVM PrintStreams restored.</h2>
 </body>
</html>
```

Listing 3

```
public class SignatureCheck
{
 public SignatureCheck()
 {

 }

  public String print(String arg)
 {
  return(new String("string: " + arg));
 }

 public String print(int arg)
 {
  return(new String("int: " + arg));
 }

 public String print(long l)
 {
  return(new String("long: " + l));
 }

 public String print(double d)
 {
  return(new String("double: " + d));
 }

 public String print(float f)
 {
  return(new String("float: " + f));
 }

 public String print(char c)
 {
  return(new String("char: " + c));
 }

 public String print(boolean b)
 {
 return(new String("boolean: " + b));
 }

}
```

Listing 4

```
<HTML>
 <HEAD>
  <TITLE>Listing 4</TITLE>
 </HEAD>

 <BODY>

 <h1>Signature Check</h1>
 <br>
 <CFSCRIPT>
 sc = CreateObject("java", "SignatureCheck");

 x = 33;

 WriteOutput("<H2>String cast</H2>");
 object_string = JavaCast("String", x);
 WriteOutput(sc.print(object_string));

 WriteOutput("<H2>int cast</H2>");
 primitive_int = JavaCast("int", x);
 WriteOutput(sc.print(primitive_int));

 WriteOutput("<H2>long cast</H2>");
 primitive_long = JavaCast("long", x);
 WriteOutput(sc.print(primitive_long));

 WriteOutput("<H2>double cast</H2>");
 primitive_double = JavaCast("double", x);
 WriteOutput(sc.print(primitive_double));

 WriteOutput("<H2>float cast</H2>");
 primitive_float = JavaCast("float", x);
 WriteOutput(sc.print(primitive_float));

 </CFSCRIPT>

 </BODY>
</HTML>
```

Listing 5

```
<html>
 <head>
  <title>Listing 5</title>
 </head>

 <body>
 <h1>Catching Errors</h1>
 <h2>object creation error</h2>
 <cftry>
  <cfobject type="java" action="create" class="Hello"
   name="hw">

  <cfcatch type="any">
  <cf_objectdump object="#CFCATCH#">
  </cfcatch>
 </cftry>

 <cfobject type="java" action="create" class="Hello"
   name="hw">

 <h2>purposefully misspelled method name</h2>
 <cftry>
  <cfset str = hw.greetin()>
  <cfcatch type="any">
  <cf_objectdump object="#CFCATCH#">
  </cfcatch>
 </cftry>

 <h2>exception generated within a successful method
  call</h2>
 <cftry>
  <cfset str = hw.error()>

  <cfcatch type="object">
  <cf_objectdump object="#CFCATCH#">
  <br>
  <cfoutput>
  #ListGetAt(ListGetAt(CFCATCH.Detail, 1, "]"), 2, "[")#
  </cfoutput>
  </cfcatch>
 </cftry>

 </body>
</html>
```

Listing 6

```
public class Hello
{
 public static final String DEFAULT_STYLE = "Hello";
 protected String m_style = new String("");

 public Hello()
 {
 this(DEFAULT_STYLE);
 }

 public Hello(String style)
 {
 m_style = style;
 }

 public String getStyle()
 {
 return(m_style);
 }

 public void setStyle(String style)
 {
 m_style = style;
 }

 public String greeting()
 {
 return(new String(this.getStyle() + ", World"));
 }

 public String greeting(String whom)
 {
 return(new String(this.getStyle() + ", " + whom));
 }

 public void error()
 throws Exception
 {
 throw new Exception("[This is a purposefully throw excep-
tion.]");
 }
}
```

Listing 7

```
<html>
 <head>
  <title>Listing 7</title>
 </head>

 <body>

  <CFSCRIPT>
  factory = CreateObject("java",
  "net.rish.coldfusion.util.DebuggingFactory");
  hw = factory.create("Hello");
  WriteOutput(hw.greeting());
  </CFSCRIPT>


 </body>
</html>
```

Listing 8

```
<html>
 <head>
  <title>Listing 8</title>
 </head>

 <body>

 <CFSCRIPT>
  factory = CreateObject("java",
  "net.rish.coldfusion.util.DebuggingFactory");
  hw = factory.flush();
 </CFSCRIPT>


 </body>
</html>
```

CODE
LISTINGS

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

Got Game? Unweb Lets You Play with ColdFusion

JOHN MORGAN
gameczar@zbzoom.net

Interview…

with Shawn “Hanzo” Holmes Creator of Unweb

INTERVIEWED BY JOHN MORGAN

Midnight approaches and you’re still battling with your latest Web application. You may feel the urge to kill the darn thing. Well, there’s good news: now you can, thanks to Shawn “Hanzo” Holmes and Unweb (www.iamunweb.com). Playing Unweb enables you to exact your revenge, with extreme prejudice, on those pesky Web pages.

Unweb is an online game that allows you to “fight” and hopefully “kill” a Web page. In the last few months it has taken off like a southern California wildfire. I had a chance to talk to Hanzo and get the developer’s perspective on this unique and addictive new Web pastime.

CFDJ: *In your own words, what is Unweb?*

Hanzo: Unweb is a game in which you create a character, arm it with equipment, and fight monsters. The catch is, the monsters are Web sites.

CFDJ: *Web sites – how do you fight a Web site? Is that anything like fighting with source code?*

Hanzo: In a sense, yes. In Unweb, there are easy Web sites to defeat, and very powerful ones that can take quite a beating and still kill you with ease. The strength of the Web site is determined by the quality of HTML found in the Web page.

CFDJ: *By “quality of HTML,” do you mean “properly formatted HTML”?*

Hanzo: That’s one of the factors, yes. It looks for simple mistakes that could be construed as poor HTML, and starts flagging them accordingly. When the Web page is completely parsed, the flags are counted up and the stat generation begins. There are also a few key features in a Web page that can be found that significantly increase or decrease the stats for that particular Web site, such as the <BLINK> tag.

CFDJ: *I hope <BLINK> would be a negative factor. Do you take into account artistic style or just technical aspects?*

Hanzo: You assume correctly on the <BLINK> tag flag. It’s pretty difficult to programmatically gauge the artistic value of anything, and this is a very simple game at the core, so really, it’s purely a technical algorithm. Unweb has a set of things it looks for, and as it finds fewer and fewer of those particulars in a given Web page, that page’s stats are reduced.

CFDJ: *Why the name Unweb?*

Hanzo: To me, it was a fairly natural title. You’re fighting Web sites, and hopefully defeating them. Since the Web is often compared metaphorically to spiders spinning webs, I thought about the reverse of spinning a web, which would be unraveling or simply unwebbing. Unweb came out of that.

I did think about calling it Deweb, but it sounded too much like Diablo (on which a lot of elements of Unweb are based), so I tossed that idea.

Needless to say, I’ve had a lot of mixed opinions on the title “Unweb.”

CFDJ: *What possessed you to write Unweb – midnight coding and bad Chinese carryout?*

Hanzo: Actually, it was a blown processor. In September of last year I had a few problems with my computer and had to borrow a temporary CPU from a friend. It wasn’t nearly fast enough to allow me to feed my Diablo II hunger at the time, so I started spending the evenings exploring a project I had started to improve my experience with ColdFusion’s Regular Expressions.

CFDJ: *What are the vital statistics for Unweb?*

Hanzo: 122 templates, approximately 7,500 lines of code. The core game took about three months to write, with some further enhancements and improvements added over the next month or so. The database is SQL Server 7.0. As for developers, I did all of the CFML coding myself. I was lucky to have a very talented graphic artist, who helped with images and game design, and I had an excellent database programmer, who came in after Unweb went live, to help improve the performance of the stored procedures. I also worked closely with a friend who helped move the majority of the bottleneck code out into a C++ COM object.

CFDJ: *It sounds like Unweb was an exercise in ColdFusion’s Regular Expressions. Were there any challenges in using ColdFusion?*

Hanzo: Plenty, but regular expressions were far from the real issues at hand. With Unweb I had to write code for an entirely new type of user: a gamer. People weren’t going to be hitting a page and sitting for 30 seconds reading the content; they were going to be hammering it. They were also going to use the Web against me, by trying everything they could to “get out of jail free,” like using the back button to escape a fight they were losing, or hacking in a URL string to try to get free items from the store.

For the most part the challenges in Unweb were unique and unlike anything I had encountered as a Web applications programmer. Writing game code is very different from writing a shopping cart.

During the initial database problems I was having, I went to my DB programmer for help and asked, “Can you write me a stored procedure that returns 10 random records from a table?” He said that he had been a DB programmer for 11 years, and had never been asked that question once.

Of course, he did it and did it wonderfully. That’s just an example of the kind of craziness a project like this entails.

CFDJ: *So Unweb was a very different project from more “serious” work, but no less challenging and time-, er, life-consuming.*

Hanzo: Absolutely, and if anything, it taught me a lot more about the many aspects of ColdFusion that I don’t use in my day-to-day programming, such as <CFHTTP>, the COM object connectivity with <CFOBJECT>, and so on.

It also pushed me to learn how to code CFML much more efficiently. Trust me, I researched every known possibility that would gain me 10ms in template execution time. For the most part, many of the tricks I learned gained me hundreds of ms.

CFDJ: *So the Unweb project was serious fun. Were there any features cut from Unweb because of ColdFusion limitations that you could not overcome?*

Hanzo: The only idea that was dropped near the end was the idea of a helper: a sort of “guide” that gave you tips and told you a bit of the story behind Unweb. I even sat down and had this entire story line written up that subtly and cutely referenced the creation of the Web and HTML, and how there was a great war between these people who followed HTML over SGML and CGI and...well...it became quite the joke near the end. I was even going to have someone read the story to you, via MP3.

Unfortunately, I just looked at it before I was about to start beta testing Unweb publicly and thought it would just come across as either too corny and stupid, or annoying. I didn't want to annoy players away, so I cut the story and guide-person idea.

CFDJ: *Let's talk about Unweb's foundations. I understand you used the FuseBox methodology. Why FuseBox?*

Hanzo: We use FuseBox pretty exclusively where I'm employed. I'm very comfortable and happy with it; I think if you're a CFML developer, there really is no reason to go with any other methodology.

For the record, Unweb is one giant FuseBox with 40 fuses. My next project will push that into separate FuseBoxes for each section of the game.

CFDJ: *In what ways did FuseBox aid in the development of Unweb?*

Hanzo: FuseBox definitely accelerated the development of Unweb. ColdFusion projects, in general terms, lack a lot of solid design processes near their inception, and are often not visually modeled out or documented. There was a lot of that in Unweb; it was a “fix as you go” project. FuseBox lends itself well to that style of project, since you always have your basic structure that works right from the start. All you need to do is add or remove components as you go, and they don't stop the site from working, even if large portions of it are causing problems.

CFDJ: *I have to ask about the “monster” calculation. I know that it's a closely guarded secret, but was it a monster to write?*

Hanzo: No, not at all. It actually does very simple HTML checking and, for the most part, is not tricky at all. Thankfully, because of its simplicity, it was easy to move out into a COM object, since it made up the core of the initial problems that Unweb first started experiencing when it was made live. Since I learned quickly that <CFHTTP> doesn't do all that well under extreme pressure, I had to get some help. Thankfully, we moved the HTTP fetching and parsing out into a .DLL and it has been much more stable since.

> “We use FuseBox pretty exclusively where I'm employed. I'm very comfortable and happy with it; I think if you're a CFML developer, there really is no reason to go with any other methodology.”
>
> *—Shawn ”Hanzo” Holmes*
> *hanzo@gamebasemnt.com*

CFDJ: *It sounds like you're an advocate of mixed-technology development. Has your mix of technology – ColdFusion, FuseBox, COM objects and SQL Server 7.0 – made it difficult to find a host?*

Hanzo: Somewhat. As you can imagine, the COM object poses the greatest risk...while at the same time, it restricts me to a Win32 host. It would have been cool to be able to have the COM object done as an EJB, because then I would've been open to any platform. Still, the COM was a lifesaver. Currently, it's in the process of being moved to a professional, dedicated CF host that's reviewing the COM object. If they give me the thumbs-up, it should be moved shortly and will be performing much better.

I was actually approached at one point by a large, game-themed Web host who liked the idea of Unweb and offered to host it. Once we started getting into the low-level issues of the hosting, however, they backed out after discovering it was written in ColdFusion. They simply weren't interested in picking up a CF license. It's too bad it didn't work out, but there are always other options.

CFDJ: *Unweb has taken on a life of its own. Did you expect and plan for this?*

Hanzo: I had absolutely no idea that it would ever get as popular as it has. I get e-mails daily from people who have just discovered it and have all kinds of questions...and I can actually feel that they're excited about it. That just blows me away. I get e-mail from people in Hungary, Germany, and France...I usually have to translate their e-mails with an online translator just to help them out.

I was surprised when I had 100 users in the database, and, only a month later, I had 100 users playing concurrently. As it stands now, there's about 10,500 users in Unweb, which equates to about 12,000 characters (each player can have several characters). And mostly, its success has spread simply by word of mouth.

The part that I'm most proud of is this little Unweb community that seems to have spawned. When Unweb was first live, and I watched the message board get its first few posts, there would be questions such as “Hanzo, what's this?” or “Hanzo, how do I beat this Web site?”

CFDJ: *Wow. With such rapid growth, have you had any problems with scale?*

Hanzo: A few. When I had my first hundred or so concurrent users, I definitely saw where my initial pitfalls were in the code. And remember, this is not an ordinary site, so we aren't talking several hundred users just reading a news blurb or looking at an image. With Unweb, users act more like an uncapped load-testing tool: they hit the site and hit it hard, clicking submit buttons and links over, and over, and over...as fast as humanly possible. Some users even open multiple windows and start several fights at once.

I also had to deal with “scripters” – players who wrote little macros or “scripts” that played Unweb for them – repeatedly fighting weak Web sites at a blinding rate with no user input needed, which, of course, ended up hammering the Web site into oblivion.

Never mind the fact that I didn't have the most incredible hardware in the world to start with (the Web server ran Unweb and a gaming news site that streamed music to users, and both sites used SQL Server that was – yep, you guessed it – also on the same box). I spent quite a few nights going back over the code, and that's when my real learning began: to discover any and every trick to optimize ColdFusion code, and implement those tricks as fast as possible. I'm confident that the build I have now could handle a much heftier load than the first public version of Unweb, providing I get it successfully moved to a new host.

CFDJ: *What does the future hold for Unweb? Will there be an Unweb II?*

Hanzo: While there are no immediate plans for an Unweb II, per se, I've definitely started developing a new game that will play very similarly to Unweb, but offer much more detail and give the player a lot more stuff to do.

This time around, I'm sitting down and laying out the plans for the game before any code is written. I already have some mock-ups of new ideas, such as player versus player (one of the most requested suggestions for Unweb) in a pseudo real-time environment (still in CFML, mind you), as well as a new layer of management strategy.

CFDJ: *Thank you, Hanzo. What parting words do you have for anyone inspired to create such a unique entertainment experience using ColdFusion?*

Hanzo: It absolutely can be done. Writing Web games is definitely a way to breathe life into Web development with ColdFusion, and if you succeed, you'll never again look at another Web project in the same way.

Caching Redundant **Dynamic Content** with **Custom Tags**

A flexible static page generation technique

BY
JON BLOCK

What do you do when you want to get even better performance out of your most efficient application code?

It can be difficult to keep total page execution time low when processing a complex page. Developers are tasked with finding ways to make apps perform better everyday. By statically storing redundant results of a dynamic page, you can get a CF site to run exponentially faster.

CFHTTP, CFCACHE, and custom tags allow you to create static versions of an entire page, but only custom tags give you the flexibility to cache the results of portions of code, unleashing some awesome possibilities.

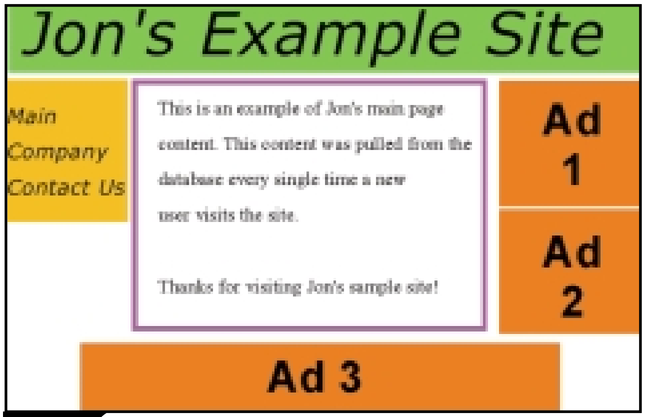
A Common Challenge

If you're like me, you do all your SQL queries with <CFQUERY> when developing a new page. Any CF developer knows this is not the preferred way to do things. We know we should use stored procedures because they're precompiled, so we create stored procedures out of each working <CFQUERY>. Some developers time their code and rewrite it in an effort to shave off milliseconds from their total execution time. Maybe you've tried everything, and you want to go even faster.

This is a situation I've been in. I wanted to make sure that each piece of my project was working at maximum capacity. My site was template-based and it became clear that each time a page was built, ColdFusion Server was doing a whole bunch of unnecessary work.

The Web application had three classic features: a navigation bar that the site administrator could change by logging in; the ability to modify content on the main page of the site; and a rotating ad management system – changes that were all stored in the database (see Figure 1).

The problem was that each time a visitor requested the main page, my application would query for the elements of the navigation bar, query for the main page content, and query for the banner advertising. If my site gets a million requests per day, that's 3 million queries right there. And for what? If the only thing that's changing between page requests is the advertising, why query for the navigation bar and the main page content over and over again? The application is running 2 million redundant queries per day.



This is an example of Jon's main page content. This content was pulled from the database every single time a new user visits the site.

Thanks for visiting Jon's sample site!

FIGURE 1: The basic visual layout with navigation on the left, main content in the center, and banners right and bottom

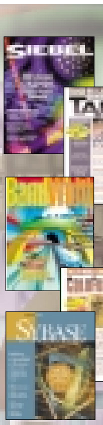
This is the point at which many CF developers would mention cached queries. However, what if you have a cluster of machines and need to run 400 hefty queries? ColdFusion can only cache 100 queries. If you're an ASP, that's not enough.

Creating Static Pages

One way you can maximize your resources is to create static pages based on dynamic ones. Then, you just serve off the static version of your page and ColdFusion Server doesn't need to do any work for each page request. You can use CFHTTP to grab the HTML being served from a dynamic page, and then use CFFILE to write it off as a regular .html file. CFCACHE does the same thing in a single step, as does the procedure of saving the output of a ColdFusion scheduled task, described in Chapter 5 of *Allaire ColdFusion Web Application Server 4.5*, "Administrating ColdFusion Server".

The main problem I had with the CFHTTP method was that by creating a static version of the entire page, the resulting static page could no longer rotate banners or count banner impressions because it was no longer a .cfm file. Even if you end the resulting file with ".cfm", CFHTTP will return the same HTML content that you get when you view the source code of a page in your browser – HTML with all CFML removed.

I needed to find a way to generate partially static main pages. I say "partially" because I needed to maintain my ability to manage dynamic banner ads in ColdFusion,


SIEBEL
TANGO
SYBASE

JAVA
wireless

wireless
JAVA
Multimedia

PREMIER ISSUE
U.S $15.00 (CANADA $18.00)
WebSphere
DEVELOPER'S JOURNAL
Consolidating Legacy Data with WebSphere
written by Brady Flowers
Feature: Programming
EJB: IBM Advanced
Feature: IBM Web
Tutorial: Introduc
XML: WebSphere
Feature: How to Install and
WebSphere & EJB Corner:
Feature: How to Configure WebSphere in Multiplatform Environments

but at the same time, reduce the load on my CF boxes. It didn’t make any sense to query for the navigation bar and the main page content for each page request because that rarely changed – that was the part I wanted to be static. Also, I didn’t want to deal with CFHTTP because it was a bit slow, and I would need to reconfigure my production firewall in order to get CFHTTP requests from my production servers that are directed at my load balancer to work right. The solution was in a special variable called thisTag.generatedContent, available in any closing custom tag.

thisTag.generatedContent

In case you’re not familiar with this custom tag feature, you must first understand that custom tags have execution modes. Most developers call only the START mode without even knowing it. That’s because you’re in START mode when you call a given custom tag like this:

```
<cf_exampleTag>
```

In exampleTag.cfm, you can check what mode you’re in by using a special variable called thisTag.executionMode. You’re in END mode only in closing custom tags, which are called when you pass a slash in the tag, like this:

```
</cf_exampleTag>
```

This allows you to use a CFSWITCH statement or a CFIF conditional statement in your custom tags to do certain things depending on which mode you’re in. You could use the following code construct in the exampleTag.cfm:

```
<CFIF thisTag.executionMode IS
"START">
  <!--- Execute your code for
   start mode here --->
<CFELSEIF thisTag.executionMode
   IS "END">
  <!--- Execute code for end
   mode here --->
</CFIF>
```

Using these modes in conjunction with thisTag.generatedContent, you can grab the results of everything that happens between an opening custom tag and the corresponding closing custom tag. In the next example, I’ll create a navigation bar based on a query:

> “Using CFINCLUDE can mean big problems if you CFINCLUDE any random user’s saved file. Somebody could write very nasty code that could do just about anything to your file system or database. Therefore, only use CFINCLUDE when you trust the author, or you want to be able to create executable CF code from within your application.”
>
> —*Jon Block*

```
<cf_grabTheNavBar>
  <cfoutput query="navBarInfo">
    <a
href="#anchor#">#navText#</a><br
>
  </cfoutput>
</cf_grabTheNavBar>
```

There would be a variable available in </cf_grabTheNavBar>’s END execution mode called thisTag.generatedContent, which contains the result of everything that happened between the <cf_grabTheNavBar> and </cf_grabTheNavBar>. In this case, thisTag.generatedContent contains the resulting HTML content for the simple navigation bar:

```
<a href="main.cfm">Main</a><br>
<a
href="company.cfm">Company</a><b
r>
<a href="contact.cfm">Contact
  Us</a><br>
```

While you’re in <cf_grabTheNavBar>’s END execution mode, you can do some cool things like change what the generated content is. In the next example, I’ll take the generated content and CFFILE WRITE it to a .txt file, then clear the variable so nothing appears on the Web page. This is what my custom tag named grabTheNavBar.cfm would look like:

```
<CFIF thisTag.executionMode IS
"START">
  <!--- I won’t do anything
here. --->
<CFELSEIF thisTag.executionMode
  IS "END">
  <!--- Write the content to a
file. --->
  <CFFILE ACTION="WRITE"
    ACTION="C:\myCFcache\sam
  ple.txt"
OUTPUT="#thisTag.generatedContent
#">
  <!--- Hide from browser dis-
play. --->
  <cfset
thisTag.generatedContent = "">
</CFIF>
```

In this example, I’ve created a text file out of thisTag.generatedContent using CFFILE. Later, I can CFINCLUDE that .txt file on the main page of my site without having to query for it every time the page is requested.

I used this technique to solve my problem. I set up my application to write off the results of the navigation bar and of the main page content when they changed. On my main page, I had two simple CFINCLUDEs that bring in the preprocessed results. This way, I no longer have to waste my resources querying to get the same information for each page build. I have to query for it only once and write it off when the site administrator makes changes. With this simple technique, I got my application to perform five times faster because the database access on my main page was reduced to banner advertisement activity only.

Some other benefits of using thisTag.generatedContent instead of CFHTTP are:

- The data is prepared much faster.
- Your server will not count the static page generation as a page view or banner impression.

- You free up CFHTTP threads for other things.
- CF server doesn't need to be able to access itself. This could be troublesome depending on firewall settings.

Security

Maybe you realize that by using the method described above, I make my main page vulnerable to a form of cross-site scripting. This means that the user responsible for editing site content could write ColdFusion code in the navigation bar or main page content that would be executed. For example, say the user wrote the following sentence in his or her main page content:

```
"ColdFusion is great. My favorite
tag is the <CFABORT> tag."
```

thisTag.generatedContent would write that string to a .txt file via CFFILE, and then it would be included via CFINCLUDE when users browse the site. CFINCLUDE not only includes a file, but executes the included code as well. The <CFABORT> would be evaluated by ColdFusion Server and page processing would subsequently halt.

Using CFINCLUDE can mean big problems if you CFINCLUDE any random user's saved file. Somebody could write very nasty code that could do just about anything to your file system or database. Therefore, only use CFINCLUDE when you trust the author, or you want to be able to create executable CF code from within your application.

If you're concerned about unexpected ColdFusion code, one alternative that seems to work quite nicely is CFFILE READ instead of CFINCLUDE.

```
<CFFILE
  ACTION="READ"

FILE="C:\myCFcache\sample.txt"
  VARIABLE="staticContent">
```

This way, the contents of your .txt file are stored in a ColdFusion variable that you simply CFOUTPUT.

```
<CFOUTPUT>
  #staticContent#
</CFOUTPUT>
```

When ColdFusion code is stored inside the text content of a ColdFusion variable, it's not evaluated. Therefore, the <CFABORT> will never be executed by ColdFusion Server. Instead, it flows into the HTML source and is sent to the browser.

Conclusion

There are lots of ways to optimize your code. It may be worth your while to figure out which parts of your code don't need to be dynamic for every page build. You may not have the need to keep some parts of a page dynamic, so you use the CFHTTP method. However, in ColdFusion 4.x, custom tags provide a faster way to render the complete page, or just certain parts of it. Let's see how Macromedia can make this technique less complex with the upcoming release of ColdFusion Server 5.

ABOUT THE AUTHOR

Jon Block is the lead ColdFusion developer at Members Connect, Inc., in Boston. He is one of the cofounders of the firm's proprietary content management system.

JON@EGRAD.COM

Handling Credit Card Transactions **Using CyberCash**

Create a potentially profitable online storefront

BY
DAVID
KEENER

Many entrepreneurs have set out to make money by selling products, subscriptions, and services on the Web. For many of these small businesses, one of the largest hurdles is implementing a solution that accepts credit card transactions.

With ColdFusion's CFX_CYBERCASH tag, this hurdle can be easily cleared and businesses can be making money on the Web within a short period of time. In this article I explain how credit card transactions work, explore the steps required to implement a solution, and show how ColdFusion can play a vital role in this process.

The Players

Handling a credit card transaction is an operation that requires the cooperation of a variety of organizations. To understand the complexity of the process, you first need to know who the players are:

- ***Merchant***: The first player is the merchant, the organization that wants to sell products online.
- ***Acquiring Financial Institution***: Put simply, this is the bank where the Merchant has set up a merchant account and hopes lots of money will eventually be deposited.
- ***CyberCash***: This organization functions as a gateway. CyberCash facilitates credit card transactions by interacting with other organizations to handle credit card authorizations and captures.
- ***Cardholder Financial Institution***: This organization issues the credit card to the consumer and maintains the account from which funds will be deducted if the transaction succeeds.
- ***Third-Party Processor:*** Banks often outsource various merchant services to a separate company, referred to as a third-party processor. This organization may handle services such as settlement, billing, authorization, and reporting. There may or may not be a third-party processor involved in the transaction process.

Handling credit card transactions may involve up to five separate organizations. While you typically don't have to deal directly with cardholder financial institutions, it's not uncommon to have interactions with the other four when initially implementing credit card functionality.

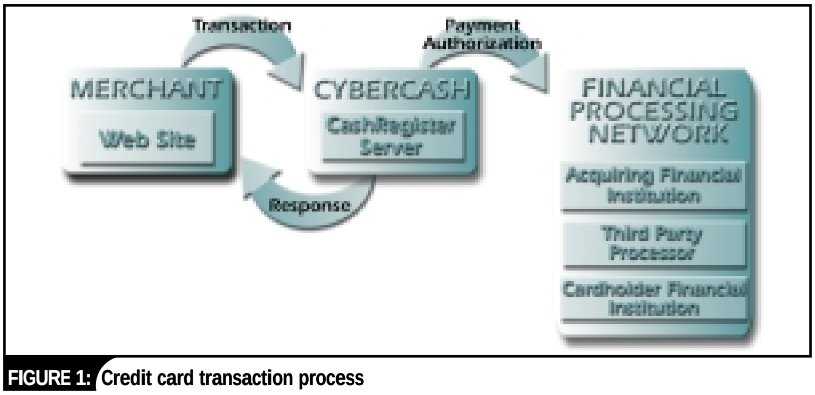


FIGURE 1: Credit card transaction process

How It Works

The credit card transaction process is illustrated in Figure 1. The customer enters all information required for a credit card transaction using a secure form provided by the merchant's Web site. The transaction information is then forwarded to CyberCash's CashRegister server.

CyberCash handles the details, interacting with the cardholder financial institution to determine whether a transaction should be approved or declined. CyberCash passes the response back to the merchant's Web site. If the transaction is approved, the Web site's software can request the "capture of the transaction," also known as settlement.

CyberCash then handles the necessary communication with the acquiring financial institution, the cardholder financial institution and, possibly, a third-party processor to ensure that the money ends up where it belongs.

Handling Credit Card Transactions

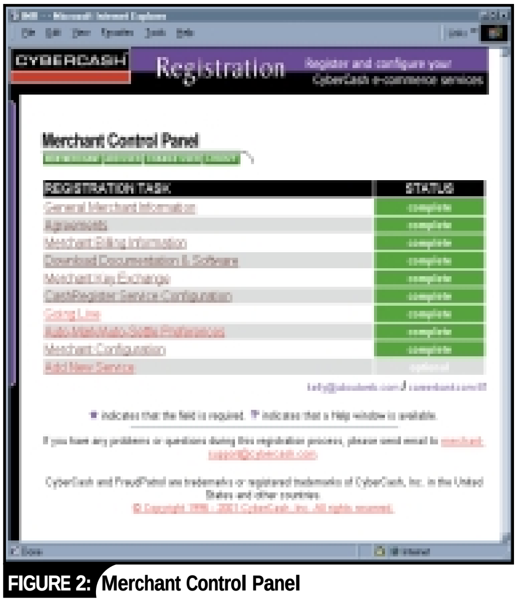
To enable a merchant's Web site to handle credit card transactions the following steps must be taken. Some of these steps are technical and are easily handled by ColdFusion code. Some are more bureaucratic and you simply need to keep trudging forward until each step is finished.

Step 1: Arrange for secure transactions

To handle credit card transactions online, you must provide customers with a secure way to supply confidential information. This translates to the Secure Socket Layer (SSL), a

protocol that allows for the exchange of encrypted information over the Web. The protocol also uses digital signatures to verify the identity of the servers communicating using SSL.

You'll need SSL to accept credit card information from customers and to communicate with CyberCash's CashRegister server.



Merchant Control Panel

REGISTRATION TASK	STATUS
General Merchant Information	complete
Agreements	complete
Merchant Billing Information	complete
Download Documentation & Software	complete
Merchant Key Exchange	complete
CashRegister Service Configuration	complete
Going Live	complete
Auto-Mark/Auto-Settle Preferences	complete
Merchant Configuration	complete
Add New Service	optional

* indicates that the field is required. ? indicates that a Help window is available.

If you have any problems or questions during this registration process, please send email to merchant-support@cybercash.com

CyberCash and FraudPatrol are trademarks or registered trademarks of CyberCash, Inc. in the United States and other countries.

© Copyright 1996 - 2001 CyberCash, Inc. All rights reserved.

FIGURE 2: Merchant Control Panel

Step 2: Get a merchant account

Get a merchant account from a bank if your business doesn't have one already. Indicate up front that you want an "Internet-ready" account that will allow you to accept credit card payments using CyberCash. This shouldn't be a problem since CyberCash works with virtually all banks. In credit card lingo, your bank will be referred to as the *acquiring financial institution*.

Step 3: Register with CyberCash

Go to the CyberCash site (www.CyberCash.com) and register for their CashRegister product. At the time this article was written, the cost of CashRegister was $495 for setup, $20 a month for the service, and 20¢ per credit card transaction. These costs may change over time, but, as you can see, the costs associated with handling credit card transactions are reasonable.

When you register, you'll receive a CyberCash ID, which will be a text string that uniquely identifies your CyberCash account. Write down this string because you'll be using it in your ColdFusion code later on.

Step 4: Perform an online CyberCash setup process

CyberCash will provide you with access to an online system called the Merchant Control Panel (see Figure 2). During the registration process you'll be asked for a user name and password. With this information, you can log in to the Merchant Control Panel at http://amps.CyberCash.com.

This Web site provides a well-organized set of tasks that need to be accomplished before the CyberCash functionality can be used. It guides you through the process of entering required billing information and viewing the necessary CyberCash agreement. This site also provides a central location from which required software can be downloaded. To use CyberCash with ColdFusion, you'll need to download the Merchant Connection Kit (MCK).

You'll need to use the site's "merchant key exchange" link to generate a merchant key to use with CyberCash. This is basically a long alphanumeric string. Write this key down as it will also be needed by the ColdFusion code later on.

This will be as far as you can get with the Merchant Control Panel. The next task shown on the list can only be accomplished by the acquiring financial institution, possibly in conjunction with a third-party processor.

Step 5: Merchant account CyberCash configuration

The acquiring financial institution should put you in contact with the necessary people to get your merchant account configured to interact with CyberCash. This may either be a department within the bank or an external organization to which the bank has outsourced these types of activities. If the activity is outsourced, the external organization will be referred to as the third-party processor.

Whoever you're interacting with, they'll need to know your CyberCash ID and your merchant key in order to get the merchant account configured to interact with CyberCash. Since this step will probably take a while, you can accomplish Step 6 while you're waiting.

Step 6: Install the Merchant Connection Kit

Download the MCK from CyberCash's Merchant Control Panel, then, following their instructions, install it on your Web server.

Step 7: Implementation and testing

Now that CyberCash has been configured, implement the ColdFusion code to use CyberCash and then test it.

At this point, you can execute credit card transactions using fake credit card numbers detailed in the CyberCash documentation. The nice thing is that CyberCash is functional, but no money is being moved.

Step 8: Go live

Once you've thoroughly tested the CyberCash functionality, execute the "Going Live" task in the Merchant Control Panel. As soon as this occurs, your Web site may begin accepting real credit card transactions.

Now, On to the Code!

As it turns out, creating the ColdFusion code to interact with CyberCash is actually one of the easiest steps in the entire process. This is because ColdFusion provides an easy-to-use CyberCash tag that uses the Merchant Control Kit to communicate with CyberCash's CashRegister Server.

Listing 1 provides a simple example of how this tag can be used to send a credit card transaction to CyberCash.

For this example, authorization and capture will be accomplished in one step (this can be set using the Auto-Mark/Auto-Settle area of the Merchant Control Panel). This is appropriate for services, such as online subscriptions, that take effect immediately upon purchase. For products that require shipping, the credit card shouldn't be billed until the product is shipped (merchants can get themselves into regulatory trouble if they bill customers too far in advance of product shipment).

As mentioned earlier, the CyberCash ID and the merchant key are needed by the ColdFusion code. These items are provided as hard-coded arguments to the tag (which is acceptable because neither value is likely to change).

The transaction requires information from the customer, including the customer's name (as it appears on the credit card), the credit card number, and the expiration date (as MMYY). Note that the type of credit card is not required, as CyberCash can determine this from the first few digits of the credit card number.

In addition, the MO_PRICE parameter will record the total cost of the transaction, the amount that will be billed to the customer's credit card. The amount should be preceded by "usd" to indicate U.S. dollars (or some other currency can be designated if required). *Note:* The price should be formatted as a valid amount; for example, 78.456 will be rejected by CyberCash.

Some other parameters can also be provided. The MO_ORDER_ID parameter allows you to record your local order ID as part of the transaction. CyberCash doesn't use this, but the merchant's accounting department will find it useful for cross-referencing CyberCash transactions with their own local order IDs. The MO_PRODUCT_DESCR parameter allows a description of the order to be recorded with the transaction.

The OutputPopQuery parameter allows you to specify where the results of the operation should be recorded. In this case, the results are stored in a structure called *pop*, which can be checked to determine whether or not the transaction succeeded.

If pop.status is not "success," the code presumably results in the display of an error message for the customer. If successful, the code updates a database table to indicate that the order has been processed, and then directs the user to a "thank you" page.

A Few Security Issues

Be very careful if you intend to store credit card information in a database. While interacting with CyberCash using SSL is extremely secure, studies have shown that in the majority of cases where the credit card information of online retailers has been compromised, it's because of inadequate security on the part of the retailer. Under no circumstances should unencrypted credit card information be stored in a database.

There's another potential issue concerning the design of the Web pages associated with a credit card transaction. Usually, the Web pages that facilitate a credit card transaction flow like this:

1. Show the customer an invoice
2. Collect credit card information
3. Confirm credit card information and invoice total
4. Display a thank you page

Depending on the design of your pages, credit card or order information may need to be passed from one page to the next in sequence. If so, they should probably be communicated as session variables, which are highly secure, as opposed to, say, URL parameters (don't store the credit card number in the ORDER table, and then pass the ORDER_ID as a URL parameter).

Conclusion

Credit cards are the enabling technology of the Web economy. For any business to make money on the Web, the ability to handle credit card transactions is an essential ingredient. With ColdFusions's CFX_CYBERCASH tag, any Web site can be converted into a potentially profitable storefront for an online business.

ABOUT THE AUTHOR

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

DKEENER@ABOUTWEB.COM

Listing 1: Sending a Credit Card Transaction

```
<CFSET order_id="345">
<CFSET price="45.78">
<CFSET ccard="1234567812345678">
<CFSET cexpire="0302">
<CFSET cname="John Doe">
<CFX_CYBERCASH
      VERSION="3.2"
      CCPS_HOST="http://cr.cybercash.com/cgi-bin/"
      CYBERCASH_ID="xyzcorp-71"
      MERCHANT_KEY="ya2eNtJiBTDog32pJS7a4yiZI8cPVc"
      MO_ORDER_ID="CB#order_id#"
      MO_PRODUCT_DESCR="Premium Subscription"
      MO_VERSION="3.2.0.2"
      MO_PRICE="usd #numberformat(price, '99999999.00')#"
      CPI_CARD_NUMBER="#ccard#"
      CPI_CARD_EXP="#cexpire#"
      CPI_CARD_NAME="#cname#"
      OutputPOPQuery="pop">
<CFIF pop.status is not "success">
   <CFSET ok=false>
   <CFSET message=pop.error_message>
<CFELSE>
   <!--- Everything ok, complete order --->
   <CFQUERY Name="updateorder"
            Datasource="#DataSource#"
            username="#DBUser#"
            password="#DBPassword#">
            UPDATE transaction
               SET order_processed = 'Y'
             WHERE order_id=#order_id#
   </CFQUERY>
   <CFLOCATION url="thanks.cfm" addtoken="No">
</CFIF>
```

CODE LISTING

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

VTML by Example Part 1 of 2

How to successfully extend the CF Studio IDE

BY
CHRISTIAN
SCHNEIDER

This article demonstrates how to customize and extend ColdFusion Studio's capabilities. Allaire has built lots of nice widgets into the Studio Integrated Development Environment (IDE) to help developers wherever they can. And they did a good job, as you can see.

Starting at the customization level we can define our own color-coding schemes, code templates, keyboard shortcuts, icons, toolbars, and more. Most of this can be customized inside Studio's Options dialog (press F8 to open it directly). At the extension level there are efficient techniques available, including Tag Completion, Tag Insight, Tag Editors, and custom Wizards.

In this article I demonstrate how to extend ColdFusion Studio by using VTML to integrate your custom tags into the Studio IDE.

What are Tag Editors, Tag Insight, and custom Wizards? I suppose most of you working with Studio are using the Tag Insight feature (can be enabled from the Options dialog) quite often while typing HTML and CFML tags. This handy pop-up box lists all possible attributes of the tag you're currently typing so you can choose the attributes you want (see Figure 1).

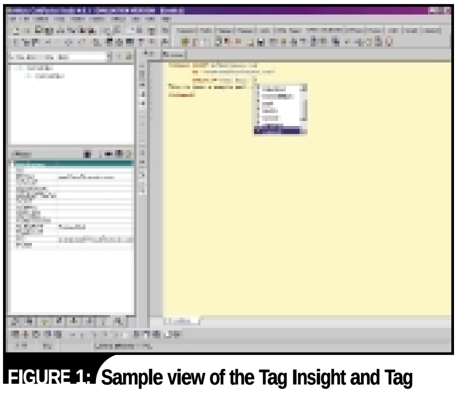


FIGURE 1: Sample view of the Tag Insight and Tag Inspector features in CF Studio

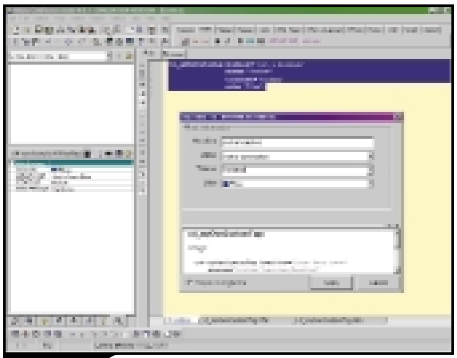


FIGURE 2: Sample view of a Tag Editor dialog in action

In addition, the Tag Insight, available at the left-handed resource tab labeled "Tag Inspector," lists all possible attributes with their values grouped together. Tag Editors are dialog boxes in which you enter tag attributes into a GUI (with some basic validation possible) that can be activated by right-clicking a tag and choosing "Edit tag..." from the context menu (see Figure 2 for a tag editor in action).

Custom Wizards are step-based dialogs that ask information and complete special tasks based on what the user has entered. Put simply, ColdFusion Studio Wizards are a series of dialogs that the user has to go through sequentially that results in a new template being generated. When choosing "New..." from the File menu you have a list of all installed wizards (there are many helpful wizards being shipped with your Studio installation) to choose from (see Figure 3 for the first step of Studio's LDAP wizard). Now let's see how to develop the features that extend CF Studio.

The Visual Tool Markup Language (VTML)

VTML is a family of markup languages used to extend the IDE of HomeSite and ColdFusion Studio. Using VTML you can define your own Tag Insight and Tag Editors, which is very helpful when developing your own custom tags, since you can simply put the VTML files inside your distribution, enabling your custom tags to tightly integrate into the ColdFusion Studio IDE. As the name (Markup Language) might suggest, VTML is a tag-based language (CFML is also tag-based) and therefore easy to learn.

The Wizard Markup Language

WIZML is also a tag-based markup language that extends VTML, making custom Wizards possible through the combination of VTML for the user interface and WIZML under the hood for flow control and input processing. The use of VTML is the main topic of this article's tutorial; how to use WIZML to develop custom Wizards will be the topic of a future article.

Beginning with Easy VTML

Now let's jump into some practical examples of VTML, since this article aims to be an example-based tutorial. The easiest (and also fastest) way to enhance your custom tags with VTML is to use the visual Tag Definitions Editor. As you can see from Figure 4, the visual VTML editor for the Tag Insight can be called from the Resource Tab's Tag Inspector (click on the red-marked icon representing a gear). All you've got to do in the VTML editor to generate a Tag Insight for your custom tag is the following:

1. Click on the "Add Tag..." button and enter the custom tag's name
2. Click the "Add..." button on the right tab to add attributes to your custom tag

3. Define the type of each attribute by choosing the appropriate type from the "Edit Type" drop-down listbox:
   - ***Text:*** For attributes taking strings as their value
   - ***Enumerated:*** For attributes taking one value out of a list of predefined values
   - ***Color:*** For attributes representing a color value
   - ***Font:*** For attributes representing a font definition
   - ***FileName:*** For attributes holding a file name
   - ***FilePath:*** For attributes holding a file path
   - ***Directory:*** For attributes holding a directory name
   - ***RelativePath:*** For attributes holding a relative path
   - ***Style:*** For attributes holding a stylesheet definition
   - ***Flag:*** For attributes acting as flags without having values
   - ***Queryname:*** For ColdFusion recordsets
   - ***Expression:*** For a ColdFusion expression
4. Optionally, you can add categories to organize your attributes into different types (for example, mandatory and optional) under the Attribute Categories tab. The logical grouping of attributes into categories is outlined in the Tag Inspector when you've selected a tag (see the left-sided attribute list in Figure 1).
5. To add documentation to your tag simply enter a filename of an HTML file that documents your tag at the Documentation tab of the Tag Definitions Editor.
6. Finally, click the "Done" button and restart ColdFusion Studio to let the changes take effect.
7. Open a new template and start typing your custom tag. Voilà! Tag Insight is working for your tag.

This editor is a good starting point for getting the basic VTML for the Tag Insight feature done quickly. Starting here, look at the VTML code generated by the Tag Definitions Editor. After using the VTML editor you can find this file on your workstation, depending on where you installed ColdFusion Studio. Just look at the subdirectories of C:\Program Files\Allaire\ColdFusion Studio\Extensions\TagDefs\. As a custom tags developer you can distribute this VTML file along with your custom tag to seamlessly integrate it into CF Studio for developers using your tag. All your clients have to do is copy the custom tag you distributed into their Custom Tag Directory and the .VTM file inside the subdirectories of C:\Program Files\Allaire\ColdFusionStudio\Extensions\ TagDefs\, and restart ColdFusion Studio.

Providing Help Files

To provide your custom tags with the help feature (every CFML tag has a help feature – press F1 when the cursor is selecting a tag), just write the docs in a simple HTML file and name it like your Custom Tag. For example, if your custom tag is <cf_myOwnCustom-Tag>, its source file would be named myOwn-customtag.cfm, the source file for the VTML code would be named cf_myOwnCustomTag.vtm, and the help file would be cf_ myOwn-customtag.htm and placed inside the directory C:\Program Files\ Allaire\ ColdFusionStudio\Extensions\Docs\ CFMLTags\. See Listing 1 for a sample help file.

VTML Syntax

Now let's look at how it was done by analyzing the results that were automatically generated for you. An outlined VTML file has the following structure:

```
<TAG>
<ATTRIBUTES>
... Defines tag attribute properties
and behavior
</ATTRIBUTES>

<ATTRIBCATEGORIES>
... Defines logical grouping for tag
attributes
</ATTRIBCATEGORIES>

<EDITORLAYOUT>
... Defines the layout of a tag editor
</EDITORLAYOUT>

<TAGLAYOUT>
... Defines the tag generation template
</TAGLAYOUT>
<TAGDESCRIPTION>
... HTML-based documentation for the tag
</TAGDESCRIPTION>
</TAG>
```

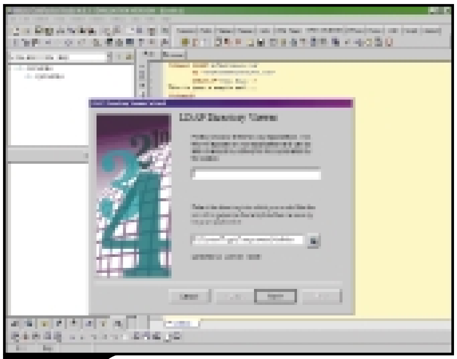


FIGURE 3: A CF-Studio Wizard in action (showing the LDAP wizard)

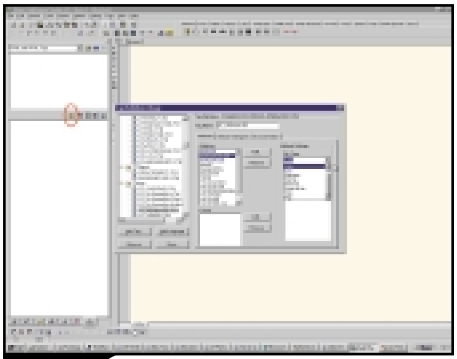


FIGURE 4: The Tag Definitions Editor (being activated from within the Tag Inspector)

The Tag Definitions Editor helps us by generating the <ATTRIBUTES>, <ATTRIBCATEGORIES>, and <TAGDESCRIPTION> sections, which is enough to provide your custom tags with Tag Inspector, Tag Help, and Tag Insight features inside Studio. However, if you wish to have Tag Editor dialogs working for your custom tags you'll have to dig a little deeper into the VTML syntax and code the user interface of the dialogs by hand. You might think it sounds difficult, but it'll be much easier than you might expect.

Part 2 of this article will cover TagEditor dialogs. Meanwhile, you should work with the VTML you've seen in this article in order to get a feeling for how this customization of CF Studio helps developers.

Code appears on page 45

ABOUT THE AUTHOR

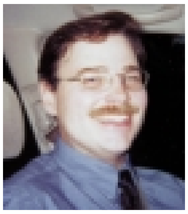
Christian Schneider is an Allaire-certified ColdFusion and Web site developer. He has over four years of intensive experience developing CF-based intranet applications for banks and logistic corporations.

E-MAIL@CHRISTIAN-SCHNEIDER.DE

cfx_kmSuite from Hopkins Technology, LLC

A handy tool belt of CFX tags

REVIEWED BY
CAREY LILLY



Being thrifty, I like to do my own basic car repairs. However, on some of my shade-tree repair jobs, I've run into the problem of not having a specific, but necessary, tool.

I often find myself in the same predicament developing applications for my clients' Web sites. I'm cruising along, happily coding away, when out of the blue I suddenly need to do something I've never had to do before.

ColdFusion, of course, makes it easy when you run into a problem. You just write a custom tag to do the job, throwing in Java or anything else that catches your fancy, and continue on with life. Sometimes, though, deadlines loom and budgets are tight. That's when you find yourself haunting the CF Tag Gallery or Java sites, trying to find that quick-and-dirty (and most especially, free) function to get you back on track. It would save a lot of time if some common functions were collected and ready to go, right?

The cfx_kmSuite from Hopkins Technology, LLC, has a number of useful functions, rolled together into a few CFX tags. If you're self-hosted, colocated, or at least on a dedicated server, you may want to consider this suite. Those of you who (like me) are hosted at national providers may not see this suite, since it includes some potentially hazardous system-access functions.

I suppose to call the cfx_kmSuite *tags* is too simple. It's more like a collection of modules, grouped by function (system, math, string, etc.). Each module will accept parameters to determine the function you'd like to perform.

How to Call Functions

When I first viewed the online documentation, for a moment I didn't comprehend that I was looking at a collection of modules covering 12 subject areas: Calendar, Convert, Format, Graphic, HTML, List, Math, Misc, Query, Security, String, and System.

Take, for example, Convert. These functions convert from English to metric and back. Now I suppose you could write a chunk of code that converts miles to kilometers:

```
<cfset miles = 2>
<cfset kilometers = evaluate( miles *
  1.61)>
```

In cfx_kmSuite, the cfx_kmConvert tag lets you do this, as well as mass, volume, area, and temperature conversions. In the suite, the function could be called as:

```
<cfx_kmConvert V=kilometers F=Length
VALUE=2 FROM=mi TO=km FORMAT="%.2f km">

Result: 3.22 km
```

In the example above, we're doing the following:

1. Using the Convert module
2. Setting the output variable (kilometers)
3. Specifying the conversion function to be performed (length)
4. Setting the value to be converted (2)
5. Specifying that the conversion be from miles to kilometers
6. Specifying the format of the output: (%) = the result, (.2f) = 2 fixed decimal places, (km) = text label for the formatted output. This frees you from the NumberFormat() function.

This Convert example is typical of the functions in the suite, depending on the complexity of the task. It frees you from constantly writing those same little (and a few not-so-little) chunks of code to handle the job. It may even include some you've always wanted.

For instance it may be just as easy to haunt the Tag Gallery for a tag that validates a 16-digit credit-card number. If that's all you're looking for, then this suite is not for you. For serious developers, the suite expands on a few existing CF tags. cfx_kmList, for instance, duplicates the list functions that already exist. Included, however, is the ListBestMatch function, which uses a similarity algorithm to find an item most like a supplied value. Or consider the ListIntersection function, which compares two lists and returns a third showing items found in both (perhaps finding which items in your online store were most popular this month).

VITALS

cfx_kmSuite
Hopkins Technology, LLC

Address: 421 Hazel Lane
Hopkins, MN 55343-7116

Web: www.hoptechno.com/kmtools/

E-mail: kmtools@hoptechno.com

Test Environment:
Windows 98 with Personal Web Server and CF4.5

Pricing: A perpetual license is available for purchase at $250 for one server ($49 each additional at a single physical site) including e-mail and Web support. Once payment clears, we will send download URL and Key.

> "For serious developers, the suite expands on a few existing CF tags. cfx_kmList, for instance, duplicates the list functions that already exist. Included, however, is the ListBestMatch function, which uses a similarity algorithm to find an item most like a supplied value."
>
> —*Carey Lilly*

A Really Neat "Almost" Mini-App

I was doing a Web site for a local art gallery some time ago. They requested a calendar of events, and asked that it be made available ASAP. If cfx_kmSuite had been available, it might have saved some headache.

Included with the suite is an innocuous-sounding function called *BasicMonth*. When I first saw it in the documentation, I thought it was fairly cool: give it a date, and it creates an HTML table for that month. What was even cooler was that you could specify an HREF for each day, thus creating a link to another page (presumably to display an event for that day). Here, half-built and just waiting for a database table and a little code, was a compact events calendar that could be easily inserted into an existing site design or application.

Conclusion

Since these tools are in compiled DLLs, you'll need to have access to your server. As I mentioned at the outset, the suite includes system-access functions that may make some hosts a little leery. Check with your provider and see if they would install this suite.

If you have your own server, rejoice! The tools you need to finish (or perhaps start) the job are close at hand!

ABOUT THE AUTHOR

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

CAREY@WORLDCONTACT.COM

CUSTOM TAGS *code continued from page 43*

Listing 1

```
<!--- /// cf_myOwnCustomTag.htm /// ---->
<!--- Place into
#StudioDirectory#/Extensions/Docs/CFMLTag
s/ --->

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.0 Transitional//EN">

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

<body>

<font face="Arial, Helvetica">
<b>&lt;cf_myOwnCustomTag&gt;</b>

<p/>
Usage:<br>

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

<p/>
Description:<br>

Some sample Custom Tag that does exactly
nothing...


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

</font>

</body>
</html>
```

CODE LISTING

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

SYS-CON MEDIA
135 Chestnut Ridge Rd., Montvale, NJ 07645
Telephone: 201 802-3000 **Fax:** 201 782-9600

president and ceo
Fuat Kircaali fuat@sys-con.com

advertising
senior vp, sales & marketing
Carmen Gonzalez carmen@sys-con.com
vp, sales & marketing
Miles Silverman miles@sys-con.com
advertising director
Robyn Forma robyn@sys-con.com
advertising account manager
Megan Ring megan@sys-con.com
associate sales manager
Carrie Gebert carrie@sys-con.com
associate sales manager
Christine Russell christine@sys-con.com
associate sales manager
Alisa Catalano alisa@sys-con.com

editorial
executive editor
M'lou Pinkham mpinkham@sys-con.com
editor
Nancy Valentine nancy@sys-con.com
editor
Gail Schultz gail@sys-con.com
managing editor
Cheryl Van Sise cheryl@sys-con.com
associate editor
Jamie Matusow jamie@sys-con.com
assistant editor
Gregory Ludwig greg@sys-con.com
editorial intern
Niki Panagopoulos niki@sys-con.com

production
vice president, production
Jim Morgan jim@sys-con.com
art director
Alex Botero alex@sys-con.com
assistant art director
Cathryn Burak cathyb@sys-con.com
assistant art director
Louis F. Cuffari louis@sys-con.com
assistant art director
Richard Silverberg richards@sys-con.com
graphic designer
Abraham Addo abraham@sys-con.com
graphic designer
Aarathi Venkataraman aartathi@sys-con.com

web services
web designer
Stephen Kilmurray stephen@sys-con.com
web designer
Purva Dave purva@sys-con.com
web design intern
Carol Auslander carol@sys-con.com

accounting
chief financial officer
Bruce Kanner bruce@sys-con.com
assistant controller
Judith Calnan judith@sys-con.com
accounts payable
Joan Larose joan@sys-con.com

sys-con events
vice president, events
Cathy Walters cathyw@sys-con.com
sales executive, exhibits
Richard Anderson richard@sys-con.com
sales executive, exhibits
Michael Pesnick michael@sys-con.com

JDJ store manager
Anthony D. Spitzer tony@sys-con.com

Querying Queries

New coding opportunities without new tags or technologies

BY
BEN
FORTA

This month I'd like to continue exploring ColdFusion 5 – this time concentrating on a radical new feature known as *Query of Queries* – a feature that lets you treat query results as if they were database tables. Intrigued? Read on.

Querying Queries

You've probably run into this one before – you retrieve data from a database, and then find that you need the same data again, just sorted differently. Or how about this one – you load an entire product catalog into the APPLICATION scope so as to avoid unnecessary database access, and then need a subset of that data which requires that you make another call to the database. In both examples, you're performing additional database operations to access data that you already have. In fact, the only reason you're going back to the database is to perform sorting or filtering, not because you actually need new data.

With Query of Queries you now have a new way to perform these operations, without having to make additional round-trips to the database server. And the best way to explain it is to look at some code:

```
<!--- Get all movies and expenses --->
<CFQUERY DATASOURCE="ows"
NAME="MovieExpenses">
SELECT MovieTitle, SUM(ExpenseAmount)
AS Expense
FROM Films, Expenses
WHERE Films.FilmID=Expenses.FilmId
GROUP BY MovieTitle
ORDER BY MovieTitle
</CFQUERY>

<!--- Sort by expense --->
<CFQUERY NAME="MoviesByExpense"
DBTYPE="query">
SELECT *
FROM MovieExpenses
ORDER BY Expense
</CFQUERY>
```

The first query is a simple <CFQUERY>. It retrieves a list of movies from a table, along with the total expenses from a second table (using a join and an aggregate function). The results are sorted by MovieTitle, and returned as query MovieExpenses.

The second query is the interesting one. It has a DBTYPE of "query", and no DATASOURCE. When DBTYPE= "query", ColdFusion executes the query against another query, instead of against an actual data source. The SQL statement itself simply retrieves all data from a table named MovieExpenses, and sorts the results by Expense. What is MovieExpenses if there is no DATASOURCE being used? It's the name of the first query – the second query is querying the first query instead of a database.

And all this is being processed by ColdFusion itself, not by a database. The second query is processed in ColdFusion, and ColdFusion returns the results itself.

Reusing Queries

And what about the second example above? This first query loads a movie list into a query in the APPLICATION scope:

```
<!--- Load movies in APPLICATION
scope --->
<CFQUERY NAME="APPLICATION.Movies"
DATASOURCE="ows">
SELECT *
FROM Films
ORDER BY FilmID
</CFQUERY>
```

Now how would you execute a search to find the movies that match user specified search text? Look at this next query:

```
<!--- Find matching movies --->
<CFQUERY NAME="MovieSearch"
DBTYPE="query">
SELECT FilmID, MovieTitle
FROM APPLICATION.Movies
WHERE MovieTitle LIKE
'%#FORM.search#%'
</CFQUERY>
```

Again, the DBTYPE here is "query", and no DATASOURCE is used. The SELECT statement retrieves data from the APPLICATION.Movies query and uses a WHERE clause to extract just the movies matching FORM.search. The result? A brand new query without ever having hit the database.

Working With Multiple Data Sources

Impressed? You should be. But wait, it gets better.

The previous examples all demonstrated code that eliminates unnecessary database calls. And as valuable as that is, this feature really shines in allowing you to perform queries that could not be performed at all with regular queries.

For example, suppose you had data in both Oracle and SQL Server, and wanted to combine the results. How would you do that? Well, prior to ColdFusion 5 you wouldn't have – there really was no way to query multiple data sources. And in ColdFusion 5? Well, look at this code:

```
<!--- Retrieve movies from SQL Server
--->
<CFQUERY NAME="Movies"
DATASOURCE="ows">
SELECT FilmID, MovieTitle
FROM Films
</CFQUERY>

<!--- Retrieve reviews from Oracle --->
<CFQUERY NAME="Reviews"
DATASOURCE="ows2">
SELECT FilmID, Review
FROM FilmReviews
</CFQUERY>

<!--- Combine the results --->
<CFQUERY NAME="combined"
DBTYPE="query">
SELECT MovieTitle, Review
```

```
FROM Movies, Reviews
WHERE Movies.FilmID=Reviews.FilmID
ORDER BY MovieTitle
</CFQUERY>
```

There are three queries used in this operation – the first queries SQL Server, the second queries Oracle, and the third uses DBTYPE="query" to join the two prior queries into a single result set. The result? A combined query derived from data from disparate data sources.

Simple. And you can't do that with straight SQL.

And Not Just Database Queries Either

As you've seen, in ColdFusion 5, queries may be used as tables. And while all the examples used here have involved database queries, in fact, any queries may be used. Consider these possibilities:

- Join a <CFLDAP> query and a <CFPOP> query so as to be able to loop through received e-mail along with information about the sender or the recipient.
- Query a database using <CFQUERY>, and then perform a Verity search using <CFSEARCH>, and join the results to simplify full text searching.
- Retrieve messages with <CFPOP>, and join them to a database query based on sender, recipient, or an ID in the subject field.

Regardless of how queries are created (by any tag, or even manually using the Query functions), ColdFusion can execute queries against them. And like any other queries, Queries of Queries may be cached, put into scopes, and even be queried against, again.

Why Query Queries?

So, why bother querying queries? There are several reasons (and no, individual query performance may not be one of them):

- Joining different data sources
- Executing SQL statements against nondatabase data
- Reducing database load (by executing SQL statements in memory instead of hitting the database server)
- Reducing network traffic (again, by not sending data back and forth unnecessarily)

And as for performance? Don't expect ColdFusion to process queries faster than a database server could. After all, database servers are finely tuned applications designed to do nothing but manipulate data; ColdFusion will never manipulate data as fast as a database server could. But having said that, by helping lower database load and network traffic, querying queries can indeed help overall system performance.

Summary

The ability to query queries is a radical and novel concept – and one that many ColdFusion developers have wanted (often without actually knowing what it is they really wanted). Used properly, this feature opens the doors to all sorts of new coding opportunities, and without having to learn any new tags or technologies at all.

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 WMLScript.

BEN@FORTA.COM

Toward Better **Error Handling** Part 4

CFTRY/CFCATCH

BY CHARLES AREHART

If you've programmed in a modern programming language (Java, C, JavaScript, or even stored procedure languages), you've probably heard of exception handling via try/catch statements. ColdFusion supports the same capability.

If you're new to using them, I'll explain how they're used, and even experienced programmers may learn a thing or two about CF's particular use of try/catch exception handling.

This is the fourth in a series on error handling (see ***CFDJ,*** Vol. 2, issues 10 and 12, and Vol. 3, issue 2). In the three previous installments, we've focused on handling errors at the page level. Something goes wrong, and you want to do something other than have the user see the typical CF error message. That rather cryptic, black-on-white message is great for developers but it's not useful to end users, and we discussed several ways to improve on that.

In this article, we'll move a level further down the "error-handling hierarchy" introduced in Part 2 (see Figure 1). With try/catch handling, or specifically the CFTRY/CFCATCH tags, we're generally designing a way to detect and handle an error that we suspect may happen at runtime, but that we can't know will always happen.

Why a CFTRY?

On a simple level, CFTRY and CFCATCH are used to handle a possible error that might arise in some code. Maybe the code is doing a database interaction that could fail due to integrity errors, or the database is unavailable. Perhaps you're attempting a CFHTTP and the connection to the remote site may fail. Or you're calling a COM or Java object and the object is unavailable.

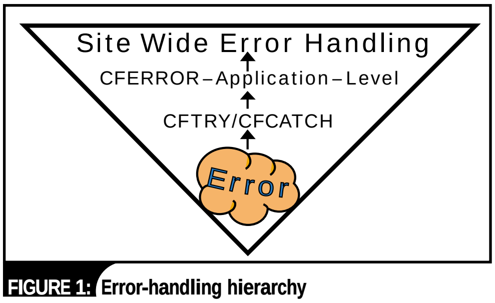


FIGURE 1: Error-handling hierarchy

In these or any circumstance like them, if you know an error may happen, you can anticipate and handle it. Of course, if you've implemented a CFERROR tag for your application (as discussed in the last issue), then it could handle the error, but that will be on a more global scale for the entire page or, indeed, application.

But if some particular tag (or function) or set of them may cause a problem that you can at least contemplate, then the CFTRY/CFCATCH pair will give you much finer control in handling the error. Indeed, handling the error may mean simply ignoring it, as we'll see later.

A Simple Example

CFTRY and CFCATCH work in tandem to:

1. Identify the code to be monitored
2. Describe how to handle any error that arises

The tags are always used together, and neither can be used without the other. The simplest form of using them is:

```
<CFTRY>
 <!--- some code you want to
  monitor--->
 <CFCATCH>
  <!--- code to handle an error
  that arises --->
 </CFCATCH>
</CFTRY>
```

A specific example might be:

```
<CFTRY>
 <CFQUERY
DATASOURCE="#request.dsn#"NAME="test">
  SELECT * FROM MyTable
 </CFQUERY>
 <CFCATCH>
  An Error has occurred while
selecting records from MyTable.<p>
The details of the error are:<br>
<CFOUTPUT>#CFCATCH.MESSAGE#
<p>#CFCATCH.DETAIL#
  <CFABORT>
 </CFCATCH>
</CFTRY>
```

There are a few very important things to note about what goes on within this process.

What Goes on Within the CFTRY/CFCATCH Process

If you're new to TRY/CATCH processing, you should be aware of the following when starting to use this processing. The processing differs depending on whether or not an error occurs in the code being "tried."

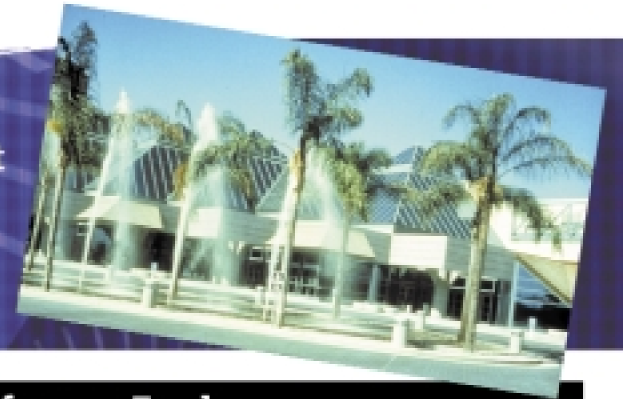
If No Error Occurs

If no error occurs while processing the code within the CFTRY block (the CFQUERY, in our example), then:

1. None of the code in the CFCATCH would be executed
2. Processing would continue with the next statement after the /CFTRY

In that sense, it's like a CFIF that tests for an error code from the code being tried and only does something if there's an error.

That's the really cool thing about TRY/CATCH processing: it's as if we now have return codes to test for CF tags. Of course, some processes (like CFQUERY) do in fact return a code (a database error code, that is), but we've



never had a way to trap that error before Release 4. More on that in a moment. But just keep in mind that if an error does not occur, then processing simply skips the CFCATCH and continues after the /CFTRY.

If An Error Occurs

The more important processing is what takes place if there is an error within the code being tried. In that case:

1. The statements within the CFCATCH (in our simple example) would be executed.
2. The normal CF error message would not be displayed to the end user. You would be responsible for determining what message (if any) to show the user.
3. No error message would be written to the CF error logs.
4. After processing the error handling code within the CFCATCH (when an error has been trapped this way), execution will continue with the next statement following the /CFTRY, which may not always be what you intend. Note that we've used a CFABORT tag in the example above to stop execution.

Also note that within the CFCATCH sample code, we're able to perform any CF tag. We're using a CFOUTPUT to display some information to the user. We could also use a CFMAIL to send ourselves (as developers) some information on the error. We might create a log entry in a database, but be careful not to cause yet another error and, yes, you can nest another try/catch block within a catch. (One of the new features in Release 5, CFLOG, will allow us to programmatically write an entry in the CF application log file.)

CFCATCH Error Variables

Note also that the example shows us referring to some special variables within the CFCATCH, such as CFCATCH.MESSAGE and CFCATCH.DETAIL. In the previous articles, we discussed the ERROR.DIAGNOSTICS variable as well as ERROR.BROWSER, and more. While the "ERROR." variables are not available, there are several specific new CFCATCH variables. They always include at least those shown in Table 1.

TYPE	The exception type being caught.
MESSAGE	The exception's diagnostic message, if any. Else an empty string.
DETAIL	A more detailed message.
TAGCONTENT	The tag stack (if enabled in the administrator).

TABLE 1 Catch variables

There are a few other variables available when specific kinds of errors are being handled, including ErrNumber, LockName, LockOperation, MissingFileName, NativeErrorCode, SQLState, ErrorCode, and ExtendedInfo. The CF documentation (the CFML Language Reference) contains more detail on each of these.

Keep in mind, too, that you have full access to all the other CF variables including CGI.HTTP_ USER_AGENT (in place of ERROR.BROWSER), so the loss of the "ERROR." variables is easily supplemented by the full range of CF variables providing the same information (indeed, the only reason the "ERROR." variables were created was because the original CFERROR handler couldn't process CF tags or variables, as was discussed in the last two articles).

Before leaving the subject of CFCATCH variables, note something about how CF will catch an exception thrown by a Java object (called via CFOBJECT). From the 4.5 New Features document: "ColdFusion checks if the exception thrown is the method exception and stores the classname of the exception in the CFCATCH.MESSAGE variable."

Anticipating Multiple Exceptions

Our simple example presumes that the error-handling routine in the CFCATCH is the only one necessary for the process being "tried." But it's certainly possible that the code may be doing several things, or that the error can have one of a number of causes. There are two ramifications of this:

1. You may have multiple CFCATCH blocks within a single CFTRY.
2. You may distinguish one CFCATCH from another, in that case, using an exception TYPE.

In fact, the Allaire documentation shows a TYPE being provided as if it's required. It's not. There is a default type (called *any*) which if it's not specified, is presumed. But you may trap for any number of error types in a manner such as:

```
<CFCATCH TYPE="Database">
```

or

```
<CFCATCH TYPE="Lock">
```

More types are provided in Table 2.

Application
Database
Template
Security
MissingInclude
Expression
Lock
Custom_type

TABLE 2 Error types

The last item – "custom_type" – literally means any phrase at all. As we'll learn later, your code can "throw" an exception within a CFTRY to be caught by CFCATCH. In such a case, you can choose to create your own "type" for the error-handling mechanism to look for.

Finally, be aware that there's also a possible type of "unknown" for certain exceptions caught by TYPE="ANY". And there are a whole host of types whose names start with "COM.Allaire.ColdFusion" that may be returned under certain conditions.

Again, you don't need to worry about catching a specific type if you're simply trying to catch any error that occurs. Leave the type off. Remember, you can view the type in the available CFCATCH.TYPE variable.

A Change in Processing Multiple CFCATCHes in 4.51

Before leaving the subject of multiple CFCATCHes, you should note that there was a change in behavior as of Release 4.51. The change can be set back to the former behavior via a new CFSETTING parameter. Following is a quote from the Release 4.51 notes:

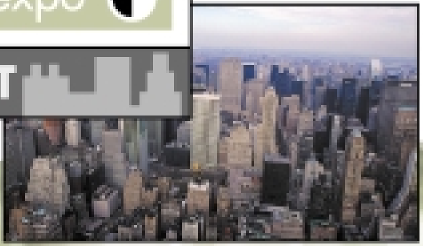
CFCATCH selection logic in ColdFusion 4.5.1 differs slightly

WEB SERVICES INTERNATIONAL
DEVELOPER CONFERENCE & EXPO

Hilton New York, NYC

web services EDGE conference & expo

EAST COAST



Santa Clara Convention Center, CA

web services EDGE conference & expo

WEST COAST

Get Up to Speed with the Fourth Wave in Software Development

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

Demystifying ebXML: A Plain-English Introduction

Authentication, Authorization and Auditing: Securing Web Services

Wireless: Enable Your WAP Projects for Web Services

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

Building Wireless Applications with Web Services

How to Develop and Market Your Web Services

Integrating XML in a Web Services Environment

Real-World UDDI

WSDL: Definitions and Network Endpoints

Implementing SOAP-Compliant Apps

Deploying EJBs in a Web Services Environment

Swing-Compliant Web Services

The First Major Conference & Expo Focusing Exclusively on Web Services

Integrating applications with Web Services

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

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

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

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

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

Plan to Sponsor

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

Plan to Exhibit

Call 201 802-3004 to reserve space

Steve Benfield
Conference Tech Chair

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

EAST
Sept 24–25, 2001

WEST
Oct 24–25, 2001

Owned and Produced by
SYS-CON EVENTS
135 Chestnut Ridge Road
Montvale, NJ 07645
201 802-3069
Fax: 201 782-9601
visit

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

Call **201 802-3004** to reserve your space today!

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

MEDIA SPONSORS: SYS-CON MEDIA, JAVA DEVELOPER'S JOURNAL, XML JOURNAL, wireless BUSINESS & TECHNOLOGY, COLDFUSION Developer's Journal, LINUX BUSINESS WEEK, WebSphere DEVELOPER'S JOURNAL *from ColdFusion 4.0.x. In ColdFusion 4.0.x, the first matching CFCATCH block encountered would be selected to handle an exception. ColdFusion 4.5 scans a CFTRY tag's entire list of CFCATCH blocks to find the closest match. For example, if a CFTRY tag has a CFCATCH TYPE=TEMPLATE block, followed by a CFCATCH TYPE=MISSING-INCLUDE block, ColdFusion 4.0.x will select the TEMPLATE block to handle a MISSINGINCLUDE exception, while ColdFusion 4.5.1 will select the MISSINGINCLUDE block. ColdFusion 4.5.1 can be reset to handle a template using ColdFusion 4.0.x rules by setting the compatibility setting, <cfsetting catchexceptionsbypattern=no>.*

What You Can and Can't Catch

With all this talk about catching errors, it may help to take a moment to clarify what kind of things you can and cannot catch with CFTRY/CFCATCH.

Don't Wrap Entire Program in a Try!

Frequently, folks getting started with this tool presume that they can surround their entire program with a CFTRY to catch any error (or indeed they try to place an opening CFTRY in application.cfm and a closing one in onrequestend.cfm). It won't work, and in fact it's not necessary since that's basically what the new CFERROR TYPE="Exception" (discussed in the last article) does for you. It can catch any exception that occurs in your program that's not otherwise being handled by a CFTRY/CATCH.

PREVENTING SYNTAX ERRORS

You may lament that you can't be warned by an error handler when you've made a syntax mistake. We all make mistakes, right? And who has time to thoroughly test every template, especially when it may be part of a long, multi-page process that can't easily be tested? But take heed: you can (and should) at least confirm that the code will compile, and you don't actually have to run it to find out. CF comes with a syntax checker. It's executed via a page in the CFDOCS directory on the Web server where CF is installed, at http://yourdomain/CFDOCS/cfmlsyntaxcheck/cfmlsyntaxcheck.cfm. It's also reached from the Welcome Page at http://yourdomain/CFDOCS/index.htm

Can't Catch Syntax Errors

Note, though, that I said *catch any exception*. I didn't say it can catch any error. CFTRY can't be used to catch a syntax error. If you think about it, it makes sense. If a syntax error is encountered, the interpreter stops processing right away. It doesn't matter if there's a CFTRY surrounding the code having the error. The interpreter will never begin executing the code.

That stresses the point that CFTRY is for catching "runtime" or execution errors. It can't catch compilation or syntax errors. (Again, CFERROR can help us here, but see the end of the article for more on when it may or may not catch syntax errors.)

Use It to Ignore Array Existence Errors

In a previous ***CFDJ*** article, "Testing Existence in Arrays," (Vol. 3, issue 4), I showed how attempting to refer to an array element that doesn't exist will generate a runtime exception. Unfortunately, there's simply no function to test for such existence (see the article for more on why IsDefined and others simply don't work in this circumstance). As the article demonstrated, you could wrap such array element references in a CFTRY and use CFCATCH to simply ignore the error (code no error message and use no CFABORT, so processing simply proceeds following the /CFTRY).

Use It to Catch Unavoidable Database Errors

I had mentioned previously that one of the neat things you can now do with a CFTRY is determine if a database error has occurred. Say "so long" to users confused about ODBC errors caused by bad SQL being created, databases being locked, or servers being down; now you can catch such an error and give the user a friendlier message.

Be Careful Using It to Catch Avoidable Database Errors

Be careful about this power for trapping database errors. Some have gone a bit too far and have used the capability in a way that wasn't intended and which may be more harmful than useful. For instance, if you're facing a decision in your code about whether to do an insert or an update, don't try an insert and then if it fails, do an update instead.

It may seem that it's saving you from having to do a test to see if the record already exists (in which case an update is the action to perform), but consider that the error causes the database connection to be lost. The time involved to re-create that connection may outweigh the performance gain from avoiding the quick check for the existence of a given primary key value. Just do a "Select keyname from tablename where keyname=value" (as opposed to a more wasteful "select *"), which should execute very quickly. (Often you can program your interface so that your insert/update page is passed information that indicates whether the process is to be an insert or update, with an update being indicated if the primary key is passed from a hidden form field.)

The same consideration about not overusing CFTRY for avoidable database errors applies when doing an insert or update to a record having a column with a uniqueness constraint. Consider a userID column. There should be only one. Whether it's the primary key or has a "unique index" on the column, there can never be more than one record with a given userID. So what can you do as you contemplate inserting or updating that column with a given value, since you don't want to proceed if there's already a record with that value?

The simple solution is to just do a select on that value to see if it already exists and report an error if it does. Some clever programmers wrap the insert or update in a try, and catch and report the uniqueness constraint error if the value already exists (which would violate the uniqueness constraint and cause the insert/update to fail). Their thinking is that database will do the same check for whether the value already exists on the insert/update, so why bother suffering the redundancy of doing it first themselves? (Thanks to Daryl Banttari, senior consultant for Macromedia Consulting, for offering that insight.)

But the cost in the lost DB connection likely outweighs the savings of just checking if the value already exists. Your mileage may vary, but consider that uniqueness is enforced via an index, so the check you do first will be very fast. Plus, while the insert or update will proceed to do another check for that value's existence, consider that the check you've done first will have caused that result to be cached in the database engine by the time the second check is done, making the "redundancy" even less painful.

Some Quirks in Studio

Before concluding, you should take note of some quirks in Studio that may make working with CFTRY and CFCATCH just a little more challenging than it should be. First, Studio doesn't create a closing tag for CFTRY or CFCATCH when you type the opening tag, as it does on all others (assuming you have tag completion enabled). You can add it to the autocompletion list in Studio using Options>Settings> Editor>Tag Completion, adding a new entry for both CFTRY and CFCATCH.

You may also notice that if you look for CFCATCH in the CFML Language Reference's alphabetical listing of tags, you won't find it. It's listed as CFTRY CFCATCH. That may make sense since it's really a subtag to CFTRY, but it's confusing at first. Worse still, pressing F1 on CFCATCH won't produce the expected help for the same reason.

More to Come

I've said in previous articles that this would be the last of four parts, but as I continue the series, I find there's lots more to cover. We still have the interesting CFERROR TYPE="monitor" to discuss. There's also more to talk about regarding intentionally "raising" an error (using CFTHROW), as well as how to provide substantial improvements in error handling when using modular programming (custom tags) with CFRETHROW.

And then there are quirks in the way a sitewide error handler will override a CFERROR TYPE="Request", as well as how a sitewide missing template handler won't catch a missing file in a CFINCLUDE or CFMODULE (you have to code your own CFERROR TYPE="EXCEPTION"EXCEPTION="MISSINGINCLUDE"). There are also quirks when a CFERROR TYPE="Exception" will catch a syntax error.

Still another quirk exists in the aforementioned "CATCHEXCEPTIONSBYPATTERN" process, which doesn't quite work as expected in some situations. When throwing a type of "x.foo", the docs say we should be able to catch it with type *x*, but it doesn't work. We'll get to all that eventually. As always, forewarned is forearmed.

We'll also eventually talk about how to get errors logged to the logs even when being handled by CFERROR or CFTRY (which will be even easier with the new CFLOG tag in CF 5), as well as how to detect and handle a long-running request (perhaps in a different way than other errors), and generally how you might detect and handle different "types" of errors in the CFERROR handlers, now that you know about the differences among such types.

I know some have wondered how I even got four articles out of the subject of error handling. From feedback so far, you agree that there is indeed more to it than many had imagined.

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

CAREHART@SYSTEMANAGE.COM

2001

JDJEDGE conference & expo

With

Cf ColdFusion fasttrack

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

Register NOW for best rates!

The Depth and Breadth of Education to Advance Your Professional Skills

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

Four information-packed JDJEdge Tracks featuring leaders in Java Technologies

Track 1 J2ME – Micro Edition & Wireless
Cutting-edge sessions for software engineers and hardware specialists working on wireless solutions.

Track 2 J2SE – Standard Edition
General Java programming for corporate programmers developing full Java applications, including several introductory sessions.

Track 3 J2EE – Enterprise Edition
Advanced sessions for software architects, Web programmers, corporate developers and consultants developing server-based applications.

Track 4 Working with *I*-Technology
Technical and management sessions for business analysts, corporate systems managers, architects, project managers and CIOs .

JAVA DEVELOPER'S JOURNAL JDJ READERS' CHOICE AWARD

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

Who Should Attend

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

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

JDJEDGE 2001 Features

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

www.sys-con.com/JDJEdge/

Keynote Panel

LYNCH

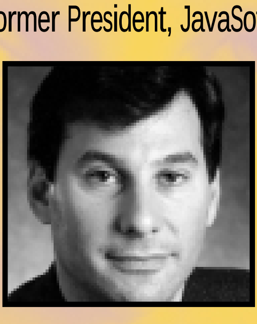
President

Macromedia

BARATZ

CEO, Zaplet, Inc.

Former President, JavaSoft



ROSS

Founder, JavaLobby

Panel Moderator

SMITH

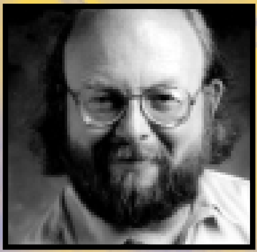
Chief Java Strategist

IBM

DIETZEN

CTO

BEA



GOSLING

Creator of Java

VP & Fellow, Sun Microsystems

SCOTT

CEO, PointBase, Inc.

Cofounder, Oracle

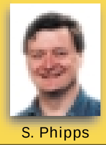
KIESSLING

CEO

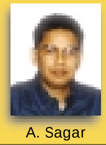
Sitraka

ADVISORY BOARD

J. Milbery

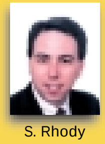


S. Phipps

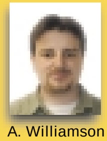


A. Sagar

J. Westra



S. Rhody



A. Williamson

SYS-CON MEDIA

Media Sponsors:

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

Owned & produced by

SYS-CON EVENTS

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

Macromedia **User Conference** 2001

The future looks Cold

BY **DAVID SCHWARTZ**

This past March Macromedia merged with Allaire. I was very excited to attend the Macromedia User Conference, UCON 2001, to see what was new and exciting in the Web-based software world, particularly as it impacted ColdFusion and its developers.

UCON was held at the Hilton New York on Sixth Avenue and 52nd Street in Manhattan, a block away from Rockefeller Center.

It was worth the two-hour battle in traffic to see firsthand demos and hear about new technology that will help shape our future as Web developers. While the conference mainly focused on Macromedia products such as Flash, there were two booths for Allaire products, JRun and ColdFusion. Both booths were manned by Allaire employees who were very familiar with the products and eager to show them off.

ColdFusion – Hotter than Ever

I spent most of my time at the JRun booth, prying as much information as possible from Allaire sales engineer Kevin Hoyt. He was very knowledgeable and accommodating, so I decided to squeeze him for every bit of ColdFusion insight I could get.

Kevin gave me an incredible demo of NEO, the next-generation ColdFusion, which is built on Java. To the ColdFusion developer, NEO's syntax looks basically the same as ColdFusion version 4.5. All the CFML-tag syntax is maintained. The main difference is on the server side. A Java runtime engine is used to process the CFML tags, and, as with the current ColdFusion server, HTML is returned to the browser.

KEVIN LYNCH

So if it looks and acts the same, why change it? Since NEO is based on a Java server, it runs on any platform that supports Java. These days everyone supports Java, from Windows NT to Linux to Sun and so on. As I see it, we can continue to create incredible Web sites for even more servers. What could be bad about that?

Currently, ColdFusion developers find themselves learning and using other languages to add functionality not directly supported by ColdFusion. For example, I use JavaScript to create a windowing interface and interactive form buttons. With NEO, you will have the entire Java language to use, when necessary, to extend your application. Theoretically this provides unlimited potential. Neo is a completely new version – not a patched-up, repackaged release. It's a cleaner, leaner, faster, and more powerful product. Yes, now you really *can* take over the world.

ColdFusion 5.0

Version 5 of our favorite program should be released by the time this article is published. While this isn't as dramatically new and different as NEO, there are still some significant and useful features I found appealing. Most notable are *Graphing* and *Querying a Query*.

The new Graphing feature lets you create charts and graphs based on query data. For example, I could query a sales table for all sales in year 2000 by product, then create a pie chart – directly with ColdFusion tags. This is a great feature for enhancing and waking up traditional tabular reports.

Querying a Query is an interesting addition that has many uses. Basically, it allows you to execute a <cfquery> query and then query the returned result set. Imagine a query that locates all customers in the State of New York. ColdFusion automatically stores the query results in RAM. Now you can further query the customer list to find all customers that live in Brooklyn, NY. Since you can query the first result set, there's no need to "hit" the database again. Simply query the first result set that's already in RAM. It's much faster and more efficient.

JRun

I think that JRun is the ColdFusion equivalent for a Java developer. Like ColdFusion, JRun has an application server and a client-side Studio product. The JRun Studio provides a convenient interface for developing Java applications, much the same way that ColdFusion Studio helps you write CFML programs.

Basically, JRun allows Java developers to use standard Java commands and functions to write 100% J2EE-compatible code. What makes it unique from other J2EE servers are the built-in functions. For example, using a JRun tag, you can query a database and display rows and columns with only a few lines of code (similar to the <CFQUERY> tag in ColdFusion), compared to a page of pure Java code.

JRun excels as a RAD tool for Java. I was very impressed with JRun's ease and approachability. I think beginner Java developers will love it since it eases the learning curve and shortens the time necessary to create finished applications.

The Rest of the World

After loitering at the Allaire booths, I decided to walk the floor and see what other exciting products were available. It seems that ColdFusion developers (and Web site developers in general) are increasingly called upon to do more than just ColdFusion. For example, some dabble in graphic design and layout –

even Web-based animation. Macromedia has become famous for Web-based authoring, graphics, animation, and an overall engaging Web experience. Macromedia estimates that Flash, its vector-based animation program, is installed on 96% of Web desktops.

Here are some products that caught my eye.

- PopChart by Corda (www.corda.com) is a graphing/charting program that's popular among ColdFusion developers. Corda introduced PopChart[D], a new version that's compliant with section 508 of the Rehabilitation Act. Section 508 is a standard for generating Web content that can be used by visually impaired persons. There are an estimated 800,000 visually impaired Web users.
- ElectricRain is a really cool program by Swift3d (www.swift3d.com) that simplifies the creation of 3D text and graphics for use with Macromedia Flash. I saw an impressive demo that automatically generated a 3D object based on a 2D drawing. It looks like a great product for Web-based demos or online stores that want to make products more engaging.

- Sams Publishing (www.samspublishing. com) had a booth that looked like a miniature Barnes & Noble computer book department. They offer several ColdFusion books, including two beginner/getting-started guides. A new version 5.0 manual is on the way. The ColdFusion guides come with CDs of sample applications, demos, and other related software goodies.
- Integration New Media (www.integrationnewmedia.com) debuted GoldenGate Database Connector at UCON. GoldenGate is a database interface for Macromedia Flash and Shockwave. It allows developers to integrate database content with Flash forms. For example, it could be used to display product photos, details, and up-to-date pricing in an online catalog.

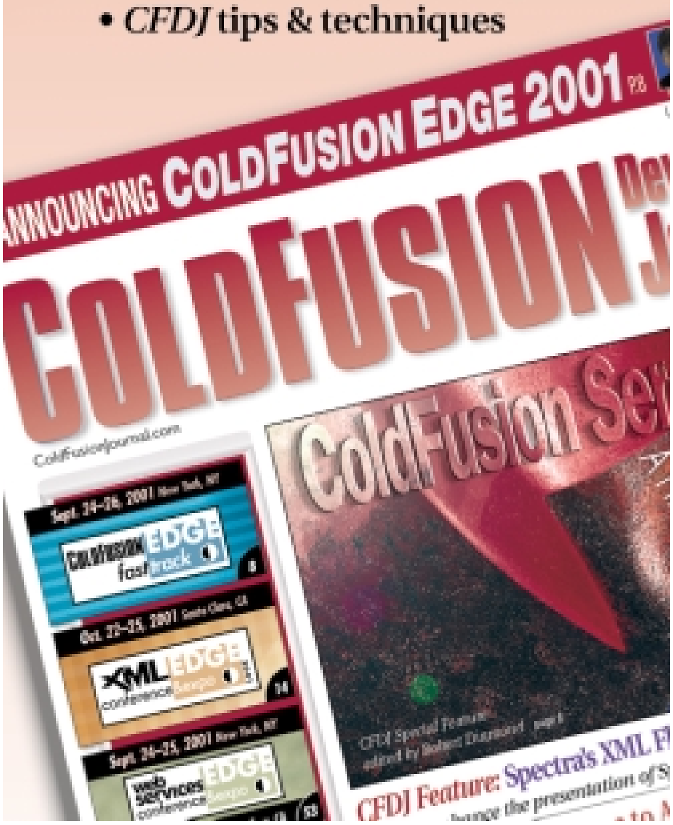
Enough Fun for One Day

We're all familiar with and love Allaire products. Most of us consider ColdFusion to be an integral component of our livelihood. Macromedia creates incredible, interactive products for authoring dynamic, engaging Web sites. I'm excited about the merger and looking forward to seeing the integration of products from these Web powerhouses. I'm confident that UCON 2002 will feature more ColdFusion-related products and enhancements to help us create even better Web sites.

ABOUT THE AUTHOR

David Schwartz is president of Array Software Inc., a software company based in New York City that publishes HotQuery. He has been developing database software for 15 years.

DS@ARRAYONE.COM


ANNOUNCING COLDFUSION EDGE 2001
COLDFUSION
ColdFusion Ser
ColdFusionJournal.com
Sept. 24-26, 2001 New York, NY
ColdFusion EDGE fast track
Oct. 22-25, 2001 Santa Clara, CA
XML EDGE conference
Sept. 24-25, 2001 New York, NY
web services EDGE conference
CFDJ Feature: Spectra's XML F

Ask the **Training Staff**

A source for your CF-related questions

BY
BRUCE
VAN HORN

Thanks again to those of you who have sent in questions or comments about this monthly column. If you'd like to see an archive of previous questions and answers, please visit www.NetsiteDynamics.com/AskCFDJ.

This month we have only three questions to consider, but the response to each is somewhat lengthy. I hope they prove helpful to you!

Q: ***How can I create a site that uses Session variables but not cookies?***

A: Ah, the "cookie-less" Web site! It can certainly be done, but it does take considerably more programming effort than a site that relies on cookies to identify one user from another. What most CF programmers are aware of is that CF looks for the existence of two cookies (CFID and CFTOKEN) to keep track of the Session variables for each user. What many don't know is that if CF doesn't find these two variables in the cookie scope, it looks for them in the URL scope. In other words, if you don't want to use cookies, you can (and have to) pass the values of CFID and CFTOKEN on the URL for every single page request in your site.

One thing you should do if you really want a cookie-less application is tell CF not to create session cookies at all. Do this in your CFAPPLICATION tag in Application.cfm. For example:

```
<CFAPPLICATION NAME="CoffeeValley"
SESSIONMANAGEMENT="Yes"
SETCLIENTCOOKIES="No">
```

Here's where the extra programming effort comes in. You now have to create these two variables and pass them along the URL for every link, form post, or redirect in your site. This isn't difficult; it's just tedious. However, there are two tools you can use to make this easier. The first is the ADDTOKEN attribute of the CFLOCATION tag. If you're doing any page redirections using the <CFLOCATION> tag, write it like this: <CFLOCATION URL=" page2.cfm" ADDTOKEN= "Yes">. This will append CFID and CFTOKEN to the page request, thus maintaining the session information for each user.

For hyperlinks and form posts your code should make use of the relatively unknown Session.URLTOKEN variable, a single variable the CF server creates that contains the equivalent of both CFID and CFTOKEN as URL variables. For example, the two links shown in Listing 1 would yield the same result, but the first requires much less coding on your part. For form posts use the code in Listing 2.

Q: ***I use a third-party hosting company to host my CF application and an MS Access database for my data source. My problem is that I sometimes need to upload a new database file to the server, but their server keeps my database file locked, preventing me from replacing it with the new file. Is there a way around this?***

A: Yes! This is a common problem encountered by people using hosting companies. The problem (which is actually a good thing as far as the performance of your site is concerned) is that the data source created in the CF Administrator has been set to "Maintain database connections." With this enabled, CF doesn't have to reconnect to your database every time someone wants information from it, thus making all your queries faster. Unfortunately for you, CF keeps that file locked as long as the connection is active. What you need is a way to force CF to release its connection. There are two ways you can do this remotely.

The first method may not be available to you if your hosting company has disabled the use of certain "undocumented" functions. Try the following to see if it works. Upload and run a .cfm file with this one line of code in it: <CFSET tmp = CFUSION_DBCONNECTIONS_FLUSH()>. This function, CFUSION_DBCONNECTIONS_FLUSH(), will cause CF to release all existing database connections. Out of respect for your hosting company, I wouldn't recommend using this method since it causes CF to release *all* database connections.

Before that function was discovered, I used a much safer method that causes CF to release only your database. It's the behavior of CF to drop a database connection if the query passed to it causes an ODBC error. Therefore, all you need to do is upload and execute a file with a query in it that will always cause an error:

```
<CFQUERY NAME="KillConnection" DATA-
SOURCE="FastTrack_Lab">
Select * From NoTable
</CFQUERY>
```

Since there's no table called "NoTable" in my database, CF will receive an ODBC error and release its connection to that file. You can now upload a new file to replace it.

***Q:** I have an application that uses MS Access so I can't write any "triggers" as you described in the Advanced ColdFusion class. What's the best way to retrieve the ID number of a record that has just been inserted into the database?*

A: This is a very common question. When you insert a new record into the database, you often need to know the primary key value the database automatically assigned to that record. As you've mentioned, the best way to do this is through a trigger. Unfortunately, as you've stated, Access doesn't support triggers. The traditional way of doing this is to run a query that will retrieve the highest number (using the SQL Max() function) in the primary key column (see Listing 3). Since the database assigns these number sequentially, Max() will usually retrieve the record you just inserted.

However, this will only work if you never have more than one user inserting records into the database. In a multiuser environment, it's easy to see that this could retrieve the wrong record. There are at least two ways to prevent this. The easiest is to ensure that no more than one user ever executes this code at the same time. To do this add a CFLOCK around the entire "transaction" (see Listing 4). Assuming that you don't insert records from any other template in your application, CFLOCK will ensure that only one user at a time will execute these two queries.

Another alternative is to insert in the database an absolutely unique value that can always be retrieved to identify that record (in addition to the primary key). You could add a column to your database table called "UUID" (it should be a text column that allows 35 characters). Then, have CF generate a Universally Unique Identifier (UUID) using the CreateUUID() function. Insert that value, then retrieve it (see Listing 5). While this increases the size of your database and can take longer to retrieve, it should never fail regardless of the number of simultaneous users inserting records.

• • •

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

Code appears on page 60

ABOUT THE AUTHOR

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

BRUCE@NETSITEDYNAMICS.COM

code continued from page 59

Listing 1

```
<CFOUTPUT>
 <A HREF="index.cfm?#Session.URLToken#">Home Page</A><BR>
 <A HREF="index.cfm?CFID=#Session.CFID#&CFTOKEN=#Session.
  CFTOKEN#">Home Page</A>
</CFOUTPUT>
```

Listing 2

```
<CFOUTPUT>
 <FORM ACTION="form_action.cfm?#Session.URLToken#"
  METHOD="post">
</CFOUTPUT>
```

Listing 3

```
<CFQUERY DATASOURCE="Coffee_Valley">
   INSERT INTO Beans (Bean_Name,Bean_Description)
   VALUES ('#Form.Bean_Name#','#Form.Bean_Description#')
</CFQUERY>
<CFQUERY NAME="qNewID" DATASOURCE="Coffee_Valley">
   SELECT MAX(Bean_ID) AS NewID
   FROM   Beans
</CFQUERY>
```

Listing 4

```
<CFLOCK TIMEOUT="10" NAME="AddNewBean" TYPE="EXCLUSIVE">
 <CFQUERY DATASOURCE="Coffee_Valley">
  INSERT INTO Beans (Bean_Name,Bean_Description)
    VALUES ('#Form.Bean_Name#','#Form.Bean_Description#')
 </CFQUERY>
   <CFQUERY NAME="qNewID" DATASOURCE="Coffee_Valley">
        SELECT MAX(Bean_ID) AS NewID
        FROM   Beans
   </CFQUERY>
</CFLOCK>
```

Listing 5

```
<CFSET UUID = CreateUUID()>

<CFQUERY DATASOURCE="Coffee_Valley">
   INSERT INTO Beans (Bean_Name,Bean_Description,UUID)
   VALUES   ('#Form.Bean_Name#','#Form.Bean_Description#',
 '#Variables.UUID#')
</CFQUERY>
<CFQUERY NAME="qNewID" DATASOURCE="Coffee_Valley">
   SELECT Bean_ID
   WHERE  UUID = '#Variables.UUID#'
   FROM   Beans
</CFQUERY>
```

CODE LISTING

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

<the world's leading independent coldfusion magazine>

ColdFusion Developer's Journal

home advertise subscribe contact SYS-CON MEDIA

< dot.com >

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

< magazine >

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

< content >

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

< conferences >

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

< search cfdj >

Search

< bestsellers >

1. VisualCafe Standard Edition v4.0 from Web Gain $97.99
2. Intr@Vision Foundation from Galileo Development Systems $3499.99
3. Goetz's Net Tools Pro Kit from Lawrence Goetz $19.99
4. JRun Server v3.0 Enterprise 1CPU License from Macromedia $4,872.99
5. Homesite v4.5 Win 32 from Macromedia $93.99

What's Online

www.sys-con.com/coldfusion

Digital Edition

For access to the most current ColdFusion information, connect to the Internet and navigate to ***ColdFusion Developer's Journal*** anywhere, anytime. Don't have your print edition on hand? Worried that you're missing out on what you need? Move at the speed of technology, with the ***CFDJ*** digital edition.

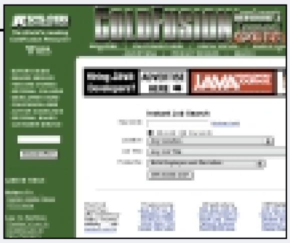
ColdFusion Edge 2001, New York September 23–26, 2001

Register early to make sure you have a seat in New York this fall at ColdFusion Edge 2001.

Join us for keynote speeches by Kevin Lynch, president of Macromedia, and Jeremy Allaire, CTO of Macromedia and creator of ColdFusion.

Check out www.sys-con.com/coldfusionedge daily for updates and conference news, and register to be a part of the experience. This is a fast track for ColdFusion certification you won't want to miss.

Search ColdFusion Jobs

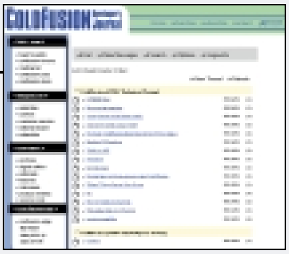


ColdFusion Developer's Journal is proud to offer our employment portal for IT professionals. Get direct access to the best companies in the nation. Find out about the "hidden job market" and how you can find it. As an IT professional curious about the market, this is the site you've been looking for.

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

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

ColdFusion Forum

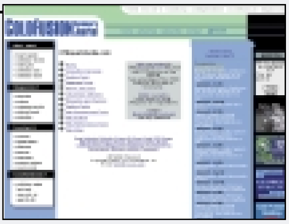


Join ColdFusion List, the new ColdFusion community. Join other IT professionals, industry gurus, and ColdFusion writers for ColdFusion discussions, technical questions, and more.

Voice your opinions and assessments on topical issues or hear what others have to say.

Join the ColdFusion List now to monitor the pulse of the ColdFusion industry.

ColdFusion Buyer's Guide



You've just found your best source on the Web for ColdFusion-related software and products in more than 20 mission-critical categories, including books, consulting services, custom tags, testing tools, Web hosting, and much more.

Check out the Buyer's Guide for the latest and best ColdFusion technologies available today.

QUICK POLL

Will You Upgrade to ColdFusion 5.0?

○ A- Yes
○ B- No

Vote

Results

CFDJnews

Backsoft Generation Released for Allaire ColdFusion Platform

(Sarasota, FL) – Backsoft Corporation, a global provider of e-business software, introduced the latest release of its e-business application framework, Backsoft Generation, for the Allaire ColdFusion platform. Backsoft Generation is designed to develop, deploy, and manage all the Web applications and information that comprise a company's e-business in a structured and coordinated environment.
www.backsoft.com

CFM-Resources.Com Offers Free Tutorials for ColdFusion Users

(Miami, FL) – CFM-Resources.Com, Corp., is lending a helping hand to the increasing number of programmers, designers, and business owners turning to ColdFusion as their primary Web site authoring language.

www.CFM-Resources.com provides all levels of Web developers with everything they need from tutorials and forums to interactive tech support. The site also provides members with free e-mail and hosting, and offers affordable hosting for more advanced sites.
www.CFM-Resources.com

Ektron Releases eWebEditPro 2.0

(Amherst, NH) – Ektron, Inc., has released version 2.0 of eWebEditPro, a multilanguage, browser-based, WYSIWYG Web content authoring and editing tool.

Major updates include simplified installation/integration, enhanced IT-side controls, new features for customization, and new content formatting functionality. eWebEditPro works with all Web application servers and ships with sample code for Microsoft Active Server Pages (ASP), Macromedia ColdFusion, JSP, PHP, and Lasso.
www.ektron.com

Macromedia ColdFusion Powers Internet Marketplace Half.com

(San Francisco, CA) – Macromedia, Inc., announced that Half.com, has implemented Macromedia ColdFusion, to power its interactive e-commerce site.

Leveraging Macromedia ColdFusion, Half.com initially built its entire Web site in 1999 in less than six months. ColdFusion's rapid deployment and cross-platform compatibility also enables the Linux-based site to coexist within its Java-based application framework. With the help of ColdFusion, Half.com recently expanded its online presence to meet the growing demand for new product offerings by adding computer equipment, sporting goods, trading cards, and consumer goods to its catalog.

ADVERTISER INDEX

ADVERTISER	URL	PH	PG
ABLECOMMERCE	WWW.ABLECOMMERCE.COM	360.253.4142	2
ACTIVEPDF	WWW.ACTIVEPDF.COM	888.389.1653	3
AMERICAN CYBERNETICS	WWW.MULTIEDIT.COM	800.899.0100	29
CFDYNAMICS	WWW.CFDYNAMICS.COM	800.422.7957	47
CFXHOSTING	WWW.CFXHOSTING.COM	866.CFX-HOST	25
CFXTRAS	WWW.CFXTRAS.COM	704.408.6186	45
CODECHARGE	WWW.CODECHARGE.COM	650.754.9810	21
COLDFUSIONEDGE 2001	WWW.SYS-CON.COM/COLDFUSIONEDGE	201.802.3069	8-9
CONCEPTWARE AG	WWW.CONCEPTWARE.COM/EYE	011.781.275.6171	4
CORDA TECHNOLOGIES	WWW.CORDA.COM	801.805.9400	11
EVOLUTIONB	WWW.EVOLUTIONB.COM/WDJA	877.622.7551	59
HOSTCENTRIC	WWW.HOSTCENTRIC.COM/CFDJ	888.932.9376	19
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM	877.215.HOST	33
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800.379.7729	60
JAVAEDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/JAVAEDGE	201.802.3069	34-35
JAVAONE	HTTP://JAVA.SUN.COM/JAVAONE/	888.886.8769	48-49
MACROMEDIA	WWW.VUE.COM/ALLAIRE	877.460.8679	39
MACROMEDIA	WWW.MACROMEDIA.COM/TRAINING	888.939.2545	31
MACROMEDIA	WWW.ALLAIRE.COM/USERGROUPS		43
PACIFIC ONLINE	WWW.PACONLINE.NET	877.503.9870	47
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	23
SYS-CON MEDIA REPRINTS	WWW.SYS-CON.COM	201.802.3024	56
WEBSERVICESEDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/ WEBSERVICESEDGE	201.802.3069	53
XMLEDGE CONFERENCE & EXPO	WWW.SYS-CON.COM/XMLEDGE	201.802.3069	14-15

Next Month...

Here's a sneak peek...

Optimal Development Environment: Working Together in CF *by Sarge Sargent*

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

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

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

ColdFusion Developer's Journal

Don't miss the July issue!