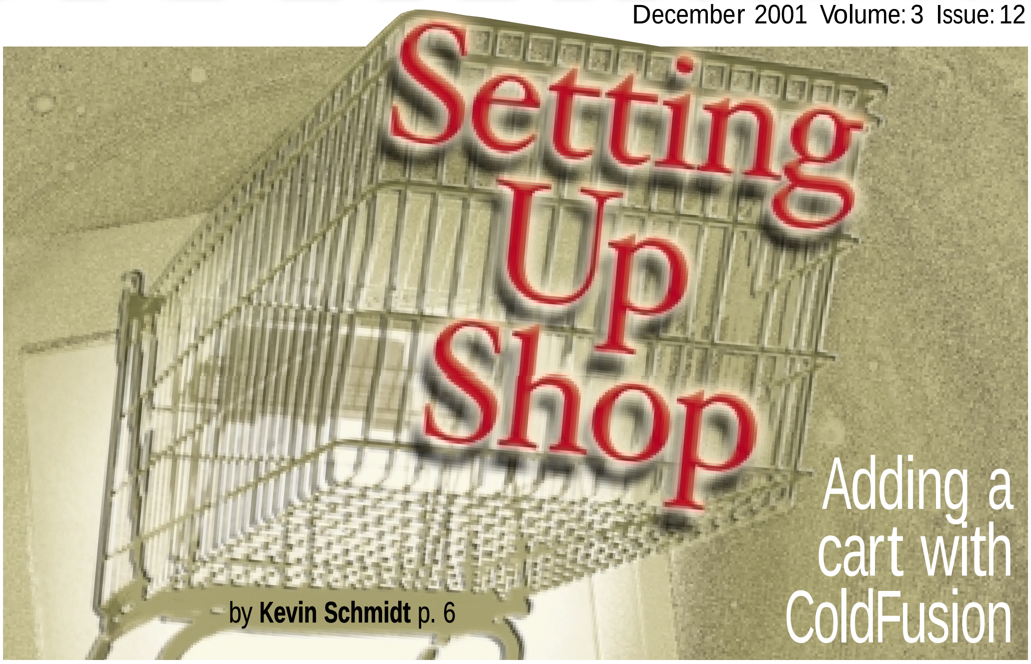
HIDDEN GEMS to Make Your Life Easier

U.S. $8.99 (Canada $9.99)

ColdFusion Developer's Journal

ColdFusionJournal.com

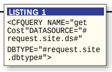
December 2001 Volume: 3 Issue: 12



Setting Up Shop

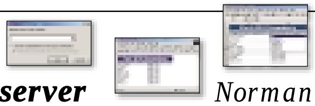
Adding a cart with ColdFusion

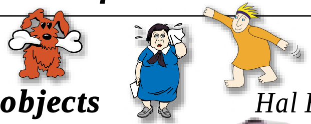
by **Kevin Schmidt** p. 6

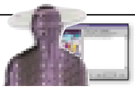


LISTING 1

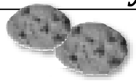
```
<CFQUERY NAME="get
Cost"DATASOURCE="#
request.site.ds#"
DBTYPE="#request.site
.dbtype#">
```

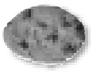






VITALS
Code Co-op
Reliable Software
Address: 1011 Boren Ave., PMB 206
Seattle, WA 98104
Web: www.relisoft.com
E-mail: info@relisoft.com
Test System:
AMD K6 266MHz PC 4.0





SYS-CON MEDIA

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
associate editor
Jamie Matusow jamie@sys-con.com
associate editor
Gail Schultz gail@sys-con.com
assistant editor
Lin Goetz lin@sys-con.com
product review editor
Tom Taulli
tips & techniques editor
Matt Newberry

writers in this issue
Charles Arehart, Robert Diamond, Norman Elton,
Ben Forta, Hal Helms, Eva Holtsmark, Ian Rutherford,
Kevin Schmidt, Christian Schneider,
David Schwartz, Bruce Van Horn

subscriptions:
For subscription requests please call
1 800 513-7111 or go to: **www.sys-con.com**
cover price $8.99/issue
domestic $89.99/yr. (12 issues)
canada/mexico $99.99/yr
overseas $129.99/yr
back issues $12 U.S. $15 all other

editorial offices: SYS-CON MEDIA, INC.
135 Chestnut Ridge Rd., Montvale, NJ 07645
Telephone: 201 802-3000 **Fax:** 201 782-9600
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:
Carrie Gebert carrieg@sys-con.com

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

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

<CF News>

BY **ROBERT DIAMOND**

Over the past few months we've previewed the development of ColdFusion 6.0 – affectionately codenamed "Neo" by the Macromedia development crew. In many of those articles we also discussed the new Java support, one of the brightest new features and also the one that could help carry ColdFusion to its next level of acceptance and support. At DevCon, Macromedia brought that promise a bit closer to reality with announcements about Neo and its brand-new partner support.

Numerous industry leaders, such as BEA, IBM, Intel, and Sun Microsystems, are joining with Macromedia to support Neo. The new CF is being developed on top of JRun and has the CFML command interpreter on top of a Java engine. Java, with its write once, run anywhere technology, will allow ColdFusion to run on top of any Java application server. Current plans, though subject to change as with any new product release, allow it to be bundled with JRun or purchased separately for installation on such Java application servers as BEA's WebLogic or IBM's WebSphere.

Integrating CF into existing enterprise-level Java applications will soon be a piece of cake, and running CF apps alongside Java ones will be a walk in the park. This is all without changing the basic ColdFusion Markup Language programming environment. Exciting times are ahead.

In addition, the new ColdFusion architecture will support interoperability with Microsoft .NET platform technologies through support for XML Web services and .NET servers. Macromedia outlined a broad technology strategy for ColdFusion to run on multiple application servers, and through this to support XML Web services and interoperate with Microsoft .NET technologies.

I was unable to attend Macromedia's DevCon this year due to a scheduling conflict – I was attending Web Services Edge 2001 West in Santa Clara, California. While covering all the Web services, XML, and .NET sessions there and walking the exhibit floor, I frequently thought of all of you in Disney World – having fun, networking, and talking ColdFusion.

I'm happy to report that we'll be expanding our Web services coverage in ***CFDJ***, which will be a great help the next time two industry conferences conflict with one another. Web services as a whole is slowly making the climb from vaporware and wild sets of standards and claims to real-world examples, and CF is along for the ride.

In this month's issue we have our usual great lineup. Norman Elton writes on using Microsoft Excel and ColdFusion together. While perhaps not the ideal method for storing your data in a Web database, it certainly comes in handy. Christian Schneider presents Part 3 in his series on customizing the CF Studio IDE.

We have two conference reviews – the TeraTech ColdFusion Conference by Eva Holtsmark and Macromedia DevCon by Kevin Schmidt. David Schwartz reviews Code Co-op from Reliable Software, which passes his tests with flying colors. Ian Rutherford discusses stored procedures, and Kevin Schmidt goes under the hood of CF-based shopping carts. We also have our regular columnists – Charlie Arehart, Hal Helms, Ben Forta, and Bruce Van Horn – who provide great insights into the world of ColdFusion.

Robert Diamond

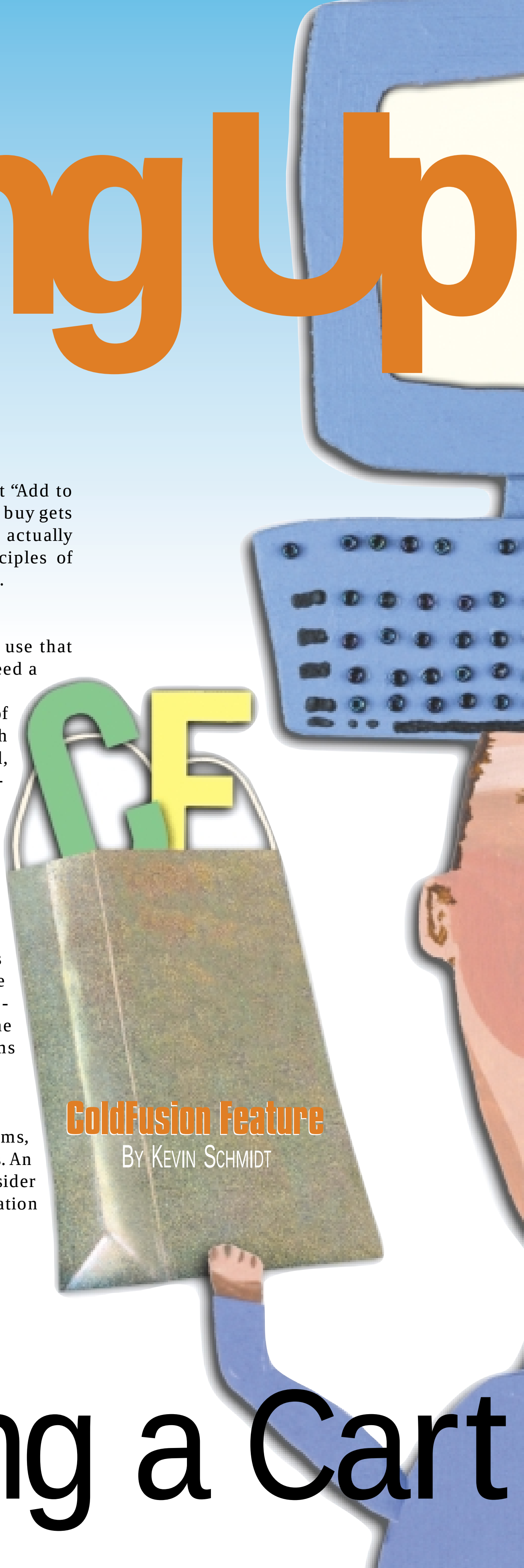
@ ROBERT@SYS-CON.COM

ABOUT THE AUTHOR
Robert Diamond is editor-in-chief of **ColdFusion Developer's Journal** *as well as* **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 with a BS in information management and technology.*

Setting Up Adding a Cart

ColdFusion Feature

By Kevin Schmidt



Ever wonder what's going on behind the scenes when you click that "Add to Cart" button on your favorite Web site? Sure, whatever you want to buy gets dropped in your cart, but what actually makes it work? There are actually several answers here, so let's go step-by-step through the principles of designing and developing a simple shopping cart system with ColdFusion.

The Shopping Cart

Say you're ready to sell on the Web and you want people to be able to use that handy little button "Add to Cart" to purchase your items. To do this, you need a shopping cart, and what, exactly, does this entail?

Several pieces must be in place for the cart to function properly: a list of the items you want to sell containing all the pertinent information about each item; a system to view the items currently in the cart and the ability to add, remove, or update them; a checkout area; and, finally, a transaction processing service. I'll cover the first three in some detail; the fourth, however, is a topic in and of itself and will be touched on only briefly.

The Items

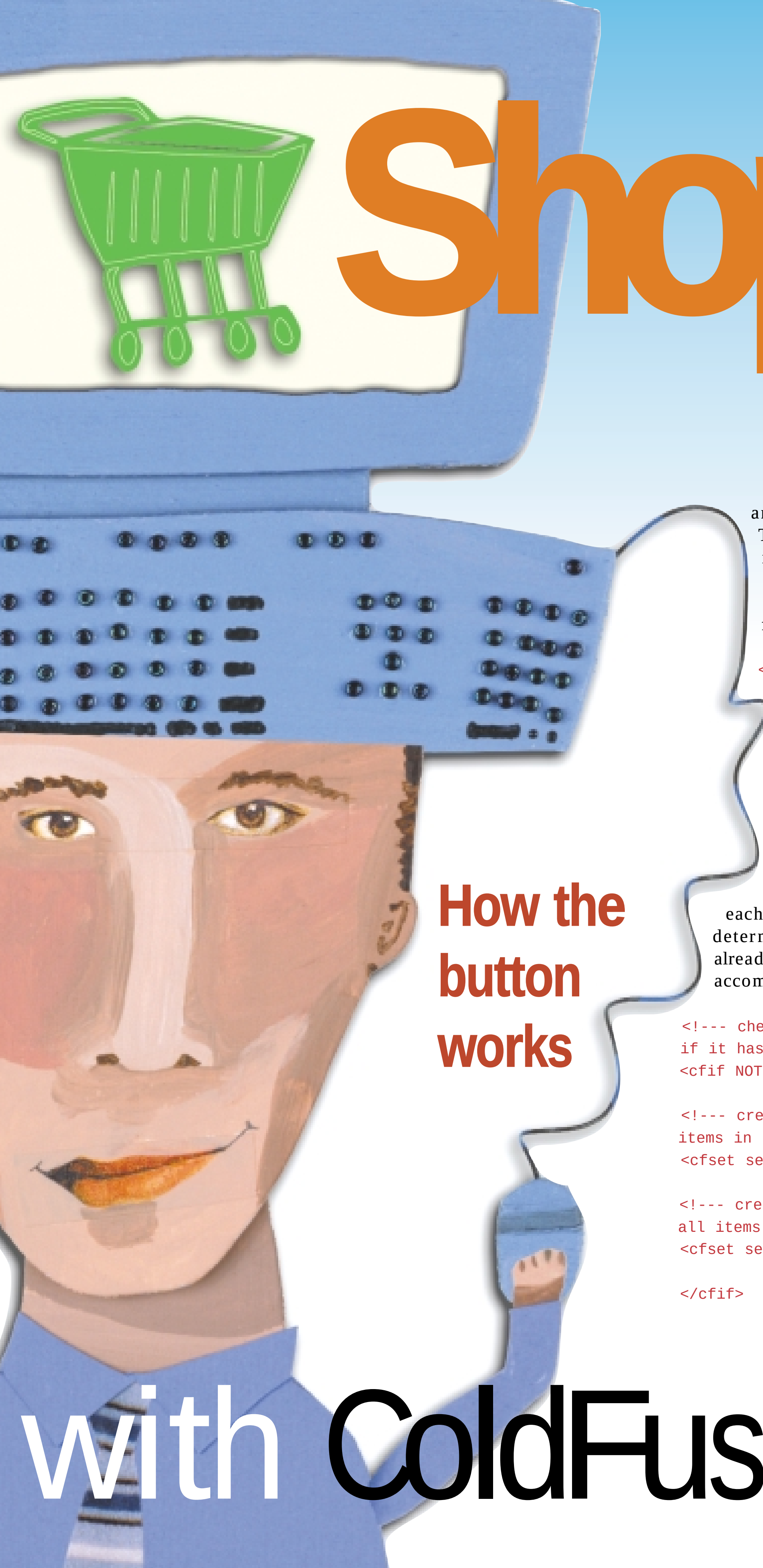
How you choose to have your items displayed for sale is up to you. You can have your products and their information stored in a database or you can code these products into a static interface. The former is easier to maintain but takes a little more effort to set up than static pages. If you're serious about creating a shopping cart, I'd suggest going with the database to store item information; however, since this article is covering the basics of a shopping cart and not database design, I've created two static items to use with the cart. The code used to create them is in Listing 1. Now that we have our items ready to sell, it's time to create a cart in which to put them.

Setting Up the Cart

In this example, our cart will consist of three templates: one for adding items, another for removing and updating items, and the third for viewing the items. An application.cfm file is also needed. There are a few things we need to consider when setting up our cart; chief among them is how to store the user's information about the items in the cart.

Shop: with ColdFusion

How the button works



I find that for simple shopping, a cart using an array stored in a session variable works well. To use session variables, session management must be enabled in the application.cfm file. The following code is an example of what this might look like (Listing 2 contains all the code for the application.cfm file):

```
<CFAPPLICATION NAME="shoppingcart"
   SESSIONMANAGEMENT="yes"


  SESSIONTIMEOUT="#CreateTimeSpan(0,0,30,0)
    >
```

The preceding code gives the application a name and tells it that we want to enable session management with the session timing out after 30 minutes. Since this page is read before each one of the cart templates, this is a good place to determine whether the user has a cart variable already or if we need to set one up. This can be accomplished by using the following code:

```
<!--- check to see if the cart has been defined
if it has not set it up --->
<cfif NOT IsDefined("session.cart")>

<!--- create a two dimensional array to store the
items in --->
<cfset session.cart = ArrayNew(2)>

<!--- create a variable to hold the total cost of
all items in cart and set it to 0 --->
<cfset session.total_cost = 0>

</cfif>
```

The preceding code checks to see if the shopping cart variable is set; if not, it sets up a two-dimensional array named *session.cart* (I'll have more about arrays when I cover adding, removing, and displaying items) as well as a variable to store the total cost of the items.

Adding Items

With our application.cfm file set up, we can now look into adding items to the cart. First, the user has to pick an item to add to the cart. And when the user clicks on that ubiquitous "Add to Cart" button, we need to ensure that the item's unique ID, as well as the cost, the quantity selected, and item description, is passed to the add template. (The code in Listing 1 shows how to set up the form.)

As you can see, each item has four variables associated with it: the item_id, item_quantity, item_cost, and item_description are all passed via the form when the user clicks the "Add to Cart" button. When the user does this, it's time to actually add the item to the cart.

We'll assume that this is the first item added, so a new array named *cart* is created and stored in a session variable. Now that we have our cart array, we can add the first item to it. To do so, we create the first record in the array and add the pertinent item information to it. (The code in Listing 3 contains the add_item template.)

The first thing that happens in the add_items template is a check to determine where we are in the array. By using the arraylen() function, we can determine the length of the array and add one to it, making subsequent items easier to add (as opposed to hard-coding the value each time). This also works on the first record, due to the fact that ColdFusion arrays start at one (as opposed to zero in other languages). The value returned by the arraylen() function is stored in the variable position. Each of the four variables is then inserted into the array using the position variable to determine the item's position in the array.

> **"Setting up a shopping cart for the Web takes a measure of time and thought"**

Our next step is to calculate the total cost and save it in the session variable total_cost. No tricks here, just some simple math. And now that we have the item in our cart and our total cost calculated, we're ready to display it to the user. I like to bring the user to the page displaying the current contents of the cart every time a new item is added. This is done by using the <cflocation> tag and specifying the view_items template in the URL attribute.

Viewing Items

The first thing we do with the view_items template is to ensure that there are, in fact, items to view. This can be done by checking the length of the cart array; if the length is zero, there's nothing to view. The following code illustrates this situation (the code for the entire view_items template is in Listing 4):

```
<cfif arraylen(session.cart) eq 0>
<font size="1" face="Arial,
Helvetica">There are no items to
display</font>
```

Once we've determined that we do have items, we need to output the item information and the total cost of the entire cart. I find that using <CFLOOP> to loop over the contents of the array works best for me. The following code sets up our loop:

```
<cfset i = 1>
<cfloop from="1" to="#arraylen(ses-
sion.cart)#" index="count">
```

The first thing we do here is set up a variable *i*. This allows us to track what position in the array we're currently accessing. The loop uses the arraylen() function to determine how many times to iterate through. Each time through the loop, several things occur: the information about the item – including the description, quantity, and cost – is output, and the user is also presented with an option to remove the item or update the quantity. Several important hidden fields are also generated. These fields are used when updating or removing an item, which is the next topic we'll cover.

Updating and Removing Items

To update or remove an item, we must first determine what action is being performed. The first thing that the remove_update_item template does is determine what to do (the entire remove_update_item template can be seen in Listing 5). This is done by checking the form variable item_quantity_2. If the variable is zero, we're removing the item. If the variable holds a value other than zero, then we're updating the item. Let's look at removing an item first.

To remove an item from the cart, we must delete it from the array. Here we simply use the ArrayDeleteAt() function and specify the cart array, the index position of the item you want to delete. The index position can be obtained from the form field array_position that was passed from the view_items template. The following code illustrates this:

```
<cfset remove_item = arraydeleteat
(session.cart, form.array_position)>
```

If the ArrayDeleteAt() function from the preceding example is successful, then remove_item will contain the Boolean value true and the item in the array at the specified position will be gone.

The next step is to update the total_cost variable. This, once again, is a simple equation and can be seen on lines 31–32 in Listing 5. The last thing we do is send the user back to the view_items template to see that the item has been removed and that the total cost has been adjusted.

But what if the user wanted to update the number of a particular item already in the cart? To update that item, two things have to happen. First, the quantity needs to be updated in the array and the total cost needs to be adjusted up or down. To adjust the quantity, simply use the code that follows:

```
<cfset session.cart[form.array_posi-
tion][2] = form.item_quantity_2>
```

The preceding code uses the array_ position variable to determine what position is to be updated; further, it uses the item_quantity_2 variable to set the quantity. Once this is done, the total_cost variable must be recalculated. This is done by subtracting the original amount of the item from the total_cost variable and adding the new cost of the item to it (as illustrated on lines 9–16 of Listing 5). Now that we can add, remove, update, and view our items, it's time to direct our customers to check out.

Checking Out

Once our customers have made up their minds about what they want to purchase and are ready to pay, we need to collect some important information from them. How much of this information you choose to gather is up to you, but name, address, credit card number, and expiration date are, for the most part, mandatory. This information can be collected through a simple HTML form and should be done through a secure connection. Once you're satisfied with the information you've collected, the next steps involve the online payment verification system and actually fulfilling the customer's order, two topics that are beyond the scope of this article.

And Don't Forget

Setting up a shopping cart for the Web takes a measure of time and thought, but we've covered most of what's necessary to set up a simple one with basic features. As always, you can customize this application to fit your specific needs and add some features we didn't cover. When you're ready to sell your wares, though, don't forget that all-important "Add to Cart" button.

About the Author

Kevin Schmidt is a senior Web developer for GeoLearning, an online learning management software company in Des Moines, Iowa.

SCHMIDT@HUNGRYCOW.COM

Listing 1: item_detail.htm

```
<!doctype html public "-//W3C//DTD HTML 4.0
Transitional//EN">

<html>
<head>
 <title>Add an Item</title>
</head>

<body>
<!-- Form for item 1 -->
<form name="add_item" method="post" action="add_item.cfm">
<table cellpadding="5" cellspacing="1" border="1"
width="500">
 <tr>
  <td bgcolor="gray" width="25"><input type="text" size="2"
name="item_quantity"></td>
  <td bgcolor="#dae1e9" width="300"><font size="1"
face="Arial, Helvetica">ColdFusion Application Server
Professional</font></td>
  <td bgcolor="#dae1e9" width="50"><font size="1"
face="Arial, Helvetica">$1295.00</font></td>
  <td bgcolor="gray" width="125" align="center"><input
type="submit" value="Add to Cart"></td>
 </tr>
</table>
<input type="hidden" name="item_id" value="12324">
<input type="hidden" name="item_cost" value="1295.00">
<input type="hidden" name="item_description"
value="ColdFusion Server (Professional)">
</form>

<!-- Form for item 2 --->
<form name="add_item" method="post" action="add_item.cfm">
<table cellpadding="5" cellspacing="1" border="1"
width="500">
 <tr>
  <td bgcolor="gray" width="25"><input type="text" size="2"
name="item_quantity"></td>
  <td bgcolor="#dae1e9" width="300"><font size="1"
face="Arial, Helvetica">ColdFusion Application Server
Enterprise</font></td>
  <td bgcolor="#dae1e9" width="50"><font size="1"
face="Arial, Helvetica">$4995.00</font></td>
  <td bgcolor="gray" width="125" align="center"><input
type="submit" value="Add to Cart"></td>
 </tr>
</table>
<input type="hidden" name="item_id" value="123245">
<input type="hidden" name="item_cost" value="4995.00">
<input type="hidden" name="item_description"
value="ColdFusion Server (Enterprise)">
</form>

</body>
</html>
```

Listing 2: application.cfm

```
<!--- set up session management --->
<cfapplication name="shopping_cart"
 sessionmanagement="yes"
 sessiontimeout="#CreateTimeSpan(0,0,30,0)#"
 >
 <!--- check to see if the cart has been defined if it has
not set it up --->
<cfif NOT IsDefined("session.cart")>

<!--- create a two dimensional array to store the items in
--->
<cfset session.cart = ArrayNew(2)>

<!--- create a variable to hold the total cost of all
items in cart and set it to 0 --->
<cfset session.total_cost = 0>

</cfif>
```

Listing 3: add_item.cfm

```
<!--- get the length of the array and add one to it --->
<cfset position = ArrayLen(session.cart) + 1>

<!--- add the item to the array --->
<cfset session.cart[position][1] = "#form.item_id#">
<cfset session.cart[position][2] = "#form.item_quantity#">
<cfset session.cart[position][3] = "#form.item_cost#">
<cfset session.cart[position][4] = "#form.item_descrip-
tion#">

<!--- set the total cost of all items --->
<cfset current_cost = form.item_cost * form.item_quantity>
<cfset session.total_cost = session.total_cost +
current_cost>

<!--- send the user to the view page --->
<cflocation url="view_items.cfm">
```

Listing 4: view_items.cfm

```
<!doctype html public "-//W3C//DTD HTML 4.0
Transitional//EN">

<html>
<head>
 <title>View Items</title>
</head>

<body>
<!--- check to see if there are items in the cart --->
<cfif arraylen(session.cart) eq 0>
<font size="1" face="Arial, Helvetica">There are no items
to display</font>
<cfelse>

<!--- loop over the array and out put each item --->
<cfset i = 1>
<cfloop from="1" to="#arraylen(session.cart)#"
index="count">
<cfoutput>
<form name="add_item" method="post"
action="remove_update_item.cfm">
<table cellpadding="5" cellspacing="1" border="1"
width="600">
 <tr>
  <td bgcolor="gray" width="25" align="center"><font
size="1" face="Arial,
Helvetica">#session.cart[i][2]#</font></td>
  <td bgcolor="##dae1e9" width="300"><font size="1"
```

```
face="Arial, Helvetica">#session.cart[i][4]#</font></td>
  <td bgcolor="##dae1e9" width="50"><font size="1"
face="Arial,
Helvetica">#dollarformat(session.cart[i][3])#</font></td>
  <td bgcolor="gray" width="125" align="center"><input
type="submit" value="Remove"></td>
  <td bgcolor="gray" width="25"><input type="text" size="2"
name="item_quantity_2" value="0"></td>
  <td bgcolor="gray" width="75"><input type="submit"
value="Update"></td>
 </tr>
</table>
<input type="hidden" name="item_quantity"
value="#session.cart[i][2]#">
<input type="hidden" name="item_id"
value="#session.cart[i][1]#">
<input type="hidden" name="item_cost"
value="#session.cart[i][3]#">
<input type="hidden" name="array_position" value="#i#">
</form>
</cfoutput>

<cfset i = i + 1>
</cfloop>
<br><br>

<cfoutput>
<font size="1" face="Arial, Helvetica">Total Cost: #dollar-
format(session.total_cost)#</font>
</cfoutput>
</cfif>
</body>
</html>
```

Listing 5: remove_update_item.cfm

```
<!--- check to see if we are updateing or deleting --->

<!--- updating --->
<cfif form.item_quantity_2 NEQ 0>

<!--- set new quantity --->
<cfset session.cart[form.array_position][2] =
   form.item_quantity_2>

<!--- refigure total cost --->
<cfset old_current_cost = form.item_quantity *
   form.item_cost>
<!-- subtract old ammount --->
<cfset session.total_cost = session.total_cost -
   old_current_cost>

<!--- add new ammount --->
<cfset current_cost = form.item_quantity_2 *
   form.item_cost>
<cfset session.total_cost = session.total_cost +
   current_cost>

<!--- back to the view items template --->
<cflocation url="view_items.cfm">
</cfif>
<!--- end updating --->




<!--- remove items --->
<cfif form.item_quantity_2 EQ 0>
<!--- remove the item from the array --->
<cfset remove_item = arraydeleteat(session.cart,
   form.array_position)>

<!--- reset the total cost of all items --->
<cfset current_cost = form.item_quantity *
   form.item_cost>
<cfset session.total_cost = session.total_cost -
   current_cost>

<!--- back to the view_items template --->
<cflocation url="view_items.cfm">

</cfif>
```

CODE LISTING

The code listing for this article is also located at www.sys-con.com/coldfusion/sourcec.cfm

BY
BEN
FORTA

Hidden Gems

Look for the little things to make your life easier

With all the talk and buzz surrounding Neo, it's important to keep in mind that ColdFusion 5 is still a relatively new product, one that many users have yet to take full advantage of. We're proud of ColdFusion 5, and rightfully so. It's the most reliable, most scalable, and most powerful ColdFusion yet, as well as the fastest server we've ever created. It's simply the best ColdFusion to date.

Although I've dedicated numerous columns this year to ColdFusion 5, I thought it worthwhile to do it once more, this time concentrating on some of the little features and enhancements that many have overlooked – features and enhancements that truly are hidden gems.

Inspecting Variables and Expressions

When debugging ColdFusion applications, it's almost always necessary to inspect the contents of variables. Simple variables can be dumped easily to screen using just a <CFOUTPUT>, like this:

```
<CFOUTPUT>#var#</CFOUTPUT>
```

But more complex variables can't be displayed that way. Arrays must be looped through, structures must be expanded, queries must be processed one row at a time, and so on. And it gets worse. Really complex variables (perhaps a structure containing an array of structures containing arrays or queries, or something like that) present developers with complicated coding challenges – not the kind of thing you want to tackle while trying to debug code. And then you need to worry about UI and presentation of this data.

Well, not anymore – I'd like to introduce you to your new debugging partner, the <CFDUMP> tag. Originally created for Spectra, this tag became so popular and useful that it made its way back into ColdFusion in CF5. So what does <CFDUMP> do? Simply put, <CFDUMP> dumps expressions, any expressions, to screen in a clean DHTML-based table.

Consider this example:

```
<CFDUMP VAR="#fname#">
```

ColdFusion will display an HTML table with the variable name on the left and its value on the right.

The next example is just as simple to invoke, but SESSION.cart is actually a shopping cart structure containing an array of selections. <CFDUMP> displays all the contents in a table that can be collapsed and expanded as needed:

```
<CFDUMP VAR="#SESSION.cart#">
```

Any data can be dumped with <CFDUMP>. The next example displays all SESSION variables for the active SESSION (SESSION is a structure internally). And if one of those SESSION variables were SESSION.cart, then all of the nested data within that variable would be displayed as well:

```
<CFDUMP VAR="#SESSION#">
```

<CFDUMP> is a powerful tool, and easy to use, one that you'll wonder how you lived without.

Client-Side Regular Expression Validation

Next is one of my favorite ColdFusion 5 features. It's also one of the least known (I don't think Macromedia has even mentioned it on feature lists; the only place you'll find it is in the CFML tag docs).

You're probably familiar with <CFINPUT>, the form tag that can be used to perform client-side form validation using automatically generated JavaScript code. <CFINPUT> is very useful, but it has one serious limitation – it can't be extended easily. Let me explain: if you need to flag a field as required, if you need to validate a U.S. phone number, if you need to validate a date, then <CFINPUT> works well. But if you need to validate anything that isn't one of the supported validation types – e-mail addresses, non-U.S. phone numbers, required minimum value lengths, for example – then <CFINPUT> falls flat on its face.

Or rather, it used to. In ColdFusion 5 <CFINPUT> has been extended to let you validate just about anything you can imagine. How? By adding one new validation type, Regular Expressions. In CF5 you may use a Regular Expression pattern as the validation rule for your field. ColdFusion will generate the needed JavaScript so the browser won't validate the form field unless the value matches the specified Regular Expression.

Let's look at an example. I recently created a form that had to prompt for RGB values (colors specified in amounts of red, green, and blue). RGB values are six characters long – three sets of two hexadecimal values (00 to FF). My form had three text fields, one for each of the three values. Here's the <CFINPUT> used for the first one:

```
<CFINPUT TYPE="text"
          NAME="color_r"
VALIDATE="regular_expression"
          PATTERN="[A-Fa-f0-
9]{2,}"
          MESSAGE="RGB value
must be 00-FF"
          SIZE="2"
          MAXLENGTH="2">
```

As you can see, I used VALIDATE="regular_expression" and then provided the Regular Expression to use in the new PATTERN attribute. [A-Fa-f0-9] matches a single character of A through F (upper or lower case) or 0 through 9. The {2,} instructs the browser to accept a minimum of two instances of the previous expression. That, coupled with MAXLENGTH="2", and I have a perfect validation rule.

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
vp, business development
Grisha Davida grisha@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
Alisa Catalano alisa@sys-con.com
associate sales manager
Kristen Kuhnle kristin@sys-con.com

editorial
executive editor
M'lou Pinkham mpinkham@sys-con.com
editor
Nancy Valentine nancy@sys-con.com
managing editor
Cheryl Van Sise cheryl@sys-con.com
associate editor
Jamie Matusow jamie@sys-con.com
associate editor
Gail Schultz gail@sys-con.com
assistant editor
Lin Goetz lin@sys-con.com

production
vice president, production & design
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
Aarathi Venkataraman aarathi@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
conference manager
Michael Lynch mike@sys-con.com
show assistant
Niki Panagopoulos niki@sys-con.com
registration assistant
Jaclyn Redmond jaclyn@sys-con.com
JDJ store manager
Anthony D. Spitzer tony@sys-con.com

web services
web designer
Stephen Kilmurray stephen@sys-con.com
web designer
Christopher Croce chris@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
accounts receivable
Jan Braidech jan@sys-con.com
accounting clerk
Patti DelVecchio patti@sys-con.com

Here's another one – one we probably all wish were a built-in rule. It validates that an e-mail address is formed correctly:

```
<CFINPUT TYPE="text"
          NAME="email"
VALIDATE="regular_expression"
          PATTERN="[A-Za-z0-9_]+@[A-Za-z0-9_]+\.[A-Za-z]+"
          MESSAGE="Please enter a
valid E-Mail address">
```

Here the Regular Expression matches one of more alphanumeric characters (or an underscore) followed by an @ sign, followed by one or more alphanumeric characters (again allowing underscores), followed by a period, then followed by one or more alphanumeric characters. This won't actually check that a specified e-mail address is a valid working address, but it will at least prevent completely invalid addresses from being entered.

As you can see, with minimal work you can write Regular Expressions to validate all sorts of things. <CFINPUT> is limited no longer.

Note: The Regular Expressions used here aren't the same as those used in the CFML RE functions. The Regular Expressions passed to <CFINPUT> are those supported by JavaScript.

Better Log File Use

ColdFusion creates log files. Lots of them. With lots of entries in each. Log files are an invaluable debugging and troubleshooting tool – if you're able to find just the entries you need, and if you can work out what the user was doing before the log entry was generated.

That's where the new <CFLOG> tag comes in. <CFLOG> lets you write your own entries to log files (either the standard log files or your own). Look at this example:

```
<CFLOG APPLICATION="yes"
        TEXT="User name is
#user.name#">
```

This code writes the log text to the Application.log file. If a particular line of code were generating an error, you could insert a <CFLOG> call like this above it so that when browsing the log file you'd have enough information to properly diagnose the problem.

When used in conjunction with CF5's new log viewer (in the ColdFusion Administrator), <CFLOG> makes those large log files far more manageable and accessible.

Accessing HTTP Data

ColdFusion provides access to all sorts of server, client, and environmental data via CGI and other variables. But have you ever wanted access to data not explicitly made available? If so, you must have discovered that there really is no way to get to it.

Until now. New to ColdFusion 5 is the GetHTTPRequestData() function, which exposes any and all HTTP data. Here's the function call:

```
<CFSET data=GetHTTPRequestData()>
```

GetHTTPRequestData() returns a structure containing headers, raw content, server information, and more. To display the raw content you could simply do this:

```
<CFOUTPUT>
#data.content#
</CFOUTPUT>
```

The structure also contains a second structure, names headers. You can loop through or search this structure to access every header and value. In other words, using GetHTTPRequestData(), you now have access to any and all relevant data.

Summary

ColdFusion 5 has lots of great new features that I've covered in previous columns – things like user-defined functions, graphing, querying queries, server probes, and so on. But just as compelling as these marquee features are all the little enhancements, many of which just slipped into the product with no fanfare at all.

As I stated at the start of this column, ColdFusion 5 is simply the best ColdFusion to date. If you haven't upgraded yet, do so quickly. And if you're already running CF5, look for the little features and enhancements that can make your development easier and more productive. I listed just four here. If you find others you think are significant, let me know.

ABOUT THE AUTHOR
Ben Forta is Macromedia's senior product evangelist and the author of numerous books including the recently published ColdFusion 5 Web Application Construction Kit *and its sequel,* Advanced ColdFusion 5 Development. *For more information on Ben's books visit* www.forta.com.

@ BEN@FORTA.COM

Using MS-SQL Stored Procedures with ColdFusion Part 2 of 3

Easier than you've imagined – if you're meticulous

BY IAN RUTHERFORD

Why should ColdFusion analyze the query results when the database could do it and you could save the bandwidth for something that really needs it?

In Part 1 of this article (***CFDJ***, Vol. 3, issue 10) I introduced stored procedures and performed some basic tasks with them. In this article I show how to do more complex tasks that involve IF/ELSE statements, looping, and string manipulation.

First, see Listing 1; how often have you had to do something similar? Let's say you just received a note from your product supplier stating that Item 20 has a new, lower everyday price of $15. Now your store is running a special in which you give away certain items, but you can't remember if Item 20 is one of those items. You could either look up the item and its cost and then decide whether to send another request to the database to update the price, or simply tell the database which item you want to edit, its new cost, and let the database do everything else.

To process this request the database needs two pieces of information: the item number and the price. The ColdFusion end of the procedure looks like Listing 2.

In the database we need to say what the procedure will be called (I begin all my procedures with "pr_" to identify them as stored procedures. It may seem to make more sense to use "sp_", but "sp_" is used for system stored procedures that can be accessed from any database on the server.)

```
CREATE PROCEDURE [pr_ItemCost]
```

We add our variables:

```
@ItemNo [int],
@Cost [money]

AS
```

Then we write the SQL statement. First we need the current value for cost because if it's zero, we aren't going to change the amount. If we only want to know if the cost is zero but don't care what it is if it's not zero, we can use the @@ROWCOUNT function instead of actually setting local variables and doing comparisons. (@@ROWCOUNT is a function that tells how many rows were affected by the previous statement.)

```
SELECT Cost
FROM Items
WHERE ItemNo = @ItemNo
AND Cost = 0

IF @@ROWCOUNT = 0
  BEGIN
    UPDATE Items
    SET Cost = @Cost
    WHERE ItemNo = @ItemNo
  END
```

Let's look at this code. In the first select statement we determined if Item 20 had a cost of zero. By querying for the item with a cost of zero we could get two results: one row or zero rows. @@ROWCOUNT tells us how many were returned. If @@ROWCOUNT=0, then Item 20 has a cost that's not zero.

Our IF statement requires a BEGIN and an END command to let the database know where to stop the execution of the IF statement. In between we put our update statement that will change the price of the item only if its original price wasn't zero.

What if we do care what the cost currently is? Maybe we have a company policy that items under $10 will only increase in cost by $3 or the new cost, whichever is less, but everything else that isn't free increases to whatever the new cost is. Normally, you would change your CF code to reflect the business rules, but with stored procedures you can keep the two separate. This will keep your Web developers happy because they probably enjoy making new site features more than getting into the minutia of business rules.

The procedure starts the same but we'll need a local variable within the procedure to hold the current cost of the item:

```
CREATE PROCEDURE [pr_ItemCost]
```

Then we need to add our variables:

```
@ItemNo [int],
@Cost [money]

AS

DECLARE
  @CurrentCost [money]
```

Next, we need to set this new variable equal to the current value using a special select statement:

```
SELECT
  @CurrentCost = Cost
FROM Items
WHERE ItemNo = @ItemNo
```

Finally, we need to use our business rules to decide if the cost should change:

```
IF @CurrentCost = 0
  BEGIN
    . . .
  END
ELSE IF @CurrentCost < 10
  BEGIN
    . . .
  END
ELSE
  BEGIN
    . . .
  END
```

Notice that each piece of the IF/ELSE statement requires its own BEGIN and END. You can include as many IF/ELSE statements as you need, and you don't need to include a final ELSE clause, just as in ColdFusion.

IF/ELSE statements are useful but what's really helpful is the ability to replace some of your CFLOOPs over recordsets with a single stored procedure, saving you many trips to and from the database.

Let's assume that the items in our store can be assigned to various special occasions. One way to do this is to have a multiple select-box form submitted to a template that would loop over the list, adding a row to the database each time. If you have the following list of events: birthday, wedding, Mother's Day, anniversary, and graduation, you could have up to five separate calls to the database. Instead, let's pass the entire list to the database and let the database do the looping.

Looping in SQL consists of using WHILE instead of LOOP. This is similar to using conditional CFLOOPs.

We want to loop over the list, add each value to the database, then remove it from the list until the list is empty. This will require some string manipulation as well as looping. This complete procedure can be seen in Listing 3.

```
CREATE PROCEDURE
[pr_AddItemEvents]

@ItemNo [int],
@EventList [varchar](40),
@EventValue [int]

AS

DECLARE @ListLength [int],
  @CommaCount [int]
```

@ListLength will hold the character count of our list. @CommaCount holds the position of the first comma in our list.

```
SET
  @ListLength =
  Len(@EventList)
```

Len() in SQL works the same as in CF

```
WHILE @ListLength >0
  BEGIN
    SET
     @CommaCount =
CHARINDEX(',', @EventList)
```

CHARINDEX tells us at what character count the first match for the pattern occurs.

We need to know this so we can properly extract our values from the list. Use the function PATINDEX if you need to match strings with wild cards (*, %).

```
IF @CommaCount > 0
  BEGIN
    SET
      @EventValue =
      SubString(@EventList, 1,
@CommaCount-1)
  END
```

@Event Value is the current value in the list. If there's a comma in the list we need to take all the characters starting at 1 and ending right before the comma, which is why we use @CommaCount –1.

```
ELSE
  BEGIN
    SET @EventValue =
@EventList
  END
```

If there aren't any commas we can set @EventValue equal to whatever is left in the list.

```
INSERT INTO Event_Items_Xref
   (Item_Fk,
   Event_Fk)
VALUES
   (@ItemNo,
   @EventValue)
```

If we were doing the looping on the CF server instead of in the database, we would repeatedly have to call this query through ColdFusion.

```
IF @CommaCount > 0
   BEGIN
      SET
         @EventList =
         Right(@EventList,
@ListLength-@CommaCount)
   END
```

If we have commas in the list we need to remove the current item and the trailing comma from the list. The Right function works the same way in ColdFusion as in SQL.

```
ELSE
   BEGIN
      SET @EventList = ''
   END
```

If there aren't any commas in the list, the list is at its last value so we can remove the last value from the list.

```
   SET @ListLength =
Len(@EventList)
END
```

We need to reset the list length after each loop so that the WHILE statement can check the current length. When the length equals zero, the loop stops running.

This article presented several useful programming constructs that will save your applications from multiple trips to the database. However, stored procedures can do much more. For example, stored procedures have access to SQL objects called *cursors*. A cursor is a result set that you can loop over one row at a time – similar to using CFOUTPUT with a query. This is handy for shopping carts in which each line item needs processing as it's added to an order.

There are also several special stored procedures that allow you to use the command line, similar to CFEXECUTE, and registry editing commands, similar to CFREGISTRY. Finally, as with ColdFusion, you can create special C++ objects that can be accessed through stored procedures, making it possible to move complex programming off the Web server and into the database.

If you want to find out more about MS stored procedure programming, I recommend the book *SQL Server 2000 Stored Procedure Programming* by Dejan Sunderic and Tom Woodhead (Osborne Press). This book covers all the basics as well as more advanced topics including OLE and XML integration. I also recommend opening up the SQL Server Books Online (www.microsoft.com/sql/techinfo/productdoc/2000/books.asp) every time you have a question about syntax or possible functions. They're a very thorough reference and better than any SQL reference book you can buy.

@ IRUTHERF@CATHOLICSTORE.COM

ABOUT THE AUTHOR

Ian Rutherford has been working with CF and SQL for two years and is the designer of CatholicStore.com and Catholicliturgy.com, both built using Fusebox architecture.

Listing 1

```
<CFQUERY NAME="getCost"
     DATASOURCE="#request.site.ds#"
     DBTYPE="#request.site.dbtype#">
  SELECT Cost
  FROM  Items
  WHERE  ItemNo = 20
</CFQUERY>

<CFIF getCost.Cost GT 0>
  <CFQUERY>
    . . .
  </CFQUERY>
<CFELSEIF getCost.Cost EQ 0>
  <CFQUERY>
    . . .
  </CFQUERY>
<CFELSE>
  <CFQUERY>
      . . .
  </CFQUERY>
</CFIF>
```

Listing 2

```
<CFSTOREDPROC
  PROCEDURE="ChangeCost"
  DATASOURCE="#request.admin.ds#">
    <CFPROCPARAM
      TYPE="In"
      CFSQLTYPE="CF_SQL_INTEGER"
      DBVARNAME="@ItemNo"
      VALUE="#Attributes.ItemNo#"
      NULL="No">
    <CFPROCPARAM
      TYPE="In"
      CFSQLTYPE="CF_SQL_MONEY"
      DBVARNAME="@Cost"
      VALUE="#Attributes.Cost#"
      NULL="No">
</CFSTOREDPROC>
```

Listing 3

```
CREATE PROCEDURE [pr_AddItemEvents]

@ItemNo [int],
@EventList [varchar](40),
@EventValue [int]

AS

DECLARE @ListLength [int],
  @CommaCount [int]

SET
  @ListLength =
  Len(@EventList)

/* Start the loop over the list */
WHILE @ListLength >0
  BEGIN

     /* Determine where our first comma occurs */
     SET
       @CommaCount = CHARINDEX(',', @EventList)

    /* If there is a comma, take the first value */
    IF @CommaCount > 0
      BEGIN
        SET
          @EventValue =
          SubString(@EventList, 1, @CommaCount-1)
      END

    /* If not, take what’s left */
    ELSE
      BEGIN
        SET @EventValue = @EventList
      END

  /* Run the query */
  INSERT INTO Event_Items_Xref
    (Item_Fk,
    Event_Fk)
  VALUES
    (@ItemNo,
    @EventValue)

  /* If there was a comma, remove the first value and its
trailing comma */
  IF @CommaCount > 0
    BEGIN
      SET
        @EventList =
        Right(@EventList, @ListLength-@CommaCount)
    END

  /* Otherwise, delete everything left in the list */
  ELSE
    BEGIN
      SET @EventList = ''
    END

  /* Set the new list length so the loop knows if it
should run again */
  SET @ListLength = Len(@EventList)
END
```

CODE LISTING

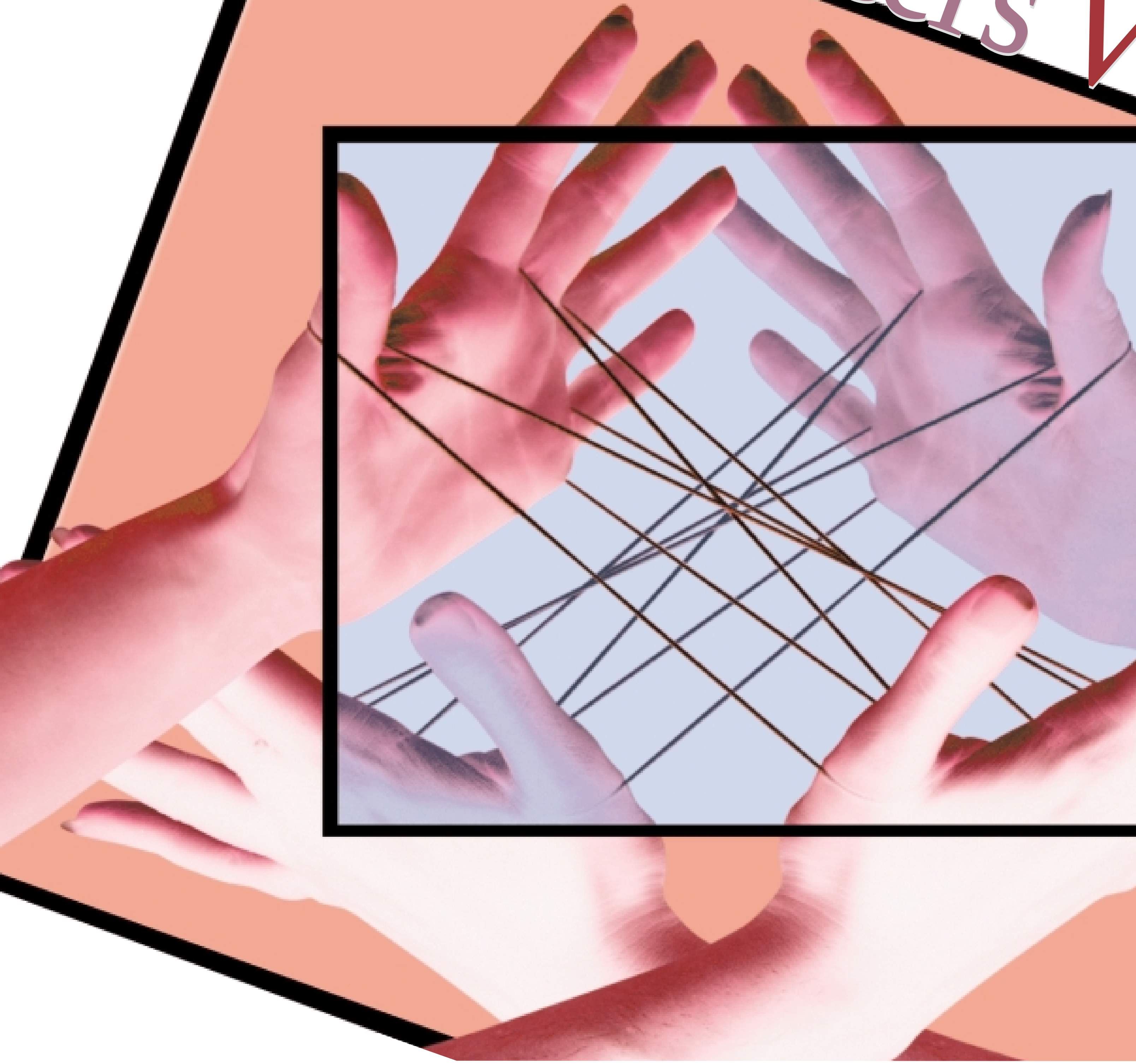
The code listing for this article is also located at

www.sys-con.com/coldfusion/sourcec.cfm

ColdFusion Feature

By Norman Elton

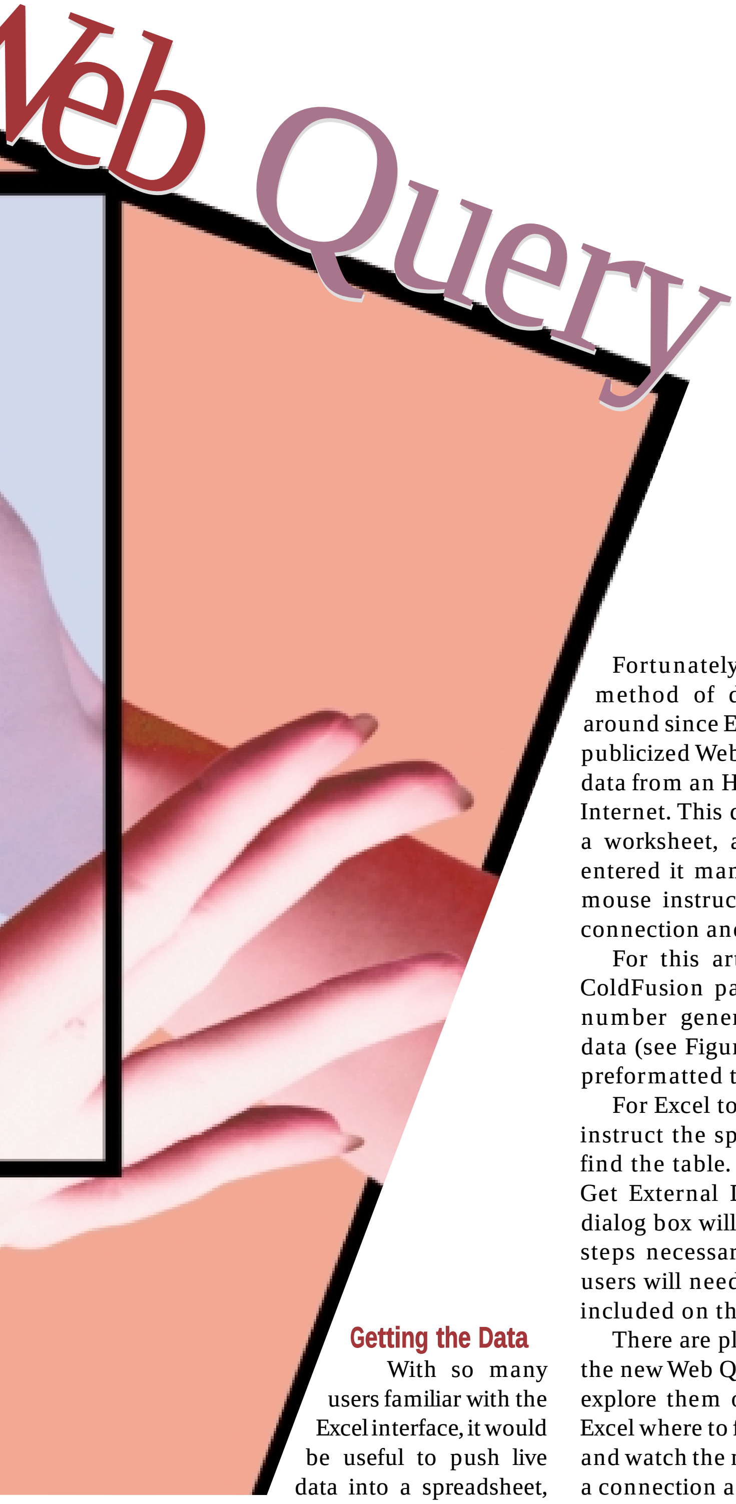
Excel's V



In today's world of e-commerce, simply viewing data in a Web browser is rarely enough. Clients constantly demand reporting, charting, graphing, and analysis of complex data. Although Macromedia has made great strides in this direction by incorporating graphing capabilities (the new CFGRAPH) and access to a reporting engine (CFREPORT), these are all server-side solutions. A simple request for a new chart can send developers scrambling. Wouldn't it be great if end users could manipulate and analyze data themselves, using a powerful suite of tools that they're already familiar with!

Since its inception in 1988, Microsoft Excel has grown to become the Swiss Army knife of data analysis. While not always the best tool for the job, users tend to make it do just about anything. Nontechnical people can easily store inventories, financial records, even data best suited for a relational database. Excel then gives them the power to twist this data into any imaginable chart or report.

Note: **All Excel examples are based on Excel 2000. Excel 97 and Excel XP users will find the same functionality with minor differences in interface and terminology.**



How to feed live data into Excel from a Web server

Getting the Data

With so many users familiar with the Excel interface, it would be useful to push live data into a spreadsheet, leaving the actual manipulation to the discretion of the user. But how do we feed live data into Excel from a Web server? Supply the users with tab delimited files? Excel has no problems there, but the user would have to download a new file every time the data changes. Copy and paste? Far too tedious. Office XP promises powerful new integration with the Web-based XML data.

Fortunately, an extremely powerful method of data acquisition has been around since Excel 97. The woefully underpublicized Web query allows Excel to fetch data from an HTML table anywhere on the Internet. This data is dropped directly into a worksheet, and behaves as if the user entered it manually. A simple click of the mouse instructs Excel to open an HTTP connection and get the latest data.

For this article, I've set up a special ColdFusion page that when given a part number generates a table of inventory data (see Figure 1). Excel can also handle preformatted text, blocked in the PRE tag.

For Excel to feed off this data, we must instruct the spreadsheet exactly where to find the table. From the Data menu, click Get External Data > New Web Query. A dialog box will then walk you through the steps necessary to import data. Excel 97 users will need to install Microsoft Query, included on the Microsoft Office 97 CD.

There are plenty of advanced options in the new Web Query window, and I'll let you explore them on your own. Once you tell Excel where to find its data, simply click OK, and watch the magic unfold. Excel will open a connection and parse the HTML file. Any formatting, whether HTML- or CSS-based, will be preserved (to a degree), so you can reliably set fonts, color, borders, etc.

Once the download is complete, right click on any imported cell and you'll see a new choice on the context menu. "Refresh Data" instructs Excel to go out and fetch the latest data, dropping it back into the table. There's also a Refresh Data icon on the toolbar (a red exclamation mark) that refreshes all the Web queries in the document. If your HTML table has changed sizes, Excel is even smart enough to expand or contract its range of cells to accommodate the change without overwriting other content on the sheet.

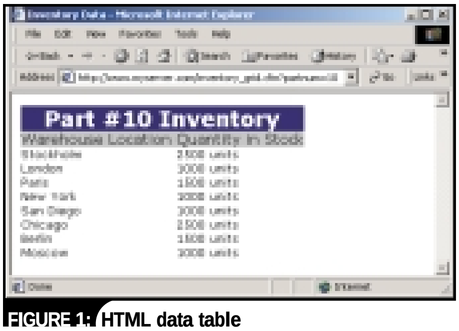
Notice that you no longer have to tell Excel where to get its data. The URL and any configuration parameters have been embedded into the document. You can convert a Web query back to plain data by right-clicking on the cell, choosing Data Range Properties, and clearing the "Save query definition" checkbox.

Excel Extensions

When composing the live Web page, Excel provides a few extensions to normal HTML. These extensions, invisible when viewing the page in a Web browser, provide extra functionality when the document is interpreted by Excel. We'll cover two of them here: filters and formulas. Additional extensions allow you to properly format data and work with PivotTables.

The first extension allows the user to quickly filter data based on a set of parameters. This applies if you have a row of column headers. In our case, the following code generates a header:

```
<TR>
  <TD Filter="ALL">Warehouse
Location</td>
  <TD Filter="ALL">Quantity In
Stock</td>
</tr>
```



Inventory Data - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address
Done
Internet

Part #10 Inventory	
Warehouse Location	Quantity in Stock
Stockholm	2500 units
London	3000 units
Paris	1500 units
New York	3000 units
San Diego	3000 units
Chicago	2500 units
Berlin	1500 units
Moscow	3000 units

FIGURE 1: HTML data table

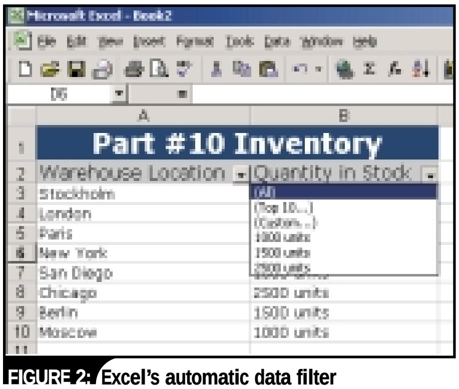
By supplying the "filter" attribute, Excel will generate a drop-down menu of filter options (see Figure 2). The user can then choose to filter for a particular value, the top 10 values, etc. All of this is accomplished using Excel's built-in AutoFilter feature. Note that for filtering to work correctly, Excel needs to use HTML formatting. This is an option in the new Web Query window.

Excel also allows you to build a formula in a cell. For instance, to find the total stock of a given item across 10 warehouses, we could use:

```
<TD Formula="=Sum(B2:B11)"></td>
```

In this case, Excel will sum the values of the range B2 to B11 and display the result in the cell. If the user decides to change one of the values, the sum will update to show the change. Although not required in a formula attribute, the equal sign inside the quotes matches Excel's function notation.

It's important to remember that a user can drop the results of a Web query anywhere on a spreadsheet. It may seem like summing the results of cells B2 to B11 could be unreliable. What happens if the user puts the query at cell D35? Fortunately, Excel corrects this. When the formula appears on the sheet, it will use the data relative to your table. Therefore, if you specify cell A1 in a formula, it will always represent the upper-left corner of the HTML table, regardless of where the query was placed on the spreadsheet.



	A	B
1	Part #10 Inventory	
2	Warehouse Location	Quantity in Stock
3	Stockholm	
4	London	
5	Paris	
6	New York	
7	San Diego	
8	Chicago	2500 units
9	Berlin	1500 units
10	Moscow	1000 units
11		

FIGURE 2: Excel's automatic data filter

IQY Files

Although Web queries are obviously powerful, they still require a degree of configuration on the part of the user. Do you really want people to have to worry about URLs and query parameters? Luckily, Microsoft has devised a standard for defining a Web query. An IQY file, a simple text file, allows system administrators to package a query for distribution. The user can then run the query from the Data menu (click Get External Data > Run Saved Query). Excel 2000 and XP users can even double-click an IQY file and Excel will open a new spreadsheet and import the data. Unfortunately, this feature was left out of Excel 97.

To understand IQY files, let's look at a simple example:

```
WEB
1
http://www.myserver.com/inventory_
grid.cfm?partnum=10
```

The first two lines of an IQY file are easy, in fact, they're always the same. The first instructs Excel how it will import the data. Notice that "WEB" is uppercase. The second line represents the version of the IQY definition. Fortunately, Microsoft has created only one version, so we'll stick with 1. Running this IQY file will produce the same results as manually configuring a Web query.

In the previous example, Excel imported and displayed inventory data for a particular part. What if we don't want to limit the user to a specific part? Let's consider a more sophisticated IQY file:

```
WEB
1
http://www.myserver.com/inventory_
grid.cfm?partnum=["Part Number","Please
enter a part number:"]
```

Now the user will be prompted for a part number to display. The first argument ("Part Number") is a descriptive name of the variable. The second argument is displayed as a prompt when the user loads the data (see Figure 3). The user's entry is passed to the Web server as a URL parameter. To pass multiple URL variables, simply separate them by ampersands, as you would an ordinary URL:

```
WEB
1
http://www.myserver.com/inventory_
grid.cfm?partnum=["Part Number","Please
enter a part number:"]&warehouse=
["Warehouse ID","Please enter a
warehouse ID:"]
```

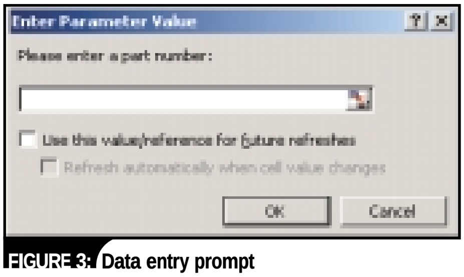
If, on the other hand, you want to pass data as form variables, simply place the prompt on the fourth line:

```
WEB
1
http://www.myserver.com/inventory_grid.
cfm
partnum=["Part Number","Please enter
a part number:"]
```

Multiple form variables are delineated by ampersands, much like URL parameters:

```
WEB
1
http://www.myserver.com/inventory_grid.cfm
partnum=["Part Number","Please enter a
part number:"]&warehouse=["Warehouse
ID","Please enter a warehouse ID:"]
```

There are a number of miscellaneous options you can specify on subsequent lines, such as "Selection=<table name>", which controls which table on the HTML page gets parsed. The best way to explore these extra parameters is to build a Web query in Excel, save it to an IQY file, and examine the results. Individual options may or may not be compatible with older versions of Excel, so be sure to test your files before distributing them!



Enter Parameter Value

Please enter a part number:

□ Use this value/reference for future refreshes

□ Refresh automatically when cell value changes

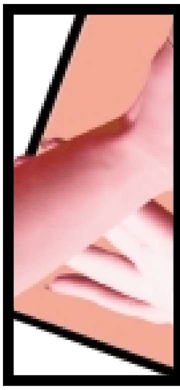
FIGURE 3: Data entry prompt

Dynamic IQY Files

Although it's fairly simple to write an IQY file and place it on the company intranet for users to download, there are times where it would be handy to generate IQY files on the fly. For instance, a Web page that displays inventory on a particular part could have a link "Connect to Excel". This could download a static IQY file, although an administrator would have to make sure that an individual IQY file exists for every part in the database.

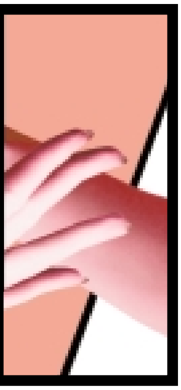
If we could pass the part number in as a URL variable, we could write the following file, saving it as "connect_to_excel.cfm":

```
WEB
1
http://www.myserver.com/inventory_grid.
cfm?partnum=<CFOUTPUT>#URL.PartNumber
#</cfoutput>
```



Data Security

If you're security-conscious, Excel uses the same security system as Internet Explorer. If you try to access a page that's protected by the Web server, a standard login prompt will appear. Once authenticated, the username would then be available to ColdFusion through the CGI.AUTH_USER variable. If you want to further restrict the data to a select list of users, code the HTML table to return an error message if the supplied user name doesn't meet the proper criteria.



When users click the link to this CFM page, however, their Web browsers will likely display the file on screen as plain text. They would then have to save the file with an IQY extension, and then open it in Excel.

There are two distinct problems with this solution. First, the file is displayed on screen and the user must know to save it locally. To remedy this problem, we'll employ the CFCONTENT tag, which changes the MIME type of the document. Originally created to handle e-mail, the MIME type instructs the browser how to display a document. The default type for Web pages is "text/html".

In this case, we'll utilize the "application/octet-stream" type that is reserved for files the browser can't display, such as binary data. This will force a "Save As" dialog box to appear. The following CFML will return the file with the new MIME type:

```
<CFCONTENT type="application/octet-stream">WEB
1
http://www.myserver.com/inventory_grid.cfm?partnum=<CFOUTPUT>#URL.
PartNumber#</cfoutput>
```

Notice that the CFCONTENT and the "WEB" specification are on the same line. If we were to put a linefeed between the two, an empty line would appear in the resulting file and confuse Excel.

Once the "Save As" dialog box appears, the file will be saved as "link_to_excel.cfm". Ideally, the file should come with an IQY extension, ready to be loaded into Excel.

Instead of directly linking to connect_to_excel.cfm (our dynamic IQY file), we'll employ the following trick:

```
<A
HREF="http://www.myserver.com/connect_to_excel.cfm/DynamicLink.iqy">
```

ColdFusion is smart enough to process connect_ to_excel.cfm, but the browser sees the file as DynamicLink.iqy.

For More Information

Unfortunately, documentation for Web queries is hard to find, especially regarding the IQY file syntax. Microsoft published the Web Connectivity Kit for Excel 97, which, to my knowledge, was its last official publication on the matter, but there is another excellent resource at www.15seconds.com/issue/991021.htm. Microsoft also includes some great examples used to fetch stock reports.

By feeding live data into Excel, you can empower users to manipulate data on the client side without requiring constant changes to the Web application. They can then use Excel's capabilities to analyze and chart data. By using Web queries, users can effortlessly update their data.

About the Author

Norman Elton is a computer science student and the ColdFusion guru at the College of William & Mary in Williamsburg, Virginia.

@ WNELTO@WM.EDU

Validating Input with **Regular Expressions**

Little known features of CF5

BY
CHARLES
AREHART

Have you ever wished CFINPUT provided a way to validate an e-mail address? Or were you frustrated that its telephone validation didn't allow parentheses around an area code among other such limitations?

Well, your wish has been granted in a new, little-known feature of CF5.

While many will know of the top 10 or so new features in CF5, few will know about the dozens (yes, dozens) of less-promoted enhancements. One of these is the ability to validate user input by way of regular expressions. (If you're new to regular expressions, we'll show you some examples and point you to the CF docs where you can learn more.)

The new feature isn't something that you couldn't do previously, if you knew how to code input validation tests in JavaScript. But, like so many features in CF, the point is that a single tag can solve the problem for you much more easily.

New Capabilities for Tags

What's new is that the CFINPUT and CFTEXTINPUT tags, which are both used within a CFFORM tag, now offer the ability to validate their input using regular expressions. That means you're no longer limited to the validations built into the tags (such as date, time, telephone, zip code, credit card, Social Security number, etc.)

If you're not familiar with these validations, they're quite useful (with some caveats). The CFINPUT tag, for instance, builds JavaScript for you that's sent to the browser to ensure that the user's input for a given field matches an expected pattern for the chosen validation. You can read more about these tags in the CF docs or your favorite CF book (see Sidebar).

Of course, the new capabilities are discussed only in the CF5 docs. If you have not yet installed CF5, you can view the documentation mentioned throughout this article by visiting http://livedocs.allaire.com. Also, if you install the new CF5 update for Studio 4.5.2 (www.macro media.com/software/coldfusion/resources/cf_tag_update/contents.html), it will install new help that appears as a new book in the Studio Help feature: "ColdFusion 5 (new manuals)."

There's one potential source of confusion, however. While the update does offer new CF5 documentation that covers this new feature as well as new tag insight, tag completion, tag editors, and tag help for most CF5 tags and functions, the tag editor and tag insight features don't include these new attributes for CFINPUT and CFTEXTINPUT. The same is true in CF Studio 5 as of RC1. However, the features do work in CF5.

Getting back to input validation, the built-in validation available for CFINPUT and CFTEXTINPUT has always been somewhat limited (phone numbers had to include an area code, but you couldn't surround them in parentheses) or just nonexistent (there's no built-in validation for e-mail addresses, among other things).

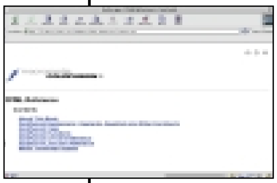
As of Release 5 of CF, you can now create any form of validation you want, if you know how to create the regular expression to describe the pattern that the input should follow. Want to validate Federal Employer Identification Numbers (FEINs, which are not the same format as SSNs)? You can. Want to let people enter phone numbers with or without area codes, and allow parentheses around the area code? You can.

Validating an E-Mail Address

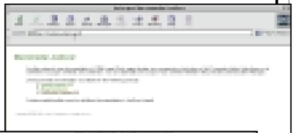
Want to add an e-mail validation? You can do that, too. It's as easy as specifying that the validation will be

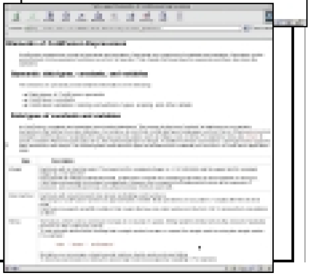
Are You Using the CF Documentation?

There are lots of great CF books on the market, but did you know that CF comes with several useful manuals? Often, the print version ends up on the shelf of the CF administrator rather than in the hands of CF programmers. If you can get the printed manuals (the set is also available for $50 from Macromedia), they're quite handy and include a reference manual, a programmer's guide, and more.



They're also available in HTML format by several means: first, they're installed within Studio (one way to reach them is to use the menu command Help>Open Help References Window). They may also be on the CF server (http://<your server>/CFDOCS/dochome.htm), though typically they shouldn't be installed on production servers for security reasons outlined in an Allaire Security Zone bulletin.





The good news is that you can also now find them online at http://livedocs.allaire.com, a new service that also allows you to post comments on each page! Last, the docs are also available in PDF format at www.allaire.com/developer/documentation/coldfusion.cfm.

using regular expressions (using VALIDATE="regular_expression") and then offering the expression in a new PATTERN attribute. Here's an example that will validate e-mail addresses:

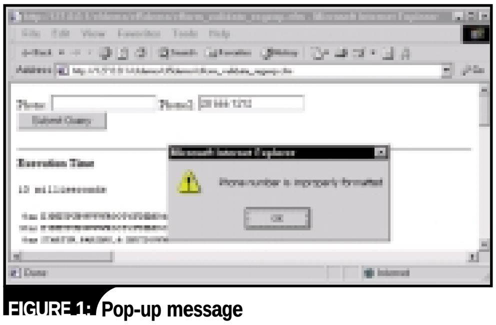
```
<CFFORM ACTION="">
Email: <CFINPUT TYPE="Text"
NAME="email"
VALIDATE="regular_expression"
PATTERN=" ^([_a-z0-9-]+(\.[_a-
z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-
]+)*\.(([a-
z]{2,3})|(aero|coop|info|muse-
um|name)))?$"
MESSAGE="Email is improperly
formatted">
<INPUT TYPE="Submit">
</CFFORM>
```

That regular expression in the PATTERN attribute may seem quite daunting. For now, just accept that regular expressions that do something useful may exist – and the cool thing is you can copy them from others to get the benefit.

For example, I borrowed the e-mail matching pattern from one created by Jeff Guillaume for his IsEmail user-defined function offered at the www.CFLib.Org Web site. (If you're not aware of this great repository of user-defined functions run by CF mavens Ray Camden and Rob Brooks-Bilson, do check it out!)

Note that the pattern isn't perfect: for instance, it won't catch if the user types somename@somedomain.om (leaving off the *c* in ".com"). The pattern does allow two or three character domain names (to allow for international domains like ".ca" and ".au".) What may surprise others is that last list of values (.aero, .coop, .info, .museum, .name). These are the newer top-level domain names that have been created recently by the InterNIC. There are still others, but they're either two or three characters.

One other thing to note: it would be best to copy and paste this code rather than try to type it. Regular expressions are pretty picky, and if you get it wrong, you don't generally get an error mesage – the expression just fails to work as expected. For instance, the character used between "aero" and "coop" is a bar (|), not a lowercase letter L (l), nor the number one (1).



Phone number is improperly formatted

FIGURE 1: Pop-up message

How CFFORM Works

I mentioned earlier that using CFINPUT within CFFORM would cause CF to build JavaScript that's sent to the browser on your behalf. That's the cool thing: you don't need to understand JavaScript or how to embed event handlers in your form; CF does it for you. If you want to see the code that's generated, use your browser's View Source or View Page Source menu command. The generated script is a little complicated, but it works and is compliant with all but the oldest browsers.

If the user's input fails whatever validation we've specified in the VALIDATE attribute (or now the optional

PATTERN attribute), the form will not be submitted. The user will receive a JavaScript pop-up message (see Figure 1) telling them what's wrong. The message can either be a default one ("Error in fieldname text.") or one that you specify in the MESSAGE attribute, as I have in the example. The bottom line is that they won't be able to proceed past the form until the validated field passes muster.

One of the things some people don't like about the CFINPUT VALIDATE attribute (both the older built-in validation and the new PATTERN-based one) is that users receive the prompt only when they submit the form. The JavaScript that CF builds isn't smart enough to detect the error when users leave the field. It also shows only one message at a time (rather than showing all validation failures at once). Finally, it isn't smart enough to leave the cursor on the field that contains the error users are being told about (which is all the more frustrating since it's only validating one at a time anyway).

If these things really bother you (and they don't have to, if you're validating only a couple of fields), you should know that you can build your own validation to do those things, if you'd like. In fact, the CFINPUT tag offers an ONVALIDATE attribute, in which you name a JavaScript function (that you must write and include on the page) to do some specific validation. If you use this approach, any value in the VALIDATE (or PATTERN) attribute is ignored, but it does indeed give you greater control. Another alternative is the little known PASSTHROUGH attribute, which allows you to pass virtually any attribute/value pair you'd like (including one that uses JavaScript) to be made available on the generated INPUT tag.

Still another thing that dissuades some people from using CFFORM is that they're afraid of its use of Java applets. It's important to clarify here that CFINPUT only uses JavaScript for validation – no Java applets. CFTEXTINPUT, on the other hand, like several CFFORM subtags, does rely on a Java applet. The applet often provides enhanced control over the user interface (in fact, Release 5 of CF offers many new enhancements to these Java-enabled features), but using Java on the client does have its disadvantages, so some are afraid to use it.

And despite what some may think, using CFFORM will not always cause the applet support files to load: they're loaded only when you use one of the applet-based subtags. But CFINPUT doesn't use Java – it uses JavaScript. Big difference.

There is still another potential issue: not all browsers support JavaScript (and some users have it disabled intentionally for security concerns). Still, there's no harm in using this feature if JavaScript isn't enabled. The JavaScript built by CFINPUT is written in such a way that if the browser doesn't support JavaScript, it will simply be ignored.

Of course, that means you shouldn't rely solely on the JavaScript validation since data coming from a browser that doesn't support it won't be validated. You need to also validate it on the server. A cool thing is that you can generally use these same JavaScript regular expression patterns in the CFML code that processes the form, using them in the REFind or REFindNoCase functions. More about these in a moment.

One last thing to note about our use of CFFORM in the preceding example: I used an empty string for the form's ACTION attribute value. CFFORM requires an ACTION attribute. It can't be left off, and even though in this particular example it doesn't really matter where we submit to, we have to provide one. Since we're only interested in observing the behavior when we try to submit it, I left the action blank so the page will just call itself. Of course, you should name the action page of whatever CF page will process the form.

About Regular Expressions

So how about that regular expression? What does "^([_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*\.(([a-z]{2,3})|(aero|coop|info|museum|name)))?$" mean? It may look a little confusing, but with a little knowledge, looking at a new pattern starts to make sense quite quickly. I can't offer a tutorial on regular expressions, or *regexps*, as they're also known, in this article, but I can show you where to learn more.

And what are regexps, anyway? They've actually been around for a long time and extend well beyond ColdFusion (and JavaScript). In fact, they're more strongly supported and more widely used in the UNIX operating system, but they've found their way into many more tools since they provide a powerful way to perform matching against a pattern, whether simple or complex. Obviously, the preceding pattern looks rather complex. There is even an entire book on the subject, called *Mastering Regular Expressions,* by Jeffrey E. Friedl (O'Reilly).

The good news is that Macromedia has provided some bolstered documentation of them just for this specific new feature, in the second section of Chapter 9 ("Building Dynamic Forms") in the *Developing ColdFusion Applications* manual. The old name was *Developing Web Applications with ColdFusion*.

The manual does have some coverage of regexps, as does the CFML Reference, but it's important to note that some of the coverage there is about ColdFusion's supported regexp syntax. What do I mean by that? It's another source of confusion.

ColdFusion Regular Expressions?

Prior to Release 5 (and still supported in CF5), there are several features in ColdFusion (and ColdFusion Studio) that leverage regular expressions already. They include the REFind and REFindNoCase functions, as well as REReplace and REReplace NoCase. Each of these performs a find or replace (case sensitive or not), respectively, using regexps, as do the Extended Find and Extended Replace features in Studio (under the Search menu). The CFLDAP tag's FILTER attribute also uses them.

These CF-enabled features, however, use a regular expression syntax that Allaire built into ColdFusion. The syntax supported is indeed documented in the CF manuals. In CF5, it's the *Developing ColdFusion Applications* manual, Chapter 14, "Using Regular Expressions in Functions." In fact, this documentation is greatly improved over the documentation in previous releases (there was no similar chapter).

Take note, however: this CF-specific version of regular expressions isn't a complete rendering of regular expressions as is used in UNIX or even JavaScript. And that's a key point for the purposes of this article: the regular expressions used in CFINPUT's PATTERN attribute are JavaScript regular expressions, passed verbatim to the client and interpreted on the browser.

But they're similar enough, and as was mentioned, the new documentation does offer a brief rundown (in Chapter 9 of that book) of some typical JavaScript regular expression syntax that might be used in the CFINPUT PATTERN attribute. The discussions are more complete in Chapter 14, and it's not too difficult to apply what's learned in one chapter to the equivalent features described in the other.

So, I'll leave it to your exploration of the docs to learn what the regular expression "^([_a-z0-9-]+(\.[_a-z0-9-]+)*@[a-z0-9-]+(\.[a-z0-9-]+)*\.(([a-z]{2,3})|(aero|coop|info|museum|name)))?$" used above means.

Be sure to carefully test your use of regular expressions in the environment in which you will deploy them. For the most part, if a regexp works in one browser, it will work in all of them. Note that the CF regular expressions (used in the functions) offer a format called POSIX-compliant regexps. These should not be used in the CFINPUT PATTERN since they'll be passed down to the browser and my testing suggests they're not supported in JavaScript (at least they weren't in my Internet Explorer 5.5).

Some Advanced RegExp Tips

I'd like to add a couple of comments for the benefit of more experienced regexp developers. First, note that I have *not* used the opening and closing slash ("/") that you may be used to using to delineate an expression. That's because, if you look at the JavaScript created by CF, it's putting those in for you. If you add them as well, it will make the pattern fail. Forewarned is forearmed.

Second, note that I've wrapped the entire pattern (other than the start/end characters, ^ and $) in parentheses. More important, I've followed that closing parenthesis with a "?". This is important as it indicates that the pattern being sought can occur either 0 or 1 time.

If you leave that off, then you'll perhaps inadvertently indicate that an e-mail address is required. If none is entered, the pattern won't match and the validation will fail.

You may indeed want to require that a value be entered, but there's an available REQUIRED attribute for the CFINPUT that will make it more clear to someone looking at your code that you're requiring an input value. I just mention this because if you didn't think to add it (the ()? around the pattern), it would fail even if no value was entered and you may be quite confused by that. (Okay, if you're really experienced with regexps, maybe not!)

Finally, keep in mind that in the case of both kinds of regexps (used in the CF functions or in CFINPUT's PATTERN attribute), it's certainly acceptable to use ColdFusion variables and other expressions to create the values to be used in the regexp. The CF expressions are interpreted by CF before the regular expression is then interpreted, so you can use form and query variables, and so on, to help build your regexp.

Validating Phone Numbers

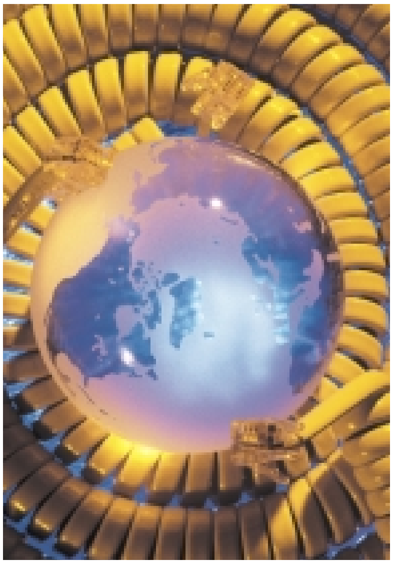
I mentioned at the opening that the built-in VALIDATE="telephone" is limited in some ways (forcing the user to provide an area code and not allowing parentheses around the area code). Here's a pattern that gets around those problems and still adds some other value:

```
^((\(?[1-9][0-9][0-9]\)?)?\s?[1-
9][0-9][0-9]\s?\d{4})?$
```

It, too, isn't perfect. It doesn't support international area codes, nor numbers formatted in other than the typical U.S. format for phone numbers of three-digit area codes, three-digit exchanges, and four remaining numbers. But while it does allow parentheses, dashes, and spaces, it doesn't require them. More experienced regexp users may wonder why I didn't just use "\d{3}" for the pattern to search for the three digits of the area code and exchange (the first three of the last seven numbers). The problem is that this wouldn't ensure they didn't start at zero, which is another rule for U.S. telephone numbers (for now, anyway).

Some Points to Remember

- The new feature in CF5 is the value VALIDATE="regular_expression" for CFINPUT and CFTEXTINPUT. The other values for VALIDATE have existed for some time.
- Using CFINPUT within CFFORM doesn't introduce the use of a Java applet. It just creates JavaScript code for us.
- The JavaScript is written in such a way that if the browser doesn't support JavaScript, it will simply be ignored.
- You can use the same JavaScript regular expressions (for the most part) in the CFML that's processing your form on the action page using the REFind and REFindNoCase functions.
- CF defines its own regexp syntax (a subset of the complete regexp syntax) for use in REFind and REReplace and their derivative NoCase functions (and in Studio Extended Find/Replace and CFLDAP FILTER).
- The regexp used in the PATTERN attribute, however, will be interpreted by the browser and can therefore use all the regexp syntax that JavaScript supports.
- Don't provide an opening or closing slash (/) in the PATTERN value. CF will add that for you before sending it to the browser.
- Consider creating the pattern in such a way that the value isn't required, using "()?" to surround the pattern.
- You can use CF variables and expressions in building regexps.



Closing Thoughts

As I noted before, you don't need to understand these patterns in detail to benefit. Just go ahead and use them.

Maybe eventually there will be a repository of patterns that can be used in a variety of ways, including the PATTERN attribute, the RE functions, Studio, and even more fancy user-defined functions, such as the IsEmail from which I got the example e-mail pattern.

Better still, perhaps you'll be motivated to create other patterns to solve common problems, to offer to such a repository, or to simply solve your own challenges. I hope this quick tutorial on a new way to use them in CF5 motivates you to consider them, or at least be able to benefit from those who do.

Additional Resources

1. To learn more about adding your own validation, see Selene Bainum's article, "Extending CFFORM with Customized JavaScript Validation" (***CFDJ,*** Vol. 2, issue 8) at www.sys-con.com/coldfusion/article.cfm?id=141.
2. To learn more about JavaScript in general, see my article, "Getting Focus(ed)" (***CFDJ,*** Vol. 2, issue 6) at www.sys-con.com/coldfusion/article.cfm?id=122.
3. To learn more about doing form and input validation with CF, see Kailasnath Awati and Mario Techera's article, "Some Thoughts on the Design of CF Data Input Applications" (***CFDJ,*** Vol. 3, issue 3), at www.sys-con.com/coldfusion/article.cfm?id=234.

@ CAREHART@SYSTEMANAGE.COM

ABOUT THE AUTHOR

Charles Arehart is a certified Macromedia trainer/developer and CTO of SystemManage. He contributes to several CF resources, is a frequent speaker at user groups throughout the country, and provides training, coaching, and consultation services. He is also a columnist for ***Java Developer's Journal.***

The Mommy of All **OO Tutorials**

An off-center approach to objects

BY
HAL
HELMS

For some time, Macromedia has outlined its vision of ColdFusion interoperating seamlessly with Java.

This raises some very interesting, exciting possibilities that ***CFDJ*** has invited authors to explore this month.

My own experience with object-oriented programming is somewhat unusual, as my first programming language was Smalltalk.

When people speak of OO languages, they often separate them into "pure" OO languages and "hybrid" languages. Smalltalk is a pure OO language – everything in Smalltalk is an object – while C++ is a hybrid language, mixing OO methodology with procedural programming concepts. The idea behind hybrid languages was that someone coming to objects from procedural programming would have an easier time if there were familiar landmarks along the way. I say *was* because, for the most part, this idea has been abandoned. This was probably a wise decision. After all, science fiction literature is littered with cautionary tales of the dire consequences of combining different types, the common moral being that it's definitely not nice to mess with Mother Nature.

At any rate, the Java creators have made a strong commitment to object-oriented design philosophy. This month I'd like to take a closer look at that philosophy, as – more than any other language I know of – the design philosophy greatly impacts *how* developers write code.

Object Basics

The fundamental building block for OO languages like Java is this mysterious thing called an *object*. Actually, not so mysterious. You've been aware of objects for about as long as you've been aware of anything. As a child, you gradually became aware of a Mommy object. This Mommy object had certain distinguishing characteristics: she was warm, soft, and big. As you came to learn, the Mommy object could actually *do* things – things like feed you, burp you, talk to you, and clean you. To the naked eye you may have appeared to be doing nothing more than drooling, goo-gooing, and eliminating, but you were actually mastering the key concept of OO languages, you clever person, you!

An object is a software simulation of something in the external world. That "something" may be a person, a thing, even an idea, but all objects share the same traits as your Mommy object: they have characteristics (or properties) and they have certain behaviors – things they can do. An object combines properties (data) and behaviors (functions) to produce a simulation. Object-oriented languages provide support for objects by allowing you to create objects.

Classes

Of course, nothing in life – or at least in programming – is *that* simple, so there are a few other concepts you'll need to learn. Still, this is nothing terribly new. As you got older, you found that your Mommy object was not the only one out there. In your first experience with *naming collisions*, you found that both you and the funny-looking kid down the street called your respective Mommy objects "Mommy." And the little red-haired girl had a Mommy object too. As you met more people, it seemed that almost everyone had a Mommy object and that, despite their obvious differences, they all had certain common traits and behaviors. You had, in fact, just discovered object *classes*.

A class is a blueprint – a set of instructions for building specific objects. All Mommies, it turns out, are produced from a central Mommy class. Mommies come off the assembly line (so to speak) with a whole set of properties and behaviors. I've managed to smuggle out the specifications for a Mommy object and while it's too large to reproduce in its entirety, I include snippets of it here.

Class: `Mommy`

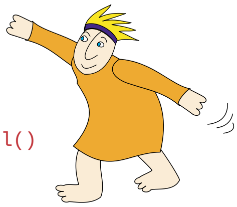
Superclass: `Parent`

Properties:

```
firstName
lastName
mood
…
numberOfActiveMommies
```

Behaviors:

```
loveChild()
feedChild()
cleanChild()
buyToys()
takeChildToSchool()
…
sayNoToChild()
```



Notice that some properties, like numberOfActiveMommies, belong to the Mommy class, but not to a specific Mommy. These are often called *class variables*. Behaviors are often called *methods*, and methods that belong to the Mommy class, but to no specific Mommy, are called *class methods*.

As you begin to take in more of your world, you realize that there is another large, warm object hovering about. It's like a Mommy object, and yet not like a Mommy object. You try out a number of names for this object, but the one the object responds to best is "Daddy." Mommy and Daddy have many things in common; as a budding software architect, you realize that such situations often create duplicate code – something you sense should be avoided.

And so, as you're watching Big Bird reveal the mysteries of the number 3, it strikes you: you will come up with a new class, something that encompasses all the commonalities between the Mommy class and the Daddy class. You decide to call it the *Parent class*. (Later you'll shorten this to "the 'rents.")

A Mommy object is a specific kind of Parent object. A Daddy object is a different kind of Parent object. In discussing this with the funny-looking kid down the street, he suggests that Parent is a *superclass* to both Mommy and Daddy. He explains that superclasses are like parents to other classes. It's a funny-sounding name from a funny-looking kid, but it'll do.

Instances

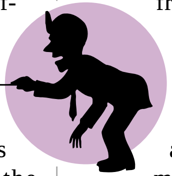
While classes as blueprints are interesting, what we really want are objects. After all, the concept of a parent never paid anyone an allowance. But if we have a class defined, we can (almost) always produce an object from the class. This is known as creating an *instance* of a class. This explains a lot, but not why your Daddy object tells you to ask your Mommy when you ask how you were instantiated. Still, you understand that when Mommy and Daddy tell you that you're going to have a new baby sister, what they really mean is that a new object will be instantiated from the class, Daughter, whose superclass is Child. Why, you wonder, don't adults just say what they mean? It's something you'll wonder about for a long, long time.

Messages and Methods

One central feature of objects is that they're separate entities and are entitled to their privacy. While you were very young, if your Mommy wanted to know how you were feeling, she would do things like feel your forehead and listen to your breathing. Later you developed the method speak(), and she was able to ask you, "How are you feeling today?" In object-speak a Mommy object sent a *message* to a Child object.

Messages are how objects communicate with each other. If one object wants to know about another object, it does so by asking the object to perform some action – even if this action is nothing more than asking it to retrieve one of its properties. Objects then have *methods* they use to respond to those messages. If you think of messages as subroutines or functions, you'll get the sense of it.

This allows the object to manage its own affairs and provides a layer of protection to the object's integrity that direct manipulation would not. It also lets the object respond differently depending on where the message is coming from. (This may explain the universal experience of parents who, when asking their child how she's feeling on a test day, receive the answer, "I think I'm sick," while no child in all of recorded history has ever had a fever on school holidays.)



The OO Trinity

ColdFusion isn't an object-oriented language, of course. In general, OO languages support three practices: *polymorphism, inheritance,* and *encapsulation,* summed up in the acronym *PIE*. I'll explain each of these in reverse order.

Encapsulation

What would we do without buzzwords? Life just wouldn't be the same. *Encapsulation* refers to the fact that both properties *and* behavior of an object are wrapped up together (encapsulated). Unlike most procedural programming models, data is not separate from functions. An object is a "black box" that accepts message requests and, if it's in a good mood, responds to you. The set of messages an object responds to constitutes its *public interface*. In addition to this, it has a private set of methods that it alone can call on.

Inheritance

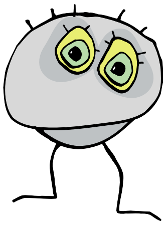
Object classes have parents or superclasses. You saw earlier that the Mommy class has a superclass of Parent (making Mommy a subclass of Parent). This means that every Mommy object produced from the Mommy class blueprint inherits both the properties and the methods of Parent. The *inheritance* tree is determined by the architect; there is no "right way" to construct this, and as your expertise in OO grows, you'll likely find your inheritance tree changes quite a bit, too.

In rummaging about in my attic, I found an old inheritance tree I constructed as a tot. (Luckily, I've always documented my code meticulously!) It's in Figure 1, showing how I modeled my world way back then.

Note that each class can inherit from only one superclass. This is known as *single inheritance*, and is the model adopted by Java. A few other languages allow a class to inherit from more than one superclass. I would have liked this model of multiple inheritance when modeling my teacher, Mrs. Sorenson, who clearly belonged to both the Teacher and the Monster class, but multiple inheritance has proven to be very unwieldy.

Polymorphism

Polymorphism is tied to the concept of inheritance, and it's the toughest piece of the PIE to grasp. Stated as simply as I can, polymorphism allows Mrs. Sorenson to tell me, "Give this note to your parents, young man!" – knowing that the Hal object, by virtue of inheriting from the Child class (which implements the method giveNoteToParents()), will know how to give a note to his parents. But, hey, nothing's keeping me from overriding that method in the Hal object!



A Sea Change

Moving to OO programming is a fundamental change in thinking about programming. It's not that it's more complex – in fact, Smalltalk was written for fifth-graders to understand – but if you're new to it, it's quite a switch. In fact, many people have discovered that people new to programming adapt to OO concepts easier and faster than seasoned veterans who must "unlearn" much of the way they approach programming.

In addition to the new concepts to be mastered, most OO languages (including Java) have a number of built-in classes that need to be mastered. When I was working with Smalltalk, we used to say that it took about a year for a new Smalltalker to stop rewriting the existing classes. I think the most helpful advice I can offer is that if you're going to tackle Java, you need to think of Java not so much as another language (obviously, it is that), but as a way of building models that simulate the "real world." Only you as the model maker determine how to organize that real world.

It's often helpful to see common ways of tackling problems. Such solutions are often called *software patterns* and there are books devoted to this aspect of OO programming.

No Panacea

The philosopher of gloom, Arthur Schopenhauer, once remarked, "All truth passes through three stages. First, it is ridiculed. Second, it is violently opposed. Third, it is accepted as being self-evident." I missed the first stage in OO's march toward ascendancy, but was there for stage 2, so it strikes me as ironic that OO has finally gotten respect. Often, though, the pendulum swings dramatically in *both* directions, and today the conventional wisdom is that OO is the only way to approach a problem.

Coming to ColdFusion *from* OO programming has given me a different perspective. Despite OO programming's near-universal acceptance, problems lurk under the surface. Gertrude Stein once wrote about Oakland, California, "There's no *there* there." OO programs can suffer from this. It's hard to see the forest, but you sure can see a lot of tree objects. This often means that as the system becomes more complex, the knowledge required to make changes to the system is substantial. Since so much of a project's life cycle involves maintenance, this can be a burden on both senior programmers and the organizations that employ them.

As I stated earlier, the learning curve for an OO programmer is usually very long, and moving to OO programming is a very large commitment. There should be better reasons to do it than its being fully buzzword compliant. Finally, OO development in distributed environments is difficult due to the great amount of interaction between objects. This type of development seems to work better in small groups, as shown by the interest in *extreme programming*, a development methodology built to deal with some of these problems.

Resources

One of the great promises of ColdFusion 6.0's adoption of a Java-based model is that it can act as a sort of front end to Java, reducing the complexity of building Java programs while offering new power and interoperability with other, Java-based development initiatives. So, while it behooves us all to learn more about Java, you don't need to abandon the ease of use and speed of development that we all love in ColdFusion.

Web Sites

- *Java tutorial:* http://java.sun.com/docs/books/tutorial/
- *OO concepts using Java:* http://java.sun.com/docs/books/tutorial/java/concepts/
- *Simple introduction to OO concepts:* http://catalog.com/softinfo/objects.html
- *Extensive discussion of OO and Java:* www.objectcentral.com/oobook/oobook.htm
- *More on OO concepts:* http://developer.java.sun.com/developer/onlineTraining/Programming/BasicJava2/oo.html
- *"A friendly place for Java greenhorns":* www.javaranch.com/index.jsp

The site that has the "Two Ralphs Java Newsletter" is www.javatrainingcenter.com. One of the Ralphs is Ralph Fiol, a good friend and creator of CFObjects. In fact, if you'd like to work with OO concepts in the comfort of ColdFusion, you should check out www.cfobjects.com. Ralph has done an amazing job of bringing OO concepts to the ColdFusion world.

Books

- **Object Technology: A Manager's Guide** *by David Taylor:* This thin, extremely well written book is the best introduction to objects I've ever seen. It's geared for people with no knowledge of object orientation.
- **On to Java** *by Patrick Henry Winston:* This book, by a professor at MIT, is simply the best book on Java I've read. In only 360 pages, Patrick Henry Winston somehow manages to perfectly blend an extemely readable style with exactly what you need to know. Don't be intimidated by Winston's credentials; the book's style is extremely approachable. If you want to learn Java, buy this book.
- **Thinking in Java** *by Bruce Eckel:* I'd read this one after you've tackled the first two.
- **Java Objects** *by Jacquie Barker:* An excellent book that provides a very good foundation in OO programming as well as Java syntax. If you buy only one book, this might be the most comprehensive.

ABOUT THE AUTHOR

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

@ HALHELMS@TEAMALLAIRE.COM

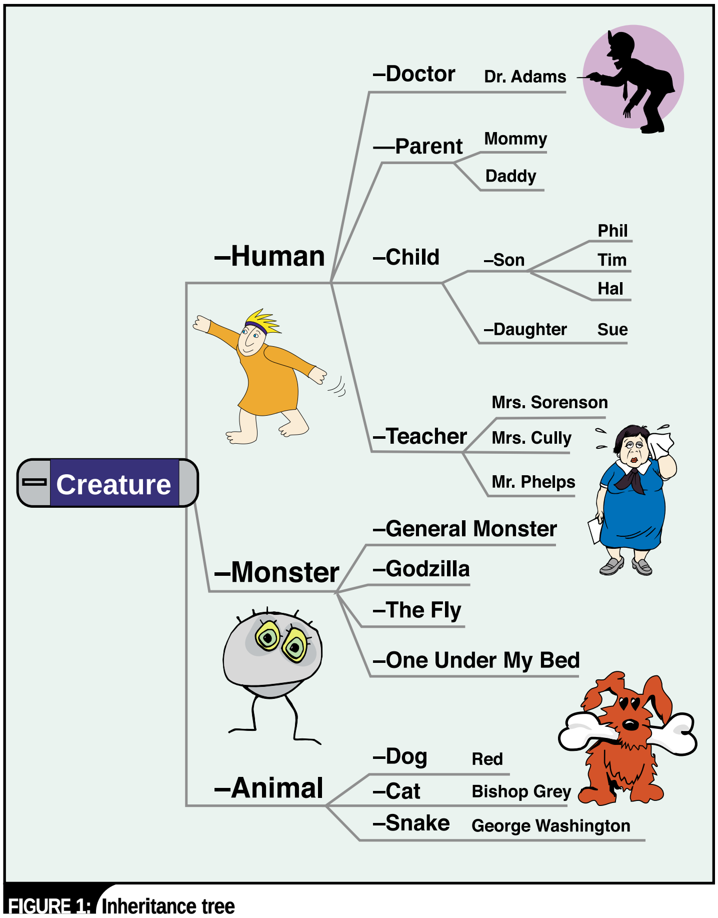
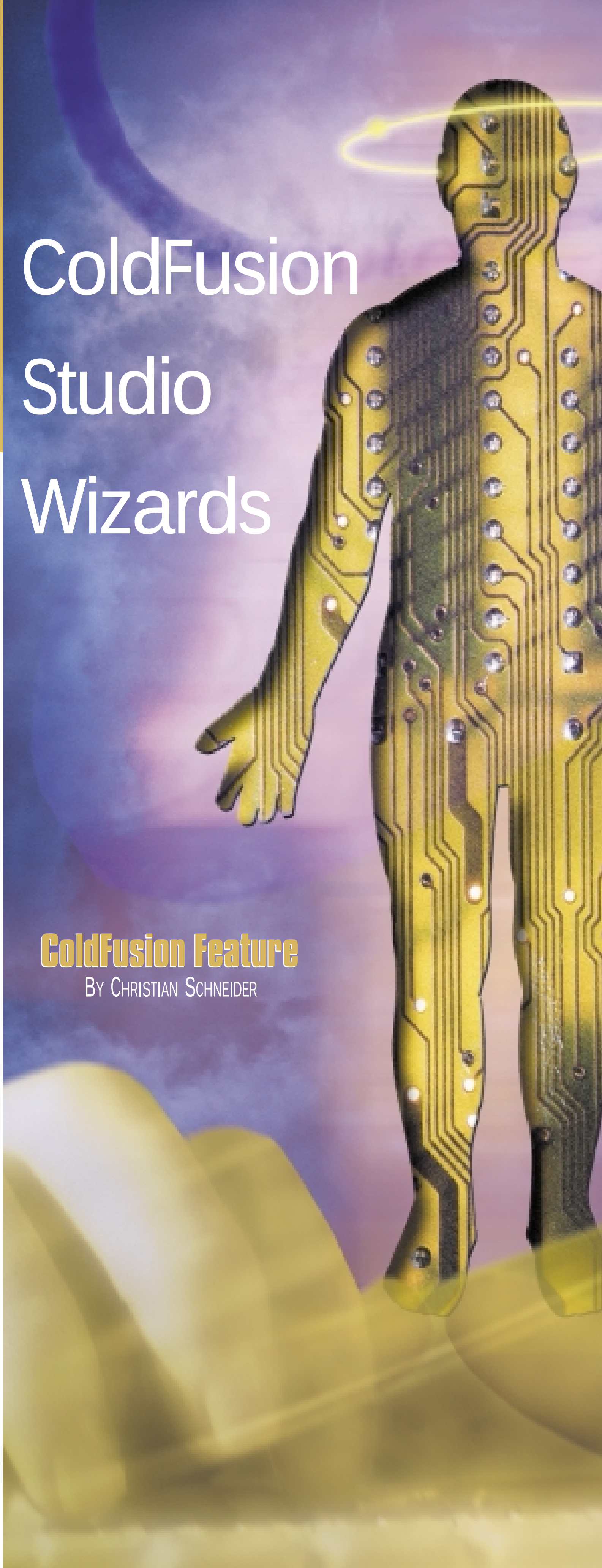


FIGURE 1: Inheritance tree

WIZML by Example – ColdFusion Studio Wizards



ColdFusion Feature

By Christian Schneider

*This article is the third in a series on customizing the CF Studio IDE. If you haven't read the first two about VTML, see "VTML by Example, Parts 1 and 2" (***CFDJ***, Vol. 3, issues 6 and 7) for an introduction to the CF Studio extensions, and learn how to use the Visual Tools Markup Language (VTML) to develop the user interfaces for CF Studio extensions. This article demonstrates how to design CF Studio Wizards using the Wizard Markup Language (WIZML).*

What Are Custom Wizards?

Custom wizards are step-based dialogs inside CF Studio that ask for user data and solve special tasks based on this data. Put simply, CF Studio Wizards are a series of dialogs that the user goes through sequentially, and when finished generates a new template. Select "New..." from the File menu for a list of installed wizards to choose from (many helpful wizards are available).

The Wizard Markup Language

WIZML, a tag-based markup language, makes custom wizards possible by adding flow control and input processing to VTML-based dialogs.

A wizard consists of a single wizard definition (also known as a wizard profile) file (named .vtm) that defines all pages, user input controls, output parameters, and one or more wizard output template files (named .wml) that use these parameters to generate code. Basically, you have to write a .VTM file that controls the user interface, input parameters, and logical flow of the wizard's steps. Then you define the resulting output pages the wizard generates after the user finishes inputting all data. Each output template file must be named .WML (for Wizard Markup Language) and contain the code the wizard should generate.

These .WML files are like templates in the sense that you can output all WIZML variables using the $${VariableName} notation. But more on this later. Keep in mind that you'll need to develop two types of files: a .VTM file for controlling the user interface and logical flow of the wizard, as well as at least one .WML file that's used as an output template.

You can deploy a custom-made wizard by placing its .VTM and .WML files in CF Studio's \Wizards\Custom folder, or create your own folder under the \Wizards directory. From now on you can use the freshly deployed wizard by choosing "New…" from the "File" menu.

The Wizard Definition File (.VTM)

An outline of a wizard .VTM file is shown in Listing 1. As you can see, each wizard .VTM file has the <wizard> tag as its root tag. Inside it holds a declarative section (all the <param> tags at the beginning) where you can set up the WIZML variables you'll output in the resulting template. This declarative section is followed by <page></page> blocks that define each wizard page by describing the page's layout inside the <pagelayout> tag and binding the input controls (e.g., textboxes, etc.) to WIZML variables using the <input> tag. Define each wizard step that way, so you'll end up with multiple <page> sections. At the end of the .VTM file you'll notice at least one <template> tag that simply assigns which output .wml templates to process after the wizard's steps are completed by the user.

Now we'll discuss the details of developing a wizard .VTM file. I'll guide you through the source code of a real-world wizard I developed. This wizard helps developers who don't have the in-depth knowledge of WDDX and JavaScript to create cool, dynamically synchronized select boxes for a parent-child relation, such as browsing a set of items within a set of categories. To understand how the wizard is built, read on. To understand what the wizard actually does (creating the WDDX and JavaScript code for the synchronized select boxes) read my article "WDDX with JavaScript" in ***CFDJ*** (Vol. 3, issue 2).

Let's assume you're primarily interested in how to build such a wizard, so I'll leave the explanation of WDDX and JavaScript aside. As you can see from Figures 1–4 the wizard prompts for some data, such as the data source, the tables, and columns to populate the synchronized select boxes to make them dynamic.

Looking at the File in Detail

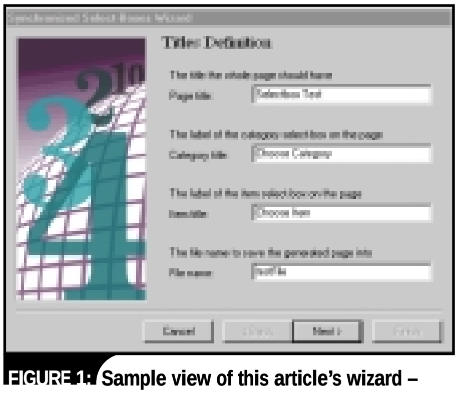
Let's start looking at the code. The Wizard Definition File (syncSelect.VTM) is available on the ***CFDJ*** Web site (www.sys-con.com/coldfusion/sourcec.cfm).

At first the root tag <wizard> needs a name, a caption, as well as (optionally) an image to display (see Figures 1–4). You can create your own images and distribute them with your wizard (they must be saved in the \wizards\images directory in BMP format and be 138x293 pixels). A nice pool of preshipped images is available with any CF Studio installation (see existing images in the wizard\images directory).

Next define all WIZML parameters that will be populated from the wizard definition file into the output templates (.wml) and are invoked after the user has completed the steps of the wizard. You can define them using the <param> tags similar to the following:

```
<PARAM name="theCategoryTitle"
value="" required="true">
```

As you can see, aside from initializing them with default values, you can set them as required or as optional parameters.



Synchronized Select Boxes Wizard

Titles Definition

The title the whole page should have

Page title: Selectbox Test

The label of the category select box on the page

Category title: Choose Category

The label of the item select box on the page

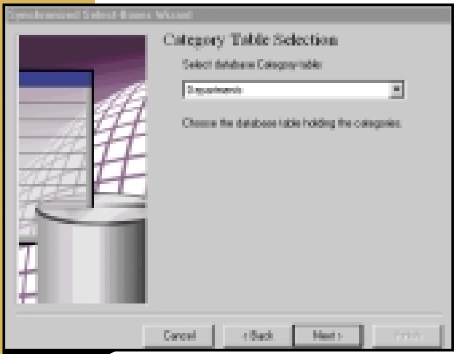
Item title: Choose Item

The file name to save the generated page into

File name:

FIGURE 1: Sample view of this article's wizard – step 2

How to successfully extend the ColdFusion Studio IDE part 3



Synchronized Select-Boxes Wizard

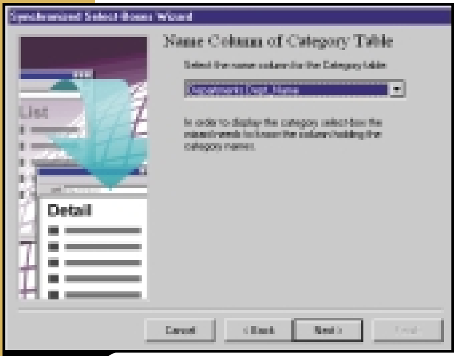
Category Table Selection

Select database Category table:

Departments

Choose the database table holding the categories.

FIGURE 2: Sample view – step 5



Synchronized Select-Boxes Wizard

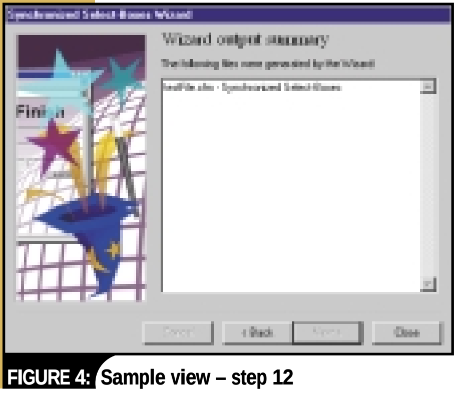
Name Column of Category Table

Select the name column for the Category table

Departments.Dept_Name

In order to display the category select-box the wizard needs to know the column holding the category names.

FIGURE 3: Sample view – step 7



Synchronized Select-Boxes Wizard

Wizard output summary

The following files were generated by the Wizard

testFile.cfm - Synchronized Select-Boxes

FIGURE 4: Sample view – step 12

After you've defined all your WIZML variables, set up the logical flow by defining the wizard's pages. This is done by placing blocks of <page> tags in the .VTM wizard definition file. Basically, you have two types of pages available: custom-made pages where you directly place the input controls on the page, or predefined pages taken from CF Studio's library of predefined wizard pages. These predefined wizard pages come in many flavors such as data-source choosing dialogs, and table and column selection pages.

First let's look at the custom-made pages and discuss the different predefined flavors later on. To prefix these two sections, both types of pages need a name as an attribute as well as a caption and an optional image reference. Custom-made pages consist of both a page layout and a variable binding section.

Custom Page Layout

All custom-made pages have the attribute type="DYNAMIC" in the <page> tag to define it as a custom page.

To design such a page place input controls (e.g., labels, text fields, select boxes, etc.) on the wizard page inside the <pagelayout> block. For details on which input controls VTML offers and how to place them inside the <pagelayout> block, consult Part 1 of this series about VTML. Using VTML controls in TagEditors is the same as in wizards, so this article should shed some light there.

Variable Binding

To use data that was entered in a VTML input field within the .wml output template, bind the VTML input control to a WIZML variable in the wizard definition file. This is done at the end of each wizard <page> block using the VTML tag <input>. In the WIZML code snippet (see Listing 2) is a simple wizard page with a label and a text box that's bound to the WIZML variable, theUsername.

As you can see, the text field is rendered using the <control> tag inside the <pagelayout> block, and it's bound to a WIZML variable using the <input> tag. Also the variable is made mandatory by setting the "required" flag to "yes," and a validation error message is provided that pops up when the user wants to proceed with the next wizard step and no text was entered in that variable. That way you can ensure that all required fields will be entered by the wizard's users. To make use of this type of page design you should read Part 1 or at least look at the Help System of the CF Studio IDE regarding the VTML layout.

Predefined Wizard Pages

Let's look at which prebuilt wizard pages CF Studio has available for our use. For the many common tasks of a wizard page, CF Studio has prebuilt pages that we can use without worrying about the VTML layout of this page. The predefined wizard pages we can use are:

- SelectNameAndLocation
- SelectDataSource
- SelectTable
- SelectTables
- SelectField
- SelectFields
- SelectTableJoins

Except for the first one, all predefined wizard pages solve common tasks when databases come into play, such as choosing data sources from the remote CF server, and selecting tables, fields, and even joins from live data. Compared to custom-made wizard pages, you can invoke these predefined ones quite easily (see Listing 3).

Set the <page>'s "type" attribute correctly and fill the predefined parameters with your own text using the <param> tag; for example, customizing the prebuilt page. Then bind the resulting values to WIZML variables using the <input> tags as described earlier. For a complete list of all prebuilt wizard pages and their use consult the CF Studio Help.

Logical Page Flow

By default, all pages are shown in the order in which they're defined in the wizard definition file using the <page> blocks. If you'd like to change this order, append the attribute nextpage="name OfNextPage" on the <page> tag to specify the next page to show. You can even go further and define a Boolean condition (that evaluates as true or false) to put into the <page>'s condition attribute, for example, condition="some-Variable EQ 23". If the <page> tag has such a condition assigned, this page is displayed only if the condition evaluates to true, otherwise the next page will be shown.

Output Templates (.WML)

We've successfully built a wizard definition file (.VTM) to render the wizard's user interface and logical flow. But this alone does not make a wizard, since the data entered by the users should be processed in a way to generate output

> "WIZML has tags to allow decision-making logic, looping, variable assignment, file including, and more to be coded inside a .wml template"

files. Basically this is very easy, since the output files of a wizard are named .wml and registered at the end of the wizard definition file using the <template> tags:

```
<TEMPLATE
    name="syncSelect.wml"
    outputfile="$${theFilename}.cfm"
    description="Synchronized
Select-Boxes"/>
```

This tag defines a .wml output file to be processed (syncSelect.wml) and saved into a .cfm file after the wizard has finished its processing. The “outputfile” attribute of this tag names the resulting file that should be generated. Though I’m using a WIZML parameter to let the user determine the exact filename, you could also code the name directly into the outputfile attribute.

Output templates are processed by the CF Studio wizard engine and saved in the specified .cfm filenames. Upon processing, the CF Studio wizard engine understands WIZML syntax, so you can output all WIZML variables from the wizard definition file inside the output template. This is done using the $${VariableName} notation. There’s even a set of WIZML tags and functions available for use in an output template. A simple output template (.wml) file would look like this:

```
<b>Hello $${theUserName} it's now
<cfoutput>#DateFormat(Now())#</cfout-
put><b>
<wizif Len(theImageFile) GT 0>
 <img src="$${theImageFile}" bor-
der="0">
<wizelse>
 [no image set]
</wizif>
```

Here you see the two WIZML variables “theUserName” and “theImageFile” being used, as well as a WIZML tag for conditional processing (<wizif>) and the WIZML string function Len(). As you can see, WIZML has tags that allow decision-making logic, looping, variable assignment, file including, and more to be coded inside a .wml template. For a detailed reference description of these WIZML tags as well as the available WIZML functions for string processing see the CF Studio Help.

In the .wml wizard output template (syncSelect.WML) of this article’s wizard (available on the ***CFDJ*** Web site) there’s a block of code in which WIZML string functions are used. This is often necessary when using the results from prebuilt wizard pages, since they return their values in strings that often need to be parsed. In the .wml wizard output template the results from our table-choosing wizard dialogs are parsed to get the table name.

All WIZML tags and WIZML functions are processed by CF Studio's wizard output generator, and the rest (which may, of course, include CFML tags) is simply written into the resulting page. You can enable WIZML to work with CFML the same way CFML works with HTML: using CFML you can dynamically create HTML code. While using WIZML you can dynamically (i.e., created by a wizard) generate CFML/HTML pages.

VTML and WIZML provide developers with tools to extend the IDE's functionality using <tags>. For further information on VTML and WIZML see the chapter "Customizing the Development Environment" on CF Studio's Help. This chapter also has a nice VTML/WIZML reference section available.

About the Author

Christian Schneider is a Macromedia Certified Advanced ColdFusion and Web site developer. He has over four years of intensive experience developing CF-based intranet applications for banks and logistic corporations.

@ MAIL@.CHRISTIAN-SCHNEIDER.DE

Listing 1

```
<WIZARD>
 <!--- wizard parameters --->
 <PARAM/>
 <PARAM/>
 <PARAM/>

 <!--- first page --->
 <PAGE>
    <PAGELAYOUT>
         <!--- user interface controls --->
         <CONTROL/>
         <CONTROL/>
         <CONTROL/>
     </PAGELAYOUT>

    <!--- binding control-variables to WIZML-variables --->
     <INPUT/>
     <INPUT/>
     <INPUT/>
 </PAGE>

 <!--- second page (more pages) --->
 <PAGE>
        <!--- (like above) --->
 </PAGE>

 <!--- output templates (.wml files) --->
 <TEMPLATE/>
 <TEMPLATE/>
 <TEMPLATE/>
</WIZARD>
```

Listing 2

```
<PAGE name="UserPage" type="DYNAMIC" caption="Enter User"
 image="..\\images\\main.bmp">
    <PAGELAYOUT>
         <!--- Ask for Username --->
         <CONTROL name="lblUserName" type="label" down="10"
          right="10" width="90" caption="User name:"/>
         <CONTROL name="txtUserName" type="TextBox"
          anchor="lblUserName" corner="NE" width="MAXIMUM"
          down="-5"/>
    </PAGELAYOUT>

    <!--- Binding control-variables to output-template-
      variables --->
    <INPUT name="txtUserName" param="theUserName" required=
     "yes" validationmsg="Please enter a user name">
</PAGE>
```

Listing 3

```
<PAGE name="DataSourcePage" type="SelectDataSource"
 caption="Datasource Definition"
image="..\\images\\SelectData.bmp">
          <!--- Fill the pre-defined parameters with your
           own text/data --->
          <PARAM name="ListBoxLabel" value="Select data
            source:">
          <PARAM name="ListBoxDescription" value="Choose
           the data source for this wizard.">

           <!--- Bind the result to a WIZML variable --->
           <INPUT name="cbDataSources"
param="theDataSource" required="yes" validationMsg="Please
 select a data source.">
</PAGE>
```

CODE LISTING

The code listing for this article is also located at

www.sys-con.com/coldfusion/sourcec.cfm

Special Introductory Offer SAVE $13.89*

WebServices JOURNAL

The world's leading independent Web Services information resource

Get Up to Speed with the Fourth Wave in Software Development

Only $69.99 for 1 year (12 issues)
Newsstand price $83.88 for 1 year

- 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
- and much, much more!

THE NEW PROGRAMMING PARADIGM???

WebServices JOURNAL

WSJ2.COM

FULL CONFERENCE COVERAGE web services EDGE conference & expo pg. 44 SEPT. 23–26 IN NYC

A Web Services Success Story DUPONT PROTECTIVE COATINGS by Catherine Marchand

Content Management & Web Services

eContent Services– the services behind Web Services Page 6

Business Basics: Web Services Fundamentals: From Whence It Came A look at future possibilities Phil Karecki 18

WSJ Feature: Web Services: Enabling Technology or New Programming Paradigm? The new integration model Rebecca Dias 22

Business Collaboration: Understanding the Depth and Breadth of the Business Opportunity Revenue growth David Russell 26

Technology: Information Syndicati… Web Services A dynamic way to share… Darren Govoni 34

WSJ Feature: A Web Services Co… A better environment for real-world deplo… …n Bernhard 38

Java: JAXM: Interoperabl… for the I… Chappell 48

…Atoms vs Bits and the Digital Middle Mile Agreement on standards expedites sharing among partners Kurt Cagle 52

UDDI: Understanding the UDDI tModels and Taxonomies The key integration point Noel Clarke

THE BEST SOURCE FOR COVERAGE ON .NET

www.wsj2.com

SYS-CON MEDIA

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

**Offer subject to change without notice*

Code Co-op from **Reliable Software**

REVIEWED BY
DAVID SCHWARTZ

Every developer faces the difficulty of maintaining and tracking changes to program source files, regardless of the platform or language.

It becomes more challenging when working with a team of developers. Code Co-op by Reliable Software is an incredible product that eases this burden and helps you create better programs.

I deviated from the standard product review format and sought out real-world, hands-on users to get objective opinions. Though I tested Code Co-op for three weeks, I felt it wasn't quite the same as using a product over a period of time under many scenarios.

Installation

The current version, 3.1, can quickly and easily be downloaded from the Reliable Software Web site. Yes, it's a free demo. There are no annoying prying forms to fill out – two clicks and the download starts automatically. The entire file is only 3.5MB (refreshing in an age of 300MB demos). Actual installation took a few minutes and I didn't encounter any problems. Once installed, the entire product, including online help, took 6.8MB. Installation is simple and straightforward, so I won't waste your time trying to detail it.

Testing 1-2-3

In a nutshell, Code Co-op keeps track of your source code and helps you share it with other members of your development team. It's a fantastic product and I recommend that every programmer and project manager evaluate it. By the way, if you're the only member of the development team you should still look into it. My only complaint (nothing's perfect) is the skimpy documentation. I would appreciate more detailed feature overviews and step-by-step guides to understanding and using key features.

VITALS

Code Co-op
Reliable Software

Address: 1011 Boren Ave., PMB 206
Seattle, WA 98104

Web: www.relisoft.com

E-mail: info@relisoft.com

Test System:
AMD K6 266MHz PC
128MB RAM
Microsoft Windows 98
Microsoft Exchange Server 5.5
Windows NT Server 4.0

Sounds Like a Salesman to Me

What makes Code Co-op so amazing are two key features. First, it's *e-mail–based* instead of server-based. Second, it automatically tracks all activity – most important, code changes.

Traditionally, code trackers and organizers have been server-centric, meaning all code resides on a central server. Developers who needed to access the code had to somehow connect to the server. This may work if your team is always in the same building at the same time, a rarity in the Internet age. In addition, with the big shift toward "collaborative" development in which many people, not just programmers, are involved in the "blood and guts" coding cycle, more people need to access the code more often. It's nearly impossible to get everyone connected to a central server. Anyone who's used a notebook computer on the road or worked from home knows the frustration of trying to connect remotely to retrieve critical code. I gave up on that framework years ago.

Using Code Co-op your team members only need e-mail access. Yes, it's that simple. All files belonging to a Code Co-op project are automatically e-mailed to team members. You get fresh, up-to-date code wherever, whenever. Considering the ubiquity of the Internet, all restrictions are lifted. I can recall several vacations when clients called in bugs and I couldn't get to the source code to review it in order to offer a temporary workaround.

"Integrating code changes is so easy and quick that we integrate several times a day instead of once or twice weekly. Once you integrate, Co-op instantly sends those changes to everyone on the team, so within minutes everyone has your code changes and automatically updates the source. It's absolutely amazing, seamless, and flawless," states Matt Brown of Ransdell Brown, Inc. (RBI), a custom CAD application and image processing–related software firm in Seattle, Washington. Prior to using Code Co-op Matt became very frustrated with his server-centric system. With Code Co-op he claims to have eliminated lost code and downtime. With offices in Seattle and Spokane, Matt is able to bridge the distance with one tool and no additional hardware or software investment. I should also point out that with server-centric systems all code usually resides in the central repository. If the server goes down, the code goes with it.

With a more diverse and disparate member base, team developers are finding it ever more challenging to keep track of code changes made by members. Code Co-op has an intriguing module called "Synch Area." The synchronization process compares different versions of a file and lists the changes/differences between them. Suppose you modify a code file and save it to the project; meanwhile, another team member modifies the same file – Code Co-op will show you what was changed and by whom.

Tom Kelleher of Tom Kelleher Consulting, Inc., an Internet technologies consulting and Web application design/development firm located in New Jersey, works with teams that consist of three developers who are geographically dispersed. Tom claims that Code Co-op has helped his company deliver better products because it allows his developers to test and experiment with code more freely.

"We are able to roll back changes when a change causes more problems than it was meant to solve. And because we know we can roll back changes like this, we are more confident with making large-scale changes (e.g., changing the spelling of a variable across 30–40 pages). Because we have a history of all changes, both for the project as a whole and for individual files, we can isolate where problems began," says Tom. Needless to say, since developers spend more time hunting down bugs than a Bushman – any help in that area sounds appealing to me.

The Big Payoff

I've covered some features that I believe make Code Co-op shine. There are many others that will help you get organized and work with more reliable sets of code. Since Code Co-op monitors what you do and keeps your files organized it really helps increase productivity. All code is kept in a "Project." You can have many projects and easily choose from a convenient "Projects" tab on the main screen. Tom Kelleher points to the "Projects" tab as one of his favorite features, "I like being able to maintain several projects at once in the interface, and being able to hop nimbly around from one to the next."

In my testing I created three projects for different versions of the same program. It was convenient to be able to switch back and forth between them and test new features.

Summary

While Reliable Software touts Code Co-op as a developers' tool, I see other uses. For example, if you share Microsoft Word or Excel files with colleagues in many locations it can help keep the files organized and everyone up to date. Developers who work solo will also benefit from the e-mail–based code storage. It's perfect for e-mailing yourself code to work on at different locations. After updating your code you can simply e-mail it back to the office, so when you arrive at work the new code is waiting for you. Since the code will exist in more than one place you'll always have a backup (since backing up is the last thing we do – and often forget to do it all together – this feature alone justifies the purchase).

Code Co-op is sold by "developer seat" at an extremely competitive and low fee. I like the model – the start-up cost is low with no huge initial purchase. Indeed, Matt has been using Code Co-op for almost a year. "I couldn't do my job without it," he asserts.

Editor's Note: Since this review, Reliant Software has released version 3.2, with a file size of 2.03MB. The command-line interface is a separate download at 1.8MB in size. (See www.relisoft.com for more information.)

@ DS@ARRAYONE.COM

ABOUT THE AUTHOR

David Schwartz is the president of Array Software Inc., a New Jersey-based software company. Array creates global data-driven Internet and intranet Web sites using ColdFusion, Oracle, MS SQL Server, and Java. David has been developing turnkey custom database software for 14 years.

Ask the **Training Staff**

A source for your CF-related questions

BY
BRUCE VAN HORN

Before you read this article look at your calendar. That's right: it's December already! Where did 2001 go so fast?

The first year of the new millennium was definitely not one of those "ho-hum-nothing-much-happened" years. It was a busy, exciting, turbulent, and tragic year for most of us. Here's hoping that 2002 will be just as busy and exciting but without so much turbulence and tragedy. Happy holidays to you and your loved ones!

Partial Retraction

I don't have any questions to address this month, but I'd like to offer a slight retraction (actually, just a word of caution) of last month's article in which I talked about sparing yourself some of the hassles of locking every reference to shared variables (those in the Server, Application, or Session scopes) by copying them into and out of the Request scope.

This subject is covered in the Advanced ColdFusion class and is offered as an acceptable and even recommended alternative (I hesitate to use the phrase "best practice"). However, after talking with several other trainers and developers and doing some actual testing, I'd like to caution you about using this method of handling shared variables and suggest that you *not* use it in most cases.

While it certainly makes writing applications easier by not having to lock every single reference to a shared variable, and while it may make logical sense that it would improve your application's performance to have only two locks (one in Application.cfm and one in OnRequestEnd .cfm) for each page rather than opening and closing many locks, this turns out not to be the case. While many applications will perform just fine by copying shared variables into the Request scope, any application that stores a significant amount of data in those scopes will see a dramatic performance loss.

> “I’d like to caution you about using this method of handling shared variables and suggest that you *not* use it in most cases”

The bottleneck actually comes while making a copy of the data using the Duplicate function. The larger the amount of data, the longer it will take to copy it to/from the Request scope. Checking or creating locks doesn’t take a considerable amount of time nor does it create significant amounts of server load. Copying data from one scope to another for every single user on every single page, however, will create performance problems as the data grows.

Another potentially problematic issue with copying data out of shared variables into the Request scope (which is a local variable) is memory management. Suppose you have a query result stored in the Application scope and you're using the duplicate function to copy it to the Request scope for each user. Well, you've just made a separate copy of that query for each user on your site. If it was a large query and you have many users, you could potentially run out of RAM on your server.

With these problems in mind, I would bet the next revision of the Advance ColdFusion class will not include this method as an example of how to use the Request scope to avoid having to lock every reference to shared variables.

Therefore, if your application doesn’t get a large amount of traffic and only stores simple data elements (Session.LoggedIn = 1, etc.) in the shared variable scopes, you may still want to consider copying the data to the Request scope as suggested last month. However, if you’re storing queries, arrays, lists, or other structures in these scopes, you’ll be better off writing explicit locks around each variable reference.

• • •

I look forward to hearing from you in 2002! Please send your questions about ColdFusion (CFML, CF Server, or CF Studio) to AskCFDJ@sys-con.com. And please visit our archive site at www.NetsiteDynamics.com/AskCFDJ.

ABOUT THE AUTHOR

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

@ BRUCE@NETSITEDYNAMICS.COM

2001: A CF Odyssey

by Eva Holtsmark

CF Squares game

I spent the weekend of July 28–29, 2001, learning ColdFusion with other programmers at the TeraTech CF2001 Conference. Why, you might ask? To learn some neat CF ideas, talk with my peers, and have fun!

Saturday

We began with a talk on disabled access called “ColdFusion Accessibility and Section 508” by Michael Smith and Jon Brundage. Next up was “Flash and ColdFusion” by Mike Chambers, using the Flash/SQL component kit <CFX_FlashSQL>.

“Building Windows Applications with HTML and CF” by Steve Nelson followed; in dramatic style he showed us how to write Web apps without an Internet connection using a Sash Weblication.

After lunch we took a break for the CF Squares game show. Out of nine squares, I was square number three. One question was: What is the maximum number of arrays allowed? All I had to do was push out my paper card with the number 3 on it; the audience groaned, “Not fair.” But I did get the answer right.

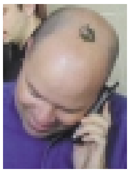
After the game we heard how our systems might be hacked in “Securing CF and IIS” by Dave Watts. He suggested taking a layered approach to security: network, application, and physical.

Next we learned how to pick a hosting company in “Running CF in a Shared Hosting Environment” by Tim Nettleton. Questions to ask a host included: What version of CF? What databases do you support? Security?

Continuing with the security theme we discovered what Linux does for security with “CF on Linux” by Matt Liotta.

Sunday

We got off to an early start on Sunday learning how to track down bugs in our apps – “Cool CF Debugging” by Shlomy Gantz. His famous last words were “It’s not a bug, it’s a feature.” Next, Charlie Arehart presented “Going from CF to Java.” He said our focus is on server-side Java, not client-side. After lunch there was a mini-play, “XFB Revealed,” from Hal Helms and the Fusebox crew that showed how projects can go wrong.



Then we played a CF Charades game in which contestants mimed CF tags and functions. It got pretty wild on the CFFLUSH tag. Back to serious stuff and mainframes and ColdFusion in the session “CF and Big Iron” by Robi Sen. Finally, we learned “CF5 Tricks and DHTML Tricks and Tips” from Steve Drucker.

Overall, the conference was a good way to learn from the pros in a fun environment. I’m already looking forward to next year.

About the Author

Eva Holtsmark is a ColdFusion programmer in Bethesda, Maryland.

@ EHOLTSMARK@BIGDOUGH.COM

Premiering...this fall
subscribe Now!
FOR FAST DELIVERY
The World's Leading Independent WebLogic Developer Resource
Go Online and Subscribe Today!
SPECIAL INTRODUCTORY OFFER
SAVE $31*
HURRY, DON'T DELAY! OFFER EXPIRES DECEMBER 31, 2001
Helping you enable inter-company collaboration on a global scale
• Product Reviews
• Case Studies
• Tips, Tricks and more!
FOR WLS DEVELOPERS BY WLS DEVELOPERS
BEA WebLogic DEVELOPER'S JOURNAL
An introduction to... WebLogic

We don't mind if they say we publish "geek" magazines, as long as we're known as...

The World's Leading *i*-Technology Publisher

June 2001
136,000 copies in print

June 2001 Preview
250,000 copies in print

June 2001
82,000 copies in print

June 2001
72,000 copies in print

June 2001
48,000 copies in print

Premiering December 2001
20,000 copies in print

Premiering January 2002
20,000 copies in print

Premiering February 2002
150,000 copies in print

and Producer of the World's Leading *i*-Technology Developer Conferences

JDJ EDGE conference&expo

XML EDGE conference&expo

web services EDGE conference&expo

COLDFUSION EDGE fasttrack

wireless EDGE conference&expo

Inc. 500 1999 WINNER

SYS-CON™ MEDIA

SYS-CON™ EVENTS

Inc. 500 2001 WINNER

www.SYS-CON.com

America's fastest-growing privately owned publishing company

MacroMickey: The 2001 Macromedia Devcon

by Kevin Schmidt

New technology and old friends at DevCon

An intrepid gaggle of geeks braved warm weather and sunny skies to attend the 2001 Macromedia DevCon at the Walt Disney World Swan and Dolphin Resort in Orlando, Florida, October 21–24, 2001.

Neo Is Next

The biggest buzz at this year's gathering was Neo, the code name for the next version of ColdFusion. Completely rewritten in Java and sitting atop JRun, the next version of ColdFusion offers numerous improvements over the current version, and it's even ready for Microsoft's .NET initiative. Tuesday morning's general session was dedicated to Neo and future product offerings from Macromedia; it included a sneak peek at the newest version of ColdFusion Studio, which took on more of a Macromedia look, similar to that of Dreamweaver. Flash, integrated with ColdFusion, also took center stage at the Tuesday session – the main message was that ColdFusion is here to stay and it's only getting better.

Workshops Galore

One of the mainstays of the event is the workshops – both hands-on and presentations. The workshops covered everything from working with ColdFusion 5, clustering servers, Web services, wireless applications, performance tuning, and error handling to other topics spanning ColdFusion, JRun, and various Macromedia products.

Vendors and the Cyber Café

Vendors were eagerly showing off their products and services to anyone who stopped by, and the cyber café allowed attendees to check e-mail or surf the Web. New this year was the client showcase, which allowed those companies currently using ColdFusion within their apps to demonstrate what they've done and how they did it. This proved helpful to any number of Neophytes, all of whom got a chance to see ColdFusion in action.

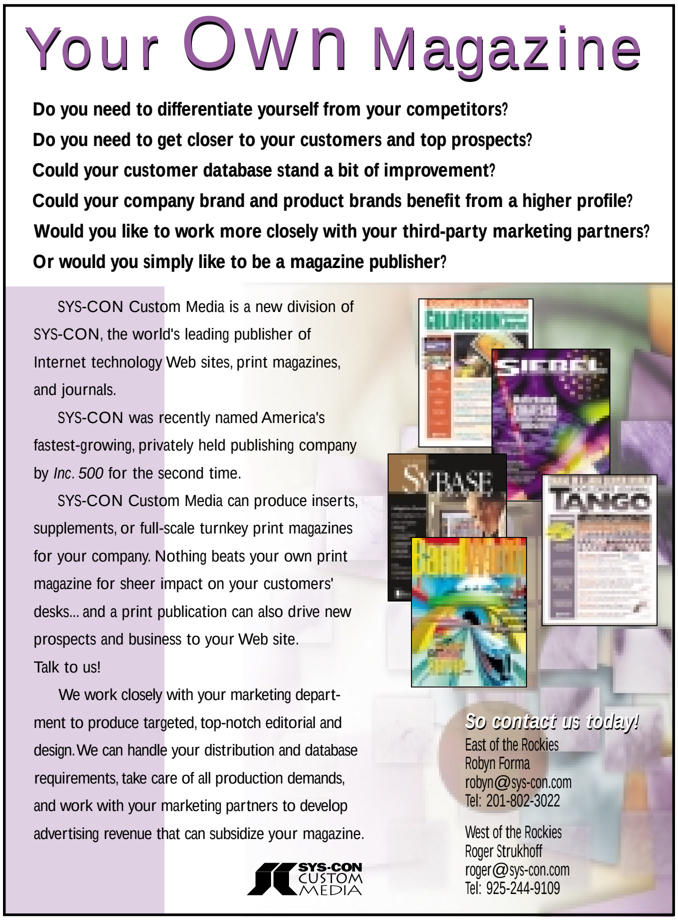
Other Stuff

Of course, DevCon didn't disappoint with the special event either. On Tuesday, Macromedia reserved MGM studios for an evening full of rides, food, and fun. No matter the time of day – there was always something to do.

Can't Wait for Next Year

Once again, I recommend DevCon to anyone who can break away for a couple of days. This year, as last, I boned up on loads of new technology, met lots of new people, and even got to see some old friends. Next year's event is again scheduled for Disney World. Hope you can make it.

@ SCHMIDT@HUNGRYCOW.COM



wirelessEDGE
conference&expo

INTERNATIONAL WIRELESS BUSINESS & TECHNOLOGY CONFERENCE & EXPO

CONFERENCE & EXPO

CONFERENCE & EXPO

Plan to Exhibit

Provide the Resources To Implement Wireless Strategy

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

Shaping Wireless Strategy for the Enterprise

Santa Clara, CA

May 7-9, 2002

WirelessEdge will provide the depth and breadth of education and product resources to allow companies to shape and implement their wireless strategy. Developers, i-technology professionals and IT/IS management will eagerly attend.

Plan to Attend the 3-DAY Conference

WHO SHOULD ATTEND

Mobile & Wireless Application Professionals who are driving their enterprise's wireless initiatives:

- Program Developers
- Development Managers
- Project Managers
- Project Leaders
- Network Managers
- Senior IT and Business Executives

Conference Tracks

Track One: Development	Track Two: Connectivity	Track Three: Wireless Apps	Track Four: Hardware	Track Five: Business Futures
WAP	Smart Cards	Education	Cell Phones/	Wireless in Vertical Industries
i-Mode	Wireless LANs incl. Bluetooth	Health Care	WorldPhones	The WWWW
Bluetooth / 802.11	UMTS/3G Networks	Entertainment	PDAs	Unwired Management
Short Messaging	Satellite Broadband	Transport	Headphones/	From 3W to 4W: Issues and Trends
Interactive Gaming		Financial Services	Keyboards /	"Always-On" Management
GPS / Location-Based		Supply Chain Management	Peripherals	Exploiting the Bandwidth Edge
Wireless Java			Transmitters/	Unplugged Valueware
XML & Wireless Technologies			Base Stations	Wireless Sales & Marketing
			Tablets	

FOR INFORMATION CALL

201 802-3069

SPEAKER PROPOSALS INVITED

WWW.SYS-CON.COM

SHAPE YOUR WIRELESS STRATEGY SAVE THE DATES!

EXCLUSIVE SPONSORSHIPS AVAILABLE

Rise above the noise. Establish your company as a market leader. Deliver your message with the marketing support of

SYS-CON MEDIA

SYS-CON EVENTS

THE LARGEST WEST COAST WIRELESS

BUSINESS & TECHNOLOGY CONFERENCE OF THE YEAR!

<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>

Web Services Edge 2002
& JDJEdge 2002
Conference & Expo
June 24–27, 2002
Jacob Javits Convention
Center, New York, NY

<search cfdj>

Search

<cfdj specials>

JDJStore.com
$229.99

JDJ, XML, CFDJ, CFAdvisor - ALL Four!!!-The Complete Works

<bestsellers>

1. IBM WebSphere Application Server 3.5 Standard Edition
$868.99
2. JRun Server 3.0/3.1 Professional 1 CPU License
$742.99
3. AltoWeb Application Platform Release 2.5
$3,640.00
4. ColdFusion 5.0 Server Enterprise
$4,672.99
5. ColdFusion 5.0 Server Professional
$1,219.99

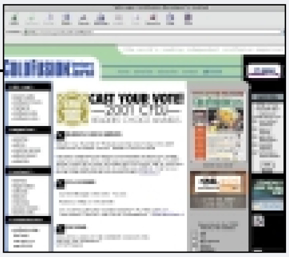
December **2001**

What's Online

www.sys-con.com/coldfusion

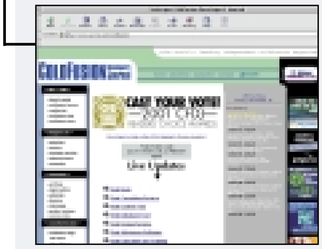
CFDJ Online

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



Readers' Choice Awards – Vote Now

Make your voice heard...this is your chance to show what you think. Categories for this year's awards include Best Book, Best Consulting Service, Best Custom Tag, Best Database Tool, Best Design Service, Best E-Business Software, Best Education and Training, Best Testing Tool, Best Web Development Tool, Best Web Hosting, Best Web Site, Best Web Application, and Most Innovative CF Application. To vote in this year's awards, visit www.sys-con.com/coldfusion/readerschoice2001.



JDJStore CD Special

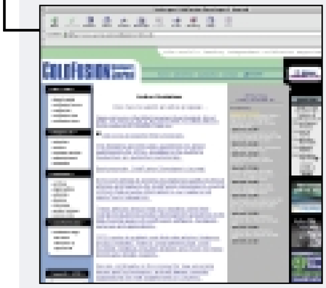
The complete library of *CFDJ, CFAdvisor, JDJ,* and *XML-J* articles are available in a four-CD set at a special price for a limited time from www.JDJStore.com.

Order now and have more than 1,200 articles on hand for research and review.

This exclusive package normally sells for $299.96, but it can now be yours for only $229.99. Order today and save!

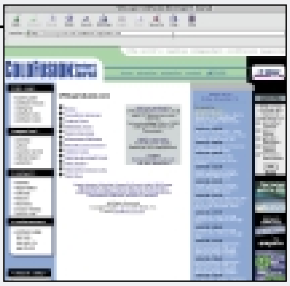
Share Your Expertise and Get a Byline!

Readers often ask if ***CFDJ*** accepts manuscripts from readers. The answer is an emphatic yes! We're continually on the lookout for emerging issues and ideas, and some of the best suggestions and articles come from readers like you. For author guidelines, go to www.sys-con.com/coldfusion/writers.cfm; this site has the complete rundown on how to submit an article proposal, the types of articles ***CFDJ*** wants, the editorial calendar, a writer's agreement, format details for text and code listings, copyright and payment information, and even some style pointers to improve your manuscript. Your CF expertise and ideas may get you into print!



The Latest in CF at the CFBuyersGuide.com

The most-read CF resource on the Internet, CFBuyers Guide.com (www.sys-con.com/coldfusion/ wbg/index.cfm), is essential for those who want to learn about the newest books, software, tools, and services related to ColdFusion. Web hosting, testing tools, books, e-business software, custom tags, Web development tools, consulting services, education and training, and much more are featured at this comprehensive site. Why waste time searching the Net for your CF needs? Discover the convenience of CFBuyersGuide.com.



QUICK POLL

What is Your Favorite New Feature of ColdFusion 5.0?
○ User Defined Functions
○ Query of Queries
○ Application Deployment
○ Integrated Charting
○ Performance Improvements

Vote
Results

Missed an issue?

We've got 'em all for you on CD!

JAVA DEVELOPER'S JOURNAL

The most complete library of exclusive *JDJ* articles on one CD!

Check out over 500 articles covering topics such as...
Java Fundamentals, Advanced Java, Object Orientation, Java Applets, AWT, Swing, Threads, JavaBeans, Java & Databases, Security, Client/Server, Java Servlets, Server Side, Enterprise Java, Java Native Interface, CORBA, Libraries, Embedded Java, XML, Wireless, IDEs, and much more!

***JDJ* The Complete Works**
~~Reg $119.99~~
Buy Online
Only $71.99

XML JOURNAL

The most complete library of exclusive *XML-J* articles on one CD!

Check out over 150 articles covering topics such as...
XML in Transit, XML B2B, Java & XML, The XML Files, XML & WML, Voice XML, SYS-CON Radio, XML & XSLT, XML & XSL, XML & XHTML, 2B or Not 2B, XML Industry Insider, XML Script, <e-BizML>, XML & Business, XML Demystified, XML & E-Commerce, XML Middleware, and much more!

***XML-J* The Complete Works**
~~Reg $59.99~~
Buy Online
Only $53.99

COLDFUSION Developer's Journal

The most complete library of exclusive *CFDJ* articles!

Check out over 250 articles covering topics such as...
Custom Tags, ColdFusion and Java, Finding a Web Host, Conference Reports, Server Stability, Site Performance, SYS-CON Radio, ColdFusion Tips and Techniques, Using XML and XSLT with ColdFusion, Fusebox, Building E-Business Apps, Applicaton Frameworks, Error Handling, and more!

***CFDJ* The Complete Works**
~~Reg $79.99~~
Buy Online
Only $71.99

SPECIAL OFFER:
Buy CFDJ & CFA
The Complete Works
For Only
$129.99

CF Advisor

The most complete library of exclusive *CFA* articles!

Check out over 200 articles covering topics such as...
E-Commerce, Interviews, Custom Tags, Fusebox, Editorials, Databases, News, CF & Java, CFBasics, Reviews, Scalability, Enterprise CF, CF Applications, CF Tips & Techniques, Programming Techniques, Forms, Object-Oriented CF, WDDX, Upgrading CF, Programming Tips, Wireless, Verity, Source Code, and more!

***CFA* The Complete Works**
~~Reg $79.99~~
Buy Online
Only $71.99

JDJStore.com

Order Online and Save 10% or More!

www.JDJSTORE.com

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

COLLECT ALL 4
FOR ONLY $229.99
JDJ, XML-J
CFDJ & CFA

CF USER GROUPS

Birmingham, AL
Jeff Bryant
jeff.bryant@mindspring.com

Huntsville, AL
Brian Yagar
brian.yager@redstone.army.mil

Phoenix, AZ
Sean Tierney
sean@pubcrawl.net

Tucson, AZ
Darcey Spears
dmspears@uswest.net

Bay Area, CA
Nathan Dintenfass
nathan@changemedia.com

Southern California
Leon Chalnick
lchalnick@advantasolutions.com

Southern California - Los Angeles
Sid Sheres
sheres@gte.net

Orange County, CA
Peter Amiri
peter@amiri.net

Sacramento, CA
Geoff Lilley
geofflilley@hotmail.com

San Diego, CA
Robert Munn
rob@illuminez.com

Orleans, Ontario
Clovis Paquette
clovispa@comnet.ca

Edmonton, Alberta
Jeff Acker
jeff.acker@topham.ab.ca

Lethbridge, Alberta
Chris Sorensen
chris@csystems.ab.ca

Montreal, Canada
John Kopanas
kopanas@videotron.ca

Ottawa, Canada
James Milks
jamesm@videotron.ca

Toronto, Canada
Kevin Towes
KTOWES@PangaeaNewMedia.ca

Vancouver, Canada
Cameron Siguenza
cameron@evolutionb.com

Colorado Springs, CO
Anthony Starkey
astarkey@mentiscorp.com

Denver, CO
Robi Sen
robi@granularity.com

Rocky Mountain
Mike Miller
mmiller@ewtechnology.com

Hartford, CT
Chris Graves
graves@rapidcf.com

New Haven, CT
Alfred DiMarzio
cfug@hbgraphics.com

Stamford/Westchester, CT/NY
Barry Hyman
bah@abodofoto.com

Delaware
John McKown
john@delaware.net

Gainesville, FL
Taner Aktas
aktas@ufl.edu

Jacksonville, FL
Russ Johnson
russ.johnson@cyberfusionsolutions.com

Orlando, FL
Randy Drisgill
rdrisgill@vshift.com

South Florida
Kevin Langevin
kevin@cfug-sfl.org

Tallahassee, FL
Benjamin Bloodworth
ben@electronet.net

Tampa Bay, FL
Kenneth Beard
kbeard2323@hotmail.com

Atlanta, GA
Adam Churvis
president@prodenhance.com

Atlanta, GA
Seth Land
gacfug@figleaf.com

Columbus, GA
Rob Seebach
rseebach@sfcts.com

Boise, ID
Paul Jenkins
pjenkins@doi.state.id.us

Chicago, IL
Robert Burns
bob-burns@mediaone.net

East Central Illinois
William Steiner
WilliamS@hkusa.com

Quad Cities
Jamie DeVolder
JamieD@4CS.COM

Springfield, IL
Gary Ashbaugh
ashbagd@mail.ioc.state.il.us

Central Indiana
Joshua Kay
josh@dataquix.com

Northern Indiana
Graham Pearson
gpearson@nicfug.org

Cedar Valley CFUG
David Chandler
David@ChandlerResearch.com

Des Moines, Iowa
Kevin Schmidt
schmidt@hungrycow.com

Louisville, KY
Tim Newton
TNewton@MercuryFusion.com

New Orleans, LA
Kathy Hester
kathyhes@yahoo.com

Annapolis, MD
Matt Schuster
mattschuster@erols.com

Baltimore, MD
Bret Peters
bmorecfug@figleaf.com

Broadneck High School CFUG
Rick Blaha
IRiCkYI@aol.com

Maryland
Michael Smith
Michael@teratech.com

Southern Maryland
Russ Mahan
rmahan@erols.com

New England Develpor's Network
Steve Casco
steve@ateaze.com

Detroit, MI
Matt McDonald
matt@stoneage.com

Mid-Michigan Group
Rick Mason
Rick@SeedChoices.com

Twin Cities, MN
Dan Chick
dan@danchick.com

Kansas City, MO
Bryan Laplante
directors@kcfusion.org

SoMoCFUG
Jeremiah Andrick
jandrick@theinteriorlife.com

St. Louis Metro Area
Dan Anderson
dan@psiwebstudio.com

Helena, MT
Tom Marino
tmarino@falcon.com

Omaha, NE
Mark Kruger
Mkruger@CFWebtools.com

Las Vegas, NV
Dave Byers
pegarm@lvcm.com

Concord, NH
Neil Giarratana
NGiarratana@ininet.com

Central New Jersey
Hien Nguyen
HNguyen7@prius.jnj.com

Northern New Jersey
Cindy Pomarlen
cjp@neotech-associates.com

South Jersey
Mark Stewart
m-stewart@home.com

Albuquerque, NM
Christopher Jungmann
jungmann@swcp.com

Albany, NY
Thomas McKeon
tmckeon@newkirk.com

Long Island, NY
Christopher Collins
chrisc@sitespan.com

New York, NY
Michael Dinowitz
mdinowit@houseoffusion.com

Rochester, NY
Dave Horan
dhoran@fusionproductions.com

Syracuse, NY
Laurie Ferger
lferger@twcny.rr.com

Charlotte, NC
Dave Crawford
dcrawford@charlotte-cfug.org

Greenville, NC
Steven Forehand
forehands@mail.ecu.edu

Raleigh, NC
Cheryl Elia
ccfug.mgr@listserv.unc.edu

Mid-Ohio Valley
Dave Hannum
hannum@ohio.edu

Cleveland, OH
Matt Newman
mnewman@newchanneltech.com

Columbus, OH
Angelo McComis
Angelo@CompuServe.com

Oklahoma City, OK
Glen Collymore
Glen-Collymore@ouhsc.edu

Tulsa, OK
Casey Bourbonnais
casey@cbfenterprises.com

Portland, OR
Jeff Cram
jcram@isitedesign.com

Roseburg, OR
Sharon Bland
sbland@douglasesd.k12.or.us

Harrisburg, PA
Neil Ross
info@codesweeper.com

Philadelphia, PA
Chris Swansen
cswansen@etechsolutions.com

Pittsburgh, PA
Kyle Jenkins
kyle@vaporbiz.com

State College, PA
Brian Panulla
brian@elmwoodmedia.com

Greenville, SC
David Goldfield
Dave@goldfield.com

Knoxville, TN
Doug McCaughan
doug@colknox.com

Memphis, TN
Michael Kubicki
mkubicki@univ-solutions.com

Nashville, TN
Chris Mosier
chris.mosier@e-maginative.com

Austin, TX
Mike Dickinson
cranch@texas.net

Dallas, TX
Patrick Steil
pmsteil@imailbox.com

Houston, TX
User Groups
usergroups@allaire.com

San Antonio, TX
Douglas White
doug@dwhite.ws

Utah State College
Jon Nelson
jnelson@nstep.net

Salt Lake City, UT
Mike George
ueberhund@hotmail.com

Provo, UT
Josh Trefethen
josh@exciteworks.com

Vermont
Sarah Moulton
info@mtbytes.com

Hampton Roads, VA
Raymond Camden
morpheus@deathclock.com

Northern Virginia
Bret Peters
novacfug@figleaf.com

Richmond, VA
Robert Baird
bbaird@btgi.net

Figleaf CFUG
Bret Peters
cfug@figleaf.com

Seattle, WA
Shannon Smith
president@cfseattle.org

Walla Walla, WA
Sarah Lewis
sarahlewis@alhena-design.com

Milwaukee, WI
Ron Kurtus
ron@ronkurtus.com

Wyoming
Matt Rodosky
mrodosky@wyoming.com

Nepal, Asia
Sudhir Raj Joshi
sjoshi@iapex.com

Australia
Marc Dimmick
marc@ibaseglobal.com

Canberra, Australia
Kunal Bhatia
kbhatia3@csc.com.au

New South Wales, Australia
Kym Kovan
kymk@microset.com.au

Northern Territory, Australia
Fred Lecul
flecul@paspaley.com.au

Queensland, Australia
Barry Moore
barry_moore@yahoo.com

Southern Australia
Brett Hales
brett.hales@baesystems.com

Tasmania, Australia
Mark Heseltine
HeseltineM@logica.com

Victoria, Australia
Jordan Thomas
jordan@vector7.com.au

Belgium
Denis Wauthy
d.wauthy@switchon.com

Rio de Janeiro, Brazil
Marcello Frutig
frutig@astrolabio.com.br

Sao Paulo, Brazil
Marcantonio Silva
marco@channelmedia.com.br

Central Europe
Sven Slazenger
slazenger@interlake.net

China
Feng Jian Feng
webmaster@cfwindow.com

Finland
Tero Pikala
tero.pikala@kolumbus.fi

France
Fabien Graille
FG500001@exchange.FRANCE.NCR.com

Ramstein Air Base, Germany
Tom Nunamaker
tom@nunamaker.com

Hong Kong
Shashank Tripathi
shashank.dp.tripathi@hk.pwcglobal.com

Hungary
Zolton Sheshler
ozoltan@yahoo.com

India
Zahid Java
zahidjava@yahoo.com

Indonesia
Agus Basuni
agus@digitaldevelopment.com

Ireland
Daryl Fullerton
daryl@thebiznet.net

Dublin, Ireland
Justin MacCarthy
maccarthy@iol.ie

Italy
Andrea Veggiani
andrea@dinamica.it

Okinawa CFUG
Michael Bradford
michael@devapps.com

Korea
HyeHyun Seo
hhseo@otozone.com

Malaysia CFUG
Halmi Yasin
affroman@affroman.com

Netherlands
Ron Pasch
ronnieo@dds.nl

Auckland, New Zealand
Amanda Irvine
amanda.irvine@delta.co.nz

Romania
Andrei Oprea
andrei@oprea.org

Johannesburg, South Africa
Steven Ringo
steven@tutuka.com

Spain
Jorge Pueyo
jpueyo@lander.es

Sweden
Stefan Gudmundsson
stefan@cfug-se.org

Zurich, Switzerland
Martin Buerlimann
bue@sercon.ch

Taiwan
Jack Jair
jack@leetide.net

Bangkok, Thailand
Paul Hastings
paul@tei.or.th

Turkey
Oguz Demirkapi
oguz@cftr.net

Dubai
Maryam Ebrahimi
CFUG@digiba.com

London
Niklas Richardson
niklas@ukcfug.org

Northern England
Stephen Moretti
stephen@cfug.org.uk

Southern UK
Stefan Elliott
cfug@four-runner.com

Subscribe Now

for Instant Access to CF Advisor

CFAdvisor.com!

12 months $89

24 months $169

Now on CD!

THE MOST COMPLETE LIBRARY OF EXCLUSIVE *CF ADVISOR* ARTICLES!

3 YEARS

40 ISSUES

200 ARTICLES

ONE CD

Since May 1998, CF Advisor™ has provided the CF community with in-depth information that quickly pays dividends on the small subscription investment. For just $89.00, you get 12 full months of uninterrupted access to CF Advisor™ – including:

- usable code
- skill-building articles
- tips
- news updates
- conference reports
- interviews with industry movers-and-shakers.

You get full access to our sizable archives covering such subjects as:

- custom tags
- functions
- variables
- framesets
- style sheets
- structures
- javascript usage
- debugging techniques
- programming practices

and much, much more...

Subscribing now guarantees that you'll immediately receive access to everything you need for keeping up-to-date with ColdFusion, no matter where you are. All the content is available exclusively over the Net and is regularly updated throughout the month.

THE MOST COMPLETE LIBRARY OF EXCLUSIVE *CF ADVISOR* ARTICLES ON ONE CD!

"The Complete Works"

CD is edited by *ColdFusion Developer's Journal* Editor-in-Chief Robert Diamond and organized into 21 chapters containing more than 200 exclusive *CF Advisor* articles.

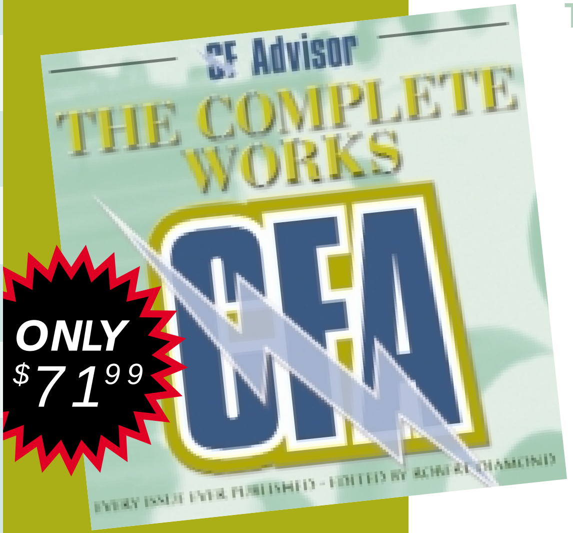
Easy-to-navigate HTML format!

E-Commerce
Interviews
Custom Tags
Fusebox
Editorials
Databases
News
CF & Java
CFBasics
Reviews
Scalability
Enterprise CF
Wireless
Verity
Source Code CF
Applications
Object-Oriented CF
WDDX
Upgrading CF
Programming Tips
CF Tips & Techniques
ProgrammingTechniques
Forms

Order Online and Save 10% or More!

www.JDJSTORE.com

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



SYS-CON MEDIA

135 CHESTNUT RIDGE ROAD
MONTVALE, NJ 07645
WWW.SYS-CON.COM

CFDJnews

CF5 Delivers Performance Gains

(Orlando, FL) – Extensive performance tests of Macromedia ColdFusion Server 5 reveal measurable gains across all supported platforms compared with the previous shipping version (4.5.1 SP2). The gains – a result of extensive internal server tuning – enable customers to deliver improved response times to end users as well as greater page throughput per server.

Macromedia ColdFusion Server 5 delivers cumulative performance gains in the key areas of request processing, memory management, and engine primitive handling. Macromedia benchmarks indicate that on Windows 2000, CF Server 5 processes page requests up to five times faster than the previous version did, 3.5 faster on Linux, and up to 1.5 times faster on Solaris.

www.macromedia.com

UltraDev 4 Studio Now Available

(San Francisco) – Macromedia ColdFusion 5 UltraDev 4 Studio, which unites the code-editing strengths of ColdFusion Studio 5 with the visual authoring environment of UltraDev for developing dynamic Web applications, has been released. This new Studio integrates the leading development environments for ColdFusion applications, providing developers with a productive toolset for creating, managing, debugging, and deploying powerful Web applications for CF Server 5.

It includes complete ColdFusion Server 5 support, simplified application deployment, enhanced custom tag creation, and improved workflow through integration with Macromedia Fireworks.

ColdFusion Server 5 offers a new, integrated charting engine and advanced full-text searching. It also delivers a significant performance increase, running some applications as much as three times faster than ColdFusion 4.5.

www.macromedia.com

Macromedia Supports Windows XP

(San Francisco) – Macromedia, Inc., has announced its support of Microsoft's Windows XP operating system. Macromedia's product line of authoring, server, and playback technologies already runs on Windows XP.

Both companies previously announced that Macromedia Flash Player 5 will ship with all versions of Windows XP and that Flash was used by Microsoft for the informational "Tour of Windows XP" experience that ships with the operating system.

www.macromedia.com

Resource Center Available to Help Developers

(San Francisco) – A new resource center to help developers combine Macromedia Flash content with Macromedia ColdFusion Server to deliver dynamic user experiences has been announced by Macromedia, Inc.

The center contains articles, tutorials and sample components, and links to third-party sites that are integrating the two products.

www.macromedia.com/go/cf_flash_resources

Macromedia Rolls Out CF Technology

(Orlando, FL) – The next major release of Macromedia ColdFusion, code-named "Neo," will support deploying solutions on leading Java technology–based application servers.

With the support of industry leaders such as BEA, IBM, Intel, and Sun Microsystems, Inc., Macromedia, Inc., has outlined a broad technology strategy for ColdFusion to run on multiple application servers, support XML Web services, interoperate with Microsoft .NET technologies, and integrate with other Macromedia products such as Flash and Dreamweaver.

www.macromedia.com

PaperThin Teaming Up with e-Zone Media

(Boston) – PaperThin, Inc., a leader in Web publishing and content management solutions, will integrate FuseTalk, a ColdFusion-based discussion forum, and FuseAds, a robust ad management application, both developed by e-Zone Media, Inc.

PaperThin began marketing FuseTalk and FuseAds as companion offerings to CommonSpot Content Server in fourth quarter 2001, both directly and through its channel partners.

www.paperthin.com
www.e-zonemedia.com

CFWebstore Version 4.5 Released

Dogpatch Software has released CFWebstore Version 4.5. Extensive rewriting of the code base in the new version focused on improving the speed and function of the code and making it fully compatible with ColdFusion 4.5 and 5.0 servers.

Shopping cart and order information is now stored in temporary database tables, stylesheets and checkout features have been improved, the use of session and application variables has been reduced, and alternate pages have been included for the removal of the CFX tag.

www.cfwebstore.com

Events

CF Europe
January 24–25, 2002
Forum der Technik
Munich
www.cf-europe.org

Macromedia Technology Conference
March 14–15, 2002
Denver
www.cfconf.com/denvercf/

CFDJ ADVERTISER INDEX

ADVERTISER	URL	PHONE	PAGE
ABLECOMMERCE	WWW.ABLECOMMERCE.COM	888.801.1333	2
ACTIVE PDF	WWW.ACTIVEPDF.COM	888.389.1653	4
ASSETNOW	WWW.ASSETNOW.COM		31
CFADVISOR.COM	WWW.CFADVISOR.COM		51
CFDYNAMICS	WWW.CFDYNAMICS.COM	800.422.7957	19
CFXHOSTING	WWW.CFXHOSTING.COM	866.CFX-HOST	11
COLD FUSION DEVELOPER'S JOURNAL	WWW.SYS-CON.COM	800.513.7111	41
EMPIRIX	WWW.EMPIRIX.COM/DOUBLE/CFM		9
EVOLUTIONB	WWW.EVOLUTIONB.COM/PARTNERS		55
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM	877.215.HOST	25
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800.379.7729	56
JAVA DEVELOPER'S JOURNAL	WWW.SYS-CON.COM	800.513.7111	23
JDJ STORE	WWW.JDJSTORE.COM	888.303.JAVA	49
MACROMEDIA	WWW.MACROMEDIA.COM/GO/MASTERING	877.460.8679	15
MACROMEDIA	WWW.VUE.COM/MACROMEDIA	877.460.8679	29
MACROMEDIA	WWW.MACROMEDIA.COM/GO/USERGROUPS	877.460.8679	13
MINERVA	WWW.MINERVA.NET	888.667.7231	27
PACIFIC ONLINE	WWW.PACONLINE.NET	877.503.9870	33
PAPERTHIN	WWW.PAPERTHIN.COM	800.940.3087	37
RACKSPACE	WWW.RACKSPACE.COM	888.850.1470	3
SYS-CON MEDIA, INC.	WWW.SYS-CON.COM	800.513.7111	45
WEB SERVICES JOURNAL	WWW.WSJ2.COM	800.513.7111	39, 43
WEBLOGIC DEVELOPERS JOURNAL	WWW.SYS-CON.COM/WEBLOGIC	800.513.7111	44
WIRELESS EDGE CONFERENCE & EXPO	WWW.SYS-CON.COM	201.802.3069	47
XML JOURNAL	WWW.SYS-CON.COM	800.513.7111	17

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

Letters to the Editor. . .

HTML E-Mailing

Thanks for Kelly Brown's eminently readable article on text/HTML e-mailing ("Harnessing the Power of HTML E-Mails," Vol. 3, issue 8).

Do you know where I can go to encode JPEGs or GIFs in order to include them in the HTML portion?

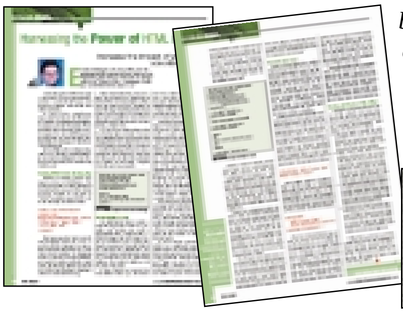
Or is this worth another article? :->

Phil Dunn
pkd@hoptechno.com

I haven't tried to do that in ColdFusion, but I have in Perl. There is a custom tag available that will allow you to send HTML messages with images attached, but I can't remember the name of it.

Sending an attachment is similar to the technique I used for the HTML e-mail. You simply need to use a different content header and encode the graphic image. The problem is encoding the data since CF doesn't have a built-in function to do this. There may be a custom tag or UDF available to encode it.

Kelly
kbrown@aboutweb.com



Timely Information

Just in time! I happened to need the information in Charles Arehart's article ("Testing Existence in Arrays," Vol. 3, issue 4). As luck would have it, I read the article the day before I needed it. Thank you so much. His article saved me from several headaches.

Sheryl Daugherty
Webmaster
Hunter Engineering Company

Well isn't that cool. :-) Very glad to hear it. It's great that you'd read it. I'm sure otherwise you might never have known there was an article addressing that very issue.

Thanks for sharing the feedback. It's always sincerely appreciated.

Charlie
carehart@systemanage.com

Letters may be edited for grammar and clarity as well as length.

Please e-mail any comments to Robert Diamond at robert@sys-con.com

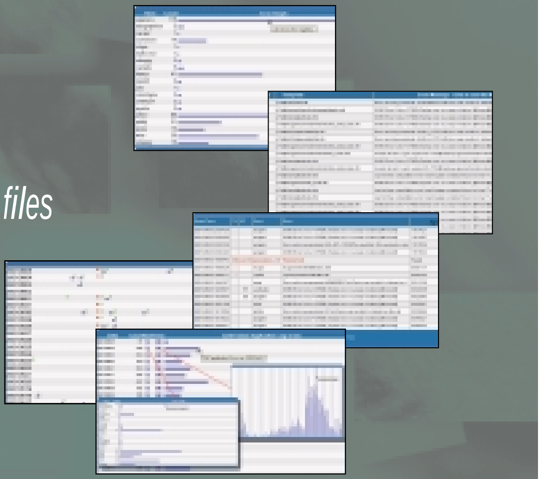
Next Month...

Managing ColdFusion
A CF server overview
by Charles Arehart

Protecting Images with ColdFusion
Use <CFIMPERSONATE> and <CFCONTENT> to protect files
by Steven Lewis

COSMOS: Managing the ColdFusion Experience
A new perspective
by Timothy Nettleton

Book Review
Core ColdFusion 5 by Eben Hewitt
reviewed by Susan Matteson



COLDFUSION Developer's Journal

Don't miss the January issue!