

COLD FUSION Developer's Journal

ColdFusionJournal.com

November 2004 Volume: 6 Issue: 11

February 22-23 2005
Hyatt Regency Center
Boston, MA

SEE PAGE 49

Join Over 3,000 Developers
App Server Shoot-Out!

FREE Web Services
Web Services Journal Pavilion
Seminars and Case Studies

\$1,000,000
Software Giveaway

Editorial
MAX Impact
Simon Horwith
page 5

Community Focus
**Talking with the
Blackstone Team**
Simon Horwith
page 7

Conference Report
**The State of the
Fusebox Union**
Sean Corfield
page 26

REVIEW
BlueDragon 6.1
Phil Cruz
page 28

RETAILERS PLEASE DISPLAY
UNTIL JANUARY 31, 2005

\$9.99US \$9.99CAN



**SYS-CON
MEDIA**

GOING TO THE MAX

By April Fleming page 34

**T-SQL: Applied Advanced T-SQL
with ColdFusionMX** *A look at new
possibilities in working with SQL Server*

Grant Szabo



8

**Platforms: Crossing the .NET Divide: CFMX,
Web Services, and .NET** *ColdFusion
Web service interoperability with .NET*

Joe Rinehart



20

CF101: ColdFusion Components
A powerful tool for CF developers

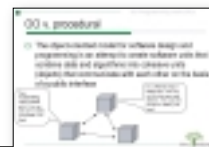
Jeffrey Houser



38

**Techniques: Thinking Outside
the Table PART 1** *How to shift workload
from the app server to the database*

Tom Peer



42

Foundations: Saving Polly *Making
polymorphism more understandable*

Hal Helms



46

macromedia®





This is:

☐ Video

☐ Flash

☐ Don't bother me. I'm staring.

The line between Flash and Video is forever gone.

Travel with us to Scandinavia for a behind the scenes tour of the award-winning, genre-bending Volvo V50 website.

macromedia.com/go/volvo



CFDynamics

Dedicated Server Packages Starting at \$189/mo.

All dedicated servers include:

- ▶ FREE STATS SOFTWARE!
- ▶ No long term commitments!
- ▶ FREE SQL server access!
- ▶ FREE MAIL SOFTWARE!
- ▶ Fair and simple pricing!
- ▶ Optional server maintenance!

As one of the premier ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to provide **better service** to ensure your satisfaction. Not only do we offer the **finest in shared hosting plans**, but we now offer the **finest in 100% dedicated server plans**! Now you can afford the freedom of having your own dedicated server!

When your needs have outgrown shared hosting look to CFDynamics for **total freedom**. With dedicated server packages they're not offering an oxymoron; "virtually private" or "virtually dedicated" is **NEITHER** private nor dedicated. CFDynamics offers a solution that is **100% completely dedicated**. They don't play games with the fake stuff; CFDynamics only offers the real deal. Real Service. Real Satisfaction. Real Value.

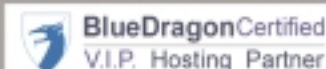
Real Freedom.



Visit us online or call to order!



PaperThin Partner



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

editorial

editor-in-chief

Simon Horwith simon@sys-con.com

technical editor

Raymond Camden raymond@sys-con.com

executive editor

Jamie Matusow jamie@sys-con.com

editor

Nancy Valentine nancy@sys-con.com

associate editors

Gail Schultz gail@sys-con.comNatalie Charters natalie@sys-con.com

research editor

Bahadır Karuv, PhD bahadir@sys-con.com

production

production consultant

Jim Morgan jim@sys-con.com

lead designer

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.comRichard Silverberg richards@sys-con.comTami Beatty tami@sys-con.comAndrea Boden andrea@sys-con.com

contributors to this issue

Sean Corfield, Phil Cruz, April Fleming, Hal Helms,

Simon Horwith, Jeffry Houser, Tom Peer,

Joe Rinehart, Grant Szabo

editorial offices

SYS-CON MEDIA

135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645

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

COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)

is published monthly (12 times a year)

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

postmaster: send address changes to:

COLDFUSION DEVELOPER'S JOURNAL

SYS-CON MEDIA

135 Chestnut Ridge Rd., Montvale, NJ 07645

©copyright

Copyright © 2004 by SYS-CON Publications, Inc.

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

Worldwide Newsstand Distribution

Curtis Circulation Company, New Milford, NJ

FOR LIST RENTAL INFORMATION:

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

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

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

MAX Impact

I've just returned to London from MAX 2004...what a conference! The announcement of the availability of Flex 1.5; cool demos of Blackstone, Flex, Flash on mobile devices; and a sneak peek at what's in Flash 8 were among the more notable announcements and content at this year's conference. For a more in-depth summary of the conference, take a look at April Fleming's article in this month's issue.



By Simon Horwith

While I was at MAX, I was able to have a private round-table interview with the Blackstone development team. We discussed the motives behind the features in Blackstone, what it was like to build Blackstone, what they would like to see in the next release after Blackstone, and about the development community and how they can help. You can read the results of my interview in this month's Community Focus column.

Also at MAX, I did a lot of brainstorming and bounced ideas for the magazine off of many developers. I also got a ton of positive feedback about the "deep focus" issue we ran a couple of months ago, so I'm pleased to announce that I will make every effort to make that the permanent format for **CFDJ** starting in January 2005. I have created an editorial calendar for next year, available at www.sys-con.com/coldfusion/writers.cfm. If you're interested in writing about anything related to any of the calendar topics, simply bounce the idea off of me and hopefully I'll get you into the schedule. I know this will make many of you happy...but possibly not as happy as what I have to say next. I went to MAX with a mission: to get more Macromedia staffers involved in **CFDJ**. The ColdFusion team has made it clear that they thrive on customer feedback and that they like interacting with the development community; so I set out to try and give them another forum in which to interact with the community. Mission accomplished!

The Blackstone development team has agreed to write articles all about the new features in Blackstone for a future issue. This issue's publish date will coincide with the public release of Blackstone. Who better to explain each feature than the person(s) who actually developed it? So, in the not-too-distant future I will be handing over the bulk of the articles to the Blackstone development team for a special "Blackstone" issue. Of

course, I still expect to run plenty of articles in subsequent issues that are written by developers about these new features, but letting the Blackstone team introduce them to us will be a treat. In addition, I'm talking with them about introducing a regular "Ask the CF Team" column in which readers can submit questions about ColdFusion behavior and problems that they'd like explained/answered

by someone from the ColdFusion development team. It's not a substitute for tech support – the Macromedia developers will respond to queries with one or two paragraph explanations.

I have ideas for other regular columns that I'll throw out for comments right now (please e-mail me and let me know what you think). Awhile ago I wanted to start a regular column to give a voice to CFUG/MMUG managers. I didn't get much response at the time – maybe now I'll get a better response? Another idea I had is for a regular column that is based around a case study of how ColdFusion and/or other Macromedia product(s) enabled a company to do great things. Being able to show our bosses and/or clients how CF saved the day at a company is very useful when you're trying to convince them to upgrade and/or simply use ColdFusion on a project. I've also been thinking about starting a regular server administration column – focusing on securing, optimizing, and administering ColdFusion.

— continued on page 41

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. He is a freelance software architect currently consulting with companies in London. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and consults on ColdFusion applications that leverage Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com simon@horwith.com

Nothing beats our racks

Absolutely nothing



Robert Marsh, Head Surfer

CARRIER CLASS DATA CENTERS

Highly Secure Guarded Facility
24/7/365 Network Operations Center
24/7/365 Technical Support
Redundant Conditioned Air Systems
Redundant Fiber Entry Points
Multiple Uninterruptible UPS Systems
Multiple 1250 KW Generators Onsite
VESDA Early Warning Smoke Detection

START YOUR OWN WEB HOSTING BUSINESS TODAY!

from **\$299*** **Dedicated Server**
Dual Xeon 2.4 GHz
2 GB RAM • 2 x 73 GB SCSI HD
Remote Console • Remote Reboot
1700 GB Monthly Transfer Included
Instant Activation!

Over 20,000 Servers!

1-800-504-SURF | ev1servers.net

PLESK7
RELOADED
Preferred Control Panel

IP Compliant. Price subject to change. Quantities Limited.

*Per month. Set-Up fees apply. See web site for complete details.

Talking with the Blackstone Team

CF development is about to change forever

At MAX I had the pleasure of sitting around a private table with several members of the Blackstone Development Team to talk about the new features, the development community, and future ColdFusion releases after Blackstone.

With so many people in the room who were all passionate about the release features, it became difficult to keep all of the information in a "typical" interview format (with question-to-question flow and nothing but direct quotes for answers). I was, however, able to gather enough information to give some insight as to the internal development process, what shapes the ColdFusion features list, and what may be in store in the future.

Attending this meeting were the following Macromedia employees from the ColdFusion Team (and their titles):

- Dave Gruber: senior product manager
- Damon Cooper: director of engineering
- Tom Jordahl: principal engineer
- Jim Schley: principal Q/A engineer
- Jim Murphy: senior Q/A engineer
- Mike Nimer: senior engineer
- Dean Harmon: senior engineer

The interview with the team members began with me asking a series of questions but quickly turned into more of a forum for open discussion. Although it ruled out a lot of direct quotes, it also contributed to the general flow of creative juices and more candidness. This article will therefore be based around not just what was said, but my overall impressions from talking with the team. Before continuing, it's important that I disclaim that in no way does this article make any promises about what is or is not in Blackstone nor what will or will not be in any subsequent release of ColdFusion (but we can hope, can't we?). The remainder of this article summarizes my discussion with members of the Macromedia Blackstone Development Team.

To begin with, we discussed how Blackstone came to be. About four to five years ago, when



By Simon Horwith

Allaire/Macromedia was beginning to think about and prototype a version of ColdFusion that was written in and ran on the Java platform, there was a list of features that they wanted to implement. At the same time they were facing the decision of re-architecting ColdFusion to run on the Java platform. They ultimately decided to put off most of the new features and focus on the Java re-architecture, which ultimately became ColdFusion MX.

After MX was released, the team returned to the feature list in a project that was code-named "Elvis" (some of you may have heard rumors about this "product"). Elvis became Red Sky, which focused primarily on bug fixes and performance. Red Sky later became ColdFusion MX 6.1. Now that the engine is streamlined and stable, the Blackstone team has been able to refocus on this list of features they'd like to implement. These features, according to Tom, came from specific customer requests and a broader effort to address a growing collection of customer needs.

Why return to the feature list now? Well, with a solid platform in place on which to deploy these features (CFMX6/6.1), the team finally has the time they need to properly tackle them.

— continued on page 30

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. He is a freelance software architect currently consulting with companies in London. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and consults on ColdFusion applications that leverage Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com simon@horwith.com

Applied Advanced T-SQL with ColdFusionMX

New possibilities in working with SQL Server

The July 2004 (Volume 6, issue 7) edition of CFDJ featured an article I authored entitled “Harnessing the Power of SQL Server Using Stored Procedures.” In that article I covered the basics of building stored procedures and using them in ColdFusionMX using ColdFusion Components. This article will expand further upon that topic, delving deeper into the advanced capabilities of T-SQL and SQL Server.

In this article, I will demonstrate some techniques that I hope you can use in your own projects, and I will provide information and examples on the following topics:

- Using a cursor
- Nesting stored procedures
- Output parameters
- User-defined functions and string manipulation
- Linked servers
- Date functions
- Specifying default values for input parameters
- Using DTS to schedule a stored procedure



By Grant Szabo

In some cases, these topics will be blended together in single examples. The source code is liberally commented to help you understand the concepts in action. The goal is to introduce you to some additional possibilities in how you approach your projects with SQL Server. Please keep in mind that the T-SQL code examples being shown are in most cases showing one possibility out of many options. It would be a good idea after reading this article to spend some time with the Microsoft SQL Server Books Online reading up on some of the statements used in the T-SQL examples.

About T-SQL

Structured Query Language (SQL) was developed by IBM in the mid-1970s. SQL is both an ANSI and an ISO standard. Transact-SQL (T-SQL) is a proprietary set of programming extensions from Sybase and Microsoft that add additional features to SQL. These features include transaction control, exception and error handling, row processing, and declared variables.

Conceptual Application Architecture

In the previous article, I discussed *n*-tier architecture. I am going to talk a bit more about this in order to get you thinking about what it means. The reason is that when using advanced T-SQL, you really have the ability to get kind of wild and crazy with how much business logic you push down into the data tier. Before you rush off to do this, I want to spend some time talking about application architecture.

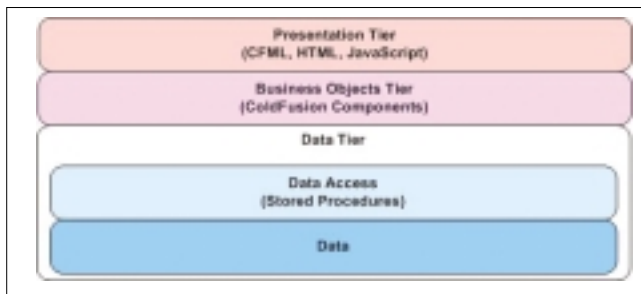


Figure 1: Conceptual application architecture

With *n*-tier architecture, efforts are made to separate presentation logic from business logic. This kind of architecture is often logical in nature – meaning there isn't necessarily (although there could be) a physical separation of the layers. The separation is popular among developers for a number of reasons. Off the top of my head, these include:

- **Maintainability:** The application is easier to maintain over time because functionality is logically separated. There isn't any (or at least much less) spaghetti code to deal with.
- **Reusability of code:** Encapsulating functionality into business objects and stored procedures means that this code can be reused by other parts of the application, and potentially by other applications, reducing complexity and increasing developer productivity.
- **Division of labor:** Data tier experts can do what they do best, class designers can do what they do best, and the presentation/designer folks can do what they do best.

In ColdFusion MX, *n*-tier architecture is done by using ColdFusion Components (CFCs) to encapsulate business logic. The presentation layer instantiates these business objects as required and calls upon the methods embedded in them to perform required tasks. Independent of the business objects, and lower in the stack, is the data tier. As depicted in Figure 1, I have differentiated data access objects from the actual data. Data access objects, for the purposes of this discussion, are the stored procedures.

Applying this architecture to your ColdFusion project means that:

1. The Presentation layer contains only the HTML, CFML, and JavaScript necessary for presenting the user interface. There is no business logic embedded in this layer. This layer will instantiate CFCs as required to return data for presentation as required.
2. The Business Objects layer contains CFCs that are designed to encapsulate business rules dictated by the project. There are no direct data calls (ad hoc queries) used in this layer. To access data, this layer calls out to SQL Server stored procedures. An excellent book on architecting CFCs as well as an OOP (object oriented programming) primer is Hal Helms' *Discovering CFCs*.
3. The Data layer contains precompiled stored procedures that provide an API for your business objects to interact with the underlying database entities.

So far so good? Now is where it starts to get muddy.

An architectural consideration when designing an application

involves deciding how much business logic to place in the data tier versus the business objects. A purist would say that all business rules should be completely encapsulated in the business objects tier leaving the data tier to simply serve up data. The reality is that there are advantages and disadvantages to each approach, making this decision even more difficult.

I'm not going into a full discussion on this topic, but I will mention what I consider to be important considerations. One is whether or not your application will be the exclusive "user" of the database. Keep in mind that there may be one or more additional applications, thin- or thick-client, that could potentially be developed now or in the future using the same stored procedures. By putting business logic into your stored procedures, you can effectively share that business logic between otherwise disparate applications.

Another important consideration when trying to figure out how much business logic to put into your stored procedures is performance. Certain capabilities may simply run faster in your application's business objects versus the data tier, or vice versa. If performance is your chief concern (and when is it not a concern?), then you will want to give this area some careful thought. My experience with ColdFusion and SQL Server is that unless I am dealing with string manipulation, SQL Server will generally outperform even my most tightly developed ColdFusion code (unless I've written some particularly bad T-SQL that is). Performance is as much an art as it is a science in my opinion. You can (and should) try to test proof of concept builds of key functional pieces of your application if time permits.

When working on .NET projects, I find that it is perhaps a bit easier to figure out where to put my business logic. With .NET, business rules can be encapsulated solely in the business objects and thin, thick, compact, or console-based .NET User Interfaces (UIs) can be plastered on top of them. In .NET, business objects are called class files. Building an *n*-tier application with .NET means that instead of baking business rules directly into your WinForm or ASP.NET code-behind, you create business objects (class files) to handle the heavy lifting. These class files can even be compiled into a stand-alone .dll file and then be referenced by any .NET application you might develop in the future. When you need the functionality that your business object provides in your application, you simply instantiate an instance of it in your WinForm or ASP.NET code-behind, and call whatever method(s) are needed from that instance of the object. As long as you use .NET for all of your UIs, those business objects can be shared across all of your .NET applications, regardless of the UI chosen for the project. Thus, for .NET projects at least, it really makes sense to keep your business rules in your business objects to the greatest extent possible.

With ColdFusion, we are dealing strictly with a UI for the Web. One way to make ColdFusion's business objects reusable is to expose them as Web services. This allows the ColdFusion business objects to be shared with non-ColdFusion applications that are capable of consuming Web services. If you go this route, make sure to consider the security of your Web services, as well as making your return types as generic as possible to ensure a high level of compatibility with consuming applications.

All right, enough of this discussion for now. I hope I've given you some food for thought on this topic. Let's move on to the core of this article – Advanced T-SQL.

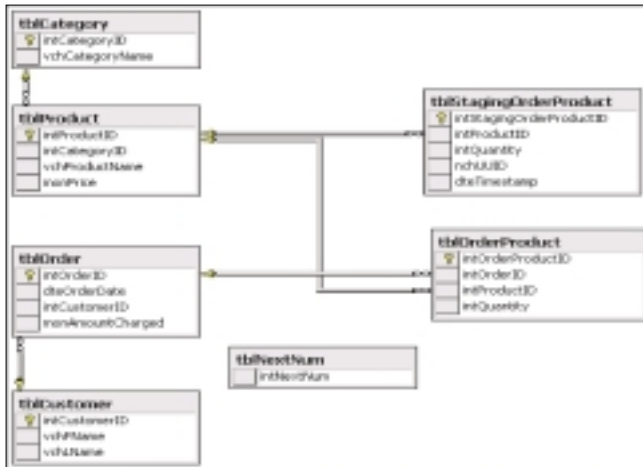


Figure 2: Shopping cart entity relationship diagram

Leveraging SQL Server

Microsoft SQL Server is a powerful relational database management system (RDBMS) that is, generally speaking, too often underutilized by ColdFusion developers. This article will not discuss how to create a stored procedure – please refer to the July 2004 article for that detail if necessary.

Using a Cursor

In the previous article, I provided a T-SQL code teaser of a stored procedure that uses a cursor. That code is reprinted here, but this time I'll step through it in detail and provide contextual information for its usage.

Cursor is an acronym that stands for a current set of records. As a ColdFusion developer you are no doubt familiar with the tag `<cfloop>`. Now imagine that you can perform a `<cfloop>` inside a record set inside of a stored procedure! Cursors are similar to while loops except instead of the while iteration happening in your ColdFusion middleware code, it can happen in your stored procedure.

The stored procedure that will be examined here is used in a custom shopping cart application that I wrote earlier this year (see Figure 2). As with all shopping carts, users visit the site and populate items into their shopping carts. `tblStagingOrderProduct` is the entity that stores the contents of

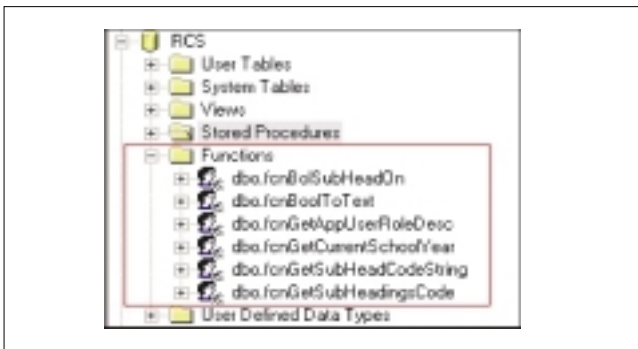


Figure 3: UDFs in SQL Query Analyzer

users' shopping carts while they browse the site. As items are added or removed from the shopping cart, the contents of this table change. The field `nchUUID` in `tblStagingOrderProduct` is a UUID that is maintained in a session scope (via a session-based user object) to link the users' carts to their session.

When users determine that they wish to check out, they are first asked to log in if they have not already done so. This allows the `intCustomerID` to be set in the session-based user object. Next, users are prompted to enter payment and shipping information. This information is then submitted to a payment gateway. If the response from the payment gateway (based on the user's credit information) is okay, then the stored procedure `spGS_Utl_ProcessOrder` is executed.

`spGS_Utl_ProcessOrder` will create the new order in `tblOrder` and seed the `tblOrderProduct` table with all of the items in `tblStagingOrderProduct` for the `nchUUID` linked to the user. This means that *only* valid, approved orders are created in `tblOrder` and `tblOrderProduct`. The contents of `tblStagingOrderProduct` are periodically purged by another stored procedure that is scheduled to run on a daily basis by SQL Server DTS (Data Transformation Service – more on this later). This stored procedure removes any records whose `dteTimestamp` is `>= 24` hours old, keeping `tblStagingOrderProduct` from filling up with potential orders not committed from more than a day ago.

A cursor is used in `spGS_Utl_ProcessOrder` to iterate over each item in `tblStagingOrderProduct` for the UUID. Each matching item is then inserted into `tblOrderProduct`.

The full T-SQL source listing is included for reference (see Listing 1). Let's break down each part of the stored procedure to see what is happening.

```
CREATE PROCEDURE spGS_Utl_ProcessOrder

@nchUUID          char(35),
@intCustomerId    int,
@monAmountCharged money

AS

/* Purpose: rolls an authenticated order over from staging into non
staging tables.*/

DECLARE @intOrderId      int,
        @dteOrderDate    datetime,
        @intProductId    int,
        @intQuantity     int

-- set the order date using the SQL Server GETDATE() function
SET @dteOrderDate = GETDATE()
```

Here, I am creating the stored procedure and defining three input parameters that the stored procedure will expect when it is executed. These fields are the UUID, the customer ID from the customer table, and the amount charged. In the production version of this stored procedure, there are additional input parameters, but I've simplified this example in order to make it easier to see what is happening.



Need a hand with your work? Get the MX Kollection.

Introducing MX Kollection - the suite of extensions designed to change the way you create dynamic web applications with Dreamweaver MX and MX 2004.

Our customers think of MX Kollection as the next level in Dreamweaver MX visual software development.

ColdFusion and PHP developers will find our product enabling them to visually develop e-Commerce, CMS, CRM, Backend and other web solutions with increased efficiency, quality and capability.

Create powerful dynamic lists

- Sort, filter and page result sets
- Perform multiple deletes
- Easily create Master/Detail lists

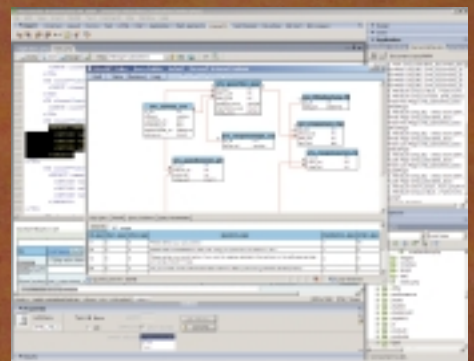
Build unified add/modify forms

- Client side and server side validation
- Insert into two tables
- Create form actions such as send mail or delete related record
- File/Image upload and resize

Create recordsets visually

- Build complex SQL queries across multiple tables quickly

... and many more



Interakt

macromedia
ALLIANCE PARTNER

Download the demo and see the features and benefits of our extensions at <http://www.interaktonline.com>

Next, I declare four additional variables that are going to be used by the stored procedure later on. The order ID is a value that I will set when I prepare to roll the contents of the user's shopping cart over to a real order. The order date will be set by the SQL Server built-in function GETDATE(), which returns the system date. The product ID and the quantity are both used by the cursor, as we will see in a moment.

```
/*get the next available order ID, note use of output parameter.
@intOrderID will hold the value that is returned by spCMS_Utl_NextNum
*/
EXEC spCMS_Utl_NextNum @intOrderID OUTPUT

-- insert the order header into tblOrder
INSERT INTO tblOrder (
    intOrderID ,
    intCustomerId ,
    dteOrderDate ,
    monAmountCharged)
VALUES (
    @intOrderID ,
    @intCustomerId ,
    @dteOrderDate ,
    @monAmountCharged )
```

Next, I call out to another stored procedure, spCMS_Utl_NextNum, to get the next available OrderID. Note that I am specifying @intOrderID as an OUTPUT parameter. I will explain this in more detail later in this article. What you need to know is that when I execute spCMS_Utl_NextNum, @intOrderID is filled with the next available order ID.

After populating @intOrderID, I then append the order header to the order table.

Last, I set up a cursor in order to move the items out of the shopping cart and into the table that holds the items ordered by the user. Here is the T-SQL syntax for declaring a cursor:

```
DECLARE cursor_name CURSOR
[LOCAL | GLOBAL]
[FORWARD_ONLY | SCROLL]
[STATIC | KEYSSET | DYNAMIC | FAST_FORWARD]
[READ_ONLY | SCROLL_LOCKS | OPTIMISTIC]
[TYPE_WARNING]
FOR select_statement
[FOR UPDATE [OF column_name [...n]]]
```

You will definitely want to read up on this in SQL Server Books Online in order to understand what all the options mean. Cursors have performance implications; when you use them, you usually want to be working with a small number of records and use Forward-Only, Fast Forward, or Read Only cursors as these perform best. There are alternatives to cursors, including while loops, derived tables, and more. In my case, I decided to use a cursor because I was dealing with a small number of records.

```
/* use a cursor to insert all of the order items (children) into
tblOrderProduct from the nchUUID
See books online Transact-SQL under topic CURSOR for explanation of
various options */
```

```
DECLARE csrRecord CURSOR LOCAL FORWARD_ONLY STATIC FOR
SELECT intProductId,
    intQuantity
FROM    tblStagingOrderProduct
WHERE   nchUUID = @nchUUID

OPEN csrRecord
FETCH NEXT FROM csrRecord INTO @intProductId, @intQuantity

-- see @@FETCH_STATUS in books online. Zero means that the Fetch was
successful.
WHILE @@FETCH_STATUS = 0

BEGIN

    INSERT INTO tblOrderProduct (
        intOrderID ,
        intProductId ,
        intQuantity )
    VALUES (
        @intOrderID,
        @intProductId,
        @intQuantity )

    FETCH NEXT FROM csrRecord INTO @intProductId, @intQuantity

END

CLOSE csrRecord
DEALLOCATE csrRecord
```

Looking at the T-SQL code, note that I first declare the cursor and then immediately below that, create a record set by selecting the product ID and quantity from the table holding the user's shopping cart data.

FETCH NEXT FROM then takes the values for those variables in the current iteration and populates @intProductId and @intQuantity.

Next you see the statement "WHILE @@FETCH_STATUS = 0." This essentially means "while there are still records in the record set that do this", effectively creating the construct needed to end the iteration when there are no more records.

Now I go ahead and insert the product and quantity for the current iteration into the table that holds the ordered items for the order. Since I have the order ID, I insert that as the foreign key into that table.

Last, it is very important to close the cursor and DEALLOCATE it in order to free up the resources used by the cursor.

The end result is that I now have a single stored procedure that I call, pass three input parameters, and have the user's order rolled over from a nonorder to a validated order. Instead of writing scores of lines of CFML, I now can write one simple <CFUNCTION> in my Order.CFC file to roll over orders (See Listing 1).

Nesting Stored Procedures

Nesting stored procedures simply means that you call out to one or more additional stored procedures from the one you are in. For example, imagine that you execute stored procedure A. During execution, stored procedure A executes stored

procedure B to retrieve some data that stored procedure A then uses to complete its task.

An example of this is depicted in the stored procedure spGS_Utl_ProcessOrder that you read about in the Cursors section. This procedure contains a nested stored procedure, spGS_Utl_NextNum. That line reads: "EXEC spGS_Utl_NextNum @intOrderID OUTPUT." As you can see, one stored procedure can easily call another, thereby reusing functionality instead of rewriting it each time from scratch.

As a data architect, this should tell you something very important when modeling your database and developing your data objects – always consider reusability when developing your database. Truly great programmers are just plain lazy, and for good reason.

Here is the T-SQL for the spGS_Utl_NextNum stored procedure:

```
CREATE PROCEDURE spGS_Utl_NextNum

@intOrderID int OUTPUT

AS

DECLARE @newVal int

-- get nextNum value
SELECT @intOrderID = intNextNum
FROM tblNextNum

-- increment the nextNum value
SET @newVal = @intOrderID + 1

UPDATE tblNextNum
SET intNextNum = @newVal

-- return the nextNum value
SELECT @intOrderID
```

Output Parameters

Output parameters can be used with nested stored procedures as shown in spGS_Utl_ProcessOrder, or can be used to return a value to ColdFusion (rather than returning a complex value such as a record set). I have found this to be most valuable when returning the next primary key value. spGS_App_Customer depicts an example of appending a new customer record to tblCustomer and providing the new intCustomerID as an output parameter.

```
CREATE PROCEDURE spGS_App_Customer

@vchFName varchar(30),
@vchLName varchar(60),
@intCustomerID int OUTPUT

AS

INSERT INTO tblCustomer ( vchFName,
```



How would you like to provide PDF generation and manipulation capabilities to your entire organization without breaking the bank? activePDF offers affordable server-based solutions that take the guesswork out of the PDF conversion process making it both simple and seamless to end users.



► activePDF Server™ is a server-side PDF print driver, capable of redirecting output from virtually any Windows application to PDF. With over 75 properties and methods available from its unique COM interface, activePDF Server provides you full control over your PDF output on a job-by-job basis.



► activePDF DocConverter™ enables you to convert over 280 different file formats directly to PDF. DocConverter can be used in batch mode, via "watched folders", or in client/server applications via a distributable remote submission component.



► activePDF WebGrabber™ dynamically converts any URL, HTML stream or HTML file to PDF on the fly, while retaining embedded styles. Custom page break tags, built-in page numbering, and dynamic headers and footers provide additional control over PDF output.



► activePDF Toolkit™ is a programmable COM object that simplifies PDF manipulation. Licensed per server, Toolkit enables you to append, stamp, stitch, merge, paint, secure, form-fill PDFs and more.



Copyright © 2004, activePDF, Inc. All Rights Reserved. "activePDF", "Leading the iPaper Revolution", "It's Everybody's PDF" and the activePDF logo are registered trademarks of activePDF, Inc. All activePDF product names are trademarks of activePDF, Inc.

```

        vchLName )

VALUES ( @vchFName,
        @vchLName )

SELECT @intCustomerID = @@IDENTITY

```

To call this stored procedure from a ColdFusion business object, the following code would be used:

```

<cfstoredproc procedure="spGS_App_Customer" datasource="#this.DSN#">
  <cfprocparam type="In" cfsqltype="CF_SQL_VARCHAR"
dbvarname="@vchFName" value="#arguments.vchFName#" null="No">
  <cfprocparam type="In" cfsqltype="CF_SQL_VARCHAR"
dbvarname="@vchLName" value="#arguments.vchLName#" null="No">
  <cfprocparam type="Out" cfsqltype="CF_SQL_INTEGER"
variable="intCustomerID" dbvarname="@intCustomerID" null="No">
</cfstoredproc>

```

Note how the third cfprocparam statement specifies type = "Out". What happens here is the next primary key value from tblCustomer is then assigned to a ColdFusion variable named intCustomerID. It's up to you what you want to do with that value – maybe you just want to return it from the cfunction, or maybe there is some other work that needs to be done before returning a value. The point is that a simple value is returned from the stored procedure call rather than a query being returned.

User Defined Functions and String Manipulation

User defined functions in SQL Server can be used for a variety of purposes. Functions in SQL Server work in somewhat the same way as functions in ColdFusion – they receive one or more arguments and return some value. Just as in ColdFusion, user defined functions in SQL Server are reusable and this is a key reason to use them. Because we are dealing with data here, user defined functions can initiate SELECT statements to retrieve data from any table in the database; however, a user defined function cannot specify any operation that modifies data in any way. For example, you cannot use a user defined function to INSERT or UPDATE data.

User defined functions can be created in SQL Query Analyzer or in SQL Enterprise Manager. If you use SQL Enterprise Manager, right-click on User Defined Functions and select New User Defined Function. The syntax for a user defined function will be displayed as:

```

CREATE FUNCTION [OWNER].[FUNCTION NAME] (PARAMETER LIST)
RETURNS (return_type_spec) AS
BEGIN
(FUNCTION BODY)
END

```

If you use SQL Query Analyzer, you won't be given any clues about the syntax – you're expected to know it. Figure 3 is a screenshot from SQL Query Analyzer showing some user defined functions.

I'll show you a simple user defined function called fcnGetAppUserRoleDesc. This function returns a string containing the word "Admin" or "User" denoting the permission

level of a user. Rather than writing conditional logic in ColdFusion for this purpose, I elected to push that logic down to the data tier in this example. The end result is that the word "Admin" or "User" is part of the record set that spGS_Sel_User returns. I elected to put this logic into a user defined function instead of using the T-SQL CASE statement because I thought there might be other stored procedures that would want to reuse the functionality of this user defined function (see Listing 2). Here are some cfstoredproc statements for interacting with spGS_Sel_User:

```

<!-- calling spGS_Sel_User this way will return all users from
tblAppUser -->
<cfstoredproc procedure="spGS_Sel_User" datasource="#this.DSN#">
  <cfprocparam type="In" cfsqltype="CF_SQL_INTEGER"
dbvarname="@intAppUserID" null="Yes">
  <cfprocresult name="RS1">
</cfstoredproc>

<!-- calling spGS_Sel_User this way will return properties for user
ID number 5 -->
<cfstoredproc procedure="spGS_Sel_User" datasource="#this.DSN#">
  <cfprocparam type="In" cfsqltype="CF_SQL_INTEGER"
dbvarname="@intAppUserID" value="5" null="No">
  <cfprocresult name="RS1">
</cfstoredproc>

```

User defined functions are a great place for handling string manipulation. In several cases, I've used user defined functions to provide formatted HTML that I then return in a recordset. Here is some T-SQL that I pulled out of one of my user defined functions that illustrates this technique:

```

DECLARE @vchSubHeadCode varchar(500)

SET @vchSubHeadCode = '<a href="multitab.cfm?intStudentCourseId=' +
CAST(@intStudentCourseId AS varchar(10)) + '&intGradeLevelId=' +
CAST(@intGradeLevelId AS char(1)) + '&StudentId=' + CAST(@intStudentId
AS varchar(10)) + '" onClick="fcnSubmit();">Modify</a>' + SPACE(2)

```

In this code snippet you can see that I am creating the first part of an HREF statement that has some query string arguments. The parameters @intStudentCourseId, @intGradeLevelId and @intStudentId are integer data types that are arguments of the user defined function that this code snippet is pulled from. In order to use those values in a concatenated string, I need to use the SQL Server built-in function CAST() to cast the integer data type to a string data type, such as char or varchar. The SPACE() built-in function creates white space in the string. Also note that the concatenation operator in T-SQL is the plus (+) symbol, not the ampersand (&) that is used in CFML. Also remember that SQL Server is a dog at string concatenation, so be smart about when and where you do string concatenation. There have been a number of times in my experience when returning preformatted text from the stored procedure made sense and provided a more elegant architecture. For this reason, I wanted to share this technique with you so that you could add it to your arsenal of T-SQL knowledge.

Specifying Default Values for Input Parameters

Normally when defining an input parameter for a stored procedure, simply type an @ symbol followed by your variable name, followed by its data type. If that is all you do, then you have created a required input parameter for your stored procedure. However, what if you want to make the inclusion of a value optional and set a default value for an input parameter if that value isn't specified when the stored procedure is executed?

spGS_Sel_User depicts an optional input parameter:

```
@intAppUserId int = NULL
```

In effect, this means that @intAppUserId is not a required argument for the stored procedure. If the user ID is not passed in, then it defaults to NULL. It would be possible to specify an integer value as the default value for @intAppUserId if NULL wasn't what you wanted to do, for example, @intAppUserId int = 5, where 5 might be a demo account, for example.

Linked Servers

Sometimes you may have more than one SQL Server that you need to access in order to retrieve data for your project. Or, perhaps you have an Oracle database in addition to the SQL Server database that you need to communicate with for your project requirements. Once two servers are linked, T-SQL statements can be written on SQL Server A to work against data objects on SQL Server B via OLEDB. This is an incredibly powerful feature of SQL Server and something that you should definitely know about in case you run into a situation where you need to use this knowledge. I'll give you an example of one of my own experiences.

I recently worked on a project for a company that used SQL Server for their Web applications and Oracle for their in-house thick-client applications. The project required that I pull large amounts of data from the Oracle database, cleanse the data (due to human data input error), make it searchable via Verity, and expose it on the Web. Because Oracle was used to drive their operational systems, anything I did with Oracle needed to be low touch. For this same reason, my access to the Oracle server was just high enough to access the tables I needed to retrieve data from.

Initially, I tried pulling the data out of Oracle using a ColdFusion ad hoc query because I did not have sufficient rights on the Oracle server to create PL/SQL stored procedures. Once I had the record set in ColdFusion, I used CF to cleanse the data through multiple iterations and then insert the cleansed data into SQL Server. Last, I would run some CFM files that would index the Verity collections. What I learned after I had built this elaborate mousetrap was that it was very slow. Worse, since the client was going to be adding even more data to Oracle over time, my architecture would not scale. What to do?

Searching for a better architecture, I learned that it would be possible to link to the Oracle RDBMS server directly from SQL Server. By linking the SQL Server to the Oracle Server I could effectively use T-SQL instead of PL/SQL to access data on the Oracle Server. Moreover, I

could create stored procedures on the SQL Server to do this job and then schedule the job to run in the middle of the night with Data Transformation Services. To put this into an easily understandable perspective, the mousetrap architecture took about an hour to complete versus the linked server architecture, which took under 5 minutes to complete. It may have taken so long to run because I am not a strong PL/SQL developer; however, there were definitely some other factors in play:

1. The client did not have an Oracle DBA on staff and the database was not tuned to do the types of joins I was requesting. Just requesting the data via TOAD for example took several minutes.
2. The record sets were quite large; once I had the record set in ColdFusion I still had to cleanse the data. This involved iterating over the data a number of times to ferret out the bad rows and then insert these row by row into a failed data table on the SQL Server.

The bottom line is that it was a very confusing process involving a lot of moving parts, and isolating where the problem was in my code became a nightmare. As a software architect I simply came to the conclusion that this was not the right architecture and my code was proof of that.

To link an Oracle server to an SQL Server, follow these steps:

1. Install SQL*Net Oracle Client on the SQL Server machine and reboot it.
2. Register the Oracle Server in SQL*Net.
3. Start SQL Query Analyzer and enter the following T-SQL Statement:

```
USE master
GO
-- create the linked server
EXEC sp_addlinkedserver 'ORACLE_DB_ALIAS', 'Oracle', 'MSDAORA', 'ORACLE'
GO
-- set login credentials for the linked server
EXEC sp_addlinkedsrvlogin 'ORACLE_DB_ALIAS', 'FALSE', NULL, 'Oracle_username', 'Oracle_password'.
```

Press CTL-E to execute your T-SQL statement. Provided you did not receive errors, the linked server relationship is established. This is a one-time process and does not have to be repeated.

Now, you can write a T-SQL statement to pull data from the

PART NAME	DESCRIPTION
linked_server	Linked server referencing the OLE DB data source
catalog	Catalog in the OLE DB data source that contains the object
schema	Schema in the catalog that contains the object
object_name	Data object in the schema
Source: http://www.schemamania.org/jkl/booksonline/SQLBOL70/html/8_qd_12_2.htm	

Table 1: Four-part name and description

GETDATE()	Used to retrieve the system date. Thus, instead of passing in the current date from ColdFusion, you can handle this in your stored procedure. An example of this is included in the spGS_Utl_ProcessOrder example.
DATEDIFF()	Used to calculate the difference between two dates.
DATEPART()	Returns a specific portion of a date. For example, if you just want to return the year from a datetime column you could say <code>SELECT DATEPART(year, dtOrderDate)</code> .
DATEADD()	Adds (or subtracts) years, months, days to a date.
DATENAME()	Returns the date as a string. For example, <code>SELECT DATENAME(month, getdate())</code> returns the name of the month, such as "November". determines if an expression is a date or not.
ISDATE() MONTH(), DAY(), YEAR()	Exactly identical to the ColdFusion equivalents.

Table 2: Useful date functions

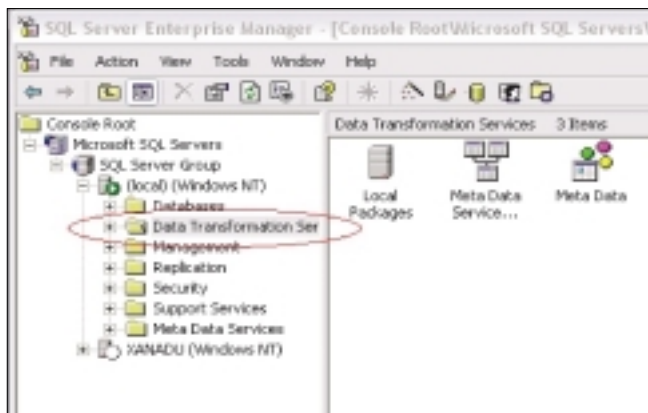


Figure 4: DTS in SQL Enterprise Manager

Oracle Server. In this example, what I am doing is joining two tables on the Oracle Server to retrieve a record set. This stored procedure uses a subquery in the WHERE clause to further refine the resulting record set. This stored procedure is slightly modified to simplify it from its production form. I am also using table aliases for the table names as the four-part names for the tables are quite long and cumbersome to type (see Listing 3).

The two periods after ORACLESERVER in the FROM and WHERE clauses of the example are not typos. Calling out to a linked server requires the use of a four-part name which is: `linked_server_name.catalog.schema.object_name` (see Table 1).

In my case, there is no catalog, but a placeholder for the non-existent catalog using the four-part name notation is required.

Date Functions

If you have become familiar with the Date functions available to you in ColdFusion, you should have no trouble picking up these useful date functions. These functions are very useful when working with datetime fields; you should strongly consider using them in your stored procedures instead of trying to manipulate dates in your ColdFusion middleware code whenever possible (see Table 2).

Note that when combined with a cursor, DATEADD() can be used with a stored procedure to return one or more record sets containing the months of any given year for the creation of an online calendar, for example.

Using DTS to Create and Schedule a Package

Here is how Microsoft defines DTS:

A DTS package is an organized collection of connections, DTS tasks, DTS transformations, and workflow constraints assembled either with a DTS tool or programmatically and saved to Microsoft SQL Server, SQL Server 2000 Meta Data Services, a structured storage file, or a Microsoft Visual Basic file.

Each package contains one or more steps that are executed sequentially or in parallel when the package is run. When executed, the package connects to the correct data sources, copies data and database objects, transforms data, and notifies other users or processes of events. Packages can be edited, password protected, scheduled for execution, and retrieved by version.



Figure 5: Simple DTS package

DTS is definitely worth reading about in the SQL Server Books Online and is arguably one of the more powerful aspects of SQL Server. I have heard that the upcoming release of SQL Server will allow DTS services to be written in .NET.

I'm going to show you how to create a simple DTS service using the DTS Designer included with SQL Enterprise Manager and then show you how to schedule it (see Figure 4). The shortest interval for a DTS Package to be scheduled is 60 seconds.

To create a DTS Package using the DTS Designer, follow these steps:

1. Open SQL Enterprise Manager on the Production SQL Server:

2. Under the (local) server, look for the folder Data Transformation Services:
3. Right-click on Local Packages and choose New Package – you will now be in the DTS Designer:
4. Add a Microsoft OLE DB object:
 - a. Existing Connection is checked, “Microsoft OLE DB Provider for SQL Server” is in the drop-down menu
 - b. Data source: “Microsoft OLE DB Provider for SQL Server” is in the drop-down menu
 - c. Server: (local) is selected
 - d. Use Windows Authentication is checked
 - e. Database: MyDatabase
5. Now add an Execute SQL Task to the designer:
 - a. Description: Execute SQL Task: undefined
 - b. Existing Connection: Microsoft OLE DB Provider for SQL Server
 - c. Command time-out: 0
 - d. SQL Statement:

`exec spMyProc`

Now save the DTS package and give it a name. Figure 5 shows a DTS package that executes a stored procedure.

Last, to schedule the DTS Service, right-click the service you just created and choose Schedule Package. Set the schedule to repeat as often as required. Please note that you will need to

have the SQL Server Agent service running on the database server to use this feature.

Conclusion

In this article I have demonstrated several advanced T-SQL techniques. Many of the topics covered could be stand-alone articles in their own right. Additional topics that could be covered in future articles include transactions, temporary tables, table variables, and handling multiple record sets. If you would like to see additional coverage of these topics, or if there are additional topics beyond these that you would like to see covered, please e-mail me and let me know. Also, please continue your learning by spending some time with SQL Server Books Online.



About the Author

Grant Szabo is a senior application developer for Global Cloud, Ltd. (www.globalcloud.net), where he delivers Microsoft .NET and ColdFusion solutions for Global Cloud's clients. Grant worked for Allaire (later Macromedia), as director, worldwide professional services, for nearly two years in 2000-2001. He is certified in CF5 and CFMX and holds numerous Microsoft certifications including MCSD, MCDBA, MCSE, and MCSA.

grant@quagmire.com

fast • easy • affordable •

CommonSpot[™] Content Server

Efficient Content Management

- 100 % browser-based
- Content object architecture
- Template driven pages
- 50+ standard elements
- Extensible via ColdFusion
- Content reuse
- Content scheduling
- Flexible workflow
- Granular security
- CSS support
- SOB compliance
- Personalization
- Replication
- Custom metadata
- Custom authentication
- Static site generation
- Multilanguage support

With CommonSpot Content Server you get it all. CommonSpot's exceptional blend of rapid deployment, ease of use, customization and scalability make it the leading ColdFusion content management solution.

Our rich Web publishing framework empowers developers with out-of-the-box features such as template-driven pages, custom content objects, granular security, flexible workflow and powerful ColdFusion integration hooks (just to name a few), allowing you to rapidly build and efficiently deploy dynamic, high performance Web sites.

For the past six years, PaperThin has been a leader in the ColdFusion community, and CommonSpot has been the solution of choice for organizations of all sizes, including Architect of the Capitol, AFL-CIO, Boeing, Kaiser Family Foundation, Ohio University, PGA.com and hundreds of others. CommonSpot's sophisticated feature set and affordable pricing are an unbeatable combination.

Call us today at 800.940.3087 to schedule a live demonstration of your site running under CommonSpot, or visit www.paperthin.com to learn more.

© Copyright 2004 PaperThin, Inc. All rights reserved.

Listing 1

```
CREATE PROCEDURE spGS_Utl_ProcessOrder
@nchUID char(35),
@intCustomerId int,
@monAmountCharged money
AS
/* Purpose: rolls an authenticated order over from staging into non
staging tables.*/
DECLARE @intOrderId int,
@dtOrderDate datetime,
@intProductId int,
@intQuantity int
-- set the order date using the SQL Server GETDATE() function
SET @dtOrderDate = GETDATE()
/*get the next available order ID, note use of output parameter.
@intOrderId will hold the value that is returned by spCMS_Utl_NextNum
*/
EXEC spCMS_Utl_NextNum @intOrderId OUTPUT
-- insert the order header into tblOrder
INSERT INTO tblOrder (intOrderId,
intCustomerId,
dtOrderDate,
monAmountCharged)
VALUES (@intOrderId,
@intCustomerId,
@dtOrderDate,
@monAmountCharged)
/* use a cursor to insert all of the order items (children) into
tblOrderProduct for the nchUID
See books online Transact-SQL under topic CURSOR for explanation of
various options */
DECLARE csrRecord CURSOR LOCAL FORWARD_ONLY STATIC FOR
SELECT intProductId,
intQuantity
FROM tblStagingOrderProduct
WHERE nchUID = @nchUID
OPEN csrRecord
FETCH NEXT FROM csrRecord INTO @intProductId, @intQuantity
-- see @@FETCH_STATUS in books online. Zero means that the Fetch was
successful.
WHILE @@FETCH_STATUS = 0
BEGIN
INSERT INTO tblOrderProduct (intOrderId,
intProductId,
intQuantity)
VALUES (@intOrderId,
@intProductId,
@intQuantity)
FETCH NEXT FROM csrRecord INTO @intProductId, @intQuantity
END
CLOSE csrRecord
DEALLOCATE csrRecord
```

Listing 2

```
ALTER PROCEDURE spGS_Sel_User
@intAppUserId int = NULL
AS
/* This stored procedure can return the entire list of users from
tblAppUser OR a specific user (if the intAppUserId is specified). A
user defined function is used to return a string denoting the privilege
level of the user. */
-- @intAppUserId will be NULL unless a value was provided when
spGS_Sel User is executed
IF (@intAppUserId IS NULL)
```

```
-- no user ID specified, return entire list
BEGIN
SELECT nchUserName,
nchFName,
nchLName,
intAppUserId,
intRole,
vchRoleName = dbo.fcnGetAppUserRoleDesc(intRole)
FROM tblAppUser
ORDER BY nchLName, nchFName
END
ELSE
-- user ID specified, return specific user
BEGIN
SELECT nchUserName,
nchPassword,
nchFName,
nchLName,
intRole,
intAppUserId,
vchRoleName = dbo.fcnGetAppUserRoleDesc(intRole)
FROM tblAppUser
WHERE intAppUserId = @intAppUserId
END
CREATE FUNCTION fcnGetAppUserRoleDesc (@intRoleId int)
RETURNS varchar(5)
AS
BEGIN
DECLARE @vchRole varchar(5)
IF (@intRoleId = 1)
BEGIN
SET @vchRole = 'User'
END
IF (@intRoleId = 2)
BEGIN
SET @vchRole = 'Admin'
END
RETURN @vchRole
END
```

Listing 3

```
INSERT INTO tblVerityTempOracleA
SELECT T1.MKEY [mkey],
T1.ID NUMBER [id number],
T1.ITEM CLASS [item_class],
T1.FIELD2 [field2],
T1.FIELD3 [field3],
T2.DERIV ID [deriv id]
FROM ORACLESERVER..MIMSY.CATALOGUE T1 INNER JOIN
ORACLESERVER..MIMSY.IMAGES T2 ON T1.MKEY = T2.MKEY
WHERE T1.ID NUMBER LIKE @vchMimsySearchStr
AND T1.FIELD15 = 'PERMANENT COLLECTION'
AND T1.MKEY IN (SELECT T3.MKEY
FROM ORACLESERVER..MIMSY.LOCATION HISTORY T3
WHERE T3.NEW LOCATION NOT LIKE 'NIM%'
AND T3.NEW LOCATION NOT LIKE '%MISSING%'
AND T3.NEW LOCATION NOT LIKE '%STOLEN%'
AND T3.NEW LOCATION NOT LIKE 'SEE%'
AND T3.NEW LOCATION NOT LIKE 'CANCELLED%'
AND T3.NEW LOCATION NOT LIKE 'TRANSFERRED%')
```

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

THE GRASS REALLY IS GREENER ON THE **SERVERSIDE.**

SUPERIOR MANAGED HOSTING

- Intelligent Routing
- Redundancy
- Network Security
- Service Level Agreement
- Environmental Controls
- Network Uptime Guarantee
- Scalability
- OC3/SONET Backbone
- Backup Power
- Physical Security
- Fire Protection
- Server Hardware Guarantee

IF YOU'VE BEEN SEARCHING for a reliable managed hosting partner, your search for greener pastures is over. There is no longer a need to settle for inferior support and lack of accountability. ServerSide takes the guess work, and the associated hassles, out of working with a technology partner.

ServerSide provides managed web hosting solutions for a wide range of customers including; Fortune 500 corporations, small and medium sized businesses and non-profit organizations located across the United States and around the World.

We provide a single point of accountability for all your web hosting infrastructure needs — while adding value, not cost.

>> SPECIAL OFFER

Sign up online for a new shared hosting account now at www.serverside.net and ServerSide will waive the set-up fee.

ENTER CODE: mxd2004

The grass is greener at
www.serverside.net
888.682.2544
hosting@serverside.net



Crossing the .NET Divide: CFMX, Web Services, and .NET

ColdFusion Web service interoperability with .NET

Like it or not, .NET is proving a powerful force in the software industry. As a ColdFusion developer, you'll probably need to interact with it at some point in time.

One of the promises of .NET is that it can provide this interoperability through Web services. Although it's easy to send simple data to and from .NET Web services, a little bit of extra work on our end can make arrays, structures, and even queries work across the two platforms.

Quick Review: ColdFusion Web Services

This article is about ColdFusion Web service interoperability with .NET. It assumes a basic knowledge of how to create and consume Web services in ColdFusion. A brief review may help everyone get on the same page.

Web services use XML to provide remote procedure calls, or services, in a platform- and language-agnostic environment. By using the XML document Web Services Definition Language (WSDL), they describe how they work in a common, agreed-on format. When they communicate, data is sent in another XML format called Simple Object Access Protocol (SOAP).

Although WSDL and SOAP are both fairly complicated, creating and consuming Web services in ColdFusion couldn't be easier! To create a service, create a ColdFusion component (CFC) file in a Web-accessible directory. Give it at least one function (or method), being sure to specify its return type and



By Joe Rinehart

that its access attribute is set to remote. If the method has arguments, you must also specify the data type of those arguments. The following code shows a very simple ColdFusion Web service that returns a simple string:

```
<cfcomponent>
  <cffunction name="getString"
    returntype="string"
    access="remote">
    <cfreturn "Hello, world!">
  </cffunction>
</cfcomponent>
```

After creating your CFC, it will be available as a Web service at <http://<servername>/<path>/<filename>?WSDL>. For example, if the code from the previous example was in "sampleService.cfc" under the root on your local machine, it would be available as a Web service at <http://localhost/sampleService.cfc?WSDL>.

Consuming a Web service from ColdFusion is just as simple. In fact, it only takes one line of code. The following code shows how to consume and then display the results of our CFC from the previous example. In this example, we use the CFInvoke tag to both instantiate and call a method of our sample Web service. We then output the result in a standard CFOutput block:

```
<cfinvoke
  webservice="http://localhost/
  sampleService.cfc?WSDL"
  method="getString"
```



```
returnvariable="result">
<cfoutput>#result#</cfoutput>
```

Now that we've reviewed creating and consuming Web services in ColdFusion, let's move on to creating Web services that can be consumed by .NET clients.

Providing Data to .NET Through Web Services

When you build a Web service in ColdFusion, the "returnType" attribute of the CFFunction tag states what type of data the Web service will be returning. The language Web services uses to communicate (SOAP), describes these return types using XML. ColdFusion automatically creates this XML when your CFC is accessed as a Web service. It is up to the client software to interpret this XML and create variables it can "understand" from the data returned by your Web service.

Sending Simple Values to .NET

ColdFusion is considered a typeless language, and simple values are the variables that fall into this "typeless" category. An easy way to think of simple values is that they can be directly printed to a page using a CFOutput block. For example, you can directly output a date inside a CFOutput tag, but not a query: A date is a simple value, but a query is not.

Lucky for us, .NET understands all of the simple ColdFusion return types without a hitch. This means that any Web service method returning these types can be understood by .NET without any changes. Table 1 shows the .NET data types that the results of your ColdFusion Web services will be translated into.

You'll notice that three of the most powerful ColdFusion data types aren't listed in Table 1: arrays, structures, and queries. These are three of the complex data types available in ColdFusion. It's when we use ColdFusion Web services to send complex data types to .NET that we begin to encounter trouble.

Sending Simple Arrays to .NET

ColdFusion arrays and .NET arrays are very different, and the technical details of these differences are beyond the scope of this article. What is relevant, however, is that .NET does not translate a ColdFusion array sent over a Web service back into a native .NET array. Instead, .NET interprets them as its generic "object" type.

As long as your array contains simple values, you're in the clear. A .NET developer can easily loop over your result in the .NET equivalent of a collection-based CFLoop to extract the data. The following example shows the VB.NET code to do this:

```
For Each i In CFArrayResult
    Response.Write("i: " & i)
Next
```

We'll tackle complex arrays, such as arrays of structures, after we've first taken a look at how a structure can be sent to .NET

Sending Structures to .NET

Presently, ColdFusion structures do not translate to .NET. If you send a structure to .NET through a ColdFusion Web service, the .NET developer receives nothing but an empty object. However, Macromedia has provided a solution to this through CFCs.

To return a structure to .NET, you create a "definition" CFC that will represent the structure and its data. Each member of the structure needs a corresponding CFProperty tag in the definition CFC.

Instead of returning a structure, you first create an instance of the CFC. Then, use CFSet tags to set its properties to their corresponding values in your structure. Last, you return the instance of the CFC instead of your structure. It's important to note that you need to change the returnType of your CFFunction tag from "structure" to the (package) name of the CFC you're returning.

In Listing 1, we take a look at how to do this. It first shows our definition CFC (employee.cfc) that represents an "employee" structure. Then, we have a Web service (employeeService.cfc) that creates the structure, translates it into the employee CFC, and returns the CFC.

When the definition CFC is returned to .NET, .NET sees it as a type named after the interface component. In other words, the employee CFC returned in Listing 1 would be seen as an "Employee" object having directly accessible properties named EmployeeId, Firstname, and Lastname. On the .NET side, if the result was stored in a variable called "myEmployee," the employee's first name would be accessible as "myEmployee.Firstname." This is exactly as it would be in a ColdFusion structure.

Combining Arrays and Structures

By combining what we've seen of sending arrays and structures to .NET, it's simple to send more complex arrays or structures to .NET. For example, if you want to send an array of structures back to .NET, you first create an array and then populate it with CFC instances that hold the structure data. In Listing 2, an array named "result" is created, and each member of this array is an instance of our employee CFC. We then return the entire array.

On the .NET side, we first store the result in an object named "employees." Then, we loop over the members of the employee object with each being an instance of an employee. Knowing that the individual employee's data is stored in directly accessible properties, we can then print it out using simple "dot-syntax."

Queries: XML to the Rescue!

After covering arrays and structures, it should be easy enough to send queries to .NET. We could represent the entire query as an array, with each row using a facade CFC. Believe it or not, we can do better by returning ColdFusion queries in a format .NET already understands.

ColdFusion ReturnType	.NET Datatype of Result
Binary	Byte Array
Boolean	Boolean
Date	Date
Guid	String
Numeric	Double
String	String
Uuid	String

Table 1: Simple value equivalents in .NET

First, let me give you some background. The .NET equivalent of a ColdFusion query is called a DataSet. It's part of ADO.NET, which is the portion of .NET that deals with databases. DataSets are XML representations of database schemas. They can hold multiple queries, maintain relationships between the queries, and perform a slew of automated database tasks.

From a ColdFusion perspective, what's great about DataSets is that we can easily create XML that a .NET client can turn into a DataSet. Listing 3 shows an example of XML that can be used to create a .NET data set. You'll see that it's fairly simple. A root-level tag defines a "table" named books, with each "book" node describing a row in that table. Inside each book node, we describe each column. In this case, the columns are the "author" and "title" nodes.

Using ColdFusion MX's XML support, we can easily turn a query into this type of XML and return it to .NET as a string. To help out, I've created a user-defined function (UDF) that transforms queries into exactly this format of XML. This UDF is part of a utility CFC I've created called QueryTool, available on www.CFCZone.org or my site at: www.clearsoftware.net/client/queryTool.cfm. Listing 4 shows an entire solution for creating a query, transforming it to XML using this UDF, returning the XML, and finally the VB.NET client code for creating a DataSet from the result.

Compatibility and Performance

I imagine some of you are wondering how these changes to your ColdFusion Web services affect both performance and compatibility with existing ColdFusion-based clients. Tasks such as translating queries to XML are costly, and they should be performed only when necessary.

In my own applications, I'm using a two-tier approach to provide data to both ColdFusion and .NET clients. The first tier consists of a CFC with no modifications for .NET compatibility. ColdFusion applications can then call this component and its Web services with no changes. The second tier is also a CFC, used solely to provide data to .NET. It does this by instantiating the first-tier CFC as a component, calling the appropriate method, and then performing any necessary .NET-specific translations on the result. This way, any performance drains such as translating queries to XML only impact .NET clients, and I maintain compatibility with existing ColdFusion clients.

Consuming Data from .NET Web Services

Similar to ColdFusion, .NET makes it very easy to create Web services. Also similar to ColdFusion, it has no reservations about returning data in its own proprietary formats, like the DataSet.

Because of these differences in architecture, consuming .NET Web services is more difficult than consuming native ColdFusion Web services. In this section, we'll take a look at how to consume the various types of data returned by .NET Web services.

Consuming Simple Types

Again, ColdFusion and .NET see eye to eye on simple data types. The .NET string type, its various numeric types, and its date type all translate to their ColdFusion equivalents automatically. This means that consuming a .NET Web service that

returns a simple value is no different than consuming an equivalent ColdFusion Web service.

Consuming .NET Arrays

When dealing with arrays, things get a little more complicated. Before we look at how to consume a .NET array, we need to understand a fundamental difference between .NET and ColdFusion. ColdFusion is typeless, whereas .NET is a typed, object oriented environment. This means that every .NET variable is of a specific type, and all of those types inherit from a root "object" class.

For ColdFusion, this means that any complex data type sent over a .NET Web service will not be automatically translated into its ColdFusion equivalent. Instead, it will be translated into a Java object. This Java object will have various properties and methods, just like a CFC or COM object. You can CFDump the resultant object at any time to get a list of its properties and methods. If any of you have ever returned a CFC from a Web service, this should look very familiar.

Now that you know this, if you're calling a .NET Web service that you think returns an array, don't be surprised when your result is an object. Unlike ColdFusion arrays, .NET arrays are of a specific type, meaning that each element of the array is of the same type. Simple arrays can be arrays of strings, numbers, or even raw bytes. Elements in more complex arrays can contain any type of object.

If you CFDump the result of a .NET Web service returning an array, the object you receive will contain a method called "Get<type>()." In this, "<type>" is replaced by the name of the type of array you're expecting, such as "GetString()." If you call this method, the result is the native ColdFusion array you'd expect from a ColdFusion-based Web service. For a more complex example, if you expected a .NET Web service to return an array of our employee structures/objects, you'd look for a method called "GetEmployee()." This is because you're looking for an array of the employee type.

Consuming .NET Structures

Now that we know that complex data returned by .NET will be represented as Java objects, we can explore how to consume .NET's equivalent of a structure. It's actually very easy. If you've ever used a ColdFusion Web service that returns a CFC, it should be familiar territory. When you CFDump the result of a .NET Web service that returns an enumeration or other similar class, you'll see a number of "get" and "set" methods. These work like their equivalents in a ColdFusion Web service returning a CFC instance. As we're concerned with consuming, it's the "get" methods we're interested in.

If we consume the .NET equivalent of our ColdFusion GetEmployee Web service, we'd see a Java object with methods named "GetFirstname," "GetLastname," and "GetEmployeeId." Calling these would then return the Firstname, Lastname, and EmployeeId values or "properties" of this object. If we stored our result in a variable named "employee" and wanted to print out the employee's first name, we could reference it as `#employee.GetFirstname()#`.

Consuming .NET DataSets

.NET's database interaction library, ADO.NET, uses what is

Complete source code and asset management in Dreamweaver MX—now possible with Surround SCM.

Dreamweaver users know a beautiful Web-based product is only skin deep. Underneath, it's a tangle of hundreds or thousands of ever changing source files. Without a good development process and strong tools, bad things happen. Surround SCM can help.

Surround SCM lets you...

Track multiple versions of your source files and easily compare and merge source code changes.

Check out files for exclusive use or work in private workspaces when collaborating on a team.

Automatically notify team members of changes to source files—push changes through your organization.

View complete audit trails of which files changed, why, and by whom.

Associate source code changes with feature requests, defects or change requests (requires additional purchase of TestTrack Pro).

Remotely access your source code repository from Dreamweaver MX.

Surround SCM adds flexible source code and digital asset control, powerful version control, and secure remote file access to Dreamweaver MX. Whether you are a team of one or one hundred, Surround SCM makes it easier to manage your source files, letting you focus on creating beautiful Web-based products.

Features:

Complete source code and digital asset control with private workspaces, automatic merging, role-based security and more.

IDE integration with Dreamweaver MX, JBuilder, Visual Studio, and other leading Web development tools.

Fast and secure remote access to your source files—work from anywhere.

Advanced branching and email notifications put you in complete control of your process.

External application triggers let you integrate Surround SCM into your Web site and product development processes.

Support for comprehensive issue management with TestTrack Pro—link changes to change requests, bug reports, feature requests and more.

Scalable and reliable cross-platform, client/server solution supports Windows, Linux, Solaris, and Mac OS X.

Want to learn more?



Download Seapine's just-released white paper, **Change Management and Dreamweaver**, to discover tips, tricks and best practices for achieving full software configuration functionality. Visit www.seapine.com/whitepaper.php and enter code WM0604 to receive your copy today.

www.seapine.com
1-888-683-6456



macromedia
ALLIANCE PARTNER



called a DataSet. A DataSet is more than a query. It's an in-memory representation of a database schema. A DataSet can contain multiple tables and maintain relations between those tables. Each one of those tables can be thought of as an individual query.

What makes DataSets interoperable is that .NET uses XML to represent them. When calling a .NET Web service that returns a DataSet, exploration of the result leads to the discovery of a `getAny()` method. This method returns an array of objects representing the schema and data contained in a DataSet. These objects, in turn, have methods that expose their underlying XML. With a little bit of work, we can use ColdFusion MX's native XML support to translate this XML into ColdFusion queries.

Again, I've created a UDF for dealing with .NET DataSets. Listing 5 shows both the UDF and an example of its use. This UDF can be downloaded from my Web site: www.clearsoftware.net/client/convertDotNetDataset.cfm. Because DataSets can contain multiple queries, my implementation returns a structure. Each member of the structure is a query named after the table's name in the .NET DataSet.

Conclusion

As more software development moves toward Service-

Oriented Architecture (SOA), it's important to ensure that ColdFusion is interoperable with other development platforms. Although some things are out of our control (e.g., the changing Web service standards), I hope this article has shown you that it's fairly easy to move data between ColdFusion and .NET via Web services. ColdFusion can be an important player in your service-oriented application, both as a client and service provider.

Supplemental Code

A complete example of interoperable ColdFusion MX and .NET Web services, along with client code, is available at:

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

About the Author

Joe Rinehart is a full-time consultant with Advanced Solutions International, Inc., and a part-time independent consultant operating in the northern Virginia area. With five years of ColdFusion experience, he's now exploring .NET and how to make the two work together. Read more about this on his blog at: <http://clearsoftware.net>

joe.rinehart@gmail.com

Listing 1: Returning a structure to .NET

```
<!-- employee.cfc (definition cfc) -->
<cfcomponent>
<!--
  Declare a property for each key in
  an employee structure
-->
<cfproperty name="employeeId" type="string" />
<cfproperty name="firstname" type="string" />
<cfproperty name="lastname" type="string" />
</cfcomponent>

<!-- employeeService.cfc (Web Service) -->
<cfcomponent>
  <cffunction name="getEmployee"
    returnType="employee"
    access="remote">

    <cfset var employeeStruct = "">
    <cfset var employeeCfc = "">
    <cfset var i = "" />
    <!-- ColdFusion Structure -->
    <cfset employeeStruct = structNew()>
    <cfset employeeStruct.employeeId = "2">
    <cfset employeeStruct.firstname = "Isaac">
    <cfset employeeStruct.lastname = "Newton">

    <!-- Create and populate CFC -->
    <cfset employeeCfc =
      createObject("component", "employee")>
    <cfloop collection="#employeeStruct#" item="i">
      <cfset employeeCfc[i] = employeeStruct[i] />
    </cfloop>

    <!-- Return the CFC -->
    <cfreturn employeeCfc>
  </cffunction>
</cfcomponent>
```

Listing 2: Sending an array of CFCs to .NET

```
<!-- employeeService.cfc -->
<cfcomponent>
  <cffunction name="getEmployeeArray"
    returnType="array"
    access="remote">
```

```
<cfset var result = arrayNew(1) />

<cfset result[1] =
  createObject("component", "employee")>
<cfset result[1].employeeId = "1">
<cfset result[1].firstname = "Leonardo">
<cfset result[1].lastname = "DaVinci">

<cfset result[2] =
  createObject("component", "employee")>
<cfset result[2].employeeId = "2">
<cfset result[2].firstname = "Isaac">
<cfset result[2].lastname = "Newton">

  <cfreturn result />
</cffunction>
</cfcomponent>
```

VB.NET Client:

```
employees = employeeService.getEmployeeArray ()
For Each myEmployee In employees
  Response.Write("<br />EmployeeId: " & myEmployee.employeeId)
  Response.Write("<br />Firstname: " & myEmployee.firstname)
  Response.Write("<br />Lastname: " & myEmployee.lastname & "<br />")
Next
```

Listing 3: XML a .NET DataSet can understand

```
<books>
  <book>
    <author>Forta, Ben</author>
    <title>SQL in 10 Minutes</title>
  </book>
  <book>
    <author>Ben-Gan & Moreau</author>
    <title>Advanced Transact-SQL</title>
  </book>
</books>
```

Listing 4: Returning a ColdFusion query to .NET

```
<!-- employeeService.cfc -->
<cfcomponent>
  <cffunction name="getEmployees"
    returnType="string"
    access="remote">
```



```

<cfset var getEmployees = "" />

<!--- Get our Employees --->
<cfquery datasource="Northwind"
name="getEmployees">
    SELECT EmployeeID, FirstName, LastName
    FROM Employees
</cfquery>

<!--- Return XML string for this query --->
<cfreturn toString(queryToXml(getEmployees)) />
</cffunction>

<!--- Builds XML from a CF Query --->
<cffunction name="queryToXml" returnType="any">
    <cfargument name="inQuery" type="query">
    <cfset var result = "">
    <cfset var i = "">

    <cfxml variable="result">
    <recordset>
    <cfloop query="arguments.inQuery">
    <item>
    <cfloop
    list="#arguments.inQuery.columnList#"
    index="i">
    <cfoutput>

<#i#>arguments.inQuery[i][arguments.inQuery.currentRow]#</#i#>
    </cfoutput>
    </cfloop>
    </item>
    </cfloop>
    </recordset>
    </cfxml>

    <cfreturn result>
    </cffunction>
</cfcomponent>

```

VB.NET Client

```

Dim st As String
Dim dsOut As New DataSet

'Create XML Document from Result
ReturnDoc.LoadXml(employeeService.getEmployees())

```

```

'Create stream from XML Document
st = ReturnDoc.InnerXml
Dim io As New System.IO.StringReader(st)
Dim stream As New XmlTextReader(io)

```

```

'Create dataset from stream
dsOut.ReadXml(stream)

```

Listing 5: Consuming a .NET DataSet from ColdFusion

```

<!--- Utility function that converts a .NET dataset to a structure of
queries --->
<cffunction name="convertDotNetDataset"
returnType="struct">
    <cfargument name="dataset" required="true">

    <!--- Local Variables --->
    <cfset var result = structNew() />
    <cfset var aDataset = dataset.get_any() />
    <cfset var xSchema = xmlParse(aDataset[1]) />
    <cfset var xTables = xSchema["xs:schema"]
        ["xs:element"]["xs:complexType"]["xs:choice"] />
    <cfset var xData = xmlParse(aDataset[2]) />
    <cfset var xRows = xData["diffgr:diffgram"]
        ["NewDataSet"] />
    <cfset var tableName = "" />
    <cfset var thisRow = "" />
    <cfset var i = "" />
    <cfset var j = "" />

    <!--- Create Queries --->
    <cfloop from="1"
    to="#arrayLen(xTables.xmlChildren)#"
    index="i">
    <cfset tableName = xTables.xmlChildren[i].
        xmlAttributes.name />
    <cfset xColumns = xTables.xmlChildren[i].
        xmlChildren[1].xmlChildren[1].xmlChildren/>

```

```

<cfset result[tableName] = queryNew("") />
<cfloop from="1"
    to="#arrayLen(xColumns)#"
    index="j">
    <cfset queryAddColumn(result[tableName],
        xColumns[j].xmlAttributes.name,
        arrayNew(1)) />
    </cfloop>
</cfloop>

<!--- Populate Queries --->
<cfloop from="1"
    to="#arrayLen(xRows.xmlChildren)#"
    index="i">
    <cfset thisRow = xRows.xmlChildren[i] />
    <cfset tableName = thisRow.xmlName />
    <cfset queryAddRow(result[tableName], 1) />
    <cfloop from="1"
        to="#arrayLen(thisRow.xmlChildren)#"
        index="j">
        <cfset querySetCell(result[tableName],
            thisRow.xmlChildren[j].xmlName,
            thisRow.xmlChildren[j].xmlText,
            result[tableName].recordCount) />
    </cfloop>
    </cfloop>
    <cfreturn result>
</cffunction>

<!--- Invoke our Web Service --->
<cfinvoke
    webservice="http://localhost/employee.asmx?WSDL"
    method="getEmployees"
    returnvariable = "result">

<!--- Convert result to CF queries --->
<cfset result = convertDotNetDataset(result) />

<!--- Display --->
<cfdump var="#result#" />

```

Download the Code...

Go to www.coldfusionjournal.com

Don't Miss CFDJ's Next Issue!



Flex Your ColdFusion Muscles: Building rich Internet applications couldn't be easier with ColdFusion and Flex.

Thinking Outside the Table, Part 2: Reducing server load by storing search results in the database itself.

Case Study: A single CFC is used to create and manipulate image files directly from ColdFusion.

CF101: CFCs are still relatively new to many ColdFusion developers; this article examines one approach to using CFCs to help speed your development.

The State of the Fusebox Union

A great conference for CFers whether you're pro-Fusebox or not

The Fusebox Conference (www.cfconf.org/fusebox2004/) is the national Fusebox and Mach-II programming conference that Rockville, MD-based consulting firm TeraTech (www.teratech.com) hosts each fall. Fusebox is a framework with an associated methodology (FlIP), which is a way to create Web programs that are easier to maintain and deliver what clients really want rather than what they said they wanted.

Fusebox 4.1

I flew overnight from San Francisco and caught the Metro out to Twinbrook which is a short walk to the hotel. Very convenient. Unfortunately, I missed Michael Smith's welcome speech and speaker introductions and Hal Helms's keynote on the state of the Fusebox union. I caught the tail end of John Quarto von-Tivadar's "New Aspects FB 4.1" talk (which ended up being just a single session – he obviously raced through everything!).

Fusebox 4.1 introduces a number of enhancements over 4.0 with the biggest features being:

- Content variables on the `<include>` verb
- "One-stop" core files (which can be shared between multiple FB apps – and should reduce the temptation to modify the core files)
- Assertions at development time (I have long-standing reservations about any sort of assert mechanism, but FB4.1 automatically disables them in production, which is a step in the right direction)
- Better support for CFCs through `<instantiate>` and `<invoke>` verbs

How compatible is FB4.1 with FB4.0? Well, I took the FB4BlogMVC app (which I was using for my Blackstone demo)



By Sean Corfield

and changed `index.cfm` to use the new core files and it all worked exactly the same. I'd say that's pretty compatible. Oh, and FB4.1 in development mode is faster than FB4.0 in development mode because it actually checks file timestamps to see if files need reparsing and regenerating instead of blindly reparsing everything.

John also gave a look at a 4.2 feature: extending the grammar with your own namespaces (aka "lexicons"). This feature is enabled in experimental form in the 4.1 core files (so it is not "supported" yet). It requires quite a bit of knowledge about how the transformer works if you want to write your own verbs, but it does offer some very powerful options for FB developers. Good work John Q et al!

Fusebox Explorer, CFCs

I chatted with Maxim Porges of Westgate Resorts for awhile about JDBC connection issues in Java – which made us both realize how much CFMX does for us in terms of database connectivity.

Steve Nelson was hawking his wares as usual at a booth near the session rooms. I sat with him while he demoed his Fusebox Explorer extension for Dreamweaver. He has incorporated his test harness generator into the explorer, which is very impressive: write your Fusedocs, click a button, instant test harness! It makes the process so seamless and so easy that I expect it will encourage a great many Fuseboxers to (a) switch to Dreamweaver and (b) start writing Fusedocs and using test harnesses. Keep an eye on Steve's blog (steve.secretagents.com/) for more details about this.

In Barney Boisvert's talk on using CFCs with Fusebox 4, he explained the benefits of MVC (encapsulation, separation of UI and controller and model) and of using CFCs for the model (encapsulation, reuse). Then he showed an MVC app where UI validation and business validation were mixed (a typical scenario) and how CFCs let you cleanly separate the validation responsibilities – a very good example. There was a lot of good information in his talk and, with FB4.1 becoming available, I'm sure more people will start using CFCs in their Fusebox apps.

XML, Case Studies, Fusepanel, Networking

After lunch I attended Jeff Peters' presentation on XML and Fusebox Configuration files. Jeff gave an overview of XML and explained why it is a good thing. Then he walked through fuse-



FUSEBOX Conference

5th Annual Fusebox conference on
Fusebox 4 and Mach-II



box.xml and circuit.xml (this was a beginners track session), talking about each section of the XML and what it meant. It was a good clear explanation for Fusebox beginners.

Next up was a case study session with several folks talking about how Fusebox had helped them get a grip on complex, runaway projects. I was very impressed by Maxim Porges who talked about a fairly major rearchitecture of a spaghetti application to provide an OO layer (CFCs) that was then wrapped in a Fusebox 4 UI layer.

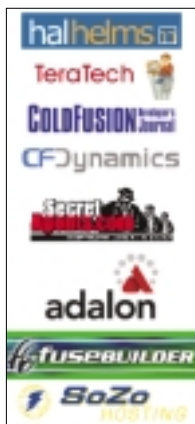
The Fusebox panel was next with the usual suspects fielding questions from attendees covering a variety of topics. I didn't take too many notes during this session so I can't remember what was being discussed... sorry... but it's always a good opportunity for folks to ask penetrating questions of the "inner circle."

I went back to my room for awhile to continue working on my Blackstone demo code and then went to the networking/Fuseball event in the hotel's gaudy nightclub (the music was mercifully quiet this time so we could actually talk). A group of us went to dinner and continued our Fusebox/ColdFusion discussions.

Blackstone, Mach II, and Ant

Sunday morning started far too early at 8:30 a.m. with my session on the next version of ColdFusion, codenamed

"Blackstone." I went over the public highlights of Blackstone and then walked through some feature demos using Sandy Clark's Fusebox 4 Blog example application (actually the version created by Brian Kotek that reworks it as a strict MVC application). The new Blackstone UI features were popu-



lar and the event gateway demo was very well received.

Next was a Mach II Birds of a Feather session, run by Ben Edwards. There weren't many Mach II users in the audience and I think some folks thought it would be more of an introduction to Mach II (which Ben was originally due to give but he gave up his slot for my session on Blackstone I think). Given the lack of other Mach II material at this year's Fusebox conference, I'm not sure how much folks actually got out of this session. Fortunately there was a full-day Mach-II class after the conference on Monday.

I caught part of John Paul Ashenfelter's session on leveraging open source Java tools (he was showing how powerful the Ant build system is) and I also stuck my head into the Fusebox tools session run by Matt Liotta. Steve Nelson was showcasing his awesome Fusebox Explorer for Dreamweaver.

More Helium


Then Matt Liotta and Howard Fore talked about He3, the new Eclipse-based ColdFusion IDE. They showcased some of the new functionality coming in Beta 2 soon (Beta 1 was launched at CFUN. Beta 2 includes substantial rewriting of core functionality according to Liotta). They showed SQL syntax highlighting and insight and said that in development they have dynamic SQL insight based on the actual tables, etc., in whichever database you are connected to. Their intent is to provide functionality close to the MS SQL Enterprise Manager directly within He3.

They also showed the new split-pane folder/file navigator tab, much like HomeSite, which was very well received by the audience. Beta 2 will also add code outlining (already present in CFclipse) and will provide a built-in Web browser (based on the embedded version of your platform's default browser selection). Probably the biggest win for the Fusebox community is the incorporation of a wireframing tool built directly into He3, complete with HTML page generation.

Other new features include a task list driven by TODO comments in the code validation based on doctype and, coming soon, a multilanguage tag insight (e.g., CFML insight for tags, but inside a <cfquery> tag you'll get SQL insight and inside a JavaScript <script> tag you'll get JavaScript insight and so on). No release date was given for the beta.

Matt and others then ran a panel on "Selling Fusebox," discussing ways to convince your boss (or clients) that Fusebox is a good way to go. Michael Smith closed the conference with the usual round of thanks to sponsors, speakers, and attendees, and a selection of raffle prizes. My evaluation form came home with me so no chance of prizes for me...my bad!

Overall, a great conference for CFers regardless of whether you are pro-Fusebox or not. If you're a Fuseboxer, you'll learn a lot of best practices for Fusebox. If you're new to Fusebox you can attend the beginners track and get some great introductory material. If you're anti-Fusebox, you can at least find out why some people love it so much. You never know, it might just open your eyes to what Fusebox (and FLiP) has done for some people. I certainly came away from the conference very excited about the new release of Fusebox.

Now it's time to start looking forward to next summer's CFUN-05 (www.cfconf.org/cfun-05/), which promises to be even bigger and better than CFUN-04 was. 

About the Author

Sean Corfield is director of architecture at Macromedia and has worked in IT for over 20 years. Sean is a staunch advocate of software standards and best practice, and maintains the Macromedia ColdFusion MX Coding Guidelines. Recently Sean has become a staunch advocate of Mach II. You can reach him at www.corfield.org/.

sean@corfield.org

BlueDragon 6.1

It's getting better all the time

If you haven't heard of BlueDragon, you're either new to the ColdFusion/CFML community or you've been under a rock somewhere. Back in September 2002, New Atlanta forever changed the CF landscape when it introduced BlueDragon 3.0.



By Phil Cruz

As a quick overview, BlueDragon is a family of server products that allows you to deploy CFML applications as an alternative to using Macromedia's ColdFusion MX. For more background, I refer you to the original product review I did for *ColdFusion Developer's Journal*, www.sys-con.com/coldfusion/articleprint.cfm?id=546. For this article, I am going to focus on what's new (and improved) with BlueDragon 6.1, which was released in June of this year.

To paraphrase Steve Ballmer's now famous rant, BlueDragon 6.1 is all about "Compatibility, compatibility, compatibility, compatibility!" For the 3.0 release, BlueDragon was targeted at ColdFusion 5 level compatibility. However, CFMX was already on the scene and developers wanted to take advantage of the exciting new features it provided. With the release of BlueDragon 6.1, New Atlanta has achieved an impressive level of language/feature compatibility with CFMX. BlueDragon now supports major CFMX features such as ColdFusion Components (CFCs), CFFUNCTION user defined functions, XML, and Web services.

More and more, CFML developers are recognizing the benefits of using application frameworks in the development process. BlueDragon does not disappoint in this regard. Mach-II, a complex CFC-based application framework, is supported on BlueDragon 6.1. Fusebox developers should take note that BlueDragon 6.1 runs on FB 3.0 and FB 4.0.4 without any modifications. Fusebox 4.1, which is in beta as of this writing, will be supported in BlueDragon 6.2, which is also in beta.

There are, of course, still some differences between CFMX and BlueDragon. There is a short list of unsupported tags (most of them related to CFGRID and CFCHART). There is a longer list of tags that are "supported with limitations." Some of the more notable issues are lack of support for HTTPS protocol in the CFHTTP tag, lack of support for username/password in the CFMAIL tag, and lack of support for COM objects in CFOBJECT. (I understand these issues have been addressed in the 6.2 beta. More on that later.)

The compatibility issues are very well documented in the "BlueDragon 6.1 CFML Compatibility and Reference Guide," available on the New Atlanta Web site. Read the document

thoroughly and you'll be aware of any potential issues that may arise. In addition, check the BlueDragon mailing list archives as others have discussed and posted workarounds to certain issues. Test early and often, and you should have no trouble deploying your application on BlueDragon 6.1

If you're looking at using BlueDragon to serve your Web site, it's a good idea to check for vendor support for any third-party applications you want to implement. InFusion Mail Server and

Tracking Tools bug tracking applications are examples of applications that are supported on BlueDragon.

(Disclosure: The author is the developer of Tracking Tools). The FuseTalk discussion forums, however, are not. FarCry, a popular, open source content management system, is also not currently supported.

Fully Featured Free Server

When trying to sell a CFML-based solution or defend CFML against other languages like PHP or ASP, one of the main points of contention is the cost of the application server. I won't go into the whole total cost of ownership issue, since others have already covered that ground, but suffice it to say justifying this cost can be a challenge to budget-strapped organizations. In February 2003, New Atlanta again changed the game by introducing a free edition of BlueDragon 3.1 Server. Although this product was head and shoulders above Macromedia's ColdFusion Express, it still held back certain key features like CFOBJECT, and CFWDDX. The lack of these key tags was still a showstopper for some developers to use the free server. With BlueDragon 6.1, New Atlanta really showed their support for the CFML community by removing all tag restrictions for the free edition. The primary restriction of the free edition is that it's limited to database connectivity via ODBC, MySQL, or PostgreSQL drivers. Now CFML developers going up against a LAMP (Linux/Apache/MySQL/Php) solution can respond with a LAMBDA (Linux/Apache/MySQL/BlueDragon) solution that is just as competitive.

If you need some of the features not supported in the free edition (integration with JSPs, ability to run pre-compiled/encrypted templates), you will need to purchase the JX or J2EE edition. When BD was first released it was offered at an introductory price of \$549 per server. This promotion has since expired and the JX edition is now offered at \$899 per server. Visit the New Atlanta Web site for full pricing details as well as a comparison matrix for the different editions.

If you don't want to run your own server, several hosting providers have stepped up to the plate to offer BlueDragon

hosting. Check the Web site for links to their hosting partners. Finding a suitable host should not be a problem.

Enhancements

With 3.0, New Atlanta showed they were not only going to implement the CFML language but they sought to improve and innovate. They did this by introducing CFFORWARD, a tag not available in ColdFusion. Since then, they have continued this effort and introduced the following tags: CFIMAGE, CFIMAP, CFASSERT, and CFDEBUGGER. Another notable enhancement is that ColdFusion Component (CFC) instances can be serialized. This would be useful in clustered environments. I'll refer you to the product documentation for details as to these tags and other enhancements. I actually haven't used them (preferring to write "vendor neutral" code), but the point is it demonstrates how New Atlanta is willing to listen to their customers and provide additional functionality.

I mentioned previously that COM objects were not supported in BlueDragon. Well, one notable "enhancement" on the horizon is a .NET edition of BlueDragon. This edition will allow you to deploy your CFML on the .NET platform just as the current offerings allow you to deploy to the J2EE platform. You'll not only be able to migrate your CFML to the .NET platform but you'll be able to integrate with and extend your existing .NET applications. I'll be watching this closely, but New Atlanta has already released a beta version that looks very solid. If your company is looking to migrate to .NET you'll want to keep this on your radar as well. This provides yet another option for CFML developers.

Support


From the initial release, New Atlanta has always provided outstanding support. It's nice to see that as they continue to grow and succeed, the quality of their support has not decreased. They are very responsive to questions on the BlueDragon mailing list. Throughout their beta program they have kept their bug database open to the public. As a developer, I really appreciate this service. It helps to know your issues are being heard and you can keep track of the status of bugs that affect you. I hope they continue this practice and wish more companies would adopt it.

What About Blackstone?

Blackstone is the code name for the next major release of CFMX that is currently in beta. From what's been publicly released about Blackstone, there are some exciting new features in store for CFML developers; things like event gateways, rich forms, and PDF generation. These are exciting times. The obvious question is "Can BlueDragon keep up with Blackstone?" It's too early to say, but personally, New Atlanta answered that question for me with 6.1. I consider it a huge accomplishment that they achieved such a high level of compatibility with CFMX. Having done that I'm not sure that keeping up with Blackstone will be that much different. (The folks from Macromedia may beg to differ!) Perhaps, considering some of BlueDragon's features/enhancements (source-less deployment, standard WAR file deployment) and the upcoming .NET edition, the question should be "Can CFMX keep up with BlueDragon?" I understand from New Atlanta that their attitude about Blackstone is "wait

and see." After Blackstone is released, if certain features are in high demand, they will build it. I think this is a reasonable strategy and avoids expending resources on less popular tags just for the sake of compatibility. Bottom line, the competition between the two companies is a win for developers.

Summary

BlueDragon 6.1 is a very compelling release that provides a high level of compatibility, fully featured free server, and popular enhancements. That, combined with support for the Mach-II and Fusebox application frameworks and their exceptional technical support, New Atlanta has transformed BlueDragon from an interesting product with lots of potential into a mature solution that stands right up there with ColdFusion MX. It's a great time to be a CFML developer and it's getting better all the time. 

About the Author

Phil Cruz is a Macromedia Certified Advanced ColdFusion developer and has over 12 years of experience in the computing industry. He is responsible for www.mach-ii.info, a community site for the Mach-II framework. As a micro-ISV, he created Tracking Tools, an easy-to-use bug tracking application built with Mach-II (www.tracking-tools.com).

philcruz@gmail.com

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

Java for ColdFusion Programmers?



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

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

Damon pointed out that “the server team is more committed than ever to not releasing any feature if they don’t have proper time to test it.” He also made a point of explaining that Blackstone is going to be a very stable release. Its features are built on top of the ColdFusion MX 6.1 code base with the Updater and hot fixes. This release has been extremely focused on quality: there are 2,000+ beta testers and some 2.2 million lines of code in a customer application repository that they are using to test backwards compatibility (without changing a single line of code in any application). In Damon’s words, this release is being “seriously punished and will be ready for immediate prime-time production deployment when it leaves the building.”

A large portion of the discussion was centered on the motives behind the new features that are in Blackstone. What factors went into determining what was on this list of features to add to the server? Tom summed it up best when he said that Blackstone, in a nutshell, is all about addressing two main goals. First and foremost, it is all about addressing customer needs. Macromedia has invested a lot of time and money in listening to customers on site, in the forums and the mail lists, and at conferences and user groups, to find out what they want. Second, Blackstone is all about allowing developers to easily do things that they cannot easily do anywhere else. Without a doubt, Blackstone achieves both goals. Jim Schley pointed out that a major goal was to allow ColdFusion developers to reach new platforms. The ability to leverage and communicate with new platforms – Flash, SMS and other protocols, or other Java applications via JMS – is in suit with both goals expressed by Tom. We developers want and need it, and it’s something that no other Web application server technology lets you do (easily or otherwise).

Flash Forms and the Event Gateway

We spent a lot of time talking about two features in particular: Flash forms and the event gateway. Many of the developers on the team not only write Java code, but CFML as well. For example, Mike has been responsible for the ColdFusion Administrator application

for many years and he wrote the Macromedia CFML Pet Market application, to name a few. Mike best described the reasoning behind Flash forms when he said that his forms have looked the same for five years and that when Flex was released he was finally presented with a better way to create forms; why not extend that same functionality to ColdFusion developers? He said the goal was to “allow developers to very quickly create very pretty forms.” When I was discussing general motives for features, Dean described one of their goals as “making things that are currently a pain, easier.” The new features for creating Flash forms and XForms are definitely consistent with this goal.

As I said, we also talked a lot about the new event gateway. The event gateway is a framework that allows direct communication between ColdFusion applications and Java components via CFC “listeners” and Java “gateway applications.” Blackstone will likely ship with many prewritten gateway applications and there is a framework that gives developers the ability to write their own as well. With the introduction of this gateway, developers are no longer bound by the constraints that have always plagued ColdFusion applications. Specifically, functionality no longer needs to be dependent on challenge and response (CFC methods can respond to events triggered by a gateway application); is no longer dependent on the HTTP protocol (gateway applications support all of the protocols supported by Java such as POP, SMTP, JMS, SMS, telnet, etc.); and any other features of the Java language can be leveraged (such as the ability to monitor the file system, for example). All of the team members agreed that the gateway might be the most radical feature ever to be introduced to ColdFusion developers. There’s no telling what creative uses developers are going to come up with for the gateway; the sky’s the limit.

Of course, there’s a lot more to Blackstone than just Flash forms and the event gateway. According to Tom, in addition to the “popular” features like reporting and the event gateway, the team also tried to cherry-pick features that they’d like to see added whenever they could. In his words, “CF has always

been about CFML and about making it easy to develop powerful applications with CFML.” The result is the addition of subtle features that make day-to-day life easier, such as the addition of support for Java null values. In fact, getting back to when I was discussing the motives behind the new features, Mike also said several times that a major goal was “to make developers’ lives easier.”

Fixing Bugs

Okay, so those are some of the motivating factors for features in Blackstone. I also discovered that, despite what Macromedia is currently making public knowledge, Blackstone isn’t only about new features. Bettering performance was a major goal for Blackstone; the development team spent a lot of time testing and optimizing the server to perform even faster than any prior version. Fixing bugs was also a huge part of the process of developing Blackstone; Jim Schley and Jim Murphy both talked with me for a while about the fact that in this release they’ve not only been addressing all of the bugs found in new features so that the features ship in great working condition, but also some old bugs in the system that they’ve wanted to tackle for some time. Thousands of bugs have been fixed so far. The team has also concentrated on making it easier to learn and use these features. A lot of time has been spent documenting and creating good examples. The CF Administrator will not only have a new look and feel, but will ship with new sample applications that take advantage of many of the new features in Blackstone.

While we’re on the topic of fixing bugs, we also discussed what ColdFusion developers could do to help make the product better. Jim Schley and Tom Jordahl both emphasized that people should use the Macromedia livedocs. The development and QA team pay a lot of attention to the comments that are submitted to livedocs and livedocs is also oftentimes the first place that product documentation is updated. They made it clear to me that this is one of the most underused resources for developers. Damon, Mike, and I also had a long discussion about the development process at Macromedia. They have adopted a system of one QA person to

one development person working on a given feature. Having a two-man team that handles development and testing together for a specific feature has allowed them to achieve terrific successes in their development process. I personally have never worked in such an environment, but it intrigues me and I'll be sure to try that myself on the next project I oversee. They both also emphasized the importance of bug submissions. Mike and Damon want beta testers for Blackstone and developers using public releases of the software, to never hesitate to submit bugs. The team looks at every submitted bug and according to Mike, "I'd much prefer to spend 10 minutes only to discover that a bug submission is a duplicate than have bugs not be submitted at all." So there you have it, people, submit your bugs!!

The Future of JRun

One of the questions I was curious about was the future of JRun. I get nervous when I hear a lot of talk about making CF better and no talk about making

JRun better. It could be said that a J2EE application, after all, is only as good as the app server it runs on. Fortunately, Dave Gruber is not only the manager for ColdFusion but JRun as well. Damon, being the director of engineering for Blackstone, also pays a lot of attention to the underlying application server. They explained that JRun is now at the heart of so many products. ColdFusion, Breeze, Flex, Flash Comm, and JRun itself of course, all depend on Macromedia constantly evaluating the features and performance of JRun. In fact, JVM and database driver optimization were examined, optimized, and tested extensively for the upcoming release. A lot of time is spent evaluating not only the current product, but also evaluating new initiatives in the Java world, whether it's a new specification from Sun or a new open source initiative or methodology. They are committed to growing JRun along with the entire suite of server products and in keeping it at the heart of everything they do. Just because some products, like ColdFusion,

are pure J2EE applications that are capable of being deployed on other third-party application servers doesn't mean that JRun will not be constantly improved and kept competitive. Far from it, in fact.

Life After Blackstone

I promised that I'd talk about what's coming after Blackstone. We spent a good deal of time discussing what the next release may or may not look like. Unfortunately, some things were said that can't be mentioned here, but many things can. Nobody knows exactly what will be in the next release after Blackstone. Every one of the developers I spoke with had his own personal list of features he'd like to see implemented. I think it's safe to say that the event gateway is certainly going to play a large part in the next release. This feature opens up the server to the addition of so much functionality and it's brand new. It was great to see the team so excited about a feature as well as so curious to see what developers decide to do with it.

Be the 1st to develop with Blackstone

HostMySite.com will be the first host to offer the innovative new version of ColdFusion.

Be the first, visit hostmysite.com/blackstone

 **HostMySite.com**

1-877-215-4678

They are also looking at the open source initiatives to see what additional functionality could be dropped into the server that would be useful for developers. The server itself already uses Axis, log4J, JAAS, SAX, and several other standards and open source components in order to add functionality to CFML developers and the server itself; and there are so many other Java technologies that could also be leveraged. The developers also pay attention to what goes into the DRK CDs. DRK applications like the CFIMAP and CFIMAGE applications are definitely candidates for functionality that might be incorporated into the ColdFusion Server. Dean mentioned that one thing he'd personally like to see (and I also think it'd be useful) is some sort of hook into the service metrics. The sky really is the limit... and ultimately it really is up to the development community what makes it to the next release. The one underlying theme, which was evident when I spoke with the team about features in the future, is that, now more than ever, they are driven by customer needs, requests, and feedback. Get involved in the beta programs, use the forums, and submit requests on the wish list – your voice is not falling on deaf ears!

A Few Examples

After the interview, I sat with Damon and Mike for a while and played with some code on Damon's laptop. He wanted to show off a few of the new features they are working on. First he showed me a ColdFusion application that has a menu – you pick a menu option and see the appropriate data. He selected option #2 and displayed Macromedia's current stock quote value. Doesn't sound impressive? Then he ran a batch file and reloaded the page. Everything still worked fine. We went back and took a look at the source code – there was no source. His batch file used a new feature to compile the CFML code into binary Java code. Yes, that's right, sourceless deployment! Still not impressed?

Damon then opened a DOS prompt and brought up the same menu – via a telnet connection with the server! I know what you're thinking – "so what, I don't think my clients will be very impressed that I can let them access

data from a DOS prompt and telnet." Maybe you'll be impressed by the fact that he then took out his cellphone and retrieved the same information by sending his laptop an SMS message? Very cool indeed – Damon's menu application demonstration was an example of how you can use several different clients (using a variety of protocols) to access the same information via the event gateway. SMS and telnet are just two supported protocols... there are many more if you need them.

Then Damon showed me an amazing example of how one of the new event gateways can be used to boost application performance. First he showed me a CFM page that creates a text file on his local system. Actually, it loops 100 times and creates 100 text files in a directory (by including a template that has nothing but a CFFILE call in it). He ran the page, everything worked, and the execution time was somewhere in the neighborhood of 21,000 milliseconds – that's 21 seconds to a layman. He then changed one line of code. Rather than including the code that creates a text file within the loop, he tells an "asynchronous gateway" to execute that file.

"The gateway might be the most radical feature ever to be introduced to ColdFusion developers"

Calling the gateway, like performing an include, also took only one line of code, and the CFML that creates the text file(s) on his local machine remained exactly the same. This time the page took approximately 230 milliseconds to run. That's right, from 21 seconds to less than a quarter of one second to run the same code! How? ColdFusion usually executes CFML from the top-down and does so in a synchronous manner. This means that it waits for one thing to finish before moving to the next. In this example it would create a file, wait for the file to finish being created, loop, and then repeat the process 99 more times. The asynchronous gateway allows CFML files to be executed in an asynchronous

manner, meaning that ColdFusion does not have to wait for one piece of code to run before moving to the next. The loop took 230 some-odd milliseconds because that happens to be the amount of time it takes his server to loop 100 times (with a fraction of a millisecond thrown in on each loop in order to make an outgoing call to the gateway). Very impressive indeed, and this is just a sample of what can be done with the new features in Blackstone.

Conclusion

My overall impression from sitting with the Blackstone Team is that they are very excited about this release. Several times I heard them say that with Blackstone they are getting back to ColdFusion's roots – back to creating great features that make it easy for CFML developers to do things that nobody else can do... and I believe them. There's so much more in the server that you'll hear about in the near future – from PDF and Flashpaper generation to new reporting tools and options, from interacting with a wide range of devices to making simple day-to-day tasks even easier. They also gave me a better understanding of what the development community can do to help shape CF – I only wish I could motivate more people to share their ideas with Macromedia because they're obviously listening. As I mentioned in this month's editorial, plans are underway to get the Blackstone team more involved in *CFDJ*, including a special issue covering the new features, with articles by the developers who built them! Most important, I came away with an appreciation for just how hard the team has worked and how excited and proud they are of what they've achieved. I share their excitement about the future of ColdFusion and can't wait to see what developers do with it when it's released. Though it's remaining true to its roots, ColdFusion development, as we know it, is about to change forever. The future looks bright, and I for one cannot wait.

Special thanks to Dave Gruber, Damon Cooper, Tom Jordahl, Jim Schley, Jim Murphy, Mike Nimer, and Dean Harmon for taking time out of their busy schedules to sit and chat, and to the entire Blackstone team for giving us all something to look forward to.



One damn good Cold Fusion hosting firm!

Webcore Technologies specializes in ColdFusion, ASP and SQL hosting. We offer these solutions in both dedicated server and shared virtual environments.

Webcore can design and implement clustered or load-balanced solutions, managed firewalls, VPN's, multi-site failover, and more. In addition, we also offer corporate Internet access, web site design, and technology consulting services.

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



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

GOING TO THE MAX

I don't think we're in Utah anymore...! Macromedia is all about the user experience, and boy, did they get it right this year! New Orleans and ColdFusion developers go together like red beans and rice, jazz and the delta, crawfish and the bayou. What a conference! There's a new energy buzzing at Macromedia and I can't think of a better place to unleash it than in New Orleans.

I've been hearing about Blackstone ever since the 2002 Macromedia conference when they released ColdFusion MX. Even then, when they were introducing us to CFCs and Web services, they were tempting us with Blackstone, asking us to be patient while they secured the foundation – and it looks as though they are about to deliver. The leap from ColdFusion 5.0 to ColdFusion MX had more to do with securing the infrastructure than new features. Of course MX introduced features that catapulted us to a new level as Web application developers – object-oriented programming on a J2EE-based foundation. At MAX this year, they gave us a glimpse not only of the features they have since built on that foundation, but how Blackstone fits into the Macromedia vision of the future.

If you missed MAX, then I hope to give you a glimpse not only of the conference itself, but also of the exciting new products and features for ColdFusion developers, as well as share with you where Macromedia is headed in the near future, and share an important message from chairman and CEO Rob Burgess.

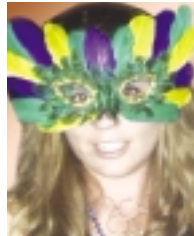
Before MAX officially kicked off, I decided to sit in on the Community College and get a little inside scoop. Community College is an event held by Macromedia for User Group managers and Team Macromedia Members. Each session focuses on a specific Macromedia product and is given by the engineers directly

The 2004 convention highlights

responsible for the design and implementation of the software. The sessions are small, informal gatherings that allow the engineers to engage directly with their audience, meant to represent the ColdFusion community. The participants can talk about any issues they have encountered while using the products and discuss what features they would really like to see included in the next release. They are then presented with a “sneak peek” of the features either currently being developed or in the planning stages. The group gives the engineers very specific feedback on these features including which ones they liked or disliked, and what features they felt were more important than others. The engineers did more than just take note of the suggestions, but discussed the reasons and implications in-depth and usually gave a pretty honest estimation of what they would consider including, and what was not going to be possible. Unfortunately Macromedia has sworn me to secrecy (literally) and I cannot divulge the details of these sessions, but I can tell you that it provides a strong argument for getting involved in the Community at a CFUG or Team Macromedia level.

Welcome Reception

The welcome reception was at 5 p.m. on Monday night at the conference center. Macromedia gave the event a little New Orleans flair with a band playing live jazz and overflowing mounds of jambalaya. The strolling servers and open bars were interspersed with the sponsors handing out beads and other fun toys to lure the somewhat socially lubricated attendees to learn more about their products. While the open bar, cyber café, and community pit are a definite plus, the networking seems to be the main draw of the reception. In fact, most of the people that I spoke with cited that as the main reason for coming to the conference. Some have a vested interest in a particular product,



By April Fleming



Fun times at MAX

some come to learn from the sessions themselves, but most are coming to learn from the other developers in the community, and to make connections that can deepen their understanding of the technology and see how their colleagues are pushing the boundaries and opening new doors for developers. Of course, there are always the purely business or purely personal types of networking going on as well, but I'll leave that at the conference!

Another main attraction is the opportunity to mingle with the Macromedia staff, instructors, speakers, and even some of the product engineers. I have to admit that Macromedia has really come a long way in making themselves available to the community. I remember a time when finding out you were in the same room with someone who had a hand in writing the nuts and bolts of the ColdFusion server was an extremely rare treat. This year they were all around, mingling with the crowd, and no matter how much they enjoyed the open bar, there was still no getting the Blackstone release date out of them.

A Preview of Blackstone's Features

The very first session that I attended on Tuesday morning was the "Future of ColdFusion: Blackstone". This session was presented by Tim Buntel, senior product manager of ColdFusion, who has been managing the development of the ColdFusion server since version 5.0 (and another Macromedia insider I frequently found mingling with attendees during the conference.) While Blackstone is still in development, we were given a preview of the features which are currently in Beta 2, which is already being used by some of the more adventurous members of the ColdFusion community. These features are of course subject to change. Buntel describes Blackstone as the most ambitious release of ColdFusion to date in terms of development hours and requested features.

We started off with a look at how Blackstone will help solve common problems ColdFusion developers face today; structured business reports, printable/portable documents, rich forms, application events, charting/graphing, and verity searching. With the new reporting features, Macromedia has vowed to do away with the third-party application woes many of us have experienced integrating reports into our applications. They



MAX conference attendees networking in the Community pit

have built not only a reporting engine, but a report builder right into Blackstone. You can seamlessly move back and forth between Dreamweaver and the report builder. The report builder allows you to create a template for the look and feel of your report. Blackstone binds the template to the data at run-time, producing banded reports, with a "pixel-perfect layout".

Blackstone includes two new output types, PDF and flashpaper. These output types can be used for reports or any other block of text. Switching between PDF and flashpaper is as simple as changing the value of an attribute. The cfdocument tag is Blackstone's answer to the "printer-friendly" dilemma we have all faced at one time or another.

In addition to specifying "PDF" or "flashpaper", cfdocument also allows you to specify the output for "print." This will retain all of your formatting and render the document suitable for printing as compared to the existing printer-friendly versions, which must strip out all formatting leaving them printer-friendly but not so user-friendly.

These new output types are a very nice feature, unfortunately, there is no PDF output for forms. This was the hot question in all of the printable/portable document discussions – maybe next time!

Blackstone's rich forms feature results in complex forms such as tabs, accordions, calendars with related selects, and pick lists built right again. The output options are "swc", which require the user to have Flash Player 7, or Xforms, which generate HTML, DHTML, JavaScript and CSS, and utilize skins. Blackstone will ship with predefined skins, or developers can create their own. Here's a word of advice: Brush up on your XSLT! Like the cfdocument tag, creating these forms is as easy as incorporating a few ColdFusion tags. Just as with the reports, you can use styles and apply skins to your forms. And, most importantly, state is automatically maintained for you.

Another long-awaited feature from Blackstone is the application events feature. Developers will soon be able to capture those ever important "on application start/end" and "on session start/end" events.

Charts and graphs have come a long way, and can be embedded into the ColdFusion reports. They include named styles, a builder, animation, and require at least Flash Player 6.

Verity can now run on a separate machine and includes highlighting, spelling suggestions using an internal dictionary, and searching through search results.

Some management and deployment features include the ability to run multiple instances of the ColdFusion server on a single piece of hardware! These instances are easy to create and can be clustered. Applications can now be packaged to EAR/WAR files and run on J2EE with or without the CFML. An administrator API has been included that allows you to do anything programmatically that you can currently do in the administrator. Imagine building a database and creating the datasource on the fly!

Of all the features that I have outlined so far, the one to really watch out for is the event gateways. Event gateways have only just begun and will lead Macromedia into its vision of the future. Tim Buntel described event gateways as "the most exciting thing for ColdFusion." For years the single purpose of the ColdFusion server was HTTP. Now with event gateways we can move beyond the browser. CFML will be able to respond to events from mobile phones using SMS and interact with

messaging clients, SMTP, TCP/IP sockets, and file systems. According to Buntel, "ColdFusion making these things easy is making it a very exciting time to be a ColdFusion developer."

Besides the event gateways, the Flex components built into Blackstone were the selling point for me. So what about Flex? I know that before I attended MAX, I was a little confused about Flex and how it related to me as a ColdFusion developer. What I discovered is that Flex gives the ability for a developer like me to create full-blown desktop-like Flash applications. I still need a back end application server, it doesn't have to be a ColdFusion server!

One goodie they do add every year for the die-hard ColdFusion developer is the ColdFusion "Birds of a Feather". I was unable to attend BOF due to scheduling conflicts this year, but I did have an opportunity to discuss this year's BOF with Shlomy Gantz, president of BlueBrick, Inc, and well-respected member of the Macromedia speaker circuit. I'll let Shlomy tell you in his own words why he made a point to fit the BOF into his busy MAX schedule.

"Getting the entire ColdFusion team in one place, you're almost sure someone is going to slip and talk about something they shouldn't. I wanted to be there when that happened. Knowing what's coming and having the ability to talk directly to the ColdFusion team is a rare occasion, so I wanted to make sure I did not miss that opportunity."

Of course, I tried to find out if they did in fact let anything slip, but Shlomy wouldn't give. I guess that's why next year I'll have to be sure to fit this one into my schedule as well.

One thing that is always sure to make it into my schedule is the MAX Special Event. This year the event was held at Mardi Gras World, a large warehouse housing floats and props from Mardi Gras' past and present. The main dining hall was lined with tables full of food, and of course, an open bar. We were all pleasantly surprised by the cover band, and by the end of the night my fellow developers were getting down to the likes of Aretha, Otis Redding, and other classics. The band even recruited some new backup singers! It took most of us a good hour to discover that there were three more rooms just like it, only instead of a live band there were psychics, palm readers, and tarot card readers. Some people were having caricatures drawn by local artists. The event drew a great crowd and it was a wonderful opportunity to dismiss our safety net of "tech talk" and really have some fun! Of course most people were still going strong when the event ended, and the Macromedia crowd took over Bourbon Street.

Keynote Speakers

I entered Tuesday's keynote, weary from work, travel, time change, and my first Halloween in New Orleans, and I left completely fired up, jumping aboard the Macromedia bandwagon and holding on for the ride. The first keynote was started off by Steven Elop, Macromedia COO. He shared Macromedia's commitment to the "rich user experience" explaining that they chose MX for the last release of software because the "MX" represents for them "great experience". Macromedia believes that while the Internet made amazing strides in technology in terms of connectivity and communications, it took us backwards in quality and style of user experience. Macromedia has taken the reigns now to transform our beloved Internet into a rich user experience combining the best of

the desktop, best of the Web, and best of communications. They did not just tell us that this was their vision, but they showed us their vision. Elop gave demos of several applications utilizing Flex to provide that rich user experience. He then outlined how this rich user experience translated into actual dollars, citing return on investment figures for the companies currently implementing their latest software – figures like one-half of miniusa.com's leads now being generated from their Web site, ROA for Ralph Lauren's TV application returning over 300% investment since it launched, and Avery.com now 20% faster since switching from Java.

He then outlined Macromedia's future for non-PC devices: cellphones, Wi-Fi, GPS, and PDAs. Between Blackstone's event gateways and Macromedia's commitment to incorporating the Flash Player into these devices, we not only have the development capabilities, but the devices on which we are able to deliver the new user experience. Another great stride is being made in e-learning and distance learning contrasting the old technology such as WebEx against the experience offered by one of Macromedia's newest applications, Breeze.

Elop then switched gears from the user experience to the Macromedia customer experience (that's us). Macromedia has been listening to the development community in an effort to determine what we need from them, so that we may in turn go out and apply these new technologies to the end user. He approached the audience as partners – what they can do for us, what we can do for them, and discussed how to ensure our future and stay on the leading edge of technology. He described his position as one that ensures Macromedia is in fact listening to the developer community and that they have been listening differently in the last year, engaging in "deep dialog" with customers. Macromedia has, in the last year, started sending Q/A engineers on the road to meet with customers and is now more confident that they are headed in the right direction. There are currently 2000 users actively beta-testing Blackstone, and they are currently working with 500 companies incorporating Flex to take their applications to the next level. They are committed to more community support in the way of involving support and Q/A in User Groups and blogs. He stressed the amount of information (sometimes controversial) available in the blogging community and encouraged us to get involved. This year at MAX, there were over 300 people from Macromedia directly interacting with attendees. You not only saw this at the sessions, receptions, and events, but also at the product support lab that was set up in the community conference hall. The product support lab remained stocked with support, Q/A, and engineers ready to talk one-on-one about specific issues related to Macromedia products.

Chief software architect and executive vice president Kevin Lynch then took the stage and showed us, even more, what we developers can do with Macromedia products. Flash Player, critical to the widespread adoption of these products, is now the most widely distributed software on the Internet. It is updated faster than XP, Internet Explorer, or Java. He talked about the Flash Player video technology and that it is now the number one video player on the Internet. He explained how companies like Toyota, Phillips Home Entertainment, and Red Bull are taking advantage of these new Flash capabilities. He demonstrated how Vodaphone, NAA, Nike, and TJ Maxx are



Mardi Gras world

using the new “guided selling” capabilities. And, how New York state, BBC news, and the University of North Carolina are incorporating visual analysis into their day-to-day business, thanks to Macromedia technology. He explained how Mazda and Seven Worldwide are using Flex to create call center applications and ERP systems, creating the largest Flash applications developed, incorporating handheld devices and processing over 100,000 invoices a day. Salesforce.com is increasing its leads using Breeze, and Microsoft and IBM are incorporating Flash video into their existing Web sites.

Ben Forta, senior technical evangelist and Tim Buntel were then introduced to officially hail the coming of Blackstone. In what appeared to be an effort to reassure developers who may be a little skeptical of jumping right in with a brand new release, they showered us with the following statistics; Blackstone spent 15 months in development with over 20,000 development hours, currently in beta with over 2,000 beta-testers, seven months of customer testing, and they have performed over 15,000 regression tests. Then they gave us a glimpse of the new wave of application development possible with Blackstone – the Ballot Application. We were asked to text our vote for President of the United States. We then watched in real time as the results came in, showing off the power of the Blackstone event gateways and new reporting capabilities.



Attendees discuss MAX highlights


They took us through the code for the application to show us just how quickly and easily an application like this is to create. We were also able to see the SMS test server and cellphone emulator in action. Evangelized I was, so much so, that I quickly added the “Mobile SMS Applications Made Easy” session to my schedule.

The Future of Macromedia

In Wednesday’s keynote, chairman and CEO, Rob Burgess, gave us a glimpse of the future of Macromedia. He took us on a journey from the past to the present and then to the future and turned our heads in a new direction and implored us all to follow...non-PC devices. Macromedia has built relationships with over 100 device manufacturers installing Flash in gaming systems and phones. In Japan there is currently a thriving system of developers for Flash-based phones; he urged us to follow suit, stressing that what we see happening in Japan is a glimpse into our own future, and that the time to get the information we will need is now.

Juha Christensen, the new president of Mobile and Devices gave a look at the future of media and entertainment on cell-phones and the possibilities for enterprise applications for mobile technology. This year at MAX, you could spend three and one-half days attending mobile sessions alone. He directed us to a new Web site for getting started in mobile development <http://mobile.macromedia.com>. We were also urged to attend the GSM World Conference in Cannes, February 2005, citing it as the “most important mobile show”.

Sitting in the auditorium, I felt as though I was being given the inside scoop from the very people who knew best. They were pointing me in the direction of the future and imploring me to forge ahead. And their message could not have come any sooner. I don’t know about the rest of you, but in the last couple of years I have found myself growing a little weary of developing the same type of Web-based business applications. I’ve been lucky enough in my own career to stumble upon some utterly amazing technologies that whet the pallet for awhile, but ultimately left me hungry. I certainly came to MAX hungry and somewhat bored, but I left with a vision. Macromedia just may have convinced me to follow them into this new frontier and they have certainly provided me with the tools that I will need on my way.

So, until next year, I look forward to sharing with the rest of you what all of this means to us, the developer community. I can’t wait to see if the new products and features do in fact deliver. I will see you at the CFUGs, MMUGS, user conferences, list servers, blogs, and of course, MAX 2005. 

About the Author

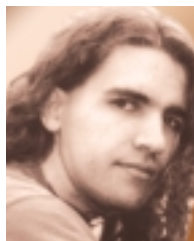
April Fleming has been developing ColdFusion applications since 1997. The majority of her work has been designing and programming Web-based applications for government and educational institutions such as the Department of Defense, NASA, and the University of Central Florida with a focus on distributed data technology. She has been an active member of the ColdFusion community for the past seven years and is currently employed by the Dixon Ticonderoga Company.

AFleming@dixonusa.com

ColdFusion Components

A powerful tool for CF developers

I'm writing this article a few weeks before MAX takes place. I'll be one of the unfortunate few unable to join everyone for the festivities. I am keeping a close eye on the Blackstone development though, and can't wait to start writing about it in this column.



By Jeffrey Houser

As a side note, I *will* be at the Powered by Detroit conference next April (mentioned in Simon's **Community Column** in last month's issue of **CFDJ**); more details on that to come (www.poweredbydetroit.org/).

This month, we're going to talk about ColdFusion Components, or CFCs as they are more commonly known. CFCs were one of the most significant features put into the ColdFusion MX release. If you haven't read last month's article on user defined functions, and are unfamiliar with them, you may want to read that before going further. User defined functions allow you to take snippets of functionality and put them into their own little package. CFCs take this concept one step further and allow you to combine data and functionality into its own little package.

The Elements of a ColdFusion Component

If you've done any development with ColdFusion, it is likely that you already know what a ".cfm" template is. You put ColdFusion code in a .cfm template and the Web server knows to send requests for that file to the ColdFusion server for processing. ColdFusion components are contained in their own file, with the extension ".cfc". The extension is how ColdFusion distinguishes the difference between ColdFusion templates (.cfm) and ColdFusion Components (.cfc).

There is more to a component than just creating a file with the extension .cfc. There are some tags involved, too. Everything in a component is wrapped in the cfcomponent tag. These are its attributes:

- **Extends:** When you're dealing with inheritance, the "extends" attribute specifies the name of the component that this component will inherit its properties and methods from. This is an optional attribute.

- **Output:** The output attribute specifies whether the component body is allowed to generate output. If you set this value to yes, then everything inside the component, but not in a method, is processed as if it were inside a cfoutput tag of a standard template (I'll tell you shortly about component code that's not in a method). Methods will inherit this property, but it can be overridden at the function level. If you set this to no, then the component code outside of methods is processed as if inside a cfsilent tag. That is to say, all output is suppressed

(more information about cfsilent is at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-pc7.htm#wp1103549>). If you omit this attribute, then the code is processed as standard CFML. It's rare and not recommended to output data from inside of a component, so I will set this attribute to no in the examples in this article.

- **DisplayName:** ColdFusion Components are self-documenting. Attempting to browse directly to a CFC is intercepted by a component browser utility that displays the component documentation on the screen. The process of generating documentation in this manner is called introspection and the data displayed is called metadata. When you view the metadata of a component, the DisplayName attribute is listed next to the component's name. This is an optional attribute. It has no effect on functionality.
- **Hint:** The hint attribute is another attribute that helps the CFC self-documenting feature. The text specified in the hint is displayed as part of the documentation. It's optional. It has no effect on functionality.

The interesting thing about the cfcomponent tag is that it is not necessarily needed in a cfc file. A cfc file is inherently a component, so we don't need to define it as such. In reality, the cfcomponent tag is almost always used, due to its attributes such as output and extends. If you really wanted to, you could use the cfcomponent tag in a cfm file to create a component.

Inside the cfcomponent tag you can define your component methods using the cffunction tag. Each component can have as many methods as you need. More information on cffunction can be found at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-p43.htm#wp2852457>. We also talked about functions in last month's column. These are the attributes to cffunction that are especially useful inside of CFCs:

- **Roles:** The roles attribute is used to specify the roles that are allowed to access this function. This refers to the roles that are set when a user logs in using the ColdFusion `cfloginuser` tag. You can use this to control the users who can access individual component methods.
- **Access:** The access attribute specifies the type of access a user must have to execute the function. The valid values for this are Private, Package, Public, and Remote. The default is Public, which means it can be executed by any local file. Private means only code inside the component can execute this function. Package means that only code inside the component package can execute it. A Package is all of the components in the same directory and subdirectories. Remote means it can be invoked remotely via Web services or Flash Remoting or (though rarely done) via URL GET or FORM POST operations.
- **Output:** The output attribute on `cffunction` acts just like the output attribute on `cfcomponent`. If set to yes, then the body of the function will

generate output. If set to no, then the body of the function will be treated as if it were inside a `cfsilent` block. In this article I'll set this to no, unless I'm trying to debug a function or if I'm creating a function whose sole purpose is to output data.

- **Displayname:** Displayname is an optional attribute used only for display in the metadata of a component. Similar to the attribute of the `cfcomponent` tag, this attribute displays its value in parenthesis after the name of the function when viewing metadata and has no effect on performance.
- **Hint:** The hint attribute is used to display comments or additional documentation in the component metadata. It is displayed under the function name as part of the metadata. It has no effect on functionality.

Other `cffunction` attributes, such as name and `returntype` still apply to functions inside a `cfcomponent`. The functions inside a component are often called methods, and the two terms can be used interchangeably: most developers will understand what you're talking about.

You may say: "I see that you mentioned something about code inside a component, but not inside a function. What can you tell me about that?" I'm glad you reminded me. The code in between `cfcomponent` tags, but not inside a `cffunction` block, is executed when you first create the component instance, but not when you make any subsequent calls on an existing instance. This is sometimes called "constructor" code. A constructor is an object-oriented programming (OOP) term that refers to a method that is executed when you create an object. A constructor is officially a method in OOP terminology, and since the CFC form of initialization is not a method, it is often called a pseudoconstructor.

Variables Inside a Component

There are two primary elements in a component: data and functionality. In the previous section we explored how to write the functionality. Here, you'll learn about the data. The data is stored in variables. You probably know all about variables in standard `cfm` templates. You can create them with `cfset` tags and they reside in different variable scopes,

Bridging the Collaboration Gap.

Collaboration

Combine your legacy collaboration applications into one easy-to-use, powerful interface.

FuseTalk for Microsoft .NET.

1-866-477-7542

Discussion Forums

www.fusetalk.com



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

depending on the intended purpose of the variable. You can read a lot more about this in the first edition of this column (February '04).

Since you know all about variables, I'm going to jump right over the basics and go right into the scopes that exist inside a component:

- **This:** This is a reference to everything public inside the component and can be thought of as the component's public scope. If you create a variable in the "this" scope, the variable can be accessed by anything that operates on the instance of the component. I don't usually use the "this" scope for variables for reasons you'll soon understand.
- **Variables:** You'll recognize the variables scope from your work with cfm templates. In a component, the variables scope is used to hold private data. This data is sometimes called instance data.
- **Arguments:** The arguments scope exists only inside a method, during the execution of a function. It contains all the arguments passed into the function (very similar to the "attributes" scope when creating custom tags).
- **Local function scope:** The local function scope is an unnamed scope. All variables that you create with the "var" keyword using CFSET or CFSCRIPT exist in this scope.
- **Shared scopes:** Inside a cfcomponent, you can access the shared scopes, such as application and session. Since the component is part of a request, request-based scopes such as URL, form, and request, are available to the component. Accessing these scopes is generally considered bad practice because it makes the code less reusable.

When developing components, I will use the "constructor" code to initialize variables in the components variables scope. I will create a number of methods to retrieve the private values (called getter methods because they only return a value) and methods to set the value of the private variables (called setter methods because they set a value).

Calling Components

There are a handful of different methods you can use to call components. I'm going to explain the one that I use most often. It's a two-step process, but is much more fun than country dancing. The first step is to create the component instance. The second step is to call various methods against the component. Component data will be persisted between component requests. Here is a sample component, entitled foo.cfc:

```
<cfcomponent>

<cfset variables.bar = " corner">

<cffunction name="Getbar" access="public" returntype="string">
  <cfreturn variables.bar>
</cffunction>

<cffunction name="Setbar" access="public" returntype="boolean">
  <cfargument name="bar" type="string" required="Yes">
  <cfset variables.bar = arguments.bar>
  <cfreturn true>
```

```
</cffunction>
```

```
</cfcomponent>
```

The foo component is very simple. It has one instance value, named bar. The bar variable is initialized to a string value "chunky".

To use the CFC you first need to create the instance of the component. You can use the cfoject tag or the createObject function to create the instance. Conceptually, both are similar. You can read about the createObject function at

<http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/uncti45.htm#wp4569859>. I'll explain the cfoject tag in more detail. It has two attributes in relation to components:

- **Component:** The component attribute specifies the location of the component, including the name of the file without the cfc. If they are in the same directory you can just use the component name.
- **Name:** The name of the return variable that will contain the component instance.

More information about using cfoject to invoke components is located at <http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/tags-pb7.htm#wp2651979>. This is a snippet of code to invoke our foo component:

```
<cfoject component="Foo" name="objFoo">
```

To invoke a method on the component, we use the cfinvoke tag. These are its attributes:

- **Component:** The component attribute is used to specify the name of component instance. To reference an instance that is already created, you must put it in pound signs. If you are calling a single method of a CFC without creating an instance, you can enter the location of the CFC.
- **Method:** The method attribute is used to specify the name of the method you want to execute.
- **ReturnVariable:** The ReturnVariable attribute specifies the name of the variable that is populated with the return value.
- **ArgumentCollection:** The argument collection attribute is used to specify the name of a structure that contains all arguments that the function requires. It is optional. You can also list arguments using name value pairs that correspond to the arguments in the function, or by using cfinvokeargument tags nested inside the cfinvoke block.

After creating our instance, we can use the cfinvoke tag to check the value of the bar instance variable:

```
<cfinvoke component="#objFoo#" method="getBar" returnvariable="variables.Out"/>
<cfoutput>
  #variables.Out#
</cfoutput>
```

This code calls the GetBar method, putting the return variable in the out value and then outputting the value. You'll see that the Bar instance variable has not changed

since we set it. We can change that using this line of code:

```
<cfinvoke component="#objFoo#" method="SetBar"
bar="Main"/>
```

Rerun the previous segment after running the Set method and you'll see that the value has correctly changed.

Are CFCs Object-Oriented?


"So, Jeff," you say, "when I create a CFC it sounds like I'm creating an object. Isn't that object-oriented programming? I've heard that the use of CFCs is object-oriented. Isn't that true?" Again, I'm glad you asked. The answer is, quite indirectly, "sort of." There is no doubt that CFCs bring many object-oriented (OO) features into the CFML language. However, just because you are using CFCs does not mean you are programming in an OO way.

When we talk about object-oriented programming or traditional (procedural) programming, we're really talking about the approach taken to design an applica-

tion. In ideal cases, this application design occurs before you sit down to write code. In CF5 and before, it was not as easy to implement an object-oriented design using ColdFusion. There is a group of custom tags called cfObjects that were used to apply OO design to ColdFusion. More info is available at <http://cfobjects.sourceforge.net/>. ColdFusion Components make the implementation of object-oriented designs much easier.

"So, then," you ask, "CFCs are object oriented, right?" They can be, yes. But, even if you are not using an object-oriented design you can still make use of CFCs. Instead of creating classes for an OO design, you can use CFCs to create Abstract Data Types (ADTs). In simple terms, an ADT is a user-defined data type. Just like classes in OO design, Abstract Data Types provide a way to encapsulate data and the functionality to operate on that data into one component. ADTs generally do not make use of OO specific concepts, such as inheritance or polymorphism.

Conclusion

ColdFusion Components are a powerful tool in the toolbox of any ColdFusion developer. They provide advanced encapsulation, so that you can more easily write reusable components and speed the development of future applications. They also open the door to the use of Web services and Flash Remoting. After reading this article, you should understand all the basics of creating CFCs. Next month, I'm going to show you how I like to use CFCs and I'll step you through the process of creating a reusable component. 

About the Author

Jeffrey Houser has been working with computers for over 20 years and in Web development for over 8 years. He owns a consulting company, and has authored three separate books on CF, most recently ColdFusion MX: The Complete Reference (McGraw-Hill Osborne Media).

jeff@instantcoldfusion.com

CFDJ Advertiser Index


ADVERTISER	URL	PHONE	PAGE
ACTIVEPDF	WWW.ACTIVEPDF.COM		13
CFDYNAMICS	WWW.CFDYNAMICS.COM	866-233-9626	4
CFDJ	WWW.SYS-CON.COM/COLDFUSION/	888-303-5282	45
EV1 Servers	WWW.EV1SERVERS.NET	800-504-SURF	6
FUSETALK	WWW.FUSETALK.COM	866-477-7542	39
HAL HELMS, INC	WWW.HALHELMS.COM		29
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM/BLACKSTONE	877-215-4678	31,43
ISSJ	WWW.ISSJOURNAL.COM	888-303-5282	49
INTERAKT ONLINE	WWW.INTERAKTONLINE.COM		11
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800-379-7729	COVER IV
MACROMEDIA	WWW.MACROMEDIA.COM/GO/VOLVO		2-3
MACROMEDIA WEB PUBLISHING SYSTEM	WWW.WEBPUBLISHINGSYSTEM.COM		51
MX DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM	888-303-5282	49
PAPERTHIN	WWW.PAPERTHIN.COM	800-940-3087	17
SEAPINE SOFTWARE	WWW.SEAPINE.COM	888-683-6456	23
SERVICESIDE	WWW.SERVICESIDE.NET	888-682-2544	19
WEB SERVICES EDGE EAST	WWW.SYS-CON.COM/EDGE	201-802-3066	43
WEBCORE TECH	WWW.WEBCORETECH.COM	877-WCT-HOST	33

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

MAX Impact

— continued from page 5

I have other ideas, but these are the ones I'm concentrating on at the moment.

So there you have it – a glimpse of what is and what may be in store for you in future issues of *ColdFusion Developer's Journal*. I for one can't wait to get more Macromedians involved, nor can I wait to get the magazine on focus each month and introduce some more new columns. As always, please let me know what you think and submit any feedback or ideas you might have to simon@horwith.com. 

2004 RCAs

Readers' Choice Awards

Nominations Open

"The Oscars of the Software Industry"



Nominate the Best of the Best:
Tell us what tools, solutions, services,
and books deserve recognition.

Cast your vote at
www.SYS-CON.com

Thinking Outside the Table PART 1

Bulk inserts

A two-part series looks at techniques for shifting workload away from the application server onto the database by using “extra” database tables.

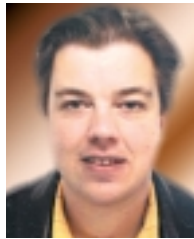
Most ColdFusion programmers understand that when it comes to bulk inserting into a database, it isn't good practice to loop over text files one line at a time with a `<cfloop>`. However, when faced with the realities of a data feed that needs preprocessing, comparing to existing data, and then postprocessing, good intentions sometimes fall by the wayside.

The theory is simple – if you have a large amount of data (typically tabbed or comma separated text) to import into your database, the best way to insert it is in a single import – either directly from SQL (with a bulk insert command for SQL Server), or else via an import utility (such as DTS for SQL Server).

In practice, it's a little more difficult than that, and far too often import routines are constructed in ColdFusion that read the complete text file, loop over it line by line, validate the data and format, and then either update an existing record or insert it into the database.

This technique is inefficient, slow, and prone to “falling over,” but it's a technique that is used time and again, usually because of one of three types of problems:

1. **The imported data is incorrectly formatted and needs preprocessing:** For instance, you may be importing a feed where the country names in addresses are free text and not ISO codes, or values such as “England” or “Hawaii.” Often these errors are consistent and can be fixed with ColdFusion but can't be imported directly into a table with a foreign key constraint requiring strict two-character country codes.
2. **Existing records need updating and new records need adding (the data needs to be “matched and added”):** Matching and adding often take place where data is supplied from an external system (e.g., user data). Existing users may have passwords or other data unique to the Web database that you don't want to overwrite. You could fudge this with one-to-one joins to separate tables (assuming you have a unique key to do it), but the neater way is to update existing data and add new data. A third stage in the process is often a “delete” – the removal of data no longer present in the feed. Ideally this stage should really be a “change status to expired,” but “match, add, and delete” is easier on the tongue.



By Tom Peer

With a line-by-line import, the usual way of determining whether a record is already in the system is to query the database – meaning that for every line in the import you end up with two queries. It can be the most inefficient part of the whole process.

3. **Business rules need to be applied to the data (postprocessing):** The final issue is the need to apply business rules after an import. Typically requiring large doses of flexible ColdFusion code, business rules are difficult to apply to data imported in a batch.

Examples of business rules that I've come across that were cited as preventing batch import include allocating free magazine subscriptions to new subscribers of a different magazine, and e-mailing all subscribers to a magazine telling them they had free access to a Web site. Both of these were far easier to achieve in ColdFusion than pure SQL, and still are, even though the import process itself now runs as a batch.

The Extra Table

All these issues can be dealt with without resorting to line-by-line imports by using a very simple technique – the addition of a separate import table to hold the data while it's processed.

Instead of importing data directly into its intended destination table, it's imported into a dedicated import table that's a copy of the destination table without any constraints, keys, triggers, or anything else that might prevent a bulk import.

Once the data has been imported, it's then processed to ensure it conforms to all the constraints on the destination table. The data can either be fixed if there are consistent structural problems with the feed, or else removed from the import process (depending on the situation you might simply delete records or, at the other end of the scale you might have a third table to store incorrect records for offline fixing).

After the preprocessing, the process of updating the live data can begin – the process of “match, add, and delete”. The crucial difference between doing this with an import table and with line-by-line is that with an import table you can use a single SQL command to match *all* the existing records, a single command to add all the new records, and a third single command to delete (or change the status of) any records no longer in the feed.

Importing the Data

There are various techniques you can use for importing the data, and this article doesn't cover them. BULK INSERT in SQL

Server works well with tabbed text. Running DTS via SQL commands was covered in **CFDJ** in September and October 2003, and is needed if you have field qualifiers in your feed (e.g., “”s around the fields). Whatever technique you choose, ensure that it imports the complete data set into an import table for preprocessing.

Checking for Errors

The next stage in the process is to check for data integrity. This might mean applying rules such as requiring e-mail addresses to be in the format %@%.%, or it might consist of “normalizing” values such as country names or categories. For instance, data may be imported with a category of “News,” which needs to be translated to a numeric key value from a related table.

All this can be done with either SQL or ColdFusion. Either way, the key is to write SQL that acts on sets of data rather than one line at a time. What you do with records that contain errors depends on the needs of your application. You can delete the records or copy them to another text file. Whether you’re attempting the process in SQL server or ColdFusion, the

ideal method is to have some way of marking them in the database.

Either use a status field in the import table or else (as I prefer to do) have a separate error table: a copy of the import table with extra fields to indicate the type of error and the date it occurred.

The exact details of this stage will depend on your requirements, but the principle is always the same – after this process, all data in the import table is ready for inserting into or updating the main table.

Matching

The match part of the process is the easiest and roughly the same for all RDMSs. Simply update all necessary fields where the existing data matches the import data:

```
UPDATE table1
SET   field1 = import_table.field1,
      ... fieldn = import_table.fieldn
FROM   table1, import_table
WHERE  table1.keyfield =
import_table.keyfield
```

The key field doesn’t necessarily have

to be the primary key of the main table – it just has to be a unique set of data that can be used to identify the existing data. If you’re importing user data, this is often the e-mail address. There should be a unique constraint on this field in the main table if it isn’t the primary key (which is unlikely).

Adding New Records

Adding new data is not so simple. By far the easiest method (and best if you’re using MySQL) is simply to delete all the matched records from the previous step and then insert any remaining records. The alternative is to use some advanced SQL to select the records that aren’t already in the destination field.

If at all possible, avoid using a sub-query such as:

```
INSERT INTO table1 (field1,field2)
table1 (field1,field2)
SELECT      I.field1,I.field2
FROM        import_table I
WHERE
import_table.keyfield
NOT IN      (SELECT keyfield
FROM table1)
```

Be the 1st to develop with
Blackstone

HostMySite.com will be the first host to offer
the innovative new version of ColdFusion.

Be the first, visit hostmysite.com/blackstone

 **HostMySite.com**

1-877-215-4678

Once you're in it...



...reprint it!

- ColdFusion Developer's Journal
- Java Developer's Journal
- Web Services Journal
- Wireless Business & Technology
- MX Developer's Journal
- PowerBuilder Developer's Journal
- .NET Developer's Journal
- LinuxWorld Magazine
- WebSphere Journal
- WLDJ

Contact Kristin Kuhnle
201 802-3026
kristin@sys-con.com

REprints



techniques

Although this is easy to follow, it's very inefficient and isn't suitable for large data sets. The most efficient solution is to use a join, for example:

```
INSERT INTO table1 (field1,field2)
SELECT I.field1,I.field2
FROM import_table I
LEFT OUTER JOIN table1
ON I.keyfield = table1.keyfield
WHERE table1.keyfield IS NULL
```

The SELECT part of this statement returns all rows from the import table where there is no corresponding record in the destination table. This can be a little confusing, and if your RDMS supports VIEWS, it can sometimes be easier to do the outer join in the view and then query that.

In the following SQL, the keyfield from the destination table has been given an alias of "Existing_keyfield" to differentiate it from the keyfield in the import table. Existing_keyfield would then be NULL for new records.

```
INSERT INTO tablename
(field1,...fieldn)
SELECT field1,...fieldn
FROM import_view
WHERE import_view.existing_key-
field IS NULL
```

This View can also be useful if you need to do any lookups while importing (i.e., to get ID numbers instead of name values from related tables).

If your RDMS doesn't support these techniques, you can simply delete all the matched records from the previous step before doing an insert.

Deleting Expired Records

The final stage of a complete import is the "delete." Of course as all good programmers know, it's best to never actually delete anything, instead you simply change its status. Using the reverse of the outer join used for the import, you can see which records in your system are no longer in the feed and update them accordingly. In SQL server, it's easy:


```
UPDATE table1
/* your logic here to mark as expired
or deleted */
SET table1.status_id = -1
FROM import_table I
```

```
RIGHT OUTER JOIN table1
ON I.keyfield = table1.keyfield
WHERE I.keyfield IS NULL
```

If you don't like that syntax, you can again set up a view, or if you have small data sets you could use a subselect to get the keys of the records to be deleted. If you're using MySQL, you may find it easier to use ColdFusion to simulate a subselect by running one query to get the keys of the records to delete, and by using the ValueList() function to create an IN () clause.

Advanced SQL

No matter how complicated the import you need to do, there's usually a simple way to do it using an extra table of some description. One of the more complicated I've worked on involved users being imported into a database, and for every new user, a user record needed to be added and then another record entered into a table of subscriptions using the primary key generated by the insert. I did this using a SQL Server temporary table - populating it with the unique import key of any new records and then joining it to the users' table when it came to inserting the subscriptions. Another way would have been to create a UUID field in the import table and the users' table and join that.

Many developers have similar tales of long, slow, line-by-line procedures that they've eliminated using the simple concept of an extra table - either physical or temporary. In the second part of this series I'll look at another technique that also uses an extra table - reducing server load by storing search results in the database itself. 

About the Author

Tom Peer is a Web developer specializing in online journals and magazines (www.tompeer@digitalmethod.co.uk). He was formerly manager of the New Scientist online, one of the top science Web sites, and now runs an agency providing design, development, and hosting services to a number of other UK publishers.

tompeer@digitalmethod.co.uk

Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE



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

That's a savings of \$29.99 off the annual newsstand rate. Visit our site at www.sys-con.com/coldfusion or call 1-800-303-5282 and subscribe today!

ColdFusion Developer's
Journal



Saving Polly

Making polymorphism more understandable



My heart beat faster as the mail carrier handed me the mail. There it was: my latest edition of *Thrilling Tech Tales* magazine. As I always do, I immediately turned to the exact center of the magazine to read the latest installment of “The Adventures of the Morphic Family.” This week, it was about Polly, the daughter and heir to the family farm. I eagerly began reading...

“And now, my dear, I fear that it’s time for me.toGo(),” said Black Bart. “I suppose I can’t talk you into coming with me? No? Well, yourSelf.suit(),” he said, rising from his crouching position. From where she lay securely tied to the railroad tracks, Polly could only see Bart in silhouette against the bright sky.

“You’re a monster!” Polly told him. “me.rather(die) than go along with your nefarious plans!” As she spoke, she could hear the whistle from the approaching train.

“Very shortly, Polly, my dear, the train.to(ElPaso) will cross this stretch of tracks and you will certainly get your wish. I, on the other hand, will get the deed to your farm.”

“No matter what you do to me, myFather.sell(theLand) will never happen!” Polly exclaimed.



By Hal Helms

“Do you know I suspect me.havePowers(psychic)?” asked Bart.

“Take your father, for example: me.haveHunch() that he may be joining you shortly in the afterlife. I, of course, will have a signed bill of sale for the farm.”

Polly lay quiet wrestling with her thoughts. Was there nothing she could do? Could this really be the end of it? Would blackBart.prevail()? There was nothing but for her to wait(theEnd). She determined to do that bravely.

Polly’s thoughts were interrupted by a movement in the sky. Above her, a tiny dark dot appeared against the clear blue. Was it a bird? Polly watched until it became clear that, whatever the object was, it was certainly not a bird and it was certainly headed toward her. For several moments, Polly watched as the speck grew larger. Then she spoke.

“Do you know, Black Bart, that me.havePowers(psychic) also? Take you, for example: me.haveHunch() that you may soon be a long-term guest of the sheriff.”

“Eh? What’s that? What are you.talkingAb —?” Bart broke off, his eye now catching the figure flying toward them. The person was too far away to recognize, but it clearly was a person and clearly was headed toward them. “So that’s what you.pinHopes(superHero) on?” asked Bart. “But aren’t you.gettingAheadOfYourself()? That could be any superhero and without knowing which one, how will you know how to ask him for help? If it’s Superman, you could ask him to burnRopesWithXrayVision(). But if it’s Batman, you’d have to ask him to cutRopesWithKnife(). And if it’s — well, I think you

see you.haveProblem(undefinedSuperhero)."

Unhappily, Polly realized that, for once, Bart was telling the truth. Without knowing exactly what type of superhero was speeding to her aid, she wouldn't be able to know what type of aid to ask for. And, she feared, by the time she could identify the superhero, it would be too late. Too late for her. Too late for her father and the Morphic family farm. Too late for... Suddenly, Polly brightened. Bart.stared(polly, astonished) as she began to laugh. "No, Bart. you.haveProblem(proceduralMindset)." And with that, Polly cried aloud: "superhero.save(me)!"

[Is Polly's fate sealed? Or does she know something evil Bart does not? Will the undefined superhero save Polly? Find out in our next action-packed episode.]

No! How could I wait another month to learn Polly's fate. Would she keep the farm or would she buy it instead? I felt sure the writer wouldn't be so cruel as to leave no clues to Polly's fate in his masterpiece, so I began looking deeper. Perhaps there was something in the characters' names that might enlighten me, but try as I might, I couldn't make anything out of Bart or Black Bart or even Evil Black Bart. The closest I came was the idea that Polly might "bart-er" for her life. Polly's name, of course, was no help.

Discouraged, I gave up and went back to work on my "Object Oriented Programming with CFCs" course. After each class I do, I try to incorporate new ideas and suggestions into the material. This is particularly true when working with tough concepts. I looked at my current slides (see Figure 1).

In my experience of teaching object oriented programming, most ColdFusion programmers have a solid grasp of the procedural method of building software, creating databases to store information and writing algorithms to operate on that data. But the OO way – combining data and algorithms together into an object – is harder to grasp (see Figure 2).

While it's relatively easy to understand the concept of an object, it's much harder to see how profoundly this changes the way software is written. An object oriented application resembles a conversation between objects more than it does a list of instructions to be carried out.

Objects "hold conversations" by sending messages to each other – things like myCar.start() and myCar.refuel(myCar.getTankCapacity()). Now, granted these are pretty "geeky" conversations, but I find that thinking of objects as alive and having a mind of their own, so to speak, is an important step in transforming procedural programmers into OO developers (see Figure 3).

In order to ensure that the conversation doesn't turn into babble, the designer must guarantee that only "safe" messages are sent to objects. A safe message is one that the object can respond to. We would expect an object named myCar to be able to respond to messages such as start, stop, refuel, and breakDown, but not to messages such as sing, dance, compose, and orchestrate.

How does an OO language help the OO designer ensure that no such babble ensues? It redefines the concept of data type (see Figure 4).

In procedural programming, data types are things like integers, Booleans, and arrays. Those are still data types in OO languages, but a great deal is added to the definition. In OO lan-

guages, a data type specifies what kind of information can be associated with a variable and what operations can be performed on that variable.

Procedural programmers get the first part of this; it's the second part they sometimes struggle with. But what we mean by "what operations can be performed on that variable" is just this: what messages can be sent to that object? The definition of the type determines what messages can be safely sent. And you, as an OO programmer, get to define your own data types, including what kind of information will be associated with your data type and the messages that can be sent.

ColdFusion components (CFCs) give us a way to write OO applications. Each component is a user-defined data type. Look at this code defining an Invoice data type:

```
<cfcomponent displayname="Invoice">
    <cfset variables.lineItems = ArrayNew(1) />
    <cfset variables.customer = "" />

    <cffunction name="getTotal" access="public" returntype="numeric"
output="false">
        <cfset var total = 0 />
        <cfloop from="1" to="#ArrayLen(variables.lineItems)#"
index="i">
            <cfset total = total +
variables.lineItems[i].getPrice() />
        </cfloop>
        <cfreturn total />
    </cffunction>

    <!-- other methods omitted -->
</cfcomponent>
```

It's easy to create a CFC:

```
<cfset currentInvoice = CreateObject('component', 'Invoice') />
```

If you send it a legal message ("legal" being defined as the cffunctions specified in the component), all goes well:

```
<cfoutput>
    Your order today comes to #DollarFormat(currentInvoice.getTotal())#
</cfoutput>
```

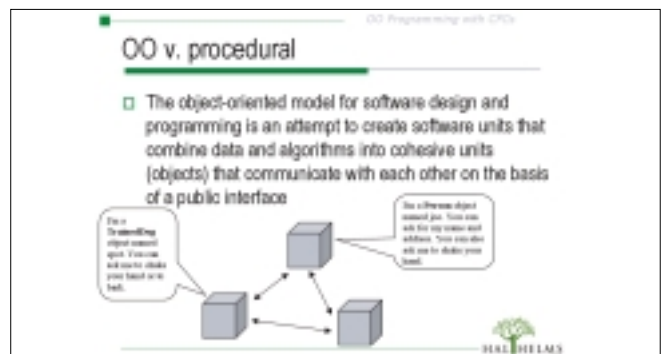


Figure 1: OO vs procedural

But an illegal message will cause ColdFusion to chide you since there's no such cffunction as *discount* defined in the Invoice component:

```
<cfoutput>
    #currentInvoice.discount(.20)#
</cfoutput>
```

So, data types are key to OO design, and part of learning OO is understanding how to define types in such a way that flexibility (think "maintenance") is maximized while still retaining robustness.

Let's take, for example, the case where we wish to model an orchestra. Of course, our orchestra members must be paid so we'll create a PayrollClerk component. Now our orchestra is a bit different from the ones found in the real world. Here, orchestra members determine their own gross pay. String players (violins, cellos, etc.) have a complicated formula based on the number of notes played, brass players compute their gross based on how loud the notes must be played, while percussion players calculate their gross pay on how many times they must strike their instrument. It's all very complicated.

If we were approaching this procedurally, we would have an unpleasant task, writing code filled with conditional code to account for the various ways pay should be calculated.

Procedural developers learning OO often take procedural ideas and try to force-fit them into objects. Such a developer might create a PayrollClerk component resembling this:

```
<cfcomponent displayname="PayrollClerk">

    <cffunction name="payStringMember" access="public"
    returntype="numeric" output="false">
        <cfargument name="stringMember" type="StringMember"
        required="true" />
        <!-- algorithm for determining pay -->
    </cffunction>

    <cffunction name="payBrassMember" access="public"
    returntype="numeric" output="false">
        <cfargument name="brassMember" type="BrassMember"
        required="true" />
        <!-- algorithm for determining pay -->
    </cffunction>

    <cffunction name="payPercussionMember" access="public" return-
    type="numeric" output="false">
        <cfargument name="percussionMember" type="PercussionMember"
        required="true" />
        <!-- algorithm for determining pay -->
    </cffunction>

</cfcomponent>
```

Where should the algorithms for determining pay be located? At first, we might think they belong in PayrollClerk, but this will get us into trouble. When later, requirements change – they always change! – and we add a winds section to our orchestra (clarinets, flutes, etc.), we must not only create a new component for WindMember, but must change PayrollClerk, adding a payWindMember method. And it's likely that it's not only PayrollClerk who will interact with different orchestra members based on their type, meaning that each time we add something to the application, we could find ourselves searching through various existing components seeing what changes might need to be made. If you're presently writing procedural code, this may sound somewhat familiar...

The OO approach is different and it makes use of data types in a different way. First, we start with the components for StringMember, BrassMember, and PercussionMember. All these components are specialized types (as in data types) of a broader class that we might call OrchestraMember. And it's in OrchestraMember that we will define that all OrchestraMembers can understand the message, calculateGrossPay(), and return a numeric value when this message is sent.

In OO languages, we can model such specialized-generalized relationships through the mechanism of subtyping. A subtype is just a specialized version of a more general type, making it ideal for modeling the relationship between, say, StringMember and OrchestraMember.

Here's the code for OrchestraMember:

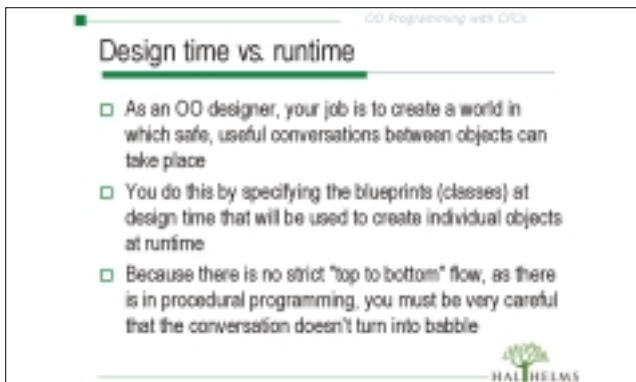


Figure 2: Design time vs runtime

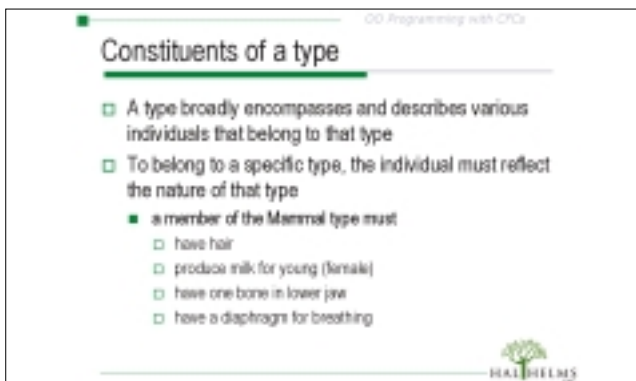


Figure 3: Constituents of a type



web services **EDGE**
conference & expo

Web Services Edge
2005 East

International Web Services Conference & Expo

Hynes Convention Center, Boston, MA
February 15-17, 2005

The Largest
i-Technology
Event of
the Year!



**Guaranteed
Minimum
Attendance
3,000
Delegates**

Tuesday, 2/15:
Conference & Expo

Wednesday, 2/16:
Conference & Expo

Thursday, 2/17:
Conference & Expo



Join us in delivering the latest, freshest, and most proven Web services solutions... at the Fifth Annual Web Services Edge 2005 East - International Conference & Expo as we bring together IT professionals, developers, policy makers, industry leaders and academics to share information and exchange ideas on technology trends and best practices in secure Web services and related topics including:

- Transitioning Successfully to SOA
- Federated Web Services
- ebXML
- Orchestration
- Discovery
- The Business Case for SOA
- Interop & Standards
- Web Services Management
- Messaging Buses and SOA
- SOBAs (Service-Oriented Business Apps)
- Enterprise Service Buses
- Delivering ROI with SOA
- Java Web Services
- XML Web Services
- Security
- Professional Open Source
- Systems Integration
- Sarbanes-Oxley
- Grid Computing
- Business Process Management
- Web Services Choreography

3-Day Conference & Education Program features:

- Daily keynotes from companies building successful and secure Web services
- Daily keynote panels from each technology track
- Over 60 sessions and seminars to choose from
- Daily training programs that will cover Web Service Security, J2EE, and ASP.NET
- FREE full-day tutorials on .NET, J2EE, MX, and WebSphere
- Opening night reception

Interested in Exhibiting, Sponsoring or Partnering?

Becoming a Web Services Edge Exhibitor, Sponsor or Partner offers you a unique opportunity to present your organization's message to a targeted audience of Web services professionals. Make your plans now to reach the most qualified software developers, engineers, system architects, analysts, consultants, group leaders, and C-level management responsible for Web services, initiatives, deployment, development and management at the regions best-known IT business address - The Hynes Convention Center in Boston.

For exhibit and sponsorship information please contact Jim Hanchrow at 201.802.3066, or e-mail at jimh@sys-con.com.

Sponsored by:



All brand and product names mentioned above are trade names, service marks or trademarks of their respective companies.

Contact for Conference Information: Jim Hanchrow, 201-802-3066, jimh@sys-con.com



www.sys-con.com/edge

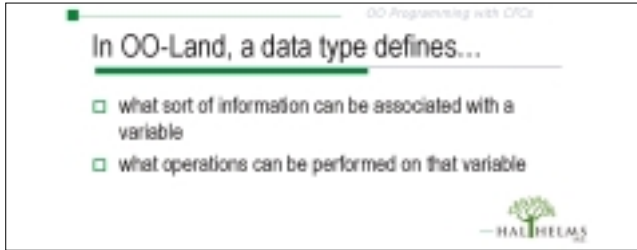


Figure 4: In OO, what a data type defines

```
<cfcomponent displayName="OrchestraMember">
  <cffunction name="calculateGrossPay" access="public"
    returnType="numeric" output="false">
    If you're seeing this, then some subtype of mine isn't doing
    its job!
  </cffunction>
</cfcomponent>
```

To show that BrassMember is a subtype of OrchestraMember, we use the extends attribute of the cfcomponent tag:

```
<cfcomponent displayName="BrassMember" extends="OrchestraMember">
  <cffunction name="calculateGrossPay" access="public"
    returnType="numeric" output="false">
    <!-- algorithm for determining gross pay for BrassMember -->
  </cffunction>
</cfcomponent>
```

Now, the payoff comes: PayrollClerk (and any classes that deal with various orchestra members) is enormously simplified:

```
<cfcomponent displayName="PayrollClerk">
  <cffunction name="payMember" access="package" returnType="numeric"
    output="false">
    <cfargument name="orchestraMember" type="OrchestraMember"
      required="true" />
    <cfset var grossPay =
      arguments.orchestraMember.calculateGrossPay() />
    <!-- algorithm for determining net pay -->
  </cffunction>
</cfcomponent>
```

But how do we create individual orchestra members?

```
<cfset firstViolin = CreateObject('component', 'StringMember') />
<cfset secondTrombone = CreateObject('component', 'BrassMember') />
<cfset timpani = CreateObject('component', 'PercussionMember') />
```

We might decide to put all orchestra members together:

```
<cfset orchestra = ArrayNew(1) />
<cfset ArrayAppend(orchestra, firstViolin) />
<cfset ArrayAppend(orchestra, secondTrombone) />
<cfset ArrayAppend(orchestra, timpani) />
```

Then, when it's time to pay the orchestra members...


```
<cfset payrollClerk = CreateObject('component', 'PayrollClerk') />
<cfloop from="1" to="#ArrayLen(orchestra)#" index="i">
  <cfset payrollClerk.payMember(orchestra[i]) />
</cfloop>
```

We created the component, OrchestraMember, only to provide a generic type that could encompass any particular subtype. We might even have a business rule that says that no OrchestraMember type will ever be created; we'll only create subtypes of OrchestraMember. And you can see from looking at the OrchestraMember component's code that there's no implementation code for payMember. So how, if PayrollClerk accepts a generic OrchestraMember, can individuals of various subtypes be passed and, once passed, how is any code actually run?

This happens because all OO languages have a feature known as "type promotion" that allows them to "promote" a subtype to a more generic type. When we sent firstViolin, for example, to the PayrollClerk's payMember method, ColdFusion type-promoted firstViolin from StringMember to OrchestraMember. As an OrchestraMember, firstViolin fit the argument type specified for payMember (an OrchestraMember).

Type promotion explains how we can send a subtype to a method expecting a more generic type, but we still need an explanation for how the appropriate calculateGrossPay method is called for each object. The mechanism for this is called polymorphism and, after encapsulation, is the most important aspect of object orientation. Polymorphism ensures that even though an object is type-promoted, a message sent to that object will call the method associated with that object's base class. In other words, when firstViolin, whose base class is StringMember, is type-promoted to OrchestraMember (by being passed to a method that accepts only OrchestraMembers), the subsequent call to firstViolin's calculateGrossPay method will execute the code defined in StringMember, not in OrchestraMember.

I was musing over all of this when I was struck by the similarity in names: Polly's last name is Morphic, making her full name Polly Morphic and here I was working on making polymorphism more understandable. I suppose life is full of these odd coincidences, though I did fervently wish there was some way to solve Polly's dilemma while helping students "get" polymorphism.

I supposed that would be too much to ask, and I was still bothered by how Polly could be so certain that the particular superhero who came to save her would respond to a save message. Did she really have psychic powers? Oh well, I guess I'll just have to wait for the next issue of *Thrilling Tech Tales*. But if you figured out what happens to Polly, please e-mail me at hal@halhelms.com. I'd sure love to know. 

About the Author

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

hal.helms@teamallaire.com



When was the last time your Intranet was updated?

Announcing the Macromedia® Web Publishing System. You build and manage the site. Users keep content fresh.

With Macromedia® Studio MX 2004, Contribute™ 3 and Contribute Publishing Services, you can affordably build, manage, and publish enterprise sites. And unlike the average content management system, this works. You determine who can edit and who can publish. Then users simply point and click to update any page. In no time, dated content will be a thing of the past. Learn more at www.webpublishingsystem.com

macromedia®
**WEB PUBLISHING
SYSTEM**



be the pilot!

FREE SETUP on Shared
Hosting Accounts With
ColdFusion MX Support
Use Promo Code CFDJ2004



WE DARE YOU TO TAKE A FREE TEST FLIGHT!

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

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

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

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

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

Managed Hosting • Shared Hosting • Microsoft Exchange Hosting

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

