USING XML IN COLDFUSION PAGE 8

ColdFusion Developer's Journal

ColdFusionJournal.com

July 2005 Volume:7 Issue:7

macromedia®

One little box. A whole lot of power.

Put it to work for you.
The shortest distance between you and powerful web applications.

Full speed ahead. The release of Macromedia ColdFusion MX 7 is changing the whole game. This groundbreaking release introduces new features to help address your daily development challenges. These features include:

› Structured business reporting? Check. Printable web content? Check.
ColdFusion MX 7 provides a structured business reporting solution that will have you creating detailed business reports for anyone who needs them. You can also dynamically transform any web content into high-quality, printable documents with ease. The days of needing a third-party application to generate dynamic reports are going, going, gone.

› Make rapid J2EE development a reality.
So, you're heavily invested in J2EE but would love to complete projects in less time? ColdFusion MX 7 is your answer: It delivers RAD productivity on top of the power and scalability of J2EE and deploys onto standard J2EE server environments.

› New mobile applications are a go.
Innovative new features enable ColdFusion MX 7 applications to reach beyond the web. So you can rapidly create new applications, or extend existing ones, to connect with mobile phones and IM clients using a variety of protocols. The new world of mobile communications is exciting, and this your invitation to the party.

To learn more or take ColdFusion MX 7 for a test drive, go to:
macromedia.com/go/cfmx7_demo

Brad Lawryk: "Before MX Kollection, I 'thought' I was doing a **pretty good job**, and now ...

Everyone tells me I´m doing a **brilliant** job!"

If you own Dreamweaver,

MX Kollection is a must!

List Database Information
Create dynamic lists
Order records within a list
Insert, Edit, Delete field in list
Automatic navigation bars
Automatic list filter
Sort column by clicking on the header
Automatic row counter
Delete multiple records directly from list
Create master/detail lists

Manage Recordsets Visually
Create, Edit recordsets visually
Visually add tables to query
Contextual menu to refresh fields
Large query management Zoom In/Out
JOINS between tables
Define and apply complex conditions
Automatically generate SQL conditions
Add GROUP BY for aggregated fields
Extract information from multiple tables
Optimized SQL generation
Smart SQL queries for list filters
Automated database introspection

Rapid HTML Form Creation
Generate Insert Record Form
Generate Update Record Form
Insert, Edit or Delete form field
Redirect to page after form submit
Table and CSS form generation

Form Validation and Error Handling
Validate form fields
Rich formats library
Allow custom validation formats
Error handling
Preserve submitted values on error

Upload Files and Images
File Upload
Image Upload
Resize and sharpen image on upload
Show Dynamic Image
Download Uploaded File
Delete file from specified folder
Show If File Exists on Server

Send E-mails
Send e-mail after form submit

Online HTML Editor
Edit HTML content visually
Use your own CSS styles
Upload and manage server images
Format tables and align images

Enhanced HTML Form Controls
Date Picker
Dependent Drop-downs
Editable Drop-down (Combo-box)
Masked Textfield
Numeric Textfield
Dynamic Search
n...1 Dependent field
Smart Date

Security & User Authentication
Login form generator
Login with User Levels
Remember me feature
Encrypted password
Restrict access to page
Generate password with fixed length
Update user password
Send password by e-mail

DREAMWEAVER PRODUCTIVITY TOOL FOR DYNAMIC WEBSITE DEVELOPMENT

Download a trial version here - http://interaktonline.com/go/MXKollection/

Find out how easy is to use our tools - **save 30%** by using the coupon in the right.

Caution: InterAKT extensions are known to cause a high increase in productivity. For those of you who can't deal with the free time please continue to hand-code or refer to our competitors for help.

MX Kollection
Suite of Dreamweaver extensions to speed up dynamic web development
Interakt

$100 coupon - 199920002001

Interakt macromedia approved macromedia ALLIANCE PARTNER http://www.interaktonline.com/

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.

macromedia ALLIANCE PARTNER

Paper|Thin Partner

BlueDragonCertified V.I.P. Hosting Partner

Visit us online or call to order!

WWW.CFDYNAMICS.COM

1 - 866 - 233 - 9626
866 - CFDYNAMICS

COLDFUSION Developer's Journal

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

editorial

editor-in-chief
Simon Horwith simon@sys-con.com
technical editor
Raymond Camden raymond@sys-con.com
editor
Nancy Valentine nancy@sys-con.com
associate editor
Seta Papazian seta@sys-con.com
research editor
Bahadir 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.com
Tami Beatty tami@sys-con.com
Andrea Boden andrea@sys-con.com

contributors to this issue
Kelly Brown, Tim Buntel, Isaac Dealey, Hal Helms, Simon Horwith, Jeff Houser, Duncan Jack, Nik Molnar, Leon Oosterwijk, Guy Rish, Tom Schreck

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

It's a Great Issue This Month - Even Better Issues on the Horizon!

By Simon Horwith

This month was a busy one for the ColdFusion development community, so it's not surprising that we have two community-related columns this month.

Kelly Brown reviewed the CFUnited conference (if you weren't there, you really missed out on a great conference) and in my regular community column I discuss all of July's community events including the CF 10th birthday party at Macromedia headquarters in Boston, CFUnited, and many announcements about upcoming conferences. The most noteworthy event announcement is that registration for MAX 2005 is now open, so read the article and then go register! I'm looking forward to seeing all of your familiar faces there again this year. Another noteworthy bit of community news is that Macromedia is now going to support the CFEclipse project. This is significant news for developers and I was extremely happy to hear it publicly announced.

We also have many XML-related articles. Our case study column by Tom Shreck is a product review and development case study of cfcPowerTools and how it uses XML. Nick Molnar has written an excellent article about using XPATH in CFML applications. We have an article by Leon Oosterwijk that introduces you to XML basics and then shows how you can use ColdFusion to parse an iTunes music library. In his regular CF101 column, Jeff Houser explains the basics of XML in ColdFusion, for those of you who need an introduction to and explanation of the basics.

In addition to the XML and community articles I've mentioned, we also have some terrific articles about other topics. In the regular "Macromedia Speaks Out" column, Tim Buntel introduces the ColdFusion Administrator API. This API is new in CFMX 7 and allows developers to programmatically access the CF Admin functionality and data. Duncan Jack has written a terrific article about UML and UML tools, which I highly recommend. In his regular column, Hal Helms has written another "etudes" article – focusing on ColdFusion structures this month. We also have a great article from frequent contributing author Isaac Dealey that explains the Rule Manager Façade design pattern. Last but certainly not least, Guy Rish has written Part 2 in his Event Gateways article series. If you're not using or experimenting with gateways yet, I strongly urge you to do so…they're unbelievably powerful and useful in application development.

While I was at the CFUnited conference, I spoke with many of our readers and authors about what we can do to make the magazine better. As a result, more monthly columns are currently in the works, including collaborating with House of Fusion, introducing more server admin topics, and (re)introducing a "letters to the editor" column. If I receive emails (simon@horwith.com) I promise to add this column to the magazine. I haven't included a contest column this month, as I have not received any entries for the last two contests. As soon as time permits, I'll go ahead and develop entries for those myself if nobody beats me to it. There is a very interesting upcoming issue in the works right now, which I won't say anything more about other than the fact that it will aim to explore all of the various frameworks and development methodologies that are popular with ColdFusion developers today. I'm really looking forward to that issue and I'm sure it's going to be of major interest to all of our readers.

About the Author
Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal. *Simon is a Macromedia Certified Master Instructor and a member of Team Macromedia. 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

Seapine Software™
changing the world of software development

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.

White Paper
Change Management and Dreamweaver
Seapine Software

Want to learn more?

Visit www.seapine.com/webdev to discover how Surround SCM can save you time and headaches. Be sure to download our white paper, **Change Management and Dreamweaver**, and learn why change management is the new must-have tool for web development.

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

Surround SCM Source Control
TestTrack Pro Issue Management
QA Wizard Automated Testing

macromedia ALLIANCE PARTNER

software development 13th annual product excellence award Jolt

ColdFusion MX 7 Administrator API

Solve key limitations of the Web-based CF administrator

By Tim Buntel

The ColdFusion Administrator is an extremely powerful and easy-to-use Web-based interface for managing your ColdFusion environment. It provides scores of pages to configure data sources, server settings, Web services, logging and debugging, and more.

Pretty much any aspect of your ColdFusion server can be controlled and configured by the administrator. That's a great feature of ColdFusion, but it does present some problems.

In developing ColdFusion MX 7, we contemplated the challenges associated with providing a single way to administer the server and, as a result, decided to introduce a new feature, the ColdFusion Admin API, which allows you to perform most ColdFusion MX administrator tasks programmatically.

A common complaint about the Web-based administrator is that there is only one password required to access all the functionality of the Admin and, once in, you can perform many different types of tasks. Some are developer tasks: adding data sources, toggling debugging on and off, monitoring log files, and so on. Some are security-related tasks such as setting passwords and creating and modifying security sandboxes. Others are administrative tasks such as configuring mail servers, Verity, and server mappings. In most companies, these different tasks fall to different individuals. For example, a developer who needs to add a new access data source should likely not be permitted to change the admin password for the whole server! And do you want to grant full administrative permission to a developer just to allow him or her to turn on debugging for a project?

Another problem with the Web-based administrator is that developers could not incorporate any administrative tasks into the context of their applications. Perhaps you make a change to the signature of a Web service and need to "refresh" the Web service stubs created by the server. Or you need to clear the trusted cache if a certain event occurs in your application. Unfortunately, any administrative task required a human administrator, which made these kinds of programmatic implementations impossible.

administrator.cfc	Contains basic Administrator functionality, including login, logout, the Migration Wizard, and the Setup Wizard. You must call the login method before calling any other methods in the Administrator API.
base.cfc	Base object for all other Administrator API CFCs.
datasource.cfc	Add, modify, and delete ColdFusion data sources.
debugging.cfc	Manage debug settings.
eventgateway.cfc	Manage event gateways.
extensions.cfc	Manage custom tags, mappings, CFXs, applets, CORBA, and web services.
mail.cfc	Manage ColdFusion mail settings. runtime.cfc Manage runtime settings for fonts, cache, charts, configuration, and other settings.
security.cfc	Manage passwords, RDS, and sandbox security.
serverinstance.cfc	Start, stop, and restart JRun servers. This CFC only works when running the multiserver configuration.

Figure 1: The ColdFusion MX7 Admin API components

–continued on page 14

Using XML in ColdFusion

A starting point

By Jeff Houser

XML is over 8 years old. How many of you have been using it that long? (Feel free to raise your hand.) I don't see a lot of hands, which doesn't surprise me. However, I bet a lot of you have used XML without even knowing it.

XML has great flexibility and there are numerous uses. Its use is becoming more and more common with every passing day. You've probably come across it in some form even if you didn't know it was XML. If you haven't used it yet, you will soon. This article is all about XML and XML within ColdFusion.

What Is XML?

An explanation seems like a good way to start this article, so what is XML exactly? XML stands for eXtensible Markup Language. This markup language is tag-based similar to HTML or (most of) CFML. Whereas HTML is used to define the look and feel of a Web page and CFML is designed to issue commands to the ColdFusion application server, XML is used to define data. The primary use of XML is to share data between different systems. Sometimes these systems are radically different.

The main advantage to using XML is that if properly designed, it can be easily read by both a computer and a person (even some non-technical people). It supports Unicode so many types of information can be translated. XML is usually provided as plain text and is free from any license restrictions and easily readable from any language. It does have some drawbacks, however. The syntax is fairly verbose. That can make for good readability, but can decrease processing efficiency especially when dealing with large amounts of data. Binary data formats are known to be much more efficient. As an example, Flash Remoting is more efficient than Web services for this very reason. XML does not have any inherent support for advanced data types, so you'll have to run some processing to turn an XML object into a native ColdFusion structure or array. As you have seen in this article it can be cumbersome to access individual elements in an XML document.

There are two requirements of an XML document: they must be well formed and the must be valid. A well-formed document is one that conforms to the syntax rules. To make sure that you conform you should follow some of these simple rules:

- Close all tags. Even tags that do not contain data between the open and close tag need to be closed. It's fine to use the shorthand syntax, which is to close it before the close bracket, but it must be closed.
- Make sure there is only a single root element. Multiple root elements are invalid in XML.
- Put all attribute values in quotes, either single or double.
- Tags must not overlap. You can nest tags, but make sure you properly close all the inner tags before you close the outer tags.

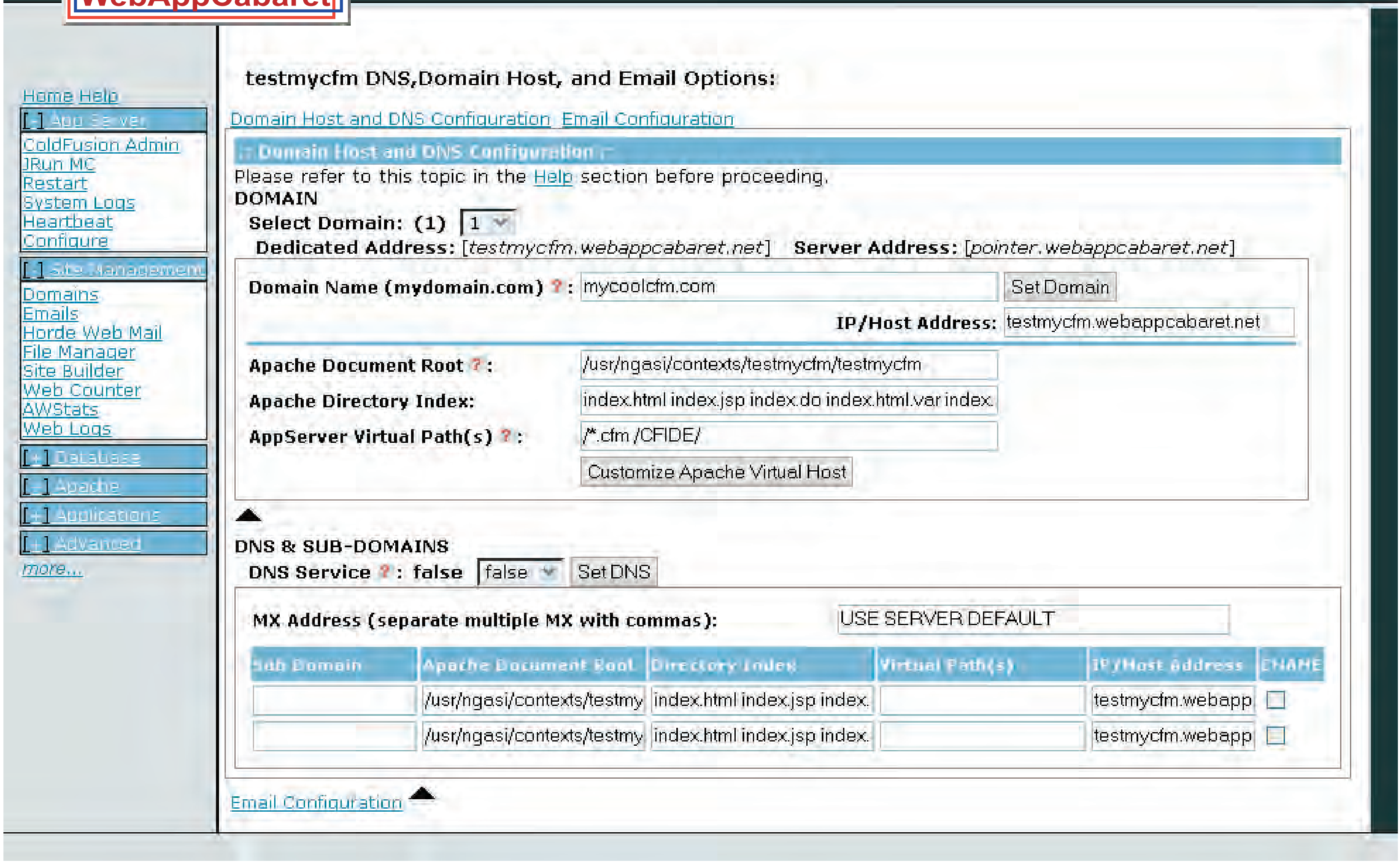
Follow these instructions to make sure that your XML is well formed.

To determine validity, we compare the XML document against the DTD (Document Type Definition) or XSD (XML Schema Definition). The DTD, or XSD, define the tags and attributes to those tags that make up your XML syntax. Both HTML and CFML have a specific set of defined tags and the attributes that can be used with those tags, but XML does not. It's up to you to define the tags that make up your XML in a DTD or XSD. You can use a DTD or XSD to define things like the names of tags, their data types, their attributes, whether they are required or optional, and their children if any. To determine if an XML document is valid, compare the XML to the rules in the DTD or XSD.

Unfortunately, in my experience with the corporate world, most developers do not bother to create either of these types of documents, so verifying for validity is often

WebAppCabaret℠

ColdFusion MX 6.1 Enterprise Hosting made easy.



Home Help
[-] App Server
ColdFusion Admin
JRun MC
Restart
System Logs
Heartbeat
Configure
[-] Site Management
Domains
Emails
Horde Web Mail
File Manager
Site Builder
Web Counter
AWStats
Web Logs
[+] Database
[-] Apache
[+] Applications
[+] Advanced
more...

testmycfm DNS,Domain Host, and Email Options:

Domain Host and DNS Configuration Email Configuration

:: Domain Host and DNS Configuration ::

Please refer to this topic in the Help section before proceeding.

DOMAIN

Select Domain: (1) 1

Dedicated Address: [*testmycfm.webappcabaret.net*] Server Address: [*pointer.webappcabaret.net*]

Domain Name (mydomain.com) ?:	mycoolcfm.com
IP/Host Address:	testmycfm.webappcabaret.net
Apache Document Root ?:	/usr/ngasi/contexts/testmycfm/testmycfm
Apache Directory Index:	index.html index.jsp index.do index.html.var index.
AppServer Virtual Path(s) ?:	/*.cfm /CFIDE/

DNS & SUB-DOMAINS

DNS Service ?: false false

MX Address (separate multiple MX with commas): USE SERVER DEFAULT

Sub Domain	Apache Document Root	Directory Index	Virtual Path(s)	IP/Host address	CNAME
	/usr/ngasi/contexts/testmy	index.html index.jsp index.		testmycfm.webapp	☐
	/usr/ngasi/contexts/testmy	index.html index.jsp index.		testmycfm.webapp	☐

Email Configuration

Imagine a hosting company dedicated to meet the requirements for complex web sites and applications such as those developed with ColdFusion MX.
At WebAppCabaret our standards based process and tools make deploying ColdFusion MX applications as easy as a point-and-click.
We call it **Point-and-Deploy Hosting**.
Our advanced NGASI Web Hosting management Control was designed for the hosting and management of ColdFusion web sites and applications thus cutting down on maintenance time.

Backed by an experienced staff as well as a **Tier 1 Data center** and network. Our network is certified with frequent security audits by reputable security firms.

All ColdFusion hosting plans have separate and individual installation AND instances of ColdFusion MX 6.1 Enterprise with **full access to ColdFusion Administrator** and JRun Management Console; so there is virtually no restriction or customization required for your application.

Complete power and control at the tip of your fingers. We take care of the system and hosting infrastructure so you can concentrate on development and deployment of your application. That is **real ROI**.

Log on now at http://www.webappcabaret.com/cdj.jsp or call today at 1.866.256.7973

NGASI POWERED

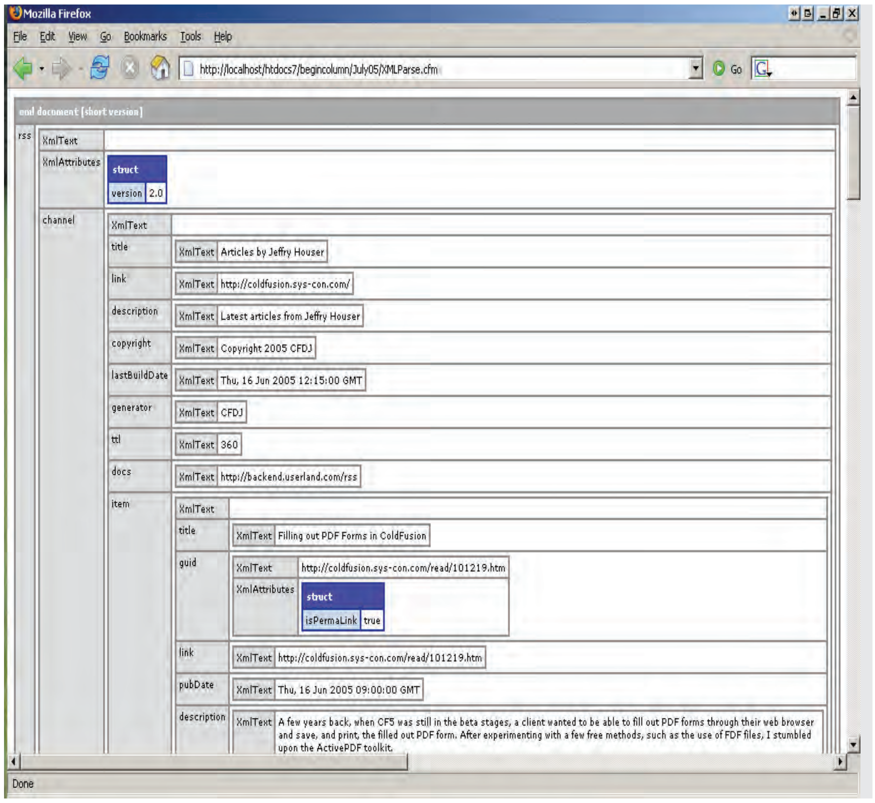
WebAppCabaret℠

Point-and-Deploy J2EE Hosting

http://www.webappcabaret.com/cdj.jsp **1.866.256.7973**

not an option. But, there are some common implementations of XML that are in use. You've probably heard or used many of them. Here are a few:

- ***WDDX***: WDDX is a specific dialect of XML and one that integrated with ColdFusion even before CF had native XML capabilities. If you ever used WDDX, you were using XML without even knowing it.
- ***RSS***: RSS is used for thesyndication of Web data. Its usage is very common in blog sites. CFDJ offers RSS feeds for their articles. This example shows an RSS feed of all the articles that I've written: http://coldfusion.sys-con.com/author/3566Houser.rss.
- ***SOAP:*** SOAP is the language of Web services. It stands for Simple Object Access Protocol and is XML.


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://localhost/htdocs7/begincolumn/July05/XMLParse.cfm
Go
Done

```
xml document [short version]
rss
  XmlText
  XmlAttributes   struct
                  version  2.0
  channel
    XmlText
    title          XmlText  Articles by Jeffry Houser
    link           XmlText  http://coldfusion.sys-con.com/
    description    XmlText  Latest articles from Jeffry Houser
    copyright      XmlText  Copyright 2005 CFDJ
    lastBuildDate  XmlText  Thu, 16 Jun 2005 12:15:00 GMT
    generator      XmlText  CFDJ
    ttl            XmlText  360
    docs           XmlText  http://backend.userland.com/rss
    item
      XmlText
      title        XmlText  Filling out PDF Forms in ColdFusion
      guid         XmlText        http://coldfusion.sys-con.com/read/101219.htm
                   XmlAttributes  struct
                                  isPermaLink  true
      link         XmlText  http://coldfusion.sys-con.com/read/101219.htm
      pubDate      XmlText  Thu, 16 Jun 2005 09:00:00 GMT
      description  XmlText  A few years back, when CF5 was still in the beta stages, a client wanted to be able to fill out PDF forms through their web browser
                            and save, and print, the filled out PDF form. After experimenting with a few free methods, such as the use of FDF files, I stumbled
                            upon the ActivePDF toolkit.
```

Figure 1: cfdump an XML object

XML Structure	Data
<?xml version="1.0" encoding="UTF-8"?>	No data to Access
<rss version="2.0">	MyXML.rss.xmlattributes.version
<channel>	No data to Access
<title>Articles by Jeffry Houser</title>	MyXML.rss.channel.title.xmltext
<link>http://coldfusion.sys-con.com/</link>	MyXML.rss.channel.link.xmltext
<description>Latest articles from Jeffry Houser</description>	MyXML.rss.channel.description.xmltext
…	…
<item>	No data to access
<title>Filling out PDF Forms in ColdFusion</title>	MyXML.rss.channel.item[1].title.xmltext
<guid isPermaLink="true">http://coldfusion.sys-con.com/read/101219.htm</guid>	MyXML.rss.channel.item[1].guid.xmlattributes.ispermalink MyXML.rss.channel.item[1].guid.xmltext
<link>http://coldfusion.sys-con.com/read/101219.htm</link>	MyXML.rss.channel.item[1].link.xmltext
…	…
</item></channel></rss>	No data to access

Table 1: How to access XML

- ***XHTML:*** XHTML is an XML version of XML. It has a stricter syntax than HTML (for example, all tags must be closed). It is expected to be the long-term successor to HTML.
- ***MXML:*** MXML is the language of Macromedia Flex and it's an XML implementation.

Next, I'm going to show you how to create and use XML objects in ColdFusion.

Creating an CF XML Object

There are two different ways that you can create an XML object inside ColdFusion. The first is to use the CFXML tag, and the second is to use the XMLParse function. CFXML is good if you are going to create an XML document from scratch within ColdFusion. The XMLParse function will take a string that contains an XML document and return the XML document.

The cfxml tag contains two attributes:

- ***Variable***: The variable attribute defines the variable that will contain the resulting XML object. It is required.
- ***CaseSensitive:*** The CaseSensitive attribute specifies whether the case used in the XML document should be maintained. It is an optional Boolean attribute with the default value of no.

Between the open cfxml tag and the close cfxml tag, specify the XML. For our example, I'm going to use some XML borrowed from the *CFDJ* RSS feed (see above for the link (see Listing 1).

I cut and pasted the XML directly from the *CFDJ* feed, then modified it to include fewer records. I set it to not be case sensitive, and stored the XML object in the variable "MyXML". The last line of code dumps the object. Inside the cfxml tags you can use any sort of ColdFusion processing that you need to create your XML object. For example, suppose you wanted to loop over the records of a database and use those records to create an XML object? We can do that with the CFXML tag.

XMLParse is a function used to turn a string into an XML object. Perhaps you are consuming a Web service that returns XML or using cfhttp to get an XML document. The XMLParse function has one argument, which is the string that contains the XML.

```
<cfhttp url="http://coldfusion.sys-con.com/
author/3566Houser.rss">
</cfhttp>

<cfset MyXML = XMLParse(cfhttp.FileContent)>

<cfdump var="#MyXML#">
```

The code grabs the CFDJ RSS feed using the cfhttp tag. The cfhttp.filecontent variable is a string. The XMLParse function is used to turn that string into an XML variable. The results dump the XML object to the screen, as shown in Figure 1.

Accessing the XML Object

Now that you've created an XML object, how do you use it within ColdFusion? This section will show you how. An XML object in ColdFusion is often a group of embedded structures and arrays. You can tell what's what by viewing the long version of the cfdump. Sometimes it can be tricky to parse this data because if its complexity. This is a list of elements that we'll need to access our data in the XML object:

- ***XMLRoot:*** The XMLRoot is stored as an "XMLElement". You'll use this to access most of the data in your object. Each additional element I discuss here is part of the XML Element object. This keyword "XMLRoot" is interchangeable with the name of the top-level node. In the RSS feed, we could use "MyXML.xmlroot" or "myXML.rss".
- ***XmlName:*** The name of the XML Element and it's a string.
- ***XmlText:*** This is the content that resides between an open and close tag. It's a string.
- ***XmlAttributes:*** The XMLAttributes is a structure. It contains the name, and values, of each attribute of the XML Element.
- ***XmlChildren:*** XMLChildren is an array of all the children's elements. Each child is treated as an XML Element of its own.

A complete list of what is associated with an XML document in ColdFusion is located here http://livedocs.macromedia.com/coldfusion/7/htmldocs/00001510.htm#1119477. Table 1 shows how to access each line of the original XML file.

This is an example of what we would use to loop over the XML document and display a link to each article in the RSS feed:

```
<cfloop from="1" to="#arraylen(MyXML.rss.chan-
nel.item)#"  index="CurrentLink">
<cfoutput>
 <a href="#MyXML.rss.channel.item[CurrentLink].
link.xmltext#">
  #MyXML.rss.channel.item[CurrentLink].title.
xmltext#
 </a><br>
</cfoutput>
</cfloop>
```

You've probably run this type of code many times against a ColdFusion query object. The code loops over the array of items. In the XML object items is an array of "XMLChildren". It displays a link using the "link" and displays the title as the link text, using a type of access similar to what is in the table.

Where to Go from Here

With this basic introduction to XML, where do you go from here? Well, you'll definitely want to read more about ColdFusion's XML functions: http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000372.htm#3468770. You may want to investigate xPath to allow you to search through an XML object to find more information. XSL (Extensible Style sheets) are also intriguing. They allow you to transform an XML document using a style sheet to make it ready for Web display. ColdFusion supports XSL with the XMLTransform function. Information XSL is located here: http://livedocs.macromedia.com/coldfusion/7/htmldocs/00001520.htm#1209889 and the XMLTransform function is here: http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000673.htm#140093. This article should give you a good starting point for using XML in your applications.

About the Author

Jeff 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 ColdFusion, most recently ColdFusion MX: The Complete Reference *(McGraw-Hill Osborne Media).*

jeff@instantcoldfusion.com

Listing 1

```
<cfxml casesensitive="no" variable="MyXML">
 <?xml version="1.0" encoding="UTF-8"?>
 <rss version="2.0">
  <channel>
  <title>Articles by Jeffry Houser</title>
  <link>http://coldfusion.sys-con.com/</
link>
  <description>Latest articles from Jeffry
Houser</description>
  <copyright>Copyright 2005 CFDJ</copy-
right>
  <lastBuildDate>Thu, 16 Jun 2005 09:32:00
GMT</lastBuildDate>
  <generator>CFDJ</generator>
  <ttl>360</ttl>
  <docs>http://backend.userland.com/rss</
docs>

  <item>
   <title>Filling out PDF Forms in ColdFu-
sion</title>
   <guid isPermaLink="true">http://coldfu-
sion.sys-con.com/read/101219.htm</guid>
   <link>http://coldfusion.sys-con.com/
read/101219.htm</link>
   <pubDate>Thu, 16 Jun 2005 09:00:00
GMT</pubDate>
   <description>
A few years back, when CF5 was still in the
beta stages, a client wanted to be able
to fill out PDF forms through their Web
browser and save and print the filled out
PDF form. After experimenting with a few
free methods, such as the use of FDF files,
I stumbled upon the ActivePDF toolkit.
   </description>
  </item>
  </channel>
 </rss>
</cfxml>

<cfdump var="#MyXML#">
```

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

Searching through your iTunes Library using ColdFusion

Accessing XML data in ColdFusion

By Leon Oosterwijk

Unless you've been stuck under a rock for the last five years you'll undoubtedly have noticed that XML is starting to become part and parcel in the software industry. In the past CSV, INI and fixed width files were the standard way to exchange information between systems.

Over the last several years with the rise in cross platform development and the need to exchange information between many dissimilar systems, the need for a better way to exchange information became evident. Building on the success of HTML and the richness of the SGML document standard developed by IBM, XML was the next step. Even though XML is a simpler version of SGML, it is still pretty complex. With the introduction of MX, ColdFusion now natively supports working with XML documents. Ten years ago ColdFusion made web application development easier and faster in the same way ColdFusion makes it possible to work with XML without being overwhelmed by its complexities.

To demonstrate the power of using ColdFusion to access XML data, we'll look at a small sample application that allows you to search through an iTunes library. iTunes stores information about your music files in an XML file. We'll look at an application that allows people to search for songs in your iTunes library through a web interface using ColdFusion.

XML Overview

An XML document at its most basic level is a collection of information stored in a structured tag-based format. Unlike HTML this format is not rigidly defined by a set of standardized tags. In XML you can come up with your own tags to describe your data. You also have the option to generate a schema that describes how different elements in the XML file fit together. There are two ways to describe a schema: Document Type Definition (DTD) and XML Schema Definition (XSD). DTD was part of the XML standard from the very beginning, but XSD is much more powerful. With either schema you can validate an XML file to confirm that it contains valid data. This can be very useful when receiving files from third parties that have to conform to a certain standard. Another powerful feature of XML documents is the ability to create an eXtensible Stylesheet Language (XSL) document. Based on such a document an existing XML file can be transformed into a different format.

XML in ColdFusion

ColdFusion supports the creation, parsing, transforming and searching of XML files through a small set of powerful functions. Like all other ColdFusion Functions, the XML functions are fully internationalized, allowing you to write XML containing information in different languages and character sets. If you have ever written JavaScript, you might be familiar with the Document Object Model (DOM). This hierarchical structure of nodes is how your browser stores a web page in memory. XML documents are also represented in a DOM-like fashion. ColdFusion represents each XML node as a structure with a number of attributes like XmlName, XmlAttributes and XmlChildren. With ColdFusion access to information in an XML document is as easy as accessing information in a structure.

Accessing XML Data in ColdFusion

To start we need to understand how iTunes stores the information about your music files. If you have iTunes installed, you can search your hard drive for the "iTunes Music Library.xml" file. If you open this file in a text editor, you'll see the information about your music library stored in an XML format. ColdFusion MX was the first version of ColdFusion to include support for working with XML files. CFMX7 built upon this to add even more features for XML transforming, validating and searching. By calling the xmlparse() function and passing in a string, file location, or URL, you can create an XML document object. To open the iTunes XML file, we call:

```
<cfset variables.xmlDoc = XmlParse(getProperty('iTunesLibrary'))/>
```

If the file is valid ColdFusion will create an XML Document object out of the XML data. To use this information, we have to understand a little about how iTunes stores the data. Here are the first three lines of the 'iTunes Music Library.xml' file.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN" "http://www.
apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
```

The first line tells your XML parser that the information that is to follow adheres to the XML standard 1.0 and that the data is UTF-8 encoded. The next line describes the document type, and in this case, also gives a DTD file that explains the schema for the XML data that is to follow. If you follow the link to the DTD file, you'll notice that it is very generic. Apple Computer uses the Property List format for much of its XML files and keeps the format very basic. The next line is the Document Element, or Root node of the XML document. All the XML defined in this document must fall between the starting and closing Document Element Tags. Between this <plist> tag we find a series of <dict> tags which enclose a succession of <key> tags followed by an <integer>,<string> and other <dict> tags. The full track listing is enclosed in the first <dict> tag that follows the <key> tag with the value 'Tracks'. This happens to be the first <dict> tag that is nested in the root <dict>. Because this is really the only part in the XML file we're interested in we'll create a variable to hold that sub-section

```
<cfset tracks = xmlDoc.plist.dict[1].dict[1]/>
```

The track data is organized as a series of <key> tags holding a unique (to this library) track ID following the <key> tag is another <dict> tag which holds a series of <key> tags like Name, Artist and Album. followed by <string> and <integer> tags holding the values for these <key> tags. This format is not very easy to eyeball, so it is possible to use an eXtensible Stylesheet Language (XSL) document to translate this schema to be more intuitive. XML.com has an excellent example of an XSL to do this conversion at http://www.xml.com/pub/a/2004/11/03/itunes.html?page=last.

In our example we won't translate the XML schema. Instead we'll turn the search results in a good old fashioned ColdFusion Query. A benefit of not translating the XML to a different schema is that we can search through the Artist, Album, Genre and other values simultaneously by searching through all the <string> tags.

Searching Through an XML Document

ColdFusion allows you to search XML documents using the XmlSearch function. This function takes arguments in the form of an XPath query. XPath is syntax to query XML files defined by the World Wide Web Consortium (W3C). ColdFusion supports a subset of the full XPath definition. One of the parts of XPath that ColdFusion doesn't support is the ability to make your search case insensitive. The syntax for our XmlSearch is 'dict[string/text()='#searchString#']'. When passing this XPath query to the ColdFusion XmlSearch function it will return an array of <dict> nodes whose descendent <string> tag content matches the searchString.

```
<cfset arrResult = XmlSearch(tracks, xPathString)/>
```

Turning XML Data into a Query

We can now loop through this array and build a ColdFusion Query Row out of each element.

```
<cfset resultcount = arraylen(arrResult)/>
<cfset rstSearchResult =  QueryNew('track,time,artist,album,genre')/>
<cfset QueryAddRow(rstSearchResult, resultcount)/>
<cfloop from="1" to="#resultcount#" index="intI">
```

Because the number of elements might be different from track to track we loop through all of them and pull out only ones in which we're interested. If the current node is a <key> tag with the value Name, we know that the next node must be a <string> tag with the value of the track name. We can now populate the different columns in the query. Notice that we access the next element in the array by referencing the current node plus one.

```
<cfswitch  expression="#arrResultDictNodes[intJ].XmlText#">
<cfcase value="Name">
     <cfset QuerySetCell(rstSearchResult, "track", arrResultDictNodes[intJ+
1].XmlText, intI)/>
</cfcase>
</cfswitch>
```

After the loop is done building the query object we can pass it to our display template to format the search results. We could have also written an XSL transformation to transform the found XML nodes into valid HTML.

Conclusion

While this article doesn't cover creating XML files in ColdFusion, the code to do so is similar for reading XML files. Check the ColdFusion documentation for a full list of the functions dealing with writing and reading XML information at: http://livedocs.macromedia.com/coldfusion/7/htmldocs/wwhelp/wwhimpl/common/html/wwhelp.htm?context=ColdFusion_Documentation&file=00001505.htm.

Knowing a little bit about accessing XML files using ColdFusion can be a great way to make your applications more powerful, cross-platform compatible and internationalized. With the growth of Web Services, RSS and other XML-based technologies basic skills in this area will only become more useful. While ColdFusion's XML support is not all inclusive it gives you enough power to quickly develop excellent data-driven applications; just like we're used to.

About the Author

Leon Oosterwijk is a senior Web developer for the Lampo Group (www.daveramsey.com). He lives in Nashville, TN, and his first exposure to ColdFusion occurred in 1996.

leon.oosterwijk@gmail.com

ColdFusion MX 7 Administrator API

–continued from page 7

The ColdFusion Administrator API solves these issues by giving developers the ability to perform most of the tasks that are possible in the Web-based administrator in their CFML code. The Administrator API consists of a set of ColdFusion components (CFCs) that contain methods you call to perform administrator tasks. For example, use the setMSQL method of datasource.cfc to add a SQL Server data source. So let's see how the Admin API could offer a solution to some of the problems mentioned above.

The Admin API allows you to create custom administrator applications. If you want a couple of developers to be able to add new data sources and nothing else, you can create a simple form-based interface that calls the data source cfc. This would also allow you to build in additional security and logging to these functions. For example, you can see which developer logged in at what time and what data sources were added.

The Admin API is also used in the ColdFusion MX 7 Extensions for Dreamweaver to enable the new debugging functionality. Prior to the availability of these extensions, you were required to enter a developer's IP address from the ColdFusion administrator to allow debugging output to be used from his or her machine. Now, the extensions make a call to the Admin API to handle this automatically. It sends a request to the ColdFusion server, asking it to enable debug mode for the current workstation's IP only, then it sends the actual page request and retrieves the results. After the request has been completed and all desired data retrieved, it sends another call asking to disable the debug mode for the current IP.

Since the Admin API is called from regular CFML code, any of the functions that it exposes can be called from any CF page. For example, the situation mentioned above for Web services could be solved with this simple code:

```
<cfinvoke
 component="CFIDE.adminapi.extensions"
 method="reloadWebService">
   <cfinvokeargument name="name" value=""/>
   <cfinvokeargument name="path" value=""/>
</cfinvoke>
```

The Admin API is broken out across several components:
The easiest way to find out all of the different functions in each is to use the CFC panel in Dreamweaver. Select CF Components from the dropdown at the top of the components tab of the application panel. Scroll down to find the CFIDE.adminapi package and expand. Each component listed above can be expanded showing all of the functions, their required parameters, and so forth.

The addition of the Administrator API to ColdFusion MX 7 allows developers to solve some key limitations of the Web-based ColdFusion administrator. You can create flexible, custom "mini-Administrators" to grant only certain abilities to certain developers; script administrative functionality into a ColdFusion application; and much more. Best of all, the API is a core, supported feature of the product: you'll never again need to worry about tinkering with the service factory or other "undocumented" approaches to programmatic administrator access.

About the Author

Tim Buntel is senior product manager for ColdFusion at Macromedia.

tbuntel@macromedia.com

HOSTING.COM

THE FIRST NAME IN HOSTING

Developers!

Developers!

Developers!

We Love Them!

SERVE. SUPPORT. SECURE. STORE.

www.hosting.com

July Was Busy for the ColdFusion Community!

By Simon Horwith

July was an exceptionally active month in the ColdFusion Community. First, the month began with the conclusion of the CFUnited Conference. Formerly known as "CFUN," CFUnited was a ColdFusion Conference the likes of which I haven't seen since the old DevCon days.

The conference boasted nearly 900 attendees, featured over 60 sessions from some of the biggest names in CF, and was a ton of fun for all. It's worth mentioning that the MiniMAX 2 conference was held the night before CFUnited and was a great success. This year, CFUnited has established itself as the premier event for ColdFusion developers. I enjoyed meeting many of our readers there and can't wait for next year's conference (to be held at the same time in the same location). You can read more about CFUnited in Kelly Brown's conference review in this month's issue. One announcement made by Macromedia at CFUnited that is of significant importance to the CF development community is that Macromedia is now officially going to help support and get involved in the CFEclipse project.

Also this month, TeraTech announced the call for speakers for their annual Fusebox conference. This year, the conference is not only covering Fusebox topics, but topics dealing with various frameworks (it's also been renamed the "Fusebox and Frameworks" conference). Not being a big Fusebox guy myself, I was happy to see the addition of topics surrounding other frameworks. This conference will be held from Sept. 28–30, 2005, in Rockville, MD. More information can be found at www.cfconf.org/fusebox2005.

Speaking of announcements, this month Macromedia officially opened registration for MAX 2005. MAX, for those of you who aren't familiar with it, is Macromedia's big annual U.S. conference. This year MAX is a West Coast event; it's being held in Anaheim, CA, from Oct. 16–19. This year, more than any previous year, I've had a lot of ColdFusion developers ask me why they should attend MAX, when ColdFusion topics make up only a percentage of the topics, when they can go to CFUnited where it's 100% ColdFusion. That's a reasonable question and I'd like to take a

moment to address this concern.

The MAX conference is not a ColdFusion conference, and shouldn't be thought of as such. MAX is a "Macromedia Technology Conference." ColdFusion is one of Macromedia's products, and an important one at that, and there are many CF topics there. Because MAX covers all of Macromedia's suite of products, there are also many sessions that teach integration between ColdFusion and other Macromedia technologies (like Flash and Flex).

Personally, I find MAX to be the most educational conference of the year, every year. If all you do is plain vanilla CF and all you care about is plain vanilla CF, you might not feel the same. For me, though, the conference is an opportunity to learn about what's new not only with CF but also with Dreamweaver, Flash, and Flex (and any other product I might be interested in). One thing I spoke about frequently at CFUnited was the fact that these days I'm using Flex a lot…and I'm loving it. The MAX conference will be an excellent chance to look at what other people are doing with Flex, including Flex/CF integrations. I personally believe that we've only just begun to see just what Flex can do…its adoption rate and functional potential has yet to be fully realized. I firmly believe that there will come a time in the near future when developers with Flex skills will be in very high demand. The other technology that I believe will be huge in the near future is Flash on devices. Looking at the session topic list, I'd say Macromedia feels the same way. The bottom line, though, is that if you want to stay on the cutting edge of what's happening with ColdFusion and the other Macromedia products, you

Savvy™
CONTENT MANAGER

Developers:
Finally, web content management that's Powerful and Affordable.

Our newest partners:
CFDynamics, Lumenbrite Training, Pixelspace, Sandstorm Design, roundpeg, Inc. and Uvault.com
– Sign up today!

See us at CFUnited
1.866.870.6358

www.BeSavvy.com
Savvy Software Inc.

macromedia®
ALLIANCE PARTNER

can't miss MAX. You can find out more about MAX at www.macromedia.com/macromedia/events/max. I hope I see many of you there.

For those of you who will be in Anaheim early, TeraTech will be holding their annual CFUnderground Conference just prior to the MAX event on Oct. 15 at the same venue. You can find out more about CFUnderground VII at www.cfconf.org/cf_underground7. No official announcement has been made about MiniMAX 3, but Adam Bell assures me that he is planning to organize a MiniMAX 3 event in Anaheim just before MAX.

The last bit of community news this month has nothing to do with any conference but rather ColdFusion's 10th birthday. Macromedia held a birthday party in their Newton office (the old Allaire headquarters) and broadcast the event to user groups and developers everywhere via Breeze Live. AboutWeb sent me there to show support along with Michael Perlstein, our VP of business development, who is in the process of developing a community-based network to assist ColdFusion developers in finding jobs.

The birthday party was held on the second floor of the Macromedia office, which, for the event, was adorned with ColdFusion memorabilia spanning the past 10 years. There were a couple dozen T-shirts, mugs, posters, and every other bit of CF-branded memorabilia you can imagine. Food and drinks were provided as the attendees, most of whom were Macromedia employees or former Allaire employees, mingled with one another. Macromedia's key community folks – Ed Sullivan, Amy Brooks, and Christine Lawson – were not only there participating but also organizing and running the event. All of my friends from Macromedia Training and from the ColdFusion Server Development Team were there in full force. Of course, Stephen Elop, Ben Forta, and Tim Buntel were in attendance, as were many former Allaire employees such as Jeremy Allaire, JJ Allaire, and Sim Simeonov.

The overall atmosphere was one of fun, relaxation, and reminiscence. To help with the reminiscing, Tim Buntel took us all on a trip down memory lane by hosting a panel made up of many former Allaire personnel as well as current Macromedia staffers. The panel members talked about the significance of the current release of ColdFusion as well as their thoughts on the future of the product, but most of the panel time was spent telling stories

from ColdFusion's illustrious past. The presentation/panel was kicked off with a terrific introduction by Macromedia CEO Stephen Elop. I spent the majority of the time socializing with my closest friends at Macromedia, and I'm no photojournalist but I did manage to take some photos while I was there. The party was a terrific event and it was great visiting the old office again and catching up with old friends. I only hope we can do it again in another 10 years.

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal *and is the CIO at AboutWeb, LLC, a Washington, DC based company specializing in staff augmentation, consulting, and training. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and specializes in ColdFusion application architecture, including architecting applications that integrate with 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

Efficient Web Content Management

CommonSpot™ Content Server is the ColdFusion developer's leading choice for efficient Web content management.

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

CommonSpot's open architecture and extensive APIs enable easy customization and integration of external applications. With CommonSpot, you can design and build exactly what you need. Best of all, CommonSpot puts content management in the hands of content owners. With non-technical users responsible for creating and managing Web content, developers are freed to focus on strategic application development.

Evaluate CommonSpot today.

To see your site *running* under CommonSpot, call us at 1.800.940.3087.

features.

- 100% browser-based
- Content object architecture
- Template driven pages
- 50+ standard elements
- Content reuse
- Content scheduling
- Personalization
- Flexible workflow
- Granular security
- Mac-based authoring
- 508 compliance
- Full CSS support
- Custom metadata
- Taxonomy module
- Extensible via ColdFusion
- Web Services content API
- Custom authentication
- Replication
- Static site generation
- Multilanguage support

1.800.940.3087
www.paperthin.com

Paper | Thin

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
ACTIVEPDF	WWW.ACTIVEPDF.COM	866-468-6733	27
BLOG-N-PLAY	WWW.BLOG-N-PLAY.COM	888-303-5282	29
CFDYNAMICS	WWW.CFDYNAMICS.COM	866-233-9626	4
COLDFUSION DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/COLDFUSION	888-303-5282	39
HAL HELMS, INC	WWW.HALHELMS.COM		33
HOSTING.COM	WWW.HOSTING.COM		15
HOSTMYSITE.COM	WWW.HOSTMYSITE.COM/CFDJ	877-215-4678	23
INTERAKT ONLINE	http://www.interaktonline.com	4031 401.68.19	3
INTERMEDIA.NET	WWW.INTERMEDIA.NET	800-379-7729	COVER IV
JAVA DEVELOPER'S JOURNAL	JDJ.SYS-CON.COM	888-303-5282	41
LINUXWORLD MAGAZINE	WWW.LINUXWORLD.COM	888-303-5282	45
MACROMEDIA	WWW.MACROMEDIA.COM/go/cfmx7_demo	415.252.2000	COVER II
MACROMEDIA MAX	WWW.MACROMEDIA.XOM/MAX	415.252.2000	COVER III
MX DEVELOPER'S JOURNAL	WWW.SYS-CON.COM/MX/SUBSCRIPTION.CFM	888-303-5282	35
PAPERTHIN	WWW.PAPERTHIN.COM	800-940-3087	19
SAVVY SOFTWARE	WWW.BESAVVY.COM	866-870-6358	17
SEAPINE SOFTWARE	WWW.SEAPINE.COM/WEBDEV	888-683-6456	6
SYS-CON E-NEWSLETTERS	WWW.SYS-CON.COM	888-303-5282	35
SYS-CON REPRINTS	WWW.SYS-CON.COM	201-802-3024	31
SYS-CON.COM	WWW.SYS-CON.COM	888-303-5282	49
WEBAPP CABARET	WWW.WEBAPPCABARET.COM/cdj.jsp	866-256-7973	9

Don't Miss CFDJ's Next Issue!

Debugging, logging/auditing, and testing – focus on the Various ways to use cfml, the coldfusion administrator, and other Technologies to debug and test code as well as techniques for Implementing auditing and logging features in your applications

Playing With(in) the Rules

The Rule Manager Facade

By Isaac Dealey

In a book entitled Finite and Infinite Games in 1986, James P. Carse wrote “Finite players play within the rules, infinite players play with the rules.” We play finite games every day, from checkers and chess to Yatzee and Monopoly. Finite games have a familiar pattern: a beginning, a middle, and an end; a winner and a loser.

A finite game is easy to play because it has a limited set of fixed parameters. Carse also wrote “Finite players play to win, infinite players play to keep playing.” Custom software development is not a finite game, and the clients who purchase it are not finite players. The objective of a workflow application is not to win an office betting pool, or even to design and complete a superior workflow application. The objective of a workflow application is to continue to distribute an ever-changing flow of work. Yet as custom software developers we tend to play the infinite game with finite rules.

We shouldn’t be taking a request from a client to change the way our application determines how work is distributed (15-day expiration instead of 30-day expiration, apply special notices to tasks above or below a particular dollar value which is later changed, etc). Instead we should be providing our clients with the tools to change these business rules themselves, so that instead of constantly trying to keep up with the stream of client change requests that result from hard-coded business logic, we can move on to the next feature or application, and play the infinite game on their terms, playing with the rules rather than playing within them.

A Brief Example

A typical business rule should be familiar to all of us and is usually described to us by a client with words like this: “commissions for this project are due to the sales person 15 days after receipt of at least 10 percent of the deposit.” This rule has one objective (indicating current payments due to the sales staff) and several parameters, including the required deposit amount, an arbitrary figure (10%), the date on which the figure is satisfied, an arbitrary duration (15 days), and by implication, the current date. These are typical logical equations and we know intuitively how to write software to perform these comparisons. The code usually looks something like this:

```
<cfif sale.getDepositPaidByDate(dateAdd(“d”,-15,now()))
  gte (0.1 * sale.getFullDepositAmount())>
```

As long as the application you’re writing will be sold only to one client, and that client never changes their rules, this line of code will survive the lifespan of our application. The problem with this is that, while it may satisfy the client it doesn’t help us much. It doesn’t allow us to sell the same application to another client without reinventing this portion of the code and so at the end of the day it amounts to being paid an hourly wage for your work instead of building equity in your software.

The Challenge

There are two barriers to transforming this finite equation into a flexible rule that many of your clients can modify without your aid or intervention:

1. ***Storage:*** Because the arbitrary rule criteria must become data that can be dynamically interpreted, the application will need a place to store rules of many disparate types with indeterminate numbers and types of parameters.
2. ***Interpretation:*** Once the storage of the rule data is accommodated, the application needs a system of managing and interpreting a vast array of indeterminate and ever-changing criteria to be applied to an indeterminate, ever expanding set of contexts.

You might notice in the wording of these barriers that I've used a lot of words like "indeterminate" and "expanding" and "changing." In order for your rule-management engine to provide maximum equity it needs to be designed with a minimum of assumptions about your clients' business model. Q. How many types of rules will there be? A. Unknown. Q. How many criteria will an individual rule have? A. Unknown. Q. In what type and how many contexts will the rules be applied? A. Unknown. Because there are so many unknowns it is necessary to use the most encapsulated and extensible tools available to store and to interpret the rules. To meet this challenge I choose Extensible Markup Language (XML) and ColdFusion Components (CFCs).

A single "RuleManager" CFC manages all interactions with the rule data in this application. In Object Oriented (OO) lingo this is known as a Façade pattern, in which a larger library of code is simplified by a single unified interface. Although the RuleManager will require a minimal set of assumptions, certain assumptions must be made in its design to ensure portability:

- Rules are stored in XML packets; each packet may contain zero or more individual rules:
  1. Each rule may contain zero or more individual criteria (nodes) of different types (date, numeric, money, text, etc.)
  2. A separate CFC must be written for each type of criteria that the RuleManager will interpret (date, numeric, text, etc.) – these criteria-type CFCs must share a common set of methods (interface)
  3. All parameters used by criteria-type CFCs in the evaluation of a rule must be provided in the criteria XML node or by an external context (CFC)

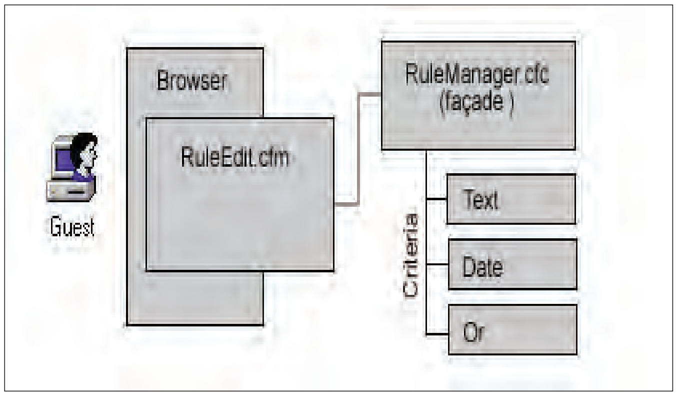


Figure 1: Assumptions

- When tested for applicability each rule must evaluate to a Boolean value (true/false)
- The criteria for a rule are cumulative, requiring the satisfaction of all criteria to apply the rule (logical and)
- A mechanism must be provided for the user to establish subsets of criteria that are not cumulative (logical or)

A Sample RuleManager Component

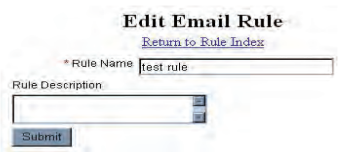
Due to publishing constraints I've simplified the sample RuleManager code quite considerably for this article. For example, the sample RuleManager doesn't provide any features for dynamically loading or unloading different criteria types or groups of criteria types – instead it merely examines a directory for CFCs and assumes each is a criteria type CFC. I've also eliminated debugging features, a considerable amount of i18n functionality for multilingual applications (which is no small challenge to implement with a RuleManager), simplified the provided criteria CFCs, and omitted nearly a dozen common criteria types. One feature I'm particularly sorry to omit is the ability for individual criteria CFCs to introspect the provided rule context to determine their own applicability to the context. Although I feel this is an important feature for the sake of encapsulation and extensibility, it would simply require too much space to include in this article.

Having trimmed the code for the RuleManager sample to a bare minimum, here are the absolutely necessary features of the façade (RuleManager.cfc) for this application to function:

- ***Rule CRUD:*** Create/Read/Update/Delete methods for individual rules must modify the loaded ruleset XML (stored in memory as a property of the RuleManager.cfc object) and return or provide a means of accessing the text-representation of the current ruleset document for storage (which can be stored either in a file or in a database). Methods: getRuleNode, setRuleNode, ruleExists, deleteRule.
- ***Criteria CRUD:*** Create/Read/Update/Delete methods for individual rule criteria. Although rules must be identified with a unique id such as a UUID, a criteria node within a rule may be identified by its position within the rule so long as the rule identifier is used when fetching the criteria node. Criteria create/update/delete methods must also return the text representation of the XML ruleset or provide a means of accessing its text-representation for storage (see above). Methods: getCriteriaNode, setCriteriaNode, getCriteriaForm, getCriteriaXML, getRuleCriteria, deleteCriteria.
- ***Criteria Object Management:*** In order for the RuleManager to perform efficiently it must cache criteria-type CFCs in memory and use the cached CFCs for managing and interpreting individual criteria nodes. The façade (RuleManager.cfc) must also provide a list of the available criteria types. Methods: getCriteriaObject, getCriteriaQuery.
- ***Test Rule:*** This is the singular feature for which all other features exist. Once a user has created their rules and assigned their criteria your application must have a means of identifying and testing the applicability of those rules to apply the various application features you want controlled by user-defined rules. Method: ruleApplies.

In addition to these features of the façade, a common interface is required for the different criteria-type CFCs, which will be managed by the façade. This is accomplished by designating a predefined set of methods (including arguments and return types) that all criteria-type CFCs must share. With these methods in place the façade can then reference any individual criteria-type without needing to know any of the internal mechanics of the criteria.

- ***void getForm(node , context [, nodeData]):*** This function displays a form for the user to update any criteria of the specified type. Although it may seem strange or even contrary to popular OO "best practice" to display the form from within the CFC, this approach provides maximum encapsulation. The content being updated by this form exists solely for the purpose of being used by this CFC and nothing else, and including the form in this method ensures that no criteria CFC will be without a user interface, and it also ensures that the façade is kept ignorant of the internal mechanics of the individual criteria-type CFCs. Called within RuleManager. getCriteriaForm.
  string getXMLNode(nodeData): Once the criteria form is submitted it is necessary to return to the criteria-type CFC to determine the syntax for the criteria node before updating the ruleset XML document. Called within RuleManager. setCriteriaNode.
- ***string describe(node , context):*** Because the façade must manage many different types of criteria it cannot contain any logic for describing the applied criteria and still maintain encapsulation. For this reason the describe method is required in each criteria-type CFC. Called within RuleManager. getRuleCriteria.
- ***boolean test(node , context):*** Finally in order to determine the applicability of any given rule, the RuleManager façade must be able to rely on the criteria-type CFCs that created the individual criteria XML nodes to interpret those same XML nodes. Called within RuleManager. ruleApplies.


Edit Email Rule

Return to Rule Index

* Rule Name	test rule
Rule Description	

Submit

Figure 2: Edit email rule

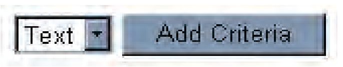
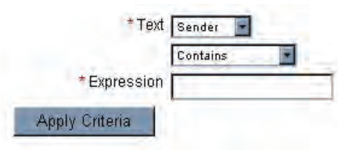


Figure 3: Selecting a criteria type


* Text	Sender
	Contains
* Expression	

Apply Criteria

Figure 4: The criteria form

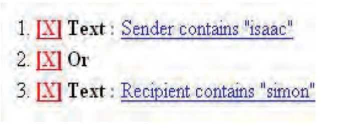

1. [X] Text : Sender contains "isaac"
2. [X] Or
3. [X] Text : Recipient contains "simon"

Figure 5: Adding additional criteria

At this point you might notice that almost all of the criteria-type interface methods include an argument named "context." If you remember from the previous sections of the article, one of the assumptions in the RuleManager application is "All parameters used by criteria-type CFCs in the evaluation of a rule must be provided in the criteria XML node or by an external context (CFC)." By providing the context in the getForm method the criteria-type object is able to determine what properties will be available for testing later. These values are supplied to the getXMLNode method through the form-scope and so are not needed in this method. The context is necessary however when describing the criteria (just as the names of context-properties are provided to the getForm method) and to fetch the current value of selected context-properties when evaluating the applicability of individual criteria when testing the rule.

Making Sense of It All

To put this all in perspective, what I'm describing is for many of us a radical shift from the world of billing for custom-software to the world of licensing self-serve enterprise applications. An email client is a good example of the need: if you designed a webmail client you might want to provide the ability to automatically filter incoming messages into different folders when mail is downloaded, however you wouldn't want to be responsible for changing the filters for each of your clients. Instead you would create a flexible filtering engine to allow individual clients to select the criteria on which messages are filtered. This is the sole purpose of the RuleManager application: to provide a flexible interface for users to assign the criteria on which other interactions occur within your application. With this in mind I'll address the application flow.

You operate a hosting company and provide email with a built-in webmail interface. A user signs up for hosting and logs in to the webmail application. Within the application they create several folders for their incoming email, then want to assign filters to automatically move incoming email into these folders when messages are downloaded. After selecting the "filters" link they are presented with an empty list page, and a link to add a new filter (index.cfm).

Once the user selects the "add new filter" link they are brought to the edit-filter page (ruleedit.cfm) where they are presented with a simple form allowing them to provide a name and a description for their new filter. Once they've saved this basic rule information they are returned to the same page with an additional form below, containing a list of different types of criteria that they can add to their new rule. This list of criteria is provided by the RuleManager.cfc, which was probably created in the session scope when they logged in (since email rules only apply to the current user).

After selecting a criteria type (Text) and selecting the "Add Criteria" button the user is again returned to the same edit-filter page. At this point a third form is presented to the user below the previous form for general filter information (name, description) and the "add criteria" form. Although the third form is ultimately created by a criteria.text.cfc object the RuleManager.cfc façade simplifies this process by instantiating the desired criteria-type CFC and calling its criteria. getForm() method within RuleManager.getCriteriaForm(). This encapsulates a significant amount of the functionality of determining the location of the criteria-type CFC and caching it in memory so that individual pages can treat the RuleManager and its criteria as a "black box."

The presented form is unique to Text criteria types – you can see this by selecting the Date criteria type in the "add criteria" form. The Text criteria form contains three elements – text to search (Sender, Subject, Message, Recipient), a user-defined

"The objective of a workflow application is not to win an office betting pool, or even to design and complete a superior workflow application"

expression, and a selection to determine how the expression is matched against the text (comparison). Although the expression is entered by the user and the types of comparison are defined by the criteria-type CFC, the list of available text properties that can be matched with a Text criteria are provided by the "email" context (EmailRuleContext.cfc).

When the user selects the "apply criteria" button the form is submitted to the criteria-update template (criteriaupdate.cfm). Like the criteria form, the RuleManager façade also simplifies this update template by providing a singular interface to inject the new (or modified) criteria into the specified rule. This is handled by the RuleManager.setCriteriaNode() method. This method internally determines which criteria-type CFC is needed (stored in memory before displaying the form) and uses the appropriate CFC object to generate the XML for the criteria node, which is then injected into ruleset XML document the RuleManager uses as the format for its rules. The RuleManager object doesn't know where this XML document will be stored (file or database – the sample documents use a file, although in a real application email filters would likely be stored in a database), so the criteria-update template then stores the updated XML document and returns the user to the edit-rule page.

After applying the first criteria for the new rule, the user is presented with the general-information and add-criteria forms and a list of the criteria attached to the current rule (with links to edit or delete each criteria). Again because there is no way of knowing how many types of rule criteria may exist or apply to a given rule, it is necessary for the RuleManager façade to rely on the individual criteria-type CFCs to generate a text description of the applied criteria for the user to read. These text-descriptions of the mechanics of each rule-criteria could be omitted on pain of many sleepless nights with technical support calls, however as most of us enjoy our sleep the RuleManager.cfc again determines the appropriate criteria-type CFCs for each applied criteria and uses their criteria.describe() methods to generate the query returned by the RuleManager.getRuleCriteria() method.

The user can now continue on to apply as many additional criteria as they like to their rules, such as the "or" criteria, which allows them to apply mutually exclusive sets of conditions in which the filter is applied. At this point however the entire workflow of the user-interface is satisfied. All that remains is the application or integration of the rules into the flow of the application.

–continued on page 27

Develop Powerful Web Applications

with ColdFusion MX 7

- Easily build and deploy multi-step data entry forms with CFML tags
- Solve business reporting problems with Integrated Reporting
- Manage your account with HostMySite's Control Panel
- Call 24x7x365 for expert ColdFusion support

Visit hostmysite.com/cfdj for a special offer

HostMySite.com

1-877-215-4678

By Kelly Brown

The crowded auditorium was abuzz with anticipation. The snatches of conversation from around you were meaningless jargon to those not in the know. A hush fell over the crowd as the moment they were waiting for arrived – Ben Forta walking on stage. Okay, he's not rock star, but the closest thing to it in the ColdFusion community.

If you didn't attend the CFUNITED (www.cfunited.com) conference this year (formerly known as CFUN), you should go ahead and put it on your schedule for next year now. CFUNITED has grown from a weekend event with 40 sessions to the premier ColdFusion conference with over 60 sessions and keynote speakers from Macromedia, Microsoft, and New Atlanta. The attendance has also skyrocketed with 887 attendees this year. It's hard to relate the kind of excitement and buzz that occurred at the conference. Where else can you walk down a hall and overhear conversations about the best uses of CFCs, or hang out in the hotel bar until 1 a.m. discussing the benefits of the various ColdFusion frameworks. If you're a ColdFusion developer, this is the conference for you.

The conference actually started earlier in the week with full-day preconference classes given by some of the leaders in the ColdFusion community. Hal Helms taught a class on object-oriented programming using ColdFusion, Simon Horwith taught a class on creating APIs and Design Patterns, and Charlie Arehart taught a class on leveraging .NET in ColdFusion. There were also several other great classes. The preconference continued the next day with MinMax2, in which many of the conference speakers provided 15-minute overviews of their topics or of other topics.

I arrived bright and early the first full day of the conference. I was there as a sponsor and had to make sure our booth and the community center were set up properly. The registration was quick and easy and I received my conference goody bag. The goody bag was really…well, good. It included a very nice CFUNITED backpack, a conference book with the slides for

CFunited is still Fun

It's bigger and better

the sessions (which I found useful for taking notes in during the sessions), a technical book, and various handouts from the conference sponsors. The technical book was on ASP.NET, which is a point of contention for many conference goers since it is a ColdFusion conference. However, I can't really fault Microsoft for pushing their products and, despite the many wonders of ColdFusion, it's good to know more than one development language. (No flames please.)

I had time before the morning keynote to explore the sponsor area. Several companies had a strong presence. Of course, Macromedia was well represented at the conference and I was surprised to see they had sent their entire ColdFusion development team. This is a good sign that Macromedia recognizes the growing importance of this conference in the ColdFusion community. Microsoft had a large booth; of course, they spent the whole conference pushing the .NET and Visual Team system. Next to Microsoft was New Atlanta, maker of the Blue Dragon ColdFusion server. It may have just been a coincidence, but I thought it was interesting that Microsoft and the company that makes a version of a ColdFusion server that runs natively in .NET were next to each other.

There were many other companies represented and I can't list them all, but I thought I'd mention a few that I found interesting. CFDynamics and HostMySite do ColdFusion hosting and I've heard good things about both. Interakt was promoting their Dreamweaver plug-in, which has some nice features including a much nicer query builder and code templates. PaperThin and Savvy Software both have nice content management systems, though PaperThin is targeting a higher-end market with a feature-rich product. SeeFusion had a very cool tool that allows you to remotely monitor your ColdFusion pages in real time and see which pages are running, the run time for each query, and other statistical data. It also has the ability to kill a page remotely in case you see a page that is taking too long to run.

Ben Forta and Tim Buntel gave the opening keynote. They discussed how ColdFusion 7 was doing in the marketplace (quite well for those of you who were not in attendance) and some of the lesser-known features of CF7, and talked about the upcoming update that will provide better support for additional platforms, especially Mac OS X. The big announcement at the keynote was that Macromedia would be supporting the CFEclipse project, a competing IDE to Dreamweaver.

The opening keynote was followed by sessions, lots of sessions. There were so many really good sessions it was hard to

choose which one to go to, as there was usually another one I was interested in at the same time. I'll mention a few I liked here, but they were all good. My favorite sessions were: all the CCS sessions by Sandy Clark, Simon Horwith's "Design Patterns," Robi Sen's "Advanced Scaling and Tuning," and Hal Helms "Creating a Domain Model."

Unfortunately, I didn't get to go to as many sessions as I wanted because I had to spend time at the booth and the community center. Not that it was bad. The community center turned out quite well, though my opinion may be biased. About Web ran several contests for the community area. There was a ColdFusion trivia game with new questions every hour. "Shoot the Guru" in which you used a Nerf gun to shoot targets of famous ColdFusion gurus, such as Simon Horwith and Hal Helms. There was also an X-box, provided by House of Fusion. Of course, all of the contest winners got to spin the "Wheel of Fusion" to win fabulous prizes. Well, some of the prizes were not that fabulous, but thanks to Macromedia we had copies of Dreamweaver, ColdFusion 7 Standard, and ColdFusion 7 Enterprise to give away. I also want to thank CFDyanmics, Interakt, HostMySite, and House of Fusion for providing prizes.

The keynote speaker at lunch was Joel Spolsky, author of "Joel on Software." Joel is a very entertaining speaker. The keynote wasn't really ColdFusion related but it made it you laugh and, in many cases, made you think about what makes good software design. I was also fortunate enough to be able to attend his session on software management. It wasn't so much a methodology, rather he presented some of the things you should be doing. Of course, Joel had his typical humorous real-work examples, which are all the more funny because you know they're true.

Thursday's keynote was from Bill Staples of Microsoft on "IIS7: The Future of Microsoft's Web Server Platform?" There are some interesting things to look forward to. Web configuration is going to be much more flexible and portable with configuration being stored in XML that can be moved from server to server. It also provides more control of the components of the Web service, such as the ability to plug in your own authentication instead of using the OS. It will also provide better metrics so you can tell what is going on in the Web server.

There was a birthday party for ColdFusion (10 years old) Thursday night on the hotel patio with a mammoth birthday cake as well as drinks and games, courtesy of TeraTech and HostMySite. There was also a band called Boxcar Collision and a video message from ColdFusion creator Jeremy Allaire.

Friday's keynote was presented by Peter Amiri of MySpace and Charlie Arehart of New Atlanta. MySpace is one of the most trafficked sites on the Web, coming in at number five just under eBay. A relatively new site, it's only 22 months old and already has over 20 million members. Some of the more distinguished readers may not be familiar with the site, as it's an online community with a target demographic of 16–24 year olds. They are in the process of converting to New Atlanta's Blue Dragon CFML compiler and got a 50% performance increase from the port. It's a good example to show people when they ask about the scalability of ColdFusion; however, the presentation did generate some controversy. During the presentation Peter said that the site had architectural issues and they were just throwing hardware at the problem. They have poor practices and do several code releases a day directly into production with the excuse that they have a "forgiving audience" that wants more features and doesn't care about the bugs.

All in all the conference was great. TeraTech Programming did an awesome job of organizing the conference and keeping everything running smoothly – kudos to Liz Frederick, the conference coordinator, and Michael Smith, who created the event. The Montgomery County Conference center was great and the Marriott staff was very helpful, especially with setting up the sponsor area with the booths and the community area.

For those of you who couldn't make it, you should definitely try to attend next year when it will be even bigger and better. As Ben Forta says on his blog, "[CFUNITED] has become the premier CF specific event." I hope to see you there at CFUNITED-06 in June 2006!

About the Author

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

kbrown@aboutweb.com

Playing With(in) the Rules

–continued from page 23

In the sample case a webmail application would contain a template (or CFC method or custom tag) for downloading email from the server. In order to filter the incoming email messages into the appropriate folders, the download process must loop over the downloaded messages and for each message must loop over and test each of the user's email filter rules. Again the RuleManager façade simplifies the process by providing a single method, RuleManager.ruleApplies(ruleid [, context]), which loops over the criteria for an individual rule and tests the applicability of each criterion to provide a boolean value which may be used by the application to determine if the message is filtered.

In order for the ruleApplies method to function properly the context CFC must provide the actual property values to the criteria-type CFCs for each downloaded email. Thus when the criteria-type CFC calls the method context.getFromHeader() the context CFC must return the "Sender" string value "s. isaac dealey <info@turnkey.to>" for any message sent by myself. This can be accomplished either by instantiating a CFC for each message and passing that CFC as an optional second argument to the RuleManager.ruleApplies() method, or a single context CFC may use an iterator property to indicate the current row of a query from which email information is retrieved. Although for this application I would recommend the latter approach for the sake of efficiency, individual context CFCs may be more appropriate for other applications.

Unfortunately the scope of this article doesn't allow me the luxury of providing a complete sample application. The sample code (which is available by viewing this article online) and the above paragraphs lack an explanation of how the folder for each filter is identified. There are several methods that could be applied, including an additional attribute in the XML node for each rule or additional structure in your database. You will need to address these details case-by-case in your applications.

Conclusion

Although I lament that this article could not examine many of the more useful nuances of the RuleManager façade it should arm you with enough information to start providing powerful self-service customization to your clients today. The previous paragraphs and code samples provide information about basic concepts, storage, formatting, user-interface and application in business-logic, as well as thorough explanations of all the necessary CFCs and methods and a step-by-step explanation of the flow of user-interface and application logic. For questions or to see a more thorough example, you may contact me directly or view the onTap framework documentation at www.fusiontap.com.

About the Author

Isaac Dealey has worked with ColdFusion since 1997 (version 3) for clients from small businesses to large enterprises, including MCI and AT&T Wireless. He evangelizes ColdFusion as a volunteer member of Team Macromedia, is working toward becoming a technical instructor, and is available for speaking engagements.

info@turnkey.to

What's your PDF?

✓ Precise Document Formatting

With activePDF Server , you gain full control over your PDF output with conversion options that allow you to specify page size, compression and resolution options, embed text, create bookmarks, concatenate to existing files, and more. Licensed per server, you can easily add PDF generation to virtually any Windows application.

✓ Populate Dynamic Forms

With activePDF Toolkit 's form-filling capabilities, you can dynamically populate PDF forms with data and images from a database, allow users using only Adobe Reader to fill-in and save forms and use PDF forms as document templates to precisely control image placement and resizing. With Toolkit's robust API, the automation of virtually any PDF manipulation task becomes possible - append, stamp, stitch, merge, paint, secure PDF and more.

✓ Promote Digital Fidelity

Do you need to standardize PDF output within your enterprise? With DocConverter, you can easily use built-in support for "watched" folders to implement server-side PDF generation in a matter of minutes, with full control over the PDF output at the server level. Or, use DocConverter's programmable COM object to integrate convert-to-PDF functionality within your enterprise application.

✓ Present Data Fashionably

Ensuring precise layout of an HTML document can be a nightmare, especially when printing. PDF guarantees pixel-perfect layout every time as what you see is what you print. With activePDF WebGrabber, you can dynamically convert any URL, HTML stream, or HTML file to PDF on the fly, while maintaining embedded styles.

Download your free trial version today at www.activePDF.com

activePDF® Leading the iPaper Revolution

Observed Benefits of the Unified Modeling Language

Using the UML with your current methodology

By Duncan Jack

Over the past 12 months, I have observed significant benefits using the Unified Modeling Language (UML) when developing Rich Internet Applications using Macromedia technology.

This article first discusses what the UML is, then lists some of the main diagram types. It highlights how these diagrams can be used and draws attention to some of the benefits I've observed when using them. It concludes with a list of resources.

What Is the Unified Modeling Language

To understand the essence of the UML, consider the elements of its name:

- ***Unified:*** The result of unifying three leading approaches to system modeling in the 1990s
- ***Modeling:*** Concerned with the simplified representation of system structure and behavior
- ***Language:*** A language, not a methodology

The UML provides a language-neutral, tool-supported, well-documented standard for modeling systems such as Web applications. It enables system requirements, structure, and behavior to be succinctly captured and effectively communicated.

At the time of writing this article, the UML 2.0 Specification was going through final editing, although you'll find that many books and tools support at least a subset of this specification. A draft version of the specification is available on the Object Management Group's UML Web site. Helpful note – don't try and learn the UML from this document, but it can make interesting reading!

The UML is not a methodology. This point is important. Some people think that you have to use every diagram type to model every aspect of system behavior all the time as part of a complex, cumbersome approach. Not at all. Simply make intelligent choices about what works for you. The UML is designed to serve you, not the other way around.

To illustrate this point, consider the ColdFusion Markup Language (CFML). CFML is a language, not a methodology. To derive full benefit from your use of CFML, adopt an effective methodology. You may adapt your approach on different projects. Use a subset of the CFML to build an application. Don't try and use every tag and every function in every application you build.

In the same way, blend the UML into the successful methodology you already use.

Main Diagram Types

Essentially, when you use the UML, draw diagrams and add notation to them. You may draw a UML diagram by hand on the back of a menu over lunch with a client or on a whiteboard. Equally, you may use a tool such as MagicDraw UML.

There are two main categories of UML diagrams defined by the UML 2.0 Specification:

- ***Structure Diagrams (six):*** Concerned with modeling static structure (architecture)
- ***Behavior Diagrams (seven):*** Concerned with modeling dynamic behavior

I have found the following diagrams most useful since I began using the UML 12 months ago:

- Use Case Diagram (behavior)
- Activity Diagram (behavior)
- Class Diagram (structure)
- Sequence Diagram (behavior)

Using these four diagrams in sequence has been very effective, so I'll address each in turn.

Use Case Diagram

Use Case Diagrams help to define the requirements of a system from the users' perspective – what they want to achieve when using the system.

It's deceptively simple yet incredibly powerful. Notes are

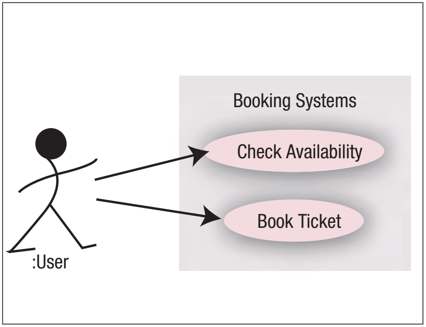


Figure 1: Use Case Diagram

added to the diagram, and may, of course, be supplemented by other documents where appropriate. This is exactly what architects and engineers in other disciplines do too, of course – use blueprints and drawings.

The user may be a human object or software object (if you are exposing a ColdFusion Component as a Web service for example). The basic syntax is very simple.

As you can see from Figure 1, the system is required to let a user check availability and book a ticket. Notice that the diagram doesn't go into the detail of how this will be accomplished. It helps them focus on desired outcomes and not the process. For me, that's the power of Use Case Diagrams – focusing minds and drawing out detail. Of course, it's important to remember that clients may:

- Not know exactly what they want or need
- Be reporting to a boss who has given them unclear, incorrect, and incomplete requirements
- Be part of a wider team amongst which requirements are fragmented
- Forget or contradict their own requirements

I have noticed a number of business benefits when using Use Case Diagrams. It's the simplicity of the diagram and the practice of going through the process with a client that really pays off. I've noticed that these diagrams help to:

- Discover what clients actually want and need
- Draw in other stakeholders (the boss, co-workers, etc.) to the requirements gathering process on an ongoing basis
- Identify any incorrect and contradiction in requirements

I was recently involved in a project to build a Rich Internet Application for a business run by three extremely capable female directors in their 50s. They found the Use Case Diagram indispensable. At every meeting, the first thing we would do was review it to confirm that all requirements had been captured and were fully up to date.

As additional requirements were identified, these were either added into the current version or added to a list for future discussion. Either way, it was up to the clients to decide. The Use Case Diagram was a living, breathing document that provided an ideal way to ensure that the interface with the clients remained cohesive.

Letting them each have a copy of the diagram that they could mark up and use in their own internal meetings proved to be a very effective way to draw everybody in and ensure we built the right system.

We became a natural extension of their business; they became a natural extension of our project team. As a result, meetings were more productive, a better application was delivered more quickly, and business was stored up for the future. In addition, our approach helped us to differentiate ourselves from our competition and ensure a strong ongoing relationship with the clients.

Activity Diagram

Once the requirements of a system from the users' perspective have been defined, Activity Diagrams help to define how this user experience will be achieved.

Activity Diagrams are also extremely powerful. They're well suited to fleshing out the details of a use case by modeling the detailed interaction between a user and a system or screen. Activity Diagrams are used to model:

- Business processes
- Flow of control in an executing program
- Details of a method

They are a close relative of the traditional flowchart (see Figure 2).

As you identify and diagram the different activities, you'll naturally see a pattern of objects emerge to which the different activities can be assigned. You

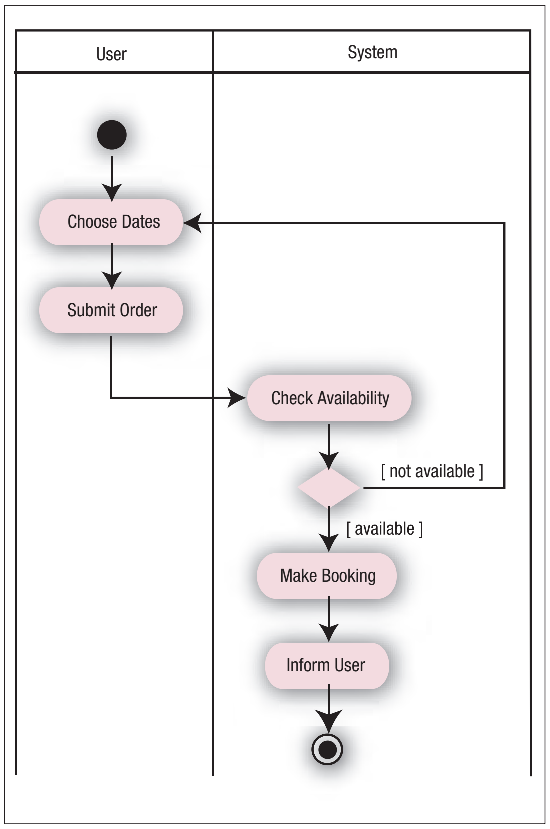


Figure 2: Activity Diagram

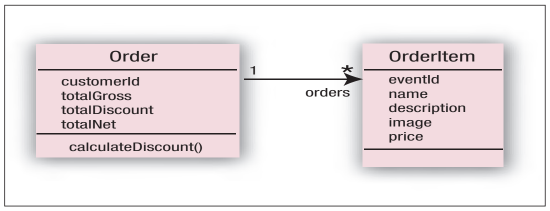


Figure 3: Class Diagram

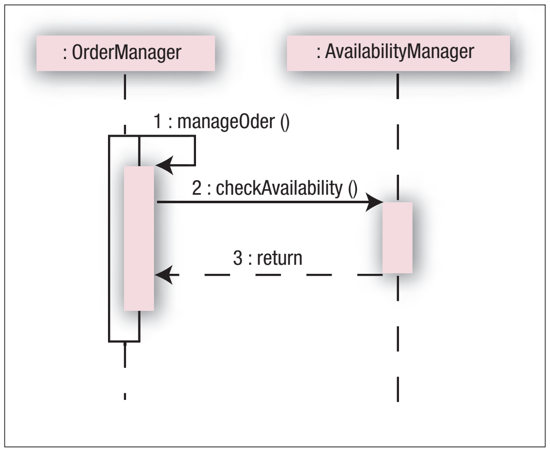


Figure 4 : Sequence Diagram

can use the swim lanes to assign responsibilities to different objects – whether those objects are people or software.

Class Diagram

Class Diagrams are used to model the classes of objects in a system (people and software). In the context of this article, the software building blocks are likely to be ColdFusion Components (CFCs) and Java classes.

Think of a CFC – it has properties and methods and can be represented as shown in Figure 3.

The Order.cfc has an orders property, which is an array of order items, each one represented by an instance of OrderItem.cfc. Although this could simply have been shown as an attribute inside the class, it's often more meaningful to represent such a property using an association as above.

On this point, I found Martin Fowler's excellent book, *UML Distilled*, particularly helpful. I highly recommend it. He goes into Class Diagrams in some detail and wisely splits his coverage into two chapters, focusing the first chapter on the essentials.

Class Diagrams seem to follow so naturally from Activity Diagrams; the activities identified often may neatly correspond to methods in a Class diagram, which helps save time and increase productivity.

There is, of course, no requirement to identify every property or method on a class in a class diagram. You may choose to show only public methods, for example. Equally, you don't have to show all classes and relationships between them. Again, use what works for you.

Class Diagrams really help when architecting the system and seem to give the design "room to breathe." It seems that if the design is elegant, the implementation is elegant too. If the implementation is elegant, it can be more pleasurable and cost-effective to evolve and maintain on an ongoing basis once the system has been put into production.

In Ian Sommerville's definitive work, *Software Engineering*, he cites (and qualifies) research that suggests up to 90% of software costs are evolution costs. Looking at this another way, if all you build for a client is the initial implementation of a system, you may only be getting as little as 10% of the revenue stream that you would otherwise get from that client over the lifetime of the system. Of course, these figures will vary significantly, but it's an interesting thought.

If you build a system that can be evolved elegantly and cost effectively, you are more likely to keep the relationship with the client, give them a better service, and make more money. Class Diagrams are great for organizing where functionality will go, and for helping to select consistent and meaningful property and method names.

Sequence Diagram

Have you ever developed an application and then had to come back and modify it six months later and tried to work out how on earth you did it? Well, the UML Sequence Diagram may be able to help you.

A Sequence Diagram models the sequence of interactions between objects. In some ways, it's a close cousin of an Activity Diagram, yet more focused on the behavior of software objects on a timeline (see Figure 4).

Sequence Diagrams are great for thinking through a design, illustrating an idea, and also getting back up to speed when changes need to be made six months or so after the system has gone into production. The design stands out so clearly.

Prior to finding out about the UML, I used my own non-standard diagrams. For me, the biggest single benefit of the UML has been the Sequence Diagram.

In the UML modeling tool I use, MagicDraw UML, I typically have a Class

Diagram open at the same time as a Sequence Diagram. As I work on the design and identify additional methods, I add these to the appropriate class. These methods are then immediately available for me in the Sequence Diagram. As a result, it's much easier to create an elegant design and enhance productivity.

It also ensures that the design is the documentation, which ensures that the documentation is done as the design evolves, changing with it. The appropriate use of annotated UML diagrams can save time, which is a significant business benefit.

In addition, it becomes a pleasure to come back and add additional functionality at a later date.

Getting Started with the UML

Here's what worked for me and what I generally suggest to anyone interested in getting started:

- Get a tool such as MagicDraw UML (free community edition and trial available). Tools have a lot of intelligence built in, which helps you get up to speed quickly on a couple of diagram types. The tool knows the specification. Have a look at the various symbols available in one or two of the main diagram types described above. Start using some of the symbols; you don't have to use them all. Grow into the tool over time. Learn the structure of the documentation and start reading it.
- Get Martin Fowler's book, UML Distilled. Read it a couple of times. Great book, aptly titled. I found Martin's real-world experience and balanced view of using the UML very helpful.

Remember that the UML is a language, not a methodology, so don't think you have to change everything you already do successfully in order to get started with the UML. Take it one diagram at a time. Used effectively, the UML offers significant benefits.

Resources

- *MagicDraw UML (includes comprehensive documentation and examples*): www.magicdraw.com
- Fowler, M. (1999). *UML Distilled*. Addison-Wesley: www.pearsoned.co.uk/Bookshop/
- Sommerville, I. (2004). Software Engineering 7. Addison Wesley: www.pearsoned.co.uk/Bookshop/
- *UML 2.0 Specification and useful links:* www.uml.org.

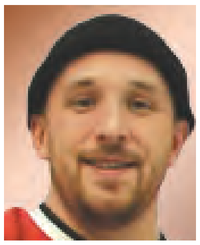
About the Author

Duncan Jack started the Scottish CFUG (www.scottishcfug.com), which is now ably run by Andy Allan. Duncan recently founded Scottish Java (www.scottishjava.com), a brand new Java community. A Macromedia Certified Flash MX 2004 Developer, his main interests are in building innovative Rich Internet Applications using Flash, Flex, ColdFusion and JRun. An accomplished mountaineer, he holds a first-class honours degree in Civil Engineering and is currently studying for an M.Sc. in Advanced Computer Science.

duncan@scottishjava.com

XML and cfcPowerTools

Using CFCs



By Tom Schreck

CfcPowerTools is a web GUI for generating Cold Fusion Components (CFCs). I started working with CFCs when ColdFusion MX 6 was released. Like many developers, I was intrigued as to what CFCs are and how I can use them to be a better, more productive developer. I played with CFCs following what documentation Google found for me and realized the potential CFCs offer.

I found some “best practices” documentation that suggested instance data (data that’s part of the CFC after it has been instantiated) should be protected from public access. The documentation recommended the use of getter and setter methods for presenting a controlled interface to instance data. Sounds like a great idea. Who doesn’t want protected data? Anyways, after a couple months of finger boning getter and setters (and fixing the subsequent errors) I wanted something better. Like any good developer, I’m lazy. So, I spent the better part of two years creating cfcPowerTools so I do not have to write getter/setter methods by hand.

cfcPowerTools has grown out of a desire to have getter/setters created for me. It has grown into an application that will:

- generate a CFC from a database table
- generate a database table from a CFC
- batch generate CFCs from multiple tables
- generate data entry form
- generate Data Access Object (DAO) CFC
- generate XML document containing all property Metadata
- generate getter/setter methods

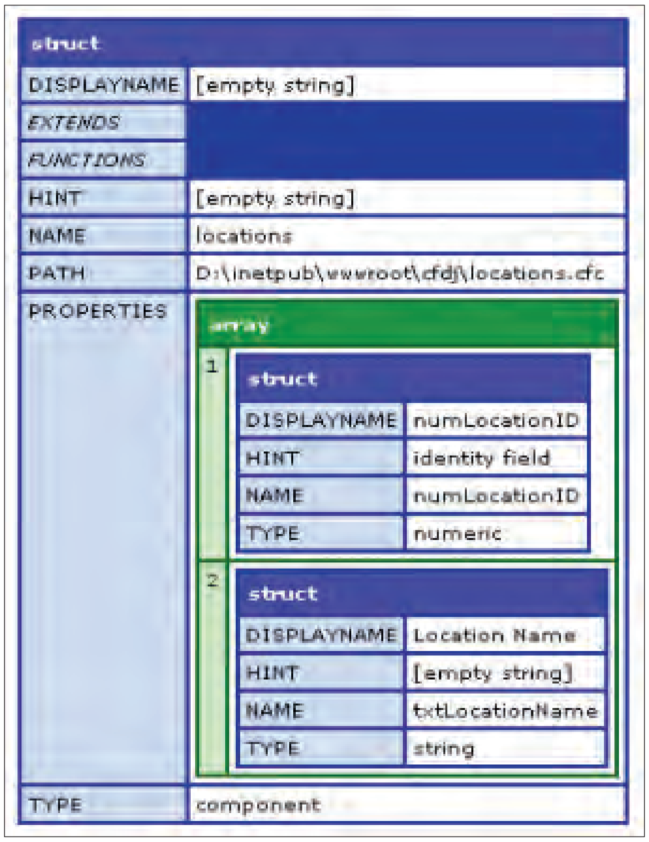
My first incarnation of cfcPowerTools put all of the metadata collected for a property into the <cfproperty> tag. A version 1 <cfproperty> looked like this:

```
<cfproperty elementmaxlength="255" hint="email from" position="1"
txtDefaultValueType="Simple Value" type="string" elementsize="35"
name="From" propertyform="textbox" datatype="varchar" displayName="From"
required="1" propertydisplaylevel="1">
```

This is not very user-friendly. A developer would not do this if he or she were writing the CFC by hand. I wanted to generate a CFC file that you could not tell if it was created by hand or by cfcPowerTools. So, I needed a better way to manage a property’s metadata.

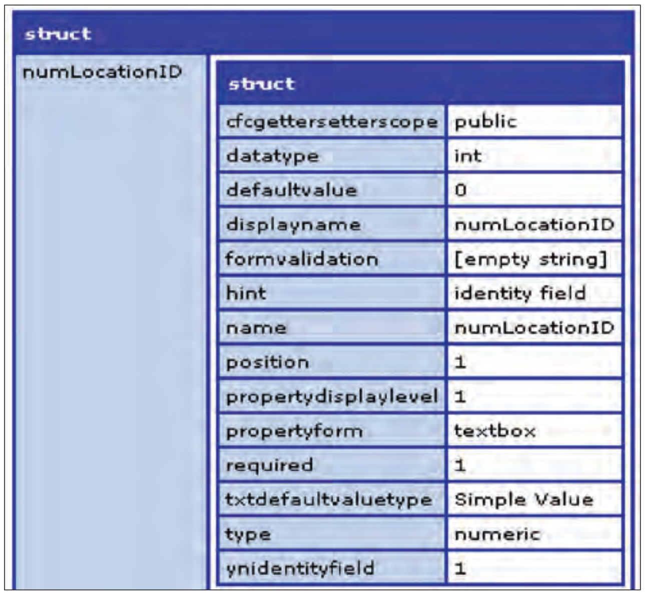
XML to the Rescue

XML seemed to be the best choice for storing the property metadata used to describe the property from a database and



```
struct
DISPLAYNAME   [empty string]
EXTENDS
FUNCTIONS
HINT          [empty string]
NAME          locations
PATH          D:\inetpub\wwwroot\cfdj\locations.cfc
PROPERTIES    array
              1  struct
                 DISPLAYNAME  numLocationID
                 HINT         identity field
                 NAME         numLocationID
                 TYPE         numeric
              2  struct
                 DISPLAYNAME  Location Name
                 HINT         [empty string]
                 NAME         txtLocationName
                 TYPE         string
TYPE          component
```

Figure 1: ColdFusion’s Property Metadata



struct	
numLocationID	struct

struct	
cfcgettersetterscope	public
datatype	int
defaultvalue	0
displayname	numLocationID
formvalidation	[empty string]
hint	identity field
name	numLocationID
position	1
propertydisplaylevel	1
propertyform	textbox
required	1
txtdefaultvaluetype	Simple Value
type	numeric
ynidentityfield	1

Figure 2: Merged Property Metadata

form's perspective (see Listing 1). Plus it allowed me to generate a <cfproperty> tag like this:

```
<cfproperty displayname="Article Name"
name="txtArticleName" type="string">
```

How Does It Work?

You describe your CFC and its properties when you create a CFC with cfcPowerTools. This metadata is collected and stored in an XML document. You describe how your properties are viewed by a database table and how they are viewed by a data entry form. Each CFC has an XML document. cfcPowerTools consumes the XML and uses the information in everything cfcPowerTools does from viewing a CFC's anatomy to generating DAO, data entry form, and database table. The CFC generated by cfcPowerTools looks like a typical CFC you might create by hand. cfcPowerTools does not introduce any of the metadata into the physical CFC file (see Listing 2).

What Benefits Does an XML Document Have?

Having the metadata in an XML document has several advantages:

- It uses an industry standard in XML. My first incarnation had metadata inside the <cfproperty> tag, which turned out to be cumbersome and awkward.
- ColdFusion MX works seamlessly with XML.
- It facilitated overwriting inherited property's metadata. This was a huge benefit. I can now extend a CFC and overwrite the parent's property's metadata.
- Manual modifications of CFC property metadata. You can change the metadata without getting into cfcPowerTools.

How Does cfcPowerTools Consume the XML Metadata?

I was challenged with figuring out how to merge my metadata with the metadata ColdFusion has on a CFC. This turned out to be surprisingly easy. ColdFusion has a getmetadata() method that introspects a CFC and produces a structure containing a CFC's metadata (see Figure 1). All I had to do is loop over each property in ColdFusion's metadata structure and append the metadata from the XML document.

The included code example shows how I am pulling data out of an XML document and merge with ColdFusion's CFC metadata. The example files are as follows:

- ***locations.cfc*** – an example CFC generated by cfcPowerTools
- ***locations.xml*** – an example XML document with property metadata
- ***mergeMetaData.cfc*** – example code for how I merge locations.xml metadata with ColdFusion's metadata
- ***test.cfm*** – page displays the results of the merged data plus the original property metadata from getMetaData()

mergeMetaData.cfc

mergemetaData.cfc is a watered down example of what cfcPowerTools does. Basically, I pass in the name of the CFC file (locations) and the first thing it does is instantiate the CFC file and retrieve ColdFusion's metadata (see lines 28-31).

```
//if stMetaData is not known, then getMeta-
Data()
if(structcount(arguments.stMetaData) EQ 0){
    oCFC = createobject("component",arguments.
cfcName);
    arguments.stMetaData = getmetadata(oCFC);
}
```

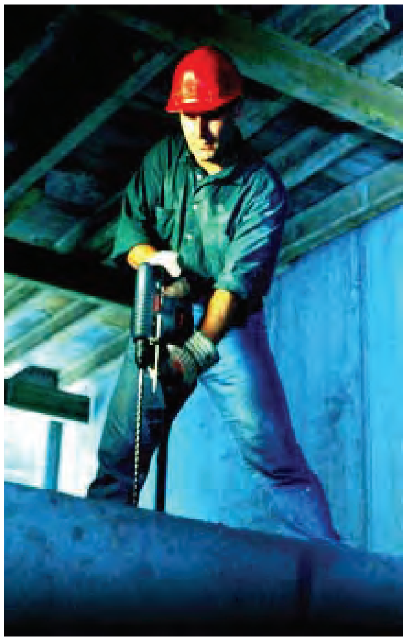
Lines 34-45 handle looping over any properties in ColdFusion's metadata, if

"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.



any exist. For each property, I make a call to getXMLProperty(), which pulls the property node out of the XML document. Then I merge the results in with ColdFusion's metadata.

```
//loop over each property in meta data structure
for(z=1; z lte arraylen(arguments.stMetaData.properties); z=z+1){
    propertyName = arguments.stMetaData.properties[z].name;

//retrieve property meta data from XML document
    stPropertyDetail = getXMLProperty(property=propertyName);

//append each XML Meta Data to CFC property
    for(x in stPropertyDetail){
arguments.stAllProperties[propertyName][x] = duplicate(stPropertyDetail[x]);
    }
}
```

Lines 60-89 shows how I read the XML file in getXMLProperty () and search for the property using XPath query and how I return a structure containing the property's attributes:

```
<cfscript>
//parse XML file into a CF XML Object
xmlMetaData = xmlParse(strXML);
//search for passed property in XML Document
arProperty = xmlSearch(xmlMetaData,"//properties/property[@name='#arguments.property#']");

//retrieve property attributes
stAttribs = arProperty[1].xmlAttributes;
return stAttribs;
</cfscript>
```

The net result is a merge of property metadata stored in an XML document with ColdFusion's metadata (see Figure 2). Once I have the merge complete, I can use this data to generate all of the cool functionality provided by cfcPowerTools.

About the Author

Tom Schreck is a Macromedia certified ColdFusion MX 6.1 developer. He has been working with ColdFusion since 1997. Check out www.cfcPowerTools.com for more information on cfcPowerTools.

tkschreck@sbcglobal.net

Listing 1: Sample Property Metadata XML

```
<?xml version="1.0"?>
<xmlcfc>
    <metadata cfcdatascope="variables" cfcdbtype="MsSQLServer" cfcdsn="byron"
cfcidentitytype="int" cfcname="locations" cfcpackage="practice.components.abstract"
cfcpath="D:/JRun4/servers/practice/cfusion.ear/cfusion.war/practice/components/abstract/"
cfctablename="tblLocations" cfctype="cfcDAO" displayname="" extends="" hint="" identityfield="
numLocationID" yncfcfromtable="0" ynhasdao="1" ynhastable="1"/>
    <properties>
            <property cfcgettersetterscope="public" datatype="int" defaultvalue="0" displayna
me="numLocationID" formvalidation="" hint="identity field" name="numLocationID" position="1"
propertydisplaylevel="1" propertyform="textbox" required="1" txtdefaultvaluetype="Simple
Value" type="numeric" ynidentityfield="1"/>
            <property cfcgettersetterscope="public" datatype="varchar" defaultvalue=""
displayname="Location Name" elementmaxlength="50" elementsize="35" formvalidation="validateNo
tNull" hint="" implementedin="practice.components.abstract.locations" name="txtLocationName"
position="2" propertydisplaylevel="1" propertyform="textbox" required="0" txtdefaultvaluetype
="Simple Value" type="string" ynnewproperty="1"/>
    </properties>
</xmlcfc>
```

Listing 2: Sample <cfproperty> Tags

```
<cfcomponent displayname="" hint="">

    <cfproperty displayname="numLocationID" hint="identity field" name="numLocationID"
type="numeric">
    <cfproperty displayname="Location Name" hint="" name="txtLocationName" type="string">

   <!section 1 -->
   <cfloop index="x" list="#form.fieldnames#">
      <cfif #
```

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

The XPath Factor

XML and XPath

By Nik Molnar

It's 3:00 P.M. on a sunny Saturday afternoon. The birds are chirping, the leaves are blowing, and you can hear the lake waters breaking on its rocky shores. The sounds of a baseball game randomly crack in the distance, and the roar of competition erupts on the basketball courts nearby.

My burgers are just getting medium-well as my wife is returning from a little potty walk with the dog. Our blanket is set and our picnic looks like it will be wonderful. Who knew a Web geek like me could pull off such a seemingly perfect day? Interestingly enough, I owe much of its success to XML and XPath.

XPath is to XML what SQL is to databases. Databases would be quite pointless if you could not query information out of them, and the same holds true for XML documents. XPath is a language for finding information in XML documents. XPath provides access to all of the elements and attributes of an XML document. It became a World Wide Web Consortium (W3C) recommendation on November 16, 1999 and since then it has become a huge part of the XML world. It is a major element of the W3C's XSLT standard, and both XQuery and XPointer are built upon XPath expressions.

An expression is XPath's primary construct, a string that, in its basic form, resembles a file path. True XPath engines examine the expression and return a node-set (more on nodes later), Boolean, number, or string from the XML document. ColdFusion's implementation of XPath only supports the node-set return type. For more complete XPath support you can use one of the many Java libraries available on the Internet.

XML has reared its head all over the Web, and with XPath being such a cornerstone of the XML world, a solid understanding of it is crucial. XML can be found any place where you need to provide or acquire data from a third-party vendor, client, application, or server. Web services, WDDX packets, and RSS feeds are all implemented on XML technology. XPath could be used to integrate various combinations of any of these technologies. A good example of this type of integration was the weekend planner application I used to find the perfect Saturday activity.

The application reads an RSS feed of a local community events calendar. It leverages XPath to read the zip codes and dates of each event. These zip codes and dates are then sent to a Web service that provides the weather forecast for that area of town, on that day. That information showed me that a picnic at Blanchard Park on Saturday was a better idea than going to the carnival at the Central Florida Fair Grounds on Sunday.

XPath can be used anytime you need a subset of data from a larger XML dataset. Its use is analogous to the use of Regular Expressions. Regular Expressions retrieve a substring based on pattern matching within a string. They are very powerful, although one might not think so at first look. This is because ColdFusion implements Regular Expressions as a single argument used in a small handful of functions. A deeper look into Regular Expressions would reveal book after book dedicated to its intricacies. In the same manner, XPath makes its ColdFusion appearance as a lone argument of one function, yet has novels dedicated to its deeper functionality.

XPath is leveraged via the function XmlSearch(). XmlSearch() requires two parameters: xmlDoc; an XML Object, and XPathString; and XPath expression. It returns an array, with each element of the array containing an XML node. As stated before, XmlSearch() cannot return strings, Booleans, or numbers.

The first parameter of XmlSearch() is an XML Object. When XML is read in by ColdFusion it is treated just like any other string, however calling the function XmlParse() on that string will load it into memory, create, and return a ColdFusion XML

Object. Figure 1 shows the comparison of an unparsed XML document dumped on the left, and the same XML document parsed and dumped on the right.

Now that our XML has been converted to an object, we can run our first XPath expression against it. In order to understand how to write XPath expressions, we must understand and identify the different parts and elements that make up an XML document.

The easiest way to identify the parts of an XML document is to compare it to a file structure. If we were to run the command "C:\inetpub\wwwroot" in Windows, the OS would open the "wwwroot" directory, showing all of its contents. In this example, "C:" of course is a reference to the disk, and "inetpub\wwwroot" tells the system to look into the "wwwroot" folder, which is nested inside the "inetpub" folder. As nomenclature of nested sets goes, it could be said that "inetpub" is the parent "wwwroot." Likewise, "wwwroot" is the child of "inetpub." Extending that paradigm out, "C:" would then be a grand parent of "wwwroot." In such instances, "C:" would be known as an ancestor and "wwwroot" would be known as a descendant. Since "C:" is the absolute oldest ancestor or the top node in the nested set, it is also given a special name – root.

If we were to change our command to "C:/inetpub" and run it, the "inetpub" folder would open showing all of its children. Besides "wwwroot," there might be other directories or files. For example, I have an "ftproot" folder nested below "inetpub." Since "inetpub" is a parent to "ftproot" as well as to "wwwroot," "ftproot" and "wwwroot" are said to be siblings.

XML nodes can also be identified using the same rules as the "family-name-game" above. Listing 1 is a breakdown of every battle from Iron Chef America: The Series (Season 1). In it, the "ICA" node is equivalent to "C:." It is the root node (also known as the document node in XML), an ancestor to the "IronChef" node, and parent to the "Battle" node. Since the "IronChef" and "Battle" nodes share the parent node "Battle," they are siblings. In XPath, just as in our file path example above, we can reference the "IronChef" node using a simple path syntax: "/ICA/Battle/IronChef."

This expression, when run against our XML file, will return a 10-element array, one element for each Iron Chef in each battle. The expression is very simple to follow. The "/" at the beginning of the string tells the XPath engine to start at the root node. Each "/*node*" combination after the leading "/" tells the XPath engine which set of nested nodes to select and finally return.

You can also start an XPath expression with a double slash "//." A double slash tells the XPath engine to search the entire XML document for the specified node. For example:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//IronChef")>
```

will return the same array as:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "/ICA/Battle/Iron-
Chef")>
```

because both expressions are selecting and returning the "IronChef" node.

This XPath shortcut can save the developer a lot of time when parsing through lengthy or deeply nested XML documents. However, since the path is not explicitly listed, it can lead to problems. "/ICA/Battle/IronChef/Score" will return an array with 10 elements, while the expression "//Score," on the other hand, will return an array with 20 elements. This is because the "//Score" expression does not differentiate between the challenger and Iron Chef's scores, thus returning all nodes named "Score."

Besides the nodes that are being returned, the attributes of those nodes are also returned when using XPath. Referencing attributes in XPath is very similar to referencing nodes, with one minor difference. XPath uses the "*@attribute*" syntax to reference node attributes. All slash "/" and double slash "//" rules apply to the attributes of nodes the same way they apply to nodes themselves.

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject,"//@food")>
```

The expression "//@food" will return the types of food that both the Iron Chefs and their challengers prepare, while

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject,"/ICA/Battle/IronChef/
@food")>
```

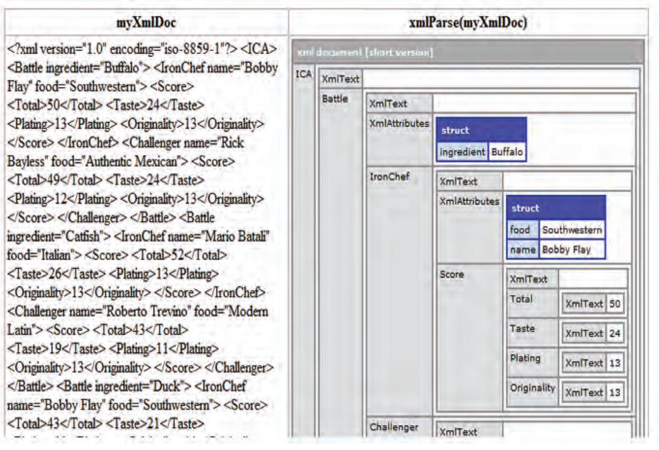
will return only the Iron Chef's culinary style.

The "/", "//," and "@" symbols can be seen as the syntax used in XPath to define the FROM clause commonly used in SQL statements. In order to filter out nodes you will need XPath's equivalent to an SQL WHERE clause. That functionality is realized through XPath's rich set of predicates, functions, and operators.

XPath predicates use a bracket notation similar to the notation used to reference individual elements of an array.

```
<cfset XPathResult = xmlSearch(ironChefXMLObject, "//Battle[1]")>
```

As you may have guessed, the expression "//Battle[1]" will return only the first "Battle" node. Likewise "//Battle[2]" will return only the second "Battle" node and so forth. Commonly with arrays you can use the arrayLen() function to find out what the last element's position will be. In XPath you can use the function last() to achieve similar results.



myXmlDoc

```
<?xml version="1.0" encoding="iso-8859-1"?> <ICA>
<Battle ingredient="Buffalo"> <IronChef name="Bobby
Flay" food="Southwestern"> <Score>
<Total>50</Total> <Taste>24</Taste>
<Plating>13</Plating> <Originality>13</Originality>
</Score> </IronChef> <Challenger name="Rick
Bayless" food="Authentic Mexican"> <Score>
<Total>49</Total> <Taste>24</Taste>
<Plating>12</Plating> <Originality>13</Originality>
</Score> </Challenger> </Battle> <Battle
ingredient="Catfish"> <IronChef name="Mario Batali"
food="Italian"> <Score> <Total>52</Total>
<Taste>26</Taste> <Plating>13</Plating>
<Originality>13</Originality> </Score> </IronChef>
<Challenger name="Roberto Trevino" food="Modern
Latin"> <Score> <Total>43</Total>
<Taste>19</Taste> <Plating>11</Plating>
<Originality>13</Originality> </Score> </Challenger>
</Battle> <Battle ingredient="Duck"> <IronChef
name="Bobby Flay" food="Southwestern"> <Score>
<Total>43</Total> <Taste>21</Taste>
```

xmlParse(myXmlDoc)

```
xml document [short version]
ICA  XmlText
     Battle  XmlText
             XmlAttributes  struct
                            ingredient  Buffalo
             IronChef  XmlText
                       XmlAttributes  struct
                                      food  Southwestern
                                      name  Bobby Flay
                       Score  XmlText
                              Total        XmlText  50
                              Taste        XmlText  24
                              Plating      XmlText  13
                              Originality  XmlText  13
             Challenger  XmlText
```

Figure 1: Comparison of XML documents

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//
Battle[last()]")>
```

In this expression only the last "Battle" node is returned. Since there is no sorting or ORDER BY mechanism in XPath, nodes are returned in the same order as they appear in the original XML document. Thus the expression "//Battle[last()]" returns the "Battle" node in which potatoes are the secret ingredient, because it is the last one in the XML file.

You can also use a combination of functions and operators to pare down the results returned. This expression will return only the second to last "Battle" node:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//Battle[last()-
1]")>
```

and this expression will return the first three "Battle" nodes:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//
Battle[position()<4]")>
```

You can also reference the values of nodes using the various set of operators in XPath:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//Battle/IronChef/
Score[Total>50]")>
```

All "Score" nodes that belong to Iron Chefs who scored a total above 50 are returned with this expression. This use of operators can apply to attributes as well as nodes. The following returns all "Battle" nodes where Bobby Flay is the Iron Chef:

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//Battle[IronChef/
@name='Bobby Flay']")>
```

and what follows returns all "Challenger" nodes where the challenger cooks Italian food.

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//Challenger[@food
='Italian']")>
```

This example brings about a problem. There is both a challenger, Roberto Donna, and an Iron Chef, Mario Batali, who each cook Italian food. However the previous example only returns Roberto Donna since the expression explicitly lists "//Challenger" before the "@food='Italian'" clause. XPath has a logical OR operator which will give us a work around to this problem. The pipe "|" character will concatenate two or more expressions together.

```
    <cfset XPathResult = xmlSearch(ironChefXMLObject, "//Challenger[@food
='Italian']|//IronChef[@food='Italian']")>
```

This expression is really composed of two separate sub-expressions: "//Challenger[@food='Italian']" and "//IronChe f[@food='Italian']." The first sub-expression returns the complete "Challenger" node, including its children and descendants, for all chefs who cook Italian food. The second sub-expression returns the "IronChef" node, including its children and descendants, for each Iron Chef who cooks Italian food. The first sub-expression returns one node set, the second four node sets, thus creating a five-node set that ColdFusion returns as a five-element array.

As you can see XPath has a full set of operators. Here is a list of all of the XPath operators and their description:

|	Node set concatenation.
+	Addition
-	Subtraction
*	Multiplication
div	Division
=	Equal
!=	Not Equal
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to
or	Or
and	And
mod	Modulus

Many of the operators have multiple uses, however, remember that since ColdFusion can only return node sets, some common uses of these operators cannot be used. For example, you cannot add together the total scores of all the Iron Chefs and see which one had the highest overall score for the season. To do this you will have to leverage other features of ColdFusion, which are out of the scope of this article.
So far we have written expressions based off of known XML schemas. Oftentimes you will not know the complete schema before you need to manipulate the data within the file. XPath has three built-in wildcard handlers for this very situation: *, @,* and node().

```
<cfset XPathResult = xmlSearch(ironChefXMLObject, "//Challenger/*")>
<cfset XPathResult = xmlSearch(ironChefXMLObject, "//Challenger/@*")>
<cfset XPathResult = xmlSearch(ironChefXMLObject, "//Challenger/node()")>
```

"//Challenger/*" will return the nodes and their values of all the children and descendants of "Challenger," but not "Challenger" itself. It also does not return any of the attributes or attribute values of any of the node sets. This is different than "//Challenger," which will return "Challenger" and it children/descendants, as well as the attributes of those nodes. "//Challenger/@*" will return the same node set, however it will only return the attributes of those nodes. If you want to return all of the nodes and attributes of those nodes, you would use the node() function, ("//Challenger/node()").

Running XPath expressions against XML documents in ColdFusion does not change the original XML document. This allows developers to experiment freely when trying to learn XPath, without worrying about the consequence (as they might when playing with SQL statements). There are scores of reference materials dedicated to the ins and outs of XPath, just be careful when investing any money into reference materials,

AN ADVANCE LOOK AT BLAC
COLDFUSION
SPECIAL ISSUE: COLDFUSION SECURITY
COLDFUSION Developer's Journal
ColdFusionJournal.com
September 2004 Volume: 6 Issue: 9
SANDBOX SECURITY
By Alex Hübner
Security: ColdFusion Security Best Practices Knowing the security risks are there is half the battle
Bryan Murphy 16
Security: Public-Key Encryption Making strong encryption nearly painless
Wayne Graham 20
cfcUnit: ColdFusion Unit Testing Framework Paul Kenney's version
Harry Klein 30
<bf> on< cf>: Defending ColdFusion Against…
Ben Forta 36
Real-World Uses: Using ColdFusion for Network Monitoring Find the exact piece of data you need
Ryan Emerle Matthew I. Fusfield 42
Security: Top Ten Web Security Tips You can't be too careful
Michael Smith 48
Editorial Reaching Deeper Simon Horwith page 5
Community Focus Problems Securing Your Applications? Simon Horwith page 7
CF101 ColdFusion's Application Framework Jeffry Houser page 28
CFUGs ColdFusion User Groups page 40
FOUNDATIONS 'Selling' ColdFusion Hal Helms page 46
Web Services Edge 2005 East February 15-17, 2005 Hynes Convention Center Boston, MA
SYS-CON MEDIA

Subscribe Today!

SAVE 16%

12 Issues for 89^{99}

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 $29.89 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

remembering that ColdFusion does not support all that XPath has to offer. Do not be put-off by ColdFusion's crippled XPath engine though, Macromedia has made sure that the most useful and timesaving features are available to you. If you would like more information on XPath and how ColdFusion implements it, you can check out both the CFML reference guide and the ColdFusion developer's guide – available on macromedia.com. The W3C also has two great XPath references, the XML Path Language guide (www.w3.org/TR/XPath) and the W3C Schools XPath tutorial (www.w3schools.com/XPath).

About the Author

Nik Molnar is a ColdFusion/Flex developer with over seven years experience. He has led teams through the development of enterprise applications for the mortgage, sports ticketing, and stock industries. He is an amateur Iron Chef and posts regularly at his blog: foodDuo.com. He lives with his wife Katy and his dog Jacques in Orlando, Florida.

nik@truetechnology.com

Listing 1

```
<?xml version="1.0" encoding="iso-8859-1"?>
<ICA>
      <Battle ingredient="Buffalo">
            <IronChef name="Bobby Flay" food="Southwestern">
                  <Score>
                        <Total>50</Total>
                        <Taste>24</Taste>
                        <Plating>13</Plating>
                        <Originality>13</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Rick Bayless" food="Authentic Mexican">
                  <Score>
                        <Total>49</Total>
                        <Taste>24</Taste>
                        <Plating>12</Plating>
                        <Originality>13</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Catfish">
            <IronChef name="Mario Batali" food="Italian">
                  <Score>
                        <Total>52</Total>
                        <Taste>26</Taste>
                        <Plating>13</Plating>
                        <Originality>13</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Roberto Trevino" food="Modern Latin">
                  <Score>
                        <Total>43</Total>
                        <Taste>19</Taste>
                        <Plating>11</Plating>
                        <Originality>13</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Duck">
            <IronChef name="Bobby Flay" food="Southwestern">
                  <Score>
                        <Total>43</Total>
                        <Taste>21</Taste>
                        <Plating>11</Plating>
                        <Originality>11</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Ming Tsai" food="East-West">
                  <Score>
                        <Total>48</Total>
                        <Taste>26</Taste>
                        <Plating>10</Plating>
                        <Originality>12</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Chocolate-Coconut">
            <IronChef name="Mario Batali" food="Italian">
                  <Score>
                        <Total>50</Total>
                        <Taste>24</Taste>
                        <Plating>13</Plating>
                        <Originality>13</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Michael Laiskonis" food="Neo-Classical">
                  <Score>
                        <Total>45</Total>
                        <Taste>23</Taste>
                        <Plating>12</Plating>
                        <Originality>10</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Crab">
            <IronChef name="Masaharu Morimoto" food="Japanese">
                  <Score>
                        <Total>39</Total>
                        <Taste>18</Taste>
                        <Plating>12</Plating>
                        <Originality>9</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Rob Feenie" food="French-Asian">
                  <Score>
                        <Total>45</Total>
                        <Taste>19</Taste>
                        <Plating>13</Plating>
                        <Originality>13</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Squash">
            <IronChef name="Bobby Flay" food="Southwestern">
                  <Score>
                        <Total>48</Total>
                        <Taste>24</Taste>
                        <Plating>12</Plating>
                        <Originality>12</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Govind Armstrong" food="Seasonal Califor-
nia">
                  <Score>
                        <Total>36</Total>
                        <Taste>19</Taste>
                        <Plating>9</Plating>
                        <Originality>8</Originality>
                  </Score>
            </Challenger>
      </Battle>
      <Battle ingredient="Scallops">
            <IronChef name="Masaharu Morimoto" food="Japanese">
                  <Score>
                        <Total>50</Total>
                        <Taste>22</Taste>
                        <Plating>13</Plating>
                        <Originality>15</Originality>
                  </Score>
            </IronChef>
            <Challenger name="Roberto Donna" food="Italian">
                  <Score>
                        <Total>33</Total>
                        <Taste>15</Taste>
                        <Plating>9</Plating>
```

```
                <Originality>9</Originality>
            </Score>
        </Challenger>
    </Battle>
    <Battle ingredient="Cheese">
        <IronChef name="Mario Batali" food="Italian">
            <Score>
                <Total>53</Total>
                <Taste>29</Taste>
                <Plating>11</Plating>
                <Originality>13</Originality>
            </Score>
        </IronChef>
        <Challenger name="Scott Campbell" food="New American">
            <Score>
                <Total>40</Total>
                <Taste>18</Taste>
                <Plating>10</Plating>
                <Originality>12</Originality>
            </Score>
        </Challenger>
    </Battle>
    <Battle ingredient="Mushroom">
        <IronChef name="Mario Batali" food="Italian">
            <Score>
                <Total>45</Total>
                <Taste>24</Taste>
                <Plating>10</Plating>
                <Originality>11</Originality>
            </Score>
        </IronChef>
        <Challenger name="Anita Lo" food="Contemporary American">
            <Score>
                <Total>54</Total>
                <Taste>27</Taste>
                <Plating>13</Plating>
                <Originality>14</Originality>
            </Score>
        </Challenger>
    </Battle>
    <Battle ingredient="Potato">
        <IronChef name="Cat Cora" food="Greek-Aegean">
            <Score>
                <Total>47</Total>
                <Taste>21</Taste>
                <Plating>12</Plating>
                <Originality>14</Originality>
            </Score>
        </IronChef>
        <Challenger name="Alex Lee" food="Contemporary French">
            <Score>
                <Total>46</Total>
                <Taste>22</Taste>
                <Plating>14</Plating>
                <Originality>10</Originality>
            </Score>
        </Challenger>
    </Battle>
</ICA>
```

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

ColdFusion

For more information go to...

U.S.

Alabama
Huntsville
Huntsville, AL CFUG
www.nacfug.com

Alaska
Anchorage
Alaska Macromedia User Group
www.akmmug.org

Arizona
Phoenix
www.azcfug.org

Arizona
Tucson
www.tucsoncfug.org

California
San Francisco
Bay Area CFUG
www.bacfug.net

California
Riverside
Inland Empire CFUG
www.sccfug.org

California
EL Segundo
Los Amgeles CFUG
www.sccfug.org

California
Irvine
Orange County CFUG
www.sccfug.org

California
Davis
Sacramento, CA CFUG
www.saccfug.org

California
San Jose (temporary)
Silicon Valley CFUG
www.siliconvalleycfug.com

California
San Diego
San Diego, CA CFUG
www.sdcfug.org/

California
Long Beach
Southern California CFUG
www.sccfug.org

Colorado
Denver
Denver CFUG
www.denvercfug.org/

Delaware
Kennett Square
Wilmington CFUG
www.bvcfug.org/

Delaware
Laurel
Delmarva CFUG
www.delmarva-cfug.org

Florida
Jacksonville
Jacksonville, FL CFUG
www.jaxfusion.org/

Florida
Winter Springs
Gainesville, FL CFUG
www.gisfusion.com/

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

Florida
Tallahassee
Tallahassee, FL CFUG
www.tcfug.com/

Florida
Palm Harbor
Tampa, FL CFUG
www.tbmmug.org

Georgia
Atlanta
Atlanta, GA CFUG
www.acfug.org

Illinois
East Central
East Central Illinois CFUG
www.ecicfug.org/

Indiana
Avon
Indianapolis, IN CFUG
www.hoosierfusion.com

Indiana
Mishawaka
Northern Indiana CFUG
www.ninmug.org

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

Kentucky
Louisville
Louisville, KY CFUG
www.kymug.com/

Louisiana
Lafayette
Lafayette, LA MMUG
www.cflib.org/acadiana/

Maryland
Lexington Park
California, MD CFUG
http://www.smdcfug.org

Maryland
Rockville
Maryland CFUG
www.cfug-md.org

Massachusetts
Quincy
Boston, MA CFUG
www.bostoncfug.com

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

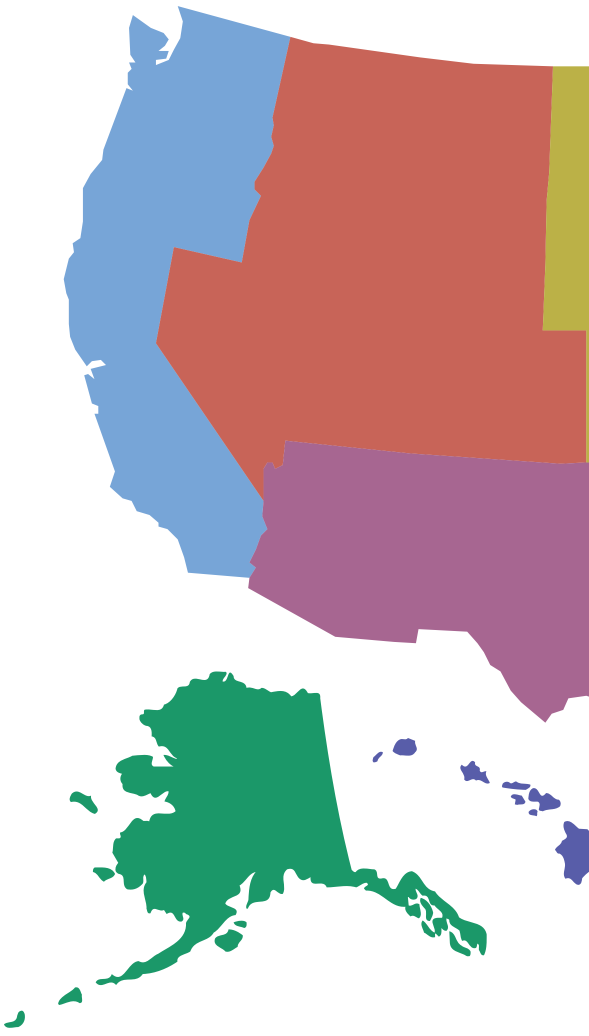
Minnesota
Brooklyn Park
Twin Cities CFUG
www.colderfusion.com

Missouri
Overland Park
Kansas City, MO CFUG
www.kcfusion.org

Missouri
O'Fallon
St. Louis, MO CFUG
www.stlmmug.com/

New Jersey
Princeton
Central New Jersey CFUG
http://www.cjcfug.us/

Nevada
Las Vegas
Las Vegas CFUG
www.sncfug.com/



New York
Albany
Albany, NY CFUG
www.anycfug.org

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

New York
Syracuse
Syracuse, NY CFUG
www.cfugcny.org

North Carolina
Raleigh
Raleigh, NC CFUG
www.ccfug.org

Ohio
Dayton
Greater Dayton CFUG
www.cfdayton.com

Oregon
Portland
Portland, OR CFUG
www.pdxcfug.org

User Groups

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

Pennsylvania
Carlisle
Central Penn CFUG
www.centralpenncfug.org

Pennsylvania
Exton
Philadelphia, PA CFUG
www.phillycfug.org/

Pennsylvania
State College
State College, PA CFUG
www.mmug-sc.org/

Rhode Island
Providence
Providence, RI CFUG
www.ricfug.com/www/meetings.cfm

Tennessee
LaVergne
Nashville, TN CFUG
www.ncfug.com

Tennessee
Germantown
Memphis, TN CFUG
www.mmug.mind-over-data.com

Texas
Austin
Austin, TX CFUG
www.cftexas.net/

Texas
Corinth
Dallas, TX CFUG
www.dfwcfug.org/

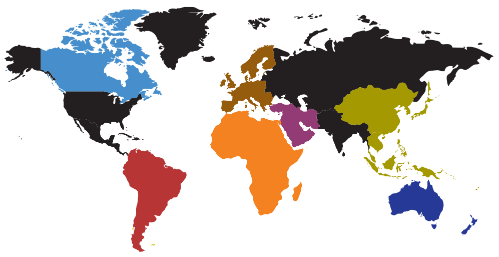
Texas
Houston
Houston Area CFUG
www.houcfug.org

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

About CFUGs

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

INTERNATIONAL



Australia
ACT CFUG
www.actcfug.com

Australia
Queensland CFUG
www.qld.cfug.org.au/

Australia
Southern Australia CFUG
www.cfug.org.au/

Australia
Victoria CFUG
www.cfcentral.com.au

Australia
Western Australia CFUG
www.cfugwa.com/

Brazil
Brasilia CFUG
www.cfugdf.com.br

Brazil
Rio de Janerio CFUG
www.cfugrio.com.br/

Brazil
Sao Paulo CFUG
www.cfugsp.com.br

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Ireland
Dublin, Ireland CFUG
www.mmug-dublin.com/

Italy
Italy CFUG
www.cfmentor.com

Japan
Japan CFUG
cfusion.itfrontier.co.jp/jcfug/jcfug.cfm

Scotland
Scottish CFUG
www.scottishcfug.com

South Africa
Joe-Burg, South Africa CFUG
www.mmug.co.za

South Africa
Cape Town, South Africa CFUG
www.mmug.co.za

Spain
Spanish CFUG
www.cfugspain.org

Switzerland
Swiss CFUG
www.swisscfug.org

Thailand
Bangkok, Thailand CFUG
thaicfug.tei.or.th/

Turkey
Turkey CFUG
www.cftr.net

United Kingdom
UK CFUG
www.ukcfug.org

Études: Studies in Structures

Get ready for a workout

By Hall Helms

The life of a writer is sometimes a lonely one. After writing something, you send it out and aren't sure how it will be received. Last month was a happy change: I received a lot of feedback from my column introducing the idea of études. In the introduction to the series on études, we looked at arrays. This month, let's do a workout on structures after a quick primer.

Structures, also known as associative arrays, are a collection of name-value pairs, typically associated with a single topic or idea. Take the idea of a book. Variables associated with a book might be its author, title, publisher, price, and synopsis. The ColdFusion code for such a book structure looks like this:

```
<cfset book = StructNew() />
<cfset book.title = "A Prayer for Owen Meany" />
<cfset book.author = "John Irving" />
<cfset book.publisher = "Ballantine" />
<cfset book.price = "7.99" />
<cfset book.synopsis = "A Prayer for Owen Meany is a long, sprawling,
coming-of-age epic. It is about the loss of innocence suffered by two
boys and by a country. It is told in the first-person by Johnny Wheel-
wright, and the book is a prayer for his lost best friend, Owen Meany.
Owen was the smallest person he ever knew, with a voice that sounded like
a permanent scream, who believed he was God's instrument. Oddly enough,
he was right." />
```

Ready for the workout?

1. Structures can be written using the dot-separator style (as shown in the code) or using an array-like style. Rework the book structure, using the array style syntax.
2. Consider the following code:

```
<cfset jack = StructNew() />
<cfset jane = StructNew() />
<cfset jack.spouse = jane />
<cfset jane.spouse = jack />
<cfset jack.job = "janitor" />
<cfset jane.job = "stylist" />
<cfset jack.job = "CEO" />
<cfoutput>
   #jane.spouse.job#
</cfoutput>
```

   What will the output be? Why?
3. Given a variable of unknown data type, myVar, write code that produces a list of keys if myVar is a structure.
4. Using code, read the following text file, preferences.txt, creating from it a structure, preferences.

```
background_color: white
text_color: black
font_size: medium
font_family: arial
link_color: red
visited_link_color: green
```

5. Store the preferences variable as a client-scoped variable, config.
6. Restore client.config to an in-memory structure, preferences.
7. Create a list of the instruments (not the players) in the orchestra.strings structure.

```
<cfset orchestra = StructNew() />

<cfset orchestra.strings = StructNew() />
<cfset orchestra.brass = StructNew() />
<cfset orchestra.woodwinds = StructNew() />
<cfset orchestra.percussion = StructNew() />

<cfset orchestra.strings.violin = "Anne Baker" />
<cfset orchestra.strings.viola = "Bob Carter" />
<cfset orchestra.strings.cello = "Danielle Edgerton" />
<cfset orchestra.strings.bass = "Edward Franck" />
```

8. Write code to determine the number of sections (strings, brass, etc.) in the orchestra.
9. Create a copy of orchestra, orchestra2, in such a way that changes to orchestra2 will not affect orchestra.
10. Remove the percussion section from orchestra2.
11. Given this code...

```
<cfset StructClear(orchestra) />
```

...is orchestra still a structure?

12. Given the following code, output the keys and values of the

structure, by alphabetical order of keys.

```
<cfset zoo = StructNew() />
<cfset zoo.zebra = "Zed" />
<cfset zoo.giraffe = "Jerry" />
<cfset zoo.hyena = "Harry" />
<cfset zoo.anteater = "Andy" />
```

13. Given this code...

```
<cfset aList = "a,e,g,m" />
```

...write code that outputs the value of any matching key in a structure, **struct**. That is, if **struct** has a key called **g**, your code should output the value of **g.**

14. Write code that models the following project using structures and arrays. The name of the project is "Plan Coyote Conference". The planner for this project is Francis Battell (francis@acmeproducts.com). She works for the marketing division of the Acme Corporation, maker of high-tech products to catch roadrunners. The conference is the annual Coyote Conference. This year, the conference is limited to only 10 conferees (it's very exclusive).

The conference will be held in Phoenix, AZ, at the Desert Inn. Francis needs to keep track of the e-mail of each participant and their shirt size (for a promotional T-shirt). The keynote speaker this year is Wile E. Coyote (e-mail: wile@ihateroadrunners.com). Francis needs to keep track of the airline he's flying in on, date and time for flight in, and date and time for flight out.

The Desert Inn will be preparing all meals for the participants, but needs to know the eating preferences for each. The choices are regular, vegetarian, kosher, and low-carb. Francis has an assistant, Jerry Capra (jerry@acmeproducts.com).

The sessions this year will be on:

- "New Products from Acme" given by Grice Brinkdot
- "Outwitting You-Know-Who" given by Omar Serappagi
- "Knowing Your Enemy" given by Wilder Frumptious
- "Tuning Your Acme Acquisitions" given by Sarquad DeFroonis

Participants need to register for the session(s) they will be attending. The participants at each session will receive a free book.

For the "New Products" session, the book will be *Technology for Dummies*, written by Richard Smurdvercker. Its retail price is $19.95 and it's published by Vanity Press, Inc.

For the "Outwitting" session, the book will be *Strategy for Dummies* written by Donald Smurdvercker. Its retail price is $34.95 and it's published by Vanity Press, Inc.

For the "Knowing Your Enemy" session, the book will be *Knowledge for Dummies*, written by Betty Smurdvercker. Its retail price is $62.95 and has the same publisher as brothers Donald and Richard have.

Finally, the "Tuning" session will use the book *Tuning for Dummies*, written by Betty and Donald Smurdvercker. Its retail price is $39.95 with the same publisher.

Structures are extraordinarily useful for building software models. There are many situations where structures express the related nature of data perfectly. The earliest object-oriented languages took structures, attached user-defined functions to them, and created the idea of classes. Taking the time to become an expert in their use will help you create software of greater flexibility and simplicity.

(My solutions to the étude problems can be downloaded from www.halhelms.com/etudes/structures.htm.)

About the Author

Hal Helms is the author of several books on programming. Hal teaches classes in Java, C#.NET, OO Programming with CFCs, Design Patterns in CFCs, ColdFusion Foundations, Mach-II, and Fusebox. He's the author of the popular Occasional Newsletter and his site is www.halhelms.com.

hal@halhelms.com

Using Gateways PART 2

The basics

By Guy Rish

Part 1 in this series (*CFDJ*, Vol. 7, issue 5) discussed the options in the ColdFusion Administrator for configuring the event gateway facility in ColdFusion MX 7 and how to register new event gateway types.

There was little in the way of explaining how to use event gateways, so in this article we'll explore the basics of using event gateways through two examples: the code from the previous article and another of the out-of-the-box event gateway types, the CFML asynchronous event gateway type. These two gateway types represent the two fundamental varieties: the types that respond to events external to a ColdFusion application and the types that respond to events generated within a ColdFusion application.

Event gateways represent a new kind of service architecture for ColdFusion that is composed of a number of logical groups, each with its own intricacies. This article glosses over much of the deeper details with the intent of getting you up and running without bogging you down. Not to worry though, the finer details ignored now will be covered in greater detail in a later article.

A rough overview of the choreography of gateways can be seen in the ColdFusion MX 7 product documentation (*Developing ColdFusion MX Applications, Using Event Gateways, About event gateways, How event gateway applications work*). Many of the framed pieces in this diagram are identified with common event gateway jargon discussed in Part 1. [http://livedocs.macromedia.com/coldfusion/7/htmldocs/images/gatewayb.jpg]

The two discrete pieces we're going to discuss in this article are the listener CFCs and CFM Pages (the calling logic being either a CFML template or a ColdFusion Component).

CFML Code for Gateways

To create a gateway instance you need a gateway type, a configuration file (which is optional in some cases) and a listener CFC. ColdFusion MX 7 comes with seven gateway types to start you off, one of which, the DirectoryWatcher type, was used in the previous article to verify the instructions for creating a gateway instance. Since the sample code was for an event gateway that responded to external OS level events, it had only a listener CFC written for it, which is a good place to begin.

Listener CFCs

Developers familiar with frameworks in other languages will recognize the role that the methods in a listener CFC fulfill – the event handler. Similar to a class implementing a component or control delegate in the .NET Framework or a Java bean event listener in Java, the listener CFC is the chunk of code that is the implementation-specific part of the more generic problem domain as defined by the gateway type.

While the gateway type defines the method names, the signature always remains the same (with the obvious exception of the method name itself): a public method with one argument, a structure, which may return a string. (*Note:* Signature is a term used to indicate a function's or a method's definition, its name, the names and data types of its arguments, and data type of the returned value [http://en.wikipedia.org/wiki/Signature_%28Computer_Science%29].)

Event Handling Methods

The one method argument is called the *CFEvent* and contains information about the gateway instance, the default listener CFC that the gateway instance will use, and a structure containing any data elements sent by the method's calling logic. These data elements can be populated by the Java code of the gateway type as well as the CFML code invoking the gateway instance.

The Members of CFEvent

Each event handler method receives a single structure, the

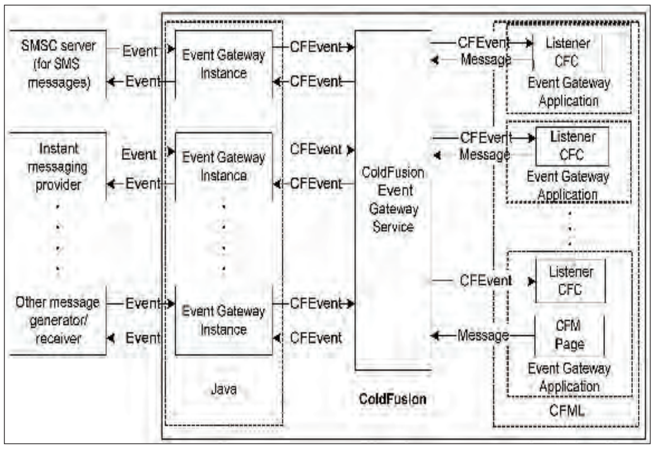


Figure 1: Event Gateway Overview

CFEvent argument. This structure contains seven members by default that are almost solely for informational purposes. They are listed in Table 1 and in the ColdFusion product documentation (*CFML Reference, ColdFusion MX Event Gateway Reference, CFML CFEvent structure*).

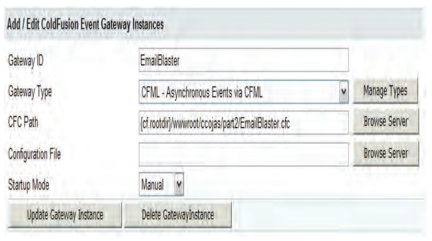
Reviewing the Previous Sample

Looking at the code from Part 1 (available at: [http://res.sys-con.com/story/may05/86134/source.html]), we find two files, CCOJASDirectoryWatcher.cfg (the configuration file) and CCOJASDirectoryWatcher.cfc (the listener CFC), which were used to create the gateway instance, CCOJASDirectoryWatcher.

Within this component we see three methods: *onAdd*, *onDelete*, and *onChange*. These methods are defined by the DirectoryWatcher gateway type (*Developing ColdFusion MX Applications, Using Event Gateways, Using the example event gateways and gateway applications*). These methods are the event handlers for the DirectoryWatcher gateway instance and none of them are required. If a listener CFC does not implement one of these event handling methods, the gateway instance does not attempt to make a call to the method. This means that the listener CFC needs to implement only what is germane to the problem domain and nothing more.

The CFEvent's data Member

In the case of this example, all three event handling methods were implemented and, as was demonstrated in Part 1 ("Cold Cup o' Joe – Another Shot: Gateways to Fun, Putting It All Together"), a log file was updated each time a file was added to, changed, and deleted from the watched directory. What is more interesting now is not just the execution of the gateway instance but the variables that get passed to the event handling methods.



Add / Edit ColdFusion Event Gateway Instances

Gateway ID	EmailBlaster
Gateway Type	CFML - Asynchronous Events via CFML
CFC Path	{cf.rootdir}/wwwroot/ccojas/part2/EmailBlaster.cfc
Configuration File	
Startup Mode	Manual

Figure 2: The EmailBlaster Gateway Instance

Referring to CCOJASDirectoryWatcher.cfc from Part 1, you'll see that the first functional line of each method (lines 5, 12, and 19) pull out a structure called *data* from each method's *CFEvent* argument. This contains data about each OS level event that the DirectoryWatcher was monitoring for. This structure contains three members, identified in the ColdFusion product documentation (*Developing ColdFusion MX Applications, Using Event Gateways, Using the example event gateways and gateway applications, Example event gateway, CFC methods*) and shown in Table 2.

Each gateway type passes different members in the *data* member. It is necessary to comb the product documentation to determine which elements will be populated by the gateway type.

The Calling Logic

While the DirectoryWatcher gateway and other gateways like it are useful for extending ColdFusion's integration with various services and OS-level functions, their general lack of ability to interact with an application in a more imperative way is limiting. Fortunately, this is not the only means of using gateways. CFML has a couple of new functions for passing data back and forth between templates and components; the most direct way is through the *sendGatewayMessage* function.

sendGatewayMessage

The *sendGatewayMessage* function (http://livedocs.macromedia.com/coldfusion/7/htmldocs/00000622.htm#153566) takes two arguments: *gatewayID* and *messageStruct*. The *gatewayID* argument is the name of a configured gateway instance. The *messageStruct* argument is a structure containing information to be sent to the indicated gateway instance – this structure will become the *data* structure member of the *CFEvent* argument to the listener CFC's event handling methods. This function may return a string value – the string returned from the listener CFC's event handling methods. The specific string value that is returned is dependent upon the gateway type of the instance to which the message is sent.

Again, as noted previously, the exact composition of the *data* structure member, the *messageStruct* argument of the *sendGatewayMessage* function, is specific to each gateway. The truly flexible thing about this mechanism is that while certain specific values are necessary, additional values can be passed without adversely affecting the basic functionality.

messageStruct

The structure argument passed to *sendGatewayMessage* not only contains various bits of data as might be required by the gateway type or that you might choose to pass, but it allows for four optional structure members as well (see Table 3).

We'll use some of these optional members in later examples both in this article and the series.

Putting It All Together

Once again, armed with a basic understanding of how events are handled and how to purposefully invoke them, let's put together a simple example.

One of the more exciting possibilities with event gateways is the ability to perform asynchronous actions, and to launch a subordinate task and not have to wait for it to complete before the parent logic can continue. Contriving a use for this facility for demonstration purposes seldom conveys its real power, but we'll have a go at it.

Popular sites often like to provide a subscription to visitors that offers timely information. Many sites have started to move toward RSS publication and some have started podcasting. But while RSS blurbs have become the condensed news and announcements for the ADD generation and podcasting the archiveable interviews and Internet soundbites, newsletters still remain a rich contextual way of sharing information. Naturally there are no special tricks to sending out newsletters or other mass mailings from ColdFusion. Launching a template that loops through a long list of e-mail addresses pretty much does the trick, though it tends to hang up a browser's output even when using CFFLUSH to provide user feedback, feedback that is seldom of interest to the person who has hit the submit button that initiates the mass mailing process. Wouldn't it be nice if after pressing the button, the administrative user could launch the mailing and then get back to doing other things in the administrative console or Intranet portal?

Enter EmailBlaster.

EmailBlaster

EmailBlaster is nothing more than a gateway instance for the asynchronous CFML gateway type, but it is an excellent example of a user-interactive gateway application. Configuring it is as simple as creating a new CFML instance that points to the EmailBlaster.cfc listener CFC (see Listing 1), the specifics of which can be seen in Figure 2.

The event handler in the listener CFC is a method called *onIncomingMessage*, which can be seen in Listing 1.

Herein we see the common action of extracting the *data* structure member from the incoming *CFEvent* argument. It is at this point that things start to become interesting as a structure member, *emailData*, is in turn extracted from it. A simple inference can be made about the nature of *emailData* when looking at the CFMAIL call on line 9; *emailData* is a query containing at least one column: *address*. In addition, the CFMAIL start tag shows two other structure members expected in the *data* structure: *sender* for the FROM attribute and *subject* for the mail's SUBJECT attribute.

All of this begs the question – how does a query and some string data get into the listener CFC? It gets passed in the *messageStruct* argument of a *sendGatewayMessage* call.

Member	Description
gatewayID	An identifier for the gateway instance invoked.
data	The data sent by the gateway type and/or the calling logic.originatorID Some value indicating the originator of the message, this value depends upon the gateway type. This might be the file path of the calling template or it might be an ID cookie used to identify a logged-in user for the messaging gateway types.
gatewayType	The gateway type.
cfcPath	The file path of the listener CFC.
cfcMethod	The name of the event handling method within the listener CFC (the method receiving the CFEvent structure as an argument).
cfcTimeout	The number of seconds the gateway type provides for the listener CFC to process the message.

Table 1: The Members of CFEvent

Member	Description
type	The event action, ADD, CHANGE or DELETE.
filename	The filename being added, changed or deleted.
lastModified	The datestamp that the file was created or modified.

Table 2: The Members of the DirectoryWatcher data Structure

Member	Description
cfcPath	This value overrides the listener CFC configured in the ColdFusion Administrator for the gateway instance being invoked. This corresponds to the cfcPath member of the CFEvent structure.
method	This value overrides the name of the event handler method within the listener CFC that gets called when the gateway instance is invoked. This corresponds to the cfcMethod member of the CFEvent structure.
originatorID	This value overrides the identifier of the calling logic or the identifier cookie used to indicate a particular "session" or "user" to the listener CFC. This corresponds to the originatorID member of the CFEvent structure.
timeout	The value overrides the number of seconds the gateway type code will allow the listener CFC to process the message. This corresponds to the timeout member of the CFEvent structure.

Table 3: Optional messageStruct Members

The Calling Logic

The other file in this pair is EmailBlaster.cfm (see Listing 2).

Despite the number of lines, there really isn't much happening in this template. Most of the action is focused upon creating the content that will get passed to the gateway instance, a structure containing a query and two strings. The query is composed of only a single column: *address* and populated with a single row of data and then added to the structure with the now-familiar name of *emailData*.

The two other members, the sender address and the subject line, are added to the structure. Finally, on line 15, the EmailBlaster gateway instance is invoked with a call to *sendGatewayMessage* and the message structure is passed.

The Change Up

Clearly there is a problem in making EmailBlaster useful in any capacity if it's only capable of calling to a single listener CFC. It would be imminently more useful if additional data elements could be passed and different e-mail letter content could be used. One way of doing this, which serendipitously helps to demonstrate a cool feature of gateways, is to indicate a different listener CFC in the *messageStruct* passed to *sendGatewayMessage*.

The Calling Logic, Part Deux

A few additions to what was our original calling logic will produce Newsletter.cfm (see Listing 3).

Herein there are two additional columns added to the query that is being sent in the *messageStruct*, *firstname*, and *lastname*. Everything else remains more or less the same until line 15 where the structure member *cfcPath* (which was shown briefly in Table 3) is added, indicating the Newsletter.cfc in the same calling directory (obtained using CFML's *expandPath* function).

The Listener CFC, Part Deux

The new listener CFC, Newsletter.cfc (see Listing 4), is merely a rehash of the default listener configured in the ColdFusion Administrator for this gateway instance. The one point of change is line 10 where the two new columns added in the calling logic are pasted into the output of the e-mail body.

It should be noted that the sample code for this article requires that you do two things before you can successfully run the examples:

1. Configure your ColdFusion Server's mail server settings
2. Change the e-mail addresses used in the templates to valid values:
   - EmailBlaster.cfm, line 7
   - EmailBlaster.cfm, line 10
   - Newsletter, line 9
   - Newsletter, 12

Wrapping It Up

The simple examples in this article demonstrate only the barest minimum of a more complicated system. Now that you have a basic grounding in setting up gateway instances, writing listener CFCs, and passing messages to instances, we can dig a little bit deeper. The next installment in this series will look at integrating gateway instance into applications using ColdFusion's persistent scopes and what some of the impacts upon some popular practices.

Be sure to drop by the CCOJ blog and post your feedback and questions about this article and the series. (*Note:* Listings 1–4 are also available on the CCOJ blog [http://**ccoj**.coldfusionjournal.com], ccojas-part2-code.zip.)

About the Author

Guy Rish is a ColdFusion and .NET developer at Vente, Inc. (http://www.venteinc.com) as well as president at Gestaltech (/www.gestaltech.com) He is an active developer and writer for various languages and technologies. He has contributed work in books on ColdFusion MX, Flash MX, and Dreamweaver MX.

grish@gestaltech.com

Listing 1: The Listener CFC code, EmailBlaster.cfc

```
<cfcomponent displayname="EmailBlaster" output="no">

    <cffunction name="onIncomingMessage" access="public"
returntype="string" output="no">
        <cfargument name="CFEvent" type="struct" required="yes"/>
        <cfscript>
            var data = CFEvent.data;
            var emailData = data.emailData;
        </cfscript>
        <cfmail query="emailData" from="#data.sender#" to="#emailData.
address#" subject="#data.subject#">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
molestie consequat, vel illum dolore eu feugiat nulla facilisis at
vero eros et accumsan et iusto odio dignissim qui blandit praesent
luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

        </cfmail>
    </cffunction>

</cfcomponent>
```

Listing 2: The Calling Logic, EmailBlaster.cfm

```
<cfscript>
    message = structNew();

    q = queryNew("address");
    queryAddRow(q);
    querySetCell(q, "address", "joesixpack@somewhere.com", 1);

    message.emailData = q;
    message.sender = "blaster@somewhere.com";
    message.subject = "EmailBlasterTest";

    sendGatewayMessage("EmailBlaster", message);
</cfscript>
```

Listing 3: The Calling Logic, Part Deux, Newsletter.cfm

```
<cfscript>
    message = structNew();
    q = queryNew("firstname,lastname,address");
    queryAddRow(q);
    querySetCell(q, "firstname", "Joe", 1);
    querySetCell(q, "lastname", "Sixpack", 1);
    querySetCell(q, "address", "joesixpack@somewhere.com", 1);

    message.emailData = q;
    message.sender = "blaster@somewhere.com";
    message.subject = "A Newsletter";

    message.cfcPath = expandPath('./Newsletter.cfc');

    sendGatewayMessage("EmailBlaster", message);
</cfscript>
```

Listing 4: The Listener CFC, Part Deux, Newsletter.cfc

```
<cfcomponent displayname="Newsletter" output="no">

    <cffunction name="onIncomingMessage" access="public"
returntype="string" output="no">
        <cfargument name="CFEvent" type="struct" required="yes"/>
        <cfscript>
            var data = CFEvent.data;
            var emailData = data.emailData;
        </cfscript>
        <cfmail query="emailData" from="#data.sender#" to="#emailData.
address#" subject="#data.subject#">
Hello #emailData.firstname# #emailData.lastname#!

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam
nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation
ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.
Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse
molestie consequat, vel illum dolore eu feugiat nulla facilisis at
vero eros et accumsan et iusto odio dignissim qui blandit praesent
luptatum zzril delenit augue duis dolore te feugait nulla facilisi.

        </cfmail>
    </cffunction>

</cfcomponent>
```

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

MAX is Angela Buraglia.

Angela Buraglia is a web developer and work-from-home mom who somehow finds time to co-author technical books, including her latest **Dreamweaver MX 2004 Killer Tips**. She is one of thousands of leading designers and developers that will gather at MAX 2005 this October to learn new skills, explore emerging technologies, share techniques with peers, and put exciting new ideas in motion.

Learn
Choose from over 90 different hands-on and workshop sessions – in five tracks – to create a schedule to meet your specific needs. Hear Angela Buraglia and other industry leaders speak on best practices and coming trends and technologies.

Connect
Exchange ideas with other designers and developers at networking sessions. Attend "birds-of-a-feather" sessions to connect with like-minded peers.

MAX 2005 happens October 16-19 in Anaheim, California. Please join us.

Register Now
Save $200 and get the best session selection with early-bird registration at macromedia.com/max (ends August 26).

Ideas in motion. The 2005 Macromedia® Conference.

Intermedia.NET...

Now serving the hottest ColdFusion.

At Intermedia.NET we go beyond the industry standard by supporting the hottest new Coldfusion software, offering power like never before. For nearly a decade, we've been providing reliable, secure hosting to thousands of companies across the globe. We can do the same for you.

Intermedia.NET's premier hosting services include:

- ColdFusion MX hosting
- Competitive plans
- Security sandboxes
- Custom tag registration
- Verity collections search engine
- Guaranteed service levels

Unprecedented power, unmatched reputation...
Intermedia.NET is your hosting solution.

INTERMEDIA.NET

Call us at: 1.888.379.7729
e-mail us at: sales@intermedia.NET
Visit us at: www.intermedia.NET