THE CHANGING STATE OF THE WEB PG 5

ColdFusion Developer's Journal

ColdFusionJournal.com

October 2005 Volume:7 Issue:10

What's AJAX Part 2

PLUS...

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

Get Dreamweaver 8 along with 93 server-side features for free

Each license or upgrade of Dreamweaver 8 or Studio 8 bought from InterAKT brings you a free license of ImpAKT.

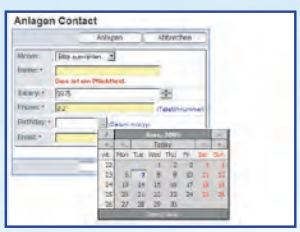
Get a $129 product **FOR FREE!**

More details on **www.interaktonline.com/Macromedia/**

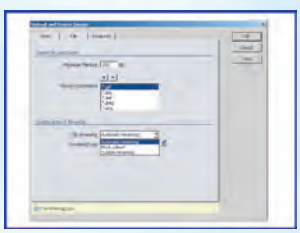
ImpAKT helps **Dreamweaver designers do complex database stuff**. The bundle includes 4 products to handle file and image upload features, user authentication section for your sites, validate your visitors' input and send e-mails on form submit.

MX Form Validation

Validate form fields
Client-side and server-side validation
Rich validation formats library
Preserve submitted values on error
Date picker, Numeric text field
Masked text field, Restricted Textarea



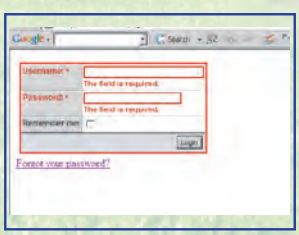
MX File Upload



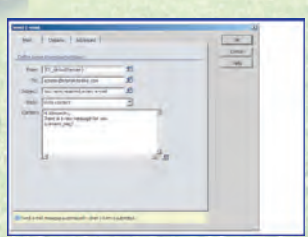
File and image upload
Proportional image resize
Restrict file size
Dynamic thumbnail creation
Download uploaded file
Delete files on record delete

MX User Login

User registration and login
Encrypted password
"Remember me" feature
Account activation by e-mail
Restrict access to page
Forgot password feature



MX Send E-mail



Send e-mail on form submit
Send e-mail to multiple recipients
Send page section by e-mail
CSS and image links in e-mail body
Set e-mail priority

work smart

Interakt
macromedia ALLIANCE PARTNER

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.

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

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

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

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ
FOR LIST RENTAL INFORMATION:
Kevin Collopy: 845 731-2684, kevin.collopy@edithroman.com
Frank Cipolla: 845 731-3832, frank.cipolla@epostdirect.com

For promotional reprints, contact reprint coordinator Megan Mussa, megan@sys-con.com. SYS-CON Publications, Inc., reserves the right to revise, republish and authorize its readers to use the articles submitted for publication.
All brand and product names used on these pages are trade names, service marks, or trademarks of their respective companies.

The Changing State of the Web

By Simon Horwith

Since our last issue, several events have transpired that have significant impact for ColdFusion developers.

I have been keeping my eye on several trends and have been evaluating whether or not they deserve focus in *ColdFusion Developer's Journal*. I've begun blogging about these trends and about my observations and ideas regarding these trends and the future of *CFDJ*. This month's editorial is an overview of these events, observations, and trends.

Web logs (blogs) are not new; they've been around for several years now, and many ColdFusion developers and companies now blog on a regular basis. These blogs have proven to be a valuable tool for developers – many of the developer blogs are becoming the best available resource to learn about specific topics and/or about how to solve specific problems. This information is dynamic and free, and comes "from the trenches," i.e., is being authored not by professional authors or evangelists but by people who have real world experience with the subject matter at hand. I think that *CFDJ* needs to begin introducing blog related articles to readers, as well as turning to the information being blogged as a source for some of the magazine's content.

There has also been somewhat of a surge in open source projects written in ColdFusion. This dates back to the old Allaire Tag Gallery and has continued to grow with the addition of sites such as cflib.org, cfopen.org, and openXCF (http://sourceforge.net/projects/openxcf/) on sourceforge. Brian Rinaldi has even compiled an impressive list of all of the open source CF projects on his blog at http://*www.remotesynthesis.com/blog/index.cfm/2005/11/1/ColdFusion-OpenSource-Project-List*. The emergence of these projects is a similar trend to blogging in that they are a free and open offering made by members of the community to the community at large, and they also deserve attention in the magazine.

Many of the open source projects and initiatives within the CF community are in the form of frameworks. For a very long time Fusebox was the only framework available, then we had one or two alternatives, but in the past six months or so there has been a surge in releases of frameworks. Many of the recent releases have been in the form of frameworks that only offer a solution to meet very specific needs (such as data access and persistence) in an application and so they are designed to integrate and work within the context of other frameworks that are out there. This is leading to the development and availability of tools that allow developers to build applications that are separated into tiers and that are not dependent on any one framework or methodology… which is a good thing. *CFDJ* needs to begin featuring articles that explain and explore these new frameworks. This December *CFDJ* is going to run what I hope will be the best issue in its history – an entire issue of articles examining all of the major frameworks available. The developers of each major framework have taken Macromedia's CFML Pet Market application, implemented it in their framework, and written about the solution. The

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

BlueDragon Certified V.I.P. Hosting Partner

Visit us online or call to order!

WWW.CFDYNAMICS.COM

1 - 866 - 233 - 9626
866 - CFDYNAMICS

SYS-CON MEDIA

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

vp, business development
Grisha Davida grisha@sys-con.com

group publisher
Jeremy Geelan jeremy@sys-con.com

advertising

senior vp, sales & marketing
Carmen Gonzalez carmen@sys-con.com

vp, sales & marketing
Miles Silverman miles@sys-con.com

advertising director
Robyn Forma robyn@sys-con.com

advertising manager
Megan Mussa megan@sys-con.com

sales & marketing director
Dennis Leavey dennis@sys-con.com

associate sales manager
Kerry Mealia kerry@sys-con.com

sys-con events

president, events
Grisha Davida grisha@sys-con.com

national sales manager
Jim Hanchrow jimh@sys-con.com

customer relations

circulation service coordinators
Edna Earle Russell edna@sys-con.com
Linda Lipton linda@sys-con.com

manager, jdj store
Brunilda Staropoli bruni@sys-con.com

sys-con.com

vp, information systems
Robert Diamond robert@sys-con.com

web designers
Stephen Kilmurray stephen@sys-con.com
Vincent Santaiti vincent@sys-con.com
Shawn Slaney shawn@sys-con.com

online editor
Roger Strukhoff roger@sys-con.com

accounting

financial analyst
Joan LaRose joan@sys-con.com

accounts payable
Betty White betty@sys-con.com

accounts receivable
Gail Naples gailn@sys-con.com

subscriptions

Subscribe@sys-con.com
Call 1-888-303-5282

For subscriptions and requests for bulk orders, please send your letters to Subscription Department
Cover Price: $8.99/issue
Domestic: $89.99/yr (12 issues)
Canada/Mexico: $99.99/yr
All other countries $129.99/yr
(U.S. Banks or Money Orders)
Back issues: $12 U.S. $15 all others

company I am CIO for, About Web LLC, has registered cfpetmarket.com – it will not only host these sample applications but will also let anyone who'd like to comment, download, and/or submit their own version of CF Pet Market.

There is one last trend that I am focusing on heavily: Web 2.0. Tim O'Reilly recently wrote an excellent description of Web 2.0 at http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html. I strongly encourage everyone to read this short paper. Kevin Lynch and many others have also been talking a lot recently about Web 2.0, which, in extremely simple terms, is all about smart clients and dynamic and shared data-driven, end-user centric experiences. What does that mean? It means that the next generation of web applications is going to be not just browser, but also platform independent. It means that these applications are going to seamlessly exchange data and services with one another. It means that the focal point of the user interface is on the user – not just by giving them an easy-to-use interface but by giving them a smart interface… a sexy client interface that knows what they're looking for before they're done describing it.

There are currently two technologies that allow these types of applications to be delivered: AJAX and Flash. Contrary to what many of you might think, AJAX and Flash are developer technologies, not designer technologies, and they do little, in the context of Web 2.0, without server-side business logic and services (i.e., ColdFusion) to talk to. AJAX is very cool but certainly has limitations that Flash does not. I see future issues of *CFDJ* with some articles on AJAX (possibly on Microsoft Sparkle if and when it's released) and much more focus on Flash.

"There has also been a surge in open source projects written in ColdFusion"

Before anyone gets bent out of shape by the prospect of the inclusion of many Flash articles in *CFDJ* or by my statement that Flash is for developers, let me qualify my statements. When I say "Flash" I refer to the Flash platform, not the tool. At the recently held Macromedia MAX 2005 conference, Macromedia announced the availability of public alpha versions of Flex Builder 2.0 – and the Flex 2.0 platform. Finally, we have a development environment and language that allows developers, not designers, to create amazing Flash applications and front ends to their ColdFusion applications. Flex 2.0 does still have a server component for enterprise customers who need it, but most CF developers are going to want to get their hands on the Flex Builder 2 IDE, which, like the traditional Flash IDE, compiles code into a SWF that must then be uploaded to your server, but like Flex uses ActionScipt (3) and MXML, a tag-based language, for authoring. No timelines, no animations. Only interfaces, effects, services, and business logic. It's the most developer-friendly environment you're likely ever to see – MXML is a language that even encourages Model View Controller design pattern use. Flex 2.0 even has a special ColdFusion adapter to allow Flex applications to interact with a ColdFusion Server via AMF or SOAP. With a very low price tag and ease of use for developers, there's really no excuse not to be building better experiences for your end users anymore. This is not only a trend I believe is coming, but one that I feel I'd be irresponsible not to promote in *CFDJ*.

Macromedia has also displayed a strong show of support for the open exchange of data, ideas, and applications. At MAX they launched Macromedia Labs (http://labs.macromedia.com) – their public alpha and beta site. That's right, some products are going to leap straight from R&D to the public for testing and sneak peek usage. You can go to Macromedia Labs today and download Flex 2.0 now. It's an Eclipse IDE plug-in (showing yet more support for open source initiatives). Speaking of Eclipse and Macromedia support for open source initiatives, at the MAX conference the ColdFusion Team sneak peek was RDS support and a query builder built for the CFEclipse project!

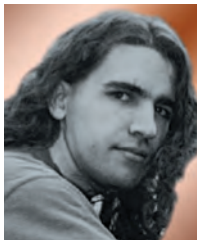
About the Author

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

Consuming Amazon.com E-Commerce

An overview



By Jeff Houser

One of the most popular articles I've ever written was on Amazon.com Web services. The article was written for the Macromedia Developer Center and is located at http://www.macromedia.com/devnet/coldfusion/articles/wsamazon.html.

This article is a bit dated. A few days before it was published, Amazon.com released version 1 of their SOAP Web services. In this issue on exchanging data, I thought it'd be great to write an article on version 4 of Amazon.com's SOAP interface, released in September. They are now called "Amazon.com E-Commerce Services."

After two days of much frustration, I was unable to successfully invoke the SOAP Web services from ColdFusion or BlueDragon; I kept getting semantic errors when compiling the WSDL. However, you can still use Amazon.com Web services without SOAP, using REST instead, which is a SOAP alternative. In this article, I'll introduce you to Web services and show you everything you need to know to implement Amazon.com's E-commerce services on your site.

Web Service Definitions

Before delving into Amazon.com specifics, I want to make sure we're all on the same page when it comes to terminology. If you're already familiar with Web services, you can probably skip this section. If you are new, or want a reminder primer, read on.

Most of the time when you hear someone talk about Web services, they're talking about SOAP. SOAP stands for Simple Object Access Protocol, an XML dialect that is used to define how Web services can communicate with the programming language calling it. Not all Web services are SOAP, though. A Web service is defined as any program run over a network by another program. If you've ever used cfhttp to process a credit card transaction through a gateway provider, you've used a non-SOAP–based Web service. These types of requests are sometimes called REST requests. REST stands for Representational State Transfer. Instead of passing an XML-based SOAP request, REST is done by using a variety of URL parameters to specify which methods to invoke and which values to pass in. More information on REST can be found at http://www.ics.uci.edu/%7Efielding/pubs/dissertation/rest_arch_style.htm.

The next definition is the WSDL, the Web Services Definition Language. It defines the functionality of a SOAP Web service. WSDL is another XML variant. The WSDL will tell you the operations you can perform against the Web service and what you would expect to get in return for that service. In many cases, you'll be provided with English documentation (such as a Word doc, PDF, or HTML) for the Web services and won't have to read the WSDL manually. If you think of a Web service as a CFC, the WSDL will define the names of methods, the input parameters to those methods, and the type of data that is returned from those methods. If you create a Web service in ColdFusion, the WSDL is generated automatically for you, just by appending "?wsdl" to the URL. In ColdFusion, you'll need to know the location of the WSDL is to create a Web service object.

There are two other definitions that you can use when speaking about Web services. The first is to publish a Web service. When you publish a Web service, you're making it available for people to use. If the Web service is public, such as Google or Amazon's Web services, anyone can use it. If the Web service is private, you may only be providing access to clients, customers, or other departments in your company. The act of publishing is merely telling someone the URL and providing the documentation on how to use the Web service.

On the other side of the coin, Web services are consumed. To consume a Web service means that you are using a Web service. It's as simple as that. Amazon.com published their Web services by making them public. As the one using the Web service, we are consuming them. Publish always seemed like a logical name for making Web services available, but it took me a while to get used to "consuming" them.

Getting Started with Amazon.com Web Services

Why would you want to use Amazon.com Web services?

JAVA J2EE-Ready Managed Dedicated Hosting Plans:

Xeon I

SAMEDAY SETUP

Dual 2.8 GHz Xeons
2GB RAM
Dual 73GB SCSI
1U Server
Firewall
Linux
Monitoring
NGASI Manager

$279 *monthly*

Pentium 4 I

SAMEDAY SETUP
FREE SETUP

2.4 GHz P4
2GB RAM
Dual 80GB ATA
1U Server
Firewall
Linux
Monitoring
NGASI Manager

$199 *monthly*
2nd month FREE

4Balance I

1 Database Server and 2 Application Servers connected to 1 dedicated load balancing device.
Dual Xeons.
High-Availability.

$1724 *monthly*

At **WebAppCabaret** we specialize in **JAVA J2EE Hosting**, featuring **managed dedicated servers** preloaded with most open source JAVA technologies.

PRELOADED WITH:
JDK1.4 . JDK1.5 . Tomcat . JBoss . Struts . ANT . Spring . Hibernate
Apache . MySQL . PostgreSQL . Portals . CRM . CMS . Blogs . Frameworks
All easily manage via a web based control panel.

Details:
-All Servers installed with the latest Enterprise Linux
-Firewall Protection
-Up to 60 GB daily on site backup included at no extra charge per server.
-Database on site backup every 2 hours
-Daily off site database backup
-A spare server is always available in case one of the server goes down
-Intrusion detection.
-24x7 Server and application monitoring with automatic self healing
-The Latest Bug fixes and Security updates.
-Tier 1 Data Center. 100% Network Uptime Guarantee
-Guaranteed Reliability backed by industry-leading Service Level Agreements

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

NGASI POWERED

WebAppCabaret sm

JAVA J2EE Hosting

Perhaps you're an associate and want to bump up your potential for income by allowing people to search the full Amazon catalog. Perhaps you're a merchant who is selling a product through the Amazon.com store. The Amazon.com e-commerce services offer many features, including:

- ***Product Searches:*** If you take away its product catalog, there isn't much to the Amazon.com Web site. You have the full array of search options available through the Web service interface that you do via the Amazon.com Web site. After a search, you can always get full details on an individual product.
- ***Product Images:*** Image data is returned as part of the search.
- ***Customer Reviews:*** You have full access to the customer reviews for any given product.
- ***Wish List Searches:*** You can search wish lists by name, e-mail address, city, or state using the Web service interface.
- ***Remote Shopping Cart:*** You can now have your users add items to their Amazon.com shopping cart without leaving your own site.

To get started setting up Amazon.com Web services on your site, first register for a subscription ID. You can do so at the Amazon.com site: http://www.amazon.com/gp/browse.html/104-7075718-2365513?node=3435361&/104-7075718-2365513. If you're not an Amazon.com associate (but want to be), you can join the associate program here: http://www.amazon.com/gp/browse.html/ref=gw_bt_as/104-7075718-2365513?node=3435371.

Deconstructing the Amazon.com URL

You're ready to perform your search. The first step is to construct your URL. The URL starts with the location http://webservices.amazon.com/onca/xml. It is the query string parameters that provide us with the search results:

- ***Service:*** The service parameter accepts the value of AWSECommerceService to access Amazon.com E-commerce services. It is required.
- ***SubscriptionId:*** The subscriptionID was provided to you when you signed up for the developer program. This is also a required parameter.
- ***Operation:*** The operation parameter describes what operation you are trying to perform. There are 18 different actions that you can perform. In this article I'll show you ItemSearch and ItemLookup. You can also look up seller info, perform similarity searches, add information to the user's Amazon.com shopping cart, or look up public customer information (such as reviews or wishlists). This is required.
- ***AssociateTag:*** If you are an Amazon.com associate, put your associate tag here to ensures that you keep getting the kickbacks when people buy something based on your referral. This is optional.
- ***ResponseGroup:*** The responsegroup parameter is used to define the type of information that is sent back to you
- ***Operation Parameters:*** The operation parameters are a set of parameters required for each operation. You might think of this as similar to the way that cffile works. You use a different set of parameters to upload a file than you do to copy a file, move a file, and so on. Other tags, such as cfobject and cfloop, operate in a similar manner.

Note that all of these parameters are case sensitive.

Searching the Store

Now you want to search the Amazon.com store. The operation that you want to perform on the store is ItemSearch. These are some ItemSearch-specific parameters:

- ***SearchIndex:*** The search index specifies the name of the Amazon.com store index that you are searching. Available indexes are locale specific. Some common ones are Books, Music, DVD, and Software. A complete list is available in the Amazon.com DVD. If you set this value to blended, all indexes will be searched. This is required.
- ***Keywords:*** The keyword search will compare your search term against various product fields, including (but not limited to) title, author, artist, and description.
- ***BrowseNode:*** The browsenode parameter is used to narrow your search to a specific category of products in the Amazon.com catalog. You can think of browse nodes as subcategories within each SearchIndex. For example, to find ColdFusion books, you would probably search in Professional & Technical. This is an optional attribute.
- ***Availability:*** This is an optional parameter that specifies that you want to only search for items that are available. If specified, you want to give it the value of Available.
- ***Title:*** The title parameter is optional and used to search for a specific title. In addition to the title, you can search using Author, Publisher, Artist, Actor, Director, Manufacturer, MusicLabel, Brand, and a whole slew of others. Check the documentation for the complete list.
- ***MinimumPrice:*** The minimum price is used to specify a price range. Items that are priced lower than this value will not be returned. Its parallel MaximumPrice can be used to specify an upper limit, where items with prices higher than this value will not be returned. If either is left out, there is no lower or upper limit, respectively.
- ***Condition:*** The condition parameter is used to filter the list using the specified value. Valid values are All, New, Used, Refurbished, or Collectible.
- ***Sort:*** Specifies the sort order of the result, similar to the "Order By" clause in SQL. Its values are dependent upon the locale; in the U.S. you can sort using these values: daterank, relevancerank, pricerank, -titlerank, inverse-pricerank, and salesrank. The minus uses a descending sort, where applicable.
- ***ItemPage:*** ItemPage is used to paginate the results. The search will return only 10 results. You can use ItemPage to specify which page of search results you want. The default is to return the top 10 results. But, if you specify two, you'll get results 11–20; if you specify 100, you'll get results 100–110, and so on.
- ***ResponseGroup:*** The ResponseGroup parameter is intended to provide greater flexibility on the type of data

Different people. Different needs. One suite solution.

With the latest versions of Macromedia Dreamweaver®, Flash® Professional, Fireworks®, Contribute™, and FlashPaper™, the new Studio 8 is quite a catch. To meet Studio 8 and find all the web design and development tools you need, visit www.macromedia.com/go/8_studio8.

macromedia®
STUDIO 8

that is returned. In previous versions of Web services, you could only retrieve data in two formats: light (some data) or heavy (all data). Response Groups are designed to give you much flexibility in what is returned. If you wanted to perform a search that only returned image data and the title, then response groups are for you.

Okay, you're digesting a lot of information. Let's see some code:

```
<cfhttp url="http://webservices.amazon.com/
onca/xml?Service=AWSECommerceService&Subscripti
onId=#variables.SubscriptionID#&Operation=ItemS
earch&SearchIndex=Books&Keywords=ColdFusion&Aut
hor=Houser" method="get">
</cfhttp>

<cfdump var="#XMLParse(cfhttp.FileContent)#">
```

To figure out what's going on, you'll need to dissect the URL. You know the service value is constant. The subscriptionID was provided by Amazon.com (in this case, the value is stored in the variables scope). The operation is ItemSearch. The SearchIndex is Books, which means we are searching the Amazon book catalog. The keyword that the code searches for is ColdFusion. The author is Houser (me). The final line of the code dumps the results of our query string. Since cfhttp returns a string, before dumping, the XMLParse function is used. It makes the dump look nicer. After all the preparation, the retrieval is really that simple. You can easily make a form to allow the user to enter parameters as necessary.

Retrieving One Item with ItemLookup

You're going to process the search results and your user will want to get more details on an item. You can get them using an ItemLookup. These are some common parameters:

- ***ItemId:*** It specifies the product ID of the value you want to look up. If you want to retrieve multiple IDs, you can do so by adding a comma-separated list of the products you want to return. Most likely you'll use an ASIN number (Amazon's internal ID) to get information on the specific product, but other options are available such as SKU and UPC.
- ***IdType:*** This parameter is used to specify which type of value is contained in the ItemId parameter. The default value is ASIN, but you can also specify SKU, UPC, or EAN.
- ***SearchIndex***: It specifies which amazon.com store you are searching for your item in. If you use the ASIN, this is ignored. If you are performing an alternate search, this is required. It operates the same way as SearchIndex does for the ItemSearch.
- ***ResponseGroups:*** Response groups here act the same way that response groups do in an ItemSearch. In the following example, I actually used a simple response group to retrieve more data than just title and author.

```
<cfhttp url="http://webservices.amazon.com/
onca/xml?Service=AWSECommerceService&Subscripti
onId=#variables.SubscriptionID#&Operation=ItemL
ookup&ItemId=0072132388&ResponseGroup=Request,L
arge" method="get">
</cfhttp>

<cfdump var="#XMLParse(cfhttp.FileContent)#">
```

You should already be familiar with the Service and SubscriptionId. The operation is ItemLookup to retrieve an item. The ItemId contains the ASIN of the value to retrieve. The ResponseGroup is set to large, which returns a multitude of information about the product including image locations, customer reviews, lists that include the book, price, and just about anything you'd want to know about the book.

Where to Go from Here

I've showed you how to get and receive data from the Amazon.com E-commerce interface using ColdFusion. Unfortunately, due to space constraints, there is barely time to scratch the surface of the full scope of functionality. You can examine Amazon.com's documentation, located at http://www.amazon.com/gp/browse.html/ref=sc_fe_l_0_3435361_3/104-7075718-2365513?%5Fencoding=UTF8&node=3487571&no=3435361&me=A36L942TSJ2AJA.

To give you an idea of the full breadth of the service suite, the PDF document is over 450 pages long. You may also look at the Amazon Web service blog at http://aws.typepad.com/ or talk with other users at the discussion forum (http://forums.prospero.com/am-assocdevxml).

This article didn't discuss what to do once you get the XML data returned. You may want to examine ColdFusion's XML handling, so you can actually do something with the data once you have it. You might start with the July 2005 issue of *CFDJ* (Vol. 7, issue 7); most of the articles in that issue focused on XML, including an XML prep in this column. Neat things can be done with XSL and the Web service suite.

Finally, on a personal note, I've officially entered the world of blogging (thanks Ray). Check out my personal site at www.jeffryhouser.com for my ramblings on ColdFusion, business, and a spattering of random thoughts. Until next time!

"A Web service is defined as any program run over a network by another program"

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

Development Perspectives

Differences between consulting and product development

By Brandon Harper

With the proliferation of many respected developers in the ColdFusion community sharing their experiences and knowledge by blogging, we've seen quite a huge jump in both information sharing as well as the discussion of best practices in software development.

These increases in enlightenment also started happening around the time that ColdFusion MX 6.0 was released, which brought a pseudo object-oriented shift to programming in ColdFusion that has greatly influenced the direction of the average CF developer. Who would have thought so many people would be fluent in design patterns and object-oriented terminology four years ago, and that ColdFusion job postings would actually specify this sort of experience?

Given the shift toward blogging as an increasing venue of communication, we've witnessed a fairly powerful community starting to build outside of mailing lists and message forums, previously the places you would most likely encounter those serious about their craft. This was more evident than ever to me when I attended CFUNITED this summer. It was my first big conference (outside of the ones that have been held in Denver), and it was an interesting experience for me. I did my best to introduce myself to people I recognized; while most of them had no idea who I was by name, I mentioned my Web site and the majority of people had a much better idea who I was, especially if they were active bloggers. I thought this was an interesting concept – to be known more for a URL than my actual name, and it demonstrated to me how powerful it is to have your own very small piece of a community.

With an online community comes both good and bad. One of the good things is the availability of so much information, as well as being able to interact directly with so many other people who are intimately involved in ColdFusion. This can also be bad as it's easier to be rude and contriving when posting to a mailing list or a blog rather than saying it to someone in person, so you must think twice about responding tactfully during a heated debate.

Earlier today when I read another great blog post from Joe Rinehart on the recent "Frameworks versus Methodology" debate that has been happening between a few popular ColdFusion blogs, it reminded me of something I've been meaning to write about in my blog, but it seemed more suitable for a long article. A point that he touches on in his entry that I think should be expanded on a bit is the consultant versus product development mindset.

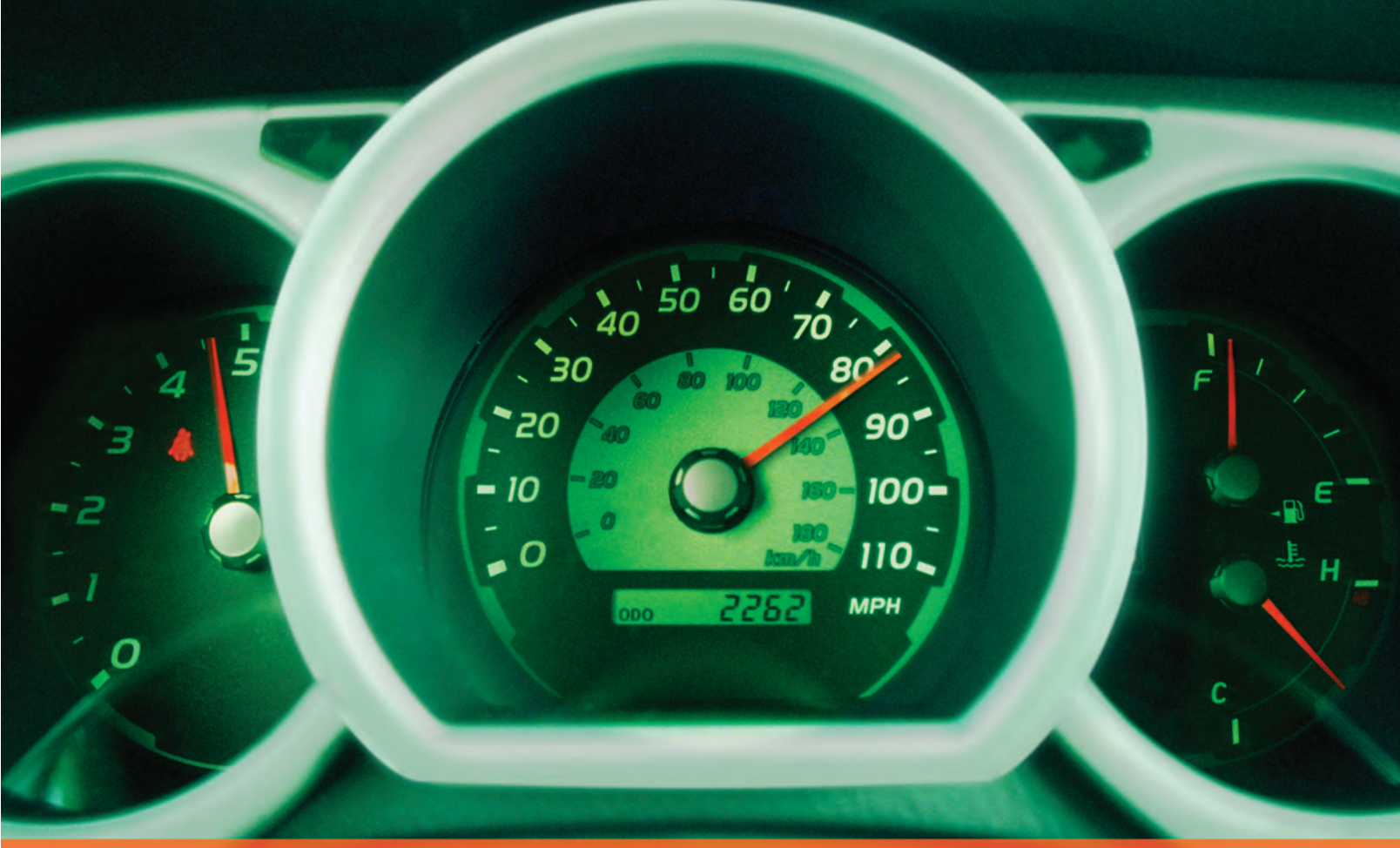
To preface this, I started off the early part of my career working in the consulting world by primarily building Web sites for companies via an ISP, followed by a branding agency for the first three years, and then I made the switch to developing products for companies about four years ago.

One thing I love about product development is that your budget for a given product at its simplest form is dictated by time rather than money. Basically, as a developer, this equates to quite a few things, but the ones off the top of my head are:

- The cool features that are fun to work on (that clients usually won't pay for in a consulting role) do eventually happen.
- Being able to spend time for requirements gathering, modeling, and other general R&D for a new initiative without trying to sell why this must be done to a client.
- In theory, you have the chance to build a mature, scalable, adaptable code base over time and even get a chance to refactor things every now and then rather than hitting a ceiling for billable hours.
- Typically the functionality is more important and more complex to write. In other words, the product has to do something to actually make money, not just have a pretty face to sell things to people.

That said, there are some things to miss about doing consulting type work for sure:

- Working with a diverse client base on a variety of high profile Web sites.
- In general, you work with more stable, predictable technology. Generally content-based sites, intranets, and e-commerce–based sites are fairly easy to scale rather than an internal enterprise integration type project.



Rev Up Your Flash Video

Stay Ahead of the Competition With VitalStream and the Enhanced Video Features in Flash 8

With over two years of experience in delivering much of today's most popular media, VitalStream® is the first and most experienced Flash™ video streaming service provider.

Enhanced Flash 8 Video Features:

- New VP6 codec delivers higher quality video at the same bit rate
- 8-bit alpha channel transparency enables you to blend video with other elements
- Improved live video capabilities

VitalStream Complete Toolset for Flash:

- MediaConsole®
- MediaOps™ SDK
- Flash Authentication
- Reporting Dashboard

Integrate Streaming Media Into Your Flash Projects

Take Advantage of the Enhanced Video Features in Macromedia Flash 8
Call (800) 254-7554 or Download Tutorials at www.vitalstream.com/go/mxdj

macromedia®
FLASH VIDEO
STREAMING SERVICE

Call (800) 254-7554
Visit www.vitalstream.com

VitalStream®

- Getting to start projects from scratch a lot of the time rather than reusing an old crufty code base, which looks more like COBOL than ColdFusion. Or in the case where you are reusing a lot of code, chances are you already have some sort of an established framework that works well and only needs to be customized each time you do a new site.
- Getting to work with cutting-edge, client-side technologies such as Flex.

One of the main differences that separate a typical developer serving in a consulting role versus a developer in a product development role is the fact that for the consultant, time is money. Ultimately the end goal is to finish a project within the time-frame and price point that the customer is looking for. On the other hand, a typical product developer is more adherent to process and is more likely to follow the various phases in the software development life cycle, especially as the team and company gets bigger. Time is money for them as well, but in a much different fashion; because they have already been budgeted into a position, the resource that they provide is development time, so they have to work more under time constraints than billing-hour constraints.

When I worked as a consultant and was much less experienced, my process to start and complete a project was something like this:

1. Gather requirements
2. Design the database to support the requirements
3. Code!

As I've learned over time, this is usually a recipe for certain disaster for many reasons that is a whole article in itself. However, this was before I started my computer science degree or had any exposure to traditional software development, and it seemed to be the right thing to do when working under billable hour limits, given my background and experience.

With more experience and mistakes to learn from, I grew to at least start using my own framework, which was akin to what Fusebox 3 ended up being. It seemed to work well at the time, but I'd always have to bring other developers up to speed about how it worked and make changes to their code if it didn't fit the "standards" of my framework. In general, it was fine if I were the only one working on a given product, but it would get in the way if someone were to work on something with me.

From my experience during the past several years in product development, the paradigm shift to OO development with the advent of ColdFusion Components, as well as completing the majority of my computer science classes, I've learned a great deal about subjects such as object-oriented programming, the software development life cycle, UML, and design patterns. Let's just say that the aforementioned three steps do not reflect the process I use today.

Generally speaking, the two most important reasons for going through the various phases in the SDLC and using proper OO design is to make sure that the resulting product does what it's supposed to, as well as mitigating the risk of the project failing to meet its objectives. Some great side benefits to this is that it gives a developer time to solve most of the business problems upfront, planning out how the objects will communicate with each other, and lays a good foundation for flexibility and expandability in the future. Specifically, it allows developers time to make sure that they design with object reuse in mind – ensuring that objects are highly cohesive, that there is a reasonable amount of coupling, and that the resulting classes fulfill all of the use cases that were specified during requirements gathering.

Just as a consultant can get too wrapped up in the minute details or a disdain of frameworks, product developers can also get hung-up "in the clouds" trying to perfect their UML with design patterns and making sure the various relations between objects are correct, and overall just being too obsessive-compulsive about the design of their application. I've certainly been guilty of this as I hate to start coding something that is not perfect because I'm very detail-oriented. However, as long as you are doing some kind of iterative development, you can always make time in the future to refactor something that doesn't seem to fit correctly. Generally I've found that it's better to take a philosophical approach and try to find the middle ground between perfection and chaos and fix the problems in later iterations as it's much better to get some code done instead of worrying about what pattern to use for a certain problem.

I'm sure you're probably asking yourself something along the lines of, "Well, that sounds great, but what does this have to do with frameworks and the differences between consulting and product development?" right now. To answer your question briefly, both types of developers usually approach problems differently but are ultimately going for the same end result: making sure the product does what it's supposed to and making sure our project stakeholders are impressed. By impressed, I mean floored enough with our results to give us fame, fortune, lots of beer, and a 30" Apple Cinema monitor. Okay, so none of these things have ever happened for me after a successful product delivery other than a celebratory round of drinks, but I haven't been holding my breath either.

In Summary

Consultants are generally trying to get things done in as few hours as possible, and product developers are trying to

"The main point I'm trying to get across is that all developers have varying backgrounds, skill levels, situations, attention to detail, and end goals"

make sure that they build a robust code base that they can add additional features to in the future, because they will never find themselves with enough time to implement every request during the first major version of a product.

However, the main point I'm trying to get across is that all developers have varying backgrounds, skill levels, situations, attention to detail, and end goals. As such, perspectives and solutions to various programming problems are always going to be different for each developer. I've been witness to some very bad code over the years (including my own), yet said code accomplished the end goal and the stakeholders in the project were very pleased with the results.

As much of a stickler that I can be at times about standards-based development and using existing frameworks as well as good OO design, I also try my best to keep in mind that there is always someone out there with a better idea or different perspective. Ultimately I'd hope that they'd see where I was going with my process and solution, and that I need to make sure to put myself in the same position if I truly think my way of doing things is much better than someone else's.

The best thing you can do in any situation of contention is participate in a friendly but logical debate using correct terminology (while admitting if you don't fully understand a term) with the end goal of learning something from it without taking it personally. If you find yourself being in the wrong at the conclusion of a good debate, just remember that humility is not a bad thing and we all make mistakes – it's one of the important characteristics that separates a good developer from a great one. Admitting when you're wrong will get you more respect than being stubborn and defensive all because you didn't want to look like someone knew something that you didn't.

In closing, keep up the good work ColdFusion bloggers! I think we've all learned a lot from each other in the past few years. Though our opinions on many topics may differ at times, it's always good to have opposing viewpoints and try to challenge the norm a bit. Whether it's those shoot-from-the-hip cowboy consultants or those dang pattern-and-framework-obsessed product developers, we stand to improve our understanding and skill level by openly acknowledging each others' viewpoints. Maybe we can even work on some of the great community projects together, including the bloated/incredible frameworks like Fusebox, Mach II, and Model-Glue.

About the Author

Brandon Harper is a senior software developer at InsightAmerica, an Acxiom company, providing risk mitigation technology to Fortune 500 companies, small businesses, and individuals using analytically enhanced data. Brandon has been working with computers for over 20 years, developing for the Web for 10, and has been implementing solutions using ColdFusion since 1998. He was also a technical editor for Inside ColdFusion MX, and maintains a blog at devnulled.com.

brandonh@gmail.com

Leveraging on Active Directory for Users Administration and Security Profiling

Make life easier for the users

By Adedeji Olowe

The majority of ColdFusion applications live far away, hidden, in enterprise fortresses as applications that small-to-large organizations depend on. In these organizations, especially the medium-to-large ones, there are well-established network infrastructures to manage the users, workstations, servers, etc.

Organizations usually implement LDAP as a directory services infrastructure but, for the purposes of this article, I will only be discussing Active Directory.

Active Directory, AD, is an LDAP implementation from Microsoft that was introduced with the Windows 2000 environment. This implementation is based on the X.500 LDAP standards. The AD is a giant database that can store as much as 16 terabytes or 1 billion objects ranging from users, printer locations, security policies, and, also important, user-defined data.

Every application that interacts with users usually has restrictions based on profiles and must also implement security. These are especially important for applications in financial organizations. Based on my experience, building user and profile management into applications, which is not a trivial matter, take considerable time and effort in the software development cycle. It doesn't stop with this; users don't like to have multiple security credentials across many applications. There is nothing more frustrating for these users than having to remember which username and password work with which application.

One of the beauties of Microsoft's Web-based/enabled applications is the ease at which they plug in to its existing network infrastructure. Outlook Web Access (OWA) users use their network security credentials to log in. Not only that, OWA knows when a user is logged into a computer, so it automatically loads the user's profile from the AD.

ColdFusion has support for LDAP, which includes AD. Using CFLAP, the ColdFusion tag for interacting with LDAP servers, you could leverage on AD for user management and profiling.

In the next few paragraphs, I'll show how you can use CFLAP to authenticate and load user profiles in AD for use in your application.

Step 1: Preparation

For the sake of this article, imagine you are writing an application for a financial institution called Bank *X* Inc. Bank *X* has implemented its AD as bankx.com. To write your application, you must know the name or IP of an AD server (Domain Controller, DC) that will authenticate your users. You must also know the structure of your AD. Please consult with your Domain Administrator for documentations.

Step 2: Login

The following is a simple login form to be used by Bank *X* users to log in to their financial applications:

```
<form action="loginADUser.cfm" method="post">
  <label for="username" accesskey="u">Username:</label>
  <input name="username" type="text" id="username" />
  <br />
  <label for="password" accesskey="p">Password</label>
  :
  <input name="password" type="password" id="password" />
  <br />
  <input name="Submit" type="submit" value="Submit" />
  <br />
</form>
```

Step 3: Authentication

The loginADUser.cfm page contains code to authenticate the user against the AD server (see Listing 1).

Explanation

In AD, any valid user can bind to the service, which is accessible on port 389 and port 636 for a secured connection. For more on secured connections, please see the security section. The bind will work only if the security credentials are correct but will throw an error if the credentials are wrong.

The **isLoggedIn** variable holds a Boolean value that determines if a user's security credentials are valid on the domain or not. Now, when authentication is attempted, the try-catch combination catches the error that is thrown with wrong security credentials.

The code in Listing 1 kills two birds with one stone by authenticating and retrieving certain records at the same time.

A user's groups are stored in the **memberof** field. The membership information is stored in DN form, which you may have to parse to extract out. It's usually in this form:

```
CN=Support Team,OU=Distribution List,DC=bankx,DC=com,
CN=InfoTech,OU=Distribution List,DC=bankx,DC=com,
CN=Administrators,CN=Builtin,DC=bankx,DC=com,
CN=Domain Admins,CN=Builtin,DC=bankx,DC=com
```

The login code can be wrapped with a CFLOGIN tag and the parsed roles passed to CFLOGINUSER as roles. Otherwise, you can implement your own role system with session management.

Extending the AD

The AD, implementing the LDAP specifications, has default fields that hold information that is important to your users and applications. However, you might want to store some application-specific data along with the default information. AD has 15 blank fields that you can use: **extensionAttribute1** to **extensionAttribute15**. You can also use some fields that are not important to your organization such as **IPPhone**.

If you use these fields, documentation of what you have done is very important. Also, note that the AD can be delicate; kindly consult with your Domain Administrator before writing anything to the database. An error could bring down the whole AD forest.

Tools

You can't do much with AD without using the ADSI Edit. Install this from the Windows Support Tools. The ADSI Edit allows you to peruse the attributes and data types of the objects in the AD. The AD has data types such as Integer, Integer8, DN, OID, Boolean, DirectoryString, and PrintableString. Consult documentations for what these stand for.

Security on AD/CF Integration

Implementing a real-life application requires that transmission of sensitive data between the servers should be encrypted. The AD supports the interchange of data between ColdFusion and itself via the Secured Socket Layer, SSL on port 636.

To use SSL, you must install an Enterprise Certificate Authority on any of the domain controllers in your organization. This forces the DCs to request certificates from ColdFusion whenever you use CFLDAP.

The next step is to install your security certificate on the ColdFusion server using the keytool. Go to the command prompt and navigate to <cfroot_install>\runtime\jre\bin directory and run the following command:

```
keytool -import -keystore cacerts -alias ldap -file ldap.crt -keypass
bl19mq
```

Please refer to the Sun JDK for full documentation.

With the certificates in place, you must add the secure = "CFSSL_BASIC" attribute to your CFLDAP.

Summary

Your application can be much more elegant if you leverage on the AD. Not only will you deliver faster, you also future-proof your application so that it can effectively connect to other sources of user security profiles.

Ultimately, the users find life easier if they can always use your applications with just a single set of universal security credentials.

Important Links

- Securing AD Servers: http://support.microsoft.com/kb/q247078/, http://livedocs.macromedia.com/coldfusion/6.1/htmldocs/ldap28.htm
- Macromedia Tutorial on LDAP: http://www.macromedia.com/devnet/server_archive/articles/integrating_cf_apps_w_ms_active_directory.html
- Active Directory Services Interface: http://www.microsoft.com/windows2000/techinfo/howitworks/activedirectory/adsilinks.asp

About the Author

Adedeji Olowe is a ColdFusion Developer.

adedeji@olowe.com

Listing 1

```
<!---a boolean variable to hold the logged in state of the user--->
<cfset isLoggedIn = false>
<cfset baseDomain = '@bankx.com'>
<cfset domainController = 'server_IP_address_server_name'>

<!---perform the authentication via CFLAP. Store the returned query in
ADResult--->
<cftry>
<cfldap action="Query"
    name="ADResult"
    attributes="cn,mail,displayname,dn,memberof,extensionAttribute1,empl
oyeeID,sAMAccountName"
    start="cn=users,dc=bankx,dc=com"
    filter="(&(objectclass=user)(samaccountname=#ReplaceNoCase(form.
username, baseDomain,'')#))"
    server="#domainController#"
    scope = "subtree"
    username="#form.username#"
    password="#form.password#"
>
<cfset isLoggedIn = true>
<cfcatch>
    <cfset isLoggedIn = false>
</cfcatch>
</cftry>
```

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

ENGAGE AND EXPLORE...

The Technologies, Solutions and Applications that are Driving Today's Initiatives and Strategies...

CALL FOR PAPERS NOW OPEN!

SOA WebServices Edge 10th International conference+expo 06

The Sixth Annual SOA Web Services Edge 2006 East - International Web Services Conference & Expo, to be held June 2006, announces that its Call for Papers is now open. Topics include all aspects of Web services and Service-Oriented Architecture

Suggested topics...

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

CALL FOR PAPERS NOW OPEN!

2006 ENTERPRISE> OPENSOURCE CONFERENCE+EXPO

The first annual Enterprise Open Source Conference & Expo announces that its Call for Papers is now open. Topics include all aspects of Open Source technology. The Enterprise Open Source Conference & Expo is a software development and management conference addressing the emerging technologies, tools and strategies surrounding the development of open source software. We invite you to submit a proposal to present in the following topics. Case studies, tools, best practices, development, security, deployment, performance, challenges, application management, strategies and integration.

Suggested topics...

- *Open Source Licenses*
- *Open Source & E-Mail*
- *Databases*
- *ROI Case Studies*
- *Open Source ERP & CRM*
- *Open-Source SIP*
- *Testing*
- *LAMP Technologies*
- *Open Source on the Desktop*
- *Open Source & Sarbanes-Oxley*
- *IP Management*

Submit Your Topic Today! www2.sys-con.com/events

Sponsored by

WebServices JOURNAL · XML JOURNAL · .NET JOURNAL · eclipse developer's journal · WebSphere JOURNAL · information STORAGE+SECURITY journal · wldj · JDJ · LINUXWORLD MAGAZINE · MX developer's journal · asp.netPRO · SD Times · CoDe · Software Test & Performance

*Call for Papers email: jimh@sys-con.com

Attention Exhibitors:
An Exhibit-Forum will display leading Web services and OpenSource products, services, and solutions

For Exhibit and Sponsorship Information – Call 201 802-3066

Produced by SYS-CON EVENTS

Handling Site Layouts with Layout Components

A flexible way to handle layouts for your sites

By Steve Bryant

Most sites have similar issues related to the site layout. I have seen many solutions that solve some challenges related to layout, but fail to handle others elegantly. I have found that using layout components elegantly solves all of the layout problems I have faced.

What Do I Want?

Here are my goals for handling layouts. I want to:

- Make edits to the layout in one place
- Easily switch layouts (per page, section, etc.)
- Switch to non-HTML formats (PDF, Word, etc.)
- Use any ColdFusion tags (like cfflush)
- Do without mappings and relative paths
- Add HTML to the <head>
- See all of the layout in one file
- Avoid passing all page calls through one file (like index.cfm)

Here is a quick example of switching to non-HTML formats:

- HTML: http://www.bryantwebconsulting.com/cfcs/
- PDF: http://www.bryantwebconsulting.com/cfcs/?format=PDF
- Word: http://www.bryantwebconsulting.com/cfcs/?format=Word
- Excel: http://www.bryantwebconsulting.com/cfcs/?format=Excel

I didn't have to switch the format using the URL, but it made a handy example. This makes "download to Excel" functionality a breeze.

As you will see, layout components also make it easy to use a different layout on your home page than on other pages. Similarly, the login page for your admin section can use your default site layout (instead of the admin layout).

Include Interlude

Before I discuss layout components, let me explain why I don't use more "obvious" techniques.

Specifically, I recommend you avoid the temptation to put output code in Application.cfm (and OnRequestEnd.cfm). When a page needs an alternate layout (a pop-up, home page, etc) then you have to add an exception to Application.cfm, which can lead to unwieldy code.

For instance, some programmers use a different layout based on the filename being requested (e.g., "index.cfm" gets a different layout from the rest). This is problematic when you

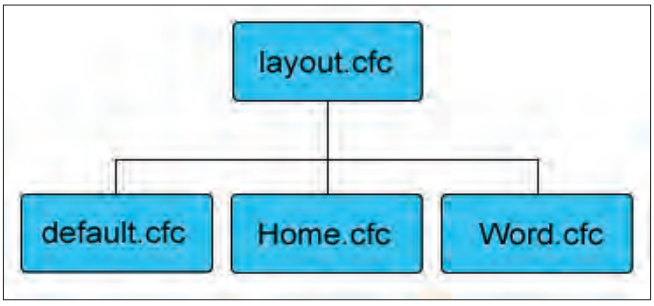


Figure 1

or your client wants to make a quick copy ("index2.cfm") to test a change.

While you could determine layout using the directory structure, this is also limiting. A home page may need a different layout than the rest of the site or the client might decide to change the layout of any page. You should be able to change a layout without moving the page (lest you jeopardize links and bookmarks).

I therefore recommend that you call output code from your page (whether you use layout components or not) to ensure this flexibility. Using layout components doesn't prevent you from including your output from Application.cfm, but I highly recommend that you include your output from your page.

Let's Take a Look

I frequently need to add code to the <head> of just one page. Using <cfhtmlhead> works, but seems clunky. I would prefer code as shown in Listing 1. My home page, using a different layout, might look like Listing 2.

In any event, I must first initialize the layout component:

```
<cfset layout = CreateObject("component","layouts.default").init()>
```

This instantiates a layout using layouts/default.cfc (from the root of your site). This can be called from Application.cfm (creating the layout variable doesn't cause any output).

The code for default.cfc is in Listing 3. The code for Home.cfc (the layout used on our home page) would be the same with slightly different HTML.

As you can see, a layout component is just a basic HTML page with some component tags in the mix. Within a week of working layout components, I learned to ignore the cffunction and cfcomponent tags and just see the HTML. You can see the resulting HTML for Listing 1 in Listing 4.

How Does It Work?

As I am sure you can tell by looking at the component code and the resulting HTML, each method of the layout component is called in turn and outputs HTML into the page.

Normally, a CFC shouldn't output anything directly. In this case, however, the whole point of a layout component is output. So, the output of each cffunction is "yes" (where it would normally be "no").

It is important that </head> is at the top of the body() method. This allows me to put HTML into the head of the document by placing the code between layout.head() and layout.body().

Of course, we still need to cover how to switch layouts…

Switch Layouts

You might have noticed that default.cfc has an extends attribute in the cfcomponent tag (look at the first line in Listing 3). That means that it inherits some behavior from the layout.cfc in the same directory (see Figure 1 for an example inheritance tree).

The methods that are common to all of our layout components go here. Among those is the switchLayout() method that we used at the top of our home page to switch to the home page layout (Home.cfc).

If you wanted to switch to a Word layout (to output to Word instead of HTML), you would switch to the Word layout like so:

```
<cfset layout = layout.switchLayout("Word")>
```

This would use Word.cfc for your layout, which would be a layout component with very basic HTML and the following tag at the start of the output:

```
<cfcontent type="application/msword" reset="Yes">
```

Make sure to call layout.switchLayout() before layout.head(). If you want to take a look at the code for layout.cfc, you can see it in Listing 5.

How Does This Compare?

The usage of layout components that I have described is most similar to using cfinclude to include a header and footer, with some notable advantages. You don't have to worry about using different paths depending on the location of your file. You can see the whole layout in one file, allowing you to easily see where your divs and table cells open and close. You can switch your layout using code in just one place.

This approach is also similar to Fusebox layouts without requiring Fusebox and without requiring all requests to run through one file. Additionally, you don't have any restriction on the use of <cfflush> (a problem with layouts in Fusebox 3).

Getting Started

Here are some steps to get started using layout components:

1. Create a "layouts" directory from the root of your Web site
2. Create (or download) layout.cfc in your layouts folder (see Listing 5)
3. Create a default.cfc in your layouts folder (see Listing 3)
4. Initialize default.cfc from Application.cfm.
   <cfset layout = CreateObject("component","layouts.default").init()>
5. Create any other layout components you need for your site
6. Use the layout components on each page (see Listing 1 and Listing 2)

A Few Other Notes

Tip 1

I find it helpful to perform any logic I need on a page before my call to layout.head(). This may include calls to components or any other logic specific to the page. I don't perform any logic that isn't directly tied to output after layout.head().

Tip 2

Once your site is complete and tested, you can eliminate all white space before your HTML by adding <cfcontent reset="yes"> before the first line of html in your layout component. Don't do this until your site is fully tested as it could disrupt the display of problems that occur before you call layout.head().

Tip 3

You can access shared-scope variables (including request variables) from within components, but you cannot access local variables that were set outside of the component. I recommend manually passing in any required variables to your component (by adding a method to layout.cfc). If you do so, make sure you store the variables in the variables.me structure so that they will be retained when switching layouts.

Tip 4

If you need to output part of your layout after some of the contents of the page, you can add a method to your layout components to handle this. For example, on some sites, I need an extra layout.menu() method because I need to show my menu below content that is specific to the page. If you add a method like this, make sure to add it to layout.cfc (with no output) first. This will make sure that the method is always available, even for layouts that don't use it.

Conclusion

Layout components provide a flexible way to handle layouts for your sites. Try it out and I think you will find that the implementation of layout components is easy.

You can download the files from this example as well as layout components for PDF, Excel, and Word from my Web site (bryantwebconsulting.com). They have worked well for me and I look forward to hearing about how they work for you.

Good Luck!

About the Author

Steve Bryant is the owner of Bryant Web Consulting LLC (www.bryantwebconsulting.com) and teaches ColdFusion at Breakaway Interactive (www.breakawayinteractive.com). He got his BA in philosophy at Oklahoma State University and still has no idea how that led to a career in Web development.

steve@bryantwebconsulting.com

Listing 1

```
<cfoutput>#layout.head('My Title')#</cfoutput>
  <style type="text/css">.myclass {color:blue}</style>
<cfoutput>#layout.body()#</cfoutput>

Page Contents…

<cfoutput>#layout.end()#</cfoutput>
```

Listing 2

```
<cfset layout = layout.switchLayout("Home")>
<cfoutput>#layout.head('My Title')#</cfoutput>
<cfoutput>#layout.body()#</cfoutput>

Home Page Contents…

<cfoutput>#layout.end()#</cfoutput>
```

Listing 3

```
<cfcomponent displayname="Default Layout" extends="layout">
<cffunction name="head" access="public" output="yes">
<cfargument name="title" type="string" required="yes">
<html>
<head>
  <title>#arguments.title#</title>
</cffunction>
<cffunction name="body" access="public" output="yes">
```

```
</head>
<body>
</cffunction>
<cffunction name="end" access="public" output="yes">
</body>
</html>
</cffunction>
</cfcomponent>
```

Listing 4

```
<html>
<head>
  <title>My Title</title>
  <style type="text/css">.myclass {color:blue}</style>
</head>
<body>

Contents…

</body>
</html>
```

Listing 5

```
<cfcomponent>

<cffunction name="init" access="public" returntype="layout" output="no">

  <cfset variables.me = StructNew()>

  <cfreturn this>

</cffunction>

<cffunction name="switchLayout" access="public" returntype="layout"
output="no">
  <cfargument name="layout" type="string" required="yes">

  <cfset var result = CreateObject("component",layout).init()>

  <cfset result.setMe(variables.me)>
  <cfset this = result>

  <cfreturn result>
</cffunction>

<cffunction name="setMe" access="package" returntype="void" output="no">
  <cfargument name="me" type="struct" required="yes">

  <cfset variables.me = me>

</cffunction>

<cffunction name="head" access="public" output="yes"></cffunction>
<cffunction name="body" access="public" output="yes"></cffunction>
<cffunction name="end" access="public" output="yes"></cffunction>

</cfcomponent>\
```

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

Making Sense of Blogs

Creating an adaptive blog filter

By Chip Temm

It's a great problem to have: thousands of people like yourself providing daily updates from the leading edge of your craft. If you want to learn where things are going in the world of Web programming, you need to read blogs and lots of them.

Aggregators help make that easier, but only by reducing the amount of running around to find new sources. The quantity of content flowing through these channels is daunting and the quality highly variable. Somewhere in this big stack of articles, sometimes hundreds a day, are things you need to read to stay ahead of the curve.

How do you find them? Maybe you use Google's deskbar and load it up with a couple of aggregators or use Firefox's live bookmarks to do the same thing. Now you have to scan the headlines flowing in every couple of hours and see if something interesting has come up. Hopefully the author gave the item a good headline. The problem is pretty clear: we need something to help us filter the flow and highlight items we would be interested in.

Some folks are looking at collaborative filtering techniques. They want to get readers to rate blog items. This could be a good approach if you can get readers to click on the rating link; then you have cheating or malicious behavior to worry about. Messy. I'd like something more empirical. Maybe measuring cross-references to aid in weighting a la Google. Complicated.

What I want is a system that watches what I read and can predict from that data what I would be interested in reading in the future. Better yet, maybe I could teach the system how to categorize items in addition to filtering out uninteresting stuff. Spam filters do this kind of thing and, with a bit of adaptation, we can bend the concepts behind spam filtering into something useful for blog filtering and build ourselves a *flog* – a filtered Web log.

Most spam filters use some kind of optimized statistical algorithm to learn what you think is spam. You train the filter by marking items as spam as they come into your inbox or moving items out of the quarantine area if the filter got it wrong. This data builds up over time and your filter becomes more and more accurate. I will discuss a very simple naïve Bayesian algorithm I found while doing research for this article ("Build Your Own Bayesian Spam Filter" by John Graham-Cumming, 2005), which we will wrap up into a CFC so that we can put it to all kinds of interesting uses. We won't do a lot of optimizations here for the sake of a simple code sample – you can check out http://anthrologik.net/flogger if you want to watch this concept evolve over time or better yet to contribute to it!

Our system will contain the following major concepts: the text we want to categorize (an *article*), the body of knowledge accumulated so far (the *corpus*), and *categories* that articles can belong in. In order to add to the body of knowledge, we *train* the system by telling it that a given *article* belongs in a given *category*. If we are confident that our training has taught the system enough, we can have it make recommendations to us by *classifying* articles that we pass to it. Eventually, we will have the system classify all incoming articles and start either simply accepting its classification recommendations or, if it's wrong, changing the categories for problematic articles. We then feed this as training data back into the system as *training*. (The source code for this article can be downloaded from http://cfdj.sys-con.com.)

Category1	Word1 Word2	Count Count
coldfusion	component cffunction	25 15

We know from this brief analysis that we need at least two methods (train and classify) and one persistent object (corpus). Our corpus will consist of a struct that looks like this:

Where *count* is the number of times the word has appeared in all articles trained into the corpus to date.

When we call *train(inputText, category)*, our new component will perform a word frequency count for our input text and add the data to the appropriate section of the corpus. In the example above, if we train a new article into the coldfusion category and the article contains the word "component" five times, our corpus would now show:

coldfusion	component	30
	cffunction	15

In Listing 1, we have an argument *parsedTextStruct* instead of just a string. This refers to a method we use to parse incoming text and create a struct containing unique words in the text as keys and their frequency of occurrence as values. We'll skip over that for the sake of brevity, but the method is included in the download for this article. Since it is off in its own method, we can experiment independently with different ways of parsing articles and see how this affects parsing speed and system accuracy including filtering out "stop words" like prepositions, conjunctions, and pronouns.

You can see that this is really very simple. If the word "coldfusion" occurs five times in an article and we have in our corpus a "CF" category containing an entry for "coldfusion" with a value of "200" (meaning that coldfusion has occurred 200 times in all of the text classified to date into the CF category), training our corpus by placing our new article into the "CF" category will bring the value of CF.coldfusion to 205. This means that for all of the articles categorized thus far into the CF category, the word "coldfusion" occurred 205 times. See? Training is easy. The magic is in the classification method.

When we call *classify(inputText)*, we are asking the system to tell us how likely it is that our article falls into any of the currently defined categories, or to be more accurate the *relative* likelihood. This prediction uses a naïve Bayesian classifier that's based on the Bayes Theorem, which states that the probability of A happening given B happening is equal to the probability of B happening given A times the probability of A divided by the probability of B. This is written as $P(A|B) = P(B|A)P(A) / P(B)$. For a more in-depth discussion of Bayes Theorem and naïve Bayesian classifiers, see Wikipedia (http://en.wikipedia.org/wiki/Bayes_rule). If you are wondering, the "naïve" means that we will make the assumption that the words in our articles have no relationship to each other. While this is clearly a false assumption, it simplifies the math considerably and in practice produces results that are acceptable for our purposes; it is not uncommon to see a moderately trained classifier reach percentages of accuracy in the high nineties.

In our case, the system is looking for the probability of an *article (A)* being placed into one of our *categories* (C_i): $P(C_i|A)$. To get that we need to calculate the probability that for category C_i the set of words in the *article* appears in the *category*: $P(A|C_i)$. We then multiply that by $P(C_i)$, the probability of a word appearing in C_i or the word count of C_i divided by the total word count for *all* categories. Now we have:

```
P(Ci|A) = P(A|Ci) * P(Ci)
```

We will drop the probability of the article occurring, P(A), from our calculation because it is unknown and we are, in any case, only interested in the *relative* values of P(C|A) for our categories and P(A) will be constant

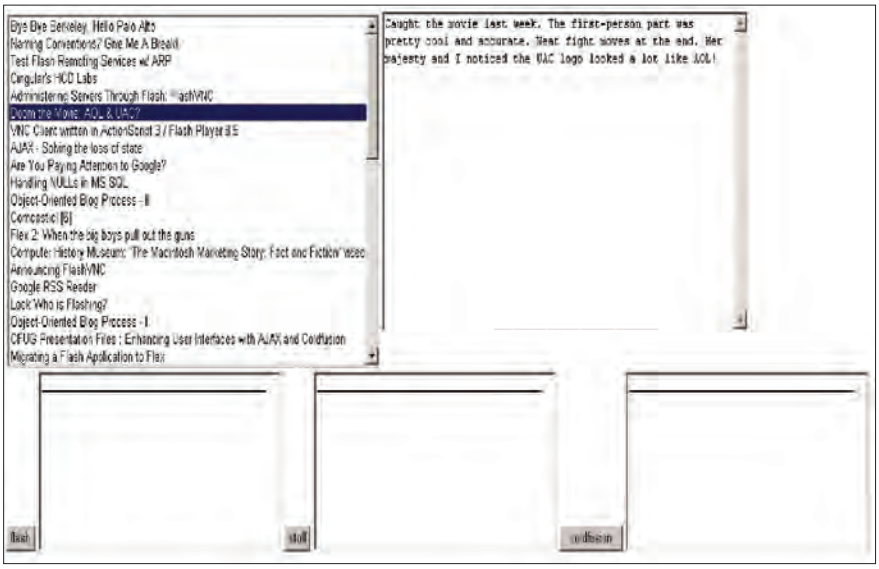


Figure 1

across all of our calculations $P(C_1|A)$ … $P(C_n|A)$.

$P(A|C_i)$ can be expanded as being the product of the probabilities for each *word (W)* in the *article* appearing in the *category*:

```
P(A|Ci) = P(W0|Ci) * P(W1|Ci) * … * P(Wn-1|Ci)
```

Each of these is just the number of occurrences of the *word* in the *category* divided by the total number of *words* in the *category*. That's not too hard.

Now multiply that by $P(C_i)$ or the number of *words* in our *category* C_i divided by the total number of *words* in the whole *corpus*.

```
P(Ci|A) = P(W0|Ci) * P(W1|Ci) * … * P(Wn-1|Ci) * P(Ci)
```

What if the word we are looking at doesn't appear in the category we are assessing against? Let's use a tiny number (0.1) to represent that we don't really know the probability but there is a chance that varies with the size of the word count of the category.

Now we have the core of our *classify* method, an equation that will produce relative scores for the relevance of an article to a category. Remember that our dropping the P(A) denominator has turned this into a ranking exercise, not a measure of probability. That also frees us up to use the log() function to reduce the size of the numbers we are operating on and increase performance by turning the multiplication of our terms into addition remembering that log(X*Y) = log(X) + log(Y).

```
P(Ci|A) = log(P(W0|Ci)) + log(P(W1|Ci)) + … + log(P(Wn-1|Ci)) + log(P(Ci))
```

The implementation in Listing 2 is translated from the Perl included in John Graham-Cummings' excellent article "*Build Your Own Bayesian Spam Filter.*"

Now we can do a small example (see Listing 3), which dumps out to:

```
coldfusion    -13.0321979407
flash         -8.93316952915
```

By comparing the numerical values, we can see that our article is more likely to be about Flash than ColdFusion (flash > coldfusion). What if we remove the word "flash" from our testdata article? Now we get:

```
coldfusion    -5.37759054744
flash         -6.070737728
```

ColdFusion is suddenly more relevant. If we want a best-guess recommended categorization, we can run our *recommendations* return through the *structSort* function:

```
sortedRecommendations = structSort(recommendations,'numeric','desc');
recommended = sortedRecommendations [1]; //currently equal to "coldfusion"
```

From here it is fairly short work to envision a system that reads feeds or an aggregator and presents something like Figure 1, allowing users to move items from the title list to the appropriate category to train the system.

After several weeks of training the system with several hundred articles, you could trust it to classify incoming articles and re-present them as filtered feeds. By the time this article reaches you, I should have such a system operational. Stay tuned at http://anthrologik.net/flogger.

This system has two things going for it: the concepts are not new or revolutionary (the math is about 250 years old and the concepts driving information-retrieval technology are of similar vintage) or complicated (the classify method is about 40 lines of code). These two properties usually make for a nice, stable library. Hopefully this library can be leveraged to make the blogosphere a more manageable source of information.

Resources

- *Graham-Cumming, J. "Build your own Bayesian spam filter":* http://www.jgc.org/antispam/05152005-92b9aec9357df88a2ce200056b18ee74.pdf
- *Lewis, D. D. "Naïve (Bayes) at Forty: The Independence Assumption in Information Retrieval":* http://citeseer.ist.psu.edu/cache/papers/cs/26985/http:zSzzSzwww.ai.mit.eduzSzpeoplezSzjimmylinzSzpaperszSzLewis98.pdf/lewis98naive.pdf
- *Wikipedia.org, "Bayes Rule":* http://en.wikipedia.com/wiki/bayes_rul

About the Author

Over the past decade, Chip Temm moved from North America to Europe and on to Africa where his company anthroLogik solutions provided analysis and development services to non-governmental organisations across seven timezones. He is currently back in Washington, DC where "remote development" means working from home and "wildlife" means raccoon.

chip@anthrologik.net

Code Sample 1

```
<cffunction
  name="train"
  access="public"
  returntype="VOID"
  hint="trains the AI to categorise
        certain words"
  >
  <cfargument
    name="parsedText"
    type="query"
    required="yes"
    hint="the parsed text to categorise"
  />
  <cfargument
    name="category"
    type="string"
    required="yes"
    hint="the categroy to place it in"
  />
  <cfscript>
  //Init private variables.
  var word ='';
  var wordcount =0;
  var excpt='';
  var i =1;
  var parsed_len= parsedText.recordCount;

  //this CFC has a private variable to
  //keep the corpus in memory during the
  //life of the object
  var myCorpus= variables.instance.corpus;

  for(i=1;i lte parsed_len;i=i+1){
    word = parsedText.word[i];
    wordCount = parsedText.wordCount[i];
    try{
      structInsert(
        myCorpus,
        arguments.category,
        structNew(),
        0
      );
    }catch(Any excpt){}
    incrementStructElement(
      inputStruct =
          myCorpus[arguments.category],
      key  = word,
      value  = wordCount
    );
  }
  //update the word counters for the category
  //and for the corpus
  updateStatistics();
  return;
  </cfscript>
</cffunction>
```

Code Sample 2

```
<cffunction
  name="classify"
  access="public"
  returntype="struct"
  hint="asks the AI to categorise words based
        on what it has learned"
  >
  <cfargument
    name="parsedText"
    type="query"
    required="yes"
    hint="the parsed text to categorise"
  />

  <cfscript>
  var excpt   ='';
  var entry   = '';
  var score   = structNew();
  var weight  = 0;
  var category ='';
  var word ='';
  var i=1;
  var v = variables;

  for(i=1;
    i lte arguments.parsedText.recordCount;
    i=i+1){
    word = arguments.parsedText.word[i];
    //for every category calculate :
    //  (word freq in category) divided by
    //  (category total wordcount)
    for(category in v.instance.count){
      if(structKeyExists(
          v.instance.corpus[category],
          word)
      ){
        //number of times this word appears
        //in this category
        weight =
        v.instance.corpus[category][word];
      }else{
        //very small number used to estimate
        //weight when we have no info
        weight = .1 ;
      }
      incrementStructElement(
          inputStruct = score,
          key  = category,
          value  =
            log(weight
              / v.instance.count[category])
      );
    }
  }
  //for every category adjust the score for
  //the input text based on
  //(category wordcount) / (corpus wordcount)
  for(category in variables.instance.count){
    incrementStructElement(
        inputStruct = score,
        key= category,
        value=
         log(variables.instance.count[category]
           / variables.instance.total)
    );
  }
  return score;
  </cfscript>
</cffunction>
```

incrementStructElement just adds a value to a key value, creating the key with a value of zero if necessary:

```
<cfscript>
function incrementStructElement(inputStruct,
  key,value){
try{
structInsert(arguments.inputStruct,
  arguments.key, 0,0);
}catch(Any excpt){}
arguments.inputStruct[arguments.key] =
  arguments.inputStruct[arguments.key] +
  arguments.value;
}
}
</cfscript>
```

Code Sample 3

```
<cfscript>
//these structures simulate input articles.
//The struct keys are words appearing
//in the articles.

//simulate an article about Flash
testdata_flash = structNew();
  testdata_flash.flash = 12;
  testdata_flash.skills = 1;
  //add in some noise
  testdata_flash.junk = 200;

//simulate an article about CF
testdata_CF = structNew();
  testdata_CF.coldfusion = 12;
  testdata_CF.skills = 2;
  //add in some noise
  testdata_CF.junk = 200;

//an incoming article we want the system
//to classify for us
testdata = structNew();
  testdata.skills = 1;
  testdata.flash = 2;

//create an instance of our new component
 myClassifier =
   createObject('component','classifier');
  myClassifier.init();
//train it with two documents worth of data
myClassifier.train(testdata_flash,'flash');
myClassifier.train(testdata_CF,'coldfusion');

//based on training, what do you suggest?
recommendations =
myClassifier.classify(testdata);
</cfscript>
```

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

What's AJAX

Part 2

Creating a better user experience on your Web site

By Rob Gonda

AJAX can make the HTML user experience almost as pleasant as Flash. The main advantage of Flash, in spite of its vector animations, is that you never reload the page. Flash Remoting allows you to interface with the server in the background and AJAX does exactly the same for HTML pages.

In my previous article, "What's AJAX?" (*CFDJ*, Vol. 7, issue 9), I covered the basics of AJAX – everything from setting it up, all the way to having it running in an MVC design with basic functionality. Thus far, we have only sent and received simple objects, which is good way to understand the principle, but far from reality.

CFAJAX allows you to send complex objects, but for some reason it's extremely difficult to receive and interpret them on the ColdFusion side. For this reason I came up with a much simpler and straightforward way: WDDX serialization.

This concept may be familiar to some of you. WDDX stands for Web Development Data Exchange and Allaire created it in 2001 to solve problems in exchanging data between Web applications. In a nutshell, WDDX is a technology that facilitates exchanging complex objects over XML. WDXX supports Booleans, numbers, strings, date-time, arrays, structures, and record sets. Modules for WDDX support exist in various languages, including but not limited to ColdFusion, Perl, Java, JavaScript, ASP, .NET, and PHP.

A JavaScript WDDX component is included in the cfide/scripts folder of every ColdFusion installation. You can locate it at /cfide/scripts/wddx.js.

Complex Object Example

Now I'll show you how to use it. Listing 1 shows a simple example in which I created a complex object in JavaScript – a structure that contains an array of structures – and serialized it before sending it to ColdFusion. All you have to do is initialize a new WDDX serializer and serialize the complex object. You can see in Listing 1 how simple it is; you don't need to know the structure of the packet.

Notice the line "DWREngine._execute(_cfscriptLocation, null, ' wddxTest', oWddx.serialize(_o), testResult);". Here the first argument is the location of the ColdFusion model; the wddxTest is the function to be called; the argument is being serialized by our JavaScript component; and the testResult is the callback function.

Listing 2 shows how simple it is to receive the call. The function takes only one argument: the WDDX packet. This packet is not an object; it's actually an XML string. Listing 2 returns it intact just so you see exactly what's happening behind the scenes, and the callback function alerts the return packet. If you run my example, you'll see that the XML string is being alerted to the screen by the testResult function.

Trick 1

Here is a very important cross-browser note for you. Pay special attention to the URLDecode function in the ColdFusion listener (see Listing 2). I found that if you don't use it, the code will work just fine in IE, Firefox, and Netscape, but it will break in Safari for Mac users. I learned this the hard way, so here I am sharing the trick. For some reason, Safari URL-Encodes the XML packet before sending it and ColdFusion cannot decode it, making the argument

impossible to de-serialize. Ideally, you'll test your code in as many environments as possible before you go live, so remember this and it may save you some debugging time.

Login Example

I'll now demonstrate a login example and some useful debugging techniques. Sometimes you'll find that the user is navigating your site and he is allowed to log in by using a form that is always available, either in the top or side navigation. Sometimes the user just posts a form, for example, a search products form, and then decides to log in. This may or may not be a problem, but if you allow him to post just the login form, you will lose all other existing form variables. One solution, you may be thinking now, is to loop through the form collection in the login form and create hidden fields; it may work in some cases, however, in other cases such as add-to-cart or checkout, resubmitting the form may cause duplicate process pages. Using cflocate after each process page is a good practice, but I am digressing and won't get into that.

How about adding some of the concepts we learned in this article? How nice would it be to let the user know if her username or password does not match any existing one without refreshing the page? Or even more! How about logging the user in, hiding the login form, and adding her login status without refreshing the page, and thus, without resubmitting, redirecting, or even losing focus of the current item she may be seeing.

This time we'll send the user and password to the ColdFusion model and expect an array for the response. The array will have a "success" status: in case of error, a message; and in case of success, the name of the user. We could also pass a structure, but you'll find that if you pass a structure back, JavaScript will interpret it as an array or keys and values. Then you'll have to loop through the array and assign the values to the keys. An alternative could be using WDDX, but this time to encode the packet before sending it back to JavaScript and use the WDDX component in JavaScript to deserialize it.

Sometimes it's not that easy to know exactly how the structure that ColdFusion sent is being interpreted by JavaScript. In these cases, I always refer to the same debugging method: I loop over the entire scope or collection to find what is being sent back. Listing 3 shows a simple one-line loop that will do the trick. Try to return a structure and see for yourself the way it's being passed back.

Let us continue with our login example. The ColdFusion model will return an array of two elements: array[1] will be the success variable (true or false), and array[2] will be the error message in case of an error, or the name of the user in case of success. For the sake of this article, the model just checks for a static user and password; however, this is exactly where your login logic should be. The trick is to update the session in the model in the background and just send back to JavaScript what it needs for the presentation layer. For instance, you could add the User-ID for the session, and send back the name and last name so it can be displayed in the left navigation.

After the user successfully logged in, we will make the login form disappear and add a simple message that reads "logged in as %name%". We will do this by using the JavaScript innerHTML function. The innerHTML function lets you modify the source code on the fly. Our form was placed inside a DIV layer that is used as a holder and, after the login, we will change the code of the holder to display the new status. When using this method, make sure that the navigation does not have to change upon login, or, if it does, you may modify the navigation too by using innerHTML as well. If you code in an object-oriented fashion, your navigation would be generated by an object you may call in the login function and send the generated HTML back to the JavaScript controller, which will replace the existing one by using innerHTML functions.

Content Example

Another great advantage of AJAX is that it improves load time. For example, imagine a typical site: it has a design frame and content in the center area. Every click has to reload the entire frame to update the content. Ideally, you wouldn't have to transfer the code for the frame and download any of its graphic components again. Caching certainly helps, but AJAX can make it even faster. Listings 4 and 5 show a simple example in which the content is being generated in the ColdFusion model and passed back to the JavaScript controller. For simplicity, in this example I actually added the content directly in the Model.cfm page. In real-life examples, the ideal way would be to actually include a view template with a cfinclude inside the cfsavecontent tag; by doing that your views will not depend on the AJAX component. Perhaps you won't see a huge speed difference just by running this example; this is due to the fact it doesn't have any design elements. As an experiment, try to add a full design frame to my example with an average page weight of 100k or 200k in graphics and compare load times.

Trick 2

Observe that when you pass back HTML content to JavaScript, you have to URL-Encode the content (see Listing 7). If you try to pass back the HTML tags without encoding it, the AJAX calls will break without notice or error messages. In the receiving counterpart, you need to decode the HTML and this is achieved by using the unescape function as shown in Listing 6.

The drawback of this method is that it completely kills search engine friendliness. Search engines will not execute JavaScript calls and will not index your content. As a rule of thumb, I try to keep all front-end related content as SEO friendly as possible. You can use these examples for back-end related purposes. For example, gmail was written completely in AJAX. Notice that you can read and write e-mails and the site will never refresh and the speed is different over hotmail or any other Web mail application.

Table Example

Next, I'll cover looping over an AJAX record set and populating tables. Although you could generate the table and send the rendered HTML by using my previous content example, by using this technique, you'll delegate the load to the client browser instead of keeping it in the server.

I will introduce now a few DOM functions: createElement, createTextNode, setAttribute, appendChild, and removeChild. The concept here is to receive a recordset from a ColdFusion function and create the HTML by using DOM functions instead of the HTML tags.

In this example, I'll only demonstrate the JavaScript part that builds the table. If you look at Listing 8, you'll see an empty table with an ID in the "tbody" tag. This "tbody" tag is extremely useful

when you have additional elements such as column headers in the "thead" tag. The "tbody" allows us to reference that section directly when adding and deleting elements.

The logical process that I follow in my pages is showing a "please wait" DIV when the user submits the request. At this point I make the AJAX call, send the request parameters, and wait for the response. The response will trigger the callback function, which immediately executes the "clear_table" function. This function loops over each row in the "tbody" and calls the DOM "removeChild" function. This will clear the results from the previous call. After I finished emptying the table, we proceed to populate it with the new recordset. If you look at the "showTable" function, you'll find two nested loops: one for each row and one for each column. Looping over each column is optional and up to you. Traditional ColdFusion programmers may prefer to loop over each row and manually code each column to their needs. Personally, I like to avoid having to write the same code over and over if I can help it. In my example, I created an array called "r" that contains the columns that I want to display. After you finish populating the table with the new recordset, you may remove the "please wait" message and the process is over.

Populating the table is actually rather simple: first I loop over each row from the recordset and create a "TR" element by using createElement('tr'); then in the nested loop I create the "TD" element for each column by using createElement('td'). The text is then evaluated in the following line "currenttext=document.createTextNode(eval('o.' + r[i] + '[j]'));". Here the "eval" function is resolving the recordset, then the column name, and then the position. For instance, the first row, first column, will read "o.uname[0]".

Creating the links is not as simple as typing "<a>". You will need to create an "A" element by using createElement('a'), and then use the setAttribute to add the href attribute.

After all is said and done, all that's left to do is to append each element to its parent. First you append the text node to the "a" element, then the "a" to the "td", then the "td" to the "tr", and finally the "tr" to the "tbody".

CFAJAX includes some utilities that facilitate this process but, the truth is, you will need to learn DOM JS if you really want to take control. For example, the example in Listing 8 only adds a link to the first column, and all the rest are just text.

Related Options Example

My last example will explain a useful built-in function with CFAJAX: related options (see Listing 9). How many times have you had to load enormous arrays of options to your HTML template to achieve this? Wouldn't it be great if you could load the dependent options only after the main option has been selected? CFAJAX includes two handy functions: "removeAllOptions" and "addOptions". These two functions will drive the JavaScript work to achieve related options to zero.

I hope the community doesn't mind, but I'll use the example from their site to demonstrate how this is achieved.

First, we'll have a brand dropdown that will call a JavaScript function called "getBrand" on its "onChange" trigger. All this function does is send ColdFusion the value of the current selected brand, and ColdFusion will return an array of values.

The array actually contains value pairs, which are the key, and the value separated by commas. It's important to pay attention to the hint value of the ColdFusion function; in this example, it's hint="type='keyvalue' jsreturn='array'", which is actually used in the callback function to set the expected value types. You can tell JavaScript the complex type of the return, set it to receive value pairs, and optionally set the delimited of the pairs. By default, it is expecting a comma for the delimiter.

The callback function cannot get any simpler. "getBrandResults" performs only two functions: DWRUtil.removeAllOptions("model"), which removes all current options from the model select component, and DWRUtil.addOptions("model", modelArray, "KEY", "VALUE"), which add the returned array to the model select component.

Conclusion

I hope you learned how AJAX could create a better user experience on your site. I covered passing complex objects, using innerHTML to modify the view layer after it has been rendered, drawing dynamic content, manipulating tables, and using related select options. AJAX is extremely fast, secure, and allows for advanced functionality that synchronous HTML does not.

About the Author

Rob Gonda is the CTO for iChameleon Group (www.ichameleon-group.com). He is an Advanced Certified Coldfusion Developer, holds a BS in computer science and engineering from F.A.U, and an MBA with a specialization in entrepreneurship from the Wayne Huizenga School of Business.

rob@gondas.com

LISTING 1: Index.cfm

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head>
   <title>WDDX test</title>

<script type='text/javascript'>_cfscriptLocation = "model.cfm";</script>
<script type='text/javascript' src='core/engine.js'></script>
<script type='text/javascript' src='core/wddx.js'></script>
<script type='text/javascript' src='core/errorHandler.js'></script>

<script language="JavaScript">
   function testResult (r) {
          alert(r);
   }

   var oWddx = new WddxSerializer();
   var _o = {"myStruct": [
          {"myEvent": "event 1", "method": "new", "regex":
"^http://.*"},
          {"myEvent": "event 2", "method": "delete", "regex": "^de-
lete.*"},
```

```
            {"myEvent": "event 3", "method": "random", "regex": "^ran-
dom.*"}
        ]
    };
    DWREngine._errorHandler =  errorHandler;
    DWREngine._execute(_cfscriptLocation, null, ' wddxTest', oWddx.
serialize(_o), testResult);

</script>

</head>

<body></body>
</html>
```

LISTING 2: model.cfm

```
<cfinclude template="core/cfajax.cfm">
<cffunction name="test">
    <cfargument name="WDDXEncodedEvent">
    <cfset var event = "">
    <cfwddx action="WDDX2CFML" input="#URLDecode(WDDXEncodedEvent)#"
output="event">

    <cfreturn URLDecode(WDDXEncodedEvent) />
</cffunction>
```

LISTING 3: Debugging

```
function loginResult (r) {
    for (var x in r) alert(x);
}
```

LISTING 4: Login HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head>
    <title>AJAX Login</title>

<script type='text/javascript'>_cfscriptLocation = "model.cfm";</script>
<script type='text/javascript' src='core/engine.js'></script>
<script type='text/javascript' src='core/wddx.js'></script>
<script type='text/javascript' src='core/errorHandler.js'></script>

<script language="JavaScript">
    function loginResult (r) {
            if (r[0] == 'false') {
                    alert(r[1]);
                    return false;
            } else {
                    document.getElementById('LoginArea').innerHTML =
'logged in as ' + r[1] + '.';
            }
    }

    function doLogin() {
            var oWddx = new WddxSerializer();

            var frm = document.getElementById('loginFrm');
            var _o = {uname: frm.uname.value, pword: frm.pword.value};

            DWREngine._errorHandler =  errorHandler;
            DWREngine._execute(_cfscriptLocation, null, 'doLogin', oWddx.
serialize(_o), loginResult);
    }

</script>

</head>

<body>

<div id="LoginArea">
    <form id="loginFrm">
            Username: <input type="text" name="uname"><br>
            Password: <input type="Password" name="pword"><br>
            <input type="Button" value="login" onclick="doLogin()">
    </form>
</div>


</body>
</html>
```

LISTING 5: doLogin ColdFusion Code

```
<cfinclude template="core/cfajax.cfm">
<cffunction name="doLogin" returntype="array">
    <cfargument name="WDDXEncodedEvent" type="string" />
    <cfset var event = "" />
    <cfset var returnArray = arrayNew(1) />

    <cfwddx action="WDDX2CFML" input="#URLDecode(WDDXEncodedEvent)#"
output="event">

    <!--- logic here --->
    <cfif event.uname eq "admin" and event.pword eq "1234">
            <cfset returnArray[1] = true />
            <cfset returnArray[2] = "Rob Gonda" />
            <!--- update session --->
    <cfelse>
            <cfset returnArray[1] = false />
            <cfset returnArray[2] = "wrong user or password" />
    </cfif>

    <cfreturn returnArray />
</cffunction>
```

LISTING 6: Content Page.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>
<head>
    <title>AJAX Login</title>

<script type='text/javascript'>_cfscriptLocation = "model.cfm";</script>
<script type='text/javascript' src='core/util.js'></script>
<script type='text/javascript' src='core/engine.js'></script>
<script type='text/javascript' src='core/wddx.js'></script>
<script type='text/javascript' src='core/errorHandler.js'></script>

<script language="JavaScript">
    function getConentResult (r) {
            document.getElementById('title').innerHTML = unescape(r[0].
replace(/\+/g,  " "));
            document.getElementById('content').innerHTML = unescape(r[1].
replace(/\+/g,  " "));
    }

    function getConent(c) {
            DWRUtil.useLoadingMessage();
            DWREngine._errorHandler =  errorHandler;
            DWREngine._execute(_cfscriptLocation, null, 'getConent', c,
getConentResult);
            return false;
    }

</script>

</head>
```

```
<body onload="getConent('aboutUs');">

<table border="1" width="600">
    <tr>
            <td width="100" valign="top">
                    <!--- side menu --->
                    <a href="#" onclick="return getConent('aboutUs');"
>About us</a><br>
                    <a href="#" onclick="return getConent('companyInfo
');">Company Info</a><br>
                    <a href="#" onclick="return getConent('ContactUs')
;">Contact Us</a><br>
            </td>
            <td>
                    <div id="title"> </div>
                    <div id="content"> </div>
            </td>
    </tr>
</table>



</body>
</html>
```

LISTING 7: Get Content Model

```
<cfinclude template="core/cfajax.cfm">
<cffunction name="getConent">
    <cfargument name="contentPage" type="string" />

    <cfset var returnArray = arrayNew(1) />
    <cfset var title = "" />
    <cfset var content = "" />

    <cfsavecontent variable="title"><cfoutput><h4>#arguments.content-
Page#</h4></cfoutput></cfsavecontent>
    <cfsavecontent variable="content"><cfoutput>
            <ul>
                    <cfloop from="1" to="4" index="i">
                            <li>#arguments.contentPage# #i#</li>
                    </cfloop>
            </ul>
    </cfoutput></cfsavecontent>

    <cfset returnArray[1] = URLEncodedFormat(title) />
    <cfset returnArray[2] = URLEncodedFormat(content) />

    <cfreturn returnArray />
</cffunction>
```

LISTING 8: Building a table

```
<table width="700" cellpadding="2" cellspacing="0">
  <tbody id="mytable">
  </tbody>
</table>

function clear_table() {
    var l = document.getElementById('mytable').rows.length;
    for (i = 0; i < l; i++) {
document.getElementById('mytable').removeChild(document.getElementById(
'mytable').rows[0]);
    }
}

    function showTable(o) {
            var r = [];
            r[r.length] = "uname";
            r[r.length] = "fname";
            r[r.length] = "lname";
        mytablebody = document.getElementById('mytable');
        for(j=0;j<o.uname.length;j++) {
            mycurrent_row=document.createElement('tr');
                    for (var i = 0;i < r.length; i++) { // Loop thru
array of fields
                mycurrent_cell=document.createElement('td'); // create
td cell
                currenttext=document.createTextNode(eval('o.' + r[i] +
'[j]')); // create text field

                                            // link to edit user
                                            if (r[i] == "uname") {
                                                    myLink=document.
createElement('a'); // create A Link tag
                                                    myLink.
setAttribute('href',o.linkto[j]); // Add href link to A tag
                                                    myLink.appendChil
d(currenttext); // Append Text to Link
                                            }

                                    if (r[i] == "uname") {mycurrent_cell.
appendChild(myLink)} else {mycurrent_cell.appendChild(currenttext)}; //
If first column, add A link, else add plain text field
                mycurrent_row.appendChild(mycurrent_cell); // append TD
cell
                    }
            mytablebody.appendChild(mycurrent_row); // append TR row.
        }
    sortTable('mytable',  0, false); // execute SORT by first column.
    }
```

LISTING 9: Related Options

```
function getBrand()
    {
            var brand = DWRUtil.getValue("brand");
            DWREngine._execute(_cfscriptLocation, null, 'makelookup',
brand, getBrandResult);
    }



<cffunction name="makelookup" hint="type='keyvalue' jsreturn='array'">
            <cfargument name="brand" required="yes" type="string">
            <cfset model = ArrayNew(1)>
            <cfif arguments.brand eq "Nokia">
                    <cfset ArrayAppend(model, "10, Nokia 7280")>
                    <cfset ArrayAppend(model, "11, Nokia N-Gage")>
                    <cfset ArrayAppend(model, "12, Nokia 7270")>
                    <cfset ArrayAppend(model, "13, Nokia 7260")>
            <cfelseif arguments.brand eq "Motorolla">
                    <cfset ArrayAppend(model, "20,E895")>
                    <cfset ArrayAppend(model, "21,Motorolla V8")>
                    <cfset ArrayAppend(model, "22,V150")>
                    <cfset ArrayAppend(model, "23,Mpx100")>
            <cfelseif arguments.brand eq "Samsung">
                    <cfset ArrayAppend(model, "30, Z130")>
                    <cfset ArrayAppend(model, "31, Z700")>
                    <cfset ArrayAppend(model, "32, X470")>
                    <cfset ArrayAppend(model, "33, D488")>
            <cfelse>
                    <cfset ArrayAppend(model, "0,Not Available")>
            </cfif>
            <cfreturn model>
    </cffunction>

function getBrandResult(modelArray)
    {
            DWRUtil.removeAllOptions("model");
            DWRUtil.addOptions("model", modelArray, "KEY", "VALUE");
    }
```

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

Visit the *New* www.SYS-CON.com Website Today!

The World's Leading *i*-Technology News and Information Source

24/7

FREE NEWSLETTERS
Stay ahead of the i-Technology curve with
E-mail updates on what's happening in your industry

SYS-CON.TV
Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!
Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST
Streaming video on today's i-Technology news, events, and webinars

EDUCATION
The world's leading online i-Technology university

RESEARCH
i-Technology data "and" analysis for business decision-makers

MAGAZINES
View the current issue and past archives of your favorite i-Technology journal

INTERNATIONAL SITES
Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

IT Solutions Guide
Information Storage+Security Journal
JDJ
Web Services Journal
.NET Developer's Journal
LinuxWorld Magazine
Linux Business News
Eclipse Developer's Journal
MX Developer's Journal
ColdFusion Developer's Journal
XML Journal
Wireless Business & Technology
Symbian Developer's Journal
WebSphere Journal
WLDJ
PowerBuilder Developer's Journal

ColdFusion U

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

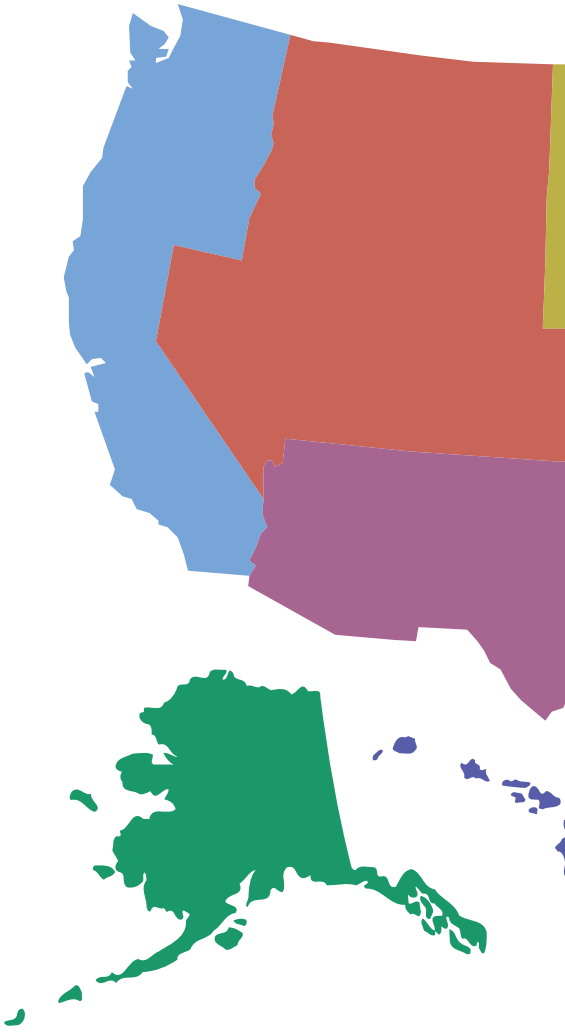
Minnesota
Brooklyn Park
Twin Cities CFUG
www.colderfusion.com

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

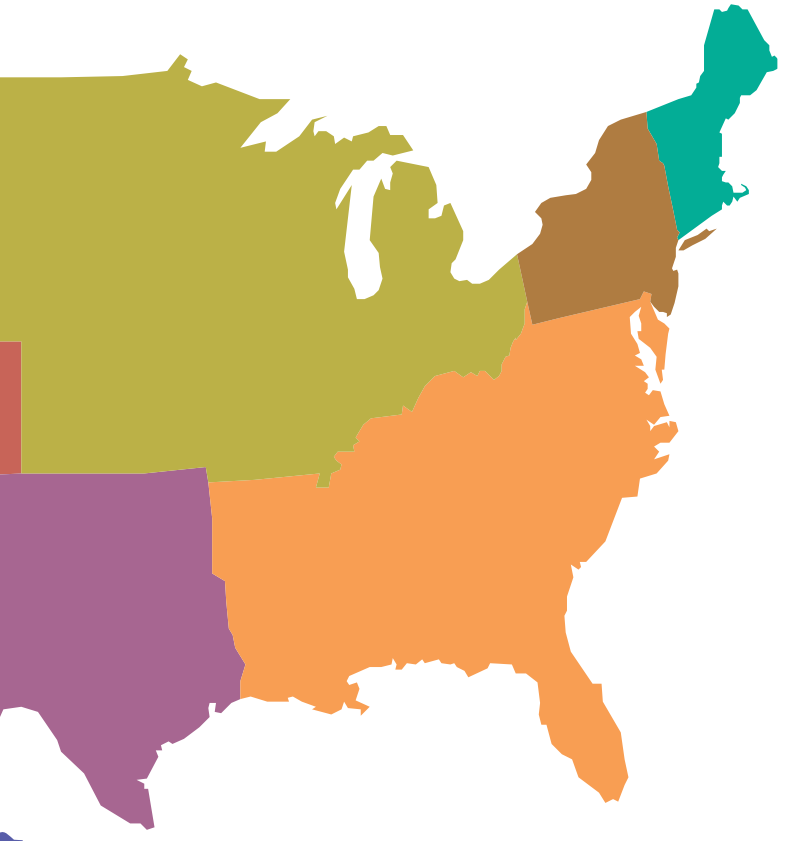
North Carolina
Raleigh
Raleigh, NC CFUG
www.ccfug.org

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

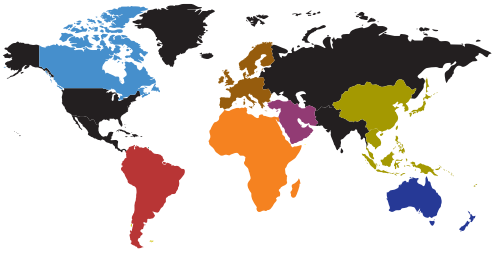
Utah
North Salt Lake
Salt Lake City, UT CFUG
www.slcfug.org

Washington
Seattle
Seattle CFUG
www.seattlecfug.com

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

Fully Automated Fusedocs Using FuseMinder

Automating the mundane tasks

By Brian Hostetler

Recently, during a Hal Helms training session, I started to look for ways to automate some of the drudgery of my upcoming application.

After Hal talked about mindmaps, he began creating the four circuits and perhaps a dozen fuse files for the application. In 45 minutes, he created the folder, files, and fusedocs. I began wondering: Would it be possible to automate this, thereby cutting down on the mindmap-to-fusestub phase?

An online search for automated tools written to automate this process led me to Jeff Peter's grokfusebox Web site (www.grokfusebox.com). I downloaded and unzipped Jeff's tool, FuseMinder for FB4. FuseMinder reads a mindmap file and creates the directory structure and files on the server. After altering my mindmap to fit Jeff's specifications, I gave it a whirl and, to my enjoyment, it worked great! All of my folders were created in the way I intended, and all of the fusefiles were created and had fusedocs already written for them.

I noticed something, though. In my mindmap, I had included both incoming and outgoing variables in the mindmap's notes section. Hal had explained that during the creation of the mindmap was the best time to identify these variables, prior to their inclusion in the fusedocs. However, if the information to help build the fusedocs is put in the notes section of the mindmap, why didn't FuseMinder use them to create the fusedocs?

After delving into the code of FuseMinder, I noticed that the tool was parsing the text entered into the notes section, placing it at the end of the fusestub. However, this was the very information that would eventually need to be the fusedoc itself. Couldn't the program be modified to handle this automatically?

I e-mailed Jeff Peters to discuss my idea with him, and from our chain of e-mails it became clear to me that one reason this was done was so that a portion of the HTML front end could be added to the mindmap and then parsed into the fuse by FuseMinder. In this way, the front end of the project was mostly complete; all the coder had to do was wrap the HTML in ColdFusion to make the display dynamic. I thought this was a great idea, but also thought that it could be expanded on.

The idea that I had was to modify FuseMinder in such a way that it would create all of the fusedocs for the architect, therefore allowing them to stay in the mindset of an architect. We all know what happens when you start switching from one task to another: none of them get done exactly the way you would like. It's too easy for our thought processes to become muddied when switching tasks.

With this task in mind, I took another look at the FuseMinder code. I found where the notes section of the mindmap was parsed and noted how the original FuseMinder handled it. From there I modified where FuseMinder sent the notes text, then searched for specifically formatted text. Confused? Let's look at an example.

Figure 1 shows a section of a mindmap outline file created by Visual Mind.

Notice the lines formatted like this:

```
IN:string:userName:comment=This is the username:scope=attributes
```

This line of code represents an incoming variable to be placed in the "in" section of the fusedocs. My modified code resulted in the following fusedocs tag:

```
<string name="userName" comment="This is the username" scope="attributes"
/>
```

There are three types of taglines that the code I wrote will recognize. Any lines that start with IN, OUT, or RESONSIBILITIES followed by the correct syntax will be parsed into the correct section of the fusedocs. Anything else will be

```
ct:userAccounts
ct:Public
        fa:invalidLogin
                do:Public.showLogin
        defaultFa:showLogin
                ff:dspShowLogin.cfm
                    RESPONSIBILITIES:I show a login screen
                    IN:string:message
                    IN:string:xfa.submitLogin
                    OUT:string:userName:scope=formorurl
                    OUT:string:password:scope=formorurl
                    OUT:string:fuseaction:scope=formorurl
                        xfa:submitLogin=Security.validateLogin
        fa:showCreateUser
                ff:dspCreateUser.cfm
                    RESPONSIBILITIES:I show the create user screen
                    IN:string:xfa.submitNewUser
                    OUT:string:firstName:scope=formorurl
                    OUT:string:lastName:scope=formorurl
                    OUT:string:userName:scope=formorurl
                    OUT:string:password:scope=formorurl
                    OUT:string:fuseaction:scope=formorurl
                        xfa:submitNewUser=Account.create
```

Figure 1

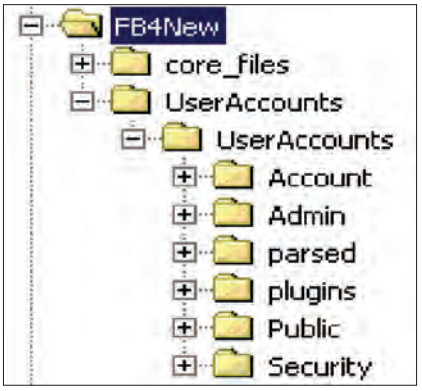


Figure 2

treated as a note and placed below the fusedocs. This allows the architect the same flexibility that Jeff Peters wanted – being able to place HTML into the mindmap outline where it can be translated into the fusestub.

Let's look at each of the three recognized keywords and their syntax. Note that all of the lines are delimited by colons (":"), just like the nodes in the mindmap, providing the architect with a common delimiter throughout the mindmap process.

The RESONSIBILITIES keyword allows the architect to supply the fusedocs with custom text for the responsibility of the fuse. Just place the free text after the keyword and the delimiter. Once you have completed the text, enter a carriage return to show the end of the line.

The IN and OUT keywords expect the same syntax following the keyword. The next entry should be the data type of the variable. This could be string, number, recordset, or any of the other fusedoc's variable types. After the type comes the name of the variable. This is free text, but should be a valid variable name for ColdFusion since it will eventually be used in code.

Up to now, everything that I have mentioned is required. But, as you may know, fusedocs allow many more fields in their variable declarations, such as comments, scope, etc. To account for this, these extra fields can be entered following the variable name. To do this, simply enter something similar to "scope=attributes" and the "scope="attributes" field will be placed into the fusedoc.

Let's take another look at the complete mindmap outline that's shown in Figure 1. When you run FuseMinder with these changes on the outline, the following directory structure will be created (see Figure 2).

Within these directories are the fusestubs for all of the fuse files we specified in the mindmap file. For those of you new to FuseMinder, these are the nodes marked with "ff:". Opening the fuse file shown in the sample outline in Figure 1, dspCreateUser.cfm, reveals the fusedoc shown in Figure 3.

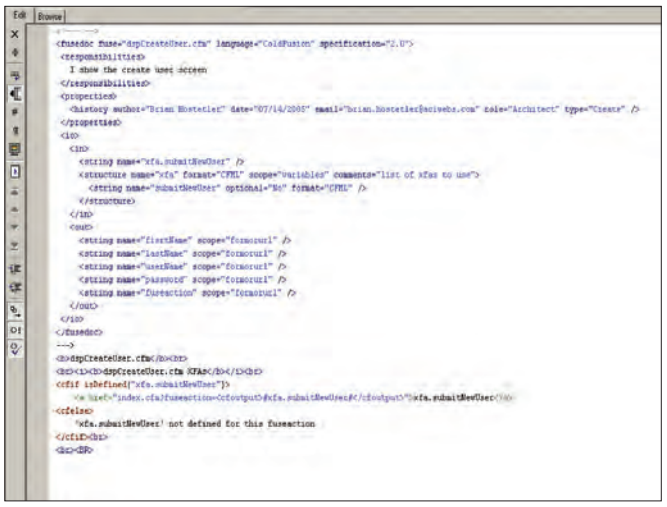


Figure 3

You can see that all of the incoming and outgoing variables specified in the mindmap outline are placed in the fusedoc and that there are no parts of the fusedoc left to be completed. This fusedoc can be passed directly to a coder and completed without any further work from the architect.

There is one other area that I have not covered – coding any structures and/or recordsets into the mindmap. When you need to enter either a recordset or structure data type, place all of the fields on separate rows starting with "-",using the colon delimiter to separate the field name and data type. To show this better, let's review an example.

Say you are expecting an incoming recordset named qryUser with columns of userID (number) and firstName (string). This is what you would enter in the mindmap:

```
IN:recordset:qryUser
  -number:userID
  -string:firstName
```

This would create the following in the fusedoc:

```
<recordset name="qryUser">
   <number name="userID" />
   <string name="firstName" />
</recordset>
```

By using this code and a properly formatted mindmap outline, any architect can run the outline through FuseMinder and be done with the architecting process much more quickly; the drudgery of creating the full directory structure and fusestubs is handled automatically.

Last, I want to thank Jeff Peters for the work he has done with creating the FuseMinder tool and for taking the time to discuss the process with me while I researched my ideas. I also want to thank Hal Helms for discussing the problems he saw in the process and helping me think through the entire process completely. I hope that the research I have done will be a help to the Fusebox community. If you are interested in seeing any of the files that I have included here, or would like the FuseMinder code that I am using to create the full fusedocs, or just want to discuss these ideas with me, feel free to e-mail me at brian.hostetler@us.army.mil.

About the Author

Brian Hostetler is a ColdFusion developer.

brian.hostetler@us.army.mil

MAX 2005

An Attendee's View

There's always something new

By Mary-Catherine Gerrey

This year MAX 2005 was great. I landed in Anaheim on Sunday, registered and then went to the welcome reception where I chatted with other developers and met with vendors.

On Monday I headed into the conference with my session schedule all planned out, ready to absorb all I could about ColdFusion, its newest features, its future, more OO, and maybe some Dreamweaver tips. My first session, Rich Internet Applications (RIAs) by Mike Sundermeyer, completely changed all that. At that session I hoped to pick up a few user interface pointers for an application I'm redesigning. Instead I was introduced to Flex. I had heard of it but thought it was some new graphical thingy that programmers like me need not worry about. Well, it isn't.

I was completely floored by Flex. The next version of the builder is simply an Eclipse plugin. I am already an Eclipse user, so the session began with language that I speak. Flex doesn't require server software but there is an option for it, which adds more capabilities. Flex Builder 2 is drag-and-drop Flash, with no timeline. It has prettiness built in. There's even code behind it, MXML. This was putting the concept of "user experience" into terms that a programmer can grasp. At that point, my focus at MAX changed and I went to a number of other RIA sessions that I hadn't planned on going to; for example, on Tuesday I caught a ColdFusion and Flex integration session. I promptly bought *Developing Rich Clients with Macromedia Flex* at the conference to read on the flight home (not that I made it through all 685 pages).

Developing Rich Internet Applications was a large focus of the conference and seems to be one of Macromedia's top priorities. Macromedia's CEO, Stephen Elop, stated that they were committed to the user experience, both in the applications Macromedia creates and in providing developers with tools to create rich applications. With tools like Flex, Dreamweaver, and Flash integration into ColdFusion, it's becoming much easier for people to create user friendly applications, and do it quickly.

Rich Internet Applications are not simply pretty Web sites. Yes, RIAs generally look nice, but that's only one aspect of a user's experience. Information has to be easily and painlessly obtainable from a system, meaning that users can navigate through an application and achieve their goals without difficulty (or crying out in frustration). Part of achieving a user's goals is including functionality and an interface that provides value to your application. Flair for the sake of flair detracts from the application's usability . You don't want people to loathe having to enter information into your application. Why? Because people won't enter their information and what they do won't be complete, making the application and its data useless.

Yes, I am a programmer won over by the user experience concept. Then again, I'm in the process of redesigning an application that users despise due to its interface inadequacies. I did attend a number of my planned ColdFusion sessions as well. Not only should usable applications be a development team's goal, but creating code that is easily maintainable should be a top priority as well.

Session speakers stressed topics such as frameworks, methodologies, and reusable code. The Model-View-Controller architecture model and having a consistent software development methodology like FLiP are both ways to achieve more maintainable applications, especially in a team environment. Application frameworks such as Fusebox, Mach II, Model-Glue, and onTap were discussed and compared. From a code perspective, using CFCs and Web services allow a developer to limit places where code needs to be modified in an application if the business logic changes. As programmers, we can plan and implement the database at whatever point in the design process. For some of us, that's the easy part. The application design itself needs to take priority over the database, especially in an object-oriented world. I found it interesting that OO programming wasn't paraded around as the second coming, but simply using components was

the path to enlightenment. True those are closely tied together, but they don't always go hand in hand. Most people there were already using components and many were doing OO ColdFusion development.

As part of the continued focus on RIAs, I was impressed with how far CFFORM has come. The Flash interfaces that can be produced were remarkable – tabbed input screens, slush boxes (two select boxes with add/ remove capabilities), and calendars are just a glimpse as to what you can do. Yes, some of this existed previously but, as recently as the 7.0.1, new functionality has been added, such as support for ActionScript. I admit to being negative on the validation provided in CFINPUT, namely because it doesn't give you enough control of the JavaScript validation that was created. After seeing Mike Nimer's presentation, I am changing my opinion. With XML forms, you can now create a skin and customize your own validation script.

In the ColdFusion MX 7 Application Framework hands-on session, we covered implementing application events. These are predefined components, like OnSessionStart(), OnApplicationStart(), OnRequestStart(), and OnError(), which ColdFusion checks for in the application.cfc and executes each at specific times. For example, OnApplicationStart() is only processed when the application first begins, regardless of the user. This structure can trim some of the fat from application functions like logins, error checking, and cleanup. If a CFC executes only when a session begins, an if statement checking for session existence is no longer needed for all those variables we set in the application.cfm.

In this class, we were in a lab that had the latest version of Dreamweaver installed. I hadn't had a chance to dig too deep into Dreamweaver yet, but this session gave me the chance to play a little. It has a very nice interface, much cleaner and things that I frequently use are much more accessible now. I was impressed with the CSS capabilities – a dropdown appears in the code view showing which styles are available. Dreamweaver now integrates with a third-party tool from Araxis for file comparison and merging. Small things like improved code collapse and a paste special feature make the newest version of Dreamweaver really nice.

Overall MAX 2005 was another great conference. The Macromedia community is always a fun, intelligent, diverse, and generous group. As a company, Macromedia is determined to continually provide quality products that allow developers to create the best possible tools for their users. As a programmer, I sat in many sessions and could see where topics directly impacted me, from form layout to application framework strategies. It seems that it will become harder and harder for programmers to "just code"; we will have to take some of the responsibility for the usability of our applications – our customers will demand it.

About the Author

Mary-Catherine Gerrey is a senior analyst for Excel Management Systems in Raleigh, NC. She has over seven years ColdFusion experience, incuding a diverse set of industries, from insurance and real estate to hospitality. In addition to a being a ColdFusion Certified Developer, she holds a computer science degree from Mercer University and a Masters in MIS from Florida State University.

duckboxxer@gmail.com

A Taste of MAX 2005

The scene

By Rob Rohan

Wow, what a conference! If you couldn't get to MAX this year, you missed out on a great conference. MAX was held in Anaheim, CA, at the Anaheim convention center. It was four days packed with classes, and the grand finale was a fun-filled night at Disneyland/California Adventure.

There was so much information available at MAX that there is no way I can impart everything I learned at the conference in this article. However, I will do is give a flavor of the classes, a few tips, and provide an overall feel for where Macromedia technology is going.

I'll start with a taste of the classes, give some brief touch points from the General Sessions (Keynotes), and give you the lowdown on the Sneak Peek session (the in-the-works products).

A Taste of the Classes

Most of the classes I attended were about presentation and code design. The classes often had examples of good and bad designs and some tips on how to architect good designs, which stands to reason because, aside from ColdFusion, Macromedia is all about the interface between the person and the computer.

The classes I attended fell roughly into four categories: RIAs (Rich Internet Application) and Flex Builder 2, ActionScript 3, ColdFusion MX, and Flash and AJAX.

RIAs and Flex Builder 2 (aka Zorn)

Macromedia coined the term RIA to describe the new type of Web application with a Flash-based user interface. Initially, it was used only for Flash-based applications, but now it's been adopted to mean AJAX-based (Asynchronous JavaScript and XML) applications as well.

RIAs look cool and have a great *wow* factor. However, the real benefit of an RIA is that the person using the RIA-based application is more productive, and has increased concentration. One of the classes I sat in on, "RIA Design - Best Practices," pointed out that making an application flow correctly and look pleasing engages the user and makes the application more usable. In the end, a more usable UI allows for a more productive user.

A tip for good UI design with RIAs is avoiding the use of excessive *Chrome*. Chrome in this context means excessive eye candy, animation, music, etc. Chrome can distract from the usability and the point of the application. In fact, it's easy to make your application completely unusable and confusing for your clients by using Chrome for Chrome's sake.

To avoid Chrome overload, try to only use transitions or sounds that orient the user when the UI is changing, for example, when a modal menu appears from the side and obscures the user's original view. This kind of transition is a good time to use a bit of animation to let the client know what just happened. Keeping the client engaged, in control, and oriented was a key theme in many of the UI/RIA classes.

The tools to build RIAs are evolving. Flex Builder 2, the IDE previously known as Zorn, was quite a highlight in the RIA track. Flex Builder 2 is the follow up to Flex 1.5 with a bit of a twist. For the uninitiated, Flex 1.5 is the Flex presentation server. Its purpose is to take MXML files, compile them into SWF files, and serve them to the client – it's the core of enterprise-level RIAs. Flex 1.5 needs to run in a J2EE server and its price can be as high as $15,000. The need for a server and the price point tend to turn off smaller shops from creating RIAs.

Flex 2.0 and Flex Builder 2, however, fix both these issues by allowing you to create SWFs based on MXML and deploy them without the server. You'll still be able to exchange data with Web services and raw XML, but instead of deploying them into a specialized presentation server, you can use your current Web server. Using Flex Builder 2 you get the power of rapid application development of RIAs using MXML with the price tag under $1,000 per developer.

Flex Builder 2, from a developer standpoint, is going to turn

some heads too. The design view is very nice. It behaves like Dreamweaver or any graphical GUI designing tool. You can drag-and-drop your way to a beautiful-looking UI in seconds.

The IDE runs on Eclipse, which means it will play nicely with source control solutions like CVS, SVN, and VSS by adding the proper third-party plug-in. It has color coding and intellisence in both the MXML blocks and the ActionScript blocks, and allows you to run and test your Flex application with a single click.

The Flex Builder 2 debugger looks incredible. If you've done any Java programming before, it's just as powerful as the Java debugger in Eclipse. It has watches, breakpoints, and variable inspection. It's a very impressive piece of software and should help keep those late night bug hunting sessions to a minimum when developing RIAs.

To find out more about Flex Builder 2 and Flex 2.0 see Macromedia Labs (http://labs.macromedia.com/wiki/index.php/Flex_Framework and http://labs.macromedia.com/wiki/index.php/Flex_Builder).

ActionScript 3

Along with Flex 2.0, Macromedia is introducing ActionScript 3. I heard a quite a few people at the conference say something to the effect, "ActionScript is becoming a real programming language." It does indeed feel that way when looking at AS3, as they move the language to more closely conform with the ECMA standard.

ActionScript 3 looks and feels a more like C# and Java 5.0 than it does JavaScript. There is a new wrapping block around classes much like the namespace in C#, and there are quite a few new function modifiers like overrides – a keyword that might sound familiar to old VB.NET programmers.

Personally, I think the language looks cool. If you know C# or Java 5.0 you'll get ActionScript 3 easily. If you know JavaScript or ActionScript 2 well, you'll have a somewhat easy time picking up the changes. However, if you've never done object-oriented programming, you're probably going to have to crack open a book or two.

For an example of the language difference, here is a basic class definition in AS2 and the same definition in AS3:

AS2:

```
import my.classes;
class my.pack.TestClass
{
    ...
}
```

AS3:

```
namespace my.pack
{
    import my.classes;
    class TestClass
    {
            ...
    }
}
```

If you've done C# programming (another ECMA-based language), the new version should look familiar. The new keywords and when to use them, however, will be where most new study will be required. A good reference for the transition from ActionScript 2 to ActionScript 3 is available on Macromedia Labs (http://labs.macromedia.com/wiki/index.php/ActionScript_3:Learning_Tips).

ColdFusion MX

The three main themes covered in the ColdFusion classes were:

1. How to design with objects.
2. Frameworks are good.
3. How to use RIAs with ColdFusion.

If you're not using ColdFusion components in an object fashion, you might not be getting the most bang for your buck out of ColdFusion. Objects in ColdFusion were a big theme at the conference – from Raymond Camden's class on the proper usage of the Application.cfc to Simeon Bateman's "ColdFusion Components as Objects" talk.

If you are interested in using RIAs with ColdFusion as the back end, you owe it to yourself to start getting into object-oriented design and frameworks with ColdFusion. Object-oriented designed frameworks tend to fit with RIA frameworks easier, and concepts between RIA and ColdFusion run in parallel.

The ColdFusion frameworks class hit a few general points about frameworks. Points like "frameworks help with the maintainability and longevity of your application," and "the only way to know if a framework is right for your team is to try it." The class also gave a brief introduction to Mach-II, Model Glue, and Fusebox.

In the ColdFusion community there are a few people who think that Macromedia is planning to abandon ColdFusion. If that is true, then they sure are wasting a lot of time making all their technologies work well with ColdFusion. RIAs and ColdFusion work well together.

The ability to pass complex data types between RIAs and servers is extremely easy with ColdFusion (I am fairly positive they did that on purpose). Flex 2.0 is no exception, thanks in part to the CF Adapter, which allows you to take advantage of built-in ColdFusion features such as CFLDAP and CFPOP.

For more information on the CF Adapter check out Macromedia Labs (http://labs.macromedia.com/wiki/index.php/CF_Adapter).

Flash and a Touch of AJAX

An interesting use of Flash is mixing it with AJAX. While at first the idea of using Flash and AJAX sounds kind of redundant, if you think about it, HTML does HTML rendering better than Flash, and Flash does certain things better than HTML. One of the classes I caught was by Mike Chambers who was showing how you can use Flash and AJAX together to take advantage of each technology's strength.

Using the two in tandem opens up some interesting possibilities with UI design. As an example, they showed the Macromedia News Aggregator reports page (http://weblogs.macromedia.com/mxna/reports/mostPopularPosts/index.

cfm?numberOfDays=1). The page is mostly HTML, but has a Flash bar graph at the top. The page itself uses HTML for most of the display, which makes sense for a mostly HTML-centric product like a blog. However, when you select an entry, Flash communicates with the page to load the selected entry into the HTML page with AJAX – no page refresh.

Flash Player 8 can communicate with JavaScript functions on the page natively, but if you want to use Flash Player 7 or lower, you'll have to use the Flash JavaScript bridge provided at OSFlash.org.

General Sessions

The session started out with a bit of comedy from Ze Frank (http://zefrank.com). After his talk I went to his Web site, and I haven't gotten much work done since. He's a funny guy.

Stephen Elop, the CEO of Macromedia, reinforced a few promises to help developers create great user experiences and keep the focus of Macromedia products in line with the customer's need. Mr. Elop was also excited about the new Flash-based mobile market that is taking off in Japan and Europe, and slowly making its way to North America. As an example, FlashCast, which is a service to push Flash content to mobile devices, was just released in Japan.

Kevin Lynch, the chief software architect for Macromedia, gave a nice talk about Web 2.0 and the roadmap for RIAs. He alluded to an in-development project called Apollo, which is a way to use RIAs online and offline, and showed off some amazing examples of RIAs in use today.

The main session was closed by the Adobe CEO, Bruce Chizen, who couldn't say much about the Adobe/Macromedia acquisition because it's not completed, but he was excited about the technology that will come out of the single entity. Mr. Chizen was looking forward to the mobile market and described the current computing environment as an "information explosion." An environment in which digital information is readily available and starting to become transparent between desktop computers, TV, cellphones, and print. He also briefly alluded to something called an "Engagement Platform," which seems to be top secret but has something to do with the bringing together of Adobe's and Macromedia's product offerings.

Sneak Peeks

The Sneak Peek was my favorite part. I love to see what's new and what's been hiding under the covers. There were peeks at pretty much every single product Macromedia has – Contribute and Director, Breeze, Flash, and ColdFusion.

Contribute and Director

Contribute now integrates with Microsoft Internet Explorer. You can browse to a page, click the Contribute icon, and it pops you right into the Contribute editor. You get full WYSIWYG editing. They've also made uploading the changes as simple as a click.

Director got a UI overhaul and moved to a more tab-based approach instead of a floating window approach. The timeline can be made to go horizontal instead of just vertical, which should please a lot of Director developers because Director movies tend to be short with many layers. It also takes advantage of all the new items in Flash 8, and had some code organizing improvements.

Breeze

The new Breeze will allow many people to work on the same screen in real time. Many people logged into a room can all be moving items around, or drawing on the virtual whiteboard.

The synching capabilities are very impressive. Macromedia has partnered with Cisco to include Breeze into some of Cisco's offerings. If I were a betting man, I'd bet you were going to start seeing Breeze used a lot more, by a lot more companies.

Flash

They showed off some of the Flash speed improvements with the new Flash Player – Flash Player 8.5. You get the most speed when you use ActionScript 3. To demonstrate the speed improvements, they showed a Flash movie with hundreds of fish swimming around. Compiled with AS2, the frame rate was around 2/3 frames per second. After writing the application in AS3, the frame rate jumped to 15/16 frames a second. It was breathtaking.

ColdFusion

One major problem with Web UIs currently, even with AJAX and Flash or Flex, is with synching. To sync data between two or more clients, you have to poll. In Flash you can write a socket server, but for the most part, in most real-world applications, your UI has to constantly ask the server at some interval: Did something change? Or, worse, your client has to keep clicking the refresh button on the browser.

The next-generation solutions out of Macromedia seem big on push technology, or pushing data to the client instead of polling. One of the push demos they did was with the new, under-construction, ColdFusion server. The demo showed two browser windows, both with an RIA Web interface with a graph in each window. They were both graphing the same information. The presenter then changed the underlying data set and, lo and behold, both graphs updated to reflect the change, instantly. Neither window had to be refreshed; it simply knew the data had changed and refreshed itself (with no page refresh, of course).

The new push technology will solve one of the Web's most irritating drawbacks – the inability for the server to let the client know something happened. Imagine being able to send a message from the server to the client to tell them their session is about to expire, or refresh a table of data when a new record gets added by a different system.

I can't wait for them to finish that one.

Another boon on the ColdFusion front, for Eclipse users anyway, is the RDS plug-in for Eclipse. Dean Harmon, the gentleman who created Homesite, has been busy creating an RDS plug-in for Eclipse. The new plug-in works famously with CFEclipse. It has an RDS file explorer so you can open and save your ColdFusion files directly on the server. The plug-in also allows you to use RDS data sources set up on the server, run queries against them, and see the results, negating the need for a separate query plug-in and having to set up JDBC drivers on your local machine. In addition, the plug-in also has a visual SQL query designer (much like the one in Microsoft Access) where you see the tables draw lines between them to do joins.

– continued on page page 50

Building a Zip Code Proximity Search with CF

The new challenges and possibilities are numerous

By Troy Pullis

Recently I was tasked with improving our Web site's Reseller Locator application. This tool helps potential customers in the U.S. find a product reseller in their state. By choosing a state from a drop-down box, a listing of all resellers located in that state is displayed.

Over the years, as more and more resellers have signed on to sell our products, some problems with this application have surfaced:

- Some states, such as California, display a very long list of resellers, and customers may never contact those listed near the bottom.
- The states with smaller populations such as North Dakota may not have any resellers to list.
- Competitors could use the tool to easily find all our resellers, and may steal these valuable partner relationships.

The Solution Is Defined

We needed a solution to solve these problems, and searching by zip codes appeared to be the answer. The new tool would ask the customer to enter their five digit zip code in a text box and select a search radius of 25, 50, or 100 miles. We wanted to limit the search to 100 miles, to keep the results to an appropriate number and prevent our competitors from entering 2,000 miles and getting a huge list all at once. For example, I might search for all resellers within 50 miles of 55113. This time a list of resellers in close proximity to St. Paul, MN, is displayed, ignoring those 200 miles north in the city of Duluth. Implementing this approach addressed each problem, respectively:

- A shorter list would result from the search, containing only the resellers located near the customer's zip code.
- The list can cross state boundaries, now that it will find all resellers located within the specified radius to the customer, and hopefully would show results for zip codes in states like North Dakota.
- Competitors can still use the tool to find our resellers, but they'll have to work much harder to get the information with the 100 mile limit.

The "Webmonkey" Demo

I've used many store locator Web applications, such as finding the nearest Quiznos or Best Buy, and always wondered how they worked. Now I had the opportunity to learn something brand new, and I began my quest for how to accomplish the Reseller Locator zip code search by starting where everybody does: Google! I quickly found a great tutorial article with sample ColdFusion code on webmonkey.com. The article was written by Robert Capili and titled Proximity Searches for Fun and Profit. The stars were aligned that day, since Robert published his article three weeks before I started my project. It was perfect timing, because as I started reading, it became apparent that this was exactly what I needed to get my feet wet. In the article Robert discusses four primary ways to calculate zip code proximities:

1. Pythagorean Theorem (remember this trig equation? a2 + b2 = c2)
2. Spherical Law of Cosines (Pythagorean theorem for triangles drawn on a sphere)
3. Haversine Formula (most accurate way to calculate distance on a sphere)
4. Square Search (Robert's speedy solution)

For each method, he explains some background information, how it calculates results, how it performs (speed versus accuracy), and his usage suggestions. I encourage you to read it when you finish this article (or now if you feel the need) to dive deeper into this topic: http://www.webmonkey.com/webmonkey/05/32/index4a.html?tw=programming. In a nutshell, Robert recommends method #4. He improved upon the speed produced by the Pythagorean method and developed the Square Search. It performs the fastest, but does not take into account the spherical shape of the earth, which the Haversine formula is best at. However, as long as you are not in need of absolute pinpoint accuracy, the Square Search is the way to go for most applications.

Testing the Sample Code

Included in the sample code is a ColdFusion Component, zipfinder.cfc, that implements each of the four search methods. It also contains a testing template, zip.cfm, that has a basic search form, containing a zip code text box and radius text box. Finally, it contains a database file, zipDB. My first step to getting set up was a quick visit to our company's DB Admin, who helped me perform a restore of the SQL database file, zipDB. The database restore creates a single table that contains 29,470 zip code records. Next, I opened Application.

query				
	CITY	DIST	STATE	ZIP
1	ROSEVILLE	0	MN	55113
2	LAUDERDALE	2.45504518128	MN	55108
3	LITTLE CANADA	2.65159057914	MN	55117
4	SAINT PAUL	3.84488523652	MN	55103
5	SAINT PAUL	3.9172475506	MN	55114
6	SAINT PAUL	4.14712855304	MN	55104
7	MINNEAPOLIS	4.22017344717	MN	55414
8	SAINT PAUL	4.38889423798	MN	55101
9	MINNEAPOLIS	4.47091704665	MN	55418

Table 1 Square Search Execution Time: 174 ms

cfm and modified the variable application.dsn to use the correct value for our datasource name. Now on to some real testing. I pointed my browser to the zip.cfm template and saw the search form. After entering a zip code and radius in the text boxes, what you get are four cfdumps of the recordsets returned from the CFC, as well as the execution times. Each query gets the city, state, zip code, and distance from the specified zip code. Tables 1 and 2 show the partial output from a search of 55113.

I soon realized why he recommended the Square Search. The execution time was faster than each of the other methods, and the distance results were very close to the Haversine Formula. After a few more tests, it was soon time to put the final pieces together.

Integrating the Solution

Now I needed to take the recordset returned by the Square Search method and put it to use in my application. We have a database table (actually a view) of resellers that includes the field: zipcode. Nothing special here; most of us are familiar with SQL tables that contain address information broken out in separate fields. All I needed to do was create a new query against the reseller table, and make sure a reseller's zip code was found among the zip codes in the recordset returned by the CFC. The code and query looked like this:

```
<cfinvoke component="zipfinder"
method="squareSearch" radius="#URL.miles#"
zip="#URL.zip#" returnvariable="results"></cf-
invoke>

<cfif NOT results.recordcount>
   Sorry, no zip codes found within #URL.
miles# miles of #URL.zip#.
   <cfabort>
</cfif>

<cfquery name="get_resellers"
datasource="#dsn#">
select *
from PartnerView
where zipcode IN (#ListQualify(ValueList(resul
ts.zip),"'")#)
and Country = 'USA'
order by CompanyName
</cfquery>
```

The first block of code invokes the CFC, calling the SquareSearch method. Next, I verify at least one record is returned. If not, I stop right there and let the user know with a friendly error message. Last, I perform a search against our resellers, using the clause:

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
ACTIVEPDF	WWW.ACTIVEPDF.COM	866-468-6733	23
CFDYNAMICS	WWW.CFDYNAMICS.COM	866-233-9626	6
COMMUNITYMX	WWW.COMMUNITYMX.COM/TRIAL		51
HOSTMYSITE	WWW.HOSTMYSITE.COM	877-248-4678	13
HOSTMYSITE	WWW.HOSTMYSITE.COM	877-215-4678	27
INTERAKT	WWW.INTERAKTONLINE.COM/MACROMEDIA	4031-401-68-19	3
INTERMEDIA	WWW.INTERMEDIA.NET	888-379-7729	52
MACROMEDIA	MACROMEDIA.COM/GO/CFMX7_DEMO.	415-252-2000	2
MACROMEDIA	WWW.MACROMEDIA.COM/GO/8_STUDIO8	415-252-2000	11
MXDJ	WWW.SYS-CON.COM/MY/SUBCRIPTION.CFM	888-303-5282	33
PAPERTHIN	WWW.PAPERTHIN.COM	800-940-3087	25
SAVVY	WWW.BESAVVY.COM	866-870-6353	17
SEAPINE	WWW.SEAPINE.COM/WEBDEV	888-683-6456	4
SYS-CON	WWW.SYS-CON.COM/MY/SUBCRIPTION.CFM	201-802-3000	37
VITALSTREAM	WWW.VITALSTREAM.COM	800-254-7554	15
WSE/OP	WWW2.SYS-CON.COM/EVENTS	201-802-3066	21
WEBAPPCABARET	WWW.WEBAPPCABARET.COM/JDJ.JSP	866-256-7973	9
WEBAPPER	WWW.SEEFUSION.COM		49

query				
	CITY	DIST	STATE	ZIP
1	ROSEVILLE	0	MN	55113
2	LAUDERDALE	2.44442437489	MN	55108
3	LITTLE CANADA	2.640477789	MN	55117
4	SAINT PAUL	3.82961292118	MN	55103
5	SAINT PAUL	3.90064412506	MN	55114
6	SAINT PAUL	4.1323354796	MN	55104
7	MINNEAPOLIS	4.20273096666	MN	55414
8	SAINT PAUL	4.3742487002	MN	55101
9	MINNEAPOLIS	4.45361461578	MN	55418

Table 2 Haversine Formula Execution Time: 742ms

```
where zipcode IN (#ListQualify(ValueList(resul
ts.zip),"'")#)
```

Based on the sample records I listed earlier, the actual query might look like:

```
select *
from PartnerView
where zipcode IN ('55113','55108','55117','5510
3','55114','55104','55414','55101','55418')
and Country = 'USA'
order by CompanyName
```

The ListQualify() and ValueList() ColdFusion functions come in handy here. ValueList() converts a query column into a comma-delimited list, then ListQualify() slaps a single quote around each item in the list. This is exactly what is needed to use as the expression on the right-hand side of the IN clause. All that's left is to display the results from the get_resellers query in a nice HTML format, and the customer has the information he or she needs to make a couple of phone calls that will hopefully lead to a future sale!

I Need Reliable Data

Once my application underwent more testing, I found that the sample database was missing some zip codes. I'm not sure where Robert obtained his data, but it definitely was not current. So off I went on another Google search, this time for a zip code data vendor. I ended up purchasing the data I needed for a whopping $5.00. That's right, only five bucks! I bought it from a small company called Team Red Line at http://teamredline.com/zc/default.asp. It included 42,492 records, about 44% more data. I also found that this new data included zip codes for many armed forces locations. However, none of our resellers are located in those zip codes. In addition, these zip codes don't contain any latitude and longitude values, so distances can't be computed. I put a check in for this case, and displayed a friendly error message to the user if he or she entered one of these special zip codes. Feel free to check out the finished Reseller Locator on my company's Web site at http://www.securecomputing.com/partner_locator.cfm.

Now It's Your Turn

As I mentioned earlier, I've used many store locator applications. These usually display the mileage from your zip code for each store location. This distance value was not needed in the Reseller Locator application, but you may want to implement it in your situation.

I also didn't need to output a graphical map, which usually shows an "X" on your location and little icons for each of the store's locations, or perhaps a link to a mapping Web site for driving directions, etc. It would be very cool to integrate the locations into mapping APIs at Google or Yahoo. The new challenges and possibilities are numerous.

I'm sure your mind is spinning right now as you think about where you might put this new technique to practice, so I'll end here and let you get to work. Make sure you visit the webmonkey.com URL and download Robert's code to get started. Keep me posted if you've put any of this knowledge to use or have any helpful tidbits. I hope a few developers out there can apply the zip code proximity search to their Web applications in exciting new ways. Good luck!

Reference

- Capili. R. (12 Aug 2005). Proximity Searches for Fun and Profit: http://www.webmonkey.com/webmonkey/05/32/index4a.html?tw=programming

About the Author

Troy Pullis works as a senior Web developer for Secure Computing Corporation (www.securecomputing.com) and also manages the Twin Cities ColdFusion Users Group (www.colderfusion.com). He is a Certified Advanced CFMX Developer. Back in 1999, Troy shifted his client/server Java programming career to focus on the Internet boom. He immediately started using ColdFusion, began attending CFUG meetings, and has never looked back.

tpullis@yahoo.com

A Taste of MAX 2005

– continued from page 46

Finally, they showed some new AJAX features that will be built into the new Dreamweaver. In about three steps, they created a dynamic AJAX-enabled page. The application wasn't of the Google Maps scope, but it was elegant and simple, and should get you writing AJAX applications in a matter of hours instead of months.

Final Thoughts

There was just so much technology and information available at MAX. I wish I could somehow pass along every nugget of wisdom and every experience I had while at MAX 2005, but that's just not possible without writing a book.

Hopefully, though, a taste of the classes, a bit of the general sessions, and the glimpse of the sneak peeks gave you a flavor of what the conference had to offer, what your development focus might need to be, and what's coming in the next few years.

About the Author

Rob Rohan has been a developer and architect for over seven years. Rob is a YellowBadger.com consultant from the Bay Area and founder of the CFEclipse, Treebeard, and Neuromancer open source projects. He teaches Macromedia products through Shcoonertech, and flies through the air with the greatest of ease.

me@robrohan.com.


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
© Copyright INTERMEDIA.NET, Inc 2005. All rights reserved. All other trademarks are property of their respective holders.