SYS-CON

WELCOME THE ARRIVAL OF ADOBE & WEB 2.0 PG 5

ColdFusion Developer's Journal

ColdFusionJournal.com

November 2005 Volume:7 Issue:11

USING OBJECTS IN YOUR COLDFUSION APPLICATIONS

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.

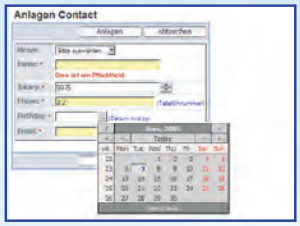
Get a $129 product **FOR FREE!**

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.

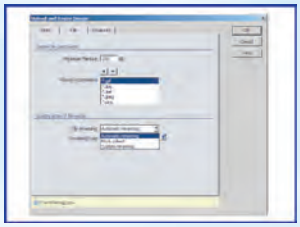
MX Form Validation

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



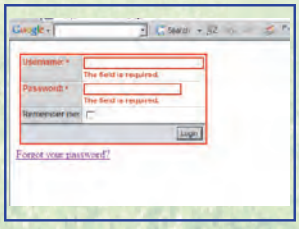
MX File Upload

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



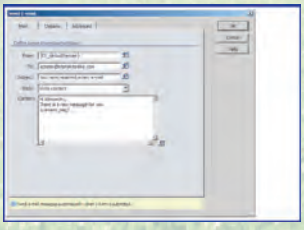
MX User Login

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 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 Future Is Here – Welcome the Arrival of Adobe and Web 2.0

By Simon Horwith

It's official – the Adobe acquisition of Macromedia has been finalized and our beloved ColdFusion has a new home. Is this a bad thing? No, not at all.

There was a lot of talk within the community about how this may adversely effect the server, but talk is cheap and, in this case, also very premature.

In order to give the community a breath of relief and put a stop to much of the hypothesizing, I interviewed Dave Mendels, senior vice president of Adobe's new Enterprise and Developer Solutions Business Unit, about Adobe's plans for the ColdFusion Application Server. Rest assured, ColdFusion's future does look bright – I for one am very excited about the possibilities for new features that lie in store for CF, not just as a developer but as a business person as well. With the right product integration, Adobe could place ColdFusion at the heart of mission-critical applications within large enterprises everywhere where Adobe products are in use. I say this mainly because, after looking at the server product offerings that Adobe had prior to the acquisition, it's clear that CF could well become poised at the center of out-of-the-box offerings in the areas of digital document creation, document management, and workflow. This would never replace ColdFusion's "traditional" role as a Web application development platform that offers rapid development and makes it easy to put databases on the Web, but it would certainly augment those features nicely. You can read my interview with Dave in this month's issue of CFDJ.

While I'm on the subject, I'd like to talk about this idea of ColdFusion's "traditional role" as a Web application development platform that offers rapid development and makes it easy to put databases on the Web. In last month's editorial, I spoke about the changing state of the Web (http://coldfusion.sys-con.com/read/154216.htm) and announced that *CFDJ* will be including more articles that focus on the use of technologies such as AJAX and Flex in order to take our ColdFusion applications, and the experiences that we as developers provide to our customers, to a whole new level. Plans are in the works to begin including Flex 2.0 articles in the near future, but in the meantime you may have noticed the inclusion of more AJAX articles in recent issues. Next month's issue includes an article by Joe Danziger on building an AJAX shopping cart for your CF applications – I recommend trying it for yourself (it's a lot easier than you might think). I welcome all of our readers to provide feedback about these articles and the addition of content focused more on AJAX and Flex for CF developers by e-mailing me at simon@horwith.com.

What I didn't talk about in last month's editorial was how this changing state of the Web will impact the role that ColdFusion plays in your "Web 2.0" applications. Let's face it: ColdFusion could be described as a tool for rapidly putting database data on the Web. This is a simple description and certainly not all-encompassing given what the server is capable of, but it's also not

CFDynamics

Dedicated Server Packages Starting at $189/mo.

All dedicated servers include:

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

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

online editor
Roger Strukhoff roger@sys-con.com

accounting

financial analyst
Joan LaRose joan@sys-con.com

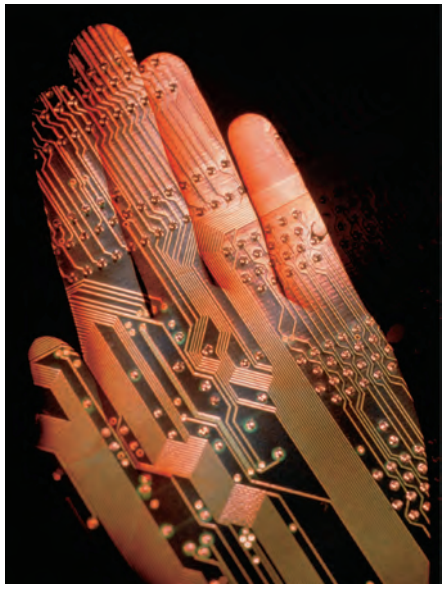
accounts payable
Betty White betty@sys-con.com

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



inaccurate. That is its most popular and most frequently used functionality. ColdFusion is not, but any means, a presentation server… yet the vast majority of ColdFusion applications have HTML front ends and this HTML is typically generated on the server. Don't get me wrong, there's nothing wrong with this, but it's not necessarily the case when we're talking about next-generation applications. In the case of AJAX yes, ColdFusion still generates the client code. In the case of Flex/Flash, this is not the case – the client application is either a precompiled SWF or a SWF that's been generated by a Flex server (unlike CF, Flex *is* a presentation server). Getting back to what Web 2.0 is all about, smart clients are just that: "smart." Whether ColdFusion generates the code for that client or not, the role that ColdFusion plays in these applications has changed. No longer does ColdFusion have to assume the responsibility of not only fetching data but also presenting the data (as well as the interfaces for managing this information). Smart clients need one thing from ColdFusion – data. Of course, sometimes applications do more than simple CRUD operations, so I'll refine my definition of this new role.

In next-generation Web applications, ColdFusion, and ColdFusion applications, will play the part of "service provider." Clients will never be able to perform complex server-side tasks like manipulating enterprise data and other resources, but they are smart enough, with ColdFusion providing the services they need, to give users a much better experience leveraging the enterprise resources that your ColdFusion code provides services for. While ColdFusion's traditional role of exposing services for working with database data may still be the same in the next iteration of the Web, it's now more focused only on that task. Persistence, the presentation of data, and playing the part of controller when users require different views are no longer going to be the responsibility of ColdFusion in these next-generation applications.

What does this mean for ColdFusion developers? Nothing is different for those developers still just working with HTML interfaces, but for the rest of us it means changing the way we architect our ColdFusion applications. There's no reason why you can't have both an HTML interface and Flex or AJAX interface to the same application, but in order to do this, especially without replicating effort, means implementing a sound architecture. By moving all business logic, data access, and manipulation functionality into ColdFusion Components, and developing applications using best practices in order to prevent your business logic from being dependent on any one specific client (including itself) for persistence, the same code can be leveraged by AJAX, Flex, Flash, Sparkle, Java, ColdFusion pages that generate HTML interfaces, and any other client on the Web. So you see, smart clients don't just require learning Flex or AJAX, they also require you to change the way you build your ColdFusion applications. Let me rephrase that, they encourage, even require, developers to encapsulate and modularize their ColdFusion application building blocks… and this is a great thing.

In the months ahead, keep an eye out for news on the Adobe front as well as more articles about the changing Web. Both of these changes in our culture are for the better, and at *ColdFusion Developer's Journal* our aim is to help developers take advantage of these useful new resources.

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

Making It As an Independent Consultant

An overview

By Jeff Houser

I haven't left the house in 11 days, yet I'm not on vacation. I spent 3 hours last Friday morning replacing a broken doorknob, yet the clients don't care. They probably didn't even know. Most of the time they don't even know if I'm out of the office. I'm one of the lucky individuals who have been able to carve out a full-time living as a consultant. Almost all of my work is done on a telecommuting basis. How did I get to this point? It was an interesting road to say the least.

Since this issue is going to be focused on marketing and management, I thought I'd stray from the usual tech related topics of this column to tell the story of how I started DotComIt, my consulting business, and how you can use some of the same tactics to start (or improve) your own consulting adventures.

How It All Begin

I received a computer science degree from Central Connecticut State University during the furor of the Internet bubble balloon. After graduation, my co-op (kind of like an internship) position turned into a full-time job. The company was a business-to-business consulting company. Although they were not a dot-bomb, the hours were crazy and the work was cutting-edge. I was having a blast. I learned something new every day and, unlike a lot of my other college friends, I was actually applying my degree-knowledge every day. Although, I didn't realize it at the time, the style burnt me out rather quickly.

Somewhere in late 1999, I had a vacation request denied. It was my third denial in the past year. A couple of days later, I had a particularly good pizza at lunch and they had my two weeks notice that afternoon. Never ever do that! Acting on a whim is not usually a good idea; always have a plan. I wouldn't recommend this approach to anyone. Any recruiter will tell you that it is easier to find a job while you have a job. Companies like the thought of luring away talent. There is a stigma associated with being jobless.

Fortunately, I was lucky and never had to experience the idiocy of my act. The company I left hired me back as a consultant after a week. I negotiated an overtime pay charge and, for the first time while working for them, they were insistent that I leave on time. From speaking with other people, it is common for the ex-employee to be hired back as a consultant during the "transition" phase while they seek a replacement.

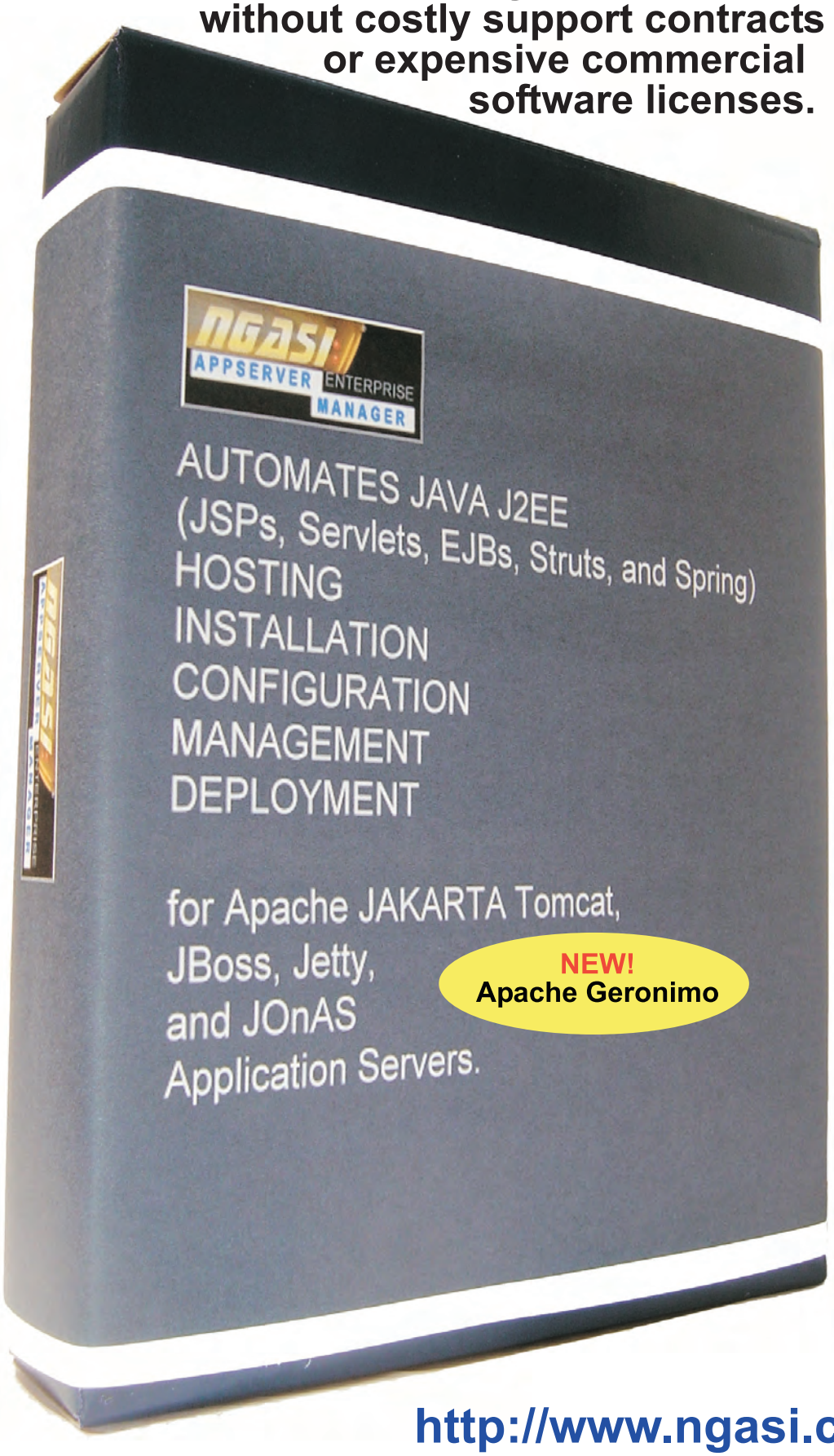
At the time, I was in the Energetic Acoustic Rock Band, Far Cry Fly (www.farcryfly.com). There was a local group of musicians who had banded together in order to help each other get gigs, argue about (the original) Napster, and share info on other music related issues. It was a user group for musicians (very similar in concept to a CFUG). I noticed their site was built in ColdFusion. I had worked with it (but didn't use it much). I asked if they needed any help, and offered it for free. A month later the company became my second client. I met my third client at the kick-off for a ColdFusion user group. Things started to snowball from there, and I incorporated the following year under the name DotComIt, and the rest is history.

Learn How to Network

You may notice the start of a trend in the previous section. I met a lot of people and they hired me. This is known as networking. Meet people! We've all heard it, but not all of us do it in the way that you should. I can't emphasize enough how important this is. "Learn to Network" is the best advice I can give you on how to find a job, consulting or otherwise. So, how do you meet people? Unfortunately, the people you meet on the curb after the bar closes are not likely to remember you, much less hire you for something. You're best bet is to find a more professional organization. Try some of these:

- ***Macromedia User Groups:*** Find your local CFUG or MUG. They are a great way to meet people who are interested in the same things that you are. Many of them work for companies that may need help. Some of them will be consultants, like you, and they may have to outsource work. Find a local user group on the Macromedia site here: http://www.macromedia.com/cfusion/usergroups/index.cfm. If one doesn't exist

JAVA J2EE Hosting Automation Software without costly support contracts or expensive commercial software licenses.



FREE DOWNLOAD

NGASI APPSERVER ENTERPRISE MANAGER

http://www.ngasi.com/jdj.jsp

1.866.256.7973

in your area, you may think of starting one.

- ***Chamber of Commerce:*** Find and join your local chamber of commerce. They usually have networking events on a routine basis. The one I'm in has multiple meetings where the only purpose is to network. They meet at least twice a month for "Business After Hours", and regularly have a business before hours. Most of these are free for members. You may be able to get into some events as a non-member, if that works better for you.
- ***Go to a Trade Show***: There was recently a small business trade show in Connecticut. I went to every booth and handed out a business card. I asked the Web development companies if they outsourced work. I asked the real companies if they had a Web site. I walked out of there with more than a dozen leads to follow-up on.
- ***Other Networking Groups:*** In addition to your local MUG or Chamber of Commerce, look for other networking groups. In my area, I know of a .NET user group and a "computer society" that has a "special interest" offshoot for Web development. There is an object-oriented group, and also a small business owner's group. Most of these are related to the things I do on a daily basis, and attending is a joy. If I attended all of them, I wouldn't do anything else.
- ***Conferences:*** Go to a conference, such as Macromedia (at the time of this writing) MAX or CFUNITED. These conferences offer plenty of networking opportunities and you can educate yourself to boot.
- ***Ask People:*** Ask people on your current contact list (you do have a contact list right?) if there is anyone else they think you should know. Offer to take that new person out to lunch to talk to find out about their job. Tell them what you are looking for.
- ***Blog Syndication:*** If you have a blog and can syndicate it on Weblogs.macromedia.com or FullAsAGoog, you'll never know who is going to start reading. Blogging wouldn't be my first choice if I wanted to meet someone local, but most of what we do can be done off-site. If you find a niche (perhaps Flex and ColdFusion or Mobile Device Development), the results can be appealing.

These are just some ideas to get you started. Let me know what you've done that worked. I'd love to hear your ideas.

Convincing Them You Can Get the Job Done

Now that you're out there and meeting people, a few of them are going to want to hire you in some capacity. They will want to know that you are capable of doing what they want you to do. This is often easy to prove to a technical person, such as yourself, since you both speak the same language. If you're dealing with a non-techie such as a recruiter, HR person, or business owner, you have a bigger hurdle to jump over. How do you prove yourself?

- ***Education:*** "I'm a computer programmer and I have a degree to prove it." As far as I know, very few people actually check to see that you have a degree (but I wouldn't recommend lying about it). Having additional classes or training outside of your education can't hurt. If nothing else, it should keep your mind sharp and on top of current development issues.
- ***Certifications:*** Get the appropriate certifications. The Macromedia certifications are a good start; look here – http://www.macromedia.com/support/training – for more info. I also routinely take new tests on Brainbench, an independent certification authority (http://www.brainbench.com). There always seems to be a debate about the merit of certifications. I understand that not everyone tests well, and that questions are sometimes out of sync with the real world, but the non-techies are almost guaranteed to see certification as a sign of competence.
- ***Demo Site:*** Create a demo site that you can point people to. There is nothing like seeing and using a demo app to help your networking buddy realize that you know what you're doing.
- ***Write:*** Contrary to popular belief, writing articles in magazines like *CFDJ* are not the realm of the super smart uber geeks (or at least they don't have to be). If you have an idea, or an area of expertise pitch them an article (http://coldfusion.sys-con.com/main/proposal.htm). My first article proposal was accepted within hours. When you sit down with the potential client and say something like "I suggest you do this, as I wrote about in x article." And hand them the article. It'll look impressive. Being successful in the book-writing world is a bit harder (see the CF5 book graveyard), but will give you even more street credibility than an article.
- ***Speak:*** Take that area of expertise that you wrote about and prepare a presentation for your local Macromedia User Group. Most likely the manager will be happy to schedule you as a speaker. You can also do general presentations to non-techie groups. I bet talking to a business owner group about how they can use the Web to improve workflow might bring up a few leads. Conferences are another source of speaking engagements, and many conferences put out a "request for papers" in advance of the conference. Watch for them and submit something. By the time you read this, the CFUNITED request for papers will have been closed and Max 2006 will probably not be open yet. Look for other options.

Most likely some combination of these will work.

Where Do You Go from Here?

It is said that the three most important things about real estate is location, location, and location. Well, the three most important things in consulting are network, network, and network. Keep your network active, and actively try to grow it. If you're lucky, it helps, but I don't want to leave that to chance.

About the Author

Jeff Houser has been working with computers for over 20 years. He owns a DotComIt, a web consulting company, manages the CT Macromedia User Group, and routinely speaks and writes about development issues. You can find out what he's up to by checking his Blog at www.jeffryhouser.com.

jeff@instantcoldfusion.com

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

CF's Place in the i-Technology Spectrum

CF, Ruby, RoR, Arf!, MachII, Coldspring – there's a lot going on

By Matthew Woodward

ColdFusion recently took a bit of a jump up to #19 on TIOBE Software's index of most popular programming languages (http://www.tiobe.com/index.htm?tiobe_index). I hadn't looked at this index in a while but it's interesting to see what CF is both above and below, and it's also nice to see the nine green up arrows next to it on the chart.

What's more interesting to me is what this really means. Yes, many have criticized the TIOBE methodology (usually when it doesn't work in their favor), so let's not focus on the specifics. Instead, let's take a look at the bigger picture, where things are going, what I think are some important convergences happening now and likely to happen in the future, and how CF fits into all of this.

First, if you haven't seen them already you need to check out Sean Corfield's blog (http://corfield.org/blog/index.cfm/do/blog.entry/entry/Ruby_and_ColdFusion) and the entry by Hal Helms (http://hal.coldfusionjournal.com/ruby.htm) to which Sean refers. Read these first and let them sink in ...

Now that you're back, let's put this into perspective. I think the major thing Hal points out that's dead on is that there's a shift happening, and it goes far beyond just Ruby. In my mind this shift started quite a while back when people started questioning the complexity of EJB and opting to use POJOs because they're far simpler and they get the job done. Why make things more complex than they need to be?

Even further back there was an initiative in the Java Community Process (JCP) to allow scripting languages to tie in with Java. (Sound familiar?) Again this is in response to unnecessary complexity with the 800 pound gorilla languages like Java. Java's great, I love it, but for a lot of purposes it's just too much.

Enter Ruby, which regardless of what you think about it (jury's still out for me personally) is gaining a lot of attention for these exact reasons. Ruby and specifically the Rails framework takes a great deal of the gruntwork out of web development and as Hal points out, seems to revel in its perceived inferiorities when compared with something like Java. I can't find the study off-hand, but about 9 months ago a study came out outlining the huge productivity gains many companies are seeing when using one of the "P" languages (typically PHP, Python, or perhaps Perl to a lesser extent these days) as opposed to Java. It took businesses a while to get it but the tide is starting to turn, and Ruby is in the right place at the right time.

The really interesting thing to me about Ruby ever since I first looked into it is how gosh darn similar it is in concept to CF. Huge libraries of common functions to make life easier for the developer, and enough power to do fantastic things without introducing unnecessary complexities. The days of wearing a Java badge of pride are starting to fade and companies are more interested in Getting Things Done than they are in telling their buddies on the golf course that they're using EJBs.

The other thing that's going on in CF these days is an explosion of great tools that allow CF to rival many of the other web development technologies out there. Things like Arf! (http://clearsoftware.net/client/index.cfm?mode=entry&entry=49CAFDF4-E081-2BAC-69AD5772ABCF584B), Mach-II (http://www.mach-ii.com/), ColdSpring (http://www.coldspringframework.org/), and the numerous other development and ORM-type frameworks sprouting up these days are great, and they all have their "CF-ness" in common, meaning give CFers the power of these tools without unnecessary complexity, all in the name of getting stuff done faster and better than with any other technology.

Add to all of this ColdFusion 7 itself, which is in my humble opinion light-years ahead of pretty much any other technology out there. At this point I feel like I've still only begun to scratch the surface of what CF 7 can do, and when you combine the

"Keep up the great work CFers and keep on keepin' on. Things are only going to get better from here on out!"

great features of the product itself with the extremely powerful tools becoming available, not to mention CF's long tradition of keeping things simple and keeping developers highly productive, you can start to see how bright the future is looking.

I'm sure I'll have more to say on this in the near future, and I'd love to hear your thoughts as well. I and most other CFers have been fighting the good fight for a long time now and I think finally the IT community as a whole is starting to see the value of what CF has been about at its heart since the beginning. The real bonus is going to be when people see not only that CF solves many of the same problems Ruby purports to (Ruby being the "new hotness" at the moment), but that CF has been around for a very long time, is an extremely mature, feature-rich product, and that it ties in so insanely well with Java so you can use that bit of added power when you need it.

Keep up the great work CFers and keep on keepin' on. Things are only going to get better from here on out!

About the Author

Matt Woodward is a Web application developer for i2 Technologies in Dallas, Texas, and also works as a consultant through his company, Sixth Floor Software. He is a Macromedia Certified ColdFusion Developer, a member of Team Macromedia, and has been using ColdFusion since 1996. In addition to his ColdFusion work, Matt also develops in Java and PHP.

mpwoodward@mac.com

CFDJ Advertiser Index

Introducing Arf! (Active RecordFactory!)

The creator of Model-Glue is back: with Rails-like ActiveRecord for ColdFusion

By Joe Rinehart

I'd like to introduce "Arf!" (Active RecordFactory!), a Rails-style ActiveRecord implementation in ColdFusion. I have to disclaim this by saying that I have mixed feelings about ActiveRecord - it's at the same time very nice for doing things quickly, but I think it's ripe for abuse, as it's easy to think it's the begin and end all of OO programming. However, I think it's also a great way to stop writing some of the same blasted code over and over again for simple projects.

Here's the basics of what the Arf! provides:

- JDBC metadata based reflection: not database specific
- Creates ActiveRecord API'd instances out of CFCs that extend a base ActiveRecord component
- Implements hasMany() and belongsTo() methods for establishing Record properties that point to other tables
- Allows for overloading any of the automagically generated methods to add custom business logic
- Automagic methods on Records include GetInstance(), Create(), Read(), Update(), Delete(), Save() [smart create/update], List(orderBy, whereClause), Validate() [does type and length checking], and SetNNN()/GetNNN() methods for each DB column

Code Demo: Arf! in Ten Steps

So, here's Arf! It's not ready for release yet, but I'd like to introduce it by showing how it'd be used to manipulate a two-table database in ten steps. If you like what you see, please let me know if you'd like to be on the alpha tester list.

Step 1: Create a database, tables, and datasource

I start by creating a database called "arfblog" (in any JDBC-compliant database, I've tested MySQL and MS SQL Server). I give it two tables: blogEntry (blogEntryId, title, body) and comment (commentId, blogEntryId, and message). The primary keys (blogEntryId and commentId) can be auto-incrementing numeric or VARCHAR(35) - if you go the VARCHAR route, Arf! will automagically use UUIDs for the key values. I then create a ColdFusion datasource pointing to the database and call it "arfblog"

Step 2: Configure an Arf! datasource bean

This'd be best done in ColdSpring or ChiliBeans (or your IoC container of choice), but you start by creating an Arf! datasource. It's just a little bean:

```
<cfset ds = createObject("component", "net.clear-
software.arf.Datasource").init() />
<cfset ds.setDSN("arfblog") />
<cfset ds.setDatabase("arfblog") />
```

Step 3: Create an Arf! RecordFactory

Next, we create an instance of the Arf! RecordFactory itself. This is the CFC that performs the Arf! magic.

```
<cfset rf = createObject("component", "net.clear-
software.arf.RecordFactory").init(ds, true) />
```

Step 4: Create an ActiveRecord extension

Now, we need to create a CFC that extends the base Arf! ActiveRecord CFC. It's pretty

simple. I create a BlogEntry.cfc and Comment.cfc from the ActiveRecordTemplate.cfc that comes with Arf!. There's a single line of code that *must* be in the CFC, not in the base ActiveRecord:

```
<!--- BlogEntry.cfc --->
<cfcomponent extends="net.clearsoftware.arf.ActiveRecord"
output="false">

<!--- Don't remove this line of code --->
<cffunction name="getFinalSuper" access="private"><cfreturn super
/></cffunction>

<!--- Add statements like <cfset hasMany() /> and <cfset belongsTo() />
here --->

</cfcomponent>
```

Comment.cfc is identical to BlogEntry.cfc at this point

Step 5: Tell the classes about each other

A comment belongsTo an entry, and an entry hasMany comments, so we add the appropriate lines to each CFC, telling it what to relate to, and what CFC to use when getting related records.

```
<!--- BlogEntry.cfc --->
<cfcomponent extends="net.clearsoftware.arf.ActiveRecord"
output="false">

<!--- Don't remove this line of code --->
<cffunction name="getFinalSuper" access="private"><cfreturn super
/></cffunction>

<!--- Add statements like <cfset hasMany() /> and <cfset belongsTo() />
here --->
<cfset hasMany("comment", "net.clearsoftware.arf.test.Comment") />
</cfcomponent>
```

Now, we do the same for the Comment:

```
<!--- Comment.cfc --->
<cfcomponent extends="net.clearsoftware.arf.ActiveRecord"
output="false">

<!--- Don't remove this line of code --->
<cffunction name="getFinalSuper" access="private"><cfreturn super
/></cffunction>

<!--- Add statements like <cfset hasMany() /> and <cfset belongsTo() />
here --->
<cfset belongsTo("blogEntry", "net.clearsoftware.arf.test.BlogEntry") />
</cfcomponent>
```

Step 6: Create a new blogEntry

Ok, I can ask my factory for some classes now. Let's ask for a new blogEntry.

```
<cfset blogEntry = rf.makeRecord("net.clearsoftware.arf.test.BlogEntry")
```

```
/>
```

That gives us an ActiveRecord. It's got CRUD/S methods, get/set methods for each column in the blogEntry table, a getComment() method added by the hasMany() statement, a validate() method that'll check each property for NULL, length, and type, as well as a list() statement that has optional ORDER BY and WHERE arguments, and a getInstance() method that'll show us internal instance data (by value, where type makes it possible).

Step 7: Populate and save a blogEntry, showing that it's added with list()

```
<cfset blogEntry.setTitle("Some new entry") />
<cfset blogEntry.setBody("A really great blog
entry!") />
<cfset blogEntry.save() />
<cfdump var="#blogEntry.list()#">
```

Not much to it, eh?

Step 8: Create a related comment

Now we'll create a related comment, and commit it.

```
<cfset comment = rf.makeRecord("net.clearsoft-
ware.arf.test.Comment") />
<cfset comment.setBlogEntryId(blogEntry.getBlo-
gEntryId()) />
<cfset comment.setMessage("I'm a wiseacre com-
ment.") />
<cfset comment.save() />
```

Notice that we set its BlogEntryId property to the BlogEntryId value of the BlogEntry we made.

Step 9: Show our relation properties

Now, we'll see where those belongsTo() and hasMany() come into play. Remember saying that the comment belongsTo() a BlogEntry? Adding the following line of code asks the comment for its entry:

```
<cfset commentsEntry = comment.getBlogEntry()
/>
<cfdump var="#commentsEntry#" />
<cfdump var="#commentsEntry.getInstance()#" />
```

The first CFDump shows that getBlogEntry() returns a BlogEntry, and the second shows that its instance data is the data for the BlogEntry.

Now, hasMany() relationships can return more than one instance, so a simple QueryIterator is returned. It's got a pretty straightforward API, and you can get to its underlying query using QueryIterator.getQuery(). Here it is in action:

```
<cfset entryComments = blogEntry.getComment()
/>
<cfdump var="#entryComments.getQuery()#">
<cfset firstComment = entryComments.next() />
<cfdump var="#firstComment.getInstance()#" />
```

The first CFDump shows the query of related comments coming back, the second shows that that we can get a Comment instance from that query, and ask it for its instance data. We could also manipulate its data and save() it because its a fully-operational instance of our net.clearsoftware.arf.test.Comment CFC that extends ActiveRecord.

Step 10: Overriding methods

Because our "shell" CFCs (BlogEntry and Comment) extend ActiveRecord, we can alter the functionality of the default methods that are created for us. If we wanted to change the getTitle() method of BlogEntry to always come back in upper-case, we'd change our BlogEntry.cfc to the following:

```
<!--- BlogEntry.cfc --->
<cfcomponent extends="net.clearsoftware.arf.
ActiveRecord" output="false">

<!--- Don't remove this line of code --->
<cffunction name="getFinalSuper" access="privat
e"><cfreturn super /></cffunction>

<!--- Add statements like <cfset hasMany() />
and <cfset belongsTo() /> here --->
<cfset hasMany("comment", "net.clearsoftware.
arf.test.Comment") />

<cffunction name="getTitle">
   <cfreturn uCase(super.getTitle()) />
</cffunction>

</cfcomponent>
```

Now, to show it works:

```
<cfoutput>
#blogEntry.getTitle()#
</cfoutput>
```

Ah, now that's good and annoying!

Conclusion

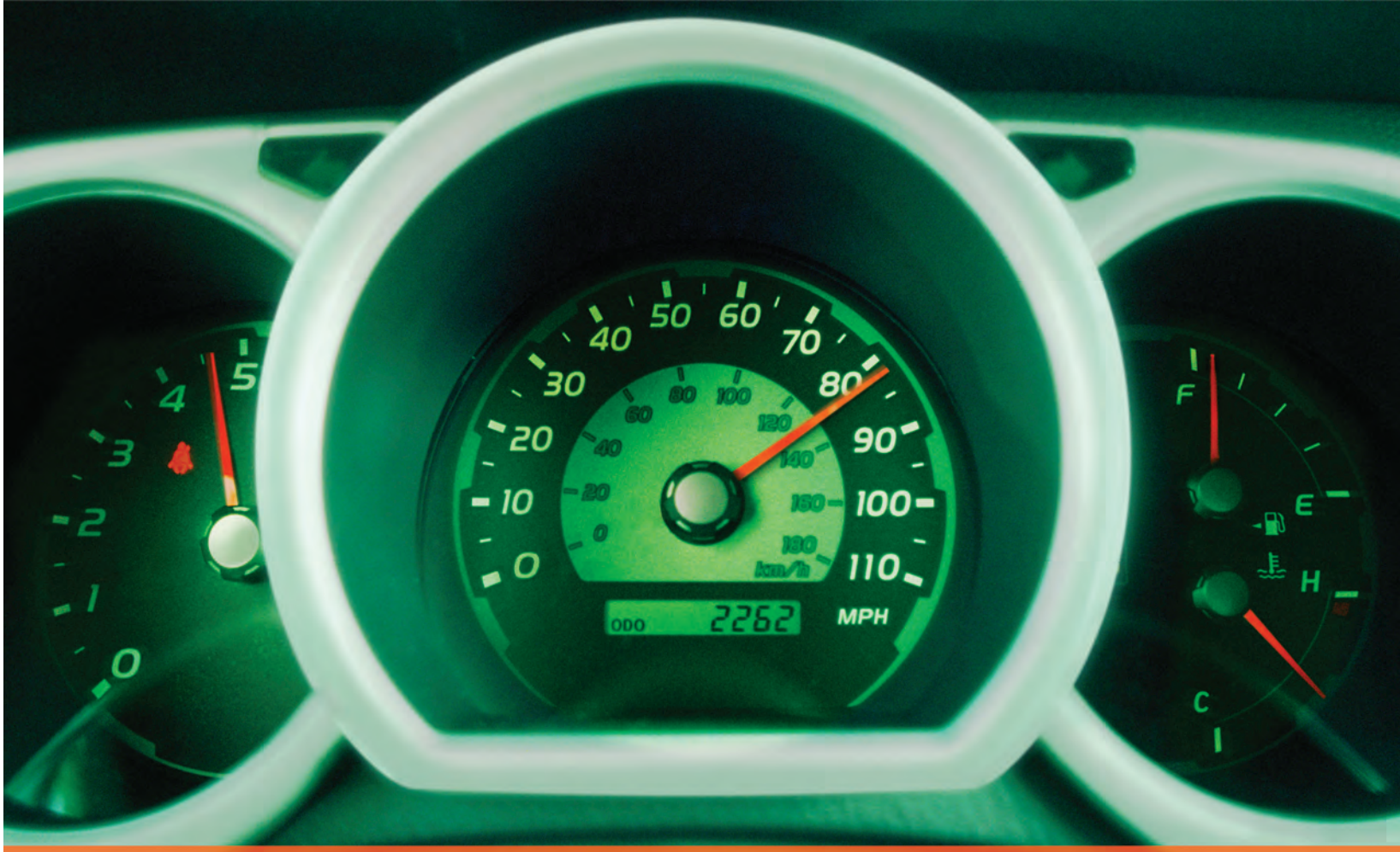
So, we've got a beginning of an ActiveRecord API / Generator working in ColdFusion. There's more to the API than what I've covered here. Hopefully, I'll get it documented and out the door before too long. If you'd like a copy to play with, please leave me a comment and I'll put you on the "alpha" tester list. I should have the code (it'll be LGPL, like ModelGlue) into the clearsoftware.net SVN repo before too long.

About the Author

Joe Rinehart is an Associate with Booz Allen Hamilton, a global strategy and technology consulting firm. He's written the Model-Glue framework (http://www.model-glue.com) to help MVC development in ColdFusion, and runs his own blog at http://clearsoftware.net.

joe.rinehart@gmail.com

"It's not ready for release yet, but I'd like to introduce it by showing how it'd be used to manipulate a two-table database in ten steps.



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

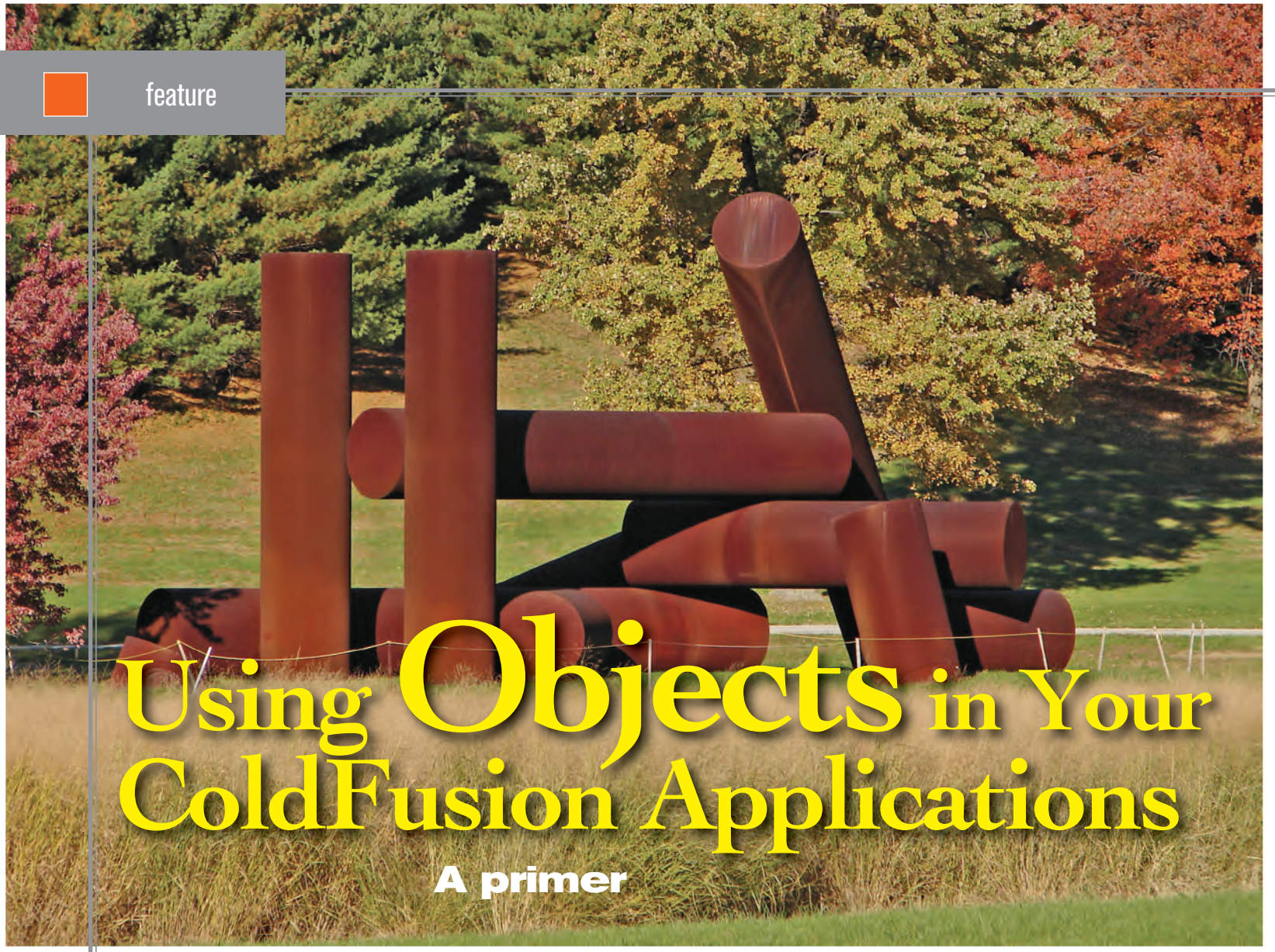
Integrate Streaming Media Into Your Flash Projects

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®



Using Objects in Your ColdFusion Applications

A primer

By Phill Nacelli & Nicholas Tunney

As we searched the Internet looking for a decent definition of objects in object-oriented programming, it became evident that objects themselves are difficult to define. Here we've included a comprehensive definition of exactly what an object is. Don't get scared off by the definition below. It will all become clear as you read on.

In object-oriented programming, an object is defined as a container of related data (properties), and the methods that control inserting and retrieving that data. This process is termed encapsulation. Encapsulation, one of the three key properties of object-oriented programming (OOP), is the process by which access to data or procedures is limited to a specified interface. An object is an instance of a class, which is a model instantiated to create a logical entity that handles storage and management of data (the object). The object can then be used as a single entity throughout your code.

Let's begin by turning this complex definition into a simple example. Objects are all around us. Objects mimic real entities in everyday life. Consider your car. The car itself is an object. It contains attributes (properties) such as its make and model. The car object also contains other objects that have their own properties like the stereo and engine, but that's beyond the scope of this article (and is actually the topic of our next article, which deals with object composition).

Why Use Objects in ColdFusion Code

The answer is pure and simple: readability, reusability, and speed. Object- oriented programming (OOP) is the future of ColdFusion, having been used for over 20 years in other languages such as C and widely popularized by Java programmers. With the introduction of CFCs in ColdFusion MX, OOP

concepts have been made available to CF programmers. It's surprising that more CF coders haven't picked up on this powerful facility.

Invoking a CFC as an Object

As we had defined above, an object has properties and methods. When an object is instantiated, these properties are defined and defaulted (note: object instantiation is defined as the process of creating an instance of a class for execution; an application can have more than one instance of a given object class in memory). Also, as defined above, the class used to instantiate the object contains methods (getters and setters) that encapsulate the object. The properties of the object shouldn't be accessed or dumped directly. To further encapsulate the objects, you must call the getters and setters to access the object's properties. Listing 1 defines an instance of the car object we introduced previously.

This component (CFC) creates an instance of the car object. As you've noticed per the definition the class contains properties of the car object, an init routine, and getter/setter methods to provide a way for you to access the data contained in the object. To instantiate this object in our code, we simply create the object by calling the init method

```
<cfset car = createObject("component", "test.cfc.car.Car").init() />.
```

In case you're new to the ColdFusion createObject function, the first attribute tells the function what type of object you're creating. The second attribute is the full path to the Car.cfc file. In this case it's webroot\test\cfc\car\Car.cfc.

We now have an empty instance of our car object. It becomes much clearer if we dump this object:

As you can see, we have no direct access to the data (encapsulation). We do have methods to access and store the data associated with this object. We can set the make and model of the car with the code:

```
<cfset car.setMake("BMW") />
<cfset car.setModel("525i") />
```

The object properties are now set. The ease of using objects now comes into play when we send the data to our DAO for writing to the database or exporting as XML or any other type of data persistence as a single entity of Car.

Persisting Your Objects – DAOs and Gateways

So now that we know how to create and use objects, let's talk about how to persist them using a relational database for future use. Keep in mind, however, that even though we're using a database here, the persistence layer can be implemented using XML/flat text files, Web Services, registry, or any other non-volatile data storage method.

Suppose we are developing a vehicle tracking application that's supposed to store and display different models of cars. We already talked about how to model a car using our Car.cfc object. Our application has to be able to add, update, or retrieve any instance of a vehicle. We do this by creating a Data Access Object (DAO). DAOs let us separate low-level data access logic from business logic by encapsulating all transactions between the object and the database. One clear advantage of this separation is that you can change the back-end database with minimum impact on the rest of the application. That's the beauty of OOP and MVC architecture.

Listing 2 shows our CarDAO.cfc object and Listing 3 shows us how to instantiate it in our application. Notice the four methods: create, read, update, and delete. These are generally know as CRUD methods. They provide the basic functionality needed for interacting with the database at the record level:

- Create – stores an object instance's current state for later retrieval.
- Read – retrieves saved entity data into an object instance.
- Update – saves an object instance back into the persistence layer.
- Delete – does a physical or logical delete of an object instance's saved state.

We also use another method to help us encapsulate these

Figure 1

transactions further.

- Commit – encapsulates the create and update methods. It frees the business logic from having to determine whether or not the create or update method has to be called.

Note that in our commit, since we can return our identity value (primary key) we're setting it back into our Car object. You can use the same principle in most databases, or even set a UUID, etc. We can now persist this car object, and continue to work with it in our application, committing back to our data provider using the primary key.

Now let's see this concept in action. Suppose a user needs to enter a new car make and model into our carTracker application. Once all relevant data is submitted and passed into an instance of our Car.cfc object using its setter methods, all our business logic has to do is simply pass that instance into the create method in CarDAO.cfc. There's an example in Listing 3.

Now let's retrieve that same instance we just stored by using the read method of CarDAO.cfc along with its known unique key value. Once again we just pass a car object and the key value and all the data retrieval operations are handled in the DAO object. (See Listing 5.)

Throughout our application, we 'll want to be able to retrieve multiple cars from our system to let the user choose from all the different makes and models. We do this by using Gateway objects; simply put, these methods handle the database retrieval of multiple records stored in the persistence layer.

Listing 6 shows our carGateway.cfc object, and listing 7 shows us how to instantiate the gateway in our application. We can now search our carTracker system for all models made by a certain maker as in Listing 8.

Next month, we'll explain how to load a query returned from your gateway into a collection of objects. We'll also introduce you to Object Composition, which lets you combine simple objects into complex objects. This means that your object can contain instances of other related objects. If you're unfamiliar with object composition, this powerful offering may change the way you program ColdFusion forever!

About the Authors

Phill Nacelli has been developing software for over 7 years, and have been using ColdFusion since version 4.5. He has engineered and developed multiple web based applications for the non-profit association/education market and enjoys playing with the latest in programming techniques, frameworks and development tools. He recently started a position as Senior ColdFusion Developer at Intelliworks, Inc., a CRM Software company based in Rockville, MD. His website and blog is http://www.phillnacelli.net/.

Nicholas Tunney is a Macromedia Certified ColdFusion developer, and has been programming ColdFusion for over 7 years. He is currently Senior Software Architect for AboutWeb, a consulting firm located in Rockville, Maryland. To learn more about using objects in ColdFusion, visit Nic's Blog at http://www.nictunney.com.

nic.tunney@nictunney.com

pnacelli@nacelliconsulting.com

Listing 1

```
<cfcomponent name="Car" hint="creates an instance of Car">
    <!--- car properties --->
    <cfset variables.ID = 0 />
    <cfset variables.make = "" />
    <cfset variables.model = "" />

    <!--- constructor method --->
    <cffunction name="init" access="public" returntype="Car"
output="false">
            <cfreturn this />
    </cffunction>

    <!--- class getter and setter methods --->
    <cffunction name="getID" access="public" returntype="string"
output="false" hint="gets ID property">
            <cfreturn variables.ID />
    </cffunction>
    <cffunction name="setID" access="public" returntype="Void"
output="false" hint="sets ID property">
            <cfargument name="ID" type="string" required="yes" />
            <cfset variables.ID = arguments.ID />
    </cffunction>

    <cffunction name="getMake" access="public" returntype="string"
output="false" hint="gets Make property">
            <cfreturn variables.make />
    </cffunction>
    <cffunction name="setMake" access="public" returntype="Void"
output="false" hint="sets Make property">
            <cfargument name="make" type="string" required="yes" />
            <cfset variables.make = arguments.make />
    </cffunction>


    <cffunction name="getModel" access="public" returntype="string"
output="false" hint="gets Model property">
            <cfreturn variables.model />
    </cffunction>
    <cffunction name="setModel" access="public" returntype="Void"
output="false" hint="sets Model property">
            <cfargument name="model" type="string" required="yes" />
            <cfset variables.model = arguments.model/>
    </cffunction>
</cfcomponent>
```

Listing 2

```
<cfcomponent name="CarDAO" hint="Data Access Object">

<!--- properties --->
<cfset variables.dsn = "" />

<!--- constructor init() method --->
<cffunction name="init" access="public" returntype="CarDAO"
output="false" hint="initiates instance of CarDAO">
    <cfargument name="dsn" type="string" required="yes" />
    <!--- initiate value(s) --->
    <cfset variables.dsn = arguments.dsn />

    <cfreturn this />
</cffunction>

<!--- create,read,update,delete methods --->
<cffunction name="create" access="public" returntype="Void"
output="false" hint="CRUD method">
    <cfargument name="car" type="cfc.test.car.Car" required="yes" />

<cftransaction>
    <cfquery name="qCreate" datasource="#variables.dsn#">
    INSERT INTO
            Cars
    (
            Make,
            Model
```

```
    )
    VALUES
    (
            <cfqueryparam cfsqltype="cf_sql_varchar" value="#arguments.
car.getMake()#" />,
            <cfqueryparam cfsqltype="cf_sql_varchar" value="#arguments.
car.getModel()#" />
    )
    SELECT @@IDENTITY AS newID
    </cfquery>
    <cfset arguments.car.setID(qCreate.newID) />
    </cftransaction>
</cffunction>

<cffunction name="read" access="public" returntype="Void"
output="false" hint="CRUD method">
    <cfargument name="car" type="cfc.test.car.Car" required="yes" />
    <cfargument name="carID" type="String" required="yes" />

    <cfset var qRead = 0 />
    <cfquery name="qRead" datasource="#variables.dsn#">
            SELECT
                    Make
                    Model
            FROM
                    Cars
            WHERE
                    ID = <cfqueryparam cfsqltype="cf_sql_integer"
value="#arguments.carID#" />
    </cfquery>
    <cfif qRead.RecordCount>
            <cfset arguments.car.setID(arguments.CarID) />
            <cfset arguments.car.setMake(qRead.Make) />
            <cfset arguments.car.setModel(qRead.Model) />
    <cfelse>
            <cfthrow type="emptyRecordset" errorcode="CarDAO.read.empty-
Recordset" message="Car with CarID #arguments.carID# not found" />
    </cfif>
</cffunction>

<cffunction name="update" access="public" returntype="Void"
output="false" hint="CRUD method">
    <cfargument name="car" type="cfc.test.car.Car" required="yes" />

    <cfquery name="qUpdate" datasource="#variables.dsn#">
            UPDATE
                    Cars
            SET
                    Make = <cfqueryparam cfsqltype="cf_sql_varchar"
value="#car.getMake()#" />,
                    Model = <cfqueryparam cfsqltype="cf_sql_varchar"
value="#car.getModel()#" />
            WHERE
                    ID = <cfqueryparam cfsqltype="cf_sql_integer"
value="#arguments.car.getID()#" />
    </cfquery>
</cffunction>

<cffunction name="delete" access="public" returntype="Void"
output="false" hint="CRUD method">
    <cfargument name="car" type="cfc.test.car.Car" required="yes" />

    <cfquery name="qDeletePresenter" datasource="#variables.dsn #">
    DELETE FROM
            Cars
    WHERE
            ID = <cfqueryparam cfsqltype="cf_sql_integer"
value="#arguments.car.getID()#" />
    ;
    </cfquery>

</cffunction>
```

```
<!--- commit methods --->
<cffunction name="commit" access="public" returntype="Void"
output="false" hint="commit method">
    <cfargument name="car" type="cfc.test.car.Car" required="yes" />

    <cfquery name="qExists" datasource="#variables.dsn#">
           SELECT
                    ID
           FROM
                    Cars
           WHERE
                    ID = <cfqueryparam cfsqltype="cf_sql_integer"
value="#arguments.car.getId()#" />
           ;
    </cfquery>

    <cfif qExists.recordcount>
           <cfset this.update(arguments.car) />
    <cfelse>
           <cfset this.create(arguments.car) />
    </cfif>
</cffunction>

<!--- public methods --->
<cffunction name="validate" access="public" returntype="Boolean"
output="false" hint="I validate all properties in Car Object">
<--- init local var(s) --->
    <cfset var valid = true />
<!--- validate properties --->
<!--- check if make is a valid string --->
    <cfif NOT isValid("String",this.getMake())>
           <cfset valid = false />
    </cfif>
<!--- check if model is a valid string --->
    <cfif NOT isValid("String",this.getModel())>
           <cfset valid = false />
    </cfif>
<cfreturn valid />
</cffunction>
</cfcomponent>
```

Listing 3

```
<!--- application.cfm or some other application init routine --->
  <cfif NOT isDefined("application.dsn")>
    <cfset application.dsn = "dsnCarTracker" />
  </cfif>

<cfif NOT isDefined("application.carDAO")>
  <cfset application.carDAO =  createObject("component","carTracker.
car.carDAO").init(application.dsn) />
</cfif>
```

Listing 4

```
<!--- code in application's controller (business logic layer) --->
  <!--- instantiate car object -‡
    <cfset car = createObject("component","carTracker.car.Car").init()
/>

  <!--- get data from submitted form --->
    <cfset car.setMake(form.make) />
    <cfset car.setModel(form.model) />

  <!--- perform server side validation; save if valid, send user back
if not valid --->
    <cfif car.validate()>

      <!--- store car into persistence layer --->
        <cfset application.carDAO.create(car) />
      … some other code …

    <cfelse>

      <!--- make object available to previous page for correction --->
        <cfset session.car = car />

      <!--- send user back to new car form --->
        <cflocation url="#cgi.script_name#?event=getCar" />
   </cfif>
```

Listing 5

```
 <!--- code in application's controller (business logic layer) --->

  <!--- instantiate car object --->
    <cfset car = createObject("component","carTracker.car.Car").init()
/>

 <!--- get instance of car from persistence layer --->
    <cfset application.carDAO.read(car,form.carId) />

<!--- code in display file (view layer) --->
  <!--- this output shows data in car instance --->
  <cfoutput>
    Car Make:  #car.getMake()#<br />
    Car Model: #car.getModel()#<br/ >
  </cfoutput>
```

Listing 6

```
<cfcomponent name="CarGateway" hint="CarGateway - Gateway Object">

    <!--- properties --->
        <cfset variables.dsn = "" />

    <!--- constructor init() method --->
        <cffunction name="init" access="public" returntype="CarGateway"
output="false" hint="initiates instance of CarGateway">
            <cfargument name="dsn" type="string" required="yes" />
            <!--- initiate value(s) --->
            <cfset variables.dsn = arguments.dsn />
            <cfreturn this />
        </cffunction>

    <!--- public methods --->
        <cffunction name="getByMake" access="public" returntype="query"
output="false" hint="I get all models for a given make">
            <cfargument name="make" type="string" required="yes" />
            <!--- init local var(s) --->
                <cfset var qGetByMake = 0 />
            <!--- retrieve cars by make --->
                <cfquery name="qGetByMake" datasource="#variables.
dsn#">
                    SELECT
                        id,
                        model
                    FROM
                        Cars
                    WHERE
                        make = <cfqueryparam cfsqltype="cf_sql_varchar"
value="#arguments.make#" />
                </cfquery>
            <cfreturn qGetByMake />
        </cffunction>

</cfcomponent>
```

Listing 7

```
<!--- application.cfm or some other application init routine --->
  <cfif NOT isDefined("application.carGateway")>
    <cfset application.carGateway = createObject("component","carTracke
r.car.carGateway").init(application.dsn) />
  </cfif>
```

Listing 8

```
<!--- code in application's controller (business logic layer) --->

  <!--- get submitted maker variable from search form --->
    <cfset searchResults = application.carGateway.getByMake(form.make)
/>

<!--- code in display file (view layer) --->
  <!--- this output shows the search results --->
    Car Maker:  #form.make#<br />
    <ul>
    <cfoutput query="searchResults">
        <li>#searchResults.model#</li>
    </cfoutput>
    </ul>
```

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 DEVELOPER'S 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

Visualize Your Visitors with Google Maps

The new art of geolocation

By Joe Danziger

The availability of the Google Maps API has opened up a slew of new development possibilities and spawned a bunch of web sites that rely on this data to function. Example applications include jogging trackers, subway map overlays, and other cool ideas all made possibly by the public availability of the mapping API.

Geolocating by IP Address

Geolocation refers to the ability to locate a user's geographic latitude and longitude from his IP address. There are several ways of doing this each with its own benefits and drawbacks.

The most cost-effective source of this data is one of the publicly available databases: NetGeo, originally set up by the Cooperative Association for Internet Data Analysis (CAIDA). Although this project was stopped several years ago, the database is still available and provides a good resource for determining rough locations. Queries to this database are free, but limited. You can only send a limited number of requests through every few minutes.

You query against the NetGeo database by calling the URL http://netgeo.caida.org/perl/netgeo.cgi?target=64.19.164.74. It should return the following data to the browser:

VERSION=1.0

TARGET: 64.19.164.74
NAME: MONMOUTH-BLK2
NUMBER: 64.19.128.0 - 64.19.191.255

Figure 1

CITY: RED BANK
STATE: NEW JERSEY
COUNTRY: US
LAT: 40.35
LONG: -74.08
LAT_LONG_GRAN: City
LAST_UPDATED: 01-Jun-2001
NIC: ARIN
LOOKUP_TYPE: Block Allocation
RATING:
DOMAIN_GUESS: MONMOUTH.COM
STATUS: OK

You have some flexibility in how you store visitor IP addresses. My preferred method is to create a login_record table so that the data is always available, even after a restart.

We can use ColdFusion's string functions to parse out the relevant latitude and longitude data. We'll also parse out the city, state, and country and display that info in a pop-up balloon when each point is clicked. Listing 1 contains the code to parse out these fields.

Another free source of data is the community-based project at http://www.hostip.info. This is a newer database but it's actively being updated. You can issue queries over the web via http://api.hostip.info/get_html.php?ip=64.19.164.74&position=true, which will return results similar to NetGeo. You can also download a full copy of its IP to the location database on the site.

There is also a commerical GeoIP City Database maintained by MaxMind at. http://www.maxmind.com/app/city.

Drawing The Map

The first thing you'll have to do if you haven't already is to sign up for a Google Maps API key at http://www.google.com/apis/maps/. This key will have to be included in any pages you're drawing maps on.

In the <head></head> section of your page, you'll have to include the following script call:

```
<script src="http://maps.google.com/maps?file=api&v=1&key=YOUR_API_KEY"
type="text/javascript">
</script>
```

The next step is to include an empty div right after your <body> tag which will hold the map:

```
<div id="map" style="width: 700px; height: 500px"></div>
```

Now that we've got all of the coordinate data, plotting points on the map is rather simple. Include the script call in Listing 2 somewhere on your page

Resources

- Google Maps Documentation - http://www.google.com/apis/maps/documentation/
- NetGeo Public Database - http://www.caida.org/tools/utilities/netgeo/
- Hostip.info Community Database - http://www.hostip.info/
- MaxMind Commercial Database - http://www.maxmind.com/app/city/
- GVisit – http://www.gvisit.com/

About the Author

Joe Danziger is the founder and president of DJCentral.com, an online promotional tool for disc jockeys and other members of the electronic dance music industry. He has been developing professional ColdFusion solutions for over six years since version 1.5.

danziger@yahoo.com

Listing 1

```
<cfscript>
citystart   = find('CITY: ',cfhttp.filecontent)+6;
cityend     = find('<br>',cfhttp.filecontent,citystart);
city        = mid(cfhttp.filecontent,citystart,cityend-citystart);
statestart  = find('STATE: ',cfhttp.filecontent)+7;
stateend    = find('<br>',cfhttp.filecontent,statestart);
state       = mid(cfhttp.filecontent,statestart,stateend-statestart);
countrystart    = find('COUNTRY: ',cfhttp.filecontent)+9;
countryend  = find('<br>',cfhttp.filecontent,countrystart);
country     = mid(cfhttp.filecontent,countrystart,countryend-countrys-
tart);
latstart    = find('LAT: ',cfhttp.filecontent)+4;
latend      = find('<br>',cfhttp.filecontent,latstart);
lat         = mid(cfhttp.filecontent,latstart,latend-latstart);
longstart   = find('LONG: ',cfhttp.filecontent)+5;
longend     = find('<br>',cfhttp.filecontent,longstart);
long        = mid(cfhttp.filecontent,longstart,longend-longstart);
</cfscript>
```

Listing 2

```
<script type="text/javascript">
  //<![CDATA[
  var map = new GMap(document.getElementById("map"));
  map.addControl(new GLargeMapControl()); // adds control for panning/
zooming
  map.addControl(new GMapTypeControl()); // adds control for switching
between Map & Satellite Mode
  map.centerAndZoom(new GPoint(-74.04, 40.32), 15); // draws map cen-
tered at this point - last parameter is zoom level which can be from 1
(street level) to 17

  <cfloop query="getIPs">
  <cfhttp
url="http://netgeo.caida.org/perl/netgeo.cgi?target=#ipaddress#"></
cfhttp>

<!--- INCLUDE PARSING CODE FROM LISTING 1 HERE --->

<!--- EACH MARKER MUST BE UNIQUE SO WE'LL USE #CURRENTROW# BELOW TO MAKE
THEM UNIQUE --->
  var point = new GPoint(#trim(long)#,#trim(lat)#);
  var marker#currentRow# = new GMarker(point);
  var html#currentRow# = "#city#, #state# #country#"; // sets HTML for
popup balloon
  GEvent.addListener(marker#currentRow#, "click", function() {
marker#currentRow#.openInfoWindowHtml(html#currentRow#); }); // adds
popup event to marker
  map.addOverlay(marker#currentRow#); // plot marker on map
  </cfloop>

  //]]>
</script>
```

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

Where's ColdFusion Headed Under Adobe?

Interview with Dave Mendels

Senior Vice President of Enterprise & Developer Solutions Business Unit, Adobe

"Adobe has been very successful in selling into the enterprise. This can only help ColdFusion going forward," says Dave Mendels, in this exclusive interview with *ColdFusion Developer's Journal*. Here Dave discusses "Scorpio," how the ColdFusion product development team is already hard at work devising the best way to harness synergies between CF and Adobe's LiveCycle products, and more.

Simon Horwith: You must be very excited about the completion of the Adobe acquisition. Can you share your thoughts on what it will be like working at Adobe?
Dave Mendels: Adobe is an innovative, exciting company that I'm thrilled to now be a part of. As we've worked through the details of the acquisition, it has become clear to me that Adobe and Macromedia share a common vision and commitment to customers and technology innovation. We have overlapping customers, and not that much overlap in technologies. The end result is that I really believe that the new combined company is greater than the sum of its parts.

CFDJ: This is the second time that ColdFusion has been acquired by a new company. The Macromedia/Allaire acquisition ended up being a very positive step in the evolution of ColdFusion, despite initial community fears and misgivings. What does this acquisition mean for ColdFusion and its legions of dedicated users?
DM: The ColdFusion product team is already working on the next great release of ColdFusion, codenamed Scorpio, a project that was started before the merger was announced, and one that has continued throughout the transition. Now that we are part of Adobe, we have many new opportunities to leverage and integrate with other Adobe technologies to bring more power to ColdFusion. The product team is excited about the possibilities and is currently hard at work evaluating how to best leverage Adobe's powerful set of technologies for use in Scorpio. In addition, Adobe has been very successful in selling into the enterprise. This can only help ColdFusion going forward.

CFDJ: The developer segment of the market is not one that Adobe had traditionally been involved in. What do the Adobe upper management and decision makers think of ColdFusion and its users?

DM: First of all, Adobe management and decision makers are a very good mix of folks who have been at Adobe and folks who have been at Macromedia and Allaire – so it is not like you have a whole new team. Kevin Lynch is still our chief software architect. I am still running the CF and Flex business (and now LiveCycle too). Damon Cooper is still leading the CF product development team. Ben Forta is still our evangelist. And so on. The new Adobe is indeed very interested and committed to the developer market. This is in fact one of the big reasons that Adobe was interested in acquiring Macromedia. Adobe recognizes that ColdFusion is an important, healthy part of the business and has a continued commitment to move the ColdFusion business forward. This begins with Scorpio

CFDJ: Okay, then where does ColdFusion fit into the Adobe product strategy?
DM: ColdFusion fits squarely in the new Enterprise and Developer Solutions Business Unit. Here we will focus on leveraging the combined LiveCycle, Flex, and ColdFusion technologies to provide a rich set of technologies for building and deploying both Web and document-based solutions. The developer market has, of course, always been a key market for Macromedia and is now also a key market for Adobe.

CFDJ: Can you offer any insight into potential future integration between ColdFusion and Adobe products and technologies?
DM: We'll it is too soon to make any specific announcements – we have a lot of work to do. After recently shipping Merrimack (aka ColdFusion MX 7.0.1) the whole CF team has been totally focused on planning for the future.

Having said that, there are two types of integration that we are looking into. First, there are capabilities in both ColdFusion and Adobe's LiveCycle products that could provide mutual benefit to each other. These include reporting, workflow, document generation, forms design, charting and graphing, and more. The respective product teams are now working together to determine how to best leverage these capabilities in both product lines.

Then there are some obvious areas like PDF document generation. Given that we introduced PDF document generation in its most basic form in the last major release of ColdFusion, it will be natural for us to extend this capability to leverage the rich set of capabilities offered in PDF. Things like digital signatures, dynamic forms, workflow, and more. Adobe also offers a forms designer product that could potentially offer ColdFusion a more productive forms design capability.

I'm not committing to anything here, but the product team is looking at all of this and more. And we want your input. As usual, send us ideas at wishcoldfusion@macromedia.com.

CFDJ: Can we expect any changes to the pricing or packaging of ColdFusion?
DM: No, we don't have any plans in this area. It is always possible at some time in the future, but you shouldn't expect any specific changes based on the acquisition by Adobe to current offerings.

CFDJ: Community interaction and involvement have long been the hallmark of the ColdFusion experience. Be it bloggers, user groups, conferences, the community has always enjoyed a close relationship with the ColdFusion team. Will this relationship continue under Adobe, and can we expect Macromedia to nudge Adobe into becoming a more developer-centric company?
DM: Yes. The same people who have been building and maintaining ColdFusion over many years are continuing on with the product. We are absolutely committed to maintaining continuing communications with the ColdFusion community. The Macromedia team is now an integral part of Adobe so you can expect that the community focus that you've experienced in the past will continue moving forward. When Macromedia acquired Allaire, lots of Allaire customers worried that Macromedia was not as developer-centric as Allaire had been. Those fears proved unwarranted; Macromedia took the best of the Allaire developer experience and combined it with the best of the Macromedia experience, and the end result proved beneficial to all. I expect that the same will happen here. By the way, check out Adobe's bloggers at blogs.adobe.com.

CFDJ: Do you have any final comments on the future of ColdFusion?
DM: Yes, I do. ColdFusion and the loyal community of developers who have played such a key role in its evolution for the past 10 years are important to Adobe, today and in the future. As ColdFusion is acquired and integrated into its new Adobe family, I remain passionate about ensuring that Adobe continues to focus resources on moving ColdFusion forward, leveraging the rich family of products and technologies now offered by Adobe.

ColdFusion MX7 has proven to be an incredibly successful product. This success is primarily the result of combining a proven solid architecture with features and technologies that solve real problems. Incidentally, the re-architecture that was ColdFusion MX means that ColdFusion and LiveCycle are built on the same Java foundation – the pieces fit together really nicely.

The ColdFusion product team is cooking up some exciting new capabilities for Scorpio and we look forward to once again providing new, critical advancements in what developers can build and how they build them. And finally, we thank you for your continuing commitment to ColdFusion and we look forward to delivering innovative, groundbreaking new releases in the months and years to come.

"Adobe recognizes that ColdFusion is an important, healthy part of the business and has a continued commitment to move the ColdFusion business forward"

Creating Subreports with the Report Builder in ColdFusion MX 7

A tutorial

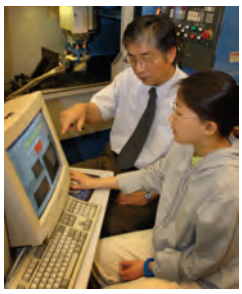
By Jim Bambrough

With Macromedia ColdFusion MX 7, you can use subreports to show more detailed reporting. A *subreport* is simply a report embedded or nested inside another report. I refer to the report that contains the subreport as the main report from this point forward. The subreport and *main report* are typically related by the nature of their content.

In this article on ColdFusion reporting, you'll learn a couple of techniques for linking a subreport to a main report. The focus will be placed on linking subreports using the ColdFusion Report Builder.

This article covers several key concepts used for linking subreports to main reports, including how to bind reports with parameters. I discuss two techniques on how to link subreports:

- Linking an existing subreport to a main report
- Creating a new subreport ad hoc and linking it to the main report



Having had the daunting task of supporting reporting applications through Java applets (and I barely survived that one!), I have come to appreciate the simplicity and ease of development that comes with the new reporting paradigm. I hope that after reading this article, you will also appreciate the power and rapid development that is now available with ColdFusion reporting in ColdFusion MX 7.

This article originally appeared on the Adobe Developer Center. Reprinted with permission

Requirements

ColdFusion MX 7.0.1

For a trial download: www.macromedia.com/cfusion/tdrc/index.cfm?product=coldfusion&promoid=devcenter_tutorial_product_090903

To buy: www.macromedia.com/software/coldfusion/buy/?promoid=devcenter_tutorial_coldfusion_090903

Note: If you have already installed ColdFusion MX 7, please install the ColdFusion 7.0.1 updater. Also install the ColdFusion sample applications that come with ColdFusion. If you have not installed the sample applications, run the ColdFusion installer again and select Sample Applications during the installation process.

ColdFusion Report Builder

ColdFusion Report Builder (EXE, 14.7 MB): www.macromedia.com/support/coldfusion/downloads_updates.html

Tutorials and Sample Files

http:download.macromedia.com/pub/developer/subreporting.zip

Prerequisite Knowledge

Recommended skills include a basic knowledge of ColdFusion and some reporting experience. However, this article will step through the sample subreport without assuming these skills. If you are new to ColdFusion reporting, please read "Building Reports with ColdFusion MX 7" by Collin Tobin and Dean Harmon (www.macromedia.com/devnet/coldfusion/articles/reporting.html).

ColdFusion Reporting

One common function of many companies is producing reports. There have been many software packages that offer these types of solutions, including Crystal Reports, Actuate, and Microsoft Access. Over time it has become more feasible and efficient to offer reporting through Web applications. Previously, ColdFusion offered Web reporting capabilities using

the cfreport tag. With ColdFusion MX 7, however, developers have access to the new ColdFusion Report Builder to develop online reports.

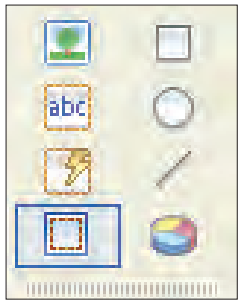


Figure 1 Report Builder Toolbox with the Subreport component selected

The ColdFusion Report Builder tool is a separate software program included with ColdFusion MX 7 that integrates with your ColdFusion server for report development. When you use the ColdFusion Report Builder to create a report, it produces a separate file with a proprietary format, and extension of CFR. You refer to the filename in the template attribute of the cfreport tag. The cfreport tag has been available in ColdFusion for some time, but with ColdFusion MX 7 it has more functionality. You can configure the Report Builder to point to the Web root of your ColdFusion server for report previews. You can easily configure your report as FlashPaper, PDF, or an Excel spreadsheet.

The Subreport Component

Typically, the ColdFusion Report Builder is used to design reports embedded in a ColdFusion page. The ColdFusion Report Builder interface is similar to that of Macromedia Flash Professional 8. There are a variety of components available in the ColdFusion Report Builder tool, including a label and a field component, each of which has its own set of properties (see Figure 1). You can change these properties when you highlight a particular component or element of your report in the Property inspector. In this tutorial, you'll use a subreport component to link a main report to an existing subreport and link a main report to a new subreport.

You use the Subreport component to embed subreports in a main report. The next section describes how to bind the subreport to the main report using subreport parameters.

Binding Subreports to Main Reports with Parameters

Generally, you can link a subreport to a main report by passing a value from the query in the main report as a parameter to the subreport; this value filters data in the subreport query. You can link the reports through the Subreport Parameters property of the subreport component. Find this property in the Data category of the subreport properties (see Figure 2).

This property corresponds to the input parameters of the subreport. The name of this property must match the name of the input parameter of the subreport. The Subreport Parameters property is an expression usually mapped to a field of the query in the main report (see Figure 3). You can view this property by clicking the ellipsis in the lower right corner in Figure 2.

Linking a Report with an Existing Subreport

You can nest an existing report as a subreport inside another report. To do this, use the sample reports included in this tutorial named Department-Employees and Company-Departments. In this exercise, the Department-Employees report is the subreport.

This tutorial assumes that you have installed the sample applications with ColdFusion MX 7. You will be using the *cfdocexamples* data source, which points to the cfdocexamples database. If you have not installed the ColdFusion sample applications, run the ColdFusion installer again and select the option for installing the sample applications.

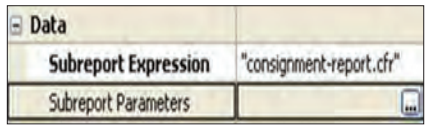


Figure 2 Subreport Parameters property

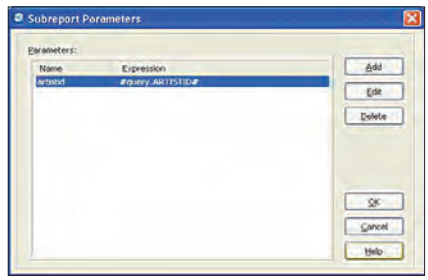


Figure 3 Subreport Parameters expression

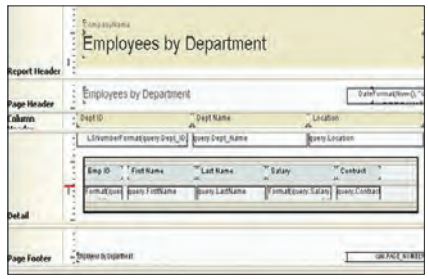


Figure 4 Subreport made from an existing report

Once you have installed the sample applications, follow these steps to link the two existing reports:

1. Copy Company-Departments.cfr report to a separate folder and rename it as **Company-Departments_v1.cfr**. Save a copy of the original report for the next exercise.
2. Double-click the Company-Departments_v1.cfr report. This opens the ColdFusion Report Builder. If you have not installed the ColdFusion Report Builder, do so now.
3. Expand the Detail band of the report to about 1.5 inches. You can do this by dragging the lower bounds of the Detail section or highlight the Detail section by clicking it and typing 1.5 as the Height property in the Property inspector.
4. Click the Subreport component in the Toolbox (see Figure 1 again).
5. Within the Detail band drag the cross hairs from upper left to lower right. You can resize this later as needed. The Subreport Wizard appears.
6. Select the From an Existing Report option and click the ellipses to navigate to the Department-Employees.cfr report. Click Next.
7. Because the sample subreport is already set up to take a parameter, the Subreport Wizard next asks you to bind the subreport parameters. For the DeptID, you can select from a pop-up list with the query's variables. Select query. Dept_ID from the pop-up list and click Next.
8. Click Finish.
9. Resize the Subreport component to see all the fields that the query will return. The width should be approximately 6.5 inches.

At this point, your report looks similar to Figure 4.

Save your report and preview it by pressing F12 or browse to the report. With a default ColdFusion installation, for example, it would be http://localhost:8500/subreports_example/ Company-Departments_v1.cfr.

Note: If you don't see a preview of your

report when you press F12, select Edit > Preferences. Select Server Connection from the Preference Groups and modify the Server Information setting so that it points correctly to your ColdFusion server.

Creating and Linking a New Subreport to Your Report

You can also create your subreport on the fly instead of linking to an existing subreport. In this section, follow these steps to create and link to a new subreport:

1. Copy Company-Departments.cfr report to a separate folder and rename it as Company-Departments_v2.cfr. Save a copy of the original report so you can use it for other tests. Double-click the Company-Departments_v2.cfr report. This opens the ColdFusion Report Builder. If you have not installed the ColdFusion Report Builder, do so now.
2. Expand the Detail band of the report to about 1.5 inches. You can do this by dragging the lower bounds of the Detail section or highlight the Detail section by clicking Detail and type 1.5 as the Height property in the Property inspector.
3. Click the Subreport component in the Toolbox (see Figure 1 again).
4. Within the Detail band drag the cross hairs from upper left to lower right. You can resize this later as needed. The Subreport Wizard appears.
5. Select the As a New Report option and click Next.
6. Click the Query Builder button to create a query to retrieve your recordset. The Query Builder window appears.
7. Select your ColdFusion server for RDS.

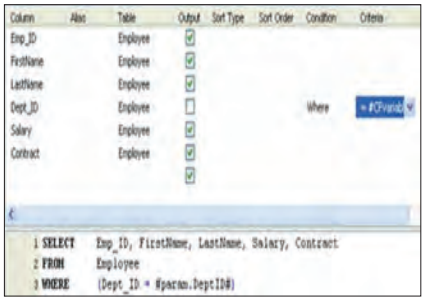


Figure 5 Building a query in the Query Builder Wizard

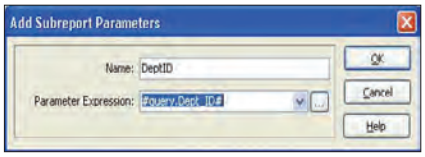


Figure 6 Subreport Parameters dialog box

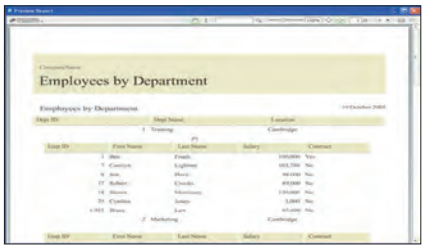


Figure 7 The final report

Note: If you have not set up an RDS server, you will need to do so now. To do so, refer to the ColdFusion LiveDocs Documentation. Expand the available data sources for your ColdFusion server, expand the cfdocexamples data source, and click Tables to expand the table choices. Double-click the Employee table. The Employee table appears in the top portion of the Query Builder.

8. Double-click the following fields: Emp_ID, FirstName, LastName, Dept_ID, Salary, and Contract.
9. For Dept_ID, deselect the Output check box so that this field will not actually be displayed as part of the report. It will, however, be part of the SQL Where clause.
10. In the Dept_ID row, under the Condition column, select Where from the pop-up menu. From the Criteria column, select = #CFvariable#. Note that this is the unquoted option, not the "#CFvariable#" option. Selecting these options creates or changes the SQL query in the panel below.
11. In the SQL query panel, modify the Where clause. Replace the value, CFvariable, in the Where clause with param.DeptID. Do not delete the pound signs around the variable. By changing the variable, the query references the input parameter for the report that you will create later. Your query appears like that shown in Figure 5.
12. Click the Test Query button, enter a test value of 1 and click Run. The query will return seven results. Close the results window.
13. Click Save. Click Next on the Define Group Reporting dialog box.
14. On the Report Layout dialog box, make sure that you have only selected the Grid option. Click Next.
15. On the Report Layout page, select the Only Detail Band option. Deselect the Create Totals for Numerical Fields option. Click Next.

Note: The report will run properly if you select the Default option at this point.

16. Click Next on the Report Theme dialog box.
17. Enter Employees by Dept for all three title fields on the Report Name dialog box. Click Next.
18. Click Next on the Subreport Binding dialog box.
19. For the Subreport Filename, save it in the same directory as the main report. Name this report Employees. Click Next.
20. Click Finish.
21. Save the Employees report.
22. In the Employees report, open the Fields and Parameters panel. If it isn't open already in the lower right corner of the Report Builder, select Window > Fields and Parameters.
23. In the Fields and Parameters panel, click the Input Parameters option once and click the plus (+) button to add an input parameter. The Add Input Parameter dialog box appears.
24. Enter DeptID for Name. Select Integer for the data type. This parameter corresponds to the param.DeptID referenced in the Where clause of your SQL statement you typed in earlier. Click OK.
25. Save the Employee.cfr report and preview by pressing F12. Enter 1 for the DeptID parameter; your report appears.
26. Toggle to the Company-Departments_v2.cfr report.
27. Click the Subreport component once to select it. In the Properties inspector, click the plus (+) sign to expand the Data property. Click Subreport Parameters once to select it.
28. Click the ellipses to add a parameter. The Subreports Parameters dialog box

appears. Click Add.

29. Enter DeptID for Name and select #query.Dept_ID# from the pop-up list (see Figure 6).
30. Click OK, and then click OK again to close the Subreport Parameters dialog box.
31. Save the Company-Departments_v2 report.
32. Preview your report (press F12). It appears similar to the report in Figure 7.

Congratulations! You have successfully created a subreport from the main report.

Where to Go from Here

This article described two different techniques for linking subreports to main reports. You can bind the parameters property in the subreport component, which will match the input parameters of the subreport itself to link reports. I have found that creating the subreport through the Subreport Wizard, the second example, has proven to be easier and reduced the guesswork in formatting the subreport with the main report.

I hope that after following the sample reports and creating subreports you can appreciate how easy it is to develop and link subreports to a main report. I believe this will help you get up and running with the ColdFusion Report Builder so you can create reports and link subreports with little trouble.

For more information about creating reports with ColdFusion, see:

- Read the article, "Building Reports with ColdFusion MX 7": www.macromedia.com/devnet/coldfusion/articles/reporting.html
- Check out the Extranet Exploration Guide (www.macromedia.com/examples/cfgettingstarted/experience/docs/extranet_sample_app_features_en.pdf). In this sample application from the ColdFusion Getting Started Experience (www.macromedia.com/examples/cfgettingstarted/experience/index_content.cfm), you can build a sample application for an art gallery and learn about building dynamic charts from your reports.
- Check out code snippets for ColdFusion reports in the Code Snippet Explorer (www.macromedia.com/examples/cfgettingstarted/experience/snippets.cfm), a part of the ColdFusion Getting Started Experience.

About the Author

Jim Bambrough taught high school mathematics and computer programming in the Phoenix area for three years. Jim currently works as a Web application developer for Amkor Technology in Chandler, Arizona. He has worked in IT since 2000 and has been developing Web-based reports at the corporate level since 2003.

Leveraging WebDAV-Accessible Directories with Mac OS X Server & ColdFusion MX 7.0.1

Opening ColdFusion to a whole new audience and platform

By Andrew Powell

Recently, with the introduction of ColdFusion MX 7.0.1, Macromedia began fully supporting a Mac OS X Server installation of ColdFusion. OS X Server used to be supported only in a development environment installed on top of JRun, Tomcat, or some other J2EE application server. Opening this platform has opened ColdFusion to a whole new audience and platform.

Mac OS X Server is regarded by many as an enterprise-class operating system overshadowed for years by Windows servers and Linux. The introduction of Mac OS X Server 10.4 (code named Tiger) brought Apple's enterprise entry to the forefront when it comes to serving files, print services, e-mail, web sites, and applications on the Apache Project's Tomcat J2EE Server. ColdFusion and Mac OS X Server offers functionality never seen in Windows installations and previously left only to Linux installations. One of these functions is access to directories based on the WebDAV standard.

A Windows 2003 Server installation of ColdFusion can't under any circumstances access a mapped drive if it's mapped via WebDAV. The share can be seen in Explorer, accessed from the command line, but can't be accessed from any of ColdFusion's tags including CFDIRECTORY and CFFILE. If access to a WebDAV directory is needed, it's a problem.

Setting Up the Environment

Mac OS X Server offers a perfectly simple solution. The first step is to mount the WebDAV directory in the operating system. This is done just like mounting any other network resource. From the finder, click on "Go," then click on "Connect To Server."

A dialog box like the one below will appear. Fill it in with the URL of the WebDAV-accessible directory that's being accessed, then click "Connect."

This should mount the directory listed as "Final" on the desktop. It doesn't mean that directory is inaccessible from the rest of the operating system or from ColdFusion. The true path of this network mount can be seen if the Mac OS X Terminal application is activated. (Applications ‡ Utilities ‡ Terminal.app). The directory "/Volumes" contains a mapping to every drive listed on the Mac OS X desktop. In this case, our mounted directory "Final" can be reached by typing at the console: /Volumes/Final. This mounting, however, won't persist through a reboot cycle. There's no good clean way to make a WebDAV share automount on OS X. The problem is easily remedied but you have to dirty your hands with some scripting work.

On the server running Mac OS X Server and ColdFusion MX 7 bring up a terminal window. (Make sure you have root privileges before doing these operations.) A directory for the mount point has to be created first. Type the command mkdir /mydirectory in the terminal. Next, an entry in /Library/StartupItems has to be created. This is what will do all the work automounting the share. The touch command will also be used to create the files needed to populate for the Automount to work properly.

```
mkdir /Library/StartupItems/AutMnt

cd /Library/StartupItems/AutMnt

touch AutMnt

touch StartupParameters.plist

Open the file AutMnt to edit with pico or any other editor of choice.

pico AutMnt
```

Type the following content into this file:

```
!/bin/sh
  . /etc/rc.common

  if [ "$1" == "start" ]
    then
      mount -t webdav http://mywebdavserver/mywebdavshare /mydirectory
  fi

Save and close this file. (Ctrl + X, then answer "Y" in pico.) Then open and
edit the file StartupParameters.plist

pico StartupParameters.plist
```

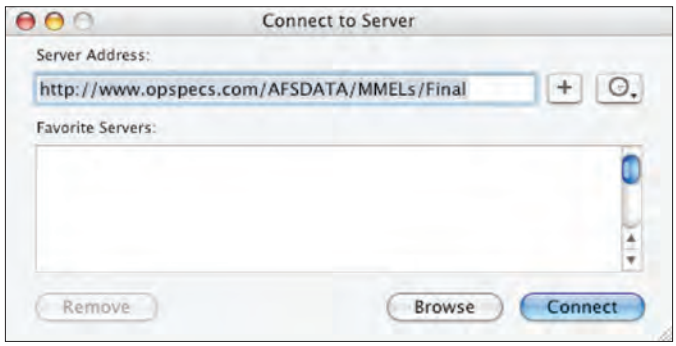


Figure 1

Type the following content into this file:

```
<?xml version="1.0" encoding="UTF-8"?>
   <!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
     "http://www.apple.com/DTDs/PropertyList-1.0.dtd">
   <plist version="1.0">
   <dict>
        <key>OrderPreference</key>
        <string>Early</string>
        <key>Description</key>
        <string>Try to mount the webdav share</string>
        <key>Provides</key>
        <array>
             <string>AutMnt</string>
        </array>
   </dict>
   </plist>
```

Save and close this file. (Ctrl + X, then answer "Y" in pico.) The proper permissions have to be set on these files before they can be used at startup. This can be done through the Finder. Browse to /Library/StartupItems/AutMnt and check the permissions by pressing Cmd + i. The Owner should be system with read/write permissions. The Group should be wheel with read-only permissions and Others should have read=only permissions.

Once done, these permissions can be checked back into the terminal with the command ls –l . The results should look like the listing below:

```
-rwxr--r--  1 root  wheel  145 17 Nov 13:43 AMount
-rw-r--r--  1 root  wheel  463 17 Nov 13:47 StartupParameters.plist
```

The OrderPreference key in the plist is important because ColdFusion's StartupParameters.plist file will be edited to indicate that it needs to start later than the script that's mounting the WebDAV share. This is to ensure that the share mounting occurs BEFORE ColdFusion starts.
Next, change the current directory to ColdFusion's directory:

```
cd ../ColdFusionMX7/
```

Open ColdFusion's StartupParameters.plist the same way the AutMnt program's plist was opened:

```
pico StartupParameters.plist
```

Change the OrderPreference key from "None" to "Late." This will ensure that ColdFusion starts AFTER the share is mounted. Save and close this file. (Ctrl + X, then answer "Y" in pico.) These changes go into effect when the system is rebooted. When

the system comes back up, the share should be mounted and ColdFusion started as usual.

Working in ColdFusion

In ColdFusion, the WebDAV shares are accessed just as any other local directory would be, as well as the files contained in the share.

CFDIRECTORY:

```
<cfdirectory action="list" directory="/mydirectory" name="myWebDAV"/>
```

CFFILE:

```
<cffile action="read" file="/mydirectory/myfile.ext"/>
```

If the Internet connection to or from the WebDAV server is slow, it will slow the performance of any actions on this directory. Keep this in mind that when the templates or CFCs that access these remote shares are called or executed.

DIRECTORYWATCHER GATEWAY

The main reason for ensuring that the mounting occurs before ColdFusion starts is to ensure that the directory being watched by an instance of the DirectoryWatcher Gateway exists. If the share isn't mounted before ColdFusion starts, then the gateway instance won't start.

Conclusion

Once the share is mounted, it appears as just another local directory to ColdFusion and can be accessed as such. There's no mystery to using the share once it's mounted. The trick is to make sure the share mounts before the ColdFusion MX 7 server starts. It's possible to extend this line of thinking to supported Linux and Unix installations of ColdFusion. If the share can be mounted, then it can be used in ColdFusion. ColdFusion can't differentiate between a local volume and a remote volume on these systems as it can on Windows. The ability to leverage remote volumes, such as WebDAV shares, makes ColdFusion that much more appealing because it lets ColdFusion reach across the entire enterprise network to do file and directory functions exploiting the power of the DirectoryWatcher Gateway.

About the Author

Andy Powell is director of information technology at D&D Aviation Services, Inc.

andy.powell@mac.com

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

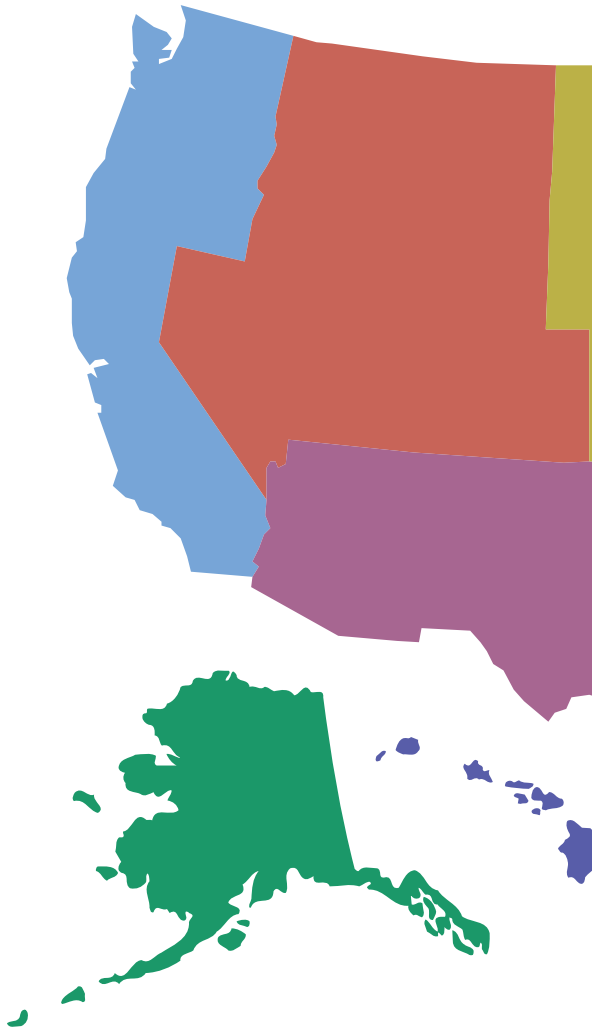
Minnesota
Brooklyn Park
Twin Cities CFUG
www.colderfusion.com

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

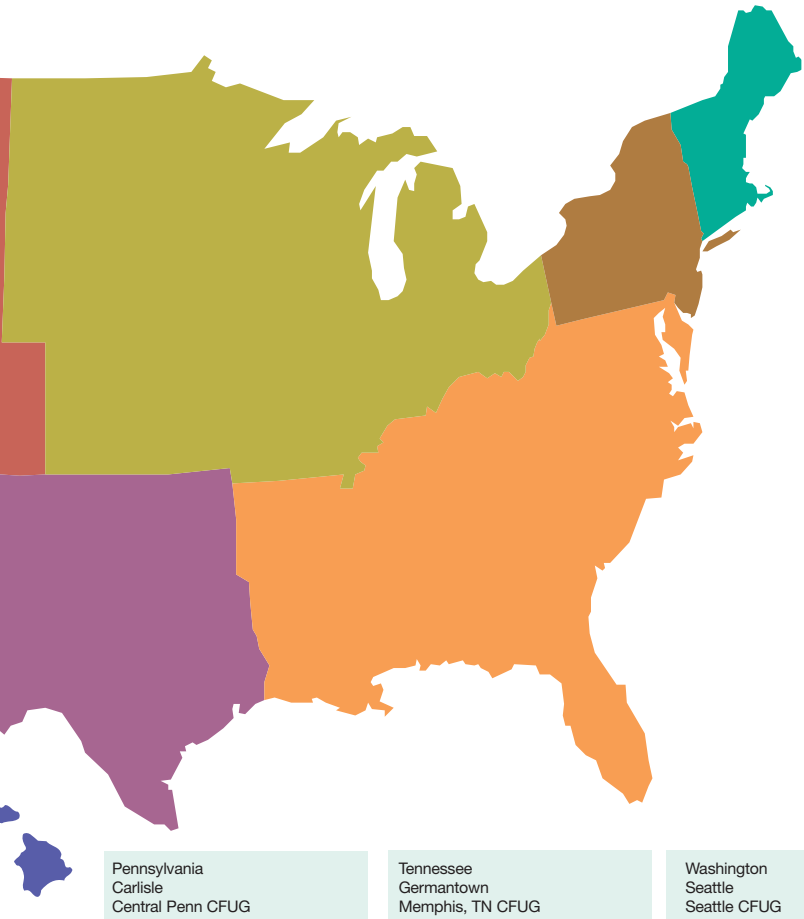
North Carolina
Raleigh
Raleigh, NC CFUG
www.ccfug.org

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

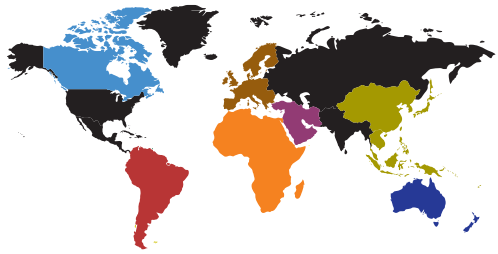
Utah
North Salt Lake
Salt Lake City, UT CFUG
www.slcfug.org

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

CFEclipse for ColdFusion Developers

Community development

By Rob Rohan

CFEclipse is a community project that builds on the open source Eclipse framework to deliver expert, code-centric developers a new option for ColdFusion development. This article explains what CFEclipse is, gives a bit of the project history, shows off some key features, and explains how you can get involved.

Requirements

To complete this tutorial you will need to install ColdFusion MX 7 or higher recommended (any version will work). For a trial download go to: www.macromedia.com/cfusion/tdrc/index.cfm?product=coldfusion&promoid=devcenter_tutorial_product_090903. To buy go to: www.macromedia.com/software/coldfusion/buy/?promoid=devcenter_tutorial_coldfusion_090903

Prerequisite Knowledge

Java knowledge helpful but not necessary.

What Is CFEclipse?

To understand what CFEclipse is, you must first learn about Eclipse. Eclipse (www.eclipse.org) is an open-source framework for creating Integrated Development Environments (IDEs). Eclipse can do much more than create IDEs, but this article focuses on the IDE aspect.

Eclipse by itself doesn't have any ability to edit or compile any type of file. It doesn't do much of anything except load plug-ins. The power of Eclipse comes from its wide array of plug-ins. Eclipse's most popular plug-in is the Java plug-in, which gives Eclipse the ability to edit and compile Java files. The Java plug-in is so popular that people mistakenly think Eclipse is a Java IDE. However, it is, in fact, an anything IDE.

Eclipse has become one of the fastes-growing IDEs in Europe, Asia, and North America. (See "Eclipse Use Grows by More Than 75% in Europe, Asia and North America": www.businesswire.com/cgi-bin/f_headline.cgi?bw.051004/241315336.) It is the basis for IBM's WebSphere Application Developer IDE, and plug-ins exist for many servlet containers and J2EE servers. Recently Macromedia announced Flex Builder 2, a completely new IDE for the Macromedia Flex Framework – and it is based on Eclipse. There are plug-ins for almost any language you can think of, Web-based or otherwise.

CFEclipse is a plug-in that gives developers the ability to author and edit ColdFusion files from within the Eclipse environment. CFEclipse provides Eclipse with an outline view, a CFC (ColdFusion component) view, code completion (code hinting), syntax highlighting text editors for CFML and CFC files, and many more ColdFusion-specific features.

One thing CFEclipse does not have is WYSIWYG ability. Its complete focus is on writing ColdFusion code (so-called hand coding) and building tools that help you write code. If you do a lot of visual or page-layout development – for instance, if you rely heavily on drag-and-drop tools, wizards, or the Design view in Macromedia Dreamweaver – CFEclipse may not be the tool for you.

A Brief History of CFEclipse

The CFEclipse plug-in was first released in January 2004, which makes the project almost two years old. The initial release was very basic. In fact, the project name was the Basic ColdFusion Eclipse Plug-in.

Since its first release, CFEclipse has undergone many improvements and feature additions by developers such as Oliver Tupman, Stephen Spike Milligan, Mark Drew, and many others.

At least 20 people have contributed code to the CFEclipse project, and the number of people who have helped market, design, support, and/or steer the product are far too numerous to list here. CFEclipse is truly a community effort.

In June 2005, Macromedia announced that

This article originally appeared on the Adobe Developer Center. Reprinted with permission

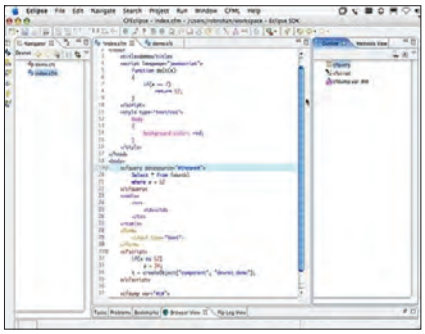


Figure 1 Code syntax highlighting in CFEclipse

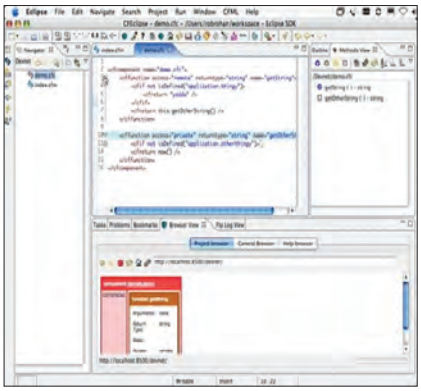


Figure 2 Methods view in CFEclipse

they would become an active supporter and contributor to the CFEclipse project as well. They are currently involved in contributing code and offering understanding of the ColdFusion developer base to help take CFEclipse to the next level in its support for ColdFusion MX 7 and beyond.

Key Features in CFEclipse

Now you know about the history of CFEclipse and where it comes from, but how does it benefit you, the developer? Well, the first major benefit in using CFEclipse is Eclipse itself.

Many companies use more than one language for their Web application development solution: Java, PHP, C#, J2EE, ActionScript, CFML, JavaScript, CSS, MXML – the list goes on. It's the nature of the Web to use more than one language at a time, and one of the benefits of Eclipse is that you can use any language that has a plug-in.

Multi-language environments are becoming increasingly common for ColdFusion developers – with ColdFusion appearing more often in the enterprise due to the ColdFusion J2EE underpinnings and its rapid application development ability gaining more and more attention. Using Eclipse and its wide assortment of plug-ins gives you the power to code in almost any language you need.

CFEclipse has one of the most advanced syntax highlighting capabilities of any ColdFusion editor. Not only does it color-code normal HTML and ColdFusion blocks, it also helps you write CFScript, JavaScript, CSS, and SQL query blocks (see Figure 1).

The editor has bracket and brace matching, auto bracket, and quote completion in JavaScript and CFScript blocks. Also, CFEclipse was one of the first ColdFusion-specific editors with the ability for code folding – a new feature called code collapse/expand in Macromedia Dreamweaver 8.

CFEclipse provides a few other ways to view your ColdFusion code. In addition to the code editor in CFEclipse, you can view your ColdFusion code through the methods view and the document outline view.

The document outline view lays out your document in a tree view. The innovation with document outline view, however, is that you can filter the view based on ColdFusion tags. For example, you could filter a 16,000-line file to show only the queries within the document. It is a helpful feature when working with legacy code – what I like to call code spelunking.

The methods view, in contrast, is perfect for a quick outline of a CFC (ColdFusion component). This view is great when writing code in the Mach-ii or Model Glue frameworks. The method's view lists public, private, and remote methods with both input and output information for each method. You can also filter the methods view based on method access (see Figure 2).

In addition to the advanced syntax highlighting, code completion, code folding, and filterable outline views, CFEclipse supports snippets (small pieces of reusable code). CFEclipse uses its own snippet format but can also natively use Dreamweaver snippets. Sharing snippets between Dreamweaver and CFEclipse improves continuity and code sharing between those on your team who focus more on design and layout and code-centric programmers.

The last few benefits I mentioned are not based on the product or feature set but more on the way the community developed the plug-in. The plug-in is open source. This gives you the ability to download and save the source code or – if you are Java-savvy – fix bugs immediately without waiting for someone else to fix the problem.

How to Get the CFEclipse Plug-in

If you would like to simply use CFEclipse, you can download the plug-in and get information at www.cfeclipse.org. If you are a developer and would like to help with coding CFEclipse, you can find the source code and collaborate with the developers at cfeclipse.tigris.org. At present, this site is also where all bug reports and developer mailing lists are located.

Because of the great community welcome and effort for CFEclipse, there are many sources of information on CFEclipse. Just type CFEclipse into Google and you'll find many sites that have tips and tricks. Dopefly at www.dopefly.com/pages/cfeclipse.cfm is a great example.

What You Can Do to Help

You don't have to be a Java developer to help with the CFEclipse project. While we like coding help, there are many other things that non-Java coders can do. For example, we are in desperate need of documentation in many languages. If you use CFEclipse or catch on quickly, answering questions from the newer participants on the mailing lists is also a great way to help.

If you know Java and would like to help, submitting patches is the best way to get committer status. If you have an idea and want to add a feature, let us know so that committers won't step on each others' toes.

If you'd like to help, Java coder or not, join the mailing list on cfeclipse.tigris.org and say you'd like to help. It might take us a bit to get back to you but we would love to hear from you.

About the Author

Rob Rohan has been a developer and architect for over seven years. Rob is a YellowBadger consultant from the Bay Area and founder of CFEclipse. He teaches Macromedia products through Scoonertech.

me@robrohan.com

Heresy! Rethinking CFCs PART 1

Empowering our development

"A good model guides your thinking; a bad one warps it." —Brian Marick

By Hall Helms

In this article, I argue that by adopting the static typing model found in languages such as Java, we have been seduced into accepting a bad model – or at least a model that is inappropriate for the dynamic typing mechanism inherent in ColdFusion. I'll explain the problems that such a mindset has caused and how the promised benefits of static typing fail to materialize in ColdFusion.

I must ask your indulgence, however. This exploration is going to be fairly lengthy and parts of it will get quite technical. And long, so long that I can't fit it into a single article, so next month I'll continue with what ramifications adopting a dynamic typing model might have on our ColdFusion development – and how such a shift in thinking can empower in surprising ways our development using ColdFusion components.

I did not come to this conclusion readily or easily. I am a great fan of static typing in languages such as C# and Java and I have spent many, many hours trying to fit ColdFusion into that mold. My own breakthrough came from studying Ruby, and from becoming reacquainted with a community that embraces dynamic typing. But, I'm getting ahead of myself. First, let's establish some terms.

What Is "Static Typing"?

To understand static typing, we first need to understand just a little about *type theory*, that branch of mathematics and logic that classifies entities into collections called types. Type theory sees individuals as having identity by virtue of their inclusion into one (or more) *types*. Furthermore, by knowing about the type, we can infer things about the individual.

Sun's Java tutorial puts it like this: "A variable's data type determines the *values* that can be associated with it and the *operations* that can be performed on it." An object receives its identity because it belongs to one or more types. The only operations (i.e., "methods") that can be guaranteed in an object are those that first belong to its type. All others are illegal, hence, "uncompilable."

In case this is too abstract, let's say we're strolling down the main street of Objectville one particularly warm day. On the other side of the street, you see an object. "Hey, I wonder if that object knows where the nearest bar is," you say. "I think I'll ask."

"Are you crazy?" I demand. "You don't know that object from Adam. How would you know if it could respond to the getNearestBar message? You need to know its type first. From that, we can determine if the type (and therefore that particular object) will be able to respond to our method call."

"Type?" you ask. "As in data type? What does that have to do with whether it knows where we can get a cold beer? Say, you're not a Java compiler, are you?"

Such is what life is constantly like for Java compilers (or so I've been assured by a highly placed Java compiler who wishes to remain anonymous). In their role of turning source code (the stuff we write) into machine code (the stuff computers read), they must be ever vigilant lest some errant code make its way to the machine. To do this successfully, they rely on static typing – that is, information that can be inferred *prior to runtime* about what types an object belongs to (and, thus, what methods may be called on that object).

Type Safety

The entire edifice of static typing is used to promote *type safety,* that blessed state where the compiler gives its blessing on our code. Author David Cardelli offers further precision: "The primary goal of a type system is to ensure language safety by ruling out *all* untrapped errors in all program runs. [The] declared goal of a type system is usually to ensure good behavior of all programs, by distinguishing between well-typed and ill-typed programs."

When I teach my "Java for ColdFusion Programmers" class, almost all students (certainly those who haven't had experience with a compiled language) are surprised and delighted when their compiled Java code runs without runtime errors. After

the initially unnerving sense that the compiler is peering over your shoulder waiting for you to make but the smallest of errors and draw your immediate attention to them, programmers learn to rely on the validation of the compiler with its insistence on proper typing along with its ultimate goal of type safety.

Polymorphism

Static typing also underlies Java's implementation of *polymorphism*. While the word transliterates from the Greek as "many forms," polymorphism typically is used to mean the ability to call the same method on different objects with (possibly) different implementations of that method.

Let's use a very simple example of polymorphism starting with a cat named Fluffy, a dog named Spot, and a lion named Leo. (For this – and some subsequent examples – I'm going to show you some Java code. I'll try to explain what's happening as we go along, although you might find it not too incomprehensible, even if you haven't done any Java yet.)

A cat, a dog, and a lion might all have a **speak** method. Call it on Fluffy and you'll hear "Meow." Call that same method on Spot and you'll hear "Bark!" Call it on the lion and the last sound you'll hear is "Roar! Chomp..." As advertised, polymorphism allows us to send the same message ("speak") to different objects and to allow them to respond in a way most appropriate for each object.

How does a statically typed language such as Java know that it's okay for you to call these methods on these objects? It insists that before you create the objects, you first *type* them. The pattern for object creation is this:

```
type objectName = new ClassName([arg1,...
argN]);
```

For this to work correctly, you must have a class (corresponding to *ClassName*) that defines a speak method. A very simple implementation of a Cat class might look like this:

```
public class Cat{
    public String speak(){
       return "Meow";
    }
}
```

Once the various classes are defined, object creation is straightforward:

```
Cat fluffy = new Cat();
Dog spot = new Dog();
Lion leo = new Lion();
```

With each object typed (notice that the **type** and the **ClassName** correspond in this instance) and created, we can call methods on the objects:

```
fluffy.speak();
spot.speak();
```

(I'll leave it for a braver man than I to call leo's **speak** method...)

At some point, we might want to place each of these animals in an array. One pattern for creating an array is this:

```
type [ ] arrayName = { fluffy, spot, leo };
```

Yes, even arrays must be typed, but what type should this array be? Don't think that the compiler is going to cut you any slack; it wants a type – and it wants it now.

If you're like many of my students, you may try this out:

```
String [ ] animals = { fluffy, spot, leo };
```

But before you can type that final semi-colon, the compiler will bleed red over your code (and some of the newer compilers will even insult your lineage). You can only imagine the internal dialog of the compiler: "String? Did you say, 'String'? Well, the last I looked, fluffy was a Cat, not a String. But maybe things have changed. Let me check. Nope. Still a Cat. Any other ideas, Einstein?"

Of supertypes, Classes, and Interfaces

Thus, we come – battered by an abusive compiler – to the idea of a *supertype* – a type that can encompass fluffy the Cat, spot the Dog, and leo the Lion. An **Animal**, perhaps. But before we can use **Animal** as a type, we must first create a type definition and we must find a way to specify that this new type is to be a supertype of our Cat, Dog, and Lion types.

For the type definition, we have two choices: we can create a new class (as we did for **Cat**) or we can create what Java calls an interface (no relationship to a graphical user *interface*, however). We'll need to look at interfaces shortly but, for now, let's stick with what we know: classes.

The **Animal** class looks like this:

```
public class Animal{
    public String speak(){
        return "Err...some generic animally
sound, I guess...";
    }
}
```

That's the class; now we need to register **Animal** as a supertype to **Cat, Dog,** and **Lion**. This relationship between classes is sometimes known as a parent-child relationship. **Animal** is the parent; **Cat, Dog,** and **Lion** are the children. True to life, the parents have no idea what the kids are up to: we don't register **Animal** as being a supertype; rather we register **Cat, Dog,** and **Lion** as being *subtypes*. If you've worked with CFCs, this should look very familiar:

```
public class Cat extends Animal{
    public String speak(){
        return "Meow";
    }
}
```

Now, we can type our **animals** array:

```
Animal[ ] animals = { fluffy, spot, leo };
```

A lot of work just to create an array, but for that work, you'll receive the compiler's assurance that your code is typesafe. Statically typed languages are insistent that whenever a variable is used, it be typed. Thus, methods must provide return types (even going so far as to specify **void** if they return nothing); a method must declare the type of any arguments it accepts, and variables must declare their type.

Type Promotion Means Type Restrictions

As I said, in a statically typed language, even polymorphism is intimately tied with types. When we placed **fluffy**, **spot**, and **leo** into that array of type, **Animal**, each object was given a sort of "battlefield promotion" to **Animal**, right on the spot. The compiler, first assuring itself that, indeed, a subtype-supertype relationship existed between, say, Cat and **Animal**, promoted **fluffy** to type, **Animal**.

To show you what a stickler our compiler is, let me change the **Cat** class, adding another simple method, one that returns every cat's favorite food:

```
public class Cat extends Animal{
    public String speak(){
        return "Meow";
    }

    public String getFavoriteFood(){
        return "lobster";
    }
}
```

Object assignments in Java work just as CFC assignments do in ColdFusion—that is, the variable refers to (or points to, if you prefer) an object. In this ColdFusion code…

```
<cfset myFavoriteBook = CreateObject('componen
t', 'Book') />
<cfset yourFavoriteBook = myFavoriteBook />
```

…both **myFavoriteBook** and **yourFavoriteBook** point to the same object in memory. Change something about one variable's reference and you change it for the other's. Since both **myFavoriteBook** and **yourFavoriteBook** refer to the same object, if I give **yourFavoriteBook** a title…

```
<cfset yourFavoriteBook.setTitle('Difficult
Compilers...and the Programmers That Love
Them') />
...myFavoriteBook now also has a title:
<cfoutput>
    #myFavoriteBook.getTitle()# --> Difficult
Compilers...and the Programmers That Love Them
</cfoutput>
```

This means that **fluffy** and **animals[0]** (array indices begin with 0 in Java) refer to the same lovable cat. But notice that while the variable, **fluffy**, was typed as a **Cat**, the object referred to by fluffy was type-promoted to Animal when it was placed in the array. This means that to the compiler, **fluffy** remains an object of type, **Cat**, while **animals[0]** is now an object of type, **Animal**.

I know: it hurts everyone's head at first. And it gets worse. Look at this code:

```
fluffy.getFavoriteFood();
animals[0].getFavoriteFood();
```

Both variables refer to the same object in memory. But while the first line of code will run fine, the second will not even compile. Don't blame me! It's the compiler's fault. In fact, anticipating that you might rise up with pitchforks and torches, I asked the compiler about this seemingly bizarre behavior. What follows is a basically unedited, frank exchange:
Me: You realize that I have to explain this to people, right?
Compiler: Oh, please. There's nothing difficult about this.
Me: Just look at this code with this cute little cat, **fluffy**.
Compiler: Yeah?
Me: So, how come it works when I say **fluffy.getFavoriteFood** and breaks when I

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

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

24/7

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

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

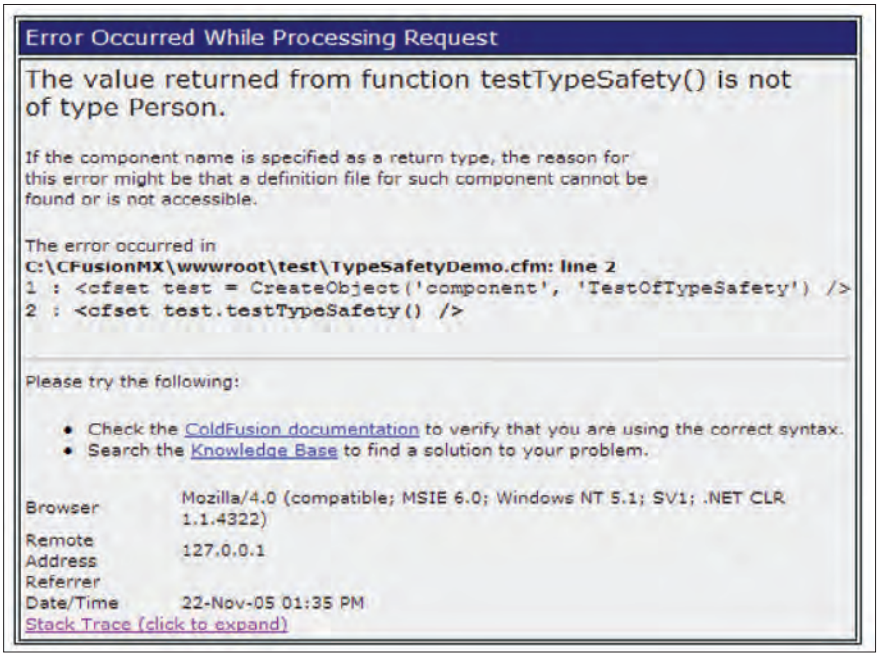


Figure 1

say **animals[0].getFavoriteFood**? They're referring to the same object!

Compiler: You just don't have a clue, do you? Look, I have to make laws, laws that everybody has to live by. So, the law is that when you get type-promoted, then, within that context, you can only be treated as that promoted type. So your cat, in the context of **fluffy**, is an object of type, **Cat**. But what's in slot 0 of the animals array is an object of type, **Animal**.

Me: But couldn't you relax the rules a little?

Compiler: Relax the rules? Oh, sure, who needs rules? I don't know why I didn't think of that. Sure, call **animals[0].getFavoriteFood**. In fact, call that same **getFavoriteFood method** on all the animals in the array. Oh, but that won't work, will it? Because what's in slot 1 and 2 aren't **Cat** objects and don't have a **getFavoriteFood** method. So, now, we'll have a runtime exception. That'll be nice. Maybe you can explain to programmers why their code is breaking at runtime when Ive promised that it's typesafe…

Me: Okay, so you're treating **animals[0]** as an object of type, **Animal**. I can see that. But does that mean that if I call **objects[0].speak**, I'm going to hear the speak response written in the **Animal** class, the "some generic animally sound" thing?

Compiler: No! You're going to hear, "Meow." That's polymorphism. I haven't forgotten that ***animals[0]*** is, in fact, a **Cat**. All I'm doing is making sure that you only call methods that are defined in **Animal**. So **speak** is cool; **getFavoriteFood** is not.

Me: Because if you let me call the **getFavoriteFood** method on one **Animal…**

Compiler: I have to let you – and everybody else – call that same method on all Animal objects.

Me: You can't make that determination at runtime?

Compiler: I don't have time to explain compiler theory to you – and frankly, I doubt you could understand it – but suffice it to say that I work at ***compile*** time, not at runtime. So I have to set up rules so that you can't pull a switcheroo on me at runtime. It's all about type safety and, to ensure that, I have to enforce certain typing rules and restrictions.

Benefits of Static Typing?

So, there you have it, straight from the compiler's mouth, so to speak: to have type safety, the Java compiler has to be quite rigid about typing. But, if you go along with the rules, you'll get some pretty significant benefits:

1. A machine-enforceable "contract" between clients (users of services) and servers (providers of those services)
2. Early detection of certain classes of errors that make code more robust and maintainable
3. Performance optimization

Maybe the static typing model isn't such a bad one – it certainly works for Java and C#. The question is, can we get compile-time type safety with ColdFusion? There are indications that we may. ColdFusion components have an optional attribute for methods: **returntype**. If you specify a return type, ColdFusion will check that what you're returning from a method matches the return type specified. Similarly, method arguments have a **type** attribute that ensures that arguments passed to a method are of the same type as was specified.

We know that ColdFusion is an interpreted language, not a compiled one. But Java, the base on which ColdFusion is written, is both a compiled and interpreted language. Perhaps we can piggyback onto Java's compilation phase? Let's see if this will work. I'll write a method that deliberately breaks the contract for a method:

```
<cfcomponent displayname="TestOfTypeSafety">

   <cffunction name="testTypeSafety"
access="public" returntype="Person"
output="false">
      <cfreturn "I'm a string, not a Person" />
   </cffunction>

</cfcomponent>
```

Now, I'll instantiate the component:

```
<cfset test = CreateObject('component', 'Tes-
tOfTypeSafety') />
```

Hmmm…no exception is thrown, even though t**estTypeSafety** violates the **returntype** contract.

From this simple experiment, we can see that, despite Java's underlying static typing model, ColdFusion solidly remains a dynamically typed language. It's not that types don't exist in ColdFusion, but rather that a variable's type is determined at runtime.

But call the offending method…

```
<cfset test.testTypeSafety() />
```

…and the runtime type checking is in-

The World's Leading Java Resource Is Just a >Click< Away!

JDJ is the world's premier independent, vendor-neutral print resource for the ever-expanding international community of Internet technology professionals who use Java.

SYS-CON MEDIA

ONLY $69.99

ONE YEAR
12 ISSUES

Subscription Price Includes FREE JDJ Digital Edition!

www.JDJ.SYS-CON.com

or **1-888-303-5282**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

voked (see Figure 1).

Missing Null

Nor do the problems of treating ColdFusion as a strongly typed language end there. Look at this component:

```
<cfcomponent displayname="LibraryBook">
   <cfset variables.checkedOutTo = "" />

   <cffunction name="checkOut" access="public"
returntype="void" output="false">
     <cfargument name="checkedOutTo"
type="Patron" required="true" />
     <cfset variables.checkedOutTo = argu-
ments.checkedOutTo />
   </cffunction>

   <cffunction name="getCheckedOutTo"
access="public" returntype="Patron"
output="false">
   <cfreturn variables.checkedOutTo />
  </cffunction>
</cfcomponent>
```

The component encapsulates the idea of a library book. When a library book has been checked out by a patron, the **Patron** object passed to the **checkOut** method will be stored in the component's **checkedOutTo** instance variable. If I call the **getCheckedOutTo method**, I'll get back the appropriate **Patron** object.

But what happens if the book hasn't been checked out? What will I get back when I call the **getCheckedOutTo** method? Let's try:

```
<cfset libraryBook = CreateObject('component',
'LibraryBook') />
<cfset patron = libraryBook.getCheckedOutTo()
/>
```

This produces an error as shown in Figure 2.

Since no **Patron** object was assigned to the component's **checkedOutTo** instance variable, the **getCheckedOutTo** method returns the empty string the **checkedOutTo** variable was initialized as:

```
<cfset variables.checkedOutTo = "" />
```

Now, we have a serious problem: we can't guarantee the type safety of our code, and type safety was the great benefit promised by static typing.

Java and C# (and many languages) "solve" the problem by providing a **null** object. A null fulfills the contract for any class. In the Java version of **LibraryBook**, I can provide the **checkedOutTo** instance variable with an initial value of **null**, like this:

```
private Patron checkedOutTo = null;
```

Even if I simply declare the variable and fail to initialize it…

```
private Patron checkedOutTo;
```

…which is perfectly legal, Java will initialize it for me as **null**. I can call my Java class's **getCheckedOutTo method** and I'll get no runtime errors.

But even this "solution" has problems. Let's say that the **Patron** class has a method called **getName**, which returns the patron's name. Now, I'll use a little Java code to instantiate a new LibraryBook (which, by default, has no Patron the book has been checked out to).

```
LibraryBook libraryBook = new LibraryBook();
```

I can even call **libraryBook's getCheckedOutTo** method without error:

```
Patron checkedOutTo = libraryBook.getChecked-
OutTo();
```

Java's compiler assures me that my code is type safe. Really? Then let me find out the patron's name:

```
String patronName = checkedOutTo.getName();
```

The code compiles just fine; it's nice to know I won't have any runtime errors. So, let me just run that code…

Oh no, a runtime error – the dread **NullPointerException**. As we can see, there are limits on what type safety can assure us of: some errors simply can't be caught until runtime. But I've digressed. Let's get back to ColdFusion, for there's a more serious problem with treating ColdFusion as a statically typed language and it has to do with supertypes.

Missing Interfaces

ColdFusion implements a subtype-supertype relationship through inheritance. If a component **extends** another component, the extending component is automatically a subtype of the component it extends.

In fact, we can translate the Java code for **Animal, Cat, Dog**, and **Lion** and get exactly the same polymorphic behavior we saw in Java. But suppose that we code to show that Lion is a dangerous animal. Phrased another way, we want to say that **Lion** is a subtype of DangerousAnimal.

But what is DangerousAnimal? As I

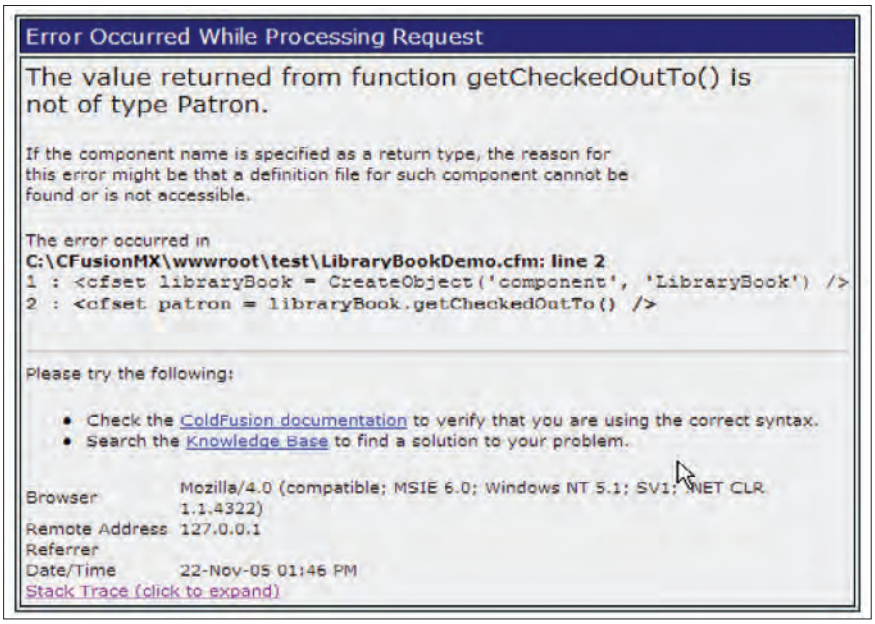


Figure 2

A LIMITED TIME SAVINGS OFFER FROM *SYS-CON MEDIA*

SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

AND SAVE UP TO $340 AND RECEIVE UP TO 3 FREE CDs!*

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY WITH YOUR PAID SUBSCRIPTIONS

3-Pack

Pick any 3 of our magazines and save up to $210.00

Pay only $99 for a 1 year subscription plus a FREE CD

- 2 Year – $179.00
- Canada/Mexico – $189.00
- International – $199.00

6-Pack

Pick any 6 of our magazines and save up to $340.00

Pay only $199 for a 1 year subscription plus 2 FREE CDs

- 2 Year – $379.00
- Canada/Mexico – $399.00
- International – $449.00

9-Pack

Pick 9 of our magazines and save up to $270.00

Pay only $399 for a 1 year subscription plus 3 FREE CDs

- 2 Year – $699.00
- Canada/Mexico – $749.00
- International – $849.00

CALL TODAY! 888-303-5282

Pick a 3-Pack, a 6-Pack or a 9-Pack					
☐ 3-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 6-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.
☐ 9-Pack	☐ 1YR	☐ 2YR	☐ U.S.	☐ Can/Mex	☐ Intl.

TO ORDER •Choose the Multi-Pack you want to order by checking next to it below. •Check the number of years you want to order. •Indicate your location by checking either U.S., Canada/Mexico or International. •Then choose which magazines you want to include with your Multi-Pack order.

☐ **LinuxWorld Magazine**

U.S. - Two Years (24) Cover: $143	You Pay: $79.99 /	Save: $63 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $32
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $79.99 /	Save: $4
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **JDJ**

U.S. - Two Years (24) Cover: $144	You Pay: $99.99 /	Save: $45 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $69.99 /	Save: $12
Can/Mex - Two Years (24) $168	You Pay: $119.99 /	Save: $48 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $89.99 /	Save: $40
Int'l - Two Years (24) $216	You Pay: $176 /	Save: $40 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Web Services Journal**

U.S.- Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **.NET Developer's Journal**

U.S. - Two Years (24) Cover: $168	You Pay: $99.99 /	Save: $68 + FREE $198 CD
U.S. - One Year (12) Cover: $84	You Pay: $69.99 /	Save: $14
Can/Mex - Two Years (24) $192	You Pay: $129 /	Save: $63 + FREE $198 CD
Can/Mex - One Year (12) $96	You Pay: $89.99 /	Save: $6
Int'l - Two Years (24) $216	You Pay: $170 /	Save: $46 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $99.99 /	Save: $8

☐ **Information Storage + Security Journal**

U.S. - Two Years (24) Cover: $143	You Pay: $49.99 /	Save: $93 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $39
Can/Mex - Two Years (24) $168	You Pay: $79.99 /	Save: $88 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $49.99 /	Save: $34
Int'l - Two Years (24) $216	You Pay: $89.99 /	Save: $126 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $59.99 /	Save: $48

☐ **Wireless Business & Technology**

U.S. - Two Years (12) Cover: $120	You Pay: $49.00 /	Save: $71 + FREE $198 CD
U.S. - One Year (6) Cover: $60	You Pay: $29.99 /	Save: $30
Can/Mex - Two Years (12) $120	You Pay: $69.99 /	Save: $51 + FREE $198 CD
Can/Mex - One Year (6) $60	You Pay: $49.99 /	Save: $10
Int'l - Two Years (12) $120	You Pay: $99.99 /	Save: $20 + FREE $198 CD
Int'l - One Year (6) $72	You Pay: $69.99 /	Save: $2

☐ **MX Developer's Journal**

U.S. - Two Years (24) Cover: $143	You Pay: $49.99 /	Save: $93 + FREE $198 CD
U.S. - One Year (12) Cover: $72	You Pay: $39.99 /	Save: $32
Can/Mex - Two Years (24) $168	You Pay: $79.99 /	Save: $88 + FREE $198 CD
Can/Mex - One Year (12) $84	You Pay: $49.99 /	Save: $34
Int'l - Two Years (24) $216	You Pay: $89.99 /	Save: $126 + FREE $198 CD
Int'l - One Year (12) $108	You Pay: $59.99 /	Save: $48

☐ **ColdFusion Developer's Journal**

U.S. - Two Years (24) Cover: $216	You Pay: $129 /	Save: $87 + FREE $198 CD
U.S. - One Year (12) Cover: $108	You Pay: $89.99 /	Save: $18
Can/Mex - Two Years (24) $240	You Pay: $159.99 /	Save: $80 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $99.99 /	Save: $20
Int'l - Two Years (24) $264	You Pay: $189 /	Save: $75 + FREE $198 CD
Int'l - One Year (12) $132	You Pay: $129.99 /	Save: $2

☐ **WebSphere Journal**

U.S. - Two Years (24) Cover: $216	You Pay: $129.00 /	Save: $87 + FREE $198 CD
U.S. - One Year (12) Cover: $108	You Pay: $89.99 /	Save: $18
Can/Mex - Two Years (24) $240	You Pay: $159.99 /	Save: $80 + FREE $198 CD
Can/Mex - One Year (12) $120	You Pay: $99.99 /	Save: $20
Int'l - Two Years (24) $264	You Pay: $189.00 /	Save: $75
Int'l - One Year (12) $132	You Pay: $129.99 /	Save: $2

☐ **PowerBuilder Developer's Journal**

U.S. - Two Years (24) Cover: $360	You Pay: $169.99 /	Save: $190 + FREE $198 CD
U.S. - One Year (12) Cover: $180	You Pay: $149 /	Save: $31
Can/Mex - Two Years (24) $360	You Pay: $179.99 /	Save: $180 + FREE $198 CD
Can/Mex - One Year (12) $180	You Pay: $169 /	Save: $11
Int'l - Two Years (24) $360	You Pay: $189.99 /	Save: $170 + FREE $198 CD
Int'l - One Year (12) $180	You Pay: $179 /	Save: $1

☐ **WLDJ**

U.S. - Four Years (24) Cover: $240	You Pay: $99.99 /	Save: $140 + FREE $198 CD
U.S. - Two Year (12) Cover: $120	You Pay: $49.99 /	Save: $70
Can/Mex - Four Years (24) $240	You Pay: $99.99 /	Save: $140 + FREE $198 CD
Can/Mex - Two Year (12) $120	You Pay: $69.99 /	Save: $50
Int'l - Four Years (24) $240	You Pay: $120 /	Save: $120 + FREE $198 CD
Int'l - Two Year (12) $120	You Pay: $79.99 /	Save: $40

*WHILE SUPPILES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Online Today **www.sys-con.com/2001/sub.cfm**

SYS-CON MEDIA

said, in the ColdFusion world, subtype-supertype relationships are implemented only through inheritance and since Lion already extends Animal, we have a problem. In fact, this is a problem for all statically typed languages. Some languages (C++ and Eiffel are examples) allow a class to extend more than one other class. If ColdFusion has this facility, my code for Lion might look like this:

```
<cfcomponent displayname="Lion"
extends="Animal, DangerousAnimal">
```

But ColdFusion forbids such *multiple inheritance,* as do Java and C#. But Java (and C#, following Java's lead) introduces the idea of interfaces, mentioned earlier. An interface is a construct similar to a class. An interface can have methods, but these methods have no bodies. Let's look at what an interface in Java for **DangerousAnimal** might look like:

```
public interface DangerousAnimal {
    public void attack();
}
```

While **DangerousAnimal** specifies an **attack** method, it provides no implementation for the method. Interfaces form contracts or specifications that other classes can "sign up" for. When a class specifies that it implements an interface, the Java compiler will ensure that it implements all methods declared in the interface specification. The advantage to interfaces is that they establish a supertype-subtype relationship with their implementing classes. And while a class can extend only one other class, that same class may implement many interfaces.

Let's look at the Java Lion class again – this time as it implements the DangerousAnimal interface:

```
public class Lion extends Animal implements
DangerousAnimal {

    public String speak(){
          return "Roar! Chomp...";
    }

    public void attack() {
          System.out.print("Slowly I creep up
on my unsuspecting prey...");
    }

}
```

Now, if you've followed the discussion this far (and congratulations, btw), I have a question for you: Of what type is leo, given the changes made to **Lion**?

1. **Lion**
2. **Lion** and Animal
3. **Lion, Animal,** and **DangerousAnimal**

If you chose number 3, you get "full marks," as our British friends say. Skeptical? Let's consult our old friend, the compiler.

Compiler: What now?
Me: So, that's right: leo is a **Lion** and an **Animal** and a **DangerousAnimal**?
Compiler: Yes, of course. Any implementing class of an interface is a subtype of that interface and I can, therefore, type-promote it when needed.
Me: Alright then, if I have a **Zoo** class, looking like this…

```
Exception in thread "main" java.lang.NullPointerException
        at test.LibraryBookDemo.main(LibraryBookDemo.java:8)
```

```
public class Zoo {

      public void cageAnimal(Animal animal){
          System.out.println("I'm going to put
you in this fenced in area");
    }

    public void cageDangerousAnimal(DangerousAn
imal dangerousAnimal){
      System.out.println("You, I'm going to
put in this high security pen");
    }

    public void feed(Animal animal){
      System.out.println("Here's something to
eat");
    }
}
```

…then this code will work?

```
Cat fluffy = new Cat();
Lion leo = new Lion();
Zoo zoo = new Zoo();
zoo.cageAnimal(fluffy);
zoo.cageDangerousAnimal(leo);
zoo.feed(fluffy);
zoo.feed(leo);
```

Compiler: Yes.

So, again, there you have it.

Conclusion to Part I

Interfaces are powerful mechanisms, not only for providing multiple supertypes, but for designing to interfaces. There's only one problem: ColdFusion doesn't have interfaces. And without interfaces (or some mechanism for multiple supertyping), static typing runs into an insurmountable problem. Single inheritance (as implemented by ColdFusion, Java, and C#) can't accommodate leo's situation. We know, from long and painful experience, that forcing some Byzantine inheritance hierarchy (if we were able to come up with such a hierarchy in this situation) produces brittle code that is highly prone to breaking.

All this has produced great consternation in the ColdFusion community. Developers have been clamoring for ColdFusion to introduce interfaces and a null object and some of the more foolish among us (that would include me) have spent far, far too long trying to work around these problems.

But are they real problems? I have come to the conclusion that they are only problems if we insist that ColdFusion be statically typed. Starting with this prerequisite, ColdFusion falls far short of what we need. But why should we start with this prerequisite? Why should ColdFusion be statically typed at all?

Next month, we'll look at ColdFusion components from the perspective of dynamic typing. It turns out that looking at components in this way removes the problems of the lack of null and interfaces and transforms CFCs from anemic imitations of Java into powerful components in their own right.

About the Author

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

hal@halhelms.com


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.