

COLD FUSION Developer's Journal

ColdFusionJournal.com

December 2006 Volume:8 Issue: 12

Directory Watcher Dangers

40

**Creating a Flex 2
Signature Grabber 18**

**Where's i-Technology
Headed in 2007 22**

**MAX 2006:
"I Can't Wait To
Get Started" 32**

Presorted
Standard
US Postage
PAID
St. Croix Press

www.AjaxWorldExpo.com

AJAX WORLD EAST

CONFERENCE & EXPO



SEE PAGE 39



Create Tomorrow's Rich Internet Applications Today

The combination of the rich user experience of Adobe® Flex™ 2 and the productivity of Macromedia® ColdFusion® MX 7 provides a fast, easy way to create rich Internet applications.

Adobe has created a deep level of integration and productivity tools for Flex and ColdFusion making it easy for ColdFusion developers to get started and become productive building Flex applications.



Download a free trial today at
www.adobe.com/products/coldfusion/flex2

Better by Adobe™

Adobe, the Adobe logo, ColdFusion, Flex, Macromedia, and "Better by Adobe" are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. All other trademarks are the property of their respective owners.
© 2006 Adobe Systems, Incorporated. All rights reserved.





ColdFusion Hosting is our Complete Focus

➤ **POWERFUL HOSTING PLANS**

FREE SQL server access | FREE account setup | Unlimited email accounts Generous disk space & data transfer
30 day money-back guarantee | Great value

➤ **RELIABLE NETWORK**

99.99% average uptime | State-of-the-art data center with complete redundancy in power, HVAC, fire suppression,
bandwidth and security | 24/7 network monitoring

➤ **FANTASTIC SUPPORT SERVICES**

24/7 support services | Knowledgeable phone support | We focus on your individual needs

CFDynamics

866.233.9626 ➤ CFDYNAMICS.COM

For years we have been involved in the Cold Fusion community and have come to know what developers and project managers look for in a web host. The combination of our powerful hosting plans, reliable network, and fantastic support sets us apart from other hosts.

Real service. Real satisfaction. Real value. Real support. Real Freedom.





**For the greatest hits
of the 70's, 80's and 90's
call your web host's
tech support.**

For answers call us at 1-866-EDGEWEB
3 3 4 3 9 3 2

When calling your web host for support you want answers, not an annoying song stuck in your head from spending all day on hold. At Edgewebhosting.net, we'll answer your call in two rings or less. There's no annoying on-hold music, no recorded messages or confusing menu merry-go-rounds. And when you call, one of our qualified experts will have the answers you're looking for. Not that you'll need to call us often since our self-healing servers virtually eliminate the potential for problems and automatically resolve most CF, ASP, .NET, SQL, IIS and Linux problems in 60 seconds or less with no human interaction.

Sleep soundly, take a vacation, and be confident knowing your server will be housed in one of the most redundant self-owned datacenters in the world alongside some of the largest sites on the Internet today and kept online and operational by one of the most advanced teams of skilled Gurus, DBAs, Network and Systems Engineers.



For a new kind of easy listening,
talk to EdgeWebHosting.net

<http://edgewebhosting.net>

By the Numbers:

- 2 Rings or less, live support
- 100% Guarantee
- 99.999% Uptime
- 2.6 GBPS Redundant Internet Fiber Connectivity
- 1st Tier Carrier Neutral Facility
- 24 x 7 Emergency support
- 24 Hour Backup
- Type IV Redundant Datacenter



2003 - 2006

◦ Shared Hosting ◦ Managed Dedicated Servers ◦ Managed Colocation ◦ Semi-Private Servers
◦ ColdFusion ◦ BlueDragon ◦ ASP ◦ .NET ◦ .Linux ◦ .Java ◦ SQL Server ◦ .MySQL ◦ Self-Healing Servers

**editorial advisory board**

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

editorial**editor-in-chief**

Simon Horwith simon@sys-con.com

executive editor

Nancy Valentine nancy@sys-con.com

research editor

Bahadir Karuv, PhD bahadir@sys-con.com

production**lead designer**

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

Louis F. Cuffari louis@sys-con.com
Tami Beatty tami@sys-con.com

editorial offices**SYS-CON MEDIA**

577 Chestnut Ridge Rd., Woodcliff Lake, NJ 07677
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
577 Chestnut Ridge Rd., Woodcliff Lake, NJ 07677

©copyright

Copyright © 2006 by SYS-CON Publications, Inc.
All rights reserved. No part of this publication may
be reproduced or transmitted in any form or by any means,
electronic or mechanical, including photocopy
or any information, storage and retrieval system,
without written permission.

Worldwide Newsstand Distribution

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

For promotional reprints, contact reprint
coordinator 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.

A New Year's Resolution: Get to Know ColdFusion



By Simon Horwith

Keeping up
to date with
the most
recent versions of soft-
ware and programming
languages is the toughest

challenge for me as a developer.

With each passing year it seems that the demands of being a top Web developer require proficiency in more and more tools and languages. Gone are the days of just knowing HTML and JavaScript Validation.

A Web developer today has to be competent not only with HTML and simple JavaScript, but also with CSS, XHTML, object-oriented JavaScript, DHTML, and AJAX to name a few...and that doesn't include server-side development or use of Flash/Flex (another industry standard). Many of us are also expected to keep up to date with the newest version of ColdFusion the minute it's out, and it's becoming more common to be expected to master ActionScript and Flex as well. I'm going to ignore all of the Web standards and Flex 2/ Flash and focus on the importance of being more familiar with ColdFusion in this month's editorial.

In the course of three years, from the release of ColdFusion MX to the release of ColdFusion MX 7.02, the landscape has completely changed for CF developers. For the first time we had native XML parsing, a report building tool, a new charting tag suite, a more feature-rich version of Verity, the native ability to cluster ColdFusion server instances, SOAP (Web services) support, the ability to do Flash Remoting, access to server-side events, an Event Gateway framework, and object-oriented CFML (in the form of ColdFusion Components). We also had dozens of other minor new features and changes added to the

language and to the CF Administrator, and an entirely new underlying platform to learn to optimize, configure, and troubleshoot. Not one thing in that list existed prior to the release of CFMX.

These are a lot of features to learn how to implement and how best to use on our applications, and many CF developers have barely scratched the surface of several of them. For example, I wonder how many of you have written an event gateway? How about even using an event gateway in an application? What about the report builder...are you able to create reports and sub-reports quickly or easily or have you managed to avoid learning that tool? Hopefully by now you are aware of the benefits of using ColdFusion Components to represent business logic, but are you doing it in your applications on a daily basis? Have you become comfortable with the idea of object-oriented CF yet? It's a lot to learn and a hell of a lot to master, and I've got news for you: ColdFusion 8, currently code-named Scorpio, is going to be another big feature release, with many, many new features being introduced.

ColdFusion MX 7, a robust feature-rich release, was very quickly and widely adopted by companies already using and new to ColdFusion. It's easy for the owner of a company to see marketing materials about the new features and say "neat - let's get it," but for many developers the implications of this aren't quite as easy because of the large new feature set to master. With another feature-rich release looming in the future, what can you do to prepare?

— CONTINUED ON PAGE 12

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal and is the CIO at AboutWeb, LLC, a Washington, DC based company specializing in staff augmentation, consulting, and training. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia.
simon@horwith.com

BALANCE

Designer/developer, front-end/back-end, clients/sanity. . .web development is a balance and we can help you maintain it. Join now and experience a wealth of training resources tailored to the tools you use every day.

www.communitymx.com



Visit www.communitymx.com/trial/ for your free 10 day trial.



founder & ceo

Fuat Kircaali fuat@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 sales director

Megan Mussa megan@sys-con.com

advertising sales manager

Andrew Peralta andrew@sys-con.com

associate sales manager

Kerry Mealia kerry@sys-con.com
Lauren Orsi lauren@sys-con.com

sys-con events

president, events

Carmen Gonzalez carmen@sys-con.com

customer relations

circulation service coordinator

Edna Earle Russell edna@sys-con.com

sys-con.com

vp, information systems

Robert Diamond robert@sys-con.com

web designers

Stephen Kilmurray stephen@sys-con.com
Richard Walter richard@sys-con.com

accounting

financial analyst

Joan LaRose joan@sys-con.com

accounts payable

Betty White betty@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



Architecting Your Objects

How to write more maintainable applications



By Peter Bell

Once you've learned

the syntax of cfc's, one of the hardest things to do is to figure out

exactly how and where to use them. The goal of this article is to run you through the most common (and useful) ways to use cfc's to make your applications easier to maintain.

The main benefit of cfc's is that they can make your applications more maintainable. If you're just creating a page with a form that saves user input to a database, you probably don't need cfc's. However, as your applications become more complex, cfc's can make them easier to maintain by breaking your code down into smaller chunks that are easier to find, edit, test, and debug as your user's requirements change.

Design Patterns

Design patterns are simply "proven solutions to recurring problems." In the programming world the term was coined by the "Gang of Four" who wrote a book full of common patterns back in 1995. Design patterns provide two main benefits. First, they show a proven way to solve a particular problem. Second, they give you a shared vocabulary to talk with other programmers about solving those problems.

Think of design patterns like recipes for "baking" successful code. It's important to realize that the only reason to use a particu-

lar design pattern is to solve an associated problem. You don't get extra points for using patterns you don't need. In fact, one of the biggest risks when you start writing code with cfc's is that you will put in patterns you don't really need – resist the temptation!

Nothing is Free

Most object-oriented design patterns were created to solve the problems of managing complexity and improving the maintainability of your code. They do this at the cost of increasing the complexity of the structure of your application and decreasing its performance.

They increase the complexity of the structure of your application so that you can break your code into smaller chunks, making it easier to find, modify, and test each piece. They decrease its performance because the overhead of getting all those little chunks to talk to each other is not zero (it's often small enough to be unimportant, but it's never zero).

However, once you get used to the more complex structure, it's actually easier to find the code you want, and on the whole programmers are more expensive than computers, so even though most of the design patterns will affect the performance of your applications, it will be well worth it to make them more maintainable.

OO-101

Just before we get started with design patterns, let's revisit why we use cfc's and object-oriented programming. Over the years it's been found that putting information (variables) and actions (functions) together into self-contained objects (in ColdFusion we use cfc's) can make for more maintainable code. Why? Because it lets you decrease the dependencies (coupling) between different parts of your application.

If you have a "user" object, as long as

TABLE OF CONTENTS

Creating a Flex 2
Signature Grabber
By Michael Givens...18



Where's i-Technology
Headed in 2007
By Jeremy Geelan...22



MAX 2006: "I Can't Wait
To Get Started"
By Paul Mignard...32



Directory Watcher
Dangers - A Follow-Up
By Dave Ferguson with Jeff Peters...40



editorial

A New Year's Resolution:
Get to Know ColdFusion

5

By Simon Horwith

cfc

Architecting Your Objects

7

By Peter Bell

workflow

Introducing Aptana:
An Alternative to Eclipse
Making your API

16

easier to use

By Robert Blackburn

cfugs

34

ColdFusion User Groups

teamwork

Frameworks, Frameworks,
Frameworks

36

Why you should use a
framework – PART 1

By Faisal Abid

foundations

ColdFusion and the Rise of
Right-Brained Thinking

46

It's time to rock the house

By Hal Helms

interfaces

Why Interfaces in ColdFusion
Are Irrelevant

48

An overview

By Brandon Harper



you don't change the methods (functions) it provides, you can change how the user works internally without breaking the rest of your application. So, when a client comes along and asks you to change how one part of his application works, it's now less likely to break the rest of the application.

What Makes an Application Maintainable?

There are a number of common “best practices” that have been proven over the years to make code more maintainable. Many design patterns can be explained by the fact that they help you to implement these best practices.

- **High Cohesion:** The code you put in a cfc wants to be as logically cohesive as possible – it should “hang together.” That's why we create business objects like users or invoices to put all of their properties (variables) and methods (functions) together and why classes with functions that have little in common (say, general utility classes) should be avoided where possible.
- **Loose Couplings:** Even more importantly, you want to minimize the coupling between different classes (classes is another term you might hear for cfc's – a cfc can be called a class). This is important because when you have to change a class you don't want it to break too many other parts of your application.
- **Program to Interfaces – Not Implementations:** One way to decrease the coupling between classes is to make sure that you treat each cfc as a black box. Pretend you don't know how it works – only what methods (functions) it provides. That way, if you have to change the implementation later (how the methods work), it won't break the rest of your code.
- **Tell – Don't Ask:** Another key to decreasing coupling is to “tell” a class to do something instead of “asking” it for information. A general rule is that if you ask a cfc for two or more pieces of information that you're operating on, you should try to replace that with a single request for the cfc to do whatever you were doing. Don't ask a TaxRate.cfc for the tax rate and whether to tax shipping. Pass it the order and ask it to calculate the tax. That way, if the rules for calculating taxes ever change, you'll know that all you have to do is change the code in the TaxRate.calculateTax() method.

Business Objects

The first step in architecting an object-oriented application is to come up with a list of business objects. Business objects are the real-world “things” that your application models. In a shopping cart you might have products, categories, orders, and order items. A content management system might have users, pages, and articles. An accounting system might have vendors, customers, invoices, and bills. Coming up with a good domain model (a well-thought out list of business objects) is one of the hardest parts of object-oriented programming and is outside the scope of this article. You might want to start by creating one business object per database table (excluding joining or extension tables), but please be aware that a well-designed object model is often very different from

the list of tables in your applications database.

MVC

The first pattern you'll want to consider is MVC – Model-View-Controller. The Model is the meat of your application – the business objects and their associated business logic. The View handles displaying the state of the application and the Controller lets the user select a view and interact with the model.

The goal of MVC is to clearly separate your business logic and views to make them easier to reuse in the application. It also lets you reuse the same model code across multiple display technologies (perhaps an HTML Web site, a Flex front-end and a Web Service). Most non-trivial applications should use MVC to get the business logic out of the page views and improve the readability and maintainability of the code. Popular community frameworks like Mach-II and Model-Glue implement MVC and provide a great example of the pattern.

Introducing the Model

Once you have your domain model, you still have a bunch of architectural choices to make. Most real-world systems don't have one cfc per business object in the model. You'll often see DAOs, Service cfc's, Gateways, and maybe even Transfer Object cfc's as well as the business objects for each object in the domain model.

- **Service Objects:** Service objects (sometimes called Managers) are used to provide an API to the Business objects. Usually named ProductService.cfc or ProductManager.cfc (for a Product object), they are singletons, which just means that there's only one of each at any given time in your application – you don't need three ProductService's running – just one. If you want to get a list of users, ask UserService.cfc. Want to delete all of your old orders? Ask OrderService.cfc. The Service objects (sometimes also called the “service layer”) along with the Business objects are the only parts of your model that the controller should speak to.
- **Data Access Objects (DAOs):** Data Access Objects are used to encapsulate (hide) your SQL code so that all of your SQL queries for each object are in one reusable place. DAOs are often used just to encapsulate single record access (insert, update, delete, and select for a single record), with the Gateway handling operations on multiple records.
- **(Table Data) Gateways:** The Table Data Gateway (usually just called a “Gateway”) is a pattern for putting all of your operations on multiple records in one place. This will always include select statements and could sometimes include delete or even insert or update methods if they're operating on collections of records.
- **Transfer Objects:** Mainly used for passing objects to remote systems (e.g., for a Flex front-end), Transfer Objects are really just structs of data without any business logic stored in a very simple cfc. They really aren't very “object-oriented” (they don't combine properties with methods) but they can be useful in solving specific problems when building a project.

Architecting the Model

Most object-oriented ColdFusion developers create a Service class, a Business object, a DAO and a Gateway for each domain object (product, user, invoice, company, etc.). The DAO and the Gateway are only ever called by the Service method and the Business object is mainly used for passing round information with getters and setters to encapsulate (hide changes in) getting and setting of individual values.

In a simple application, a product might have a ProductService.cfc with methods like getProductbyCategory(CategoryID), getProductbyID(ProductID), getProductbySKU(ProductSKU), deleteProduct(Product), and saveProduct(Product). The Product Business Object (Product.cfc) would have methods for getting and setting its properties so there would be getPrice() to get the price of a product and setPrice() if you wanted to set the price of the product – perhaps as part of processing a product administration form for the site manager to update the product prices. The controller would only ever speak to the ProductService and Product objects.

The Service class would then call the ProductDAO's getProductbyID(), getProductbySKU(), deleteProduct(), and saveProduct() methods to get, delete, and save a single product respectively and would call the ProductGateway's getProductbyCategory() method (because there could be more than one product in a category) to get all of the products in a category. The getProductbyID() and getProductbySKU() typically return a loaded object ready for passing back to the controller. The deleteProduct() and saveProduct() either return just the status of the request or an object containing the status of the request. The ProductGateway.getProductbyCategory() returns a recordset ready for displaying using a simple cfoutput.

Improving the Model

First, you should consider putting all of your “Product” objects in one directory, all of your “User” objects in another and so on. That way you can make all of the DAO and Gateway methods “package” types (rather than public or private) so that they are only available from other objects in the same directory – in this case the Service class and the Business object. Second, you need to ask whether you'll ever have to calculate the values of your objects properties (perhaps a product's price or a user's age) when you are dealing with lists (like getProductbyCategory(), which returns multiple records). If you do, the above approach doesn't work very well since you need to put all of your business calculations in the database, loop through your recordset in the Service object to do any calculations, or put your calculations in your views (not a good idea). See *CFDJ* October 2006 (p. 16) for more information on why you might want to encapsulate your recordsets.

If you choose to take this approach, the Gateway disappears, the ProductService always calls the ProductDAO for any

database-related tasks, and instead of getting an object from getProductbySKU() and a recordset from getProductsinCategory(), both now return an Iterating Business Object that gives you the benefits of encapsulating your getting and setting (making your code more maintainable) while still providing good performance. I'd personally recommend starting with Gateways and recordsets and then moving to this approach only if you find that you have values in your recordsets whose calculation you'd like to be able to vary.

Handling Dependencies

We now have a bunch of objects that need to talk to each other (the Product service needs to talk to the Product DAO, and so on). But how does that work? We could just throw them all into the application scope. Then when ProductService wants to get a product by ID, it would just call application.ProductDAO.getProductbyID(ProductID). This works for a while, but as your application grows it can become a little unwieldy. It also makes it harder to test components using Unit Testing (see <http://www.cfcunit.org> for more information on Unit Testing in ColdFusion). You could create a Factory (another design pattern) to create and/or return your objects, but then you'd need to access the Factory (either by calling it in the application scope or by passing it into every object).

A better solution is to use a framework to automatically provide all of your objects with any other objects they need. The most popular framework for doing this in ColdFusion is ColdSpring. With ColdSpring, you just add a cfargument to your init() method for every dependent object (so ProductService, which needs ProductDAO, would have a ProductDAO cfargument in its init() method) and create a simple XML file describing your objects and ColdSpring will take care of providing the right objects when you need them.

Sometimes people without an object-oriented programming background get confused by what ColdSpring does because it solves a problem they don't yet know they have. I'd start by throwing your objects into an application scope to get a sense of the problems that causes. Once that starts to become a problem, head on over to <http://www.coldspringframework.org> and it should make perfect sense!

If you know what an Active Record pattern is and find yourself wishing that ColdSpring was optimized for injecting dependencies into transient Business objects as well as singletons (which it isn't), you might want to check out LightWire at <http://lightwire.riaforge.org>.

Understanding the Controller

The most misunderstood part of a well-designed object-oriented application is the controller. In the past our “controller” methods often did a lot of work. In a well-architected OO application, they do very little indeed. The job of a controller is to do two things. First it should take values from the URL, form, and

“As your applications become more complex, cfcs can make them easier to maintain by breaking your code down into smaller chunks that are easier to find, edit, test, and debug as your user's requirements change”

Other companies in this magazine spent a lot of time on pretty ads. As you can see, we did not. We spent our time hiring the best people and training them to deliver outstanding support for your website. We spent our time building a state of the art datacenter and staffing it with people who care about your website like it's their own. Compassion, respect, credibility, ownership, reliability, "never say no," and exceed expectations are words that describe our service philosophy. From the first time you interact with us, you'll see what a difference it really makes. And you'll also forgive us for not having a pretty ad.



WEB HOSTING • MANAGED DEDICATED SERVERS • COLOCATION • VPS • ECOMMERCE • BLOGGING • EMAIL

(possibly) session scope to make well-parameterized calls to the appropriate method in the model. Second it should take the information returned by the model and pass it to the appropriate view for displaying on-screen. That's pretty much it!


Recently there have been a lot of questions about how to integrate Model-Glue and Mach-II with Flex front-ends or Web Service calls. The answer is that you don't. The Flex front-ends and Web Service calls should speak to the model directly (well, actually via a simple remote façade to handle any differences in data formatting between the model and the remote system). They shouldn't speak to your HTML controller (such as a Mach-II or Model-Glue controller) at all. If you feel you need to make your Flex applications or Web Services speak to your HTML controller because otherwise you'd have to replicate business logic, that's a perfect sign that you've put too much business logic into your controller and that you need to refactor that and put it into your model instead.

Looking at Views

The goal of a view is to display the information that the model provided to the controller along with any templates, formatting, and the like. There should be no business logic in the views (such as calculating prices or deciding what value to display based on business rules). The view should never include a cfquery that runs against a database (a query of queries for sorting or ordering a recordset may sometimes make sense).

Other than the general rules above, there aren't well-accepted patterns in the ColdFusion community for handling view issues like skinning, view composition (so you can reuse parts of your views on multiple pages), and view helpers (to provide a separation of concerns by removing any programming logic from your templates so designers can safely edit the templates without breaking view-specific code like alternating row tables or "records per page" display logic). However, there are many approaches being explored so it's well worth looking at them.

Conclusion

Using cfcs for your application can make complex applications much more maintainable, but there's a substantial learning curve. Wherever possible, start with small test applications and then as you get more confident using the patterns you'll find your development speed increasing. If you have to start with an existing system, solve one problem at a time. Pull your data access logic into DAOs. Then start to add a service layer and remove business logic from your views and controllers one at a time. The trick? Lots of little steps! 

About the Author

Peter Bell is CEO/CTO of SystemsForge (<http://www.systemsforge.com>) and helps web designers to increase their profits and build a residual income by generating custom Web applications – in minutes, not months. An experienced entrepreneur and software architect with 15 years of business experience, he lectures and blogs extensively on application generation and ColdFusion design patterns.

<http://www.pbell.com>
pbell@systemsforge.com

A New Year's Resolution: Get to Know ColdFusion


– CONTINUED FROM PAGE 5

First and foremost, try to get as up to speed with the current features as you can. In ColdFusion MX 7.0.2, there are 291 CFML functions and approximately 106 CFML tags. It's not reasonable to expect that you would have complete mastery of the skills required to properly use them all, but it's certainly feasible that a developer can learn enough about their syntax and purpose in order to have a better understanding of how to solve any problem they face in code and to quickly know where to turn for the answers. The first thing every developer should do when sitting down to a new development environment is download the free CF Documentation from Adobe (available at <http://www.adobe.com/support/documentation/en/coldfusion/>). These documents (available as pdfs for off-line use) are an invaluable source of information for any developer using ColdFusion so always keep them handy.

I guarantee that being more familiar with all of the current features will aid you tremendously in getting up to speed with the next batch of enhancements. In addition, I recommend starting small – pick focused areas to master. Rather than learning everything at once, pick a specific feature set (charting, reporting, CFCs, etc.) and spend anywhere from a few hours to a few weeks just learning and practicing using that feature set. In no time, you'll find that you've learned a huge amount about what the server can do and about how the features are implemented.

I also recommend being ready. You know CF 8 is coming. You may not know when and you may not know exactly what it will contain, but you know it's going to have a lot of great new features and you know it's coming, so put together a game plan now about how you're going to make the time to learn it whenever it does come out. Talk to your boss about it – letting him or her know now that you want a week or more (if possible) to study the new version when it is released can't hurt (and shows a lot of initiative). If you or your department has a training budget, make sure that you've kept some of that budget stashed away for training in the form of classes and/or conferences after the new version is released. There's a lot you can do to be ready.

It's often said that most CF developers use 10 particular tags for 90% of what they do. While this is true, it also accounts for the fact that most CF developers aren't getting their money's worth from the server or from their own code; when a new feature is released that they want to use, implementing the new feature will take longer, possibly resulting in the feature not being implemented efficiently or effectively, and in any case could make implementing the new feature a daunting task. This is why, though you may not necessarily need to implement every feature or even more than 10 tags in your applications, it is essential for ColdFusion developers to be studious and passionate when it comes to the onslaught of features that Adobe's server team has hurled at us. They won't relent and neither should we.

A very senior member of the CF server team once said to me, "We just make the tools – it's up to you developers to figure out the best way to use them." Brilliantly put. It's not just our livelihood but our responsibility to explore every capability the server has to offer. This is a New Year's resolution we should all be making to ourselves and our employers for 2007 and so long as Adobe and the ColdFusion Server Team continues delivering the kind of feature-rich versions we're fast becoming used to for many more years to come. 

<cf_essentials>



Indispensable CFMX tools.



FusionReactor

CFMX / J2EE Server, database
and application monitoring

from **\$249**



FusionDebug

Interactive Debugging
for ColdFusion MX

from **\$99***



www.fusion-reactor.com

* Using GOT2DEBUG discount coupon. Offer ends October 31st.

Trademarks and Registered Trademarks are the property of their respective owners.

Meet Robert

A business executive at a popular social media site

Challenges

- Encountering poor video quality due to exponential growth in traffic.
- Increasing bandwidth costs due to growing audience size.
- Experiencing compatibility issues due to user-generated video arriving in multiple formats.

Solutions

- VitalStream Streaming Services – Improved quality of end-user video experience using a scalable and global content delivery network.
- VitalStream Advertising Services – Transformed the delivery of digital media from a cost center into a profit center.
- VitalStream Transcoding Services – Automatically converted all user-generated content into the leading streaming media format.

Providing End-to-End Solutions for Your Business

VitalStream is more than a rich media delivery company. We are your professional partner providing solutions that meet your unique business challenges. To learn more about VitalStream solutions, call 800-254-7554 or visit www.vitalstream.com/go/solutions/



Digital Media Solutions for Your Business





Introducing Aptana: An Alternative to Eclipse

Making your API easier to use

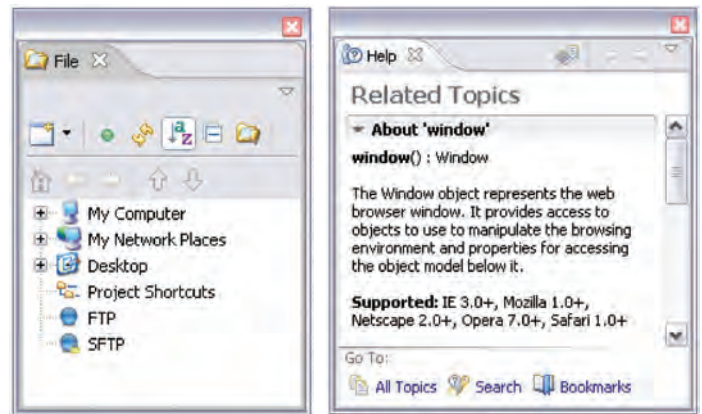
By Robert Blackburn

A few months ago I stumbled on an IDE named Aptana and I instantly recognized that it was the perfect IDE for developers who have found it difficult to convert to CFEclipse. And why is it such a great replacement?

For starters Aptana is actually based on Eclipse. That means it has all the great features that you've heard of about Eclipse, including the ability to use the CFEclipse plug-in with it. You can install CFEclipse the same way you'd install it for Eclipse, and there's almost no difference in how it works.

But why is this any better than Eclipse? Well, the biggest roadblock for new CFEclipse users is the project-based workflow that it requires. But Aptana supports both the project-based workflow and the traditional file-based workflow that these users are accustomed to. This is huge, because it means that developers can continue to work the way they're used to while leveraging the advantages of CFEclipse. Yet it still supports Eclipse-style project-based workflow and so you can leverage the advantages that provides. Because of this you can experiment with that workflow so you're not giving up on learning that alternative work style. You're just not forced to work in a way you may not be comfortable with.

But if that isn't enough, Aptana gives you more advantages than Homesite+/CFStudio or even Eclipse by itself. Aptana is primarily a JavaScript and CSS editor. That means you get full code assist, outlining, and support for CSS or JS files. It even includes code insight and assist with user-defined JavaScript functions. This won't help with any inline JavaScript or CSS em-



bedded in a CFML file, but it's very helpful for JS and CSS files.

Aptana also has extensive help documentation. Its web site contains wiki-style documentation that's quite extensive and clear (<http://www.aptana.com/docs>). Not only that, the Aptana IDE has extremely well-developed help documentation. You can hover over a code element and a help overlay appears with a description, or click on the code element and hit F1 and the help view displays with information about the selected element. It also picks up any user-defined JavaScript function to provide code help for them too. Just put a JSDocs-style comment ahead of your JavaScript functions. For example:

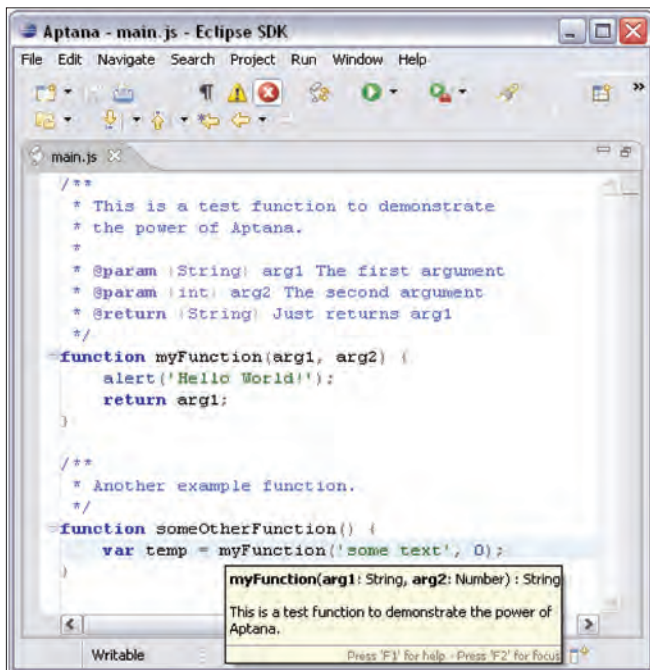
```
/**
 * This is a test function to demonstrate
 * the power of Aptana.
 *
 * @param {String} arg1 The first argument
 * @param {int} arg2 The second argument
```

"Over the last couple of years many ColdFusion developers have attempted to make the transition to CFEclipse, but failed because the mindset and workflow with CFEclipse is so different from what they are used to with Homesite+ or CFStudio. Well, I finally have the solution for you"


```

* @return {String} Just returns arg1
*/
function myFunction(arg1, arg2) {

```




```

    alert('Hello World!');
    return arg1;
}

```

By putting this comment before my function, help information will be displayed whenever you use it or press F1 while it's selected.

Aptana comes in two flavors: a standalone IDE and an Eclipse plug-in. So if you're already an Eclipse user, or ever become one, you can still leverage Aptana and all its great features just by using it as a plug-in like CFEclipse.

There are many features I haven't covered here. For a list of all the great features and for instructions on how to install Aptana, visit the web site at <http://www.aptana.com>. 

About the Author

Robert Blackburn is a developer and team leader of the Internet Application Development team at American Power Conversion Corporation (<http://www.apcc.com>). He's been using ColdFusion since 1999 and has revived and manages the CFUnit open source project (<http://cfunit.sourceforge.net>). He currently has a blog at <http://www.rbdev.net/devblog> for his occasional ramblings.

rwblackburn@gmail.com

Efficient Web Content Management

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

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

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

Evaluate CommonSpot today.

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

features.

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

1.800.940.3087
www.paperthin.com

Paper | Thin

© Copyright 2005 PaperThin, Inc. All rights reserved.

Creating a Flex 2 Signature Grabber

My MAXUP Experience



By Michael Givens

By the time you read this, another MAX will be in the history books. In fact, with a record attendance of around 3,500 developers and designers, this MAX in Vegas will also be in the record books.

Along with the usual great conference sessions, this year there was a special event referred to as the “BarCamp – The un-conference event” called MAXUP. I presented a MAXUP demo, A Flex 2 Signature Panel at MAXUP, where I showed the audience how easy it was to combine a Flex 2 UI for capturing a mouse-created signature with the ActionScript 3 Drawing API and a remote object call to a ColdFusion component. Leveraging work that James Ward blogged about on cayambe.com, I retooled the back-end as a ColdFusion CFC (I make remote object calls to the CFC’s functions), and made some minor UI changes to create a signature panel that could be used for online NDA agreements or e-signatures.

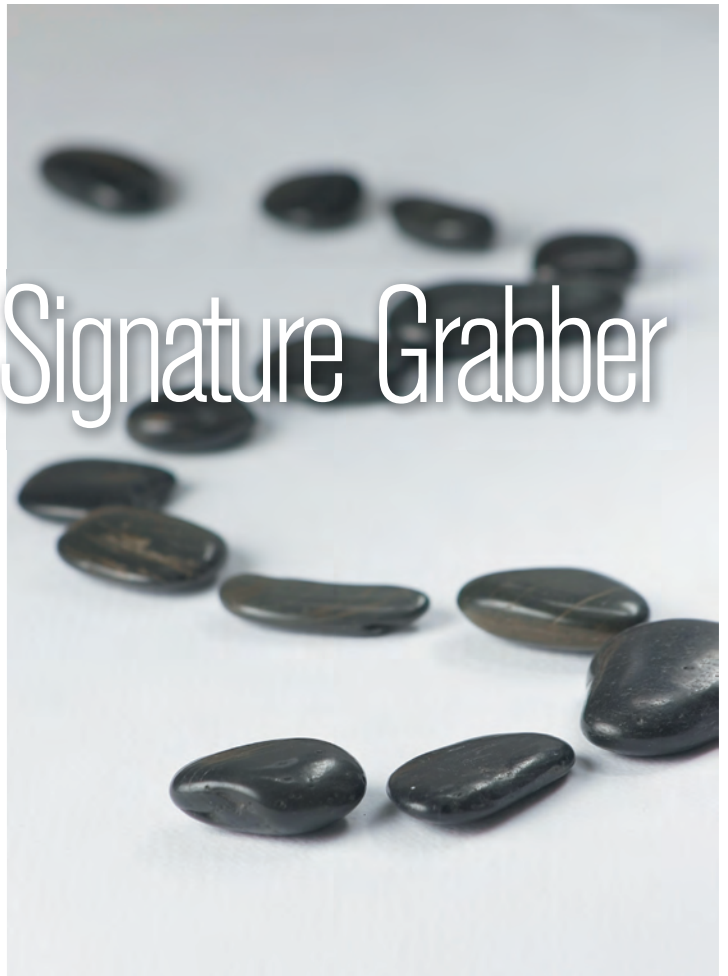
Adobe Flex 2 Builder allowed me to lay out very quickly a simple UI where a user can use a mouse (or even better a drawing pen) to write out messages (See Figure 1).

The signature pad captures a visitor’s IP address via a remote object call to a CFC method `getVisitorIP()` during the Flex application’s initialize event:

MXML:

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" initialize="roSign.getVisitorIP()" creationComplete="setColor()" styleName="plain" viewSourceURL="srcview/index.html" horizontalAlign="center" verticalAlign="top">
```

```
<mx:RemoteObject id="roSign"
destination="ColdFusion"
source="flex2_training.components.signature"
fault="mx.controls.Alert.show(event.fault.faultDetail.toString(),
'Alert');"
showBusyCursor="true">
<mx:method name="getVisitorIP" result="my_IP_handler(event.result)"/>
<mx:method name="doUpload" result="my_CFC_handler(event.result)"/>
</mx:RemoteObject>
```



CFC:

```
<cffunction name="getVisitorIP" displayName="Get IP Address"
hint="Returns Visitor's IP address" access="remote" returnType="string">
<cfset sIP = "#CGI.Remote_Addr#">
<cfreturn sIP />
</cffunction>
```

Clicking the ‘Accept’ button triggers the click event that calls the remote object `doUpload` and corresponding CFC method:

MXML:

```
<mx:RemoteObject id="roSign"
destination="ColdFusion"
source="flex2_training.components.signature"
fault="mx.controls.Alert.show(event.fault.faultDetail.toString(),
'Alert');"
showBusyCursor="true">
<mx:method name="getVisitorIP" result="my_IP_handler(event.result)"/>
<mx:method name="doUpload" result="my_CFC_handler(event.result)"/>
</mx:RemoteObject>
```

```
<mx:Script>
<![CDATA[
import mx.rpc.events.ResultEvent;
import flash.events.Event;

private var sIP:String;
private function doSave():void {
```

```

var bd:BitmapData = new BitmapData(canvas.width,canvas.height);
bd.draw(canvas);
var ba:ByteArray = PNGEnc.encode(bd); // See the PNGEnc.as code
roSign.doUpload(ba,sIP); // pass the ByteArray and the Visitor's IP address
}
]]>
</mx:Script>
CFC:
<cffunction name="doUpload" displayname="Save Signature" hint="Saves a
PNG Signature" access="remote" output="false" returntype="any">
<cfargument name="sigbytes" required="true" type="binary">
<cfargument name="ip_suffix" required="true" type="string">
<cfscript>
var myUUID = "";
    var SigFileName = "";
</cfscript>
<!-- create a FileOutputStream -->
<cfobject type="java" action="CREATE" class="java.io.FileOutputStream"
name="oStream">
<cfscript>
    myUUID = CreateUUID(); // create a unique id

// call init method, passing in the full path to the desired jpg
oStream.init(expandPath("../converted_pngs/signature_arguments.ip_suf-
fix#_myUUID#.png"));
oStream.write(arguments.sigbytes);
oStream.close();

SigFileName = getSigFileName("#arguments.ip_suffix#_myUUID#");
</cfscript>
<cfreturn SigFileName />
</cffunction>

```

Once the doUpload method is complete, the event.result is returned (the dynamic file name [including both an IP address and a UUID] of the stored PNG image file) and assigned to a Flex string variable, SigFileName:

```

MXML:
<mx:Script>
<![CDATA[
..... abbreviated for space saving .....

    private var SigFileName:String;

private function my_CFC_handler(result:Object):void {
SigFileName = result.toString();
btnView.enabled = true;
}
]]>
</mx:Script>

```

Clicking on the 'View' button triggers the click event and the function doView(), which opens the page and the saved PNG image as shown in Figure 2:

MXML:

```

<mx:Script>

```

```

<![CDATA[
..... abbreviated for space saving .....

```

```

private function doView():void {
var u:URLRequest = new URLRequest("../welcome.cfm?sig=" + SigFileName +
".png");
navigateToURL(u,"_self");
}
]]>
</mx:Script>

```

The complete code listing is available at: <http://webcfmx.no-ip.info/flextraining/signature/bin/srcview/index.html>

PNGEnc.as:

```

package {

import flash.geom.*;
import flash.display.*;

```



Figure 1



Figure 2

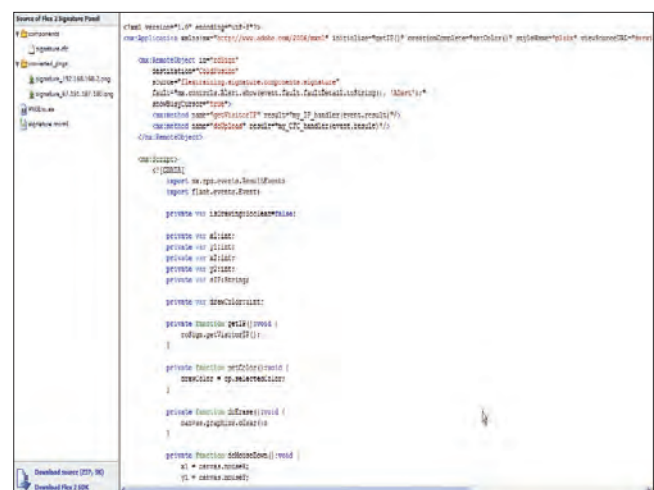


Figure 3



```

import flash.utils.*;

public class PNGEnc {

    public static function encode(img:BitmapData):ByteArray {
        // Create output byte array
        var png:ByteArray = new ByteArray();
        // Write PNG signature
        png.writeUnsignedInt(0x89504e47);
        png.writeUnsignedInt(0x000A1A0A);
        // Build IHDR chunk
        var IHDR:ByteArray = new ByteArray();
        IHDR.writeInt(img.width);
        IHDR.writeInt(img.height);
        IHDR.writeUnsignedInt(0x00000000); // 32bit RGBA
        IHDR.writeByte(0);
        writeChunk(png,0x49484452,IHDR);
        // Build IDAT chunk
        var IDAT:ByteArray= new ByteArray();
        for(var i:int=0;i < img.height;i++) {
            // no filter
            IDAT.writeByte(0);
            var p:uint;
            if ( !img.transparent ) {
                for(var j:int=0;j < img.width;j++) {
                    p = img.getPixel(j,i);
                    IDAT.writeUnsignedInt(
                        uint(((p&0xFFFFF) << 8)|0xFF));
                }
            } else {
                for(var k:int=0;k < img.width;k++) {
                    p = img.getPixel32(k,i);
                    IDAT.writeUnsignedInt(
                        uint(((p&0xFFFFF) << 8)|
                            (p>>>24))));
                }
            }
            IDAT.compress();
            writeChunk(png,0x4944154,IDAT);
            // Build IEND chunk
            writeChunk(png,0x49454E44,null);
            // return PNG
            return png;
        }

        private static var crcTable:Array;
        private static var crcTableComputed:Boolean = false;

        private static function writeChunk(png:ByteArray,
            type:uint, data:ByteArray):void {
            if (!crcTableComputed) {
                crcTableComputed = true;
                crcTable = [];
                for (var n:uint = 0; n < 256; n++) {
                    var c:uint = n;
                    for (var k:uint = 0; k < 8; k++) {
                        if (c & 1) {


```

```

                            c = uint(uint(0xedb88320) ^
                                uint(c >>> 1));
                        } else {
                            c = uint(c >>> 1);
                        }
                    }
                    crcTable[n] = c;
                }
            }
            var len:uint = 0;
            if (data != null) {
                len = data.length;
            }
            png.writeUnsignedInt(len);
            var p:uint = png.position;
            png.writeUnsignedInt(type);
            if ( data != null ) {
                png.writeBytes(data);
            }
            var e:uint = png.position;
            png.position = p;
            var d:uint = 0xffffffff;
            for (var i:int = 0; i < (e-p); i++) {
                d = uint(crcTable[
                    (d ^ png.readUnsignedByte()) &
                    uint(0xFF)] ^ uint(d >>> 8));
            }
            d = uint(d^uint(0xffffffff));
            png.position = e;
            png.writeUnsignedInt(d);
        }
    }
}

```

MAXUP was a great success. Although somewhat nervous before my presentation, I was grinning ear-to-ear afterwards from Adobe's Ted Patrick's encouragement. He said, "Mike, I like the way you used AMF to pass the byte-array to the server-side code." Thanks, Ted.

Reflecting on how much easier it was to create this example compared to the two-month-long project of doing it with the Flash 7 IDE/Flex 1.5/CFMX 6.1/Apache Batik, I am simply amazed by Flex 2's versatility and efficiency in getting the job done quickly. If you still haven't had a chance to try out the Flex 2-CFMX 7.0.2 dynamic duo, I hope this example helps lure you in a bit further. 

About the Author

Mike Givens is the CTO of U Saw It Enterprises, a Web technology consulting firm based in Marietta, GA. He is an Adobe Corporate Champion known to share his experience and evangelism of all things Adobe. Certified in both ColdFusion 5 and as an Advanced CFMX Developer, he has been using ColdFusion since the days of Allaire Spectra. For the last 11 years, he has been seen with his head down – deep in the code.

msgivens@thomcoins.com


```
function optimizeRIA() {  
    if (omniture.actionsource == true) {  
        businessSuccess();  
    } else {  
        if (javascript.futile == true) {  
            businessFail();  
        }  
    }  
}
```

businessSuccess();



OMNITURE®
ActionSource™

A new way to measure the impact of your **Rich Internet Applications**.
No JavaScript required. No hassles. More accurate. More success.

WANT TO LEARN MORE ABOUT OMNITURE ACTIONSOURCE™?

→ www.omniture.com/actionsource

1.877.722.7088

© SEPTEMBER 2006 Omniture, Inc. Omniture, and the Omniture logo are trademarks of Omniture. All other trademarks and logos are the property of their respective owners. All rights reserved.

OMNITURE®
— — —



Where's i-Technology Headed in

SYS-CON Media's Annual Poll of Industry Prognosticators



By Jeremy Geelan

At the end of each year, when SYS-CON informally polls its globe-girdling network of software developers, industry executives, commentators, investors, writers, and editors, our question is always the same:

where's the industry going next year?

Every time, the answers are surprisingly different from the year before, and of course throw light not just on where the industry is going but also how it's going to get there, why, because of who, within what kind of time-scale – all that good stuff. Enjoy!

Ruby on Rails • JRuby • AJAX • Rules-Based Programming



JASON BELL

Enterprise Developer, Editorial Board Member, Java Developer's Journal

My predictions for 2006....

1. **Incremental mainstream adoption of Ruby on Rails**
It's going to happen, isn't it? Keep an eye out for Sun's offering of JRuby. Whether this is the death of other open source scripting languages like Groovy remains to be seen. Ruby has been a wake-up call and has now drawn the line dividing serious scripting languages from "hobby" languages (ones that wouldn't see enterprise adoption). For me, my job just got a whole lot easier, a whole lot quicker.
2. **A slowdown in the AJAX hype**
I think the shine has worn off. There are some nice applications about but at the end of the day it's a Web page with some very fancy JavaScript.

3. **2007 is the year of the business rule**

Rules-based programming will be big business. With the likes of JBoss acquiring Drools it's certainly an area to keep an eye on.

LAMP • REST • ATOM • Apple



DAVID HEINEMEIER HANSSON

Creator of (Ruby on) Rails

1. 2007 will be the year where LAMPers finally decide to stop being neutral about the WS-* mess and pick the side of REST: the next wave of Web APIs will stop supplying both a SOAP and REST API and just go with the latter.
2. On the leading edge, we'll see the same for RSS vs ATOM. For techies in the know, ATOM will become the assumed default syndication format and that'll mark the slow decline of RSS (though more as a technology than as a brand, RSS will remain synonymous with feeds).
3. Apple will continue to trounce everyone else for the preferred geek platform. The stigma of being a Web programmer still using Windows will increase.

Vista • Office 2007 • Zune • AJAX • Ruby • Java Ruby on Rails • Flash Memory



GARY CORNELL

Founder & Publisher, Apress

In no particular order:

1. IE 7 will have a fast adoption curve and so Firefox will cease gaining market share.
2. Vista will have a slow adoption curve.
3. Office 2007 will have a slower adoption curve.

2007?

Enter



4. Oh, the Zune will have no adoption curve.
5. The AJAX bandwagon will gain even more speed.
6. Ruby's momentum will slow down as Python and PHP frameworks to combat Rails grow in popularity.
7. The open-sourcing of Java will have no effect whatsoever on Java's slow decline in favor of dynamic languages (Ruby, Python) and C#.
8. Sales of high powered desktop will slow.
9. Apple will no longer gain market share for its desktops and will stabilize at its current meaningless level.
10. Ultra lightweight notebooks based on flash memory with instant on/off will start coming out in large numbers.

SOA & Web 2.0 • "Outside-In SOA" • Semantic Web • AJAX



DAVID S. LINTHICUM
CEO, The Linthicum Group

1. **The worlds of SOA and the Web 2.0 blur together.** While many who think SOA don't think Web 2.0, and many who think Web 2.0 don't think SOA, those days will come to a fast end in 2007. So, what does this mean to those standing up SOAs today? It's clear that many of the services we consume and manage going forward will be services that exist outside of the enterprise, such as subscription services from guys like Salesforce.com, or perhaps emerging Web services marketplaces from guys like StrikeIron, Google, Amazon, and others. This is outside-in SOA, in essence reusing a service in an enterprise not created by that enterprise, much as we do today with information on the Web. Thus, those services outside of the enterprise existing on the Internet create a Universal SOA, ready to connect to your enterprise SOA, perhaps providing more value.
2. **The rise of the Semantic Web.** The Semantic Web is the abstract representation of data on the World Wide Web, based

on the Resource Description Framework (RDF) standards and other standards. Although this notion has been around for some time, in 2007 it will greatly affect how we design, build, and deploy Web 2.0 applications and SOAs, providing a mechanism to track and leverage application semantics, local and remote.

3. **Enterprise applications continue to move outside the enterprise.** With the success of Salesforce.com and many others, we'll continue to see applications move to the Web including accounting, CRM, HR management, logistics, inventory management, etc. While many Global 2000 companies will fight this trend, the success of the younger and more nimble up-starts will drive this movement quickly.
4. **The success of AJAX drives traditional software back to the drawing boards.** With the ability to finally provide dynamic rich content and applications over the Web, traditional software vendors will find that they need new products to play in this new world. Indeed, as Google Mail is giving Microsoft fits, so will other more innovative Web-delivered applications leveraging rich client technology such as AJAX. Entire interfaces will have to be rewritten to support AJAX, and end users will demand that we move away from traditional pump-and-pull HTTP programming.

Mobile AJAX • "Mobile Web 2.0" • SMS • LBS Flash Lite • On-Device Portals



LUCA PASSANI
Wireless Guru & Technology Evangelist, Openwave

Here are my predictions for 2007:

1. AJAX will still be hyped, but we will still see no mobile AJAX-based killer apps, only proofs of concept.
2. JAVA ME will not gain much more ground. Too fragmented. Games and some other apps. No killer apps though.
3. What people call "Mobile Web 2.0" is not Web browsing.



Saying that mobile and Web will converge is trendy in some environments these days. This is wrong and that's hardly surprising: people buy phones to make calls, not to browse the Web, so why should we expect phones to get so much better at browsing the Web?

4. SMS will still represent 80% or more of data traffic. The rest will be downloads: ringtones, wallpapers and games. WAP will be mostly used as a discovery mechanism to get to those contents. Reformatting proxies to adapt Web content for mobile will be implemented by most operators. They will increase browsing a bit, but nothing earth-shattering.
5. Not sure about Location-Based Services. LBS have been on the verge of explosion for some time now.
6. Flash Lite will make significant progress in Europe and North America, also on operator portals.
7. On-Device Portals are an interesting development: content gets pushed to devices while the user isn't watching and they may decide later to buy it or not. This will be trendy next here. It will be interesting to see which actual implementations of the concept deliver.
8. More people will realize that device fragmentation is one of the main hurdles for mobile.

Flash Memory • AJAX Productivity • Red Hat • Vista Notebooks • Ubuntu



MARK HINKLE

Editor-in-Chief, Enterprise Open Source Magazine

Here are my predictions:

1. **Flash-bootable PCs** – It's been a long-time coming but laptop PCs will start booting from flash memory. This will make a huge difference in battery life. Intel will lead the way pushing their NAND flash boot memory on a new laptop platform and Apple will be among the first to adopt. The One Laptop per Child initiative will also provide a demonstration of the first zero disk drive PCs albeit small. Devices like this will inspire creativity on higher end models especially as the price of non-volatile memory continues to drop.
2. **New Crop of AJAX Productivity Applications** – While the buzz around AJAX may fade, the number of robust new AJAX-enabled applications will increase. These applications will be built on evolving AJAX frameworks like Dojo and Rico and commercially backed platforms like OpenLazlo. Of course every new start-up will be secretly hoping for Google to make a bid and join the family that has been expanded this year by Writely and Jotspot.
3. **Red Hat Will Become an Acquisition Target** – Someone will make a bid on the #1 Linux vendor. Maybe Oracle who has

done a number on the leading Linux vendor with Unbreakable Linux will take advantage of Red Hat's near 52-week low. Uncertainty and ambiguity in the enterprise Linux market will send Red Hat looking for another partner to avoid being swallowed by the DB maker. Maybe IBM will become Red Hat's white knight.

4. **Open Source Everywhere** – More and more companies will open source legacy products and launch new ones under open source licenses. Database vendor Ingres is going to set the standard that other more conservative infrastructure vendors will follow. Look for new open source initiatives from major infrastructure vendors like BMC, VMware, and even Microsoft.
5. **Microsoft Vista Launch Will Boost Sales of Other OSes** – Microsoft's launch of Vista will start to prompt hardware refreshes which can be nothing but good for Apple. Apple already has momentum, Intel hardware, dropping prices and all the tumblers are becoming aligned for it to creep above its measly 5% market share. Linux desktop vendors will likely see a few defectors from the Redmond camp, though big ships turn slowly. Look for Ubuntu to be the Linux desktop distribution of choice.
6. **Half of All New PCs Will Be Notebooks** – PC buyers are buying more notebooks every quarter and sometime in 2007 the number of shipping notebooks will match the number of desktop PCs or come very close.

IT Enabled Services • Web TV • Visual AJAX IDE Microsoft Atlas • Apache XAP

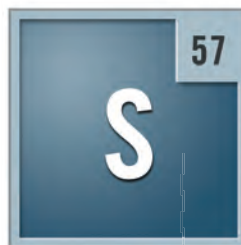


COACH WEI

Founder, Chairman, & CTO, NexaWeb

Here are my submissions:

1. IT Enabled Services is going to fly high in 2007. As a result, we will see:
 - a. A lot more venture capital investments into IT Enabled Services;
 - b. Of course, a lot of startup activities in IT Enabled Services (new company creation, merger and acquisition);
 - c. There will be some significant moves made by "traditional, big companies" into IT Enabled Services too. For example, some of the possibilities are:
 - i. Massive reality shows on the Web, instead of being on TV. Can you imagine "American Idol" on the Web? Speaking of this, I think highly of Yahoo's initiative into this area, including its recent acquisition of Bix.
 - ii. A major entertainment company (NBC, ABC, etc.) fully embracing Web TV.
2. AJAX grows up – which means the following are available and useable:
 - a. Visual AJAX IDE (solving the ease-of-development issue.



STREAM57



Customized Flash-based media solutions

Software and services for collaboration, video conferencing, and e-learning

visit us
stream57.com

e-mail us
streamline@stream57.com

call us
212.909.2550 x1012

Most likely based on Eclipse ATF);

- b. Declarative AJAX Framework (solving the ease-of-development issue. Most likely based on Microsoft Atlas and Apache XAP);
 - c. Adoption of AJAX within less leading-edge enterprises.
 - d. AJAXWorld Conference overtakes JavaOne conference. JavaOne is being renamed as JavaScriptOne Conference.
3. Growing adoption of Web 2.0 technologies within the enterprise
 - a. Enterprise Mashup Server emerges as a product category.
 - b. Less leading-edge companies start to adoption Web 2.0 technologies.
 4. The IPO market shows signs of opening up
 - a. One or two Web 2.0 companies go public, the majority of the exits are acquisitions.
 - b. An increase of IPO filings and going public.

WS-BPEL 2.0 • BPM & Web 2.0 • SOA • XSLT • JSON



JOHN EVDEMON

Architect, Microsoft, with the Architecture Strategy Team focusing on BPM and SOA

E.F. Schumacher, a well-known British economist, once wrote: "I cannot predict the wind but I can have my sail ready." With that thought in mind here are ten predictions and hopes to help get your sails ready for 2007:

1. **The WS-BPEL 2.0 specification will finally be approved as an OASIS standard.** Adoption of WS-BPEL will initially be slow, driven by customer demand. BPEL will evolve beyond a "check box requirement" if people begin using it as a foundation for defining process profiles (conceptually similar to how people use WS-Security today). An updated mapping from BPMN to WS-BPEL will also be published.
2. **The convergence of BPM and Web 2.0 begins.** BPM is about improving performance by optimizing key processes. Web 2.0 is about capturing the wisdom of crowds (or as O'Reilly puts it, the architecture of participation). The convergence of BPM and Web 2.0 enables collaborative development and tagging of sub-processes, establishing a "process folksonomy" where the best processes can evolve organically. Collaboration can occur over simple but highly scalable pub/sub mechanisms (like Atom or SSE). Lightweight tools will enable users to model or reuse sub-processes using a broad set of metadata. While this is an exciting opportunity, there are several technical and non-technical issues that must be addressed before this convergence becomes a reality.
3. **Improvements in SOA management and governance.** Tools, frameworks and platforms will emerge that better enable:
 - Defining and enforcing service development guidelines
 - Modeling, managing and enforcing operational policies (e.g., security, service level agreements and others)
 - Service simulations (what-if scenarios, impact analysis, etc.)
 - Modeling and managing service dependencies
 - Service provisioning and de-provisioning
 - Configuration management
4. **Workflow isn't confined to the datacenter anymore.** Lightweight, extensible frameworks like Windows Workflow Foundation (WF) enable workflow in places where it may not have been previously considered.
5. **Better UI Experiences.** Declarative user interfaces will enable rich user experiences that can be easily modified or extended with simple mechanisms like XSLT. Familiar business applications like Office provide the user interface to back-end line-of-business systems. The line between AJAX-based UIs and rich desktop UIs will blur, enabling users to enjoy both connected and occasionally-connected experiences. Tools and guidance will make building, testing and deploying these composite UI experiences much easier.
6. **A new category of architecture emerges: Software + Services.** It is hard these days to find an architectural concept that is not somehow tied to services. The line between Web services, SaaS and traditional applications will blur to the point where the location, contract and hosting of a service are less important than the capabilities exposed by the service.
7. **JSON without AJAX.** We'll start to see more people using JSON to address the XML bloat problem outside of simple AJAX-based applications. The downside is that this may result in more tightly-coupled applications.
8. **Events and states instead of EDI-style messaging.** Lightweight frameworks will empower developers to start thinking about solutions in terms of event notifications instead of simple messages passing from point A to point B. Hierarchical state machines enable state synchronization across complex, federated processes.
9. **We stop talking about SOA and "just do it."** Sometimes we spend more time arguing about IT trends than actually using them. In 2007 the tools and specifications we need for enterprise-strength, loosely-coupled solutions have finally arrived – it's time to roll up our sleeves and get to work.
10. **IT finally admits that there is no silver bullet.** Every year I hope to see this happen and every year my hopes are crushed by buzzword-of-the-minute hype machines. (Hey I can dream, can't I?)



Visit the *New*

www.SYS-CON.com

Website Today!

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

24/7

FREE NEWSLETTERS

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

SYS-CON.TV

Watch video of breaking news, interviews with industry leaders, and how-to tutorials

BLOG-N-PLAY!

Read web logs from the movers and shakers or create your own blog to be read by millions

WEBCAST

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

EDUCATION

The world's leading online i-Technology university

RESEARCH

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

MAGAZINES

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

INTERNATIONAL SITES

Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

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



AJAX Over-use • JSF • Relational Object Mapping • Macs



BILL DUDNEY

Editor-in-Chief, Eclipse Developer's Journal

1. AJAX will continue to gain momentum as folks continue to have the epiphany that Web 1.0 UI is not good for users. Overuse of the technology will be a real problem. JSF will finally start to become a de facto as well as actual standard due to its ease of integration with AJAX.
2. Open Source enablement will continue to be a hot spot for VC investment. I don't think the perfect business model will emerge in '07 though so the market will still remain 'immature.'
3. Java Persistence API will bring relational object mapping to the long tail of the market. Early adopters will be wondering what all the hype is because the technology is so old in their eyes.
4. Macs will continue their 'thought leader' adoption curve. This is not the year they start to penetrate the corporate IT department.

Web Service Orchestration • Web Services Explosion



ADAM KOLAWA

Co-Founder & CEO, Parasoft

1. I anticipate a significant demand for Web service orchestration in the upcoming year, especially in the United States.

Many organizations now have at least one Web service, and a growing number already have two or more related Web services. Managing multiple related Web services is considerably more challenging than managing the same number of separate, unrelated Web services. To use these related Web services to achieve your business goals, you need to consider how high-level operations pass through the Web services, then determine how to implement this high-level flow—from start to finish. This can be accomplished in two ways:

- By programmatically coding the application logic required to tie the involved elements together.
- By using an orchestration tool to direct the flow through the involved elements, which remain separate.

I predict that the latter method will be the favorite because it is easier.

2. I also expect an explosion of Web services because they are so easy to expose. Once exposed, Web services basically create interfaces which can be reused. This will significantly reduce the amount of code that needs to be written, which will in turn cut the demand for "bare bones" development.

Server Virtualization • Container-Based Hosting • Linux Rails • Django • Agile Development



BRANDON HARPER

Senior Software Developer at Acxiom Corp.

The top five technology trends I see happening in the New Year are:

1. Server virtualization is just getting started, and will really make itself known in the coming year. Once we start seeing the quad core CPU architectures as a part of standard infrastructure, it really starts making a lot of sense to start deploying and managing servers and applications as virtual entities rather than specific pieces of hardware. This helps manage the cost and pain of software configuration management, take advantage of being able to process many tasks simultaneously because of hardware support, as well as allows legacy hardware to be retired in favor of applications running on virtual servers.
2. Container-based hosting is the new kid on the block, and will also start making its presence known in the upcoming year. Commonly labeled as "grid" hosting (which is a technical misnomer if you understand distributed computing), it essentially claims to be an infinitely scalable hosting platform. This technology still seems to be half-baked at the moment, but you could have said the same thing about Linux ten years ago.
3. People who normally wouldn't use Linux start to explore it and even replace Windows with it permanently. With Vista, Microsoft seems to be moving to a model in which the Windows operating system is a method to police users with DRM and other nonsense rather than provide developers with a good platform on which to use hardware, which is what operating systems are really supposed to be. A lot more consumers who haven't noticed this happening in the past will stand up and notice this year.
4. Dynamic languages and frameworks will continue to make leaps in popularity and adoption. Given the current squeeze on technology talent in the US, companies are going to have to learn how to do more with fewer resources. Moving to dynamic languages and frameworks as well as other simplification such as varying Agile software development practices will enable this to take place. I think the obvious leading candidates here are Ruby on Rails and Django.
5. The enterprise will embrace ways to simplify development by continuing to embrace open source software and Agile Development strategies. While there are a lot of cries to the effect of Ruby on Rails replacing Java, I think that's complete nonsense as Java is a language and Ruby on Rails is a framework. Rapid development languages will certainly make some inroads, particularly where heavy tools have been used to build simple applications, Java is still going to be a major part of the service-oriented enterprise for years to come because of the power and tools it provides as well as its industry support.

SaaS • Open Source • Mobile • Enterprise IM Social Networks
• China • Virtualization • RFID • Internet Video • Prediction
Networks • Intelligent Wikis



The Z Generation CIO & IT Professional

By Bob Zurek

Director of Advanced Technologies with IBM Information Integration Solutions

The Z Generation is typically described as those who are born sometime during the early 2000s and continue to the 2017 time frame. So what will these Generation Z IT Professionals be experienced with. Here's my prediction for the Top Ten characteristics (and this is just the tip of the iceberg):

1. Grew up in the world of SaaS and open source and wonders why you would ever license and install software. If you still needed to install software, then it should be available in the form of open source. Expects all internal projects to be developed using the open source model.
2. Grew up with a mobile technology and wonders how anyone could run a business without it. Insists everything be available on a highly portable digital device and everyone in the organization have a device. No exceptions.
3. Grew up knowing how to leverage the power of social networks for the benefit of the corporation. This includes the ability to build out these networks and use them to help build new products and technologies. Generation Z CIOs will have a huge advantage as they have grown up as participants in many social networks. China will be a big source of these networks. Websites will be built by the Z Generation CIO to invite outsiders in to help build new and innovative products that have yet to be thought of by the enterprises internal employees.
4. Grew up using Instant Messaging and will insist that the enterprise use IM as a priority over email and that email will only be used if the communications can't be done using the features of the future enterprise IM platform.
5. Will tap into offerings such as TopCoder.com to supplement project teams. There will be a world of competing Topcoder.com like sites where the best coders in the world will be found to solve very complex algorithms and other challenging software projects facing the IT department. China will be a major provider of these teams.
6. Grew up with a complete understanding of the value of virtualization and therefore, their datacenter will be virtualized and the IT operating fabric will be grid-based, tapping the power of external grids of CPU.
7. RFID Everywhere. The Z Generation will be the ones that take RFID to new heights. Everything that is taggable will be tagged and tracked.
8. CIO and Z Generation IT Professionals will leverage the power of Internet video by taking advantage of companies like BrightCove Networks which will bring knowledge workers engaging channels like "The Customer Service Channel", "The Corporate Strategy Channel", "The M&A Channel" and others. I can see companies like Harvard Business Review and others producing content for these channels. Imagine the "The DataCenter Channel." The topics are endless and will be as easy to find as bringing up your favorite search engine. New content will be generated targeted for companies like "The IBM Channel" or the "GE Channel." I would love to see "The Institute of The Future Channel"
9. Intelligent Wikis will be the primary source of knowledge in an enterprise and will eventually do what data warehousing did for business intelligence. Furthermore, new internal employee-generated communities will spin up to voluntarily invent new projects during their off-hours to showcase their creativity that is typically not known by the employer.
10. CIOs will aggressively adopt Prediction Networks as part of the core business strategy to better help the enterprise gauge where everything from sales to new product development will be successful.

Open Source Java • General Public License v2 • GPL v3



Consequences of Open-Sourcing Java

by Tony Wasserman

Professor of Software Engineering Practice at Carnegie Mellon West and Executive Director of the Center for Open Source Investigation (COSI) The open-sourcing of Java under the GPL 2 license will have a ripple effect on various technical and business issues in 2007:

- First, people will closely study the Java source code and find one or more serious bugs, at least one of which has been there since the earliest days of Java.
- Second, a real-time systems vendor will fork the source code, as permitted by the GPL, and create a variant that is "tuned" for real-time applications. This step will be the focus of a major debate within the Java community.
- Third, the open-sourcing of Java will have a positive impact on the adoption and use of open source software in general.
- Fourth, the use of the GPL 2 for open-sourcing Java will inhibit the completion and acceptance of the GPL 3 proposal.

Open Sun • iPod Uno • IT2 • Microsoft VAPOR



...And in Other 2007 News

by Richard Monson-Haefel

Award-Winning Author & Senior Analyst, Burton Group

1. Jonathan Schwartz open sources Sun Microsystems.
In a move that will surprise everyone Sun Microsystems will announce that it will open source its entire company. Sales, marketing, finance, and even operations will be open to the community for anyone to contribute.
2. **Apple computer announces the iPod Uno.**
The size of a match stick with no screen or controls, the iPod Uno plays one song in a constant loop. Despite its limited capabilities, the tiny device becomes an instant hit and a cultural icon.
3. **In what is heralded as the seminal article on the subject, Tim Berners-Lee mentions "IT2"**
Overnight the term morphs into "IT 2.0," spawning thousands of blog entries and press articles, a dozen books, five conferences, and millions of dollars in venture capital. It turns out that the original article, incomprehensible to most readers, was actually another attempt to explain the Semantic Web and the IT2 reference was just a typo.
4. **Microsoft will create the first CMO (Chief Marketing Officer) position.**
The new CMO will immediately change his own title to Chief Command & Control of Packaging Officer (C3PO) and then announce that Vista will be delayed and renamed Microsoft Virtualization Application Program Operating system Reloaded (Microsoft VAPOR).

About the Author

Jeremy Geelan is Sr. Vice-President, Editorial & Events of SYS-CON Media. He is Conference Chair of the AJAXWorld Conference & Expo series and of the "Real-World Flex" One-Day Seminar series.

jeremy@sys-con.com

Scary Question.

Exactly who is developing your next app?



Contact Us



Address:

555 Not My Home St.
Big City, CO 12345



Your App Starts Here.

We are the leaders in RIA development services.

INCREDIBLE APPLICATIONS

PASSIONATE USERS

PROVEN SUCCESS



Unlock your potential
with the help of industry leaders in
Rich Internet Application development.
10 Years. 1400+ Customers.

Your app starts here.

CYNERGYSYSTEMS.COM



Solution
PARTNER



MAX 2006: “I Can’t Wait To Get Started”

**How I came to give myself
more work and love
every minute of it**



By Paul Mignard

I can't wait to get started." The sentiment struck me pretty hard as I struggled to get some sleep on a packed red-eye flight leaving Las Vegas at 11 o'clock at night.

I had visions of components and Flex Forms dancing in my head and I was still reeling from the sheer sense of empowerment that I felt. It's funny, but sitting there with my laptop loaded with the ColdFusion Developers Edition and a copy of Flex Builder, I felt as if I could write the most elegant piece of Web software that my mind could conjure up. I still feel that way now.

But I'm getting ahead of myself.

Simon Horwith, through his blog, recruited me into checking out Adobe MAX 2006 in Las Vegas and report my findings back to you. Having never been, I wasn't exactly sure what to expect. Other than a lot of swag, I anticipated copious demonstrations of Adobe technology and not much else.

I was sorely mistaken.

It all started with CFUnderground on a beautiful Sunday Las Vegas morning. Having no idea what to expect when I stepped into an abandoned bar at 9 o'clock in the morning, I soon relished my time as I made some quick friends and learned something that was true during my entire time at MAX: these developers are some of the nicest and most transparent people

I had ever met. Numerous times throughout the day it would not be uncommon for someone to open his laptop and expose his code, either to show you some uncanny way of overcoming some coding roadblock or ask how you could make the code better. People weren't arrogant in the way they explained things and it was welcomed when another developer would join the conversation or even just look over a shoulder to gain understanding. CFUnderground was a fantastic icebreaker, and, for me, was the first time I saw ColdFusion as a scalable object-oriented language. Michael Dinowitz gave a remarkable talk about how to think about ColdFusion in an object-oriented way (which was a helpful precursor to the 23-page article he wrote about the subject that I read just three days later). Mini-MAX soon followed (in the same bar no less) and was a rapid-fire succession of some of the speakers and topics we were going to see at MAX.

Monday morning, after some initial issues with my registration, I was presented with the marvelous MAX 2006 conference bag that would be used by many over the next few days to identify themselves in the throng of Las Vegas gamblers and gastroenterologists who had a much less exciting conference right below ours. I found the exhibit hall to be a great hideout during the pre-conference and in between sessions as developers took root in MAX beanbags and couches strewn all about when they weren't trying to pilfer swag away from the vendors that were strewn about as well. The conference store was stocked (although the Flex books flew off the shelves like they were printed on sheets of \$100 bills) and there was always

something interesting going on in the small pockets of people gathered around a laptop looking at some chunk of code or wild Flash movie.

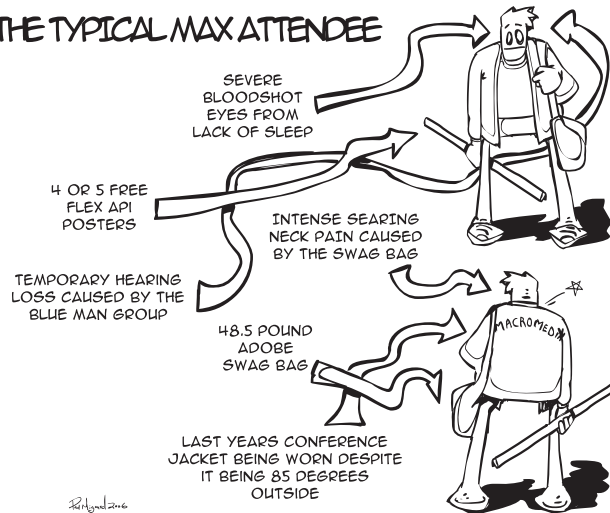
Tuesday is when things really began to roll with the first of three keynote sessions. The first keynote was kicked off by an awesome performance from the Blue Man Group and then went on a whirlwind tour of some of the really exciting things that Adobe has up its sleeve. It started with a talk about the marriage of Adobe and Macromedia and gave way to demonstrations of some of the new features in Photoshop, Dreamweaver, Flash, Acrobat, and other things. Ben Forta rocked by showing off the ColdFusion/Flex Wizard in Flex Builder where, in a matter of minutes, he was able to create an entire Flex front-end and ColdFusion back-end to a few tables he had written for a Flex version of iTunes. From there Kevin Lynch took the stage to demo Apollo, the Adobe desktop publishing application. It was cool that it wasn't this new complicated architecture that we'd all have to learn and pore over to get things published. It was a way for us developers to take the stuff we're doing already with ColdFusion and Flex and bring it to the desktop. Needless to say, it started the conference on a good note. Well, that and the \$100 million venture fund announced by Adobe to invest in Apollo developers.

The Wednesday keynote focused on Verizon and the ability of U.S. phones to run Flash content now through Brew and the different ways Flash developers can get their content to those phones. The "Can you hear me now?" Verizon guy made an appearance as well as the Flash-enabled Chumby. Thursday was sneak peek day as features were finally revealed from Fireworks to Dreamweaver and, of course, ColdFusion. Ben Forta took the stage aptly dressed as Scorpio Man as he and Tim Buntel showed off the ColdFusion eight-server monitoring tool and the much anticipated cfimage tag. There was some other cool stuff in the form of Soundbooth, the Flex-AJAX bridge, a demo of the ability to export MXML from Fireworks, and a very welcome debugger panel to be found in Flash 9. The keynotes were quite a production as over 3,000 people converged into the giant auditorium each morning in fervent anticipation of hearing the morsels of information Adobe was going to release next.

The developer sessions were something that I would really come to appreciate as being the greatest part about MAX. Gone was the lofty manager-speak and 25-cent developer phrases – here's where we got down to code and, at the risk of sounding like a haughty superficial mind shrink, I really enjoyed watching people who knew what they were talking about teach their chosen craft. Any Google search will return the content of those sessions but certainly not the spirit – I learned so much from the guy sitting next to me or from the bowling-shirt-wearing ColdFusion team member who always seemed eager to answer my constant barrage of what I thought to be annoying and elementary questions. Two really great sessions from this track were the "Component Development to ColdFusion MX 7" session taught by Dave Gallerizzo and "Leveraging ColdFusion Components in ColdFusion" taught by Ray Camden (who is about as pleasant and approachable as they come). The Flex sessions were simply amazing and as I sat through them I had to bolt myself to the chair so I wouldn't run out to try all of the new effects and techniques that they were covering. The "Leveraging Flex

2 and Flash Player 9 for Truly Cinematic Experiences" session given by Alex Uhlmann was, well, truly cinematic and the "Building Rich Internet Applications with Flex Builder" by Mark Shepherd was as great a teaching session as it was an example of how fast someone who is experienced with Flex and ColdFusion can build a working and impressive application in a brief amount of time. In short, I couldn't take enough notes and I find myself wishing now that I had had more time to attend more sessions – there was simply too much for one person to take in at once.

THE TYPICAL MAX ATTENDEE



So here I am back on the plane. It's funny, but as I'm sitting here thinking of the various development projects that I know I have to rework, based on my new understanding of how to employ the various Adobe products I use on a day-to-day basis – I can't help but think of how my users are going to react to the new things they're about to experience. What are they going to say when I plunk down that first Flex app in front of them and they see how much more quickly and easily they can do their jobs than previously thought. What are they going to say when the time it used to take me to develop a Flash application drops by 75% or when I ask them to download their first Apollo app to use right from their desktop? Although MAX is a slickly produced and well-funded event, it's really the developers who made the experience for me – the networking alone is worth the price of admission. I left knowing that the ColdFusion/Flex/Flash community isn't one of fierce competition and cutthroat politics, but of friends and a general spirit of cooperation. I'll certainly be coming next year – even if Simon doesn't pay my way again.

About the Author

Paul Mignard is an application developer for Liberty University in Lynchburg, VA, where he uses ColdFusion, Flash, and Flex. When not working he is usually trying to keep his blog updated at www.onekidney.com and loves spending time with wife, family, and friends.

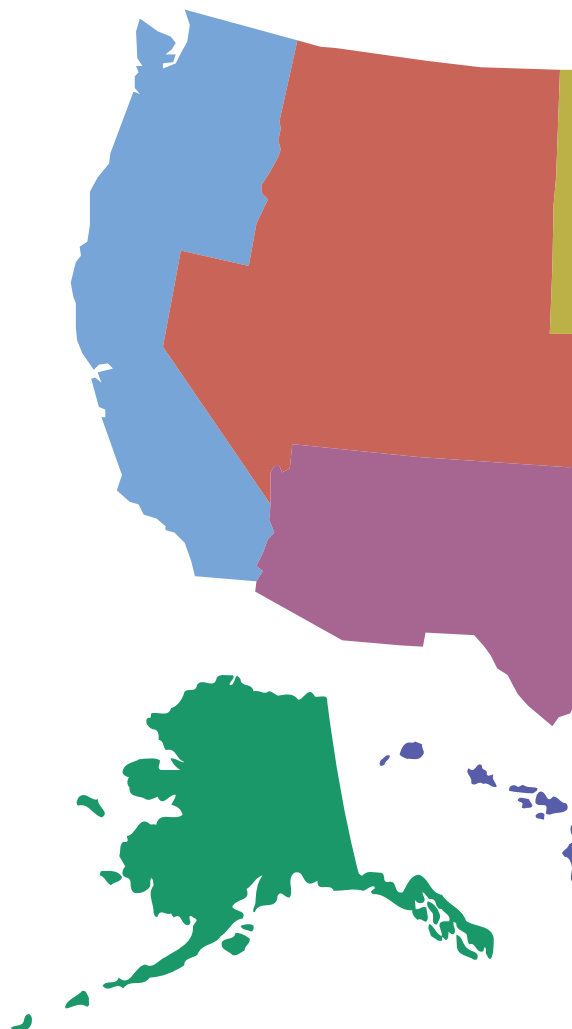
pjmignard@liberty.edu

ColdFusion

For more information go to...

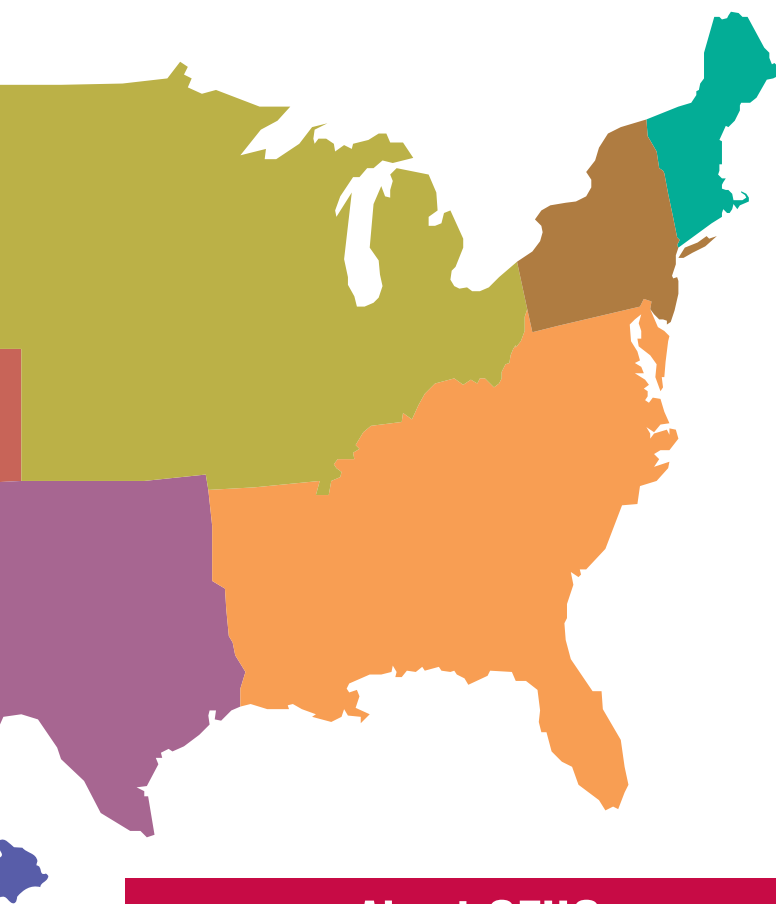
U.S.

Alabama Huntsville, AL CFUG www.nacflug.com	Louisiana Lafayette, LA MMUG http://www.acadianammug.org/	New York Albany, NY CFUG www.anycfug.org
Arizona Phoenix, AZ CFUG www.azcfug.com	Maryland California, MD CFUG http://www.smdcfug.org	New York New York, NY CFUG www.nycfug.org
California Bay Area CFUG www.bacflug.net	Maryland Maryland CFUG www.cfug-md.org	New York Syracuse, NY CFUG www.cfugcny.org
California Sacramento, CA CFUG http://www.saccfug.com/	Massachusetts Boston CFUG http://bostoncfug.org/	North Carolina Raleigh, NC CFUG http://tacfug.org/
California San Diego, CA CFUG www.sdcfug.org/	Massachusetts Online CFUG http://coldfusion.meetup.com/17/	Ohio Cleveland CFUG http://www.clevelandcfug.org
Colorado Denver CFUG http://www.denvercfug.org/	Michigan Detroit CFUG http://www.detcfug.org/	Oregon Portland, OR CFUG www.pdxcfug.org
Connecticut SW CT CFUG http://www.cfugitives.com/	Michigan Mid Michigan CFUG www.coldfusion.org/pages/index.cfm	Pennsylvania Central Penn CFUG www.centralpenncfug.org
Connecticut Hartford CFUG http://www.ctmug.com/	Minnesota Southeastern MN CFUG http://www.bittercoldfusion.com	Pennsylvania Philadelphia, PA CFUG http://www.phillycfug.org/
Delaware Wilmington CFUG http://www.bvccfug.org/	Minnesota Twin Cities CFUG www.colderfusion.com	Pennsylvania State College, PA CFUG www.mmug-sc.org/
Florida Jacksonville CFUG http://www.jaxcfusion.org/	Missouri Kansas City, MO CFUG www.kcfusion.org	Tennessee Nashville, TN CFUG http://www.ncfug.com
Florida South Florida CFUG www.cfug-sfl.org	Nebraska Omaha, NE CFUG www.necfug.com	Tennessee Memphis, TN CFUG http://mmug.mind-over-data.com
Georgia Atlanta, GA CFUG www.acfug.org	New Jersey Central New Jersey CFUG http://www.cjcfug.us	Texas Austin, TX CFUG http://cftexas.net/
Illinois Chicago CFUG http://www.cccfug.org	New Hampshire UNH CFUG http://unhce.unh.edu/blogs/mmug/	Texas Dallas, TX CFUG www.dfwcfug.org/
Indiana Indianapolis, IN CFUG www.hoosierfusion.com	New York Rochester, NY CFUG http://rcfug.org/	Texas Houston Area CFUG http://www.houcfug.org



User Groups

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



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
<http://www.actcfug.com>

Australia
Queensland CFUG
<http://qld.cfug.org.au/>

Australia
Victoria CFUG
<http://www.cfcentral.com.au>

Australia
Western Australia CFUG
<http://www.cfugwa.com/>

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Germany
Central Europe CFUG
www.cfug.de

Italy
Italy CFUG
<http://www.cfmentor.com>

New Zealand
Auckland CFUG
<http://www.cfug.co.nz/>

Poland
Polish CFUG
<http://www.cfml.pl>

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
<http://www.cfugspain.org>

Sweden
Gothenburg, Sweden CFUG
www.cfug-se.org

Switzerland
Swiss CFUG
<http://www.swisscfug.org>

Turkey
Turkey CFUG
www.cftr.net

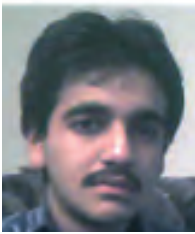
United Kingdom
UK CFUG
www.ukcfug.org





Frameworks, Frameworks, Frameworks

Why you should use a framework Part 1



By Faisal Abid

Your boss tells you that he wants you to build a ColdFusion application based on his requirements, which will take you about a month to complete. You are now faced with two options, you can either start

coding your application in spaghetti code fashion, or you can explore the many frameworks ColdFusion offers and choose one of them.

Writing spaghetti code for many is an easy solution, by approaching the application development with a top-to-bottom declaration of code. Using the spaghetti code method will seem logical at first but you'll soon realize that there's so much code you'll be lost debugging the program for months. A framework for ColdFusion is just like books with a table of contents at the front and an index at the back. Because this pattern is always used, anyone who uses the book knows how to access information they need from that book, so frameworks have the same pattern for universal use. Using a framework(s) is the way to go because it will cut your development time, give you better productivity as well as increased code reusability, and help you organize your code and make it easier for future enhancements.

There is nevertheless a lot of contradiction as to whether frameworks are needed. The main argument on the flip side of the coin is that the learning curve for any frameworks is steep. Well, there is quite a challenge, but if you're going to build something sophisticated in the first place then this learning curve is no challenge to you at all. Another reason why this learning curve is worth the turn is because if you don't use a framework in the long run the code might get so unmanageable that you'll have no choice but to start from scratch, wasting valuable time and in most cases money. Another well-known argument is that frameworks are for beginners who want to use the RAD methodology of development. This, however, isn't true, because

it's a well-known fact that all the big software development firms use frameworks when developing applications. There are many famous frameworks such as the .NET framework for the Win32 and Win64 platforms, and in terms of Web application frameworks there are numerous frameworks that are used by major Web sites such frameworks Coldbox, Fusebox, and Mach II. All in all, yes, there's quite a learning curve, but accepting the challenge will make your life much easier and keep your code managed and clean.

All developers know that time is always against them. A deadline is always looming over a developer's head and he or she is always in a rush to complete the application's lifecycle. It's not that the developer can't write code quickly, it's that once the code is written, debugging takes a much more time. By using an application framework, you'll have enough time to develop an entire operating system. An application framework is a pre-defined structure that will help you organize your application by using various object-oriented programming methodologies. Another reason why your development time will be reduced is that in almost every framework there's a set of predefined "solutions" that you can use to unravel a lot of the problems you might encounter during development. A good example of a framework that has predefined solutions is the Fusebox framework in which you simply tell the circuit.xml file when and what you want to load and it'll do the processing and code generation without any errors.

Another great motive for using an application framework is that you won't have to do repetitive tasks over and over again. For example, if you wanted every page on your site to check if the user is logged in, and you have well over a thousand pages of information the non-framework way of doing it will be to open each page and add a `isDefined(Session.user)` function to do the validation. However, what if you change the session variable to something other than user, you'll have to open thousands of pages again to change that variable. Using this approach will cost a lot of time and valuable resources. On the other hand, if you used a framework, you could have defined that on each page-request executing the function `isDefined(Session.user)` and later on if you were to change the session variable, you could easily change just one line of code in your application framework and it would do the job for you.

Better productivity is what every software developer wants and by using an application framework you'll get just that. In

Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

SPECIAL ISSUE: COLDFUSION SECURITY

COLDFUSION Developer's Journal

September 2004 Volume 6 Issue 9

Sandbox Security
By Alex Hübner

Security: ColdFusion Security Best Practices
Knowing the security risks are there is half the battle
By Bryan Murphy

Security: Public-Key Encryption
Making strong encryption nearly painless
By Wayne Graham

cfUnit: ColdFusion Unit Testing Framework
Paul Kenney's version
By Harry Klein

<bf> on <cf>: Defending ColdFusion Against...
By Ben Forta

Real-World Uses: Using ColdFusion for Network Monitoring
Find the exact piece of data you need
By Ryan Emerle
Matthew I. Fusfield

Security: Top Ten Web Security Tips
You can't be too careful
By Michael Smith

Editorial: Reaching Deeper
Simon Horwith page 5

Community Focus: Problems Securing Your Applications?
Simon Horwith page 7

CF101: ColdFusion's Application Framework
Jeffrey Houser page 28

CFUGS: ColdFusion User Groups
page 40

FOUNDATIONS: 'Selling' ColdFusion
Hal Halm page 46

Training: Fast Track to Flex
Macromedia Training 'Flexes'...

CFML: Harnessing the Power of SQL Server Using Stored Procedures
Use of stored procedures in your CFML architecture in your CFML

Foundations: A Web and Data Abstraction
A perfect marriage between basic object function

<BF> on <CF>: ColdFusion and Data Abstraction
basic object function

PDFs: Creating Your CF Application
only functional...

Editorial: Blackstone
Robert Diamond page 5

CF Community: Tales from the List
Simon Horwith page 7

CF101: What's the Best Approach to Software Development?
Jeffrey Houser page 32

CFUGS: ColdFusion User Groups
page 42

RETAILERS PLEASE DISPLAY UNTIL SEPTEMBER 30, 2004
\$9.99US \$9.99CAN

0 0928102689 1 10

SYS-CON MEDIA

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

That's a savings of \$29.89 off the annual newsstand rate. Visit our site at www.sys-con.com/coldfusion or call 1-800-303-5282 and subscribe today!

COLDFUSION Developer's Journal





Once you're in it...



...reprint it!

- ColdFusion Developer's Journal
- Java Developer's Journal
- Web Services Developer's Journal
- Wireless Business & Technology
- XML Journal
- PowerBuilder Developer's Journal
- .NET Developer's Journal

Contact Megan Mussa
201 802-3024
megan@sys-con.com

REprints


SYS-CON
MEDIA

general, most application frameworks use object-oriented programming methodologies, which give you a lot of code reusability that in turn give you better productivity. For instance, if you were to create a class or cfc component (in this case) of all the items your authentication system does in your application then by using a framework like MACH II or even designing your own proprietary framework you can call the functions in those classes numerous times without having to write the code that many times.

The basic ColdFusion application framework that many like to call the Application.cfc framework is an example of how a framework can lead to better productivity. There are several solutions built into the framework that help you reuse your code. One useful method is the OnError, which is called whenever there's an error on your cfm page. In between this method you can declare some code such as `<cfinclude template='error.cfm' />`, which tells the user to contact you in case of an error and every time an error occurs your code will be executed. This is a good example of code reusability because you don't have to go through all your pages and error check; instead by using that method the framework will do it for you. Also, some developers who are looking to use their time well can rest assured that all of these frameworks are simple to install, simple to use, and simple to deploy. To get any of these frameworks running, all you have to do is create a ColdFusion mapping to the core framework files and then use the framework skeleton application to develop on it. In sum, you'll have a much more productive environment and achieve a higher rate of code reusability.

Another reason to consider using a framework is that over time your code will grow, with all the patches and maintenance you make to it, and if you're using the spaghetti code method your code will look like it's been encrypted using AES. So rather than letting that unfortunate event occur, using a framework will help organize and maintain your code. An example would be using a framework such as Fusebox 5.1 and its MVC architecture to organize your code. Using an MVC architecture helps separate your code logic tier from your presentation

tier and by doing this if you ever have to make modifications you can rest assured that it will have no visual impact on your presentation tier and vice versa. If you were to compare spaghetti code style to MVC-type architecture, you'll clearly see why life is easier on the MVC side. A framework will also increase code readability by organizing the code in a human-readable logical way, an example would be the Coldbox framework, which has a framework dashboard used to organize all your configurations of the application, so instead of digging deep into the code, you can easily configure your framework and application with a simple dashboard. Another great framework that you can use for a simple MVC-type architecture is MACH II in which you'll be able to develop applications faster than MACH II. What a fun framework. Generally speaking, a framework is a really good way to keeping all your code organized and will help you a lot when it comes to updating and adding new features in your code over the course of the application lifecycle.

All in all, using a framework can help increase code reliability by minimizing errors, increase productivity, and maintain your code through out the application lifecycle. As you can clearly see, these give you a significant advantage over the old spaghetti code method and learning how to use a framework can and will give you a good asset in your skill set. Many large corporations are looking for people skilled in many of the top application frameworks such as Fusebox and MACH II. Overall, after using a framework you'll wonder how you ever developed applications without it. 

About the Author

Faisal Abid is a young passionate ColdFusion and Flex developer. He has a Rich Internet Application consultancy in Toronto. You can visit it at www.G-uniX.com. You can also e-mail him about anything relating to Rich Internet Applications. He has a blog at www.G-uniX.com/blog that you can visit to get latest news and updates on G-uniX, Flex, ColdFusion, Spry and the rich Web in general. .

faisal@g-unix.com

Rich Internet Applications: AJAX, Flash, Web 2.0 and Beyond...

REGISTER TODAY AND SAVE!

www.AjaxWorldExpo.com

AJAXWORLDTMEAST

CONFERENCE & EXPO



NEW YORK CITY

THE ROOSEVELT HOTEL LOCATED AT MADISON & 45th

**SYS-CON Events is proud to announce the
AjaxWorld East Conference 2007!**

The world-beating Conference program will provide developers and IT managers alike with comprehensive information and insight into the biggest paradigm shift in website design, development, and deployment since the invention of the World Wide Web itself a decade ago.

The terms on everyone's lips this year include "AJAX," "Web 2.0" and "Rich Internet Applications." All of these themes play an integral role at AjaxWorld. So, anyone involved with business-critical web applications that recognize the importance of the user experience needs to attend this unique, timely conference – especially the web designers and developers building those experiences, and those who manage them.

BEING HELD MARCH 19 - 21, 2007!

We are interested in receiving original speaking proposals for this event from i-Technology professionals. Speakers will be chosen from the co-existing worlds of both commercial software and open source. Delegates will be interested in learning about a wide range of RIA topics that can help them achieve business value.

Directory Watcher Dangers – A Follow-Up

A roadmap from the trenches



By Dave Ferguson
with Jeff Peters

In the July 2006 issue of CFDJ, I wrote about the Directory Watcher event gateway, and how easy it was to set up and how powerful a tool it could be for managing files and external interfaces. While this is true, there are some potential hazards waiting for the unsuspecting developer who jumps into DW waters without a life preserver.

Fortunately, Dave Ferguson has used this particular gateway extensively in his work, and offered to share some of his experiences and solutions with us. Never one to turn down a good follow-up article, I immediately agreed to an education courtesy of Dave, and here it is. – Jeff Peters

When I read Jeff's article, I was interested because I use the DW gateway extensively and I was struck by memories of the double-edged nature of the DW gateway in action. Glory does not come cheap.

Imagine that our hero needs to build a DW gateway to monitor an FTP directory. He writes all the code, does all the tests, then puts the code into the production environment. When it's all done he marvels at his accomplishment, basking in the early glow of success. Then the phone rings – it's Murphy. The new DW gateway system is failing. Files aren't being processed, or only parts of files are being processed. Our hero checks the log files and to his horror it's filled with file-read errors from the gateway.

That beautiful, streamlined, straightforward process, written in accordance with all best practices, has gone to hell. Shaken, our hero begins banging his head against his desk and staring at the screen, wishing it worked the way it's supposed to. Everything is there, the gateway is firing when a file is written, but it's not processing files. Oh, yes –and to make matters worse,





the problem only happens about 30% of the time.

I know all this because I was that guy. This exact scenario happened to me. I went from hero to zero in about 2.3 seconds. About three weeks of coding turned into two weeks of pain. It took me roughly five days to figure out what was going wrong. Then another week of trying to solve it, and when I reached the point of scrapping the whole thing, a light went off in my head. I had one of those moments of clarity where the solution seemed so clear. Now all I had to do was create a test and see if my bright idea would work. Or was it just a flash of light?

My pain and anguish is your reward. I'm going to walk through the problems with the DW gateway and provide some tools to overcome them. I tried to make the code as lean and mean as possible for performance reasons, and the principles can be applied to your DW gateway with minimal effort. For purposes of this article I'm going to assume that the reader has a general understanding of the DW gateway. Details on setting up a DW gateway and how it operates are available in the article entitled "DirectoryWatcher Event Gateway: Ditching the Scheduler" in the July '06 issue of CFDJ.

I was using DW to watch a directory for any new files uploaded via FTP. I needed to process each new file after it uploaded. The root cause of the scenario I experienced was, in my opinion, a flaw in the DW gateway that causes the gateway onAdd event to fire before the file is finished writing. fires A second problem happens when a file is locked for writing and usually occurs when a DW gateway is watching a network shared directory.

Both of these problems are fatal for the DW gateway, but it's possible to get around them and process the file. (Note: The failures I encountered had almost nothing to do with the size of the file being written; it was all about the timing. However, the bigger the file, the more apt we are to run into problems.)

A third issue is that it's hard to access debugging information from the DW gateway. It's not just an issue with the DW gateway but any gateway or other call that doesn't output to a browser. This issue is fixable too, but it takes a bit of effort on the part of the developer. The important thing to remember is that any piece of code can fail at any given time.

With the DW gateway there's no user to get an error message, so we need to send those messages to someplace useful.

Before we can try to fix these issues we have to understand their nature and how CF reacts to each one. Like I said, file size has almost nothing to do with the problem, it's all about timing: when the gateway fires and when the file stops writing. Unfortunately, the gateway doesn't have a built-in way to detect that a file write is complete.

Let's take a closer look at the first issue. Since the server starts writing the file to the drive as soon as it's received, the gateway can pick up the file as soon as a single byte is written. While this in itself isn't a problem, it can become a problem for the gateway if the gateway kicks off a process that needs the entire file. The gateway itself doesn't fail, but if we used CFFILE to read the file content we wouldn't get the whole file. Only that part of the file that was uploaded when you tried to read it would be returned. So we need a way to have our process continue only if the whole file has been uploaded.

The second issue is a file that's write-locked, which normally happens when writing a file to a shared directory or a UNC path using a Windows server. If we write a 5MB file to a shared directory Windows instantly creates an empty 5MB file as a placeholder. It then copies the file to the destination. During this operation there's an exclusive write-lock on the file that prevents anything from trying to read the file until the write is complete. Unlike the first issue, if we try to use CFFILE to read the file while it's locked, the gateway crashes. So we also need a way to avoid reading locked files.

While both these issues seem pretty extreme they're fairly easy to overcome. We can even couple the fixes for each scenario together for a more robust solution. When I first wrote the fixes they were independent but over time I've combined them and put them in every DW gateway I create. Let's take a look at them.

We can avoid reading a file that's still being written when the onAdd event fires by putting the CF thread to sleep and checking the file's size repeatedly until it stops changing. Of course this can tax server performance and I have experimented with other solutions that work, but it's the most straightforward approach.

Note: All the code examples in the article have been stripped to a minimum to save

space. The complete code is available online.

Here's a normal onAdd event inside a gateway cfc file:

```
<CFFUNCTION NAME="onAdd" ACCESS="public" RETURNTYPE="string">
<CFARGUMENT NAME="CFEvent" TYPE="struct" REQUIRED="yes">

<CFSET thisFile = cfevent.data.filename>
<CFFILE ACTION="READ" FILE="#thisFile#" VARIABLE="fileContent">

</CFFUNCTION>
```

At the third line we get the file name and path then on the fourth we read the file. If the file write wasn't complete, the file read would get a partial file.

Take a look at the code in Listing 1. There we attempt to overcome the issue by checking the size of the file. This is a delicate operation because we don't want to disturb the write of the file. To do this we'll leverage the java.io.FileInputStream. Let's walk through the code and examine what's going on.

First we create a Java object to read the file system:

```
fileRead = createObject("java","java.io.FileInputStream");
```

This will enable us to check the file size. This can be done with any of several Java file objects, or we could do it with CFDIRECTORY. However, this way lets us do a non-blocking read on the file. We can use it to get a byte count that can be read from the file without causing a block. We want to do our best to make sure that we don't disrupt anything that's happening to the file while it's still uploading.

Next we create a Java object that lets us manipulate the current thread. Now we can pause the code:

```
thisThread = CreateObject("java", "java.lang.Thread");
```

Initialize the Java file object with the file that triggered the onAdd event:

```
fileRead.init(thisFile);
```

Next we set a variable loopCT to count the loops and control a while loop that, if gone unchecked, could run infinitely. Then we get the current size of the file and pause the thread for a second:

```
sizeA = fileRead.available();
thisThread.sleep(1000);
```

After the one-second pause we check the file size again and do a comparison. If the file is still being written the sizes won't match. If the file sizes match, we do one more check, just in case there's an FTP pause or some other delay that caused a pause in

the file writing:

```
if (sizeA EQ sizeB){
thisThread.sleep(1000);
sizeC = fileRead.available();
if (sizeC EQ sizeB){
break;
}
}
```

If the file sizes still match then the file is finished writing. If the sizes don't match then we continue looping and start the process all over again. Keep an eye on the loop count. We don't want the loop to go on forever. If every pause was hit on every loop then this check would stop the loop after about two minutes of runtime. We would then have to put some code in that would handle this condition. In the example we just return:".

```
incrementValue(loopCT);
if (loopCT GT 60){
return ;
}
```

Lastly, we close the fileRead object. If this isn't done we could end up with a lock on the file that could only be released by stopping and starting ColdFusion.

```
fileRead.close();
```

After all this is done we can process the file as the application requires. In this example, I've removed the error checking from the code so we can focus on the file size check and not clutter the code.

The example above handles a file that we can read even though it's still in a write state. The other scenario is a little trickier to get around. There we were trying to get around a file being locked when we try to read it. Even though java.io.FileInputStream does a non-blocking read it still can't get around this hurdle. The only thing to do is to attempt to read the file and trap the read error. It's not very elegant but it gets the job done. The code in Listing 2 works to overcome the lock issue. There are some advanced Java objects you can use to check for locks. However, they do just about the same thing. They try to put a lock on a file to see if it worked. It works on the basis that only one process can have a write-lock on a file.

The flow of this code is just about the same as the code from Listing 1. The difference is that we put a try/catch around the fileRead.init(). This lets us try to read the file without causing the code to fail. If the file is locked when we try to read it, we fail to catch and sleep the thread. The code continues to loop until it can read the file or the loop count hits 60.

The code in Listing 3 is the best of both worlds. We combine the file lock check with the file size check. My suggestion would

“The DW gateway is a very powerful tool, if a bit unrestrained; fortunately, we can create code to tame DW’s uncivilized tendencies”

be to always do it this way. You can never be too safe when it comes to running a DW gateway.

Now that we know how to check for file-write completion and overcome file blocking, we can start to get creative. We can create a process that can handle multiple files as one batch. This is done by using CFDIRECTORY to check file counts in the directory. We can then cause the current DW gateway event to quit if the directory count increases. The next gateway event will pick up the new file. Each gateway event does the same check. When the file counts no longer increase then the last gateway run processes all the files in the directory. The code and detailed explanation is too lengthy for this article, but the full code, with documentation, will be available for download.

If we continuously process large files there are a few considerations we need to take into account. The first is to make sure that the server has the processing power and RAM to run the DW gateway. In my experience it takes more power to run them than a normal server. Because of the methods used to overcome the write completion and file locks, we could end up with a lot of sleeping/running threads at one time. You can't control the number of running DW threads...or can you? This is a topic for another time but suffice it to say, it is possible. (I can hear the e-mails coming now...)

I've said a few times already that error handling is very important in a DW gateway. This can't be overemphasized, and finding out what went wrong is also important. While preventing errors is straightforward, doing something with the error isn't. First, we'll focus on preventing errors then we can focus on displaying them.

For example, in Listing 2, I used a try/catch to trap an error that may occur. Now, going one step further I can put the whole run in a try/catch. Using try/catch is probably the best thing to do to prevent a gateway from failing. The other side of the coin is to figure out what to do when something does go wrong. This is really up to the specific occurrence of the gateway.

One thing that I've done is set up an onError event in my application.cfm for the gateway CFCs. This lets me e-mail any errors that go untrapped to myself or others. I also write the error to an error log specific to the gateway that was running. Normally we can do a CFDUMP in CF code to see what's happening. With a gateway this isn't possible. I solve this by setting up a debug function to get information out while I am working on the CFC. This takes an input and dumps it to a file that I can call from a Web browser. This is great for trying to find out what's happening inside the CFC.

We must also keep an eye on the file space used for a DW gateway. If there are failures, the failed file must be cleaned up, otherwise we could end up using lots of disk space. I handled this by creating a scheduled process which deletes files that are a day old. Your situation may call for some variation on this theme.


There are some other tricks we can do to add more robust functionality to the DW gateway.

We can have a file exclusion list based on the file extension. If the file's extension is on the list, we can delete it or handle it some other way. However, if we're going to delete it we have to make sure it's in a state that allows deletion. We could just add the code from Listing 4 to the end of the other code examples to do this.

We can also build a function to attempt to correct file naming issues, for example a file that starts with a period. This is common if a user on a Mac uploads files. In this situation we'd want to delete the file. We can also make sure the file has a valid extension and that the file name doesn't contain any invalid characters.

The DW gateway is a very powerful tool, if a bit unrestrained. With it you can do a multitude of things, both good and bad. We've seen how it lets us kick off processes that may not be able to handle the very files that are monitored by the gateway. Fortunately, we can create code that can tame DW's uncivilized tendencies and reliably manage transferred files coming from both FTP and network transfers. Hopefully this article gave you some tools to build a better DW gateway and make you a hero in your shop, too.

...

Thanks for the great article, Dave. I'm always glad to hear from folks "in the trenches." For you readers, if you have a war story you'd like to see printed in the pages of the Journal, drop me an e-mail at jeff@grokfusebox.com. 

About the Authors

Dave Ferguson is a system architect and principal programmer. He has been doing website design and development for over 10 years. He is also a Certified Advanced ColdFusion Developer. You can read his blog at <http://dfoncf.blogspot.com>

Jeff Peters is a program manager and application architect for Operational Technologies Services in Vienna, Virginia. His ColdFusion-related books are available at www.protonarts.com; e-mail to jeff@grokfusebox.com.

dave@dkferguson.com

jeff@grokfusebox.com

Listing 1: Checking file upload completion

```
<CFFUNCTION NAME="onAdd" ACCESS="public" RETURNTYPE="string">
<CFARGUMENT NAME="CFEvent" TYPE="struct" REQUIRED="yes">
<CFSCRIPT>
thisFile = cfevent.data.filename;
fileRead = createObject("java","java.io.FileInputStream");
thisThread = CreateObject("java","java.lang.Thread");
fileRead.init(thisFile);
loopCT = 1;
while (1 EQ 1){
sizeA = fileRead.available();
thisThread.sleep(1000);
sizeB = fileRead.available();
if (sizeA EQ sizeB){
thisThread.sleep(1000);
sizeC = fileRead.available();
if (sizeC EQ sizeB){
break;
```



```

}
}
incrementValue(loopCT);
if (loopCT GT 60){
fileRead.close();
return ;
}
}
fileRead.close();
</CFSCRIPT>
<CFFILE ACTION="READ" FILE="#thisFile#" VARIABLE="fileContent">
<CFRETURN >
</CFFUNCTION>

```

Listing 2: Checking for file lock

```

<CFFUNCTION NAME="onAdd" ACCESS="public" RETURNTYPE="string">
<CFARGUMENT NAME="CFEvent" TYPE="struct" REQUIRED="yes">
<CFSCRIPT>
thisFile = cfevent.data.filename;
fileRead = createObject("java","java.io.FileInputStream");
thisThread = CreateObject("java", "java.lang.Thread");
loopCT = 1;
while (1 EQ 1){
try {
fileRead.init(thisFile);
break;
} catch(any ecpt){
thisThread.sleep(1000);
}
incrementValue(loopCT);
if (loopCT GT 60){
fileRead.close();
return;
}
}
fileRead.close();
</CFSCRIPT>
<CFRETURN >
</CFFUNCTION>

```

Listing 3: Checking for file lock and upload completion.

```

<CFFUNCTION NAME="onAdd" ACCESS="public" RETURNTYPE="string">
<CFARGUMENT NAME="CFEvent" TYPE="struct" REQUIRED="yes">
<CFSCRIPT>
thisFile = cfevent.data.filename;
fileRead = createObject("java","java.io.FileInputStream");
thisThread = CreateObject("java", "java.lang.Thread");

loopCT = 1;
while (1 EQ 1){
try {

```

```

fileRead.init(thisFile);
break;
} catch(any ecpt){
thisThread.sleep(1000);
}
incrementValue(loopCT);
if (loopCT GT 60){
fileRead.close();
return;
}
}

loopCT = 1;
while (1 EQ 1){
sizeA = fileRead.available();
thisThread.sleep(1000);
sizeB = fileRead.available();
if (sizeA EQ sizeB){
thisThread.sleep(1000);
sizeC = fileRead.available();
if (sizeC EQ sizeB){
break;
}
}
incrementValue(loopCT);

if (loopCT GT 60){
fileRead.close();
return;
}
}
fileRead.close();
</CFSCRIPT>
<CFRETURN >
</CFFUNCTION>

```

Listing 4: Using an Exclusion list

```

<CFSCRIPT>
exclusionList = "bat,dll,exe,zip";
theFileName = listLast(CFEvent.data.filename, "\\"); if
(left(theFileName, 1) EQ "."){ //handle file delete }

theFileExtension = ListLast(theFileName, "."); if (listFindNocase(exc
lusionList, theFileExtension)){ // handle excluded file } </CFSCRIPT>

```

Download the Code...
Go to <http://coldfusion.sys-con.com>

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.



ONLY
\$69⁹⁹
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

ColdFusion and the Rise of Right-Brained Thinking

It's time to rock the house



By Hal Helms

Recently, I've been reading a book recommended to me by my friend, Clark Valberg. The

book is *A Whole New Mind*; its author is

Daniel Pink. In this article, I'll discuss why

I think the premise of the book holds such

promise for ColdFusion programmers – and how it challenges

us to rethink how we define what our work is.

Pink begins by recounting what we know of brain chemistry: the left brain is the processor of logic and language; the right brain finds meaning, beauty, and context. One is sequential and discrete; the other grasps truths in whole and synthesizes parts into a whole.

Pink's opening argument is that three forces have conjoined to impeach the dominance of left-brained thinking – the very thing that has given programmers such prominence in the “information economy.” Those forces are abundance, Asia, and automation.

The pre-eminence of logical, left-brained thinking has produced wealth on a scale that would have been unthinkable several decades ago. This abundance has met the physical needs of most of those fortunate enough to live in so-called “first-world” countries. But it has done nothing to speak of for the deeper needs, aspirations, and hopes of those thus blessed. This deficit has fueled the growth in movements that span the philosophical/spiritual spectrum from New Age to fundamentalism.

Meanwhile, the source of much of this abundance, the triumph of left-brained thinking, is rapidly moving to countries long thought of as “third world” – notably India, the Philippines, and China. The motivation for this shift? Money. The same, basic left-brained thinking-related work previously done by first-world workers can now be performed by similarly educated and trained third-world workers, but at a fraction of the cost. This trend, Pink asserts, will only continue. “Outsourcing” will become not so much a novel phenomenon as a new paradigm. Finally, automation – primarily computer automation – threatens to displace many whose knowledge work can be turned into computer code. And it turns out that a surprisingly large amount of work we previously thought of as the preserve of humans can be automated. Perhaps no event was so seminal as the 1997 match between Grand Master Gary Kasparov and the IBM supercomputer, Deep Blue. The man who for over a decade

had not lost a match was beaten by a collection of semi-conductors that ran a computer program.

Pink argues that left-brained thinking, of the kind practiced by us computer programmers, is not so much antiquated as it is insufficient. It can produce wealth, but not fulfillment. It delivers value but not meaning. It provides, but does not delight.

In this new age, adequacy is inadequate. Our customers are rapidly becoming inured to applications that fail to anticipate their needs. We're all very familiar with such applications; the Web is filled with them: forms that are clumsy to use, links that fail to enlighten us, information that does not sufficiently inform us to make decisions.

In the rapidly passing “Information Age,” it was enough to dump data on users. That is no longer the case. Users want interactivity; they want their applications to show that the application's designers took care to make their communication with machines as personal as possible. In an age where so much customer interaction is relegated to machines, they want something human.

Pink is quick to point out that this new era will be both a bane and a blessing. People who have learned to use more than half their brain are at a great advantage while people who insist on employing only a portion thereof will find themselves on very rough waters. My argument in this article is that this dichotomy holds true not only for people, but for computer languages.

Traditional computer languages were built by engineers to get a job done. But what job? The virtues long espoused by computer engineers are speed, power, and robustness. So little thought was given to human interaction with computers that the term “lipstick” was used to derogatorily refer to the user interface. Traditional computer languages reflect this bias: try creating a beautiful interface with, say, Java – you're in for quite a challenge.

If Pink is right, those virtues are all necessary, but by no means sufficient. What we need are languages that facilitate the creative part of ourselves – the right brain. Past a certain threshold, the old-line virtues rapidly reach a point of diminishing returns. We need languages that offer access to those who can use both sides of their brains. If that sounds to you like a good description of ColdFusion, well, I'd agree.

After the debacle of the “dot-bomb” implosion, a new mentality has emerged. Though it has been given the perhaps too-affected name of “Web 2.0,” there is an underlying reality that blends nicely with Pink's ideas, as much of what this new perception embraces is the idea of user interfaces that not only inform but inspire.

People like Jason Fried, known for his company 37 Signals and the software it produces (Basecamp, most noticeably), said in a recent interview: “We do things a little different; we start with the [user] interface first.”

When the user interface – that part of the application where computer code directly interacts with a real user – is done last, as is the case with much traditional development, it reflects its lowered status as an afterthought. But by using that much-neglected hemisphere of our cerebrum to create the user interface first, we can move beyond the merely sufficient into this new world of design and meaning.


If all this sounds a bit too foo-foo, consider the not so ephemeral success Apple has had with their iPod. By some estimates, Apple controls 80% of the market for mobile music devices, a number that would warm the heart of any predatory monopolist.

What do people love about the iPod? Its design. By understanding that consumers are looking beyond the essential to the elegant, Apple's whole-brained designers have created something that transcends mere success: the iPod is a cultural phenomenon. The actual functioning of the iPod is somewhat less successful: there have been numerous complaints of cases scratching, iPods freezing – and then there's the whole aspect of the execrable iTunes software and Apple's none-too-generous digital rights management. But those have been overlooked by consumers who wanted something more than a functional music player.

ColdFusion, I believe, has a large role to play as we move beyond the merely denotative to the connotative. The world is catching up with the original vision of the Allaire brothers. The time is ripe for us to illustrate how truly powerful, beyond baseline issues of speed and

security, is the language. For that to happen, though, we need a new attitude toward application development, just as Apple's new thinking produced the iPod. This can be difficult when the current is moving so strongly in the opposite direction. It takes courage and vision to leave the exclusively left-brained habitués to their dry applications while we pursue meaning.

To those “who have ears to hears,” however, the opportunity is unbounded. Computer programmers have for too long adhered to the Hippocratic Oath: “Above all, do no harm.” That worked in the past; now is the time to do some positive good, to offer users applications that are built around the way they work rather than forcing them into a Procrustean bed imposed by the limitations of purely left-brained computer languages.

ColdFusion programmers of the world, unite: it's time to rock the house. 

About the Author

Hal Helms is a well-known speaker/writer/strategist on software development issues. His monthly column in CFDJ contains his Musings on Software Development and he has written and contributed to several books. Hal holds training sessions on Java, ColdFusion, and software development processes. He authors a popular monthly newsletter series. For more information, contact him at hal@halhelms.com or see his website, www.halhelms.com.

Get more with Hot Banana!

Web Content Management + Active Marketing



Hot Banana software builds efficient, affordable, fast, very easy-to-use and measurable Web sites. And they're ColdFusion-based and optimized for marketing performance. That's what makes us different and better!

Visit us today at:
hotbanana.com/webinar
or call 1-866-296-1803
to schedule a
live 1-on-1 demo.



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

Java for ColdFusion Programmers?



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

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

Why Interfaces in ColdFusion Are Irrelevant

An overview



By Brandon Harper

One of the many hotly contested items about ColdFusion in recent times that some developers have been clamoring for are the addition of interfaces to ColdFusion Components as part of the standard CFML

language. At one time I was definitely in this camp myself – interfaces would indeed be a good way to make sure that components follow a given contract.

However, after much thought about the underlying implementation of them at a compile time and runtime level, I find them to be only a semantic difference to what you can already do with CFCs. While something akin to a `<cfinterface>` tag would make the implementation of a pseudo interface a slightly cleaner affair, the underlying implementation of interfaces in a dynamically typed language such as ColdFusion does not add much to the existing capabilities. In particular, the main purpose of an interface is to ensure that an object follows a contract, and this is technically infeasible in a dynamically typed language like ColdFusion.

A reasonable high-level comparison of dynamic languages to static languages could be summed up as a pair of safety scissors versus a razor blade. Static languages are the safety scissors – they help protect you from cutting yourself and will probably last longer, but it might take a long time to cut something. Dynamic languages are a razor blade – they are very sharp and nimble and cut through most things very quickly, but they are easy to cut yourself with if you're not careful and may have a shorter shelf-life depending on how they are used. This comparison is not valid for all languages, but works well for comparing a dynamically typed language such as ColdFusion and Python to a statically typed language like Java.

ColdFusion is dynamically typed because you don't have to

specify the primitive data type of variables when they are created, as the CFML runtime examines a given variable and takes care of that for you. In general, this allows you, the developer, to concentrate on getting things done rather than managing each variable you create. Statically typed languages such as Java typically require the developer to declare the data type for all variables, and provide compile-time checking to make sure all variables and method signatures are typed correctly. While it is generally more time-consuming to program in statically typed languages, it does provide more type safety and at a micro level has greater performance. One of the perfor-

CFDJ Advertiser Index			
ADVERTISER	URL	PHONE	PAGE
Adobe	www.adobe.com/products/coldfusion/flex		2
CFDynamics	www.cfdynamics.com	866-233-9626	3
EdgeWebHosting	http://edgewebhosting.net		4
CommunityMX	www.communitymx.com		6
HostMySite	www.hostmysite.com		11
Integral	www.fusion-reactor.com		13
Vitalstream	www.vitalstream.com		14,15
Paperthin	www.paperthin.com	800-940-3087	17
Omniure	www.omniure.com/actionsource	877-722-7088	21
Stream57	www.stream57.com	212-909-2550 x101225	
SYS-CON Media	www.SYS-CON.com	201-802-3000	27
Cynergy	www.cynergysystems.com		30,31
CFDJ	http://coldfusion.sys-con.com	1-800-303-5282	37
SYS-CON Reprints	www.sys-con.com	201-802-3024	38
AJAXWorld	www.AjaxWorldExpo.com	201-802-3020	39
JDJ	www.jdj.sys-con.com	1-888-303-5282	45
Hot Banana	www.hotbanana.com/cfdj-demo	1-866-296-1803	47
Halhelms	www.halhelms.com		47
AJAXWorld Uni Boot	www.AJAXBootcamp.sys-con.com		49
Teratech	www.teratech.com		51
HostMySite	www.hostmysite.com		52

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

AJAX ONE-DAY HANDS-ON INTENSE TRAINING!

ALL ATTENDEES RECEIVE:
Certificate of Completion **PLUS** Course Materials on DVD!*

AJAXWorld University's "AJAX Developer Bootcamp" is an intensive, one-day hands-on training program that will teach Web developers and designers how to build high-quality AJAX applications from beginning to end. The AJAX Developer Bootcamp is intended to be the premier AJAX instructional program presently available anywhere.

AJAXWORLD UNIVERSITY BOOTCAMP

www.AJAXBOOTCAMP.sys-con.com

Roosevelt Hotel
New York, NY
January 22, 2007

Roosevelt Hotel
New York, NY
March 18, 2007

What You Will Learn...

Overview of AJAX Technologies

- HTML vs. DHTML
- Network Concerns
- Asynchronous Conversations with Web servers
- The characteristics of high-quality AJAX applications
 - The Web page is the application
 - What the server provides
 - User interaction

Understanding AJAX through the basics of AJAX

- Asynchronous server communication
- Dynamic HTML
- Javascript Design patterns
- User interface strategies for building elegant, highly addictive Web sites and applications
- The Essential AJAX Pieces
 - Javascript
 - Cascading Style Sheet (CSS)
 - Document Object Model (DOM)
 - XMLHttpRequestObject
- The AJAX Application with Javascript
- Using CSS
- Structuring the View Using the DOM
 - Applying Styles with Javascript
 - Communicating with the Web Server in the Background
 - Designing AJAX Applications
 - Design Patterns
- Introduction to AJAX Frameworks
 - Dojo, script.aculo.us, Prototype
 - Overview of framework capabilities
- Examples of frameworks in use
- Best Practices

Hand-On Development The Fundamentals:

Building the Framing for an Ajax Application

- Review the courseware code with the Instructor
- Begin building a working AJAX application and start applying technique and technologies as introduced in class
- Create the basic AJAX application by creating HTML, Javascript, and CSS files
- Learn Best Practices and Validation
- Learn and add script.aculo.us effects
- Learn and add the Dojo Framework

Adding Basic Ajax Capabilities to a Web Page: Going Deep Into the AJAX User Experience

- Elements on the Rich Internet Experience
 - Interactivity
 - Robustness
 - Simplicity
 - Recognizable Metaphors
 - Preservation of the Browser Model Bookmarks/Back Button
- Background operations
- Building a AJAX Notification Framework
- Provenance and Relevance
- Rich Experience Support with Third-Party AJAX Client - Framework
- Using AJAX layouts, containers, and widgets
- Patterns for Animation and Highlighting
- User Productivity Techniques
- Tracking Outstanding Network Requests

Hands-On Development:

Expand the Application with more Advanced Ajax

- Review the courseware code with the Instructor
- Expand the Mural Application
- Add Features using Dojo
- Add specific Dojo Libraries to support Ajax widgets

Advanced AJAX Concepts

- Review Ajax Concepts
- SOA and Mashups
- Current state of Ajax Frameworks
- Web 2.0 and the Global SOA
- Ajax Constraints
- Design Patterns
- Javascript Timers
- Ajax Programming Patterns
- Performance and Throttling

Hands-On Development:

Working with Advanced Ajax Capabilities

- Review the courseware code with the Instructor
- Work with the Accordion control
- Learn how to use the Tree control
- Explore Dojo's animation capabilities
- Explore how the debug output can be used in <div> elements
- Tour Dojo and RICO demos
- Experiment with new Dojo features from the Dojo demos source code and attempt to add them to various parts of the Mural application
- Overview of Future of AJAX and Rich Internet Applications

What Attendees Are Saying...

**“ The trainer was excellent.
The material too!**

**“ The hands-on, although long,
was useful and educational!**

**“ The instructor was good. He
answered questions thoroughly!**

**“ Well designed and organized.
Good mix of lecture vs lots of
hands-on!**

*ALL RELEVANT COURSE MATERIALS WILL BE OFFERED ON DVD AFTER THE EVENT

HURRY! REGISTER NOW FOR EARLY-BIRD DISCOUNT!



For more great events visit www.EVENTS.SYS-CON.com

mance advantages would be that because the runtime can already be assured of which data type each variable contains, fewer CPU cycles are used introspecting data and turning it into code that a CPU can understand. There are plenty of other advantages and disadvantages between the two types of languages as well that are beyond the scope of this article.

When I find a need for an interface in ColdFusion, I essentially create my own pseudo interfaces that at least provide runtime checking in case an inheriting object's method is called. It's not necessarily the best solution, but it does at least offer some sort of contract that an inheriting object can be tied to, mostly for documentation purposes. I've seen several other people do it as well, but it looks something like this:

foobar/Foo.cfc

```
<cfcomponent hint="Interface for foo">
    <cffunction name="init" returntype="any">
        <return this />
    </cffunction>

    <cffunction name="implementThis" returntype="any">
        <cfthrow message="foobar.Foo: implementThis() must be
implemented">
    </cffunction>
</cfcomponent>
```

foobar/Bar.cfc

```
<cfcomponent hint="Provides foobar functionality">
    <cffunction name="init" returntype="any">
        <return this />
    </cffunction>

    <cffunction name="implementThis" returntype="any">
        <cfreturn true />
    </cffunction>
</cfcomponent>
```

These are just some quick pseudo-code examples coded for brevity, but you get the idea.

As I started to question the usefulness of interfaces in ColdFusion in light of the way a lot of people are already doing them, I began to realize that the implementation of an interface tag or attribute to CFCs would not really offer any technical advantages


at a runtime level, so largely their implementation is only a semantic difference.

In Java, one of the main advantages to using an interface is that it does compile-time checking to make sure that classes implementing interfaces follow the contract dictated by the interface as well as the full method signature of required arguments. As ColdFusion is a dynamically typed language, there really isn't a good way to do compile-time checking of interfaces. It is also impossible to do a full method signature of an object at compile time because of the nature of dynamically typing – the full signature of an object can change at runtime because of the flexibility inherent in dynamically typed languages.

One of the reasons I decided to write about this is that it's a subject that has been debated heavily in the community as a needed feature for the upcoming ColdFusion 8, currently code-named Scorpio. BlueDragon 7, currently in beta, does include an implementation of interfaces. I wrote a shortened preview version of this article on my blog, which drew some valuable dialog, in particular from Vince Bonfanti, the CEO of New Atlanta.

Vince discussed the nature in which interfaces are implemented in BlueDragon, which I think are technically sound and the way I would personally attempt to add interfaces into a dynamically typed language. However, any sort of interface implementation in a dynamic language such as ColdFusion are only semantics because you can't check the full method signature of an object until runtime, rather than at compile time as you can with static languages. While I do concur that the syntax is cleaner than in the example I provided earlier, I disagree in principle that it's a true interface. A counter-example I provided is how many developers using the open language Python have proposed interfaces over the years, but it's never been implemented for the reasons I've cited as well as due to disagreements in the syntax of its proposed implementation, among other reasons.

Summary

Due to the nature of dynamically typed languages, the implementation of interfaces in ColdFusion would not prevent someone from breaking the contract of an interface at runtime. Any implementation of interfaces in the CFML language would essentially only represent a difference in semantics to other ways already available to implement interface-like objects without providing the safety that interfaces should provide. There are bigger problems to solve and better features to implement in the next version of ColdFusion than interfaces. 

About the Author

Brandon Harper has been programming in ColdFusion since 1998 and also actively writes applications in Python and Java. He is currently a senior software developer at Acxiom where he works on an enterprise service platform that powers their risk mitigation products. Brandon was also a technical editor for Inside ColdFusion MX, and maintains a blog at devnullled.com.

E-mail: brandonh@gmail.com



www.frameworksconference.com

FRAMEWORKS 2007 CONFERENCE

Washington D.C.

February 1st & 2nd

NEW TOPICS GREAT SPEAKERS NETWORK LATEST TECHNOLOGIES

The Frameworks 2007 conference will be located in the Washington DC area, on February 2007. We will have lots of great speakers like last year. This year in addition to Fusebox and FLiP we will have sessions on other frameworks such as Mach-ii, Model Glue, Ruby on Rails, ColdSpring and Struts, and other methodologies such as XP and test-driven development.

We will be looking at frameworks for several web-based languages including ColdFusion, Java and .Net. There will be sessions for beginning and advanced developers, with lots of opportunities for learning from "foreign frameworks" and cross-pollination.

WHY SHOULD YOU COME TO THE FRAMEWORKS CONFERENCE?



"Frameworks is FOR developers, put on BY developers. So the information is useful and there is a ton of it... Are you into "Networking", how would you like to talk to the people who created Fusebox, FLiP, or Fusedoc's? Would you like to meet leaders in our industry using these tools? Well, you can, and that is why you should come."

Eric R. L.

"I'm coming to the conference because our core applications are far behind the Fusebox times—we're running on Fusebox version 2! So I'm interested in learning about other frameworks that are out there, as well as ways of migrating our existing apps smoothly."

Troy B.

"To stay current with the latest fusebox and frameworks conference."

Anne S.



TeraTech
Programming
www.teratech.com

Other companies in this magazine spent a lot of time on pretty ads. As you can see, we did not. We spent our time hiring the best people and training them to deliver outstanding support for your website. We spent our time building a state of the art datacenter and staffing it with people who care about your website like it's their own. Compassion, respect, credibility, ownership, reliability, "never say no," and exceed expectations are words that describe our service philosophy. From the first time you interact with us, you'll see what a difference it really makes. And you'll also forgive us for not having a pretty ad.

