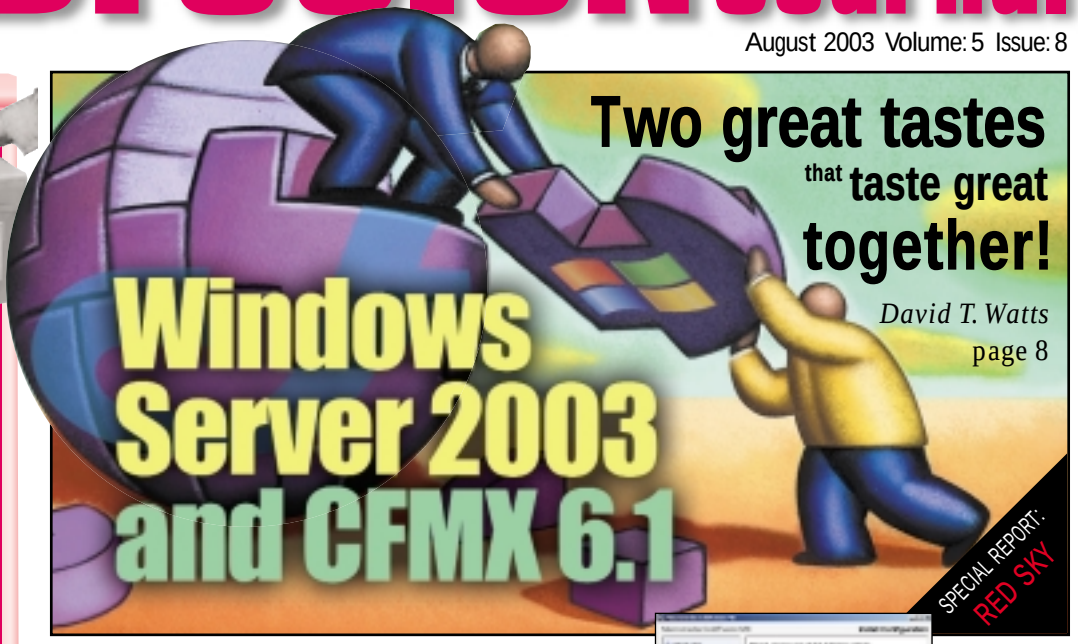
U.S. $8.99 (Canada $9.99)

ColdFusion Developer's Journal

ColdFusionJournal.com

August 2003 Volume: 5 Issue: 8



RETAILERS PLEASE DISPLAY UNTIL OCTOBER 31, 2003
$8.99US $9.99CAN
09
0 71486 02689 1
SYS-CON MEDIA

ColdFusion Developer's Journal

editorial advisory board

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

editorial

editor-in-chief
Robert Diamond robert@sys-con.com
technical editors
Charlie Arehart charlie@sys-con.com
Raymond Camden raymond@sys-con.com
editorial director
Jeremy Geelan jeremy@sys-con.com
executive editor
Jamie Matusow jamie@sys-con.com
editor
Nancy Valentine nancy@sys-con.com
associate editors
Gail Schultz gail@sys-con.com
Jean Cassidy jean@sys-con.com
assistant editor
Jennifer Van Winckel jennifer@sys-con.com

production

production consultant
Jim Morgan jim@sys-con.com
art director
Alex Botero alex@sys-con.com
associate art directors
Louis F. Cuffari louis@sys-con.com
Richard Silverberg richards@sys-con.com
Tami Beatty tami@sys-con.com

contributors to this issue

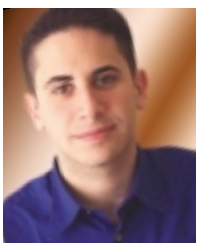
Charlie Arehart, Steve Drucker, Drew Falkman, Ben Forta, Hal Helms, Simon Horwith, Aaron Johnson, Dominic Plouffe, Dave Watts

editorial offices

SYS-CON MEDIA
135 CHESTNUT RIDGE RD., MONTVALE, NJ 07645
TELEPHONE: 201 802-3000 **FAX:** 201 782-9600
COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)
is published monthly (12 times a year)
by **SYS-CON Publications, Inc.**
postmaster: send address changes to:
COLDFUSION DEVELOPER'S JOURNAL
SYS-CON MEDIA
135 Chestnut Ridge Rd., Montvale, NJ 07645

What Did I Do Before Red Sky?

Many folks on the ***CFDJ*** editorial team (myself included) have been observing and participating in the beta process for Red Sky since day one, and now the release is finally here. I'm proud to present what we've put together, along with Macromedia's help, to cover the first batch of new highlights in the product as it's being released.



By Robert Diamond

Ben Forta and Dave Watts have written articles on Red Sky that are well worth reading. I've been itching to write about many of these features for weeks, as has our editorial team, and with the August 5 release now behind us, we have the official go ahead to do so. Also, I can report that at SYS-CON we're eating in our own kitchen and that *CFDJ*'s Web site, along with most of SYS-CON.com, is now running off of Red Sky without a hitch. The release, by the way, is officially labeled as ColdFusion 6.1 for those keeping numerical count.

I've been running Red Sky on both my development machine and personal server for a few months now, and I can say unequivocally and without reservation that it's well worth the upgrade. Since it's free, upgrading is generally a no-brainer to most folks but it's still worth encouraging. In fact, if you're running CFMX, stop reading this column and start downloading it…right now. After a quick registration on Macromedia.com, if you're not already set up there, you can hit the download button and then come back and finish reading this editorial. By the time you get to "Till next month, happy coding" it should be ready to go. On the flip side, if you've been holding back on upgrading to MX, and are still on 4 or 5 waiting for the always expected .1 release in a product's life cycle, it's now here, so start testing your apps.

At the *ColdFusion Developer's Journal* Keynote Panel at CFUN 2003 in Maryland this past June, the panel asked what everyone's favorite new feature of Red Sky was. Back then much of the product was under NDA though we still managed to get out a series of good and differing answers from the participants without raising any legal red flags (no pun intended.) My personal favorites, both then and now, are most notably the speed improvements – especially when files are compiled for the first time. I was doing some development work recently for a friend on regular MX, which I haven't used in months, and the difference was like night and day. That to me is in and of itself a worthy upgrade.

We wanted to get one other perspective on Red Sky, and that was from Mr. ColdFusion himself, Jeremy Allaire, who, while busy with his new position as technologist in residence at General Catalyst Partners, is still keeping up with the world of ColdFusion. When asked to comment on Red Sky he said, "The Red Sky release of ColdFusion MX demonstrates Macromedia's commitment to listening to customers by addressing longstanding feature requests such as mail and HTTP protocol enhancements, to just making development and deployment faster. The updated Apache Axis release combined with some minor yet very significant enhancements to CFCs will ensure that CFMX remains the easiest way to deliver Web services–based applications." You can't describe it much better than that!

Till next month, happy coding!

About the Author

Robert Diamond is vice president of information systems for SYS-CON Media, and editor-in-chief of **ColdFusion Developer's Journal.** *Named one of the "Top thirty magazine industry executives under the age of 30" in* Folio *magazine's November 2000 issue, Robert holds a BS degree in information management and technology from the School of Information Studies at Syracuse University. Visit his blog at* www.robertdiamond.com.

robert@sys-con.com

SYS-CON MEDIA

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

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

advertising
senior vp, sales & marketing
Carmen Gonzalez carmen@sys-con.com
vp, sales & marketing
Miles Silverman miles@sys-con.com
advertising director
Robyn Forma robyn@sys-con.com
director of sales & marketing
Megan Mussa megan@sys-con.com
advertising sales manager
Alisa Catalano alisa@sys-con.com
associate sales managers
Carrie Gebert carrieg@sys-con.com
Kristin Kuhnle kristin@sys-con.com

sys-con events
president, events
Grisha Davida grisha@sys-con.com
conference manager
Michael Lynch mike@sys-con.com

customer relations
circulation service coordinators
Niki Panagopoulos niki@sys-con.com
Shelia Dickerson shelia@sys-con.com
Edna Earle Russell edna@sys-con.com
manager, jdj store
Rachel McGouran rachel@sys-con.com

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

web designers
Stephen Kilmurray stephen@sys-con.com
Christopher Croce chris@sys-con.com

online editor
Lin Goetz lin@sys-con.com

accounting
financial analyst
Joan LaRose joan@sys-con.com
accounts receivable
Kerri Von Achen kerri@sys-con.com
accounts payable
Betty White betty@sys-con.com

subscriptions
Subscribe@sys-con.com
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

Tales from the List

How clever is your SQL?

Because of the time difference between my new location in the UK and the vast majority of the ***CFDJ*** subscribers in the U.S., one of the first things I do every morning is read over all of the threads that transpired on the ***CFDJ*** List the night before. Not only does it help me to stay aware of any pressing issues that the List might have, but it also allows me to begin the day with some thought-provoking material. Recently, I had one of these thought-provoking mornings, which I thought I'd share with you this month.

By Simon Horwith

The thread began with a post from Danna Swain, who was having trouble getting back the result set she wanted from her database. Danna has two tables that store quantitative amounts (rain and discharge volumes). Every entry is also associated with a year. What Danna needs is to retrieve the total quantity grouped by year, from both of the tables in a single recordset. So how can Danna tell the database to return a recordset with each row looking something like "raincentimeters, dischargecentimeters, year"?

Remember that raincentimeters is stored in one table, discharge centimeters in another, both have a year column, and there are likely to be many rows for each year in each table. It might sound simple, but try drawing the tables on paper and write the SQL statement to return this information… it's not as easy as you might have thought. As a hint, Danna confirmed that she is able to retrieve exactly what she wants from one table with the following SQL: "SELECT Rain.year, Sum(Rain.raincm) AS SumOfraincm FROM Rain GROUP BY Rain.year".

I-Lin Kuo immediately suggested that it would be very easy to retrieve two queries and join them in CFML, adding that in Oracle 9 you could also join them together before returning them to CF. Several other posts offered suggestions – none of which appeared to work quite the way they needed to. Two of these posts were extremely vague but after follow-up posts, did lead to the solution.

One post, from Amit Talwar, suggested that a UNION query should work. He suggested using a UNION within a subquery and then running calculations on that. John Hatcher added that there should be a full outer join somewhere in Amit's proposal in order to ensure that all of the data would display if a year happened to exist in one table but not the other. Still, how does this look in code? I-Lin Kuo ran some tests and reported back that it did not appear to work quite right, but that he thought they definitely were on the right track.

To better clarify his post, Amit offered the actual solution code he was suggesting, which as it turns out works very well and is very clever indeed. The SQL offered by Amit is as follows:

```
select sum(totalrain) as train,sum(totaldischarge) as
tdischarge , year
From (select sum(raincm) as totalrain , 0 as totald-
ischarge ,year from t1 group by year union all select
0  as totalrain ,sum(dcm) as totaldischarge ,year
from t2  group by year) group by year
```

This SQL creates one recordset from which to select another that is returned. This nested SQL statement

```
select sum(raincm) as totalrain , 0 as totaldischarge ,year
from t1 group by year union all select 0  as totalrain
,sum(dcm) as totaldischarge ,year from t2  group by year
```

—continued on page 32

About the Author

Simon Horwith is chief technology officer of eTRILOGY Ltd., a software development company based in London, England. Simon has been using ColdFusion since version 1.5 and is a member of Team Macromedia. He is a Macromedia Certified Advanced ColdFusion and Flash developer and is a Macromedia Certified instructor. In addition to administering the CFDJ-List mail-list and presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers.
simon@horwith.com

Windows Server 2003 and ColdFusion MX 6.1

TWO GREAT TASTES THAT TASTE GREAT TOGETHER!

With the release of ColdFusion MX 6.1 (Red Sky), ColdFusion will now run on the latest version of the Windows operating system. This article takes a brief look at the new features in Windows Server 2003 that may be of interest to CF developers and administrators. We'll also see how CFMX 6.1 integrates with Internet Information Services 6.0, the Windows Server 2003 version of the Microsoft Web server.

By David T. Watts

What's New in Windows Server 2003?

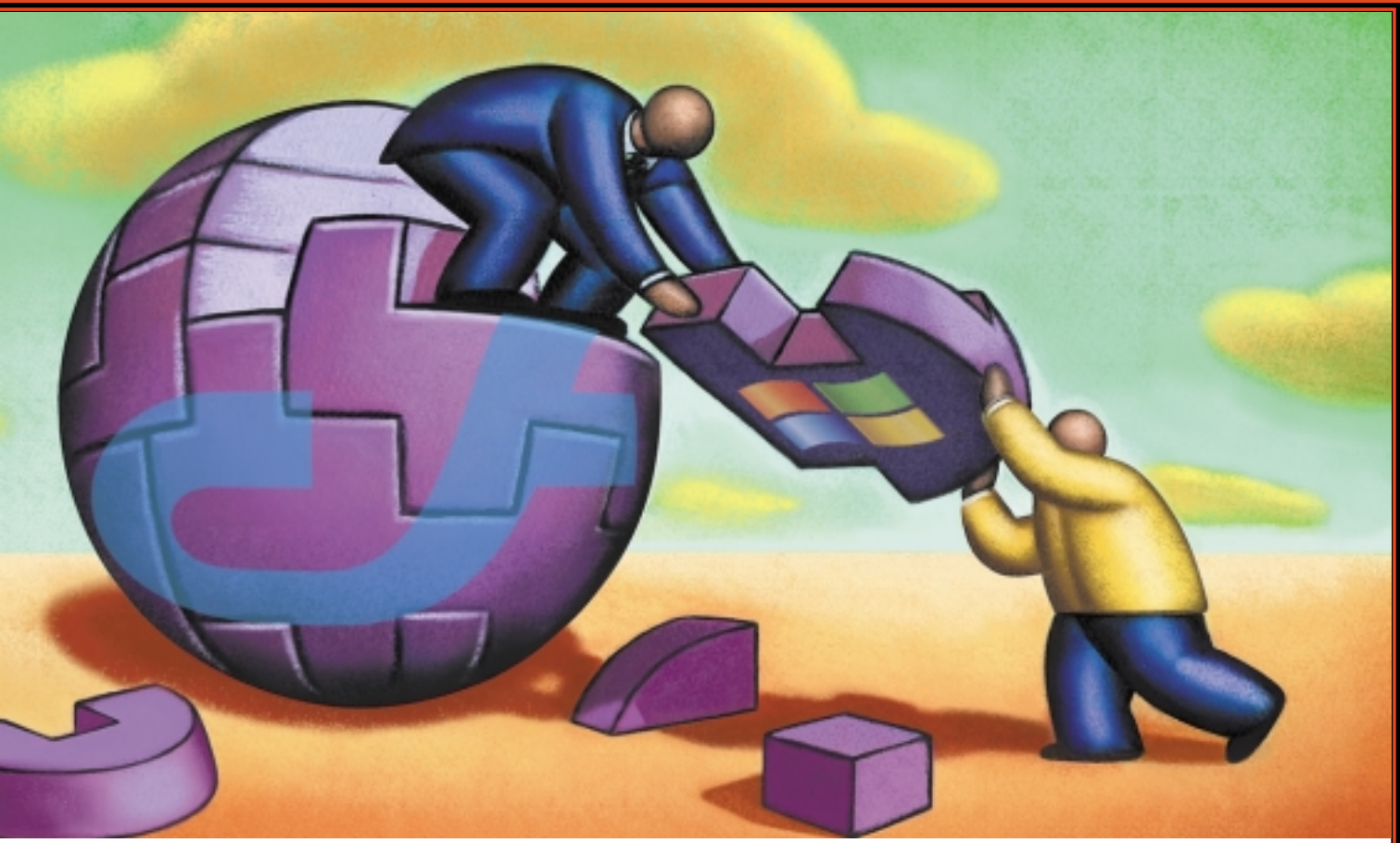
The most apt description of Windows Server 2003 I've heard is, slightly paraphrased, "It's the best version of Windows 2000 yet!" That is, it's not a sea change, but rather a welcome revision to an existing operating system. In that respect, I suspect it will be very exciting to those who currently manage numerous Windows servers.

There are four new versions of Windows Server 2003. As with Windows 2000, there are Standard and Enterprise Editions, but there's also a Datacenter Edition that supports high-end Intel servers. Of more interest to most of us is the Web Edition, which is designed for one thing: running Web servers. This edition should be suitable for most ColdFusion application servers, although it does have some significant limitations – it supports only 2GB of RAM and dual processors, and allows only 10 concurrent SMB connections for file and print services. This won't affect most production Web servers, but if you're planning on publishing an intranet, it may affect you. Best of all, though, Windows Server 2003 Web Edition is a lot cheaper than previous Windows licenses. It's approximately $400 and available only through OEMs, so you'll be able to get it when you buy a new Web server from your hardware vendor.

Some nice touches have been added to the OS to make it easier to manage. For example, functionality that's not especially useful to servers, like DirectX, is disabled by default. Internet Explorer is configured, by default, to disallow all functionality beyond basic HTML rendering, so you can't easily download malware onto your server console if you forget that you shouldn't be using the browser on your server console. The "Remote Assistance" functionality introduced in Windows XP is also included here, which should make it a bit easier for server administrators to help each other out. Fortunately, it didn't borrow the default look and feel of Windows XP, but provides a relatively spartan Windows 2000 style.

Internet Information Services 6.0

In the new version of Windows is a completely new version of IIS as well, with several substantial changes. Prior versions of IIS had a lot of security problems and required quite a bit of management knowledge to configure securely. IIS 6 is a lot closer to "secure by default," with ISAPI extensions and filters disabled unless you specifically enable



them yourself. In addition, most IIS functionality runs within a very low privileged security context; in IIS 5 and earlier versions, security holes in ISAPI extensions or filters could allow malicious code to run as SYSTEM.

There are lots of other interesting changes to IIS 6. Some functionality has been moved into a kernel-mode driver to improve performance. IIS management has been significantly improved; IIS configuration information, or the "Metabase," has been converted into an XML text file so you can work with it directly without needing a special editor. From within the IIS management console, you can easily work with portions of the Metabase, exporting and importing text files. You have more control over ISAPI application isolation as well – you can create your own application pools directly, and place each virtual server or directory within the application pool of your choice. This allows you to keep one ISAPI application from bringing down the whole server.

ColdFusion MX 6.1: 'Red Sky'

The next release of ColdFusion MX will be available at almost the same time as Windows Server 2000 – probably by the time you're reading this. ColdFusion MX 6.1, code named "Red Sky," will be the first version of ColdFusion to support the new Windows OS, and it provides lots of new features.

To investigate these features, I installed the release candidate of CFMX 6.1 onto Windows Server 2003 Web Edition. Before installing, I configured several virtual servers, but otherwise left IIS settings at their default values.

Anyone who's heard anything about Red Sky probably knows that Macromedia has taken great pains to simplify and ease the installation process. Many people have had quite a bit of difficulty installing the initial releases of CFMX, especially when it comes to integrating CFMX with a Web server. I'm happy to report that there were no such problems during this installation – CFMX 6.1 detected all of my virtual servers and configured each one appropriately (see Figure 1).

After the installation, a "configuration wizard" will open within a browser window. However, if you're installing to multiple virtual servers, you may have to edit the URL within the browser window, as it'll use 127.0.0.1 by default; if you're using host header names or don't have the right virtual server bound to that IP address, it won't resolve correctly. Fortunately, that's easy to address.

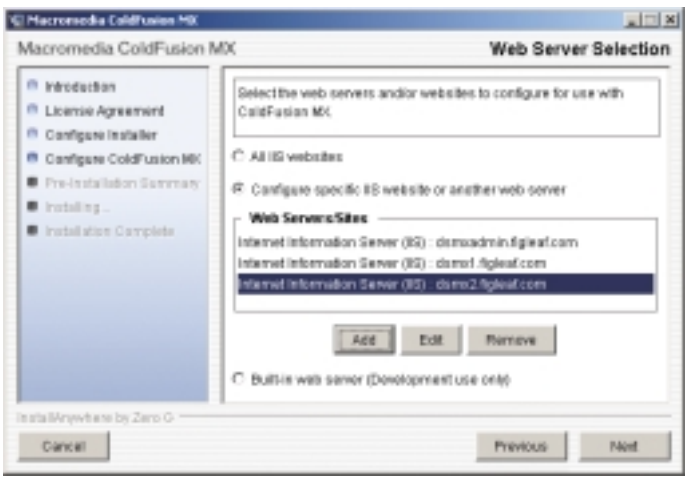


Figure 1: Configuring IIS virtual servers during install

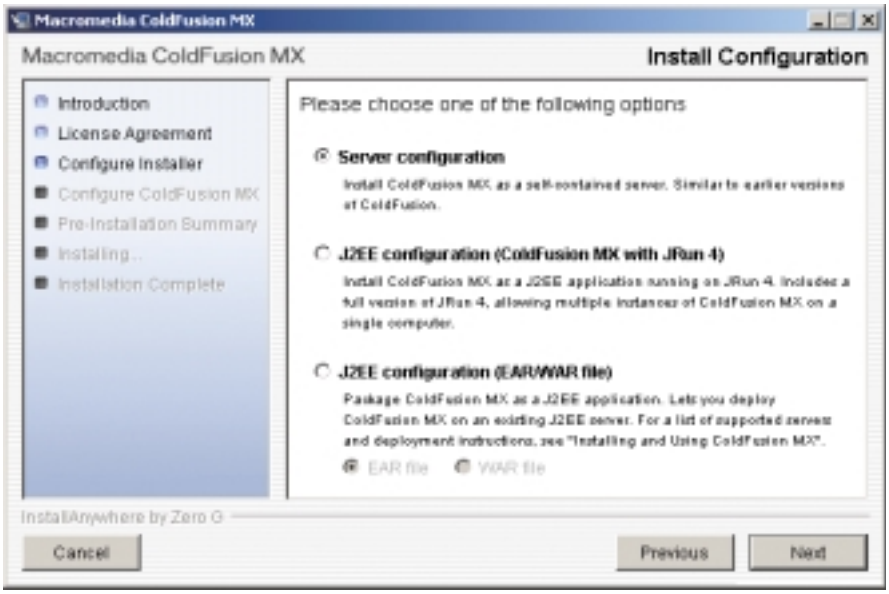


Figure 2: CFMX configuration options

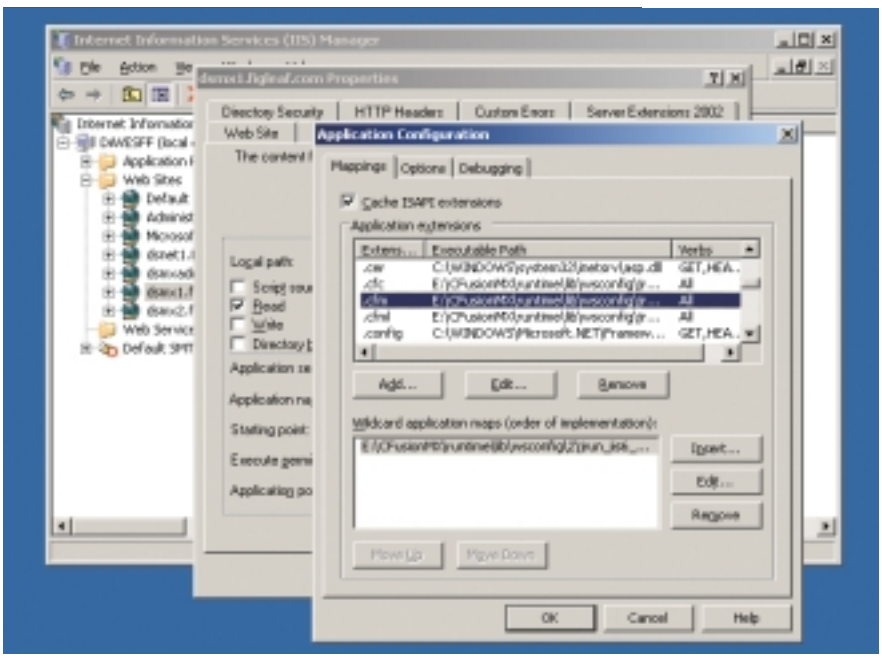


Figure 3: IIS management console showing application mappings

In addition, the CFMX installer provides many other niceties (see Figure 2); for example, it allows for a "standalone" install like CFMX Enterprise or Professional, a full J2EE install including JRun, or a J2EE application package for deployment to an existing J2EE application server if you have one. It also makes installing to a production server easier by allowing you to disable RDS during the install rather than by editing a text file afterwards.

As mentioned earlier, ISAPI extensions and filters don't work by default, but the CFMX 6.1 installer takes care of that for you by creating an entry called "Macromedia Server Extensions" and enabling it.

In prior versions of CFMX, you could integrate it with IIS via an ISAPI filter or extension, and it wasn't too clear how this should be done or what the ramifications would be either way. CFMX 6.1 takes another approach – it creates something new in IIS 6 called a "Wildcard Application Map" (see Figure 3). This is similar to a filter in that it's not invoked based on the filename requested (for example, "*.cfm"). However, it's not a filter either, which means that it's not subject to the security problems that ISAPI filters may face; since ISAPI filters execute so early in the request process, in some cases they may sidestep things like filesystem permissions and the like. Also, if an ISAPI filter has a security problem, like a buffer overflow, it will typically execute within a more privileged security context, although that's been addressed somewhat within the IIS 6 security architecture anyway.

Another nice touch about the CFMX 6.1 install was that it automatically configured the appropriate IIS virtual servers to enable "index.cfm" as a default document and ensured that the Flash Remoting gateway was set up. It appears that Flash Remoting requests are handled by the "Wildcard Application Map."

Summary

If you're currently managing Windows servers and you're getting ready to deploy CFMX, make sure you look at CFMX 6.1 on Windows Server 2003 – it's a lot easier to install and manage than it used to be. Current Windows Web server administrators can take advantage of new IIS management functionality to make their lives a lot easier by working directly with the Metabase using their tools of choice, rather than limiting themselves to writing WMI and ADSI scripts to manage IIS. Current CFMX administrators can ease their server deployments by taking advantage of the much cleaner and easier installation process of CFMX 6.1.

About the Author

Dave Watts is the CTO of Fig Leaf Software, a ColdFusion development shop and training provider in Washington, DC. He's worked with CF since version 1.5, and is a Certified Macromedia Advanced ColdFusion MX developer and trainer. He frequently speaks at industry events and user groups.

dwatts@figleaf.com

Introducing ColdFusion MX 6.1

Jump on board!

It's been over a year since Macromedia released the most important and ambitious ColdFusion ever, ColdFusion MX. Considering the scope of the undertaking, ColdFusion MX has been an incredible success.

By Ben Forta

Reengineering and rewriting a mature product from scratch is not a task undertaken lightly, but the customer feedback we've received has affirmed that we did the right thing. The enhancements in ColdFusion MX have given developers important new capabilities, and the move to the Java-based architecture has helped us increase performance and provide developers with exciting new deployment options.

But as with all ambitious projects, there were things we couldn't get done and there were things we didn't get quite right the first time. And that's what brings us to ColdFusion MX 6.1, an absolutely vital update to ColdFusion MX. It's faster, simpler, and much more powerful than even ColdFusion MX. And so, for the first time in printed form, it is my pleasure to introduce you to ColdFusion MX 6.1.

Simplified Installation and Migration

Installation and migration have proven to be the primary gotchas for CFMX users. ColdFusion MX 6.1 features a brand new cross platform installer that:

- Provides clear instructions and explains options in detail
- Gracefully handles the non-plain-vanilla installations (the ones that gave CFMX a hard time)
- Works out-of-the-box with multi-homed systems
- Includes an advanced Web server configuration wizard
- Provides improved upgrade and migration tools
- Continues to support silent installation for those who need it

ColdFusion MX 6.1 also features increased backward compatibility. Thanks to all the feedback from early adopters of ColdFusion MX, we were able to identify dozens of areas where we'd inadvertently broken compatibility. As a result, it should be much easier to bring forward your ColdFusion 5 and 4.5 applications.

New OS Support

There have been several important operating system upgrades since ColdFusion MX shipped, and many users have been clamoring for support for these operating systems. New to CFMX 6.1 are:

- Windows Server 2003 (and IIS6)
- RedHat Linux 8
- RedHat Linux 9
- SuSE Linux 8
- Solaris 9
- AIX 4.3.3 and 5.1

Of course, all currently supported platforms and operating systems remain supported.

Faster Development

By now just about everyone knows that CFMX is a compiler – it compiles CFML code to Java bytecode. Actually, that is almost true; what it actually does is generate Java source code corresponding to the original CFML, and then compiles that generated Java to bytecode.

I'm not going to explain the benefits of ColdFusion being a compiler (this has been covered before extensively). What I do want to point out is what many of you have already discovered: ColdFusion the compiler improves execution speed at runtime, but it hurts performance at development time. Why is this? The initial code generation and compile process is time consuming, and each time you tweak a CF tag and then try your change, ColdFusion has to go through that entire process again.

ColdFusion MX 6.1 comes with a brand new compiler, one that compiles from CFML to Java bytecode directly (without needing to generate Java source code and spawn another compiler). The result? Blinding fast execution, so fast that you'll likely not even notice the difference between the initial compile and subsequent requests.

In fact, the compiler is so fast that you may not want to bother saving the compiled .class files anymore. CFMX compiles to disk, the CFMX cfclasses directory contains a .class file for each .cfm file, and once compiled, ColdFusion accesses those .class files directly so as to not have to recompile the CFML source again. But in CFMX 6.1 the compiler is so fast that you'll likely find that there is no real value in storing the .class files. Instead, ColdFusion can compile to memory and execute the bytecode directly from there (this will also solve the problem that some of us ran into where .class files were left over or went out of synch). Of course, this will mean that if the server restarts, ColdFusion will need to recompile your .cfm files, but this process is so fast that it may be worth it (after all, checking file time stamps and reading .class files from disk takes time too).

ColdFusion MX 6.1 supports both compiling to disk (CFMX behavior) and compiling to memory. The ColdFusion Administrator lets you define how you'd like the compiler to behave.

Faster Runtime

The new compiler has no real impact on runtime, its job is to improve development time. But ColdFusion MX 6.1 improves runtime performance too. While exact numbers were not available at press time, initial testing using example application testing (identical applications on identical hardware) has shown significant performance gains even over ColdFusion MX (which was already faster than ColdFusion 5, which was already faster than… you get the idea).

Improved Protocols

The protocol tags are a very important part of the CFML language, and ColdFusion MX 6.1 improves and enhances them all. Key improvements include:

- <CFHTTP> now supports all HTTP operations (GET, POST, HEAD, PUT, DELETE, TRACE, OPTIONS).
- <CFHTTP> now provides access to all headers and content, and provides explicit control over timeouts and proxy support.
- <CFPOP> now supports the retrieval of multipart e-mail messages (those with text and HTML parts contained within a single message).
- <CFINVOKE> now supports secure connections (via https).
- <CFINVOKE> now provides control over timeouts and proxy support.

Improved <CFMAIL>

The most used Internet protocol tag has to be <CFMAIL>, and <CFMAIL> has been dramatically enhanced too. For starters, in ColdFusion MX 6.1 (Enterprise) it is possible to allocate multiple mail delivery threads, and also keep SMTP connections open. The combination of these two features introduces mail delivery throughput that exceeds anything possible in prior versions of ColdFusion. On test boxes <CFMAIL> has been clocked delivering over 1,000,000 messages an hour!

<CFMAIL> also now supports SMTP logins (required by many SMTP servers to prevent mail relaying). The <CFMAIL> attributes USERNAME and PASSWORD allow for the login information to be provided within the tag. (It is also possible to provide login information in the SMTP server definition in CF Admin).

Another frequently requested <CFMAIL> enhancement is support for multiple SMTP mail servers (so that if one is unavailable an alternate may be used). This is now supported in ColdFusion MX 6.1 (Enterprise) at both the CF Admin level and the <CFMAIL> level.

In addition, the new <CFMAILPART> tag allows developers to create multipart messages so that a single message may contain both HTML and text versions of the message body. The syntax looks like this:

```
<CFMAIL ...>
   <CFMAIL PARAM ...>
   <CFMAILPART type="text">
   Text version goes here
   </CFMAILPART>
   <CFMAILPART type="html">
   <B>HTML version goes here</B>
   </CFMAILPART>
</CFMAIL>
```

Improved CFCs

ColdFusion Components (covered in detail in ***CFDJ,*** Volume 4, issues 6 and 7) are the most important CFML language enhancement in ColdFusion MX. ColdFusion MX 6.1 fixes several issues with CFCs, and adds the single most requested enhancement:

- Within a CFC it is now possible to use the "super" scope to access overridden methods.
- CFCs can now safely be placed in a variety of scopes, and CFC code has access to all scopes.

Other Bits and Pieces

There are also all sorts of other little goodies. For example:

- A new Wrap() function which inserts breaks into text to force wrapping (used internally by the new <CFMAIL> WRAP attribute)
- Lots of COM improvements, and the introduction of a ReleaseCOMObject() function (which does exactly what its name suggests)
- Improved <CFCHART> performance
- An update to the Flash Remoting engine
- The embedded AXIS engine has been updated to v1.1 (this provides numerous SOAP enhancements including better interaction with .NET Web services)

Versioning Changes

Even if none of what I have mentioned thus far makes you sit up and take notice, this next one will.

With ColdFusion MX 6.1 we've changed the product editioning. ColdFusion Professional has been replaced by ColdFusion Standard, and ColdFusion Enterprise is now a combination of ColdFusion Enterprise and ColdFusion for J2EE (and we've even included a full version of JRun as well). What does this mean to you? As a ColdFusion Enterprise user you now have several different ways to install ColdFusion MX:

- ***As a standalone:*** Like in CFMX (using the embedded JRun).
- ***On top of JRun:*** You get a full JRun installation, and the ability to run multiple CF instances on top if it. This translates into better performance, greater security, superior scalability, and more control over specific applications (as explained in last month's column, ***CFDJ,*** Volume 5, issue 7).
- ***On top of a J2EE server of your choice:*** For example, IBM WebSphere, BEA WebLogic, and Sun ONE.

In other words, you are getting CFMX, CFMX for J2EE, and JRun 4 – all for the same price and the same upgrade.

Summary

ColdFusion MX 6.1 is an important upgrade to an important product. If you are already using ColdFusion MX then 6.1 is free, and you'll enjoy greater performance and stability than ever before. And if you are not yet using ColdFusion MX, well, there couldn't be a better time to jump on board.

About the Author

Ben Forta is Macromedia's senior product evangelist and the author of numerous books, including ColdFusion MX Web Application Construction Kit *and its sequel,* Advanced ColdFusion MX Application Development, *and is the series editor for the new "Reality ColdFusion" series. For more information visit* www.forta.com.

ben@forta.com

Parsing RSS Feeds Using ColdFusion

Use ColdFusion to bring all your favorite RSS feeds together

XML is popping up all over the Internet and the need to transfer information from system to system, independent of the platform, is greater then ever. XML (eXtensible Markup Language) allows you to define your own human readable markup language (similar to HTML).

By Dominic Plouffe

Many vendors are now operating with XML, due to the fact that it is text based and platform independent. A number of XML-based standards have already been created to facilitate data exchange. This article will focus on one of these standards, and probably the most popular, RSS. In this article you'll learn how to read and parse an RSS document using the ColdFusion MX XML tags and functions, and some tips will be offered on how you can aggregate the data for later use.

What Is RSS?

RSS is an XML 1.0–based standard that was developed by Netscape to transfer data between Web sites from their portal environment. It is not any different than any other standard and has an umbrella of different versions that have been developed by different organizations. The original specification was developed by Netscape as a format to supply news items from their portal environment to other news Web sites. When Netscape withdrew from the portal business, RSS was also subsequently dropped from development. Other software vendors (most notably Userland Software) picked up the idea and continued to develop it.

Today, there are many different versions of RSS in use: version 0.91, which was standardized by Netscape; versions 0.92, 0.93, 0.94, and 2.0, which were developed by Userland; and version 1.0 (also called RDF), which was developed by independent organizations.

As with most XML documents, the meaning and content can be fairly easily gleaned by simply looking at it. Many Web sites now provide a summary of their content through RSS documents, such as Macromedia (www.markme.com/mxna/blogview.cfm?blogid=8). Also, more and more industry leaders such as Jeremy Allaire (http://radio.weblogs.com/0113297) and Kevin Lynch (www.klynch.com) are now sharing their thoughts through weblogs that are also publishing their content using the RSS standard.

This article will concentrate mainly on version 0.91 (RSS 0.91) since it is the simplest form of RSS, but the techniques represented in this article will apply for most versions of RSS and other XML documents. Listing 1 shows a sample of a basic RSS version 0.91 document.

The XML document starts with the <channel> tags which include the title, description, and URL of the site the document is describing along with items that the channel currently contains. For each <item> tag, there is a title of the article, the description, and a URL that points back to the original Web site. Take note that this is a bare bones RSS document and that most documents that are in use have more metadata such as author, publish date, and images.

ColdFusion and XML – Parsing

Because RSS provides an easy way for Web sites and weblogs to share data, we now have a wealth of information at our fingertips. This overload of information also means that we need some way to compile it all, in order to make it easier and faster to read. The following section will describe the methods you can use to read an XML document.

There are two methods widely used to parse XML. The first is API-based functions; the second is a Document Object Model (DOM). When using a DOM, the XML document is read into memory and manipulated through functions that are provided. The DOM approach is generally considered much more powerful. However, it suffers from serious drawbacks that include the inability to process large documents (mainly due to memory constraints).

ColdFusion MX offers you a set of functions to process XML. It is these functions that we will use to parse our RSS document and create the aggregator.

The first step of the aggregator is to retrieve the RSS document. We accomplish this by using the <CFHTTP> tag and pointing it to the URL of the RSS document.

```
<cfhttp url="http://www.mywebsite.com/rss091.xml"
   method="get">
```

The next step is to use the XMLParse() ColdFusion function and create an XML document using the text that we have already retrieved with the <CFHTTP>:

```
<cfset objRSS = xmlParse(cfhttp.filecontent)>
```

Creating an XML object will parse the XML document into a series of XML elements and structures that will enable us to use the predefined ColdFusion variables to manipulate the RSS document. If you wish to view the structure of the XML document you can perform this command after you have done the XMLParse():

```
<CFDUMP var="#objRSS#">
```

Once the object has been parsed by ColdFusion, you are now ready to retrieve the document meta data and the items. The data within the object can be retrieved by referencing each XML node using the hierarchy of the document beginning by the root node. For example, if we would like to retrieve the text within the <title> tag of the channel we do the following:

```
<cfoutput> #objRSS.channel.title.xmltext#
   </cfoutput>
```

Xmltext is a special reserved variable created when ColdFusion parses an XML object, which is used to refer to the data appearing between a given XML element's open and closing tags. We can

easily reference the <title>, <link>, and <description> tags under the channel, but we still need to retrieve the data from the <item>s. Since there is more than one <item> element in the document, we cannot reference them by doing the following because ColdFusion will not know which item to retrieve:

```
<cfoutput> #objRSS.channel.item.title.xmltext#
</cfoutput>
```

The ColdFusion MX XMLParse() function gets around this issue by creating an Array with the elements when it encounters more than one of the same name within a given parent. The array permits us to reference each item like this:

```
<cfoutput> #objRSS.channel.item[1].title.xml
text# </cfoutput>
```

In our example we use a <CFLOOP> to loop through all the <item> tags. We use the following ArrayLen() function to identify the number of <item> tags:

```
<cfset Item_Length =
arraylen(objRSS.channel.item)>
```

Listing 2 has the complete listing of the XML parser with all the tags and functions described in this section.

Notice that the XmlParse function is wrapped in a CFTRY statement. Since the file being retrieved may or may not be a properly formatted XML document and could throw an error, it would be wise to handle any such error this way.

The Next Step

Now that you have retrieved all the data from the RSS document, you'll probably want to do something useful with it. The next step would most likely be to take the data that you have retrieved and insert it into a database, and then create a front end interface to view the data. Also, since most RSS documents are created dynamically and change on a regular basis, it's a good idea to create a scheduled task in the ColdFusion administrator to run your RSS parser/database update on a regular basis.

Conclusion

Hopefully this article has shown you how to properly parse and display an RSS document using the ColdFusion functions and will get the ideas flowing in your mind. More thought would be required to create an RSS aggregator that can be used on your Web site or intranet. Although we used the RSS standards as an example, you can use the same function and ideas to create a parser and aggregators for any XML document.

Resources

If you would like to purchase an RSS aggregator for your Web site as opposed to creating your own, you can have a look at the Macromedia DevNet Resource Kit Volume 3 (www.macrome dia.com/software/drk/productinfo/product_overview/volume3). Not only does the DRK3 come with its own aggregator, it also allows you to create and manage your own RSS feeds.

More information on the RSS standards is available at: http://backend.userland.com/rss, http://backend.userland.com/rss091, http://web.resource.org/rss/1.0/spec.

More information on the ColdFusion XML Functions is available at: http://livedocs.macromedia.com/cfmxdocs/CFML_Reference/functions-pt022.jsp#3468770

(Listings are on page 16)

About the Author

Dominic Plouffe, cofounder and VP R&D of FuseTalk, Inc., has been designing Web applications since 1997.
dominic@fusetalk.com

Listing 1

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="0.91">
  <channel>
    <title>My Articles</title>
    <link>http://www.website.com/</link>
    <description>This is a listing and short descriptions of my recent
     articles.</description>
    <language>en-us</language>
   <item>
          <title>My First Article</title>
       <link>http://www.website.com/articles?articleid=1000</link>
          <description>This is the description of my first
           article.</description>
   </item>
    <item>
      <title>My Second Article</title>
      <link>http://www.website.com/articles?articleid=2000</link>
      <description>This is the description of my second
       article.</description>
    </item>
    <item>
      <title>My Third Article</title>
      <link>http://www.website.com/articles?articleid=3000</link>
      <description>This is the description of my third article.</description>
    </item>
    <item>
      <title>My Fourth Article</title>
      <link>http://www.website.com/articles?articleid=4000</link>
      <description>This is the description of my fourth article.</description>
    </item>
  </channel>
</rss>
```

Listing 2

```
<!--- Retrieve the RSS document --->
<cfhttp url="http://dplouffe.fusetalk.com/rss/rss901.xml" method="get">

<!--- Validation flag --->
<cfset XMLVALIDATION = true>

<cftry>
   <!--- Create the XML object --->
   <cfset objRSS = xmlParse(cfhttp.filecontent)>

   <cfcatch type="any">
          <!--- If the document retrieved in the CFHTTP
          is not valid set the validation flag to false. --->
          <cfset XMLVALIDATION = false>
   </cfcatch>
</cftry>

<cfif XMLVALIDATION>
   <!--- If the validation flag is true continue parsing --->

   <!--- Set the XML Root --->
   <cfset XMLRoot = objRSS.XmlRoot>

   <!--- Retrieve the document META data --->
   <cfset doc_title = XMLRoot.channel.title.xmltext>
   <cfset doc_link = XMLRoot.channel.link.xmltext>
   <cfset doc_description = XMLRoot.channel.description.xmltext>

   <!--- Output the meta data in the browser --->
   <cfoutput>
          <b>Title</b>: #doc_title#<br/>
          <b>Link</b>: #doc_link#<br/>
          <b>Description</b>: #doc_description#<br/><br/>
   </cfoutput>

   <!--- Retrieve the number of items in the channel --->
   <cfset Item_Length = arraylen(XMLRoot.channel.item)>

   <!--- Loop through all the items --->
   <cfloop index="itms" from="1" to="#Item_Length#">
          <!--- Retrieve the current Item in the loop --->
          <cfset tmp_Item = XMLRoot.channel.item[itms]>

          <!--- Retrieve the item data --->
          <cfset item_title = tmp_item.title.xmltext>
          <cfset item_link = tmp_item.link.xmltext>
          <cfset item_description = tmp_item.description.xmltext>

          <!--- Output the items in the browser --->
          <cfoutput>
                    <b><i>Item #itms#</i></b><br/>
                    <b>Title</b>: #item_title#<br/>
                    <b>Link</b>: #item_link#<br/>
                    <b>Description</b>: #item_description#<br/><br/>
          </cfoutput>
   </cfloop>

<cfelse>
   <!--- If the validation flag is false display error --->
   Invalid XML/RSS object!
</cfif>
```

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

ColdFusion MX/J2EE Hybrid Applications

10 best practices

Since the release of ColdFusion MX, the ColdFusion community has been rumbling about the possibilities made available by CFMX's powerful Java 2 Enterprise Edition (J2EE) underpinnings; using servlets, JavaServer Pages (JSP), JavaBeans, and Enterprise JavaBean (EJB) components are the most notable examples.

By Drew Falkman

Out of this discussion has arisen the concept of ColdFusion/J2EE hybrid applications. As an ideal, CFMX/J2EE hybrids offer the rapid development and robust high-level services of CFMX, while integrating the deeper transactional and low-level programming power of Java.

With all of the possibilities for architecting and developing these hybrid applications, it seems to me that it's time for the ColdFusion development community to seriously discuss what some of the best practices might be when developing these hybrid applications. Surely there are infinite possibilities for integrating CFMX with J2EE, but which of these are viable and truly offer the best that both frameworks have to offer?

This is the question I have been asking myself again and again over the past few months while I collaborated on a book about this topic (*Reality ColdFusion: J2EE Integration*), developed applications (including the Lindex component using Apache Lucene for the Macromedia Developer's Resource Kit 3), and did a case study for Macromedia's DevNet site (www.macromedia.com/devnet/mx/coldfusion/j2ee/articles/hybrid.html) on hybrid applications. In short, I've been living a hybrid life. This article is not only to share what I've learned, but also to contribute to and further an ongoing discussion on best practices for CFMX/J2EE hybrid applications.

Note: in order to best focus on the subject at hand, I have foregone providing background on what the two environments, CFMX and J2EE, are. If you are looking for more background information on this subject, I suggest the following resources:

- Chapter 32 of the CFMX manual, "Developing ColdFusion MX Applications with CFML," online at http://livedocs.macromedia.com/cfmxdocs/Developing_ColdFusion_MX_Applications_with_CFML/Java.jsp
- Ben Forta's online presentation at Macromedia DevNet (www.macromedia.com/software/coldfusion/j2ee/special/presentations/tech_intro/)
- Ben Forta's *CFDJ* column in Volume 5, issue 3, "ColdFusion & Java: More than just the sum of their parts"
- *Reality ColdFusion: J2EE Integration*, Forta, Falkman, et al (Macromedia Press)

ColdFusion MX and J2EE: A Good Match

What I've realized most in this process is that these two technologies really complement one another well. ColdFusion provides high-level services and an intuitive and rapid development framework, while J2EE provides low-level application programming interfaces (APIs) and powerful development objects. The most important thing to understand for anyone looking at developing hybrid applications is what each environment does well. Table 1 shows the key strengths and differences between both environments.

CFMX/CFML	J2EE/Java
Short Development Times	Long Development Times
High Level	Low Level
Tag-based	Object-oriented
Short Learning Curve	Long Learning Curve
Provides Tags for High-Level Services	Provides APIs for Low-Level Services

Table 1: CFML and J2EE primary strengths and differences

At the core, CFMX brings rapid development to J2EE, something it was clearly lacking before. Developers can learn and become proficient in using CFML in significantly less time than Java, which requires understanding object-oriented programming concepts, strict data types, and a vast library of programming constructs. At the same time, because of its object-oriented nature and vast library collection, Java offers a considerable amount of control for those who know how to use it.

In addition, the J2EE framework contains a number of programming constructs – such as EJBs and messaging services – that simply can't be emulated using ColdFusion alone. The best part of these differences is that there is no reason to choose one or the other anymore, just when to use which. This leads us to the heart of the matter: the best practices.

The 10 Best Practices

Best Practice #1:
Use only what you need

There's no reason to add complexity – ever. Introducing a second programming environment when you really need only one is nonsense. If you are building a CFMX application, don't use Java if you don't need to. The Java language is not as accessible as CFML – you'll add hours of learning, planning, and coding into your development process. Understand why you should use it first. With J2EE-only applications, you can likely gain a significant amount of time by using CFMX, even with the learning curve. However, if using CFMX means rewriting a significant amount of JSP/servlet code you may want to reconsider.

Best Practice #2:
Use what you already have

Okay, this is another "no-duh" best practice, but I wouldn't feel right if I didn't mention it. If you have a CFML template that's handling a core, complex piece of business logic, don't recode it into an EJB or something else unless you have a compelling reason to do so. It needs to be distributed? Make it into a ColdFusion Component (CFC) and deploy it as a Web service. The same goes even more so for existing servlets and Java classes. It is relatively easy to integrate these items from within CFML. See the ColdFusion manual reference above for more information.

In fact, capitalizing on existing code is a central argument for hybrid applications – use the Java code you have already developed, but build around it in CFMX (and in half the time it would take to do so in Java). In addition, this point should include capitalizing on open-source (or closed-source, for that matter) code in both environments. The ColdFusion community has a number of good resources for CF tags and user-defined functions, and the Java community has a huge open-source and third-party product community – use this prebuilt, pretested code whenever possible.

Best Practice #3:
Use CFMX for prototyping

ColdFusion, because of its rapid development capabilities, makes an ideal prototyping solution. You can easily plug in a Microsoft Access (or mySQL) database, some CFQUERY and CFOUTPUT tags, and bam! (sorry Emeril…) you're up and running with a pretty deep prototype. You can then capitalize on your prototype code by tightening it up, adding error handling, and maybe converting the back-end processing into EJBs or other Java components. This is especially useful if you use Best Practice #4 in your conversion process.

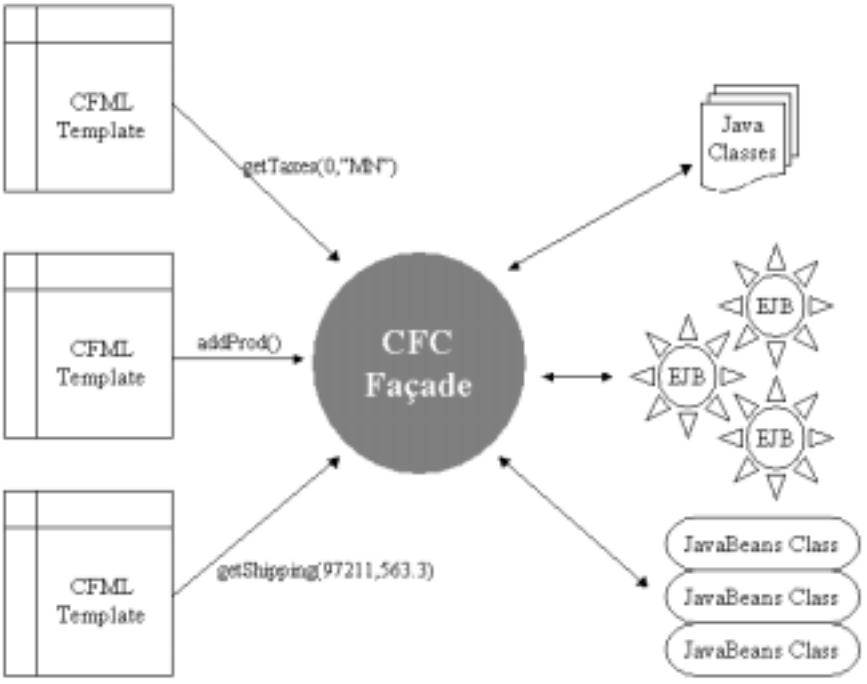


Figure 1: Use the CFC Façade pattern to provide abstraction for J2EE objects

Best Practice #4:
Use the CFC Façade pattern

If I had to pick a *best* best practice, this would be it. CFCs allow CFMX developers to create their own components containing functions. These functions are then called on from within any CFML template, and can even be exposed as Web services. CFCs are perfect for calling on Java classes of any kind. Let's say you have an EJB that calculates sales tax for in-state orders. The taxBean EJB has a method called getTaxOwed() that you pass the total sales amount to. In a hybrid application, you would create a CFC called eCommerce.cfc or something like that. Inside of this CFC, you would create a function called getTaxOwed() (sound familiar?). This is simply an abstraction of the Java code. Then you can use CFINVOKE or CFSCRIPT to call on the getTaxOwed() CF function.

This layer of abstraction makes irrelevant how the eCommerce.cfc works internally, thus allowing the getTaxOwed() CF function to call on an EJB or some other type of Java class, or even perform the logic itself. Using this pattern is great for prototyping – simply return dummy data in the prototype, and then when you plug it in for real, only the CFC needs to be changed. This is based on a well-established J2EE pattern called the Session Façade, whereby a session bean (EJB) is used to abstract logic of other EJB components. Figure 1 shows the CFC Façade pattern.

Best Practice #5:
Use the M-V-C pattern, with CFMX handling the V and usually the C

M-V-C stands for Model-View-Controller. In this architecture, you have three primary sections of an application: the Model is the data model and includes data storage and logic processing; the View handles the user interface (UI); and the Controller handles the request and response (input/output) of the application. If you do not have existing servlets or JSP pages handling the UI and request/response processing (see Best Practice #2 above), let ColdFusion handle this.

Between the Application.cfm, OnRequestEnd.cfm, CFMX's built-in security, and other services, as well as other frameworks (such as Fusebox and variations thereof), CFMX can effectively control an application, and output the UI – whether the UI is HTML, Flash, or XML. It is in this capacity that CFMX really shines. Developing the

View/Controller aspects of an application in CFML will take so much less time than developing servlets and JSP pages that your job might become a part-time position, or you at least will have time (finally!) to get up to date on the water-cooler gossip.

Best Practice #6: Use J2EE event listeners

Of all the J2EE constructs at your disposal in CFMX, the event listener is one of the most unique. Listeners can "listen" to the application server for certain events to happen – such as sessions being destroyed and initiated – and then execute a block of code. There is nothing like this in CF, and it's something that many applications can gain from. For example, by listening to session destroys, a developer could program a process that saves abandoned shopping cart information to a database. Listeners can be turned on and off using the application server's deployment descriptor – an XML file – thus allowing for easy implementation without having to mess around with code and multiple if statements.

Best Practice #7: Understand EJBs

Before I go on touting the virtues of EJBs (see my *CFDJ* article, "Beefing up Your Applications with EJBs," Volume 4, issue 8 for more information), I should mention that some still debate whether EJBs are really necessary. That said, EJBs are deployable components, meaning that components reside in a container much like CFML templates and servlets/JSPs do. This container can enable beans to be called on by remote applications, and can allow beans to be clustered in their own array of servers and more.

There are four types of beans: entity beans that handle interactions with datasources; message beans that handle messaging services; stateful session beans that store session data; and stateless session beans that handle business logic processing. All of these have a unique purpose. Because EJBs can be distributed across an enterprise, they can be an integral part of applications that are called on by other Java applications – be they desktop, applets, J2EE, or CFMX. They are also easy to deploy as Web services and to call on from .NET or other types of applications.

Best Practice #8: The best J2EE elements to use (in my opinion)

These are EJBs, JavaBean components, Java classes, event listeners, and maybe servlet filters (but only for server-wide processing directives). All of these components bring something to ColdFusion applications. In the case of JavaBean components and Java classes, they are especially useful when they encapsulate programming that requires low-level processing. I have already discussed the strength of EJBs and listeners. Servlet filters (for more information, see Charlie Arehart's article, "Fun with Filters," in *CFDJ*, Volume 5, issue 2) work much like the CFMX framework's Application.cfm and OnRequestEnd.cfm templates, processing code before or after a template (or servlet) is called on.

Deciding whether to use the CFMX framework or servlet filters is a matter of choice, remembering that servlet filters are controlled (like event listeners) in the deployment descriptor, so they can be easily turned on/off and are especially handy when they provide processing for multiple applications – something that can't be done with the CFMX framework.

Best Practice #9: Some J2EE elements are best used only if they already exist

Specifically, these are controller or view servlets, JSP custom tags and JSP pages, and single-application or template servlet filters. Instead of servlets, use CFML templates. Of course some things are still best handled with servlets (note that Flash Remoting is servlet-based), but unless there is a specific reason to use servlets, CFML templates are way faster to build and still end up as compiled Java bytecode just like servlets. JSP tags can be extremely useful – there are numerous libraries available to handle everything from database calls to internationalization to Web services, but in most cases you shouldn't need to develop these specifically for an application. (See Charlie Arehart's article, "Using JSP Custom Tags in CFMX: What, Why, and How," in *CFDJ* Vol. 4, issue 5 for more information.)

UDFs and ColdFusion custom tags are quicker to build. If you need low-level processing, Java classes are probably a better way to handle this. (For more on integrating Java classes in CFMX, see that CFMX manual reference as well as Guy Rish's eight-part series, "A Cold Cup o' Joe," starting in *CFDJ* Vol. 3, issue 1 and ending in Vol. 4, issue 8. While it discussed integration in CF 4.5 and 5, many of the same concepts for integrating with Java classes using CFOBJECT apply.)

Finally, servlet filters, though handy, really don't offer much beyond what Application.cfm and OnRequestEnd.cfm do. If you need to process logic for only certain pages or folders, do a <cfif> check on the CGI.Script_Name variable; of course the exception is server-wide processing.

Best Practice #10:
Don't use JSP, Struts, Velocity, or other J2EE RAD "frameworks"

I can hear my inbox already filling over this one. Any developer who has ever used both JSP and CFML will understand why eliminating JSP in its entirety is a good thing. Performing loops, controlling flow, and other programming logic either require external custom tags (which is an added pain compared to CFML) or scripting elements that make the code confusing and don't offer a clear separation of presentation and programming.

Only use JSPs if they already exist and you won't need to modify them significantly to work with your application. Having used both, I have found it is often faster to rebuild in ColdFusion (using the existing JSP page as a starting point) than to make the modifications to the original JSP pages and maybe their controller servlets. For example, Java developers will often put queries in the controller servlet, then output in the JSP page.

So if you want to add a field to the output, you need to edit (and recompile) the servlet to change the query (usually in a Java IDE), and edit the JSP (usually in an HTML editor) to change the output. Multiple development environments alone make this a hassle. Even if you are retrieving a query from a CFC into a CFML template, the change can at least be made in your CFML editor of choice. (For more on a comparison of CFML and JSP, see Vince Bonfanti's article, "Making the Case for CFML," in *CFDJ*, Vol. 5, issue 6.) As far as the other frameworks such as Velocity and Struts, simply put, these are J2EE-only frameworks. I know there are a very few people out there using Struts with CFMX, but in general I wouldn't recommend it. CFMX offers more services and a better (in my opinion) framework, and this only adds a third element to hybrid applications. What you will end up with is a hydra, not a hybrid.

Conclusion

Overall, this is just a start – Hybrid Best Practices 101. The truth is that hybrid applications are only in the infancy stage – J2EE developers are starting to realize the RAD capabilities that CFMX can add to their applications and ColdFusion developers are realizing the power and control they can add using J2EE. I encourage any and all developers (both J2EE and CFMX) to contribute their thoughts on this matter in the Macromedia and *CFDJ* forums.

About the Author

Drew Falkman is the author of JRun Web Application Construction Kit *and coauthor of* Reality ColdFusion: J2EE Integration, *both published by Macromedia Press. Currently Drew offers technology planning, project management, and development services through his consulting company, Veraison, LLC. Drew also speaks at events, writes for numerous publications, and teaches Web development courses at Portland State University. He is a member of Team Macromedia, a certified ColdFusion Developer, and a certified Macromedia instructor. You can learn more about Drew at www.drewfalkman.com.*

drew@drewfalkman.com

Mach-II

Breaking the procedural barrier

With the release of the MX version, ColdFusion has moved from its strict procedural programming background toward object–oriented design and programming. This move has evoked both hope and fear in developers, some welcoming the decidedly new concepts of object orientation (OO) and some dreading that they will lose the language they love.

By Hal Helms

By Ben Edwards

Within the Fusebox community, the introduction of ColdFusion components (CFCs) has stirred a great deal of interest: Would Fusebox leverage these new capabilities and, if so, how? Almost a year ago, we – along with John Quarto-vonTivadar – began work on a new version of Fusebox that would leverage the new capabilities of MX. At once there was a good deal of speculation on what "Fusebox MX" would look like. Would Fusebox become an OO framework? Would developers have to understand polymorphism and inheritance to use it? This article answers these questions and will, we hope, inspire people to look at a new, unashamedly object-oriented framework called Mach-II. In next month's issue of ***ColdFusion Developer's Journal***, we'll explore Fusebox 4 – the worthy successor to Fusebox 3.

ColdFusion's initial appeal was to "Webmasters" who wanted to make their sites more dynamic – and it succeeded admirably. But just as the term "Webmaster" is an anachronism, the call for more dynamic Web sites was succeeded by the need for true Web applications. As these applications became more involved and more ambitious in scope, ColdFusion developers discovered that even a thorough knowledge of tags and functions was just not enough. Realizing that the same problems presented themselves again and again, a group of developers wondered if these problems couldn't be solved – or at least ameliorated – by building some sort of framework on which Web-based applications could be developed. These visionaries traded ideas and code over e-mail. Ideas evolved and Fusebox was thus born, and it too succeeded admirably.

The Rise of OO

When ColdFusion MX was released, it generated a great deal of interest. For many developers, the most interesting aspect of the new release was the inclusion of a new encapsulation mechanism, the ColdFusion component (CFC). CFCs promised ColdFusion developers the ability to move from writing purely procedural code to object-oriented code. Macromedia began touting CFCs as "objects without all the fussiness."

For some developers, this represented a profound "this changes everything" shift – not without reason. Object orientation represents a shift in thinking. The procedural approach views each problem as a series of actions that must be taken in the correct order to achieve a desired result. Data is separate from functions. The object-oriented approach identifies separate, encapsulated components ("objects") in which data and functions are combined.

Fusebox represents a procedural approach to building applications. The object-oriented approach is very different from the procedural approach. Solutions result from first building classes that provide abstracted and simplified models of real-world counterparts. For example, a billing system might have classes for Invoice, Receipt, Customer, etc. When the application is run, instances of these classes (objects) are created. The running application then resembles an extended conversation between objects, with different objects sending messages to other objects to request information or ask that a service of some type be performed.

Shortly after the release of CFCs, we wrote "Discovering CFCs" (Techspedition Press, 2002), in which we explored in what ways CFCs were – and were not – truly object oriented. And since we were both involved in the Fusebox world, we decided to write a new version of Fusebox that would make use of the new capabilities of CFCs. We labeled the initiative, seemingly sensibly enough, Fusebox MX.

When we began talking about it openly, we heard from people who asked whether, with CFCs, the need for Fusebox was gone altogether? "Why do we need Fusebox when we have objects?" they asked. As we saw it, there were three ways to answer this question.

First, as we showed in our book, there were aspects of CFCs that simply could not be squared with an object-oriented approach. Second, if objects obviated the need for frameworks, why would Java, a language with first-class objects, have strong communities for over a dozen framework projects? Finally, however helpful individual components might be, they didn't address the issue of an underlying architecture. In short, we felt that Fusebox remained a valuable framework for the ColdFusion community.

OO and You

But a funny thing happened on the way to Fusebox MX. We began to ask questions about the appropriate software architecture on which to build the framework. Would Fusebox MX essentially be Fusebox – doing little more than adding components into a still procedural toolkit? That would be the easiest change for existing Fuseboxers to get used to, but would be little more than adding "super custom tags" to procedural code.

We felt strongly that the continued reliance on procedural code would put ColdFusion programmers at risk. Over the past 10 years, essentially the current lifespan of ColdFusion, a revolution has taken place. This revolution was not political, but technological, and it represented the ascension of object orientation as the dominant software construction paradigm. While languages such as Java, C#, Delphi, Visual Basic .NET, and Ruby (to name only a few) represented the new guard in this revolution, ColdFusion – and Fusebox – remained solidly procedural. Developers using ColdFusion simply didn't need to come to terms with object orientation.

Being involved in the Fusebox community, we felt a duty to ensure that Fuseboxers would not be using a proprietary framework cut off from the bigger IT world. We decided to rethink everything about Fusebox – including its architecture.

A Focus on Architecture

For many developers, the idea of "software architecture" simply means "how I put my application together." While obviously inexact, that definition of architecture was reflected by the great architect, Ludwig Mies van der Rohe, who wrote: "Whenever someone puts two bricks together, there architecture begins."

In practice, software architectures are commonly treated as a collection of components and connectors. Components are the system's functional elements, such as a shopping cart, a contact manager, or a database. Connectors define the protocols for communication between components. Examples of connectors include method calls, SQL queries, and HTTP requests. The architecture chosen for a system determines the vocabulary of components and connectors that can be used as well as the set of constraints defining how they are combined.

The choice of a particular software architecture is made on the basis of the goals of the designers, which is to say that there is no single-fit, perfect architecture. Over time, several different software architectural styles have risen to the fore, each with its own strengths and weaknesses. Our choice of a software architecture for a new Fusebox would then depend on what we wanted to accomplish. To determine this, we examined what others had come to rely on with Fusebox.

Fundamental Fusebox Goals

We began to speak with other Fuseboxers: What was it they most valued about Fusebox? From many voices, a few themes emerged. What Fuseboxers wanted from Fusebox was:

- ***The ability to more rapidly develop applications:*** Developers voicing this concern told us that they were being asked to develop applications without sufficient time or resources. They were far less concerned about "future-proofing" their careers by learning OO than they were simply about getting out from under the current backlog of work.
- ***The ability to help manage complexity:*** Simple apps are…well…simple, but they seldom remain simple. We've often noted that the most dangerous words clients can utter are, "You know what would be nice…." Despite our best efforts at requirements gathering, new requirements evolve and suddenly a simple application becomes more complex. Nor does ongoing development ever stop. Each new change becomes the basis for a new round of "improvements."
- ***The ability to do effective team development:*** In corporate environments especially, application development is done by teams. Each member has unique strengths to offer, but very few have all the needed skills to solely craft an entire Web application. Any change to the current version of Fusebox needed to ensure that the support for team development was not diminished.
- ***The ability to document an application:*** What developer wants to be forever chained to an application he or she has built? Yet without good documentation (and that meant documentation that was simple and easy to produce), developers were locked into maintaining their existing applications. It was simply too difficult for someone else to take over the responsibility for the application's maintenance.

Our own concerns centered on software maintenance. It is an often overlooked fact that between 70–90% of the life-cycle cost of an application is spent on maintenance. A framework that misses this point, by treating maintenance as an afterthought, gives away the enormous leverage that cost and time savings in maintenance can offer adopters of the framework. And if saving money is not a worthy enough goal, surely saving the sanity of

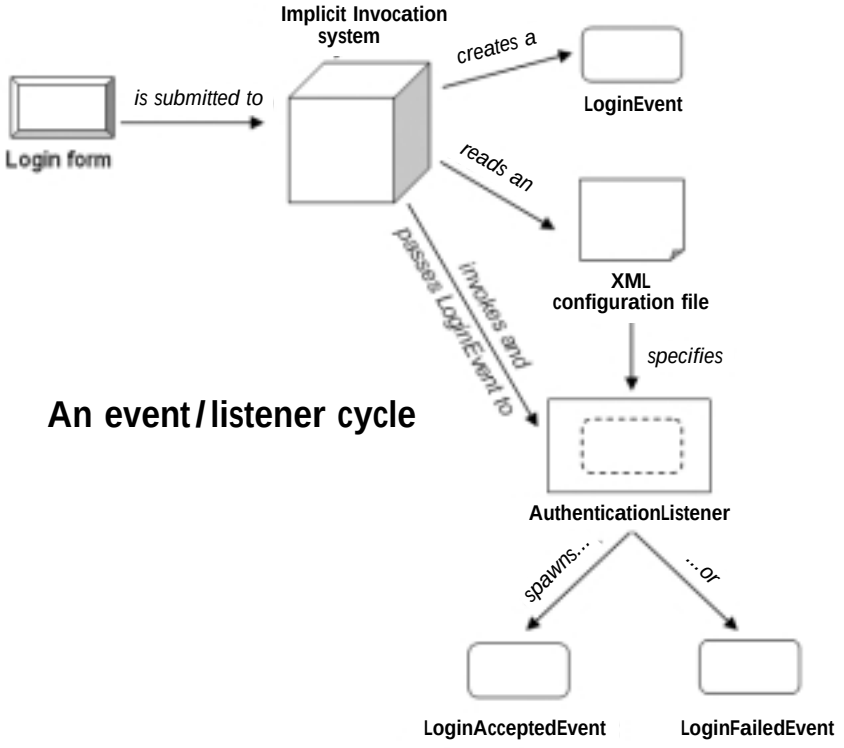


Figure 1: An event/listener cycle

coders who must work on poorly maintained code is. Thus, software maintainability became one of our chief goals.

Maintainability, we knew, was intimately tied to code reusability, since the same problem that causes an application to be difficult to maintain causes the components to be difficult to reuse – namely, the tight coupling between components.

Excessive dependencies between code components adversely affects all. Therefore, we felt that by tackling software maintainability, we would increase the chances for significant code reuse.

Cohesion and Coupling

Two metrics important for consideration in defining the publicly exposed interfaces of an architecture's components and connectors are a system's cohesion and coupling. Cohesion is the measure of the degree to which a component has a singular purpose. The greater cohesion a component exhibits, the more focused the component and the fewer the assumptions about contexts for reuse.

Coupling is the degree of interdependence between components. The less a component relies on other components (the looser its coupling), the more independent and reusable it is. Maximized cohesion (simple components) and minimized coupling (fewer connectors) are hallmarks of a flexible, maintainable architecture.

One of the more broadly accepted architectural styles in software engineering is known as implicit invocation. Implicit invocation architectures intrigued us. Event-based, implicit invocation is an example of a well-crafted architectural style with high cohesion and loose coupling. As such, it is one of the more broadly accepted architectural styles in software engineering. Examples of implicit invocation systems abound, including virtually all modern operating systems, integrated development environments, and database management systems.

Events and Listeners

Implicit invocation systems rely on the idea of events and listeners. Events are triggered whenever the system needs to do something, such as respond to an incoming request. Events can take many forms across different types of implementations; often for object-based systems, an event is an object whose properties contain any contextual information needed to process the event (similar to how an HTTP request carries with it all its form and query-string variables).

Listeners are business logic components that are registered with the system. When an event is announced, the system finds the listeners registered for that event and announces the new event to those listeners. Listeners fit the same criteria for components that we've already discussed – they are functional modules of the system. Components that wish to act as listeners are registered to listen for certain events at configuration time (by specification in an XML file, for instance). When an event is triggered, all registered listeners of that event are passed the event by means of a dynamically determined method call. In this way, functions are implicitly invoked. This process of notifying listeners of an event is called event announcement.

Events and listeners can themselves trigger other events. Let's consider how a common login/authentication scenario can be represented in terms of events and listeners. In this example, a login form is filled out by a user and then submitted. The incoming HTTP request triggers the creation of a LoginEvent, and the system populates the event with information in the request.

Next, the system determines the listeners for LoginEvent; in this case there is only one – the AuthenticationListener. Determined by a configuration file, the system invokes the AuthenticationListener's tryLogin() method, passing to it the event. Based on information in the event, the tryLogin() method will seek to authenticate the user. If the authentication succeeds, a new LoginAcceptedEvent is triggered. If authentication fails, a new LoginFailedEvent is triggered. The cycle then continues, with any listeners of the new event being notified (see Figure 1).

Loosely coupled components work together, but do not rely on each other to do their own jobs. The interaction policy is separate from the interacting components, providing flexibility. Components can be introduced into a system simply by registering them for events of the system, aiding greatly in

reusability and maintainability. Introduction of new components does not require change in other component interfaces, providing scalability as new features are added. Overall, implicit invocation eases system evolution.

Rather than starting with the idea of making a Fusebox for ColdFusion MX, we began with the idea of making a true, object-oriented framework that would meet the goals we adopted. Work on the framework proceeded without regard to a specific language. Instead, we wanted to ensure that this new framework would be adaptable to any language that implemented object-oriented principles. Only after all architectural decisions were made did we set about the task of implementing it in ColdFusion, where we found CFCs provided us the encapsulation we needed.

The end results have surprised and gratified us. We find that the system offers enormous flexibility. New functionality (a.k.a., "You know what would be nice…") is often as easy as adding a new listener to the system. Loose coupling ensures that other components do not need to be altered. We found that the implementation of the Model-View-Controller design pattern was a natural fit with an event-based invocation architecture. Finally, we found that the system could easily work with Web services, Flash remoting, Enterprise JavaBeans, and more.

Only one problem remained: What should we call this new framework? We began this journey with the idea that we were creating a "Fusebox MX". What emerged was something different and better than what we had hoped for. But was it Fusebox MX? In the end, we decided that a different name would better serve all involved. Certainly, the underlying architecture had little in common with Fusebox. Better, we felt, to give it a separate name. Fusebox 4 would be the successor to Fusebox 3, and would continue to offer enormous benefits to procedural programmers.

After too many late-night discussions (for some reason, naming something takes on enormous importance!), we decided that we wanted the framework name to reflect the fact that it was meant to help developers break the procedural barrier. Mach-II was thus born.

Next month, we'll explore the inner workings of Mach-II. In the meantime, we invite developers who want to use a powerful, flexible, object-oriented framework to build robust and maintainable applications to explore Mach-II at www.mach-ii.com.

About the Author

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

Ben Edwards is a Sun Certified Java Programmer and holds a degree in computer science from the Georgia Institute of Technology. He currently trains developers on software engineering practices focusing on Java, object-oriented programming, and software architectures. Ben is also cofounder of the Mach-II project.

hal.helms@teamallaire.com
ben@ben-edwards.com

Browsing Within CF Studio/HomeSite+

A whirlwind guide to this powerful feature

Do you know how to use the internal browse capability in CF Studio/HomeSite+? Did you know that you don't need to use the RDS feature to be able to do so? And that you can use it to browse not only CFML templates running on both ColdFusion and BlueDragon, but also JSP, ASP, PHP, and others?

By Charlie Arehart

Even if you don't care for the internal browser, did you know Studio can load your templates in any external browser you have installed? And that there are keyboard shortcuts for doing either?

In this month's **Journeyman** column, I'll walk you through the steps to enabling internal and external browsing in CF Studio (4 or 5) and HomeSite+. Along the way, I'll address several sources of confusion and frustration. If you've tried and failed, I'm sure you'll succeed after reading this. By the way, somewhat similar concepts apply to Dreamweaver MX, though you need to set up a "site" there. See the DWMX help for more information on that (and the View->Server Debug and File->Preview in Browser commands).

Challenges in Getting It Set Up

Many CFML developers remain loyal to CF Studio or have moved to HomeSite+, the latest incarnation of that tool offered with DWMX, as I discussed in my June 2003 ***CFDJ*** article, "Getting into HomeSite+." For the purposes of this article, the two (CF Studio and HomeSite+) are essentially the same, and for the sake of simplicity I'll just refer to them simply as "Studio" in the vernacular common to many developers (and not to be confused, of course, with the Studio MX suite of tools).

I find that many have never enabled/leveraged the Studio internal browsing ability. I think part of the problem is that it's just not obvious how to do it. Indeed, if you hit the Browse button in Studio or press the shortcut (F12), when you've not taken the steps to set up your Studio environment, you'll see an error message that may or may not clarify for you what's wrong: "The ColdFusion file that you are trying to browse cannot be resolved into a URL. Add a server development mapping to enable URL resolution for this file." (See Figure 1.)

The first question may be, what is a "server development mapping?" If you're familiar with Web servers, you may be confused into thinking it has something to do with Web server mappings (it doesn't). Beyond that, how do you set things up properly? Where is it documented? The message doesn't tell you. And when you do find the documentation, you'll also encounter the notion of an "RDS Server." What's that? By the end of this article, you'll understand all of these concepts.

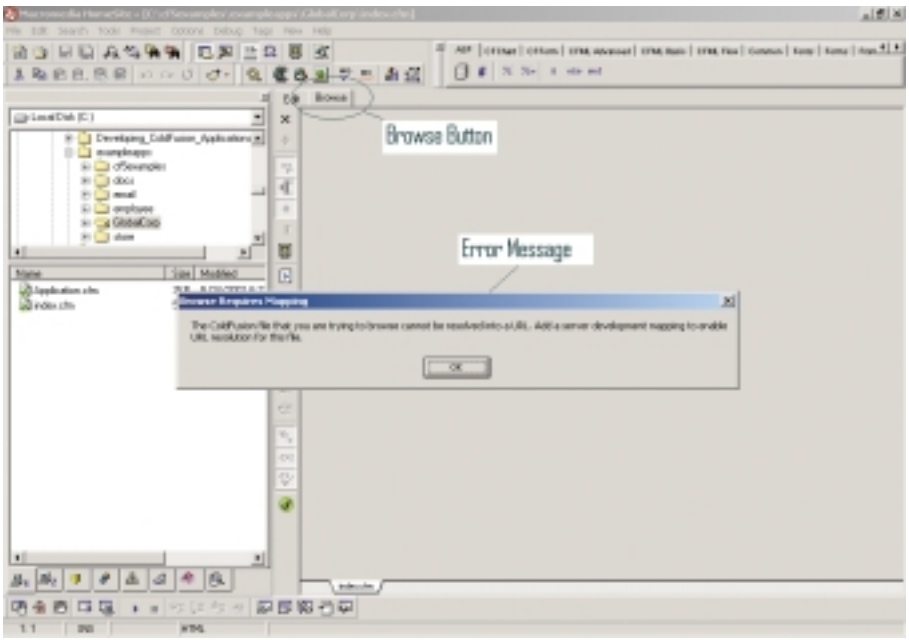


Figure 1: Typical internal browse error message

Understanding Development Mappings

A development mapping is the real key to enabling browsing. It's really quite simple in concept, being just a definition of which URL to use when browsing files in a given directory. You can have many mappings if you have files in different directories, or one mapping for all files and subdirectories under a given one.

The key point is that when you try to browse a template in Studio, it looks at the location (disk and directory path) of the file you're trying to browse and then checks to see if a development mapping has been defined for that directory or any ancestor (parent, grandparent, and so on).

For instance, if the file you're editing is in c:\inetpub\wwwroot\somedir\, then Studio will look first to see if there's a Studio development mapping defined for c:\inetpub\wwwroot\somedir\; if not then it will look for one for c:\inetpub\wwwroot\, and so on. This happens very quickly, of course, much the same way in which ColdFusion looks for an application.cfm file before processing each template.

If there's no mapping for the directory path (or ancestor) of the file being browsed (and note that there are none set up by default), then you are shown the error message above. If there is a mapping, then Studio uses that mapping to determine which URL to use for browsing files in that directory and it will attempt to browse it. (We'll see how such a mapping is defined in the next section.)

This approach of using a mapping means, of course, that it works for any files that are located in a directory ready to be browsed by a Web server. If you put your code in a directory called c:\mycode\ or some subdirectory of that, and if there is a URL that can be used to browse it in that location (such as if you've defined a Web server mapping or virtual directory), then you can define a Studio development mapping for it.

It even works for mapped drives (such as f:\, that may be mapped to a network directory) or a UNC path (such as \\myserver\). All that matters is that a URL can be defined to browse the file in the given directory. If you can type a URL to browse files in a given directory, then you can tell Studio how to map that URL to that directory (and its children).

That's really all that development mapping is. This also shows why any kind of file can be browsed within Studio. It's just a mapping of a directory path to the corresponding URL to browse that directory. You could even create a mapping to your PHP, JSP, and ASP apps, if you wanted to. Indeed, Studio supports color coding and other coding tools for PHP, and even more for JSP and ASP.

The Mapping Editor and RDS Servers

So how do you define these mappings? Let's assume for the moment that you're simply interested in creating a mapping for code running on your localhost machine, or 127.0.0.1. As I just mentioned, a mapping can also be created for files executed on other machines as long as you have a drive mapping or UNC path (or RDS Server) defined on your machine. We just need to declare what URL to use for the given directory of a file we're editing.

In fact, there are several ways to add a mapping. The original one was to use the menu command Options->Settings (or F8) and then choose the Browse link in the folder in the tree control displayed on the left (about halfway down the list). You'd see several fields and buttons, one of them labeled "Development Mappings."

A shortcut for all that is to simply press the shortcut Alt+M (while holding the Alt key, press "m" for mapping).

Still another approach, introduced in Studio 4 along with the internal debugger feature, is the menu command Debug->Development Mappings (and a corresponding button in the Debug toolbar at the bottom of the Studio interface).

Any of these approaches will open a window or editor for development mappings (see Figure 2). The first time you use it, it may appear mostly blank as in the figure, since no development mappings are defined by default upon installation of Studio.

If you look closely, you'll notice that in fact there first is a dropdown at the top of the window labeled "RDS Server." Since in this first example we're setting up browsing against our local machine, we want to be sure that "localhost" is selected. It will be selected by default if no other RDS Servers have been defined (more on those later), because Studio does install a "localhost" RDS Server definition on installation.

If you see anything other than localhost in that dropdown, or think you will use other RDS Servers, see the sidebar "Multiple RDS Servers" for more information.

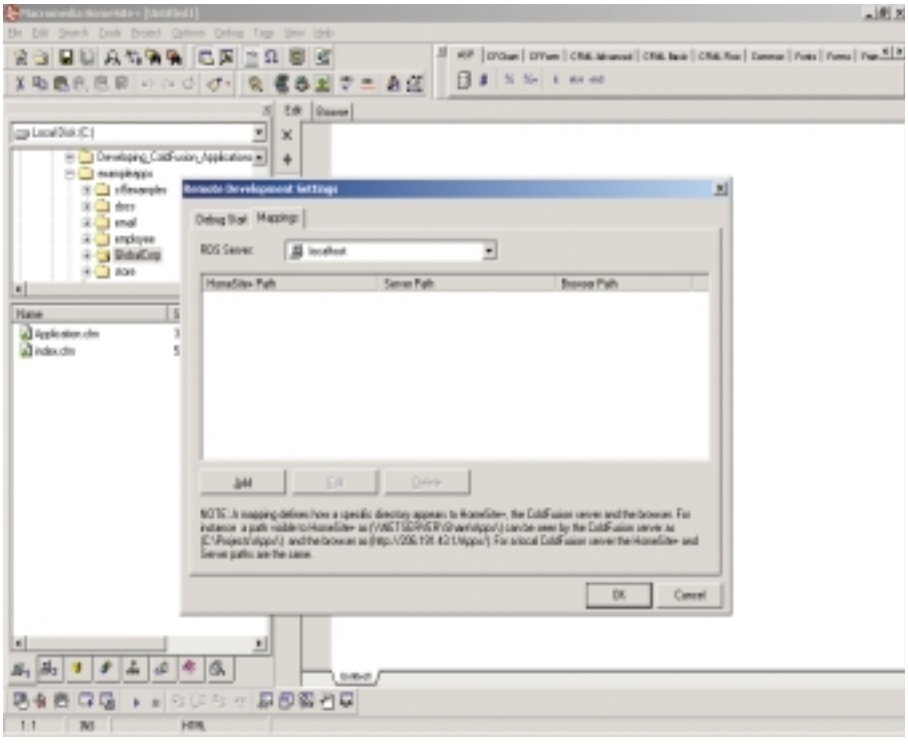


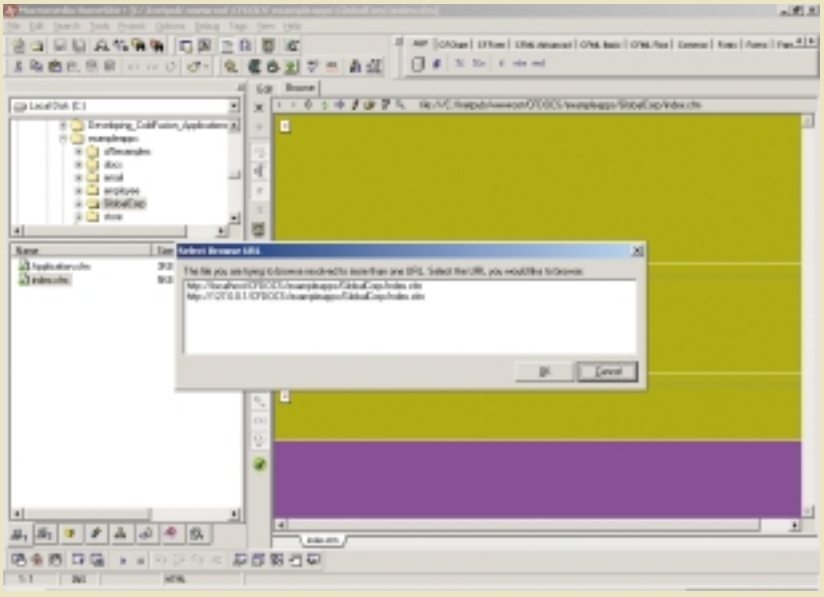
Figure 2: The development mappings editor window

Multiple RDS Servers

If you or anyone else has defined another RDS Server in Studio, be aware that they're shown in alphabetical order in the first dropdown of the Development Mappings editor, and the first server in that order is selected by default when the editor is opened. Any mappings shown (or created) apply to the selected server. Be sure to select the correct RDS Server before creating or reviewing your mappings.

Technically, it's far more critical to select the correct RDS Server when creating mappings for use with the internal debugger than with the internal browser. Indeed, only for debugging and using the Database tab, among other things, is it important that the RDS Server definition even be set up with the correct RDS password, etc. We'll discuss RDS Servers, RDS passwords, and RDS processing more in a later article, but there's help in "Using ColdFusion Studio" or "Using HomeSite+," shown in the Help tab of the Resource Toolbar (Help->Open Help References Window).

For browsing, no RDS processing takes place and all that matters is that a mapping exists for the given URL. In fact, Studio will search through all mappings for all servers to locate a match. (It even prompts you if it encounters more than one mapping among the multiple RDS Servers for a given directory, as shown below.)



When multiple mappings are found for a path

If you never use the internal debugger, the association of mappings to a given specific RDS Server isn't critical just for browsing (since it searches through all of them). Still, for consistency, it may be best to select the RDS Server representing the server where the code being executed is located, and define its mapping there.

So, continuing our example, make sure that localhost is selected in the RDS Server dropdown. If you have any development mappings defined for the selected server, they will be shown in the lower area of the mapping editor window. Note the available Add button to create a new mapping, as well as the Edit and Delete buttons for managing any existing ones.

Adding a Development Mapping

To add a new development mapping, select the Add button, and a new window is opened (see Figure 3). (If you're using Studio 4.5 or earlier, you don't need to click Add to start the process.) You'll be prompted with three empty fields. For purposes of browsing, only the first and third fields are significant.

The first field, labeled "Studio Path" (or "HomeSite+ Path" in HomeSite+), is another source of frequent confusion. Newcomers may be inclined to interpret this as being the path where Studio/HomeSite+ is located. In Studio 5 and HomeSite+, the prompting associated with these fields has improved, so if you read the screen it should be less of an issue.

Anyway, from what we've discussed so far, it should be clearer now that the Studio/HomeSite+ Path is simply this: the path to the directory where your files that you will want to browse from within Studio are located.

Referring back to our example of code in c:\inetpub\wwwroot\somedir\, we might just simply name that directory as the path here. But if you think about it, we're going to next declare (in the third field) the URL to use when browsing files in that path. Let's assume that the URL would simply begin http://localhost/somedir/, then clearly the /somedir is just a subdirectory indicator. As we'll see in a moment, we don't need to define a mapping for this specific subdirectory but instead for the Webroot above it.

So we should fill in c:\inetpub\wwwroot\ in the first field. Notice that there's an available Browse button (a yellow folder icon) to the right of the field, which allows you to use a tree-like directory explorer interface to locate and select the desired path.

The second field, the "CF Server Path," is yet another source of confusion. It's really relevant only to using the internal debugger. Since that works only with CF 4, 4.5, and 5, if you're not using any of those to execute your code, you can safely leave the field blank. Debugging is not enabled in ColdFusion MX (nor currently in BlueDragon). To learn more about the debugger, see the Studio/HomeSite+ manual referenced in the sidebar.

The third and critical field is the "Browser Path," or simply the URL needed to browse templates found in the first field's path. If we've defined the first field to hold the Webroot directory, then we would specify here the URL needed to browse files in that Webroot. In our example, we'd put in http://localhost (or

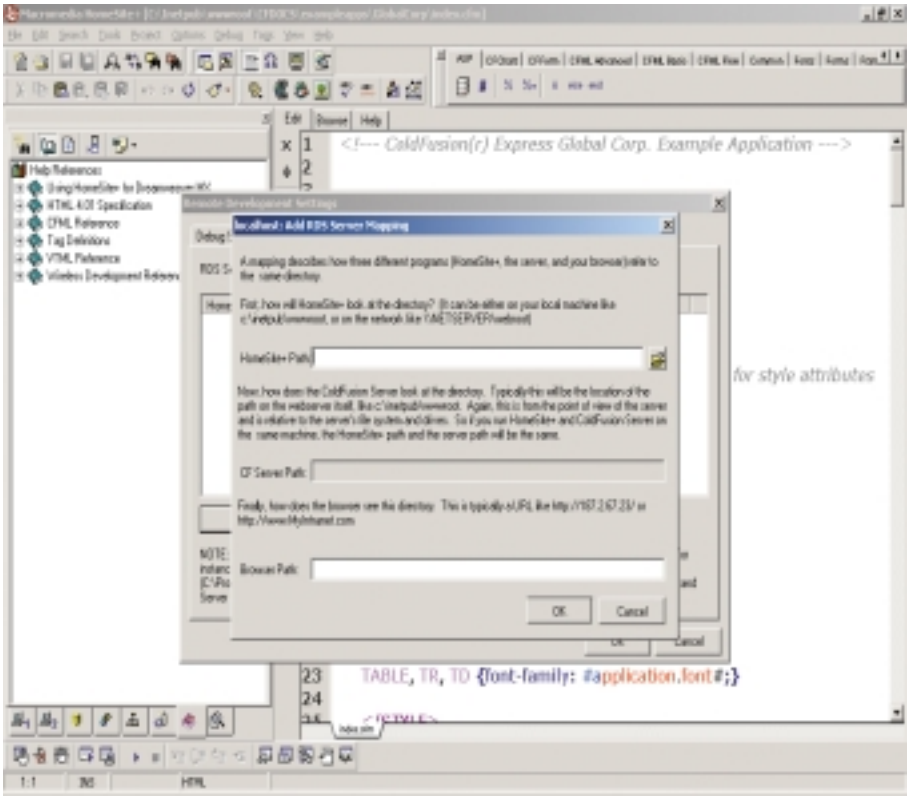


Figure 3: Adding a development mapping

http://127.0.0.1, if the difference is significant on your machine). If a port is needed, such as when using the built-in Web server in CFMX or BlueDragon, provide that on the URL, such as http://localhost:8500 for CFMX or http://localhost:8080/ for BlueDragon.

As stated, we can specify just one mapping for all templates and subdirectories under a given path. If a file is browsed in a subdirectory, Studio will properly determine the extra path information needed relative to the first path specified and will add that extra path information (such as "/somedir") to the URL when the template is browsed.

That's it! You've now defined a development mapping for all files in your localhost Webroot. As I said, it's really simple once it's been explained. You're probably annoyed that you never learned about it (or learned all the details) before. Naturally, if you have multiple directories such as virtual directories or aliases in your Web server pointing to code that's outside your Webroot, you'll want to create mappings for each of those.

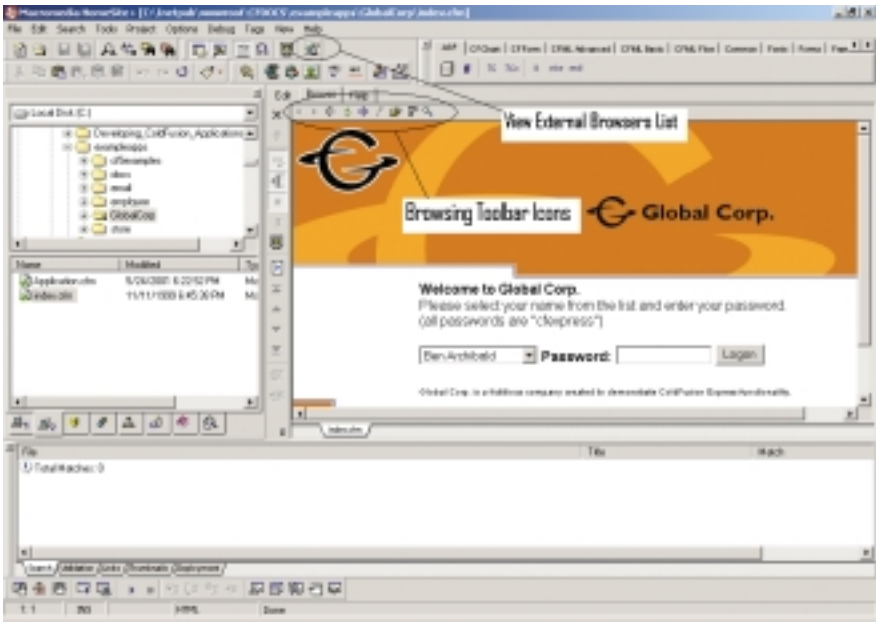


Figure 4: Browse toolbar icons and view external browsers list

And if you ever have trouble determining what the "Studio path" should be for a given file (especially if you have a file open via a drive mapping, UNC path, or RDS Server), simply open the file to be browsed within Studio and look at the title bar. Studio shows the file path up there. That's the path to be put in the Studio path (minus the filename, of course, and perhaps leaving off some subdirectories to create a more general mapping).

It would be nice if Studio had a mechanism to capture the current filepath of an opened document when creating a new mapping. (You might think it would be nice to have Studio come preconfigured with a mapping for the Webroot, but the Webroot path would depend on which Web server you're using and there's no way for Studio to know.)

Before concluding, let me point out one more annoyance for those of you still working with Studio 4.5 or earlier. The interface for adding a mapping is just slightly different and a possible source of frustration (thus its improvement in Studio 5 and beyond). Instead of an Add button opening a new window in which to define a mapping, you are shown the three blank fields for either editing existing mappings (if selected) or adding a new one. But you must remember to click the Add button after typing in the mapping fields. If you press "OK", the editor window goes away and your mapping is not added. Untold woe has befallen those who didn't realize that and wondered why their mappings weren't working. Studio 5 and beyond prevents the confusion by opening a new window for adding/editing mappings.

Browsing Your Goods

You can now try your spanking new development mappings to use the internal browser. Open a file in Studio located in your Webroot (or whatever directory you named in the mapping). Cross your fingers, click the Browse tab (highlighted in Figure 1) or press F12.

If you've set up everything correctly, your page should be executed and the edit area where your source code was shown should be replaced with a browsed version of the page. Woo-hoo! Life as a developer should be much easier now.

You can hit F12 again (or the Browse tab) to return to the editor, edit some code, hit F12 again, and so on. No more need to have an external browser open and flip back and forth. There are just a couple of final points to keep in mind.

First, if you get a 404 or some other indication that the URL is not correct, you can return to the development mappings editor and edit whatever mapping you just added. Also note that when using the internal browser, it may be important to refresh your document after changing it since the internal browser may show a cached version. The default internal browser is Internet Explorer, and Studio's internal browser will respect whatever setting you have in your real IE browser (the two are the same). There are several possible settings in IE for browser caching. In IE 6, for instance, see Tools->Internet Options->Temporary Internet Files->Settings. The current setting may cause Studio to not show the latest version of output from a template you've just edited.

The good news is that you can (and usually should) simply refresh the display. One way is to right-click in the browsed output (not on a graphic or form field) and choose "refresh" from the list of options – the same ones you'd see when right-clicking the output in your regular IE browser. Note that you can also right-click on a browsed document and choose "view source" to be able to see the underlying HTML of a browsed page.

I should add that there are indeed times when it may still make more sense to browse a file externally (meaning, not in the internal browser). If you're editing the action page of a form, you wouldn't typically want to browse that file. You'd want to browse the form instead, and if you're meaning to edit the action page and see the results of your changes, you'd need to save the changes, switch to the form, and submit it. Then if you needed to change it, you'd have to return to the editor, switch to the action page, and repeat. That's clumsy.

It would be smarter to open the form in an external browser, submit the form, and view the action page results in the browser. Then if you need to edit the action page, do so in Studio and save the changes. Go back to the external browser and simply hit the "refresh" button (Ctrl-R) while viewing that action page result. The browser will ask if you want to "resend" the form data. Click "OK" or "retry" and you'll see the results of your changes. Switch back to Studio and continue editing. That's much easier than trying to work in the internal browser.

Another case when it may make sense to use an external browser is when you want to pass query string or URL values into a page being browsed, though there is, in fact, a way to do that in Studio. When browsing a file in the internal browser, you'll notice a series of buttons atop the browsing area, with little icons that give

access to several interesting functions (see Figure 4). Explore these icons on your own. One of them is "open url" (the folder icon) and another is "refresh". Still, it's sometimes just easier to open your template in an external browser.

There's one more reason you may prefer to use an external browser, if you don't care to launch your templates in Internet Explorer: the default internal browser. There is, in fact, a way to change what the internal browser should be. See Options->Settings->Browse, and the Studio documentation, for more help.

Launching an External Browser

You don't need to do it manually. Studio can launch the external browser for you. Launching your template in an external browser can be as powerful and useful (if not more so) than internal browsing, and it's certainly a time saver.

If you had any external browsers already installed when Studio was installed, you'll find that you can ask Studio to launch a template in an external browser. Instead of clicking the Browse tab or F12, click the F11 key. That will launch your "default" external browser, at least the "default" as far as Studio is concerned. The browser will open and the URL for the given template will be populated automatically and browsed. Cool!

To configure which browser is your default, or indeed to make Studio aware of one that may have been installed after Studio, see Options->Configure External Browsers (see Figure 5). Note that you can put them in order placing the desired one first, which controls which is launched by F11.

Besides using F11, there is also an available "Open in External Browser" icon in the Browse toolbar icons (the last one, a magnifying glass, as shown in Figure 4). And there is also a more powerful "View External Browser List" button at the top right of the toolbar buttons under the menu commands. Again see Figure 4. This option will give you a choice of available browsers if multiples are installed.

Note also that even if you're not interested in external browsing, there is one more setting to be aware of, and it's set (perhaps curiously) in that Configure External Browser window (Figure 5). Notice that it offers a choice for how Studio should react when you try to browse a file whose changes you haven't saved. By default, it will try to save a temporary file. For dynamic files like CFML, it's best to change that to either prompt you to save it or automatically save it. Browsing dynamic pages with a temporary name can get confusing.

Just be aware that the same issue applies to browsing internally. Unfortunately, you have to know how to get Studio to recognize that setting for internal browsing as well. Notice in Options->Settings->Browse (not shown), there is an option "use external browser configuration for internal browser." That's what this checkbox is there for. You'll want it checked (though I think the default behavior may have changed in Studio 5 or HomeSite+, as my HomeSite+ shows it not checked but it is not saving as temporary files. Perhaps this setting isn't as critical anymore).

Conclusion

That's a whirlwind tour of browsing files internally and externally in CF

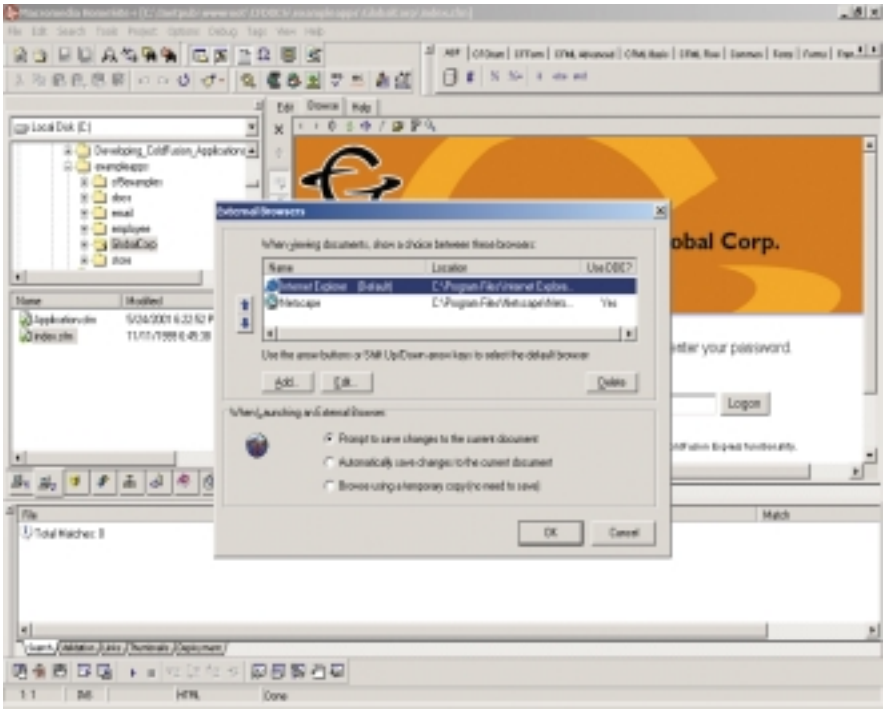


Figure 5: Options->Configure External Browsers

Studio/HomeSite+. It's a powerful feature, misunderstood by many. I hope that these tips help save you lots of time. Let us know. You can comment on any article by visiting the ***CFDJ*** Web site, at www.sys-con.com/coldfusion. View the article in the archives and enter comments at the bottom of the article page. Or drop me a line. I'd love to hear how my articles have helped you.

About the Author

Charlie Arehart is co-technical editor of **ColdFusion Developer's Journal** *and a Macromedia Certified Advanced ColdFusion developer and trainer. He has recently become CTO of New Atlanta Communications, makers of BlueDragon. In his new role, he will continue to support the CFML community, contributing to several CF resources, and speaking frequently at user groups throughout the country.*
charlie@newatlanta.com

cf community *—continued from page 7*

UNIONs a recordset containing the sum of the quantitative column, "0" for the quantitative column in the second table (because the column doesn't exist), and the year all from the first table, with a recordset that contains the sum of the quantitative column, "0" for the quantitative column in the first table (because the column doesn't exist), and the year from the second table. These zeros are selected in order to be able to UNION the two queries. Zero is used so that the actual SUM value that is selected from this UNIONed recordset is not affected.

I-Lin Kuo then added that implementing John Hatcher's suggestion of using an outer join is most efficient because they have a tendency to perform better than UNIONs on most databases (most likely because you're retrieving less data). The final solution would then look like:

```
select totalRain, totalDischarge, r.year
from (select sum(raincm) as totalRain, year from rainTable group by year) as
r
  full outer join
  (select sum(dischargecm) as totalDischarge, year from dischargeTable group
by year) as d on r.year = d.year
```

And there you have it. How to select the year along with total quantities for different columns in different tables, all grouped by year, in a single recordset. This thread not only showed a very creative solution to a problem, but also helped many of the List members to better understand more advanced key database concepts. It also reinforced the fact that there are usually many different ways to arrive at the same result, and that you should try to make your database do the work for you whenever possible.

Extending ColdFusion with Java

Writing a Java-based CFX tag

Last month I introduced you to Lucene and then showed you how you can access the Jakarta Lucene APIs using native CFML scripting syntax. This month I'm going to show you how easy it is to create a Java CFX tag that performs the same functions as the two tags I illustrated last month: <cf_luceneindex> and <cf_lucenesearch>.

As a short review of last month and as many of you know already, one of the many reasons to use ColdFusion MX is that it comes standard with the majority of the tools you'll need to write full-featured, dynamic Web applications. Tags like <cfquery> and <cfmail> make it relatively simple to query a relational database and send e-mail. In the same way, you can use <cfsearch> and <cfindex> to create and search Verity full text indexes.

There are, however, a couple of situations when you can't use the full text searching capabilities of Verity. The ability to run ColdFusion MX on the Apple OS X operating system, while a boon to developers who code on the Apple platform, does not include the ability to use Verity. Programmers who work in a hybrid J2EE/ColdFusion MX environment (possibly using ColdFusion MX for J2EE) cannot natively use the Verity search capabilities in the J2EE environment. Finally, programmers who need customized searching and indexing capabilities may find the standard Verity integration limiting.

Enter Lucene, an open source full-text searching framework from the Apache Jakarta project, which, when combined with ColdFusion MX, can be run on Apple OS X, programmatically accessed by both J2EE and ColdFusion MX developers, and fully customized and extended. Additionally, when accessed with the Java CFX API, Lucene can be utilized in ColdFusion 5, 4.x, and possibly earlier versions.

In this article, I'll discuss CFX tags in general, walk through the CFX API, and then jump into the creation of two Java classes that combine the CFX and Lucene APIs into two CFX tags: <cfx_luceneindex> and <cfx_lucenesearch>. Finally, I'll conclude with some tricks and hints to help you when writing CFX tags in Java.

This article is not intended to be an in-depth introduction to the Lucene API, to Java programming, or to writing C++ CFX tags. If you're interested in learning more about the internal workings of Lucene, one of SYS-CON's sister publications, ***Java Developer's Journal***, featured an article entitled, "Search-Enable Your Application with Lucene," see the resources section at the end of this article.

If you're new to Java, I'd suggest heading to your local bookstore to pick up one of the many Java books for beginners. Finally, if you're looking for information on writing C++ CFX tags, check out the CFX C++ documentation link, also in the resources section at the end of this article.

While I'll try to cover the basics of writing a Java CFX tag, if you'd like more information about Java and C++ ColdFusion CFX tags, I encourage you to check out the "Building Custom CFXAPI Tags" resource in the CFMX documentation (see resources section for the URL); included in the documentation are sample CFX tags for browsing ZIP file archives, creating JPEG images, and using sockets.

By Aaron Johnson

What Are CFX Tags?

If you've worked with ColdFusion for any length of time, you know that ColdFusion provides almost everything (if not everything!) you need to rapidly build dynamic Web applications. Sooner or later you'll come to a situation where you need something that ColdFusion doesn't provide out of the box. But don't fret! With some knowledge of Java or C++, you can create a CFX tag that wraps up the extra functionality you need into an easy-to-use CFX that you can then access using the standard tag-based syntax. In short then, a CFX tag is a compiled Java or C++ module that performs a task that ColdFusion doesn't natively offer.

Setup and Configuration

Before beginning, you'll need to make sure that your system (be it Unix, Linux, Windows, or Apple) is appropriately configured:

- ColdFusion must be installed. You'll see that I'm using the ColdFusion MX integrated Web server running on port 8500 throughout the examples.
- You'll need to download Lucene; binaries and source code are available on the Jakarta Apache site; you'll see links to those resources at the end of this document.
- After downloading the Lucene JAR file, add the location of the JAR file to the classpath in ColdFusion Administrator (http://localhost:8500/cfide/administrator/, click on "Java

and JVM" under "Server Settings", and type the full path to the location of the Lucene JAR file you downloaded). Make sure that you restart the ColdFusion service after saving your changes.
- Create a folder in your /cfusionmx/wwwroot/ called "cfxlucene", into which you'll put the source code written during this article.
- To compile the source code I've provided (sys-con.com/coldfusion/sourcec.cfm), you'll need to have the Java 2 SDK installed on your system; I'm using J2SDK 1.4.1.
- Finally, locate the "cfx.jar" file on your system. On my system with the default CFMX installation, the "cfx.jar" file is located at "C:\CFusionMX\lib\cfx.jar".

Let's get started!

Creating a Lucene Index Using the CFX API and Java

If you've completed the setup and configuration steps above, open your favorite Java IDE (Notepad, VI, Eclipse, or even CF Studio all work just fine), create a new document, and save the document as "LuceneIndex.java". This file will contain the source code for the CFX tag that will perform the indexing of documents later in this article. The first items you'll need to add are the following Java import statements:

```
import java.io.*;
import java.util.*;
import com.allaire.cfx.*;
import org.apache.lucene.analysis.*;
import org.apache.lucene.document.*;
import org.apache.lucene.index.*;
```

The lines of code above "import" the java.io and java.util libraries, the com.allaire.cfx library (which I'll explain in a bit), and three Lucene libraries – all of which you'll use when writing the LuceneIndex class. If you're new to Java, you can think of the previous import statements as doing something analogous to using <cfinclude> to include a file that contains one or more user-defined functions. Basically, I've just imported a number of Java functions (although in Java you only have "methods" on "objects", not "functions") into the LuceneIndex class.

Next, you'll need to add the following line:

```
public class LuceneIndex implements CustomTag {
```

Without going into too much detail, this line of code says that the name of the class is "LuceneIndex" (the name of the file must match this name) and that it "implements" CustomTag. The word *implements* has a special meaning in Java; when you implement something, it means that the class you are writing will have a method with the same signature as each method in the class you are implementing (which is called an "interface"). In this case, I'm implementing the "CustomTag" interface, which has only one method:

```
public void processRequest(Request req, Response res)
```

Now that you understand interfaces (don't worry if you don't, interfaces and the "implements" keyword aren't that important to this article), you should know what the next line will be:

```
public void processRequest(Request req, Response res) {
```

Look familiar? It should! This is the method from the CustomTag interface that you must implement. Let's look a bit closer at it. There are two classes passed to this method: Request and Response. If you've done any work with ASP/VBScript or JSP, you might think that you understand these classes right away, but they aren't the same. We'll look at the Response class in the next section. The Request class contains seven methods, none of which deal with the "request" made by the browser (unlike the ASP request object or the JSP request object). I'm only going to cover two in this article: attributeExists() and getAttribute(), but if you're hungry for the details, you can read about all the methods by visiting the ColdFusion Java CFX reference resource at the end of this article. The method "getAttribute()" retrieves the value of the variable given as an attribute to the method. So, for example, if I wrote this in ColdFusion:

```
<cfx_luceneindex path="c:\cfusionmx\wwwroot\cfdocs\">
```

and I wanted to retrieve the value of the attribute "path" in the Java code, I'd write this:

```
req.getAttribute("path")
```

Simple right? In the same way, if I wanted to check that a specific attribute has been passed to the tag, I'd use the attributeExists() method. For instance, if I wanted to make sure that the attribute "path" has been passed to the Java code, I'd use a statement like this:

```
req.attributeExists("path")
```

Back to the LuceneIndex class. In order to index a directory or a file, you'll need to know which directory or file, where to create the index, whether to recurse the directories, and so on. To do so, you'll add the following lines of code:

```
String indexName = req.getAttribute("indexName");
String indexPath = req.getAttribute("indexPath");
String directory = req.getAttribute("directory");
String file = req.getAttribute("file");
String urlpath = req.getAttribute("urlpath");
```

For the sake of brevity, I'm not covering the error checking code, which you can see in the source code that you can (and should!) download (www.sys-con.com/coldfusion/sourcec.cfm). Next, add a statement that checks to see if the user wants to "optimize" a Lucene index or "index" a directory of files:

```
if (action.equalsIgnoreCase("index")) {
```

Finally, you're ready to begin using Lucene to index a directory.

```
Analyzer analyzer = new StopAnalyzer();
IndexWriter writer;
try {
          writer = new IndexWriter(indexPath, analyzer,
                   bCreateIndex.booleanValue());
          if (req.attributeExists("directory")) {
                    File f = new File(directory);
```

```
        if (f.exists() && f.isDirectory()) {
                        indexDirectory(urlpath, f,
                        recursive.booleanValue(), writer);
                }
    } else {
                File f = new File(file);
                indexFile(urlpath, f, writer);
    }
        writer.close();
} catch (java.io.IOException e) {
        errors.add(e.toString());
}
```

Start by creating a Lucene Analyzer (in this case using the StopAnalyzer) object and a Lucene IndexWriter object, which are then used in the index process. You'll notice that the IndexWriter constructor and the indexFile()/indexDirectory() methods are wrapped in a try/catch block so that errors resulting from insufficient permissions and nonexistent directories/files are caught appropriately. I won't discuss indexFile() and indexDirectory() methods as they aren't pertinent to the topic at hand; you can find the source code for both methods in the LuceneIndex.java file.

To optimize a Lucene index, add these short bits of code:

```
StopAnalyzer analyzer = new org.apache.lucene.analysis.StopAnalyzer();
IndexWriter writer;
try {
        writer = new IndexWriter(indexPath, analyzer, false);
    writer.optimize();
    } catch (java.io.IOException ex) {
        ex.toString();
}
```

At this point, you're ready to save and compile LuceneIndex.java. Since I haven't included all the necessary code, I'd suggest that you download the source code and unzip the archive to

```
[cfusionhome]/wwwroot/WEB-INF/classes/
```

where [cfusionhome] is the directory that contains your ColdFusion MX installation. After completing that (and keeping in mind the location of both the cfx.jar and the lucene.jar file that you downloaded/located during the previous setup and configuration steps), you can compile LuceneIndex by bringing up a command prompt and typing the following from the [cfusionhome]\wwwroot\WEB-INF\classes\ directory:

```
C:\CFusionMX\wwwroot\WEB-INF\classes>javac -classpath
[path_to_lucene.jar];[path_to_cfx.jar] LuceneIndex.java
```

substituting the appropriate values for [path_to_lucene.jar] and [path_to_cfx.jar] like I did in Figure 1.

After successfully compiling LuceneIndex.java, you should have a file called "LuceneIndex.class" in the [cfusionhome]\wwwroot\WEB-INF\classes\ directory. In order to test out the code you've written so far, you need to make ColdFusion aware of its existence. To do that, log in to the ColdFusion Administrator, click on "CFX Tags" under "Extensions," and then click the "Register

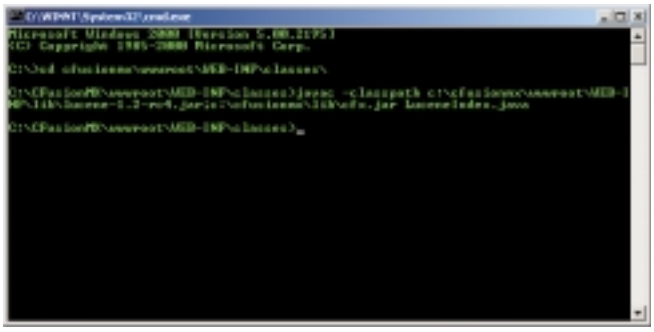


Figure 1: Compiling LuceneIndex.java from the command line

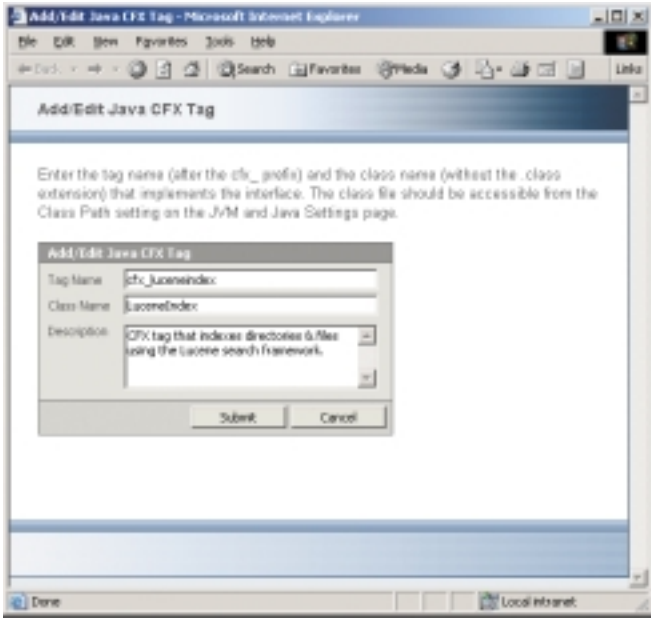


Figure 2: Adding a CFX tag to ColdFusion Administrator

Java CFX" button. Enter the tag name (in this case "cfx_luceneindex"), the class name ("LuceneIndex"), and optionally, a description just like I have in Figure 2.

You can then test the indexing and optimizing operations by running a CFML script that contains the following code:

```
<cfx_luceneindex
        action="index"
        indexpath="c:\cfusionmx\wwwroot\cfxlucene\cfdocsindex\"
        bCreateIndex="true"
        directory="c:\cfusionmx\wwwroot\cfdocs\CFML_Reference\"
        urlpath="http://localhost:8500/cfdocs/CFML_Reference/"
        recursive="true">
```

again, substituting the appropriate values for your environment. I included a sample script in the source code called "test_luceneindex.cfm" that contains code for indexing a file, indexing a directory, and optimizing an index. All set to search that index? Great! Let's move to the next section…

—continued on page 38

Searching a Lucene Index Using the CFX API and Java

As in the previous section, start by creating a new file using your IDE; save this one as "LuceneSearch.java". I'll start off the new class by importing the appropriate libraries, just as I did in the previous example:

```
   import java.util.*;
import com.allaire.cfx.*;
import org.apache.lucene.analysis.*;
import org.apache.lucene.document.*;
import org.apache.lucene.index.*;
import org.apache.lucene.queryParser.*;
import org.apache.lucene.search.*;
```

Not much different there, you can move quickly to the next lines:

```
public class LuceneSearch implements CustomTag {
   public void processRequest(Request req, Response res) {
```

Again, use the "implements" keyword in the class definition, and then immediately add a method "processRequest()" that matches the method signature of the processRequest() method from the CustomTag class. After that use the req.getAttribute() method to retrieve the value of the attributes given in the tag:

```
   String indexPath = req.getAttribute("index");
String queryString = req.getAttribute("keyword");
   String r_query = req.getAttribute("r_query");
```

Perform some error checking (again removed for the sake of brevity) and then you're into the heart of the tag:

```
   IndexSearcher searcher = null;
org.apache.lucene.search.Query luceneQuery = null;
Hits hits = null;
searcher = new IndexSearcher(IndexReader.open(indexPath) );
Analyzer analyzer = new StopAnalyzer();
luceneQuery = QueryParser.parse(queryString, "body", analyzer); //parse
the
hits = searcher.search(luceneQuery);

String[] columns = { "URL", "TITLE", "SUMMARY" } ;
com.allaire.cfx.Query q = res.addQuery(r_query, columns) ;

for (int i = 0; i < hits.length(); i++) {

   Document doc = hits.doc(i);
          String doctitle = doc.get("title");
   String url = doc.get("url");
          String docSummary = doc.get("summary");

   int iRow = q.addRow() ;
          q.setData( iRow, 1, url) ;
   q.setData( iRow, 2, doctitle);
   q.setData( iRow, 3, docSummary);
}
```

This block of code creates an IndexSearcher object, a Lucene Query object, a Lucene Hits object, and an Analyzer object, all of which are used in various ways to parse the keyword into manageable bits and then search the index you created in the above example.

Remember the Request and Response classes I mentioned? In this example you're going to use the Response object, which has four methods, only three of which you'll probably use on a regular basis. The addQuery() method enables you to return a ColdFusion query to the calling template, which is what you'll do in this example. Additionally, you can use the setVariable() method just like you might use the SetVariable function in ColdFusion – for example, to return a string or a number to the calling template. Finally, you can use the write() method to write text directly to the screen, just as you might use

```
<cfoutput>Hello World!</cfoutput>
```

or

```
   WriteOutput("Hello World!");
```

when writing CFML code. Back to the last example, the portion of code that is pertinent to this article starts with these lines:

```
String[] columns = { "URL", "TITLE", "SUMMARY" } ;
com.allaire.cfx.Query q = res.addQuery(r_query, columns) ;
```

The first line creates an array of String objects: "URL", "TITLE", and "SUMMARY"; each represents the name of a column in a query. This array is passed to the method addQuery() of the Response object, along with the "r_query" value. After creating the query, loop over each "hit" that Lucene returned, using a "for" loop, and then add the URL, the TITLE, and the SUMMARY to the query:

```
   q.setData( iRow, 1, url) ;
   q.setData( iRow, 2, doctitle);
   q.setData( iRow, 3, docSummary);
```

where iRow is the row number and 1, 2, and 3 represent the number of the column you're adding a value to.

I left out the error checking and try/catch statements so if you want to compile the code, again I'd suggest that you download and unzip the archive to the [cfusionhome]\wwwroot\WEB-INF\classes\ directory and then compile LuceneSearch.java by typing this from the command line:

```
   cd [cfusionhome]\wwwroot\WEB-INF\classes
   javac -classpath [path_to_lucene];[path_to_cfx] LuceneSearch.java
```

substituting the appropriate values for [cfusionhome], [path_to_lucene], and [path_to_cfx]. After successfully compiling the Java class, you should register the CFX tag in the ColdFusion Administrator (i.e., click on CFX Tags under Extensions and click the "Register Java CFX" button, tag name should be "cfx_lucenesearch", tag class should be "LuceneSearch"). You should be able

to test the search functionality using the included "test_lucenesearch.cfm" script in the source code archive, or by creating a simple script that looks like this:

```
<cfx_lucenesearch
r_query="r_query"
index="c:\cfusionmx\wwwroot\cfxlucene\cfdocsindex\"
keyword="software">
```

Tips and Tricks

As with any technology, there are things you need to watch out for. Here are a couple I found while working with Java CFX tags:

- When using the Response object's write() method, the text is not written to the browser if you have <cfsetting enablecfoutputonly="true">. If you want to see error messages written by the tag or if the tag you've written uses the write() method, you must either (a) surround the tag invocation with a <cfoutput></cfoutput> block or (b) not use the <cfsetting enablecfoutputonly="true"> tag above the CFX tag invocation.
- During development, you can use the DebugRequest, DebugResponse, and DebuqQuery objects to emulate your Java class being called from ColdFusion as a tag. The Building Custom CFXAPI Tags section of the documentation has directions on setting up your Java IDE for interactive debugging as well as an example of how you might use those classes. The LuceneIndex.java file in the source code archive also provides an example of the use of those classes.
- According to the CFX Java tag documentation, changes to the class files located in the [cfusionhome]/wwwroot/WEB-INF/classes are automatically reloaded upon being changed. I did not observe this behavior; to view the change in the class file, I had to restart the ColdFusion MX service using the Services control panel applet.
- When using a CFX Java tag in CFML, there is a special attribute called "reload" that supposedly controls whether or not ColdFusion will reload changes to the class file. The documentation states that by default the reload attribute is set to "Auto" and recommends that during development you set the reload attribute to "Always"; set it to "Never" to increase performance.

Conclusion

The Java CFX Tag API enables developers to extend ColdFusion almost anywhere Java can travel. I hope that this article has given you an excellent introduction to writing CFX tags in Java and gets you rolling on your own Java CFX projects. I'd encourage you to download and explore the source code (www.sys-con.com/coldfusion/sourcec.cfm) in which I've included error checking code, try/catch blocks, and attribute validation, and to explore Lucene further.

Resources

- *Jakarta Lucene:* http://jakarta.apache.org/lucene/
- *Jakarta Lucene Downloads:* http://jakarta.apache.org/builds/jakartalucene/release/v1.2/
- *"Search-Enable Your Application with Lucene:"* www.sys-con.com/java/article.cfm?id=1777
- *ColdFusion MX Documentation:* http://livedocs.macromedia.com/cfmxdocs/
- *ColdFusion MX Documentation: Building Custom CFXAPI Tags:* http://livedocs.macromedia.com/cfmxdocs/Developing_ColdFusion_MX_Applications_with_CFML/CFXTags.jsp
- *ColdFusion MX Documentation: Developing CFX tags in C++:* http://livedocs.macromedia.com/cfmxdocs/Developing_ColdFusion_MX_Applications_with_CFML/CFXTags7.jsp#1112325
- *ColdFusion MX Documentation:* ColdFusion Java CFX Reference: http://livedocs.macromedia.com/cfmxdocs/CFML_Reference/CFXRef_Java.jsp

About the Author

Aaron Johnson is a senior software architect for Mindseye, Inc., and has been developing large-scale Web sites for companies like FAO Schwarz, FootJoy, and Macromedia using ColdFusion, ASP, C#, and Java since 1996. He is a Certified ColdFusion Developer and a Microsoft Certified Systems Engineer. You can find out more about Aaron via his blog (http://cephas.net/blog/).

ajohnson@cephas.net

LayerIT Content Management System from LayerIT AS

An impressive, price-busting initial release from 'across the pond'

After evaluating and deploying a number of content management systems over the past five years, I guess I've become a bit jaded. Frankly, they all just started to look the same. To be sure, each one has core strengths and weaknesses. Each is usually geared to fit a certain vertical market (education, for example), and many have similar levels of functionality. All of them have significant usability issues – particularly when authoring content that requires HTML tables.

Reviewed By
Steve Drucker

So when I was asked to write a review of LayerIT CMS (priced at $2,695 for the enterprise edition), I expected to find another run-of-the-mill low-end system that wrapped a minor bit of code around a third-party WYSIWYG editor. What I discovered was an innovative, technically sophisticated, easy-to-use content authoring system that may someday be a real force to be reckoned with.

What Is a Web Content Management System?

A good CMS enforces business rules and facilitates the addition and editing of content on a Web site. Typical features include the ability to modify content through a Web browser based on a series of templates, full-text searching, approval workflow, security, data versioning, and library services. The goals of deploying a CMS include, but are not limited to, pushing the task of Web site authoring down to the original content authors, ensuring consistency in the site branding through the use of templates, keeping content up-to-date, and making content easier to locate for both contributors and end users. Oddly enough, price does not necessarily correspond to features or quality in this particular market segment of computing. Frequently, less expensive products yield better results than those positioned for the "enterprise."

To Boldly Go Where No CMS Has Gone Before

One of the most common complaints I hear from CMS content contributors regards the overall usability of their systems. In particular, users have a difficult time with content layout – largely due to the morass of table-based positioning. LayerIT eschews the traditional CMS authoring model of using table-based formatting in favor of the long-ratified W3C CSS positioning standards. Content is generated following XHTML guidelines and wrapped by tags that stipulate x, y, and z coordinates on a page.

As depicted in Figure 1, positioning content is a simple matter of using LayerIT's drag-and-drop DHTML-based interface. I found the content creation experience to be not unlike using Microsoft Publisher, whereas most other Web CM systems follow a Microsoft Word–centric approach to authoring and placement.

Worship at the Altar of Dan Steinman

Dan Steinman is an independent software developer who created a cross-browser compatible abstraction layer for JavaScript and Dynamic HTML. His open-source DynAPI project (www.danstein man.com/dynapi) set the gold standard for demonstrating how to implement complex UI constructs (including drag and drop) across different browser implementations. LayerIT wisely chose to take advantage of Dan's work. As a result, content may be inserted and modified in LayerIT through Microsoft Internet Explorer 5.x+ as well as Netscape 7/Mozilla.

Many facets of the authoring environment even functioned correctly in Netscape 4! Entering WYSIWYG content, however, remains browser-specific as LayerIT utilizes the SiteObjects editor (www.siteobjects.com), which does not support Netscape. Content generated from LayerIT displayed identically in all browsers tested.

Preventing Use of the <BLINK> Tag

Templates are created in LayerIT using the same authoring mechanism that's used to create content. No programming is required. It uses a metaphor that's usually found only in more expensive content management systems – treating templates

as virtual transparency layers. If any template "layer" is modified, all pages that include that layer will reflect the change. This design is much more flexible and maintainable than using the single, monolithic template approach found in most CMSs. It also ensures that specific aspects of a site's branding will always remain intact, despite the desires of the amateur "designers" on your staff.

WISINWIW – What I See Is Not What I Want

As previously mentioned, LayerIT uses the SiteObjects Lite 2.5 editor (soEditor Lite 2.5). The integration of this component into their overall framework seems far from complete, however. There is no way to restrict access to WYSIWYG features for "challenged" users. Furthermore, I was shocked when my attempt to insert an embedded image into the WYSIWYG field was met with a generic image dialog box instead of a popup referencing the LayerIT asset repository. Furthermore, the editor is completely unaware of CMS-based pages when attempting to create a hyperlink between content.

Repeat After Me: 'It Wasn't Me…I Didn't Do It'

My evaluation version did not contain any versioning features; however, LayerIT has implemented this functionality for their next release. Planned functionality includes the ability to preview the current work-in-progress page, published page, and all prior versions of a page.

'If the Code Don't Fit, You Must Decrypt!'

Source code for LayerIT is encrypted; however, they provide an extensibility model for adding custom ColdFusion code. You can drop a reference to a custom ColdFusion module anywhere on a page. The output from the script will render within LayerIT's generated tags. In addition, you can embed a <CFHTTP> call to "scrape" content off of another HTTP URL. They also include a few "portlets" that can render a table of contents, login form, content list, and "edit" button that invokes the authoring environment. Regretfully, no portlet for a search form/search results is included in this release. Also, since the portlet code is encrypted, you can only modify the look/feel of the output by tweaking the stylesheet of the site.

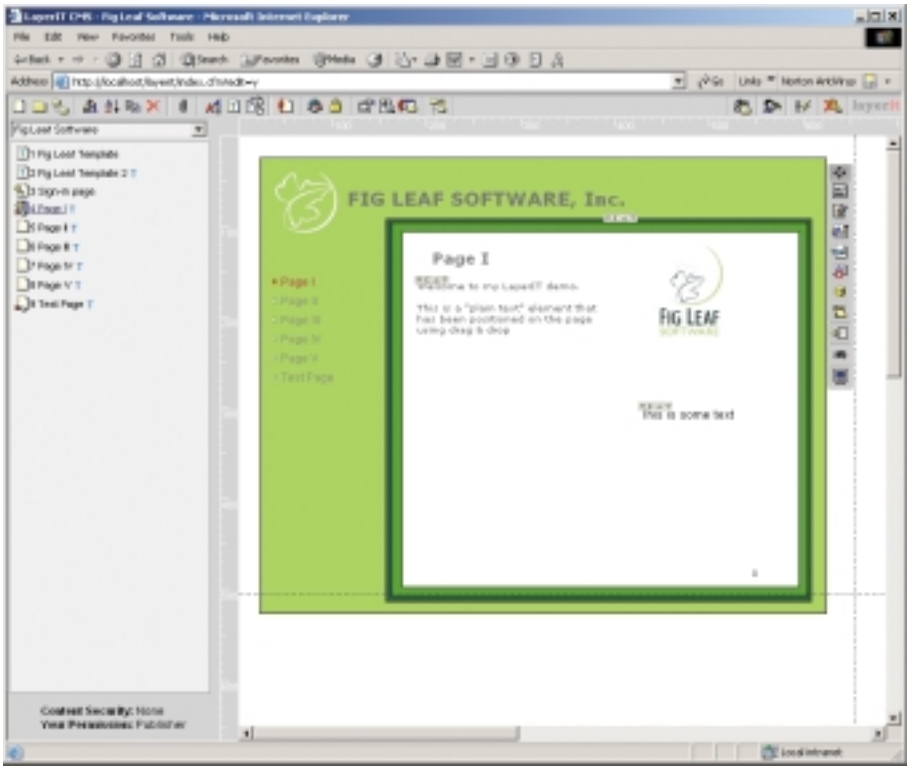


Figure 1: LayerIT uses a simple drag–and–drop metaphor for placing content

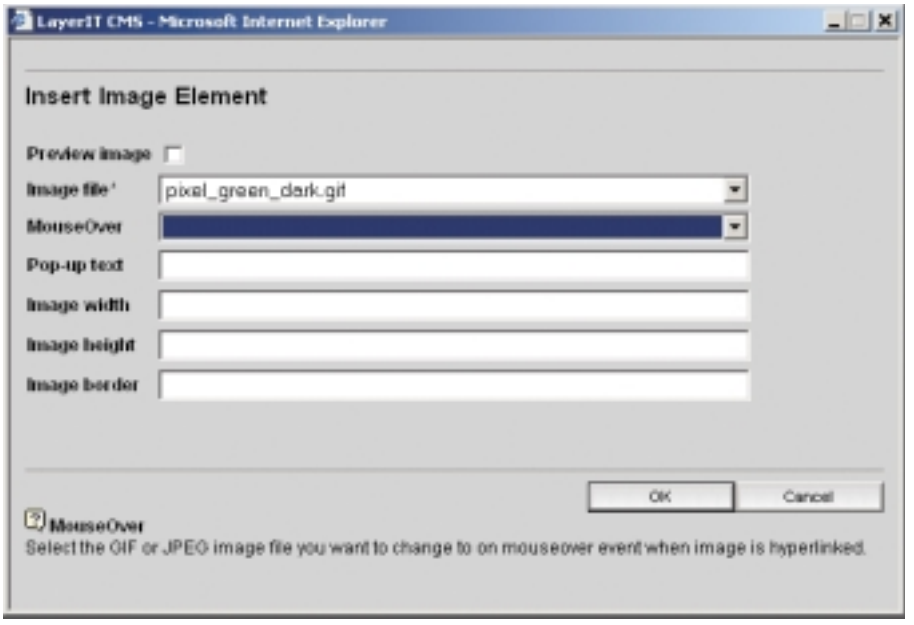


Figure 2: While positioning is a core strength, library services need work

Conan the Librarian

Library services, or the ability to upload, search, and deploy non-HTML content items such as images, are limited in this release. As depicted in Figure 2, while you can upload any document type into specified directories, there is no additional classification scheme that can be applied to the files. I suspect that this UI would start to become ungainly once hundreds of images were uploaded into a production system.

Worse still is LayerIT's "insert image" feature which takes all .gif and .jpg files and unceremoniously drops them into a single drop-down box from which the user must make his or her selection.

Content Approval and Security

Content approval and security have been implemented as a rigid, six-level hierarchy. Users may be assigned the following rights and each succeeding level inherits the rights, of those above it:

1. A READER can only view published pages
2. A CONTRIBUTOR can edit existing pages
3. An EDITOR can create, edit, and delete pages
4. An EDITOR II can create and edit templates, and edit the CSS
5. A DEVELOPER can create and initiate portlets
6. A PUBLISHER can approve pages for publication

This scheme should suffice for organizations that require only a single level of approval prior to publication. Pages can be marked for either anonymous login or member login.

HTML Is Good

The product allows you to create static HTML files from the dynamic content. This feature is typically found only on high-end content management systems and has a direct impact on scalability and portability of the site. Back when I was in the hosting business, we had an old P-90 that would regularly serve up 100K HTML page views a day. Unfortunately, I could not verify this particular feature as it failed when tested with an HTTP server 500 error.

I Am Not the Center of the Universe

I freely admit that I am an "ugly American" when it comes to localizing my applications for a non-U.S. market. Working with LayerIT reminds me that in order for an application to succeed on multiple continents, some attention must be paid. At the time of this review, all documentation for the product was in Norwegian; however, an English translation should be available by the time you read this. More regrettable is that while the system can automatically output the last updated date on a page, the list of formats does not include mm/dd/yyyy.

Fill in the Blanks

LayerIT shows remarkable poise for being essentially a 1.0 release. However, there are a number of issues with their architecture that could prove to be show-stoppers for those of you interested in the product. In particular, there appears to be no direct support for generating "printer-friendly pages." Also, the system utilizes a URL-based approach for distinguishing content. For example, the home page of my sample site is accessed through the following URL: http://localhost/layerit/index.cfm?pageID=2, while a different section page is displayed by typing this URL: http://localhost/layerit/index.cfm?pageID=23.

This URL-based data representation scheme can be difficult for Internet search engines to catalog appropriately and has been abandoned by higher-end CM systems in favor of more traditional "human readable" URLs. The lack of deep integration with the SiteObjects editor is particularly troubling.

I Could Have Been a Dentist...or a Doctor...or a Journalist...

One of the great "life lessons" that I have learned is that an innovative use of technology does not necessarily guarantee market success. In fact, sometimes the relationship between the two is inversely proportional. While LayerIT has a strong technical foundation for being a force in the CMS world, at the time of this writing they have but six commercial customers. Most established players have over a hundred. The company officially launched in February of this year, so there is a bit of risk associated with their "staying power" in the marketplace.

Conclusion

LayerIT is a dichotomy. While conquering the battle for creating a highly usable interface and embracing CSS positioning, its weak implementation of the WYSIWYG editor seriously compromises its usefulness. While striving to include high-end features like HTML generation, it contains a pedestrian implementation of library services. Considering its price-point, however, much of this is forgivable. For those of you on a tight budget wanting to deploy a CMS, LayerIT is certainly worth considering. I anxiously await their "official" 2.0 release that should clear up many of the issues I encountered. Then it may actually qualify as a "best value" for low-end CMS implementations.

About the Author

Steve Drucker is the CEO of Fig Leaf Software, a Macromedia premier solutions and training partner with offices in Washington, DC and Atlanta, GA. He is also a certified Macromedia instructor and MM certified Dreamweaver, Flash, and Advanced ColdFusion MX developer.

sdrucker@figleaf.com

Vitals

LayerIT 2.0 (Pre-release)
LayerIT AS
Britveien 4
N-6411 Molde
Norway

Office: (+47) 71 11 50 00
Direct: (+47) 71 11 50 80
Fax: (+47) 71 11 50 01
E-mail: contact@layerit.com

Test Platform
Dell Latitude C640, 512MB RAM 2.2 GHz P-4
ColdFusion 5.0, Windows 2000 SP2, Microsoft Access DB

Professional Edition: $1,398 USD
Enterprise Edition: $2,695 USD
Hosting Edition: $149/month USD

Target audience: Organizations that have small (<500 page) HTML-based Web sites

Pros: Low-cost leader, usable interface, easy to deploy

Cons: Limited approval/security framework, WYSIWYG editor integration needs work, new product with limited market penetration

Client platform: Administrative GUI supports IE 5+, Netscape 7/Mozilla. Server OS: Tested on Windows platforms only, CF 5/MX

Database support: MS Access, MS SQL Server

International Web Services Conference & Expo

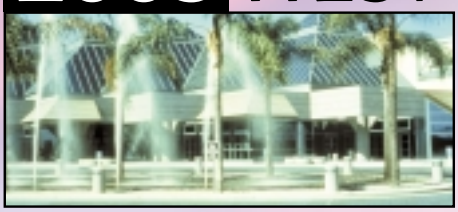
Web Services Edge

3rd Annual

web services conference EDGE &expo

Delivering .NET, JAVA, Mac OS X, & XML Technologies

2003 WEST



SEPT. 30 - OCT. 2, 2003

Santa Clara, CA

web services conference EDGE &expo

REGISTER TODAY!

CALL 201-802-3058

www.sys-con.com

Register before September 1st and

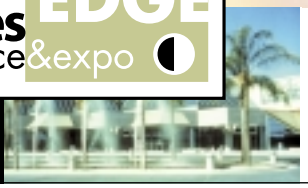
SAVE Up To $200

Produced and Presented: SYS-CON EVENTS

Media Sponsors: WebServices Journal, Java Developer's Journal, WebSphere Developer's Journal, XML Journal, .NET Developer's Journal, WebLogic Developer's Journal, Linux World Magazine, wireless Business Technology, Linux Business Week, SE Advisor, ColdFusion Developer's Journal, PowerBuilder Developer's Journal, market WIRE, HSP Street.com, OASIS, SD Times, LinuxWorld.com, SAMS, Spotlight

WEB SERVICES EDGE CONFERENCE & EXPO

web services EDGE conference&expo



SEPT. 30 - OCT. 2, 2003

Santa Clara Convention Center

It is our pleasure to bring the latest edition of the highly successful Web Services Edge Conference & Expo to the Santa Clara Convention Center, September 30 – October 2, 2003. The Third Annual Web Services Edge West Conference & Expo will continue to build on our past success to make available the most current and relevant information to you, our valued attendee.

With the widespread adoption of Web services across the industry, developers are facing new challenges. In this year's conference program, we will address these challenges with our most comprehensive program to date. Web Services Edge 2003 West will provide practical approaches and solutions to overcome the hurdles in developing and deploying Web services in today's competitive markets. Once again Web Services Edge will feature dedicated tracks covering Java, Web Services, .NET, and XML - along with the newly added Mac OS X Track. Sessions in this track will highlight the use of the Mac OS X platform, which combines the ease of use of a Mac with the power of Unix, in applications and Web services development, deployment, and management.

Your three days will include highly informative keynotes, conference sessions, tutorials, industry-leading university certification programs, case studies, and demo presentations. The Expo Hall will be open on September 30 and October 1, featuring the largest grouping of quality exhibitors prepared to field your questions and solve your development needs.

All the best,
SYS-CON Events

FEATURES & ATTRACTIONS

TO ALL CONFERENCE & EXPO REGISTRANTS

- 3 Days Packed with Education and Training
- Keynotes & Panel Discussions from Industry Leaders
- 60 Hard-hitting and Informative Seminars
- FREE .NET Tutorial with Russ' Tool Shed
- Java University Certification Training
- Industry-Leading Certification Programs
- "Birds of a Feather" Discussions
- Round Table Discussions
- Opening Day Welcome Reception
- SAMS Meet the Authors Hot Topics Lounge
- Compelling Case Studies & Best Practices
- Hands-On Labs
- Featured Product Demonstrations
- Exhibit Floor featuring more than 40 companies and hundreds of products
- Real-time SYS-CON Radio Interviews

WHO SHOULD ATTEND

- Software Developer
- Software Engineer
- Development Manager
- Application Developer
- Technical Director
- Analyst/Programmer
- IT Manager
- Technical Architect
- Team Leader
- Software Consultant

HIGHLIGHTED SPEAKERS

Allan Vermeulen

CTO, Amazon.com

CTO and vice president at Amazon.com directly oversees the Platform Technologies group. This group is responsible for guiding Amazon.com's technology architecture, including building and acquiring foundational components. Prior to his move to Amazon.com, Vermeulen was CTO and vice president of development at Rogue Wave Software. He holds a PhD in Systems Design Engineering from the University of Waterloo.

John Schmidt

Leader of Systems Integration and Middleware, Best Buy Co.

John Schmidt is the chairman of the Methodology Committee for the EAI Industry Consortium and leader of systems integration and middleware at Best Buy Co., a leading specialty retailer of consumer electronics, personal computers, entertainment software, and appliances.

Dave Chappell

VP, Chief Technology Evangelist, Sonic Software

Dave Chappell is the vice president and chief technology evangelist for Sonic Software. He has more than 18 years of industry experience building software tools and infrastructure for application developers, spanning all aspects of R&D, sales, marketing, and support services. Dave has also been published in numerous technical journals, and is currently writing a series of contributed articles for *Java Developer's Journal*.

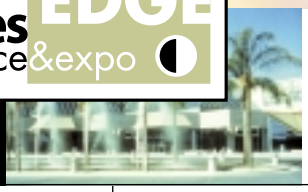
Anne Thomas Manes

Research Director, Burton Group

Anne Thomas Manes is a research director at Burton Group, a research, consulting, and advisory firm. Anne leads research for the Application Platform Strategies service. Named one of NetworkWorld's "50 Most Powerful People in Networking" in 2002, and one of Enterprise Systems Journal's "Power 100 IT Leaders" in 2001, Anne is a renowned technologist in the Web services space. Anne participates in standards development at W3C and OASIS. She is a frequent speaker at trade shows and author of numerous articles and the book *Web Services: A Manager's Guide*.

WEB SERVICES EDGE CONFERENCE & EXPO

web services EDGE conference&expo



SEPT. 30 - OCT. 2, 2003

Santa Clara Convention Center

WEB SERVICES TECHNOLOGY

Presentations will include discussions of security, interoperability, the role of UDDI, progress of the standards-making bodies, SOAP, and BPM. Case studies cover the design and deployment of Web services in the marketplace.

Sessions will focus on:

- Interoperability
- Enterprise Networks
- Web Services Management
- Web Services Standards
- Web Services Orchestration
- Security (WS-Security, SAML)
- BPEL4WS
- UDDI: Dead or Alive?
- ebXML & Web Services
- EAI & Web Services
- RPC vs Documents: Uses and Differences
- User Interfaces for Web Services
- Web Services Best Practices
- Service Oriented Architecture

JAVA TECHNOLOGY

The Java Track features presentations aimed at the beginner, as well as the seasoned Java developer. Sessions will explore the whole spectrum of Java, focusing on J2EE, application architecture, EJB, and J2ME. In addition the track will cover the latest in SWT, Ant, JUnit, open source frameworks, as well as an in-depth look into the vital role that Java is playing in building and deploying Web services.

Sessions will focus on:

- Enterprise Java 1.4
- Ant Applied in "Real World" Web Services
- Developing Application Frameworks w/SWT
- Empowering Java and RSS for Blogging
- JUnit: Testing Your Java with JUnit
- JDK1.5: The Tiger
- Simplifying J2EE Applications
- Using IBM's Emerging Technologies Toolkit (ETTK)
- Apache Axis
- Meeting the Challenges of J2ME Development
- Integrating Java + .NET
- Squeezing Java

XML TECHNOLOGY

Presentations will focus on the various facets of XML technologies as they are applied to solving business computing problems. Sessions will include emerging standards in XML Schemas, XML repositories, industry applications of XML, applying XML for building Web services applications, XML/XSLT/XQuery-based programming using Java/.NET, XML databases, XML tools and servers, XML-based messaging, and the issues related to applying XML in B2B/EAI applications. The XML Track is geared for audiences ranging from beginners to system architects and advanced developers.

Sessions will focus on:

- XML Standards & Vocabularies
- Introduction to XForms
- Securing Your XML and Web Services Infrastructure
- XQuery Fundamentals: Key Ingredient to Enterprise Information Integration
- XML and Enterprise Architecture: Technology Trends
- Standards-Based Enterprise Middleware Using XML/Web Services
- XML and Financial Services
- Canonical Documents for Your Business: Design Strategies
- XPath/XSLT 2.0: What's New?
- XML Schema Best Practices
- XML in EAI, Enterprise Portals, Content Management

.NET TECHNOLOGY

Presentations will explore the Microsoft .NET platform for Web services. To the average developer, it represents an entirely new approach to creating software for the Microsoft platform. What's more, .NET development products - such as Visual Studio .NET - now bring the power of drag-and-drop, GUI-based programming to such diverse platforms as the Web and mobile devices.

Sessions will focus on:

- ASP.NET
- Security
- VB.NET
- .NET and XML
- Smart Device Extensions for VS.NET
- Best Practices
- Shared Source CLI
- .NET Remoting
- Smart Devices in Health Care Settings
- Mobile Internet Toolkit
- ROTOR
- Portable .NET
- ASP.NET Using Mono
- Using WSE with IBM's WSTK
- GUI applications Using Mono
- Portals – Windows SharePoint Services/SharePoint Portal Server
- Windows Server 2003 and IIS 6
- .NET and Java Interoperability
- Distributed .NET for Financial Applications
- Developing C# with Eclipse

MAC OS X

OS X represents a new wave of operating systems. It combines the ease of use of a Mac with the power of Unix. Sessions in this track will highlight the use of the Mac OS X platform in applications and Web services development, deployment and management.

Sessions will focus on:

- Introducing OS X (Panther): What's New?
- Quick Applications using AppleScript
- Enterprise Java and OS X
- Developing Web Services Using WebObjects
- Xserve: Ease of OS X and Power of Unix
- Introducing Quartz: 2D Graphics for Apple
- OS X for the Unix Developer
- Securing OS X Applications
- Java and OS X: A Perfect Marriage
- Programming Rich User Interfaces Using Cocoa

SPECIAL DISCOUNTS AVAILABLE

Take advantage of the Early Bird and Pre-registration values available right now, or save even more with a group of 5 or more. For special group discounts contact Michael Lynch at mike@sys-con.com, or by phone at (201) 802-3058.

For more information visit
www.sys-con.com
or call
201 802-3069

PRODUCED BY SYS-CON EVENTS

Conference at-a-Glance

		JAVA	.NET	WEB SERVICES	XML
TUESDAY, SEPTEMBER 30 DAY 1	8:00AM – 4:00PM	**REGISTRATION**			
	9:00AM – 9:50AM	Enterprise Java 1.4	Using WSE 2.0	Web Services Management	Introduction to Xforms
	10:00AM – 10:50AM	Opening Keynote - Allen Vermeulen, CTO, Amazon.com			
	11:00AM – 6:00PM	**EXPO OPEN**			
	2:00PM – 2:50PM	Keynote Panel Discussion - Enterprise Application Integration			
	3:00PM – 3:50PM	Ant Applied in "Real World" Web Services	Smart Devices in Health Care Settings	Service Oriented Architecture	Securing Your XML and Web Services Infrastructure
	4:00PM – 4:50PM	Developing Application Frameworks with SWT	Using the Mobile Internet Toolkit	Web Services Orchestration	XQuery Fundamentals: Key Ingredient to Enterprise Information Integration
	5:00PM	**OPENING NIGHT RECEPTION**			
WEDNESDAY, OCTOBER 1 DAY 2	8:00AM – 4:00PM	**REGISTRATION**			
	9:00AM – 9:50AM	Integrating Java and .NET	Introduction to ROTOR	Security (WS-Security, SAML)	Standards-Based Enterprise Middleware Using XML/Web Services
	10:00AM – 10:50AM	Morning Keynote			
	11:00AM – 4:00PM	**EXPO OPEN**			
	2:00PM – 2:50PM	Keynote Panel Discussion - Interoperability: Is Web Services Delivering?			
	3:00PM – 3:50PM	JUnit: Testing Your Java with JUnit	Using Portable .NET	WS-BPEL	XML and Enterprise Architecture: Technology Trends
	4:00PM – 4:50PM	JDK1.5: The Tiger	ASP.NET with Mono	UDDI: Dead or Alive?	Using XML Schemas Effectively in WSDL Design
	5:00PM – 6:00PM	Squeezing Java	Using WSE with IBM's WSTK	Web Services Choreography, Management, and Security - Can They Dance Together?	Canonical Documents for Your Business: Design Strategies
THURSDAY, OCTOBER 2 DAY 3	8:00AM – 4:00PM	**REGISTRATION**			
	9:00AM – 9:50AM	Using IBM's Emerging Technologies Toolkit (ETTK)	Distributed .NET for Financial Applications	eAI & Web Services	XML and the Fortune 500
	10:00AM – 10:50AM	Morning Technical Keynote			
	11:00AM – 11:50AM	Apache Axis	Developing C# with Eclipse	RPC vs Documents: Uses and Differences	XPath/XSLT 2.0: What's New?
	12:00PM	**BREAK**			
	1:00PM – 1:50PM	Meeting the Challenges of J2ME Development	Windows SharePoint Services	The Seven Habits of Highly Effective Enterprise Service Buses (ESBs)	ebXML & Web Services
	2:00PM – 2:50PM	Keynote Panel Discussion - Summit on Web Services Standards			
	3:00PM – 3:50PM	Empowering Java and RSS for Blogging	BizTalk 2003	See www.sys-con.com for more information	See www.sys-con.com for more information
	4:00PM – 5:00PM	See www.sys-con.com for more information	See www.sys-con.com for more information	See www.sys-con.com for more information	See www.sys-con.com for more information

MAC OS X

	Introducing OS X (Panther) What's New?
	Programming Rich User Interfaces Using Cocoa
	Quick Applications using AppleScript
	Java and OS X: A Perfect Marriage
	Enterprise Java and OS X
	Developing Web Services Using WebObjects
	Cocoa, Carbon, Java: Application Frameworks for OS X (When to use what)
	Securing OS X Applications
	Xserve: Ease of OS X and Power of Unix
	OS X for the Unix Developer
	Introducing Quartz: 2D Graphics for Apple
	See www.sys-con.com for more information

PROGRAM SUBJECT TO CHANGE

Microsoft .net

FREE Microsoft® Tutorial

RUSS' TOOLSHED

TINKERING WITH VISUAL STUDIO

RUSS' TOOL SHED

Join Russ as he shows you how to use Visual Studio .NET

INTRO TO WEB SERVICES USING VS.NET

One of the key ideas behind the .NET strategy is the concept of software as a service, or in short, Web services. This session will explain what a Web service is and provide an overview of its related technologies like XML, SOAP, and UDDI. We will demonstrate how the .NET Framework makes it easy to implement them for new and existing applications. This session will also provide concrete best practices for building XML Web services using Visual Studio .NET. We'll answer many common questions like: How will my Web service scale? How can my XML Web services enable interoperability with Web services from other vendors as well as within my own organization? We'll delve into building highly reliable and secure Web services. Also, we will discuss issues such as dealing with complex data types using WSDL (Web Services Description Language), as well as securing SOAP messages using encryption. We'll see how developers can use enterprise-level XML Web services to simplify customer solutions.

ADVANCED WEB SERVICES USING ASP.NET

This session will explore some of the more advanced areas of SOAP in ASP.NET's support for Web services. ASP.NET Web services are the preferred way for Web developers to expose Web services on the Internet. The goal is quick, easy, and high-performing SOAP services. We will look at how to use the SOAP extension classes to create some very interesting applications on top of the core SOAP architecture found within the .NET Framework. For instance, you can implement an encryption algorithm or screen scraping on top of the Web service call. We'll dig into more advanced topics, explore the SOAP headers, and see ways to ensure security in our Web services.

.NET REMOTING ESSENTIALS

Microsoft .NET Remoting is the .NET technology that allows you to easily and quickly build distributed applications. All of the application components can be on one computer or they can be on multiple computers around the world. .NET Remoting allows client applications to use objects in other processes on the same computer or on any other computer to which it can connect over its network. During this presentation we will discuss what you will need to know to get started with .NET Remoting. We will talk about how .NET Remoting compares with DCOM, how to host remoted objects in a variety of applications, how to call remoted objects from a client application, how to control the lifetime of remoted objects, and how to secure remoting applications.

REGISTRATION FORM

CONFERENCE: Sept. 30 – Oct. 2, 2003 EXPO: Sept. 30 – Oct. 1, 2003

Santa Clara Convention Center • Santa Clara, CA

THREE WAYS TO REGISTER FOR CONFERENCE

1) On the Web: Credit Cards or "Bill Me." Please make checks payable to SYS-CON Events.
2) By Fax: Credit Cards or "Bill Me" 201-782-9651
3) By Mail: 135 Chestnut Ridge Road, Montvale, New Jersey 07645, Attention: Registration

Please note: Registrations are not confirmed until payment is received.

Please complete sections 1, 2, 3 and 4

1 YOUR INFORMATION (Please Print) ☐ Mr. ☐ Ms.

First Name _____ Last Name _____

Title _____

Company _____

Street _____

Mail Stop _____

City _____

State _____ Zip _____ Country _____

Phone _____

Fax _____ E-Mail _____

2 PAYMENT METHOD: (Payment in full due with registration)

☐ Check or Money Order Enclosed (Registration confirmed upon receipt of payment)

Check # _____ Amount of Check $ _____

Charge my ☐ Visa ☐ MasterCard ☐ American Express ☐ Discover

Name on card _____

Card # _____ Exp. Date _____

Signature _____

Billing Address (if different from mailing address)

_____

3 PLEASE INDICATE YOUR CONFERENCE CHOICE

Total Registration fee $_____

	Before 8/1/03	Before 9/26/03	Onsite
❑ **GP Gold Passport** Good for all three days of the .NET, Web Services, XML, Java, and Mac OS X Tracks, including Keynotes, Panel Discussions, preferred seating for Microsoft .NET's Russ'"Tool Shed" Tutorial, and your choice of one Sun Microsystems Java™ University Class	$1,195.00	$1,395.00	$1,495.00
❑ **3D Three Day Conference** (Does not include Sun Java™ Education)	$1,095.00	$1,295.00	$1,395.00
❑ **2D Two Day Conference** (Does not include Sun Java™ Education) (select any two days: ❑ Tue. ❑ Wed. ❑ Thurs.)	$995.00	$1,195.00	$1,295.00
❑ **1D One Day Conference** (Does not include Sun Java™ Education) (select any one day: ❑ Tue. ❑ Wed. ❑ Thurs.)	$495.00	$595.00	$695.00
❑ **JU1 Sun Java™ University Class** Select one: ❑ Web Services Programming Using Java™ Technology and XML (Sept. 30) ❑ Java™ Fast Path: Programmer (Oct. 1) ❑ Java™ Fast Path: Architect (Oct. 2)	$595.00	$695.00	$795.00
❑ **JU2 Sun Java™ University Class** Select two: ❑ Web Services Programming Using Java™ Technology and XML (Sept. 30) ❑ Java™ Fast Path: Programmer (Oct.1) ❑ Java™ Fast Path: Architect (Oct. 2)	$1,095.00	$1,295.00	$1,395.00
❑ **JU3 Sun Java™ University Class** Select three: ❑ Web Services Programming Using Java™ Technology and XML (Sept. 30) ❑ Java™ Fast Path: Programmer (Oct. 1) ❑ Java™ Fast Path: Architect (Oct. 2)	$1,195.00	$1,395.00	$1,495.00
❑ **EO Expo Only**	FREE	FREE	$50.00

4

A. Your Job Title

❑ CTO, CIO, VP, Chief Architect
❑ Software Development Director/Manager/Evangelist
❑ IT Director/Manager
❑ Project Manager/Project Leader/Group Leader
❑ Software Architect/Systems Analyst
❑ Application Programmer/Evangelist
❑ Database Administrator/Programmer
❑ Software Developer/Systems Integrator/Consultant
❑ Web Programmer
❑ CEO/COO/President/Chairman/Owner/Partner
❑ VP/Director/Manager Marketing, Sales
❑ VP/Director/Manager of Product Development
❑ General Division Manager/Department Manager
❑ Other (please specify)_____

B. Business/Industry

❑ Computer Software
❑ Computer Hardware and Electronics
❑ Computer Networking & Telecommunications
❑ Internet/Web/E-commerce
❑ Consulting & Systems Integrator
❑ Financial Services
❑ Manufacturing
❑ Wholesale/Retail/Distribution
❑ Transportation
❑ Travel/Hospitality
❑ Government/Military/Aerospace
❑ Health Care/Medical
❑ Insurance/Legal
❑ Education
❑ Utilities
❑ Architecture/Construction/Real Estate
❑ Agriculture
❑ Nonprofit/Religious
❑ Other (please specify)
_____

C. Total Number of Employees at Your Location and Entire Organization (check all that apply):

	Location	Company
10,000 or more	01 ❑	01 ❑
5,000 - 9,999	02 ❑	02 ❑
1,000 - 4,999	03 ❑	03 ❑
500 - 999	04 ❑	04 ❑
100-499	05 ❑	05 ❑
100 or less	06 ❑	06 ❑

D. Please indicate the value of communications and computer products and services that you recommend, buy, specify, or approve over the course of one year:

❑ $10 million or more
❑ $1 million - $9.9 million
❑ $500,000 - $999,999
❑ $100,000 - $499,999
❑ $10,000 - $99,999
❑ Less than $10,000
❑ Don't know

E. What is your company's gross annual revenue?

❑ $10 billion or more
❑ $1 billion - $9.9 billion
❑ $100 million - $999 million
❑ $10 million - $99.9 million
❑ $1 million - $9.9 million
❑ Less than $1 million
❑ Don't know

F. Do you recommend, specify, evaluate, approve or purchase wireless products or services for your organization?
01 ❑ Yes 02 ❑ No

G. Which of the following products, services, and/or technologies do you currently approve, specify or recommend the purchase of?

❑ Application Servers
❑ Web Servers
❑ Server Side Hardware
❑ Client Side Hardware
❑ Wireless Device Hardware
❑ Databases
❑ Java IDEs
❑ Class Libraries
❑ Software Testing Tools
❑ Web Testing Tools
❑ Modeling Tools
❑ Team Development Tools
❑ Installation Tools
❑ Frameworks
❑ Database Access Tools / JDBC Devices
❑ Application Integration Tools
❑ Enterprise Development Tool Suites
❑ Messaging Tools
❑ Reporting Tools
❑ Debugging Tools
❑ Virtual Machines
❑ Wireless Development Tools
❑ XML Tools
❑ Web Services Development Toolkits
❑ Professional Training Services
❑ Other [Please Specify]

If you require special assistance covered under the Americans with Disabilities Act, please call 201-802-3058 by September 16, 2003.

CANCELLATIONS, SUBSTITUTIONS, REFUNDS

Fax written request to SYS-CON Registration 201-782-9651. Requests for refunds received prior to August 29, 2003 will be honored, less a 10% handling charge; requests received after August 29, 2003, and before September 12, 2003, will be honored less a 20% handling charge. No requests for refunds will be honored after September 12, 2003. Requests for substitutions must be made in writing prior to September 26, 2003. No one under 18 is permitted to attend. No warranties are made regarding the content of sessions or materials.

Speakers, sessions, and schedule are subject to change without prior notice.

No solicitation by anyone other than official exhibitors, sponsors or marketing partners is permitted. Such behavior is cause for expulsion without refund.

Web Services Edge WEST 2003
SEPT. 30 - OCT. 2, 2003
Santa Clara Convention Center

web services conference&expo EDGE
WEB SERVICES EDGE CONFERENCE & EXPO

Macromedia Introduces Macromedia Contribute 2

(*San Francisco*) – Macromedia has introduced Macromedia Contribute 2, the easiest way for individuals and teams to update and manage content on existing Web sites. Macromedia Contribute 2 now works on Mac OS X, and connects to Web sites faster and more securely. Scheduled for release this month, it will be priced at $99, with upgrades available for a limited time through Macromedia at only $9.99.

"Macromedia Contribute unlocked the power and potential of the Web for a whole new class of users," said Norm Meyrowitz, president of products, Macromedia. "With Macromedia Contribute 2, we added Mac OS X support, as well as new features to specifically benefit customers in the education, government, and small business markets where the first version of Contribute was particularly successful."

Macromedia Contribute dramatically simplifies Web content updating while maintaining the overall look and feel of a site. Users simply browse to the page they want to modify, make the necessary changes, and then publish the updated page back to the server. The product features the same look and feel as traditional business productivity applications and supports content created in Microsoft Word and Excel.

More than 15,000 companies now use more than 70,000 seats of Macromedia Contribute to keep their Web sites up to date. Contribute has also been broadly adopted in the education, government, and small business markets.

Macromedia Contribute 2 Brings Paypal Online Transactions to Small Businesses

(*San Francisco*) – Macromedia Contribute 2 will allow small and medium-sized businesses to use the leading global payment service, PayPal, as their online payment solution. By incorporating this payment solution into Contribute, Macromedia and PayPal are enabling users to integrate e-commerce functionality seamlessly into their Web sites.

Contribute users on Windows or Mac OS X can incorporate PayPal functionality into their Web sites through a simple wizard-based process. By using PayPal on their sites, online businesses can utilize a highly secure and easy payment solution. PayPal allows merchants to receive payments fast – a few seconds in most cases – and to accept most forms of payment including credit cards and bank transfers.

The integration of PayPal Merchant Tools within Contribute provides a shopping cart solution, a "Buy Now" button, and a host of other business services and tools such as instant payment notification, shipping, subscription services, tracking, and inventory management tools. Additionally, PayPal is a very safe way to pay online, with one of the industry's most successful fraud-prevention systems.

Macromedia Renews J2EE License Agreement for JRun

(*San Francisco*) – Macromedia has renewed its Java 2 Platform, Enterprise Edition (J2EE) licensing agreement with Sun Microsystems, Inc. Macromedia has been a licensee for more than three years. This renewal means that Macromedia JRun can continue to deliver commercial-grade, highly affordable J2EE technology to businesses and ISVs worldwide.

Macromedia JRun has helped spur the adoption of Java technology since its introduction in 1997. More than 10,000 organizations use Macromedia JRun because of its ability to help make J2EE technology accessible, approachable, and affordable.

Macromedia embeds JRun in both the Macromedia ColdFusion MX and Macromedia Breeze product lines and relies on JRun to power the macromedia.com corporate Web site.

"Macromedia is pleased to reaffirm its commitment to providing affordable, no-hassle J2EE technology by extending our licensing agreement," said Norm Meyrowitz, president of products, Macromedia. "We remain heavily involved in shaping the future of Java technology and ensuring our developers benefit from the full potential of the J2EE platform."

"Macromedia JRun addresses an important part of the J2EE compatible marketplace," said Mark Bauhaus, vice president, Java Web Services, Sun Microsystems, Inc. "The success of JRun in thousands of companies worldwide has helped drive the adoption of J2EE technology into a broader base of customers on multiple platforms."

Macromedia is an elected member of the Java Community Process Executive Committee, which is comprised of 16 companies charged with leading the community of developers and vendors in creating future standards for the Java platform. Macromedia is also active in a number of Java Specification Requests (JSRs), committees that shape the actual descriptions of proposed and final specifications for the Java platform.

Macromedia Announces Affiliate Program – Turn Your Suggestions into Sales

(*San Francisco*) – Macromedia has launched its initial pilot of an affiliate program to enable Web design and development agencies to earn a referral commission when they sell Macromedia products such as Macromedia Studio MX and Contribute to their clients. For more information on this program, visit www.macromedia.com/go/affiliate.

Last Call for *CFDJ* Readers' Choice Votes!

Vote for your favorite ColdFusion products/services in the 2003 ***ColdFusion Developer's Journal*** Readers' Choice Awards competition now in full swing at www.sys-con.com/coldfusion/readerschoice2003. Widely referred to as the "Oscars of the Software Industry," the Readers' Choice Awards program has become the most respected industry competition of its kind. Voting began on March 1, and will continue until August 30, 2003. Winners will be announced at Web Services Edge 2003 West, in Santa Clara, CA, September 30–October 2.