

COLD FUSION Developer's Journal

ColdFusionJournal.com

June 2006 Volume:8 Issue: 6

Introducing Spry 7

How I Became a Hero at the Office 22

Security Matters 32

Interface Customization Part 1 34

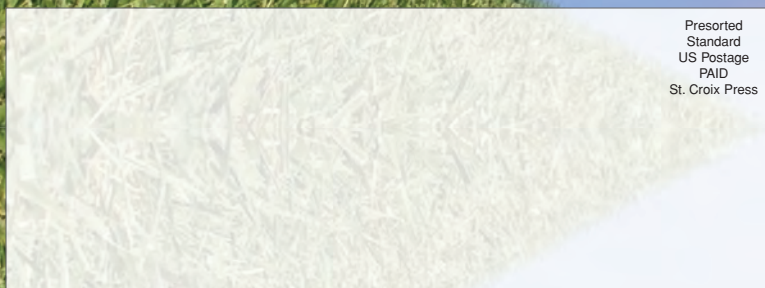
Getting Started with Flex 2 38

User Interface Layout in Flex 2.0 42

YOUR FIRST FLEX APP WITH A CF BACKEND

16

Flex 2.0 View States & Transitions 48



Presorted
Standard
US Postage
PAID
St. Croix Press





ColdFusion Hosting is our Complete Focus

➤ **POWERFUL HOSTING PLANS**

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

➤ **RELIABLE NETWORK**

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

➤ **FANTASTIC SUPPORT SERVICES**

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

CFDynamics

866.233.9626 ➤ CFDYNAMICS.COM

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

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





One little box. A whole lot of power.

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

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



› **Structured business reporting? Check. Printable web content? Check.**

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



› **Make rapid J2EE development a reality.**

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



› **New mobile applications are a go.**

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

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

BALANCE

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

www.communitymx.com



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



editorial advisory board

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

editorial**editor-in-chief**

Simon Horwith simon@sys-con.com

technical editor

Raymond Camden raymond@sys-con.com

executive editor

Nancy Valentine nancy@sys-con.com

research editor

Bahadır Karuv, PhD bahadir@sys-con.com

production**lead designer**

Abraham Addo abraham@sys-con.com

art director

Alex Botero alex@sys-con.com

associate art directors

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

editorial offices**SYS-CON MEDIA**

135 Chestnut Ridge Rd., Montvale, NJ 07645
Telephone: 201 802-3000 Fax: 201 782-9638
COLDFUSION DEVELOPER'S JOURNAL (ISSN #1523-9101)
is published monthly (12 times a year)
by SYS-CON Publications, Inc.

postmaster: send address changes to:

COLDFUSION DEVELOPER'S JOURNAL
SYS-CON MEDIA
135 Chestnut Ridge Rd., Montvale, NJ 07645

@copyright

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

Worldwide Newsstand Distribution

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

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

Thoughts from My Blog



By Simon Horwith

I was really busy
for several weeks
prior to CFUnited

and am back on the
road again... but there's a
silver lining.

I'm basically done with my first round of SAM docs and I will be making those available very soon; I have the code and stories/advice to post as well – I just haven't had the time. I do promise I'll be blogging and posting interesting stuff very soon, though.

Frameworks vs Methodology Debate

I am interested to know what's on the agenda for the "The Big Framework and Methodology Debate" session at MAX (www.adobe.com/events/max/sessions/wd301s.html) since all of the listed speakers are framework advocates. I also want to take a moment to publicly state that, as much fun as I've had creating controversy and being a human piñata in the interest of encouraging developers to think, I feel like the whole "methodology vs frameworks" debate is being beaten to death.

I believe that people should go with whatever works for them and there is no such thing as a silver bullet. I don't generally use frameworks, though I know a lot of good developers who do. One reason I don't generally prescribe to any one framework is the simple fact that I spend the bulk of my time modeling business domains and coding and architecting the really complex stuff...and that has absolutely nothing to do with frameworks or whether or not a framework is being used on a project.

Because I'm a little tired of debating about frameworks, I'm not going to be quite so vocal about them anymore, aside from a few possible special events that are planned, and/or if people prompt me to discuss them (for good reason). This isn't me "throwing in the towel" and I still feel the same way I've always felt, but I think that the community is


now discussing these topics more, hopefully as a partial result of all the "trouble" I caused. I now feel like moving on and spending time and energy on more productive ventures, including getting back to releasing code.

Speaking of code, if there's anything that folks would like me to write and release, let me know. No promises, and it's not that I don't have plenty to work on already, but I'd be interested to know what you all would like.

Shoot the Guru

At CFUnited last year, in addition to other games in the community area, my company – AboutWeb – had "Shoot the Guru." Conference attendees received a nerf gun and had to shoot and knock over small cardboard cutouts of CF Gurus in order to win prizes.

This year, at CFUnited, Shoot the Guru was back, only this time we had a Flash version of the game. Gurus popped out of cubicles (on a big screen TV) and conference attendees shot them with a laser gun (a USB gun that from a programmer's view acts like a mouse-click whenever you pull the trigger).

Kelly Brown has just released "Shoot the Guru" on his blog for folks to play... and he made the code available as well. The blog entry URL is www.kellyjo.com/blog/index.cfm/2006/7/11/Shoot-the-Guru. Have fun! 

About the Author

Simon Horwith is the editor-in-chief of ColdFusion Developer's Journal and is the CIO at AboutWeb, LLC, a Washington, DC based company specializing in staff augmentation, consulting, and training. Simon is a Macromedia Certified Master Instructor and is a member of Team Macromedia. He has been using ColdFusion since version 1.5 and specializes in ColdFusion application architecture, including architecting applications that integrate with Java, Flash, Flex, and a myriad of other technologies. In addition to presenting at CFUGs and conferences around the world, he has also been a contributing author of several books and technical papers. You can read his blog at www.horwith.com. simon@horwith.com



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

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

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

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



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

<http://edgewebhosting.net>

By the Numbers:

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



2003 - 2006

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

president & ceo

Fuat Kircaali fuat@sys-con.com

group publisher

Jeremy Geelan jeremy@sys-con.com

advertising

senior vp, sales & marketing

Carmen Gonzalez carmen@sys-con.com

vp, sales & marketing

Miles Silverman miles@sys-con.com

advertising director

Robyn Forma robyn@sys-con.com

advertising manager

Megan Mussa megan@sys-con.com

associate sales manager

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

sys-con events

president, events

Grisha Davida grisha@sys-con.com

national sales manager

Jim Hanchrow jimh@sys-con.com

customer relations

circulation service coordinator

Edna Earle Russell edna@sys-con.com

manager, jdj store

Brunilda Staropoli bruni@sys-con.com

sys-con.com

vp, information systems

Robert Diamond robert@sys-con.com

web designers

Stephen Kilmurray stephen@sys-con.com

Wayne Uffelman wayne@sys-con.com

online editor

Roger Strukhoff roger@sys-con.com

accounting

financial analyst

Joan LaRose joan@sys-con.com

accounts payable

Betty White betty@sys-con.com

accounts receivable

Gail Naples gailn@sys-con.com

subscriptions

Subscribe@sys-con.com

Call 1-888-303-5282

For subscriptions and requests for bulk orders, please send your letters to Subscription Department

Cover Price: \$8.99/issue

Domestic: \$89.99/yr (12 issues)

Canada/Mexico: \$99.99/yr

All other countries \$129.99/yr

(U.S. Banks or Money Orders)

Back issues: \$12 U.S. \$15 all others



Introducing Spry

The promising pre-release of Adobe's AJAX framework

By Kelly Brown

If you've been to Adobe Labs lately or keep up with

the blogosphere you've probably heard the buzz about Spry. It's a new AJAX framework from Adobe.

It lets designers build rich Internet applications with little or no programming. The Adobe-Macromedia combine has been pushing RIAs for years and its vision is just now moving into the mainstream.

Adobe had three goals when creating Spry: open access, easy use, and enable innovation.

Open Access

Proprietary tags or server-side code weren't used in the Spry framework, instead the capabilities of existing HTML tags were enhanced. Spry has been released under a BSD license and it's freely available on the Adobe Labs site.

Easy-To-Use

Spry was created with designers in mind. It uses regular HTML tags, CSS, and JavaScript. Very little coding is required, although its capabilities can certainly be enhanced with coding. The syntax reminds me of how ColdFusion integrates into Web pages as a tag-based language. You use the tags you're already familiar with, but with new properties. Adobe certainly accomplished its goal; Spry is very easy to use.

Enable Innovation

Enabling innovation is one of those warm and fuzzy goals. In this case what it's trying to do is give developers the ability to create rich applications while still allowing them freedom of design.

All that sounds great, but what is it really? Spry boils down to a couple of JavaScript libraries that you include in your Web page.

Just a couple and you can be adding dynamic interactive content to your site. It can't be that easy you say, but it really is.

Loading Data in Spry

The two main JavaScript libraries that have to be included are xpath.js and spryData.js. The xpath.js is the Google implementation of XPath 1.0; spryData.js is the Spry library itself. Once these two files are included in your page Spry is enabled. Of course just enabling Spry doesn't do anything. You have to load some data and display it. Once you're Spry-enabled you can dynamically load data. This is one of the core features of the framework and is done by creating a Spry XMLDataSet. Right now XML is the only type of data supported, but there are plans to support other formats in the future. The sample below shows the code needed to enable Spry and load some data:

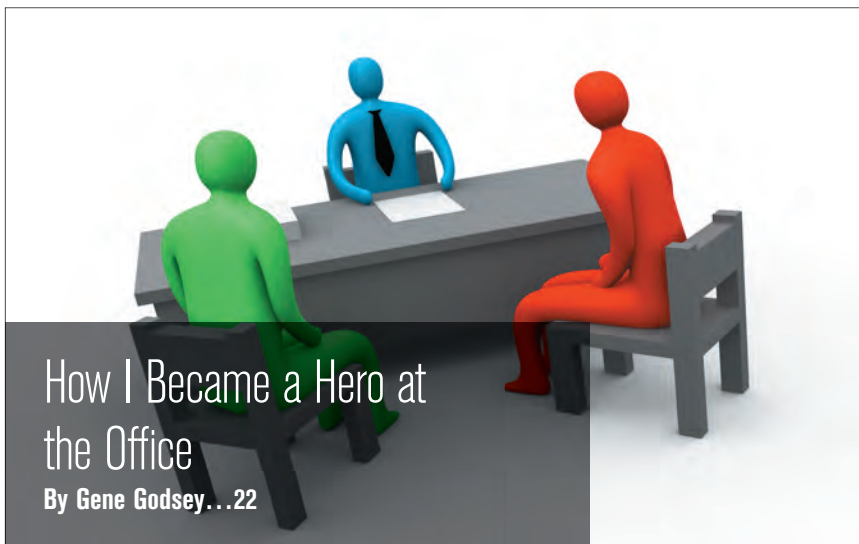
```
<script type="text/javascript" src="xpath.js"></script>
<script type="text/javascript" src="SpryData.js"></script>
<script type="text/javascript">
    var mydata=new Spry.Data.XMLDataSet("mydata.xml","/people/person");
</script>
```

Two pieces of information are needed to load data into the Spry, the URL of the file you're loading and an xpath statement to select from the data that's returned. I used an XML file in the sample, but it could just as easily have been a dynamic file such as ColdFusion page that returns data from a database.

When using ColdFusion make sure you include a <cfcontent type="text/xml"> or Spry won't read the data. This has tripped server people up (include me) the first time they tried to retrieve data from a ColdFusion page. The URL can access data from anywhere, but it will generate a warning or may disallow loading data from other sites depending on the browser security settings. The xpath statement lets you filter the data that's retrieved.

— continued on page 10

TABLE OF CONTENTS



editorial

Thoughts from My Blog

By Simon Horwith

5

spry

Introducing Spry

The promising pre-release of Adobe's AJAX framework

By Kelly Brown

7

flex

Flex for ColdFusion Developers

An introduction

By Oliver Merk

14

cfugs

ColdFusion User Groups

28

foundations

Security Matters

Is more always better?

By Hal Helms

32

design

Interface Customization Part 1

The basics of themes, styles, and skins

By Tariq Ahmed

34

cf 101

Getting Started with Flex 2

An introduction

By Jeffry Houser

38

mxml

User Interface Layout in Flex 2.0

Planning your space

By Peter Ent

42



WEB ALL-STAR seeks an Integrated Software Suite that's up for anything. Understand my need to create cool graphics one day, while tweaking CSS the next. Should be intuitive, multi-platform and not phased by my turbo-charged pace. High-maintenance is a no-no.



Different people. Different needs. One suite solution.

With the latest versions of Macromedia Dreamweaver®, Flash® Professional, Fireworks®, Contribute™, and FlashPaper™, the new Studio 8 is quite a catch. To meet Studio 8 and find all the tools you need to design, develop and maintain online experiences, visit www.macromedia.com/go/studio8_mxdj

macromedia®
STUDIO 8

This is required even if the only thing you're selecting is the root node of the XML. Using the same data source you can select different XML nodes or search for a specific node or series of nodes in the XML.

Displaying Data in Spry

Once we have the data we want to display it. For this example I'm assuming we return a list of people with firstname, lastname, and e-mail as the fields. To output data you have to define a Spry Region. This is done by adding the attribute `spry:region="dataset"` to the tag such as a `div` that encloses the area using the dataset. This lets Spry know that it should update this area of the page if the specified dataset changes. The data output is very similar to data binding in Flex and has similarities to ColdFusion except with curly brackets and colons instead of hash marks. For instance, to output the first name returned in our data set the syntax would be `{mydata::firstname}`. Much like a ColdFusion query this would display the firstname field of our dataset from the first row of data. What we need to go with this is the ability to loop over a dataset when multiple data items are returned. This is done using the `spry:repeat` tag. But for a couple of minor differences, the syntax should feel familiar to you if you've been using ColdFusion. You don't use an output tag, you just embed it in the tag you want to loop with. The example below demonstrates how the tag would be use to display a table with our data:

```
<div spry:region="mydata">
<table>
<tr>
<th>First Name</th><th>Last Name</th><th>Email</th>
</tr>
<tr spry:repeat="mydata">
<td>{firstname}</td><td>{lastname}</td><td>{email}</td>
</tr>
</table>
</div>
```

This example is creating a table, but the same technique can be used to create other dynamic elements such as a select box by putting the `SpryRepeat` in the option tag. It makes chores like have multi-select boxes where the items in the second box depend on the item chosen on the first a breeze. You can update the second box without reloading the page by changing the URL of the data source for the second dataset. For instance, I could add a button that updated the data source to my table example:

```
<input type="button" onClick="mydata.setURL('mydata2.xml')">
```

When you click on the button the XML data would be read from a different file and the table would automatically update without reloading the page. You're not limited to text, you can bind the `src` attribute of an image to a Spry data field and the image will automatically be updated when the data changes.

Other Capabilities

The dynamic data capabilities are the core of the Spry framework, but this is still a pre-release of the technology. Though it's not covered in the documentation, there are two other Spry

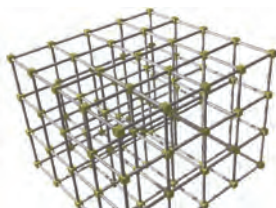
libraries included as part of the demos: `SpryAccordion.js` and `SpryEffects.js`. `SpryAccordion.js` contains code to produce an accordion-style menu. `SpryEffects.js` contains code for animation effects such as Move, Size, and Opacity. I expect these libraries will be expanded to include a full set of dynamic UI components and animation effects.

Limitations

All this dynamic functionality sounds great and you may be tempted to jump right in and start making Spry-based Web pages, but there are some limits you should keep in mind when working with Spry. Spry is AJAX-based and as such has the same limitations as AJAX. It can't work without JavaScript. If a user has JavaScript turned off your page won't work, not only that but the data might be inaccessible. If you were to browse to the sample used in this article with JavaScript turned off, you would just see a table with the variables in it. There would be no way for you get to the data.

This situation is of concern to the Spry team too and the latest release includes documentation on how to make your page usable even if Spry can't run. Since we're dealing with JavaScript you face the dreaded JavaScript-compatibility issues. Spry has been tested with Firefox 1.5 (Windows and Mac), Netscape 7.2 (Windows), Internet Explorer 6 (Windows), and Safari 2.0.3 (Mac). If you're not using one of these browsers, don't count on Spry working. It may not work at all or you may only have partial functionality. You also have the accessibility issues that come with dynamic Web content. Most screen readers do a poor job or just don't work with dynamic updates in Web pages.

Also keep in mind that Spry is in pre-release. The code could change at any time and it's not guaranteed to be backwards-compatible. There are already differences between the May and June releases. For instance, the format of the Spry Region attribute was changed from `spryregion` to `spry:region`. A minor difference, but it broke the code I had written and I had to update the examples in this article.



Where Is Spry Going?

The future of Spry is up in the air at this point. This is a pre-release of the code, not even an alpha or beta. The Spry team isn't even sure where it's going. They want feedback from the community to help with Spry's direction. Since many of the developers of Spry are on the Dreamweaver team I expect it to be included in Dreamweaver at some point. I also expect to see many more UI components and hopefully a component standard so that the community can create its own custom Spry components. This is just the beginning of a very interesting technology with a lot of potential. Hopefully we'll see it fulfill that potential soon.

About the Author

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

kbrown@aboutweb.com

Welcome to the Future of Video on the Web!

REGISTER NOW!
www.iTVcon.com

CALL FOR PAPERS NOW OPEN!

 **LIVE SIMULCAST!**
AROUND THE WORLD ON SYS-CON.TV

iTV CON.COM
INTERNET TV CONFERENCE & EXPO 2006

Coming in 2006 to New York City!

“Internet TV is wide open, it is global, and in true ‘Web 2.0’ spirit it is a direct-to-consumer opportunity!”



For More Information, Call 201-802-3023
or Email itvcon@sys-con.com

Welcome to the Future!

Did you already purchase your “.tv” domain name?

You can't afford not to add Internet TV to your Website in 2006!

2005 was the year of streaming video and the birth of **Internet TV**, the long-awaited convergence of television and the Internet. Now that broadband is available to more than 100 million households worldwide, every corporate Website and every media company must now provide video content to remain competitive, not to mention live and interactive video Webinars and on-demand Webcasts.

20 years ago the advent of desktop publishing tools opened the doors for the creation of some of today's well-known traditional print media companies as well as revolutionized corporate print communications. Today, with maturing digital video production, the advent of fully featured PVRs, and significant advances in streaming video technologies, **Internet TV** is here to stay and grow and will be a critical part of every Website and every business in the years to come.

It will also very rapidly become a huge challenge to network and cable television stations: **Internet TV** is about to change forever the \$300BN television industry, too.

The Internet killed most of print media (even though many publishers don't realize it yet), Google killed traditional advertising models, and **Internet TV** will revolutionize television the way we watch it today. You need to be part of this change!

Jeremy Geelan
Conference Chair, iTVCon.com
jeremy@sys-con.com

PRODUCED BY
SYS-CON
EVENTS

List of Topics:

- > Advertising Models for Video-on-demand (VOD)
- > Internet TV Commercials
- > Mastering Adobe Flash Video
- > How to Harness Open Media Formats (DVB, etc)
- > Multicasting
- > Extending Internet TV to Windows CE-based Devices
- > Live Polling During Webcasts
- > Video Press Releases
- > Pay-Per-View
- > Screencasting
- > Video Search & Search Optimization
- > Syndication of Video Assets
- > V-Blogs & Videoblogging
- > Choosing Your PVR
- > Product Placement in Video Content
- > UK Perspective: BBC's "Dirac Project"
- > Case Study: SuperSun, Hong Kong

- | | |
|----------------|--|
| Track 1 | Corporate marketing, advertising, product and brand managers |
| Track 2 | Software programmers, developers, Website owners and operators |
| Track 3 | Advertising agencies, advertisers and video content producers |
| Track 4 | Print and online content providers, representatives from traditional media companies, print and online magazine and newspaper publishers, network and cable television business managers |

Innovative Tools

To Manage, Deliver and Track Streaming Media On the Internet

MediaConsole

Real-time



Using the Right Tools For the Job Is Critical to The Success of Your Business

VitalStream reliably delivers streaming media to large global audiences using an award-winning content delivery network that powers all of our services. These services come with innovative tools that support and enable a variety of business models for online media distribution including advertising, subscriptions, pay-per-view and more.

Introducing the New VitalStream Media Player for Flash

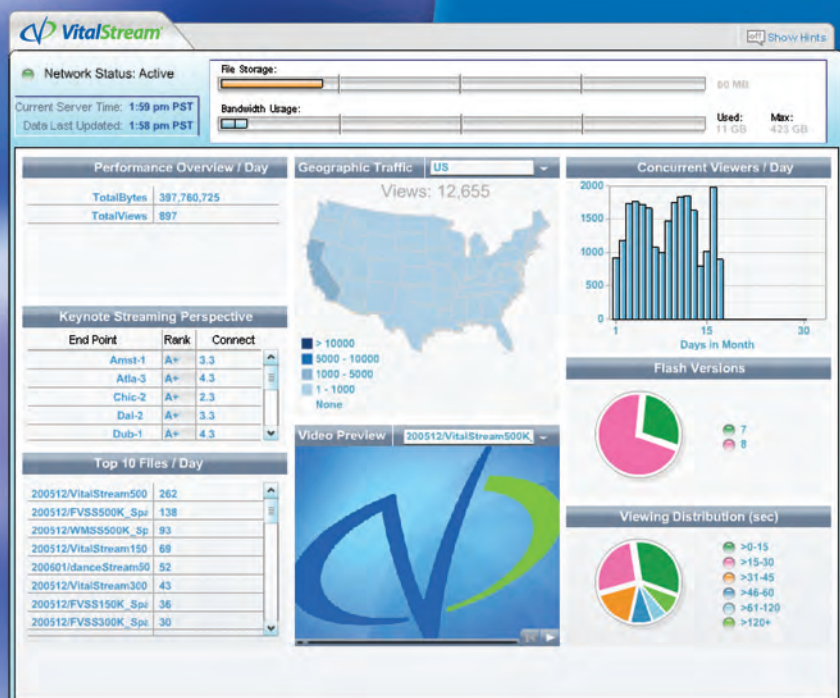
Continuing its commitment to innovation, VitalStream is proud to announce the new Media Player for Flash to help speed up your workflow by allowing you to test and play streaming Flash video locally on your computer before you upload the files to our servers.

Download this player for free at <http://www.vitalstream.com/tools/mxdj.asp>

Reporting



Reporting Dashboard



To learn more about VitalStream, call (800) 254-7554 or visit www.vitalstream.com



The World Leader In Audio and Video Streaming



By Oliver Merk

Flex for ColdFusion Developers

An introduction

There's been a lot of talk in the ColdFusion community lately about the newly released Flex 2. If you're new to Flex or haven't tried it yet, this article provides an introduction, from a ColdFusion developer's perspective, to what Flex is and is not.

I'll also put Flex in the perspective of what you already know as a ColdFusion developer, so that your introduction to this exciting and powerful technology will have a familiar context.

Like many of you, I've been using ColdFusion for several years, keeping up with the latest bells and whistles as they were introduced: the Java platform, CFCs, reporting, Flash forms. As an Adobe Certified ColdFusion Instructor, I made sure I kept up with these new features, even if I didn't use them day-to-day, simply because I knew I'd get questions about them.

TECHNOLOGY	ARCHITECTURAL CONCERN
Flex-created Flash & Actionscript	VIE
-HTML & Javascript	
ColdFusion Application Server	CONTROLLER
J2EE Enterprise	
Database XML	MODEL

TABLE 1

One thing has always remained constant though. I wrote code that would be interpreted by the ColdFusion Application Server and returned to the user's browser as HTML. Over the years, I learned how to combine technologies like JavaScript and CSS with my ColdFusion code to create robust, intuitive applications for my clients. I started separating a lot of the application logic into CFCs so that my code was reusable. I even started learning some basic Java so that I'd be better able to take advantage of ColdFusion's underpinnings.

Little did I know that arming myself with these snippets of knowledge and best practices would help ease my introduction to Flex enormously.

What Is the ColdFusion-Flex Relationship?

As Table 1 illustrates, Flex-created Flash takes the place of standard HTML output. The rest of your application architecture remains unchanged.

What Exactly Is Flex?

To understand what Flex is, you must first understand what a Flash movie is. A Flash movie, at its most simple, is a file with an SWF extension (the pros refer to them as "swiffs") that contains compiled code that renders an interface and occasionally make calls to back-end systems. The SWF is read by the Flash player, which most people (an estimated 97% of Web users) have installed as a browser plug-in, and rendered on-screen within the confines of the Flash movie. The Flash movie shows up on screen via some HTML tags that define its dimensions and other attributes.

To create a Flash movie, developers traditionally use Adobe's Flash IDE, a timeline-based application that is exceptionally well suited to creating animations. If you've never seen the Flash IDE, it's worth taking the time to download the free trial. The Flash IDE, however, was not originally intended to be a tool for application developers like us. While some incredible applications have been created using the Flash IDE, developers who come from a coding background have had a hard time getting comfortable building applications this way. I certainly did.

The Flex team was well aware of this and, to their credit, recognized that if Flash was going to play a significant role in the

next generation of Web applications, especially at the enterprise level, they would need to entice coders to get in the game. Wouldn't it be great, they thought, to be able to create Rich Internet Applications (RIAs) using only code?

And so, Flex was born.

Flex allows you, the developer, to create Flash applications using code. The good news is that the code is very similar to the HTML and CFML you already know.

Here is a simple example:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/
mxml">
  <mx:Label text="Hello World!" fontSize="22"
color="#ffff00"/>
</mx:Application>
```

Notice that Flex uses opening and closing tags, just like HTML and ColdFusion. In this case, the markup language is called MXML. Also, the first line tells you that MXML documents are standards-compliant XML files.

What Is ActionScript?

Good news about the tags. It gets better. You would be unlikely to create a complex HTML-based application these days without using JavaScript and CSS for things like form validation and DHTML effects. Well, Flex also has a scripting language, analogous to JavaScript, called ActionScript. If you know JavaScript, ActionScript will look fairly familiar. If you know Java, ActionScript will look very familiar.

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml">
  <mx:Script>
    <![CDATA[
      import mx.controls.Alert;
      public function doAlert():void {
        Alert.show('Hello Actionscript!');
      }
    ]]>
  </mx:Script>
  <mx:Button id="myBtn" label="Click me!" click="doAlert();"/>
</mx:Application>
```

You can probably guess that this snippet of code displays a button on screen that, when clicked, calls the doAlert function. Inside the function is a call to Flex's Alert class, which generates alert boxes, just like in JavaScript.

ActionScript is where the real work of your Flex application occurs. It can be used to manipulate the application's visual elements as well as gather, send, and retrieve data from remote sources such as Web services, Java objects, and even ColdFusion Components (CFCs).

Flex also uses Cascading Style Sheets syntax to style the visual

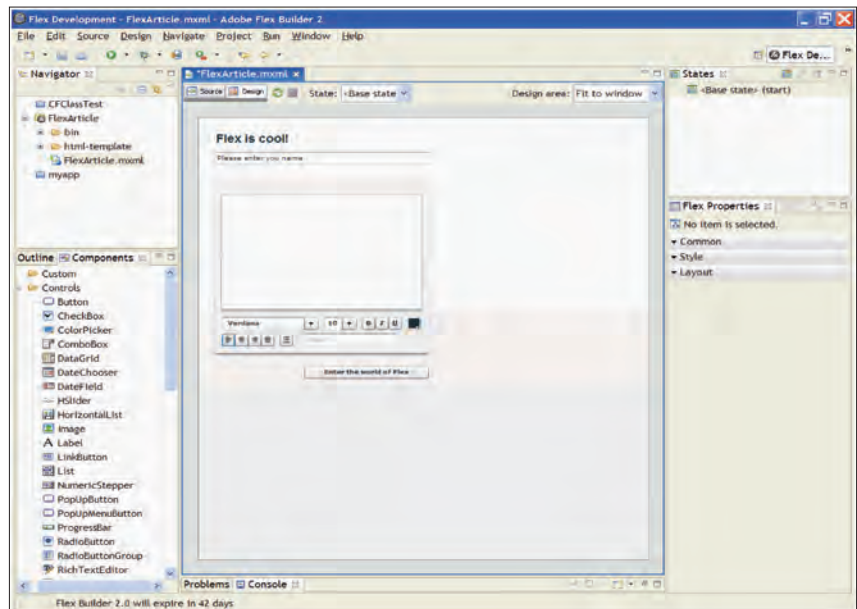


Figure 2 Flex Builder 2

elements of your application, so those CSS skills will come in handy with Flex as well.

What About ColdFusion?

Is ColdFusion no longer needed? Are your hard-earned skills going to be abandoned now that Flex has arrived?

Absolutely not.

Flex represents what's known as the presentation layer of an application; the view part of that model-view-controller stuff that's all the rage these days (see Table 1 above). That is, its primary concern is to render the interface for the user and provide an environment that is comfortable and intuitive. For example, we're all used to dragging and dropping things within our desktop environments, so Flex has this functionality built in – and it's easy to implement.

What Flex does not do is all the back-end transactional things that an applications needs. It cannot directly send a query to a database. It cannot POP an e-mail account. It cannot manipulate files on the server. Is this a drawback to using Flex? No. In fact, it's its greatest strength. Flex leaves the choice of back-end technology up to the developer or organization. This means that the initial investments made to create such systems can be repurposed with a rich front end.

Flex contains several built-in functions that can talk to various back-end technologies, including native connections to ColdFusion CFCs and Java objects, XML files, as well as Web services. That last one is important; Flex can talk to any Web service, regardless of the technology used to create it. This means that an organization that has invested heavily in their technology of choice, say .NET, can retain that code base, while adding a sophisticated presentation layer that is difficult or impossible to achieve with a straight HTML interface.

– continued on page 21



Your First Flex Application with a ColdFusion Backend

The wow factor plus usability



By Nahuel Foronda & Laura Arguello

Flex is a complete set of tools to develop rich Internet cross-platform applications based on the Flash platform. With Flex, you can create applications that not only have the “wow factor” necessary to please clients and users alike, but the “usability factor” necessary to make your application a real success.

Flex 2 has recently been released and can be downloaded at Adobe's Web site. This release includes Flex Builder 2 (an

IDE based on Eclipse) and Flash Player 9. At Adobe's Web site, you'll also find tools specifically for ColdFusion such as the ColdFusion/Flex Connectivity package and ColdFusion extensions for Flex Builder. If you know how Flex 1.5 works, you'll be happy to hear that Flex 2 doesn't require a Flex server and that you can develop and deploy applications with Flex 2 for free using the Flex SDK (if you don't use the IDE).

A Flex application communicates to external services, and we find ColdFusion to be a perfect tool for providing those services. In this article, we'll walk you through the construction of an application with a Flex frontend and a ColdFusion backend. We believe that the integration with ColdFusion is so smooth you won't even notice you're transferring data from a remote server. The application you'll construct is a simple to-do list, but it will let us show you several Flex and ColdFusion features.

To start, you'll need the tools mentioned above, specifically Flex Builder IDE and the ColdFusion connectivity package and extensions downloaded and installed.

Setting Up Your Environment

The application will use a database with only one table, so

you need to create a database with your favorite DBMS. This database should contain a table called “Task” with the following columns: id (varchar 35), description (varchar 1000), done (bit), and priority (smallint). Then register the data source in the CF administrator with the name “mytodolist.”

Now you’re ready to open Flex Builder and create a new project. While running the New Project wizard, specify that you’ll be using ColdFusion Flash Remoting Services because the application will be communicating to your ColdFusion server via Flash Remoting rather than other alternatives such as XML or Web Services. Assuming that you’re running the development version on your local computer (<http://localhost:8500>), you can select “Use local ColdFusion server” in the second screen of the wizard. In the next screen enter “mytodolist” as the name of the project in the default location and finally, in the last screen, specify “<http://localhost:8500/mytodolist/>” as the Output folder URL.

The extensions you downloaded and installed include RDS support for viewing files and data sources. Make sure you have

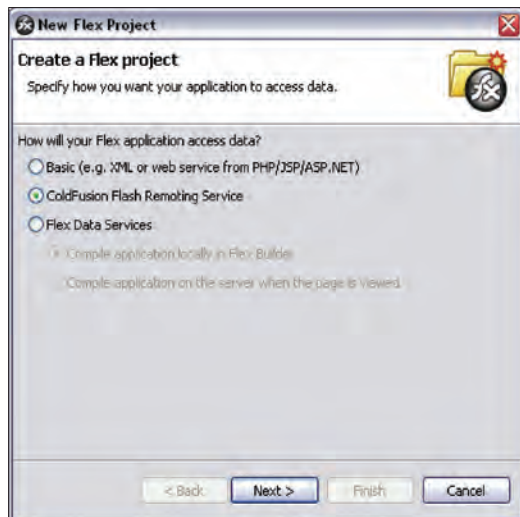


Figure 1 New project wizard

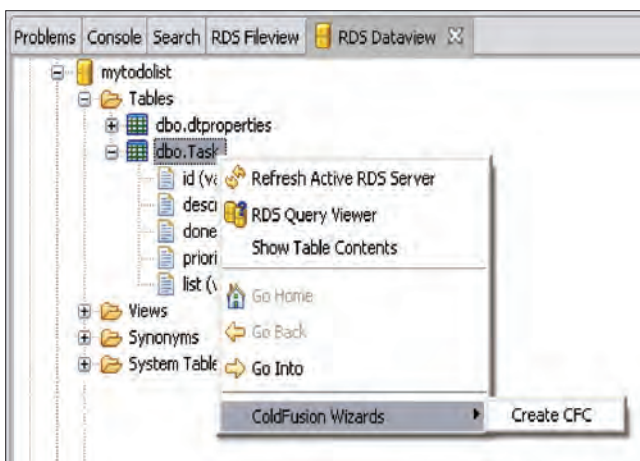


Figure 2 RDS dataview and create CFC wizard

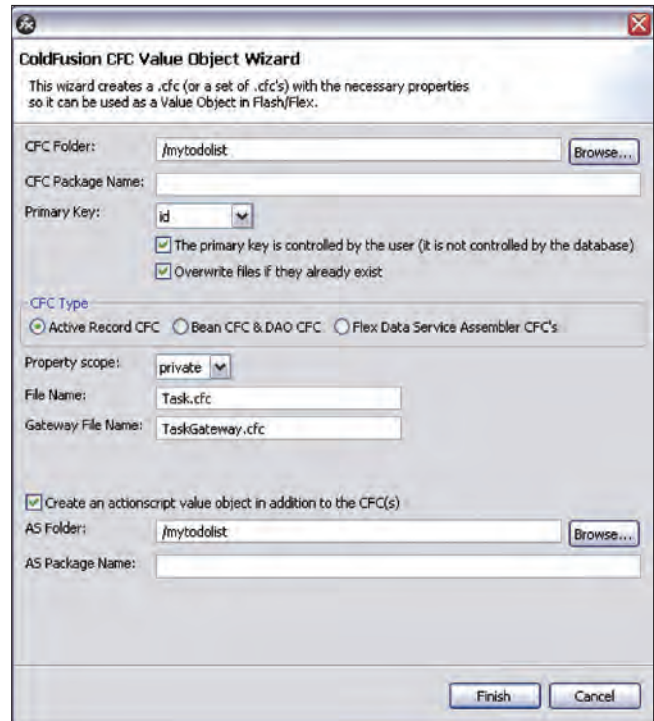


Figure 3 Create CFC wizard dialog

RDS enabled in your CF administrator and that you’ve set up the RDS preferences in Flex Builder. If your RDS connection is set up properly, you should be able to open the RDS Dataview (you can open it from Window > Other Views > RDS) and see the data source you created earlier.

Running the ColdFusion Wizard

Another handy feature included in the ColdFusion extensions is the ability to run automatic generation of basic ColdFusion components. The “Create CFC” wizard will generate all the ColdFusion code you need for this application. Then you’ll make a few changes to the components to meet specific needs.

From the RDS Dataview, look for your data source (“mytodolist”), open the tables node, and right-click on the “Task” table to run the ColdFusion wizards > Create CFC option of the context menu. This action will open a dialog that asks you what type of CFC you want and some other basic options. Make sure the CFC folder is your current project folder (mytodolist). Set the CFC package name to “mytodolist.” Also check the option “The primary key is controlled by the user” because the component will set the id instead of letting the database choose the id. You also want to create an ActionScript value object that matches your Task CFC. So you need to check “Create an actionscript value object.”

When you’re finished, you should have three new files in your project: Task.cfc, TaskGateway.cfc, and Task.as.

Overview of the Application

The application consists of a list of undone tasks and a form to add a new task. Each task contains an icon that indicates its

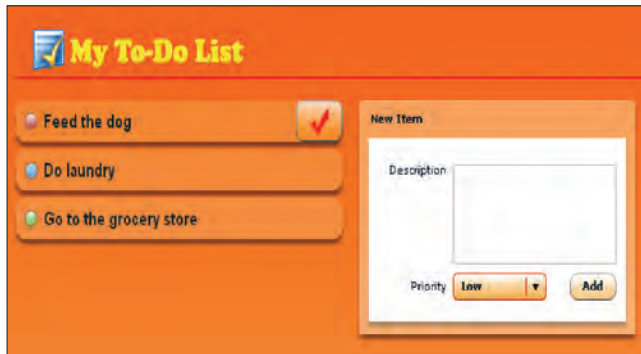


Figure 4 Application overview



Figure 5 TaskItem.mxml custom component

priority, a label that shows the description of the task, and a button that can mark the task done and remove it from the view. Figure 4 shows the finished application with styles applied.

To construct it, you'll use these components: VBox, Repeater, Panel, Button, Label, and form controls such as TextArea and ComboBox. You'll also create a custom component that will represent the view of each task item (TaskItem.mxml) and a custom component that will contain the edit form (EditForm.mxml). By creating custom components, you can reuse them throughout your application.

If you are wondering about the data, the application will get the list of tasks from a ColdFusion service and send requests to add new tasks and mark them as done.

Main Application File

The New Project wizard created an empty main application file called mytodolist.mxml. You'll use this file as the main canvas where you'll place the components and most of the ActionScript code. In a larger application, you will probably not want all your code in the same file, but for simplicity, you'll keep it there.

We have not yet described how to get the data from the server, but once the application gets the list of tasks, you'll need to store it in a local variable. To do that, declare the variable "tasks" inside a <mx:Script> block.

```
<mx:Script>
    <![CDATA[
        import mx.collections.ArrayCollection;

        [Bindable]
        private var tasks:ArrayCollection = null;
    ]]>
</mx:Script>
```

It's important to set this variable as "bindable" because you'll use it as the dataProvider of the repeater component.

In our layout, we want to have the list of tasks vertically positioned with one task below the other. By using a VBox component, we can add all the tasks and the VBox will position them how we want them. But the task list data will come from a ColdFusion service, so you don't know how many of them there'll be. So you'll use a repeater component that will "loop" over the collection of tasks and add them to the VBox.

```
<mx:VBox width="350">
    <mx:Repeater dataProvider="{tasks}" id="taskList">

        <TaskItem taskData="{taskList.currentItem}" />

    </mx:Repeater>
</mx:VBox>
```

TaskItem Custom Component

If you save mytodolist.mxml file, you'll get an error indicating the TaskItem can't be resolved. This happens because we put the tag <TaskItem> inside the Repeater but the component TaskItem doesn't exist yet.

To create this custom component, you'll go to the File menu and click on New > MXML component. It will extend the Canvas and have 100% width because we want it to take the width of the parent container: the VBox. Save it as TaskItem.mxml.

However, the compiler still complains that it can't find the TaskItem component. "But I just created it!" you say. The problem is that the main application doesn't know where to find it. Since you saved it in the same folder, you have to indicate that other components can be found there by adding the following attribute to the Application tag in the mytodolist.mxml file:

```
xmlns="**"
```

Lastly, declare the taskData variable in your new component (TaskItem.mxml) so that it will finally compile without errors:

```
<mx:Canvas xmlns:mx="http://www.adobe.com/2006/mxml" width="100%">
    <mx:Script>
        <![CDATA[
            [Bindable]
            public var taskData:Task = null;
        ]]>
    </mx:Script>
    <!-- place other code here -->
</mx:Canvas>
```

Why wouldn't it compile? Because when you put the tag <TaskItem> in the main application, you also specified the attribute: taskData="{taskList.currentItem}." This was necessary to be able to send the current task to the custom component during each loop of the Repeater. This variable has to be set as bindable so that you can bind to its properties in the controls inside this component.

If you look carefully, you'll see that this variable has the type "Task." This type refers to the ActionScript file automatically

generated by the ColdFusion wizard. The properties of this class match exactly those of your Task ColdFusion component.

An instance of the TaskItem component (Figure 5) shows the description and priority of the task and a button to remove the task. As described above, the data is set when the component is instantiated by the Repeater and accessible in the variable “taskData” inside the component.

You can use the Design view to drag an Image, a Label, and a Button and put it in the Canvas. The Properties panel of the Design view shows common properties of the selected component. In that panel, set the text property of the Label as: {taskData.description} and the source property of the Image as: images/priority{taskData.priority}.png. Note that you should have a folder called “images” with an image for every priority you want to show. These images should be called “priorityNUMBER.png,” where NUMBER should be replaced with the priority number (i.e.: 1, 2, 3).

Everything should compile, but unfortunately you won’t be able to see anything because there’s no data.

Getting Data from a ColdFusion Service

To load data from a service, you must use a RemoteObject. The simplest way to use a RemoteObject to call a ColdFusion service is by setting its destination to “ColdFusion” and its source to the path to that component. The application will call two methods on the TaskGateway component (generated by the wizard) so you can specify them explicitly inside the RemoteObject tag. When the response from the service comes back, you want a function to be called. The function to call will be different for each method, so you can specify it in the <mx:method> tag with the “result” attribute. Set the result of the method “getAll” to the function “tasksReceived(event)” and the result of the method “save” to the function taskSaved(event).”

```
<mx:RemoteObject
    id="myRemoteObject"
    destination="ColdFusion"
    source="mytodolist.taskGateway"
    fault="Alert.show(event.fault.faultstring, 'Error');">
```

Figure 6 Add new item form

```
<mx:method name="getAll" result="tasksReceived(event)" />
<mx:method name="save" result="taskSaved(event)" />
</mx:RemoteObject>
```

Inside a script block, implement the tasksReceived function as follows to set the source of the task ArrayCollection to what the service sent as a response of the getAll method:

```
private function tasksReceived(event:ResultEvent):void {
    tasks = new ArrayCollection();
    tasks.source = event.result as Array;
}
```

If you open TaskGateway.cfc, you’ll see that the method “getAll” returns an array of Task objects. When you set that array to the source of the tasks ArrayCollection, the Repeater will get populated with this data since you declared it as the Repeater’s dataProvider.

If Flex Builder didn’t add the required import statements automatically, add:

```
import mx.controls.Alert;
import mx.rpc.events.*;
import mx.events.*;
```

The other function you need to implement is taskSaved. For now leave it empty; you’ll complete it later.

```
private function taskSaved(event:ResultEvent):void {
}
```

Everything works, but...you still don’t see anything on the screen. That’s because you haven’t called the service yet. Since you want to load the list of tasks as soon as the application loads, put a function call in the applicationComplete attribute of the Application tag. You’ll set it to call a custom function called setUp():

```
applicationComplete="setUp()"
```

Then implement the function so that it sends the actual request to the server:

```
private function setUp():void{
    myRemoteObject.getAll.send();
}
```

Of course you still won’t see anything if your database is empty. To test it out, just add a couple of records manually. You’ll build the form to enter new items in the next section.

Adding a Form To Insert Items

Just like you created the TaskItem component, create another one called “EditForm.mxml” that extends the Canvas. In the Design view of this newly created component, add a Textarea with the id “descriptionArea,” a ComboBox with the id “priorityList” and a Button. In the Source view, add the priority items to the

ComboBox:

```
<mx:ComboBox id="priorityList">
    <mx:dataProvider>
    <mx:ArrayCollection>
    <mx:source>
    <mx:Object label="High" data="1"/>
    <mx:Object label="Medium" data="2"/>
    <mx:Object label="Low" data="3"/>
    </mx:source>
    </mx:ArrayCollection>
    </mx:dataProvider>
</mx:ComboBox>
```

Going back to the main application file (mytodolist.mxml), switch to the design view and drag a new panel and inside drag your EditForm component. At this point, your main application should have a list of tasks and a form to add tasks. But the form doesn't send the new task data to the server yet.

Using Events To Communicate Between Components

The goal is that when the user clicks on the “New item” form button, the task data gets sent to the server. Likewise, when the user clicks on the button on the task item, the task gets marked as done and this information gets recorded in the server. But both of these buttons and forms aren't in the main application where the RemoteObject tag is declared. That means you need a way to make the different components communicate with each other. Flex 2 has a powerful built-in event architecture that you can use to do exactly that.

First you need to make the form button respond to the click event. You'll implement a function called addItem() that will be invoked when the user clicks on the submit button:

```
<mx:Button label="Add task" id="addButton" click="addItem()"/>
```

The function must create a new event called saveItemEvent. Store the new task in the event and dispatch it so that other components that might be listening for this event can act on it.

```
<mx:Script>
<![CDATA[
import mx.events.ItemClickEvent;

public function saveItem():void {
    var saveItemEvent:ItemClickEvent = new ItemClickEvent("saveItem",
true);
    //create a new Task object
    var taskData:Task = new Task();

    //set its properties using the data entered in the form
    taskData.description = descriptionArea.text;
    taskData.priority = priorityList.selectedItem.data ;
    saveItemEvent.item = taskData;

    //announce this event
```

```
        dispatchEvent(saveItemEvent);
    }
    ]]>
</mx:Script>
```

Now you must make other components listen to that event. In the main application file, you created a method setUp() that was invoked when the application was completely loaded. In that same method you can specify the main application as a listener to any number of events, including the “saveItem” event your form is dispatching.

```
addEventListener("addItem",saveTask);
addEventListener("setAsDone",taskDone);
```

saveTask and taskDone are functions that will be called whenever the application gets notices about the “addItem” and “setAsDone” events.

saveTask simply sends the request to the server by using one of the methods defined in the RemoteObject tag:

```
private function saveTask(event:ItemClickEvent):void{
    var taskItem:Task = event.item as Task;
    myRemoteObject.save.send(taskItem);
}
```

Similarly, the setAsDone function sends a save request, but ensures that the task is set as done before sending the Task object:

```
private function taskDone(event:ItemClickEvent):void{
    var taskItem:Task = event.item as Task;
    taskItem.done = 1;
    myRemoteObject.save.send(taskItem);
}
```

Receiving Responses from the Server

If you recall, when you created the RemoteObject, each of the methods of the RemoteObject tag specified a function to be called when the response came back from the server:

```
<mx:method name="getAll" result="tasksReceived(event)" />
<mx:method name="save" result="taskSaved(event)" />
```

You already implemented the tasksReceived() function but left taskSaved() incomplete. This function will be called either because a new task was added or because a task was set as done. In each case, you should do something different. You could also implement two different methods in the service if you'd like to separate them.

If a new task was created, you want to add it to the main list of tasks, that is, to the “tasks” ArrayCollection that feeds the Repeater. If the task was updated (set as done), you want to remove it from the list.

```
private function taskSaved(event:ResultEvent):void {
    var taskItem:Task = event.result as Task;
    var i:Number = -1;
```



```
if (taskItem.done){//we must remove it
    //find task in list
    for each (var thisTask:Task in
tasks){
        i++;
        if (thisTask.id ==
taskItem.id){
            //remove this
            item
            tasks.
            removeItemAt(i);
            break;
        }
    }
}
else {
    tasks.addItem(taskItem);
}
}
```

Making Changes to the Generated ColdFusion Components

To make everything work, you'll have to make a couple of changes in the generated ColdFusion components. First you'll need to make the Task component generate a unique key before storing the data in the database:

```
<cfset var local0 = createUUID() />
<cfset setId(local0) />
```

That should be added to the "create" method.

Then your TaskGateway component should return the task it got in the "save" method:

```
<cffunction name="save" output="false"
access="remote">
    <cfargument name="obj" required="true" />

    <cfset obj.save() />
    <cfreturn obj />
</cffunction>
```


While you're at it, you may also want to change the getAll method so that it returns only the undone items, or you may want to delete the task when it's set as done.

Final Touches

As a final touch, you can add styling information to the application and to

each individual control. An application-wide style file can be added by using the <mx:Style> tag that references an external style sheet. In each control you wish to apply specific style information, you can use the styleName attribute and set it to be the name of one of your classes defined in the style sheet. We invite you to look at the source code to get ideas about styling. (See <http://www.asfusion.com>.)

As you can see, integrating Flex with ColdFusion is very simple and the CFC wizards greatly simplify the task. In addition, with the new Flex/ColdFusion connectivity features, you can send object instances from Flex to ColdFusion and from ColdFusion to Flex as if they were instantiated locally like you did with the Task.cfc-Task.as class mapping.

We hope this sample application gets you started using Flex and ColdFusion together to offer your users a nicer experience. We've only scratched the surface with this article; you can do much more with Flex. As your next steps, you can look at the sample code, read the tutorials at Adobe, and get your feet wet with an application of your own. 

About the Authors

Nahuel Foronda is one of the founders of Blue Instant (<http://www.blueinstant.com>), a web development firm specializing in Rich Internet Applications where he has been creating award-winning applications and offering training for the last five years. He also maintains a blog, called AS Fusion (<http://www.asfusion.com>), where he writes about Flash, ColdFusion and other web technologies.

Laura Arguello is one of the founders of Blue Instant (<http://www.blueinstant.com>), a web development firm specializing in Rich Internet Applications where she has been creating award-winning applications and offering training for the last five years. She also maintains a blog, called AS Fusion (<http://www.asfusion.com>), where she writes about Flash, ColdFusion and other web technologies.

nahuel.foronda@authors.sys-con.com

laura.arguello@authors.sys-con.com

Where Do I Start?

Download Flex Builder 2 from <http://www.adobe.com/products/flex/> and start playing. Once you've installed the product, start in Design View. This is the simplest way to learn MXML; simply drag the prebuilt components (lower left) to where you want them, then study the code that Flex Builder generates for you (see Figure 1). After a while, you'll likely just go straight to the Source View.


I would also suggest an official Adobe Flex training course. There are three Flex courses currently available, with more to follow. Visit the link in my bio for more information.

What Other New Things Will I Learn?

In addition to quickly creating a decent-looking interface, you will find that, as you get deeper into the RIA world, there are some concepts that may be new to you:

- For example, you may, though it certainly is not required, use object-oriented frameworks. The most popular one for Flex is called Cairngorm (http://www.macromedia.com/devnet/flex/articles/cairngorm_pt1.html).
- You will quickly learn about the ubiquitous Event object that permeates and gives life to Flex applications.
- You will start to abandon the restrictions of the traditional page request/response nature of the Web that we have tolerated for far too long.
- And much more.

Summary

As ColdFusion developers we are living in a very exciting time. Our years of experience are being put to a new use. The same joy we felt when we first learned ColdFusion is about to be rekindled with another technology that, like ColdFusion, makes seemingly difficult tasks easy and satisfying. 

About the Author

Oliver Merk is a senior consultant with New Toronto Group (www.newyyz.com). He has been using ColdFusion since version 1.5 and is a certified ColdFusion MX 7 Developer as well as an Adobe Certified ColdFusion and Flex Instructor. oliverm@newyyz.com.



How I Became a Hero at the Office

DOIM Online Application Services Module



By Gene Godsey

The purpose of this article is to provide an overview of the enterprise applications that I developed using the ColdFusion MX 7.1 product suite.

Currently I am the only ColdFusion developer on contract at the Ft. Stewart Department of Information Management (DOIM). Ft. Stewart, the largest Army base east of the Mississippi River, is home to the 3rd Infantry Division (Mechanized). It also has one of the largest installed rollouts of

desktop and laptop computers in the southeast, numbering in the thousands of workstation computers. The DOIM has the mission of supporting all units, activities, and tenants assigned to Ft. Stewart. If a unit or asset is deployed, support is still part of the mission.

Before implementing CFMX7.1, DOIM was running an implementation of another purchased CRM that did not meet their business needs. Management decided to look at other solutions that would reduce the cost of the product, ease administrative responsibility, speed up development, and be Web based. A product was needed that could integrate with the Army's infrastructure. CFMX7.1 was that product. It integrated with Active Directory, Windows Server products, IIS, Exchange, SMS, WSUS, and SQL Server.

Applications Developed

- Help desk
- Asset management
- Accreditation
- Computer repair
- LAN/WAN work order
- Security training and testing

Help Desk

The DOIM operates a staffed help desk to support our users. Help desk personnel use the help desk application to log trouble calls and create work orders. As they are entered into the system, each trouble call is categorized and routed to the appropriate personnel as needed. This not only includes the initial assignment of trouble, but escalation as well. Business rules are built into the application that allow the users to not only see what is assigned to them, but to receive e-mail alerts if necessary.

This application also allows the unit's Information Management Officer (IMO) to submit trouble ticket/work orders online. The IMO no longer has to wait for the next available help desk technician to answer the phone. The IMO can also check the status through the Web interface. This has been a great improvement from the business processes that were in place prior to CFMX7.1

Asset Management

This application was created to enable the appropriate personnel to manage the assets or real property that they are responsible for. When the organization purchases equipment, it is entered into the asset management system. Data recorded is relevant to the procurement cycle of that end item. Purchase order numbers, warranty expiration dates, and maintenance contract dates are some of the items tracked in the asset management application.

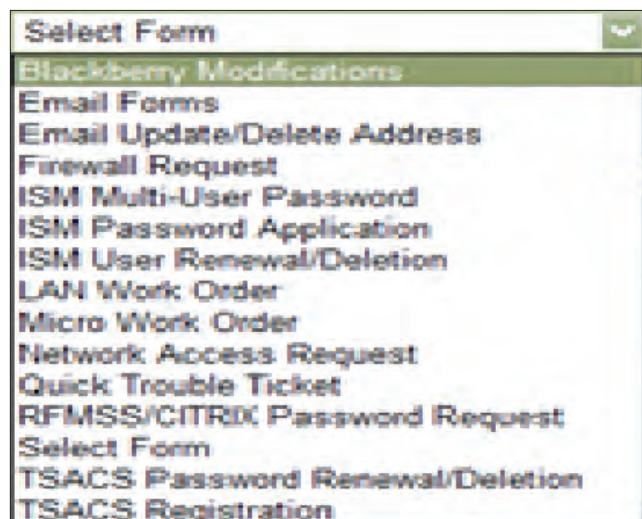


Figure 1 Dropdown Window



Figure 2

Again, the business rules are built in to provide on screen, as well as in e-mail alerts, information that is relevant to managing the end items. This application gives the appropriate personnel the ability to manage a product from cradle to grave. From the date the request is entered until the item is turned in for disposition, managing the asset, and its support agreements, is handled by this application.

Accreditation

One of the mandates imposed upon Army installations is the mission to accredit every item that is placed on the network. With CFMX7.1, I have created a Web-based system that allows the unit IMO to enter specific pieces of information about a device that will connect to the enterprise WAN. Embedded business rules in the application ensure that a request for accreditation flows to the appropriate personnel.

Once a device has been cleared to connect, various events occur, such as obtaining IP address assignments and receiving a machine name. At this point, the DOIM is now required to report this new asset, as well as perform various levels of management. This is where the accreditation application really shines. Through various techniques, I have CFMX7.1 tying into AD, SMS, and WSUS. The application has the ability to drill in or out depending on the needs of the user.

The application suite is an online help desk, asset management, LAN accreditation systems database, micro repair ordering system, and a division reentry triage used to reaccredit systems from the Middle East battlefield to the Ft. Stewart home base. There is also associated educational security training and testing with this existing suite. The first area of review will be the online help desk. This system allows users to submit a series of different requests via their given unit or organization. The different units can track their individual data and see if the DOIM approving authority has accepted or rejected their requests.

There is an extensive list of request types for the units to use with built-in routing and approval authority processing. The following is a brief list of those request types.

The other part of the help desk is based on dynamic e-mails sent to the Information Management Officers (IMOs) of the different units and organizations. This is critical for the com-

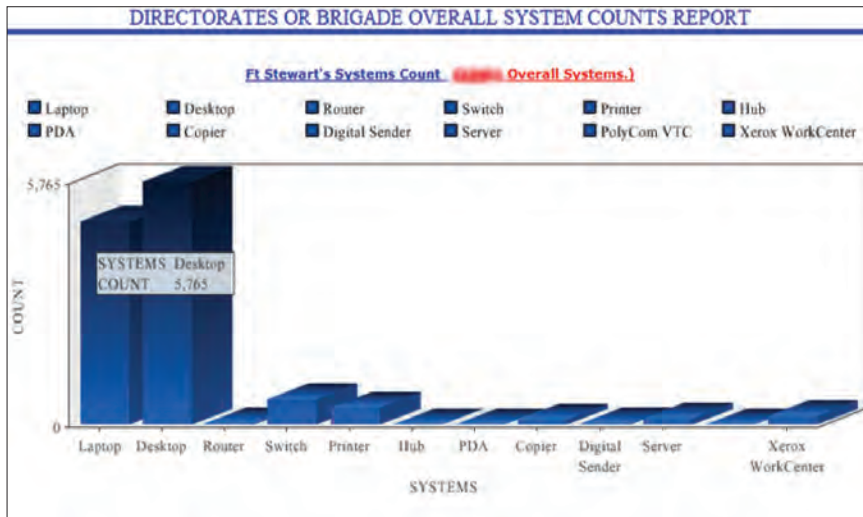


Figure 3

INSERT NEW COMPUTER SYSTEM

Date: May 02, 2006 16:06:52

☐ Check to insert multiple systems

POC Information

Serial:
 Phone:
 Room:
 Type:
 Warranty:

IMO Rank: **IMO:**
Email:
Building:
Purchased:

Figure 4

UPDATE COMPUTER SYSTEMS

Date: May 02, 2006 08:06:38

POC Information | **Computer Info** | **Accreditation Info** | **Network/Fielded Info** | **System Software**

Sys Name:
Device:
IPADD:
MAC:
DNS 1:
DNS 2:
User:

Processor:
Users Cnt:
Brand:
Q-W-N:
WDRS 1:
Modem:

Figure 5

UPDATE COMPUTER SYSTEMS

Date: May 02, 2006 08:06:38

POC Information | **Computer Info** | **Accreditation Info** | **Network/Fielded Info** | **System Software**

Application	Publisher	Version
Windows XP Media - KB933333	Microsoft Corporation	20060330.000218
hp deskjet 6400	Hewlett-Packard	1.03.0000
128 Memory Disc	SanDisk Corporation	1.0.0.000
Windows XP Media - KB933333	Microsoft Corporation	20060330.000218
Net-Golf-Cross 9.8 Client	Microsoft Corporation	20060330.000218
Windows XP Media - KB933333	Microsoft Corporation	20060330.000218
Microsoft Office 2003	Microsoft Corporation	11.0.0.000

Figure 6

munication between the IMO and the DOIM help desk personnel. LAN network connectivity, Systems Management and general computer-related help requests are mission critical.

Figure 1 shows an e-mail request form that is a Flash format form built with ActionScript validation to prevent users from sending bad requests.

Other forms and requests are specific to the required process, but the open-ended form is called the quick ticket. This is a fast and easy form that uses a custom tag with AJAX programming. There isn't any refreshing for this form so users can get to the point with their request and send it off fast. The turnaround for these requests is usually about an hour.

In this form the category and short summary fields are listed in a dynamic

AJAX nonrefresh mode. There is an HTML editor for a text editor that has a built-in Java spell check. This adds to a better user data entry experience. The user also has an immediate way to escalate the ticket to a person of experience by selecting the moved-up field. This will send an automatic e-mail to the administrator via ColdFusion's CFMail tag and relay off of the Exchange 5.5 mail server. Relaying is turned off at the LAN level to prevent spam from leaving the domain.

This particular form has a great deal of validation and user interface tools. Data binding is a part of the form that's required and it makes the user's experience easier. The form has sensitive data, which is processed so only the necessary administrators see the data they need. There is also an interactive interface with other databases so that the security clearance information can be dynamically verified.

The turnaround time to create an account is about one day. Delays are usually caused by the security database or issues with security clearances.

The IMO also has an online discussion capability that allows them to have discussions within a forum of technical topics that pertain to their needs. This discussion board is linked right off of the main help desk navigation pane.

The administrative dashboard for the site is the console, which can access pending tickets or individual reports that pertain to the admin user that is logged in at that time (see Figure 2). The network credentials are passed and captured in the database via CGI variables. This allows the administrators to track who is doing what to different requests. In addition, the dashboard allows administrators to see where tickets have been elevated to and to lock transactions. When an administrator grabs a trouble ticket, that ticket is locked and belongs to them until the ticket is closed or escalated to a higher level of expertise.

The accreditation database is the army's way of meeting the needed Information Assurance requirements. This part of the DOIM suite is integrated with SMS, Microsoft Active Directory, and a WSUS server. The systems are entered by the IMO and the SMS server pulls data, which populates the rest of the required

information that the information management officer needs to report. A basic report would look like Figure 3 using CFMX charting. The report taker can click on the given bar and drill into the records. In addition, they can recursively drill into those records to manipulate the data if necessary.

Figure 4 shows the basic data entry form for adding a new system. There is very little data to enter here and the check box allows the template to loop for ease of use in adding more than one computer. This reduces the data entry for the IMO. Then a populated IP request form is sent where the accreditation

process begins. Once the system has been granted an IP address by the network-approving authority, the system is added to a Microsoft SMS systems collection. That data is queried by hooking the Active Directory's (AD) system-naming convention into the organizational databases units' names. This is done with the SQL substring function. Once the AD name is in the table, the military unit identification code (UIC) kicks in and the SMS database and the accreditation database tables are joined. The UIC codes are unit to unit all over the military. Using the report queries from SMS, I have been able to generate data grids of software, hardware, and other network protocol information about these individual systems.

Figure 5 shows an SMS pull of all IP and required hardware information. Figure 6 shows a data grid of all of the software that the computer system has loaded. There was filtering to pull out files that may or may not support these applications. If I didn't do this, there would be thousands of data records that would not have been of use for standard security reporting. This is for the information management officers and information assurance manager to use for security reporting. There are many more functions

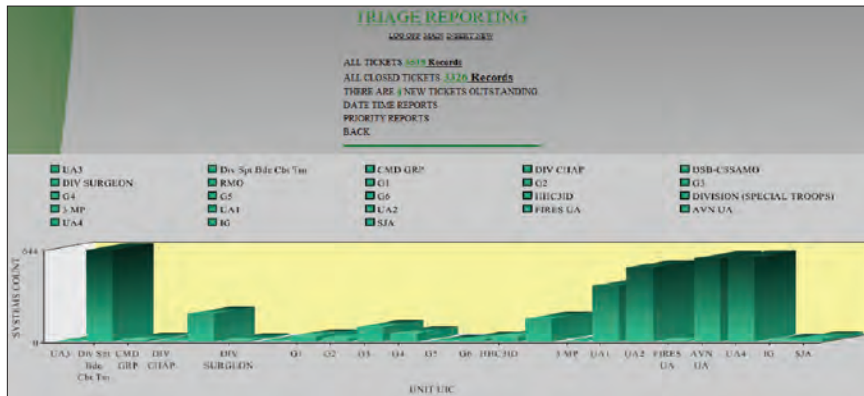


Figure 7

Efficient Web Content Management

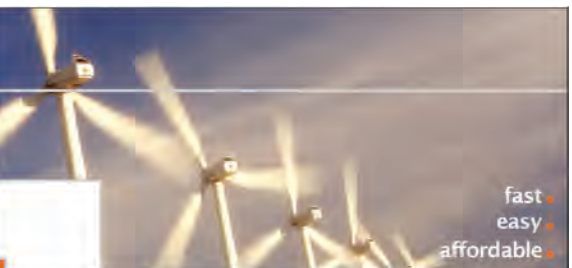
CommonSpot™ Content Server is the ColdFusion developer's leading choice for efficient Web content management.

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

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

Evaluate CommonSpot today.

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



features.

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

1.800.940.3087
www.paperthin.com

Paper Thin

© Copyright 2005 PaperThin, Inc. All rights reserved.

but the primary purpose here is to use this module as a means to control the unit's network systems assets and get the approval to operate on the LAN from this database.

The Triage system module was developed for the division's systems reentry from the Middle East to the Ft. Stewart Domain. All the systems needed to be checked, imaged, and prepared for the security posture of this current network. Once the systems go through the hardware overhaul, they are given back to the unit's users to do their daily business at their given places of work. The Triage ties into the accreditation database because once the systems are added into the Triage, there is a dynamic IP and domain request through the accreditation database (ADAT2005). The approval authority then grants or denies the request to be added. This is tracked in the help desk and reported back on the Triage. There are many elements to measure with the Triage. There is also a service-level agreement that is time stamped and put on the clock when the system is brought in. The repair people have a limited amount of time to get work completed per system and will also measure the amount of time required by manufacture warranty for repair work. Once the systems are successfully completed, the organization is contacted and they claim their systems. The military authorities track the progress of the Triage's activities in real time through a ColdFusion chart (see Figure 7). This chart allows them to click through it and go to the individual record sets and




see what the status is of the given system.

There are other modules to this ongoing scaling CRM application that are critical to the daily business of the DOIM and Ft. Stewart's network. To maintain the brevity of this article, I will give a short description of a few more. There is a Micro Repair Ordering System. This is basically the installation's solution for a computer repair shop for classified and unclassified systems. The order request is sent and tracked through the help desk from

the side of the end user. From there a ticket is generated at the Micro repair shop and there is an SLA in place to expedite the order of repair. Repair work, parts ordering and hours required are tracked for the DOIM and the division's resource management office. These are the people that maintain the budget for the military. This is the way that the Micro shop is compensated as well as the repair tickets are tracked for various levels of reporting and tracking.

Another module of interest in this CRM is the asset management system. The three main areas here are the tracking of DOIM automation assets, the life cycle of vendors' maintenance agreements, and vendors' contracts. All of the routers, switches, hubs as well as all other network devices that are currently on the network or are still sitting in the box are tracked in this database. Devices are tracked by contract number and contract number can be tracked by device. This is a time driven application and there are events written to send alerts to the hand receipt holder of the given device that expiration dates are coming to fruition with the conclusion of the contractual agreements.

The last area I want to mention is the educational online training that we do. Before a user can become an information management officer, he or she needs instructor-led and Web-based training. Once the training is complete, there are several online exams that are fully dynamic for the reporting and certification of the proposed training taken. Reports are generated and roll ups and recursive drill downs of the unit's numbers and users are recorded. That enables the higher command groups to see how the division and installation stacks up for their reporting purposes.

The DOIM online application services will continue to grow and scale based on the ever changing technologies that come down the pike. The Ft Stewart Garrison and 3rd Infantry Division's rapid deployment ability forces the need for this CRM to keep up with the ongoing changes in business processes. Currently CFMX7.1 is an outstanding solution for RAD iterations to keep up. There will eventually be integration with other technologies like .NET, Java, and XML but the mainstay of this will most likely remain in the ColdFusion arena. 

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

Java for ColdFusion Programmers?



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

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

About the Author

Gene Godsey is a ColdFusion developer with the Department of Information Management at Ft Stewart. He was a senior ColdFusion developer and system integrator for 3ID during OIF-I (Operation Iraqi Freedom). Gene holds a masters in education from the University of Miami.

gene.c.godsey@us.army.mil

REPRINT!

Once
you're in
it...



...reprint it!

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

Contact Dorothy Gil
201 802-3024
dorothy@sys-con.com

REprints

SYS-CON
MEDIA

CFDJ Advertiser Index

ADVERTISER	URL	PHONE	PAGE
AjaxWorld	www.ajaxseminar.com	201-802-3022	30, 31
CFDynamics	www.cfdynamics.com	866-233-9626	2
CFDJ	http://coldfusion.sys-con.com/	800-303-5282	43
CommunityMX	www.communitymx.com/trial		4
EdgeWebHosting	edgewebhosting.net	1-866-334-3932	6
Hal Helms	halhelms.com		26
HostMySite	www.hostmysite.com	877-215-4678	39, 52
Hotbanana	hotbana.com/cfdj	866-296-1803	27
InterAKT	www.interaktonline.com/		51
IT Solutions Guide		201-802-3021	47
iTVcon.com	www.itvcon.com	201-802-3023	11
Macromedia	www.macromedia.com/go/cfm7_demo		3
Macromedia Studio 8	www.macromedia.com/go/studio8_mxdj		9
Paperthin	www.paperthin.com	1-800-940-3087	25
Real World Flex	www.flexseminar.com	201-802-3020	45
VitalStream	www.vitalstream.com/tools/mxdj.asp	800-254-7554	12, 13

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

Get more with Hot Banana!

Web Content Management + Active Marketing

Hot Banana software builds efficient, affordable, fast, very easy-to-use and measurable Web sites. And they're ColdFusion-based and optimized for marketing performance.

That's what makes us different and better!

Call us today at

1-866-296-1803 to schedule a live 1-on-1 demo, or visit
hotbanana.com/cfdj
to learn more.



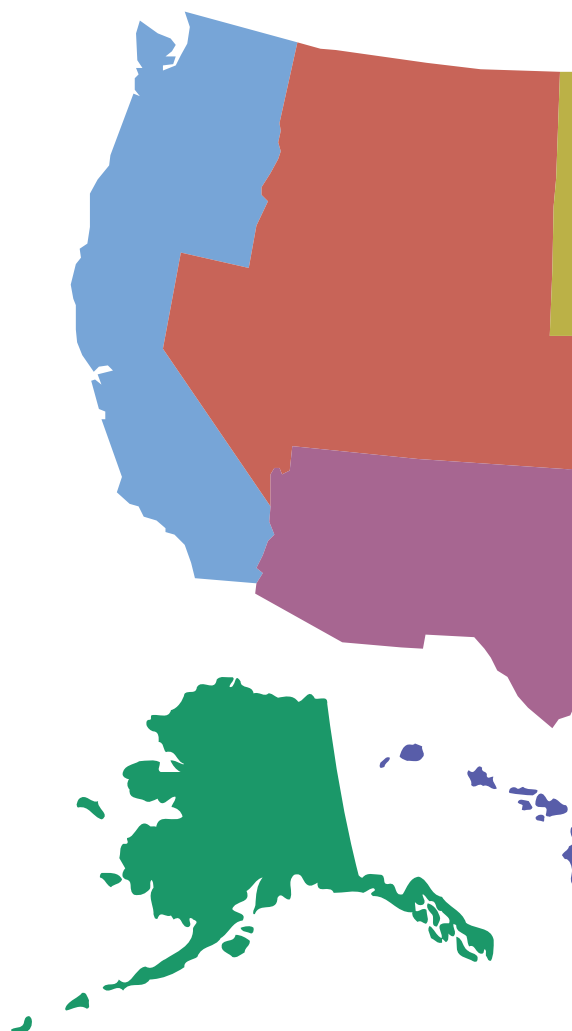
hotbanana

ColdFusion

For more information go to...

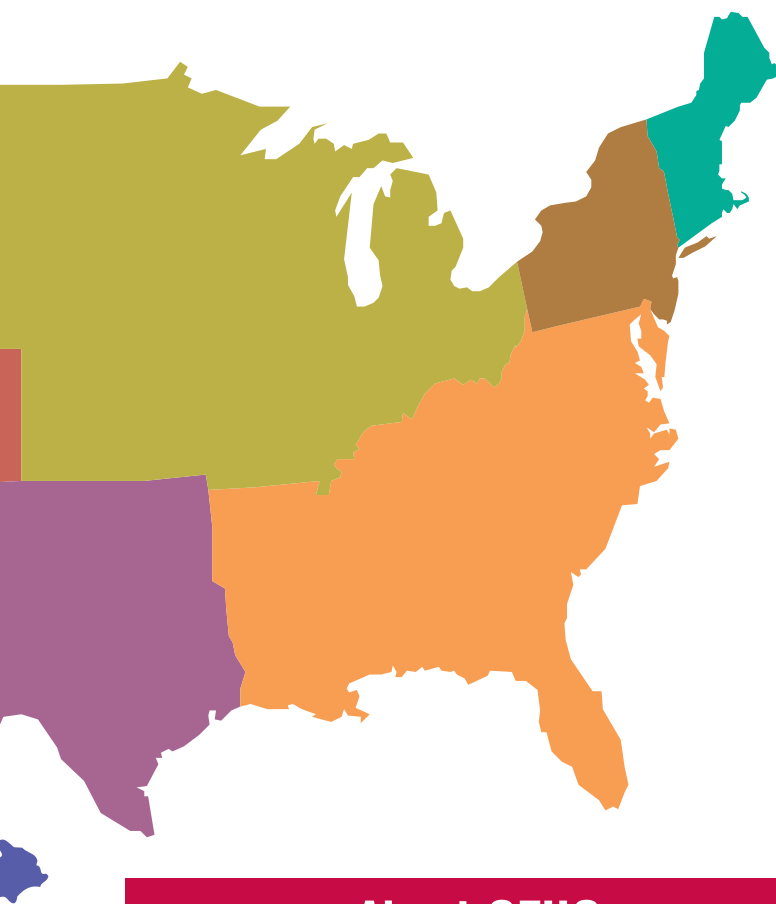
U.S.

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



User Groups

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



About CFUGs

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

INTERNATIONAL



Australia
ACT CFUG
<http://www.actcfug.com>

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

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

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

Canada
Kingston, ON CFUG
www.kcfug.org

Canada
Toronto, ON CFUG
www.cfugtoronto.org

Germany
Central Europe CFUG
www.cfug.de

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

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

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

Scotland
Scottish CFUG
www.scottishcfug.com

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

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

Spain
Spanish CFUG
<http://www.cfugspain.org>

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

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

Turkey
Turkey CFUG
www.cftr.net

United Kingdom
UK CFUG
www.ukcfug.org



Rich Internet Applications: AJAX,

www.AjaxWorldExpo.com

AJAXWORLDTM

CONFERENCE & EXPO

SANTA CLARA SILICON VALLEY

**SYS-CON Events is proud to announce the first-ever
AjaxWorld Conference & Expo 2006!**

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

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

CALL FOR PAPERS NOW OPEN!

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

Flash, Web 2.0 and Beyond...

REGISTER TODAY AND SAVE!

→ **October 3-4, 2006**

→ **Santa Clara Convention Center**
Hyatt Regency Silicon Valley
Santa Clara, CA

→ **To Register**

Call 201-802-3020 or
Visit www.AjaxWorldExpo.com

→ **May 7-8, 2007**

First International AjaxWorld Europe
Amsterdam, Netherlands

“Over the two information-packed days, delegates will receive four days’ worth of education!”

Early Bird*

(Register Before August 31, 2006)
..... **\$1,495****
See website or call for group discounts

Special Discounts*

(Register a Second Person)
..... **\$1,395****
See website or call for group discounts

(5 Delegates from same Company)
..... **\$1,295/ea.****
See website or call for group discounts

On-Demand Online Access

(Any Event)
..... **\$695**

*Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

**OFFER SUBJECT TO CHANGE WITHOUT NOTICE,
PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

“It Was The Best AJAX Education Opportunity Anywhere in the World!” —John Hamilton

Topics Include...

Themes:

- > Improving Web-based Customer
- > Interaction
- > AJAX for the Enterprise
- > RIA Best Practices
- > Web 2.0 – Why Does It Matter?
- > Emerging Standards
- > Open Source RIA Libraries
- > Leveraging Streaming Video

Technologies:

- > AJAX
- > The Flash Platform
- > The Flex 2 Framework & Flex Builder 2
- > Microsoft's approaches:
ASP.NET, Atlas, XAML with Avalon
- > JackBe, openLaszlo
- > JavaServer Faces and AJAX
- > Nexaweb
- > TIBCO General Interface

Verticals:

- > Education
- > Transport
- > Retail
- > Entertainment
- > Financial Sector
- > Homeland Security

GROUP DISCOUNTS AVAILABLE:
— 5 Delegates from Same Company —
for only \$995 (each)
— Register a Second Person —
for only \$1195

**Hurry! Limited Seating
This Conference Will Sell-Out!**



LIVE SIMULCAST!
AROUND THE WORLD ON SYS-CON.TV

Receive **FREE** WebCast Archives of Entire Conference!

The best news for this year's conference delegates is that your "Golden Pass" registration now gives you full access to all conference sessions. We will mail you the complete content from all the conference sessions in seven convenient DVDs after the live event takes place.



► This on-demand archives set is sold separately for \$995



SYS-CON
EVENTS

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

VISIT WWW.AJAXWORLDEXPO.COM FOR THE MOST COMPLETE UP-TO-DATE INFORMATION

Security Matters

Is more always better?



By Hal Helms

It seems that not a week goes by without another story of a major organization inadvertently leaking private data. In one recent week, representatives of a Rhode Island government agency reported that tens of thousands of

credit card transactions on a government-run site had been stolen by Russian data thieves. In the same week, the private firm, Providence Home Services, announced that backup tapes of patient records – some 365,000 of them – had been stolen from one of their employees' car.

Horror stories like these leave executives dizzy with fear of the liability they could bear if sensitive customer data were to be exposed. Network administrators, too, feel the pressure to ensure that such breaches will not happen on their watch. But fear rarely leads to wise decisions and, in the case of security, management too often neglects to examine the security risk profile of their organization, opting instead for a security paranoia that can lead to the equivalent of a security lockdown.

A recent experience got me thinking about the hidden costs of security. A network administrator, rightly concerned about maintaining security within the company, instituted a series of policies that made the business of software development very difficult to carry out. Networks were locked down to specific MAC addresses, and outside

source control repositories were portrayed as disasters waiting to happen. At one point, the idea of using Superglue to seal USB ports (to prevent unauthorized data removal) was seriously discussed.

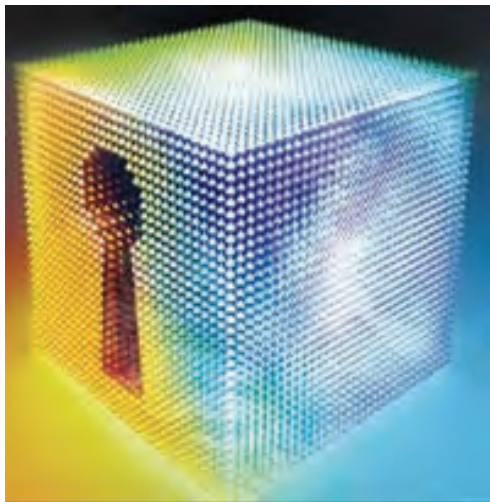
It would be both easy and inaccurate to view this as the sole result of an overzealous network administrator. Worse, such a simplistic analysis that localizes blame prevents the organization from examining the security choices that are being made. If we are to make sane choices, we must first recognize that choices are being made.

It's often assumed that where security is concerned, more is always better. But is it? The discussion (if there is any discussion at all) over the amount, type, and degree of security needed is skewed by the lack of differentiation between the risk profiles that different organizations have. Certainly where such things as payroll, credit card, social security, or health data are concerned, the need for data security is paramount. But for software development houses, the real damage of losing source code or employee contact information is much, much less. The problem is that the same solution is often urged for both types of risk profiles. "More is better" only works if we fail to recognize the real cost of security.

Thought leaders in the security community often come from the type of environments described above where it would be hard to overstate the validity of security concerns.

Security conferences, security whitepapers, security briefs all detail the possible ways security can be compromised and this mindset filters down to network administrators. We can't blame them for wanting to do their job well!

An appropriate security response is impossible without a clear understanding of the risk profile of the company. Unfortunately, talking about risks is often unpopular. One tool I've found to be very helpful in thinking clearly about risks is what I call the "postmortem game." It's played like this: imagine that at some point in the future, a catastrophe occurs. It could, for example, be the failure of a project, a large initiative, or



even an entire company. If you've ever been involved in such a failure, you may have also been involved in the obligatory "post-mortem" where concerned parties meet to try to understand why the failure occurred and to determine how to prevent its reoccurrence. At their worst, postmortems are exercises in affixing blame to certain individuals. But such "analysis" is a terrible mistake for it deprives the organization from understanding the real nature of such failure.

Edwards Deming, the legendary management consultant often credited as the person who transformed Japanese products from the decidedly inferior perception they held in the 1960s to their present lofty status, had much to say about the nature of failure. Deming recognized that in a complex system such as a business, a significant failure is almost always a part of the system itself, not of individuals.

Quality (or the lack of it), he asserted, is built into the system. To illustrate this, audience members at his lectures were asked to come to the stage to participate in his "red bead game." Players were given special paddles that were used to draw small red and white beads from a large bowl. The paddles were designed to hold exactly 50 beads.

The players had no control over which beads fell into the cavities in the paddles. The ratio of red beads to white beads was fixed so that, over time, each draw of the paddle averaged 40 white beads and 10 red beads.

The goal given to the players was to draw as many white beads as possible. Players whose draws selected more white beads were praised while those whose draws selected too many red beads were criticized. White bead players were given promotions, made "Employee of the Month," given "bonuses," and even selected as management material, while their red bead counterparts were scheduled for special training, received bad reviews for their performance, and even risked firing. To inspire better performance, banners signaling that "Quality is Job # 1" were installed, but since the outcome of the draws was outside of the players' control, no amount of "management programs" could possibly affect the outcome. The "red bead game" was a powerful illustration that success or failure is determined by the system, not the individual.

But postmortems don't have to go down this dreary route. Instead, they can be used to find ways in which the system produces undesirable results. In the "postmortem game," failure is imagined – and players are asked why such failure occurred. They are, in effect, asked to uncover the most likely risks of failure. Given the imaginary nature of the failure, there are no individuals to blame. Instead, the players find themselves analyzing the viability of the system. The "post-


mortem game" can produce a remarkably clear risk profile and lead to discovering ways to improve the system so that failure remains imaginary.

What does this have to do with security? I recommend that you and your colleagues try the "postmortem game." What role does lack of security play in possible failure? If it plays a significant role, you will know that security ought to be a major concern. Often, though – especially with software development – other reasons come to the fore: lack of product acceptance, slowness to market, etc. Now, we can begin to examine the costs of excessive security.

What are those costs? Lack of productivity is one of the chief ones. Developers may find that the constant drag caused by excessive security undermines their best efforts. Implementing security that imposes draconian restrictions is like weighing down an athlete. Performance cannot help but suffer.

Worse, developers may lose their enthusiasm for producing great software. Working against constant friction imposed by security constraints can sap even the most energetic of developers.

There is another kind of friction – that which occurs between different departments. Developers will resent the network administrators who make their work so difficult while network administrators will be upset that developers seem so heedless of the need for security. Infighting and conflict results. Relations are strained. But the real cause for these problems is the failure of management to (1) properly assess their company's risk profile and (2) communicate this throughout the company.

One of management's key tasks is mediating between competing interests such that decisions in the best overall interest of the company are made. The amount, type, and extent of security within an organization are issues too crucial to be left for any single department to decide upon. Where security is concerned, a naïve posture that "more is better" places the entire system at risk and may produce a result where the "postmortem game" is no longer an imaginary one. 

About the Author

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

hal@halhelms.com

"Developers may find that the constant drag caused by excessive security undermines their best efforts"

Interface Customization Part 1:

The basics of themes, styles, and skins



By Tariq Ahmed

One of the great things about Flex is that you get to spend most of your time working on the business logic and workflow, and less on the cosmetic aspect. This is because the Flex default's look-and-feel

is great! The only problem of course is that most people won't spend much time customizing the look, resulting in many Flex apps looking the same. In Flex 1.5 many folks customized the color scheme, but few ventured into modifying the actual skins of the components.

The designer approach would be to modify the graphical aspects of a theme, a laborious process of editing many items in various Flash source files ("FLAs") using the Flash Professional IDE.

The second approach let advanced Flex programmers create extremely customized interfaces; however this approach made it difficult for designers and developers to hand off the graphical "assets" of a project.

In Flex 2 things got a whole lot easier. In this article we'll look at the basics of changing the look-and-feel of your Flex application by introducing Flex Themes, Styles, and Skins.

Terminology

Before we get into it, let's quickly define what all the terms means.

Themes are simply a collection of style definitions, images (a.k.a graphical assets), and skins. Themes give you a lot of control and convenience in terms of how you package it all together and distribute it. You can distribute your theme as separate individual files, or ideally compile it into a theme SWC (commonly pronounced as "swick," a sort of JAR/ZIP-like archive format for Flash).

Styles are similar to the CSS you've been using for your Web applications. In fact, Flex's styles follow the CSS 2.0 syntax and support many of the common style properties that you're used to.

Although many of the visual components and controls support a variety of style properties that let you control fine-grain details like the corner radius of a button, fill gradients, and transparency. They're still based on a certain look; using skins you can completely alter the graphics and interface behavior.

Out of the Box

Out-of-the-box there are a couple of things you can do to easily change the look of your app. By default the Theme Flex uses is called Halo and it supports a style property called `themeColor`.

In HTML you can assign predefined colors like Red and Blue to certain CSS properties. Flex supports the same thing, but includes four interesting colors ready-to-use. They include:

- `haloBlue` (the default)
- `haloOrange`
- `haloGreen`
- `haloSilver`

To use them with the `themeColor` is pretty simple. To change it on an application-wide basis, modify your main Application tag to include this:

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
  themeColor="haloOrange">
```

You can also do this on a per-component instance basis as well. For example:

```
<mx:Button label="My Button" themeColor="haloGreen">
```

Now don't get too excited. The difference is rather subtle in that the most noticeable change is when you mouse over various components. In the case of the button it changes the outer highlight, and with the Datagrid (see the graphic below) it changes the Header column.

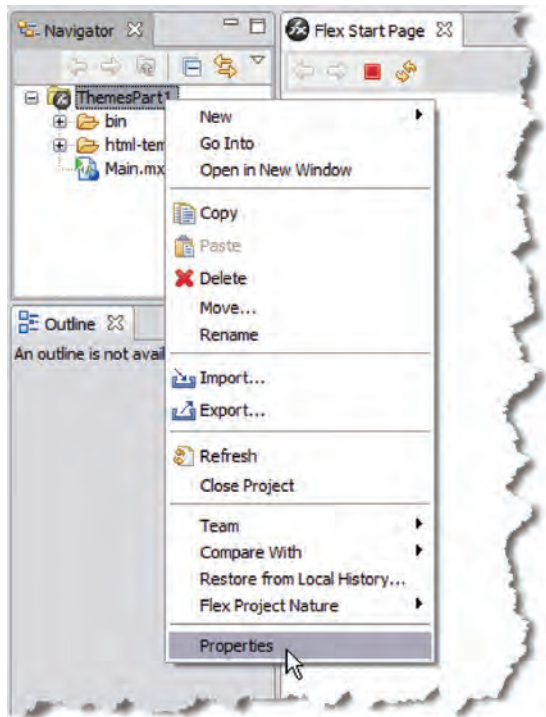


Fortunately there's a much more dramatic change that you can employ with the out-of-the-box themes. The themes that come with Flex 2 are:

- Halo (the default, embedded in the framework.swc)
- HaloClassic (haloclassic.swc)
- Ice (Ice.css)
- Institutional (Institutional.css)
- Wooden (Wooden.css)
- Smoke (Smoke.css)

You'll find the source of these files located under the <flex builder root>\Flex SDK 2\frameworks\themes directory. To make quick use of them:

- Start off with a Project in Flex Builder 2 either by opening an existing one or creating a new one.
- Using the mouse, right-click on the project folder, and select properties



- Select the Flex Compiler section
- Under "Additional compiler arguments," append the -theme parameter located as the desired theme file. For example:

```
-theme "C:\Program Files\Adobe\Flex Builder 2\Flex SDK 2\frameworks\
```

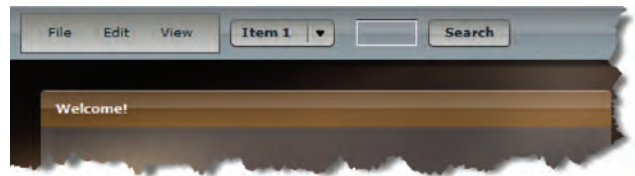
```
themes\Smoke.css"
```

- Now build and run the application

If all goes well you've instantly transformed your application from this:



To this:



Get Your Style On

Let's take things a bit farther and talk about the essence of styles. For all intensive purposes, styles are your theme and give you a vast amount of control.

At the basic level there are three ways of defining your styles and the style properties.

The first is called the Inline Style, and this is the same as in HTML where you style on-the-fly as needed.

For example in HTML, you could use an Inline Style to customize some text:

```
<span style="font-size:18px;font-family:Tahoma">Hello Style!</span>
```

Similarly in Flex, you can use Inline Styles that are implemented as properties on the component or control that you're using. For example:

```
<mx:Text text="Hello Style!" fontFamily="Tahoma" fontSize="18"/>
```

The second and more flexible approach is to use a local style definition. This would be similar to using the <style> tag in HTML somewhere in your Web page.

As in HTML, there are two types of these definitions. The Type Selector style is one that applies to all instances of that definition, and the Class Selector style let's you apply it to specific components.

In HTML you could do this:

```
<style>
input
{
    font-family:Tahoma;
    font-size:14px;
```

```

        color:#FF9900;
    }
    .myStyle
    {
        font-family:Verdana;
        font-size:18px;
        color:#CC0000;
    }
</style>

<input value="use type definition">
<input value="use class definition" class="myStyle">

```

Which results in:



With Flex you can do the same thing using the `<mx:Style>` tag as follows:

```

<mx:Style>
    TextInput
    {
        font-family:Tahoma;
        font-size: 14px;
        color:#FF9900;
    }
    .myStyle
    {
        font-family:Verdana;
        font-size:18px;
        color:#CC0000;
    }
</mx:Style>
<mx:TextInput text="use type definition"/>
<mx:TextInput text="use class definition" styleName="myStyle"/>

```

Which produces nearly the exact same thing:



Although you can find all the CSS properties for all the components in the Flex documentation, the easiest way to experiment with styles in Flex is to use the Flex Style Explorer (See figure 1). You'll find the link to this in the Flex Start page (select Help->Flex Start Page), and it provides an interactive tool to generating the style definitions that you can cut and paste into your Flex application.

Lastly, and ultimately the most modular approach would be to store your style definitions in a separate file known of course as the external style sheet. Once again Adobe makes this aspect easy to take advantage of by making it similar to what you already know from the land of HTML.

So pretty straightforward stuff; copy your Type and Class definitions into a .css text file then in your MXML file put the fol-

lowing in the file in use:

```
<mx:Style source="/path/to/your/stylesheet.css"/>
```

One thing you may have noticed when I introduced the out-of-the-box themes – a bunch of them were CSS files. Which begs the question: Can you source them in using the `<mx:Style>` tag? Yup! As mentioned before, style definitions are the root of your theme.

Just something to remember, though, is that you can't use the `<mx:Style>` to source in a theme SWC file; and using the `-theme` compiler option gives you extended flexibility to control what theme files to use during the build and release phase of the project (particularly useful for large projects).

Flex gives you many deployment options here so use whatever's most convenient.



Figure 1: Flex Style Explorer

The Skinny on Skins

For those of you who use WinAmp or Firefox you may have played around with their skins and seen how the look can be drastically altered. Flex lets you do the exact same things and more.

There's actually two ways to skin. Graphical skinning uses images to define parts or all of a component, and programmatic skinning lets you use code to define the way a component looks and behaves.

Here we're going to look at the graphical way of using JPEGs, GIFs, and PNGs to define the look of a component. When skinning you take into consideration the states of a component or control; the common states that most controls exhibit are the mouse off, mouse over, and mouse down states. One of the most popular things in HTML to represent graphically is the button, and to add some pizzazz you could use some JavaScript to add a rollover effect that swapped out the default image to a mouse over state image.

It's very simple in Flex; you use CSS/Styles to specify what assets to use for your graphical skins. In the Button Class (`mx.controls.Button`) documentation you'll notice it supports many Style properties, but namely the following skin-related properties:

- disabledSkin
- downSkin (mouse down)
- overSkin (mouse over)
- upSkin (mouse not over)
- selectedDisabledSkin

- selectedDownSkin
- selectedUpSkin
- selectedOverSkin

So using your favorite editor such as Photoshop, create three images to represent the states for mouse down, over, and up. Note that Flex will recognize transparent backgrounds in GIFs, PNGs, and SWFs.

If you want to take it a bit further you can also create images for the disabled state (i.e., `<mx:Button enabled="false"/>`), as well for the selected skins that are used to skin the background and the border of the button.

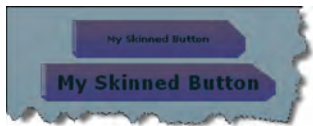
To test them quickly, use the inline approach:

```
<mx:Button label="My Skinned Button"

overSkin="@Embed('button_over.png')"
downSkin="@Embed('button_down.png')"
upSkin="@Embed('button_up.png')"/>
```

Compile and run your application and you should see a button that changes when you mouse over it and click down on it.

If your button is big enough to contain the text it will retain its original size, but if it wasn't Flex will automatically scale it to a big enough size. Add a `fontSize="25"` to your Button tag and see how Flex scales it:



The whole Embed business is a directive that tells Flex to import the image during compile time. You can abstract the name of the actual image files out of the Button into a script area of your MXML and embed the image elsewhere so that your Button doesn't need to know the exact filenames involved by doing:

```
<mx:Script>
<![CDATA[
[Embed(source="button_over.png")]
[Bindable]
public var buttonOver:Class;
[Embed(source="button_down.png")]
[Bindable]
public var buttonDown:Class;
[Embed(source="button_up.png")]
[Bindable]
public var buttonUp:Class;
]]>
</mx:Script>

<mx:Button label="My Skinned Button" overSkin="{buttonOver}"
downSkin="{buttonDown}" upSkin="{buttonUp}"/>
```

Of course even better is just wrapping that all up into a Style:

```
<mx:Style>
```

```
.skinnedButton
{
    overSkin: Embed(source="button_over.png");
    downSkin: Embed(source="button_down.png");
    upSkin: Embed(source="button_up.png");
}
</mx:Style>

<mx:Button label="Skin Class" styleName="skinnedButton"/>
```

Fairly painless wouldn't you say? Well, you may notice there's a bit of an issue; when it comes to scaling, your image gets distorted. You want only the middle to scale, not the corners.

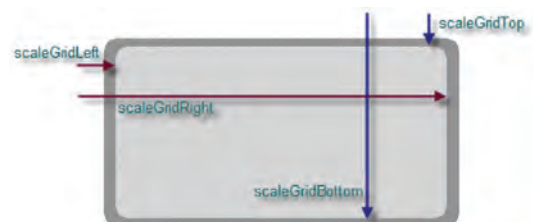
There's a simple remedy for this and it's called Scale-9 formatting. It's similar in spirit to using HTML tables to create a header at the top of your Web page that resizes properly by repeating the cell backgrounds, while using non-repeating images for the parts that remain fixed in dimension. Except in Flex the inner part of the image will scale as needed, corners won't scale, and the top and bottom parts will stretch accordingly.

This is done by using four additional attributes called `scaleGridTop`, `scaleGridBottom`, `scaleGridLeft`, and `scaleGridRight` when embedding the image. For example:

```
[Embed(source="button_over.png",scaleGridTop="18",scaleGridBottom="31",scaleGridLeft="8",scaleGridRight="158")]
```

The numbers are relative to the top and to the left as follows:

Now you've got a skin that will preserve itself more proportionally!



Conclusion

The goal of this article was to warm you to altering the look-and-feel of your Flex apps by relating the things that you're used to when building HTML-based ColdFusion apps with how they're done in Flex.

There's actually a lot more to discuss such as Style inheritance, limitations, programmatic styling, and skins, using SWFs for themes and graphical assets, compiling theme SWC files, and a whole lot more.

About the Author

Tariq Ahmed is the manager of product development at Amcom Computer Services, a former project lead at eBay, and host of www.cflex.net. He specializes in leveraging technology with process engineering to reduce operating costs while maximizing revenue potential. You can find him on the Web at www.dopejam.com.

tariq@cflex.net

Getting Started with Flex 2

An introduction



By Jeff Houser

I'm going to postpone the second part of my RSS aggregator article to tie this column into this Flex-themed issue. Have no fears, though, it will be back in full force in the next issue.

Flex, as I'm sure most people know, is a

way for programmers (you, me, and us) to create Flash movies.

The focus of Flex is not on animation and drawing little fancy pictures; it's on creating advanced interfaces, which are used to create Rich Internet Applications (RIA). It is a "Flash for programmers"-oriented product. Macromedia had long been pushing the concept and benefits of Rich Internet Applications, so it's great to see Adobe taking up the charge and finally making them accessible to all.

Flex 2 was released at Adobe's CFUNITED keynote (a few days ago to me), so this issue seems appropriately timed. I thought I'd take the space in this beginner's column for an overview of Flex 2 and talk about why you want to care.

Putting the Pieces Together

Everyone who has seen Flex from the beginning viewed it as an amazing and revolutionary product. If you think of ColdFusion as a way to build HTML pages on the fly, Flex was a way to build Flash movies on the fly. Unfortunately, most people found the price tag to be a serious setback to Flex usage. Adobe has addressed those concerns head on with the release of Flex 2.

These are the components that make up the Flex 2 suite of products:

- **ActionScript 3:** The language of Flash has always been ActionScript. With the release of the Flash 9 Player comes a new version of ActionScript. ActionScript has always been used to provide advanced functionality in a Flash movie.
- **MXML:** MXML stands for Maximum Experience Markup Language and is a form of XML. You can use MXML to create Flash movies with Flex. Most things in ActionScript have a parallel in MXML, and vice versa.
- **Flash 9 Player:** Flex applications will run only on Flash Player 9. Flash Player 9 adds support for ActionScript 3,

and offers many performance enhancements over Flash 8.

- **Flex SDK:** The Flex SDK is everything you need to build Flex applications. It contains a command-line compiler along with all the built-in Flex components (which includes a bunch of user interface elements). You can write MXML code in any editor of your choice, use the SDK to compile it to a swf file, and then deploy it to a Web server of your choosing. There have been rumors that Adobe hopes the release of the SDK will allow for the creation of third-party tools for generating Flex applications. I haven't heard of any tools being built yet, but this definitely bodes well for the long-term release of the community. Did I mention that the Flex SDK is free?
- **Flex Builder 2:** Flex Builder 2 is an Eclipse-based editor for building Flex applications. Although you can use the Flex SDK to build them for free, Flex Builder offers many advantages, including tag insight and a step-through debugger. Flex Builder is available as a standalone product or as an Eclipse plug-in. I like to use the plug-in version so that my Flex applications can easily reside next to CFclipse applications.
- **Flex Charting Components:** The Flex charting components are available as a standalone package or as an add-on to Flex Builder. They make it easy for you to generate charts in Flex, as well as allow for drill down and roll over functionality.
- **Flex Data Services:** Flex data services allow you to push data

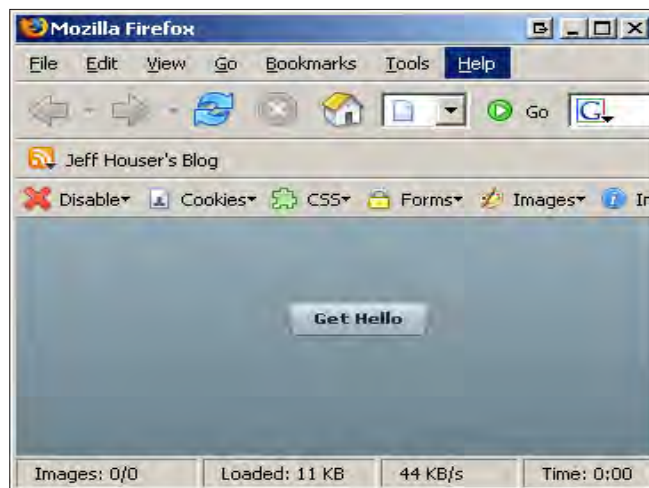


Figure 1: Hello World Pre-click

to the browser. It integrates with ColdFusion very nicely through the use of an event gateway. This is a feature that was unavailable in Flex 1.5, and can be very powerful in some applications.

Those are the important pieces of Flex. You can download the Flex components from the Adobe Website at <http://www.adobe.com/cfusion/tdrc/index.cfm?product=flex>. You can update

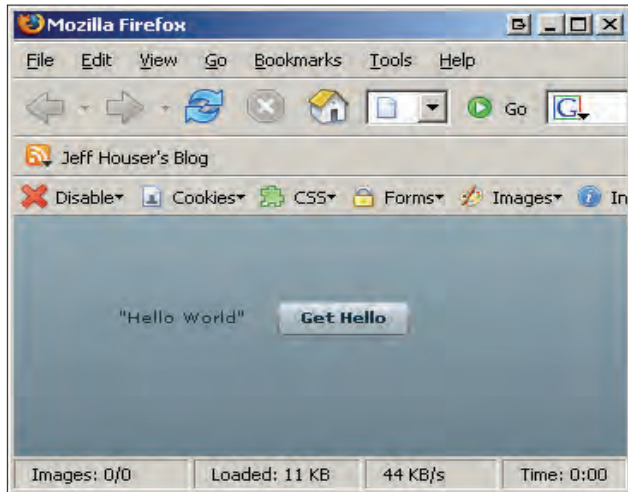


Figure 2: Hello World Post-click

your Flash player at http://www.adobe.com/shockwave/download/download.cgi?P1_Prod_Version=ShockwaveFlash. For the remainder of this article, I'm going to give you an introduction in how Flex and ColdFusion work together. When working with Flex 2 and ColdFusion, you'll want to install the ColdFusion 7.02 updater. You can download that from http://www.adobe.com/support/coldfusion/downloads_updates.html.

Accessing a CFC from Flex

Flex can call CFCs directly using Flash Remoting; an update to ColdFusion's Flash Remoting components is located in the 7.02 updater. This update helps ColdFusion talk to something you built in Flex. To demonstrate, I'll start with a simple `helloworld.cfc`:

```
<cfcomponent>
<cffunction name="GetHello" output="false" access="remote"
returntype="string">
    <cfreturn "Hello World">
</cffunction>
</cfcomponent>
```

If you are not familiar with CFCs, there are a plethora of resources for learning about them. You might start with one of my previous columns on them at <http://coldfusion.sys-con.com/read/47203.htm>. This component has no instance variables and only contains a single "GetHello" method. The method returns

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



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

the string “Hello World.” This is simple stuff, and you all know it, right? Great, let’s look at some Flex code!

First, you’ll need to define a Flex application, like this:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
layout="absolute">
</mx:Application>
```

All the MXML code that you write will go in the `mx:Application` block. In CFML, all tags start with `CF`. In MXML, all tags start with `mx:`. Code blocks work the same way in either language. The `mx:Application` works, conceptually, the same way that a `cloop` does. Now we can add a label and a button to our code:

```
<mx:Label id="Result" x="59" y="58"/>
<mx:Button x="154" y="56" label=" Get Hello"/>
```

Remember that this code goes in the `mx:Application` block. I used Flex Builder 2 to easily place the label and button, but if you are using the SDK without Flex Builder, you can specify the location of the elements using the `x` and `y` coordinates as shown in the code. This code will show you an empty label with a button next to it. The label, at present, doesn’t contain any text. The button displays the text “GetHello” but doesn’t actually do anything yet. The label is given an ID “result”. This is so we can reference it later to assign it a value. I did not give the button an ID because we won’t need to access it programmatically.

Next you need to tell Flex how to find your CFC. To do that, I used the `RemoteObject` tag and placed this code in my MXML file:

```
<mx:RemoteObject id="helloWorld" destination="ColdFusion" source="htdocs.
experiments.flex.helloworld">
  <mx:method name="GetHello" result="GetHello_handler(event)" />
</mx:RemoteObject>
```

The code, once again, goes inside the application block. The `RemoteObject` tag has an ID, which allows us to access it in code, a destination that is a special distinguisher, and the source. The source is the Web location of your CFC. When accessing the CFC remotely, it must be Web accessible (unless you change settings to allow you to access CFCs via a ColdFusion mappings, but such a configuration is beyond the scope of this article). Inside the `RemoteObject` block there is one method that we respond to: “GetHello”. The `mx:method` tag accepts two arguments: a name and the result. The name is the name of the method on the CFC. The result is the name of a local method that will be called when the Flash Player gets the results from calling that event.

The next step in our code base is to write the `GetHello_Handler`, which is written in ActionScript. You can put ActionScript in a MXML page using the `mx:Script` tag:

```
<mx:Script>
<![CDATA[
import mx.rpc.events.ResultEvent;
import mx.utils.ObjectUtil;

private function GetHello_handler( event:ResultEvent):void {
```

```
Result.text = ObjectUtil.toString(event.result);
}
]]>
</mx:Script>
```

After the script tag comes the CDATA. This tells the XML parser to ignore the text in the script tag. ActionScript is not a valid XML dialect. (I’m not saying that’s bad, though.) Then I import the two objects that are used in the function. Importing objects in this manner is not common in ColdFusion development, but if you’ve worked with older versions of ActionScript or Java, you’ve probably seen it. It just says, “I need this object, so make it available to me.”


The function should look similar to a CFScript function; I specified the function as private. Then comes the function keyword, followed by the name of the function. Next comes the list of arguments. This function only has a single argument, the `ResultEvent`. Then comes a colon followed by the return type. This function doesn’t return anything. The one line of code takes the result from our function call, translates it to a string, and assigns it to the text of our `Result` label. It sounds more complicated than it actually is.

Type in the code (or copy and paste from the Web version), compile it, and execute it. You should see something similar to Figure 1. Click the button. Unfortunately nothing happened as the button was not told what to do yet. A click event needs to be added to the button. The new button code will look like this:

```
<mx:Button x="154" y="56" label="Get Hello" click="helloWorld.GetHello()" />
```

The click event refers to the `helloWorld` remote object and says, “Execute the `GetHello`” method on that object. You should recognize this syntax for calling the method from your use of CFCs inside ColdFusion. When the button is clicked, the Flash player goes to the remote object and calls the method. When the method result is returned, the Flash player looks for the “`mx:method`” tag and executes the result function. Recompile the code and try it out. Click the button and you should see the Hello World text display next to the button.

Conclusion

Adobe has done a fantastic job of making ColdFusion the best back-end tool for developing Flex applications. Included with Flex Builder are some Eclipse extensions that work well with ColdFusion, including RDS support and some code generators. I’m just scratching the surface of what can be done with Flex and how you can combine it with ColdFusion. I’d love to see what you are going to do with this technology, so be sure to let me know! See you in a month. 

About the Author

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

jeff@instantcoldfusion.com



Visit the *New*

www.SYS-CON.com

Website Today!

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

24/7

FREE NEWSLETTERS

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

SYS-CON.TV

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

BLOG-N-PLAY!

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

WEBCAST

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

EDUCATION

The world's leading online i-Technology university

RESEARCH

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

MAGAZINES

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

INTERNATIONAL SITES

Get all the news and information happening in other countries worldwide

JUMP TO THE LEADING i-TECHNOLOGY WEBSITES:

IT Solutions Guide

Information Storage+Security Journal

JDJ

Web Services Journal

.NET Developer's Journal

LinuxWorld Magazine

Linux Business News

Eclipse Developer's Journal

MX Developer's Journal

ColdFusion Developer's Journal

XML Journal

Wireless Business & Technology

Symbian Developer's Journal

WebSphere Journal

WLDJ

PowerBuilder Developer's Journal

User Interface Layout in Flex 2.0

Planning your space

By Peter Ent

Creating a user interface is just like making a floor plan. You have your “room,” which is the screen, and you have the “furniture,” which is your user interface elements. This article describes many of the ways to position those elements on the screen using XML tags (which we call MXML.)

Flex applications are written with a combination of component declarations as MXML tags and ActionScript programming code. A “component” refers to any object in Flex that has a visual aspect. Buttons, Sliders, Labels, and CheckBoxes are examples of “control” components. There are other components that are very important to the layout of a Flex application. These other components are “containers” in which you put controls to position them on the screen.

There’s a Layout Manager in the Flex architecture. This is the entity that is responsible for carrying out the layout instructions. Once the components are created, the Layout Manager asks every container for its size. Each container in turn asks each child for its size. This continues until the Layout Manager has a pretty good idea of how big everything is. Then it carries out the placement instructions as best it can.

Layout

The easiest way to place components is with absolute positioning and setting the x and y coordinates. The origin, (0,0), is in the upper left-hand corner. For example, to position a button so that it’s 10 pixels from the left edge and 30 pixels from the top edge, type this MXML tag:

```
<mx:Button label="Push Me" x="10" y="30" />
```

Simple enough. You can use this technique to position all of your components.

That’s all well and good, but now suppose you want a layout like this:



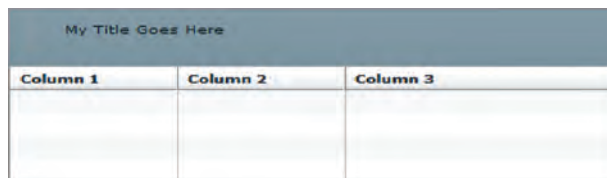
Besides position, most components have width and height properties. To start, you could create your layout like this:

```
<mx:Text text="My Title Goes Here" x="35" y="10" />
<mx:DataGrid x="0" y="50" width="200" height="200" />
```

Now comes the question: How do I make the DataGrid take up the entire width of the application space and how do I make the DataGrid use up the rest of the height? In other words, you have no idea how big the user will make their browser. It could be 200x250. Or it could be 1024x768. It would be nice if the components could automatically expand to fill the space. Actually, that’s easy. Not only can you specify the width and height of a component in pixels, you can also specify it as a percentage of the container in which they are placed. This is an important fact and we’ll come back to it shortly.

```
<mx:DataGrid x="0" y="50" width="100%" height="100%" />
```

Using 100% for the width and height makes the DataGrid use up the space in those directions. When I first saw this I expected that I would have to know the height of the Text control, or the percentage of space that the Text control was using. Not necessary. The Flex Layout Manager figures it out. It knows you want the top of the DataGrid to be positioned at 50 pixels. Having a height of 100% means it takes up the rest of the vertical space. When you run this Flex application you can resize the browser and the DataGrid will adjust its size.



Subscribe Today!

SAVE 16%

12 Issues for **\$89⁹⁹**

OFFER SUBJECT TO CHANGE WITHOUT NOTICE

SPECIAL ISSUE: COLDFUSION SECURITY

COLDFUSION Developer's Journal

September 2004 Volume 6 Issue 9

SANDBOX SECURITY
By Alex Hübner page 8

Security: ColdFusion Security Best Practices Knowing the security risks are there is half the battle
Bryan Murphy 16

Security: Public-Key Encryption Making strong encryption nearly painless
Wayne Graham 20

cfcUnit: ColdFusion Unit Testing Framework Paul Kenney's version
Harry Klein 30

<bf> on <cf>: Defending ColdFusion Against...
Ben Forta 36

Real-World Uses: Using ColdFusion for Network Monitoring Find the exact piece of data you need
Ryan Emerle
Matthew I. Fusfield 42

Security: Top Ten Web Security Tips You can't be too careful
Michael Smith 48

AN ADVANCE LOOK AT BLACKSTONE

COLDFUSION

ColdFusionJournal.com

web services EDGE
Web Services Edge 2005 East
February 15-17, 2005
Hyatt Regency Center
Boston, MA
See page 18 for details

Editorial Reaching Deeper
Simon Horwith page 5

Community Focus Problems Securing Your Applications?
Simon Horwith page 7

CF101 ColdFusion's Application Framework
Jeffrey Houser page 28

CFUGS ColdFusion User Groups
page 40

FOUNDATIONS 'Selling' ColdFusion
Hal Helms page 46

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

CF Community Tales from the List
Simon Horwith page 7

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

CFUGS ColdFusion User Groups
page 42

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

Training: Fast Track to Flex
Macromedia Training 'Flexes' Its Muscles
By Issue 10

CFML: Harnessing the Power of SQL Server Using Stored Procedures
Use of stored procedures in your CFML architecture
page 32

Foundations: A Web Application Framework
A perfect marriage between ColdFusion and .NET
page 28

<BF> on <CF>: ColdFusion and Data Abstraction
basic object function
page 30

PDFs: Creating Your CF Application
only functional, not just a pretty face
page 40

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

SYNCON MEDIA

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

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

COLDFUSION Developer's Journal



But what about that title Text? Suppose you'd like it to be centered above the DataGrid and move left and right as the browser window's width changes. That's easy too.

Flex has another way to position components called constraints and it's probably the method you'll use the most. Constraints let certain container components (those that allow absolute positioning) shift their content according to the parameters you specify. Using constraints is a lot better than using x, y, width, and height, because it positions the components dynamically and lets them adapt to the size of the browser window.

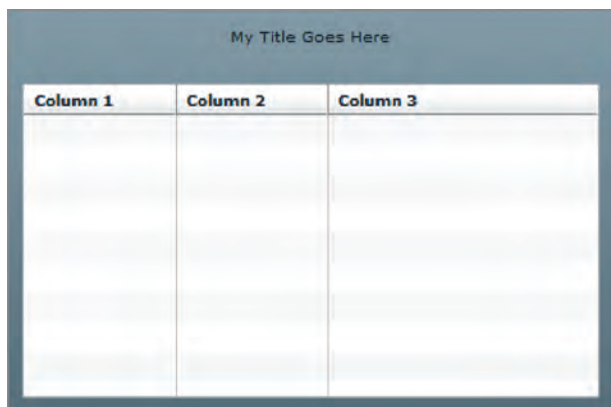
Let's start with the DataGrid. So far the DataGrid is at the edge of the browser space. Using constraints can give it some margins to make it look nicer:

```
<mx:DataGrid left="10" top="50" right="10" bottom="10" />
```

The constraints in this MXML tag are named left, top, right, and bottom. Each constraint gives the position, in pixels, from that edge of the container component. The tag above says DataGrid should always be 10 pixels from the left edge, 50 pixels from the top edge, 10 pixels from the right edge, and 10 pixels from the bottom edge.

Centering components uses another constraint:

```
<mx:Text text="My Title Goes Here" horizontalCenter="0" />
```

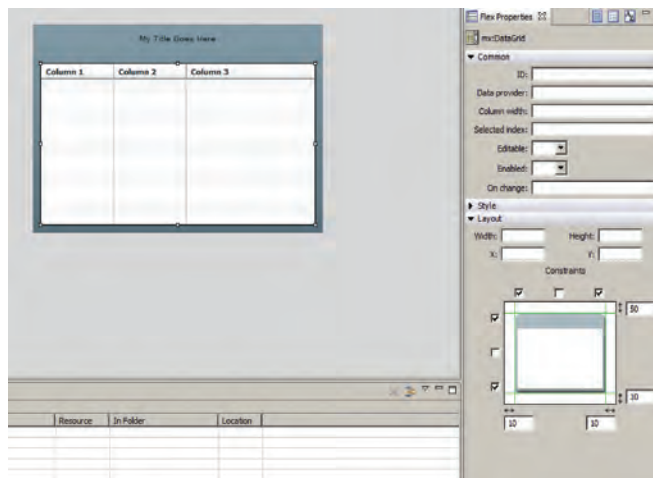


The horizontalCenter constraint is the number of pixels that the center of the component should be from the center of the container. That is, when set to 0, the center of the Text component is exactly at the center of the application. If you set horizontalCenter="-20" then the center of the Text component would be shifted left of center by 20 pixels. A positive number shifts the center of the component to the right. There's a corresponding verticalCenter constraint too.

Note: You don't have to use all the constraints with every control. Use only the ones that make sense. For example, to make a component hug the right edge, use right="0" but don't use left. If you do use both left and right the component will be stretched across the application.

Flex Builder

If you have Flex Builder 2 you can use the Design View function to easily position components.

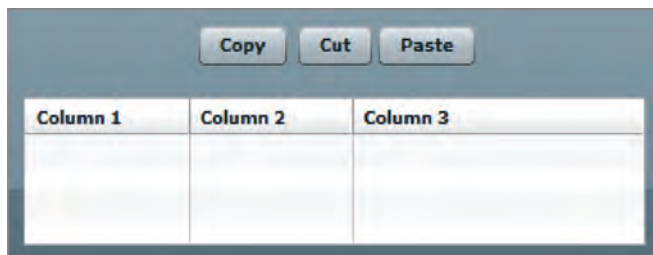


The layout tool (lower right) allows you to give exact values for size, position, and constraint.

Containers

In the examples I've been using so far, there's been one essential container component all along: the Flex Application component. This is the main component. Positioning (x, y, width, height) and constraints (left, right, top, bottom, verticalCenter, horizontalCenter) are used to put components in the application.

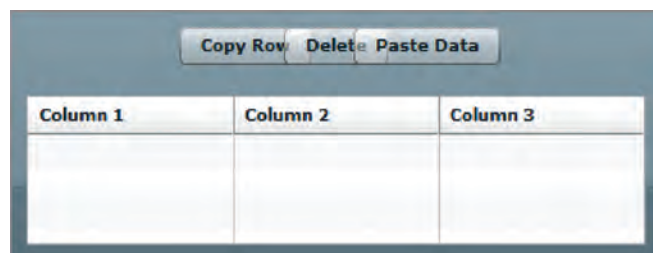
Suppose you want to have a more complex layout like this:



Constraints can be used to keep the buttons together as the application's width changes:

```
<mx:Button label="Left" top="10" horizontalCenter="-56" />
<mx:Button label="Center" top="10" horizontalCenter="0" />
<mx:Button label="Right" top="10" horizontalCenter="56" />
```

You have to experiment with the horizontalCenter values; the values depend on the size of the font used for the labels and the text of the labels. Should any of that change you'll need to recalculate the horizontal positions. For example, simply changing



Learn How to Build the Next Generation of Web Apps **from the Experts!**

August 14, 2006

The Roosevelt Hotel
New York City

Adobe Flex 2 is a complete, powerful application development solution for creating and delivering cross-platform rich Internet applications (RIAs), within the enterprise and across the web. Not until now has there been a way for enterprise programmers and architects to work with existing tools of choice, familiar programming models and integration with existing systems and infrastructure. Multi-step processes, client-side processing, direct manipulation and data visualization are all key factors in the Flex solution.

The "Real-World Flex" One-Day Seminar will delve deep into the central workings of Flex so that the seminar delegates can integrate this timely new technology into their applications, creating powerful interactive content. Delegates will learn from the experts who not only created the product but also those who are using it to the massive benefit of their employers, their customers and the Web.

"Go Beyond AJAX with Flex."

The list of topics at the Real-World Flex Seminar includes:

- ✓ *The Flex Approach to RIA Development*
- ✓ *Bridging Flex and AJAX*
- ✓ *Integrating The SPRY Framework*
- ✓ *Using Flex Builder 2*
- ✓ *Flex for Java Developers*
- ✓ *ActionScript 3.0 Tips and Tricks*
- ✓ *How To Use ColdFusion with Flex*
- ✓ *Leveraging Flash*
- ✓ *Preparing for Adobe Apollo*
- ✓ *MXML Master Class*

Who Should Attend?

- ✓ *CEOs and CTOs*
- ✓ *Senior Architects*
- ✓ *Project Managers*
- ✓ *Web programmers*
- ✓ *Web designers*
- ✓ *Technology Evangelists*
- ✓ *User Interface Architects*
- ✓ *Consultants*
- ✓ *Anyone looking to stay in front of the latest Web technology!*

Early Bird:

(Hurry for Early Bird Pricing) **\$395***

Conference Price:

(Register Onsite) **\$495***

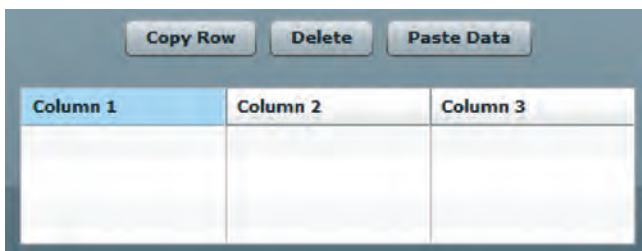
OFFER SUBJECT TO CHANGE WITHOUT NOTICE, PLEASE SEE WEBSITE FOR UP-TO-DATE PRICING

* Golden Pass access includes Breakfast, Lunch and Coffee Breaks, Conference T-Shirt, Collectible Lap-Top Bag and Complete On-Demand Archives of sessions in 7 DVDs!

the labels causes the Buttons to overlap:

An easier way to keep the Buttons together is to use another container. In this case the HBox is the perfect choice. The HBox container defines a rectangular area and automatically positions its children horizontally.

```
<mx:HBox top="10" horizontalCenter="0">
    <mx:Button label="Left" />
    <mx:Button label="Center" />
    <mx:Button label="Right" />
</mx:HBox>
```



Notice that it's the HBox that's now positioned in the application and kept 10 pixels from the top and always centered.

The HBox children, the Buttons, don't need any positioning or constraints because the HBox will automatically put them side-by-side. When the application is made wider, the HBox is moved to the center; then it moves its Button children with it. Now you can change the font and the labels on the Buttons without having to worry about any recalculations.

As you might expect, there's also a VBox container for when you want to align components vertically.

Note: The HBox and VBox containers ignore positioning and constraints because these containers impose their own positioning rules.

Another useful container is the Canvas. This container requires you to place its children using positioning or constraints. If you don't supply any positioning or constraints, the Canvas will put its children at x=0, y=0 and they'll be on top of each other.

```
<mx:Canvas top="10" left="10" bottom="10" width="200">
    <mx:Button top="10" label="top" />
    <mx:Button verticalCenter="0" label="middle" />
```

```
<mx:Button bottom="10" label="bottom" />
</mx:Canvas>
```

The Canvas above is constrained to always be 10 pixels from the top, bottom, and left sides of its parent container (perhaps the application) and have a width of 200 pixels. The children of the Canvas are themselves constrained in it.

Note: You can nest containers in containers. That's what you're doing when you put an HBox in an application. You can then put a VBox in that HBox if it suits you. As your application design becomes more complex, you'll have to do this, but try not to make the nesting too complex. Use constraints whenever possible and avoid nesting containers when possible.

Layout Options

The application container is actually more versatile than it appears. If you look at a Flex program you'll most likely see this:

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mx" layout="absolute">
```

The layout setting of "absolute" means that you must treat the components using positioning or constraints just like a Canvas. But you can change the layout type to either "horizontal" or "vertical." When using either of these values, the application container will act like an HBox or VBox, respectively, and ignore your positioning or constraint settings. The ability of the application container to do this lets you reduce the number of nested containers. For example, if you want to put three Canvas containers next to each other across the application you could do it with constraints but it would be awkward because you'd have to experiment again and again to get the left and right values correct.

Setting the application's layout to "horizontal" is easier:

```
<mx:Application xmlns:mx="http://www.adobe.com/2006/mx"
layout="horizontal">
    <mx:Canvas id="left" width="33%" height="100%" />
    <mx:Canvas id="center" width="34%" height="100%" />
    <mx:Canvas id="right" width="33%" height="100%" />
</mx:Application>
```

Other Useful Containers

Panel: This container has a space for a title at the top and sur-

"Creating your Flex application requires that you plan your space using containers; from as little as the application container to nested containers of containers, you build up your application to look correct and respond to changes in the size of the application"

rounds its children with a frame. As with Application, you can set the Panel's layout to absolute, horizontal, or vertical.

TitleWindow: This container is just like Panel and is used for dialog boxes.

HDividedBox and **VDividedBox:** These containers arrange their children horizontally and vertically, but provide a divider between each child that the user can move to make one component larger or smaller than the others.

Tile: This container places its children one after the other until it reaches an edge then starts a new row or column. You can specify which direction the tiles go.

Form: This container aligns its children with labels. These MXML tags describe a form:

```
<mx:Form x="10" y="10">
    <mx:FormItem label="User Name:">
        <mx:TextInput/>
    </mx:FormItem>
    <mx:FormItem label="Password:">
        <mx:TextInput displayAsPassword="true"/>
    </mx:FormItem>
    <mx:FormItem>
        <mx:Button label="Login"/>
    </mx:FormItem>
</mx:Form>
```

Which appears as:

Summary

Creating your Flex application requires that you plan your space using containers. From as little as the application container to nested containers of containers, you build up your application to look correct and respond to changes in the size of the application.

Judicious use of containers can make your Flex application not only look great, but perform great as well.



About the Author

Peter Ent is a Flex product support engineer at Adobe Systems. He has more than 20 years of experience working with interface design in education, science, and financial services.

pent@adobe.com

Reach Over 100,000 Enterprise Development Managers & Decision Makers with...



Offering leading software, services, and hardware vendors an opportunity to speak to over 100,000 purchasing decision makers about their products, the enterprise IT marketplace, and emerging trends critical to developers, programmers, and IT management

Don't Miss Your Opportunity
to Be a Part of the Next Issue!

Get Listed as a Top 20* Solutions Provider

For Advertising Details
Call 201 802-3021 Today!

*ONLY 20 ADVERTISERS WILL BE DISPLAYED. FIRST COME FIRST SERVE



The World's Leading i-Technology Publisher

Flex 2.0 View States & Transitions

How programming got to be fun again

By Walter Ferguson

Many ColdFusion developers expressed this sentiment after downloading and working with the Flex 2 beta. Flex 2 has made developing rich Internet applications in the Adobe Flash environment possible for everyone (not just those of us who know and love timelines and tweens).

If you've had the opportunity to experience rich Internet applications developed with Flex 2, you likely need little convincing about its benefits from a user's point-of-view. The concept of rich Internet applications (RIAs) is transforming the way we think about using and developing Web-based applications. However, as developers, many of us are just getting introduced to this concept of rich Internet applications and what they offer users of our applications.

Over the past several months a number of technologies have emerged as candidates for developers to add rich interfaces to

browser-based Web applications. Among these technologies, Adobe Flex 2.0 has surfaced as one of the most powerful tools for developers looking to build rich Internet applications. With Flex, Adobe has provided developers with the tools needed to deliver on the promises of RIAs. One of those tools involves the newly introduced concepts of view states and transitions. Let's take a closer look at just what a view state is and how we can use it along with transitions to enhance our applications.

View States

View states are what let Flex developers declare multiple states or views for an application or component. In fact, the Flex documentation defines a view state as simply a particular view of a component. This view can be defined in either the visual or functional capabilities of the component. Every component, or application for that matter, has a default or base state in which we define how the component looks and behaves when it's initialized. From that base state, we can define additional view states by specifying how the component or application changes from its base state, or any other defined state, to the newly created state.

This capability lets our applications have separate states or views based on any number of criteria that we may define. For instance, when developing a classic Web application, we have all

had to address requirements to display data differently for different users perhaps depending on their login status and role, the application state, or other external factors such as a deadline passing.

In a classic Web-based application we could approach this problem in a number of different ways using everything from inline logic to control the display of specific application components to creating entirely separate pages to be included based on the state of the application. View states in Flex let us get this functionality without having to load multiple Flash applications or produce spaghetti logic to control how our application is viewed by a given user or when the application is in a particular state.

In Flex, view states can be created and managed with either MXML tags or ActionScript code. For the sake of this discussion, we'll focus on how MXML tags let us define and use view states. Using MXML, we'd use the `<mx:states>` and `<mx:state>` child tags to define named states in our component or application.

You create a new named view state by altering the base state or another view state of a component or application by adding or removing child components, setting application and component properties, styles, and event listeners. Flex builder makes this very convenient since we can create and manage our view states in the design view using the States tab. The code sample shown in Figure 1 provides a basic example of a search interface. An "Advanced Search" view state has been defined to give the user additional search parameters.

In this example code you can see that the "Advanced Search" state is being defined by listing the changes that have to be made to the base state of the application. This code uses the `<mx:State>` tag to make use of the `mx.states` class. This class lets us define our state by giving it a name using the `name` property, specifying the view by which our newly created state is based using the `basedOn` property.

As is the case in our example, if the `basedOn` property isn't set, the view will be based on the base state or default view of the component or application. The MXML `<mx:AddChild>` tag is also used to add and position additional components in our "Advanced Search" view states. It also uses the `<mx:SetEventHandler>` tag to define event handlers that are active only when their parent's view state is active.

Transitions

Almost anytime you read or hear about view states in Flex 2, transitions can't be far behind. While view states, functionally speaking, are a powerful new feature in Flex 2, transitions are what make them a beautiful new feature.

By default, when we change view states in a Flex 2 application, Flex makes all of the changes at the same time. Visually, it appears like any changes that we've defined in our view state; adding, removing, or resizing components all appear to change instantly as our application jumps from one state to the next.

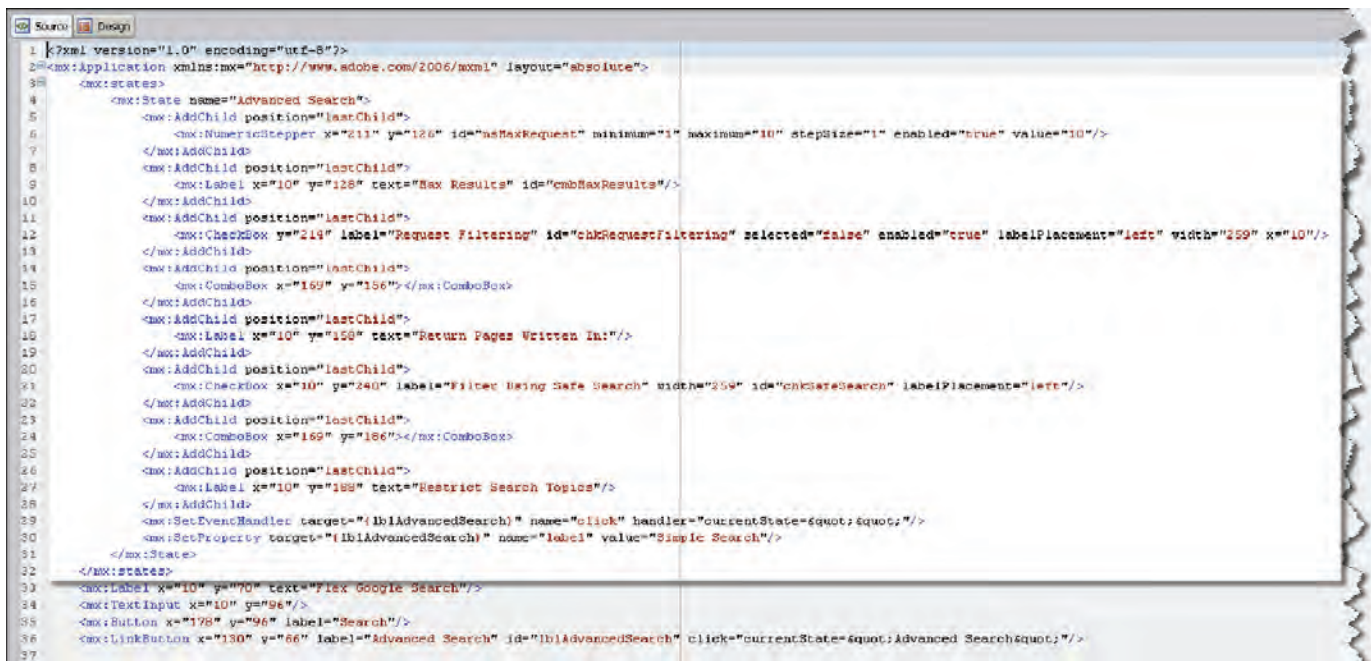


Figure 1:

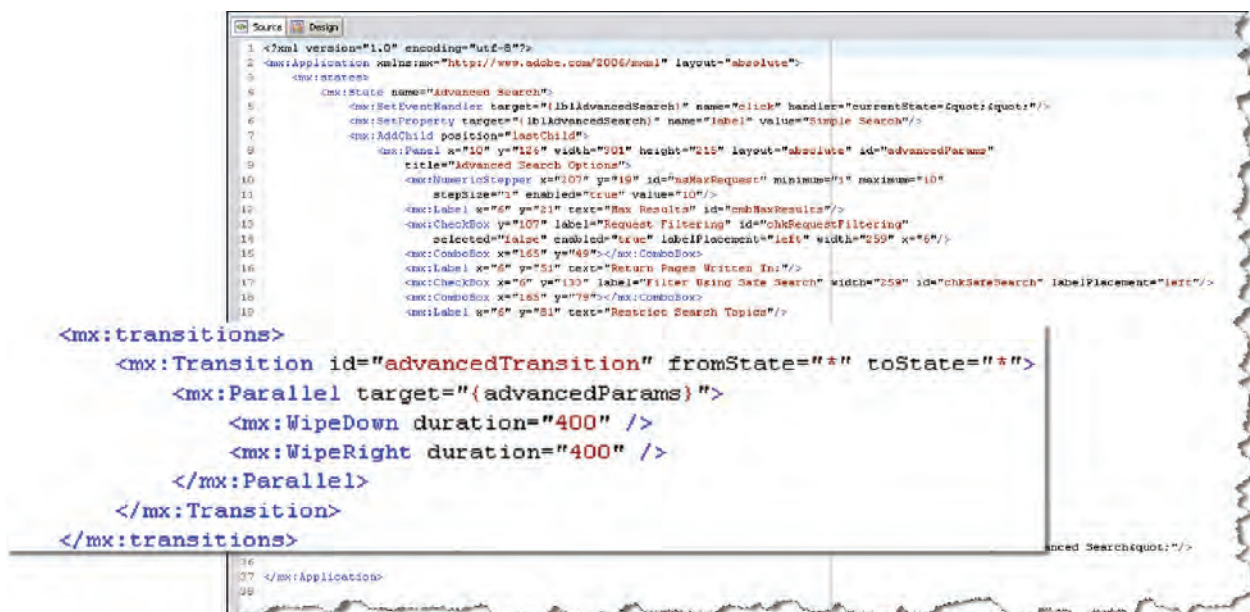


Figure 2:

From a usability standpoint, it's much more effective for users to see the changes take place and transitions let us do just that. Using transitions lets us define a smooth visual change as our application changes from one state to the next.

As we have seen with many other powerful features, Flex 2 makes it very easy to implement transitions when switching from one view state to the next. We can do this by using the Transition class to create a transition and then specifying the fromState, toState, and effect properties of the Transition class. The fromState and toState properties simply tell Flex when the transition should be applied and the effect property defines an effect object that tells Flex which effects to play and how they should be played.

The code sample in Figure 2 defines a parallel transition to be applied whenever we change states in an application. Specifying a value of * in the fromState and toState tells Flex to run this transition anytime we change states.

History Management

Another powerful feature of Flex 2 that can be used when defining view states is the Flex HistoryManager class. Many developers have experienced the pain of providing history management by way of the browser's back and forward navigation commands when using alternate RIA development technologies such as AJAX.

Adobe Flex 2 provides access to the Flex HistoryManager class. Registering our application with the HistoryManager class can let our users use their browsers' back and forward navigation commands like they do with traditional browser-based Web applications. Flex does this by letting developers track and save the state of an application each time we change to a new view state. No need for any complicated ActionScript or JavaScript.

With Flex, we can enable history management in our applications via the IHistoryManagerClient interface. Our

application can implement this interface by implementing loadState() and saveState() methods. After registering the application with the HistoryManager class, we can call the HistoryManager.save() method every time the application state changes to save the state of the application. The code sample below shows how this can be done. It even remembers to run our transition.

We've seen how view states address a very common requirement when building Web applications by letting developers display data differently to users based on any number of other factors. View states in Flex allow us to do this in an elegant manner that's simple to maintain. Transitions, when coupled with view states, can enhance our applications by using effects to provide a visual indication to our users that the state of our application has changed. Additionally we can use the Flex HistoryManager class to allow our users to use the browsers back and forward navigation commands to navigate our applications. Flex 2 has provided developers with powerful options to deliver next generation Web applications.

Yes! Programming is fun again!

References

- Adobe Systems Inc. Flex 2.0 Developers Guide. 2006.
- Adobe Systems Inc. Adobe Flex 2.0 ActionScript and MXML Language Reference. 2006.
- <http://labs.adobe.com> 

About the Author

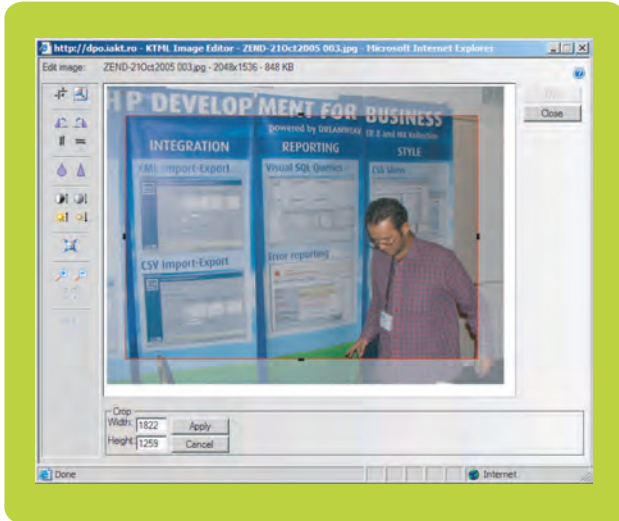
Walter Ferguson is a Certified Advanced ColdFusion MX developer

walter@fergusonfam.com

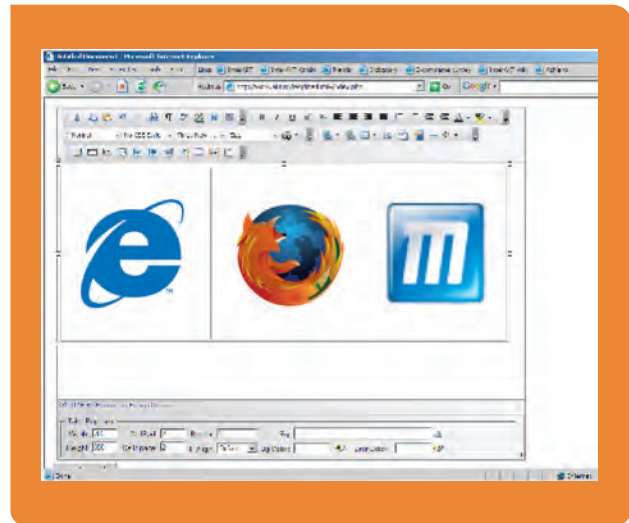
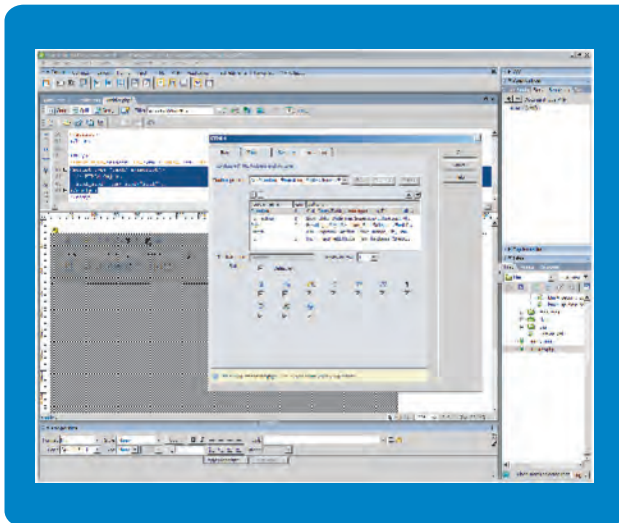
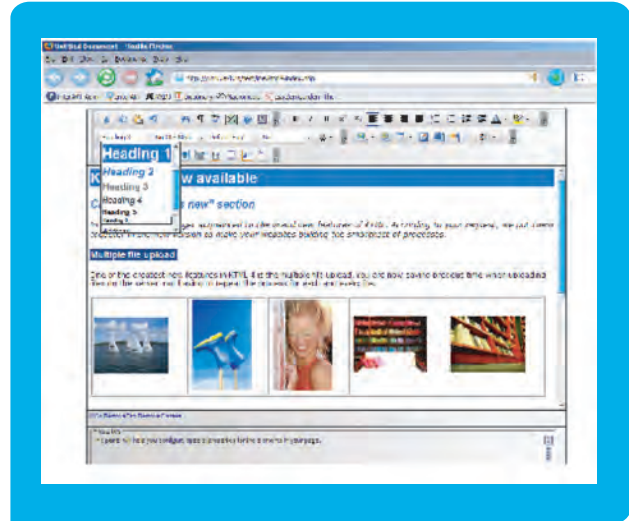
KTML4 Word editing in browser

Along with the most affordable **UNLIMITED** licence

Revolutionary Image Editor



Word-like visual CSS styles



Fast Dreamweaver integration

- Instant paste from Word
- Incredible speed
- Easy to use Word-like toolbars
- Improved CSS authoring
- Remote File Explorer
- XHTML 1.1 compliant

Wide browser compatibility

- Multiple file upload at once
- HTML Table Editor
- Support for multimedia (Flash, Avi)
- Documents management (.doc, .pdf)
- Page templates
- WAI compliant

Please visit www.interaktonline.com/ktml4/ for details

work smart

Interakt

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

