SYS-CON

WALL STREET - FLEX YOUR MUSCLES IN REAL TIME

FORMERLY MX DEVELOPER'S JOURNAL

Web Developer's & Designer's Journal

WWW.WEBDDJ.COM

VOLUME 4 ISSUE 8 2006

BUILDING ENGAGING APPLICATIONS AND CONTENT WITH ADOBE TECHNOLOGIES

Rich Internet Applications

Getting trained with Total Training's Flex 2

AJAXWorld Conference & Expo October 3 ADOBE FLEX TRACK

Plus: 10 Glorious Years of Flash

Presorted
Standard
US Postage
PAID
St. Croix Press

Michael Uman and Luis Blanco,
INTERspectacular

Adobe® Creative Suite 2

Say hello to the next generation.

It's found in the world's most inspirational places. It's pushing the boundaries of art direction and design. Adobe® Creative Suite 2. Today's most talented designers and art directors are working smarter and faster because of it. Just ask the masterminds behind INTERspectacular—they rely on Adobe CS2 to make their ideas tangible reality. See what Adobe Creative Suite 2 can do for you. It's everything but the idea. Better by Adobe.™

adobe.com/go/designer

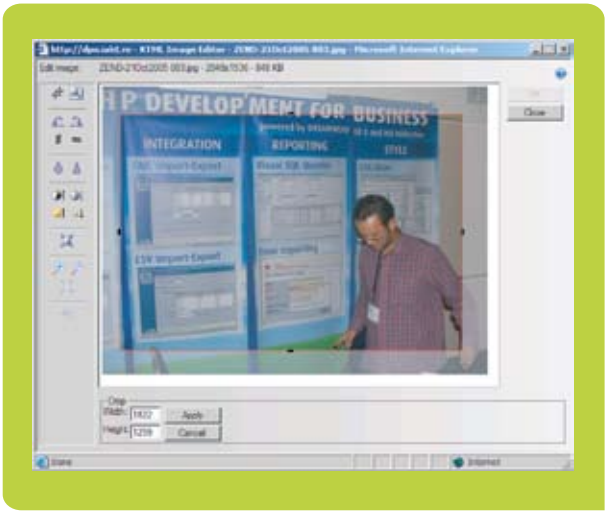
Adobe®

Web Developer's & Designer's Journal
August 2006

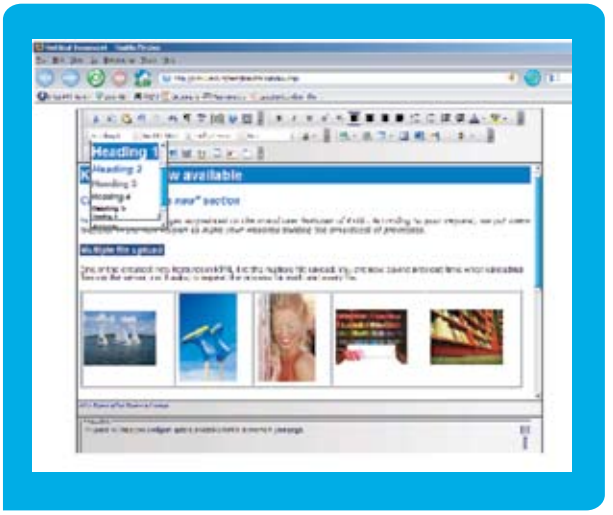
KTML4 Word editing in browser

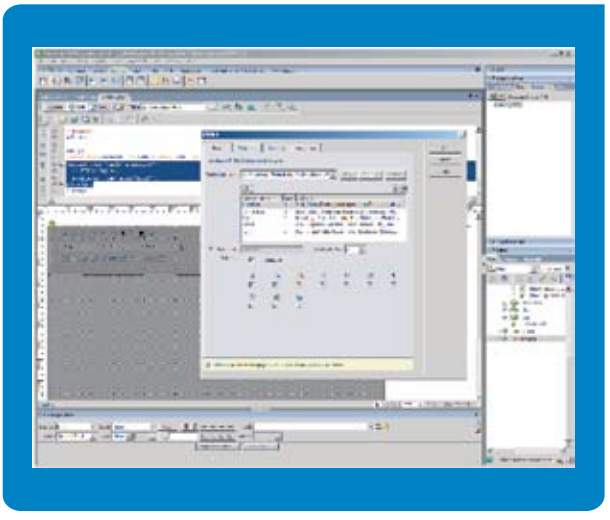
Along with the most affordable **UNLIMITED** licence

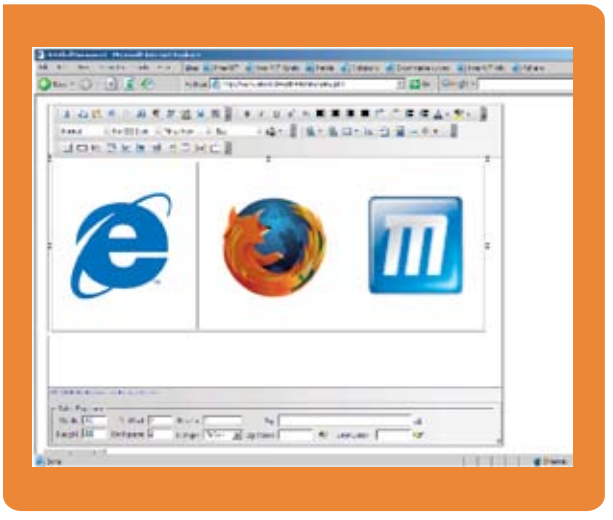
Revolutionary Image Editor



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

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 suppport 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 suppport sets us apart from other hosts.
Real service. Real satisfaction. Real value. Real support. Real Freedom.

macromedia ALLIANCE PARTNER

BlueDragon Certified V.I.P. Hosting Partner

Web Developer's & Designer's Journal

Group Publisher Roger Strukhoff
Art Director Louis F. Cuffari

Editorial Board
Dreamweaver Editor
Dave McFarland

Flash Editor
Brian Eubanks

Fireworks Editor
Joyce J. Evans

FreeHand Editor
Louis F. Cuffari

Director Editor
Andrew Phelps

Captivate Editor
Tom Green

International Advisory Board
Norway
Jens Christian Brynildsen

UK
David Hurrows

USA
Joshua Davis, Jon Gay, Craig Goodman, Phillip Kerman, Danny Mavromatis, Jesse Nieminen, Gary Rosenzweig, John Tidwell

Canada
Colin Moock

Editorial
Editor
Nancy Valentine, 201 802-3044
nancy@sys-con.com

Technical Editors
Jesse Warden • Sarge Sargent

To submit a proposal for an article, go to http://grids.sys-con.com/proposal.

Subscriptions
E-mail: subscribe@sys-con.com
U.S. Toll Free: 888 303-5282
International: 201 802-3012
Fax: 201 782-9600
Cover Price U.S. $5.99
U.S. $29.99 (12 issues/1 year)
Canada/Mexico: $49.99/year
International: $59.99/year
Credit Card, U.S. Banks or Money Orders
Back Issues: $12/each

Editorial and Advertising Offices
Postmaster: Send all address changes to:
SYS-CON Media
577 Chestnut Ridge Rd.
Woodcliff Lake, NJ 07677

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ

List Rental Information
Kevin Collopy: 845 731-2684,
kevin.collopy@edithroman.com,
Frank Cipolla: 845 731-3832,
frank.cipolla@epostdirect.com

Promotional Reprints
Megan Mussa, 201 802-3024
megan@sys-con.com

Web Developer's & Designer's Journal (ISSN#1546-2242) is published monthly (12 times a year) by SYS-CON Publications, Inc., 577 Chestnut Ridge Road, Woodcliff Lake, NJ 07677.

Hey, Wall Street, Flex Your Muscles!

by Yakov Fain

On August 14, I attended the seminar RealWorld Flex in New York City. I was really impressed by the endless number of presentations showing the use of Adobe Flex 2 technology in the real world applications. Real-time processing, vector graphics, collaboration, messaging, multimedia, shopping carts, geo maps, customer support, and more. This was a really interesting event about the technology that will become a hit of 2007.

Each of the presenters was emphasizing that it does not take a rocket scientist to create these applications. This reminded me of the process of adoption of Java at the end of the nineties based on a similar premise that an average Java programmer could do things that only gurus could have done in C++. And now a similar story: an average Flex 2 developer will be able to do more that a Java Swing guru.

I was co-presenting with Victor Rasputnis on using Flex with Java (you can download this presentation at Farata Systems Web site), and our message was twofold: first we've shown how Flex on the front can help and work with Java on the back, and then we've shown a couple of our open source components where Java was used to automatically generate Flex code and a Flex business intelligent tool.

As of today, Java is the most adopted language in the world, and it'll stay this way on the server side and in the mobile devices arena for years. Java Swing is the most comprehensive but the most expensive tool available for developing the front-end for Rich Internet Applications. You can do everything with Swing… if you can afford it. While the software is free, the Swing professionals charge premiums for their working hours. And they'll need lots of hours to develop, debug and test these applications.

Regarding Wall Street… Adobe Flex 2 can work in a real time, has its own messaging and supports JMS, so it's about time to re-think the design of your trading applications. I can almost hear now the voices of the die-hard stock traders looking all day long at the black screen with constantly changing large numbers – price quotes. Some vendors came out with special keyboards/devices that allow making a trade with two successive hits of the finger and the bottom of their palms. They say, "We did it yesterday, and we'll be doing it tomorrow". They just did not know any better.

But Wall Street is seriously considering Flex 2. I know two Fortune 100 brokerage houses that purchased Flex 2 licenses while this software was still in Beta. If you are not impressed, this tells me that you did not work for these giants that may start considering any new software in a year after its official release.

Flex Data Services can be used as a gateway to your existing, powerful and scalable Java EE applications. Just replace the end-user facing part of your Web applications. Flex (as opposed to AJAX) supports the server push using the publish-subscribe mechanism. It supports binary sockets as well. Move the MVC pattern implementation to the client where it belongs in the RIA world. You had to maintain the session in the Web applications on the server only because HTTP is a stateless protocol, but now your compiled code runs in the Flash virtual machine on the client (by the way, raise your hand if your Java Swing Web application keeps the state on the server? Boo!).

Any pragmatic corporate development manager can say, "I like the way Flex applications look, but who else is doing this? Are there Flex developers available? Are there Flex components available? We have all this in Java". The answer is yes, there are people now and there will be plenty of those in 2007.

Still unsure how to get familiar with this technology before making this decision? Sign up for my evening Flex 2 class at New York University (it's five sessions over five weeks). It's a hands-on class where we'll start from the basics and will move toward the creation of business applications.

You know better than me that competition is real tough in the financial IT sector. This technology will give you an edge over your competition, and most importantly, will make your business users happy.

Yakov Fain is a senior IT architect consulting Wall Street companies. He's authored several Java books, dozens of technical articles and his blog is hugely popular. SYS-CON Books will be releasing his latest book, "Rich Internet Applications with Adobe Flex and Java: Secrets of the Masters" this Fall. Sun Microsystems has nominated and awarded Yakov with the title Java Champion. He leads the Princeton Java Users Group. Yakov teaches Java and Flex 2 at New York University.
yfain@sys-con.com

Innovative Tools

To Manage, Deliver and Track Streaming Media On the Internet

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

VitalStream®

The World Leader In Audio and Video Streaming

10 Glorious Years of Flash

According to Mike Downey – and why would anyone doubt the word of the Senior Product Manager for Flash in the entire Adobe empire – a lot of people in the post-merger world of Adobe "didn't realize" that mid-August marked the ten-year milestone of the second jewel in Adobe's crown after PDF…Flash.

The Early Days

"FutureSplash Animator 1.0, as it was called at the time," explains Downey, "was released in August 1996. So August 2006 was a major milestone for Flash."

I ask Downey for more detail on how FutureSplash Animator was deployed 'back in the day' and he explains that the technology was used on several big-name websites such as Microsoft's MSN, Fox TV's official site for The Simpsons, and Disney's own Disney Daily Blast.

"At this point, the company behind FutureSplash – called Future Wave Software – consisted of just six people," Downey says.

It was at this point, he adds, that Macromedia enters the picture. Disney, it turned out, also used a player from Macromedia called Shockwave, and once it learned aboiut FutureSplash from Disney, Macromedia approached FutureSplash about maybe working together – an approach which led to Macromedia acquiring FutureWave. That was in December of 1996, so things were already moving fast.

FutureSplash Animator became Macromedia Flash 1.0 and almost overnight became a popular method for adding animation and interactivity to web pages.

Downey notes the strange coincidence that Flash should have its 10th birthday just as the World Wide Web is having its 15th. But many developers I know will actually be interested in going even further back in time, so before we get on to the 10 Glorious Years of Flash that this feature celebrates, you might want to glance at the sidebar labeled "From Tiny Acorns…"

From Tiny Acorns

Before it even became FutureSplash Animator, what we now know as Flash had a pre-history. Two engineers, Jonathan Gay and Robert Tatsumi, started exploring new devices – the precursors of the Tablet computer – and felt that people ought to be able to draw on a computer, so they came up with a sketching program that they called SmartSketch, a vector-based drawing program for pen-based computers.

Their company, Future Wave Software, started demo'ing SmartSketch at trade shows like SiGraph. And it was from the feedback at such show that the idea came that what would be really great would be if you could combine this sketching functionality with some kind of timeline.

By chance this epiphany came at exactly the same time as the debut of Netscape's plug-in architecture. The resulting product was released in May of 1996. Its name: FutureSplash Animator.

The Middle Years

From December 1996, when Flash took on the name its had ever since, the key metric that Macromedia began to focus on was market penetration: if the Flash runtime could only be widely

June 1997

Flash 2 is released with features such as support of stereo sound and enhanced bitmap integration. Released with Flash Player 2.

May 1998

Flash 3 is released with Flash Player 3, new features include the movieclip element, JavaScript plug-in integration, transparency and an external standalone player.

June 1999

Flash4 is released with Flash Player 4, new features included: internal variables, an input field, advanced ActionScript, and streaming MP3.

August 2000

Flash 5 is released with Flash Player 5 and ActionScript 1.0 (based on ECMAScript, making it very similar to JavaScript in syntax), XML support, Smartclips (the precursor to components in Flash), HTML text formatting added for dynamic text.

•

March 2002

Macromedia announces the availability of Macromedia Flash MX and Macromedia Flash Player 6, with support for video, application components, and accessibility.

September 2003

Flash MX 2004 is released, with features such as faster runtime performance up to 8 times with the enhanced compiler and the new Macromedia Flash Player 7, ability to create charts, graphs and additional text effects with the new support for extensions (sold separately), high fidelity import of PDF and Adobe Illustrator 10 files, mobile and device development and a forms-based development environment.

December 3, 2005

Adobe Systems acquires Macromedia and its product portfolio (including Flash)

---

propagated, the belief was that revenue streams would follow.

This strategy took a tactical Big Leap Forward when Macromedia approached Netscape with the idea of paying them to distribute Flash – a hefty investment that, Downey says, paid off. Netscape's requirement was that the Flash player was kept by Macromedia's engineers to under 150KB in size, and to this day the Flash Player team working on any release always starts off with a "code budget" – and in fact it has only increased by 1MB in 10 years.

For a snapshot of how Flash changed through its middle years, see Sidebar B.

The Flash Future

The future of Flash is inextricably bound up with the future of Flash video. Google Video and MyTube ought to be proof of concept enough for anyone: once ordinary Joes can put video on their web sites, the sky for Flash video is the limit.

"There is is always room for a better quality codecs of course," says Downey, who adds that v 8.0 is even better at video-on-demand. Internally, he reveals, Flash video is right up there among the Top Five priorities of the entire company.

Anyone lucky enough to have been at the inaugural "Real-World Flex" seminar produced by SYS-CON Events in mid-August in New York City will have heard

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.

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

EDGE WEB HOSTING

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

http://edgewebhosting.net

CFDJ READERS' CHOICE AWARD · MXDJ READERS' CHOICE AWARD

2003 - 2006

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

Flash Player version	File size	Size increase	Size gain
2	192 KB	-	-
3	216 KB	24 KB	12.5%
4	269 KB	53 KB	24.5%
5	311 KB	42 KB	15.6%
6	512 KB	201 KB	64.6%
7	683 KB	171 KB	33.4%
8	926 KB	243 KB	35.6%

Flash 2 was only 192KB in size, and by Flash 8 was still under 1MB. The table below shows the file size of Flash Player releases, from version 2 to the present.

[Datasource: Rob Reinhardt, CommunityMX]

Adobe's Luis Polanco give all manner of insights into "Apollo," Adobe's next-generation runtime. Here, too, Flash video will play a key role: Polanco promised that Apollo will have "unique video capabilities" and in the October issue of WebDDJ – which will be available at this year's MAX in Las Vegas – he will be writing in more detail about them.

Polanco also committed Adobe to a public beta of Apollo – which it will release via the Adobe Labs site – before the end of the year.

The Team Behind Flash: Bios & Sidelights

Kevin Lynch

- SVP and Chief Software Architect, Platform Business Unit
- Named one of CRN's "Top 25 Innovators" in 2003 and honored as one of the "First Annual Web Innovators" by CNET in 1998.
- Joined Adobe through the company's 2005 acquisition of Macromedia, Inc., where he served as chief software architect and president of product development.
- Responsible for the company's ubiquitous Portable Document Format (PDF), Adobe® Reader®, and Macromedia® Flash® Player, as well as alignment of Adobe's servers and tools with the company's technology platform.
- Also oversees Adobe's developer relations program.

Mike Chambers

- Senior Product Manager for Developer Relations
- Has spent the last 8 years building applications that target the Flash runtime.
- Has worked during those 8 years with numerous technologies including Flash, Generator, Central, Flex, and Ajax.
- Currently in charge of developer relations for Apollo, a project which aims to bring the power of Rich Internet Applications to the desktop.
- Has co-authored a number of books and articles, spoken at numerous conferences, and maintains a Flash Platform development weblog at http://weblogs.macromedia.com/mesh/.

Mike Downey

- Senior product manager, Flash
- Passionately involved with web technologies since the late 90s.
- Joined Macromedia in 2000
- Experienced instructor, presenter, and developer.
- Has met with thousands of designers, developers, and technologists through numerous live and online events.
- Has also been a featured presenter at conferences such as FlashForward, FiTC, Macworld, Spark, WebDU, Oracle OpenWorld, JavaOne, Macromedia MAX and several others.
- Maintains a weblog devoted to Flash and other Internet technologies at http://weblogs.macromedia.com/md.

Robert Tatsumi

- One of the original Flash engineers – worked with Jon Gay
- Co-Creator (with Jonathan Gay) of SmartSketch (later FutureSplash Animator) the precursor of Flash.

Slavik Lozben

- Flash engineer #3
- Principal engineer on the Flash Media Server team.
- Involved in the design and implementation of many of the features in FMS.
- Joined Macromedia in January of 1995.
- Originally worked on the Director product line, but when Macromedia bought what is now known as Flash, saw a tremendous opportunity and joined Robert Tatsumi and Jon Gay, the creators of Flash.
- Has been on the Flash team ever since.

Justin Everett-church

- Sr. Product Manager for Flash Designer/Developer relations
- Co-author of four books on game development in Flash.
- A frequent speaker at conferences on the topics of Flash for mobile and devices and Flash outside the browser.
- Formerly Rich Media Evangelist for Yahoo!
- First saw FutureSplash on MSN as a spinning globe animation in 1996.

Ext.	Explanation
.avi	AVI file is a video file, standing for Audio Video Interleave. Flash includes some compression codecs, including some from Radius.
.gif	GIF image, can be a single static frame or animated.
.png	.png files are Portable Network Graphics
.jpg	.jpg files are [JPEG] Joint Photographic Expert Group, a popular lossy 24bit image compression format.
.ssk	.ssk files are SmartSketch drawings.
.3ds	Some third-party software converts animated 3D file formats, notably the common 3D Studio Max format, to SWF files as rendered vector animations.

Generic file formats used by Flash
[Datasource: Wikipedia]

Voices from the Community

Blogosphere

by WebDDJ News Desk

Post "Real-World Flex" One-Day Seminar

by Jesse Randall Warden
http://www.jessewarden.com

I gotta say, I had a lot of fun at the Real-World Flex Seminar SYS-CON put together. Things went, for me, really smoothly. Registration, getting Internet at the booth, and hooking up with my team. Free coffee was readily available, so I was set. It's always great because at every conference, I always get to put faces to people I've known online & collaborated with for years. I also get to meet new people and hopefully leak some of my passion to them and hopefully inspire them to want to create cool apps with Flex. I also got to rant via an interview; any excuse to talk about technology is fun.

Unfortunately, I had no idea I was in Manhattan. Things move so fast lately I just assumed I was going to some hotel in New York. I had no clue I was like... in THE Manhattan. What an idiot.

To spare the details, I basically had to go through 3 laptops in 15 minutes to get my presentation started. Since the laptop didn't have the apps I needed on it setup, I just breezed through an abbreviated version of my slides in 30 minutes. Thanks Jeremy Geelan & Dave Wolfe for the laptops. My test run with the Atlanta Adobe User's Group went to about an hour and a half, so naturally I talked at Mach-2, and skipped some of the lower level Flash SWF integration slides. I'll upload my full PowerPoint later this week for those who didn't get to the see the main six I removed.

Really great answering questions, really great meeting new people, and cool to talk albeit briefly about what people are working on; it's always fun. Thanks a lot SYS-CON, it was a great seminar! Hopefully your audience is inspired to at least investigate what it would take to get an information architect/designer on some of their existing software engineering teams if they don't already have one.

I'm working on getting the design rights to the design I'm using to build my real-time YouTube clone, so hopefully I can not only share the source this October, but also some of the nitty gritty FLAs and SWFs that I used to integrate such a complicated, traditional Flash design, into Flex.

Thanks to those who came, and again, thanks SYS-CON for the invite to speak, much appreciated.

Flickr Changes Bust Flexamp and other Flash Apps

by Code Zen
http://www.arpitonline.com/blog/

Flickr changed the crossdomain file on their servers a few days back breaking Flash applications across the www. The flickr crossdomain file at www.flickr.com has been removed and Flash developers have been asked to point their applications to api.flickr.com.

Unfortunately I missed the news and so far the last few days, related media has not been working. Hopefully I'll get down to recompiling it soon. I looked around the many Flex apps using Flickr and most haven't updated their apps.

In other news I have also started a new page to point to Flex links I have found on the web. I keep jumping from computer to computer and am not completely in love with del.icio.us (mostly because of the UI …. I wish it looked more like popurls basically one big page of links) so it seemed like a good idea.

Large Applications for Impatient Developers

by Anatole Tartakovsky & Yakov Fain
http://flexblog.faratasystems.com/

Half year ago I made few comments regarding tricks I like to use when developing large Flex 2 applications. Since then I have been receiving steady stream of emails - once every week in the beginning, up to few per days lately. They are asking for information from the upcoming RIA Book we are writing (http://www.riabook.com). Apparently, there are quite a few developers that a) become impatient if the build takes more then 5 seconds and b) are concerned that their application will be delivered to users with a similar attention span who could walk away in 10 seconds or less - regardless of artwork in the progress meter.

That pretty much forces developers to break the applications in the manner similar to the current generation of loosely coupled applications:

- The application has to use RSL methodology to speed up development and minimize linkage time
- The bootstrap application has to be kept to bare minimum - just initialize global managers and list common runtime libraries - to keep "rebuild" time

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

low and initial load fast

- Optimized runtime libraries - while it is tempting to load complete framework SWC with every application, it would amount to extra 500-600KB on the initial download - better managememt via automatic extract to shared RSL is required
- The application should make use of automatic download of "pages" - essentially parts of application that are either optional or independent or can be customized/added due to subscription /security.

Given those requirements, the book talks about static linkage of Flex applications (unlike classical environments that compile everything and then link object modules Flex starts from the "application" and pools necessary resources via sophisticated linker/preprocessor/compiler/optimizer/packager workflow). The book describes the differences between MXML and ActionScript applications, differences between compile-time and run-time linkafe, class loading, self-initialization of dynamically loaded SWFs, and other small details that you only need to know when your application grows over 10-15 screens /1MB of statically linked SWF.

Flex-AJAX Bridge and the Unsupported setAttribute() method By IE

By Marco Casario
http://casario.blogs.com/

I'm testing the Flex AJAX bridge and I spent some time to solve this unsupported tag by Internet Explorer myDiv.setAttribute("onblur", u_onblur).

In Firefox and Mozilla all worked but in order to make it works in IE I had to change the JavaScript code into :

```
i_elemento.onblur = u_onblur;
```

where u_onblur is the second parameter passed as argument to my function :

```
changeTag("username",checkUsername,"username_check","[betweeen 5 and 20
characters]");
```

Here's my function:

```
function changeTag(id_elemento, u_onblur, id_span, testo_span)
{
        var l_elemento = getByID(id_elemento);
        var parentDiv = l_elemento.parentNode;
        i_elemento=document.createElement("input");
        i_elemento.setAttribute("id", id_elemento);
        i_elemento.setAttribute("value",
        Trim(getByID(id_elemento).innerHTML));
        i_elemento.setAttribute("maxlength", "20");
        i_elemento.setAttribute("size", "30");
        i_elemento.setAttribute("name", id_elemento);
        i_elemento.setAttribute("type", "text");
        // it does not work on IE
        //i_password.setAttribute("onblur", u_onblur);
        i_elemento.onblur = u_onblur;
        parentDiv.replaceChild(i_elemento, l_elemento);

        s_elemento=document.createElement("span");
        s_elemento.setAttribute("id", id_span);
        var s_elemento_content = document.createTextNode(testo_span);
        s_elemento.appendChild(s_elemento_content);
        var sp2 = document.getElementById("labelpassword");
        var parentDiv = i_elemento.parentNode;
        parentDiv.insertBefore(s_elemento, sp2);
}
```


Intermedia.NET...
Now serving the hottest ColdFusion.
At Intermedia.NET we go beyond the industry standard by supporting the hottest new Coldfusion software, offering power like never before. For nearly a decade, we've been providing reliable, secure hosting to thousands of companies across the globe. We can do the same for you.
Intermedia.NET's premier hosting services include:
ColdFusion MX hosting
Competitive plans
Security sandboxes
Custom tag registration
Verity collections search engine
Guaranteed service levels
Unprecedented power, unmatched reputation...
Intermedia.NET is your hosting solution.
INTERMEDIA.NET
Call us at: 1.888.379.7729
e-mail us at: sales@intermedia.NET
Visit us at: www.intermedia.NET
© Copyright INTERMEDIA.NET, Inc 2005. All rights reserved. All other trademarks are property of their respective holders.

Using Resource Bundles in Flex

The basics

by Brian Deitte

There comes a day when we all need an application in multiple languages. Eso es la verdad. In Flex, the solution to this problem is resource bundles. In this article, I'll describe the basic use of resource bundles and create a small example in Flex Builder. I'll also suggest some resources for further exploration and ponder some potential future directions of this feature.

The Basics of Resource Bundles

A resource bundle is a simple file, commonly called a properties file, where keys and values are stored. The file format for resource bundles is similar to Java properties files, with a "key=value" on each line. The main difference from the Java properties format is that files with non-ASCII characters in them should be stored as UTF-8.

Properties files are put in directories where they can be searched for by the compiler in the same way that other source files are found. The properties files should be arranged in directories in a specific way that's explained in the example below.

The values in the resource bundles can be accessed in Flex through @Resource in MXML or through ResourceBundle metadata and the ResourceBundle class in ActionScript. There's two pieces of information that are needed to access resource bundles in MXML or ActionScript- a bundle name and a key. The bundle name is the same name as the properties file. So if you create HelloWorldBundle.properties, the bundle name is HelloWorldBundle. The keys are found in the properties file, to the left of the values.

bdeitte@adobe.com

Setting Up the Flex Builder Project

We'll start off the example by creating a new Flex project in Flex Builder. When creating the project, select Basic for data access and name the project whatever you'd like. Click on "Next" to enter more information.

Before entering anything else in Flex Builder, we need to create directories for the properties files. The files should go in their own directories that are outside of any current source path. I've created the main directory for the properties file in the default directory for projects:

```
C:\Documents and Settings\bdeitte\My
Documents\Flex Builder 2\locale
```

We also need to add subdirectories for each language we plan to have resource bundles in. The subdirectory names should match the locale names we plan to use We'll be using en_US for our English strings and sp for Spanish strings. So now I have:

```
C:\Documents and Settings\bdeitte\My
Documents\Flex Builder 2\locale\en_US
```

```
C:\Documents and Settings\bdeitte\My
Documents\Flex Builder 2\locale\sp
```

We'll get back to putting files in these directories later. Now we want to finish setting up our Flex Builder project. To have the project find the resource bundles, we need to add the directories we've just created to the source path. We don't add a source path for both en_US and sp, but rather we use the special "{locale}" signifier. So I add this folder as a source path in Flex Builder:

```
C:\Documents and Settings\bdeitte\
My Documents\Flex Builder 2\locale\
{locale}
```

The Main application file name should be changed to HelloWorld.mxml. Then click Finish.

Creating HelloWorld.mxml

You should now have a HelloWorld.mxml file in front of you in Flex Builder. In Source mode, change HelloWorld.mxml to the following:

```
<?xml version="1.0" encoding="utf-
8"?>

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

  <mx:Label text="@Resource(
key='hello', bundle='HelloWorldBundl
e')"/>

</mx:Application>
```

This is all of the MXML for the Hello World application that we are building. Note the use of @Resource for the Label's text and the key and bundle information. At this point you can try to compile HelloWorld.mxml, but you'll get the following error:

Unable to resolve a class for ResourceBundle: HelloWorldBundle.

A class is mentioned here because the Flex compiler thinks of the properties files as classes, and it can't find a properties files.

Adding Properties Files

Create the file HelloWorldBundle.properties in the en_US directory created earlier. Add the following to the file:

```
hello=Hello World
```

Create HelloWorldBundle.properties in the sp directory, and add the following to the file:

```
hello=Hola Mundo
```

The names of the properties files correspond to the bundle value we specified in @Resource, and the key value corresponds to the left side. We could add more key values as needed, like this:

```
hello=Hola Mundo

one=uno

bye=Adios
```

Building and Running the SWFs

We can then go back to Flex Builder and run HelloWorld. When you run the project, you should see on the screen the most exciting of Flex applications, "Hello World".

To run the project in Spanish, right-click on the project name and bring up the properties for the project. In the additional compiler arguments, change the locale from "en_US" to "sp". The "sp" matches the folder name where we put the second HelloWorldBundle. properties. When you run the project after this, you will see "Hola Mundo".

If you were building these Flex applications for later use, you would have to copy the directory of output files from the English version before building the Spanish version. Alternatively, you could build these files with the command-line compiler when creating the final SWFs. You would also need to build a page in HTML or Flash for choosing between the different languages.

Further Exploration

We only explored using @Resource within MXML and did not use the ResourceBundle metadata in ActionScript. This is an essential part of using resource bundles. Check the "Localizing Flex Applications" section of "Flex 2 Developer's Guide" in the Flex documentation to learn more about this.

We didn't discuss a few of the more advanced features of resource bundles in this article:

- Resource bundles can be used inside of SWCs if you create resource bundle SWCs.
- Complete classes and media such as images can be internationalized by using custom resource bundles.
- All of the framework uses properties files, and by using the framework source you can localize the framework.

More information about these features can also be found in the "Localizing Flex Applications" documentation.

The ResourceBundle API documentation can be used to learn more about resource bundles, but try to ignore the main summary on the page. Significant pieces of the description are currently incorrect.

Future Directions

There are a few issues with resource bundles which will be fixed in the next update to Flex. Most notably, Flex Builder can show an error after updating a properties file. You'll know you are encountering this problem if you see an "Unable to resolve a class for ResourceBundle" error where the class mentioned ends with "_properties". Clean the project in order to remove the error.

We haven't decided on this yet, but Flex Builder could remove the need to copy directories when building localized applications. This could be done through a publish dialogue that allows multiple locales to be chosen.

We haven't decided on this change, but Flex could allow resources to be dynamically retrieved at runtime. For now, Flex only supports the compiling of resource bundles into the SWF. We plan to allow media and classes to be added to properties files. This will mean that custom resource bundles won't be needed for complex resource bundles.

In the far future, Flex may allow different translation formats than properties files, such as the XML Localization Interchange File Format.

If you know what you'd like to see in the future of resource bundles, you can email me at bdeitte@adobe.com. I'll also post updates and future work on resource bundles on http://www.deitte.com.

Putting RSS Content into Your Dreamweaver Web Pages

Sharing is good

by Ronald Northrip

RSS – Really Simple Syndication – is about sharing content with new audiences. Lots of websites create RSS feeds so that their audiences can get updates using their favorite Aggregators (like NetNewsWire). Other websites collect that content and republish it in new venues, to new audiences, or just in a different medium. That's what syndication is all about -- sharing. And since my mamma always taught me sharing is good, I'm gonna share the secret to putting RSS content onto your web page.

Now, you might have an RSS feed already. However you choose to build that RSS feed -- from RSS DreamFeeder to FeedForAll to a custom script -- the content in the feed is intended to let you reach out to a new audience. That audience has to subscribe to the content, so it might be useful to help them see the sorts of headlines they'll be getting. Consider publishing those headlines on a webpage -- it's easy and helps them make the decision to subscribe to your feed.

You might also want to republish content from other websites. There are lots of sources of RSS content out there. Republishing those feeds on web pages that your audience reads regularly can help to keep them better informed. If the goal of an RSS feed is to find new audiences, putting their content in front of your audience helps achieve the aim of the original publisher (though don't forget to let people know where the headlines came from).

This tutorial will show you how to do both of these things with a simple Dreamweaver extension called RSS Replay.

What You'll Need

First you'll need to download and install the RSS Replay extension. You can get the extension from the RNSoft website at http://www.rnsoft.com/en/products/rssreplay/

Second, I've written a few example files that you'll want to use so download them from the RNSoft website as well http://www.rnsoft.com/en/products/rssreplay/content/tutorial/RRPTutorialWDDJ.zip

Once you have downloaded the example files and unzipped them you will have a directory called "RRSTutorialWDDJ" that contains a few files. You'll use that directory as the root directory for a website in Dreamweaver. So the next step is to launch Dreamweaver and setup a website with "RRSTutorialWDDJ" as the local root.

I am assuming that you know and use Dreamweaver -- which might not be a fair assumption. If these instructions are not specific enough, take a look at the tutorial on RNSoft's website for more details at http://www.rnsoft.com/en/products/rssreplay/tutorial/

Replaying a Local Feed

Say you've got this website about Dreamweaver called "Dreamweaver News" and you publish an RSS feed of the active issues in the Dreamweaver community. You want to put your feed ("dwnews.rss") onto your web page ("dwnews.html").

So begin by opening the "dwnews.html" file in Dreamweaver.

Select the text that says "Put Replay Here" and delete it. In its place we will put an RSS Replay. Depending on your version of Dreamweaver the icon for RSS Replay is the Insert panel's Media tab or the Media popup in the Common tab (wherever you would normally find the icon for Flash or ActiveX) (see Figure 1).

Click the icon to insert an RSS Replay. An icon is inserted into your layout indicating that you now have a Replay. With that icon selected you should now see the RSS Replay Property Inspector. We'll use the property inspector to configure the RSS Replay.

The property inspector allows you to specify the sources (the URLs of the feed), the CSS style for displaying the content (headlines and stories), the format for display, and the method of transferring the feed.

The next thing you need to do is specify the source of the content you want to display. The feed is an RSS file called "dwnews.rss". Put that into the list of sources by clicking the "Sources" button to display a dialog containing the list of sources. At first, the list will be empty. Click the plus (+) button to add a new source to the list, then select

it and enter "dwnews.rss". Then click the "OK" button to close the dialog (see Figure 2).

The next step is to specify the format for displaying the content. RSS Replay comes with 10 formats for displaying content (and you can create your own too). You'll just select one from existing formats in the "Format" menu. Right now the menu indicates that there are "No Settings". Select "Headlines w Stories", which will display the headline and the story associated with that headline (see Figure 3).

Now that you've specified that you'll want to see what the page will look like. RSS Replay provides an in-Dreamweaver preview of the content so that you can better assess how the content fits into your page and make adjustments as needed. To play the Replay press the "Play" button in the inspector.

Now your page will change to display the RSS content in the layout and the "Play" button will change to say "Stop" – press this when you are done previewing (see Figure 4).

You can try other formats too – for example: "Headlines w Stories (Definition List)". Change the format to that and now your layout will have hanging indents (standard definition list formatting).

OK, that looks pretty good, but say you wanted to make those headlines a little bigger than the other content. You can do that with CSS (see Figure 5). Create a CSS class and call it "replayheads" and set the font size to be "larger" and set the weight to be "bold".

Now apply that style to the headlines by selecting it from the "Headln Class" popup.

Preview the Replay again (press the "Play" button) and you should now see that the headlines are larger.

Now if you're really paying attention to details in the layout, you'll see there's a blank line above our feed content that appears to be part of the preview selection. What's happening here is that RSS Replay is inserting content where it is placed in the layout. By default, Dreamweaver puts any content into paragraphs (P tags). So what we have is an extra P tag wrapping our Replayed content. It's easy to fix this -- switch off the preview, so you are looking at the little icon in the layout. The icon should still be selected, and the Tag Insepector (the status area at the bottom of the Dreamweaver window) shows which tags are wrapping the current content. Right now it says "<body><p>". Right click (or control click for you Mac folks) on the P tag there and select "Remove Tag" from the contextual menu.

If you preview again (press the "Play" button) you'll now see that the extra space is gone.

Perfecto! So now you'll want to take a look at the page in the web browser. Save the page and open it in your favorite web browser.

Doh! Until you register RSS Replay the content in the browser will only show "RSS Replay is Unregistered". Once registered, the JavaScript file will be unlocked and you'll see the same thing in the browser that you do in Dreamweaver.

Replaying a Remote Feed

There are really only two important differences between a local feed and a remote feed. First, you have to change the source so the URL is a full URL that includes the protocol (e.g. http://www.whatever.com/path/file.rss). Second, you have to change the method being used to include a proxy that will make the request for RSS content for you (RSS Replay offers two methods -- ASP & PHP).

Start by opening "wddjarticles.html". You're going to replay the feed from Web Designers and Developers Journal so that your readers can see the headlines, and maybe a little bit of the stories.

Select the text that says "Put Replay Here", delete it and insert a Replay. Now to configure the replay, begin with the source. Click the "Sources" button in the inspector. Press the plus button and then enter the URL for the feed: http://webddj.sys-con.com/index.rss (see Figure 6).

Then select the format to display this feed with. Choose "Headlines w Stories (25 Word Stories w UL)". This will display the headlines and stories as a bulleted list, and trim the stories to only the first 25 words.

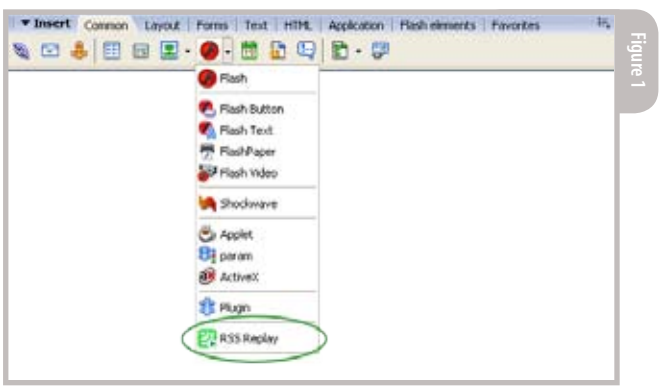
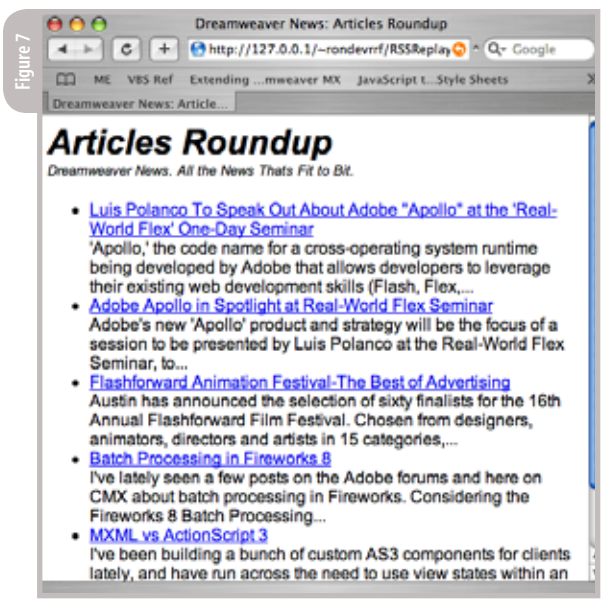


Figure 1

Figure 2



Finally, we're going to specify the method for transferring the feed using the "Method" popup. The default method is "AJAX/JavaScript (local feeds only)". This method is restricted to displaying feeds that are from the same domain as the page -- a restriction based on the way the web browsers handle data requests. To get around that limitation RSS Replay uses a proxy to request the RSS feed from a remote server but to provide the data as if it were from the local domain. The proxy is a server script and has been included in both ASP and PHP -- so all you have to do is pick the server language that you are using. Of course, you will not be able to test this method without having a server to put in on. Select the proxy that is appropriate for your server.

Now if you press the "Play" button you will see the layout including content from a remote source. Don't worry if you see an HTTP error -- sometimes Dreamweaver times-out before the data is completely transferred -- pressing "Stop" and then "Play" again usually solves this problem.

So save this and test it on your server.

Of course, if you haven't registered yet you'll still see "RSS Replay is Unregistered". But once you do register the JavaScript and ASP/PHP pages will be replaced with unlocked versions and your page will work perfectly (see Figure 7).

Lastly, I want to leave you with one more idea. You can customize your display not only with CSS but also by changing the HTML that is being generated by RSS Replay. In the "Format" menu of the property inspector you will also find the "Advanced" option. Selecting that option will present you with a dialog where you can enter the HTML codes that will be used along with special Replay tags that are used to identify where content is placed and what to do with it -- like<<headline>> and <<story>>. I'll leave that for you to play with.

I hope you enjoy the software and build great websites.

Ronald Northrip is the creator of the award-winning RSS DreamFeeder, an extension for Dreamweaver that helps people create RSS feeds from existing Dreamweaver content. His company, Ronald Northrip Software, is dedicated to bringing Web 2.0 tools to Dreamweaver developers everywhere. ron@rnsoft.com

E-Commerce 2.0

The quiet revolution

by Simeon Simeonov

With all the noise the Web 2.0 revolutionaries are making, it's easy to ignore another—this time velvet—revolution. E-commerce 2.0 is coming into maturity and getting ready to relieve its now 10+ year old predecessor. It's about time.

What makes an e-commerce 2.0 site? Well, as Supreme Court Justice Stewart's famous saying (about pornography) goes: "I know it when I see it." This may not be a satisfactory answer but it's an honest one. It's too early to tell. E-commerce 2.0 sites look better, they are easier to use but, most importantly, they drive better economics. Let's look at the e-commerce 2.0 drivers, the trends that define the transition and the architectures and technologies that enable it.

What drives the change? For starters, maturity and perspective. E-commerce has been around for over a decade now. Online retail spending will near $100B in 2006 with close to 20% Y/Y growth. The Internet influences an increasing portion of total retail sales. JupiterResearch predicts the number to go from 27% in 2005 to over 50% in 2010. E-commerce has become strategic. Businesses need to find differentiation because, as Esther Dyson puts it, "You can no longer tell people about your brand; you have to let them experience it." Well, there is something wrong when the real-world Old Navy and Banana Republic stores, both owned by Gap, look nothing like each other yet have identical online experiences.

Another driver is entropy. Many of the largest etailers are running systems that are 6-8 years old, built by long-gone bubble-time integrators on top of highly customized e-commerce platform infrastructure. Brand managers and merchandisers have to talk to IT to get things done. Everything takes too long because IT is understaffed and they often care more about running a site well than about the site making lots of money. These are some of the factors that make Forrester predict a major re-platforming of e-commerce sites, starting this year.

Online consumer behaviors have also changed. Trust in online shopping and payment mechanisms has increased. Consumers are buying more online and they are buying more complex products as well. On the flip side, they are becoming pickier. A survey by Allurent found that when consumers are faced with a frustrating experience, 82% are less likely to return to the site, 55% form a negative impression of the retailer and 28% are less likely to shop at the retailer's physical store. That last one really hurts…

Last but not least, broadband penetration has gotten to a point where etailers can safely start downplaying dial-up traffic and experimenting with richer content and experiences.

I see three big trends framing the evolution of e-commerce 2.0 and helping address the challenges of finding shoppers, turning them into buyers and collecting payment.

First, expect a significant move to more interactive user experiences delivered through rich internet applications (RIAs). The early signs are everywhere: from Gap's QuickLook to Amazon's & Angara's Diamond Search to MyRatePlan's phone chooser to Harvey Davidson's bike configurator. This is just the beginning. With e-commerce 2.0, RIAs will dominate. The main goal will be reducing shopping cart & checkout abandonment which, believe it or not, is sometimes in excess of 50% and is the key ingredient of low conversion rates (some of the best etailers are in the 3-4% range). In a world where shoppers can increasingly get the same product from ever more etailers, innovation will also target search integration and optimizing consumers' browsing & product selection experiences to attract a bigger audience.

The second trend is accelerating disaggregation, brought about by the dual forces of focusing on core competencies and leveraging network effects. Disaggregation is not foreign to e-commerce 1.0. Some businesses have pushed it to the extreme, outsourcing their entire online operations to Amazon, GSI Commerce and the like. For years, Amazon's affiliate Web services have allowed businesses to build sites using the Amazon catalog and backend. That's nothing new, though even more highly-customized, audience-specific sites will serve the Long Tail of consumer interests. What's new in e-commerce 2.0 are network-wide services that provide only a portion of the e-commerce experience yet benefit from focused network effects. Ratings and reviews are a good example. E-commerce 1.0 has aggregators such as NexTag and Epinions. E-commerce 2.0 has BazaarVoice and PowerReviews, which bring ratings & reviews capabilities to all

sites. Payment is another good example. E-commerce 1.0 has PayPal while Google Checkout belongs in 2.0. The key distinguishing features of the 2.0 services are tight integration with the e-commerce experience and the ability to go beyond simple hosting and leverage network effects. The most successful services will reduce the barriers to purchase across sites. Google Checkout and ARPU are two early innovators that merit keeping a close watch on. It is also interesting to ponder whether there is a 2.0 version of e-commerce analytics driven by the following trend.

The third trend is social commerce, which comes in two flavors: content-driven and interaction-driven, or passive vs. active. Examples of content-driven social commerce are already present, albeit not automated. Our purchase choices are influenced by those in our social networks. Brands know that. That's why promotion on MySpace is the cool new thing. What about being able to see what products your friends have viewed or purchased? It's coming to an e-commerce site near you. Interaction-driven social commerce is different yet, again, a new spin on an offline idea—multi-level marketing. Remember Tupperware house parties? Well, the company has started hosting these online. Why can't iTunes start offering incentives to people who recommend songs to others? And, since any individual can become an Amazon affiliate, anyone can have product links in their social network profiles that reward them for sending interested buyers to Amazon. With e-commerce 2.0 these types of social interactions will infiltrate the shopping experience. Combined with disaggregation, it means that social commerce will happen everywhere, not just on the e-commerce sites.

There are three architecture elements that define e-commerce 2.0 sites and help break the virtual store (everything under the same roof) mentality:

- A composite front end that integrates disaggregated services into a coherent, fluid user experience. How is this different from a portal? In a portal, the various pieces of content are often independent of one another. Here, everything is highly integrated from a data and user experience standpoint. The front end will initially run in parallel to the existing e-commerce 1.0 site because etailers will experiment and make the switch to e-commerce 2.0 gradually. Pieces of the front end will be embeddable in other sites. (Yes, even as MySpace widgets.)
- A backend suite with three main purposes: (1) tying into existing e-commerce functionality that doesn't need to be replaced such as the catalog, order processing and customer service, (2) creating an intermediate data layer optimized to support the user experience and (3) maintaining interaction state, a task which becomes a lot more complicated with disaggregation.
- A sophisticated battery of tools for brand managers, merchandisers and analysts that takes IT out of the equation. Control of content, promotions, design, layout, interactivity and analytics should be firmly in the hands of business users and the creative types. IT should worry about scalability, reliability and security.

Great e-commerce 2.0 sites leverage a multitude of technologies, many of which are unfamiliar to e-commerce 1.0 developers.

Rich interactivity requires some combination of Flash and AJAX. Going from simple mashups to flexible composite front ends becomes easier with advanced platforms such as Flex & Atlas from Adobe and Microsoft. Audio and video usage will increase, bringing new tools and servers into the mix. Looking at the work of companies such as Kaon Interactive, I even expect 3D to get a second chance in certain verticals. Core Web services protocols enable disaggregation with a special role for Atom/RSS in dynamic syndication. Social commerce brings in the rest of the Web 2.0 compendium such as user-generated content (some of it pulled from blogs & wikis, the rest captured on e-commerce sites), trust/reputation building and information discovery & management through folksonomies and social networks. Of course, all of this has to be tightly integrated with both horizontal and vertical search, which is no small task. Backend innovations will target complex product configuration (often an information architecture problem more than a presentation problem), navigation path optimization (for example, see Endeca) and new kinds of personalization (based on social context, real-time site activity, etc.). Agile response to user behavior requires integrating real-time business intelligence with traffic flow modeling.

That's a lot of technologies to wrap in a solution bundle. It begs the question about what will the e-commerce platform vendors do. Existing e-commerce platforms are too tied to HTML on the presentation side and lack sophisticated data processing and state management capabilities. Incumbent e-commerce vendors' attempts to make an evolutionary transition to e-commerce 2.0 will fail until they commit to significant re-architecture. This creates an opportunity for three kinds of players: (a) system integrators who move quickly to assemble necessary pieces into solutions, (b) a new generation of e-commerce platform companies such as Allurent, which I'm an investor in, and (c) the network-wide service providers such as ARPU, Google, BazaarVoice and PowerReviews.

This is but a quick glimpse into the future. For etailers, consumers and everyone building Web sites, it'll be exciting times ahead. Know of any e-commerce 2.0 sites & services? Let me know at simeons.wordpress.com.

Simeon (Sim) Simeonov used to be chief architect at Allaire/Macromedia where he shipped more products than he can remember on a summer day. Now he is a technology investor at Polaris Venture Partners in Boston. He still codes on weekends. You can reach Sim at sim_at_polarisventures.com, or enjoy his blog about innovation and venture capital in the post-broadband era at http://simeons.wordpress.com. sim@polarisventures.com

Understanding Classpaths

Step beyond the built-in toolset

by David Stiller

All objects in recent versions of ActionScript are defined by something called classes. Think of classes as blueprints that determine the unique combination of characteristics, actions, and reactions that comprises a particular object of a certain type. By "object," we are talking about the familiar things a Flash developer deals with every day: movie clips (the MovieClip class), text fields (the TextField class), buttons, sounds, math functions, components, you name it. They are all defined by classes.

As of Flash MX 2004 (aka Flash 7), classes are stored in external text files, usually with the .as file extension, and imported into a SWF at compile time, which is the point at which Flash gathers together all your ActionScript, symbols, and other relevant content, and "compiles" them into the special code structure (bytecode) of the SWF file played by the Flash Player.

Out of the box, Flash provides hundreds of built-in ActionScript classes. The great part is, you can even write your own! But there's a catch: the Flash compiler, which converts ActionScript into bytecode, must be told where new classes are located, or it won't be able to find them. That's what this article is about.

A Brief History

Flash 5 was the first to support the #include directive, which made it possible to compile external ActionScript into a SWF. This was a very cool feature, because it encouraged developers to share solutions to common problems in the form of code snippets and function libraries located in external text files. The fact that these snippets could be conceptually distilled from a given FLA made them particularly convenient, especially in light of other ActionScript advances at the time, such as the ability to write custom functions.

Note: It should be mentioned that #include has caused a bit of confusion over the years. This directive is effectively a placeholder: its purpose is to insert external ActionScript into your FLA as if the code were already in the FLA. This is not the same thing as loading ActionScript dynamically — a feature many have asked for, but one that is not yet available in Flash as of this writing.

By the time Flash MX (aka Flash 6) arrived, ActionScript featured a then-new Object.prototype property, and hotshots began to employ object-oriented principles in Flash for the first time. A full discussion of prototype-based object-oriented programming (OOP) is beyond the scope of this article, but for the sake of this brief history, suffice it to say that Object.prototype allowed developers to extend native objects such that, for example, all normal arrays might support a new shuffle() method. This was also true for custom-made objects. Ah, those were the heady days!

Even so, prototype-based OOP was "merely" the sort used in JavaScript: certainly practical, but often sneered at by "real programmers," who tended to prefer the greater breadth of class-based OOP. It wasn't until Flash MX 2004 that ActionScript supported bona fide classes. Thanks to new strict datatyping and something called classpaths, which will be covered in the next section, Flash developers could finally write classes in the same format as those shipped with the product — classes they could create, use, and share in ways more advantageous than ever before.

Classpaths Point the Way

To build the objects required of it, the Flash compiler looks up whatever classes it needs to — for directions, basically. One of the places it checks by default is a special folder it knows as $(LocalData)/Classes, much like you might head to the pantry for staples like rice or sugar. The actual location of this folder is typically one of the following, depending on what operating system and version of Flash you have:

Windows XP

- C:\Documents and Settings\<user name>\Local Settings\Application Data\Macromedia\Flash MX 2004\<language code>\Configuration\Classes
- C:\Documents and Settings\<user name>\Local Settings\Application Data\Macromedia\Flash 8\<language code>\Configuration\Classes

Mac OSX

- Hard Disk: Users: <user name>: Library: Application Support: Macromedia: Flash MX 2004: Configuration: Classes
- Hard Disk: Users: <user name>: Library: Application Support: Macromedia: Flash 8: Configuration: Classes

Article courtesy of Community MX. For more quality articles such as this one, go to www.communitymx.com

In addition, the compiler looks for classes in whatever folder contains the current FLA. These two conceptual locations are stored as user preferences in the global classpaths setting.

To see these classpaths for yourself, proceed as follows. In Flash MX 2004, select Edit > Preferences, choose the ActionScript tab, and click the ActionScript 2.0 Settings button. In Flash 8, select Edit > Preferences, choose ActionScript from the listbox at left, and click the ActionScript 2.0 Settings button.

Note the existing entries of both the dot (.), which represents "current folder," and the aforementioned $(LocalData)/Classes. If either of these is accidentally deleted, simply add it back.

To add a path, click the Add New Path button and either type a folder by hand or click the Browse To Path button to browse. Now, why would you want to add a path? In a word: convenience. With #include, you have to remember the actual file location of your .as file ... if you keep all your function libraries in one place, chances are good that folder isn't the same one that contains your current FLA. With classpaths, however, you can maintain all your class files in a single folder and add its location once to this dialog. From then on, Flash simply "knows" where those classes are.

Global versus Document

If you like, you may specify classpaths on a per-FLA basis. To configure the settings of a given FLA only, select File > Publish Settings, choose the Flash tab, and click the Settings button in the ActionScript version row. You'll see the same dialog, but this time its affect does not apply to all FLAs — only the one you have open. In addition to adding paths, you may specify a frame in which to export these classes in the compiled SWF.

Components offer a similar export option. To see what I mean, drag any item from your Components panel to the Stage. Doing so will place a copy of that component into your Library panel. Right-click or Control-click it in the Library to see its Linkage... properties. You'll see a checkbox for "Export in first frame." This option, and the one similar to it in the document classpaths setting, determines which frame of the SWF should contain the associated class(es); that is, which frame should contain all that ActionScript code. Component classes are actually quite large (anywhere from 26KB to 68KB!). That much weight can actually interfere with preloader code.

Your own custom classes are not likely to be that huge. Unless they are, I would leave the export frame as 1; classes are just text files, after all.

Avoid Collisions by Using Packages

Let us get practical, to illustrate what we have discussed. Say you are interested in scripted animation. You have read Joey Lott's "Programmatic Tweening in Flash" (http://www.communitymx.com/abstract.cfm?cid=9B986) and are now able to move objects around the Stage with instances of the Tween class. Perhaps you are even applying naturalistic movement, thanks to the native easing classes Back, Bounce, Elastic, Regular, and Strong (of course, there's also the None class, but that's specifically for not easing). Now, wouldn't it be nice if there were additional flavors of easing in the world? You bet it would! As it happens, there are.

Author Robert Penner is famous for his easing equations, available free from his website. To use his, simply download the .as files (they're archived in a .zip file) and put them into a folder whose location is specified in your classpaths setting. Wherever you would have referenced a native easing class in your ActionScript, reference one of Penner's instead.

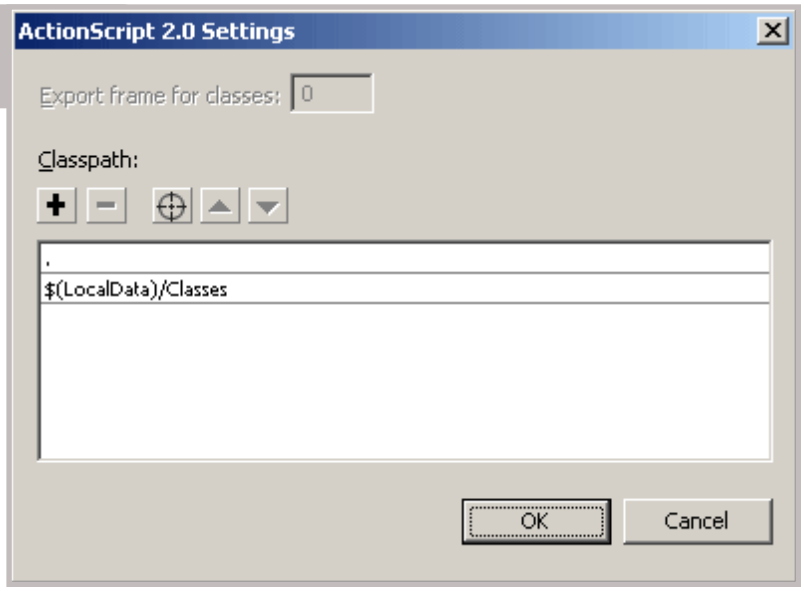
But wait, we already have a problem! Although his collection includes plenty of new classes — Cubic, Expo, Quad, Quart, and more — three of them have the same name as existing native classes (Back, Bounce, and Elastic). If this situation weren't managed, you'd have a class collision on your hands! If you wanted to use Penner's Easing instead of the native one, how would the compiler know which class you meant? Flash would be confused and would either output an error message or compile with unpredictable results.

Thankfully, a handy solution exists. Borrowing from Java and other OOP languages, Flash supports something called packages. Packages sidestep the abovementioned ambiguity by logically categorizing classes in a hierarchical manner. They are manifested on the hard drive by way of nested folders and are used to organize classes with a common purpose. Flash's native easing classes, for example, are located in the package mx.transitions.easing, because they're related conceptually to Flash's transition classes. If you look into the real location of $(LocalData)/Classes on your computer, you'll find an mx subfolder, and inside that (among many others), a transitions subfolder, and finally an easing folder, so you can see how this ties together. As long as packages are unique, classes will not be confused.

By convention, developers usually arrange packages based on the reverse of their Internet domain names, which are guaranteed to be unique. This means that classes written by Robert Penner, available from robertpenner.com, are organized in the package com.robertpenner (his easing classes, specifically, in com.roberpenner.easing). Classes written by CommunityMX partners are organized in the package com.communitymx . Those written by Flash guru Grant Skinner are organized in the package com.gskinner . You get the idea.

As long as your global classpaths setting includes the root folder that contains these package subfolders, you'll be fine. For example, I personally keep a folder named AS2 Classes inside my My Documents folder. The path to AS2 Classes on my computer is C:\Documents and Settings\David Stiller\My Documents\AS2 Classes — and this path is listed in my global classpaths setting.

Figure 1



Because of that, I'm already done. I do not need to further specify \com\robert-penner\easing\, even though that's where the Cubic.as file actually resides. On a Mac, this location might be Hard Disk: Users: <user name>: AS2 Classes, and that path would likewise be enough for the global classpaths setting.

To recap, then, once your classpaths are set, all you have to remember is a class's name and optionally its package, if it has one.

Save Keystrokes by Using Import

Let's go back to the native easing classes for a moment. To invoke the Regular class, you could specify its fully qualified package and class name.

```
mx.transitions.easing.Regular
```

But after repeated references, all that typing does a number on your fingers. Are there any shortcuts? You bet. Enter, the import statement.

The import statement allows you to access classes without having to spell out their packages. You could shorten repeated references to the above, for example, by putting the following at the top of your script, just once:

```
import mx.transitions.easing.Regular;
```

After that line, you would only have to type Regular when referring to that class. In fact, you may even use the asterisk (*) character as a wildcard. Because you know there are several easing classes in the mx.transitions.easing package, you could import them all at once by typing this:

```
import mx.transitions.easing.*;
```

Note: The import statement must be repeated for each new frame of code you write. If you import classpaths in frame 1, all scripts in frame 1 will be able to take advantage of them, but scripts in frame 2 will not. Multiple use of the same import statement in any single frame will generate an error, so when you use it, use it at the top of each relevant frame.

Summary

The basic classes in Flash reside in a special $(LocalData)/Classes folder. Flash already knows this classpath location, so common tasks, such as instantiating a Sound object, are straightforward:

```
var myMusic:Sound = new Sound();
```

Even the more advanced classes, such as transitions, easing, and Flash 8's new BitmapData class, are located in packages within the $(LocalData)/Classes folder. Because of this, the classpath is already taken care of, but any reference to these classes requires the package's full path:

```
var myImage:flash.display.
BitmapData = new flash.display.
BitmapData(parameters here);
```

... Or, to save keystrokes, the import statement:

```
import flash.display.BitmapData;var
myImageA:BitmapData = new
BitmapData(parameters here);var
myImageB:BitmapData = new
BitmapData(parameters here);// etc.
```

Custom classes, such as those you might acquire online or write yourself, may be located in the same folder as the current FLA, thanks to the dot (.) entry in the global classpaths setting. Even if your custom classes use packages (which is to say, the class files are located in subfolders), you're in good shape, provided those subfolders reside in the same folder that contains the FLA.

It is recommended, however, that you keep third-party classes in a centralized location, such as a folder under My Documents. If you go this route, make sure to add the full path of this location to your global classpaths setting. Once that's out of the way, you may refer to these classes in any new FLA by name alone — or by package and name, unless you use the import shortcut.

Conclusion

In this article, you learned about classpaths and their usage in Flash MX 2004 and 8. You learned where to configure global and document classpath settings, how packages organize classes, and even how to save keystrokes by using the import statement. Armed with this information, you're ready to begin using third-party classes. Have fun stepping beyond the built-in toolset!

David Stiller is a career multimedia programmer/designer whose portfolio includes NASA, Adobe, and major US automotive and boat manufacturers. He likes anaglyph 3D photography, finely crafted wooden game boards, Library of Congress field recordings, and Turkish coffee. David is self-taught and gets a kick out of sharing "aha!" moments with others through consultation, mentoring, and regular contributions to a variety of Flash forums. He lives in Virginia with his amazing wife, Dawn, and his beguiling daughter, Meridian.

New Development
Strategies for New Markets

Flexible solutions

by Barry Neu

traditional application development processes make it challenging to meet the time and budget constraints facing businesses today. Going from the requirements gathering phase to writing up design documents to delivering applications can take many months—which is often too long for executives wanting to address new market opportunities or solve old business problems.

The need to shorten application development cycles is familiar to programmers. Yet, several systems and business requirements in place today seem to conspire against it. Corporate applications and data residing in departmental silos, threats to information inside and outside an organization, and demands for more integrated, dynamic user interfaces are just a few of the issues that can drain development resources and slow application rollout.

Rapid development of complex applications

As a performance improvement solutions developer serving health care companies nationwide, bConnected develops intranet- and Internet-based solutions that deliver accurate, real-time information to empower managers to change an organization's behavior, processes, and performance. The flexible solutions overcome organizational silos and give managers accurate views into many facets of their businesses. Integral to our success has been the ability to cost-effectively develop complex solutions in as little as six weeks—from initial client meetings to full deployment.

The streamlined development relies on industry standards such as Java and Adobe Flex. Using the open solutions enables us to address business demand for intuitive front-end processes fully integrated with enterprise systems. Our rapid development environment centers around a toolset with three in-house-created components: Link Vision, Link Design, and Link Execution.

Built upon Adobe Flex, Link Vision is a user interface design tool for creating intuitive business intelligence dashboards and report cards. Link Design is a Java desktop application that acts as a process design tool, enabling us to visually program data collection, translation, and presentation. The third tool, Link Execution, runs in a J2EE-certifed application server and is the execution engine for performing actions outlined in Link Design. Our solutions support multiple application servers and services, as well as databases from small MySQL systems all the way up to enterprise Oracle systems.

Focus on customer problems, not browser incompatibilities

bConnected's work in the health care industry is indicative of many of the challenges that developers typically face. The industry is slow to adopt technology, and professionals frequently rely on manual processes. As a result, solutions have to be easy to implement and manage, and particularly easy to use. Otherwise, staff will resort to familiar manual processes.

In addition, health care organizations often have information scattered throughout the enterprise. Operational data, patient care information, financial records, and other critical data lives in spreadsheets, Microsoft Access databases, and dozens of other systems large and small. With the help of Adobe Flex and our Link Design tool, we can quickly create intuitive business intelligence dashboards that securely aggregate and present any detail managers require. The dashboard can reach into databases, pull information from file systems, and interpret data on spreadsheets.

Significant from a developer perspective is the speed in which we can accomplish this. Because we are not writing individual lines of code for user interfaces and integration points, dynamic dashboards can be created in days. An added advantage is that Flex overcomes a major obstacle facing developers—getting web applications to run reliably in any browser.

Typically, the only way developers streamlined creating sophisticated HTML applications was to target a specific browser platform and version, so that it was possible to gauge an application's behavior. Unfortunately, this approach to development does not reflect the reality of how individuals and businesses work, with end users relying on a wide range of browsers.

With Flex, we are not concerned about whether a person is running our application in Microsoft Internet Explorer 5.5 or version 6.0, Firefox, or any other browser. Because the Flex application runs on the Adobe Flash player—already installed on more than 98 percent of Internet-connected computers—we

know that our interfaces will behave in the same way regardless of the browser. It is an important feature that can eliminate a week or longer from development and help ensure applications reach as wide an audience as possible.

By developing in Flex, our team stays focused on solving customer problems, not browser problems. Rich, interactive applications can be deployed on the free Flash Player, without requiring users to wait for special downloads, time-consuming page refreshes, or separate applications to launch.

Keeping clients engaged and satisfaction high

Equally important for enhancing development is becoming better able to respond to changing customer requirements, especially in the design stage. A benefit of using Adobe Flex is that we can mock up UI screens quickly in hard code data, without having to fully develop data services. This has improved development processes, as well as customer satisfaction.

We can now take a more iterative approach to design, providing clients with interfaces to evaluate throughout a project. The ongoing communication helps ensure that all parties are engaged in the process. And in the end, clients get exactly what they want. Development reviews can be done as often as weekly, giving clients the opportunity to evaluate the look and feel of an interface, explore drill-down capabilities, and determine the best ways to present information.

For example, if one of our clients wants to evaluate medical services received by patient populations, we can use Flex capabilities within our Link Vision tool to create a display container—pie chart, graph, tabular chart, or other form—without writing a single line of code. If the client opts for a pie chart presentation of data, we can develop a portlet, with each slice representing a service category and each slide defined by an XML information feed. To connect the database information to the UI, our Link Design tool is used to create the database query.

A key advantage of using Flex is the ability to easily amend development to meet changing client needs. For instance, if a customer decides to use a column chart instead of a pie chart, all we have to do is remove the pie chart and drag and drop a column chart into the portal container. The same XML stream can be used. We just need to configure what each column should represent. The revisions can be handled in minutes, as opposed to the hours it could take trying to rework lines of HTML code.

Breaking down data silos

Our work with Rotech Healthcare, a leader in providing medical equipment, respiratory equipment and services, and respiratory medications for home use, highlights the advantages of more open, object-oriented development. As the medical industry grows and competition heats up across all areas, health care companies like Rotech are striving to gain more insight into operations to improve patient care and profits.

Rotech's Director of Operations, John Sullivan, describes a problem familiar to many health care companies. "We need to pull data out of legacy systems, dig out from under the paper, and better analyze the metrics we use to drive our business," he says.

Across more than 500 company locations nationwide, Rotech business managers are responsible for overseeing staff, monitoring patient satisfaction, and tracking on-time delivery of medical supplies and patient care. In addition, they must ensure that products and services are billed to Medicare as well as to private insurers in a timely manner in order to get reimbursed.

Traditionally, Rotech used several 'silo'd' legacy systems, requiring managers to constantly toggle among various databases to find information about patient care, procurement, financial performance, and other activities. Important issues monitored daily by managers include items held during the billing process, equipment not yet received into inventory and asset tracking systems, and equipment not yet confirmed as delivered to or picked up from Rotech patients.

Enhanced productivity and user experiences

To address Rotech's challenges, we used our Flex-based Link Vision to develop a business dashboard—an intuitive desktop panel that provides consolidated views of multiples sources of back-end data—for the company's managers nationwide. Using the powerful dashboard, managers can drill down into aggregated information fast, analyzing revenue, patient care, and inventory metrics that help them meet performance and budget goals. Not only have operations improved, but managers also find that they have more time to spend assisting field staff and generating new business.

"The Location Manager Dashboard that bConnected built around Adobe Flex has changed the behavior of our organization," says Mike Dobbs, chief operating officer at Rotech. "Since deploying the Flex-powered dashboard, we've seen a 20 percent reduction in assets waiting to be processed in inventory and asset tracking systems."

Raising performance standards

Performance boosts in Flex 2 further increase the benefits that Rotech and other clients are realizing. The rendering lag time for large data sets displayed in grids has been eliminated and scrolling across data is seamless. Additionally, the new Flex Builder based on the popular Eclipse IDE helps accelerate development by as much as 50 percent and gives us access to an even greater pool of skilled developers.

Our focus is always on building mutually beneficial relationships with clients. Integrating Adobe Flex into our processes goes a long way in supporting that goal by accelerating application development and expanding the capabilities of web systems. For us, that means more growth and better services, while our clients continue to discover how to improve their operations.

Barry Neu is vice president of engineering at bConnected Software, Inc.

Rich Internet Applications

Getting Trained with Total Training's Flex 2

By Tariq Ahmed

You read the blogs, see a couple of articles, and hear all the buzz about Flex 2 and how it can really transform how your users can fluidly interact with information and the business in an exciting and efficient new way.

But at the same time you're under tight deadlines, you're an expert in the technology you're already using, and are able to maintain a certain pace and deliver projects within a predictable timeframe.

You want to learn, but don't have the time to scour through thick technical books (which at this time this article was written didn't even exist for Flex 2). And even if you did, you find that your ability to absorb a lot of information from a tech book is challenging.

You would love to go to a hosted course that is presented by a knowledgeable expert who will not only visually guide you through the learning process, but verbally as well. Personally, I find I'm able to retain a lot more information if presented by someone, so I'll always take the in-class option over books.

Not to say books are bad, I love books, but to me I find their value as being a reference. At the same time, taking 3-4 days off of work to attend in class training is often difficult to swing (need to get your managers buy-in, the courses are often $2000+, the cost skyrockets if you have to fly to another city to attend, etc…).

Well, if you've been using time or budget an excuse not to learn, I'm glad to say that your excuse is gone!

Introducing Flex 2 – Rich Internet Applications by Total Training

The folks at Total Training provide a very cool product for computer based training courses (CBT). CBT of course is nothing new and has been around for years; but traditional CBT's are nothing more than an online book that is a little bit interactive (via some quizzes here and there that you have to answer correctly to be able to proceed).

If you're the kind who can read the words for hours, but later realizes you haven't retained a thing, you end up having to re-read it again – that CBT approach is not much better than a book.

Total Training's approach is to combine the best of both worlds. CBT gives you the price advantage and time saving convenience; while hosted courses give you that 'in class' presentation approach where someone is talking and explaining the content to you.

Figure 1

Figure 2

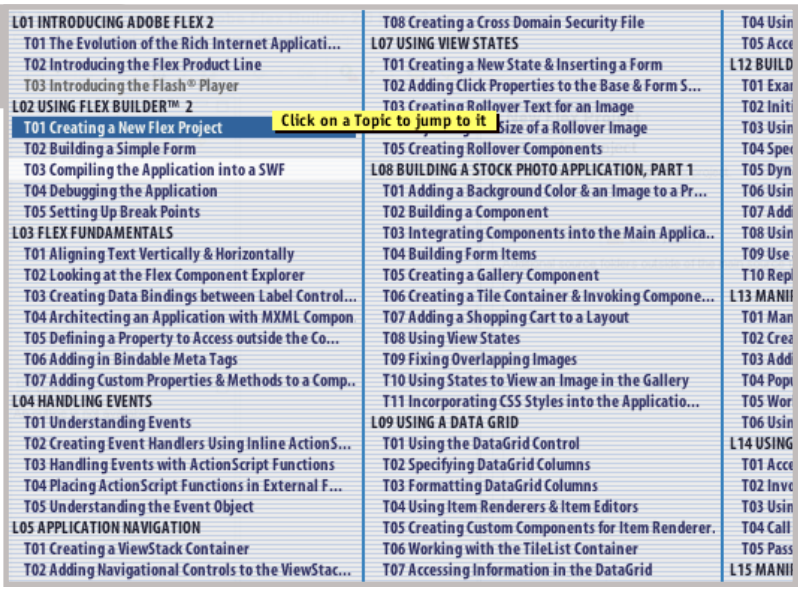


Figure 3

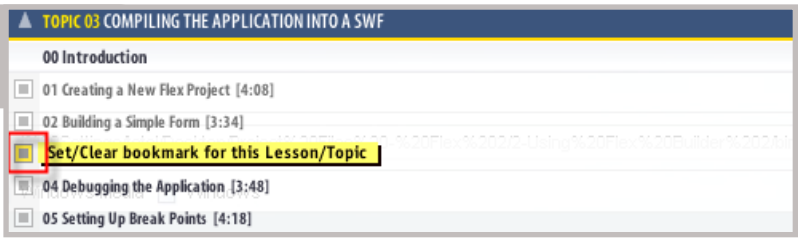
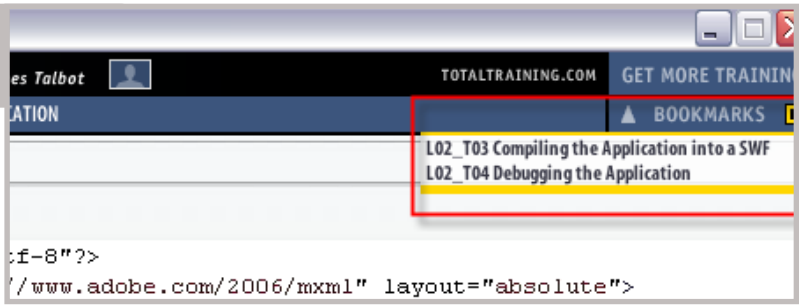


Figure 4



How It Works

The DVD-ROM based product is based on high-resolution video that combines recorded footage of the presenter, presentation slides, and programming demonstrations.

When installing the product, it installs the player as well as the course files. In this particular case it installs a whopping 135 Megs of course files – which is 16 chapters worth of source code ready to go.

So the end result is like being in a classroom while at home, with the full control of being able to navigate around the topics. As the presenter codes through code samples, you have those exact samples that are on your computer allowing you to work on the labs/exercises just like in a classroom. Instead of reading pages and pages of code, the presenter builds the samples as he goes along – just like in the classroom.

Meet James

Say hi to James Talbot! He's the presenter of this course. James has been with Adobe for six years now, and has worked in both the sales engineering and training teams for the Flash and Flex products.

James is well known in developer circles, and you can often catch him giving presentations to user groups and conferences. You may also be familiar with his literary works such as Object Oriented Action Script 2.0 by New Riders Press, and the upcoming *Flex 2: Training from the source* by Adobe Press.

On the consulting side of things he's worked on some heavy duty RIA applications with AOL/Time Warner, JP Morgan, and TV Guide.

So what more could you want? An extremely seasoned pro that works with the company that makes the product, and even better - his background is both technical and training!

Navigation

The heart of the product of course is the player. Like any media player it has the usual play, pause, rewind, and forward buttons. But they really went through a lot of trouble to ensure super rapid topic navigation.

Content is structured by Lessons and Topics. There are 16 lessons that include:

- Introduction to Flex
- Using Flex Builder 2
- Flex Fundamentals
- Handling Events
- Application Navigation
- Retrieving XML Data
- Using View States
- Building a Stock Photo Application
- Using a Data Grid
- Architecting an Application
- Using a Repeater
- Manipulating XML
- Connecting to Web Services
- Manipulating Complex Data
- Formatting and Validating

Each of these lessons include many topics. At the top of the player, the current Lesson and Topic will be displayed; but by using your handy dandy mouse you can easily jump across topics in a lesson, or jump to completely different lessons (see Figure 1).

If you're not sure what Lesson a Topic is in, or just want to browse all the topics, you can click on the Contents button, which will bring up the full index so you can jump to it quickly (see Figure 2).

One of the benefits of a book of course is that you can jump back to a piece of information at any point in time by bookmarking it, which of course you can't do in a class.

Total Training's product also allows you to do this. Any topic can be bookmarked, and easily retrieved in the Bookmarks drop down menu (see Figures 3 and 4).

The Content

If the player is the heart, the content is the soul. The courseware includes stunningly rich visuals that are not only interesting to watch and hold your attention, but at the same time are laden with valuable knowledge.

But it's more than just presentation slides; James shows in a live fashion everything you need to know to build a Flex application. So although you see James himself, as well as the slides, you also get into the nitty-gritty of the code itself.

Who This Is For

The prime candidate for this course is anyone interested in learning Flex, but it also encompasses anyone evaluating RIA technologies and interested in seeing how Flex works up close and what the anticipated learning curve is for their team.

I would also recommend this product for intermediate level Flex developers, and I think seasoned pros would also learn a couple of tricks that they didn't know about. For example, a lot of experienced Flex developers have been able to get by without having to learn leveraging breakpoints, but James shows you how easy it is to take advantage of this feature in Flex Builder 2.

Pricing and Availability

The product is available for both Windows and MAC, and retails for $149.99. For more information on the product, as well as to order, visit Total Training's website at http://www.totaltraining.com.

Figure 6

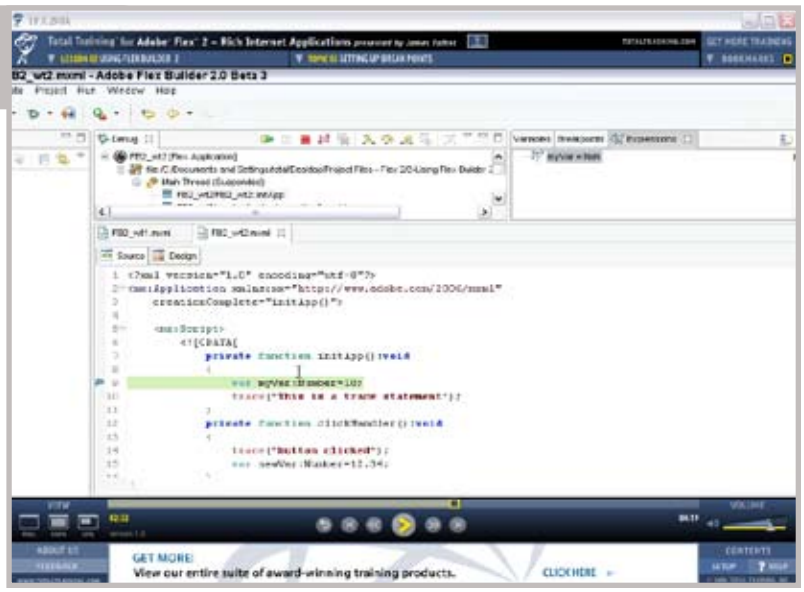
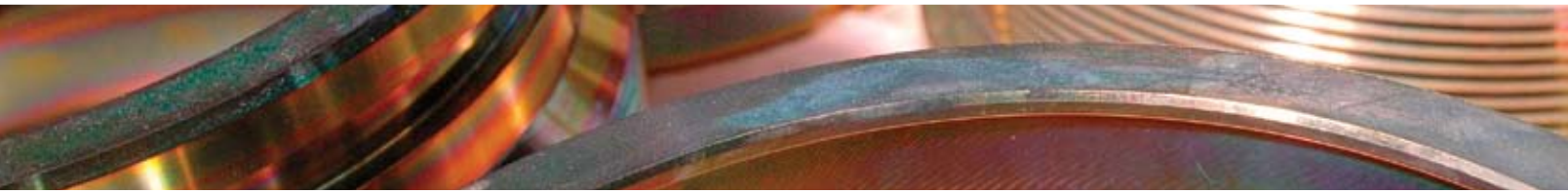


Figure 6



Designing Loosely Coupled Flex 2 Components

Powerful tools

by Marco Casario

flex 2 applications may have complex architectures and they are usually made up of multiple files: MXML, Actionscript and CSS.

Coding the entire application within a single file is not a best practise. It makes the code harder to maintain and reuse and the application is not architected into a logically pieces.

Flex 2 allows developers to break into small external pieces the project, creating separated MXML files and maintain them separately.

The process of dividing Flex application into logic modules has many benefits. It allows the developer team to work and debug a single module independently, isolating errors and functionality of applications. It makes the code easier to maintain and reuse across multiple applications.

Each MXML file is an MXML component but only the main MXML application can load external MXML components. In fact only one Application tag can exist for a single application, everything else is MXML components.

In order to design MXML components you create a MXML file where the root tag is not an Application tag but it's a component tag (such as Vbox, Panel, Canvas, Button, Datagrid, etc.) with the declaration of the http://www.adobe.com/2006/mxml namespace.

The following code shows a simple MXML component, named custDataGrid.mxml, that is made up of a Datagrid to store data :

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

<mx:DataGrid id="myDG" >
<mx:columns>
 <mx:DataGridColumn headerText="Posts"
dataField="title" />
<mx:DataGridColumn headerText="Date"
dataField="pubDate" width="100" />
 </mx:columns>
</mx:DataGrid>

</mx:VBox>
```

The root tag is a simple Vbox component where the xmlns:mx="http://www.adobe.com/2006/mxml" is declared. Once created external modules, the main application invokes the MXML component as the following example :

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.
adobe.com/2006/mxml" layout="absolute"
xmlns:cust="comp.*" >
<mx:Panel title="Comtaste's Blog
Reader">

<cust: custDataGrid width="80%" />

</mx:Panel>

</mx:Application>
```

The main application adds a new XML namespace and uses a package name to represent all the components defined in a folder :

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

The "cust" namespace refers to MXML components into the "comp" folder, in fact as a best practise all components should be stored in a subdirectory

Component invocation is the same as with any other MXML tag, the only difference being the customized prefix rather than "mx" :

```
<cust: custDataGrid width="80%"  />
```

Note that the MXML tag name corresponds to the component's filename.

Tightly coupled versus loosely coupled component

To make components configurable and reusable throughout your applications, you might want to create MXML components that accept properties, launch methods and dispatch events. To achieve this, components must not depend on a specific application (tightly coupled component), their variable's name or tag instance's name. In this scenario if the code of the application or the component changes, reliant code won't work anymore and we'll have to modify the tightly coupled component to reflect that change.

A much cleaner and better approach is to develop a loosely coupled component that dispatches the events to propagate return data to the application and contains property declarations for passing information to it.

This approach allows to create MXML components as "black boxes" with the fol-

lowing benefits :

- They are easier to reuse and maintain
- They don't know nothing about any other components' inner workings
- They are not dependent on a variable's name and tag instance's name

As we said, in order to create a loosely coupled component we need properties to store information and an event model to dispatch events that contain the return data.

Properties are usually defined with Actionscript (although you can define them with MXML) and placed within a <mx:Script> block :

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

  [Bindable]
  public var lista:ArrayCollection;
 ]]>
</mx:Script>
```

The same is for methods of MXML components :

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

  [Bindable]
  public var lista:ArrayCollection;

  public function justWrite():String
  {
   return "This is a method of the
component";
  }
 ]]>
</mx:Script>
```

In the following example the component defines a property that represents the dataprovider that will populate the Datagrid and a method that returns a string. This is the entire code for the custDataGrid.mxml file :

```
<?xml version="1.0" encoding="utf-8"?>
<mx:VBox xmlns:mx="http://www.adobe.
com/2006/mxml">
<mx:Script>
 <![CDATA[
  import mx.collections.
ArrayCollection;

  [Bindable]
  public var lista:ArrayCollection;

  public function justWrite():String
  {
   return "This is a method of the
component";
  }
 ]]>
</mx:Script>

<mx:DataGrid id="myDG"
dataProvider="{lista}" >

</mx:DataGrid>

</mx:VBox>
```

NOTE FROM FLEX 2 LIVEDOCS : A public variable declaration or a set function in an <mx:Script> tag becomes a property of the component. A public ActionScript function in an <mx:Script> tag becomes a method of the component.

We can access to the component's methods or pass data to component's properties from the main application file :

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.
adobe.com/2006/mxml" layout="absolute"
xmlns:cust="*">
<mx:Script>
 <![CDATA[
  import mx.collections.
ArrayCollection;
  [Bindable]
        public var myData:
ArrayCollection = new ArrayCollection
(

  [{A:2000},{A:3000},{A:4000},{A:4000}
,{A:3000},{A:2000},{A:6000}]);

 ]]>
</mx:Script>
<cust:custDataGrid id="custDG"
x="258" y="89" lista="{myData}"/>

<mx:Label text="{custDG.justWrite()}"
x="300" y="44"/>
 </mx:Application>
```

In the main application we populate an ArrayCollection variable named "myData" and we pass it within the MXML component tag declaration through the following line of code :

```
<cust:custDataGrid id="custDG"
x="258" y="89" lista="{myData}"/>
```

The Flex data binding mechanism was used to show that properties defined in custom components can take advantage of data binding feature. The "lista" variable is a public property of custDataGrid custom component so we passed it to the custDataGrid tag declaration.

Similiarly, it's possibile to creata a data binding invoking a method of the custom component :

```
<mx:Label text="{custDG.justWrite()}"
x="300" y="44"/>
```

The text attribute of the Label control will receive the text returned by the public function of custDataGrid custom component :

```
public function justWrite():String
  {
   return "This is a method of the
component";
  }
```

The last step for designing a loosely coupled component is to handle the dispatching of an event that contains the return data. Each MXML custom component dipatches events that can be customized in 3 simple steps :

- using the [Event] metadata tag
- creating an event object
- Dispatching the event and create the function to handle the event

The [Event] metadata tag defines the events that components can dispatch. It's possible to declare the [Event] metadata tag in an Actionscript classes, just after the package definition and above the class definition :

```
Package com.casario
```

```
{
[Event(name=" changeBlog", type="
flash.events.Event ")]
public class custComp
{

}
}
```

or in the <mx:Metadata> tag of an MXML file :

```
<mx:Metadata>
 [Event(name="changeBlog", type="
flash.events.Event ")]
</mx:Metadata>
```

Once the [Event] metadata was created the component has to dispatch the event using the dispatchEvent() method. The dispatchEvent() method accepts the event object as argument as shown in the following code :

```
dispatchEvent(new
Event("changeBlog"));
```

We modify our custDataGrid.mxml file to dispatch an event when the user clicks on an item of the datagrid. The main application will handle the event fired by the custom component and write some text in a Text Area control.

Open the custDataGrid.mxml with Flex Builder 2 and add the [Event] metadata tag as shown below:

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

<mx:Metadata>
  [Event(name="changeBlog",
type="flash.events.Event")]
 </mx:Metadata>

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

  [Bindable]
  public var lista:ArrayCollection;

  public function justWrite():String
  {
   return "This is a method of the
component";
  }

  private function changeHandler():
void
  {
   dispatchEvent(new
Event("changeBlog"));
  }
 ]]>
</mx:Script>

<mx:DataGrid id="myDG"
dataProvider="{lista}"
change="changeHandler()" >

</mx:DataGrid>

</mx:VBox>
```

The [Event] metadata defines a "changeBlog" event with a generic Event type and makes the event public so that the MXML compiler recognizes it :

```
<mx:Metadata>
[Event(name="changeBlog", type="flash.
events.Event")]
</mx:Metadata>
```

The event is dispatched when the change event of the DataGrid is triggered :

```
<mx:DataGrid id="myDG"
dataProvider="{lista}"
change="changeHandler()" >
```

As you can see the "changeHandler" function simply executes the dispatchEvent() method :

```
private function changeHandler():void
  {
   dispatchEvent(new
Event("changeBlog"));
  }
```

The main application has the role to handle the event triggered by the custom MXML component with a simple event handler function defined into the <mx: Script> block :

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.
adobe.com/2006/mxml" layout="absolute"
xmlns:cust="*">
<mx:Script>
 <![CDATA[
  import mx.collections.
ArrayCollection;
  [Bindable]
        public var myData:
ArrayCollection = new ArrayCollection
(

  [{A:2000},{A:3000},{A:4000},{A:4000}
,{A:3000},{A:2000},{A:6000}]);

        private function change-
BlogHandler (event:Event):void
        {
          myLabel2.text += "Event
fired by Datagrid:"+ event.type + "\n"
;
        }

 ]]>
</mx:Script>
<cust:custDataGrid id="custDG"
x="258" y="89" lista="{myData}" change
Blog="changeBlogHandler(event)"/>

<mx:Label text="{custDG.justWrite()}"
id="myLabel"  x="283" y="63"/>
 <mx:TextArea  id="myLabel2"  x="205"
y="239" height="102" width="273"/>
</mx:Application>
```

The custDataGrid component fires the "changeBlog" event and associates an event handler function to it :

```
<cust:custDataGrid id="custDG"
x="258" y="89" lista="{myData}" change
Blog="changeBlogHandler(event)"/>
```

The changeBlogHandler function takes the event object as argument and prints a simple text into the TextArea control :

```
private function changeBlogHandler
(event:Event):void
        {
          myLabel2.text += "Event
fired by Datagrid:"+ event.type + "\n"
;
        }
```

Observe the use of the event object with the syntax event.type that returns the type of the event triggers (in the example it returns changeblog as value !)

NOTE: Flex Builder 2 automatically finds the reference to the the custom event defined into the component and shows it with the code hint :'

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

iTVCON.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

[fig. flexbuilder_customEvent.png]
Flex Builder 2 identifies the event name

While we have seen how simple it is to create a custom event for our loosely coupled component, but this approach also has some limitations if we want to send data to the custom event. The fact is that the flash.events.Event class doesn't permit developers to add properties to it.

In fact we use the dispatch event just to notify that something happened in the MXML custom component. But what if we need to send complex data to a custom event ?

No need to be worried. This is done by creating custom event classes with Actionscript ! I'll show you how to develop custom event classes with a real example !

Creating Custom Event Actionscript Classes for sending complex data

With Flex it's possible to create a custom event that uses an event object of a customized type. One just needs to create a subclass that extends the Event class and adding properties to it. Don't forget that custom components that extend existing Flex classes inherit all the events of the base class.

To develop a subclass we have to create an actionscript class that typicaly extends the flash.events.Event class, the base class for all event objects. After that we call the super() method of the Event class in order to call the constructor of the superclass and pass the event type to it. We are required to override the "clone()" method to return a new copy of the event object by setting the type property and any new properties.

To dispatch a new event from your custom component we create a new istance of the custom actionscript class and we pass properties to it.

Let me show you a real example to explain this approach. We develop a simple blog reader that allows users to select from a list of blogs using a combo bo. The application is made up of 2 MXML components that contain the DataGrid, to show the blog's post, and a combobox with the blog's list.

We'll start from the custom event class (saved as "evtClass.as" placed into the "com" folder) that accepts a simple String property :

```
package com
{
 import flash.events.Event;

 public class evtClass extends Event
 {

public var evProp:String;

  public function evtClass(evParam:
String,type:String)
  {
   super(type);
   this.evProp = evParam;
  }
   override public function clone():
Event
  {
   return new evtClass(evProp,type);
  }
 }
}
```

As we said before, our class extends the flash.events.Event class, calls the super() method of the Event class and overrides the clone() method.

Now, create the first component from your Flex Builder 2 (File > New > MXML Component) and save it as custBlogList.mxml. This component contains the combobox and defines its dataprovider as an ArrayCollection object :

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

 <mx:Metadata>
  [Event(name="changeBlog", type="com.
evtClass")]
 </mx:Metadata>

 <mx:Script>
  <![CDATA[

   import com.evtClass;

   private function init():void
   {
    myArray.addItemAt({label:"Alistai
r McLeod", data:"http://weblogs.macro-
media.com/amcleod/index.xml"}, 0);
   }

   private function changeHandler():
void
   {
    var eventObj:evtClass =
new evtClass(myCombo.value as
String,"changeBlog");
    dispatchEvent(eventObj);
   }

  ]]>
 </mx:Script>

 <mx:ComboBox id="myCombo"
change="changeHandler()" creationCompl
ete="init();myCombo.selectedIndex=0" >
 <mx:ArrayCollection id="myArray">
            <mx:Object label="Mike
Chambers" data="http://weblogs.macro-
media.com/mesh/index.xml"/>
            <mx:Object label="Matt
Chotin" data="http://weblogs.macrome-
dia.com/mchotin/index.xml"/>
    </mx:ArrayCollection>
</mx:ComboBox>

</mx:VBox>
```

We used the [Event] metadata tag that defines the events that the MXML component can dispatch. The value assigned to the type property is equal to our custom event class :

```
<mx:Metadata>
  [Event(name="changeBlog", type="com.
evtClass")]
</mx:Metadata>
```

Once the user selects a blog under the ComboBox, the change event is fired and the changeHandler() is called :

```
<mx:ComboBox id="myCombo"
change="changeHandler()" creationCompl
ete="init();myCombo.selectedIndex=0" >
```

The event handler has the only purpose to create an instance of the custom event class and to pass the value selected by the user to the new istance (myCombo.value as String) :

```
private function changeHandler():void
   {
```

```
     var eventObj:evtClass =
new evtClass(myCombo.value as
String,"changeBlog");
     dispatchEvent(eventObj);
    }
```

The second MXML component contains the DataGrid control and accepts the dataprovider from the main application. Now you should save the file as custBlogData.mxml:

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

<mx:Script>
 <![CDATA[
  import mx.collections.
ArrayCollection;
  import flash.net.*;
  [Bindable]
  public var lista:ArrayCollection;
 ]]>
</mx:Script>

 <mx:DataGrid id="myDG" hori-
zontalCenter="14" verticalCen-
ter="18.5" dataProvider="{lista}"
width="100%" change="navigateToURL(new
URLRequest(myDG.selectedItem.link),'_
blank');" >
  <mx:columns>
        <mx:DataGridColumn
headerText="Posts" dataField="title"
/>
        <mx:DataGridColumn
headerText="Date" dataField="pubDate"
width="100" />
    </mx:columns>
 </mx:DataGrid>

 </mx:VBox>
```

The DataGrid defines two columns. The first one receives data from the RSS feed the user has selected (custBlogList component). The second one receives data from the ArrayCollection variable ("lista").

Last but not least there is the main application (BlogReader.mxml) file that invokes the two components, calls an HTTPService for retrieving rss data and defines the event handler function:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.
adobe.com/2006/mxml" layout="absolute"
xmlns:cust="*" creationComplete="hs.
send()" >
 <mx:HTTPService
     id="hs"
     url="{selectedMenu}"
     useProxy="false"/>

<mx:Script>
  <![CDATA[

    import com.evtClass;

    [Bindable]
    private var selectedMenu:String=
"http://weblogs.macromedia.com/
amcleod/index.xml";

    private function eventFired(event:
evtClass):void
    {

      selectedMenu = event.evProp;
      if (selectedMenu == "null")
      {
      mx.controls.Alert.show("Please
Choose a valid Blog");
      return;
      };
      hs.send();
    }
  ]]>
 </mx:Script>

 <mx:Panel width="70%" height="70%"
layout="absolute" title="Comtaste's
Blog Reader" horizontalCenter="0" ver-
ticalCenter="-14.5">

 <mx:Label text="{hs.lastResult.rss.
channel.title}'s Blog" id="myLbl"
x="48.5" y="43" fontWeight="bold"/>
     <cust:custBlogList  x="161" y="10"
changeBlog="eventFired(event)" />
     <mx:Label x="48.5" y="10"
text="Select a Blog" height="25"/>
 <cust:custBlogData width="80%"
x="48.5" y="69" lista="{hs.lastResult.
rss.channel.item}" />
 <mx:ControlBar>
 <mx:Label text="Developed by Marco
Casario" />
 <mx:LinkButton label="http://casario.
blogs.com" click="navigateToURL(new
URLRequest('http://casario.blogs.
com'),'_blank');"/>
  </mx:ControlBar>
 </mx:Panel>

</mx:Application>
```

The HTTPService tag has a data binding for its url attribute. The {selectedMenu} value is populated when the changeBlog event of the custBlogList fires :

```
<mx:HTTPService
     id="hs"
     url="{selectedMenu}"
     useProxy="false"/>

<cust:custBlogList  x="161" y="10"
changeBlog="eventFired(event)" />
```

The eventFired() event handler function uses the event object to retrieve the evProp defined into the actionscript class (evtClass.as) that contains the myCombo.value :

```
private function eventFired(event:evt-
Class):void
    {
      selectedMenu = event.evProp;
      if (selectedMenu == "null")
      {
      mx.controls.Alert.show("Please
Choose a valid Blog");
      return;
      };
      hs.send();
    }
```

Loosely coupled components are powerful tools for Flex 2 applications because they enable developers to improve the reusability and maintainability of their code. Using this approach is possible to define logic modules that respond to different user interactions throughout multiple applications.

Modules can also be customized with properties to pass information with one another and events to return information back.

Marco Casario is CEO of Comtaste, a company devoted to develop Rich Internet Applications on the Web and for mobile devices. He collaborates intensively with Adobe Italy as a speaker at conferences and as a consultant for Flash, Flex, and Flash Lite. Learn more about Marco Casario at his blog http://casario.blogs.com. m.casario@

Rich Internet Applications: AJAX,

www.AjaxWorldExpo.com

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 $AVE!

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

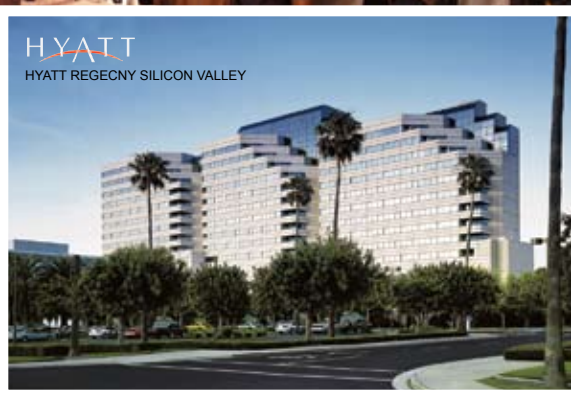
GROUP DISCOUNTS AVAILABLE:
— 5 Delegates from Same Company for only $995 (each) —
— Register a Second Person for only $1195 —

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

End-to-End Rapid Application Development with Data Services & DAOFlex

Excertps from chapter 6 of Rich Internet Applications

by Yakov Fain, Victor Rasputnis, & Anatole Tartakovsky

The simplest way to explain Flex Data Services (FDS) is to compare them with Flex Remoting. Simply put, FDS addresses only a subset of operations facilitated via Flex Remoting – result set requests. However, whereas Flex Remoting enables one-way requests, FDS combines one-way requests with the publish/subscribe mechanism so that besides the original result set FDS sends the client live updates produced by other clients of the same destination. And there's one more dimension in which Data Services depart from Flex Remoting – support for hierarchical collections, but we won't be covering that subject in this book.

In other words, FDS resolves the task of programming data collaboration: Several users may edit different rows and columns of the "same" DataGrid and see each other's changes automatically pushed by the server. Now, what if they overlap each other's work? In terms of FDS that's called a conflict and the FDS API provides for flexible conflict resolution, which may require the user's intervention.

An FDS destination can be configured for working with the data that is persisted to a data store as well as supporting scenarios that persist the data in the server's memory. To that end, FDS provides Java and ActionScript data adapters that are responsible for reading and updating a persistent data store according to its type. In this chapter we'll focus on use cases involving Java adapters.

Flex Data Services & Automation: Problem Statement & Solution

Robust in enabling collaborative manipulation of data, FDS demands a substantial development effort in case of persistent data stores. In particular, you need to build:

- A Java Data Access Object class that implements retrieve, update, delete, and insert of the data;
- Java Data Transfer Objects (DTO);
- A matching ActionScript data transfer object class;
- A configuration file, which registers identity columns of the result set and, optionally, argument types for every retrieval method and other parameters.

We just mentioned four classes/files containing hard-coded names of the fields and there are more. To function properly, these hard-coded values have to be kept in sync, which is an additional maintenance effort whenever the data structures change.

Addressing this complexity, the main idea of this chapter is not to cover every twist of the FDS API, but rather automate the development effort that FDS takes for granted. We'll start with a "manual," albeit simplified, example of using DataServices. Then we'll introduce you to the methodology of complete code generation based on the pre-written XSL templates and FDS-friendly XML metadata, which will be extracted from the annotated Java abstract classes.

This methodology is fully implemented in DAOFlex – an Open Source utility that's a complementary addition to this book. We'll gradually introduce this tool by leading you through a process of creating the most comprehensive template that generates a complete DataServices Data Access Object DAO. Finally, we'll show you how to run and customize DAOFlex in your development environment so that writing and synchronizing routine DataServices support classes becomes a task of the Ant building tool and not yours!

A "Manual" FDS Application

Let's handcraft the application presented in Figure 6.1. This application displays a Panel with a scrollable DataGrid that we consciously did not size in the horizontal dimension, so that all columns can be viewed without shrinking. The database result set is ultimately produced by the following SQL query that will use a bound variable in place of the question mark:

```
select * from employee where start_
date < ?
```

There are two buttons below the DataGrid: Fill and Commit. As the names imply, these buttons pull the original data from the database table and submit the data changes back to an FDS des-

tination. A separate Parameters panel permits entering parameters of the back-end method behind the Fill button, which, in our case, is the employee start date :

Building the Client Application

Let's build the client application first. The full listing of the application is presented in Listing 6.1. We start with defining the mx:DataServices object (a k a ds), which points to the destination "Employee." Later, when we get to the server components we'll discuss mapping this destination to the backing Java class:

```
 <mx:DataService id="ds"
destination="Employee"
fault="onFault(event)"  />
```

We provide only a rudimentary handler of the fault event that's sufficient to keep us aware of any anomalies that may occur along the way. Dynamic referencing fault and faultString properties will spare us from casting to a specific event:

```
 private function onFault(evt:Event):
void {
  Alert.show(evt["fault"]["faultString
"], "Fault");
 }
```

Then we define a handler of the application's onCreationComplete event where we instantiate a collection to be eventually associated with our mx:DataService object and, most importantly, set both autoCommit and autoSyncEnabled of the ds to false:

```
 private function onCreationComplete()
: void {
  collection = new ArrayCollection();
  ds.autoCommit=false;
  ds.autoSyncEnabled=false;
 }
```

By setting autoCommit to false we state that all updates have to be batched and explicitly submitted to the server as a single transaction during the ds.commit() call. By setting autoSyncEnabled to false we effectively protect our local instance of data from delivery of messages caused by other clients connected to destination "Employee." Setting autoSyncEnabled to false is entirely optional, and we use it to avoid dealing with application specific conflict resolution. In particular, in the handler of the Commit button's click event you might uncomment the first line to support the "optimistic" way of handling the conflicts:

```
 private function commit_onClick():
void {
  //ds.conflicts.acceptAllClient();
// Optimistic conflict handling, as
oppose to ds.conflicts.acceptAllServ-
er();
  ds.commit();
 }
```

Last, we have to initiate the population of the local collection with the ds.fill() method, which we do inside the click event handler of the button Fill:

```
 private function fill_onClick():void
{
  ds.release();
  ds.fill(collection,         param_
getEmployees_startDate.selectedDate);
 }
```

The scripting portion of the application is completed so let's build the UI. We create a DataGrid with the dataProvider bound to our collection in Listing 6.1. For brevity's sake, we didn't list all the columns here: you'll have a chance to scrutinize them in the subsequent section of this chapter.

The DataGrid and ControlBar with Fill and Commit buttons are put inside a Panel, with DataGrid's title bearing the name of the destination and a specific getEmployees method of that destination, which will ultimately be invoked during the ds.fill() call. The second panel, titled Parameters, contains a form with a single item mx:DateField. Both panels are embraced by the VDividedBox.

We've included a linkage variable of the data transfer type to ensure that the corresponding ActionScript class (EmployeeDTO) will be linked into the generated SWF file.

```
<?xml version="1.0" encoding="UTF-8"?>
<mx:Application xmlns:mx="http://www.
adobe.com/2006/mxml" backgroundColor="
#FFFFFF" creationComplete="onCreationC
omplete()">
 <mx:DataService id="ds"
destination="Employee"
fault="onFault(event)"  />
 <mx:VDividedBox width="800"
height="100%">
  <mx:Panel title="Employee::getEm-
ployees()" width="800" height="70%">
   <mx:DataGrid id="dg" dataProvi
der="{collection}" editable="true"
height="100%">
     <mx:columns><mx:Array>
       <mx:DataGridColumn
dataField="EMP_ID" headerText="Emp Id"
/>
       <mx:DataGridColumn
dataField="MANAGER_ID"
headerText="Manager Id"  />
       <mx:DataGridColumn
dataField="EMP_FNAME" headerText="Emp
Fname" />
       .  .  .  .
     </mx:Array></mx:columns>
   </mx:DataGrid>
   <mx:ControlBar>
     <mx:Button label="Fill"
click="fill_onClick()"/>
     <mx:Button label="Commit"
click="commit_onClick()" enabled="{ds.
commitRequired}"/>
   </mx:ControlBar>
  </mx:Panel>
```

```
  <mx:Panel title="Parameters"
width="100%" height="30%">
    <mx:HBox height="100%"
width="100%">
      <mx:Form label="getEmployees()">
        <mx:FormItem
label="startDate:">
          <mx:DateField id="param_getEm-
ployees_startDate" selectedDate="{new
Date()}"/>
        </mx:FormItem>
      </mx:Form>
    </mx:HBox>
 </mx:Panel>
</mx:VDividedBox>
<mx:Script>
 <![CDATA[
 import mx.controls.Alert;
 import mx.collections.
ArrayCollection;
 import com.theriabook.datasource.dto.
EmployeeDTO;
 private var linkage:com.theriabook.
datasource.dto.EmployeeDTO = null;
     [Bindable]
 private var collection :
ArrayCollection;
 private function fill_onClick():void
{
  ds.release();
  ds.fill(collection,         param_
getEmployees_startDate.selectedDate);
 }
 private function onCreationComplete()
: void {
  collection = new ArrayCollection();
  ds.autoCommit=false;
  ds.autoSyncEnabled=false;
 }
 private function commit_onClick():
void {
  ds.conflicts.acceptAllClient();
  ds.commit();
 }
 private function onFault(evt:Event):
void {
  Alert.show(evt["fault"]["faultString
"], "Fault");
 }

 ]]>
</mx:Script>
</mx:Application>
```

The application above doesn't cover all use cases of the FDS API. We tried to keep it as small as possible for one reason: to enable metadata-based code generation. Ultimately, it will be entirely up to you which code you'd elect to generate by modifying the DAOFlex templates. Finally, we present the listing of the ActionScript class EmployeeDTO that our collection uses in communicating with the Employee destination:

```
package com.theriabook.datasource.dto
{

 [Managed]
 [RemoteClass(alias="com.theriabook.
datasource.dto.EmployeeDTO")]
 public dynamic class EmployeeDTO
 {

  // Properties
  public var EMP_ID : Number;
  public var MANAGER_ID : Number;
  public var EMP_FNAME : String;
  public var EMP_LNAME : String;
  public var DEPT_ID : Number;
  public var STREET : String;
  public var CITY : String;
  public var STATE : String;
  public var ZIP_CODE : String;
  public var PHONE : String;
  public var STATUS : String;
  public var SS_NUMBER : String;
  public var SALARY : Number;
  public var START_DATE : Date;
  public var TERMINATION_DATE : Date;
  public var BIRTH_DATE : Date;
  public var BENE_HEALTH_INS : String;
  public var BENE_LIFE_INS : String;
  public var BENE_DAY_CARE : String;
  public var SEX : String;

  public function EmployeeDTO() {
  }
 }  //EmployeeDTO
}
```

Creating Assembler & DTO Classes

The time has come to work on the Java side, which is a rather tedious process, so we'll gradually go top-down.

Our first stop is an Assembler class that the FDS Employee destination should map to. As the Flex documentation suggests, you can implement the methods on your Assembler class in several ways:

- Extend flex.data.assemblers.AbstractAssembler and override the fill(), getItem(), createItem(), updateItem(), and deleteItem() methods as needed;
- Configure these methods via XML definitions against a class that doesn't extend the AbstractAssembler class;
- Or a combined approach, where methods defined via XML declarations are used if defined, otherwise the AbstractAssembler methods are invoked.

We'll take an XML approach that lets us declare a so-called sync-method. The XML contract of the destination's sync-method prescribes that it accepts a single parameter: a List of flex.data.ChangeObject elements. We find it convenient to control how we want to process data changes. In particular, we'd like to maintain the following order: all deletes, then all updates, and then all inserts. After all, if the user deletes a record for an employee with EMP_ID=123 and then inserts a new record with EMP_ID=123, we certainly wouldn't want our sync-method to issue the INSERT, followed by DELETE FROM employee WHERE EMP_ID=123 during the batched FDS data modifications.

Let's keep in mind that our ultimate focus is the metadata-based code generation. Should you decide to have your Assemblers as descendants of the AbstractAssembler, you'd have the liberty of modifying the corresponding DAOFlex template.

Listing 6.3 presents the complete XML describing the destination Employee. Under the default configuration scenario, this XML would go inside the <services> node of the flex-data-services.xml file, located in the WEB-INF/lib/flex folder of your Web application.

We set com.theriabook.datasource.EmployeeAssembler as the exact name of the class mapped by our destination, with the methods java.util.List getEmployees_fill(java.util.Date dt) and the List getEmployees_sync(List lst) acting as the fill and sync methods respectively.

Even though XML doesn't explicitly declare that that the fill-method returns a List or that the sync-method takes a List, this is a part of the XML contract for Assembler classes in destinations:

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

```
<destination id="Employee">
 <adapter ref="java-dao"/>
<properties>
 <source>com.theriabook.datasource.
EmployeeAssembler</source>
 <scope>application</scope>
 <metadata>
   <identity property="EMP_ID"/>
 </metadata>
<network>
<session-timeout>0</session-timeout>
<paging enabled="false"/>
<throttle-inbound policy="ERROR" max-
frequency="500"/>
<throttle-outbound policy="ERROR" max-
frequency="500"/>
</network>
<server>
 <fill-method>
 <name>getEmployees_fill</name>
 <params>java.util.Date</params>
 </fill-method>
 <sync-method>
   <name>getEmployees_sync</name>
 </sync-method>
</server>
</properties>
</destination>
```

The structure of the EmployeeAssembler Java class is pretty straightforward. This class delegates the actual data retrieval and update of the data store to EmployeeDataServiceDAO class, which we'll discuss next:


```
package com.theriabook.datasource;
import java.util.*;

public final class EmployeeAssembler
{
 public EmployeeAssembler()
 { }

 public final List getEmployees_
fill(Date startDate) {
   return new EmployeeDataServiceDAO().
getEmployees(startDate);
 }

 public final List getEmployees_
sync(List items) {
   return new EmployeeDataServiceDAO().
getEmployees_sync(items);
 }

}
```

The chapter then leads the reader line by line through the entire set of programming tasks required to build a fully functional create-retrieve-update-delete (CRUD) application based on Employee Data Service destination. Once the job is well-know, the reader is invited to automate it once and for all. Here is one more fragment.

Introducing Metadata

Let's look at the snippet from the XML file generated by the DAOFlex utility – Employee.xml. Please note the name of the Java package – com.theriabook.datasource, the name of the Assembler's fill method – getEmployees(), names on the transferring structures on the Java and ActionsScript side, both pointing to array of com.theriabook.dto.EmployeeDTO objects, the name of the connection pool – jdbc/theriabook, and the name of the method's parameter – startDate:

```
<?xml version="1.0" encoding="UTF-8"?>
<WEBSERVICE NAME="Employee"
PACKAGE="com.theriabook.datasource"
TYPE="DAOFlex" >
<SERVER LANGUAGE="Java" MODE="JEE">
<SQL ACTION="SELECT"
NAME="getEmployees" POOL="jdbc/theri-
abook" SCOPE="public"
ASTYPE="com.theriabook.dto.
EmployeeDTO[]" JAVATYPE="com.theri-
abook.
dto.EmployeeDTO[]"
>
<PARAM IN="Y" INDEX="1"
JAVATYPE="Date" NAME="startDate"/>
</SQL>
</SERVER>
</WEBSERVICE>
```

Starting at this point, we'll be working our way through this XML while building the complete XSL stylesheet from scratch. Once we make this stylesheet, it'll be capable to manufacture any DataServiceEmployeeDAO, DataServiceDepartmentDAO, etc. – as long as we have the metadata XMLs like the above one. You're probably wondering at this point: "What's the input of the DAOFlex that allows it to generate this XML?"

The input for DAOFlex is an annotated Java class, like the one presented in Listing 6.11:

```
package com.theriabook.datasource;
import java.util.Date;
import java.util.List;
/**
* @DAOFlex:webservice pool=jdbc/the-
riabook
*/
public abstract class Employee {
/**
* @DAOFlex:sql
* sql=select * from employee where
start_date < :startDate or start_
date=:start_Date
* transferType=com.theriabook.dto.
EmployeeDTO[]
* updateTable=employee
* keyColumns=emp_id
*/
public abstract List getEmployees(Date
startDate);
}
```

Reference

- Rich Internet Applications with Adobe

STREAM57

Customized Flash-based media solutions

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

visit us
stream57.com

e-mail us
streamline@stream57.com

call us
212.909.2550 x1012