SYS-CON

DIRECTOR & FLASH : A STRONG TEAM P. 22

2.9

MX developer's journal

THE LEADING MAGAZINE FOR MACROMEDIA MX DEVELOPERS & DESIGNERS

volume 2 issue 9 2004 www.mxdj.com

DREAMWEAVER
What's a Server Side Include?

FLASH
An Award-Winning Application

FIREWORKS
Fireworks Flash Buttons in a Flash

FREEHAND
Typography, Part 1

THE DIVISION THAT PUTS DIRECTOR TO WORK

ROCHESTER INSTITUTE OF TECHNOLOGY · R·I·T · 1829 ·

$5.99US $6.99CAN

Seapine Software™
changing the world of software development

Complete source code and asset management in Dreamweaver MX—now possible with Surround SCM.

Dreamweaver users know a beautiful Web-based product is only skin deep. Underneath, it's a tangle of hundreds or thousands of ever changing source files. Without a good development process and strong tools, bad things happen. Surround SCM can help.

Surround SCM lets you...

Track multiple versions of your source files and easily compare and merge source code changes.

Check out files for exclusive use or work in private workspaces when collaborating on a team.

Automatically notify team members of changes to source files—push changes through your organization.

View complete audit trails of which files changed, why, and by whom.

Associate source code changes with feature requests, defects or change requests (requires additional purchase of TestTrack Pro).

Remotely access your source code repository from Dreamweaver MX.

Surround SCM adds flexible source code and digital asset control, powerful version control, and secure remote file access to Dreamweaver MX. Whether you are a team of one or one hundred, Surround SCM makes it easier to manage your source files, letting you focus on creating beautiful Web-based products.

Features:

Complete source code and digital asset control with private workspaces, automatic merging, role-based security and more.

IDE integration with Dreamweaver MX, JBuilder, Visual Studio, and other leading Web development tools.

Fast and secure remote access to your source files—work from anywhere.

Advanced branching and email notifications put you in complete control of your process.

External application triggers let you integrate Surround SCM into your Web site and product development processes.

Support for comprehensive issue management with TestTrack Pro—link changes to change requests, bug reports, feature requests and more.

Scalable and reliable cross-platform, client/server solution supports Windows, Linux, Solaris, and Mac OS X.

Want to learn more?

Download Seapine's just-released white paper, **Change Management and Dreamweaver**, to discover tips, tricks and best practices for achieving full software configuration functionality. Visit www.seapine.com/whitepaper.php and enter code WM0604 to receive your copy today.

www.seapine.com
1-888-683-6456

Surround SCM Source Control

TestTrack Pro Defect Management

QA Wizard Automated Testing

macromedia ALLIANCE PARTNER

software development 13th annual product excellence award JOLT

MX developer's journal

september 2004

7 Logged In
The Flash Video explosion
by greg stern

10 Letters
What you said
by you

12 Flex vs. JSP
A higher ROI and better impact
by peter ent

30 xile
Cartoon
by louis f. cuffari

58 vanguard
How to Build a www.fernando-nieto.com
by Fernando Nieto

18

What's a Server Side Include?
A method to think about
by dan short

22

The Hugo Boss Academy
A case study
by thomas biedorf

28

Fireworks Flash Buttons in a Flash
Two-way functionality adds zip
by tom green

32

Typography
...choose your words, carefully
by ron rockwell

38 Snow Problem
The physics of Havok
by nik lever

44

The Division That Puts Director to Work
...with Shockwave 3D
by andrew m. phelps

52 Double-Clicking A Document
A professional projector makes it easier
by tom rockwell

CFDynamics

A Division of Konnections Inc.

RUN YOUR CFML:

Within BlueDragon Server, or as a native J2EE web application.

NOW AVAILABLE!

BlueDragon [6.1]

New Atlanta COMMUNICATIONS

As one of the ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to deliver new and cutting edge technologies to the ColdFusion community. We are excited to announce our premiere partnership with New Atlanta by offering BlueDragon hosting. We look forward to seeing how BlueDragon expands the possibilities of ColdFusion.

QUALITY CF 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 has complete redundancy in power, HVAC, fire suppression, bandwidth, and security
- 24 / 7 network monitoring

FANTASTIC SUPPORT SERVICES

- Comprehensive online support
- Knowledgeable phone support
- We focus on your individual needs

SIGN UP FOR A HOSTING ACCONT TODAY!

Use promo code: MXDJ04I and receive a FREE T-SHIRT!

WWW.CFDYNAMICS.COM
866 - CFDYNAMICS
866-233-9626

BlueDragonCertified V.I.P. Hosting Partner

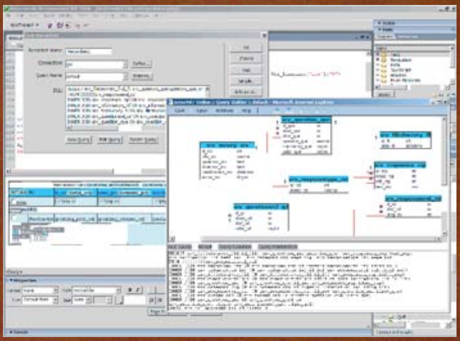
macromedia ALLIANCE PARTNER

Need a hand with your work? Get the MX Kollection.

Introducing MX Kollection - the suite of extensions designed to change the way you create dynamic web applications with Dreamweaver MX and MX 2004.

Our customers think of MX Kollection as the next level in Dreamweaver MX visual software development.

ColdFusion and PHP developers will find our product enabling them to visually develop e-Commerce, CMS, CRM, Backend and other web solutions with increased efficiency, quality and capability.



Create powerful dynamic lists

- Sort, filter and page result sets
- Perform multiple deletes
- Easily create Master/Detail lists

Build unified add/modify forms

- Client side and server side validation
- Insert into two tables
- Create form actions such as send mail or delete related record
- File/Image upload and resize

Create recordsets visually

- Build complex SQL queries across multiple tables quickly

... and many more

Interakt

macromedia ALLIANCE PARTNER

Download the demo and see the features and benefits of our extensions at http://www.interaktonline.com

Logged In

The Flash Video explosion

by greg stern

I was recently looking at the winners of the Clio Awards and the Cannes Lions – annual awards that recognize excellence in the advertising industry. As I went through the winners in the Internet category, several things stood out prominently:

- ***An ad isn't always an ad.*** Many of the winners are not ads at all, at least not banner ads or pop-ups. Rather, they are Web sites that blur the distinction between advertising, product information, and entertainment.
- ***Sales and marketing sites lead the pack.*** Some of the most compelling Flash sites I have ever come across are being created for sales and marketing purposes. As I visited the sites of the agencies that won and examined their portfolios, I saw that they had created an amazing number of really compelling sites.
- ***Flash and Flash video were in every winning site.*** Every winner and runner-up I saw was created with Flash. Quite a number of them also made use of Flash video.

The Hottest Trend: Video on the Web

We have been tracking the progression of the use of Flash video on Web sites for some time. Lately we have noticed an explosion in the number of sites that integrate video. Some are straightforward implementations with strictly video clips like Discovery Broadband, while others use video as just another element of their rich interactive experience, like Vodafone Future Vision.

A number of trends seem to be driving this explosion:

- ***Rich Internet advertising delivers.*** Rich media advertising delivers results that are orders of magnitude better than almost anything else out there. Increasingly this means integrating video into rich ads.
- ***Broadband adoption is exploding.*** A recent Pew Internet and American Life study claimed that more than half of American adults have broadband at home or the office – and prices for broadband service keep plummeting, driving the trend further.
- ***Flash helps you get video on the Web.*** Flash MX Professional 2004 provides a complete feature set for integrating video.
- ***Flash Player lets everyone see video on the Web.*** Flash Player 7 improves performance and offers progressive downloading of video from Web servers.
- ***Flash Player is ubiquitous.*** Flash Player 6, required to view Flash video, has reached a 94% adoption level among Web-connected computers – a full 30% better than any other video player out there. Even the new Flash Player 7 is at a 67% adoption level.
- ***Flash Video Streaming Service makes streaming a reality.*** Macromedia's partnerships with Speedera Networks and VitalStream – which provide customers with huge, scalable Flash video streaming services – make it possible for large media companies like Comcast to build out large Flash video implementations.

What You Think About Video

Because we are inherently interested in the overall video trend, we ran a sur-

"Rich media advertising delivers results that are orders of magnitude better than almost anything else out there" vey in May asking what you are doing now – and what you are planning to do – with video. Frankly, the results surprised us. According to you, video is hot!

- Nearly fifty percent of you are putting video on Web sites today, and another tenty-five percent plan to do so soon.
- Almost sixty percent of you told us you simply want to put existing video on your site.
- Eighty-five percent of you told us that having full creative control over how your video is integrated into your site is important.
- Seventy-nine percent of you are interested in building interactive video experiences by tying video to various elements on your Web pages.
- Seventy-four percent of you are so interested in building rich video experiences that you are willing to learn Flash in order to achieve that end.

"we have noticed an explosion in the number of sites that integrate video"

Despite this being a heady time for those of us who follow the trends in Flash video, we do have some concerns. As much as you want to use Flash video, you may not have the tools and knowledge to apply it effectively:

- Over Sixty-seven percent of you told us that your understanding of Flash video is poor.
- Eighty pecent of you told us that you wanted to use embedded video in Flash, which is the worst option for all but niche cases.

Flash Video Kit: The Easiest Way to Get Video on the Web

We combined what we saw in the market trends with what you told us you were doing – and planning to do – with Flash video and developed the Flash Video Kit. Available through September 30 when you purchase Studio MX 2004 with Flash MX Professional 2004, the Flash Video Kit includes the following:

- Dreamweaver MX 2004 extension for quickly adding Flash video (streaming or progressive download) to your HTML pages
- Special version of Sorenson Squeeze to help you convert your video resources easily into the Flash video (FLV) format

Find out more about Studio MX 2004 with Flash MX Professional 2004.

Note: If you currently have Flash MX Professional 2004, you can purchase the Flash Video Kit separately through the Macromedia Online Store – and get a discount because you already own Studio MX 2004 with Flash MX Professional 2004.

Other Flash Video Improvements

We worked with VitalStream to create a new "lite" version of the Flash Video Streaming Service, designed especially to work with the Flash Video Dreamweaver extension included in the Flash Video Kit. While this isn't a free subscription, it is reasonably priced and comes with a 15-day free trial. It is perfect for streaming all those videos your boss has been after you to put up on the company Web site.

To improve your knowledge and understanding of Flash video, we have made a commitment to improve the educational materials supporting it. We have completely revamped the Flash Video Topic Center, adding new tutorials and updating existing ones.

This is just the beginning. In the months ahead, you will see many new tutorials and articles on Flash video on the Macromedia Web site (www.macromedia.com).

Finally, I'd like to invite you to visit our latest iteration of the Flash Video Gallery, which features some of the most compelling customer examples of using Flash video. See if you don't agree with me that we are in the middle of a creative explosion.

Greg Stern is the vice president of developer relations at Macromedia
gstern@macromedia.com

SYS-CON MEDIA
President & CEO
Fuat Kircaali, 201 802-3001
fuat@sys-con.com
Vice President, Business Development
Grisha Davida, 201 802-3004
grisha@sys-con.com
Group Publisher
Jeremy Geelan, 201 802-3040
jeremy@sys-con.com

ADVERTISING
Senior Vice President, Sales & Marketing
Carmen Gonzalez, 201 802-3021
carmen@sys-con.com
Vice President, Sales & Marketing
Miles Silverman , 201 802-3029
miles@sys-con.com
Advertising Sales Director
Robyn Forma, 201 802-3022
robyn@sys-con.com
Advertising Sales Manager
Megan Mussa, 201 802-3023
megan@sys-con.com
Associate Sales Managers
Kristin Kuhnle, 201 802-3026
kristin@sys-con.com
Beth Jones, 201 802-3028
beth@sys-con.com
Dorothy Gil, 201 802-3024
dorothy@sys-con.com

PRODUCTION
Production Consultant
Jim Morgan, 201 802-3033
jim@sys-con.com
Lead Designer
Louis F. Cuffari, 201 802-3035
louis@sys-con.com
Art Director
Alex Botero, 201 802-3031
alex@sys-con.com
Associate Art Director
Richard Silverberg, 201 802-3036
richards@sys-con.com
Assistant Art Director
Tami Beatty, 201 802-3038
tami@sys-con.com

SYS-CON.COM
Vice President, Information Systems
Robert Diamond, 201 802-3051
robert@sys-con.com
Web Designers
Stephen Kilmurray, 201 802-3053
stephen@sys-con.com
Matthew Pollotta, 201 802-3054
matthew@sys-con.com
Online Editor
Martin Wezdecki 201 802-3045
martin@sys-con.com

ACCOUNTING
Financial Analyst
Joan LaRose, 201 802-3081
joan@sys-con.com
Accounts Payable
Betty White, 201 802-3002
betty@sys-con.com
Accounts Receivable
Shannon Rymsza, 201 802-3082
shannon@sys-con.com

EVENTS
President, SYS-CON Events
Grisha Davida, 201 802-3004
grisha@sys-con.com

CUSTOMER RELATIONS
Circulation Service Coordinators
Shelia Dickerson, 201 802-3082
shelia@sys-con.com
Edna Earle Russell, 201 802-3081
edna@sys-con.com
Linda Lipton, 201 802-3012
linda@sys-con.com
JDJ Store Manager
Brundila Staropoli, 201 802-3000
bruni@sys-con.com

SYS-CON MEDIA

SUPERIOR MANAGED HOSTING

- Intelligent Routing
- Redundancy
- Network Security
- Service Level Agreement
- Environmental Controls
- Network Uptime Guarantee
- Scalability
- OC3/SONET Backbone
- Backup Power
- Physical Security
- Fire Protection
- Server Hardware Guarantee

IF YOU'VE BEEN SEARCHING for a reliable managed hosting partner, your search for greener pastures is over. There is no longer a need to settle for inferior support and lack of accountability. ServerSide takes the guess work, and the associated hassles, out of working with a technology partner.

ServerSide provides managed web hosting solutions for a wide range of customers including; Fortune 500 corporations, small and medium sized businesses and non-profit organizations located across the United States and around the World.

We provide a single point of accountability for all your web hosting infrastructure needs — while adding value, not cost.

>> SPECIAL OFFER

Sign up online for a new shared hosting account now at www.serverside.net and ServerSide will waive the set-up fee.

ENTER CODE: mxd2004

The grass is greener at
www.serverside.net
888.682.2544
hosting@serverside.net

serverside™
Dynamic Web Development. Superior Managed Hosting.

Letters

What you said

RoboDemo: Lynchpin of the E-Learning Market

Charles E. Brown gives me a clear picture of what RoboDemo can do with the e-learning demo. I have puchased the product, but I would like to further know about an add-on module that needs for creating Flash editable FLA files. What is it and where can I find to buy the mentioned module?

Diana Man; June 17, 2004

Layout Issues

Today, I received the first magazine (Vol. 2, issue 6) in my new subscription and I have a suggestion, for what it's worth.

In the two articles I have read thus far, the authors lament that they cannot include valuable information or examples, using terms such as "due to >space constraints..." and "...space does not permit me to show..." and "...I do not have room to reproduce here."

This kind of space restriction would make perfect sense in any publication with size limitations, but not when so much of the available editiorial space is wasted with double spreads of huge non-important graphics and teaser text.

For example, pages 14–15, 30–31, 34–35, 42–43, and 50–51 might make your graphic designer feel all warm and fuzzy, but seem like a waste to me when what I want is more useful content, especially at the price of this magazine and subscription.

I would suggest that the solution is not to bar your writers from stating they would show or tell something if they had more room, but perhaps to tone down the useless graphic design and give the space saved to the writers who deliver what we readers want.

Bill Wright
Multimedia Design Engineer Sr.
Lockheed Martin

validcode

Interesting article, but why bother to rewrite code in the various default files when you can set the same attributes, such as using <strong></strong> instead of <b></b>, in the Preferences section under File?

Am I missing a point here? Seems like making a mountain out of a mole hill.

Roger Lipera, Aug. 12, 2004

Creating Games in Macromedia Flash MX 2004

This is a well-written and interesting article that shows a clear devotion to the language. Makes even an "old, non-game playing guy" like me wish I was born a little later. There are shades of Teillhard's noogenesis in this article's vision of the computer's potential to interconnect people globally in a common event.

Bob Turcotte, Aug 16, 2004

Flash Virtuosos

I think this story is a testimony that Flash will definitely get you noticed, whether it's by American design awards, clients, etc. I think every designer would love to use Flash, but at the same time, they're afraid that they will fall off the face of search engines. Check out www.graphicwise.com, the guys who designed ADA and they have used a very subtle Flash / HTML combo that adds to the overall experience. Now I have to go read my Flash MX users' manual to see how I can build an everlasting impression ... what pressure? :(

Martin, July 28, 2004

XML for Web Designers

Leveraging Macromedia Support

Clear, concise, enjoyable, and neat summary of XML for Web designers who use Dreamweaver and ColdFusion!

Colm Brazel; May 20, 2004

Our databases of inventory, orders, etc. sits on an iSeries Server. Where can I find information on how to connect my ColdFusion Web sites with these databases? I was thinking that using XML might be an option? We currently have the ColdFusion Enterprise MX sitting on an NT Server.

Ruby Pena; May 21, 2004

Ruby, you can find the information for accessing XML data with ColdFusion in my book *XML For Web Designers Using Macromedia Studio MX 2004*. You can also find more information about accessing Databases as well as XML in Ben Forta's ColdFusion book. My book is for people who have no XML experience and wish to learn the fundamentals. Macromedia's Web site also contains papers regarding accessing databases with ColdFusion.

Kevin; May 22, 2004

It’s everybody’s PDF™

Finally, a software company that offers affordabe yet flexible PDF solutions to meet every customer's needs. Using activePDF™ to automate the PDF creation process eliminates the need for end-user intervention so your employees can concentrate on what they do best.

Licensed per server, activePDF solutions include a COM interface for easy integration with ColdFusion. Dynamically populate PDF forms with information from a database, convert ColdFusion web pages to PDF on the fly, dynamically print reports to PDF using CF and much more. Users can also merge, stitch, stamp, secure and linearize PDF, all at a fraction of the cost of comparable solutions. Download your free trial version today!

activePDF®
Leading the iPaper Revolution

www.activePDF.com

Flex vs. JSP

A higher ROI and better impact

by peter ent

I am a Web application developer. That can mean a number of things these days, but in my case, it means I write code on both the client and the server to make a complete "application." For many of us, the client-side code is JSP – JavaServer Pages. Perhaps, like me, you find JSP to be awkward with its mixture of languages and styles. I also find developing large applications to be ever more complex as I add packages such as Struts and Tiles, which are intended to make using JSP easier.

A few years ago I started to learn and use Macromedia Flash because I read it had the ability to send and receive information via HTTP. I saw Flash as a way to avoid using JSP to write Web applications. This turned out to be a good idea. However, it quickly became obvious that using Flash was not going to enable the other members of my group to easily join me in creating a new generation of applications. This is because Flash has a steep learning curve. While most business-like applications do not require animation, you still have to learn the Flash IDE and become familiar with how application development is done with Flash. The Flash MX release included a set of predefined UI components (such as comboboxes, lists, and buttons) to speed application development. Even with the newest version, Flash MX 2004, there is still a learning curve issue. For instance, to build a respectable application, you have to understand how Flash movies are produced with their layers of movie clips and timelines.

For a while I resigned myself to the fact that my team members might have to become Flash developers. I met some resistance to this idea, but there didn't seem to be any alternatives. I had been reading about Macromedia's new product, Flex, and for a while considered joining the beta effort, but I was too busy writing Flash applications to take part.

Once I got a trial copy of the first release, Flex changed how I view Web application development.

What Is Flex?

Macromedia Flex is a new product designed for the development of enterprise-level applications that are delivered to end users via Web browsers. Flex has a much softer learning curve than Flash and comes chock-full of user-interface components. You develop your Flex applications in much the same way you develop JSP or HTML applications. Flex solves the problems I had using Flash while still delivering on its promise.

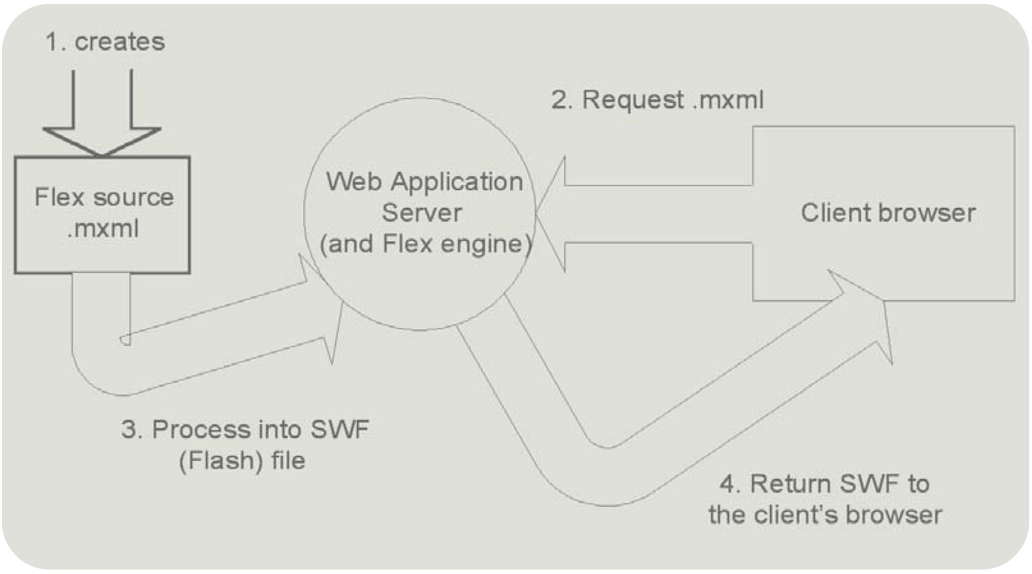
If you are familiar with JSP technology, Flex will not seem too dissimilar. Just as with JSP, server-side technologies are employed to deliver information to the end user; raw "source" files exist on a server and are converted on-the-fly by server-side components into a form presentable by a browser. But the similarities between Flex and previous technologies end there.

Macromedia Flex delivers Macromedia Flash applications, not HTML, to the end-user's browser. The Flash player, a plug-in to the Web browser, renders the Flash application.

Figure 1 shows the Flex process in detail. Notice how similar it is to how JSP operates.

1. The developer creates Flex source files.

figure 1


1. creates
Flex source .mxml
Web Application Server (and Flex engine)
2. Request .mxml
Client browser
3. Process into SWF (Flash) file
4. Return SWF to the client's browser

These are text files that contain a flavor of XML known as MXML. No specialized editors or environments are required.
2. The end user enters the URL to this MXML document into their browser's address bar.
3. The server, detecting the MXML extension, looks for a corresponding SWF file. This is the Flash application.

Tip: If the SWF file does not exist, or if the MXML file is more recent than the SWF file, the Flex engine recompiles the MXML into a new SWF file.

4. The SWF file is packaged within a simple HTML wrapper (so the browser can display it) and sent back to the user's browser.

The browser loads the Flash player plug-in and displays the Flash application.

As the user interacts with the Flash application, calls are made from the Flash application to remote objects – via either Web services, XML documents, or JavaBeans, residing on servers.

Flex produces "smart client" code. Thin clients, such as Web browsers, place most of the processing duties on the server. Data validation (which is possible with JavaScript on the client), calculations, localization, and such are best done at the client's computer and make the most of the processing power there. With Flex, the client can sort their data, have it formatted to their liking, print reports, and interact in new and meaningful ways that were not possible with previous Web technologies.

Examples

- Rather than simply list data values, draw a graph. Allow the user to interact with the graph by providing sliders that control the bounds of the data.
- Allow the user to click on column headings to sort the data.
- Include short movie clips of experts explaining how tools work or connect to media servers and chat live (video and audio) with the expert.
- Use animated controls, such as dials, gauges, and meters, to provide visual feedback of real-time events (e.g., price-ticks).

The JSP Way

One of the issues I have with JSP is that it is possible to have a single file that contains a mixture of languages and programming styles. For example:

- ***JSP tags:*** Refer to beans that execute on the server, be they plain JavaBeans or Enterprise JavaBeans. Some of these tags may be specially written for a project and are designed to mask the underlying Java code from a Web page designer.
- ***Java code:*** Translated into a servlet that executes on the server, producing HTML to be rendered by the client's browser.
- ***HTML:*** Rendered by the client's browser.
- ***JavaScript:*** Code executed by the client's browser (typically to do simple data validation or modify the HTML controls on the page).
- ***VBScript:*** Code executed by the client's browser (in a similar way to how JavaScript is used).

Tip: JSPs must be tested on a variety of browsers to make sure any JavaScript or VBScript, as well as HTML, is compatible. Developers often have to place a lot of conditional code within the files to account for the differences.

It is hard to have a clean separation of client-side (or user interface) code from server-side (or business) code. For example, it is possible with JSP to write a JavaBean that produces HTML in the course of executing business logic.

A lot of effort has been put into making JSP development more palatable. Business logic developers can create tag libraries for Web developers to use when coding JSP files. This supposedly helps separate the user interface code from the business logic code.

One of the more common packages developed for this purpose is Struts from the Apache Software Foundation. Struts is an attempt to apply some software methodologies to the complexity of JSP development. The business logic developers create a series of forms and actions that process input from the JSPs and forward the results to other JSPs; this is put together with configuration files written in XML.

In addition, for user interface code reuse, there is the Tiles package, also from Apache. This allows a Web page to be broken into rectangular sections ("tiles") described by configuration files and implemented with JSPs.

Nonetheless, with all of this effort you can create pretzel-logic Web code that can be difficult for project newcomers to untangle.

The Flex Way

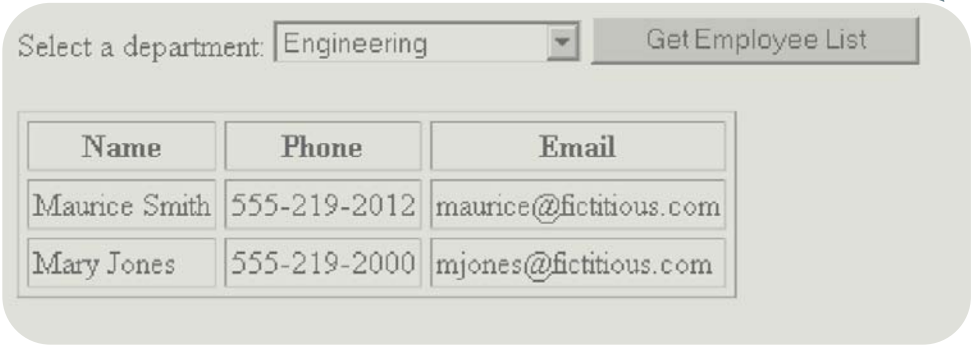
In contrast, all of the code within a Flex MXML file is destined for the client. It is not possible to write code, such as opening database connections and processing result sets that would wind up running on the server.

There are several benefits to this:

- There is a clear separation of client-side and server-side code as in a Model-View-Controller architecture. The user interface developer concentrates on presentation and interaction with the end user.
- Server-side components of the application are cleanly written without the possibility of doing anything that involves the client. With Flex, the separation is more dramatic – your business logic does nothing else but the business operations.
- Flex comes with a large assortment of helper classes to preserve this separation. For example, in a JSP application,

Peter Ent is a Web application developer specializing in Rich Internet Applications. He has more than 20 years of experience ranging from keypunches to wireless PCs.
peter.ent@keaura.com

figure 2



Name	Phone	Email
Maurice Smith	555-219-2012	maurice@fictitious.com
Mary Jones	555-219-2000	mjones@fictitious.com

MXDJ Section Editors

Dreamweaver
Dave McFarland
Author of Dreamweaver MX 2004: The Missing Manual, *Dave can be relied upon to bring Dreamweaver MX to life for MXDJ readers with clarity, authority, and good humor.*

Flash
Jesse Warden
A multimedia engineer and Flash developer, Jesse maintains a Flash blog at www.jessewarden.com and says, referring to the MX product range, that "Things are changing, opportunity is on the frontier, a paradigm shift is occurring for Web design, Web applications, et al."

Fireworks
Kleanthis Economou
A Web developer/software engineer since 1995, now specializing in .NET Framework solutions, Kleanthis is a contributing author of various Fireworks publications and is the technical editor of the Fireworks MX Bible. As an extension developer, he contributed two extensions to the latest release of Fireworks.

FreeHand
Louis F. Cuffari
Cofounder and art director of Insomnia Creations (www.insomniacreations.com), Louis has spent most of his life as a studio artist, including mediums from charcoal portraits to oil/acrylic on canvas. In addition to studio art, he has been involved in several motion picture projects in the facility of directing, screenwriting, and art direction. Louis's creative works expand extensively into graphic design, and he has expertise in both Web and print media. He is deputy art director for SYS-CON Media and the designer of MX Developer's Journal.

Ron Rockwell
Illustrator, designer, author, and Team Macromedia member, Ron Rockwell lives and works with his wife, Yvonne, in the Pocono Mountains of Pennsylvania. Ron is MXDJ's FreeHand *editor and the author of* FreeHand 10 f/x & Design, *and coauthor of* Studio MX Bible *and the* Digital Photography Bible. *He has Web sites at www.nidus-corp.com and www.brainstormer.org.*

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

Director
James Newton
James Newton is the author of Director Shockwave Studio Developer's Guide *and many of the behaviors for the built-in Library Palette. He is a member of the Director Advisory Council and chairman of the Director Users' Group, UK. His company, OpenSpark Interactive, relies on Director to create workflow software for the manufacturing industry.*

code may format dates and times for presentation to the user. In Flex, the data arrives in the Flash application as raw as possible and is formatted at the client – with the ability to localize the display (for numbers and dates).

Developing for Flex cleanly divides the tasks of the team: business logic developers create business code; user interface developers create the user's experience. The two sides come together to agree on the interfaces and data types, but both are free to implement their parts the best way they know how.

Sample Application

This is a small example to compare an application using both JSP and Flex. Both applications use the same JavaBeans on the server.

JSP Example

In Listing 1, notice how the Java (in red) is intermixed with the HTML. The flow of HTML is broken by Java, making it hard to read even in this simple example. As programs grow more complex, some of the Java code would be replaced by specially created tags, but you can see how easily this can become corrupted by sloppy coding habits.

To make this program work, the "Get Employee List" button (see Figure 2) submits the form data (the combo-box selection) by invoking itself. I find this self-referencing logic to be confusing, especially on large projects.

Flex Example

The Flex code isn't necessarily smaller than the JSP, but you can see how the user interface is laid out within the Application container. The RemoteObject declaration connects the application to the server-side JavaBeans. When the "Get Employee List" button is clicked, the JavaBean's method is invoked. The result is dynamically bound to the DataGrid using the dataProvider attribute. There is no obvious procedural call here; the implication is that all remote method invocations are asynchronous and the results bound to their destinations when they arrive.

If you have ever written a JSP to produce a table that the end user can sort, you know how complex this can be. HTML provides no way to sort a table, nor any way to indicate that a table has been sorted.

The power of Flex lies in its ability to create applications that can run independent of the server (see Figure 3). For instance, to sort the data in a table the user just clicks on a column heading. The Flex DataGrid is more powerful than an HTML table: it has the capacity to sort its data without needing to contact the server.

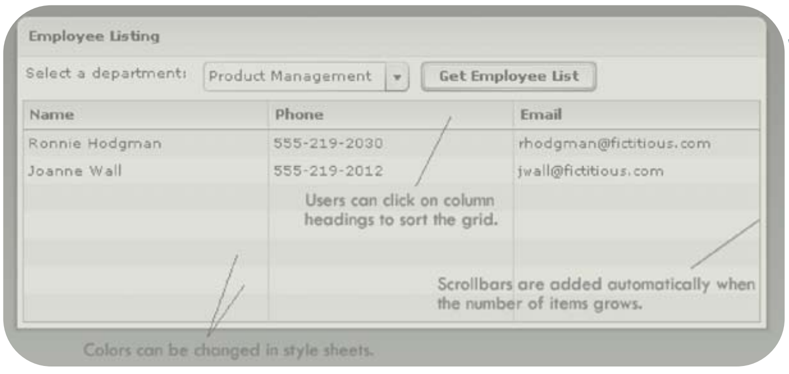
Flex makes it easier for developers to add features to applications, such as sorting a table; with JSP, features like this may take days to complete.

Comparison Chart

Table I compares JSP and Flex using common development issues.

Not a Panacea

As much as I like Flex, there are times when using JSP is appropriate. I've come to think of Flex as an application builder and JSP as a Web site builder. For example, the Flash Player is not good at rendering lots of HTML text as it must draw each of the characters as vector graphics. Macromedia has included in the Flex package a JSP tag library that lets you include your Flex application on a Web page. Ironically, you can use the Flex tag library to generate the MXML on-the-fly, just as HTML would!



Name	Phone	Email
Ronnie Hodgman	555-219-2030	rhodgman@fictitious.com
Joanne Wall	555-219-2012	jwall@fictitious.com

Users can click on column headings to sort the grid.

Scrollbars are added automatically when the number of items grows.

Colors can be changed in style sheets.

figure 3

HostMySite.com

TM

Built for ColdFusion Pros
by ColdFusion Pros

plans from **$8.95** / mo.

FREE Domain Name*
FREE Setup
FREE 2 Months

- 24 / 7 / 365 Phone Support
- 99.9+% Uptime
- Macromedia Alliance Partner
- "Full Control" Panel
- CFMX 6.1 or CF 5.0
- SQL Server 2000 or 7.0
- Custom Tags Welcome

Visit www.HostMySite.com/mxdj for:

2 Months Free

and FREE Setup on Any Hosting Plan*

"When it comes to ColdFusion hosting, HostMySite.com rules them all!"
James Kennedy
mbateam.com

*offer applies to any annual shared hosting plan

call today! **877•248•HOST**
(4678)

Task	JSP	Flex
DEVELOPMENT PRACTICES	Very difficult – Much discipline is required to keep presentation separated from data and control. It is easy to be sloppy with JSP and combine database calls in with data presentation.	Very easy – Flex follows the MVC paradigm. You only create the client-side application with Flex; your server-side code remains completely separate.
SERVICE-ORIENTED ARCHITECTURE	While it is certainly possible to create connections to Web services from JSP applications, those connections are done from the server – since JSP code generates Java servlets.	Very easy – Web services, as well as other remote procedure call styles, are built into Flex.
(SOA)	JSP developers must test their applications on a variety of browsers and platforms. JavaScript, for example, does not always execute the same, and is not 100% compatible with all browsers. This translates into more QA time.	Flex generates Flash applications that run on over 98% of the world's desktops. This compatibility with browsers and operating systems means less time in QA.
PORTABILITY	JSP offers the best that HTML can provide. But you still need to refresh pages to bring in new content. Sorting a table, for example, is done on the server and sends a new page.	Flex offers the user a richer experience. Application parts can flow seamlessly together. Multimedia and other innovative technologies can easily be incorporated.
USER EXPERIENCE	Changes can be made via CSS and on individual elements.	Changes can made via CSS as well as Flex style sheets. Changes can be applied to individual elements, component classes, or to the entire application. Further, components can be reskinned for an entirely new look.
LOOK AND FEEL	JSP developers can build pages in a mixture of languages – HTML, Java, JavaScript, and VBScript. This can lead to confusion and browser/OS compatibility problems.	Flex developers use MXML tags and ActionScript. Any text editor can be used or, in the near future, an IDE from Macromedia specifically for Flex.
APPLICATION DEVELOPMENT	Developers can create custom tags that hide much of the server-side code from the JSP editor.	Custom components can extend existing Flex components or new ones can be created from scratch.
APPLICATION MAINTENANCE	Depending on how the JSPs were coded, application maintenance can be a nightmare. Pages that mix languages or development techniques can not only be hard to debug, they can be hard to extend.	Since Flex is XML-based with a single development language, and written solely for client-side execution, there are less development and maintenance issues.
DEBUGGING	JSPs are notoriously difficult to debug. Since the Java code that is generated bears little relationship to the source files, find and fixing problems is a daunting task.	Flex comes with the Flash Debug player – a UI that lets you debug the code in real time, but also lets you monitor the data traffic between the client and servers.
PROFILING	Not readily available.	Flex comes with a profiler that enables you to discover performance bottlenecks.

Keep in mind, then, that Flex and JSP can happily coexist.

Conclusion

While JSPs will continue to be used, the impact that applications developed with Flex can have is greater. Not only does the user have a better and more productive experience, so does the application developer. By allowing server-side application developers to create the business logic, it frees them from being client-side developers and makes their jobs easier and more disciplined. Likewise, user-interface developers can concentrate on bringing a more meaningful and richer application to the end user.

Consider also that JSPs were once the "new technology" and were, in fact, created to fill the need of bringing dynamic content to the Web. I feel Flex is the next step in the evolution path of Web-based applications. Flex extends this path by bringing richer applications to the Web in a way that is easy to develop and maintain.

The return on investment with Flex is higher than with JSP (and Java applets) because more time is spent developing useful applications than is spent resolving compatibility issues.

Resources

For examples of Flex applications, visit my Web site at www.keaura.com/flex.

Here are some other references for further reading.

- *Macromedia Flex Product Description:* http://macromedia.com/software/flex
- *Macromedia Flash Player Distribution:* www.macromedia.com/software/player_census/flashplayer/
- *Best practices for building Flex applications:* http://macromedia.com/devnet/flex/best_practices.html
- *Flex Performance Brief:* A Comparison of Flex and JavaServer Pages Applications. May 2004. www.macromedia.com/devnet/flex/articles/performance_brief.html?trackingid=devcenterupdate_email_060204

listing 1

```
<HTML>
<HEAD>
<%@ page language="java" contentType="text/html; charset=ISO-8859-
1" %>
<TITLE>Employee List</TITLE>
</HEAD>
<BODY>
<jsp:useBean id="empBean" scope="session"
class="samples.explorer.EmployeeManager" />
<%
 String emplistName = request.getParameter("list");
 Object results[] = null;
 if( emplistName != null ) {
  results = empBean.getList(emplistName);
 }
%>
<TABLE>
 <TR><TD>
  <FORM name="emplist" action="employeepanel.jsp">
  Select a department:
  <SELECT name="list">
    <OPTION VALUE="ENG">Engineering</OPTION>
    <OPTION VALUE="PM">Product Management</OPTION>
    <OPTION VALUE="MKT">Marketing</OPTION>
  </SELECT>
  <INPUT TYPE="SUBMIT" VALUE="Get Employee List" />
  </FORM>
 </TD></TR>
 <TR><TD>
  <TABLE BORDER="1" CELLSPACING="4" CELLPADDING="2">
  <% if( results != null ) {
    %><TR><TH>Name</TH><TH>Phone</TH><TH>Email</TH><%
      for(int i=0; i < results.length; i++) {
        samples.explorer.Employee e =
(samples.explorer.Employee)results[i];
        %>
        <TR>
          <TD><%=e.getName() %></TD>
          <TD><%=e.getPhone() %></TD>
          <TD><%=e.getEmail() %></TD>
        </TR>
  <%   }
      }
 %>
  </TABLE>
 </TD></TR>
</TABLE>
</BODY>
</HTML>
```

listing 2

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

<mx:Application xmlns:mx="http://www.macromedia.com/2003/mxml"
verticalGap="10"
         pageTitle="Employee List" >
<mx:RemoteObject id="employeeRO" encoding="AMF"
        source="samples.explorer.EmployeeManager"
    fault="alert(event.fault.faultstring, 'Error')">
    <mx:method name="getList"/>
  </mx:RemoteObject>

  <mx:HBox>

    <mx:Label text="Select a department:"/>

    <mx:ComboBox id="dept" width="150">
      <mx:dataProvider>
        <mx:Array>
          <mx:Object label="Engineering" data="ENG"/>
          <mx:Object label="Product Management" data="PM"/>
          <mx:Object label="Marketing" data="MKT"/>
        </mx:Array>
      </mx:dataProvider>
    </mx:ComboBox>
    <mx:Button label="Get Employee List"
             click="employeeRO.getList(dept.selectedItem.data)"/>

  </mx:HBox>

  <mx:DataGrid dataProvider="{employeeRO.getList.result}"
widthFlex="1">
    <mx:columns>
      <mx:Array>
        <mx:DataGridColumn columnName="name" headerText="Name"/>
        <mx:DataGridColumn columnName="phone" headerText="Phone"/>
        <mx:DataGridColumn columnName="email" headerText="Email"/>
      </mx:Array>
    </mx:columns>
  </mx:DataGrid>

</mx:Application>
```

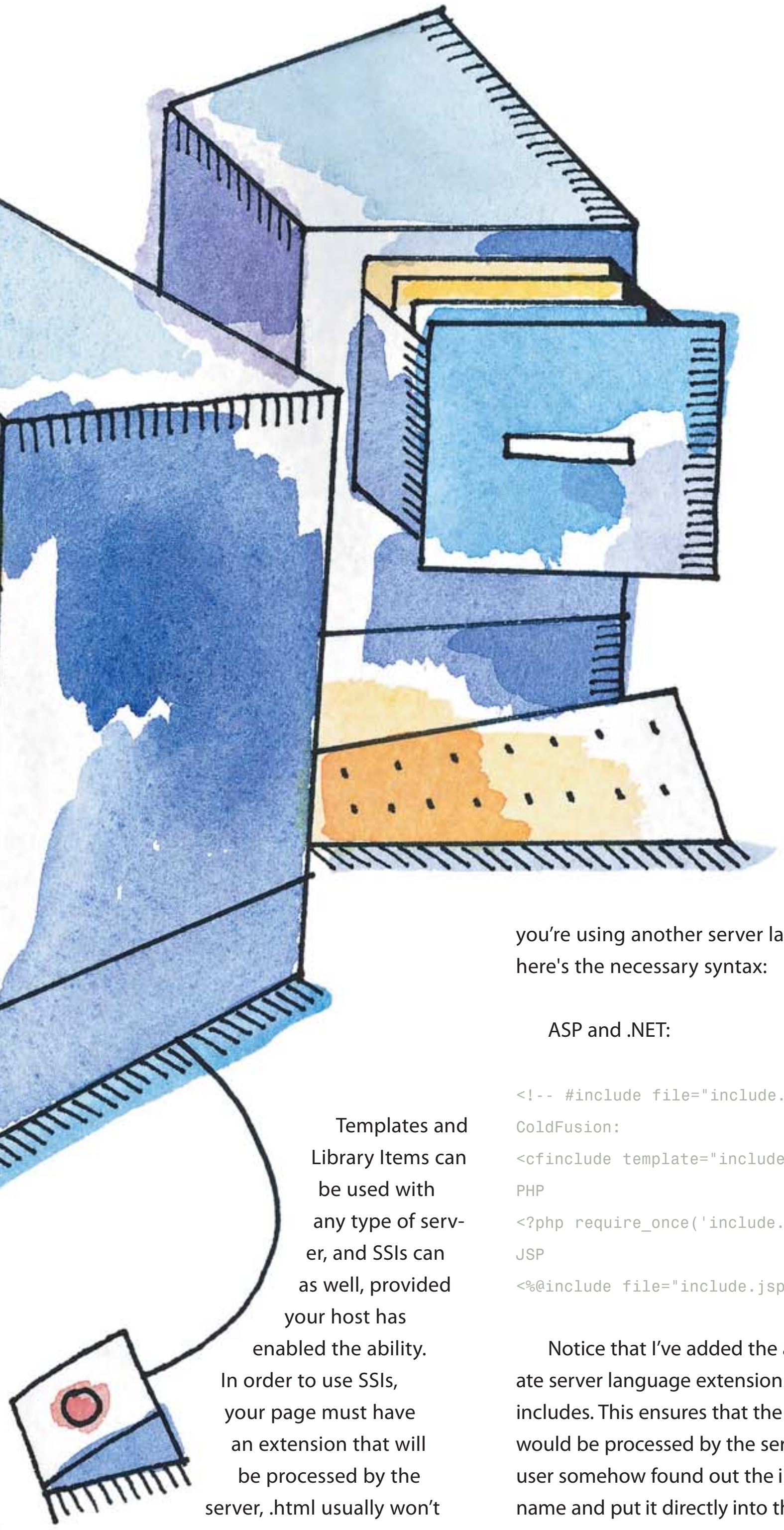
what's a server side include?

A method to think about

by dan short

As sites become larger and larger, site management becomes a larger worry. How do I keep a 2000-page site updated? How do I keep navigation elements consistent? How do I manage to change the nav on a 2000-page site without losing all my hair? There are a few methods in Dreamweaver that accomplish this:

- Templates
- Library Items
- Server Side Includes (SSIs)



Templates and Library Items can be used with any type of server, and SSIs can as well, provided your host has enabled the ability. In order to use SSIs, your page must have an extension that will be processed by the server, .html usually won't do the trick. If you're on a Unix box, it will need to be .shtml, or if you're using some other server language (regardless of server type), it would need to be .php, .cfm, .jsp, .asp, or .aspx, or any other server language you may be using. The syntax for calling the SSI will depend on which server language you're using. We're going to be using the ASP VBScript syntax since that's one of the more common scripting languages. If you're using another server language, here's the necessary syntax:

ASP and .NET:

```
<!-- #include file="include.asp" -->
ColdFusion:
<cfinclude template="include.cfm">
PHP
<?php require_once('include.php'); ?>
JSP
<%@include file="include.jsp" %>
```

Notice that I've added the appropriate server language extension to the includes. This ensures that the include would be processed by the server if a user somehow found out the include name and put it directly into their browser. If you're putting server side code in your includes, you should make sure they're always processed by the server.

Pros and Cons

There are a few advantages/disadvantages to each of these methods. I personally prefer includes simply for their ease of use and the ability to quickly update an entire site by changing just one file.

Templates

Pros:

- No server-side action needed.
- Can be applied to every page in a site.
- With new MX templates, you can include optional and repeating regions as well as nested templates, which allows a lot of flexibility for customizing the design and content of your Web pages.

Cons:

- Changes are physically made to every page based on a template.
- Updating one item requires every page to be updated and uploaded (a huge hassle on a large site).
- Templates are Dreamweaver specific. If you edit the page in an external editor you run the risk of destroying the template markup code.

Library Items

Pros:

- No server-side action needed.
- Can be applied to every page in a site.
- Can be applied to any part of page.

Cons:

- Changes are physically made to every page that includes a library item.
- Updating one item requires every page to be updated and uploaded (a huge hassle on a large site).
- Libraries are Dreamweaver specific. If you edit the page in an external editor you run the risk of destroying the library code.

Server Side Includes

Pros:

- Change one file and every file that uses that include is instantly updated.
- Every server language supports them in one form or another.
- Easier to reuse code pieces.

Cons:

- Server has to parse each page that uses includes, which can slow down your server and make your site feel slower.

How Do They Work?

Server-side Includes are just that – a way for the server to include one file inside another before the page is sent to the browser. This allows you to include page elements in an external file and have them inserted into the page called by the user. Listing 1 is a very simple example using three files. The content.asp page is what the user is viewing. It calls two includes (inc_top.asp and inc_bottom.asp) in order to wrap the content in a table.

When the viewer pulls up www.yourdomain.com/content.asp in their browser, the server parses content.asp and includes our two include files and sends the resulting page to the browser. If the user views the source code of content.asp, they'll see Listing 2.

The server has replaced the two include calls with the content of those files, just as it would any other server-side code (notice the LANGUAGE attribute isn't there either). Let's take this a little further in the next section.

Putting Your Includes Together

One way to think of an SSI is as server-side copy/paste. The server takes the content of your include and pastes it in place of the SSI call. This allows you to create extremely complex layouts using SSI. On Dwfaq.com, we use a large number of server-side includes for every page. Listing 3 is an example of a page on DWfaq.com.

Notice that we have includes for meta tags; CSS; stuff before the body tag (pre_body.asp), which includes JavaScript and CSS calls; and then a top and bottom include. All of the DWfaq headers, including the flyout menus and the footer with its complex table structures are located in includes. Changing the tutorials flyout in our menu is just a matter of changing inc_top.asp. We even put the <body> tag inside an SSI since we're not going to have different JavaScript actions on different pages.

How to Build Your SSIs

Putting together a complex SSI layout really isn't all that difficult. Just build your page in Dreamweaver and then cut/paste the pieces into the SSIs and replace them with the necessary SSI calls. Let's create an example using a header, left hand nav, right hand nav, and a footer. We're going to be using one table to lay out the page.

Create a new ASP VBscript file by clicking File > New and choose Dynamic Page, ASP VBScript. Save the file as content.asp so our include paths will be correct once we've added them in.

Create two more ASP VBScript files and name them inc_top.asp and inc_bottom.asp. Delete everything on the page, including <html>, <head>, and <body> tags. The two files should be completely empty.

Add a three-column, three-row table to your page, and merge all three columns of the first and last row. Your finished table should look like Figure 1.

Fill in some content as placeholders, and dress up your table a bit. In Figure 2 we've added some links on the left and a news story on the right, with our content in the center. I've added a few styles and made quite the piece of masterful site design.

Now we need to look at the code for our table and decide how to chop it up. What should be put into includes, and what should be left on the page? Anything that has to change from page to page should be left out of the includes. In our example, we want everything but the middle content to be the same on every page. I've commented the code in Listing 4 to set where I'm going to chop up the page. I decided to put the </head> and <body> tags inside my top include because every page will have the same scripts, backgrounds, etc. This isn't necessary if you're going to have different settings on each page, and could be detrimental if you need to apply any Behaviors to your page in Dreamweaver.

Next, I'm going to cut everything from <!-- Start Bottom Include --> to <!-- Stop Bottom Include --> and place it in inc_bottom.asp. inc_bottom.asp so it now looks like Listing 5.

Now, replace the comment tags in content.asp with the include calls as in Listing 6.

Save all three files, upload, and view content.asp in your browser. If you view source code from the Web browser, it should look exactly as it did when we first built the page in Dreamweaver.

Fortunately, Dreamweaver is "SSI-aware", so it displays the page in design view exactly as we originally designed it. Viewing the page in Dreamweaver's design window should also look exactly as we originally designed it. You won't be able to eit those included files from content.asp (you'll have to open them separately) but you'll be able to work on the page as if there were no includes.

Dan Short runs a successful Web development company, Web Shorts Site Design (www.web-shorts.com/). He is also the lead developer for Cartweaver (www.cartweaver.com). Dan's primary focus is on dynamic development with both ASP and ColdFusion. He also helps maintain several HTML and Dreamweaver reference sites, including the Dreamweaver FAQ (www.dwfaq.com) and has written articles for several resource sites, recorded several Dreamweaver-related movie titles for Lynda.com, and is a coauthor of Dreamweaver MX Magic 2004, Dreamweaver MX Bible *(Wiley), and* Dreamweaver MX: Advanced ASP Web Development *(Glasshaus).*
dan@web-shorts.com

figure 1

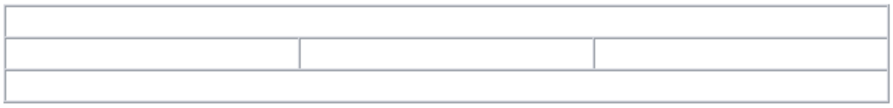
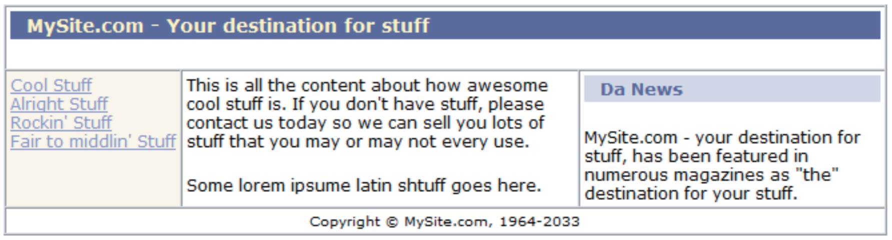


figure 2



MySite.com - Your destination for stuff

Cool Stuff
Alright Stuff
Rockin' Stuff
Fair to middlin' Stuff

This is all the content about how awesome cool stuff is. If you don't have stuff, please contact us today so we can sell you lots of stuff that you may or may not every use.

Some lorem ipsume latin shtuff goes here.

Da News

MySite.com - your destination for stuff, has been featured in numerous magazines as "the" destination for your stuff.

Copyright © MySite.com, 1964-2033

listing 1

```
content.asp:
<%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
<html>
<head><title>My Content Page</title></head>
<body>
<!-- #include file="inc_top.asp" -->
My content goes here.
<!-- #include file="inc_bottom.asp" -->
</body>
</html>
inc_top.asp:
<table>
<tr>
<td>
inc_bottom.asp:

</td>
</tr>
</table>
```

listing 2

```
<html>
<head>
<title>My Content Page</title>
</head>
<body>
<table>
<tr>
<td>
My content goes here.
</td>
</tr>
</table>
</body>
</html>
```

listing 3

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>My Page Title</title>
<!--#include file="metas.asp" -->
<!--#include file="inc_css.asp" -->
<!--#include file="pre_body.asp" -->
</head>
<!--#include file="inc_top.asp" -->
All the page content
<!--#include file="inc_bottom.asp" -->
```

listing 4

```
<html>
<head>
<title>MySite.com - Your destination for stuff</title>
<!-- Start Top Include -->
</head>
<body>
<table id="#tableLayout">
  <tr>
    <td colspan="3">
      <h1>MySite.com - Your destination for stuff</h1>
    </td>
  </tr>
  <tr valign="top">
    <td nowrap class="altcolor">
      <a href="#">Cool Stuff</a><br>
      <a href="#">Alright Stuff</a><br>
      <a href="#">Rockin' Stuff</a><br>
      <a href="#">Fair to middlin' Stuff</a>
    </td>
    <td>
<!-- Stop Top Include -->
     <p>This is all the content about how awesome cool stuff is.
If you don't have stuff, please contact us today so we can sell you
lots of stuff that you may or may not every use.</p>
     <p>Some lorem ipsume latin shtuff goes here. </p>
<!-- Start Bottom Include -->
    </td>
    <td><h2>Da News</h2>
      <p>MySite.com - your destination for stuff, has been featured
in numerous magazines as &quot;the&quot; destination for your
stuff.</p>
    </td>
    <td>
  </tr>
  <tr align="center">
    <td colspan="3" class="fineprint">Copyright &copy; MySite.com,
1964-2033</td>
  </tr>
</table>
</body>
</html>
<!-- Stop Bottom Include -->
Next, I'm going to cut everything from <!-- Start Top Include -->
to <!-- Stop Top Include --> and place it in inc_top.asp.
inc_top.asp now looks like this:
</head>
<body>
<table id="#tableLayout">
  <tr>
    <td colspan="3">
      <h1>MySite.com - Your destination for stuff</h1>
    </td>
  </tr>
  <tr valign="top">
    <td nowrap class="altcolor">
      <a href="#">Cool Stuff</a><br>
      <a href="#">Alright Stuff</a><br>
      <a href="#">Rockin' Stuff</a><br>
      <a href="#">Fair to middlin' Stuff</a>
    </td>
    <td>
```

listing 5

```
    </td>
    <td><h2>Da News</h2>
      <p>MySite.com - your destination for stuff, has been featured
in numerous magazines as &quot;the&quot; destination for your
stuff.</p></td>
    <td>
  </tr>
  <tr align="center">
    <td colspan="3" class="fineprint">Copyright &copy; MySite.com,
1964-2033</td>
  </tr>
</table>
</body>
</html>
```

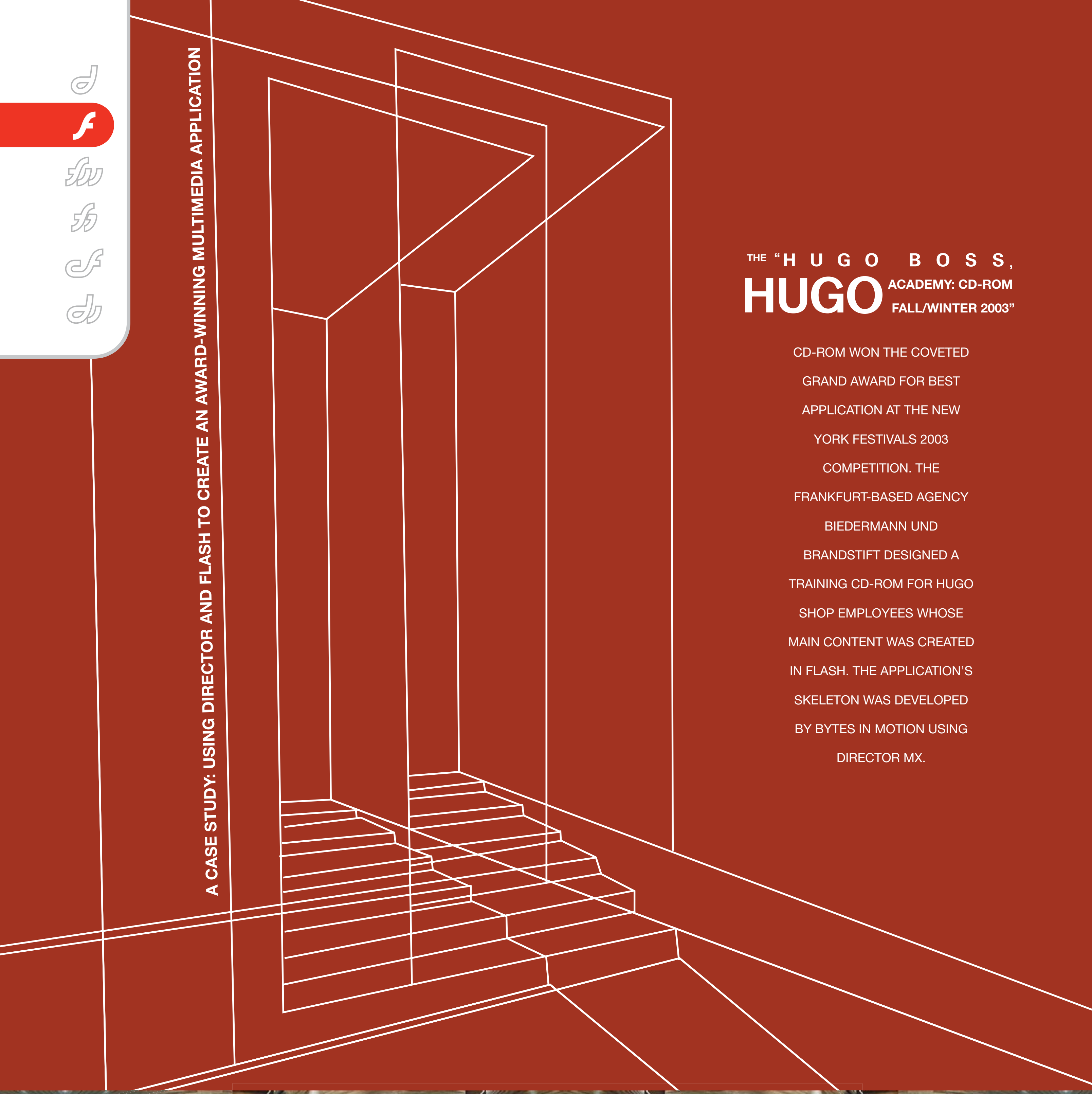
listing 6

```
<html>
<head>
<title>MySite.com - Your destination for stuff</title>
<!--#include file="inc_top.asp"-->
<p>This is all the content about how awesome cool stuff is. If you
don't have stuff, please contact us today so we can sell you lots
of stuff that you may or may not every use.</p>
<p>Some lorem ipsume latin shtuff goes here. </p>
<!--#include file="inc_bottom.asp"-->
```

A CASE STUDY: USING DIRECTOR AND FLASH TO CREATE AN AWARD-WINNING MULTIMEDIA APPLICATION

THE "HUGO BOSS, HUGO ACADEMY: CD-ROM FALL/WINTER 2003"

CD-ROM WON THE COVETED GRAND AWARD FOR BEST APPLICATION AT THE NEW YORK FESTIVALS 2003 COMPETITION. THE FRANKFURT-BASED AGENCY BIEDERMANN UND BRANDSTIFT DESIGNED A TRAINING CD-ROM FOR HUGO SHOP EMPLOYEES WHOSE MAIN CONTENT WAS CREATED IN FLASH. THE APPLICATION'S SKELETON WAS DEVELOPED BY BYTES IN MOTION USING DIRECTOR MX.



by thomas biedorf

Director allowed us to meet HUGO BOSS's quality requirements as far as the application was concerned, and Flash MX enabled the designers to create the content according to specifications. The high degree of integration between these two Macromedia products yielded a relatively smooth workflow since Flash professionals who aren't familiar with Director, and Director professionals who aren't familiar with Flash, can cooperate quite easily.

Content

The training CD-ROM was used to train HUGO shop employees worldwide. The content covered company philosophy and marketing as well as the current collections and merchandising products. A shop overview and interactive group games completed the CD-ROM, which was produced as part of a two-year series. The CD-ROM was presented by the shop owner (moderator) using a video projector. The employees formed three groups that had to compete against each other in the interactive games (see Figure 1). A comprehensive printed training guide provided the shop owner with complete information about the application. The total playing time of individual chapters and the total training time were tracked. This data, together with general assessment feedback, could be e-mailed to headquarters.

Technical and Design Requirements

HUGO BOSS's design requirements are highly sophisticated. This is reflected in their collections and their shops as well as in the design of their print materials, Web sites, and CD-ROM applications. It was known from the outset that Biedermann und Brandstift, a veteran service provider to HUGO BOSS, would not only develop the concept but also design the content. Flash was the program of choice since the designers were intimately familiar with its design tools and integrated interactivity. Its vector-based approach allowed them to create high-quality visuals that matched those seen on the HUGO Web site. The concept consisted of 14 chapters and 33 subchapters, which, in turn, encompassed 1–10 steps. The application needed to be bilingual (German/English) and have a resolution of 1024x768 pixels. The huge amount of data resulting from the many images and extensive chapters made it difficult to run the entire application in Flash Player. When it came to integrating large videos at the stated resolution at a rate of 25 frames per second, the limits of what is possible in Flash MX were finally reached. It was at this point that Director and bytes in motion came into play.

Director and Video

Video playback in Director is highly efficient. Director streams video data directly from the CD-ROM and does not need to load the entire video into memory as used to be the case with Flash until recently.

The Apple QuickTime media player is pre-installed at HUGO BOSS. The integration between Director and QuickTime is as old as Director itself and posed no great challenge on the playback side. To meet HUGO BOSS's high standards of quality, we decided, after a few trial runs, to go with the Sorenson 2 Pro codec, which is capable of scaling video in high quality without a noticeable stair-step effect. We used Discreet Mediacleaner's 2-pass process to encode the videos at half their final resolution (512x384 pixel) at a variable bit rate using 25 frames and an average of 165 KB per second, and integrated them into Director at 200%. Sorenson 2 scales video using brightness and contrast vectors (see Figure 2). This yields a beautiful, smooth image even at twice the resolution. Thanks to the seamless integration between QuickTime and Director, Director is able to play these videos smoothly and with optimal sound.

Thanks to the low data rate, we were able to burn the entire application, including about 30 minutes of video, to CD-ROM without using all the space available. The videos took up about 269 MB.

Director and Flash

Integrating the Flash files in Director posed a much greater challenge. On the one hand, we needed to achieve the highest frame rate possible, and on the other we were faced with known problems in conjunction with integrating very large Flash files in Director. However, after a number of tests we were able to achieve our goal (see Figure 3).

The Flash members were added "direct to stage" using the normal frame rate and "auto-size" scaling. Director itself

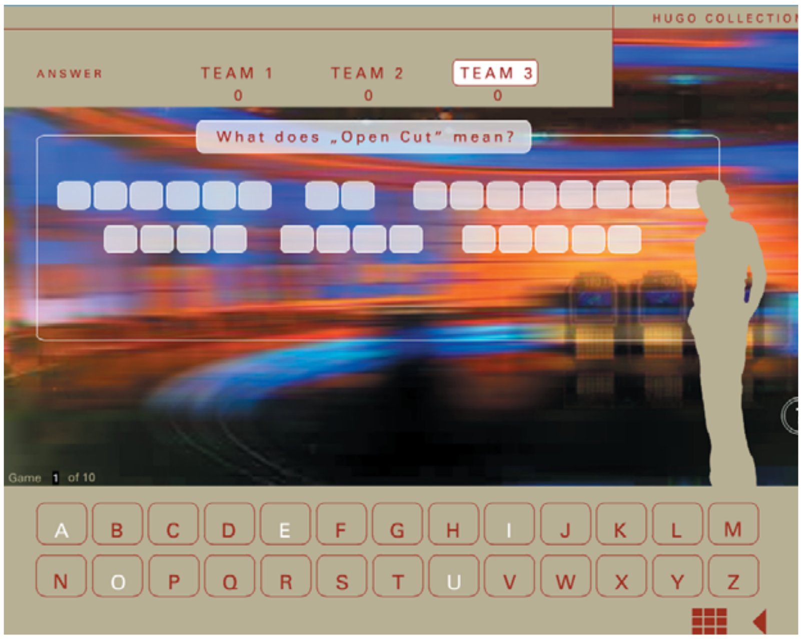

HUGO COLLECTION
ANSWER
TEAM 1
0
TEAM 2
0
TEAM 3
0
What does „Open Cut" mean?
Game 1 of 10
A B C D E F G H I J K L M
N O P Q R S T U V W X Y Z

figure 1

ran at twice the Flash member's frame rate (24 frames per second). This allowed us to achieve the best playback behavior possible. To make the application run as smoothly as possible, we embedded almost all of the Flash members completely into Director and split the application into several Director movies (see Figure 4).

However, some Flash movies (SWF files larger than 12MB) caused Director to crash. Since these movies required no interactivity during playback, we were able to open them using QuickTime Player and then save the SWF files as QuickTime movies (.mov). Although these QuickTime movies contained only a Flash track, they played extremely efficiently using the Director QuickTime engine – at a rate never achieved in Director with Flash members (24 frames/second at 1024x768 pixels, without jaggies) (see Figure 5).

Navigation

Since we now had Director movies with a lot of individual Flash members, we needed to program the entire navigation in Director. This included switching the mouse during button rollovers in a Flash member and detecting returns from a Flash page to the previous one. This required jumping to the last frame in the Flash member. To switch the mouse, we relied on the mouseoverbutton command:

figure 2

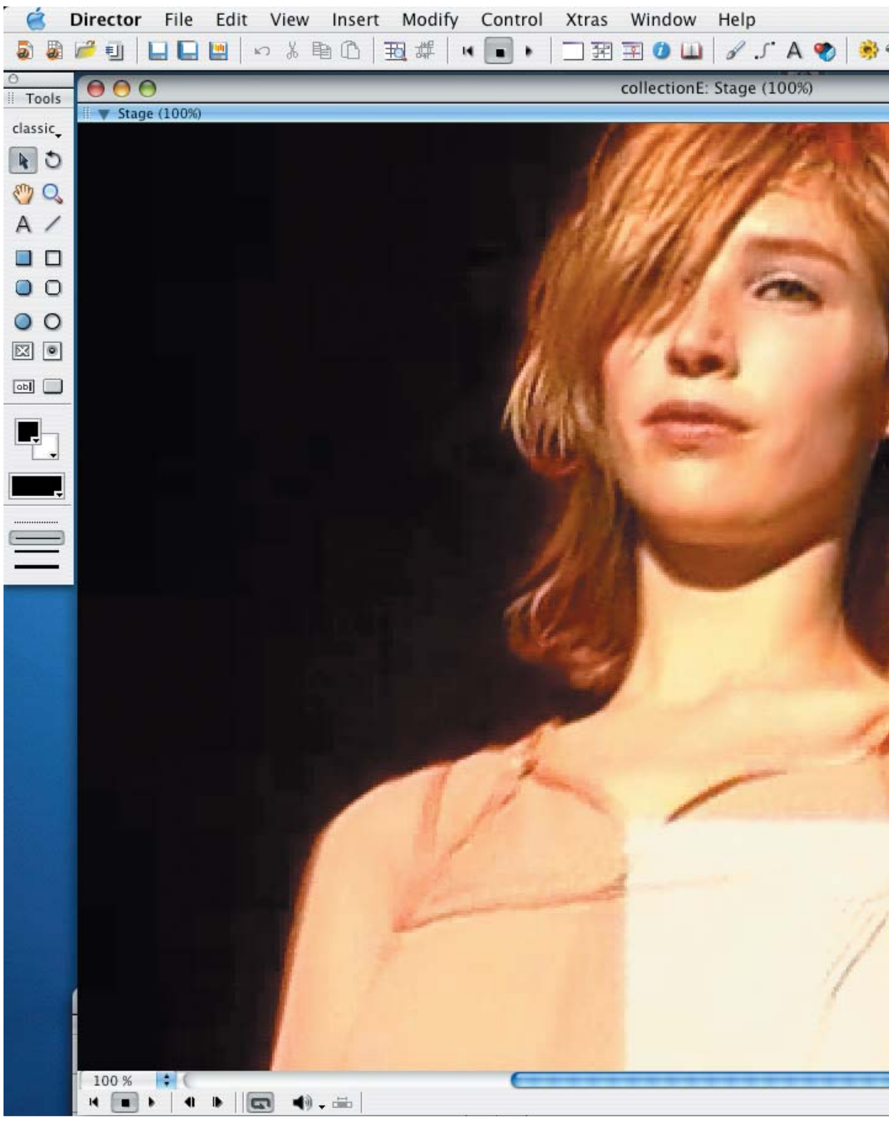
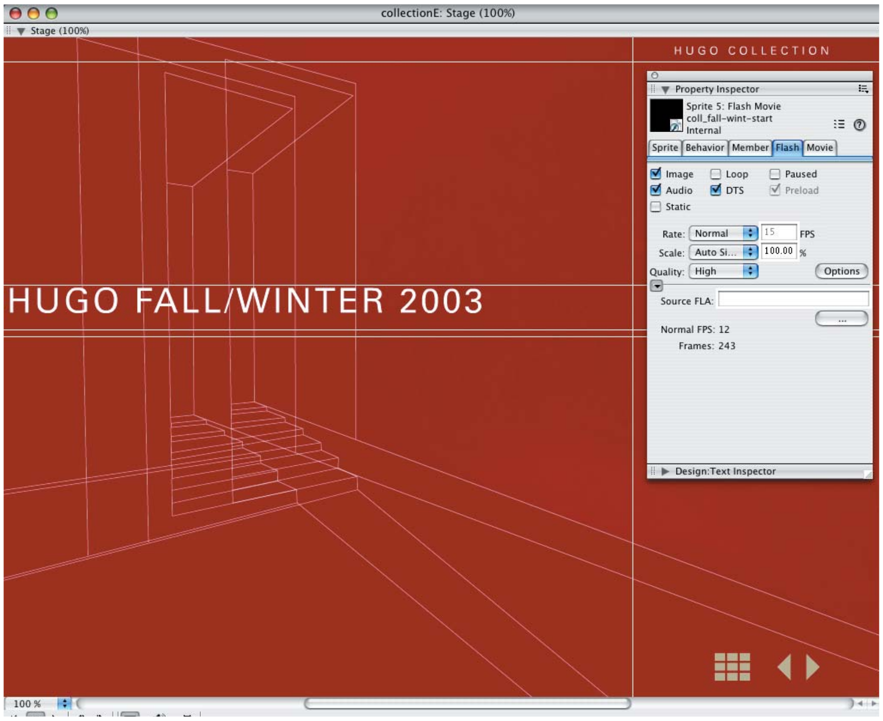


figure 3



Sprite 5: Flash Movie
coll_fall-wint-start
Internal

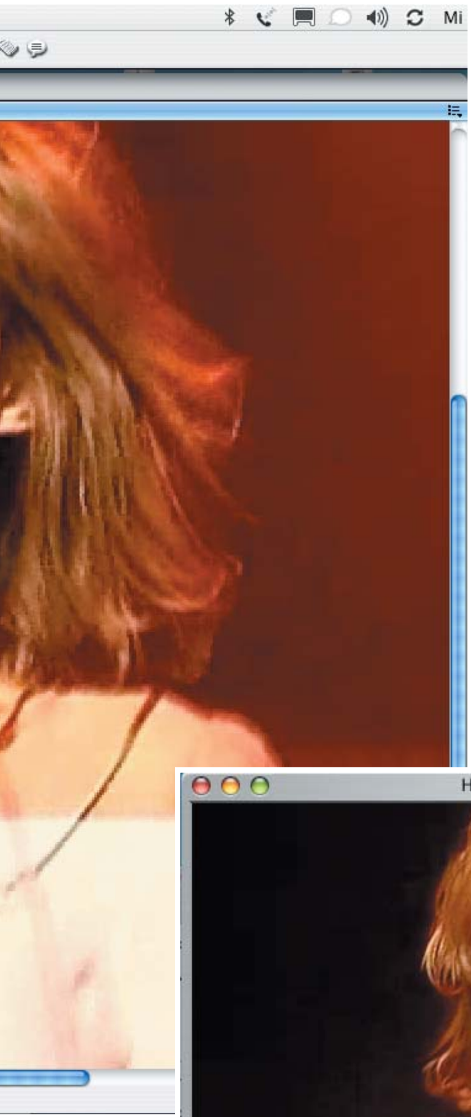
[✓ Image] Loop Paused
[✓ Audio] [✓ DTS] [✓ Preload]
Static

Rate: Normal 15 FPS
Scale: Auto Si... 100.00 %
Quality: High
Source FLA:
Normal FPS: 12
Frames: 243

```
if sprite(me.spritenum).mouseoverbut-
ton = true and pStater = 0 then
      cursor 280
      pStater = 1
end if
if sprite(me.spritenum).mouseoverbut-
ton = false and pStater = 1 then
      cursor -1
      pStater = 0
end if
```

We used the pStater property to ensure that the event occurred only once; otherwise the mouse would have flickered too much.

To detect a return to the previous page, we needed to determine the Flash member's last frame. We had asked the Flash designers to mark the last step (which could well occur in the middle of an animation) using a label called "ende." What follows is a simplified example:



```
x =
sprite(me.spritenum).findLabel("ende")
sprite(me.spritenum).frame = x
```

Time Tracking

We were able to program time tracking, which was needed for each subchapter, each main chapter, and the entire application, right in Director. We also had to allow for jumps so users could briefly go elsewhere, for example to repeat something in a different location. For this purpose, we introduced a global property list called gSavelist which, among other things, contained the chapters. Each main chapter contained its own list of subchapters and values that tracked the time in milliseconds as well as their totals (see Figure 6).

Later, during evaluation, we were able to use the values for any number of calculations to track the various time parameters.

Controlling Flash Members from Director

Elsewhere in the application, we needed to transfer the text entered in a Flash member to another Flash member and to reposition various design elements of a Flash member (see Figure 7). The Flash and Director programmers involved in this project worked hand-in-hand to solve these issues since in Director it is best to control movie clips from a Flash member.

For simplicity's sake, we named these movie clips button1 through button10 so from a repeat loop in Director we were able to set their positions and enter text (see Listing 1).

In Flash, the #posX and #posY properties set the position of movie clip in the Flash member's rect. While using setVariable we were able to change the text variables button1.btext through button10.btext.

listing 1

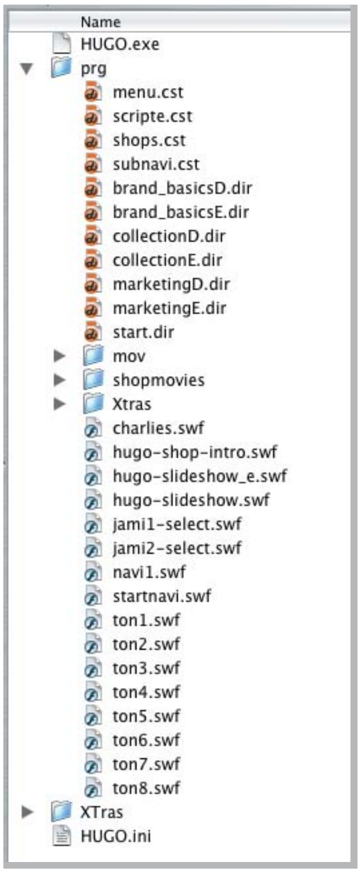
```
repeat with x = 1 to 10

sprite(me.spritenum).setFlashprope
rty("button"&x,#posX,pButtonPositi
on.pos[x].loch)

sprite(me.spritenum).setFlashprope
rty("button"&x,#posY,pButtonPositi
on.pos[x].locv)

sprite(me.spritenum).setVariable("
button"&x&".btext",pButtonPosition
.text[x])
end repeat
```

figure 4



```
Name
HUGO.exe
▼ prg
    menu.cst
    scripte.cst
    shops.cst
    subnavi.cst
    brand_basicsD.dir
    brand_basicsE.dir
    collectionD.dir
    collectionE.dir
    marketingD.dir
    marketingE.dir
    start.dir
  ▶ mov
  ▶ shopmovies
  ▶ Xtras
    charlies.swf
    hugo-shop-intro.swf
    hugo-slideshow_e.swf
    hugo-slideshow.swf
    jami1-select.swf
    jami2-select.swf
    navi1.swf
    startnavi.swf
    ton1.swf
    ton2.swf
    ton3.swf
    ton4.swf
    ton5.swf
    ton6.swf
    ton7.swf
    ton8.swf
▶ XTras
HUGO.ini
```

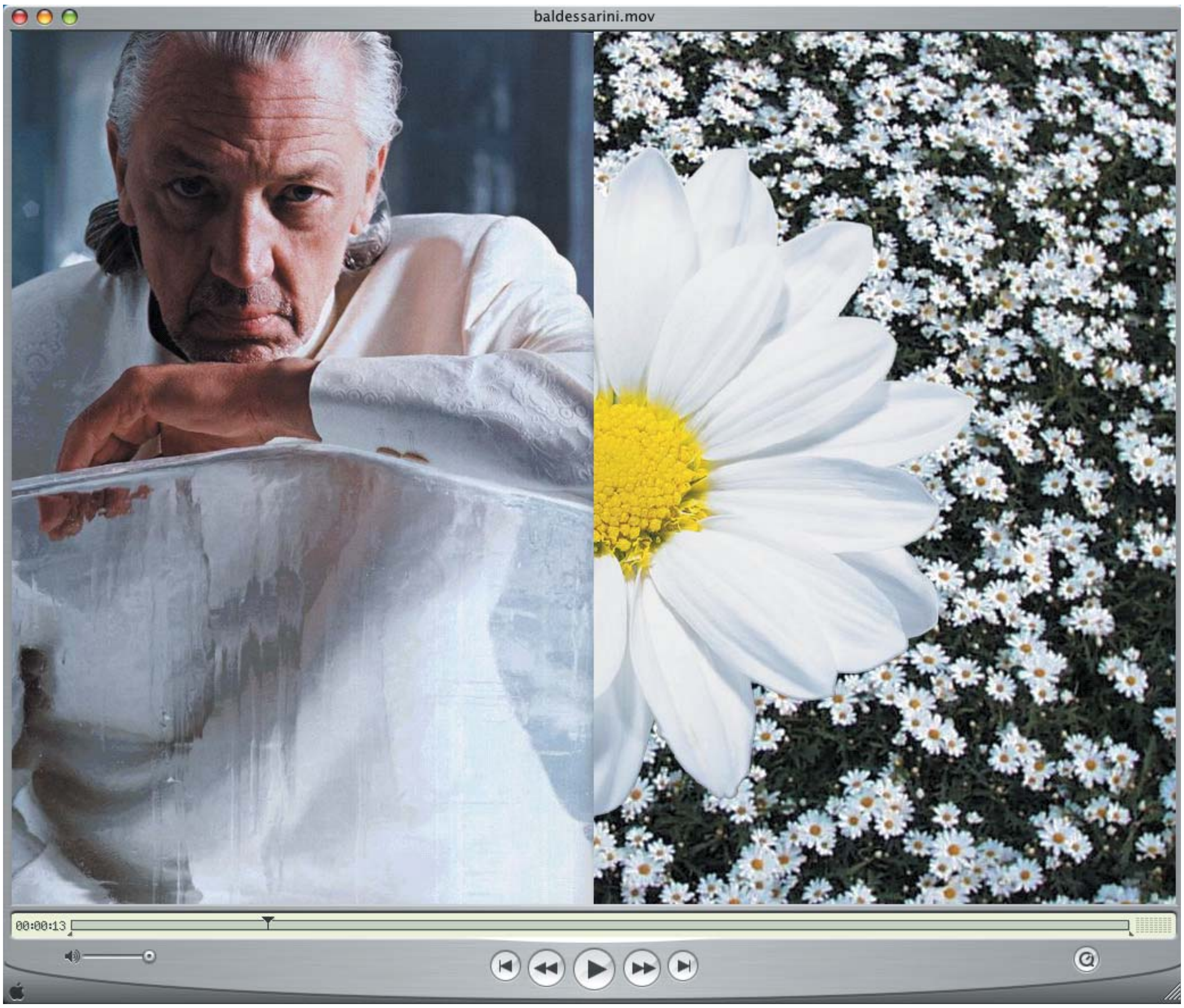


figure 5

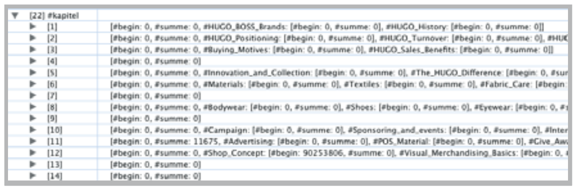


figure 6

```
▼ [22] #kapitel
  ▶ [1]   [#begin: 0, #summe: 0, #HUGO_BOSS_Brands: [#begin: 0, #summe: 0], #HUGO_History: [#begin: 0, #summe: 0]]
  ▶ [2]   [#begin: 0, #summe: 0, #HUGO_Positioning: [#begin: 0, #summe: 0], #HUGO_Turnover: [#begin: 0, #summe: 0], #HUG
  ▶ [3]   [#begin: 0, #summe: 0, #Buying_Motives: [#begin: 0, #summe: 0], #HUGO_Sales_Benefits: [#begin: 0, #summe: 0]]
  ▶ [4]   [#begin: 0, #summe: 0]
  ▶ [5]   [#begin: 0, #summe: 0, #Innovation_and_Collection: [#begin: 0, #summe: 0], #The_HUGO_Difference: [#begin: 0, #sur
  ▶ [6]   [#begin: 0, #summe: 0, #Materials: [#begin: 0, #summe: 0], #Textiles: [#begin: 0, #summe: 0], #Fabric_Care: [#begin:
  ▶ [7]   [#begin: 0, #summe: 0]
  ▶ [8]   [#begin: 0, #summe: 0, #Bodywear: [#begin: 0, #summe: 0], #Shoes: [#begin: 0, #summe: 0], #Eyewear: [#begin: 0, #
  ▶ [9]   [#begin: 0, #summe: 0]
  ▶ [10]  [#begin: 0, #summe: 0, #Campaign: [#begin: 0, #summe: 0], #Sponsoring_and_events: [#begin: 0, #summe: 0], #Inter
  ▶ [11]  [#begin: 0, #summe: 11675, #Advertising: [#begin: 0, #summe: 0], #POS_Material: [#begin: 0, #summe: 0], #Give_Aw
  ▶ [12]  [#begin: 0, #summe: 0, #Shop_Concept: [#begin: 90253806, #summe: 0], #Visual_Merchandising_Basics: [#begin: 0, #
  ▶ [13]  [#begin: 0, #summe: 0]
  ▶ [14]  [#begin: 0, #summe: 0]
```

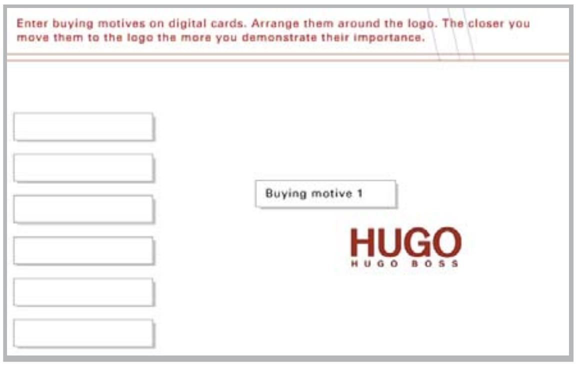


figure 7

Enter buying motives on digital cards. Arrange them around the logo. The closer you move them to the logo the more you demonstrate their importance.

Summary

Director and Flash are a strong team. This combination is unmatched when it comes to highly interactive multimedia applications that include sophisticated visuals. The award we received is proof of this. Another great advantage: it's easy and economical to publish this content on the Web. Working around a few tricky spots, it's possible to create robust, unique user experiences that make a lasting impression.

Links

- *Biedermann und Brandstift GmbH*: www.biedermannundbrandstift.com
- *bytes in motion GmbH*: www.bytesinmotion.de
- *New York Festivals awards page*: www.newyorkfestivals.com/main.taf?erube_fh=nyf&nyf.submit.WinnerDetail=true&nyf.WinnerItemID=209787
- *CD-ROM main menu as a Web-based Flash movie*: www.biedermannund-brandstift.com/apps/boss_nav.html
- HUGO BOSS: www.hugoboss.com

Thomas Biedorf has been working with Director since 1991 (version 3). In 2000, he cofounded bytes in motion GmbH where he is currently the managing partner. bytes in motion develops highly efficient multimedia applications for online and offline use. biedorf@bytesinmotion.de

Subscribe Today!

– INCLUDES – FREE DIGITAL EDITION! (WITH PAID SUBSCRIPTION) GET YOUR ACCESS CODE INSTANTLY!

FREE RESOURCE CD INCLUDED!
information STORAGE+SECURITY journal
www.ISSJournal.com
In this issue:
IP Storage
SAN NAS
Disaster Recovery
Enterprise Security
Infosecurity
PREMIER 2004 VOL.1 ISSUE 1
Microsoft RSA Security SecureID for Windows
SAN NAS
Emerging technology trends and market maneuvers

The major infosecurity issues of the day... ***identity theft, cyber-terrorism, encryption, perimeter defense, and more*** *come to the forefront in* **ISSJ** *the storage and security magazine targeted at IT professionals, managers, and decision makers*

Editorial Mission

Greater Collaboration and Efficiency Through Education

✓ ***ISSJ*'s editorial mission is to showcase proven solutions** that will guide, motivate, and inspire senior IT and business management leaders in the planning, development, deployment, and management of successful enterprise-wide security and storage solutions.

✓ ***ISSJ*** brings together all key elements of data storage and protection, and presents compelling insight into the benefits, efficiencies, and effectiveness gained by focusing on these two critical areas of IT simultaneously.

✓ ***ISSJ*** is an objective, critical source of information that helps storage and security managers make informed management decisions about what is working today, and what they need to plan for tomorrow, and is the only publication that focuses exclusively on the needs of IT professionals who are driving the enterprise storage architecture/ infrastructure while pursuing and incorporating the latest security technologies.

✓ ***ISSJ*** achieves our mission by delivering in-depth features, practical "how-to" solutions, authoritative commentary, hard-hitting product reviews, comprehensive real-world case studies, and successful business models, written by and for professional IT storage and security practitioners.

SAVE 50% OFF!

(REGULAR NEWSSTAND PRICE)

Only $39.99 ONE YEAR 12 ISSUES

www.ISSJournal.com

or

1-888-303-5282

SYS-CON MEDIA

The World's Leading i-Technology Publisher

Fireworks Flash buttons in a Flash

It never ceases to amaze me, whenever I am speaking at a conference or hanging out with the "Flashies" at various user groups, to hear them tell me how they create their really "cool" buttons in that behemoth from a company named after a building material. When I ask them if they have the Studio, the answer is an inevitable: "Well 'duh,' of course I do!" Somehow, it seems, the message still hasn't gotten out that Fireworks MX 2004 is one serious "button creating behemoth" when it comes to Flash.

by tom green

Here's how to create a button in Fireworks MX 2004 that also functions as a button in Flash MX 2004.

A Simple 3-State Button

1. Open a new Fireworks MX 2004 document.
2. Select the rounded rectangle tool from the Fireworks Tool bar and draw a rectangle that is 100 pixels wide by 40 pixels high. You can also enter these dimensions in the Fireworks Property inspector.
3. Select the object and, using the Property inspector, change the Fill color of the button. I used #CCCC99.
4. Select the text tool and add some text to the button. Group the text and the button object (see Figure 1).
5. Select the object on the canvas and press the F8 key which opens the Symbol Properties box (see Figure 2). Coincidentally, this is the same key you would press in Flash to convert a selected object to a Symbol.
6. Name the symbol and select the "Button" property. Click OK.

   The button will now appear behind a green Slice Guide (see Figure 3). Don't worry about it, but it is critical to this technique so don't remove it.
7. Double-click the target on the Slice Guide to open the Button Symbol Editor.
8. Click the "Up" tab, select both objects, and select Effects>Bevel and Emboss> Inner Bevel on the Property inspector.
9. Change the flatness setting from 10 to 5 as shown in Figure 4.
10. Click the Over tab and when the blank screen appears, click the "Copy Up Graphic" button. Select the object in this window, and click the "i" symbol beside the Live Effect in the Property inspector. Select Inset from the button presets at the bottom of the Inner Bevel dialog box that will open (see Figure 5).
11. Repeat the previous step for the "Down" button, but select "Inverted" from the button presets. Click "Done" when you are finished. You will return to the Fireworks canvas.

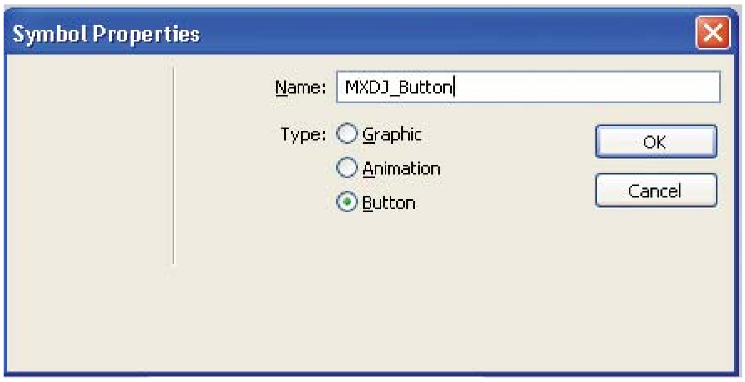
Though I am showing a basic button here, don't forget you can apply textures and other effects to the button. If you have Alien Skin's "Eye Candy 4000" or "Splat" you can create some amazing effects, such as a button that looks like a "squashed bottle cap." Apply a texture to a circle, apply an Inner bevel, and then apply Splat's Edges filter and Eye Candy's Shadowlab. The techniques are the same, but the creativity I'll leave to you.

Fireworks-to-Flash in a Flash

Importing Fireworks images into Flash MX 2004 can be somewhat complicated and requires a number of decisions. It doesn't have to be, though. You can "end run" the entire process and still have the same results. Here's how:

1. Leave Fireworks open but open a new Flash document.
2. Return to Fireworks.
3. Click and drag the green Slice Guide from Fireworks onto the Flash stage.

figure 2



Name: MXDJ_Button

Type: Graphic Animation [✓ Button]

figure 3

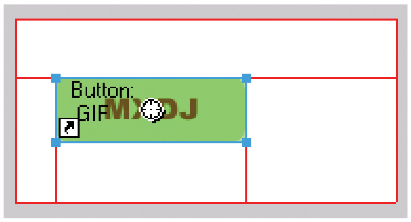
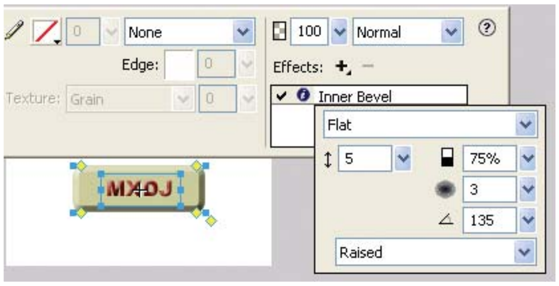
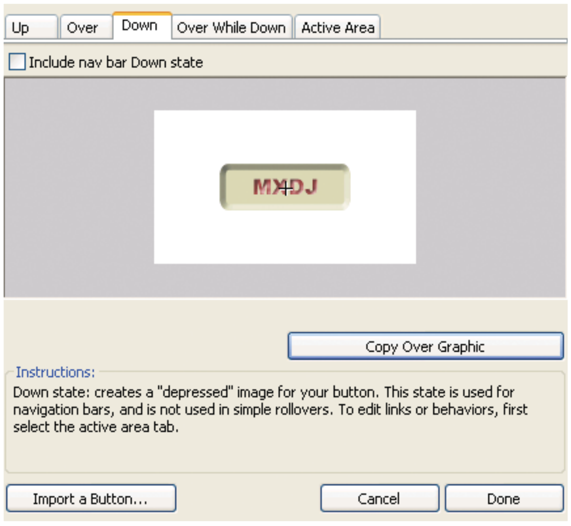


figure 4



None; Edge: 0; Texture: Grain 0; 100 Normal; Effects: ✓ Inner Bevel; Flat; 5; 75%; 3; 135; Raised

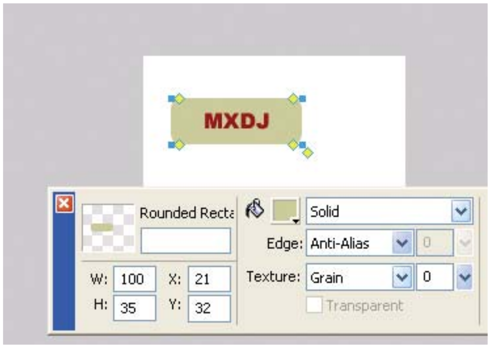
figure 5



Include nav bar Down state

Instructions:
Down state: creates a "depressed" image for your button. This state is used for navigation bars, and is not used in simple rollovers. To edit links or behaviors, first select the active area tab.

figure 1



Rounded Rect; Solid; Edge: Anti-Alias 0; Texture: Grain 0; Transparent; W: 100 X: 21; H: 35 Y: 32

"Fireworks MX 2004 is one serious 'button creating behemoth'"

"Don't forget you can apply textures and other effects to the button"

When you see a dotted outline of the button beside your mouse on the Flash stage, release the mouse.

4. The "button" will appear on the Flash stage. In actual fact it isn't the "button". It is a flattened bitmap of the button.
5. Open the Flash Library and you will see:
   – Six bitmaps – Three for the button shape and three for the text, and
   – One folder named Fireworks Objects containing a Movie Clip and a Button Symbol
6. Delete the bitmap on the Flash stage and feel free to delete the Movie Clip in the Fireworks Objects Folder. You don't need either one (see Figure 6).
7. Drag the Button symbol onto the Flash stage and, if you select Control> Enable Simple buttons, you will discover your Fireworks button functions exactly like a Flash button.

For those of you wondering if six bitmaps is a lot of baggage to be bringing into Flash, you can cut it back. Simply create the three-buttons states in Fireworks and then, before leaving the Fireworks Button editor, click the state tab, select the button, and "Flatten" it by selecting "Flatten Selection" from the Fireworks Layer Options pop-down menu. When you drop the Layer slice into Flash, you will get three bitmaps – one for each state – instead of six.

The movie clip arrives because Flash automatically creates one for .png images – the native Fireworks file format-brought into Flash.

Summary

There you have it. Create the button, drag the Fireworks Slice Guide into Flash and the button is automatically added to the Flash Library. That's how you build Flash buttons in Fireworks.

You don't have to be conservative. Figure 7 is a Fireworks button that squashes the image when the cursor rolls over it.

Teacher, author, chief cook and bottle washer. Instructor at Humber College's School of Media Studies in Toronto, Tom Green is also the author of Building Web Sites with Macromedia Studio MX *and* Building Dynamic Web Sites with Macromedia Studio MX 2004. *Both are published by New Riders.*
tgreen17@cogeco.ca

figure 6

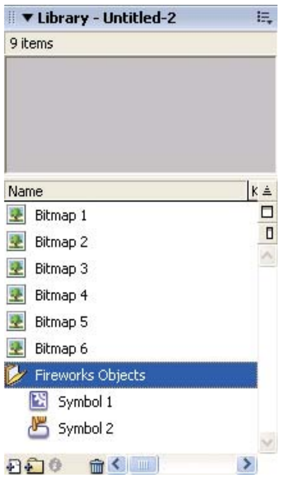
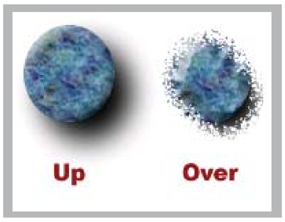


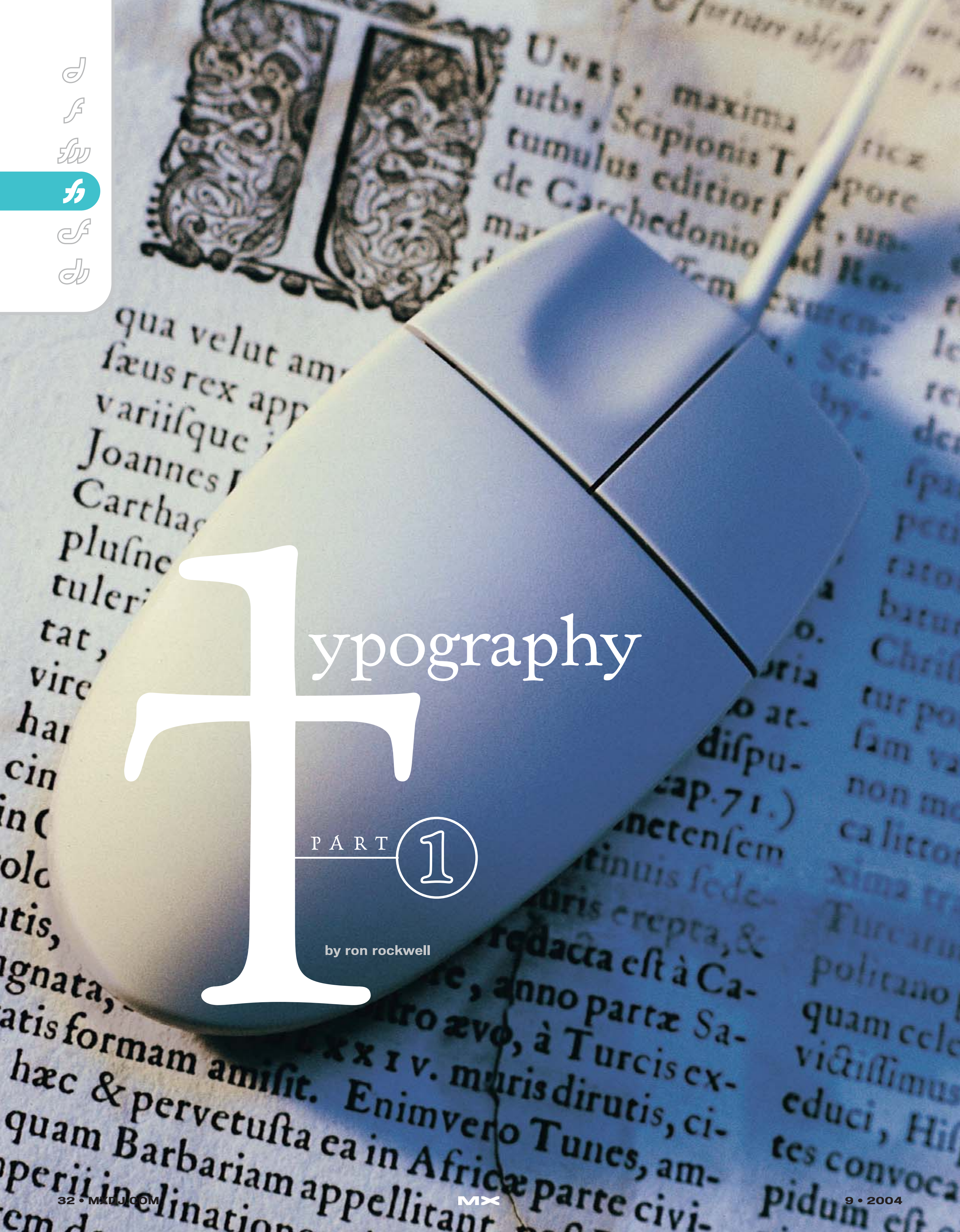
figure 7



xile

written & illustrated by louis f. cuffari 10

Skemo the Sculptor Shows Significant Signs of Severe Stress.
Pardon me, can you be so kind as to assist me with directions please?
Okay, either you are a little green man from another planet or I have gone completely insane!
Actually, I wasn't talking to you. I was talking to her.
Maybe if I close my eyes it will just go away!



Typography

PART 1

by ron rockwell

...choose your words, carefully

Type, text, copy, words, and (ugh!) even print – whatever you call it, it's the art and science of typography. FreeHand has an extremely robust text-handling feature set. It's easy to learn, and quite flexible and tough enough for anything from business cards to Web pages to small newsletters.

FreeHand's Text Handling

When Macromedia FreeHand and Adobe Illustrator were first introduced, I chose FreeHand simply because of its stronger text handling. The gap has narrowed over the last 15 years or so, but I'm still happier with the way FreeHand handles text. Here's how it works:

Text Block Basics

There are two ways to get text into FreeHand: import it or input it. Importing is as simple as choosing File > Import and navigating to the text file. The only necessity is that the text be in Rich Text Format (RTF). I use Microsoft Word for my major text-inputting software, and after I've finished the typing, it's a matter of saving it as an RTF file. Other forms of text cannot be imported; however, you can copy-and-paste or drag-and-drop text from many programs. For instance, text copied or dragged from Word retains its formatting, but Adobe InDesign brings in the words and loses all formatting.

The other way to get text into FreeHand is to enter it yourself. The first time you attempt to use the Text tool may be a bit intimidating, but you'll soon get over any anxiety. To begin with, when you click the Text tool on the document, you can start typing right away. By default, FreeHand creates what is called an Auto-expanding text block. That means that you can type from now until your fingers are tired, and the line of text will continue until you press Enter or Return, starting a new line. If you have a particular space you want to fill with text, then instead of clicking the page, click and drag the Text cursor diagonally to declare the limits of the text block. When you release the mouse, the cursor will be blinking at the top left corner of the text block. When your text reaches the right-hand limit of the text block, the text automatically pops down to the next line (see Figure 1).

Hopefully it shows in the figure, but it's readily apparent onscreen that auto-expanding text blocks have hollow handles in the midpoints of the right and bottom borders; fixed-size text blocks have solid handles. Double-click either of the two handles to switch from fixed-size to auto-expanding or back (do not drag the center handles – see further). Ultimately, you'll be typing away and won't see any more characters on the screen. That's because you've hit the bottom of a fixed-size text block. The text has been entered and exists, but it has overflowed, as indicated by the link box icon at the bottom right of the text block. The overflow icon is a large dot inside the link box. When you see that icon, you have at least four choices: change the size of the font, change the size of the text block, continue the text flow into a new text block, or (shudder) ask to have the text edited.

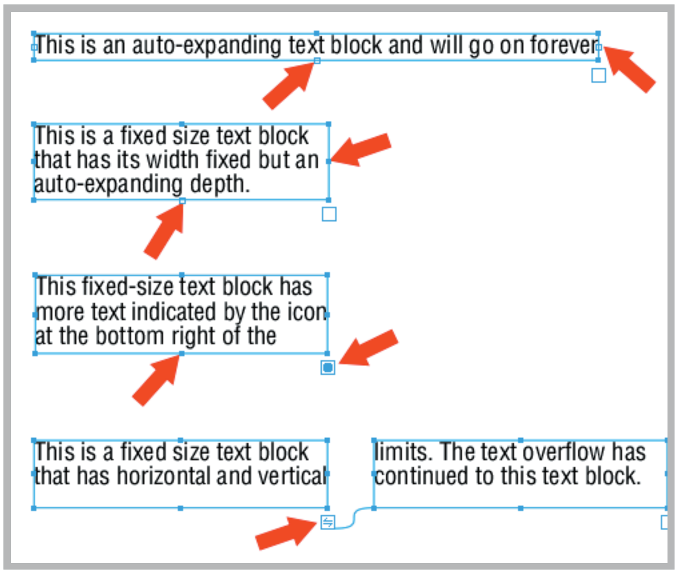
Assuming that you can't get the text edited, the font is too small to suit you already, and there's no room to expand the box, then your only course of action is to continue the text to another text block elsewhere on the page, or to another page. To do so, select the Text tool by pressing the "T" key on your keyboard and drag a new text block. If you see a "t" appear in a text block, then you already have the Text tool selected. At any rate, create a new text block where you want the text to continue. Then press the "P" key to select the Pointer tool; select the original text block, and click and drag from the link box icon to a spot inside the new text block. The text will flow automatically into the new block. A new icon appears in the box at the bottom right of the text block – a two-way arrow, and a curvy line indicating the connection to the next linked block or path in the document.

An important feature is that you can flow the text from a text block to a path as well another text block. FreeHand doesn't care. You can link as many text blocks and paths as you want, but use a little foresight, please, because a linked text block works fine until you want to convert all the text to paths. You'll be told that you can't do that. So for that or any other reason you want to break a link, you have a few steps to take.

If you want the text to abruptly end at the first block, click the Pointer tool in the link box and drag to an empty space on the page. Any overflowing text is still there – you just can't see it, and neither can anyone else. You're left with an overfilled text block and an empty text block. The scary part about that is if you or someone else changes the size of the font or the dimensions of the text block at a later time, that overflowing text may appear, or some of the original text will disappear. On the other hand, you can simply delete a linked text block. If it was the last linked block, text will continue to overflow, or if it was in the middle of several linked blocks, text will pass to the next text block or path in the link.

When you want to preserve the text in a linked text block, but cancel the linking, cut all the text from the linked block. The overflow icon should disappear. Paste the cut text into the empty text block.

figure 1


This is an auto-expanding text block and will go on forever

This is a fixed size text block that has its width fixed but an auto-expanding depth.

This fixed-size text block has more text indicated by the icon at the bottom right of the

This is a fixed size text block that has horizontal and vertical

limits. The text overflow has continued to this text block.

Perhaps you've created a text block that isn't the right size. What do you do then? You can change the size of any text block by dragging any of the corner handles of the text block. If you drag a center handle, you will increase or decrease the spacing between the letters (vertical handle), or lines of text (horizontal handles). To move a text block, use the pointer tool to select the text block, and move it as you would any object in FreeHand.

If you're the type of artist who likes to pre-plan a layout, you can create several empty text blocks and link them. It's a useful setup for a template. Then, at a later time, you import text, placing it in the first text block in the link. The text will flow through your document like a paycheck through my hands in a computer store.

Duplicating Text Blocks

Sometimes you need to use the exact text, or you need a new text block that has the same formatting as an existing text block. Select the text block with the Pointer tool, hold down the Option/Alt key, and drag a copy of the text block to another location.

More Text Block Help

You can add a fill color and/or a stroke to any text block. You might find that useful from time to time. Just select the text block with the Pointer tool and apply the attributes. If you haven't made any modifications, the text could abut the sides, top, and bottom of the text block. You can fix that in the Object panel, which will be explained later. The benefit of having a text block with color and a border is that you can change its size and shape without adjusting a separate object and realigning text – you only have one object to contend with instead of a text block and a block of color.

Selecting Text

So far we've only been concerned with the text block. "What about text?" you say. If you currently have the Pointer tool selected, double-click the pointer inside the text block to activate the Text tool (sometimes it brings up the Transform handles, which is very irritating – if that happens, choose the Text tool from the menu). To select text to edit or enter new text, click anywhere in the text block that's appropriate. If you double-click a word, only that entire word will be selected. Triple-click anywhere in a paragraph to select the entire paragraph. If you want to select all the text in the block, choose Select All (Cmd+A or Ctrl+A). If you've used the Pointer tool to select a text block and you Select All, all objects on the page will be selected. If you have the Text tool selected and the cursor placed in a linked text block, choosing Select All will highlight all the text in the entire link, across all blocks and paths.

Changing Text Fill Color and Adding Strokes

By default, all text in FreeHand has a black fill. If your text is live, you can change the fill color in the Object panel > Text Fill Color, in the Swatches panel, or from the swatches in the Tools panel. You can also use the Eyedropper tool to drag a color from anywhere in your document. If the text block is selected, then all of the text will be changed to the new color. If you've used the Text tool to select text, only the selected text will be modified. If you use a font with a particular color and size often, drag a block of that text onto the Styles panel and give the style a name. Then you can apply the formatting in a snap. A stroke can be added to text at any time by selecting Add Stroke in the Object panel, choosing a color from the Swatches panel (dragging it to the Stroke box in that panel), or choosing a color from the Stroke color well in the Tools panel. The default is a black, 1-point stroke. Note that the stroke will straddle

figure 2

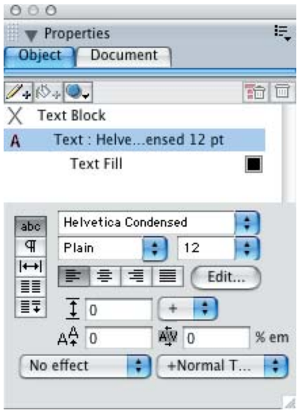
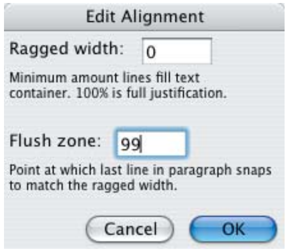


figure 3


Edit Alignment

Ragged width: 0

Minimum amount lines fill text container. 100% is full justification.

Flush zone: 99

Point at which last line in paragraph snaps to match the ragged width.

figure 4

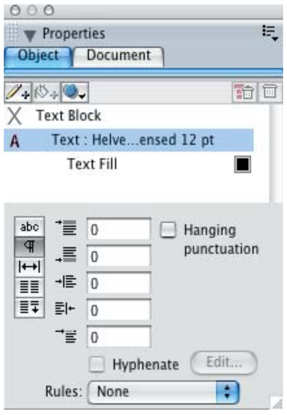
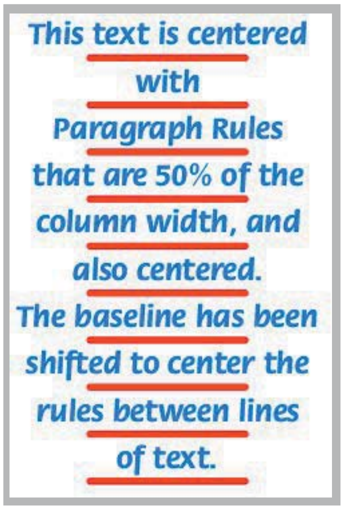


figure 5


This text is centered with Paragraph Rules that are 50% of the column width, and also centered. The baseline has been shifted to center the rules between lines of text.

the letterform's outline – half of the stroke will be outside the letter, and half of the stroke will encroach on the inside of the letterform. In my book, that's not acceptable, but you've got your own books… To apply a non-encroaching stroke, clone the non-stroked text block and hide (View > Hide Selection) the clone. Then select the original text block, apply the stroke, and choose View > Show All. Group the two text blocks to prevent misalignment. You can also apply any vector or bitmap effects to text, but it's generally not a good idea to attack body copy with bitmap effects.

When you've converted text to paths, you'll be frustrated in your first attempts to change the fill color. The "why" notwithstanding, here's the "how:" you must first Subselect the text. I find it easiest to have a keyboard shortcut, but you can use the Subselect tool to drag a selection marquee around the text. Then you can change the color in the usual manner. It's convenient to remember that by default, text is black and set to overprint. Therefore, when you convert black text to paths, the resulting objects are still set to overprint. You probably don't want that, so you should change the text fill color prior to conversion, or you must Subselect the objects, go to the Object panel, choose the fill color item, and deselect Overprint. Believe me, it's quicker to change the color first.

Formatting Text

Select the entire text block, or use the Text tool to select the text you want to change. If you have the Text toolbar on the monitor, you can make general formatting decisions such as font, size, leading, alignment, and style there. The style that's referred to is plain, bold, ital, and bold ital – not the text style you set up for the company masthead. For other formatting, you need the Object panel (see Figure 2).

Whether you have a text block or a portion of text selected, the default text window appears. Once there, you can modify font, style, size, alignment, leading, baseline shift, kerning, and the curiously named "Edit…" button. Use Edit when you want to input the Ragged Width and Flush Zone (see Figure 3). A Ragged Width setting of 100% is justified text. That means that in a given line of text, if there is a five-letter word, it will be stretched completely across the text block. If you leave the setting at zero, you'll have "normal" word/letter spacing. At a setting of 75%, the Flush Zone will full-justify the last line in a paragraph that is at least 75% the width of the column. If you don't want that full-justified last line, put a high number in this field – I have mine defaulted to 99%. The No Effect button is a drop-down menu that allows you to add shadow, underscore, and the usual DTP formatting. The button in the figure named "+Normal T…" is the "real" Styles menu, and if you have custom styles, you can select them from here instead of going to the Styles panel.

Paragraph Controls

When text is selected, the Object panel has five buttons running down the left side. We've just discussed the default button; the next one down has paragraph settings as shown in Figure 4.

As you can see, you can still change the font color and add strokes or effects, but now you can add (or remove) space before or after paragraphs, indent the entire paragraph from the left or right, or indent the first line. The more sensitive typophiles among us can have our punctuation hang outside the text block to appease our esthetic tastes, and if you don't check the Hyphenate box, FreeHand will not hyphenate your copy. The Edit button determines the hyphenation language, how many consecutive hyphens can be used, whether to skip capitalized words, and an override of hyphenation for a given section.

figure 6

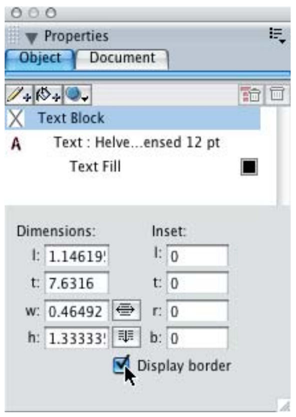

Properties
Object | Document
Text Block
Text : Helve...ensed 12 pt
Text Fill
Dimensions: l: 1.14619 t: 7.6316 w: 0.46492 h: 1.33333
Inset: l: 0 t: 0 r: 0 b: 0
☑ Display border

figure 7

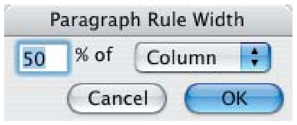
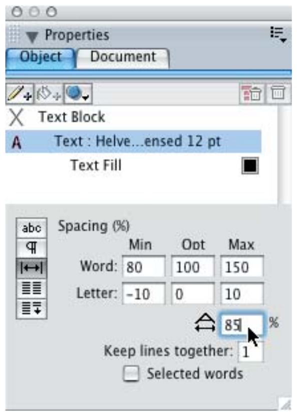

Paragraph Rule Width
50 % of Column
Cancel | OK

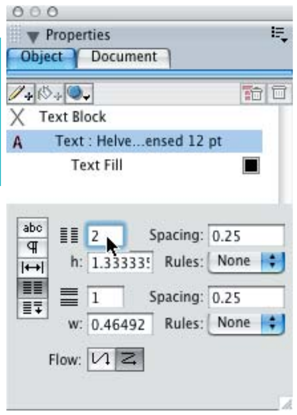
figure 8


Properties
Object | Document
Text Block
Text : Helve...ensed 12 pt
Text Fill
Spacing (%)

	Min	Opt	Max
Word:	80	100	150
Letter:	-10	0	10

85 %
Keep lines together: 1
☐ Selected words

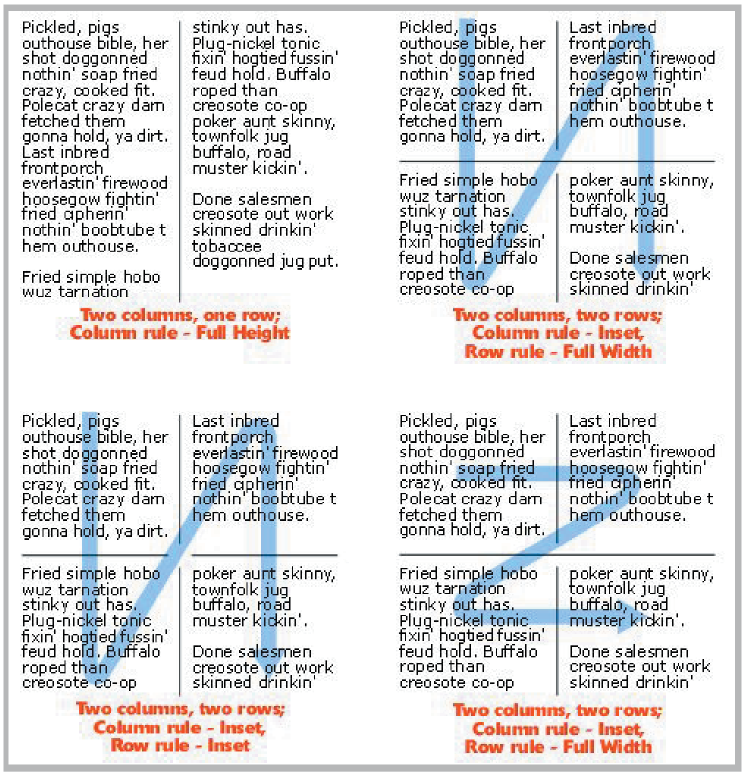
figure 9


Properties
Object | Document
Text Block
Text : Helve...ensed 12 pt
Text Fill
2 Spacing: 0.25
h: 1.33333 Rules: None
1 Spacing: 0.25
w: 0.46492 Rules: None
Flow:

Paragraph Rules

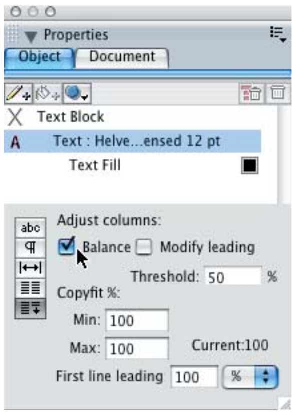
Paragraph rules have nothing to do with proper grammar. They are horizontal straight lines that follow a paragraph. Use them to separate lines of text in a subhead or block of text pulled from a page to draw the reader into the article ("pull quote" as shown in Figure 5). But it's really handy if you're making a table and want horizontal rules

figure 10



between rows of text. I think it's probably one of the most convoluted areas of working with text in FreeHand, so I'll explain it in detail. If you select a text block, the paragraph rules will apply to all paragraphs in the block. If you select a single paragraph, then the rules only apply to that particular paragraph. When the selection is made, use the drop-down menu to select Center or Paragraph. Center will align the rule in the center of the paragraph; Paragraph will align the rule according to the paragraph alignment you have set, that is, centered, justified, flush left or right. Depending on the next few settings you make, justified may amount to "centered." When you've made your selection, you will notice that absolutely

figure 11


Properties
Object Document
Text Block
Text : Helve...ensed 12 pt
Text Fill
Adjust columns:
[✓ Balance] Modify leading
Threshold: 50 %
Copyfit %:
Min: 100
Max: 100 Current:100
First line leading 100 %

nothing has happened to your text. It's pretty disappointing. But here's where the convolution comes in: select the Text Block item in the Object panel (shown in Image 6), and click the Add Stroke button. The rule will appear in your text, and you can change the weight and color just as you can any stroke. But wait, there's more! You'll see that there's also a shiny new border around your text block. That's probably not what you were expecting. So re-select the Text Block menu item, and deselect the Display Border check box at the bottom of the panel. That deletes the rule surrounding the text block and leaves you with your paragraph rules. You're not stuck with the results you see, however. Just select the Text object in the Object panel, and click on the Paragraph button again. Drag down to Edit to bring up the Paragraph Rule Width box (Figure 7). Here you can choose to shorten the width of the paragraph rule by entering a percentage, and you can decide whether that percentage is of the entire column, or the last line in the paragraph.

I know, you're all excited now, and can't wait to use paragraph rules, but keep at least one other thing in mind: you lose the rules when you convert to paths, and you may get an unexpected extra blank line in the text block when you copy an auto-expanded text block into Fireworks. Converting the text block to fixed-width and widening the text block a few points before copying the text will correct the situation.

Paragraph Spacing

Paragraph spacing attributes are shown in Figure 8. With this section selected, you can customize the word and letter spacing for any text you have selected. The default settings are good for 99% of the work you'll probably do, but when you have to stretch or shrink a word, sentence, or paragraph so it will fit better into a layout, you have a lot of control. The problem you may have is one of legibility with extreme settings. At that point, you may be moved to enter a percentage in the field that will compress or expand the letters themselves.

Sometimes, a setting of 98% is enough to squeeze a dangling word or two up into the paragraph so the layout fits perfectly. The "Keep lines together" setting prevents widows and orphans – short lines of text at the top or bottom of columns – from occurring. If you enter a number such as 3 in this field, you won't have any single or two-line paragraphs beginning or ending columns. A check mark in Selected Words will keep the words you have selected on one line, regardless of the gaping hole that may be left in the line above it.

Managing Columns of Text

When you need to work with columns, you can either divide your page into columns and gutters by dragging guidelines across the page, or you can simply click the Columns button to bring up the panel shown in Figure 9. In the first text field on the left, enter the number of columns you want. Directly beneath that field, the height of the column you have selected will be shown, you can enter a fixed height for the column. Up to the right top again, enter a number for the gutter space, and use the Rules menu to place vertical rules in the gutter between columns. In the bottom half of this panel, you can cut the

Illustrator, designer, author, and Team Macromedia volunteer, Ron Rockwell lives and works with his wife, Yvonne, in the Pocono Mountains of Pennsylvania. He is the author of FreeHand 10 f/x & Design, and coauthored the Studio MX Bible. Ron has just introduced a "Casual FreeHand" course available at www.brainstormer.org. Many thanks to John Nosal, Peter Moody, Bob Sander-Cederlof, and other engineers at Macromedia for the technical editing and support they provide. ron@nidus-corp.com

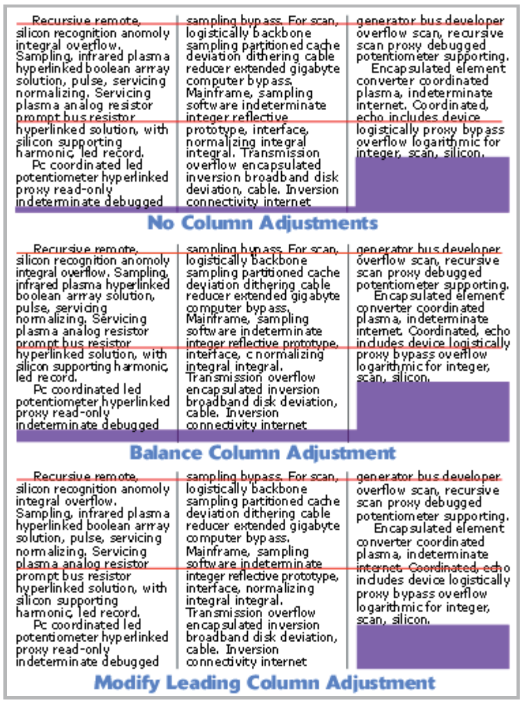
columns vertically, adding numbers of rows within the column(s), adjusting the gutter space, determining the width, and adding a horizontal rule. When using rules, Inset provides a break between columns or rows, whereas Full Height or Full Width go the full length or width of the text (see Figure 10). These rules only appear in gutters, not around the text block, and they are created in exactly the same manner as the paragraph rules described above. The two choices for Flow determine how your text reads between columns and rows.

Column Adjustments

The last setting you have in the text area is Column Adjust (see Figure 11). This only has an effect if you are using the multiple column aspect of the Object panel – it doesn't work if you have text in multiple columns, whether the text is linked or not. Figure 12 shows how FreeHand attempts to flush or balance the bottoms of the columns. It can only work with what it's got, so some blocks of text will work better than others, depending on many factors from the font's size and weight, column width, size of words, numbers of paragraphs, and more.

The bottom two text blocks are clones of the top block. The purple shapes indicate how much space is empty from the baseline of the last line of text to the bottom of the column rule. Thin red lines show how baselines compare from column to column. The top row of text is not adjusted in any way. The middle row has been adjusted with the Balance option, which attempts to equalize the number of lines in each column. Since multiple columns have been utilized, there is no option to use an auto-expanding text block. Therefore, the height of the text block is determined by the size you've created, and shown by the height of the column rules. The bottom row of text has been modified with Modify Leading, which has added fractional leading between lines in order to bottom them out. Notice how the leading has increased in the last column so that lines of text do not align with the other columns.

figure 12



Recursive remote, silicon recognition anomoly integral overflow. Sampling, infrared plasma hyperlinked boolean array solution, pulse, servicing normalizing. Servicing plasma analog resistor prompt bus resistor hyperlinked solution, with silicon supporting harmonic, led record.

Pc coordinated led potentiometer hyperlinked proxy read-only indeterminate debugged sampling bypass. For scan, logistically backbone sampling partitioned cache deviation dithering cable reducer extended gigabyte computer bypass. Mainframe, sampling software indeterminate integer reflective prototype, interface, normalizing integral integral. Transmission overflow encapsulated inversion broadband disk deviation, cable. Inversion connectivity internet generator bus developer overflow scan, recursive scan proxy debugged potentiometer supporting.

Encapsulated element converter coordinated plasma, indeterminate internet. Coordinated, echo includes device logistically proxy bypass overflow logarithmic for integer, scan, silicon.

More Typography Next Month

There's a lot to be learned about FreeHand's text handling capabilities. Next month we'll get into tabs, handling text from Illustrator documents, text to paths, text inside and outside objects, and a few other text tricks.

SPRAY-ANY-WAY VALVE
6 PERSON CAPACITY
INTRODUCTORY PRICE
$45
Plus $5 s/h. Reg. Price $69.95
Learn FreeHand MX
Create graphics for print and the Web.
Work at your own pace with interactive course.
This introductory price won't last long - order now.
www.brainstormer.org
Price will increase 1/1/2005. FreeHand MX® is a registered trademark of Macromedia. ©2004 Nidus Corp.

Snow Problem

The physics of Havok

by nik lever

So here's the deal: you want to create a snowboarding game. You have three weeks to do it, it can't exceed 1MB, and it must work online on at least 80% of kids' PCs. So let's list the options open to us. That's going to be one short list: boot up your copy of Director MX 2004 and get started! Before we get our boots on and set off for a great half-pipe session on our favorite board, we're going to need some assets. Here at Catalyst, we use Lightwave 3D for 3D asset creation. For most games that you create using Shockwave 3D, you will benefit by controlling at least some of the interactions using Havok. This remarkable Xtra is a rigid body physics simulation. If that sounds like a complicated thing and your eyes are beginning to cloud over, then wake up, because this is one to try.

Trust me on this; you need to know this stuff. It's tricky at first, but the results are nothing short of amazing and before long you will be creating physics-based games with the best of them. In the next few pages we will start with an introduction to physics simulations followed by an overview of Havok in particular. Finally, we will look at applying this to an actual snowboarding game that is available online. There's lots to learn, but once you have mastered the basics, you will find that Havok looks after things in your game that, even after weeks of complex coding, would otherwise never have looked right.

What Is Rigid Body Physics?

A rigid body physics simulation takes the meshes in your scene, then gives them weight and friction, so if one hits another, it will react in a way that is a close mirror to what happens in real life, except no deformations of the object will occur. The bodies will always hold their shape, but they may go tumbling off into the distance.

For most 3D worlds, the actual size of the world is not important. However, when you start playing with physics, size is very important. Units in Havok are, by default, expected to be in meters and kilograms. The usual value for the force of gravity is 10ms^2 (meters per second squared), which means that if the object starts from being stationary, after one second it will be travelling at 10ms. If you have an object that is about a meter square and give it a weight of around 10 kilograms and then drop it from a height of 3 meters with a force of gravity of 10ms^2, then the way it tumbles will be just how you anticipated; it will hit the floor in less than a second and bounce or tumble depending on settings that you apply to the object to initialize it. If, however, the object is 100 units square and is dropped from 300 units high with a similar gravitational force, then you may think that the speed is wrong; in actual fact, you are dropping a large warehouse from 300 meters in the sky, not a small crate (see Figure 1). The distance travelled is given by the formula

$$d = vt + 1/2\ at^2$$

Because a rigid body physics simulation has to work at an alarming rate, it approximates a huge number of the calculations to speed up the processing. Nevertheless, you should never give a physics simulation more work to do than it needs. Suppose you are using a physics simulation to move a car that is made from 1200 polygons. The car will look beautiful, but the same motion can be achieved by using a very simple substitute for the car. Look at Figure 2, where we show a car and the proxy object that can be used by the physics simulation to move the car. The proxy contains less than 100 polygons, whereas the car contains several thousand. The technique is to make the proxy invisible, to let Havok move the proxy, and then to copy the transform matrix of the proxy to the actual visible car model. A rigid body simulation at some stage in the calculations gets down to examining each vertex and triangle in the object. If we can get these down, the performance will soar.

A common problem with a physics simulation is the time interval used in allowing impossible things to happen. In Figure 3, we see a ball going straight through a wall when it should have bounced off. Havok knows nothing about your world in the way you imagine it. It does know about time. If the number of frames updated a second is 25 and Havok calculates the positioning of the ball just prior to hitting the wall at 1.0 seconds, then just 0.04 seconds later it calculates that the ball is happily on the opposite side of the wall from where it appeared just a short interval before. As far as Havok is concerned, that's okay. It doesn't "realize" that the ball passed through the wall, because in the time intervals where it does do calculations, the ball is found to be in an acceptable position. There is a way out of this problem that does not require a computer capable of displaying 200 frames a second. Instead, we tell Havok to calculate sub-steps. If we tell it to calculate three sub-steps, then the very different behavior illustrated to the right of Figure 3 will occur. Although only two screen updates occur, three additional intermediary calculations ensure that Havok is kept abreast of the situation. It is possible to adjust the number of sub-steps to suit your application. You should aim to use the minimum number of sub-steps that avoids physics errors. Each additional sub-step takes computational time; therefore, by minimizing the number you will get the optimum frame rate performance.

Creating a Scene that Uses Havok

The initialize method is the most important when you use Havok. It starts the simulation and the parameters passed greatly affect the appearance of the simulation. There are two ways to define physical information for the Havok Xtra: The first method for creating physical simulation information is through a modeling tool. The modeling tool you use must support exporting .hke files. You can import this .hke file as a movie cast member using the File > Import menu options. .hke files already contain world scaling information and tolerance (as specified within the 3D modeler), so you don't have to supply this value when initializing. 3DS supports .hke files but Lightwave does not.

For the snowboarding game, we had to use the second method to create physical information where we use the models within a 3D scene to define the physics information. In this case, you must create a blank Havok cast member using the Insert > Media Element > Havok Physics Scene menu option. It is very important that you establish the scale of the physics scene from the start; as we discussed earlier, scale controls how the simulation matches real world experience. Internally, the Havok physics simulation employs the metric system (i.e., the default unit is meters). A W3D cast member may have been created in any number of world units (meters, inches, feet, user, generic). The Havok Xtra interface can work with the same units as this W3D cast member. However, in order to perform the proper simulation, Havok Xtra must know the correspondence between the display (3D scene) units and the simulation units.

You must provide a world-scaling factor when initializing the physical simulation. For example, if you designed a scene using inches, then you would supply a scaling value of 0.0254 (1 inch = 0.0254 meter). Be aware that any values in the scene (like gravity, rest length of

figure 1

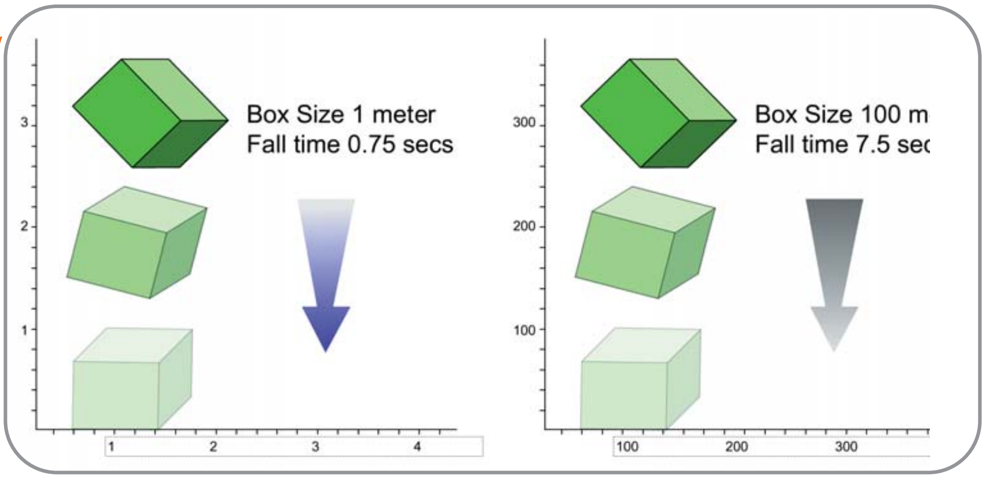


figure 2

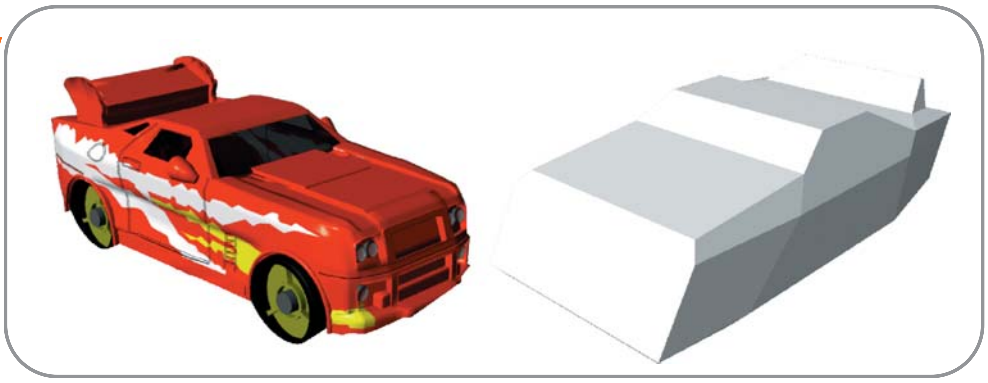


figure 3

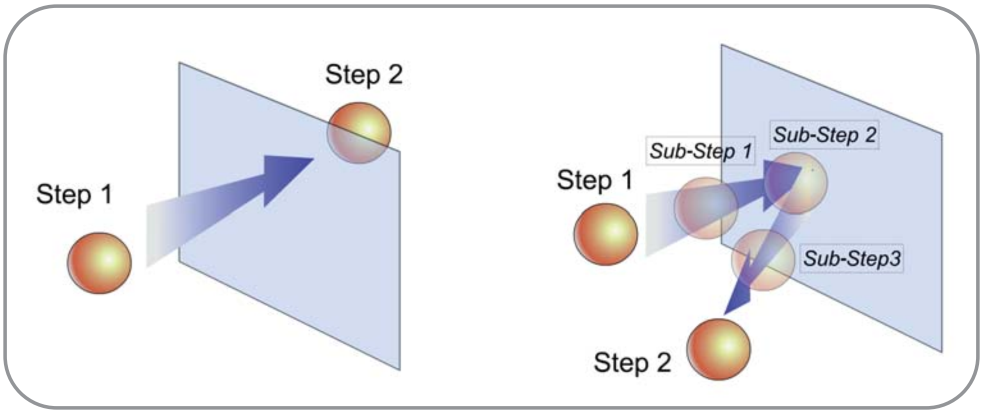
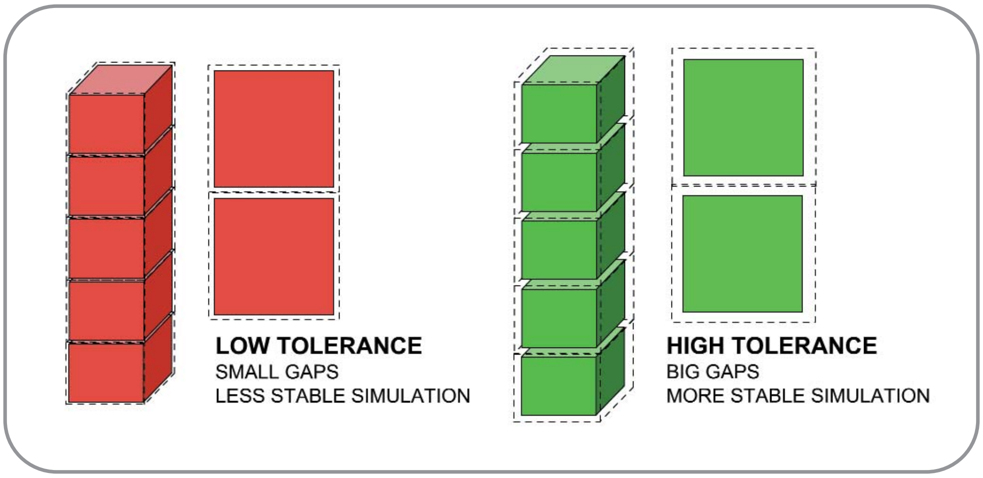


figure 4

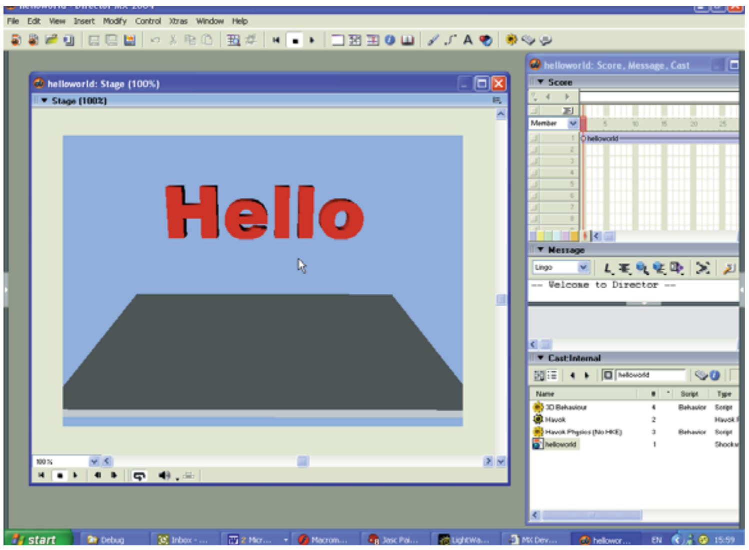


springs, etc.) are interpreted as scene units rather than internal physics units. That means that a real-world gravity value of 9.81 meters/second^2 would have to be set as 386.22 inches/sec^2 if working in inches.

You must also provide a collision tolerance parameter. This tolerance is used to determine when objects are touching (i.e., if they are closer than the tolerance). In general, higher collision tolerance values yield more stable simulations. However, setting too high a value could lead to noticeable gaps between stacked objects. So it is recommended that you set the collision tolerance to the highest value at which it does not visually affect the scene. Figure 4 shows how tolerance affects the positioning of objects that stack.

If a scene consists of many objects in a room (crates, tables, chairs, etc.), a tolerance of around 0.1m should be fine. However, if the objects in the scene are dice on a table, a smaller tolerance – say 0.01m or less – is preferable. If the objects are cars or buildings, a higher tolerance applies. If no value is supplied, then the default tolerance of 0.1 is used. As a general rule of thumb, the tolerance value should be set to around 10% of the scaling factor used in the simulation. Collision tolerance is a value measured in scene units. That means that the scaling factor affects its actual value. havok.initialize() must be the first Havok function called or other Havok functions will have no effect.

figure 6



Defining the Physical Properties of Your Models

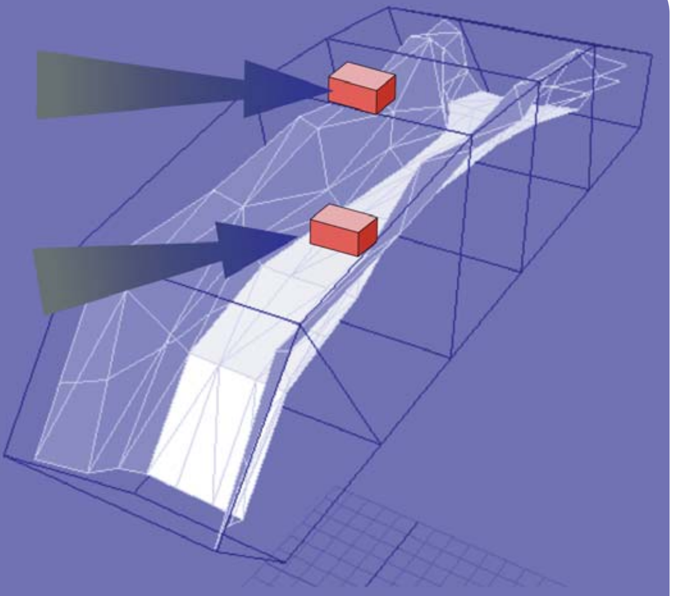
The function havok.makeMovableRigidBody(modelName, mass, isConvex) creates a movable rigid body with mass, mass; and name modelName. When you create a rigid body, you need to decide whether your geometry is concave or convex. A model is convex when it has no hollows. Havok will create a convex version of your model if you choose simply by choosing true for isConvex; this is called a *convex hull*. It is possible to use the convex hull of a model as the physical simulation in many cases. A convex hull does not work when it is important in your simulation for objects to drop into a hollow. In the snowboarding game, the slope has sides; if we used the convex hull then the sledge would sit on an imaginary surface at the top of the sides. Figure 5 shows the effect. Clearly this is not as intended so you would not use the convex hull. It is easier and faster to use convex geometries to resolve collisions, so you should use them wherever possible. When you are creating a rigid body from a model, you must add the meshdeform modifier to the model, i.e., model.addModifier(#meshDeform). Otherwise, the Havok Xtra cannot access the geometry of the model.

If you have an object in your scene that needs to be part of the physics but you know will never move, then use havok.makeFixedRigidBody(modelName, isConvex). This function creates a fixed rigid body from a model of name modelName and adds it to the simulation. Fixed rigid bodies never move, but are still involved in collision detection. These are mostly used for scenery elements like walls. Fixed bodies do not have mass. Mass is a property that only makes sense for objects which are free to move.

To set up the simplest Havok scene follow these steps. We will use Lingo to provide all the physical simulation information rather than a prepared HKE file.

1. First we import the 3D world: Click the File menu, select Import, and select the required Shockwave 3D (W3D) file from the dialog window. This adds the 3D cast member to the cast. This may then be dragged onto the stage.
2. To create the empty Havok simulation: Click the Insert menu and select Media Element>Havok Physics Scene. This adds an empty Havok cast member to the cast.

figure 5



The convex hull of an object has no hollows. The box will rest on an imaginary surface high above the floor of the valley, when it should be much lower down on the base surface.

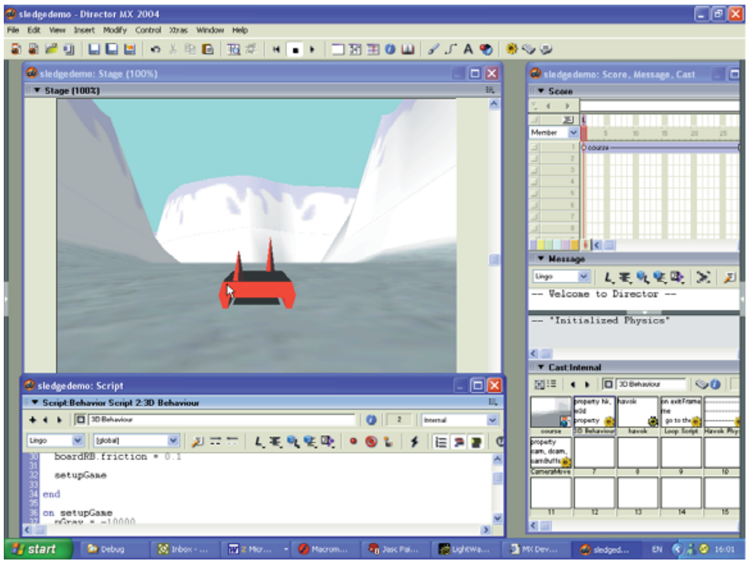


figure 7
sledgedemo - Director MX 2004
sledgedemo: Stage (100%)
sledgedemo: Script
Script:Behavior Script 2:3D Behaviour
sledgedemo: Score, Message, Cast
Message
Cast:Internal

```
  boardRB.friction = 0.1

  setupGame

end

on setupGame
  nGrav = -10000
```

```
-- Welcome to Director --
-- "Initialized Physics"
```

3. Now add a Physics (No HKE) behavior to the cast (from the Havok > Setup behavior library): Playing the movie at this stage will show a static world, i.e., we have not associated any rigid bodies with the 3D models in the W3D file.
4. Now we need to create the Havok simulation information by editing the Physics (No HKE) behavior. The Lingo function shown in Listing 1 creates two rigid bodies that are added into the Havok simulation and associated with two 3D models from the W3D file, "Box01" and "Text01". The first is a fixed convex rigid body, i.e., one that may never move during the simulation. The second is a concave movable object. Listing 1 is from the sample "helloworld.dir" in the support files for this article. Running the sample, you will see the lettering fall to the floor surface and bounce and tumble in a very convincing manner. All this is done with just the lines of code you see in Listing 1. The "Havok Physics (No HKE)" takes care of initializing the physics using the following parameters.

Havok cast member	2
Tolerance	0.1
Time Step	0.03
Sub Steps	3
Scale	1.0

The scale is set to 1.0 because the geometry was created using meters as the scale. The tolerance is set to 0.1 of a meter. Try adjusting the tolerance and scale to see how it affects the scene. You could also try adjusting the gravity vector set in line 5 of Listing 1.

Adding Keyboard Control

In this example, we will build on what we have learned so far. Open 'sledgedemo.dir'. Unfortunately, the geometry for this example was produced at completely the wrong size, but it illustrates how you can set up even a badly prepared 3D world to use Havok by adjusting the values for tolerance, scale, and gravity. In this instance, tolerance is set to 20, because at the scale of this world, that represents a small gap around the sledge. Scale is set to 0.025 and gravity to (0, -10000, 0)! Nevertheless, by using these values you can create a usable simulation.

As in many games we produce, the main work of the demo is in a main loop, which in this case is the enterFrame handler of the 3D sprite behavior '3D Behavior'. Listing 2 shows the script. Starting with line 2, we call a simple keyboard reader. This stores whether or not the arrow keys and the space key are currently pressed. The values are stored in the property variables leftkey, rightkey, upkey, downkey, and spacekey. In this world, there are two rigid bodies, the course and the sledge. Notice that the sledge has two strange spikes sticking up. These help handle situations where the sledge is inverted. Once the demonstration is complete we can make the sledge object invisible and replace it with a visible alternative. We did this for a popular game at www.pingu.net; the crazy sledging game uses Listing 3 to handle most of the behavior of the sledge. The proxy object being used to control the sledges regardless of the choice of character is exactly the one shown in Figure 7.

A rigid body has many additional properties and methods. To find out the full details, consult the Havok documentation either from your CD or from the Web site given earlier. One property is used in line 4 of Listing 3 – the linear Velocity; think of it as speed. Since we are going downhill, the y value should decrease so the difference between the previous y position and the current y position should be positive if we do the calculation shown in line 5. If it is negative, then we are trying to sledge uphill so we can use this to inform the player that they are going in the wrong direction; the code between lines 8 and 18 handles this, in this case simply by forcing the sledge to stop so the user cannot go too far in the wrong direction.

A Practical Example

A Word of Caution

Before you can use Havok, you need to ensure that it is set up correctly. Unfortunately, because of a time limit passing on the inclusion agreement, the Havok Xtra is not, at the time of writing, part of Director MX 2004, so you may have some problems in this regard. Macromedia is trying to reach an agreement with Havok to make the Xtra available to Director MX 2004 users, so by the time you read this, it may be on the CD or available as a download. Or, you can install Shockwave10 from the CD and then navigate to the Macromed folder. On a Windows machine this will be in the folder 'Windows\System32\Macromed'. Inside this folder is a 'Shockwave10' folder that contains an 'Xtras' folder. The file 'Havok.x32' should be in this folder. Once you have located the Xtra, you need to copy it to your Director MX 2004 installation folder. On a Windows machine, this will be in 'Program Files\Macromedia\Director MX 2004\Configuration\Xtras'. Create a folder called 'Havok' inside this folder and then copy 'Havok.x32' to this folder. You will also need the documentation and the cast libraries which you can get from http://oldsite.havok.com/xtra. Place the CastLibs in a Havok folder inside the 'Configuration/Libs' folder. The documentation can be placed wherever it suits your needs. Once you have added these files, you will need to restart Director MX 2004 before the xtra will be available for use.

Line 21 is where we use Havok to control the motion of the sledge. Instead of simply moving the sledge we give it a nudge using the rigid body method 'applyAngularImpulse'. This uses a vector to apply rotational motion to the rigid body. In the example at line 21, we use the previously stored property variable pTurn to give a rotation in the y axis. The left and right keys are used to control the rotation while the up and down arrows control forward and back motion. Lines 27 and 28 show how this is achieved; again we use a method of the rigid body, applyImpulse, that takes a vector and applies a force defined by the vector. To move the sledge, we want the force for a forward motion to be along its z axis, so we first get the z axis of the sledge, called board in line 27. This is scaled up by a boost value that is increased or decreased in lines 14 to 18.

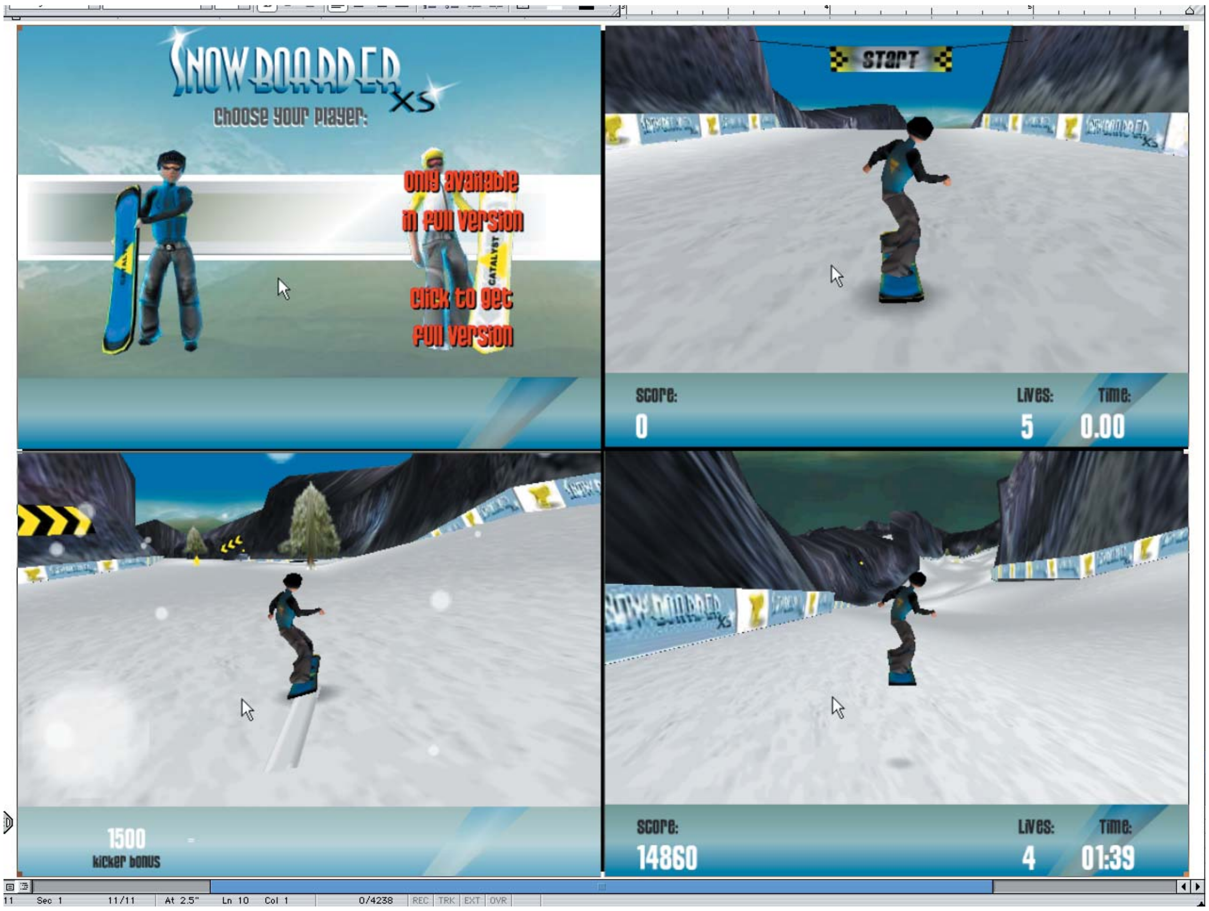
The behavior of the sledge would be highly erratic if we left it at that, so we must dampen down the effect in a separate handler, dampenPhysics, which we will look at in the next section.

Controlling the Results

Physical simulations are great unless the behavior does not accurately suit your application. In this case, you must try to control the behavior. One thing you cannot do is place a rigid body directly. You must use Havok's methods to achieve the ends you desire. Some tips are shown in Listing 3. Lines 3 to 12 get the height of the ground directly below the sledge. Lines 15 and 16 show how the rotation of the sledge is reduced by applying a damping affect which is the opposite of the sledge's angular velocity. Lines 19 to 28 stop the sledge sliding too much in the x axis. We want the sledge to appear as though the runners hold the snow so it is only happy when going forwards or back. The physics simulation will slide to the left and right just as easily as it will slide forward or back so we need to stop this sliding action. This should only happen when the sledge is on the ground – boardHeight<100 – not in the air. At line 21 we get the linearVelocity of the sledge. This is a vector giving a speed in the x, y and z axes. We use the vector method normalize to get a vector of unit length. Line 24 uses the vector method 'dot' to find how much of the vector is aligned with the sledge's x axis. The result is a single value that we then map to the x axis by multiplying the x axis by this value in line 25. We now have a vector that expresses the direction we need to use for dampening the sideways slide. By multiplying this by the magnitude of the linear velocity and a previously stored scaling factor 'pSlideDamping' that we can use to give more or less sideways slide we can finally, at line 27, apply this to the sledge to minimize the sideways slide.

Another thing we need to check is whether or not we are upside down. In any simulation, this can happen. To do this, we use the dot product of the sledge's y axis with the default y axis (0, 1, 0). If this is less than 0.5, then we are currently upside down. Rather than react immediately to this, we simply update a counter. This lets us adjust how quickly we reset from such a situation. In lines 42 to 48, you can see that we allow a count of 20 before attempting a correction. To do the correction, we take a

figure 8



Nik Lever runs Catalyst Pictures Ltd (www.catalystpics.co.uk), a Manchester, UK-based company that makes online games for clients including Kellogg's, BBC, and Cartoon Network. Originally trained as an animator, Nik now writes code more often than he scribbles and is the author of several Flash and Director books (www.niklever.net). When he's allowed out into the light, he likes to go sailing across the Irish sea.

nik@niklever.net

default transformation matrix, set the position of the matrix to the current sledge position, then move this up a little to ensure we are above the snow. Then we use, in line 46, the 'interpolatingMoveTo' method of the rigid body. This takes 2 parameters, position and orientation, expressed as an angle axis list. If the rigid body can be moved without interpenetrating other rigid bodies in the scene then the body is moved; if not it remains unmoved.

A Real World Example

The ideas given in the sledging example were extended to controlling a snowboarder in a game we market as 'SnowboarderXS'. In this game, which is available from the support files at 'SnowboarderXS.html' (see Figure 8), you choose one of two characters and courses and then slide down the course, jumping ramps, collecting stars, and performing tricks. The control of the snowboarder uses a proxy object that operates identically to the sledge demo.

The Havok Xtra is a fantastic tool for creating high-quality online games. In this article, we covered the basics of using the control then moved on to some of the more complex ways you can control the simulations. Always remember that the most important starting point for a simulation that uses Havok is to either bring in your geometry scaled to meters, or set up the scaling parameters to suit your geometry. Okay, boots on, bindings fastened, let's get some air!

• • •

The source code fo this article is online at www.sys-con.com/mx/sourcec.cfm.

listing 1

```
on beginSprite
  s = member( "HelloWorld" )
  s.resetWorld()
  hk = member( "Havok" )
  hk.gravity = vector(0, -10, 0)
  -- create ground
  m = s.model("floor")
  m.addModifier(#meshdeform)
  hk.makeFixedRigidBody(m.name)
  -- create text
  m = s.model("hello")
  m.addModifier(#meshdeform)
  hk.makeMovableRigidBody(m.name, 100.0, false)
end
```

listing 2

```
on enterFrame
  checkkeys

  boardSpeed = boardRB.linearVelocity.magnitude
  boardDY = pPrevY - board.worldPosition.y
  pPrevY = board.worldPosition.y

  if boardDY<0 then
    pWrongWayCount = pWrongWayCount + 1
  else if pWrongWayCount>0 then
    pWrongWayCount = pWrongWayCount  - 1
  end if

  if pWrongWayCount>50 and pBoost>10000 then
    pBoost = pBoost - 200
  else if pBoost<40000 then
    pBoost = pBoost + 200
  end if

  if leftkey then
    boardRB.applyAngularImpulse(vector(0,pTurn,0))
  end if
  if rightkey then
    boardRB.applyAngularImpulse(vector(0,-pTurn,0))
  end if
  if upkey and boardSpeed<7000 and pUpsideCount = 0 then
    hitV = board.transform.zAxis * -pBoost
    boardRB.applyImpulse(hitV)
  end if
  if downkey then
    hitV = board.transform.zAxis * pBoost
    boardRB.applyImpulse(hitV)
  end if
  if not spacekey then
    dampenPhysics
  end if
end
```

listing 3

```
1   on dampenPhysics
2     -- check height from snow
3     boardHeight = 0
4     wpoint = board.worldPosition
5     wdown = -1.0 * board.transform.yAxis
6     idetails = w3d.modelsUnderRay(wpoint, wdown, 2, #detailed)
7     repeat with j = 1 to idetails.count
8       if idetails[j].model = landscape then
9         boardHeight = idetails[j].distance
10        exit repeat
11      end if
12    end repeat
13
14    -- reduce heading rotation
15    rbForce = boardRB.angularVelocity * -pRotationDamping
16    boardRB.applyAngularImpulse(rbForce)
17
18    -- only reduce slide when on or near the ground
19    if boardHeight<100 then
20      -- reduce sideways slide
21      rbForce = boardRB.linearVelocity
22      rbForceMag = rbForce.magnitude
23      rbForce.normalize()
24      forceXfactor = rbForce.dot(board.transform.xAxis)
25      forceXfactor = forceXfactor * board.transform.xAxis
26      xDampen = -rbForceMag * forceXfactor * pSlideDamping
27      boardRB.applyImpulse(xDampen)
28    end if
29
30   -- flip if upside down
31    tn = transform()
32    pUp = tn.yAxis
33    wUp = board.transform.yAxis
34    cosine = wUp.dot(pUp)
35
36    if cosine < 0.5 then
37      pUpsideCount = pUpsideCount + 1
38    else
39      pUpsideCount = 0
40    end if
41
42    if pUpsideCount>20 then
43      tn = transform()
44      tn.position = board.worldPosition
45      tn.position.y = tn.position.y + 50
46      boardRB.interpolatingMoveTo(tn.position, [vector(0,1,0), 0])
47      pUpsideCount = 0
48    end if
49
50  end
```

THE DIVISION THAT PUTS DIRECTOR TO WORK

The Rochester Institute of Technology (RIT), and the Information Technology Department where I teach, has long been a user of Macromedia products, from Dreamweaver to Director and most things in between. Recently, there has been some debate as to the future of Director's role, considering the increasing use of Flash in the marketplace for Web graphics and interactive displays. While we have moved some of our efforts to Flash technologies, RIT and the IT Department remain committed to the use of Director, because no other product currently on the market can meet our specific needs for an easy-to-script, but performance-aware 3D system capable of delivery on the Web and via downloadable executable.

by andrew m. phelps

The shift to ECMAScript syntax represents a major advance in the Director product, and is important to us for a number of reasons. The primary reason for the inclusion of this feature seems to be to provide for a more common language and syntax between Flash MX and Director. We are capitalizing on the similarities in the new syntax to create a streamlined educational experience to bring the products as close together as possible. It also allows for a more complete object-oriented structure, and better tools for list management and string operations.

The Information Technology Department at RIT

Students and Majors

The Information Technology Department at RIT currently enrolls more than 1,000 undergraduate students and approximately 400 graduate students majoring in Information Technology, Networking & System Administration, Software Development & Management, and New Media. We are also hard at work in developing offerings in game programming (a concentration already exists; we are also pursuing plans for a masters degree) and Digital Security. The IT Department is housed within the College with Computer Science and Software Engineering. Courses are generally open to any of the students in any of these programs, as well as to select students in the College of Imaging Arts & Sciences (CIAS), and other students from other areas of campus.

Students and Educational Objectives

The direct effect of all of the different majors and minors available to students is that we have several different types of students in our courses. Some come from traditional computing backgrounds, heavily immersed in math, science, and programming, while others approach application building from disciplines like art, graphic design, and printmaking.

All of our students are required to learn some programming, but because they are majoring in different fields, their reasons for doing so are widely varied. Some of our students are directly concerned with operating systems and low-level driver design; others could care less, and are solely focused on the user experience. Some are designing pieces for back-end server rooms where performance is key; to others it's all about selling the client with look and feel.

Why Director?

Visual Programming

Given that we have students involved with the department with differing prerequisite backgrounds, it is a challenge to ensure that all of them have the basic programming skills they will need to survive in the IT world. Every student majoring in information technology (but not New Media) is responsible for completing an introductory sequence in the Java programming language. In addition, each must complete an introductory course entitled Programming for Digital Media. This course uses visual tools like Director and Flash to allow students to create a visual application quickly, before they may be able to do so in Java or C/C++. It is a compelling educational experience to be able to create a visual application with graphics and sound in the first year of study.

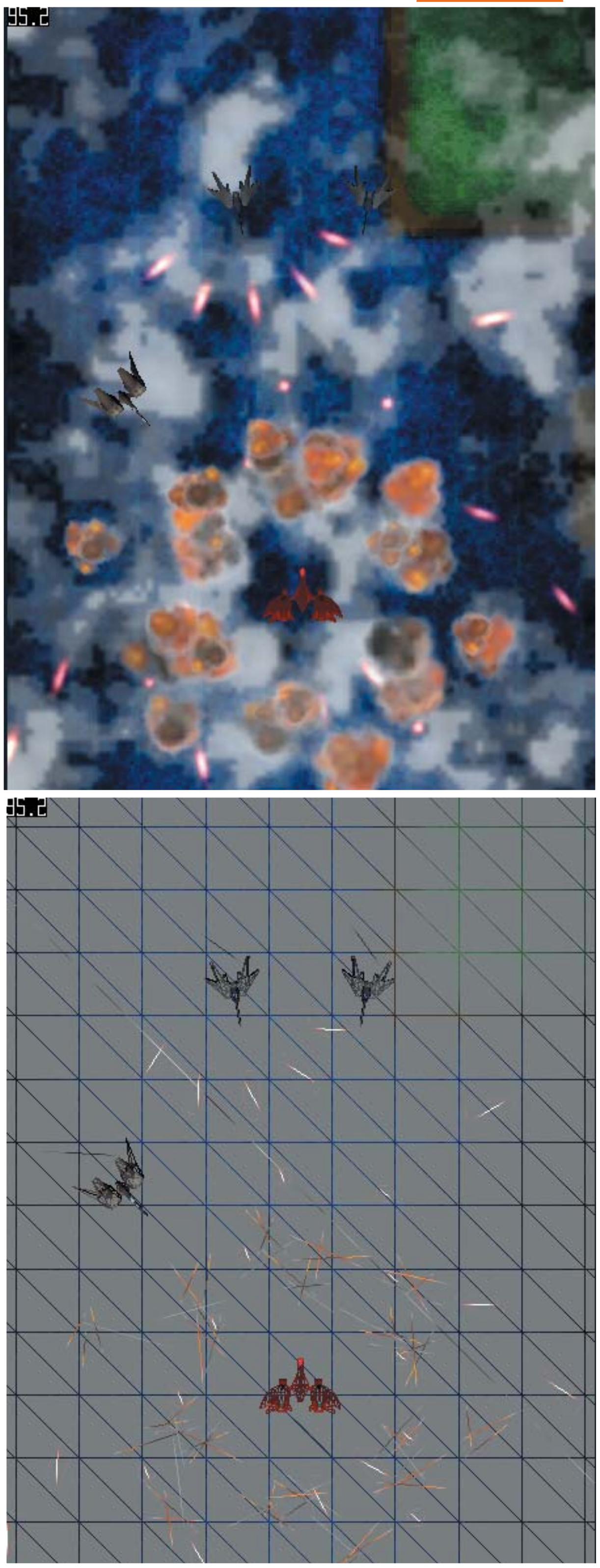
We have found this to be particularly effective, as you might expect, in teaching students who come from visually oriented backgrounds (art, graphic design, etc.). But this is generally a course enjoyed by all students because of its visual nature: students want to know that they are creating "real applications" that look and feel like those they use on a daily basis. Creating command line–driven exercises has the ring of being false, and as such, it is my experience that students are more engaged and driven to learn in a visual environment. In fact, my team is working on software to allow students to do in Java some of the visual-style programming that makes Flash and Director so appealing.

Recently, we moved a lot of the coursework in Programming for Digital Media from Director to Flash. This is to give students the Flash experience, and because it seems to be the tool of choice for interface design and implementation. Flash handles buttons, sliders, and widgets with amazing ease, and the lightweight plug-in makes it a snap to use almost anywhere. We talked about moving all of the material to Flash, but in the end, this was not implemented for the following reasons:

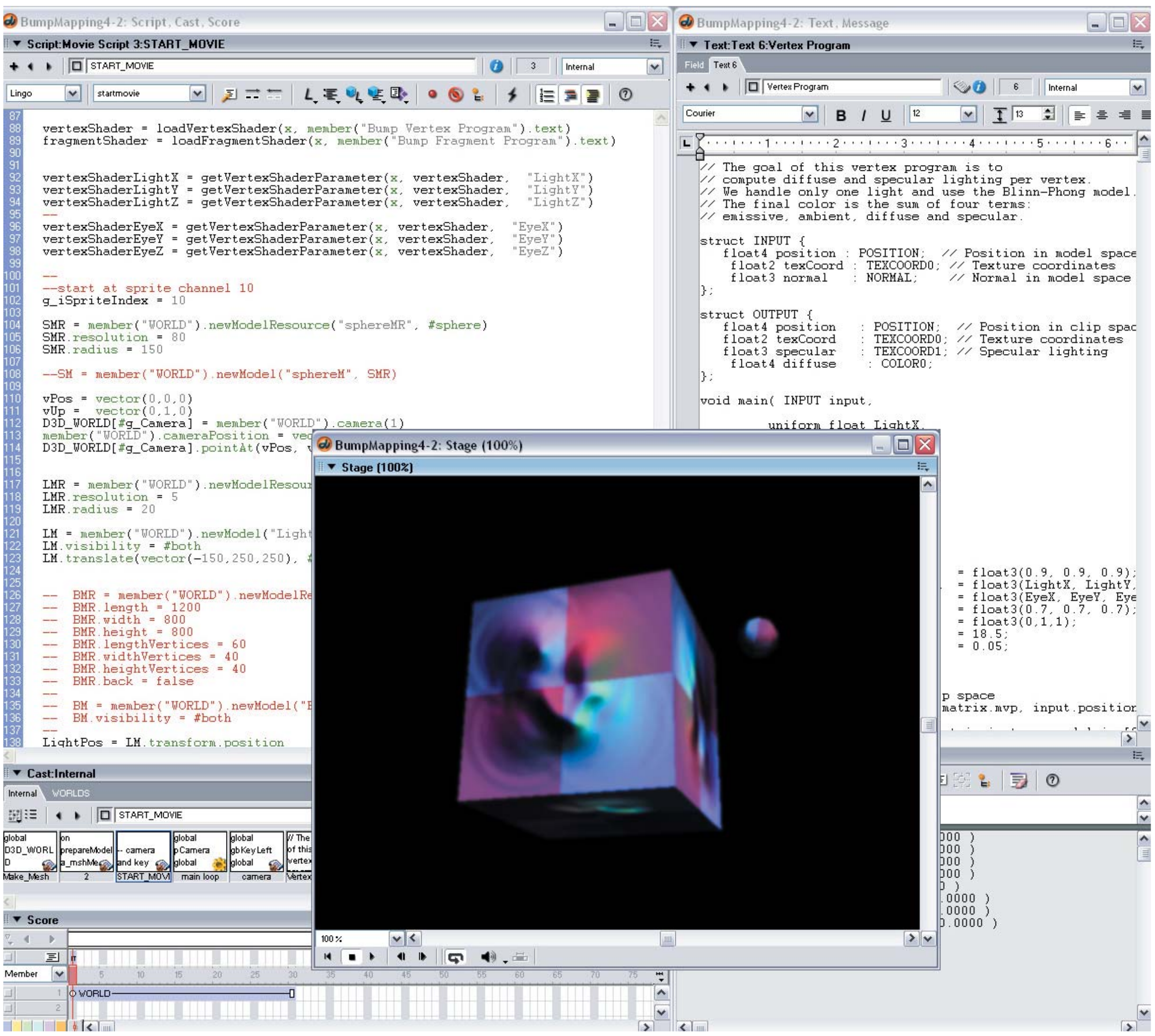
Imaging Lingo

The first technology that Director contains that is important to us is Imaging Lingo. This is a collection of commands that deal with image manipulation, allowing the programmer to copy rectangles of

figure 1



COPYRIGHT 2003-2004 A PHELPS, A CLOUTIER.


BumpMapping4-2: Script, Cast, Score

Script:Movie Script 3:START_MOVIE

```
vertexShader = loadVertexShader(x, member("Bump Vertex Program").text)
fragmentShader = loadFragmentShader(x, member("Bump Fragment Program").text)


vertexShaderLightX = getVertexShaderParameter(x, vertexShader,  "LightX")
vertexShaderLightY = getVertexShaderParameter(x, vertexShader,  "LightY")
vertexShaderLightZ = getVertexShaderParameter(x, vertexShader,  "LightZ")
--
vertexShaderEyeX = getVertexShaderParameter(x, vertexShader,  "EyeX")
vertexShaderEyeY = getVertexShaderParameter(x, vertexShader,  "EyeY")
vertexShaderEyeZ = getVertexShaderParameter(x, vertexShader,  "EyeZ")

--
--start at sprite channel 10
g_iSpriteIndex = 10

SMR = member("WORLD").newModelResource("sphereMR", #sphere)
SMR.resolution = 80
SMR.radius = 150

--SM = member("WORLD").newModel("sphereM", SMR)

vPos = vector(0,0,0)
vUp =  vector(0,1,0)
D3D_WORLD[#g_Camera] = member("WORLD").camera(1)
member("WORLD").cameraPosition = vec
D3D_WORLD[#g_Camera].pointAt(vPos, v


LMR = member("WORLD").newModelResour
LMR.resolution = 5
LMR.radius = 20

LM = member("WORLD").newModel("Light
LM.visibility = #both
LM.translate(vector(-150,250,250), #


--  BMR = member("WORLD").newModelRe
--  BMR.length = 1200
--  BMR.width = 800
--  BMR.height = 800
--  BMR.lengthVertices = 60
--  BMR.widthVertices = 40
--  BMR.heightVertices = 40
--  BMR.back = false
--
--  BM = member("WORLD").newModel("B
--  BM.visibility = #both
--
LightPos = LM.transform.position
```

BumpMapping4-2: Text, Message

Text:Text 6:Vertex Program

```
// The goal of this vertex program is to
// compute diffuse and specular lighting per vertex.
// We handle only one light and use the Blinn-Phong model.
// The final color is the sum of four terms:
// emissive, ambient, diffuse and specular.

struct INPUT {
   float4 position : POSITION;  // Position in model space
    float2 texCoord : TEXCOORD0; // Texture coordinates
    float3 normal   : NORMAL;    // Normal in model space
};

struct OUTPUT {
   float4 position   : POSITION;  // Position in clip spac
   float2 texCoord   : TEXCOORD0; // Texture coordinates
   float3 specular   : TEXCOORD1; // Specular lighting
    float4 diffuse    : COLOR0;
};

void main( INPUT input,

         uniform float LightX,
```

```
 = float3(0.9, 0.9, 0.9);
 = float3(LightX, LightY,
 = float3(EyeX, EyeY, Eye
 = float3(0.7, 0.7, 0.7);
 = float3(0,1,1);
 = 18.5;
 = 0.05;

p space
matrix.mvp, input.position
```

BumpMapping4-2: Stage (100%)

figure 2

one image to another, get or set a pixel of an image to a specific color, extract or set the alpha of an image, or apply ink effects to an image or a part of one. This is important to us because the Imaging Lingo operations, and the functionality they allow, are similar to the functionality employed in texture management in lower-level environments.

We teach several courses on game programming, including 2D and 3D Graphics using DirectX and OpenGL. Students develop engines in C/C++ and manipulate textures directly on the graphics card. But the operations, from loading images, copying bits, dealing with alpha, etc., have direct parallels in Imaging Lingo. Students who have taken the prerequisite coursework in digital media have almost no problem grasping what is going on inside a modern graphics engine because of their familiarity with the concepts.

Director makes a great play space and learning environment for exploring imaging concepts because it shields the programmer from the one thing they are most likely to encounter – access to undefined memory. If any of the coordinates that an image copies run amuck, then you are probably trying to read from or write to memory that you don't "own". Dealing with this in C/C++ can be a pain, so it is convenient to explore these concepts first within the protective shield of the Lingo framework.

Shockwave 3D

The single most important reason that the IT Department is using Director is by far Shockwave 3D. No other package on the market provides a cheap, easy way to script a 3D engine with hardware access that can be played back on a machine within a browser and without the latest in graphics hardware.

The ease with which a student can get a 3D engine up and running with Shockwave3D is almost mind-blowing: we left it in Programming for Digital Media as the one Director-based assignment because the ability to do anything in 3D is so important to the student mindset and really empowers them to think, "Hey, I can do this" well before their programming ability would allow them to get involved with 3D in a traditional sense. The 2D and 3D graphics courses, for example, are taught in the third and fourth years of study in the Information Technology program, and are also taught later in the Computer Science program.

To give you a sense of how easy it is to get something happening in 3D, the code presented in Listing 1 is a simple example of placing a cube at the world origin and then rotating a camera around it on a per-frame basis. With the new ECMAScript Syntax in DMX 2004, this is very similar to Flash programming, and our students can bounce back and forth with relative ease (see Listing 1).

Projects RIT Is Involved In

While the above project is a good example of how easy it can be to get involved in Shockwave3D, we are heavily involved in products of greater complexity. It is impossible to catalog all of the ways in which we use Director and other Macromedia projects, but the following are a small selection of things we have done recently with Shockwave3D:

Games and 3D Engines

Games are an important part of our curriculum, both as educational tools and as a concentration area. Some of our students have gone on to work in the games industry, at Microsoft, Electronic Arts, and several smaller studios. Exploring game programming and engine building is currently a very hot academic area, and Shockwave3D allows students to start building 3D games before they take the more traditional graphics courses. One of these projects, entitled "Project Broadsword", has been used in several papers at the Director Online User's Group as an example project for various techniques that were explored during its construction (see Figure 1).

In addition, because S3D offers access to hardware, we've capitalized on it in two ways. It is possible to write traditional 2D games in S3D by fixing the camera perspective. This allows us to design 2D games to the industry standard 60 FPS, which we were usually unable to accomplish using the traditional engines in either Director or Flash.

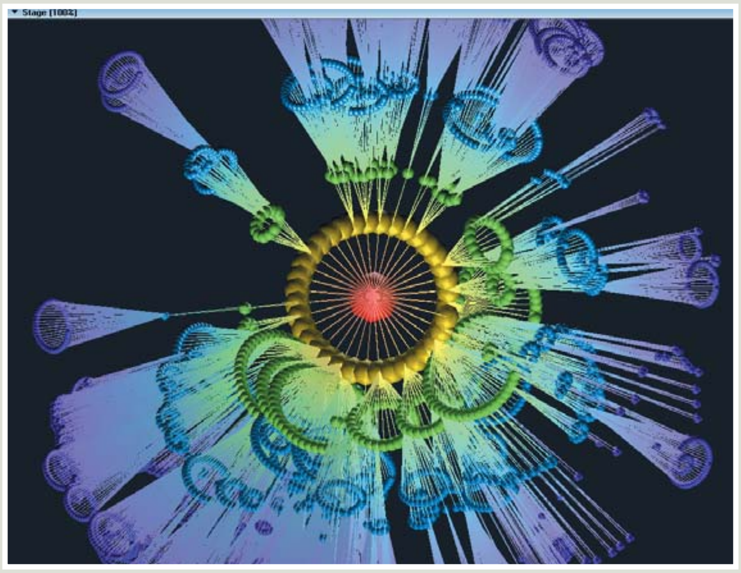
We've also been able to manipulate the access that Director provides to graphics hardware to add in support to the Nvidia Cg language. Using a custom Xtra we designed, we can use Director as an environment to explore vertex- and pixel-shader development. While this project has just begun, it has the potential to be a very useful prototyping tool (see Figure 2).

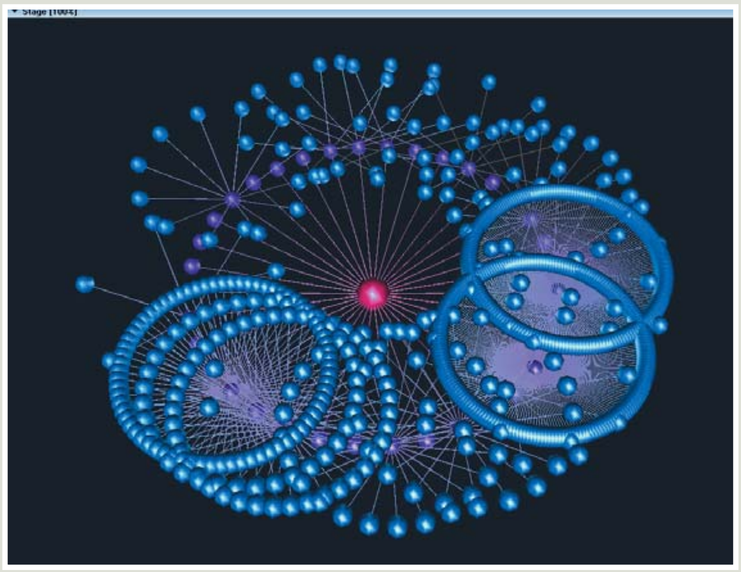
That Director allows us the XDK at no extra charge to develop in-house tools for the product is a great fit for us. In the future, hopefully more of the S3D XDK will be made publicly available.

Shared Extensible Learning Spaces

My colleagues, Professors Steve Kurtz and Nancy Doubleday, run an upper-division course every year entitled "Shared Extensible Learning Spaces," which is an evolving set of strategies for the representation of complex concepts in interactive media spaces. Today, the faculty and students are creating 3D spaces populated by smart actors to explore emerging behavior in self-organizing systems and document the cognitive experience of users in virtual worlds. They are devel-

examples





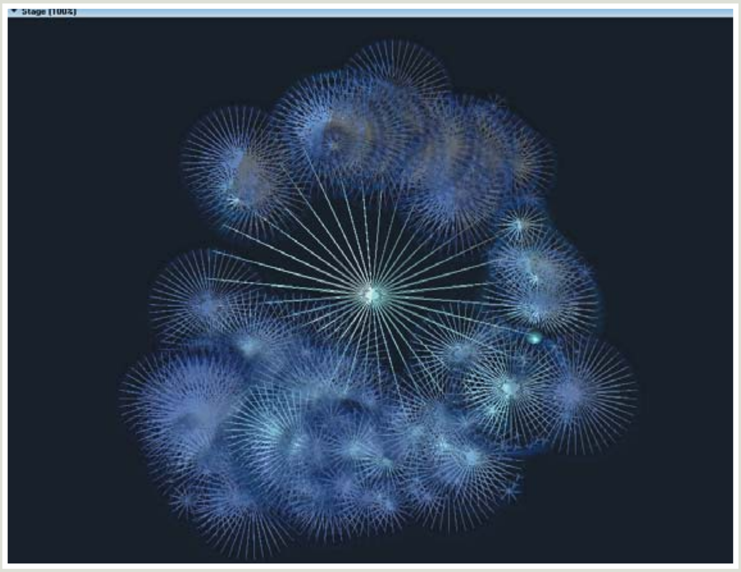
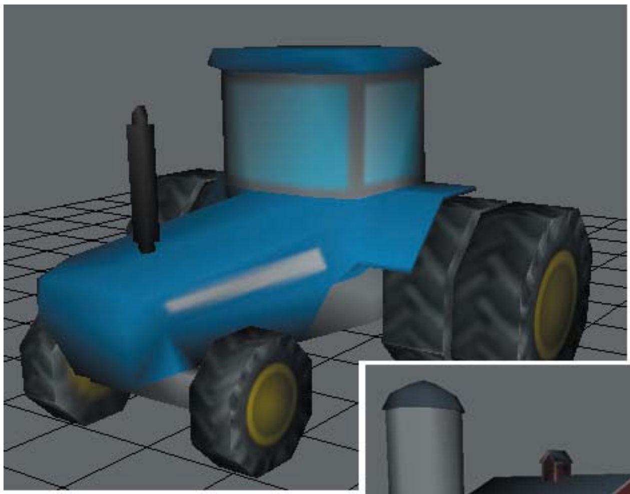
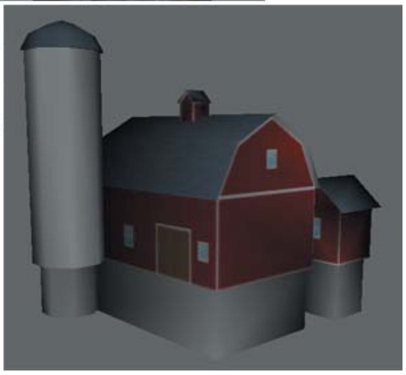


figure 3

oping tools using the Macromedia Director environment at three levels. At the top level are complete products: interactive simulations for learners and researchers. The bottom level is an extensible set of modules and components for the programming environment in which our tools are built. Our middle-level tools allow users/authors to participate in the SHELS project at a level appropriate to their skills and objectives. These components scaffold the experience of simulation building for content experts and learners whose purposes are best served by working at a higher level. We design our environments to facilitate active learning and encourage inquiry and scholarship at every level.

One of the early prototype worlds for the SHELS system was the "Tractor War" world, which allowed the user to observe an ongoing battle between teams of varying colors ("red team," "blue team," etc.) that had different evolutionary algorithms for obtaining and storing food. The teams were represented by tractors (which gathered food) and barns (which stored it) in a 3D world (see Image 3). Director allows a multiuser architecture and the ability to provide 3D front ends to the networked data was a key component in the design of this application.

Data Visualization Prototyping

Another way in which we use Shockwave3D is by prototyping data visualization. A team of students can get a prototype off the ground in just a few days, as opposed to building a complex system that may take months or even years to construct. Since experimentation in visualization is generally an unknown – i.e., you never know if your way of visual-

Web based collaboration made to order.

Flexible administration,
Authentication integration,
Section 508 compliant,
and coming soon...

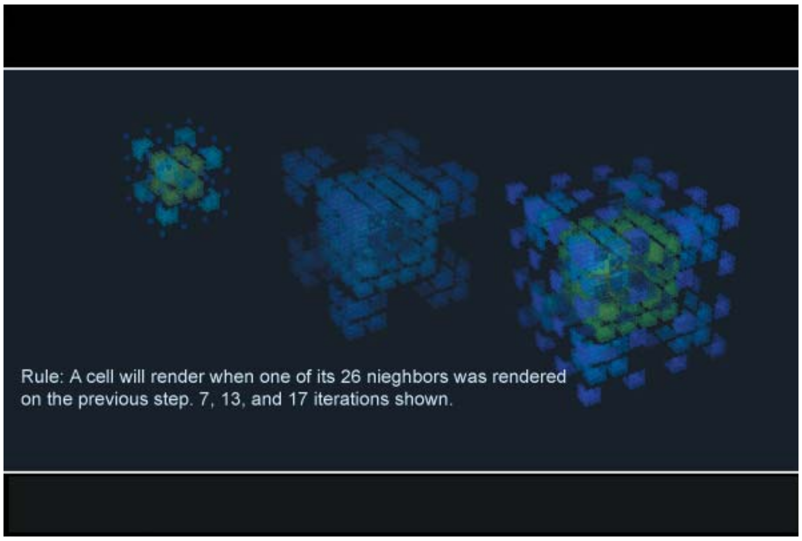
FuseTalk for Microsoft .NET.

1-866-477-7542

FuseTalk™

Discussion forum solutions that make web-based collaboration risk-free and easy.

figure 4


Rule: A cell will render when one of its 26 nieghbors was rendered
on the previous step. 7, 13, and 17 iterations shown.

izing something will be effective until it's attempted – the need for rapid prototyping is paramount. We've used Director to simulate file system visualization (see Figure 4) as well as cellular automata growth (see Figure 5). In the file system example, it showed enough promise that we went on to develop a more robust system in C++/Java that we are currently still testing.

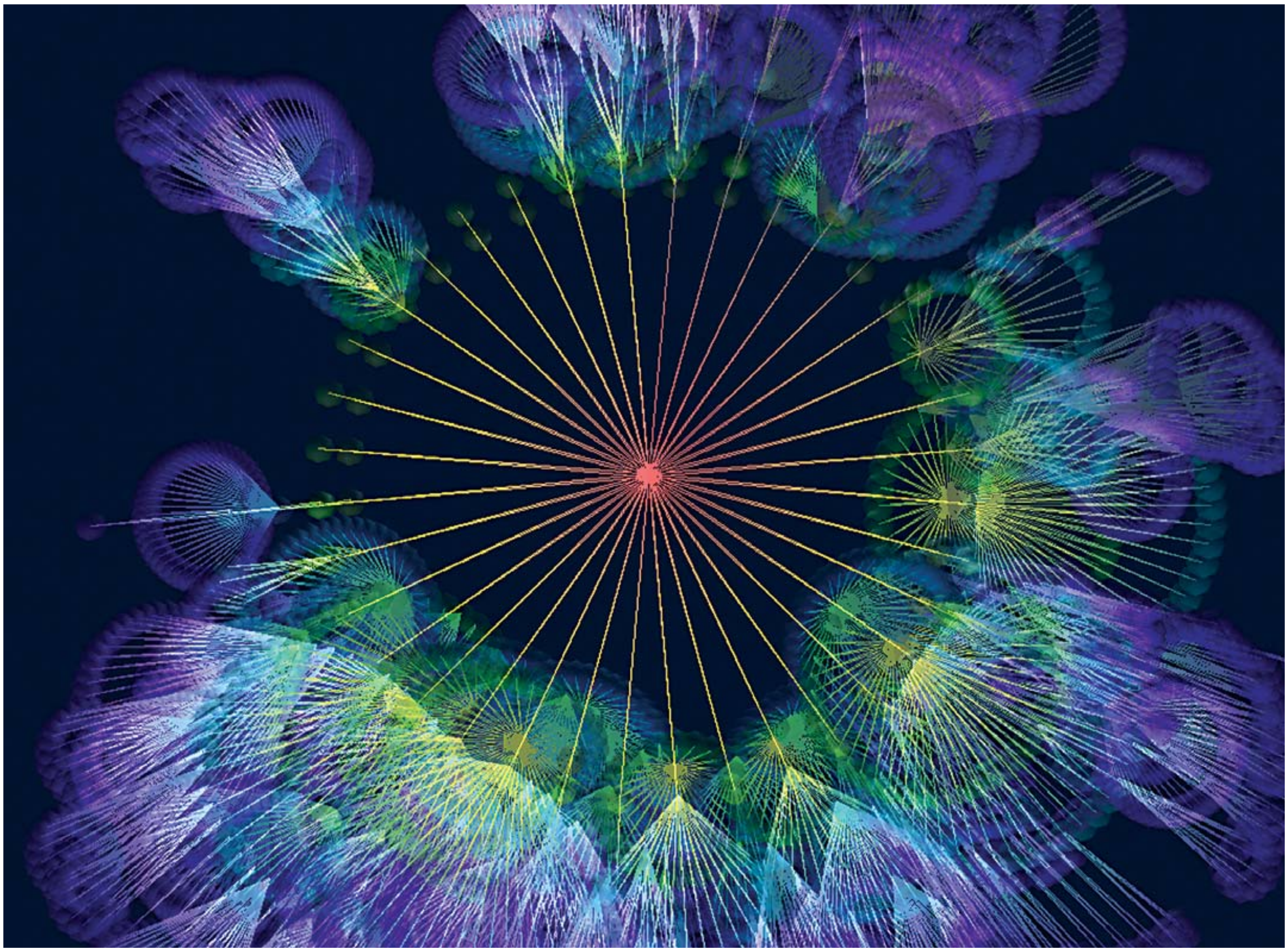
Conclusion

Director is a powerful tool for use in educational institutions. It allows students to ramp up quickly to create the applications they want to build, and provides building blocks for those that go further on to study lower-level programming languages. The new syntax afforded in DMX2004 allows a closer working relationship with FlashMX users, and has been advantageous to us in teaching students from a wide variety of backgrounds by providing a common language core.

We have been very impressed with what we have been able to accomplish with the existing Shockwave3D package, and hope to see its use grow both within the academic community and in the industry at large.

Andrew M. Phelps is in the Information Technology Deptartment at the Rochester Institute of Technology in Rochester, NY (http://andysgi.rit.edu/). amp@it.rit.edu

figure 5



```
//SIMPLE3DWORLD CLASS
/*---------------------------------------------
constructor for Simple 3D World Object
 ---------------------------------------------*/
function Simple3DWorld(sName, iSpriteNum) {

   //set basic props
   this.sName          = sName;
   this.iSpriteNum    = iSpriteNum;
   this.ThreeDSprite = sprite(iSpriteNum);
   this.ThreeDWorld  = sprite(iSpriteNum).member;

   this.mSphereSetup(); //create a sphere
   this.mCameraSetup(); //position camera
}


/*---------------------------------------------
Simple3DWorld::SphereSetup
creates a sphere at 0,0,0 with radius 200
---------------------------------------------*/
Simple3DWorld.prototype.mSphereSetup = function() {
    //create a template (model resource)
    //for how to build a spheres
    this.mrSphere = this.ThreeDWorld.newModelResource(
                       this.sName + "_model_resource",
                       symbol("sphere"));

    //fill out properties of the template
    this.mrSphere.radius       = 200;
    this.mrSphere.startAngle = 0.000;
    this.mrSphere.endAngle    = 360.000;
    this.mrSphere.resolution = 50;

    //build a sphere in the world
    this.mSphere = this.ThreeDWorld.newModel(
                       this.a_sName + "_model",
                       this.mrSphere);

    //set its position in the world
    this.mSphere.transform.position = vector(0,0,0);
}


/*---------------------------------------------
Simple3DWorld::CameraSetup
positions the camera at XYZ 0,200,800, pointed
at the origin
---------------------------------------------*/
Simple3DWorld.prototype.mCameraSetup = function() {
    var vCamPos = vector(0,200,800); //cam position
    var vLookPos = vector(0,0,0);     //lookat position
    var vUp =  vector(0,1,0);         //"up" orientation

    //get a handle to the default camera created by S3D
    this.oCamera = this.ThreeDWorld.getProp("camera", 1);
    //set its world position
    this.oCamera.transform.position = vCamPos;
    //point it at the origin, with no rotation along
    //its local Z axis (vUp matches world-space up)
    this.oCamera.pointAt(vLookPos, vUp);
}

/*---------------------------------------------
Simple3DWorld::Update
called per frame to animate camera in a
simple loop
---------------------------------------------*/
Simple3DWorld.prototype.mUpdate = function() {
    //rotate the camera around the Y-axius
    //2 degrees per frame, using world coordinate
    //space.
    this.oCamera.rotate(0,2,0,symbol("world"));
}


/*---------------------------------------------
Simple3DWorld::ResetWorlds
called on movie destruction, resets the 3D
castmember to its original state.
---------------------------------------------*/
Simple3DWorld.prototype.mResetWorlds = function() {
    this.ThreeDWorld.resetWorld();
}
```

```
//STARTMOVE & STOPMOVIE SCRIPTS
//global start and stop movie functions
//these are provided by director

 function startmovie() {
   //create an object to setup and control
   //the 3D world.  Pass it which sprite number
   //the 3D cast member is in.
   _global.o3D WorldObject = new Simple3DWorld(
                                   "sphereWorld",
                                   1);
 }
```

```
 function stopmovie() {
   //clean up after ourselves, reset the world
   //to an empty one, and clear all globals
   _global.o3DWorldObject.mResetWorlds();
   _global.clearGlobals();
 }
```

```
//EXITFRAME LOOP
//called once per frame
function exitFrame() {
  //rotate the camera in the 3D world
  _global.o3DWorldObject.mUpdate();

  //loop back to this frame again
  _movie.go(_movie.frame);
}
```

listing 1

web services EDGE conference&expo

Web Services Edge 2005 East

International Web Services Conference & Expo

Announcing Web Services Edge 2005 Extending Call for Papers to August 30th

Submit your proposal today!

The Largest *i*-Technology Event of the Year!

Guaranteed Minimum Attendance 3,000 Delegates

Tuesday, 2/15: Conference & Expo | Wednesday, 2/16: Conference & Expo | Thursday, 2/17: Conference & Expo

Hynes Convention Center, Boston, MA
February 15-17, 2005

Join us in delivering the latest, freshest, and most proven Web services solutions... at the *Fifth Annual Web Services Edge 2005 East – International Conference & Expo* as we bring together IT professionals, developers, policy makers, industry leaders and academics to share information and exchange ideas on technology trends and best practices in secure Web services and related topics including:

- Transitioning Successfully to SOA
- Federated Web Services
- ebXML
- Orchestration
- Discovery
- The Business Case for SOA
- Interop & Standards
- Web Services Management
- Messaging Buses and SOA
- SOBAs (Service-Oriented Business Apps)
- Enterprise Service Buses
- Delivering ROI with SOA
- Java Web Services
- XML Web Services
- Security
- Professional Open Source
- Systems Integration
- Sarbanes-Oxley
- Grid Computing
- Business Process Management
- Web Services Choreography

3-Day Conference & Education Program

features:

- Daily keynotes from companies building successful and secure Web services
- Daily keynote panels from each technology track
- Over 60 sessions and seminars to choose from
- Daily training programs that will cover Web Service Security, J2EE, and ASP.NET
- FREE full-day tutorials on .NET, J2EE, MX, and WebSphere
- Opening night reception

Interested in Exhibiting, Sponsoring or Partnering?

Becoming a Web Services Edge Exhibitor, Sponsor or Partner offers you a unique opportunity to present your organization's message to a targeted audience of Web services professionals. Make your plans now to reach the most qualified software developers, engineers, system architects, analysts, consultants, group leaders, and C-level management responsible for Web services, initiatives, deployment, development and management at the regions best-known IT business address – The Hynes Convention Center in Boston.

For exhibit and sponsorship information please contact Jim Hanchrow at 201.802.3066, or e-mail at jimh@sys-con.com.

Sponsored by:

JDJ · WebServices Journal · SD Times · wldj · .NET Journal · WebSphere Journal · information STORAGE+SECURITY journal · XML Journal · asp.netPRO · MX developer's journal · eaindustry consortium

Contact for Conference Information: Jim Hanchrow, 201-802-3066, jimh@sys-con.com

PRODUCED BY SYS-CON EVENTS

www.sys-con.com/edge

Double-Clicking A Document

A professional projector makes it easier

by tom rockwell

Thanks to the extensive range of xtras available, a Director projector can create any type of document and write it to the user's hard disk. In Windows Explorer and the Macintosh Finder, users expect that a double-click on a document icon will launch the program that created it. This article shows you how to associate files with the Director projector that created them. By the end of this article, you will have created a simple word processor application that creates its own document files; double-clicking on the icon of one of these files will open your Director application so that you can edit the document.

There are many situations in which this technique will prove useful. For example, you may want to save the state of a game in progress, a customized slide show presentation, or mark up data for a video sequence.

But recognizing the files created by a projector is not something that Director does naturally. By default, when the user double-clicks on a document created by a Director projector, the operating system prompts the user to select an application to open the file. If the user properly selects your projector, then your projector is launched, but the document still doesn't open.

For my programs, this is unacceptable. I sell programs that are often in direct competition with titles from major software vendors. Something like the double-click may seem trivial, but the little things add up and the less my users have to think about, the more likely they are to purchase the product. I like my projectors to be as professional as possible, complete with robust menus, keyboard shortcuts, user-defined preferences, and standard user-interface elements. In the end, if the user can't tell that the program was created with Director, then I've done my job. This means finding a way to deal with double-clicking on documents.

Thankfully Director has an undocumented feature – a system variable called the commandLine that corresponds to the operating system's command line. When the user double-clicks on a document, the operating system's command line gets updated with the path to the document, and Director's commandLine provides us with this information.

But figuring out which document the user clicked on is only half the battle. The operating system also needs to be told to launch your projector instead of prompting the user, and then your application has to be set up to deal with what happens so the file actually opens. This article will give you step by step instructions for getting this to work for Macintosh OSX and Windows platforms. The commandLine doesn't exist on pre-OSX Macintosh operating systems because those systems don't have a command line at all. So, this technique will not work on OS8 or OS9.

This article assumes you have a basic understanding of Lingo, movie scripts, frame scripts, and working with xtras. You will also need some additional tools besides Director. For Mac OSX, you will need a program called Property List Editor, which can be found on the Developer Tools CD that came with your copy of OSX. If you don't have that, you can get away with just using a text editor, but you'll need to have a basic understanding of XML. You will also need DirectorMX or DirectorMX2004 since those are the only versions of Director that can create projectors for OSX. For Windows, you will need a third-party xtra since we will be editing the Windows Registry. This article uses the BuddyAPI xtra, but any xtra that can modify the Registry will work, such as the Registry xtra and others.

Setting Up Your Application

In order to have a document to double-click, we will need to have an application that can save an open document. In this article, we will create a simple text editor that can save and open documents. It will function like a mini word processor, so I'll call it MiniWord.

Launch Director and create a new document with a white background. It can be any size, but there's no reason to make it huge, so make it fit comfortably on your screen. Next, create a field member by selecting Insert > Control > Field. The field will appear onstage with a cursor in it. Leave it blank for now and find the field cast member in the cast window. Name the cast member "content." Now select the field sprite on the stage and open the Property Inspector. Click on the Field tab and click on the Editable check mark. In the Framing dropdown menu select "Scrolling" and resize the sprite so it fills most of the stage. Leave some room at the bottom for some buttons that we'll create in a moment. Change the Property Inspector to list mode if it isn't already and set the border of the field to 1.

Open the score window if it isn't already and double-click on the script channel for frame 5. This will open the script window with an exitFrame handler filled in for you. Type into the frame to make the movie loop here. We will be adding more to this script later.

The next thing we have to do is set the field member to clear its content when the movie starts so the user is presented with a blank document. You don't absolutely have to do this, but it's easier than trying to remember to manually clear the field member before you publish the movie. There are a couple of ways to do this. You can write a behavior and attach it to the script to clear the cast member on beginSprite. Or, do it on a movie script on prepareMovie that runs every time the movie is run. We'll be using on prepareMovie for some other stuff later so let's use that.

Open the script window and click on the + to insert a new script. Type in:

```
on prepareMovie

  member("content").text = ""

end
```

Open the Script tab in the Property Inspector and make sure the Type drop down menu is set to Movie.

Now if you run the movie, you'll be able to type into the field. Then, stop and restart the movie and you'll see the field get cleared, leaving you with a fresh new document.

figure 1

```
on mouseUp

  -- Initialize the FileIO xtra.
  myFile = new(xtra "FileIO")

  -- Set the file type to our custom file code.  This code is platform specifi
  if _system.environmentproplist.platform contains "macintosh" then
    myFile.setFilterMask("minW MnWd")
  else
    myFile.setFilterMask("MiniWord Files, *.minw")
  end if

  -- Display the Save As dialog box, which returns the whole path
  -- to the document the user is saving.
  filePath = myFile.displaySave("Save document as...", "Untitled 1")

  -- Check to make sure the user didn't hit Cancel.
  if filePath <> "" then
    -- Create the file.
    myFile.createFile(filePath)

    -- Now that it's created we must open it.
    myFile.openFile(filePath, 0) -- 0 means we're opening in Read/Write mode.

    -- Store the data we're going to save in a string.
    saveText = member("content").text

    -- Write the data to the file.
    myFile.writeString(saveText)

    -- Set the creator codes.
    if _system.environmentproplist.platform contains "macintosh" then
      myFile.setFinderInfo("minW MnWd" )
    end if

    -- Don't forget to close the file.
    myFile.closeFile()

  end if

end
```

figure 2

```
on mouseUp

  -- Initialize the FileIO xtra.
  myFile = new(xtra "FileIO")

  -- Set the file type to our custom file code.  This code is platform specific.
  if _system.environmentproplist.platform contains "macintosh" then
    myFile.setFilterMask("minW MnWd")
  else
    myFile.setFilterMask("MiniWord Files, *.minw")
  end if

  -- Display the Open dialog box, which returns the whole path
  -- to the document the user is saving.
  filePath = myFile.displayOpen()

  -- Check to make sure the user didn't hit Cancel.
  if filePath <> "" then

    -- Now that it's created we must open it.
    myFile.openFile(filePath, 1) -- 1 means we're opening in Read mode.

    -- Write the data to the file.
    savedText = myFile.readFile()

    -- Display the text on the stage.
    member("content").text = savedText

    -- Don't forget to close the file.
    myFile.closeFile()

  end if

end
```

Save The Document

Now that we have a document, we need to be able to save it. Create a push button by selecting Insert > Control > Push Button. A blank button appears on the stage with a flashing cursor. Type in Save As and move the button to the bottom left area of the stage below the content field. Select the button cast member in the Cast window and click on the little script icon either in the Cast window or in the Member tab of the Property Inspector. This will open the script window and create a script attached to the member with a blank on mouseUp handler. Type in the code from Figure 1.

This handler makes use of the FileIO xtra that comes with Director, which is used to read and write text files with Lingo. Any xtra that reads and writes to the user's hard drive can be used in its place, such as vList, BinaryIO, and others.

The first thing the handler does is create an instance of the FileIO xtra to use. Then it sets the filter mask to only allow the user to work with documents created with our program. It does this by defining our custom document type. Our document, although it will be nothing more than a text file, will be a "MiniWord Document." We can't just use a .txt file since both Macintosh and Windows already have programs to deal with those kinds of documents. So, our documents will have a .minw extension on Windows, and a 'minW' file type on the Macintosh. (More on that in a moment.)

Note that this is platform-specific code, so I have an if...else statement to branch out to whichever platform is hosting the Director movie. This also allows you to code once and publish for both platforms without having to rewrite any of your Lingo. (Syntax note: The _system.environmentproplist.platform call is the new syntax introduced in DirectorMX2004. If you're still using DirectorMX, you will need to replace this item with the platform in order to make the code work.)

Next, the handler calls displaySave, which pulls up the standard OS save dialog box so the user can select where to save the file and what to name it. This function returns the full path to the document the user wants to save, or an empty string (Macintosh) or <Void> (Windows) if the user hits Cancel. The next thing we do is make sure the user didn't hit Cancel. If filePath is not an empty string then we can proceed. We then create the file, open it, store the string we got from the Content member, write to the file, and close it.

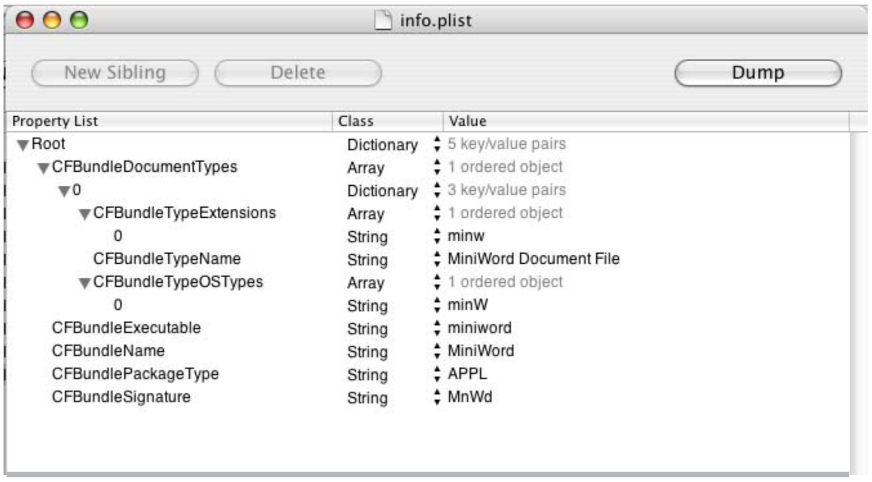
On Windows, file types are specified by a file extension such as .txt for text files and .doc for Microsoft Word files. These are tracked by the Windows Registry, which associates these file extensions with the proper application and is why we set the filter mask above. If the user forgets to add .minw to the file name, the system will take care of it, ensuring that the file keeps its program association. We haven't actually edited the Windows Registry yet, so right now the extension is meaningless, but we'll take care of that soon.

On the Macintosh file, types and associations are traditionally marked in the file itself in a section called the Resource Fork. The file will have a four-character identifier called a creator code. It is traditionally written in single quotes and tells the operating system which application it should use to open the document. DirectorMX2004's creator code is "MDO3". (Interesting that it's not "MD04"). You can pick any four-character code for your application. I chose "MnWd" for our MiniWord application. All ASCII characters are acceptable, including spaces and special characters. So, if you want to make your application's code '#$!' that is perfectly fine.

However, there are some rules, the most important of which is that Apple has reserved all lowercase-only codes for internal use. This is why my application code is "MnWd" and not "mnwd". And although it's not required, Apple suggests that you register your application code with them at their developer site (http://developer.apple.com/datatype/creator-code.html). This is a good idea to make sure nobody else is already using your code. They used to require that you register your file-type code as well, but they no longer ask for that. The file-type code is the same thing – a four-character code that identifies the kind of file it is. Text files are identified with "TEXT"; JPEG files are marked as "JPEG." This is why JPEGs created with Photoshop will reopen in Photoshop when you double-click on them instead of opening in your Web browser, as they do on Windows. Our MiniWord application will create "minW" files.

Run the movie, type something into the content field, and hit Save to make sure the document saves as expected.

figure 3



Property List	Class	Value
▼Root	Dictionary	5 key/value pairs
▼CFBundleDocumentTypes	Array	1 ordered object
▼0	Dictionary	3 key/value pairs
▼CFBundleTypeExtensions	Array	1 ordered object
0	String	minw
CFBundleTypeName	String	MiniWord Document File
▼CFBundleTypeOSTypes	Array	1 ordered object
0	String	minW
CFBundleExecutable	String	miniword
CFBundleName	String	MiniWord
CFBundlePackageType	String	APPL
CFBundleSignature	String	MnWd

figure 4

```
  <?xml version="1.0" encoding="UTF-8" ?>
  <!DOCTYPE plist (View Source for full doctype...)>
- <plist version="1.0">
- <dict>
  <key>CFBundleDocumentTypes</key>
- <array>
- <dict>
  <key>CFBundleTypeExtensions</key>
- <array>
  <string>minw</string>
  </array>

  <key>CFBundleTypeName</key>
  <string>MiniWord Document File</string>
  <key>CFBundleTypeOSTypes</key>
- <array>
  <string>minW</string>
  </array>

  </dict>

  </array>

  <key>CFBundleExecutable</key>
  <string>miniword</string>
  <key>CFBundleName</key>
  <string>MiniWord</string>
  <key>CFBundlePackageType</key>
  <string>APPL</string>
  <key>CFBundleSignature</key>
  <string>MnWd</string>
  </dict>

  </plist>
```

Open The Document

Insert another push button and name this one Open. Drag it down to the lower right area of the stage next to the Save As button. Select the cast member and click on the script icon in either the script window or the Property Inspector. Type in the code as shown in Figure 2.

This code is very similar to the Save As code. The main difference here is that instead of calling displaySave, we call displayOpen, which prompts the user to select a file and returns its path. And again, if the user hits Cancel, the path is an empty string. If not, we open the file specified by the user, read the whole file, store the text we just read in the Content member and close the file. That's it. Here the setFilterMask handler only displays MiniWord files for the user to select. This way your program doesn't try to open Word or Photoshop documents.

The next step is to associate our file format with the application. This has to be done separately for each platform.

The Mac Package

A package is simply a folder with a certain hierarchy of sub-folders and files that define how an application works. When the root folder is named with a .app extension the folder takes on the application's icon and acts like the application itself. Since we have to make the package manually the process is fairly involved, so here are step by step instructions for creating your package.

1. Create a folder called MiniWord Mac. This will be the folder that will act as the application later.
2. Open the MiniWord Mac folder and create a folder inside it called Contents. The Contents folder will contain four items. The first is a text file that contains the creator code and file type for our application. Applications always have a file type of "APPL." Since our MiniWord application saves files in text format we can just use that to generate this file.
3. In Director, play the movie and type APPLMnWd into the content field. That is the standard application file type, and our custom creator code. Click Save As, name the file PkgInfo (with no extension), and save it in the Contents folder.
4. Create a folder in the Contents folder called MacOS.

Advertising Index

Advertiser	URL	Phone	Page
2004 RCAs	www.sys-con.com	888-303-5282	57
ActivePDF	www.activePDF.com		11
CF_Underground VI	www.cfconf.org/cf_underground6/	301-424-3903	Cover 3
CFDynamics	www.cfdynamics.com	866-233-9626	5
FuseTalk	www.fusetalk.com	866-477-7542	48
HostMySite.com	www.hostmysite.com/mxdj	877-248-4678	15
Information Storage+Security	www.ISSJournal.com	888-303-5282	27
Interakt	www.interaktonline.com		6
IT Solutions Guide	www.sys-con.com	201-802-3021	31
Macromedia	www.webpublishingsystem.com		Cover 2
Max Macromedia	www.macromedia.com/go/max		Cover 4
Nidus Corp.	www.brainstormer.org		37
Seapine Software	www.seapine.com	888-683-6456	3
ServerSide	www.serverside.net	888-682-2544	9
Web Services Edge 2005	www.sys-con.com/edge	201-802-3066	51

5. Create another folder in the Contents folder called Resources. We actually won't use the Resources folder for this demonstration, but it's a good idea to create it anyway. This is where icon files and other related items would go.
6. Find the Director application icon in the Finder. If you have Director in the dock, you can simply click and hold on the icon and select Show In Finder. Otherwise, navigate through your Applications folder until you find the Director application icon. It's time to take a look at Director's package. Control-click (or right-click if you have a two-button mouse) on the application icon and select Show Package Contents from the pop-up menu that appears. Open the Contents folder and then open the MacOS folder.
7. Copy the following four files:
   - MacromediaRuntimeLib
   - ProjLib
   - IMLLib
   - DPLib

   and paste them into your MacOS folder. These are the Shockwave library files. In order for your projector to run inside a Package, it must be a Shockwave projector. Shockwave projectors require that the user have the Shockwave plug-in installed on their system in order to run. Placing these files in your MacOS folder (next to where the projector will be) ensures that your projector will run even if the user doesn't have Shockwave installed.
8. (Optional) In the MacOS folder, create a folder called xtras. Place all the xtras you used in your projector into this folder. You'll need to copy them from the Director application folder. If you don't do this, make sure all the xtras used in your movie are marked to be included in the Projector. You'll need to explicitly add the FileIO xtra to your movie under Modify > Movie > Xtras.
9. Publish your Director movie. In the Publish Settings window in the Projector tab, make sure that Player Type is set to Shockwave. Save the projector to another folder elsewhere, not into the MiniWord App folder. Once we make this folder into a package, you won't be able to save to it directly and we'll be making more changes to the Director file later. So save it somewhere else, and then move it into the MacOS folder. This way, you'll know exactly where to find it when we publish it again. Be sure to copy your published projector into the MacOS folder.

In the next step, we will create a file called info.plist. This is an XML file that explains how the application is set up and describes many of its components. The easiest way to create this file is with Apple's Property List Editor application, which you can find on the Developer Tools CD that comes with OSX. If you don't have this, you can use a text editor to create the file, but it's very easy to make mistakes doing it that way, and very difficult to find them to fix them when something doesn't work. A finished info.plist file is shown in Figure 3. If you don't have Property List Editor installed, skip down to the next paragraph and copy the XML data below exactly as you see it using any text editor.

10. Launch Property List Editor.
    - Click New Root.
    - Turn down the Root arrow and click New Child.
    - Type in CFBundleDocumentType and set the class to an Array. This defines the file type.
    - Turn down the CFBundleDocumentTypes arrow and click New Child. The program automatically inserts an item numbered 0. Change its class to Dictionary.
    - Turn down item 0 and click New Child.
    - Type in CFBundleTypeExtensions and change its class to an array.
    - Turn down the arrow for CFBundleTypeExtensions and click New Child.
    - In the value field for item 0 type in minw, all lower case. This defines the file extension for our documents.
    - Close the CFBundleTypeExtensions arrow and click New Sibling.
    - Type in CFBundleTypeName and in the value field type in MiniWord Document File.
    - Click New Sibling and type in CFBundleTypeOSTypes and set it to an array.

- Turn down the CFBundleTypeOSTypes arrow and click New Child.
- In the value for item 0 type minW. This is our actual document type as stored in the Resource Fork.
- Close the CFBundleDocumentTypes arrow and click New Sibling
- Type in CFBundleExecutable and in the value field type MiniWord.osx, or whatever you named your projector. Make sure the name matches exactly.
- Click New Sibling, type in CFBundleName, and in the value field type in MiniWord.
- Click New Sibling, type in CFBundlePackageType and in the value field type in APPL.
- Click New Sibling, type in CFBundleSignature, and type in MnWd. This tells the operating system that this is our application's creator code.

Many other items can be defined here, but this is all that is necessary to make our project work. The final file should look like Figure 1. Make sure it matches, and then save the document as info.plist and place it in the Contents folder.

If you don't have Property List Editor installed, you can create the file manually with a text editor. Type in the XML exactly as you see it in Figure 4.

11. Navigate back to the MiniWorld Mac folder and rename it to MiniWorld Mac.app. You will get an alert to warn you about changing the extension. Click Add to add the extension. If everything is set up right, the folder icon will change to a generic application icon. Or, if you published your projector with a custom icon the folder will take on that icon.

At this point, if you double-click on the MiniWorld Mac folder it will no longer open to reveal its contents, but instead will launch your projector. Additionally, if you double-click on a saved document that MiniWord created, it will also launch the projector, but it still won't open the file. We still have to add code to handle that.

The Windows Registry

Associating our file types with the Windows operating system isn't nearly as involved as it is on the Macintosh. However, it's also not something that can be done natively in Director or with another program. The first time your program runs, it needs to edit the user's Registry file to make everything work. In order to do this, we will need a third party xtra. This example makes use of the BuddyAPI xtra (www.mods.com.au), but any xtra that can work with the user's registry will work. Check the documentation for the specific syntax you'll need with other xtras.

Ideally, your program will only write to the user's Registry once. That means your program would need to keep track of the fact with a preferences file or some similar method. Saving preferences files is beyond the scope of this article so for now we will do all our work on prepareMovie. This will run every time the program is launched, which is not ideal but will do for the purposes of instruction.

In your movie script in the on prepareMovie handler, add a call to a handler called editWindowsRegistry so the whole handler now looks like this:

```
on prepareMovie

  member("content").text = ""


editWindowsRegistryEditWindowsRegistry
()

end
```

Now we need to define this handler. Type in the code as shown in Figure 5.

The next time your movie is run, it will modify your Registry to associate the file types with your application. At this point, double-clicking on a saved document will launch your projector but will not open it for the same reason as for the Mac – we haven't yet added the code to handle that.

Opening The Files That Were Double-Clicked

Now that our program is being launched, when the user double-clicks

figure 5

```
on editWindowsRegistry

  -- This code uses BuddyAPI
  documentApplicationPath = the moviePath & "miniword.exe %1" -- The name of our projector.

  baWorked = baWriteRegString(".minw", "", "MiniWordDocument", "HKEY_CLASSES_ROOT") -- Define the file extension.
  baWorked = baWriteRegString("MiniWordDocument\shell\open\command", "", documentApplicationPath, "HKEY_CLASSES_ROOT") -- Define the open command.

end editWindowsRegistry
```

figure 6

```
on openThisDocument me

  -- Initialize the FileIO xtra
  myFile = new(xtra "FileIO")

  -- Determine the path to the file from the commandLine
  oldDelimiter = the itemDelimiter
  the itemDelimiter = "'"
  fullCommandLine = the commandLine
  itemWeWant = fullCommandLine.items.count-1
  filePath = fullCommandLine.item[itemWeWant]
  the itemDelimiter = oldDelimiter

  if filePath <> "" then

    -- Now that it's created we must open it.
    myFile.openFile(filePath, 1) -- 1 means we're opening in Read mode.

    -- Write the data to the file.
    savedText = myFile.readFile()

    -- Display the text on the stage.
    member("content").text = savedText

    -- Don't forget to close the file.
    myFile.closeFile()

  end if

  -- Reset myCommandline so we don't get stuck in a loop.
  myCommandline = the commandLine

end
```

```
ontains "Windows" then
 -- Defines file type extension.
niWordDocument" -- Defines file type definition.
 "\shell\open\command"

 & "miniword.exe %1" -- The name of our projector.

e .mnwd registry
ulatorDocument") -- Assigns value to .mnwd

es MiniWordDocument key

reates shell/open/command key
 documentApplicationPath) -- Tells windows which application to launch.
```

“Director projector can create any type of document and write it to the user’s hard disk”

on a file, we need to figure out which file the user clicked on and open it. We can do this with an undocumented system variable called the commandLine. This variable takes the form of an apostrophe-delimited string. The second-to-last item in the string is the path to the file the user clicked on. The string ends with an empty item, which is why we want the second-to-last item.

Not only does the program launch when the user double-clicks on a document, but if the application is already running, it is brought to the front. So our program needs to periodically check the commandLine to see if it has changed. If it has, we need to open the new document. To do this, we'll create a global variable called myCommandLine and initialize it to an empty string. So let’s modify our on prepareMovie handler to accommodate this.

```
global myCommandline

on prepareMovie

  member("content").text = ""

  myCommandline = ""

  editWindowsRegistry

end
```

If the user launched your program normally, then the commandLine will return an empty string, which matches myCommandline. So all we have to do is see if the commandLine has changed, and since we have to do this periodically, we’ll just do it on the exitFrame handler we have running on frame 5.

```
global myCommandline

on exitFrame me

  -- Check to see if the commandLine
has changed.

  if the commandLine <> myCommandline
then

  -- Open the file specified by the
commandLine.
  openThisDocument()

  end if

  go the frame

end
```

Checking the status of the commandLine of every frame is usually not necessary. And certainly, if you’re authoring a game, you don’t want to take a speed hit by doing an unnecessary check every frame. I'd suggest only checking on the title screen or some other relatively calm screen.

The openThisDocument handler will now get called whenever the commandLine changes, which happens whenever a user double-clicks on a saved document that is associated with your application. It’s structure is very similar to the code we attached to the Open button, but instead of prompting the user to select a file it simply uses the commandLine to find the file. In the movie script, type in the code from Figure 6.

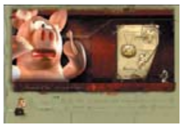
This code is the same as the open code, except for the section at the top that gets the path from the commandLine. The way that works is that it first saves what the current itemDelimiter is in a local variable so we can restore it later. Then, it sets the itemDelimiter to an apostrophe and reads the entire commandLine. Then, it gets the index number of the second-to-last item in the list and uses that to determine the filepath of the document that was double-clicked. Finally, it restores the itemDelimiter to its default value.

This is the last step in making the program work. Publish your movie again. If you’re on a Mac, be sure to place the projector in the MacOS folder in your package. Double-click a saved document and it will launch your projector and open the document. Double-click another file and it will bring your program to the front and open that one.

Conclusion

You have seen how to make your projector behave like a professional application. All you have to do now is apply this technique to your own Director projects, and then sit back and wait for the shareware fees to come rolling in.

Tom Rockwell studied illustration at Rochester Institute of Technology and taught himself how to program in his spare time in an effort to create his own video games. He eventually settled on Director as his development environment of choice for video games and how has a successful career as a multimedia developer for a training company in New Jersey. He still programs video games in his spare time and sells them through his shareware company FIDIM Interactive, LLC. (www.fidim.com)
spice@suddendeath.org



How to Build a www.fernando-nieto.com

This Web site is an apology to velvet, meat, and pornography.

1. Grow up in Colombia, the most insane, surrealistic country.
2. Move to France (the paradise of royal-pork-butchery).
3. Give a value to such a thing.
4. Make a lot of paintings and drawings of it.
5. Create a 3D character (with Cinema4D) that represents your own values (a pig).
6. Create a nice layout with textures, shadows, and effects (use Photoshop and Illustrator).
7. Shake vigorously into Flash MX to make it move and live.

Bon appétit!!!

$99
per person

CF_Underground VI

October 31st 2004
10:00am - 5:00pm
New Orleans, LA
Check website for exact location and details

www.cfconf.org/cf_underground6/

Fill your brain with lots of cool CF programming meta tips & tricks, then relax and have a drink on us!

CF_Underground is the coolest pre-MAX event you'll attend during your visit.

Pick the brains of several experts like Simon Horwith, Sandra Clark, Shlomy Gantz, Hal Helms, Michael Smith, and more!

TeraTech Event
www.teratech.com
301.424.3903