

MX

developer's journal



THE LEADING MAGAZINE FOR MACROMEDIA MX DEVELOPERS & DESIGNERS

volume 2 issue 6 2004 www.mxdj.com

macromedia

The New Age of

FLASH

DREAMWEAVER
Avoiding CSS Pitfalls

FIREWORKS
Call in the Specialist

FREEHAND
The Tricks to Tracing

COLDFUSION
Freaks & Geeks

DIRECTOR
Building Tooltips

The Most Powerful Authoring Enhancement Ever Seen



\$9.99US \$9.99CAN 07>

0 09281 01597 0



Avoiding CSS Pitfalls
Hacks for frustrated designers
by zoe gillenwater



Custom Tools
A new world in the Flash IDE
by keith peters



Call in the Specialist
Fireworks MX does a lot; these tools do the rest
by charles e. brown

14



22

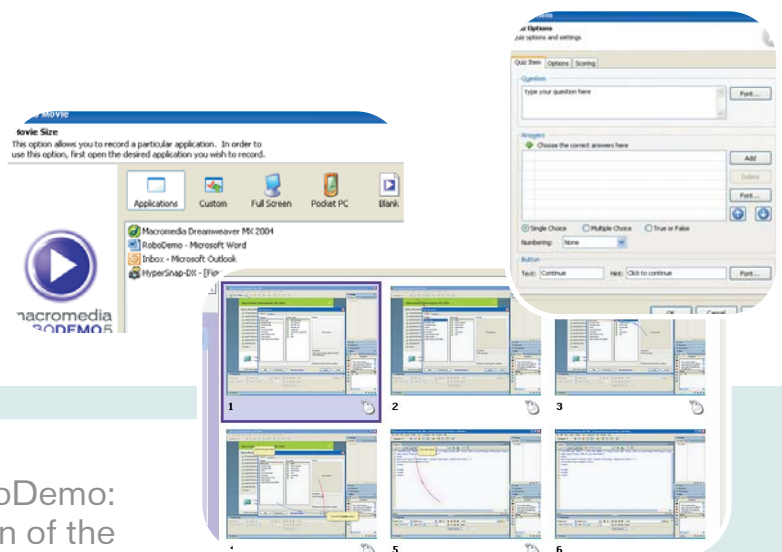


30



20

NeXTensio2
by InterAKT
Create databases in next to no time
reviewed by russell stearman



7

RoboDemo:
Lynchpin of the
E-Learning Market
*The future of how we
learn software*
by charles e. brown



The Tricks to Tracing
*Save your energy for
 designing and drawing*
 by ron rockwell



Freaks & Geeks
The saga continues
 by tom green



Building Tooltips
What does this button do?
 by irv kalb

34



42



50



on the cover

the whole process of creating tools is fairly complicated – "Custom Tools: A new world in the Flash IDE" aims to make the process a bit more clear. Hopefully it will inspire you to begin making your own...and sharing them.



12

news
NewsFlash

29

xile
Cartoon
 by louis f. cuffari

58

vanguard
Church of Fools
 by specialmoves

Dreamweaver Website Development

MANY needs - ONE solution



MX Kollection for ColdFusion and PHP

Create powerful dynamic lists

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

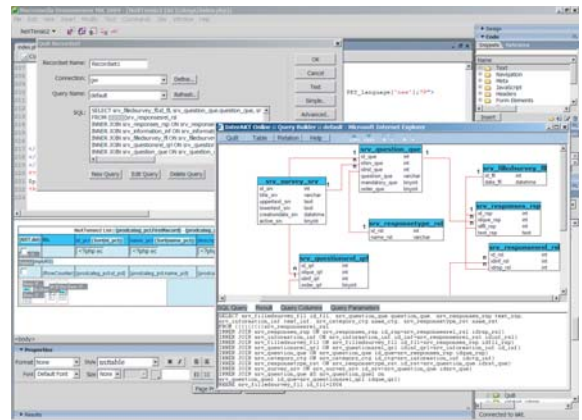
Build unified add/modify forms

- 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 queries across multiple tables quickly

... and many more



The MX Kollection is a **suite of extensions** designed to **change the way you create dynamic web applications** with Dreamweaver MX.

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.

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

Download the demo and see the features and benefits of our extensions:

<http://www.interaktonline.com/>

MX Kollection - Professional web tools



RoboDemo: Lynchpin of the E-Learning Market

The future of how we learn software

by Charles E. Brown

There is a little bit of irony going on here: this is an article about a software package that will go a long way toward rendering printed technical media, such as this journal, obsolete.

Last year Macromedia purchased the company eHelp and, as a result, acquired RoboHelp, RoboInfo, RoboHelp for Framemaker, RoboPDF, and RoboDemo. This purchase, in my opinion, has made Macromedia a major player in the e-learning/e-publishing market. In this article, we are going to focus on the lynchpin of that market: RoboDemo.

RoboDemo is a way to create highly interactive software simulations. This means that you can record the steps necessary (mouse movements, key

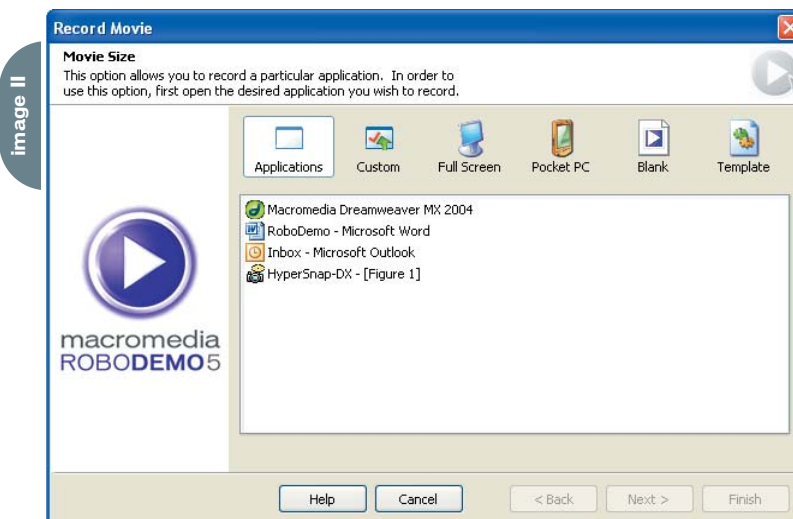
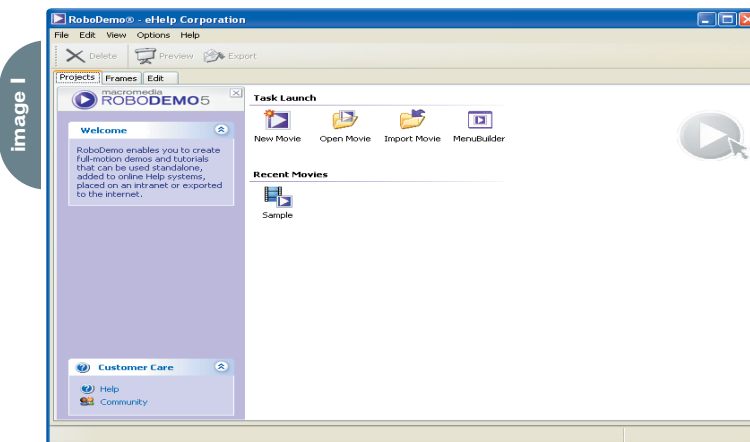
strokes, menu options, etc.) to accomplish a task in a video. However, as you will soon see, that is not the end of the possibilities.

Creating a Movie

As a simple example, let's assume that I want to demonstrate how to start a new HTML document in Dreamweaver. I would begin by opening Dreamweaver and RoboDemo. Image I shows the opening RoboDemo screen.

A screen will come up allowing you to select the open program you want to record the video in. This screen is shown in Image II.

After selecting, in this case, Macromedia Dreamweaver MX 2004, a



Group Publisher Jeremy Geelan
Art Director Louis F. Cuffari

EDITORIAL BOARD
Dreamweaver Editor
Dave McFarland
Flash Editor
Jesse Warden
Fireworks Editor
Kleanthis Economou
FreeHand Editor
Louis F. Cuffari
Ron Rockwell
ColdFusion Editor
Robert Diamond

INTERNATIONAL ADVISORY BOARD
Jens Christian Brynildsen **Norway**,
David Hurrows **UK**, Joshua Davis **USA**,
Jon Gay **USA**, Craig Goodman **USA**,
Phillip Kerman **USA**, Danny Mavromatis **USA**,
Colin Mook **Canada**, Jesse Nieminen **USA**,
Gary Rosenzweig **USA**, John Tidwell **USA**

EDITORIAL
Executive Editors
Gail Schultz, 201 802-3043
gail@sys-con.com
Jamie Matusow, 201 802-3042
jamie@sys-con.com

Editors
Nancy Valentine, 201 802-3044
nancy@sys-con.com
Jennifer Van Winckel, 201 802-3052
jennifer@sys-con.com

Assistant Editor
Torrey Gaver, 201 802-3041
torrey@sys-con.com

Technical Editors
James Newton • Sarge Sargent

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

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

Editorial and Advertising Offices
Postmaster: Send all address changes to:
SYS-CON Media
135 Chestnut Ridge Rd.
Montvale, NJ 07645

Worldwide Newsstand Distribution
Curtis Circulation Company, New Milford, NJ

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

Promotional Reprints
Kristin Kuhnle, 201 802-3026
kristin@sys-con.com

Copyright © 2004
by SYS-CON Publications, Inc. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information storage and retrieval system, without written permission.

MX Developer's Journal (ISSN#1546-2242) is published monthly (12 times a year) by SYS-CON Publications, Inc., 135 Chestnut Ridge Road, Montvale, NJ 07645.

SYS-CON Media and SYS-CON Publications, Inc., reserve the right to revise, republish, and authorize its readers to use the articles submitted for publication. MX and MX-based marks are trademarks or registered trademarks of Macromedia, in the United States and other countries. SYS-CON Publications, Inc., is independent of Macromedia. All brand and product names used on these pages are trade names, service marks or trademarks of their respective companies.

screen comes up that allows you to pick the hot keys you want to use to start and stop the recording process. There are suggested defaults which, in most cases, work fine.

You are then taken to the application you are recording with a panel, as shown in Image III, in the upper left corner.

At this point, there are adjustable graphic handles that will allow you to block out the area of the application in which you want to record. In most cases, however, you will use the default of the full screen.

After selecting Record, you simply go through the steps necessary to accomplish the task you want to show. Once you complete the steps, press the End key and RoboDemo will build the video. When completed, you will be returned to

RoboDemo with thumbnails of the key frames of the video. You can see this in Image IV.

You can now preview the video in either RoboDemo's own player, or in your Web browser. The one thing that will strike you is that RoboDemo automatically added some captions for you. – an example of this is shown in Image V.

These captions are fully editable for both content and style. In addition, you can create your own captions and place them where they are necessary.

E-Learning

If you are on a sales team and want to demonstrate the features of your software, you have a powerful tool. However, RoboDemo now goes one step further. If you are doing this demo as part of an e-learning project, RoboDemo will now allow you to create an interactive situation that will allow users to try the steps themselves. You can even program the responses to right and wrong choices.

To create the interactivity, all you need to do is double-click on the thumbnail frame to which you want to attach the interactivity. Once open, you select Insert>Click Box. You will be brought to the dialogue box shown in Image VI.

Here you can decide how the movie will respond to right and wrong responses. Notice that you can add Success and Failure captions. Also, if you are comfortable with programming, you can tie the movie into a JavaScript file.

If you are in an e-learning environment, chances are you will need to test the student and track the scores. You can insert Quiz Frames as shown in Image VII.

Here you can decide the questions asked, the type of answers, scoring, and responses to right and wrong answers.

To aid in the learning process, RoboDemo even allows for branching. For instance, let's assume someone answers a question incorrectly about attaching a template in Dreamweaver. You could branch into a movie that would review those steps in greater detail. If the user answered the question correctly, you could simply move on to the next topic.

If all this is not enough, you will notice in Images VI and VII that there is a tab to help decide scoring for the response. This could be important for

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, 201 802-3025
 kristin@sys-con.com
 Beth Jones, 201 802-3028
 beth@sys-con.com
 Dorothy Gil, 201 802-3024.3
 dorothy@sys-con.com

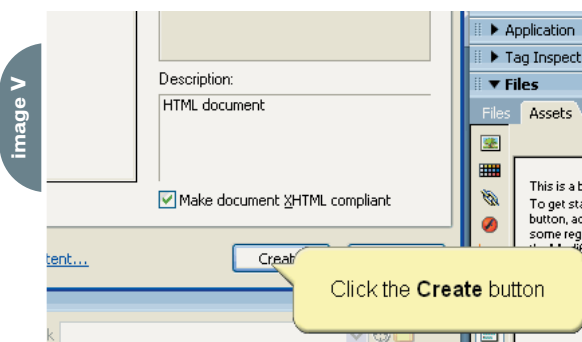
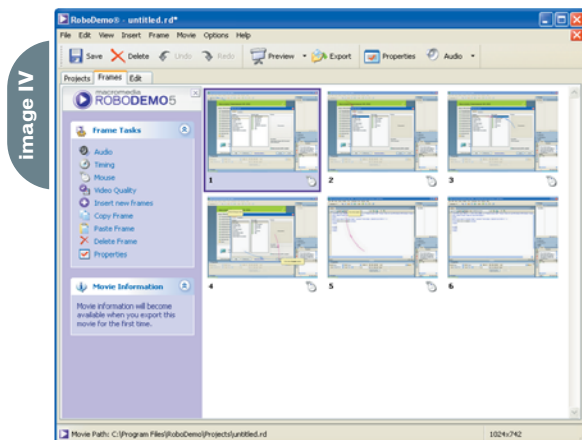
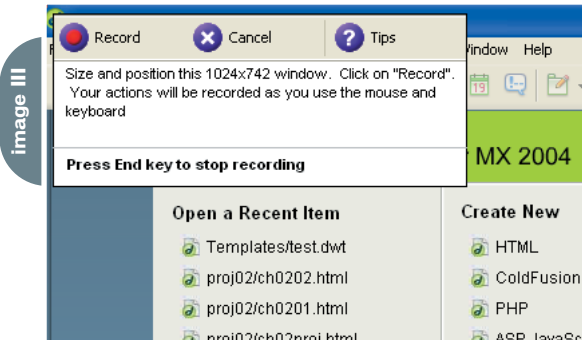
PRODUCTION
Production Consultant
 Jim Morgan, 201 802-3033
 jim@sys-con.com
Lead Designer
 Louis F. Cuffari, 201 802-3035
 louis@sys-con.com
Art Director
 Alex Botero, 201 802-3031
 alex@sys-con.com
Associate Art Director
 Richard Silverberg, 201 802-3036
 richards@sys-con.com
Assistant Art Director
 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
 Lin Goetz, 201 802-3045
 lin@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

EVENTS
President, SYS-CON Events
 Grisha Davida, 201 802-3004
 grisha@sys-con.com
Conference Manager
 Lin Goetz, 201 802-3045
 lin@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



BlueDragon [6.1]

RELEASED!



RUN YOUR CFML:

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

On the platform, web server,
and operating system of
your choice.



Get BlueDragon hosting:

CFDynamics is now offering BlueDragon hosting!
Find out more by visiting: www.cfdynamics.com/bluedragon

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. Come join CFDynamics and New Atlanta in expanding the ColdFusion horizon!

CFDynamics

A Division of Konnections Inc.



POWERFUL HOSTING PLANS

- FREE SQL server access
- FREE account setup
- UNLIMITED email accounts
- GENEROUS disk space / data transfer
- 30 day MONEY-BACK GUARANTEE
- GREAT VALUE!

RELIABLE NETWORK

- 99.99% average uptime!
- State-of-the-art data center 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
ACCOUNT TODAY!**

Use promo code: **MXDJ04F**
and receive a **FREE T-SHIRT!**



WWW.CFDYNAMICS.COM

866 - CFDYNAMICS

866-233-962-6427

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.



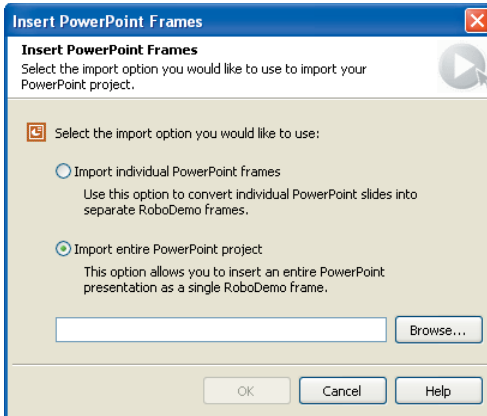
ColdFusion

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

image VI

image VII



corporate and academic environments. As a matter of fact, RoboDemo is SCORM- and AICC-compliant in order to integrate with a Learning Management System.

If you are in a corporate environment and need to analyze scores, RoboDemo will generate a manifest that will integrate with an XML file created for this purpose.

I work with Macromedia's Authorware often to create learning programs. Integration of RoboDemo into Authorware is nearly seamless. However, I got my best results when I either exported the movies as SWF files or incorporated them into larger Flash presentations.

PowerPoint Integration

One of the features I found especially interesting was how RoboDemo integrated with Microsoft PowerPoint.

You can easily place PowerPoint slides between existing frames of the RoboDemo movie. This helps to make the presentation run smoother by allowing you to add slides with descriptive text. By simply selecting Insert>PowerPoint Frame you are presented with the dialogue box shown in Image VIII.

Here you can import a single slide or an entire PowerPoint presentation. This can be handy in both demonstration and e-learning situations.

Nowhere is this integration more noticeable than in RoboDemo's MenuBuilder feature.

In many situations you may want to break your presentation down to a series of shorter movies rather than one long movie. MenuBuilder employs Microsoft

PowerPoint to create a menu so that the user can choose what movie/demonstration they want to see. The menu can then be incorporated into the project.

As an interesting note, I recently did a RoboDemo tutorial with 10 movies and a PowerPoint menu to connect them together. I then exported the menu as an EXE file and put the whole tutorial on a CD for distribution.

If you are not a PowerPoint user, don't worry. RoboDemo has a built-in means of creating text slides with a feature called the Text Animator. This is shown in Image IX.

Here we can create simple text slides and transition effects for the text. While, in my opinion, it does not have the flexibility of PowerPoint, it does get the job done quite nicely. I found it especially helpful in creating an Introduction and Exit for the presentation.

Of course, the best solution of all is to integrate your movies within Flash. Let's take a quick look at that, as well as various other output options.

Output

RoboDemo offers a variety of outputs. As we just mentioned, you can export as an EXE file. However, if you want to really create a very professional looking presentation, you might want to consider exporting the movies as Flash SWF file. By doing that, you can incorporate your movies within a larger Flash presentation. Using the power of the Flash timeline allows you to integrate sound, movies, text, and transitions seamlessly. In my opinion, after integrating RoboDemo and Flash, you will not want to go any other way. The presentational possibilities are nearly endless.

Unfortunately, if you want to create Flash editable FLA files, you will need to purchase an add-on module for \$99.00.

During output you can also set an expiration date. This option will prevent the movie from being seen past a certain date. This can be handy for versioning control.

You can also e-mail your movie from

within RoboDemo. You can email it as a SWF, EXE (Windows or Linux), or HQX for the Macintosh.

While exporting, you can also select from a number of playback control styles or, if you want, design your own control graphics using BMP graphic images.

If bandwidth size is a concern, you can quickly get the statistics of your movie with the Bandwidth Monitor shown in Image X.

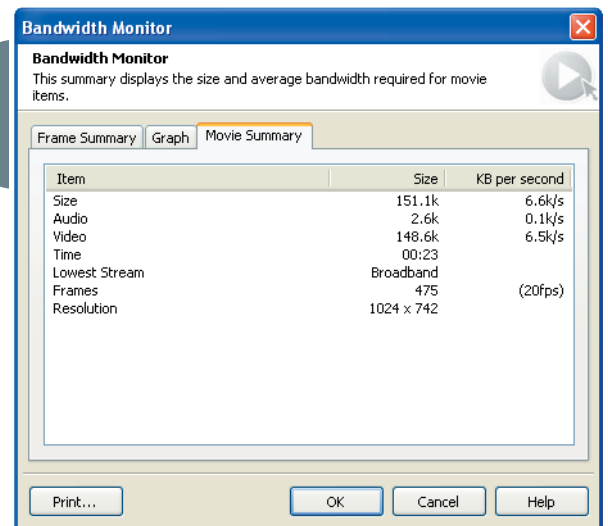
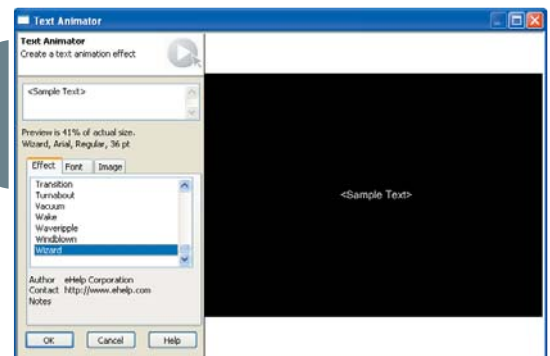
Here you can analyze your movie either textually or graphically. In addition, you can break the analysis down to individual frames and components.

Conclusion

Due to space constraints we could only touch on the main points of this remarkable and feature-rich program. Whether for sales presentation or instructional design, this program is going to change how you design your solutions.

Macromedia has a 15-day trial version available for download on its site. Trust me, once you try it you will be sold. ☺

Charles E. Brown is the author of Fireworks MX 2004 Zero to Hero and Beginning Dreamweaver MX 2004. He also contributed to The Macromedia Studio MX Bible. charles@charlesebrown.net



NewsFlash

Macromedia Flash Player 7 for Linux Now Available

Macromedia has announced the immediate availability of Macromedia Flash Player 7 for Linux. This new version of Macromedia Flash Player, the leading rich Internet client, offers improved performance, security, and powerful new development capabilities. Macromedia Flash Player is bundled with Linux operating systems distributed by Novell, Red Hat, Sun Microsystems, and Turbolinux.

"Macromedia is committed to the Linux platform and wants to make sure Linux users can experience the proven effectiveness of Flash technology on their platform of choice," said Jeff Whatcott, vice president of product management, Macromedia. "Developers can now take advantage of the breakthrough performance and advanced capabilities of Flash Player 7, enabling them to deliver a consistent cross-platform experience."

Flash Player 7 offers increased performance and ensures a consistent cross-platform experience. With support for Cascading Style Sheets (CSS), Flash Player enables developers to blend HTML and Flash with consistent formatting. New support for Simple Object Access Protocol (SOAP) Web services connectivity allows developers to create rich Internet application user interfaces that handle enterprise data in a service-oriented architecture.

Macromedia Positioned in the Visionary Quadrant

Macromedia Breeze is positioned in the "visionary" quadrant in Gartner's 2004 Magic Quadrant report on Web conferencing. With Breeze, Macromedia is delivering the first rich Web communication system, allowing organizations to communicate, collaborate, and train via a sin-

gle, easy-to-use, integrated solution that includes everything from live meetings to on-demand presentations.

Breeze was evaluated based on the "completeness of vision." According to Gartner, enterprises "who desire more than a slide-sharing event should look to visionaries who provide more advanced capabilities." Gartner defines vendors listed in the visionary quadrant as having a clear vision of market direction and are focused on preparing for that, but they can still improve in terms of optimizing service delivery.

The Magic Quadrant is a graphical representation of a marketplace at and for a specific time period. It depicts Gartner's analysis of how certain vendors measure against criteria for that marketplace, as defined by Gartner. Gartner does not endorse any vendor, product, or service depicted in the Magic Quadrant, and does

not advise technology users to select only those vendors placed in the "Leaders" quadrant. The Magic Quadrant is intended solely as a research tool, and is not meant to be a specific guide to action.

CommonSpot Content Server 4.0

PaperThin, Inc., has announced the availability of CommonSpot Content Server version 4.0, the company's flagship Web content management solution. This major release introduces new scalability options, added developer and administrator capabilities, and expanded content creation features. Dozens of enhancements to existing features are also introduced in this release.

Version 4.0 builds on an already rich feature set, which includes 50+ pre-built standard elements. New scalability options in version 4.0 include enhancements to the replication feature and a static content generation module. By separating the process for handling dynamic and static content, a more scalable, reliable, and higher performance site can be realized. This new feature also enables easier incorporation of third party applications, providing support for a broader range of technologies.

For content contributors, new 4.0 features like comprehensive spell check, visual difference, and pop-up calendars help authors create and publish content more accurately. New administrator

tools include the ability to easily customize the page creation interface, the ability to

restrict the type and size of files that can be uploaded by contributors, the ability to copy subsite parameters, and support for international date formats. www.paperthin.com

Macromedia Simplifies Teaching with Technology

The Macromedia Contribute Higher Education Site License has recently been released to provide a higher education solution enabling greater faculty use of technology in teaching, collaboration, and research. This standardized workgroup productivity tool includes a one-year, department-wide site license for Macromedia Contribute 2 for use at school and at home, five perpetual licenses of Studio MX 2004 with Flash Professional, digital learning assets, and professional development resources.

Macromedia Contribute 2 is the easiest way for individuals and teams to update, create, and publish Web content to any HTML Web site. Contribute allows non-technical faculty and staff to update Web content and share class lectures, projects, and research online while maintaining site standards for style, layout, and code. Studio MX 2004 with Flash Professional enables campus Web professionals to create templates that can be used by faculty and staff members with Contribute.

The Macromedia Contribute Higher Education Site License helps colleges and departments turn a course Web site into an interactive teaching resource by putting content contribution and revision directly in the hands of faculty. This solution supports college deans and provosts who want to motivate faculty members to actively integrate technology into their teaching, whether to meet program accreditation requirements, or to demonstrate commitment to teaching excellence and innovation.

Several tutorials are included with the site license digital assets. Among these is a tutorial on how to create e-portfolios with Contribute, and a white paper on the pedagogical and practical value of teaching with e-portfolios. Additional learning resources include narrated multimedia presentations explaining why and how e-portfolios enrich student learning, and suggestions on ways faculty can use e-portfolios to make digital learning experiences more relevant and engaging for their students.

The Contribute Education Site License also includes five copies of Studio MX 2004,

Studio MX 2004 Step-by-Step, and a copy of the Macromedia Faculty Development Guide.

BookFlash: Web Design with Macromedia Studio MX Plus

Web Design with Macromedia Studio MX 2004, written by Eric Hunley and published by Charles River Media, provides beginning designers with a step-by-step process for creating a Web site with the new Macromedia Studio MX 2004. Structured around a good Web development cycle of plan, design, build, test, and maintain, the book starts by planning the site and creating a wire frame with Freehand MX, along with vector graphics for use in Macromedia Flash MX 2004 and Fireworks MX 2004. From there it moves to Flash, where the designer learns how to create dynamic content for use on the Web. Next, Fireworks is used to manipulate graphics and create Web page designs. Then the content is brought into Dreamweaver MX 2004 and the site launched. Throughout the book, the focus is not only on how to use each tool, but how to use them together to create a seamless workflow. www.charlesriver.com

Introducing the new FuseTalk.

Collaboration

Discussion Forums

Flexible Security, 508 Compliance, Offline Capabilities, Reporting, Easy Integration and much more...

1-866-477-7542

2003 CFDJ Readers' Choice Award Winner for Best eBusiness Software & Best Web Application

www.fusetalk.com

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

avoiding CSS pitfalls

CSS has been around for years, but many Web designers still do not think it is ready to be used extensively because of the host of browser rendering inconsistencies that exist. However, by knowing a few CSS hacks and tricks, you can learn how to write code that is cross-browser compatible and allows you to fully separate your content from its presentation.

by zoe gillenwater

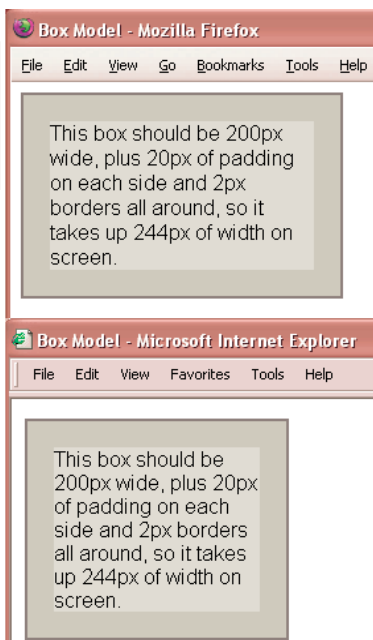


This article assumes you know what CSS is and have some idea how to use it to style Web pages, but is aimed at Web designers who have not yet taken the leap of using it as their primary layout method due to frustrations with browser inconsistencies. It also assumes you know the benefits of CSS-based layout and are eager to make use of it. Many of the hacks I'll go over depend on your document using a doctype that will put it into browsers' "standards rendering mode," so be sure you are using a proper doctype as well. (Dreamweaver MX 2004 uses such doctypes on its HTML and XHTML templates, so you should be good to go. If in doubt, read my article in the last issue of *MXDJ* or check out the CSS Wiki – <http://css-discuss.incutio.com/?page=DoctypeSwitch> – for information about doctype switching.)

Squashing Bugs

Even the most modern browsers have bugs in the way they render CSS styling. Worse still, many people continue to use older, buggier browsers, so achieving a consistent cross-browser design can be frustrating even when you adhere to the best CSS practices. Luckily, you can use the browsers' own bugs against them to fix many of these problems. CSS hacks are code tricks that take advantage of various browsers' bugs or shortcomings to hide certain rules from or feed different rules to browsers that need special treatment. I'll outline some of the most irksome bugs and how to work around them with CSS hacks.

image 1



A word of caution: keep in mind that not everyone supports the use of hacks. Some developers argue that the browser failings we are forced to work around will never be fixed because our hacks create the appearance that everything is working correctly. Hacks can also be dangerous because they rely on browser bugs that may be corrected in future browser releases, rendering your hacks useless or perhaps introducing new problems of their own. However, by keeping our hacks in the CSS instead of the (X)HTML markup, we should be able to change or remove broken hacks in one central file and update an entire site if it becomes necessary in the future. If we're going to create the best experience for our users, hacks are a necessary tool in commercial Web site design – as long as they are used sparingly and thoughtfully with periodic testing to make sure they are still functioning correctly.

Box Model

One of the most common problems with CSS-based layouts is getting all of your columns and boxes to line up and fit together due to Internet Explorer's (IE) faulty implementation of the box model. According to the World Wide Web Consortium (W3C), the assigned width or height of an element refers to its content area only. The padding and borders are then added to this value to arrive at the total area of the element that you see on screen. This means that if you set a div's width to 200 pixels with 20 pixels padding and 2 pixels borders, the total area it will take up is 200px + 20px (left padding) + 20px (right padding) + 2px (left border) + 2px (right border) = 244px. Unfortunately, all CSS-enabled versions of Windows IE before IE6 (in standards mode) use a different box model that counts the padding and borders of a box as part of its assigned width. This means that your div that should take up 244px on the screen only takes up 200px in WinIE5.x because it subtracts the padding and border values from the content area of the box. You're left with a box that looks completely different and has much less room for content in WinIE5.x than it does in IE6 and other current browsers (see Image 1 for a comparison).

In many cases, the box model problem can be worked around without any

hacks. For instance, instead of applying padding to a box to get its content to move away from the edges, you can apply the padding to the content inside the box:

```
div {
    width: 200px;
}
div p {
    padding: 20px;
}
```

But we also wanted 2px borders on our box. You could choose to ignore the four pixel difference they would introduce in WinIE5.x – some layouts don't rely on such precision. Or, you could nest another div inside the first one and apply the borders to the nested div, but this mucks up your markup. What you really want is a way to give WinIE5.x a bigger value than other browsers use, a value that is equal to the total box width, including padding and borders, so that it can happily subtract them without making the box too small. Tantek Çelik developed a box model hack to do just that, and since then several simplified versions have followed. My favorite is the Modified Simplified Box Model Hack (MSBMH) developed by Edwardson Tan:

```
div {
    width: 200px;
    padding: 20px;
    border: 2px solid red;
}
* html div {
    width: 224px;
    width: 200px;
}
```

The first rule gives the correct width, padding, and border values to browsers that correctly implement the box model. The second rule is seen only by IE through the use of the Star HTML Hack. `* html div` tells the browser, "apply this rule to any div element that is a descendant of an HTML element that is a descendant of any element" (the asterisk is the universal selector and means "any element"). This rule is nonsense: HTML is not a descendant of any element – it is the root element – so this rule should not apply to anything and compliant browsers should (and do) skip over it.

However, IE seems to ignore the universal selector when it precedes "html," so this rule gets applied by IE and IE only.

The first width value in the second rule is read by all IE versions, giving WinIE5.x the total on-screen box width that it uses in its box model.

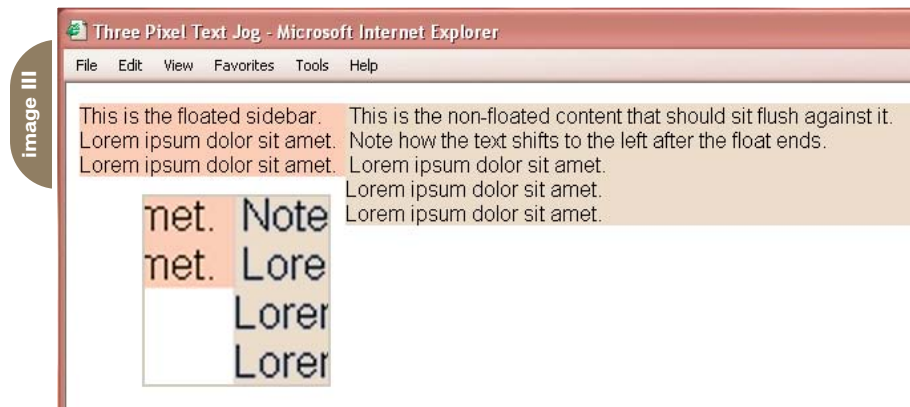
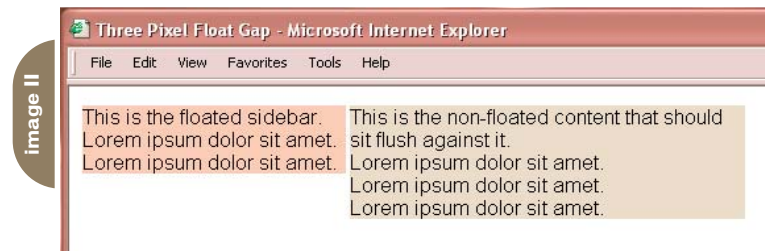
Unfortunately, IE6 for Windows and IE5 for the Mac also see this 244px value, but they don't have a box model problem and need the correct 200px value. The second value with the backslash character sets things back correctly for these browsers. Since WinIE5.x cannot understand a rule with a backslash in it, it keeps the larger 244px value, and all other browsers including IE6 and MacIE5 get the correct 200px value, making things look the same cross-browser.

Keep in mind that IE6 needs to be in standards-rendering mode for this to work correctly. Otherwise it will use WinIE5.x's incorrect box model, and since the hack only targets WinIE5.x, IE6 will not get fixed.

The Star HTML component of the MSBMH comes in handy quite often: IE has a host of bugs, and the Star HTML Hack allows you to feed it different values when it's misbehaving.

Three Pixel Gaps

You'll want to have the Star HTML Hack at your disposal when you're using floats. WinIE, including IE6, inserts an extra three pixels of space between the edge of a float and the edge of the following content. If the following content has a width or height assigned, the three pixel gap shows up as a space between the boxes (I'll call this the Three Pixel Float Gap; see Image II). If it does not have a width or height, the boxes seem to sit next to each other, but the content within the following box is pushed over by three pixels for as long as the float extends beside it, shifting back over to the edge where it belongs after the end of the float (Big John [John Gallant] of positioniseverything.net calls this the Three Pixel Text Jog; see Image III, with closeup). Although both are seemingly minor, the Text Jog is pretty unsightly, and the Float Gap can throw everything off in precise layouts, making it impossible to create a two-column layout with the float and following content sitting flush against each other.



To get rid of the Three Pixel Float Gap, you need to assign the float a negative margin to pull the following box back against it. Let's say we're using the following code to create two columns that sit flush against each other:

```
#sidebar {
float: left;
width: 300px;
}
#content {
width: 400px;
margin-left: 300px;
}
```

We've assigned #content a margin-left exactly the same size as #sidebar's width so that they will sit flush against each other, but in IE the Float Gap shows up, preventing them from touching. Add the following hack below your regular rules to get rid of the gap:

```
/* hide from MacIE */
* html #sidebar {
margin-right: -3px;
}
* html #content {
margin-left: 0;
}
/* end hide */
```

We've used the Star HTML Hack above to give values to just IE, then enclosed it

in a Mac Backslash Hack that hides the values from MacIE due to the backslash in the comment preceding the rule (a MacIE bug). Thus, only WinIE (the problem browser group) sees the hacked values.

What if you do want some space between your boxes, you just want to kill the extra three pixels WinIE adds? Just give WinIE a margin-right that is three pixels smaller than it should be. Code I is an example that leaves 10 pixels between the two boxes.

The examples in Code I have a width assigned to #content, but when you leave off this width, the Three Pixel Text Jog shows up. How can you get rid of the Text Jog if you have a fluid layout and can't assign #content a width? Holly Bergevin discovered that a height works just as well, and since IE will (incorrectly) expand a box's height to accommodate its content, you can assign a very small height and still keep your fluidity! Code II shows the Holly Hack to kill both the Float Gap and Text Jog.

The Holly Hack can be used in a variety of situations when IE is misbehaving. If IE is doing something strange, check to see if the offending box has a dimension assigned. If not, try giving it a height of 1%, and often this will fix things.

Peek-a-Boo Bug

Another IE problem you may run into while using floats occurs when you have

a container with a float inside and content alongside the float. In IE6, the content following the float may not appear at all, or appear only partially, until you scroll down or switch to another window and switch back. This bug was aptly named the Peek-a-boo Bug by Big John, who lists several workarounds for it on his site. The easiest one to apply was discovered by Matthew Somerville. If you give the container holding the float a line-height of any value, it cures the Peek-a-boo. If assigning the line-height would cause problems with child elements of different font sizes, you can use the Holly Hack on the container for an alternate quick fix.

Doubled Float-Margin Bug

Since we're on the subject of floats in containers, we'd better cover the Doubled Float-Margin Bug, again a WinIE-only problem. If you apply a margin to a float to move it away from the edge of its

container, WinIE doubles that margin. So the following code produces a 10 pixel margin on the left of the float in everything but WinIE, which shows a 20 pixel margin on the left:

The CSS:

```
#float {
float: left;
width: 100px;
margin: 10px;
}
```

The (X)HTML:

```
<div id="container">
<div id="float">float</div>
</div>
```

Your reaction to this is probably, "I know! Use the Star HTML Hack to feed IE a halved value!". This would work, but luckily can be fixed even more easily and cleanly. Steve Clason discovered that if you apply "display: inline" to the float it gets rid of the

doubled margin. Since "display: inline" on floats is ignored by other browsers (floats are block elements by definition) you don't even have to hide this rule from them via the Star HTML Hack! Easy.

Flash of Unstyled Content

Another IE bug with an equally simple solution is the Flash of Unstyled Content, or FOUC, named by Rob Chandanais of bluerobot.com. A page afflicted with this bug will show a quick flash of the page without any styles before the CSS takes hold. This only occurs in WinIE on the first page view, before the CSS is cached, and it happens when you are using @import to call your style sheet. Adding just one link or script element to your document will fix the problem. The addition of a link element is an easy and natural fix because most pages can benefit from an alternate style sheet, such as a print style sheet:

```
<head>
<title>My Page</title>
<style type="text/css">@import
"screenstyle.css";</style>
<link rel="stylesheet" type="text/css"
media="print" href="printstyle.css">
</head>
```

If you decide to add a script to fix the problem instead, keep in mind that the script doesn't have to be in the head to prevent the FOUC. Placing it inside the body but before all visual content works just as well.

Text Inheritance

WinIE5.5 has a problem inheriting the correct text size into tables, but a simple rule gets it back on track:

```
table {
font-size: 1em;
}
```

This makes sure the font size of the table displays at the same size as the surrounding text, which it ought to do by default, so the rule doesn't need to be hidden from other browsers.

Netscape Navigator (NN) 4.x also has major – and less predictable – text inheritance problems. Although you may want to present NN4.x with an unstyled version of your pages (using @import instead of <link>

image IV

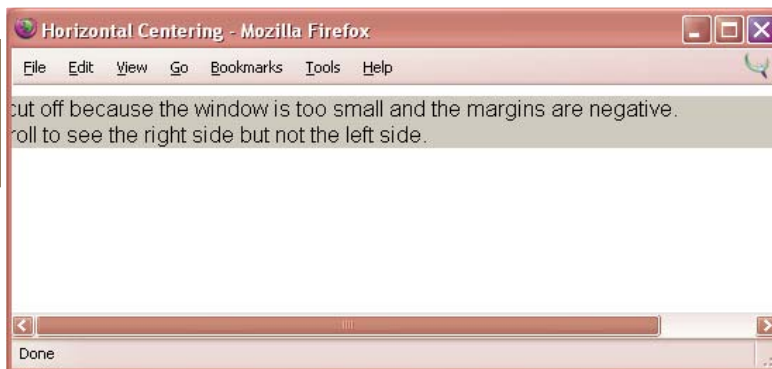


image V



to call your style sheet), there may be times when you want to give it at least some basic text formatting. Since NN4.x often loses text properties at seemingly random spots in your page, it's best to explicitly set the text parameters on everything that could possibly need them in NN's style sheet:

```
body, div, p, blockquote, ol, ul, dl,
li, dt, dd, td {
  font-family: Arial, Helvetica, sans-
  serif;
  font-size: 12px;
}
```

Opera has its own text problem when you set font-size to 100%: it computes it to be one pixel smaller than it should be. Using 100.01% instead of 100% fixes this.

Horizontal Centering

To center your entire layout in the browser window, use the following CSS:

```
body {
  min-width: 700px;
  text-align: center;
}
#wrapper {
  width: 700px;
  margin: 0 auto;
  text-align: left;
}
```

Setting the left and right margins to auto centers #wrapper in the window because the margins are set to equal values. Since WinIE5.x does not recognize this technique, add "text-align: center" to the main div's container, in this case "body," to achieve the centering in that browser (this property won't work in other browsers because it's only supposed to center inline content, not blocks, but WinIE5.x ignores that little detail).

Set a min-width on the body equal to the width of the centered box to avoid problems in Gecko-based browsers. If you don't set this min-width, when you size your window below the size of the centered box, its left side will get cut off without the ability to scroll over to the left to see the cut-off content (see Image IV). This is because when the window is too small, the auto margins must get set to negative values, thus extending the box equally off both sides of its containing block (in this case, the body).

Containing Floats

Keep in mind while using floats to create columnar layouts that floats, not just absolutely positioned boxes, are removed from the flow of the document. This means that a parent box doesn't know the float is there and thus will not expand to hold the child float. This may seem illogical at first, but think about when you are not using floats for columns, but for images placed in paragraphs – the most traditional use of floats. In this case, you want all the text to flow around the image – not just the text of the first paragraph where the image is placed, but all subsequent paragraphs that encounter the float as well. If the float made its container, a paragraph, expand to hold it, the next paragraph wouldn't start until the image ended, leaving a potentially large gap (see Image V). So, floats naturally stick out of their parent elements instead.

But don't worry – there is a way to overcome this behavior when your layout calls for it! All you need is a block-level element, set to clear the float, placed within the container but after the floated element. This forces the container to expand down around the element beneath the float, enclosing the float in the process.

The CSS:

```
br.clear {
  clear:both;
  height:0;
  margin:0;
  font-size: 1px;
  line-height: 0;
}
```

The HTML:

```
<div id="container">
<div id="float">float</div>
<br class="clear">
</div>
```

Another easy way to contain a float is to float its parent: a floated parent automatically expands to hold children floats.

While we're on the subject of floats, one caution: make sure you assign your floats an explicit width, as required by the CSS 2.0 spec. Although most browsers are lax about this (so much so that the requirement's been dropped from the upcoming 2.1 spec), MacIE takes floats that don't have a width and expands them to fill up their entire container. This

is a problem if you are using floated s to create a horizontal nav bar, for instance. In this particular case, you can float the <a>s inside the s to get them to sit beside each other in MacIE, but most of the time you have to bite the bullet and give all floats a width to keep MacIE from expanding floats to 100%.

Conclusion

I wish I could tell you that the bugs I've described are the only ones you will run into and that the hacks I've outlined will cure all your CSS ills, but I can't – I'd be lying. Unfortunately, there are quite a few other bugs I haven't touched on, and new bugs will certainly continue to pop up as browsers evolve and Web developers push the limits of CSS and (X)HTML. But, I can tell you that you're now armed against the most common CSS bugs and ready to create your first all CSS, cross-browser layout without fear. ☺

Zoe Gillenwater is a Web designer at the University of North Carolina at Chapel Hill with a passion for standards-based development. She also keeps busy with graphic design and multimedia projects. Zoe is an active participant in the css-discuss community and is one of those who believes CSS-based layout is ready for prime time. zoe@pixelsurge.com

```
#sidebar {
  float: left;
  width: 300px;
}
#content {
  width: 400px;
  margin-left: 310px;
}
/* hide from MacIE */
* html #sidebar {
  margin-right: 7px;
}
* html #content {
  margin-left: 0;
}
/* end hide */
```

code I

```
#sidebar {
  float: left;
  width: 300px;
}
#content {
  margin-left: 300px;
}
/* hide from MacIE */
* html #sidebar {
  margin-right: -3px;
}
* html #content {
  height: 1%;
  margin-left: 0;
}
/* end hide */
```

code II

d

f

fw

fb

cf

db

NeXTensio2

Create databases in next to no time
reviewed by russell stearman

As a developer who often finds myself with less time to develop projects than I would perhaps like, the lure of purchasing a plug-in that saves time on database creation is more than a little tempting. This is what the NeXTensio2 plug-in offers for PHP-driven systems that use Dreamweaver.

NeXTensio2 (note that it is not just developed for the experienced programmer, but also for the novice) enables the developer, in next to no time, to provide a database system that includes the access screens for the end users.

InterAKT, the company that created NeXTensio2, has taken into account the needs not only of a developer creating a suitable database on a tight timescale, but also of the user that has to utilize the completed system. NeXTensio2 aims to provide a full list of tools that offers the developer a fully conceived structure via a behavior interface. The NeXTensio2 system should also save vital planning time, allowing the developer to concentrate on the content issues of the system. By using precoded behaviors, this shortened development cycle should improve turnaround time and give the client a clear

sight of the intended goal from early on.

InterAKT admits that much of what NeXTensio2 is capable of is possible for an experienced programmer who has the time, but that is not the aim of the product – NeXTensio2 is a labor-saving alternative. It provides the end-user interface that is essential, which can be tailored during production to take into account most standard data input and extraction needs. This also extends to master/detail list management enabling database linking, expanding the capabilities of the system.

This is not InterAKT's first product. NeXTensio2 appears to have been developed to answer the needs of their previous systems, including their tNG transaction system, from which NeXTensio2 is partially developed. This development and extension into an InterAKT family of products results in a whole transaction data management system in PHP. All this has been designed with speed and efficiency in mind, aiming to allow developers to use NeXTensio2 to create powerful databases quickly.

The user interface of NeXTensio2 is a series of entry boxes running with a wiz-

ard-style interface, which guide you through the process of creation of the list information and end-user interface (see Image I). While I do not prefer wizard-styled interfaces, this one fulfilled my expectations and was flexible in its execution. The acceptable data types cover most of the types needed by the typical system; if the required database system is more specialized, then it's likely that it would be developed from scratch, which is outside of the program's intended role. The layout of the input screens is clear and concise, although a little difficult to grasp at first, owing to the lack of a clear overview of all the table field information. An impressive feature is the open-ended way in which you can use this system, implementing as much or as little of it as desired. The developer is free to pick and choose, although the creation of specialized additional behaviors for integration into the existing NeXTensio2 code would obviously be more difficult than for a bespoke system.

The system end-user entry behaviors are particularly handy. They enable the creation of a clear and separate data-entry style interface for general usage and the list management system to be used by authorized users, via the form creation behavior (see Images II and III). The ability to create a multiple-user entry system is attractive when displaying the final database system to a client, since it would save the client time and money in transferring files to their new system. This is supported by the standard safeguard features for data entry with validation for entry fields. It would be nice to see some development made in further segregating access between standard data entry screens and the more powerful list management areas. This could be via an automatic password facility or similar for page access – the “check credentials” function does not extend far enough for my liking.

NeXTensio2 extends from InterAKT's

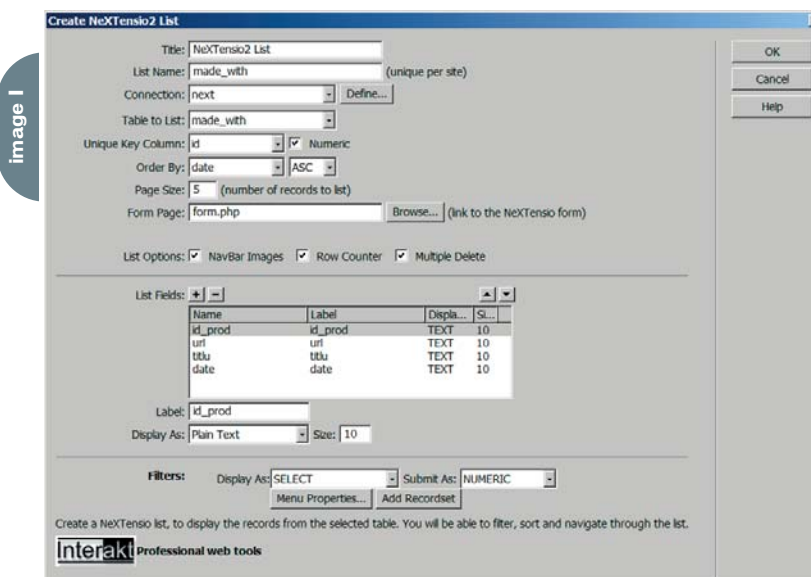
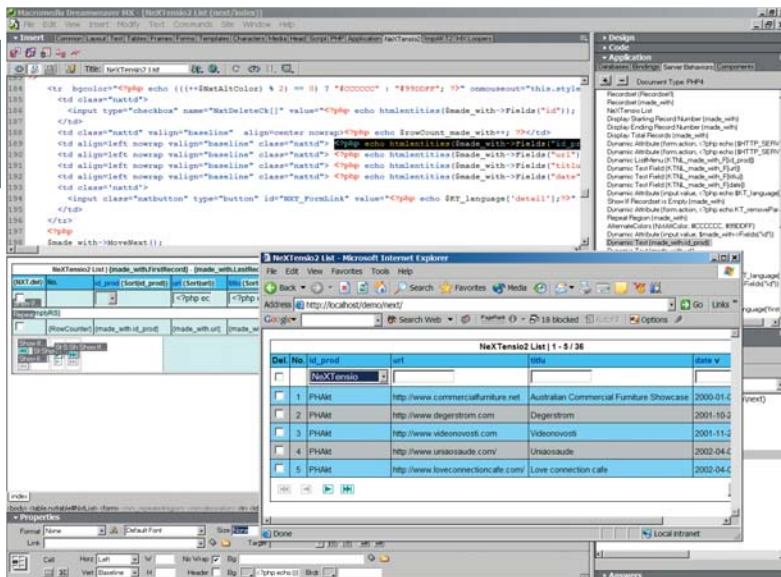


image II



tNG (transaction Engine) system, making this family of packages very useful for a small- to medium-size retail company. A user interface that is clear and easy, which takes no more than a few minutes to create, makes NeXTensio2 powerful indeed.

InterAKT is well aware of the needs of their clients and developed this tool with them in mind, so it fulfills many requirements. The documentation is well presented and easy to digest, and the company is very forthcoming and helpful. This lends to an image of the company as dependable and loyal to their customers. They are aware of the current version limitations and already have plans for the next update.

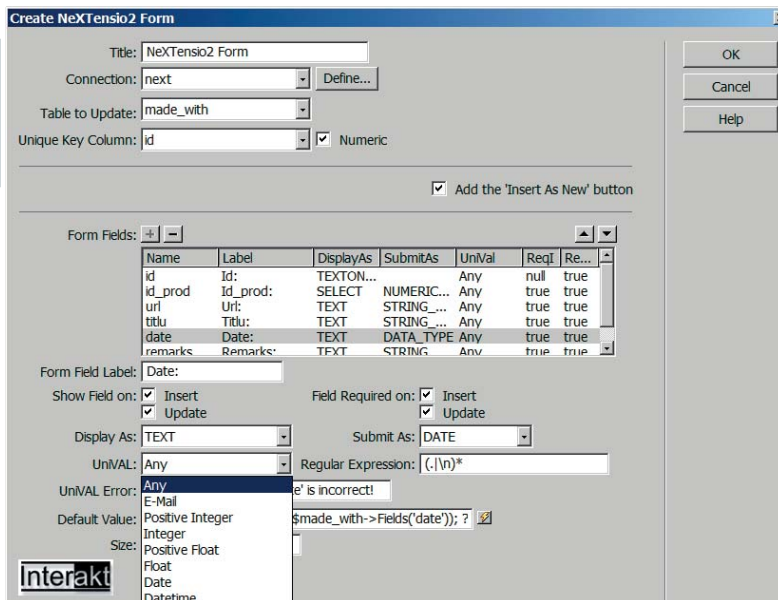
A welcome feature would be to enable further customization of the PHP pages created, which is presently handled by the KHTML program

included in the package. This is helpful, although the next KHTML release will be a separate product.

Conclusion

NeXTensio2 is a very useful tool for creating small- to mid-scale database systems whose main lure is the shortened development cycle. This makes the product suitable for creating smaller test and display systems for larger specialized projects. It is a very handy plug-in to have around for the database-oriented developer. The accessibility for the nonprogrammer is also good, although the level of database programming knowledge needed is still slightly above that of the average nonprogrammer. With NeXTensio2, you can create powerful and versatile transaction systems to suit a multitude of business needs. Visit www.interakt.ro/products/NeXTensio to learn more. ☺

image III



ONE PUBLISHER ALL THE SOLUTIONS



I-58450-283-5 \$49.95



I-58450-301-7 \$49.95



I-58450-315-7 \$41.95



I-58450-309-2 \$49.95



I-58450-316-5 \$49.95



Titles also available at Amazon, Borders, Barnes & Noble, and other fine retailers.
800 382 8505
CHARLESRIVER.COM

*custom*tools





when

I first laid my hands on Flash MX 2004, the thing that really made me say "Wow!" was the extensibility layer. If you are not familiar with it, this incorporates a new language, JavaScript Flash or JSFL, that allows you to do whole new things with the Flash IDE. by keith peters



The extensibility layer covers several new sections of the IDE. There are commands, which are relatively linear programs in JSFL. You run a command and Flash steps through the code and does whatever you've told it to do. Then there are custom tools that you can add to the toolbar to make any shape or drawing function that you can describe in code. Behaviors are like ActionScript templates that can be applied to any symbol. Timeline effects are essentially canned tweens created in JSFL, which again can be applied to any shape or object on stage. WindowSWFs are custom panels that you can open like any of the standard panels in the IDE. These make use of the new ActionScript command, MMExecute, to run JSFL commands on the current document being edited. Finally, JSFL scripts can be run from the command line or by other programs, making it possible to automate Flash functions or create whole new front ends for Flash.

A handful of people have now jumped onto the extensibility bandwagon. For the most part, people seem to be making commands and WindowSWFs, and some very useful ones at that. Some have also played around with command-line execution. The usefulness of Behaviors and Timeline Effects is still up in the air. I've seen very little being done in these areas at all.

What really surprised me, though, was the total absence of tools being created. I thought this was one area that would really take off. My guess is that the whole process of creating tools is fairly complicated and there is really no documentation on how to do it. My hope is that this article will make the process a bit more clear, and inspire a few people to begin making their own and sharing them.

The Structure of a Tool

Unlike commands and MMExecute statements in WindowSWFs, which execute in a fairly linear fashion and can be programmed like any simple script, tools take a fair bit of setup and require a specific structure. If you don't get it just right, not only will your tool not work, but chances are it won't even show up on the toolbar (see Image I).

To begin with, a tool generally consists of a JSFL file, and a PNG file to serve as its icon on the toolbar. It may also

include an XML file for specifying options via the Properties panel.

Here is the general outline for creating a tool:

1. Create the icon file.
2. If you will be using options, create the XML file.
3. Create the JSFL file.
4. Add the tool to the toolbar.
5. Test and debug.
6. Reload the toolbar.
7. Repeat steps 5 and 6 until it's done.

The Invisible Button Tool

As an example, we will be creating a tool that allows you to draw an invisible button on the stage. Invisible buttons are buttons that have no graphics except in the hit state. They are useful as "hot spots" or for making other static graphics function as buttons. They don't have any graphical representation in the final movie, but in the authoring environment appear as a translucent blue shape. I got sick of creating and resizing these buttons by hand all the time and realized a tool would be very useful for this.

Creating the Icon

You just need a 16 x 15 pixel PNG file to serve as an icon. This can be created in Fireworks, Photoshop, or any other graphics application. The easiest way is to load in the existing PolyStar.png file and edit it. You'll find that in the Tools directory in your Flash Configuration directory. In Windows, that's generally `c:\Documents and Settings\\Local Settings\Application Data\Macromedia\Flash MX 2004\en\Configuration\Tools`. On a Mac, it's `<hard disk>/Users/<user name>/Library/Application Support/Macromedia/Flash MX 2004/en/Configuration/Tools`.

Make an icon that will represent what the tool is. I just made a blue rectangle the same shade as the invisible buttons appear (see Images II and III). Save the PNG in the Tools directory with the name "InvisibleButton.png".

We won't be using any options at first, so don't worry about the XML file just yet.

JSFL Program

Now comes the big part – creating the JavaScript Flash program that makes the tool work.

Tools are actually event-driven programs that just sit there and wait for something to happen, such as a mouse click or key press. As such, the only code in a tool should be in predefined event handlers or additional functions that are used by them. There should be no "loose code" in a tool's JSFL file. Here are the standard tool event handlers, with a brief description of each:

- `configureTool`: This is run each time Flash starts up or tools are reloaded. Flash searches for this function in every JSFL file in the Tools directory and executes it. It needs to contain a few standard lines of code to make the tool available to be added to the toolbar – generally nothing else.
- `activate`: This is called whenever the tool is selected on the toolbar. Here is where you initialize various aspects of the tool, which usually includes grabbing a reference to the active tool, and getting any default options from the options panel, if one has been set up.
- `deactivate`: This is called whenever the tool is active, and another tool is selected. Although generally not much is needed here, you can use it to do any cleanup.
- `notifySettingsChanged`: This handler is called when a user chooses the options for a tool from the Property Inspector, changes some settings, and clicks "OK" in that dialog. The code in this function will often be quite similar to the activate function, as both need to get and process the options data.
- `setCursor`: Often Flash or the operating system needs to take control of the cursor and change it to a particular shape – hourglass, arrow, etc. When they are done with it, this function will be called, allowing you to change the cursor to one of several predefined shapes.
- `keyUp`: Fires whenever a key is released.
- `keyDown`: Fires whenever a key is pressed.
- `mouseUp`: Fires whenever a mouse button is released.
- `mouseDown`: Fires whenever a mouse button is pressed.
- `mouseMove`: Fires each time the mouse is moved.
- `mouseDbClick`: Fires whenever the mouse is clicked twice within a certain

image I



short period. Note that the first click will still generate a mouseDown and mouseUp event.

To create a tool, you just need to define some or all of these functions in a text file and save it as a .jsfl file in your Tools directory. Minimally, you need a configureTool handler, so let's see what needs to be in there.

configureTool

As mentioned, this function needs to define certain properties so that it can be seen by the program and available to place on the toolbar. Code I shows the configureTool function for the invisible button tool (Code listings are available at www.sys-con.com/mx/sourcec.cfm).

A little note on the JSFL Document Object Model: JSFL has described most of the Flash authoring environment and most aspects of a Flash movie in a series of objects. These are arranged in a logical structure to represent how things are actually arranged. For example, the "fl" object (you can also call it "flash") is the root of this model and represents the entire program. It contains an array called "documents", with one element for each Flash document that is currently open. Each document has an array called "timelines", equivalent to scenes. From there, each timeline has layers, and each layer has frames. Anything on the stage in a particular frame is in that frame's "elements" array.

The fl object also contains a "tools" object, which has an "activeTool" property. This always points to the tool that is currently being used or configured. Here we grab a reference to that and store it in curr_toolObj. Then we run several "set" functions on it. These tell Flash what to use for the tool's icon, menu string, tool name and tool tip. Here is also where you can set the XML file to be used for the tool's options, if you have made one. I've commented that one out as we're not using it at this point.

setCursor

As we don't have any options and don't need any other setup, we can skip activate, deactivate, and notifySettingsChanged for now. Most of our work will be in the mouse handlers. But first, let's tell Flash that we want a crosshair cursor whenever the tool is active. This will let the user know that

he's supposed to draw something. The following code shows the setCursor function.

```
function setCursor(){
    fl.tools.setCursor(0);
}
```

The tools object has its own setCursor function. This is easy to get confused with the setCursor event handler. fl.tools.setCursor() takes a single argument, a number from 0 to 7. These will create the following cursor shapes:

- 0: Crosshair
- 1: Black arrow.
- 2: White arrow.
- 3: Four-way arrow.
- 4: Two-way arrow (horizontal).
- 5: Two-way arrow (vertical).
- 6: An "X" cursor.
- 7: Hand cursor.

mouseDown

The next event we want to handle is the user pressing the mouse button down. This is our signal that he wants to draw something starting at the point where the mouse was clicked. We'll need to find out where that point is, and begin drawing. The following is the mouseDown code.

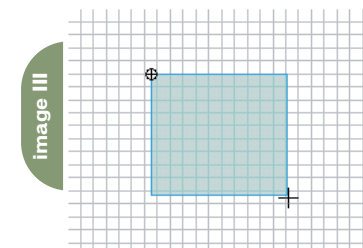
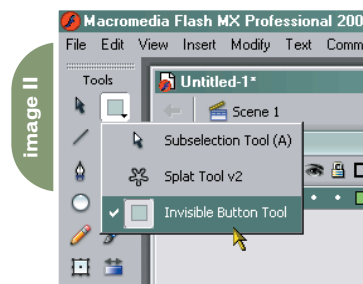
```
function mouseDown(){
    startPoint = fl.tools.penDownLoc;
    fl.drawingLayer.beginDraw();
}
```

The tools object has a couple of properties relating to the mouse position, penLoc and penDownLoc. penLoc holds a reference to the current position of the mouse, and penDownLoc holds the position of the mouse the last time the mouse was clicked. Both are objects containing x and y values.

Here we are storing the penDownLoc coordinates in a variable called startPoint. This variable will be available to any function in the tool, much as a timeline variable in ActionScript. Also similar to AS, had we declared it using the keyword "var", the variable would be local to the mouseDown function and not be available when we needed it in other functions.

The poster for the Web Services Edge 2005 East conference is divided into several sections. At the top, it features the 'web services EDGE conference & expo' logo on the left and the text 'Web Services Edge 2005 East' in a large, bold font on the right. Below this, it says 'International Web Services Conference & Expo'. A prominent black banner with white text reads 'Call for Papers Now Open!'. The main body of the poster has a green background on the left and a photograph of the Hynes Convention Center in Boston, MA on the right. Text on the green background includes 'The Largest i-Technology Event of the Year!', 'Guaranteed Minimum Attendance 3,000 Delegates...', and the website 'www.sys-con.com/edge'. At the bottom, it lists the dates: 'Tuesday, 2/15: Conference & Expo', 'Wednesday, 2/16: Conference & Expo', and 'Thursday, 2/17: Conference & Expo'. A small logo for 'PRODUCED BY SYS-CON EVENTS' is also present.

Next we see another object, "drawingLayer". If you open up Flash and use any of the built in drawing tools, you'll see that while you are drawing with them, you'll get a preview of what you are drawing using a thin black line. When you release the mouse, the shape is drawn with whatever stroke and fill attributes are set. That temporary preview drawing





occurs on the drawingLayer, and you now have full access to it.

All drawingLayer activity needs to commence with a call to its beginDraw() function, and when you are done, you should exit out of drawing mode with endDraw(). These two calls usually occur in mouseDown and mouseUp respectively. All the drawing itself generally happens in the mouseMove function.

mouseMove

We have the point where the mouse went down, and we've entered drawing mode. Each time the user moves his mouse, we want to give him a preview of what he will see if he releases the mouse button. We'll do this in the mouseMove function, shown in Code II.

First, realize that mouseMove will be fired every time the mouse is moved, regardless of whether the user has pressed the mouse button or not. So we check another tools property, "moueslDown" to make sure that the user is actually trying to draw something. It would be more efficient to create the drawing code in a separate function, such as "drawPreview". You could then assign this function to mouseMove within the mouseDown function like so:

```
mouseMove = drawPreview;

and delete it in mouseUp:

delete mouseMove;
```

But for simplicity, we will leave it as is.

In addition to bracketing the entire drawing cycle with beginDraw() and

endDraw(), we need to surround each individual preview-drawing session with beginFrame() and endFrame(). This clears the drawingLayer of all previous material and sets it up to receive additional commands. The usual sequence is:

```
mouseDown:
  beginDraw
mouseMove:
  beginFrame
  drawing actions
  endFrame
mouseUp:
  endDraw
```

If you notice flickering or leftover artifacts on the screen, you are not following this sequence. There's no reason that drawing to the drawingLayer in your custom tools shouldn't be as smooth as in the built in tools.

In the drawingLayer frame, we grab a local reference to fl.tools.penLoc, and issue a series of moveTo's and lineTo's. The drawingLayer line drawing functions work identically to the ActionScript drawing API commands. This simply draws a rectangle from where the mouse was clicked to its current location.

mouseUp

Hold on to your hat, here's where the real action starts. The user is happy with the preview we've given him, and wants an invisible button in that exact spot. Rather than simply drawing a shape, we need to create a button in the library, put it on stage, and position and size it to where the

user asked for it. Code III shows the code. We'll walk through it step by step to see what's happening.

The first few lines end drawing mode on the drawingLayer, grab a reference to the current mouse location and compute the width and height that the button should be.

Next, we grab a refer-

ence to the currently active document in the authoring environment. Although you can access any document through the documents array, usually you only want the current document, which is always available by calling fl.getDocumentDOM().

Once we have the document, we get a reference to its library. We use the "itemExists" method to see if the library contains an item called "invisButton". If not, we create it with the "addNewItem" method. We specify "button" as the type of item to add, and "invisButton" as its name. You can also add videos, graphics, movie clips, bitmaps, or even folders with this method.

Now we have a blank button in the library. We begin editing it with the "editItem" method. This is equivalent to double clicking on the item in the library. If we were to end the code right here, the user would actually be left with the button open for editing in the authoring environment.

As we now need to deal with the frames of the button, and as frames are part of a layer, which is part of a timeline, we first need a reference to the current timeline of the current document. We get that with the "getTimeline" method of the document.

A newly created button has only one frame – the "Up" frame. Remember that an invisible button only has content in its "Hit" frame, which is frame 3 (starting with 0). So we need to insert three more frames, and make frame 3 the current frame, and a keyframe. These three lines take care of exactly that:

```
curr_tl.insertFrames(3);
curr_tl.currentFrame = 3;
curr_tl.convertToBlankKeyframes(3);
```

It takes a bit of practice to get the hang of manipulating frames and layers and elements in JSFL. It helps if you think beforehand what you would do by hand, list out those steps, and then find the functions that do those in JSFL.

Now we are sitting inside the hit frame, ready to add a graphic. Rather than drawing a rectangle the exact size we need it, however, we'll just draw a generic 100 x 100 pixel square, and resize it whenever we need to use it. This allows us to use the same button symbol for multiple instances of invisible buttons.

We use the addNewRectangle method of the document object to draw a rectangle on the stage. The first argument to this method is an object containing top, left, bottom, and right coordinates of the rectangle to be drawn. We've set these to 0, 0, 100, and 100 to create our 100 x 100 rectangle. The second argument is the roundness of the corners

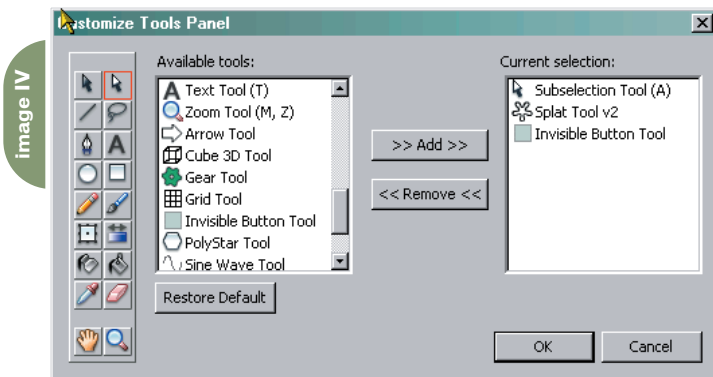


image IV

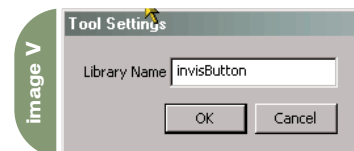


image V



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





of the rectangle. We'll use zero to have sharp corners.

The next two arguments can be used to suppress the fill and/or stroke of the shape drawn. We definitely want the fill, so we leave that false. Specifying true as the last argument suppresses the stroke, for which we have no use.

Having finished our work here, we exit edit mode, which puts us back on the main timeline. Note that there is room for improvement here. If the user was not on the main timeline to begin with, we really shouldn't return him there. A more complete strategy would include taking note of the current timeline, layer, and frame the user was on, and returning him there when we were done. For the sake of simplicity I'll leave that as an exercise for you to practice.

At this point, we've either skipped over the if block because the invisButton symbol existed, or we've created it. Thus, the first time someone uses the tool, the symbol will be created. Successive uses will reuse the existing symbol.

Next we add the button to the document with, obviously enough, "addItemToDocument". The first argument of this method is the point where you want the item placed. What do you know? We already have that stored in the variable startPoint! The next argument is simply which item to add.

Now we need to size the button to what the user drew. We've stored the size in the variables, w and h. Knowing that the button starts out at 100 x 100, we can scale it up or down to the correct size by using "scaleSelection". You'll notice that many of the document methods operate on whatever is currently selected, hence "scaleSelection", "moveSelectionBy". Luckily, when we add an item to a document, it is automatically the only thing on stage selected.

Unlike _xscale and _yscale in ActionScript, which use percentages to scale, scaleSelection uses fractions. Thus a value of 1 is 100%. If we divide w and h by 100, we'll get the right amount to scale.

For example, if the user drew a shape 233 pixels wide, w will equal 233 – which divided by 100 is 2.33. So Flash will scale the 100 pixel button 2.33 times its width, making it 233 pixels.

One minor problem here is that the point you specified in addItemToDocument is used as the center point of the new button, not its top left corner or even registration point. So the button will not be centered on the point where the user originally clicked. We simply need to move the button by half its width and half its height and it should be properly positioned. We do that with moveSelectionBy. This takes an point object containing x and y values, so we have to encode our w and h variables into an object before passing them to the function, as so:

```
{x:w/2, y:h/2}
```

Add the Button to the Toolbar

We've finished our first round of coding and are ready to add the button to the tool bar. Save the JSFL file as "InvisibleButton.jsfl" in your Tools directory, making sure your icon png file is there too (see Image IV).

Start a new Flash movie and open the Customize Tools Panel (Edit->Customize Tools Panel...). You'll see a dialog showing all the existing tools on your toolbar, and any tools that are available to add. The Invisible Button Tool should appear in the list alongside the icon you created. If it is not there, there is a problem in the configureTool function. If Flash cannot find or cannot successfully execute that function, or if it doesn't have the necessary code telling Flash its name, etc., it won't be listed.

Click on the little toolbar representation on the left, on the spot where you'd like to put your tool. You can remove the existing tool and replace it with your own, or simply add it to the list.

Click OK and your tool should now be

on the toolbar. If it is not, there is a major error in your code. In ActionScript, certain bugs will cause run time problems. Others will prevent the movie from being compiled at all. Similarly in JSFL, some errors will throw JavaScript errors when you try to run your tool, more serious errors will prevent the tool from appearing at all.

Test and Debug

In a perfect world, the tool would be sitting there on the toolbar and when you went to use it, it would work exactly as you planned. In reality, if something can go wrong, it will. If your tool doesn't show up on the toolbar, or you run into some problem with it while testing, you already know you need to dive into the code and fix it. Then what?

Reload the Toolbar

You need to reload the toolbar to make the latest version the active one (or perhaps to make it show up at all). You have a few choices. You can restart Flash. A bit extreme. You can go back into the Customize Tools Panel, remove and re-add the tool. Still a bit much. Fortunately there's an easier method, using JSFL itself.

The fl object has a method called "reloadTools" for this exact purpose. We just need a way to execute this method whenever we need it, and that's what commands are for.

Create a new JSFL file containing a single line:

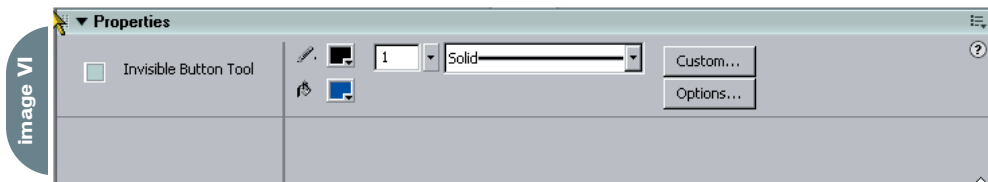
```
fl.reloadTools();
```

Save this in the Commands directory in the Flash Configuration directory. This will now appear on the Commands menu, and you can reload the toolbar each time you change your tool JSFL file.

Adding Options

The Invisible Button Tool is useful, yet pretty simple (see Images V and VI). You draw on the stage and it puts a button there. There is no real need to specify any options in it. But in order to demonstrate how options are used, we'll add in an option to allow the user to specify the library name of the button.

The first thing we need to do is uncomment the line in configureTool that



calls `setOptionFile`. This will set the option file to an xml file named "InvisibleButton.xml".

Naturally, we need to create this file next. It will go right in the Tools folder along with the other tool assets. The contents of this file are shown below.

```
<properties>
  <property name="Library Name" variable="libName"
  defaultValue="invisButton"
  type="String" />
</properties>
```

The root node is `<properties>`. Each option you want to define goes in its own `<property>` node. You define the option in the node's attributes. All options will contain name, variable, defaultValue and type attributes. The name is the label shown in the Tool Settings dialog. The variable is the identifier used to pass the value back to the tool JSFL file.

Type can be "Number" (integer), "Double" (floating point number), "Boolean", "String", "Strings", or "Color". Number, Double, and String will present the user with a text field. Boolean will create a checkbox, Strings will create a drop-down list, and Color will create a color chooser.

The defaultValue is the value with which the tool will be initialized.

Depending on the type, other attributes may be available. The numeric types have max and min attributes, and Strings has a list attribute, which is a comma-separated list of values in the form, list="Dog,Cat,Mouse".

If you reload tools now, and select the Invisible Button tool, you'll notice there is an options button in the Property Inspector. Click that and you'll get the Tool Settings dialog allowing you to change the setting for Library Name. Of course, we need to program something into the JSFL to retrieve this value and use it when it creates a button.

We need to retrieve the value at two points. First, when the tool is activated, and then any time the settings are changed. Now you see the use for the activate and notifySettingsChanged event handlers. As mentioned earlier, these two functions are often quite similar. In our case they are exactly the same and are given in the following code.

```
function activate(){
  libName =
  fl.tools.activeTool.libName;
}
function notifySettingsChanged(){
  libName =
  fl.tools.activeTool.libName;
}
```

As you can see, the option is passed in via its variable name, which becomes a property of `fl.tools.activeTool`. We simply grab this value when we activate this tool and update it whenever the settings are changed.

The variable `libName` will now initially contain the default value, "invisButton". If the user ever changes it, `libName` will be updated automatically. We simply need to substitute `libName` everywhere we previously hardcoded "invisButton". This

only affects the mouseUp function. The updated version of that function is in Code IV.

Save that, reload tools, and test it out. Initially, the tool should create a symbol named "invisButton" in the library. Additional uses of the tool will continue to use the same symbol. However, if you click on the Options button and change the Library Name, the next time the tool is used, a new symbol with that name will be created.

Conclusion

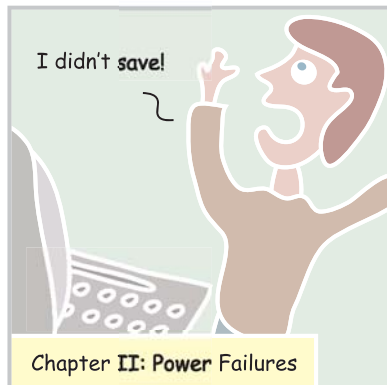
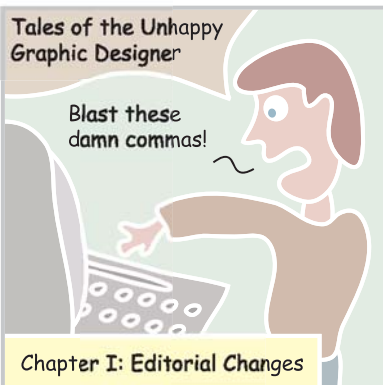
We've just barely scratched the surface here with custom tools. In addition to the possible improvements noted in the article, there are ways to make tools snap-to-grid, and constrain them when you hold down the shift key while drawing. There is also a nasty "but" that arises if you try to draw inside a symbol that has been translated, scaled, or rotated. And beyond just fixing things, there are many powerful things you can build into your tools that we didn't even touch.

In Chapter 3 of the book *Extending Flash MX 2004, Complete Guide and Reference to JavaScript Flash*, published by Friends of ED, I cover custom tools much more exhaustively, and handle all the points brought up above. And before you accuse me of a cheap book plug, I'll also throw in that the entire chapter is available as a free download at www.flashextensibility.com.

At any rate, I hope that this article has shed a little light on how to make tools and given a few people a little nudge into giving them a shot. ☺

xile

written & illustrated by louis f. cuffari ⑦





by charles e. brown

I HAVE TO BEGIN

this month's article with a confession. Each year, because of my articles and books, software publishers send me piles of free software with the hope that I will do an article about their product. I use the software and then, at some point, start feeling guilty. So, in that light, this is my periodic attempt at assuaging my conscience; hopefully, I'll give you some good advice.

CALL
IN
THE
SPECIALIST



Fireworks MX is a great program that can do a lot. But in today's "job must be done by yesterday" world, the concept of object-oriented programming is here to stay. The whole philosophy behind OOP is that you use a prebuilt solution and just plug in the information. This saves hours of potential trial-and-error development.

The software tools I discuss here are both relatively inexpensive and compatible (directly or indirectly) with the MX environment.

Alien Skin Software

This company has had a strong association with Fireworks for quite some time now. As a matter of fact, a "lite" version of its products, Eye Candy 4000 and Alien Skin Splat, ships with Fireworks MX 2004. These are filter programs that will allow you to create special effects quickly.

The full version of Eye Candy 4000 comes with 23 easy-to-use filters. As an



example, you could apply the melt filter to a photograph and end up with something like Image I. This could be the easy road to becoming the next Salvador Dali in melting time.

How many hours have you spent trying to get that glass button effect? The glass filter will give you the effect shown in Image II, among others, quickly and easily. The interface will allow you to change color, shading, transparency, etc.

The full version of the Splat program will give you some great frame and edging effects.



As an example, the cameo shown in Image III was created with just a few clicks.

I could have achieved the same effect by using a masking technique. However, this image was created within seconds with just a few simple clicks.

Xenofex 2 offers 14 additional filters for such exotic effects as electricity, smoke, crumpled paper, jigsaw puzzle, etc.

Image Doctor will allow you to repair damaged images by removing scratches, spots, and various compression errors.

Why reinvent the wheel? It took me a considerable amount of time to recreate what these filters did with a few simple keystrokes in an easy-to-use interface.

You can purchase these filters, and download demo versions, at www.alienskin.com. They also have special package pricing.

Auto FX

This series of filters are available in two versions: stand alone and plug-in. However, as of this writing, the plug-in versions are not available directly for Fireworks. The plug-in versions are available only for Photoshop, Corel Photo Paint, and Jasc Paint Shop Pro.

I happen to be a user of Paint Shop Pro, and as a result I was able to install these filters as a plug-in. Once I did that, I was able to access the plug-ins with Fireworks.

These filter packages offer some very sophisticated effects. As an example, the Mystical Lighting package gives you ways to play with an image's lighting. Image IV offers an example.

As an experiment, I found the graphic in Image IV on their Web site. I was able to reproduce the results within about 15 minutes. Unfortunately, space does not permit me to show you many of the interesting variations I was able to create.

As a designer, I'm always looking for interesting things to do with text. One of the filters of the Dreamsuite Series of filters will allow you to turn ordinary text into something like Image V.

The Mystical Tint package will allow you to work some magic with the image's colors. This is shown in Image VI.

Again, I was able to create a number of variations that I do not have room to

reproduce here. Experimenting with the interface of these filter packages opened up a number of fascinating possibilities.

Demo versions, a number of examples, and pricing are available at www.autofx.com. As I stated at the outset, you can use this as a freestanding program and then export into Fireworks.

You will be amazed.

WildForm

Not all the samples were for Fireworks. As you may know, there are a number of programs to augment Flash

image V



MX. Among the best programs I have seen is Wildfx. This simple, and inexpensive, package will give you an unbelievable 400 text effects.

You open it up, type the text, pick the effect, and the job is done.

The interface is shown in Image VII.

I can easily export the SWF file and then incorporate it into a Flash movie. If 400 text effects are not enough, you can get a low cost supplemental package that will give you an additional 200 effects. If you want to put a Flash

movie together, using multiple SWF files, quickly, you can use the companion product called Linx.

This will allow you to easily take SWF files and just dropping them onto a timeline.

The interface is shown in Image VIII.

Please remember that I am not using Linx as a substitute for Flash MX. However, it helped me speed up some of the more routine tasks.

I am finding the Wildform programs increasingly important to help speed up my workflow. Information about these programs can be found at www.wildform.com.

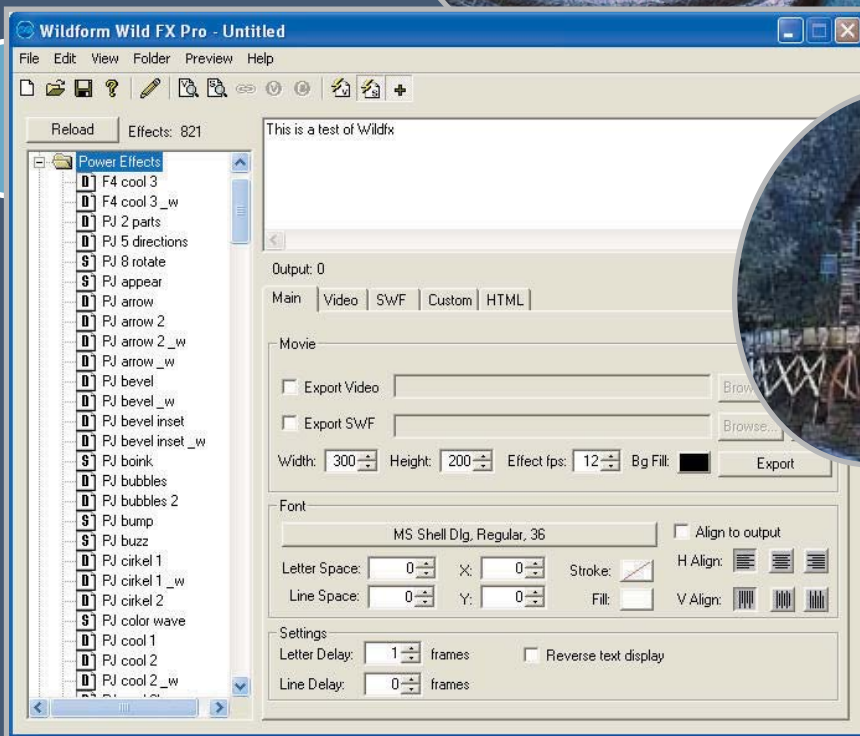


image VII



Image VI



image VIII

Conclusion

This is only a small sampling of the third-party market building around Macromedia's MX line. Each of these programs can speed up the workflow process by offering specialized functionality and reducing graphics process down to a few simple keystrokes.

Are they worth the investment?

Only you can decide that. However, in my practice, each of these programs has saved me hours and, in one case, days of work. This translates out to profitability that could outweigh the initial cost.

Give the demos a try. I don't think you will be disappointed. ☺

Charles E. Brown is the author of Fireworks MX 2004 Zero to Hero and Beginning Dreamweaver MX 2004. He also contributed to The Macromedia Studio MX Bible. charles@charlesebrown.net

the tricks to tracing

tracing

Many casual users of
FreeHand MX ask
how to turn a bitmap

or photograph into
vector art. Naturally,
the hope is that there's
a button to click and
the job is done. There's
not, but it's close.

by ron rockwell







Each image will be different in terms of both its content and the look you have in mind for the final artwork. That makes it pretty tough to create a cut-and-dried tutorial, but there are a few basics that you can follow.

The Trace Tool

The Trace tool is one of those areas in FreeHand where nothing is cast in stone. There are three or four main variables, and thousands more within them. There's absolutely no single-shot approach – you have to experiment. There are, however, a few points to consider when using the Trace tool. The bitmap you use should have a resolution of between 300 and 600 lines per inch. If you use a bitmap outside these limits, you'll get jaggies (on the low side) and too many points (on the high side). Higher resolution scans also take up a lot more RAM, and that can slow down your operation or even bring it to a crashing halt. A high-res tracing is a common reason that a FreeHand document won't print. You will also get different results depending on how your FreeHand Redraw preferences are set. If you have them set to Low, you will get a jaggier scan than if you set the on-screen image resolution to High or Full.

Your tracing can have color results, or grays, depending on the choice you make in the Color Mode menu (see

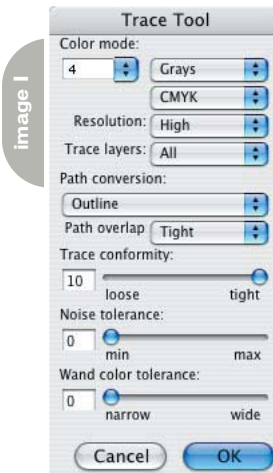


image I

image II



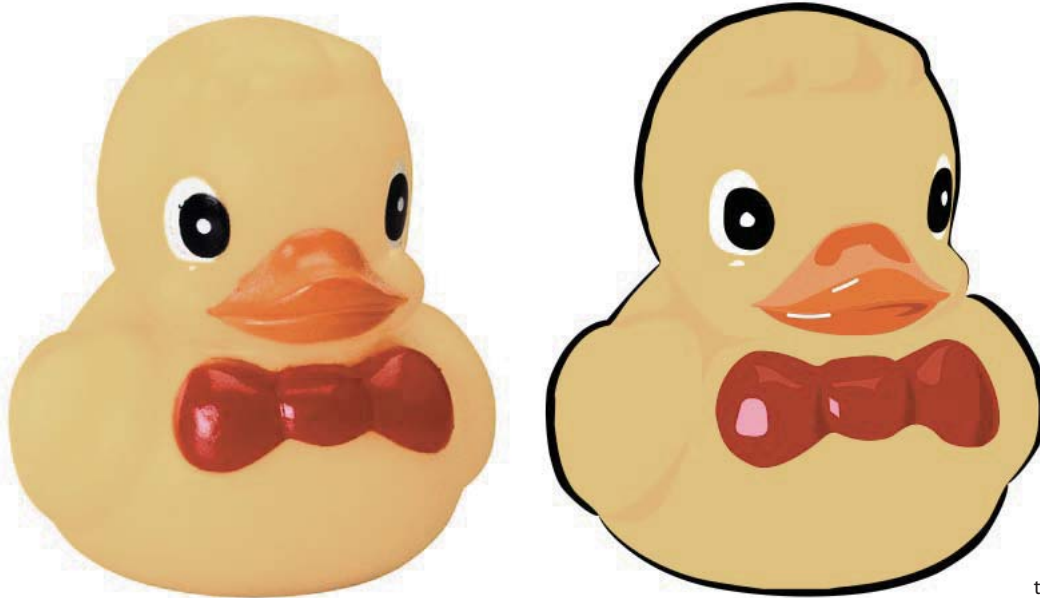
Image I). Beyond that, you can choose from 2 (black and white) to 256 color/gray steps in RGB or CMYK. The resolution settings are Low, Normal, and High. Low will give you a looser trace in a short time; High takes longer but provides many more details and a tighter tracing. Normal is adequate for most jobs, as it splits the difference between Low and High.

The Trace tool can trace on all layers, just foreground layers, or just the background layer. Use the tool to drag a rectangular marquee over the area you wish to trace. The tool will trace every-

thing inside the rectangle, so you can surround an entire object or trace a rectangular portion. When working with a bitmap, it's best to place the bitmap in an area that doesn't overlap any other objects when you do the tracing. For instance, if you have drawn a rectangle to surround the bitmap and it's on a foreground layer, it will become part of the tracing if you have All or Foreground selected. Moving the rectangle to the Background layer and choosing Foreground would solve the problem. You could also hide (View>Hide Selection) the rectangle until you're



image III



given
a blue
tint fill.
Remember
that the tracing
you get consists of

vector paths. Those paths are
fully editable, and are in every way just as
if you had drawn them yourself.

So much for a relatively good trace, but if you're looking for clip art or a different feel to the artwork, try adding strokes and changing colors in the Object panel as seen in Image III. For this approach, the Name All Colors Xtra was invoked (on the 4-color gray tracing), adding three grays to the Swatches panel (with black already in the Swatches panel). Then create new colors in the



through with the tracing operation. At any rate, look at your layer and placement situation and make a Trace Layers choice accordingly.

Moving down the Trace tool options, you come to the Path Conversion menu. Each choice yields a totally different tracing – luckily, we have the Undo command. The Outline conversion outlines contiguous areas of color and creates closed, filled paths. You have the further options of the type of Path Overlap. If you're tracing line art or text, use None; for photographic-type images, choose Loose; and if you want a close representative of the bitmap in the tracing, pick Tight.

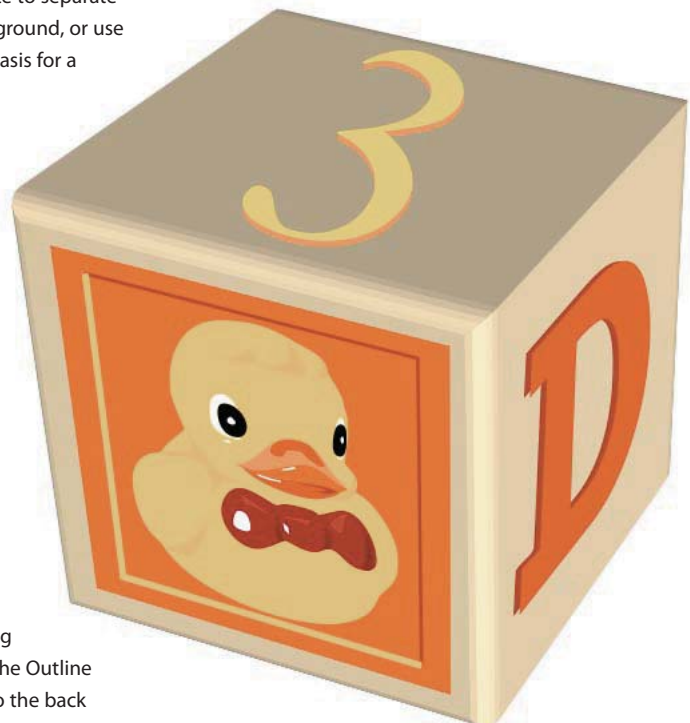
If your image uses a lot of strokes, and few filled areas, then Centerline is a good choice for the trace. You can choose to have Uniform, 1-point strokes, or deselect Uniform to end up with a more fluid, hand-drawn tracing.

For more ticklish drawings with strokes and filled areas, pick the Centerline/Outline conversion method. This method combines the other two methods and allows you to decide how to treat paths according to their width.

Your choice is to determine that paths with widths lower than 2 to 10 points (your choice) are left open. These options may seem trivial, but make a huge difference in the resulting tracing.

The last conversion method is called Outer Edge. I call it the wire outline, and use it all the time to get a very faithful perimeter outline of my vector artwork (as long as I'm tracing closed paths). All I have to do is change the stroke width to something heavier and I'm done. I can also fill the trace with white to separate the drawing from its background, or use (without a stroke) as the basis for a drop shadow.

All that is confusing to read about, so look at Image II for a visual description. The lacy metalwork and solid areas make it a good test subject. Other than changing the Path Conversion option, none of the settings were modified. Notice the loss of detail from Outline to Centerline, and the addition of black to the Centerline/Outline methods. A second tracing using Outer Edge was made of the Outline tracing. Then it was sent to the back (Modify>Arrange>Send to Back) and





Mixer panel and add them to the Swatches panel. Open Edit>Find & Replace, and select the dominant menu color within the tracing. In this case, it was the middle gray. Add a stroke (I added a black 1-point stroke), and change the fill color to something new. Here, I chose a yellow-orange tone. Then using Find & Replace again, select another gray in the image (here, the light gray). This time, I gave it a Hairline (0.25-point) stroke and a lighter fill color. Finally, I traced the image using the Outer Edge conversion method, applied a heavy stroke, and sent it to the back.

Uses of the Trace Tool

With a basic understanding of how the Trace tool works, you can save time and put your energies into designing and drawing instead of tedious manual tracing with the Pen or Bezigon tool (think how long it would take you to trace the birdcage in Image III with the Pen tool). Tracings can be used as masks or clipping paths, and also a quick way to turn a line illustration into a compound path. When you convert line art into composite paths by tracing, you are basically doing the work of the Expand Path Xtra. The resulting object can be filled with solid color, given a gradient of some kind, or used as a clipping path.

Distortion

For this spot illustration (Image V), I found a photo of a rubber duck. Normally, this image would suffice for use on the Web or in print, but I wanted to apply the image to a child's building block. That process would involve distorting the image in a way that can't be done in FreeHand, so I had to make a bitmap-to-vector

version. I decided to use the Trace tool to make the switch. However, the yellow color of the duck was so even that a regular tracing created too many shapes and points to make a viable vector graphic. In order to clarify all the yellow tones, there were nearly 2,500 objects in the tracing. That's overkill. Because there isn't a lot of detail in the image, I decided to do a combination of Trace tool and hand-drawing. I used the Pen tool to outline shadow areas in the yellow section, and placed the shadows on a separate layer. Still using the Pen tool, I drew the other objects, placing each on its own layer to make later alterations easier to select. (If you haven't used the shortcut, you can select everything on a layer by holding down the Option/Alt key and clicking the layer's name in the Layers panel.) Some areas, such as the eyes and bowtie, were traced with the Trace tool. The Trace tool can be dragged diagonally to trace everything within the confining rectangle, or you can pull another of its tricks out of the bag.

What tricks? Well, click the cursor on an area in a bitmap or vector object, and according to the parameters you've set in the Trace dialog box, the tool will create a selection outline. Yup, the army of marching ants will surround the clicked area. Hold down the Shift key to add to the selection. It can be the same color in a different area, or it can be a different color – whatever you click will be added to the selection. If you hold down the Option/Alt key, you will deselect a given area. It's pretty much the same as working in Fireworks or Photoshop, but not quite as predictable or consistent in its operation. Once the selection is complete, you can't just add

a fill or a stroke – the ants just keep on marching around – what have you accomplished?

Well, move the cursor over any part of the selection and the cursor will display a small square beneath it. Click inside the selection, and a new option box will appear. You will be able to choose Trace Selection or Convert Selection Edge (see Image IV). The former will do a regular tracing according to the parameters you've already set in the Trace dialog box. That means as many levels of color or black and white and all the other options you've selected will be in play. When you click this option and the OK button, the tracing will proceed. On the other hand, if you choose to Convert Selection Edge, each of the areas will be selected and filled with black or white, and given a black stroke, even though you may have chosen to show 256 CMYK colors. Just for reference, clicking on just the bowtie and both eyes of the duck provided 1,189 objects with Trace Selection, and 22 objects with Convert Selection Edge using the same parameters for both scans. The upside to objectively selecting parts of a bitmap or vector graphic with the wand is that you don't have the extra parts of the image that you have when using the rectangular marquee method.

When the tracing was complete, colors were chosen with the Eyedropper tool and added to the Swatches panel. By selecting all objects on particular layers, colors were applied to the various elements. I ended up with ten colors, plus black and white. Finally, I used the Trace tool again to draw a rectangle around the duck, using the Outer Edge option. I chose to add a custom Brush stroke to the path, and sent it to the back of the layer stacking order. The default brush from FreeHand looked a little boring, so I used the Knife tool to cut the path at various intervals around the duck (see Image V).

In FreeHand, you can change the height and width of a bitmapped object, and you can rotate or skew the object, but you can't use the 3D Rotate tool, the Fisheye tool, Envelope distortion, and you can't apply the bitmap to the Perspective Grid. But we're artists, and we need to do stuff to pictures. How do we get around FreeHand's limitations? That's where turning bitmaps into vectors

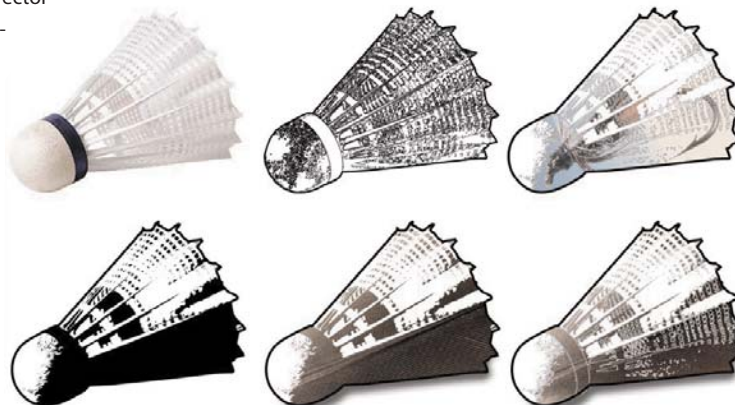


Image VII

comes in. I created a 3D child's block with the Extrusion tool. I wanted the duck (sans brush outline) and letterforms on the sides of the block. All I had to do was place an Envelope on the duck, and adjust the corners of the envelope to fit the face of the block, as shown in Image VI. The type characters could have been extruded with the Extrusion tool, but for the purpose of this short tutorial, they started as shadow color, and were converted to paths, then applied to the block using Envelopes. Then the letterform was cloned and shifted with the keyboard arrow keys to create a small amount of depth, and the color was lightened.

Masks and Clipping Paths

To continue on with the bird train of thought I'm on, I traced a badminton birdie. By the way, all the images in this article (except the assembly illustration near the end) came from the Hemera 50,000 Photo-Objects CD set; other images can be downloaded from their site at www.hemera.com. Image VII shows the original bitmap, followed by several versions of a single scan. The scan was done with eight (gray) colors. I moved each color to its own layer. Immediately to the right of the bitmap is the scan, with a white fill and paths stroked at a quarter-point. Still working on the bird metaphor, I thought of feathers, therefore fishing flies, and I used one of the layers in the scan as a clipping path. I placed the fishhook over the birdie and Cut it (Cmd/Ctrl+X). Then I selected the layer I wanted and chose Edit>Paste Inside, and turned off all the other layers except one that I gave a light gray stroke. That looked a little too viscous for a backyard game, so I did an Undo and filled my clipping layer with black, getting a nice black and white approach appropriate for clip art. I was on a roll, so I found a good feather image and did a Paste Inside - in just the black shape, but then I added the gray-stroked layer for a little color in the bottom-right image.

More Clipping Paths

I can't get enough of birds lately, so I imported an image of an eagle. This time, I used the Trace tool in the Outer Edge mode to get a really fast outline. I placed the eagle's outline over a flag

Advertiser	URL	Phone	Page
ActivePDF	www.activePDF.com		41
CFDynamics	www.cfdynamics.com	866-233-9626	9
FuseTalk	www.fusetalk.com	866-477-7542	13
HostMySite.com	www.hostmysite.com/mxdj	877-248-4678	47
Interakt	www.interaktonline.com		6
Macromedia	www.macromedia.com/go/dwupdated		Cover 2
Macromedia	www.macromedia.com/into		C4
Seapine Software	www.seapine.com	888-683-6456	27
Charles River Media	www.charlesriver.com	800-382-8505	21
Flashforward2004	www.flashforwardconference.com	877-435-2744	Cover 3

image, Cut the flag image, and pasted it inside the eagle shape (Edit > Paste Inside) as shown in Image VIII. Add a calligraphic stroke and you have instant patriotism.

Stacking Up

Depending on your settings, when a trace is created in FreeHand, it appears

pretty much the same as the original object. But the actual tracing is much different when you tear it apart. This is because individual colors are placed on separate levels. Not layers, but levels. There will be one color that completely encompasses the background, even though only a small area of that color appears in the tracing. If you have traced



Image VIII



from the pop-up window. In just a few seconds, the entire drawing had been converted to a compound path! I was

Illustrator, designer, author, and Team Macromedia member Ron Rockwell lives and works in the Pocono Mountains of Pennsylvania. He is the author of FreeHand 10 f/x & Design, and is about to introduce a FreeHand MX course. He has Web sites at www.nidus-corp.com and www.brainstormer.org. Contact him at guru@brainstormer.org with questions or article requests.

with 256 colors, you'll have 256 levels of chunky objects stacked on top of each other – and you'll watch the image build every time you change position on the screen. There are times you may want to edit color levels, delete them, or combine them with other colors. The ostrich in Image IX was scanned in only 4 colors. To separate those colors quickly, I chose Name All Colors from the Xtras menu (keep in mind that if you have other vector graphics in the document, their colors will also be named and could confuse your editing). Then I used Find & Replace to select each of the colors, which were then grouped and sent to their own new

layer. After a few minutes, each of the colors was available for editing or manipulating. On the right side of Image IX, I've separated the layers so you can see how the color levels are created. Notice that the bottom layer is a solid fill of black, and shows through holes in the dark green and other layers.

Getting Creative

Having completely exhausted bird relationships, I chose an assembly illustration that I needed to spice up for a manual. The problem with computer-generated illustration is that they usually look as if they'd been drawn on a computer – imagine that! It's an easy situation to get around, and there are a couple ways that FreeHand MX can help.

First, I had to select all the strokes in the drawing and convert them into a single compound path. This drawing was done with three or four line weights, and had been enlarged or reduced at various times, so it wasn't as simple as using the Find & Replace panel to select each of the stroke sizes, and then convert them to paths. That would take a lot of time and energy. There's also the chance that one or two paths would somehow be overlooked. But by using the Trace tool it was an easy matter of dragging the wand marquee across the drawing. I clicked the selection and chose Convert Selection Edge

warned that there were too many points to trace at the resolution I had selected, and I took the option to trace at a lower resolution. I deleted the stroke and applied a dark blue fill color. Due to the roughness of the tracing, I had a very casual drawing style working for me (as seen in the bottom half of Image IX) compared to the "computer generated" original (top half, same image). I added a Ragged vector effect in the Object panel, with three copies. The final effect is similar to a rough pencil sketch that has been tightened up with a felt tip pen. With or without the vector effect, the drawing certainly has lost its computer generated look.

Using the same tracing technique to select all the line work and create a single compound path, you have other options that are pretty easy, but look complicated. For one thing, you can apply a gradient to the compound path so the lines fade out or blend into the background. That gradient can be a linear gradient from left to right or up and down, or a radial gradient that fades out at the terminus of the lines. It's a quick trick to apply to soften up any drawing.

Summary

The Trace tool has been overlooked for many years, and it's too bad. There are many uses for the tool that will make your job easier. As with a lot of features and tools in drawing programs, it requires a bit of experimentation, but it is certainly time worth spending.

Acknowledgments

Many thanks to John Nosal, David Spells, Peter Moody, and other engineers at Macromedia for the technical editing they provide. ☺

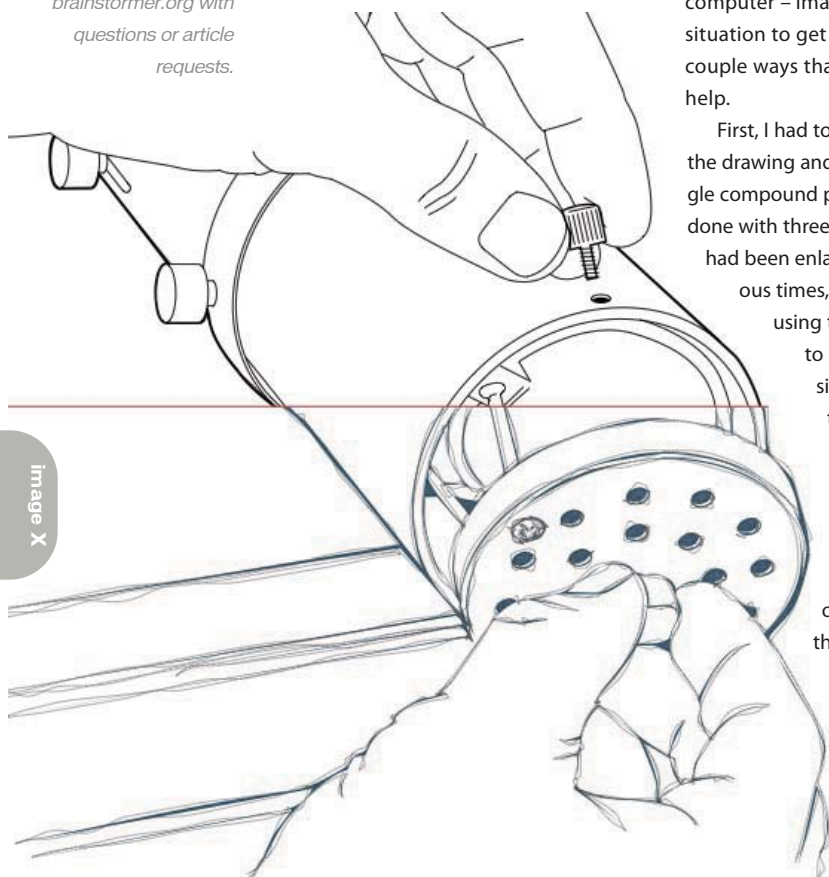


Image X



It's everybody's PDF™

Finally, a software company that offers affordable 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!



www.activePDF.com

FREAKS & GEEKS

The great thing about being a “Freak” and working with a “Geek” is not having to concern yourself with the nitty-gritty details of coding a dynamic site. The bad thing is that you will get involved with the nitty-gritty details whether you like it or not.





the
saga
continues

by tom green



In my previous article (MXDJ Vol. 1, issue 4), I walked you through my “epiphany.” I had redesigned my site, basked in the accolades of my students and colleagues, and then discovered to my chagrin that I had screwed up – big time. I had a tutorial area that was a classic case of a freak focusing on the design and not wondering, “Can anybody use this page?” There were a few dozen tutorials available but they were impossible to find.

The entrance page simply laid them out and said to the visitor, “You have a

brain. Figure it out.” This is a huge error and once I “figured it out,” I wandered down the hall from my office to that of my colleague James Cullin, the ideal ubergeek, and described my error. I also added the fact that, in my humble opinion, the solution was to get dynamic and could he help. Rather than point fingers or laugh up his sleeve at me, James did something that is so typical of James – he pulled out a notepad and said, “Let’s get to work.” We quickly sketched out a broad plan of attack and I left James alone.

This article describes how James, in typical geek fashion, attacked the problem. It shows how he worked from concept to code and offers you a few ideas as to how you can approach the task of building a database and then “hook” it into MySQL through ColdFusion MX. We started the task with good old-fashioned pragmatism.

Once we understood the scope of the project, our first question was, “How does it work?” It is all well and good, in typical freak fashion, to say, “Here’s what I want the page to look like.” It is also all well and good for the geek to say, “Here’s what I want the data to look like.” What both are overlooking is a fundamental question: How does the data get into the design? The first step for both of us was a rather intense discussion around that very question.

From a design point of view, there were a couple of approaches to solving this issue. The first was to simply flow the images and text into the page from the database. This seemed to be the ideal solution. I write the tutorial, do a few screen shots, toss it all into a database, and somehow the page, thanks to James, is magically constructed on the server and shot into the browser. As James and I fully explored this idea, it became evident it was the worst possible solution. This was due to the way the page is designed.

The page is constructed from a series of div tags (see Image I). All of the words and images are placed in a <div> with the ID of “content”. The major elements in this area are a headline, body text, sub-heads, images, and captions. As the freak, I controlled the placement of all of these elements, and there was no way to control the placement of the images and captions if they were to be inserted into the <div> on the server without a lot of unnecessary extra work on both our parts. Plan A, the most obvious solution, was discarded.

The solution we eventually arrived at is based upon how I work with Community MX. Having written for them for a couple of years, I always found the process to be rather smooth. All articles are contained in a folder, named using an article number generated by a database, and all of the images, pages, and uploads sit in that folder. This is a rather tidy solution to the problem at hand. Everything is

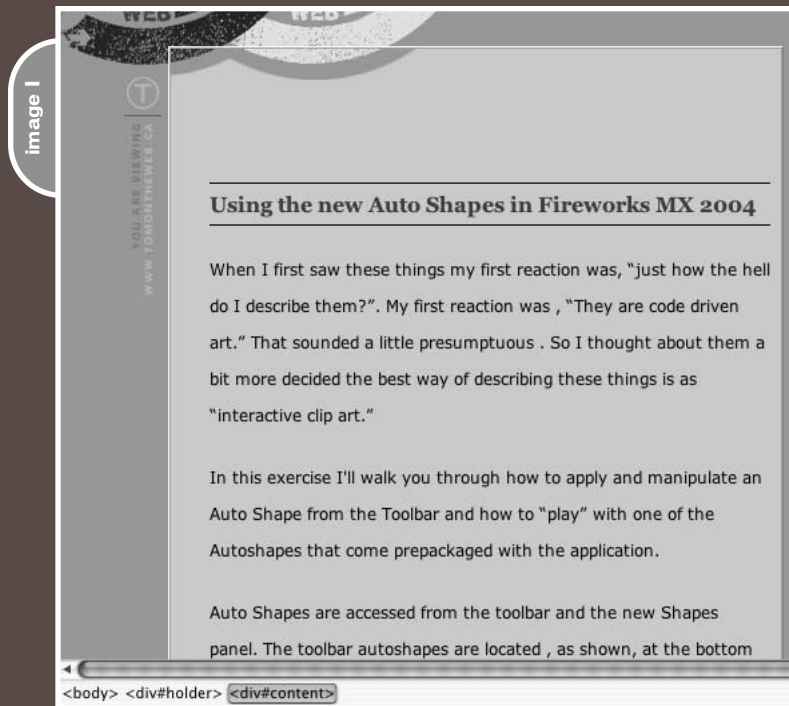


image I

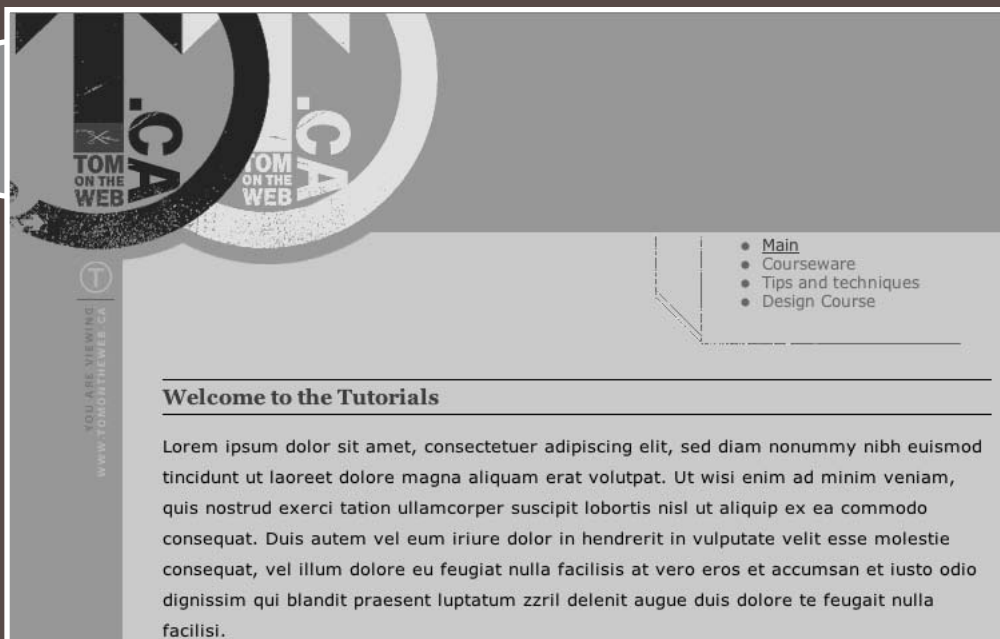


image II

in one place, making development and maintenance extremely easy.

We quickly decided this model met our needs. James saw how easy this would make his life and, rather than an article number, he suggested we use the date the article was created for the folder name. My first reaction was a bit negative. I foresaw a potential situation where two or three tutorials could be posted on the same day. James' response was essentially, "That's my problem. Worry about something more important." The solution, as you will see, had more to do with me than anything else.

What was important to me was how the user would locate a tutorial. I felt it important that the user be able locate it by either viewing all of the tutorials in a category or through a keyword search. I toyed with a couple of ideas, but the design of the page presented me with the most logical solution. On the right side of the page is a <div> that contains link information based upon the subject of the page. I decided to have the main content area welcome the user to the page and to use the link area for the search. When the user selects a subject or does a keyword search, the results flow into the "Content <div>". From there the user can select the tutorial and go to work.

This is an invaluable step because it gave James a clear idea of how the user would access the information and the functionality involved. It also gave both of us a roadmap to follow. The concept, in the form of a Dreamweaver MX 2004 page, was sent to James with the following question: "Is this doable?" This is an important question because it puts the freak and the geek on common ground. The geek can understand my thinking, and I learn (rather quickly) whether my idea is too ambitious. This is the best time to discover this because changes can easily be made. Discover it in the middle or at the end of the process and you are essentially relegated to starting all over again (see Image II).

Building the 'Tutorial Engine'

The first step was to build the MySQL database tables that reflect the architecture we had agreed to. James discovered that this job, based upon our discussions,

was a relatively simple undertaking because it only needed one table, as shown in Image III.

You will note the naming conventions are fairly intuitive. James' explanation for this should have several of you nodding and saying, "Been there, brother."

"Every table I create starts with 'tbl_', said James. "I wish I could say I had this good sense right from the start but I can't. It is only after your 9th or 10th 'all-nighter' where you meet a deadline by the skin of your teeth that you realize there just has to be a better way. When it is 3:00 a.m., and you started work at 8:00 a.m., you will discover that you either develop a distinct naming pattern or just live with the confusion."

This logical naming convention is used throughout. For example, the primary key in the database uses "pk_" in the name. Columns use "col_" and so on. This not only helps James build the database using clearly labeled elements but also it gives anybody else maintaining the database a very clear idea of what's what.

When geeks and freaks work together for as long as James and I have, they tend to become sensitive to each other's quirks. For example, the DataType for the article titles, "col_title" is text. By doing

this James allows me the flexibility to get a bit wordy with the titles. This DataType supports from 1 to 65,535 characters or about 8,000 words without truncating the title. I am not to sure whether this was shrewd planning or self defense because, according to James, "I can sleep easy knowing there is no way in hell Tom will ever have his title truncated."

The rest of the database design follows a similar plan. Using the category drop-down list from the concept shown in Image I, James created a category column that plans for product names of more than eight words. There is a column for the abstracts I will write and it, again, allows up to 8,000 words. The Key Words column allows for the search. The interesting aspect of that name is the use of the upper case lettering in the name. "As my naming convention evolved," said James, "I found it confusing to have multiple underscores. I established that if a

Image III

Name	Type	Null	Default	Extra
pk_ArticleID	int(3)	No		auto_increment
col_title	text	Yes		
col_category	text	Yes		
col_description	text	Yes		
col_KeyWords	text	Yes		
col_DatePosted	datetime	Yes		
col_FolderName	text	Yes		

Image IV



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)




```

AddArticle.cfm  index.cfm
Code Split Design Title: TomOnTheWeb.ca | Add Article
39 <cfquery name="InsertArticleInformation"
40     datasource="Cold_Fusion_DataSource_Goes_Here"
41     username="Your_MySQL_UserName_Goes_Here"
42     password="Your_Password_Goes_Here">
43
44 INSERT into tbl_articles (col_title, col_category, col_description, col_KeyWords, col_DatePosted, col_FolderName)
45 VALUES ('#form.title#', '#form.category#', '#form.description#', '#KeyWords#', #DatePosted#, '#FolderName#')
46
47 </cfquery>

```

were going to be blank fields when I clicked the "Create" button. Line 3 is a rather elegant solution to this issue.

The `<cfif>` tag inspects the first field to see if it is blank. James also knows that if there is a way to screw it up, I will find it. For example, my "accidental" use of the spacebar has resulted in some pretty legendary stories amongst our faculty. This is why he added the ColdFusion "trim" function to strip out any "accidental" spaces to the right or the left of the keyword. If the field is blank, there will be no value in the field. This explains the `neq` operator. It is the way one expresses "not equal to" in ColdFusion MX and, in this case, if there is a word, then the value is not equal to null (the empty quotation) and the next line of code is executed. If it does equal null, then the code skips to the next CFIF statement and repeats the process.

The next line – `<cfset KeyWords = "#KeyWords##form.kw1#"` – tosses the word into the string that is being assembled and the XHTML form variable being used is `#form.kw1#`. The `"#"` signs enclose each of the two variables and the comma at the end is used because the keywords in the database will be separated by commas. This whole process repeats itself nine times until the string is assembled in the database.

With keywords out of the way, James next turned his attention to the date. This too is rather interesting because the form kicks out three values – Month, Date, Year – while there is only one column for the date in the database. The code used to assemble this single date element is:

```

<cfset DatePosted =
CreateDate(form.YearNumber, form.MonthN
umber, form.DayNumber)>

```

The ColdFusion MX function `CreateDate` is used to create the data string for the "DatePosted" variable. This function requires three arguments for Year, Month, and Day, which must have numeric values. This wasn't terribly difficult to accomplish because James did

just that when he coded the drop-down menu earlier.

The final issue was making the folder-naming process idiot-proof. The name is set using this code:

```

<cfset FolderName =
"#form.category##form.DayNumber##form.
MonthNumber##form.YearNumber#">

```

"I created a variable named `FolderName` and, to make it unique, I assemble the `FolderName` using the category of the article Tom chooses and the three date elements." says James. "My assumption here is that Tom won't post two Dreamweaver articles on the same day." I am glad I asked. This is important information for me to be aware of.

From the Page into the Database

With the data assembled, the time had arrived to move it into the database and, according to James, "ColdFusion MX makes inserting a record into a database table incredibly easy." The code that accomplished this task is shown in Image IX.

Lines 39 to 42 are the opening `<cfquery>` tag. Each ColdFusion query has a name and James' protocol is to make the name as verbose and intuitive as possible. "I learned that lesson the hard way," he told me. "When a file is due at 9:00 a.m. and it is 3:00 a.m., the last thing you need is to be wondering which query does what. If the name is verbose and obvious, even the most sleep-deprived coder can figure it out."

The `datasource` is how ColdFusion MX maps to the database. In this case, our ISP has configured our ColdFusion MX server so that each site has a `datasource` that maps to a corresponding MySQL account. To access that account, we need a username and password which are detailed in the `<CFQUERY>` tag.

Lines 44 and 45 are not ColdFusion code. Those lines are pure SQL, which is a

database manipulation language. The first line may, at first glance appear to be a bit convoluted:

```

INSERT into tbl_articles (col_title,
col_category, col_description,
col_KeyWords, col_datePosted,
col_FolderName)

```

In fact it is simply saying where to insert the values from the `AddArticles.cfm` page into the database. The locations are contained in the brackets. Now that we know where the values are to be placed we also have to answer the question, "What values?" The next line of code handles this:

```

VALUES ('#form.title#', '#form.catego-
ry# ',
#form.description#, '#KeyWords#', '#Dat
ePosted#', '#FolderName#')

```

What is absolutely critical in this code line is that the order of the items in the `VALUES` statement precisely matches the order of their counterparts in the `INSERT` statement. As well, the "title", "category", and "description" come from the form created in the Admin page, which explains why they are prefaced with "form". One item, "KeyWords", doesn't use this convention because its value was established in line 1 of this page's code.

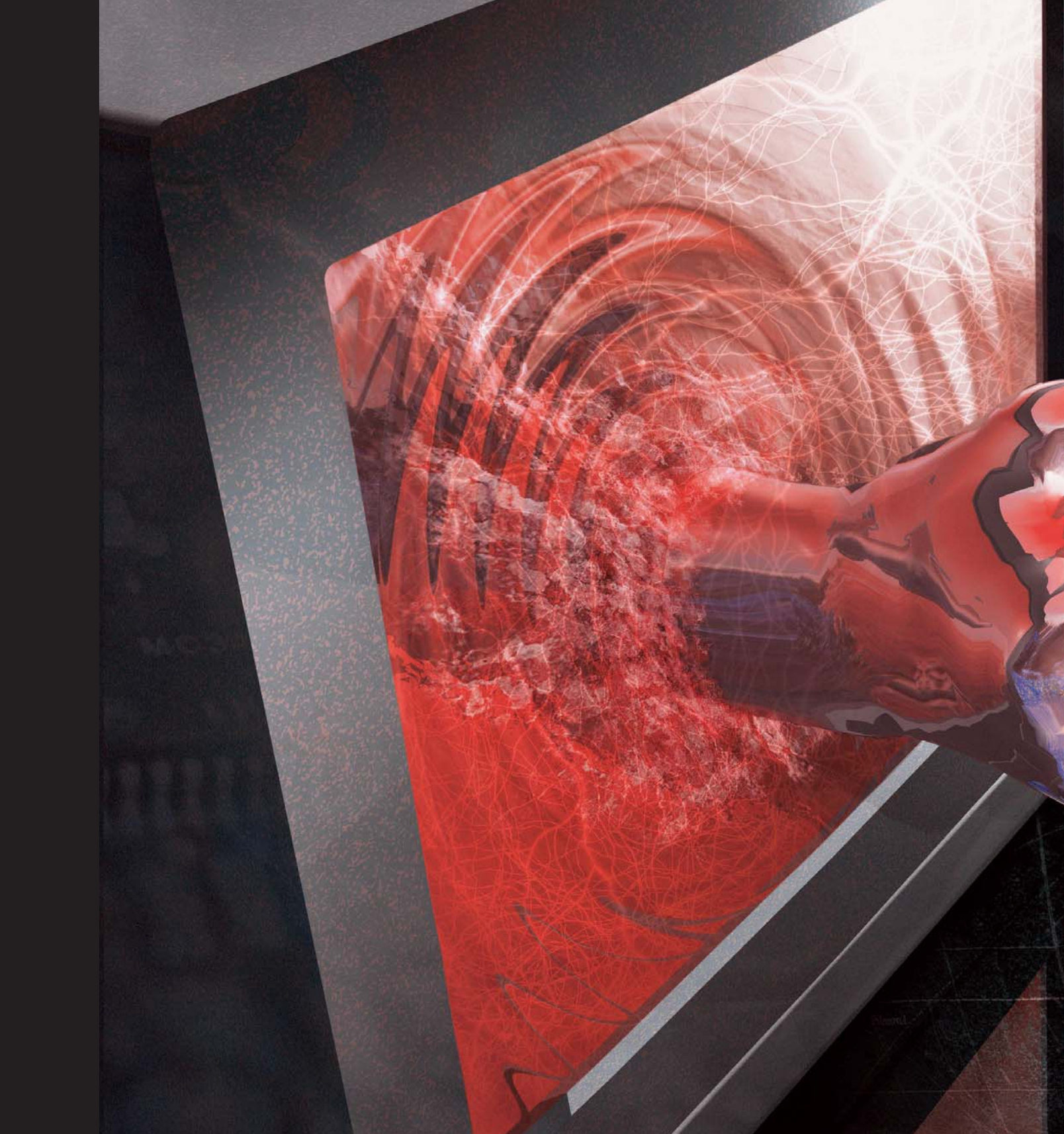
You may also have noticed the variables are surrounded by single quotes. This is because the information is going into the database as text.

Another inconsistency you may have noted is the variable being inserted in the "col_FolderName" column of the database. It is named "FolderName". This variable name was also set earlier in the page.

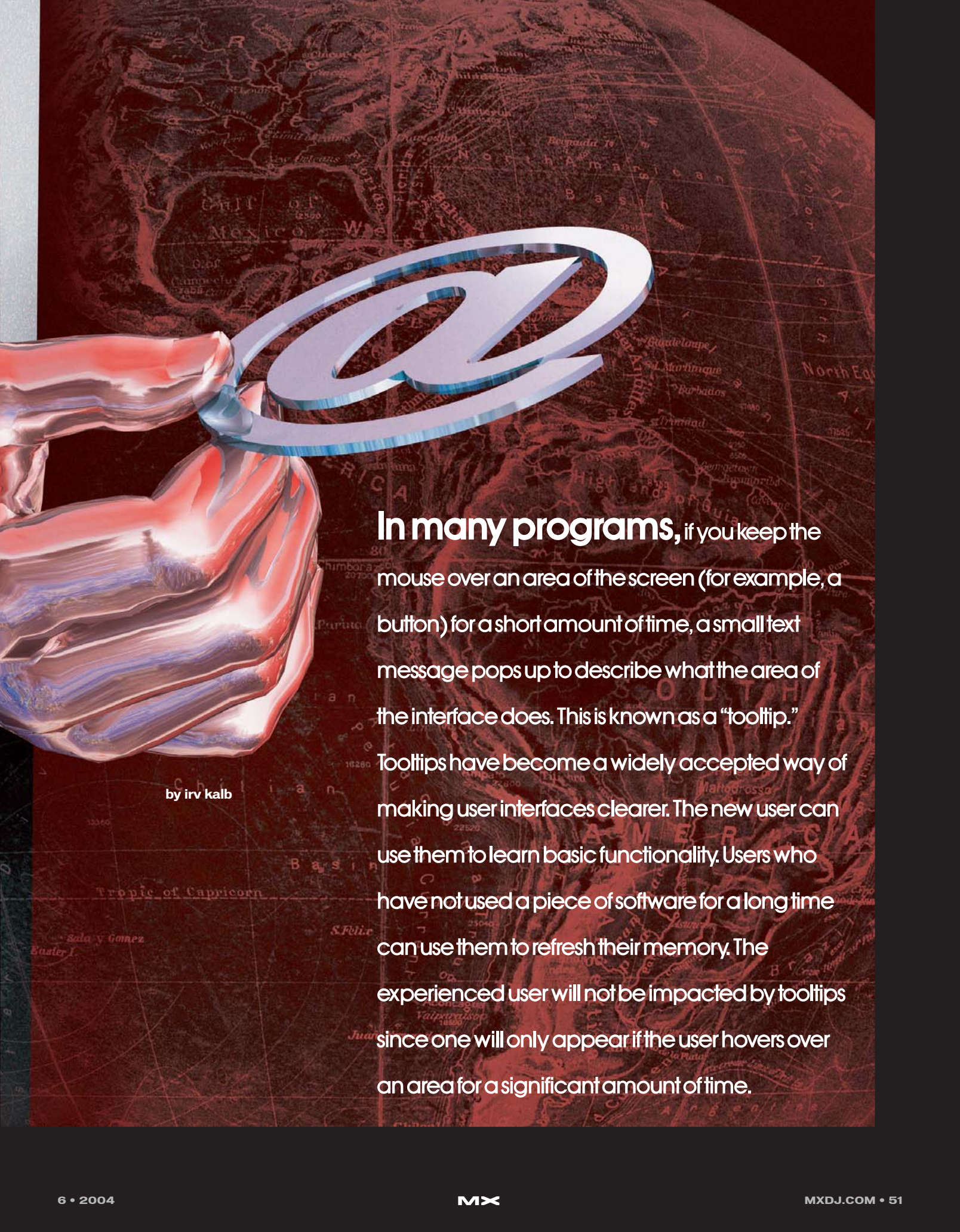
Now that we have the data I input sitting in the database, how does it appear on the page? That, ladies and gentlemen, is a whole other story in the *Freaks and Geeks* saga. ☺

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.

The course coordinator and "Lead Geek" for the Internet Management and Interactive Multimedia programs through the School of Media Studies at Humber College in Toronto, James Cullin currently teaches courses in Internet technology and Web programming. tgreen@coge-co.ca



Building Tooltips



by irv kalb

In many programs, if you keep the mouse over an area of the screen (for example, a button) for a short amount of time, a small text message pops up to describe what the area of the interface does. This is known as a “tooltip.”

Tooltips have become a widely accepted way of making user interfaces clearer. The new user can use them to learn basic functionality. Users who have not used a piece of software for a long time can use them to refresh their memory. The experienced user will not be impacted by tooltips since one will only appear if the user hovers over an area for a significant amount of time.



Even in Director, the meanings of some of the buttons might not be immediately clear. In the Director main toolbar, there is a button with two right-angled arrows. The novice Director user might think that this is some sort of traffic diagram from England showing where cars may turn left (see Image I).

However, if you move the mouse over this button for a while, a tooltip shows up to explain the meaning of this button (see Image II).

This article shows how to build a set of two related behaviors that will implement tooltips in Director. The first behavior will be the "Tooltip Rollover." You can attach this behavior to any interface element you wish to have a tooltip. The other behavior will be the "Tooltip Display" behavior. This behavior will be attached to a single sprite made up of a field member. This one sprite will be used to display the text of the tooltip.

We'll discuss the following concepts:

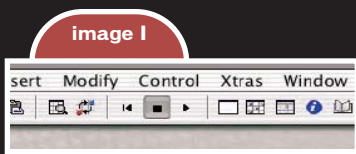
- Instances of behaviors
- Events and event handlers
- State machines
- Timing in milliseconds
- Intersprite communication
- Calculating positions and rectangles on the Stage

Tooltip Rollover

First we'll discuss and write the Tooltip Rollover behavior. The Tooltip Rollover behavior will detect when the mouse has been within the rectangle of a sprite for a given amount of time, and then send the text to be displayed to the Tooltip Display sprite that lives in a different channel.

Let's think about the essential pieces before writing some code. Among the things we must do are the following:

- Determine what text should be displayed



- Build a timing mechanism to see if the user has kept the mouse inside the rectangle of the sprite for an appropriate amount of time
- Find out where the Tooltip Display sprite is
- Send text to the Tooltip Display sprite at the appropriate time

Text to Display

In Director each cast member can have a name. For simplicity, we will start by using the name of the rolled over member as the text of the tooltip. We can get this name by using the following code:

```
property spriteNum
property pName

on beginSprite me
    pName = sprite(spriteNum).member.name
end
```

"spriteNum" is a special property variable that can be used in behaviors. When it is declared as a property and used in a behavior, Director automatically gives it the number of the channel to which the behavior is attached.

One of the big buzz phrases in programming these days is object-oriented programming (also known as OOP). Behaviors are a great way of learning more about OOP. Programmers write behavior scripts. But when a behavior script is attached to a sprite, Director creates a new object from the script. Each such object is called an "instance" of that behavior.

Here's a simple example. Create a new Director movie. Go to the paint window and create a square. Name the member "Square". Then in the paint window, create a circle and name the member "Circle". Now open a script window and enter the code above. Before the last line (the "end" line) of the beginSprite handler, add the following line:

```
put "spriteNum" && spriteNum &&
    "pName" && pName
```

Name this script "Tooltip Rollover". Using the Property Inspector, make sure that you set the type of this script to "Behavior".

Now drag the Square member into channel 1, and drag the Circle member

into channel 2. Finally drag the Tooltip Rollover behavior script onto both sprites. Now run the movie.

If you open the message window, you should see the following:

```
--"spriteNum 1 pName Square"
--"spriteNum 2 pName Circle"
```

At run time, Director creates two instances (objects) from the single behavior script. Each instance uses the exact same code, but each instance gets its own copy of the data – the property variables. Each gets its own copy of spriteNum and its own copy of pName. As we will see in a moment, the value of any property variable declared in a behavior script is "remembered" and can be accessed by any other handler in that script.

Any change you make to the code in a behavior script will affect all sprites to which the behavior is attached. This is a very important part of object oriented programming.

Timing

Now we need to build a way for the behavior script to know when is the right time to show a tooltip. Again for simplicity, we'll just pick a set amount of time. Let's say that we will trigger a tooltip when the user has kept the mouse within the rectangle of the rolled-over item for one second.

To build this timing mechanism, we must understand two important concepts: the "on exitFrame" handler and the concept of a state machine. At each frame, Director sends out special messages to all sprites. One of these messages is the exitFrame message. The rate at which this message is sent is dependent on the frame rate of the Director movie. In a behavior, you can write an on exitFrame handler to receive this message. The content of the exitFrame message is completely up to the programmer. You can tell Director to do anything you want it to do on each such call. A "trick" is to make the same handler do different things under different circumstances. In our case, in the exitFrame handler, we can constantly check how long the mouse has been within our rectangle. If it has been over a second, we can make a call to display the tooltip.

But how do we do this checking? The answer lies in a concept called a state machine. When implemented in software, a state machine is a piece of code that does something different based on the value of a variable. This maps nicely into a Lingo case statement:

```
case someVariable of:
  State1:
    --execute this code here
  State2:
    -- execute this code here
  ...
  StateN:
    -- execute this code here
end case
```

The key is that in any branch of the state machine – or in other handlers in the same script – we can change the state to some other state.

Here’s how we’ll apply this idea to our Tooltip Rollover behavior. We will declare a property called `psymState` that will be used to keep track of what state the current behavior is in. In thinking through the different states, the behavior will be in one of three states:

- `#notOver`: The mouse is not over our rectangle
- `#overAndWaiting`: The mouse is over our rectangle and we are waiting for one second
- `#showing`: The tooltip is showing

We’ll start `psymState` in the `#notOver` state. Whenever the mouse enters our rectangle, we should go into the `#overAndWaiting` state. When we are in this state for one second, we trigger the tooltip to display, and change our state to `#showing`. Whenever the mouse leaves our rectangle, we hide the tooltip and change the state to `#notOver`.

In computer science classes, students are asked to draw “state diagrams” to describe these type of interactions. Another approach is to create a table with a list of events as rows (`exitFrame`, `mouse-Enter`, `mouse-Leave`), and a list of states as columns (`#notOver`, `#overAndWaiting`, `#showing`). This is often a good idea to be sure to account for all cases. In each cell, you describe what action takes place and any change in the state variable. Then the array is turned into code.

However, this state machine is rather simple and will become clearer with some simple code. Code I shows what this state machine would look like in pseudo-Lingo. Notice the addition of the Lingo on `mouseEnter`, on `mouseDown`, and on `mouseUp` handlers to handle these Lingo events.

By declaring `psymState` as a property variable, it means that each instance of this behavior will have its own version of `psymState`. Further, `psymState` can be used in any handler in the script, and its value will be remembered across these calls. For example, when the user moves the mouse into the rectangle of the sprite, the on `mouseEnter` handler is called by Director, and here we set the state to `#overAndWaiting`. The `exitFrame` handler is called many times each second. When the value of `psymState` changes, the `exitFrame` handler will do something different – it will take a different branch in the case statement.

To finish the timing aspect, we need to figure out how to count from zero to one second. There are many ways to do this, but here is the simplest. When the user first brings the mouse over the sprite, we’ll calculate what time it will be one second from that point. Lingo keeps time in milliseconds. You can find out the current value of milliseconds from when the computer was turned on by using the Lingo function “the milliseconds”. To find one second from the current time, we can add 1,000 to the milliseconds. We can save this value in a new property variable like this:

```
pmsShowTime = the milliseconds + 1000
```

Finally, in the `exitFrame` handler, in the `#overAndWaiting` part of the case statement, we can check for the one second elapsed by checking if the current value of the milliseconds is greater than the `pmsEnd` that we calculated earlier. If so, then we show the tooltip and change our state to `#showing`. The check looks like this:

```
If the milliseconds > pmsShowTime
then
  -- Show the tooltip
  psymState = #showing
end if
```

It turns out that in this behavior script, the `#waiting` and `#showing` states don’t really do anything, and they could be collapsed into a single state. However, for the sake of clarity, I like to show all possible states. It is also a good idea to keep these states in the code in case you decide to do something else when in one of these states.

There is one extra note here. If the rolled-over area is not a rectangle, for example a circle, you need to use `Matte Ink` in the score. When using `Matte Ink`, Director will send the `mouseEnter` and `mouseLeave` messages only when the mouse enters and leaves a pixel that is in the cast member, not just within the rectangle of the sprite.

InterSprite Communication

The main thing that’s left to do is to send a message to the `Tooltip Display` to show the tooltip. Obviously we have not written the `Tooltip Display` behavior yet. However, we can now define what messages we expect to send it. This is known as defining the “interface” of a behavior or an object. We know from our pseudo-code above that we will need to tell it to show and hide the tooltip. Clearly, we will need to define one handler for each of these actions. However, when we want to show a tooltip, thinking ahead, the code to show a tooltip will need to know two things: the text to display and the rectangle of the rolled-over item (in order to position the tooltip accurately). When the behavior instance starts up, we can save away the rectangle of the sprite in a property variable. When we call the `show` routine, we will need to pass in the text and this rectangle.

The final issue is to find out to which channel the `Tooltip Display` behavior instance is attached. The way to do this is to broadcast a message to all sprites and ask them if they have the `Tooltip Display` behavior attached. This can be done using `sendAllSprites`. But `sendAllSprites` can be “expensive” in terms of time, so we don’t want to do this a lot. Here’s how we will handle this: the first time we show a tooltip, we will use a `sendAllSprites` to send out a special message asking for the sprite number of the tooltip display sprite. When we find out the channel number, we will save it in yet another property variable. All property variables



(in fact, all variables) start out with a value of VOID. We can use the voidP() function to check if we have figured out the channel number. From then on, whenever we want to send messages to the Tooltip Display behavior, we will use the saved channel number. When we want to send a message to the specific sprite with the Tooltip Display behavior attached, we use the sendSprite command.

Tooltip Rollover Code

Putting all this together, we get the code for the Tooltip Rollover (see Code II).

Notice that I added an on endSprite handler to the end of the script. This just sends a message to the Tooltip Display sprite to turn off the tooltip when the Rollover sprite ends.

Tooltip Display

Now we have to build the Tooltip Display behavior script. We know from our earlier discussion of the “interface” of this behavior, that this behavior script will at least need a Show, Hide, and GetChannel handler. Let’s start by building a skeleton behavior script (see Code III).

The outline in Code III gives us most of the functionality we need. In fact, at this point, you can test if the intersprite communication is working correctly. To do this, create a field cast member and give it a name like “Tooltip Display Field”.

Drag it to the stage and place it in the highest-numbered channel. This way, the tooltip will show over all other sprites. In practice, this sprite will need to be stretched across the entire movie so that Tooltip Rollover behaviors can communicate with it from anywhere in the movie.

Now create a new behavior script, call it “Tooltip Display”, and enter the code from Code III. Be sure to make it of type Behavior in the Behavior inspector. Finally, drag the behavior from the cast and drop it onto the field sprite in the score. Open the message window and run the program. When you hover over the square or the circle for one second, the word square or circle should appear in the message window, along with the appropriate rect.

Formatting the Text

When dealing with a field member, there are many different properties that can be manipulated to alter the way the text is displayed. You can find a list of these properties by clicking on the member in the cast, then looking at the field tab in the Property Inspector (see Image III).

Of these properties, for a tooltip display you will want to set:

- editable to false
- boxType to #adjust
- wordWrap to true

You can experiment with and customize many of the other properties here (e.g., border, color, bgColor, font, fontStyle, fontSize, etc.) so that the tooltip will appear the way you want it. Any changes you make here will be reflected for the display of all tooltips.

However, since the text to be displayed in the tooltip changes each time you rollover a different item, the rectangle (width and height) of the member needs to be set on the fly. Therefore, setting the rectangle must be done in code.

To begin the full Tooltip Display behavior, we will define a few properties for saving information, and set a few property variables as constants to be used in calculations later. At the end of beginSprite handler, we make a call to a move the tooltip off-screen so it is not

visible at the start of the program (see Code IV).

An important thing to notice here is that in the first line, we “cache” away a reference to the tooltip member into pmTooltipText. This will make it easier to refer to the member in the rest of the code. Since Lingo only has to figure out this member reference once, it helps make the code fast. You’ll also notice that one of the properties we created was called pMaxWidthTooltip. This will be used to set a maximum width for the tooltip. If the text goes over the maximum width, the text will wrap onto two or more lines. We’ll pick an arbitrary size of 200 pixels to start with.

As we discussed earlier, when an instance of the Tooltip Rollover behavior instance calls the Tooltip Display instance, it will pass in the text to display, and the rectangle of the rolled over item. Now we have all the information we need to code the mTooltipDisplay_Show handler.

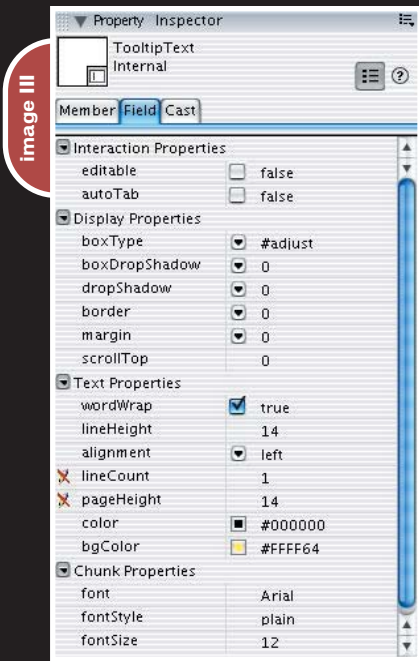
To size the rectangle properly for the tooltip, we first set the rectangle of the member our maximum width. Then we put the text into the display member:

```
on mTooltipDisplay_Show me, theString,
rectRolledItem
    member(pmTooltipText).rect = rect(0,
0, pMaxWidthTooltip, 0)
    member(pmTooltipText).text =
theString
```

When we put the text into the member, Director automatically adjusts the height of the rectangle for us, but it leaves the width at pMaxWidthTooltip. For multiline tooltips this is fine. However, if the tooltip is only one line of text, then we must manually adjust the right side of the tooltip member’s rectangle.

Lingo has a routine called charPosToLoc that tells us the position of any character in a string. We want to know the position of the right edge of the last character. To do this, we ask Director the position of the number of characters in the string, plus one. While this is a non-existent character, Director gives us the width of the full string. Knowing this width, we can adjust the rest of the member. Here’s how:

```
if pmTooltipText.lineCount = 1 then
    -- Only one line, must shrink the
```



SUBSCRIBE TODAY TO MULTIPLE MAGAZINES

AND SAVE UP TO \$340 AND RECEIVE UP TO 3 FREE CDs!*

RECEIVE YOUR DIGITAL EDITION ACCESS CODE INSTANTLY WITH YOUR PAID SUBSCRIPTIONS



3-Pack
Pick any 3 of our magazines and save up to **\$210**! Pay only \$99 for a 1 year subscription plus a **FREE CD**

- 2 Year - \$179.00
- Canada/Mexico - \$189.00
- International - \$199.00

6-Pack
Pick any 6 of our magazines and save up to **\$340**! Pay only \$199 for a 1 year subscription plus **2 FREE CDs**

- 2 Year - \$379.00
- Canada/Mexico - \$399.00
- International - \$449.00

9-Pack
Pick 9 of our magazines and save up to **\$270**! Pay only \$399 for a 1 year subscription plus **3 FREE CDs**

- 2 Year - \$699.00
- Canada/Mexico - \$749.00
- International - \$849.00

CALL TODAY! 888-303-5282

Pick a 3-Pack, a 6-Pack or a 9-Pack

<input type="checkbox"/> 3-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 6-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.
<input type="checkbox"/> 9-Pack	<input type="checkbox"/> 1YR	<input type="checkbox"/> 2YR	<input type="checkbox"/> U.S.	<input type="checkbox"/> Can/Mex	<input type="checkbox"/> Intl.

TO ORDER •Choose the Multi-Pack you want to order by checking next to it below. •Check the number of years you want to order. •Indicate your location by checking either U.S., Canada/Mexico or International. •Then choose which magazines you want to include with your Multi-Pack order.

LinuxWorld Magazine

U.S. - Two Years (24) Cover: \$143	You Pay: \$79.99 / Save: \$63 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 / Save: \$32
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 / Save: \$48 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$79.99 / Save: \$4
Intl - Two Years (24) \$216	You Pay: \$176 / Save: \$40 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 / Save: \$8

JDJ

U.S. - Two Years (24) Cover: \$144	You Pay: \$99.99 / Save: \$45 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$69.99 / Save: \$12
Can/Mex - Two Years (24) \$168	You Pay: \$119.99 / Save: \$48 + FREE \$198 CD
Can/Mex - One Year (12) \$120	You Pay: \$89.99 / Save: \$40
Intl - Two Years (24) \$216	You Pay: \$176 / Save: \$40 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 / Save: \$8

Web Services Journal

U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 / Save: \$68 + FREE \$198 CD
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 / Save: \$14
Can/Mex - Two Years (24) \$192	You Pay: \$129 / Save: \$63 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$89.99 / Save: \$6
Intl - Two Years (24) \$216	You Pay: \$170 / Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 / Save: \$8

.NET Developer's Journal

U.S. - Two Years (24) Cover: \$168	You Pay: \$99.99 / Save: \$68 + FREE \$198 CD
U.S. - One Year (12) Cover: \$84	You Pay: \$69.99 / Save: \$14
Can/Mex - Two Years (24) \$192	You Pay: \$129 / Save: \$63 + FREE \$198 CD
Can/Mex - One Year (12) \$96	You Pay: \$89.99 / Save: \$6
Intl - Two Years (24) \$216	You Pay: \$170 / Save: \$46 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$99.99 / Save: \$8

Information Storage + Security Journal

U.S. - Two Years (24) Cover: \$143	You Pay: \$49.99 / Save: \$93 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 / Save: \$39
Can/Mex - Two Years (24) \$168	You Pay: \$79.99 / Save: \$88 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$49.99 / Save: \$34
Intl - Two Years (24) \$216	You Pay: \$89.99 / Save: \$126 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$59.99 / Save: \$48

MX Developer's Journal

U.S. - Two Years (24) Cover: \$143	You Pay: \$49.99 / Save: \$93 + FREE \$198 CD
U.S. - One Year (12) Cover: \$72	You Pay: \$39.99 / Save: \$32
Can/Mex - Two Years (24) \$168	You Pay: \$79.99 / Save: \$88 + FREE \$198 CD
Can/Mex - One Year (12) \$84	You Pay: \$49.99 / Save: \$34
Intl - Two Years (24) \$216	You Pay: \$89.99 / Save: \$126 + FREE \$198 CD
Intl - One Year (12) \$108	You Pay: \$59.99 / Save: \$48

ColdFusion Developer's Journal

U.S. - Two Years (24) Cover: \$216	You Pay: \$129 / Save: \$87 + FREE \$198 CD
U.S. - One Year (12) Cover: \$108	You Pay: \$89.99 / Save: \$18
Can/Mex - Two Years (24) \$240	You Pay: \$159.99 / Save: \$80 + FREE \$198 CD
Can/Mex - One Year (12) \$120	You Pay: \$99.99 / Save: \$20
Intl - Two Years (24) \$264	You Pay: \$189 / Save: \$75 + FREE \$198 CD
Intl - One Year (12) \$132	You Pay: \$129.99 / Save: \$2

WebSphere Journal

U.S. - Two Years (24) Cover: \$216	You Pay: \$129.00 / Save: \$87 + FREE \$198 CD
U.S. - One Year (12) Cover: \$108	You Pay: \$89.99 / Save: \$18
Can/Mex - Two Years (24) \$240	You Pay: \$159.99 / Save: \$80 + FREE \$198 CD
Can/Mex - One Year (12) \$120	You Pay: \$99.99 / Save: \$20
Intl - Two Years (24) \$264	You Pay: \$189.00 / Save: \$75
Intl - One Year (12) \$132	You Pay: \$129.99 / Save: \$2

PowerBuilder Developer's Journal

U.S. - Two Years (24) Cover: \$360	You Pay: \$169.99 / Save: \$190 + FREE \$198 CD
U.S. - One Year (12) Cover: \$180	You Pay: \$149 / Save: \$31
Can/Mex - Two Years (24) \$360	You Pay: \$179.99 / Save: \$180 + FREE \$198 CD
Can/Mex - One Year (12) \$180	You Pay: \$169 / Save: \$11
Intl - Two Years (24) \$360	You Pay: \$189.99 / Save: \$170 + FREE \$198 CD
Intl - One Year (12) \$180	You Pay: \$179 / Save: \$1

*WHILE SUPPLIES LAST. OFFER SUBJECT TO CHANGE WITHOUT NOTICE

Subscribe Online Today www.sys-con.com/2001/sub.cfm





```
width to match the text
    nChars =
pmToolTipText.text.char.count
    locLastChar =
pmToolTipText.CharPosToLoc(nChars + 1)
    textMemberRect = pmToolTipText.rect
    textMemberRect.right =
locLastChar.locH
    pmToolTipText.rect = textMemberRect
end if
```

Placing the Tooltip

Field members work differently from graphics members as far as the registration point is concerned. Since you can change text on the fly, field members have their registration point in the upper left hand corner. This makes the math slightly tricky. We will calculate the horizontal and vertical locations to display the tooltip separately.

We want to position the tooltip centered on, and a little below the rolled item. Earlier we declared a property variable, pnPixelsBelow, and set it a small number of pixels (4) to give us some separation from the rolled item. To display the tooltip we make the calculations shown here:

```
-- Calculate locH and locV
centerH = (rectRoll.left +
rectRoll.right) / 2
locHTip = centerH -
(pmToolTipText.width / 2)
locVTip = rectRoll.bottom +
pnPixelsBelow
```

Finally, we use the locV and the locH we just calculated to properly position the tooltip:

```
sprite(spriteNum).loc = point(locHTip,
locVTip)
```

Code V is the resulting full script of the Tooltip Display behavior.

Extensions

We have shown the basic functionality of tooltips. In practice, there are a few things we would want to do to extend these behaviors.

In the Tooltip Rollover behavior, we would want the ability to alter the text to be displayed. Ideally, by default we could use the name of the cast member, but we would like to have the ability to change

that to any text. Further, we would like to have the ability to change the delay period. While one second may be fine, we might like to set some tooltips to show up in less or more time.

Both of these can be accomplished relatively easily. To allow for these types of modifications, there is a very powerful handler called getPropertyDescriptionList that can be built into the Tooltip Rollover behavior. Using getPropertyDescriptionList, we can specify that when the developer drops a behavior onto a sprite, Director will bring up a dialog box that shows defaults, but allows the developer to change values.

We would probably want the Tooltip Display behavior to be a little smarter about placement of the tooltip. If an interface element is too close to the bottom of the screen, we would want the tooltip to display above the item. Likewise, if the interface element is too close to the left or right edge, we would want the display to be positioned so that the entire tooltip could be readable on the stage.

Further, we might want to be able to modify more of the field properties such as Margin and Framing, and have the Tooltip Display behavior adjust automatically. Doing so requires a little more math in the mToolTipDisplay_Show method.

I am placing a fully functional and fully documented set of behaviors that includes these extensions in a sample movie on my Web site at: http://furrypants.com/ftp/tooltips_dir.

Conclusion

Tooltips have become valuable additions to improving the clarity of user interface designs. Using the two drag-and-drop behaviors described here, Director developers can now add this functionality quickly and easily. ∞

Irv Kalb has been working as an independent software developer in Director and Lingo for over ten years. He has written extensively on object oriented programming in Lingo, and his on-line Ebook on this topic can be found at www.furypants.com/loope. Irv is always interested in discussing new projects. IrvKalb-MXDJ4114@mailblocks.com

```
property psymState

on beginSprite me
    psymState = #notOver
end

on mouseEnter me
    psymState = #overAndWaiting
    * Start a timer *
end mouseEnter

on mouseLeave me
    * Hide the tooltip *
    psymState = #notOver
end mouseLeave

on mouseUp me
    * Hide the tooltip *
    psymState = #notOver
end

on exitFrame me
    case psymState of
        #notOver:
            nothing

        #overAndWaiting:
            * if we've been in this state for
            more than 1 second then *
                * show the tooltip *
                psymState = #showing
            end if

        #showing:
            nothing
    end case
end
```

```
property spriteNum -- the channel in
which thhe sprite is located
property pName -- the text to display
property pmsShowTime -- the milliseconds
at which to show the tooltip
property psymState -- #notOver,
#overAndWaiting, #showing
property pRect -- the rectangle of
the rollover area
property pchToolTipDisplay -- channel
number of the Tooltip Display sprite

on beginSprite me
    pName =
sprite(spriteNum).member.name
    psymState = #notOver
    pRect = sprite(spriteNum).rect
end beginSprite

on mouseEnter me
    psymState = #overAndWaiting
    pmsShowTime = the milliseconds +
1000
```

code I

code II


```

end mouseEnter

on mouseLeave me
    sendSprite(pchToolTipDisplay, #mToolTipDisplay_Hide)
    psymState = #notOver
end mouseLeave

on mouseUp me
    sendSprite(pchToolTipDisplay, #mToolTipDisplay_Hide)
    psymState = #notOver
end

on exitFrame me    case psymState of
    #notOver:
        nothing

    #overAndWaiting:
        -- See if it's time to show the tooltip
        if the milliseconds > pmsShowTime then
            me.mShowToolTip()
            psymState = #showing
        end if

    #showing:
        nothing
end case
end

on mShowToolTip me
    -- The first time, the channel of the tooltip display is
    not known yet, let's find it
    if voidp(pchToolTipDisplay) then    pchToolTipDisplay =
        sendAllSprites(#mToolTipDisplay_GetChannel)
    end if
    sendSprite(pchToolTipDisplay, #mToolTipDisplay_Show,
pName, pRect)
end

on endSprite me
    sendSprite(pchToolTipDisplay, #mToolTipDisplay_Hide)
end

```

```

property spriteNum

on beginSprite me
    me.mToolTipDisplay_Hide() -- start off hidden
end

on mToolTipDisplay_GetChannel me
    return spriteNum
end

on mToolTipDisplay_Hide me
    sprite(spriteNum).locV = -1000 -- move offstage
end

on mToolTipDisplay_Show me, theString, rectRolledItem
    put theString && rectRolledItem
end

```

```

property spriteNum
property pmToolTipText
property pMaxWidthToolTip
property pnPixelsAboveOrBelow

on beginSprite me
    pmToolTipText = sprite(spriteNum).member
    pnPixelsBelow = 4 -- spacing between rect and tooltip
    pMaxWidthToolTip = 200 -- maximum width of the tooltip
    me.mToolTipDisplay_Hide() -- start off hidden
end

```

```

property spriteNum
property pmToolTipText
property pMaxWidthToolTip
property pnPixelsBelow

on beginSprite me
    pmToolTipText = sprite(spriteNum).member
    pnPixelsBelow = 4 -- spacing between rect and tooltip
    pMaxWidthToolTip = 200 -- maximum width of the tooltip
    me.mToolTipDisplay_Hide() -- start off hidden
end

on mToolTipDisplay_Hide me
    sprite(spriteNum).locV = -1000 -- move offstage
end

on mToolTipDisplay_GetChannel me
    return spriteNum
end

on endSprite me
    pmToolTipText.text = " "
    me.mToolTipDisplay_Hide()
    updateStage
end

on mToolTipDisplay_Show me, theString, rectRoll -- Set
the max width of the tooltip
    pmToolTipText.rect = rect(0, 0, pMaxWidthToolTip, 0)
    pmToolTipText.text = theString

    if pmToolTipText.lineCount = 1 then
        -- Only one line, must shrink the width to match the
text
        nChars = pmToolTipText.text.char.count
        locLastChar = pmToolTipText.CharPosToLoc(nChars + 1)
        textMemberRect = pmToolTipText.rect
        textMemberRect.right = locLastChar.locH
        pmToolTipText.rect = textMemberRect
    end if

    -- Calculate locH and locV    centerH = (rectRoll.left +
rectRoll.right) / 2
    locHTip = centerH - (pmToolTipText.width / 2)
    locVTip = rectRoll.bottom + pnPixelsBelow

    -- Finally, assign the tooltip location
    sprite(spriteNum).loc = point(locHTip, locVTip) end

```



Church of Fools

Specialmoves has launched Church of Fools, a groundbreaking 3D multi-user virtual church built on top of their flexible interaction engine, using Shockwave and Flash Communication Server. Created for shipoffools.com, an irreverent Christian Web site, Church of Fools allows people from around the globe to talk, pray, and attend services in a stunning Romanesque virtual church.

Users choose a screen name and customize their avatar, then they're free to explore the environment. They can sit on pews, kneel at the altar, and even wander down to the crypt for a discussion with the preacher after a service. Controls allow worshipers to view the real 3D environment from any angle and to perform gestures such as "praise the lord" hallelujah.

It's a massive worldwide hit with tens of thousands of visitors a day; at its peak Church of Fools welcomed 41,000 people within 24 hours. Church has never been so much fun.

www.specialmoves.com





FLASHFORWARD & FLASH™ FILM FESTIVAL

NEW YORK CITY 2004, JULY 7-9
THE NEW YORKER HOTEL

THE WORLD'S PREMIER FLASH EVENT

The 12th Flashforward conference and Flash™ Film Festival, the oldest and largest Flash user conference in the world, shares the latest in design, development, education and inspiration.

FLASHFORWARD 2004 FEATURES:

- 4 In-Depth Flash Workshops
- 22 One-Hour Seminars
- 20 Q&A Sessions
- 17 "Ask the Experts" Sessions
- 11 Technology Showcase Seminars
- Flash™ Film Festival
- Macromedia Keynote
- Exhibition Area
- Flash Authors' Book Signing Event
- Networking Receptions
- Exclusive Conference Workbook with Speaker Notes

REGISTRATION:

\$499 through June 11
\$599 through July 6
Additional \$50 Macromedia User Group discount.
\$699 at the door.

ADDITIONAL INFORMATION:

- www.flashforwardconference.com
- Call toll free 1.877.4.FLASH.4
- 1.805.640.6679 outside the US and Canada

GOLD SPONSOR


macromedia®

BRONZE SPONSORS



electricrain



introNetworks



macromedia
PRESS

METALYZO



New
Riders



Peachpit Press

MEDIA SPONSOR



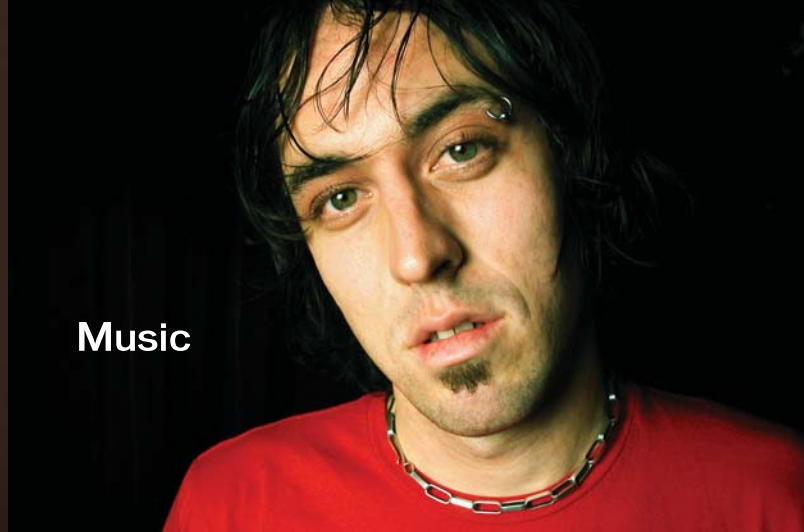
ANIMATION
MAGAZINE



MX
developer's journal



Consumer
Products



Music



Fighting AIDS



Water

INTO

WHAT ARE YOU INTO?

At Macromedia, we're continually inspired by the passion of our customers. Visit "Into" to see their stories, or share your own. www.macromedia.com/into



Announcing Macromedia MX 2004:
New versions of Macromedia® Flash™, Dreamweaver®,
Fireworks® and Studio. [Run with it.](#)

Copyright © 2003 Macromedia, Inc. and its licensors. All rights reserved. Macromedia, the Macromedia logo, Dreamweaver, Flash, Fireworks, and Macromedia Flash are trademarks or registered trademarks of Macromedia, Inc. in the United States and/or other countries. Other marks are the properties of their respective owners.