MX
developer's journal

FOR MACROMEDIA MX DEVELOPERS & DESIGNERS

january 2004   www.mxdj.com

DREAMWEAVER
Search... and You Shall Find

FLASH
Streaming Media with MX

FIREWORKS
Optimizing JPEG Slices

FREEHAND
The Art of Pages

COLDFUSION
CF Explained

building a
cyborg suit
with director

Tangled
in the
Machine

on the cover

A custom hardware system uses Director to generate a prototype 3D environment that is dynamically generated and controlled by Lingo and can be modified in real time.

MX developer's journal

From the MAX 2003 Show Floor

Macromedia takes its vision to MAXimum heights

sponsors

Eidoserve offers a complete, highly interactive, rich multimedia platform where "real answers are only a simple question away."

Electric Rain, Inc. is a fast-moving 3D and multimedia software company with a vision of bringing easy-to-use 3D and multimedia tools to the masses.

Oddcast licenses a dynamic rich media platform that enables companies to improve business results by bringing Web and wireless content to life quickly, easily, and cost-effectively.

Peachpit publishes how-to books on graphics, multimedia, and Web publishing. Peachpit represents an elite collection of the world's top computer book imprints, including Macromedia Press as well as flagship imprint Peachpit Press and New Riders.

Sorenson Media is the recognized technology leader in video services, video compression, and video communication to improve the quality of communication over the Internet.

Sybase enables the "Unwired Enterprise" by delivering mobile and enterprise software that integrates platforms, databases, and applications.

Universal Mind is a professional services organization focusing on Macromedia solutions.

Webvertising provides Web application and rich Internet application development services for clients in a variety of industries.

t he annual Macromedia Conference may have undergone a name change as well as a location change, but one key thing remained the same: the overall level of excitement of Macromedia's development community.

Macromedia made many interesting announcements at MAX: the official announcement of the availability of Central and the Central SDK; the announcement of "Flex" (formerly code-named "Royale") and the first public demonstrations of the new technology; and a sneak peek at the next version of ColdFusion, currently code-named "Blackstone," to name a few.

By many reports, attendance at MAX 2003 in Salt Lake City, Utah, was up from last year's show, and if traffic to the SYS-CON Media/*MX Developer's Journal* booth was an indication, that was certainly true – a fantastic sign.

Dozens of sponsors showed off their products and services, and CF editor Robert Diamond was lucky enough to meet with representatives from some of the most innovative companies out there. On the following pages is our MAX 2003 scrapbook, complete with pictures of some of our friends that were at the show.

CF Editor Robert Diamond with Macromedia's Ben Forta (above) and Al Ramadan (left)

activePDF is a leading provider of PDF creation, conversion, and development tools for Windows, offering the ability to dynamically create PDFs from over 280 file types, including HTML, as well as numerous PDF manipulation features.

Affinity Internet has been providing feature-rich services and solutions since 1996 and is the leading Web hosting services provider. Affinity operates multiple hosting brands ValueWeb, HostSave, and WinSave.

B-Line Express specializes in developing rich Internet applications that streamline the user's accessibility and interactivity with complex information. Whether it's multimedia content or warehouses of legacy data, B-Line delivers engaging front ends seamlessly integrated with robust back-end processing.

CFXHosting, founded by ColdFusion developers, is one of the leading Macromedia hosting companies. Recipient of the *ColdFusion Developer's Journal* Readers' Choice Award for excellence in hosting in 2000 and 2002, and a Macromedia Premier Alliance Partner, CFXHosting offers a wide array of plans ranging from shared to dedicated hosting.

eHelp is the maker of the award-winning RoboDemo tutorial software, and FlashHelp, the revolutionary online Help format. RoboDemo records any on-screen activity to create software tutorials, eLearning simulations, and product demos in Flash format.

F5 Networks keeps IP-based traffic flowing and business information always available to any user from any device, anywhere in the world. Our products ensure secure and reliable access to servers and the applications that run on them. F5 also provides tools to automate communications between applications and the network.

FuseTalk Professional Edition makes it easy to get started quickly and easily. FuseTalk Enterprise Edition delivers powerful tiered administration and database tools for managing large, complex, or dynamic discussion forum deployments. FuseTalk Virtual Meeting leverages Macromedia Flash Communication Server to turn forums into effective, inexpensive virtual workspaces.

Integration New Media is a North American leader in multimedia development. The company offers top-level multimedia project consulting services as well as a successful line of multimedia products and Xtras.

MXDJ Section Editors

Dreamweaver
Dave McFarland

Author of Dreamweaver MX: 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.jesse warden.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.

ColdFusion
Robert Diamond

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

Director
Niqui Merret

After many years as an instructor and developer in South Africa, Niqui now works at London-based rich Internet solutions firm, Prismix. She feels that she is a Director addict in a Flash-mad world.

InterAKT Online
is an experienced company that develops powerful Dreamweaver MX extensions for dynamic Web site development. Our professional Web tools transform Dreamweaver MX into a very powerful visual code generation platform for complex dynamic Web sites.

NACIO Systems, a Macromedia Alliance Partner, is a developer-friendly company offering shared and dedicated managed hosting for ColdFusion MX, JRun, and Flash Communication Server. NACIO Systems specializes in high-reliability dedicated hosting, server load balancing, clustering, storage, MX migration services, database administration, application/content management and other ASP platform services.

O'Reilly & Associates is the premier information source for leading-edge computer technologies. The company's books, conferences, and Web sites bring to light the knowledge of technology innovators.

PaperThin's award-winning CommonSpot Content Server delivers empowering Web content management capabilities that enable organizations to create, manage, and publish Web content affordably and efficiently – within a controlled, out-of-the-box, scalable framework. Built on ColdFusion, CommonSpot's open architecture enables easy customization and integration of external applications.

RemoteSite Training provides hands-on instruction for all levels of Web application design and development. Since 1999, RemoteSite Training has been a Macromedia Partner and has delivered enterprise-wide solutions globally. Based in Raleigh, NC, RemoteSite Training offers training courses throughout the country.

Site Executive is a leading ColdFusion-based Web Content Management System for middle-market organizations. Developers can take advantage of its extensibility and platform interoperability, end users are empowered via its intuitive interface, and business people love its price tag.

TechSmith is the maker of Camtasia Studio, which captures onscreen activity to create videos to train, inform, and demonstrate. With it you can produce videos as Macromedia Flash files to distribute software demos, training, technical support, documentation, and marketing presentations.

Advanced Templating

Moving around regionally
by murray summers

dreamweaver hides many wonderful (and often overlooked) tools. While you may be a long-time user of Dreamweaver templates you might not know some of their more powerful features. This article explains how two of these "template secrets" interact, and how you might use them on your pages. I'll assume you have some experience with the basics of Dreamweaver templates, which were greatly improved in Dreamweaver MX (DMX), but remain nearly the same in MX 2004.

Among the many features Macromedia added to Dreamweaver MX's template capability is a collection of new regions, one of which is called an optional region (see Image I). The name of this region is a blatant giveaway of its function – it is a region that may, or may not, be written into, hence visible on pages built from templates. We'll call such template-based pages *child pages* in this article. To understand how optional regions work, it's important to review the template process.

An engine that drives all of DMX's template functionality is called into play at several times during your use of templates. When you spawn a child page (using FILE | New > Templates > Site > (template file name) > Create), this engine is called to conditionally (I'll talk more about the condition part later). Copy the template's code into the child page. When you make edits to the child page, the engine is invoked to check those edits and to make sure that you have not tried to change any non-editable region's content (there are those who believe that the engine contains a random trigger for the error message associated with this check!). And finally, when you make changes to a template page, the engine serves to propagate those changes forward into all existing child pages based on that template (and to once again check to make sure that you have not tried to nest editable regions, or something equally troublesome). The important idea here is that this template engine is deeply and continuously involved in the state of your template-managed site's pages.

The name "optional region" implies that there is some condition on which this region somehow depends, and that is true. These conditions are communicated to the template engine by way of one or more variables that are accessible to each child page, and that can be changed as desired by you, the developer. These variables are called "template parameters" (catchy name, don't you think?). A template parameter is automatically created for you whenever an optional region is inserted into a page. You can then use this parameter's value to modify the display of that optional region's contents on any given child page of this template. To do this, you would open the child page, select MODIFY | template properties…, find and select the parameter's name in the ensuing list of available parameters, and enter a new value for that parameter. On making this change, the template engine examines the parent template page first and then the parameter's value as just changed on the child page, and adjusts the child page's code (and its optional region, if any) accordingly.

Since parameters can be central to the way in which the template's code is written to the child page, it is important to understand a bit more about them. A parameter can hold a value of any sort. This is a bit dangerous when accepting free-form input from a developer. As a result, Macromedia has confined things a bit by allowing only five kinds of parameters: text, URL, number, color, and Boolean. The type of parameter defines which type of user interface you are offered on making a change to that parameter's value, and this is mighty convenient. A text parameter allows the entry of free-form text into an input field, whereas a URL parameter gives you the option to browse to and select a new target file.

In addition to parameter type, there is also a way to assign a parameter default

image I

MX
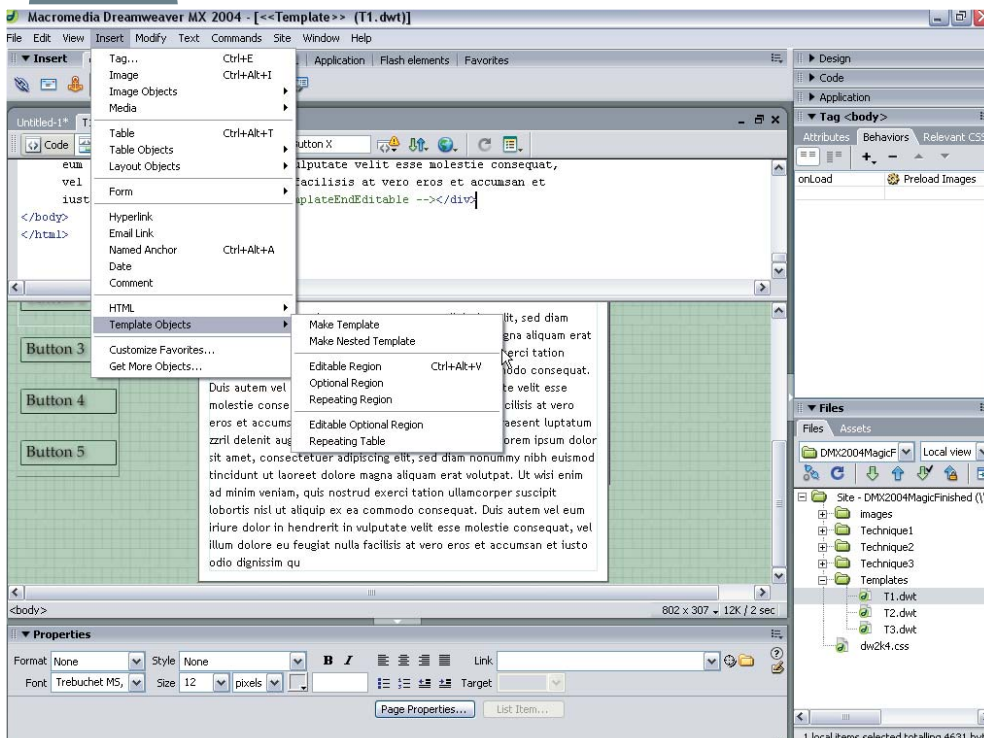
value. This value is used by the template engine to determine how to interpret any use of optional regions or template expressions (a discussion of template expressions will be reserved for a subsequent article) on the page so that the correct code can be written to the child pages. The default value for a parameter is usually determined when that parameter or optional region is defined. Of course, the default value must be consistent with the type assigned to the template parameter or bad things will likely happen.

Finally, each parameter must have a name associated with it so that uses of that parameter's contents can be easily located on child pages.

To see how this works, create a new page in DMX and save it as an HTML template. Use INSERT | Template Objects > Optional Region, leave the default values (name="OptionalRegion1" and Show by default selected), and click OK. Look at the code. In the header of the document you will find the parameter's declaration (see Image II).

```
<!-- TemplateParam
name="OptionalRegion1" type="boolean"
value="true" -->
```

In the body of the document you will find the optional region.

```
<!-- TemplateBeginIf
cond="OptionalRegion1" --
>OptionalRegion1<!-- TemplateEndIf -->
```

Save this file again (ignore the no editable region prompt) and spawn a child page from it. Note that the child page uses the value of the Parameter (true) to satisfy the test

```
cond="OptionalRegion1"
```

hence the contents are displayed on the child page. By using MODIFY | Template Properties, you can set the value of this parameter to "false" when you select the OptionalRegion1 parameter and then click the Show OptionalRegion1 checkbox (note that the value is shown under the column labeled "Value"; see Image III). On clicking the OK button, this optional region's content will disappear from the child page. Note carefully that
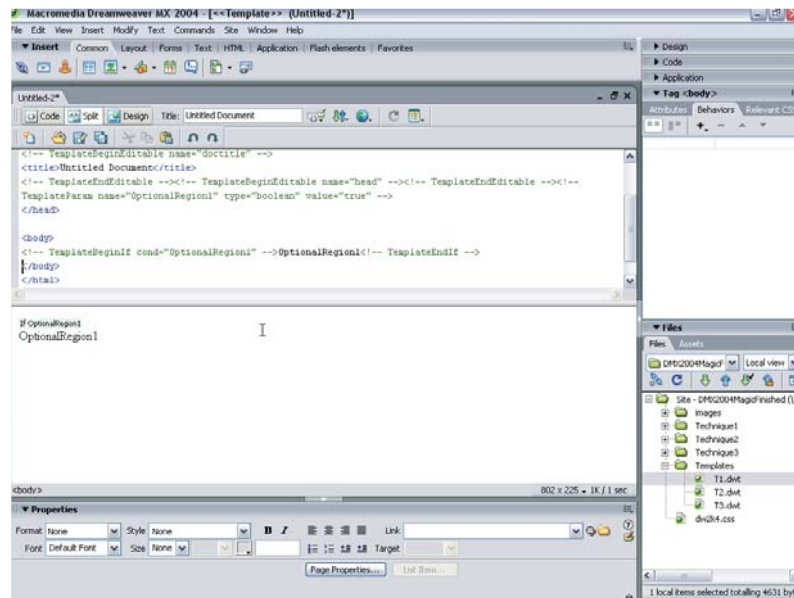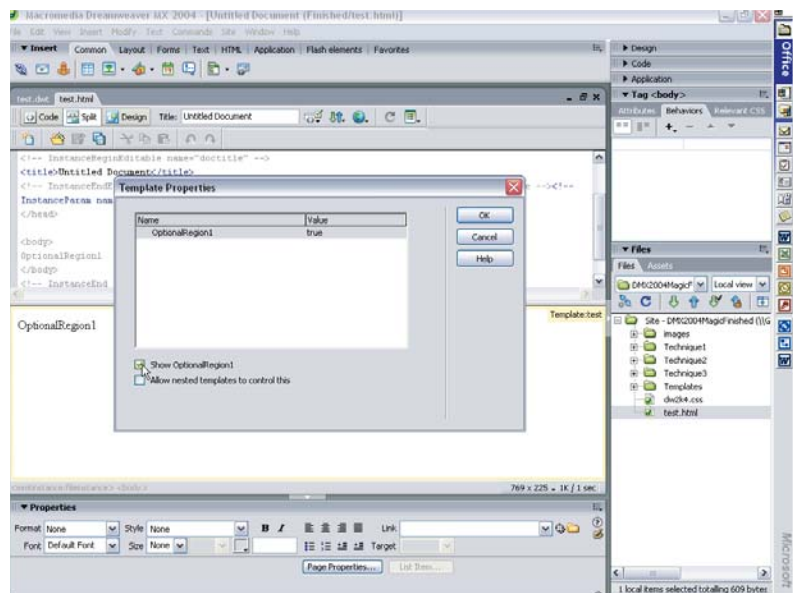
this content disappears because it's not written to the page. When an optional region is not "optioned," that code is never written to the child page (and if it was written before changing the parameter, it is erased from the page by the template engine).

So, parameters can be used to control regions, and regions can be used to control page content display. But consider this: an optional region is not necessarily usable only for things that can be displayed. Remember that when the condition fails, the code is not written to the child. So we should broaden this concept to the following: an optional region can control what code is written to the child page.

Note: The code for the parameter's

definition is exposed on the child page in code view, as if it were in an editable region. But it is important that you not make changes to the parameter by editing it in code view; otherwise, the page will not be properly updated by the template engine. Changes to parameter values should always be done using the MODIFY | Template Properties approach.

Why bother with this detail? It can be mighty useful, that's why! Here are some scenarios in which you might want to use this method:

Scenario I

Your pages contain images that are section specific. In a single template file, you can insert each image and place an optional region around it with a default

Murray Summers is a biochemist by training, but has spent the last 20 years working in the computer industry. In 1998, Murray started Great Web Sights (www.great-web-sights.com). As a Team Macromedia Volunteer, he also participates in the sponsored newsgroups for Dreamweaver and other Macromedia products. Murray is a Macromedia Certified Web Site Developer and Dreamweaver Developer.murray@ great-web-sights.com

value of false. Name these regions something like Section1, Section2, etc. This will give you code that looks something like what is shown in Code I.

Since all of these parameters evaluate false, none of this code will be written to the child pages, i.e., the disadvantage of bulky template markup is minimized. On any given child page where these images are required, use MODIFY | Template Properties to set the desired image's controlling parameter's value to true, and they will then appear on that page when the code is added to reflect this change.

Scenario II

Obviously, the method in Scenario I can be easily adapted to navigation buttons in a "two-state" menu (i.e., each menu button has an up and a down state). Each up and each down button would be placed on the template page, and each button's image would be controlled by an optional region. While somewhat awkward for a complex menu, this method would easily allow you to set the "down state" for any button on any child page.

A simple extension of this concept would also allow this method to work for CSS menus rather than image-based menus, using something like the following:

```
<!-- TemplateBeginIf cond="Link1up" --
><a href="link1.html"
class="linkup">Link1</a><!--
TemplateEndIf -->
<!-- TemplateBeginIf cond="Link1down"
--><a href="link1.html"
class="linkdown">Link1</a><!--
TemplateEndIf -->
```

(Be aware that there are better ways to do this using template expressions or multipleIf conditions, but that will be explored in future articles.)

Scenario III

Nothing prevents you from adding an optional region to the head of your template document. This can be handy in some situations.

Say you are working in a collaborative environment, and you want your developers to be able to easily select from one of several stylesheets for the child pages. It is quite simple to put the alternate stylesheet links into optional regions, as shown below.

```
<!-- TemplateBeginIf
cond="Stylesheet1" --><link
href="../styles1.css" rel="stylesheet"
type="text/css"> <!-- TemplateEndIf --
>
<!-- TemplateBeginIf
```

```
cond="Stylesheet2" --><link
href="../styles2.css" rel="stylesheet"
type="text/css"> <!-- TemplateEndIf --
>
```

Using MODIFY | Template Properties provides easy access to these parameters, allowing instant changes to the referenced stylesheet without unnecessarily giving access to any of the remainder of the head region.

Scenario IV

It is quite possible, and sometimes useful, to use optional regions for selecting between server-side includes on a page, e.g.,

```
<!-- TemplateBeginIf cond="Include1" -
--><!-- #include file="myinclude1.asp"
--><!-- TemplateEndIf -->
<!-- TemplateBeginIf cond="Include2" -
--><!-- #include file="myinclude2.asp"
--><!-- TemplateEndIf -->
```

This might permit the easy selection of two different navigation schemes within a single template–controlled site.

Note: Although this discussion focuses on optional regions, for which parameters are automatically created by DMX, it is important to mention that this is the only instance in which DMX does the automatic writing of the parameter's code. If you want to use parameters without inserting an optional region, you will have to add the parameter code manually. In addition, the optional region parameters are always Boolean type. If you want another type, you will have to make that change manually, too. Snippets in DMX can work really nicely for this. You can find a template parameter snippet extension on www.dreamweavermx-templates.com for free download.

End Note

Now that the groundwork has been laid for understanding how template parameters work, and how they can be used to control the display of optional regions, it will be much simpler to move on to more advanced applications of these new template features, like multiple conditional tests and template expressions. I hope to be able to share those with you in coming issues.

These ideas and much more about Dreamweaver MX templates are explored in some depth on our book's companion site at www.dreamweavermx-templates.com, developed and written in collaboration with Brad Halstead. I hope that you find this site and its contents useful in your exploration of template capabilities!

Menu Magic II DHTML Menu Kit

Project Seven's Menu Magic II makes creating dynamic menus (and sub-menus) a simple process. Choose from a variety of menu types such as the drop-down menu pictured above.

Extension Developer: Project Seven Development
Developer Web Site: www.projectseven.com
Price: $90

You've probably seen navigation bars that include pop-up submenus like the one pictured here. Many sites now include them, as they let you put numerous links out of sight until a user needs them. But the complex mixture of JavaScript and CSS isn't always easy to implement, especially when you factor in the often-inconsistent browser support for these technologies. Dreamweaver includes a simple behavior – show pop-up menu – to create these menus, but it doesn't offer many options; and when it's used with Dreamweaver's Templates and Site Management tools, it frequently breaks.

If you want to add dynamic menus like these to your site, but aren't interested in investing the hours required to master the JavaScript and CSS necessary to build them, Project Seven's Menu Magic II is the answer. This easy-to-use tool lets you create a variety of menu types, including horizontal and vertical navigation bars, that use a combination of graphics (for the main menu buttons), and CSS-styled text (for the submenu buttons.) The extension comes with built-in themes (designs) so you can quickly deploy well-designed navigation systems. But if the default graphics aren't to your liking, you can edit the included PNG files in Fireworks to make navigation buttons that exactly match your vision. Additional themes are available from Project Seven, and a free Theme Builder

extension lets you create your own designs.

The support material for this extension is awesome. A 153-page user's guide provides straightforward explanations of how to use the extension, including excellent step-by-step tutorials. And the user-centered interface demonstrates what a mature, well-designed, and thoughtful commercial extension should be.

Dave McFarland is the Dreamweaver editor of MX Developer's Journal *and author of* Dreamweaver MX 2004: The Missing Manual. *davemcfarland@sys-con.com*

Search

...and You Shall Find

Powerful and easy "hosted search" services like Google help your customers unearth everything from general Web results anywhere on the Internet to pages located on your site.

Search engines share the common goal of bringing content from all over the world together into one convenient place. Hosted search technology can be added to your Web site in minutes. There's nothing to download or install. The best part is that it's free and easy. To get a handle on hosted search services, it is important to first understand a couple of search techniques familiar to anyone who spends time online – "Web search" and "site search."

by eric ott

Inside the Engine

Web Search

Search sites like Google and Yahoo! have popularized Web keyword search, where users enter words or phrases and click a button to return result pages containing these terms. These sites leverage indexes of publicly available content on the Web and use various algorithms such as Google's well-known page-ranking algorithm to display the most relevant information first.

Site Search

In addition to this level of Web search, most people have also used the simple keyword search found on individual sites. Site search uses the same concepts as the big search engines but on a much smaller scale. However, the goals of a site search are different from those of a Web search. Site search needs to fit seamlessly into any site design. You need complete control over the look and feel of search results.

Make a Good Match

There's an overpopulation of search services right now. However, as a Dreamweaver user it is important to find a vendor that has integrated their services into Dreamweaver so that these services appear as a native feature in the tool. Only a few companies have created extensions for Dreamweaver. The most popular search extension for Dreamweaver has been the Atomz Search extension, which incorporates the Atomz Express Search service into Dreamweaver. However, this is a trial version – it is feature limited and has a 500-page limit. WebAssist recently released the WA Google Search extension that incorporates Google technology into Dreamweaver.

In this article, we'll take you step-by-step through three easy steps to add the Google Free - Search service to your site. First, you define the type of search you would like to add to your site. Next, you define the design properties of the Google results page. Finally, test your pages with the Google service and make any necessary edits. The entire process is visual and does not require any hand coding or knowledge of Google technology.

image I

image II

image III
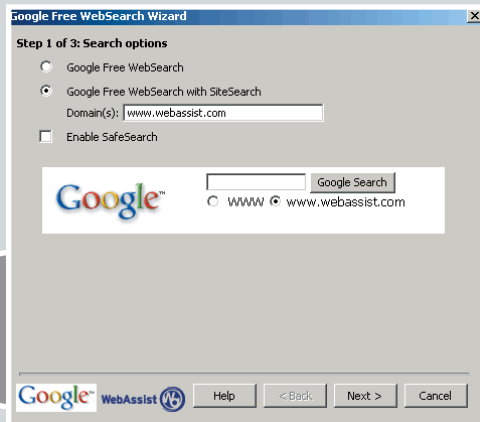
Google Free WebSearch Wizard

The WA Google Search extension adds a Google tab to the Insert Panel – you will need to select this tab in order to view the Google Search object. Google has many other hosted services and you can imagine that in the future other Google technologies will be available from this Object Panel tab. Click on the Google Search object to open the insertion wizard. This wizard walks you through the process of configuring the Google Search in three easy steps. Once you have successfully completed the insertion wizard, the extension will insert the code into your page and display the Google object in the Design View.

Step 1

The first step in the wizard allows you to define the type of search – either Internet search or Site search. Google offers three different search options with the Google FreeSearch service – WebSearch, SiteSearch, and SafeSearch. The extension will automatically generate the proper code depending on your selection on this page in the wizard.

Follow these steps to successfully complete the first step in the wizard:

1. From the Google Object Panel, choose the WA Google Search object to start the wizard.
2. Select the Google Search option if you want to insert a Google search object that will search the publicly available content on the Web.
3. Select the Google Search with SiteSearch option if you want to insert a Google search object that allows users to search either the Internet or just pages with your domain(s). When you select this option the Domain field becomes enabled – you'll need to update the placeholder domain with your domain(s). Multiple domains should be separated by semi-colons.
4. Select the Enable SafeSearch checkbox if you want to restrict the results to exclude Web pages containing adult-themed and explicit sexual material.
5. The preview will automatically update as you change your selections on this page of the wizard. The preview provides instant feedback to you on what the Design View will look like in Dreamweaver after the object is inserted.
6. When you're ready, click Next to proceed (see Image I).

Step 2

The second step in the wizard allows you to define a custom logo for the results page on the Google server. This step is optional – you can leave the URL text field blank and click Next to proceed if you don't want to show your logo on the Google server. If you would like to add a custom logo you must enter a valid HTTP path to the logo image and enter the design properties for the logo. Your logo will appear at the top of the page, above the Google Search Form. You can also customize the background of the Google Search Form to match your branding.

Follow these steps to successfully complete the second step in the wizard:

1. In the Logo URL field enter a valid HTTP address to your custom logo. It is common for users to post a custom logo on their Web server and then enter the URL that references the logo image. This image does not need to be used on your site – it can be a custom logo just for Google. Leave this option blank and click Next to proceed if you want to skip this step.
2. If you're going to use a custom logo, it is recommended that you enter the design properties to ensure the logo is properly formatted on the Google server. The Width, Height, and Alignment options define the positioning of your logo.
3. The Google Background Color allows you to adjust the background color of the table cell that contains the Google Search Form – use this option to match the Google Search Form with the rest of your page. You can select white, grey, or black for the background color. The background color is white by default.
4. When you're ready, click Next to proceed (see Image II).
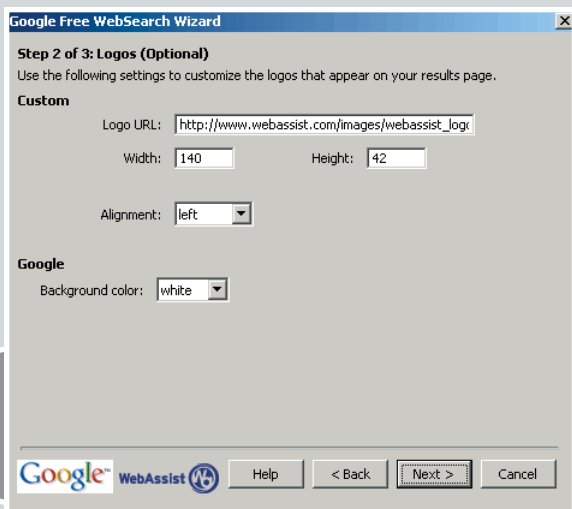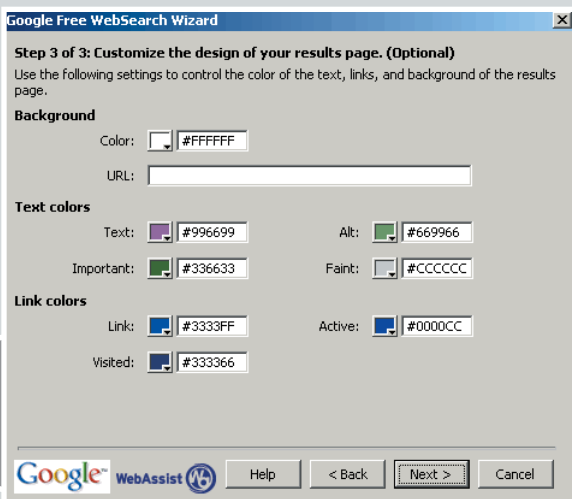
Step 3 : Customize the Design of Your Results Page

The third step in the wizard allows you to define the design properties of the results page on the Google server.

This step is optional – you can leave all the fields blank and click Next to proceed if you want to use the standard design for the Google results page. If you want to customize the results page, you can enter a HEX number or use the color picker to define the custom color scheme. You can leave individual fields blank if you just want to override a few of the design properties.

Follow these steps to successfully complete the third step in the wizard:
1. In the Background section, enter the color and URL to define the background color and pattern for the results page.
2. In the Text Colors section, enter the

Back to find and modify the choice. When you're done, click Finish. The wizard inserts the custom code required to work with the Google server and provides a visual representation in the Dreamweaver Design View. Your site is now ready to work with the Google Search service and you didn't have to write any code or study the Google API to implement this feature on your site.

Google Free WebSearch in Design View

Once you have completed the wizard, the extension will insert the proper code on your page. The extension includes a custom translator, a pattern-matching technique, to properly render the Google

Eric Ott has over 10 years of experience as a pioneer in the Web industry. His position at Macromedia as the product manager for Dreamweaver enabled him to define the direction of this highly regarded Web development platform from its early days. His knowledge and experience in Web design and development have given him strong insight into ways to achieve success on the web. Eric is currently the president of WebAssist, the leading provider of extensions for Macromedia MX.
eott@webassist.com

"Site search uses the same concepts as the big search engines but on a much smaller scale"

HEX color values for Text, Alt, Important, and Faint. Each of the options define the design properties for text that is included in the results page. These Text Color settings are not very inituitive, so I recommended that you define the colors and then do some trial-and-error testing to get the Text Colors defined properly for your requirements.
3. In the Link Colors section, enter the HEX color values for Link, Active, and Visited. These design properties control the link colors on the entire page. You probably have these colors defined as part of the color scheme for your site.
4. When you're ready click Next to proceed (see Image III).

Summary Page: Review Google Search Selections

In the final screen of the wizard, review your selected choices. If you see anything you'd like to change, choose

object in the Dreamweaver Design View. Additionally, pages that already have the Google Search code footprint are supported by this extension. This enables you to pass your pages on to other developers and, as long as they have the Google extension installed, the page will render correctly. The Google object is represented as an HTML Form with an embedded HTML table for positioning (see Image IV).

Google Free WebSearch Footprint

What's happening behind the scenes in the Dreamweaver Code View? The Google Search code footprint is a hybrid of HTML Form elements and HTML Table cells. You can add code to the footprint to customize further. The only code that should not be changed is the FORM ACTION value. This is the unique identifier for the code footprint and is used to tell Dreamweaver that this is a Google Search object (see Image V).
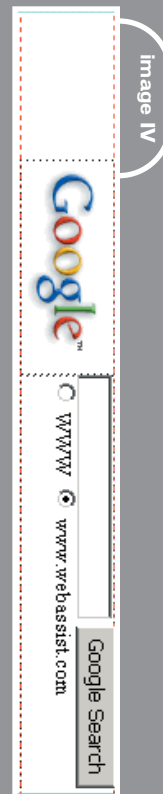
image IV

> *"...it is important to find a vendor that has integrated their services into Dreamweaver so that these services appear as a native feature"*

Edit Google Free WebSearch

Once you have inserted the Google Search service on your page it is possible to visually edit the properties. Dreamweaver has a Property Inspector that is used to edit objects in Dreamweaver. The Google object has a custom PI , but there are too many options to include them all in the small area of the Property Inspector. You must click on the Edit button in the Google Property Inspector to bring up the Edit interface for the Google Search service. There are three tabs in the Edit Interface that correlate to the three steps in the wizard.

Follow these steps to successfully edit the Google Search object:

1. In the Dreamweaver Design View select any region of the Google Search Form.
2. Select the Edit button in the Google Property Inspector. Update any of the fields in the Edit Interface and select OK to update the code footprint on your page (see Image VI).

WA Google FreeSearch for Contribute

Macromedia Contribute is the easiest way for individuals and teams to update, create, and publish Web content to any HTML Web site. Contribute allows non-technical users to update Web content while maintaining site standards for style, layout, and code.

The WA Google FreeSearch extension is also available for Contribute. The user experience is exactly the same as with the Dreamweaver extension – except there is a custom Google toolbar for insertion. It is also important to note that Macromedia Extension Manager does not support Contribute so the installer uses platform-specific installers instead of the standard MXP file format.

Download Information

You can download the Dreamweaver and Contribute extensions directly from the WebAssist Online Store. These extensions are free downloads and work with Dreamweaver MX, Dreamweaver MX 2004, and Contribute 2. The extensions work on Windows and Macintosh platforms. Visit http://webassist.com/journal/google to learn more.
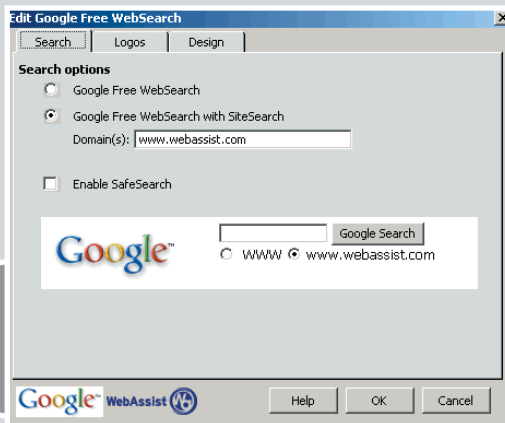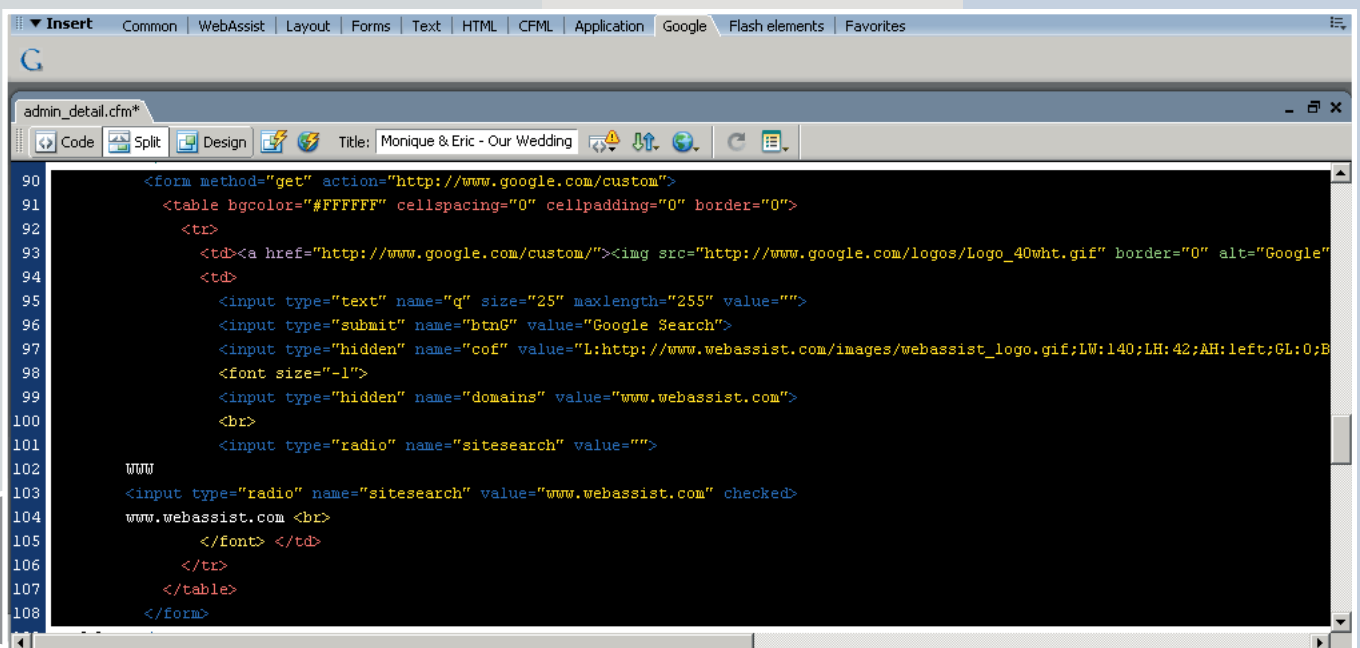
image V

image VI

What's the easiest way to update a web site?

Through your browser

Introducing:
- Property inspector
- Spell-checker
- Image management
- Dreamweaver MX support
- ColdFusion, ASP, PHP

Experience:
- "MS Word" style text editing
- Text formatting using CSS
- Intuitive table alignment

Compatibile with:
- MSIE/Netscape
- Windows/MAC/Linux

KTML 3 WYSIWYG editor
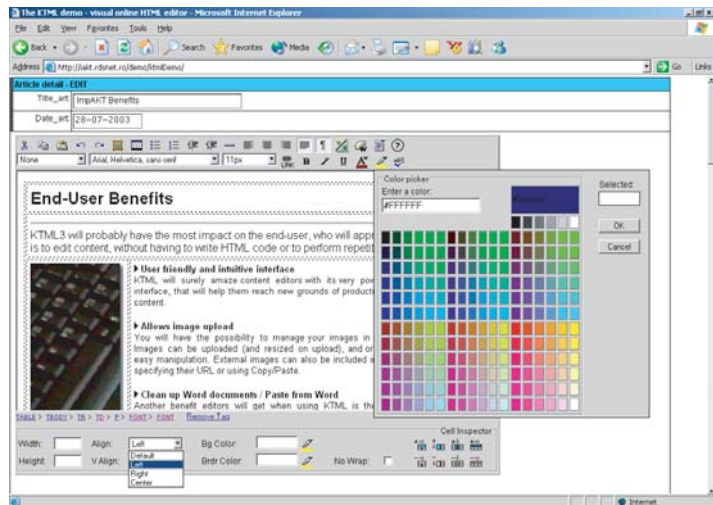
See all the features, benefits and a live demo at:

http://ktml.interaktonline.com/

KTML3 - Coming Soon to a Browser Near You

Flash MX Programmer's Guide to ActionScript 2.0

The more tools you have, the more you can build

by erik bianchi

flash developers who are still using ActionScript 1.0 are drastically limiting their capabilities, wasting valuable time and resources. Because most Flash developers come from a nonprogramming background, they don't fully understand the true power and benefits of ActionScript 2.0. This article attempts to clear up some common misconceptions about AS2 and gives some clear examples demonstrating the new concepts and methodologies that can be applied using it.

AS1 Is Not Deprecated

What is now referred to as AS1 is the same AS you'll find in Flash MX. Contrary to popular belief, the AS1 programming style – using prototype objects as classes – is not deprecated. You can still write prototype objects and modify existing objects' prototypes in AS2 using the exact same AS1 syntax. AS2 does not deprecate AS1; it extends and refines it.

Class Declarations

With AS1 you could only use prototype objects to define a class (see Code I).

For component development you would also have to use the #initclip and #endinitclip pragmas as well as the Object.registerClass method to assign your class to a MovieClip in the library.

With AS2 you could still write the same prototype object or you could write a true class. To write a class you need to define your class in a separate AS file much as you would in Flash MX for external AS. For classes, the AS file must have the same name as the class itself. Some people might find this to be a hindrance; however, this requirement actually helps you better organize and structure your code (see Code II).

The prototype object and the class example do the exact same thing. For comparison, the constructor function was left in the example in Code II. However, a class does not explicitly have to have a constructor function defined as it does for prototype objects. In addition, no other code is required for component development. You can simply specify which class a MovieClip should use in the MovieClip's Linkage Properties panel found in the library.

When compared side-by-side you can see that the class definition is actually cleaner looking and is less redundant. In the prototype example, for every method of a class you must type the name of the object you're adding the method to and the location of the method (specified by prototype).

Some might argue that despite the redundancy the prototype approach is actually more flexible because you can add a method to any class dynamically at anytime in AS1. (This is known as delegation.) Again, AS2 doesn't deprecate AS1, so you can still reference an object's prototype even inside a class definition (see Code III).

Properties

In AS1 you could add a property to a prototype object by first declaring a method to be used as a getter and then declaring another method to be used as a setter. Then you had to use an addProperty method on the prototype object to wire the getter and setter methods together as one property (see Code IV).

You can still use the exact same syntax in AS2 or you can just declare a getter and setter function (see Code V).

In the example in Code V, Flash MX 2004 Pro does all the work of the addProperty method for you and automatically ties the getter and setter methods together in a property named

code I

```
// An Example of a Prototype
Object

// Constructor Function
function MyObj(){}

// Inheriting From MovieClip
MyObj.prototype = new
MovieClip();

// Specify a Method
MyObj.prototype.method = func-
tion(){
    trace("Hello World");
};
```

code II

```
// An Example of a Class


// MyObj.as

// Defines Class and Inherits
From MovieClip
class MyObj extends MovieClip{

        // Constructor
Function
        function MyObj(){}

        // Specifies a Method
        function method(){
                trace("Hello
World");
        }

}
```

code III

```
// An Example of a Class
Definition

// Inheriting From MovieClip
class MyObj extends MovieClip{

        // An example delega-
tion
        var inheritedMethod =
AnotherClass.prototype.method;

// Constructor Function
        function MyObj(){}

        // Specifies a Method
        function method(){
                trace("Hello
World");
        }

}
```

"_prop". Note that this is no longer needed in most cases, saving you some extra typing.

Static, Public, Private

In AS1 there was no supported way of specifying a public, private, or static method. However, with AS2 this is possible (see Code VI).

A public method simply means that an outside object is allowed to directly access the specified property or method. A private method or property may only be accessed internally from within the class. Static methods or properties are similar to private methods in that they may only be called from within the class itself. However, static methods or properties are created only once per class rather than being created in every object based on that class.

Strict Data Typing

Another new feature of AS2 is Strict Data Typing (aka Type Casting). This allows you to cast (assign) a specific data type or class to a variable. It is especially useful when debugging. Say, for example, in AS1 you have a prototype object called List that has a method called "addNewValue":

```
var MyList = new List();
MyList.addNewVlaue(1);
```

However, when you test the movie the value never gets added to the List and you are forced to figure out the problem on your own. Granted, a typo like this by itself is pretty easy to find.

However, buried in a couple thousand lines of code it may not be as **obvouis**.

Strict Data Typing deals with this problem by allowing only public methods or properties of nondynamic classes to be called (see Code VII).

Case Sensitivity and Member Variables

Another new addition to AS2 is case sensitivity. This means that a variable called "someVar" would have a different memory location than "somevar" would. This allows each variable to exist without one overwriting the other. In addition, gone are the days of declaring an instance/member variable anywhere inside of a prototype object as in Code VIII.

In AS2 classes you have to declare all your member variables first (see Code IX). Some developers find this to be an inconvenience initially (especially when converting old AS1 code to AS2). However, over time this will help make you a more consistent programmer and will make your code easier to read and maintain. Note the use of m_ in front of someVar. This is just a naming conven-

tion I've adapted in my source code to specify a private member variable.

Conclusion

With all the features of AS1 and the many new additions to AS2, most of which I didn't even get a chance to cover in this article, Flash developers refusing to use AS2 are limiting the tools and capabilities they have for completing a job. You wouldn't want to build a house with just a hammer. A good developer knows, just like any construction worker, that the more tools you have in your tool box, the more things you can potentially build.

Erik Bianchi is a software engineer with more than five years of experience developing Flash-based RIAs and enterprise-wide applications for Fortune 50 and 500 companies. In his spare time he enjoys building Flash-based games, writing or tech editing Flash-related books, and when burned out on code, playing video games on his PC/console systems. You can get more information about Erik and his latest projects on his blog at www.erikbianchi.com.
erik@erikbianchi.com

> "Contrary to popular belief, the AS1 programming style – using prototype objects as classes – is not deprecated"

xile

written & illustrated by louis f. cuffari ②

Why is it every time Glisp starts a new painting, he can never finish it?

I hate it!

Later on...

Pure genius.

Profound, yet compelling.

It's monumental.
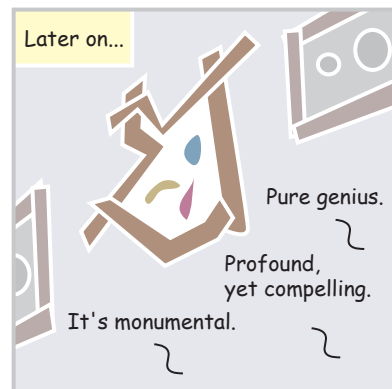
NORM MEYROWITZ
PRESIDENT OF PRODUCTS

WEB DEVELOPERS ARE USING OUR MX PRODUCTS IN WAYS WE NEVER DREAMED. JUST IMAGINE WHAT THEY'LL DO WITH THE NEW MX 2004.

I've been endlessly amazed by the things our customers have done using our MX generation of products. And with all the new features in Studio MX 2004, it's going to be even easier and faster for them to realize their visions.

Dreamweaver MX 2004 helps you get that picture in your head turned into a web site faster than ever. That should be welcome news to the millions of web professionals who use Dreamweaver to create sites and applications. We've added things like CSS support, target browser check and improved code hinting to help you get through projects in far less time. And with Fireworks MX 2004 you can optimize web graphics up to 85% faster.

The new Flash MX Professional 2004 takes our industry standard tool for building rich content and applications to a whole new level. It really helps stretch your abilities—no matter where you fall on the designer-developer continuum. For development, we've added things like data-aware components and an extensibility layer so you can add your own features. For design, we've added high-quality, long-format video that's really impressive.

These are just a few of the new features. The products are available individually, but they really work well together in Studio MX 2004. Don't take my word for it. Download a free trial and read more at our web site.

Let me know what you dream up. norm_01@macromedia.com

Introducing
Macromedia MX 2004.
www.macromedia.com/go/2004

streaming media

with

mx

by tom green

MX

Could it be that we're about to see
the disappearance of QuickTime,
Windows Media Player, and
RealOne as viable Web players
for streaming video and audio?

Now that I have your attention, this question is not as radical as it appears at first glance. The release of Flash Player 7 offers you one serious tool for the Web playback of streaming rich media without the use of third-party plug-ins that hijack valuable Web real estate and bandwidth.

Why do we need the QuickTime, Windows Media, and RealOne players when it comes to video and audio on the Web? I really don't appreciate Apple's begging me to upgrade to Pro. Windows Media Player inevitably tells me I am missing a codec, and when I ask it to show me which one, the answer is inevitably: "Don't know, but I am sure you are clever enough to figure it out." RealOne, apart from the fact that it too has trouble finding codecs, also litters my desktop with an assortment of RAM files. Not only that, I have absolutely no guarantee that users even have the plug-ins or can see the content.

The other issue is purely selfish. As I said in an article regarding this topic on the Community MX site, "If I control the Web turf, why am I ceding a piece of it to a third party?" Which brings us to that Trojan horse known as the Flash Player.

Jeremy Allaire has been quite adamant over that past year or so, telling anyone who will listen that the Flash Player really is a Trojan horse. He claims the release of the Flash 6 Player was like "distributing hundreds of millions of next-generation communication devices, and then later turning them on." He is not far from the truth on that one, and the release of Flash MX 2004, Flash MX Professional, and Flash Player 7 not only turned them on but also turned those communication devices up to full power.

To see an example of the Trojan horse at work head over to www.brendan dawes.com/sketches/voices. As soon as you arrive at the page the video, with U2 soundtrack, starts to play. Brendan Dawes, who is a principal of Magnetic North in Manchester, England, shot the video during a trip to New York and chose Flash MX for a variety of reasons.

The video was assembled and output using Final Cut Express. Once Brendan was satisfied with the final product, he used Sorenson Squeeze (more about that later) to output the final .swf file. "I could have exported it as an FLV," says Brendan, "but for this purpose I had no need to add anything like interactivity to the video, so the exported SWF was ideal. I use Squeeze because it's superior to the built-in encoding in Flash MX. A definite buy if you're going to do video in Flash."

The piece that provided me with the revelation that Jeremy Allaire and Macromedia may be onto something is a short clip I use on my site, which can be found at www.tomontheweb.ca/ MMDesign/Storytelling3.htm. With this one I not only embedded the video in the page but created the video controls in Flash MX 2004 using the buttons and behaviors that ship with the application. This revelation didn't hit me until after the fact. When I was writing my latest book, *Building Dynamic Web Sites with Macromedia Studio MX 2004*, I included a review of the new video features of Flash MX 2004. When I got around to putting the Storytelling lecture up on my site, I had a couple of QuickTime videos to include so I dug out the chapter and created the player. It was when I used a browser to visit the page that it occurred to me that something very profound is going on here. No streaming indicator. No player. No codecs. Just a video with an audio track and an on/off switch.

What makes this so interesting is you can now create your own video interfaces and controls in FreeHand MX or Fireworks MX 2004, assemble and code them in Flash MX 2004 or Flash MX Professional, and quickly embed the resulting .swf file into a Dreamweaver MX 2004 Web page.

The remainder of this article details the creation of the video controls in

FreeHandMX and Fireworks MX 2004, the creation of the FLV file for use in Flash, and the assembly of the Flash movie in Flash MX 2004 Professional.

Creating Video Controls in FreeHand MX

We are going to create a rather simple Play button that contains the Up, Over, and Down states needed in Flash MX 2004. Here's how to create a 3D look using the tools in FreeHand MX.

1. Open a new document and select the Ellipse tool. Create a circle that is about 70 pixels in diameter.
2. With the circle selected open the Properties panel and select the Fill tab. Select a linear gradient and fill the gradient with two colors of your choosing.
3. Click on the circle and, with the Option (Mac) or Alt (PC) key depressed, drag a copy of the circle from the selection. Click the Fill tab, click on the gradient in the Fill panel, and when the gradient handles appear, change the gradient direction to the opposite of the original circle, which should be either vertical or horizontal.
4. Change the selected circle's dimensions to 50 pixels wide and 50 pixels high in the Property Inspector. Press the Enter/Return key and the circle will shrink.
5. Select the text tool and click once on the small circle. Enter the word "Play". With the word selected, open the Text Inspector and set the word's font to a sans serif and the text color to white.
6. Move the small circle on top of the large one. Select the circles and the word and select Modify > Align > Center Horizontal and Modify > Align > Center Vertical. This will line up all of the objects on the center point of the large circle.
7. To create the Over and Down states, copy the object and change the fill direction of the gradient for each object. You also might consider either changing the color of the text or applying an Envelope – Modify > Envelope > Create – to the text to give it the illusion of being pushed in or raised.
8. Convert each button object to a symbol by pressing the F8 key. Save the file. When finished, your buttons could resemble those shown in Image I.

Creating a Video Player in Fireworks MX 2004

Why Flash developers usually don't see much value in using Fireworks MX 2004 for interface creation is one of those great mysteries in life. The ability to add textures and patterns to shapes, special effects, AutoShapes, and to the ability to output button states from the application make its use a no-brainer.

The parameters for the video player are rather simple. The video is 240 x 180, meaning I need to create an interface large enough to hold the video and the buttons. The controls are rather simple as well. They need to be able to play the video and stop it.

I started with a blank Fireworks MX 2004 canvas whose dimensions were 350 pixels wide by 240 pixels deep. This left enough room to "hold" the video and the two controls.

I next selected the Rounded Rectangle AutoShape, drew out a rectangle that would fill the canvas, and then filled the AutoShape with the "Blue Wave" pattern. I adjusted the handles of the fill and then applied a small Gaussian Blur to it. In order to "contain" the video, I selected the Inner Bevel from the Effects menu and applied a 21-pixel Ring Bevel to the selection. Finally, to provide a "screen" for the video, I selected the Rectangle tool, drew a 240 x 180–pixel rectangle in the player, filled it with white, and applied a drop shadow to the selection (see Image II).

The button controls weren't terribly difficult to build. Recognizing that a Flash Button requires Up , Over, and Down states, you can quickly build these buttons by using the Symbol feature of Fireworks MX 2004. Once you create the base artwork for a button, select all of the pieces and then select Modify > Symbol > Convert to Symbol or, if you are familiar with MX Studio 2004, simply press the F8 key, which converts selected objects to symbols in Flash, FreeHand, and Fireworks.

When the Symbol dialog box opens, name the symbol and select Button from the Properties list. This will open the Button Editor. The selected object will appear in the "Up" area. To give the user a visual clue that the button in the Flash interface is live, I clicked the Over tab, clicked the Copy Up Graphic button,

selected the button on the Canvas, and darkened it by reducing the brightness to -66. (Effects > Adjust Color > Brightness and Contrast). I then clicked the Down tab, copied the Over button, and selected Invert instead of Brightness and Contrast in the Adjust Color menu. I simply repeated these steps to create the Stop button, see Image III.)

The Down state of the button to be used in Flash is created in the Fireworks Button Editor.

The buttons were then added to the interface as shown in Image IV.

Moving an interface from Fireworks MX 2004 into either version of Flash MX is not hard. What you have to do is remember that old adage, "Let the software do the work." In this case, you need only the interface and the three button states from each of the buttons created. Turn off the visibility of the two button layers and select File > Export Preview. The interface without the buttons will be visible, and this can then be exported out of Fireworks MX 2004 as a .jpg image.

The buttons are a little bit trickier. Turn off the visibility of the interface and a button layer but ensure the Web layer is still visible. This will leave only one button, with the green Web layer overlay, visible on the Fireworks Canvas. Select File > Export Preview and make sure the button is exported as a .jpg image. Due to its size you can apply up to 60% .jpg compression and not be too concerned about quality. To ensure you are exporting as small a file as possible, select the Crop tool in the Export Preview window and crop out everything but the button. Click Export.

The Export window that opens is the key. Name the button. (I used 'PlayButton'.) Select Images Only from the Save As Type dropdown menu and select Export Slices from the Slice dropdown. This will ensure that each button state is exported. Click Save and the three states will be exported to the selected folder. Before quitting, save the interface with buttons as a .png file just in case you have to make changes. Quit Fireworks MX 2004 (see Image V).

Preparing Video for Flash MX 2004/Professional

Though Flash MX 2004 contains a vastly improved video import feature, the
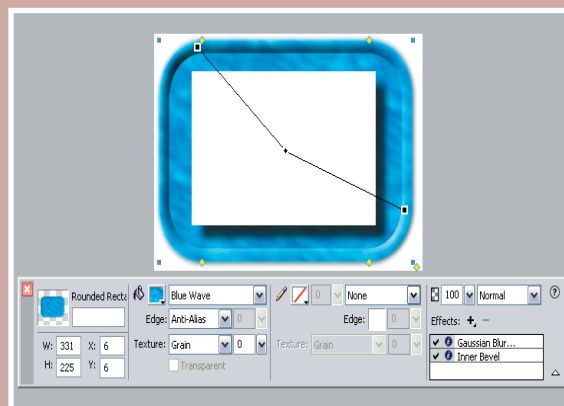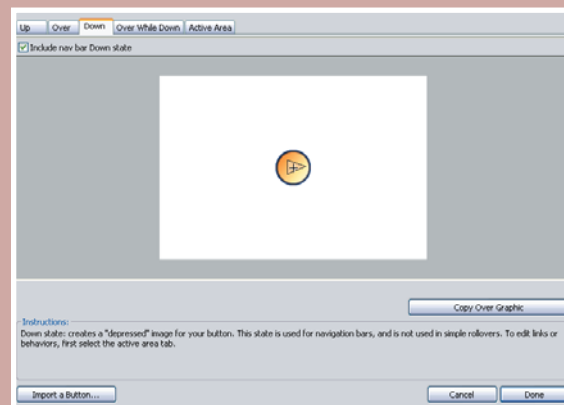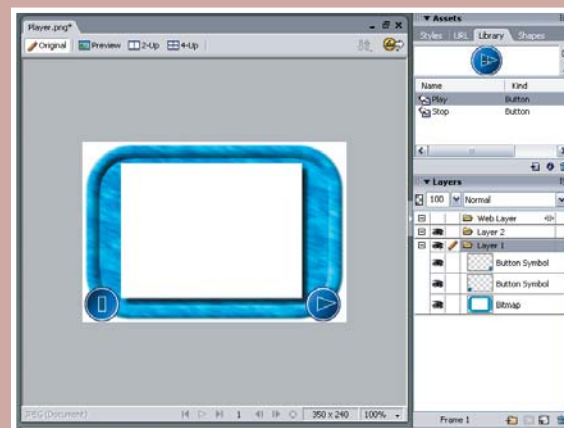
best results are obtained from Sorenson Squeeze. The reason is, though Flash contains a "Lite" version of the Spark codec, Squeeze offers full control of the output, ranging from the video and audio compression to be used to the final output, which can be the .flv file to be placed into Flash MX 2004 or the .swf file, which can be dropped into a Dreamweaver MX 2004 page.

In Brendan's case data rate and bandwidth were prime concerns. "I used the 300K streaming option in Squeeze." said Brendan. "I experimented by 'tweaking' the options before settling on that one. I

MX

wanted to maintain a certain level of quality but also have it load pretty quick on a good connection. I also wanted a good window size – there's not really much point in anything less than 320 x 240 when you're doing video. 300K streaming option seemed the best bet. Plus, the uptake of broadband files of this size is less of an issue. The piece clocks in around 6MB total, which is amazing considering the raw video file was 628MB."

Though that may seem like a rather large file size, the video I compressed was a 30-second capture output to QuickTime whose raw video capture weighed in at 108MB. The final FLV, output from Squeeze, using the settings shown, came in at 828K (see Image VI).
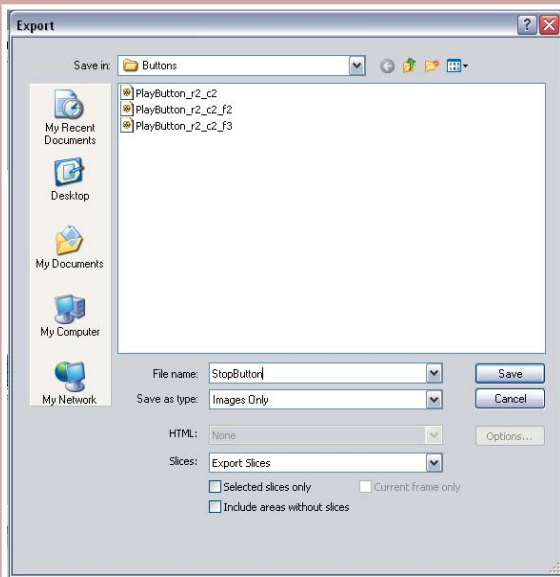
Assembling the Video in Flash MX

With the assets created, it's time to assemble the video player in Flash MX. The plan is to play the video in the interface and to have the buttons controlling the video sit on the stage just under the interface.

Once Flash MX is open, import all of the assets – interface, button states, and .flv file – to the Library (File > Import > Import To Library). In order to reduce clutter in the Library, I created a folder for all of the button pieces and moved them into the new folder. I then added layers for the interface, Play button, Stop button, actions, and labels.

I then created a new button symbol named "Play" and put the appropriate button states, as shown in Image VII, in the proper areas of the Flash button. I also created a Stop button in the same manner.

The Play button is created in Flash using the Button symbol exported from Fireworks MX 2004.

If you prefer to create the buttons in FreeHand MX, the steps to creating the Flash button are no different. The only major difference you will encounter is that the Vector Import dialog box will open upon import into the Flash MX 2004 or Flash MX Professional Library.

Once the buttons are created, drag the various interface elements onto the stage and place them in their final positions. Select each of the buttons and, in the Property Inspector, give them an instance name.

image VI
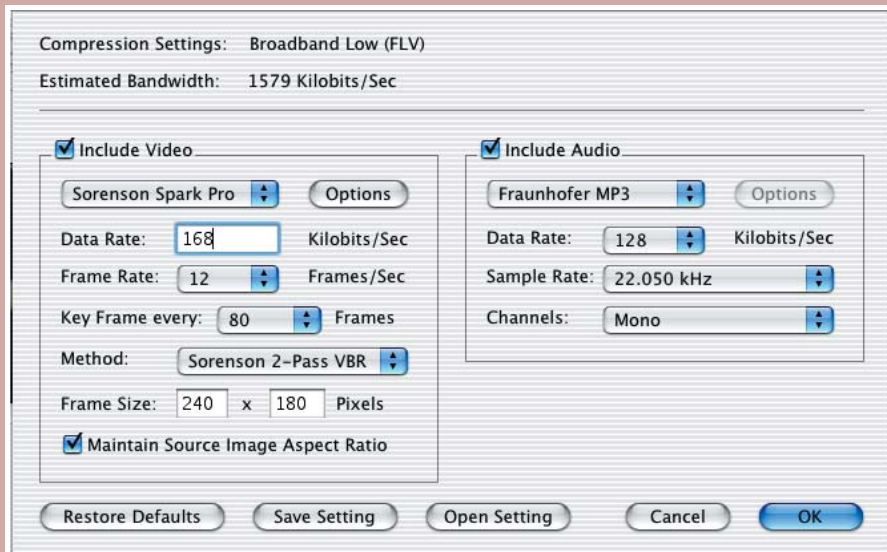
When you add the video don't be surprised to see Flash ask if it should extend the time line. Click OK and then add a frame at the end of the time line in the Interface and Button layers. Select the video on the stage and also give it an instance name. When finished, your Flash stage should resemble the one shown in Image VIII.

The assets are in place. Note the instance name in the Property Inspector for the selected button.

The next step is to add a label to Frame 1 so that when the video is finished, the playback head will be returned to the start of the video. The other reason is that the Stop button is also going to be used to turn off the video.

With Frame 1 labeled, select that frame in the Actions Layer and add a key frame. Select that key frame and add a "Stop" action- stop (); - to the frame. (You can press the F9 key to open the ActionScript Editor.) Scroll to the last frame of the Actions layer and add a key frame. Instead of a stop action you can add one that scoots the playback head back to the label in Frame 1 or back to Frame 1. I prefer to use the marker simply because I can move it around in the movie without worrying about the ActionScript.

For you ActionScript purists, the next step is heresy. This version of Flash MX 2004/Professional contains a new feature: the Behavior panel. With the loss of the Basic Mode for the ActionScript Editor, the thinking is to provide those new to ActionScript with a relatively painless method of coding. The video on the time line is "embedded," meaning you have access to the new Video Behaviors.

Select the Play button on the Stage. Open the Behaviors panel. If you don't see the Behaviors panel in your Panel Group, select Window > Development

image VII

Panels > Behaviors (Shift-F3) to open the Behaviors panel. The first thing you will notice is that the button's name and its instance name appear at the top of the Behavior panel. Click the "+" sign to open a dropdown menu of the behaviors available to you (see Image IX).

Select Embedded Video > Play. This will open the Play Video dialog box. Select the instance of the video in the dialog box. You should see "this.video Name", as shown in Image X, in the text input of the dialog box. You also have to tell Flash MX where which time line is in use. You have the usual choices – Relative or Absolute. Select Relative and click OK. The behavior will then appear in the Behavior panel.

You have to tell the behavior which video instance is being controlled.

Select the Stop button and add a Stop Behavior from the Embedded Video list. In this case, clicking the Stop button will send the playback head back to

Frame 1. If you are feeling adventurous, the rewind behavior is rather interesting. In this case, you tell Flash MX how many frames backward the playback head skips when the button is pressed. For instance, assume you tell Flash to go back five frames when the button is pressed. In this case, the playback head will skip backwards at five-frame intervals until the first frame of the movie is reached. (see Image XI).

Test the movie. At this stage you have built a Video Controller that looks nothing like that created by Apple, Microsoft, or RealOne. In order to keep the video's footprint on the Web page as small as possible, the last step is to reduce the stage size to fit only the interface including buttons.

The example I have used here just scratches the surface of the new video features of Flash MX 2004. If you have Flash MX Professional you have access to a full range of video features. For example, the new Flash Player 7 enables progressive download using the netStream controls to pull video from your server and start playing without waiting for the video to download. You can also avoid scripting altogether if you use the new Media Components and Component Inspector to build your own interfaces with full interactivity in Flash MX Professional.

Regardless of your approach, a reliance on third-party players is now a

thing of the past. When I asked Brendan about this his response was rather revealing. "Well, one plug-in is better than two!" he said. "Flash is everywhere so it makes natural sense to start to use it for showing video rather than asking people if they have QuickTime or RealOne. I've always had a 'shonky' experience with RealOne on a Mac anyway. Now you can use the same plug-in you use for your site, your app, or whatever. Plus, it can be integrated really well into a project – no pop-up windows!"

As I said at the beginning of this article, if I control the Web turf why should I cede a part of it to Apple, Microsoft, or Real? With the release of Studio MX 2004 and a Trojan horse, I may just have won the turf war. ⌒

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

Creating Cool Text Effects

Working with what's already there
by charles e. brown

I recently received a letter from a reader asking if I knew of any plug-in programs to create text effects in Fireworks MX. The reader seemed a bit surprised when I wrote back and told them that Fireworks already had the tools to get the job done. "Where is the button?" they asked. Many users feel that if a button isn't on the toolbar, the program can't do it.

In this article, we're going to create some stunning text effects with just a few simple steps and, more importantly, no additional plug-in programs.

Good text effects start with a good variety of fonts. Fortunately, there are a variety of Internet locations for downloading great fonts for free. One of my favorites is www.highfonts.com.

Once you've downloaded the fonts you want, check the directions for your operating system with regard to proper installation. Remember, you don't install the fonts just for the program but for the OS.

Three-Dimensional Text

Here is a simple way to create three-dimensional text. Once you master this basic technique, it will be easy to vary it for your own needs.

1. Open a new canvas to the dimensions you need. I usually like to start with a much larger size and then adjust it down later on. This gives me extra room to work and move things around. For this little exercise 500 pixels wide by 400 pixels high is sufficient.
2. I find that fonts with rounded edges make for nicer three-dimensional effects. After selecting the Text tool, I used the Property Inspector to select the Comic Sans MS font. I set the size to 100 pixels and the font to bold. Of course, you'll need to adjust these settings for your own particular needs.
3. For this example, I'm just going to type the word Welcome as our working text. You could use whatever text you want.
4. Once typed, you want to make sure that your whole text box is selected so that any settings you make will apply to all the text and not just where the insertion point is. I usually instruct my beginner students to click on the Pointer tool, click away from the text, and then once on the text again. This should ensure that the whole box is selected.
5. The next step is to clone the text by selecting Edit @> Clone.
6. In my own work, when I create layers I like to move the cloned text, temporarily, to a different area of the canvas. I find this makes it easier to see fill effects.
7. Open the Layers panel. You should see two layers now. Click on the two layer icons to see what layer goes with what text. It's always good to name your layers. Name the original layer Bottom_Text, and the cloned layer Top_Text.
8. Make sure the Top_Text layer is selected and open the Fill Color Box in either the Tool panel or the Property Inspector. Once open, select Fill Options…
9. For this example, I'm going to select the Fill Category of Gradient and a pattern of Linear. For the color, I'm going to use silver. Later on, once you have this basic technique, you could try any combination of settings. I like textures also.
10. Your Top_Text layer should now have a metallic look. We 'll leave the Bottom_Text layer set for black.
11. From here on in, the 3-D effect is easy. Using the Pointer tool, drag the Top_Text layer over the Bottom_Text layer and offset it a little to the left. You can use the arrow keys to move the text one pixel at a time. This allows you to adjust the amount of 3-D that you want.

You now have great looking 3-D text that was easy to create and, more importantly, very easy to control in terms of offset, colors, patterns, etc. (see Image I).

Attaching Text to a Path

Sometimes you might want to curve your text around an ellipse. For example, we can use the technique you are about to see have the text surround an image.

1. Start a new canvas. For this second demonstration, let's make the canvas 500 by 400 once again (as before).
2. We are going to begin by drawing the path we want our text to follow. Let's select the Ellipse tool and draw an ellipse approximately 475 pixels wide by about 250 pixels high. If necessary, set the fill color to none and the stroke color to black so that we just see the outline.
3. For the time being, we just want to use the top half of the ellipse as the guide. The bottom half will be used shortly. In order to split the top and bottom of the ellipse, select the Knife tool from the Tools panel. While slicing the ellipse from left to right, use the Shift key to keep the slice straight. After doing that, you can select each half and, if you want, move them apart slightly.
4. In the Layers panel you should now see two Path layers. Name the layer associated with the top half Top_Half. Name the bottom layer Bottom_Half.

5. Select the Text tool. For this exercise, I am once again using the Comic Sans MS. Set it to 75 pixels and Bold. I'm using black as the color, but you can use any color you want. To help position it properly on the guide, set the alignment to Center (using the Property Inspector).
6. Type the text Welcome To. Name this layer Top_Text. Its position on the canvas is not critical at this point.
7. Using either the canvas or the Layers panel, select the Top_Text layer. Using the Shift key, select the Top_Half layer. You should see both layers selected.
8. Select Text @> Attach To Path. (Note: If that option is unavailable, you don't have both paths selected.) Your text should now be aligned with the Top_Half path (see Image II). Notice that the two paths are now merged together.
9. Select the Text tool again and, with the same settings, type Fireworks MX.
10. Name this layer Bottom_Text.
11. Like before, select the Bottom_Text and Bottom_Half and, again, Text @> Attach To Path.
12. Your text is probably backwards and upside down. Unless you want to keep it like that, it is easily fixed by selecting Text @> Reverse Direction. Your finished result should look something like Image III.

You can fill in the center with an image of your choice. Also, the two halves can be moved at will.

Filling Text in with an Image

Here is a very startling effect: we're going to fill in the text with an image. For this one, I selected a photograph taken by a friend, June Daniel, in New York State. Also, to make this effect stand out, you should use a thick font. I am going to use an Arial Black set to 90 pixels and bold.

1. For this demonstration, I'm using a canvas set to 800 pixels wide by 275 pixels high. I imported my friend's JPEG image, taken with a digital camera, and stretched it to accommodate the canvas.
2. Name this layer Base_Photograph.

3. I select the Text tool and, as discussed, use Arial Black set to 90 pixels and bold. Type Fall In New York.
4. Name this layer Fill_Text.
5. Use the Pointer tool and position the text so that it is approximately centered on photograph. Precise placement is not critical. Image IV will show you the initial setup.
6. Select the Base_Photograph layer and select Edit @> Cut.
7. Click on the Fill_Text layer and select Edit @> Paste Inside. The results will be something similar to what you see in Image V.

The photograph is now inside the font as a fill. Of course you could change the background to anything you want to create some interesting effects.

Freeform Text

For a final demonstration, we're going to give each letter a life of its own.
1. Start a new canvas and decide what size you want.
2. Select a font and size it so that the words Fireworks MX fit on the canvas.
3. As before, use the Pointer tool and select the entire text box.
4. We are now going to break the text apart. Select Text @> Convert to Paths. This converts each letter to a vector path. Select Modify @> Ungroup. Each letter is now a layer of its own with its own life.
5. In addition to moving each letter around, you can use the Pen tool to reshape the letters. Image VI is an example of what is possible.

Summary

As you can see, we created some great effects by using some simple, built-in tools. In addition, they were not very difficult to create. Try combining two or more of the above effects to do even more.

There is no question that plug-in filters add a lot of great functionality. In addition, most are very easy to use. However, before you go out and spend money, take a look at what you can do with existing tools. You might be pleasantly surprised.

Next month we're going to look at built-in, and third-party, filters for Fireworks MX. ⬿

image I

image II

image III

image IV

image V

image VI

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

MX

Optimizing Your JPEG Slices in

Fireworks

by kleanthis economou
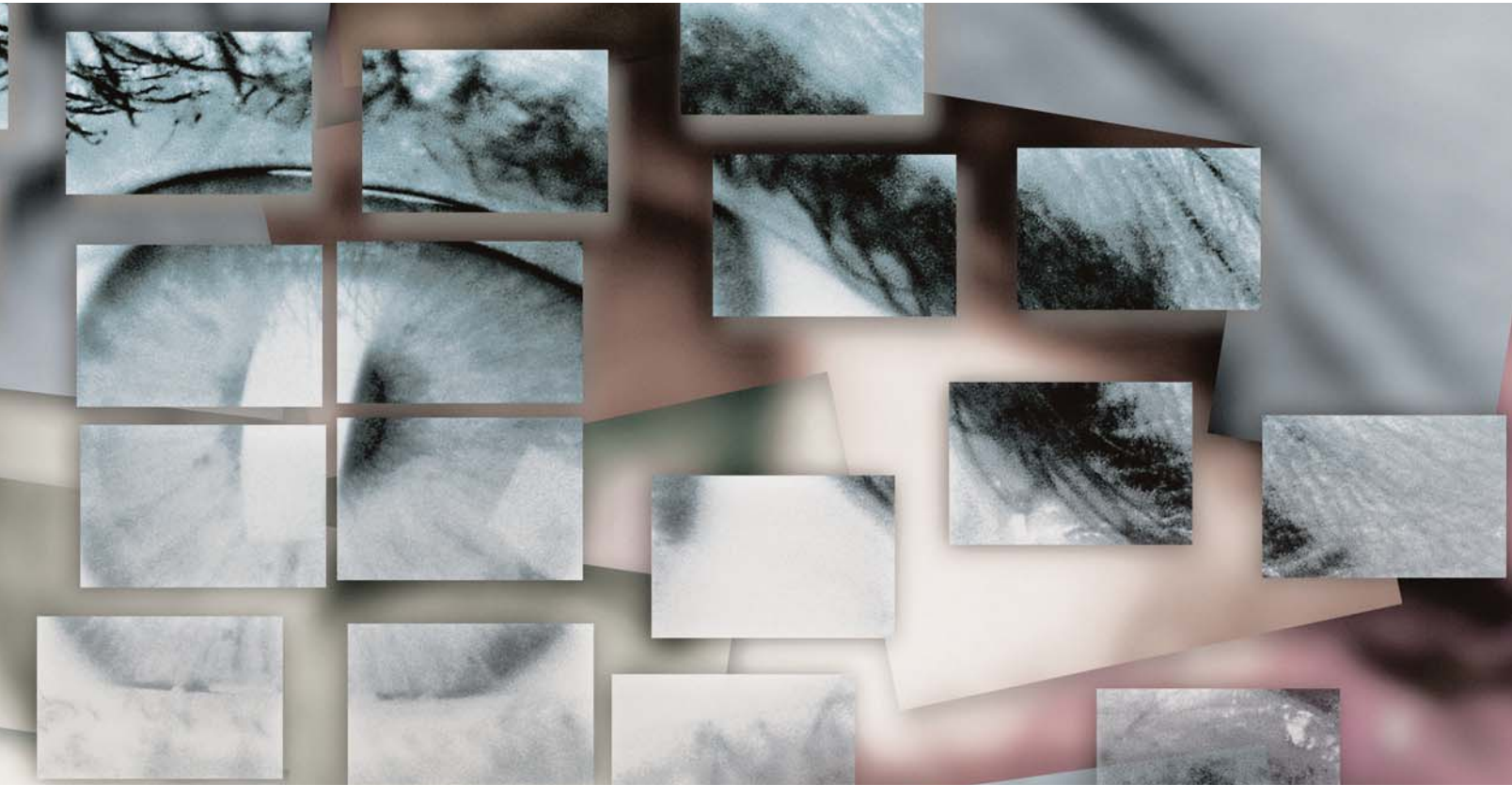
W

hen you want to make an image available on the Web, there are two important constraints. It needs to look its best and it must download fast. Those two rarely, if ever, go together. Better image quality means a larger file size, which translates to longer download times. Broadband connections help reduce download times but not everyone is sitting behind a fat pipe that is sucking up the Internet.

This month I'll discuss the JPEG format (pronounced "jay-peg"). When I sat down to write this article, a voice in my head kept screaming "this is so '97", and in a way I must admit it is. Then again, why is it that so many folks today still choose wrong image formats? Or don't know how to use selective JPEG compression in Fireworks? Um, selective what? Where? I seriously hope you didn't ask the last question.

The Basics

JPEG is a standardized image compression mechanism designed to compress full-color or grayscale images of natural, real-world scenes. The JPEG file structure splits the data into two parts – monochrome and color information – and applies different compressions to each of them. The monochrome portion defines the objects in a picture — it provides detail and brightness. The human eye is more sensitive to small brightness variations than it is to color variations. So discarding color information is less noticeable to the human eye than discarding monochrome information. JPEG exploits this fact. It compresses color information more than its monochromatic counterpart, and as a result it produces smaller file sizes without sacrificing their viewing quality —at least the way the human eye understands it. For the same reason, JPEG does a poor job when blocks of solid color are present, resulting in visible artifacts.

JPEG is "lossy," meaning when you use it to compress an image, information is lost permanently. The resulting image is not the same as the image you start with.

If you keep editing that JPEG and export it back as a JPEG as you make edits to it, image degradation will be obvious before too long.

The Joint Photographic Experts Group was the committee that wrote the standard, and this is how this acronym came about.

Compression Levels

You control the data you lose as you compress an image by altering its compression level. Higher compression levels result in smaller images with less image quality. You can trade off file size for image quality.

The compression levels are not standard in all the imaging applications that use JPEG. Some use a rating of 1-10, others 0-100. Some use this rating to refer to image quality while others use it to refer to compression level. Fireworks uses a rating for image quality between 0 and 100. Zero means minimum quality (maximum compression) and 100 means maximum quality (minimum compression). But don't let the scale trick you. Setting image quality to 80, for example, doesn't mean you will get 80% of the quality of the original image. It simply means that if 100 is the maximum quality then 80 is 20 clicks lower.

A common question is, "how much compression should I use?" There aren't any magic numbers. Experiment by adjusting the values until you get a result that has enough image quality and a satisfactory file size for your application. Fireworks defines two presets: JPEG - Better Quality and JPEG - Smaller Size that you can use as your starting point. Often they may be just right. However, you should experiment and compare results if you want to squeeze the most out of your JPEGs.

Optimize Panel

The Optimize panel (Window > Optimize) gives you access to various settings for each slice. With a JPEG slice selected, your Optimize panel will look similar to the one shown in Image I.

As I mentioned earlier, Fireworks uses a 0–100 rating for the JPEG image quality. You can adjust this value using Quality on the panel.

Selective quality is available when a JPEG mask is present. In a nutshell, it
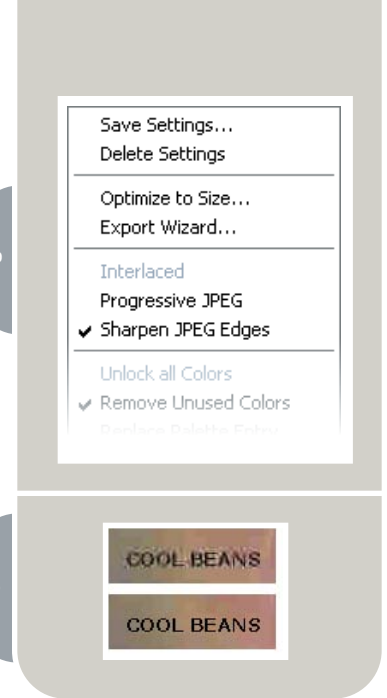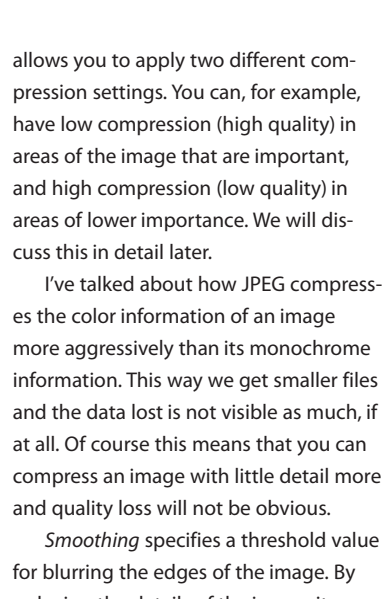
allows you to apply two different compression settings. You can, for example, have low compression (high quality) in areas of the image that are important, and high compression (low quality) in areas of lower importance. We will discuss this in detail later.

I've talked about how JPEG compresses the color information of an image more aggressively than its monochrome information. This way we get smaller files and the data lost is not visible as much, if at all. Of course this means that you can compress an image with little detail more and quality loss will not be obvious.

Smoothing specifies a threshold value for blurring the edges of the image. By reducing the details of the image, it becomes more compressible. A value of zero will not make any changes to the edges of the image. A value of eight (maximum value) will reduce details in the image heavily and the resulting JPEG will achieve a higher compression level, resulting in smaller file size.

I try to avoid resorting to smoothing to reduce file size since the images produced are "soft." There are cases, however, when the trade-off is acceptable and smoothing by 1 or 2 does not have a large impact and I can squeeze a couple of Ks out of it. It's all about trial and error.

Another option for a JPEG slice is sharpening the edges. This increases the details of the JPEG by sharpening it, which based on what we already know results in an image that appears to have more detail since our eye is more sensi-

tive to brightness changes. At the same time, since the edges of the image are more pronounced, the image can't be compressed as much and maintain acceptable quality. You can locate this setting under the options menu of the Optimize panel (see Image II).
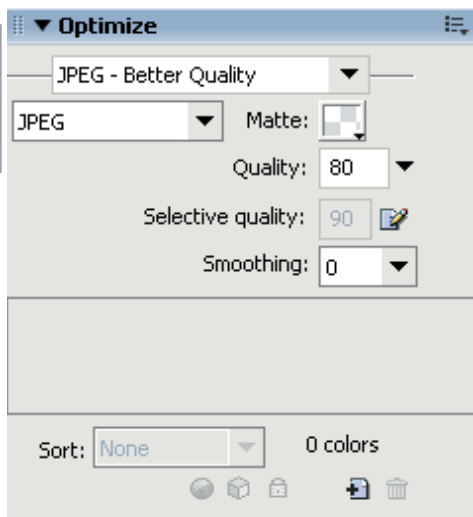
I suggest toggling that option on and off while in Preview so you can see how much it affects the quality of the image and the file size.

Progressive JPEG

Another choice for your JPEG slices in Fireworks is Progressive JPEG. A simple JPEG is stored as one top-to-bottom scan of the file. The decoder (that is, the browser when viewing the image on the Web) performs a single scan of the image and displays it. Progressive JPEG divides the file into a series of scans. The first scan shows the image at a low quality (heavily compressed) that has small file size and can show up quickly. Each scan that follows adds to the quality of the image. A Progressive JPEG is slightly larger in file size compared to a simple JPEG, but loading over a network seems faster.

You can set a JPEG slice as a Progressive JPEG in Fireworks using the options menu of the Optimize panel (see Image II).

Selective JPEG Compression

When you define a JPEG slice some areas may be more important than others. One such example is a portrait photograph where the background is out of focus. In a case like this you want to keep your subject's quality high; the rest, which is not as important, can be compressed more.

Although it is not the best approach to include text in a JPEG (remember our discussion about edges) sometimes you just have to. You can apply low compression around the text and a higher compression to the rest of the image. This way you can keep your text crisp, and achieve a small file size.

The technique of applying different compression levels in different areas of an image is called Selective JPEG Compression. Fireworks makes it easy to use selective compression.

Let's say I have this photo and I must include a label on it. I can compress it a

lot and still get acceptable results, but the text looks horrible (see Image III).

Selective JPEG compression to the rescue! With the JPEG slice, select Selective JPEG compression and bring up the Optimize panel. The *Selective quality* field is disabled. Click on the edit button next to it and the Selective JPEG Settings dialog comes up (see Image IV).

Check Enable selective quality and Preserve text quality. By doing so, any text block in the selected slice will have different compression levels. In my example I have a JPEG for which I can use quality 40 and selective quality 90. You can see how much difference it makes to the clarity of text in Image III.

Let's look at an example where we can define the area to apply selective quality manually. I'll start with the photo on the left in Image V.

In this case I want to keep as much quality as I can on the subject but I don't care about the quality of the foliage in the background. My first step is to create a selection of the area I am interested in. In this case I created a selection of the background (middle photo, Image V) and using Select > Select Invert I got the subject selected. With the selection active, choose Modify > Selective JPEG > Save Selection as JPEG Mask. In the last photo of Image V, the area with the red overlay is the JPEG Mask I've created.

Now I can create a JPEG slice for that photo and set its Quality low while the Selective quality is at a higher value. This way I compress the background more than the subject, dropping data from the less important areas of the photo.

Again, there are no magic numbers. Experiment with the values and find the ones that produce the best results for your test or the file size requirements you may have.
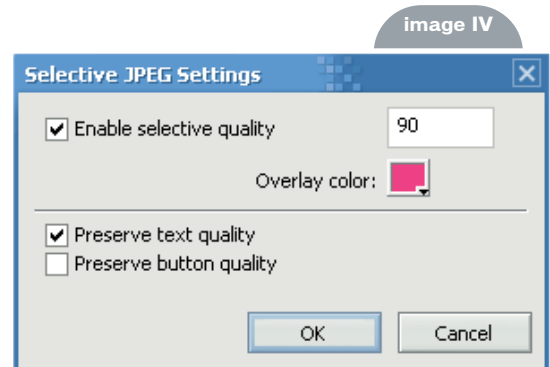
Under Selective JPEG, you'ill find other options that become available after you enable it. Use Restore JPEG Mask as Selection to restore your selection and manipulate it further. Then save this new selection again as the JPEG Mask you want to use. Of course, if you change your mind and want to start fresh, you use Remove JPEG Mask.

Kleanthis Economou has been a Web developer/software engineer since 1995 and specializes in .NET Framework solutions. He is a contributing author of various Fireworks publications and is the technical editor of the Fireworks MX Bible. *As an extension developer, Kleanthis contributed two extensions to the latest release of Fireworks.* kleanthiseconomou@sys-con.com

THE ART OF Freeh

by ron rockwell

F
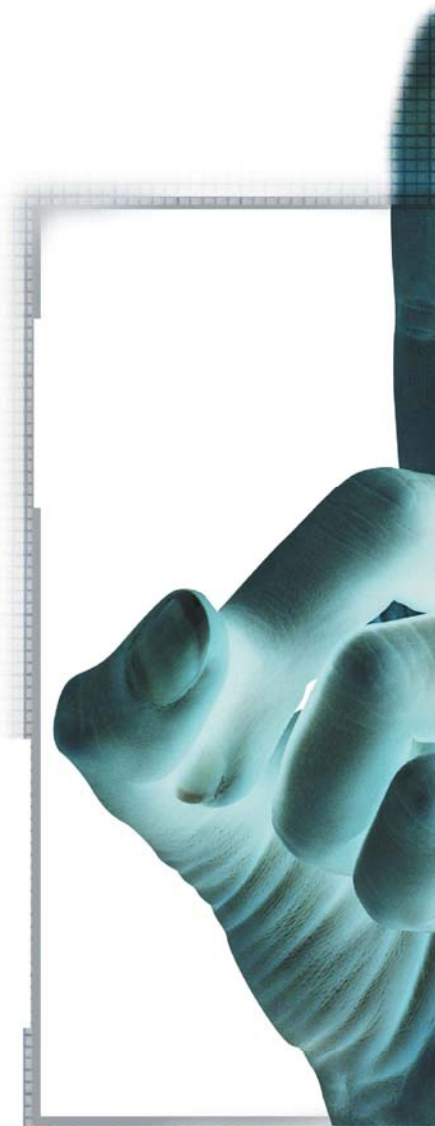
FreeHand MX has to be one of the most powerful programs you can have in your arsenal of graphic tools. By purchasing Studio MX, Illustrator users are now finding out that FreeHand is capable of many things that they couldn't do in Illustrator. One of the greatest advantages of working in FreeHand MX is the capability to create multiple pages in a single document.

With multiple page support, you can create anything from two-sided flyers and brochures, to 24- or 40-page catalogs or presentations. I wouldn't say that FreeHand would be your program of choice for a text-heavy 60-page catalog, but it could be done. For brochures, packaging, small to medium-sized catalogs, and instruction manuals, however, FreeHand is a killer app.

Finding Pages

When you first start FreeHand, a default letter-sized page is located in the lower left corner of the pasteboard. New pages will be placed to the right until the far right of the pasteboard is reached. At that point, the next page will go beneath the first page, and following pages will continue across the pasteboard as before. Eventually – depending on page size and quantity – you'll reach a point where the pages are added above the first row of pages. It starts to get confusing because page 1 is no longer at the top left of the group of pages.

To make a more logical layout, open the Document panel and click on the page icon in the bottom left corner. Then drag the icon to the top left corner. From this point on, any pages added will go to the right and down.

If you were to save and close the document, the page would still be in the top left corner the next time you opened the document. But if you opened a new document, the page would be at the bottom left of the pasteboard again. To remedy that situation, create a new default page.

Your Personal FreeHand Page

Start with a fresh, untitled FreeHand document and open the Document panel as shown in Image I. At the top of the panel you have the option to select a master page if you have any (we'll get into the construction of master pages later). The FreeHand default page is letter-sized for English versions. Open the drop-down menu to see the other choices. If you don't see an appropriate page size for the work you usually do, select Custom and enter the height, width, printer resolution, and bleed information in the appropriate fields. For this exercise, we'll stick with letter.

The page is dimensioned using X and

image I

image II

image III

Y numerical entries, where X is width and Y is height. In this case, the entries are 8.5 x 11 inches. If I had the units of measurement for the document set to picas, the entries would read 51 picas x 66 picas. There are two buttons that indicate

whether a page is portrait or landscape – make your choice and the X and Y values will change accordingly. If you have artwork that will print off the edge of the paper, you will need to set a bleed amount. Usually something in the range of one-eighth of an inch (0.125) or 1 pica will be fine for offset printing. Other methods of printing may have different requirements; check with your printer. Printer resolution tells FreeHand what you think the final output resolution will be, and uses this information to generate auto-created objects such as blends. The higher the number, the more steps. The three icons beneath the thumbnail window show the entire pasteboard, a medi-

um view, and a single-page view. As noted before, I like to have my pages laid out from the top right corner of the pasteboard, so I drag the page icon to that location. It's very important that after you've moved the page in the thumbnail window you double-click the page icon. Doing so targets the view to that location on the pasteboard.

Now the mechanical page is set up. Let's move onto the page itself. Start by defining guidelines and margins. Drag guidelines to fit your general workflow. If you usually work with two columns of text, place guidelines on the page in the appropriate spaces. If you're like me, there's one font that is your workhorse. Do yourself a favor and select that font and size once and for all now.

Go to File>Document Settings>Raster Effect Settings, and enter the printer resolution you expect to use. This also sets on-screen resolution – higher numbers will slow down your display.

You're trying to make life simpler for yourself, so think about your most common jobs. What colors do you usually need? Do you work in CMYK or RGB, picas or points instead of inches? Are 4-point PANTONE 209 strokes with square end caps the norm? Pull colors from the Swatches option menu, or create your own custom colors and add them to the Swatches panel. Set a common stroke and fill if you have one, and go to the bottom of the application window and set the units of measure to reflect your work.

Making Your Page the Default Page

Go to File>Save As. In the Save dialog box, open the Format dropdown menu and change the entry to FreeHand Template. Navigate to the Settings folder that's inside the English folder. On a Mac, it's found in this path: Library>Application Support>Macromedia>FreeHand MX>11>English>Settings. The path is similar on Windows machines. Name the file something easy to remember – mine is called Ron's MX Default Page. Click Save.

Now, open FreeHand's Preferences and select Document from the list. You'll see a menu named Select a New Document Template and a dropdown menu (see Image II). Click the Select button and navigate that long path to the page you just created. That's all there is to it. Every time a new document is opened in FreeHand, all of your personal touches be in place and you can get right to work. The neat part is if you share FreeHand with other artists, they can have their own default page by making the switch in Preferences. To top it off, you can put your default page on a floppy or e-mail it to yourself at another location. Just place it in the Settings folder and you'll be up and running in comfortable surroundings.

Note: Whatever you do, do not delete or move the FreeHand Default pages from the Settings folder. You may need them in case you have to troubleshoot at

a future date. They're labeled Default Letter, Default A4, and so on.

Changing Pages

What if your boss is the really artsy type that has the letterhead trimmed to an even 7 inches by 10 inches? Well, you can open the Document panel, open the page-size dropdown menu, and select Edit from the list. Click the New button. This allows you to enter a name for the page and customize dimensions. The list field at the bottom of the window (see Image III) will say Size-1, Size-2, and so on unless you double-click the name you gave the page. Doing so will put that name in the list and also in the page-size dropdown menu. Click the Close box to exit the dialog box.

Adding Pages

There are a few ways to add another page to your document. First, you should think about turning on Show Grid and Snap to Grid in the View menu. If you work on a Mac, turn the sounds on (in Preferences>Sounds). This is a great feature for an audible alignment of pages, objects, paths, points, and so on – but it can annoy other people.

At the bottom of the application window there are several very useful menus and indicators (see Image IV). The page view size and its dropdown menu, buttons to navigate forward or back a single page, the currently selected page, a dropdown menu to choose a page by its number, the Add Page button, and the view type and units of measure dropdown buttons. Click the Add Page button and a blank page the same size as the currently selected page will be added to your document. You won't get your guidelines, but there are other tricks to get around that.

Another way to add a page is to go to the Document panel and choose Add Page from the Options dropdown menu in the top right corner of the panel.

But possibly the coolest way to get a new page is to duplicate the page. First, you can go through the Document Panel Options dropdown menu and choose Duplicate. If you're like me and don't like to travel through menus, you can install the Duplicate button in your Main Menu bar (Windows>Toolbars>Customize> Tools/Panels). Duplication gives you

everything on the currently active page.

Possibly the most underused tool in the FreeHand arsenal is the Page tool. It provides still another way to duplicate a page. Simply select the tool from the toolbar and click the page you want to duplicate. Hold down the Option/Alt key and drag the page. You'll get a complete duplicate of the page, and you can place it anywhere on the desktop you wish. With Snap to Grid activated (whether the grid is visible or not), perfect alignment is guaranteed. Use the page tool to move any page to a new location on the pasteboard.

To make the best use of multiple pages in Web page layout and presentations, use the Connect tool to draw lines from objects on one page to objects on another page in order to show the flow.

The beauty of FreeHand's multiple page system is that you don't have to have pages of the same size. You can have eight letter-sized pages, a couple of tabloid pages, and a suitcase-sized package design all in the same document. Now, don't get me wrong, I don't advise that kind of an arrangement for the simple reason that if your single document heads south on you, you have a lot of reconstruction to do. But the capability is there, and it's great for experimentation in packaging and other collateral print work.

Master Pages

Much as a custom default page saves you time, a master page can really

image IV
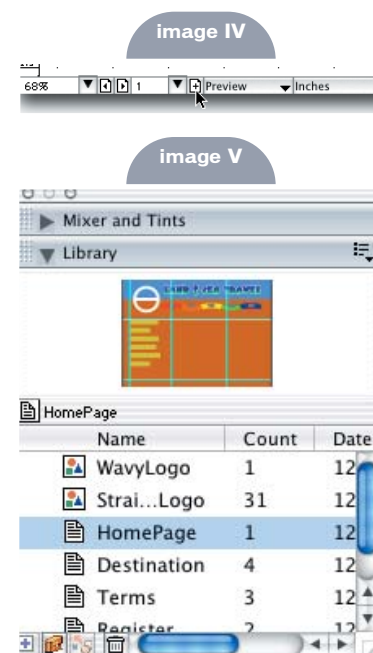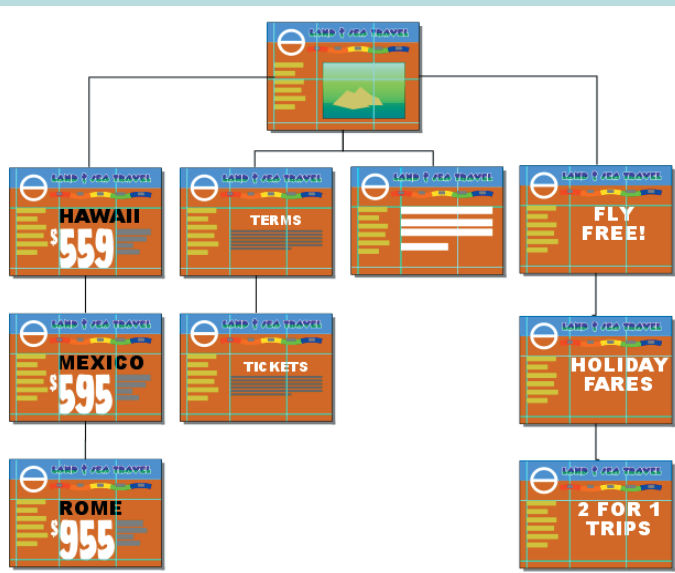
image V
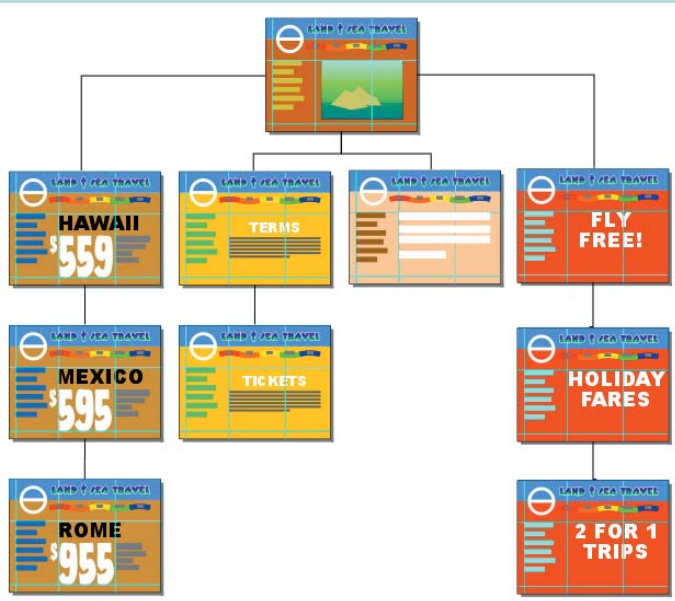
image VI

image VII

image VIII
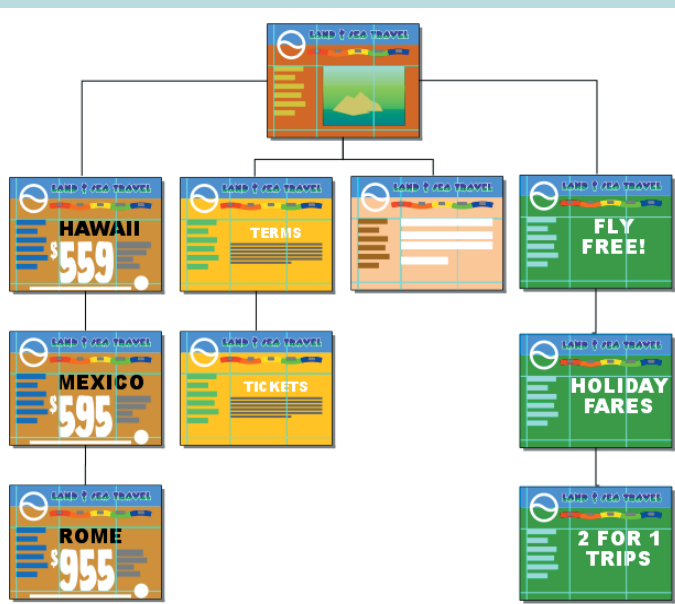
increase your output. At first you might think that you don't need to create a master page because you have templates set up for documents that you reproduce frequently. But a template can only be opened or imported, whereas a master page can be applied to any page at any time. If a global change is needed in the type of document you've made from the template, you'll have to do each document individually. With a master page, all you have to do is change the master page, and all its child pages will reflect the change (when the child document is opened again – obviously it won't affect a document that's sitting on a floppy disk somewhere). Master pages are stored in the Library, and appear in the Document panel, making them very convenient to use.

Prepare a master page as you would any other document. I like to create a page and convert it to a master page, but you can start a new master page and lay it out as you go. To convert a page, get everything exactly where you want it – guidelines, logos, phone numbers, photos – whatever elements you want in a fixed location. Then, go to the Document panel Options menu and choose Convert to Master Page. A generic name will appear in the Master Page menu. Open the Library and you'll see a thumbnail image of your new page (see Image V). Hover the cursor over the generic name in the Library for a second or two and the name will become highlighted so you can type an appropriate name.

To create the master page from scratch, choose New Master Page from the Library or Document panel Options menus. A new document opens with a page the same size as the currently active page. Put the page together the way you want it and close the window. The Library and Document panel have the new page in their menus, but the page is not visible in the document.

In order to retrieve a master page, you must export it. The process is simple – just select the master page in the Library list and choose Export from the Library Options menu. The Export Symbols window that shows all the symbols in the Library will open. Just as you would export any symbol, select, Shift-select for adjacent symbols, or Command/Control-select for nonadja-

cent symbols. When your selection is complete, click the Export button. That will open a standard Save window, but you'll be saving in the Settings>Symbols folder. Type a name for the master page and click the Save button. Nothing changes in the current document, but now you have access to the master page in future documents. All that's necessary is to select Import from the Library Options menu. The Symbols folder will open, and you can choose the master page you want. A dialog box opens that contains all the symbols and master pages in the group you've saved. Select any or all of them and click the Import button. The symbols and master pages will be listed in the Library.

Applying a master page to a standard page is a matter of dragging the master page icon onto an existing page, or selecting the master page from the drop-down menu in the Document panel.

Releasing the Child Page

Everything in the newly applied child page is locked unless you release the child page or modify its master page – you cannot change anything but the layering order or visibility. In fact, you can't even select anything! That's the design, it's not a flaw or bug. The master page is to remain intact. If you need to make adjustments in the layout, you must first release the child page. Select the page with the Page tool and go to the Document panel. Choose Release Child Page from the Options menu and the page is now a simple FreeHand page. Any changes you make to the master page that gave this page life will not be reflected in this page – there is no longer a connection between the page and the master page. If you have multiple pages to release, select them with the Page tool and release them all at once. To work on the new free page, you must first ungroup it.

Applying Master Pages - Making Child Pages

Starting a job with a master page is as easy as choosing it from the Document panel menu. The fun starts when you have several master pages that are similar, such as might be used in a Web site, newsletter, or product data sheet. You can prepare a "normal" page with new

elements, and then by choosing a master page from the Library, have a completed page on your screen. If you don't like the graphic arrangement, try another master page.

Making the Most of Master Pages

To get the most use out of a master page you need to plan ahead. I made a Web site for a hypothetical travel agency. I needed a home page and four other types of pages (in my example, the pages have a very common appearance). I created the home page with all the common elements that had been converted to symbols, placing key elements on individual layers. Then I chose the Page tool and selected the page. Holding down the Option/Alt key, I dragged a duplicate of the page to a new location beneath the original. Using the same technique, I made three more duplicates. Master pages were made from each of the pages in two ways. I wanted to keep the home page as a possible working space later, so I selected everything on that page and copied it. Then I chose Create New Master Page from the Document panel Option menu. I pasted the objects in the Master Page window, closed the window, and named the page in the Library. Then each of the other four pages was selected in turn, Convert to Master Page was selected from the Document panel Option menu, and the pages were named. Using the Page tool again, I Option/Alt-dragged copies of pages and arranged them as shown in Image VI. I added the necessary text.

I thought (hypothetically) that the layout looked a little boring, so I made changes to some of the master pages by double-clicking their master page icon in the Library. This opened the page for editing, and it was a simple matter to change background and text colors. When I closed the Master Page window, all of the child pages had changed (see Image VII), reflecting the edits.

I still wasn't satisfied, so I made a few more changes and modified the logo symbol. With two or three mouse clicks, I changed the logo on all 10 pages, as shown in Image VIII. All of this could have been done with a few template pages, but the process would have taken hours instead of minutes. By placing elements

on individual layers, I gained one more versatility factor: layers can be turned on or off, and rearranged in their stacking order – even in a child page, which is normally "locked." The key is that now I can add new elements on new layers that I can move under or over existing layers. With a little foresight, many design options can be built into the layout.

Printing Multiple Pages

Printing is fairly straightforward with a PostScript printer. But an old tool has been brought into the daylight with FreeHand MX – the Print Area tool. Use it to drag a red-dashed rectangle (Image IX) around any portion of your desktop that you want to print, then choose File>Print. In the FreeHand section of the dialog box, check Print Area. The page outline can be printed by checking Print Page Boundary. If the area is larger than the paper you will be printing to, choose Fit On Paper from the Scale% menu. Then click the Print button. *Only* the area you have selected will print – it's pretty slick.

Moving Pages

It's pretty common to need to move entire pages from one document into another. One way is to do a Save As and delete pages that you don't need. Another is to draw a rectangle around the borders of your page. Select the rectangle and everything else on the page and copy it. Then open a new document of the same size and paste the elements. I usually group the objects before maneuvering the rectangle to fit the page. Then I ungroup and delete the rectangle.

• • •

Multiple pages are just one of the strong points about using FreeHand MX. The more you experiment with multiple and master pages, the more you'll find you can build just about any graphic project in FreeHand. ⌒

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

MX

CF EXPLAINED

by simon horwith

For those of you who don't know who I am, let me begin by explaining that I am not a designer. I can't make pages or pictures pretty – heck, I haven't even figured out how to use some of the features in PC Paint yet. My expertise is in software architecture, programming, and business logic. In Web terms, I make all the stuff that runs on the server – the stuff that isn't displayed in Web pages. My tool of choice for doing this is Macromedia's ColdFusion Application Server. In addition to building applications with ColdFusion, I tech edit and write for *ColdFusion Developer's Journal* (*CFDJ*).

and now I use it all the time. Two years ago I also thought that the closest I'd ever come to doing Flash would involve a trench coat and being chased by the police.

Now I love working with Flash (but I'm still no designer). I also thought about the fact that design tools are so scary for developers, and that designers could very easily feel the same way about ColdFusion. The purpose of this article is to explain what ColdFusion MX is, how it works, and what its place is among the other Macromedia MX products, to all of the *MXDJ* readers who are unfamiliar with it. I'll begin by explaining exactly what ColdFusion is.

CF Works on the Server

ColdFusion is an application server – by that I mean that you install it on your server and it does all of its work on the server. You can think of it as being similar to a Web server. Web servers sit on the network and wait for an HTTP request. When somebody tells their Web browser to go to a specific URL, the Web server receives this request, finds the file (usual-

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Hello World</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
<cfset msg = "Hello World">
<cfoutput>
<h1>#variables.msg#</h1>
</cfoutput>
</body>
</html>
```

Well, when the folks at SYS-CON Media (the publishers of *CFDJ* and many other magazines, including the one you're holding right now) told me about *MXDJ* – one magazine to cover all of the Macromedia Studio MX products – I thought, "Who, me? Read articles about Fireworks MX, Director, and FreeHand MX? Yeah right. That's all I need… to try and wrap my head around more new technologies and IDEs."

But the more I thought about it the more I felt that maybe I really do need to better familiarize myself with the more "design-oriented" Macromedia products. After all, two years ago I detested the idea of writing code with Dreamweaver

ly an image, HTML, CSS, or JavaScript file), and sends the contents of that file back to the Web browser. The browser parses what the server sends it and renders the contents for the end user. Thus, the browser is doing most of the work and the Web server is acting as a file server, nothing more.

In the case of ColdFusion, when a browser sends a request to the Web server and the request is for a page written in CFML (ColdFusion Mark-up Language – the programming language used in ColdFusion pages), rather than sending the contents of that file to the browser, the Web server hands the request off to the ColdFusion Application Server. The ColdFusion Application Server then

opens the file and parses it, looking for any special instructions that tell the server to do something, and sends the result of running all of this code back to the browser.

What the Web browser receives is nothing specific to ColdFusion – it is still just text, HTML, JavaScript, and CSS. The difference is that rather than coming from a static "hard-coded" file, this content was generated dynamically. With ColdFusion, developers have the ability to create, manipulate, and output variables; perform conditional logic; redirect the user; retrieve data from a database and loop over that data; send e-mail; manipulate the local file system on the server; and so much more. In order to understand how the ColdFusion Application Server knows what to do when it looks at a ColdFusion page, and to better understand the reasons for creating dynamic pages rather than static HTML pages, let's look at a few typical ColdFusion code examples.

A Few Examples

We learned that ColdFusion looks through the contents of a page to find instructions telling it to do something before returning a page to the browser that requested it. These instructions are written in CFML. As the name implies, CFML is a mark up language. Similar to HTML, CFML is a tag-based language, which tends to make its syntax easy for HTML developers and designers to learn and to use. Unlike the tags in HTML, CFML tags all begin with the letters "CF" – this is how we tell the server what to do. Like HTML (which is also just text) you can write CFML in any text editor, but Dreamweaver MX is the recommended development tool and has many features to make writing ColdFusion applications even faster and easier than it already is.

When the ColdFusion Application Server opens a ColdFusion page (ColdFusion pages typically have a ".cfm" file extension) it parses the contents looking for any tags beginning with "<CF" and does whatever those tags tell the server to do. All other text (text, HTML, JavaScript, CSS, etc.) is ignored and the tags are processed in place. Let's look at an example. We'll begin with the classic "Hello World" application.

In order to tell ColdFusion to say hello, we will create a variable to hold the words "Hello World" and will then tell ColdFusion to output the value of that variable as an HTML header 1. The code is simple enough to look at without ever having seen CFML before (see Image I).

Looking at this code, you see that it is a typical HTML template, with the exception of a few cf tags. The <CFSET> tag is used to create a variable called "msg" with the value "Hello World." In order to display a variable, you first place opening and closing <CFOUTPUT> tags in the page. ColdFusion will resolve any variables contained within the <CFOUTPT> block. You put these variables in the page by surrounding the name of the variable with hash marks (#). Variables may exist in different places in server memory. These places in memory (commonly referred to as "scopes") may be omitted in most cases, but for code readability and performance it's a best practice to prefix variable names with their scope and a dot (to separate the scope name from the variable name). In this example you see that regular (local to the page) variables exist in the "variables" scope.

In this example the HTML header tags are wrapped around the variable within the <CFOUTPUT> block. ColdFusion not only ignores any tags that are not CF tags, but also any text that is not part of an instruction set. So, the <H1> tags are ignored even though they are within the <CFOUTPUT> tags. The header tags could have been moved outside the <CFOUTPUT> block (<H1><CFOUT-PUT>#variables.msg#</CFOUTPUT></H1>); the result is the same.

It's also worth noting that ColdFusion doesn't require the tags to be uppercase or lowercase and that variable names aren't case sensitive (#variables.msg# is the same as #Variables.MSG#); that's up to the developer. Also worth noting is that no data type is declared when creating variables. ColdFusion does not care if a variable is a number, string, date, complex variable, etc. – when you write code that expects a certain data type, the server will attempt to cast the variable value to that data type for you (this is known as "loose typing"). The result of browsing to the code above is shown in Image II.

The previous example is fine, but the results could have been achieved just as

"CF isn't pretty or glamorous like a Flash animation or a slick Fireworks DHTML menu, but it is an extremely robust and powerful tool that allows developers to create the kind of complex functionality that keeps visitors coming back to your site again and again"

easily with a static HTML file. Dynamic publishing is very powerful because of its ability to retrieve data from a database or some other source, and dynamically show this data.

Here's an example. You have a public Web site and want to display a catalog of products that you sell. Visitors should also be able to place an order by placing products in a shopping cart. Your company also wants internal applications that allow managers to process the orders, track the current inventory of products, and generate sales reports to show sales per month, YTD sales, most popular products ordered, etc. HTML alone will not allow you to create shopping-cart functionality – but of course you could do that with JavaScript or Flash if you wanted to.

If you create all of these pages with static HTML (and JavaScript or Flash), every time a product is added, removed, or changed you'll have to open each HTML file and modify it. This is a lot of maintenance, and can lead to inconsistencies if product information is not accurately updated in every page that uses it. What makes more sense is to store all of the product catalog information in a

database and create ColdFusion pages that retrieve data from the database and use that data to display the catalog. All of the reports would also pull their information from the database, and the current inventory would be stored (and updated when orders are placed) in the database. Now, by adding new products (or modifying or removing current products) to and from the database, all of the Web pages, reports, etc., do not need to be touched. This is because they reflect whatever is in the database in real time. Changes to your data now happen in one place, which keeps things more consistent and requires little or no maintenance of the Web pages and applications that use that data.

Explaining everything there is to know about working with databases is beyond the scope of this article, but like before, the code is very straightforward and easy to read. Image III shows what the code might look like to retrieve all product information from a database and display the catalog as an HTML table. I've added CFML comments (similar to HTML comments but they aren't sent back to the browser) to make the code easier to read.

image II

How CF Works with Other MX Products

Now that we understand what ColdFusion is and the basics of how it works, where does this fit in relation to the other Macromedia MX products? As mentioned earlier, when building a Web application or site, most data should come from a database. ColdFusion's role first and foremost is to retrieve that data. After retrieving the data, ColdFusion manipulates and/or outputs it directly within pages that have the look and feel of your site or app. This look and feel includes graphics and/or menus that might have been created with Fireworks MX or FreeHand. The placement of these graphics within HTML and other text is defined by the page layout (HTML or CSS) as it was defined within Dreamweaver MX. If the site uses Flash, again, graphics might have been created with Fireworks or FreeHand (or the Flash authoring environment), layout of all display was determined within the Flash authoring environment, and data should again be retrieved by Flash from ColdFusion (either using Flash Remoting or Web services).

As you may have guessed, the way the development process typically works is that static versions of HTML pages are created using Fireworks, Dreamweaver, FreeHand, and Flash (if applicable) and then "handed over" to a ColdFusion developer (which certainly might be the same person). A little bit of ColdFusion code is added (using Dreamweaver MX) and the static text is then replaced with dynamic text. The look and feel remains exactly the same as the "static" version, but the actual content is now easily maintainable in one place and is reflective of "real world" data.

Macromedia has been telling us and showing us that the future of Web development is all about user experience. Flash MX and the other Studio MX products allow developers to give these applications an amazing look and feel. Pretty much all of these applications require the ability to allow users to interact with a database, talk with other Web applications, or perform some other sort of complex functionality, and it is here that ColdFusion fills its role in the user experience. CF isn't pretty or glamorous like a Flash animation or a slick Fireworks DHTML menu, but it is an extremely robust and powerful tool that allows developers to create the kind of complex functionality that keeps visitors coming back to your site again and again. To get started with ColdFusion development, I recommend taking Macromedia's Fast Track to ColdFusion course (you can find a classroom near you at http://trainingreg.macromedia.com/tregexternal/). When seeking advice, Macromedia hosts many ColdFusion forums online (http://webforums.macromedia.com/coldfusion/) and the ColdFusion documentation includes a language reference, a "Developing ColdFusion Applications" guide, and a "Getting Started" guide to name a few – all in either HTML or PDF format (www.macromedia.com/support/documentation/en/coldfusion/). There are sev-

eral popular books on the market and SYS-CON's *ColdFusion Developer's Journal* (www.sys-con.com/coldfusion/) is well worth the subscription cost as well. Also at SYS-CON's site, you can subscribe to the *CFDJ-List*, a mailing list that allows you to seek and offer advice to other ColdFusion developers. You can take design only so far before you need to make your Web sites more interactive and functional than traditional tools allow, and ColdFusion makes this complex functionality fast and easy to achieve. There are many resources available to help get you started; I've named just a few.

You can download ColdFusion from Macromedia and use it for free for development and learning. The free license is the "Developer Edition" option when you run the installer, and is fully functional except that the server will respond only to requests from one browser, which is fine for development. Optionally, you can run the free 30-day trial (which is fully functional) and will revert to a developer edition after the 30-day trial expires (all ColdFusion versions share the same installer). Visit www.macromedia.com/go/trycoldfusion/ and download your copy today. Enjoy!

Simon Horwith is co-technical editor of CFDJ, and chief technology officer of eTRILOGY Ltd., a software company based in London. Simon has been using ColdFusion since version 1.5. He is a Macromedia-Certified Advanced ColdFusion and Flash developer and is a Macromedia-Certified instructor. simon@horwith.com

image III

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>My Company Catalog</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
<!--:: get data from DB ::-->
<cfquery name="qProducts" datasource="myDB">
    SELECT * FROM products
</cfquery>
<!--:: create table with one row of headers ::-->
<table>
    <tr><th>Product Name</th><th>Description</th></tr>
    <!--:: loop over query and output product names and descriptions ::-->
    <cfoutput query="qProducts">
        <tr><td>#qProducts.productname#</td><td>#qProducts.productdescription#</td></tr>
    </cfoutput>
</table>
</body>
</html>
```

MX

CFDynamics

A Division of Konnections Inc.

ColdFusion hosting

is our complete focus

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

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% uptime guarantee!
· 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

macromedia® ALLIANCE PARTNER

WWW.CFDYNAMICS.COM
866 - CFDYNAMICS
866-233-962-6427

Stage Fright!

Jump right in, the movie's fun!
by niqui merret

panic seems to be a common feeling for those new to Director, those who dare to launch it. We're going to try to ease the feeling of panic and create familiarity with common interface elements. You should also feel inspired to use Director for projects where you may be using Flash for comfort's sake. Director is a great tool that just needs a bit of patience and less fear.

What causes this fear? The interface would be one of the main factors. Why? It contains so many options. Most of the tools that you use extensively in a project are open and ready. This may, for a beginner, cause the interface to look far too busy and complex. In essence, the elements are very user friendly and easy to

work with. The ideal work environment would be dual monitors, one for the panels and the other for the stage. If this is not available, the MX interface allows for panels to be grouped together and hidden quickly.

The Director Metaphor

Let's begin with the Director metaphor. When you work with Macromedia Director, you become the director. This is your show! You get to sit in the chair and tell the cast members what to do and how to act, and they listen to you – what a treat! You have a cast full of actors that you call onto the stage when needed and then send them back after they have done their bit. How do they fit in? They join the score. The score in Director is not just for music; all media elements join the score when they are called to perform. At this point the elements are referred to as sprites and we add behaviors (or script) to add control. Behaviors tell any element how to behave – even the whole movie.

The Property Inspector

It is worth noting that the Property Inspector is one of the most basic ways to edit the properties of all elements. This is the same, in essence, as any other application that has a Property Inspector. It is content sensitive and will display the properties of the selected item. The great thing that sets it apart from the average Property Inspector is that it contains properties for all elements, at all times. You can switch between options without having to deselect an item. If you have a Flash sprite selected on the stage, you can access properties ranging from the sprite's location or size of the movie, to the Flash member's frame rate. Although it may appear to be a very busy inspector, it is extremely useful. Another key difference is that it is structured vertically rather than horizontally (see Images I and II).

The Stage

The stage is the window in which actions are viewed. This is also what the end users will see when they are playing the project. The stage size, color, and position are some of the basic properties that can be set in the Property Inspector. These properties can also be set on the fly through Lingo. Position and zoom can be set to make the authoring environment easier to use and then reset for final output.

At the base of the stage is a control panel. It is a useful tool that allows you to control the playback of your movie in the working environment. These playback options not only allow you to play, stop, and rewind, but also to play selected frames only, change the sound level, loop the playback, and jump between markers (see Image III).

The Cast

The cast is the storage space for all content that will be used and stored. Any content created on the stage is automatically added to the cast. When a new Director movie is created, a cast is automatically created with it. You can have multiple casts within a movie. The casts can be either embedded or linked. This is useful for sharing casts between Director movies. You can view multiple casts or keep the view to only one and swap between them as needed.

Everything can be targeted by reference to its location in a cast, either by name or by numerical position. If you are working with multiple casts, a reference needs to be added to the cast that the member is stored in. If you do link to cast members by numerical position, you should not move them within the cast as this will break the link. This reference is the same regardless of whether the cast is internal or external.

The property inspector will display information about a cast's name and its number in the list of casts. It will also dis-

play the file size and memory and whether it is stored internally or externally. The cast window can also be used to display information on the cast members. The list view of the cast can display a range of information from file size to date created (see Image IV). The thumbnail view is useful to view thumbnail previews of the cast members and is set to display the media type by default (see Image V).
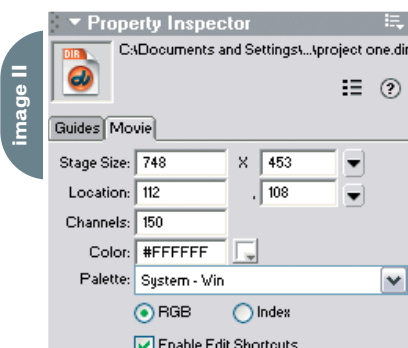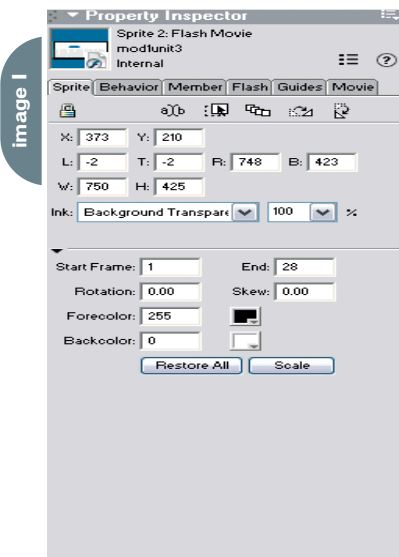
Cast Members

Cast members are media elements that are either created within Director content except for digital video. QuickTime, Avi, and Mpegs will automatically be linked to its external file and cannot be embedded in Director. Linked content can be very useful for quick updates to a project as it is easy to replace. The project can then be repackaged and distributed with minimal work.

The Score

The score is the timeline where all the elements in a movie are organized and controlled. The elements are placed in channels that define the stacking order able in Director, multiple views of the score can be viewed.

Markers can be used in the score for navigational purposes or for general authoring. They are useful for quickly jumping to a specific section of a Director movie. Markers are located at the top of the score window and are displayed any time the score is open. The control panel at the base of the stage allows you to navigate back and forth between them. If you wish to move a section to a different location on the timeline, and you have linked to markers,

"It is worth noting that the property inspector is one of the most basic ways to edit the properties of all elements"

using the basic bitmap, vector, text, and shape tools or they can be imported. Director can use external editors (applications) if you wish to edit imported cast members. You may edit some media types in Director.

The cast members can be either embedded or linked to an external file. If you link to external files, they must be bundled with the project for distribution. The default import option embeds all and sequencing. The main channels can contain all media types except for sound, which has its own special channels. There are also extra channels for controlling playback options such as the tempo (frame rate) and color pallette used in the project. These can all be changed at any time during playback. There is a channel dedicated to frame scripts that control a movie's behavior in a time-based manner. Although this is the only timeline avail- you would move the marker, maintaining all links (see Image VI).

Sprites

As soon as a cast member is placed on the stage it becomes a *sprite*. A sprite is merely an instance of a cast member. Any changes that are made to a cast member are reflected in every instance. Should you modify the properties of the sprite, however, the cast member is not
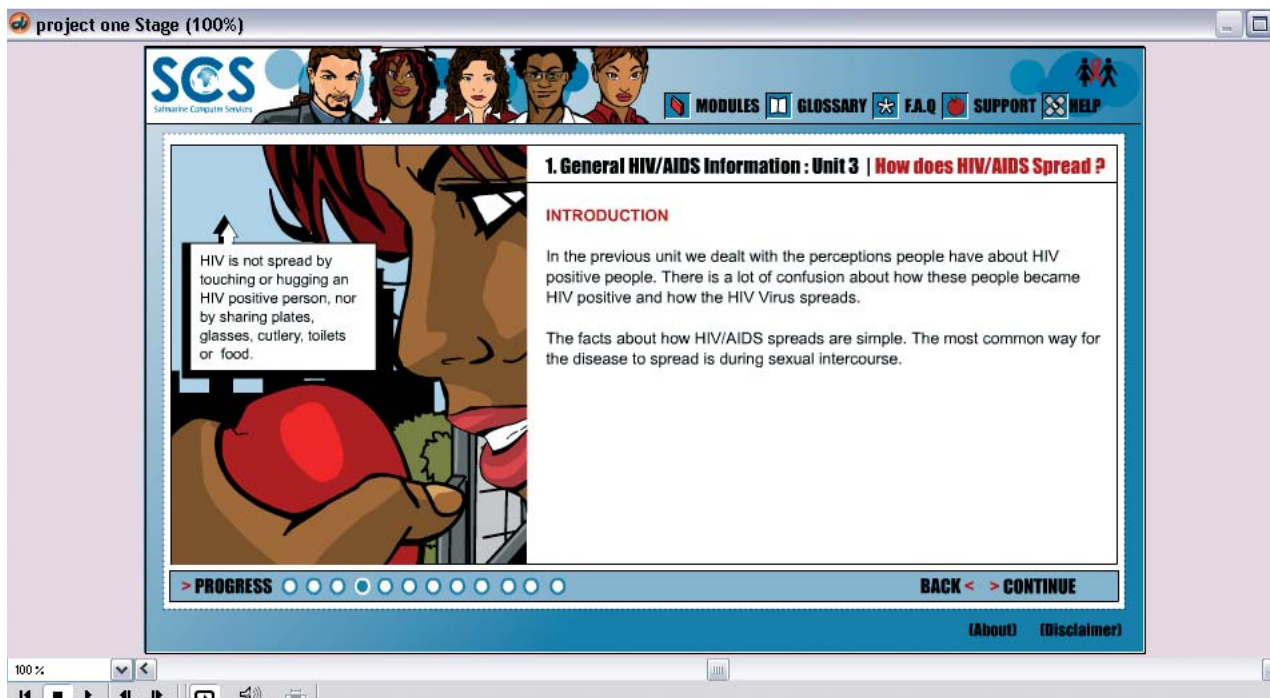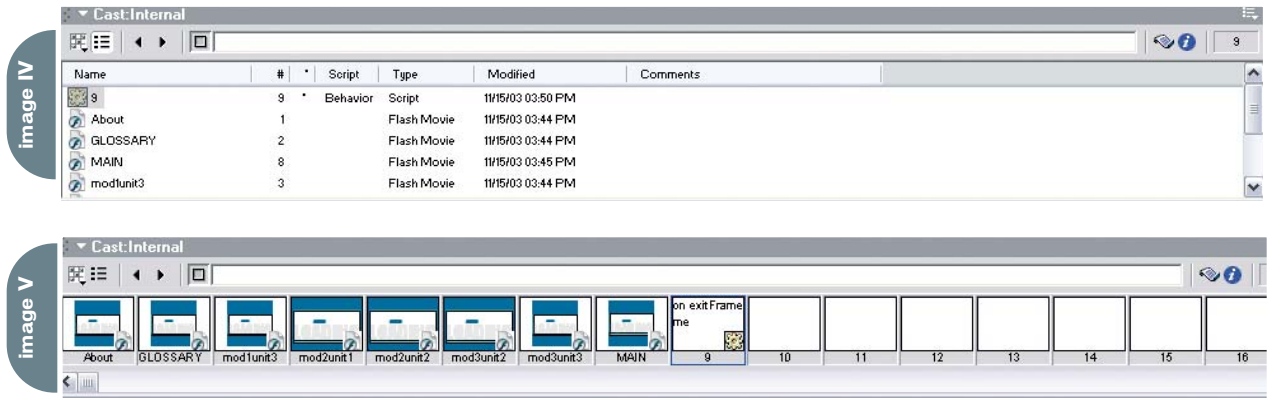
affected. The property inspector will display the cast and sprite properties. The cast properties will affect the cast member and all instances even if you have only the sprite selected.

If you wish to target any of the sprites, you will refer to them by the channel number in the score. You cannot give a sprite an instance name. This offers the benefit that you will never be in a situation where a sprite cannot be called because you have forgotten to name it.

The disadvantage is that if you do refer to them by number you lose the freedom to move them up and down the scores' channels.

Behaviors

Behaviors are prewritten Lingo scripts that allow you to add interactivity and special effects to a movie. They are attached to either sprites or frames, are stored in the library under logical categories, and are attached by dragging

them onto the target object. Behaviors respond to user actions or can respond to time-based events. Some behaviors may require parameters to be passed and will bring up a dialogue box that allows you to select from the various possible options. For example, if you wish to have rollover functionality you would use a rollover member change behavior, which will prompt you to select the cast member to display on rollover.

The property inspector has a behav-

iors tab that displays a list of behaviors and the associated parameters. These parameters can be modified if required and do not affect the other instances of the behavior. The behaviors are reusable throughout the Director movie and are stored in the cast. Should you wish to reuse a behavior, you can drag it from the cast onto the new object or use the menu in the behavior tab of property inspector. Additional behaviors can be downloaded from the Internet or created as needed (see Image VII)
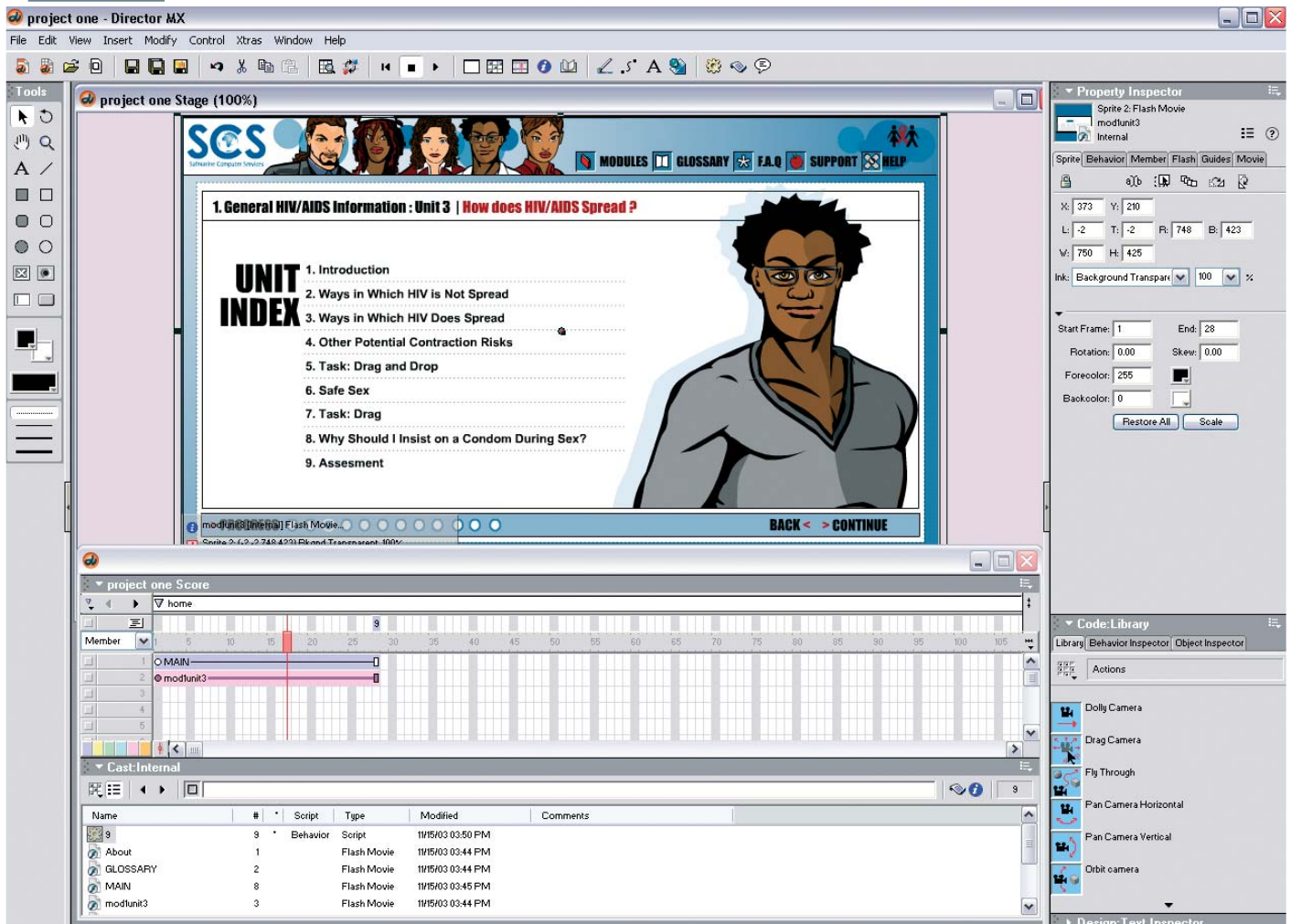
Xtras

Xtras are components that extend Director. They are used, even without your knowledge, in the most basic project. For example, Director uses a Flash asset xtra to view Flash movies within Director. Additional xtras can be downloaded from Macromedia's Web site or from third-party vendors. It is also possible to create your own xtras. Xtras are located in the program directory under xtras. To install a new xtra you need to place it in this directory. Once Director has been relaunched the xtra will be available for use.

Should an xtra not be available during playback, an alert will appear when the movie starts playing and a red binding box will be displayed as a placeholder for the missing content. This placeholder will be displayed in all instances where the missing xtra is being used.

Director knows which xtras are used in each movie. A list of the xtras can be viewed and controlled in the movie xtra dialogue found under Modify menu > Movie > xtras. All xtras that are used in a movie must be deployed with the project. They should either be placed in an xtras folder, added into the projector or available for download off the Internet.

Configure the Workspace

Due to the size of the Director interface, it is not always easy to know how to optimize the workspace (see Image VIII). One of the easiest ways of managing the workspace is to get to know the short keys, or quick links, for some of the more commonly used panels. The panels can be grouped together into tabbed view and then minimized and maximized on demand. The panels can be docked to each other and to the Director MX application window. Panel layouts can be created and saved for future use. Preferences can be set to control some of the default settings. Options range from the number of cast members that are viewable in one cast window to showing tool tips.

Time to Play

The best way to start is to jump straight in and get your hands dirty. Work through the sample movie that comes with Director to get comfortable with the workflow and reinforce the basic principles of Director. Work your way up from a simple project to more complex applications. The first step should always be to plan a project fully so that there is no confusion about the required functionality. Create as much content as possible before you begin. Most importantly, have fun. 〰

After many years as an instructor and developer in South Africa, Niqui Merret now works at the London-based rich Internet solutions firm, Prismix. She feels that she is a Director addict in a Flash-mad world. Illustrations by Kyra Merret. niquimerret@sys-con.com

image VII

image VIII

Tangled
in the
Machine

I have been using Director for many years now, for a wide variety of projects from Web games to kiosks, and it's a really great platform for developing complex projects quickly. The ability to "get my hands dirty" with Director's scripting language, Lingo, allows me the opportunity to take control of media in a way that's just not feasible with other software, and allows me to use Director in some very unusual ways.

by adam montandon
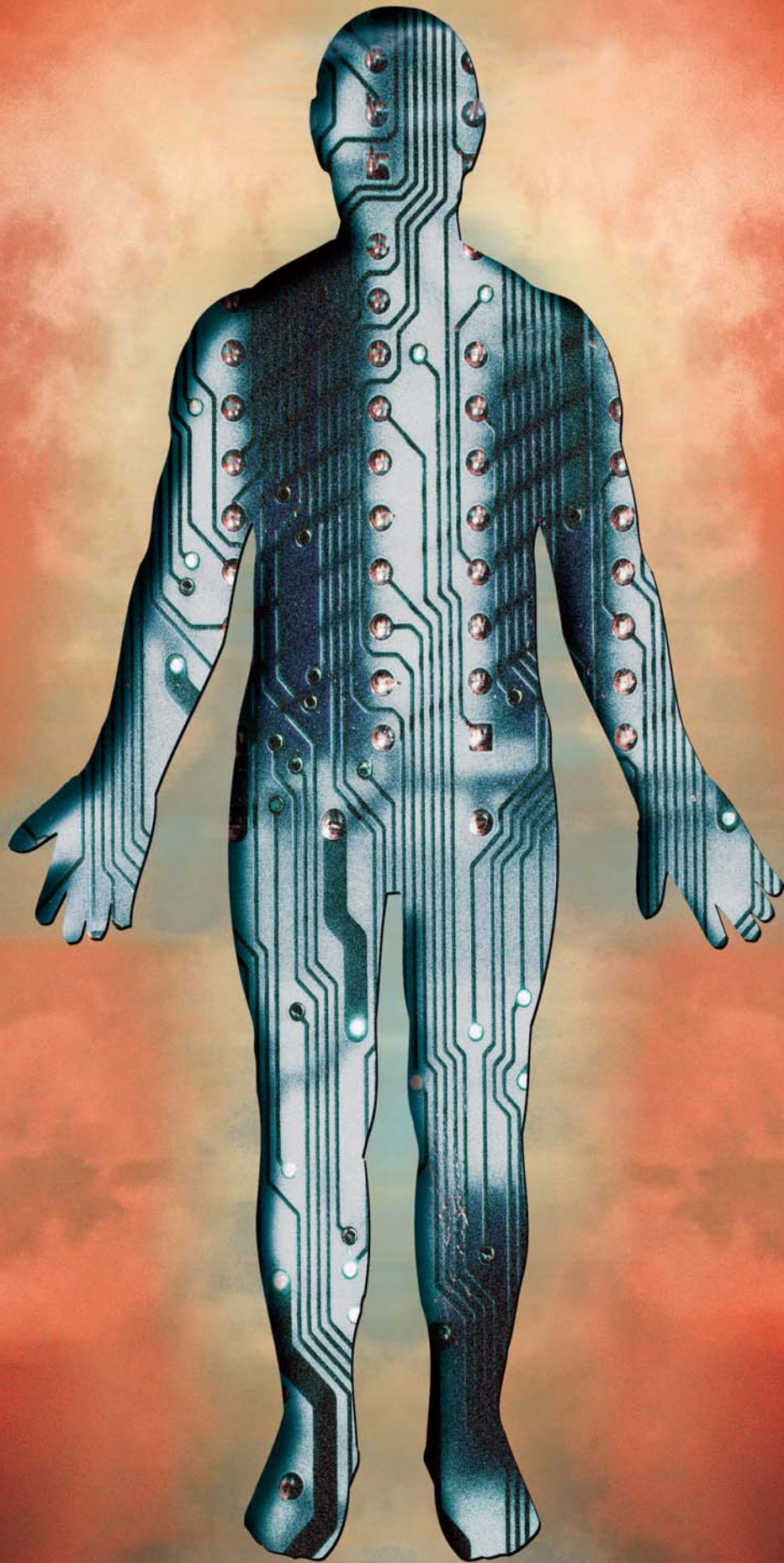
In my research at the University of Plymouth (England), I have been studying the relationship between the human body and its surrounding architecture. As we find ourselves in the dawn of a cyborg era, how can we expect our architecture to alter as we cybernetically enhance our bodies? How will we engage with what we know as space and place?

I spend a lot of my research time and effort creating customized hardware installations as a way of getting information from my body into a computer, and I don't want to waste a second worrying about whether my software can keep up. Director is an ideal prototyping tool that allows me to create a wide variety of multimedia output for my projects.

The MyBorg Suit

I set out to explore and experiment with cyborg systems, linking my body to its surrounding architectural environment. I created a custom hardware sys-

All this information wired across my body was then converted into MIDI data via a custom chipboard mounted inside my jacket pocket. I then fed the MIDI data into Director via an Xtra that enabled me to create software patches in Lingo that could assign any one of my sensors to absolutely anything within Director's authoring environment. This removed the need for a keyboard and mouse entirely and led to the development of a new range of projects that could be controlled directly with the body. By combining the MyBorg with Director's ActiveX components on the PC it was possible to patch into applications such as Web browsers, DVD players, and audio mixers. This worked particularly well as it gave a new physicality to a liquid, virtual concept.

By far the most intuitive use for the MyBorg is navigating 3D spaces. Using the MyBorg with Shockwave 3D, it is possible to spin, rotate, tilt, dip, and fly

graphics packages. It gave my MyBorg suit total control of every aspect of a virtual environment. This meant that I could spend less time on software development and more time on experimenting and researching the applications of my MyBorg system.

Lacuna

Based on the success of the original MyBorg suit, I set about creating a new architectural experiment that allows very physical architecture to extend itself into the realm of zero physicality, to become a hyperflexible space. Taking it a step further, I used sensors built within the architecture itself, upgrading from 16 digital nerves to thousands of high-tech sensors embedded within the brand new Portland Square building at the University of Plymouth. I was able to use the building as a fully extended sensory body. Data from absolutely every corner of the building, heat, light, movement,

> ## "By far the most intuitive use for the MyBorg is navigating 3D spaces"

tem that extended my body's nervous system by adding 16 new "digital nerves" that allowed me to sense changes in my internal body and my external architecture. I wired sensors for temperature, light, tilt, and vibration into the fabric of a jacket, placing the sensors in strategic places on my upper body. This jacket became known as the MyBorg, a wearable, portable, personal diary cyborg, digitally recording my every move. As I navigated through my environment the MyBorg's digital nerves sensed every interaction; vibration sensors on my chest allowed my breathing to be recorded, tilt sensors on my shoulders recognized which direction I was facing, light sensors allowed me to judge proximity to objects, and light sensors on my back enabled me to ascertain if someone was walking up behind me.

through a virtual architecture as if you were in it yourself. Because of the ability to completely control a 3D scene with Lingo, I could alter any aspect in real time with the movements of my body and my surroundings. It quickly became possible to relate changes in the real-world architecture, such as lights and sounds, to directly manipulate the virtual 3D architecture.

Director was the fast development environment I needed to make prototype 3D environments dynamically generated and controlled by Lingo that could be modified in real time by my movements. This created a unique reactive body architecture in virtual 3D space. Using the real-time 3D mesh features of Director, combined with imaging Lingo, meant that everything was created on the fly, no need for expensive 3D applications or

color, CO_2 levels, humidity, data, telephone calls, wind, rainfall, gases, emergency systems, electronics, Web-site usage, even how far open or closed a window or door is, can be digitally recorded and monitored. I used all of these inputs and more to alter in real time my software art project, called Lacuna. This created a unique representation of the building not in traditional space and time or virtual electronic space, but somewhere between the two.

Lacuna is an active response to the relationship between body and architecture in the cyborg era. Existing in the perceived "gap" between physical volume and electronic volume, Lacuna is customized software that communicates between the visual medium of the screen and a high-resolution electronic skin of real-world architecture. This in turn cre-

MX
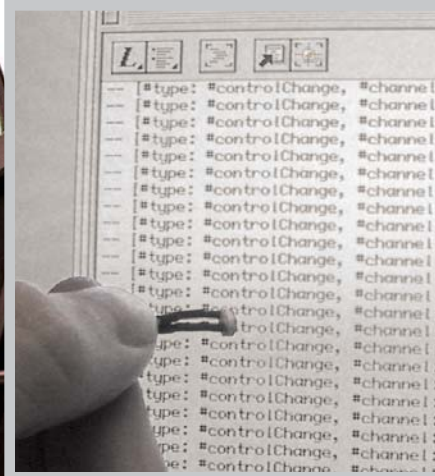
ates virtual counterpart architecture, enhancing the electronic cybrid dimension of construction.

Turning the architecture of the building into a giant sensory machine was made possible by a complex building management system that converted data taken from the building's sensors into XML. This was then passed on to a Web server that updated three times per second, providing a constant stream of fresh XML data over the Web. I was able to harness the XML and feed it directly into a Shockwave 3D screensaver version of Lacuna that was installed on every machine throughout the building, adding a new level of social interaction to the virtual dimension of the architecture. This allows people to navigate and explore the space around them in ways that are not normally possible with "dumb" buildings.

University of Plymouth's Portland Square building interact not through the dreary key-press or mouse-click but by their very presence, or lack of presence, and usage of the building is converted into electronic signals. As the occupants of the building turn on lights, open windows, use computers, or even flush the toilet, they change and control the animations on screen, but if they try to touch the keyboard or mouse, the screensaver automatically quits, directly challenging traditional notions of human/computer interaction. This, for the first time, allows visitors to exist simultaneously in the real and the virtual, they are everywhere and nowhere.

Lacuna is in a constant state of flux, and intelligently updates, modifies, and mutates itself many times a second. As a software organism it makes its own decisions, although it is open to interactive

inhabitants of our own software, we are tangled in the machine.

Personal Note

The MyBorg was initialized on May 8, 2003, with a live performance at MLAX, extending the 16 digital nerve endings into audio systems to create a real-time, body-responsive sound mix. I am continuing to develop new applications for cyborg systems. Lacuna is the result of two years of research into the production of cybrid spaces; my 46-page document entitled "Supermodern: Architecture for a Culture Without Boundaries" is available for download from www.stormsky.com.

Next Steps

My next project is to explore the playful relationship between humans and machines by writing and illustrating a number of bedtime stories especially for

> ## "Lacuna is an active response to the relationship between body and architecture in the cyborg era"

The visuals and graphics of the Lacuna screensaver are not the product of an expensive 3D program, or Photoshop, or any other graphical program, they are the result of pure XML data and code, an entirely generative piece played out by the interactions within the building itself. In this respect Lacuna can be seen as a "building player" that responds in real time to the data it is fed. This allowed me to keep file sizes low as there was no need to include graphics or animation. They are all generated live by the precise activity of the building at any given moment. This led to a file size of just 175k, small enough to be distributed quickly over the Web and the building's internal network.

Distributing Lacuna as a Web-update-able screensaver challenged traditional notions of interaction; visitors to the

influence by opening doors and windows, by adjusting heating, by the weather, by the very human interactions in, on, and around the building.

Lacuna is the first art project in the world to use this data to generate not a representation of a real building in cyberspace, but an electronic counterpart building that allows its inhabitants to fly beyond walls, to exist without physicality and to engage with the digital/spiritual nature of zero physicality. This transient data gathered from the real world is fed into a virtual building, creating light, colors, and shapes in a way that we cannot accomplish in the physical world.

Lacuna has no hard-wired ideas. Nothing in the program can be broken. It is generated entirely by interaction, a feedback loop between the body and digital/real architecture. We become

computers. Described as "stories to shut-down to", work on this project can be seen online at www.onceuponawebsite.com.

Finishing with a First Class Degree in MediaLab Arts, Adam Montandon recently won the Submerge Award 2003 for his programming and the Being Digital award for dedication in the field of multimedia. He is a serious multimedia developer who has worked for North America's largest contractor of virtual reality to the architecture sector and specializes in cyber-architecture and its relationship with the cyborg reality. Adam is now a researcher in digital futures at the Institute of Digital Arts Technology in the UK. Research Web site: www.stormsky.com. adam@onceuponawebsite.com
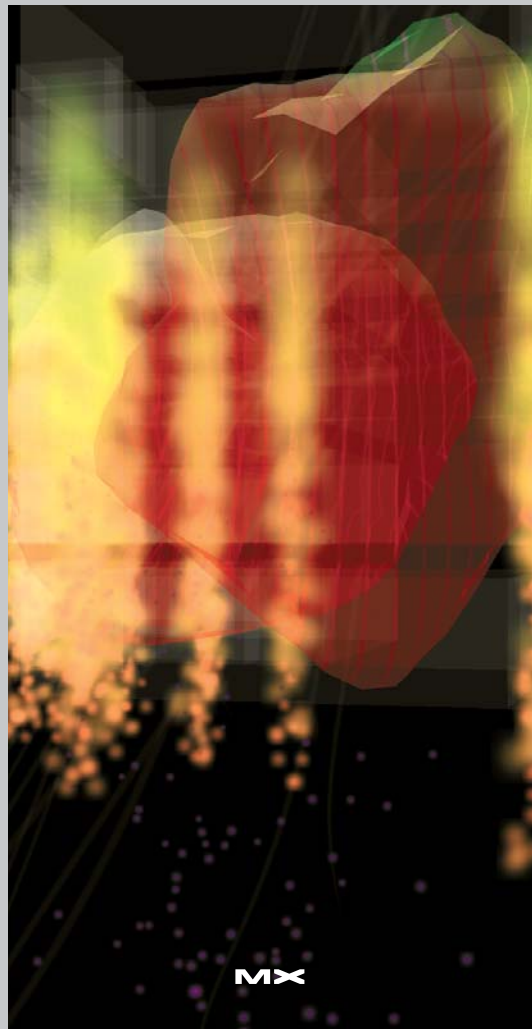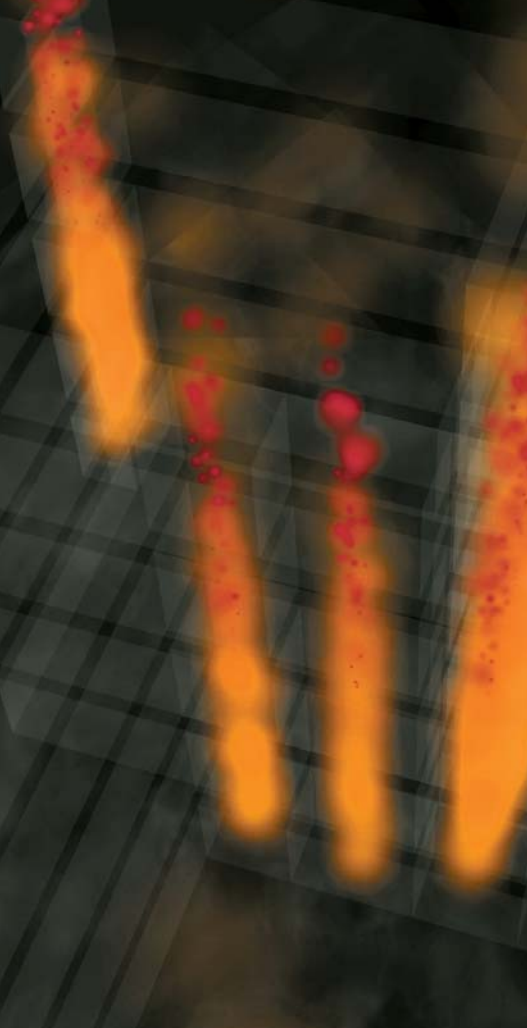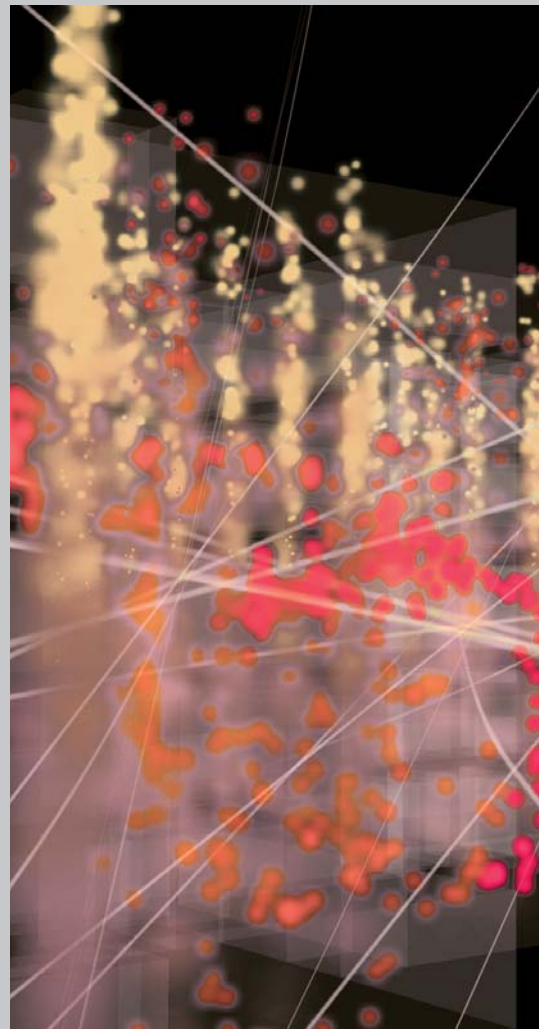
Multiple Undo

Power and flexibility
by james newton

application users change their minds and make mistakes. The undo item in the Edit menu has been a standard feature in all applications since the dawn of menu-driven applications. undo buttons in toolbars are more recent. Most modern applications provide a multiple undo facility. This requires a second redo entry in the Edit menu, and a second button on the toolbar.

Director MX is used by many developers to create professional-strength applications.

This article shows you how to provide a multiple undo button set for a simple Paint program in Director MX. This includes undo and redo buttons, each with a popup menu that allows you to undo or redo a whole sequence of changes with one action.

By the end of the article, beginner users of Director will be able to create a simple paint program with undo and redo buttons for the paint action. Users with a limited knowledge of Lingo will be able to extend the multiple undo feature to reset the choice of brush color. Intermediate users should be able to customize the Undo scripts for their own projects, and advanced users will have found a number of subtle tricks with standard Lingo features.

Test-Driving the Multiple Undo Feature

Point your browser at http://nonlinear.openspark.com/tutorials/undo/undo.dcr. (If you don't have the Shockwave plugin, you will be redirected to the download site.) This is a simplified version of the final movie that requires no third-party xtras. Once the movie has loaded, click on the yellow canvas area and drag the mouse to paint (see Image I). Paint several lines, keeping in mind that the movie is set to provide up to 20 undo steps. Now click on the undo button on the left. Keep on clicking until the canvas is blank again, or until you run out of undo steps. You can re-create your original painting by using the redo button.

Try selecting a different brush or the eraser tool, and using that on your painting. (Note that a double-click on the eraser will erase the entire painting.) The exact same script members are used to undo any of these actions. All the script members you require are available in the tutorial download (see below).

Now try changing the brush color by clicking on the color palette. Notice that the color swatch behind the eyedropper tool changes. Click on undo: the original color reappears. This action requires a different undo script, which you will have to create yourself by modifying a copy of an existing script.

All You Need To Get Started

You don't need any knowledge of Director to build this little paint application. All you need is the tutorial download for this article, which you can find at http://nonlinear.openspark.com/tutorials/undo/undo.zip or http://nonlinea.openspark.com/tutorials/undo/undo.sit, and a free 30-day trial version of Director, which you can find at www.macromedia.com/cfusion/tdrc/index.cfm?product=director.

If you already have Director 8.0 or later, you don't even need to download Director MX.

Building the Paint Application

Inside the tutorial package you downloaded, you should find a file named Canvas01.dir. Open this up in Director 8.0 or later. You should find a set of nine sprites already positioned on the Stage, and a set of behaviors in the Internal Cast library (see sidebar, "Improved Paintbox Behaviors").

To build the Paint application with its Multiple Undo feature, you simply need to drag the appropriate behaviors onto the appropriate sprites. Accept the default parameters except where noted in Table 1.

That's it. You're ready to run your movie and test the Multiple Undo and Redo buttons. Feel free to add more brush

table I			
Sprite 1	"Canvas" bitmap member	Paint behavior	(Defaults)
Sprite 2	"Undo arrow" bitmap member	Undo/Redo Button Set	(Default)
Sprite 3	"Redo arrow" bitmap member	Undo/Redo Button Set	Redo
Sprite 4	"Palette" bitmap member	Select Color	-
Sprite 5	"Color Chip" rect member	Show Brush Color	(Default)
Sprite 6	"Eyedropper bitmap member	Select Eyedropper	-
Sprite 7	"Eraser" bitmap member	Select Eraser	-
Sprite 8	"Round Brush" bitmap member	Select Brush	-
Sprite 9	"Slant Brush" bitmap member	Select Brush	-

image I

shapes or to modify the Stage layout.

How the Undo Feature Works

If you open up the Undo/Redo Button Set behavior in the Script window, you'll see that the mouseUp() handler says, in essence:

```
property action  -- #undo | #redo

on mouseUp(me) ---------------------
  -- ACTION: Undoes or redoes the
      most recent action
  ---------------------------------

  if action = #redo then
    Redo() -- in Undo Broker
  else
    Undo() -- in Undo Broker
  end if
end mouseUp
```

The action property is set when you drop the behavior on the sprite and choose Undo or Redo from the popup menu in the Behavior Parameters dialog.

The Undo Broker is a movie script. In fact, it's somewhat more than a movie script, but I'll leave the discussion of its complexities until later. All you need to know for now is that the Undo Broker contains a record of the last undoable action that the user made. This record is saved as an instance of the Paint Line parent script, which contains instructions on how to undo the paint action and

redo it again.

When you tell the Undo Broker to Undo(), the Undo Broker forwards the message to the instance of the Paint Line script, which actually does all the work. Let's first look at what an instance is, then look at what work it needs to carry out in order to undo the painting of a line.

Parent Scripts and Instances

A script instance is like a trained dog. When you give it the right commands, it does exactly what its handlers have taught it to do. A parent script is like a cross between a dog breeder and an obedience class. All the offspring of the script have the same general behavior but different characteristics.

You can breed as many offspring from a parent script as you like, and give them their different characteristics from the moment of their birth. They will all obey the same commands.

Or to use Lingo terms: you can create as many instances of the script as you like, and provide them with property values in the new() handler. An instance is in fact an address in the computer's memory that stores a number of property names and the values that are associated with them. When you give a command to an instance of a parent script, the command handler "knows" which property values are stored by that instance at that particular address in memory. Depending
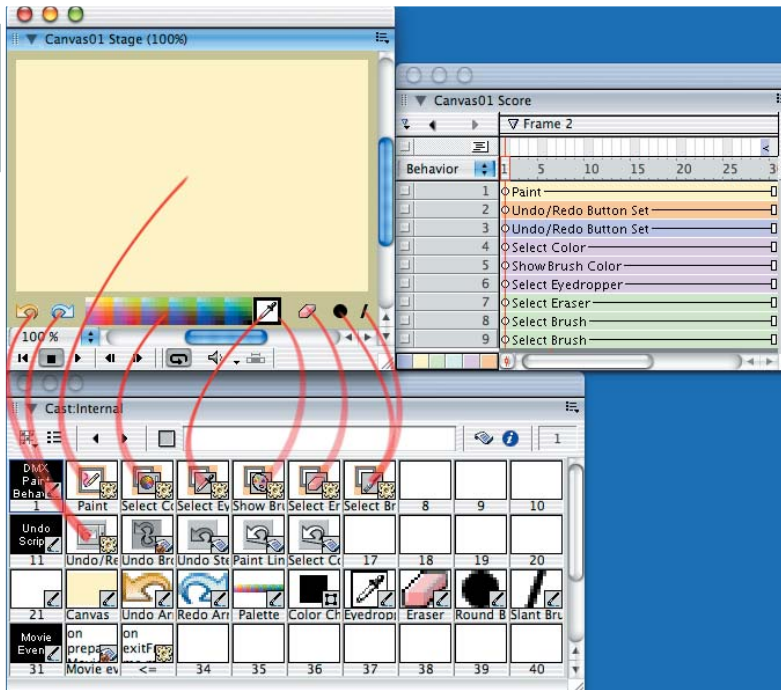
Improved Paintbox Behaviors

If you have already used the behaviors in the Paintbox section of the Library Palette, you might recognize some of the icons used for the behaviors in the tutorial movie. Don't let this fool you: these behaviors are a completely different set from the ones I wrote for Director 8.0.

Director MX runs on Mac OS X. On Mac OS X, the system automatically displays a busy cursor (fondly known as the spinning pizza of death) if an application runs a lengthy process. The original Canvas behavior, bundled with Director 8.0, uses a single process as long as the mouse is held down. In Director MX on Mac OS X, this brings up the busy cursor after about a second. This is not an issue on Windows.

I therefore rewrote the Paintbox behaviors for my own personal use. You will notice that I added two new ones: Select Eyedropper and Show Brush Color. This new set is much simpler than the original set of Director 8.0 behaviors… except for the Undo Paint behavior.

The Undo Paint behavior has been replaced by a set of four scripts: an Undo/Redo Button Set behavior, an Undo Broker movie script, and two parent scripts – Undo Step and Paint Line. The advantage of adopting this new technique for the undo feature is that it is not limited to undoing a paint action. As you will see, you can create a simple script, similar to the Paint Line behavior, to undo and redo other actions. The undo feature thus becomes generic: you can use it for any undoable user action. I explain how the Undo Broker and the Undo Step scripts work later in the article.

on the values it holds, each instance will thus do something different, by following the same rules.

Undoing a Paint Line Action

It's easier to understand this if you can see it in action. Let's take a moment to consider what information is needed to undo a paint line action. We need to know:

- Which bitmap member is being altered by the paint line action

a different rect and different before and after images for that rect. Code I shows you how the Paint Line parent script produces a new instance.

Given these starting points, undoing the paint action is simply a matter of swapping the current image inside the dirtied rectangle for the image that was there before.

```
on undo(me) -------------------------
  -- ACTION: replaces the current
```

To make the Paint_SetBrush Color() action undoable, you need to tell the Undo Broker what the color was before and after the change, and which script will handle undoing the change (see Code III).

What purpose does the isUndo parameter serve? And how does the Select Color Undo script work? The answer to both of these questions is in the extracts from the Select Color Undo script shown in Code IV. The

- What is the smallest rectangle that can fit around the new line
- What that rectangle looked like before the line was drawn
- What that rectangle looks like now

Each new line drawn "dirties" a different rect, so each undo action will refer to

```
image within pRect with the
--     original image
-------------------------------------
  pMember.image.copyPixels
(pBefore, pRect,
    pBefore.rect)
end undo
```

Redoing the change is just as simple:

```
on redo(me) -----------------
-
  -- ACTION: replaces the
original image within pRect
with the
  --     altered image
---------------------------
--
  pMember.image.copyPixels
(pAfter, pRect, pAfter.rect)
end redo
```

Creating a Customized Undo Script

As your movie stands, you can change the color of the brush but you can't undo that change. When you click on the color palette, the event #Paint_Set-BrushColor is sent to all sprites. Here is how the Show Brush Color behavior, on the sprite behind the Eyedropper tool, currently handles it (see Code II).

undo action simply calls the same Paint_SetBrushColor() handler you saw above. Since it passes a value of TRUE to isUndo, no new instance of the Select Color Undo script is created.

As you can see, the principles are simple. You define the before and after state in the new() handler, and you provide a means of switching from one to the other (and vice versa) in the undo() and redo() handlers.

I have removed the six crucial lines from the Show Brush Color behavior, and another six lines from the Select Color Undo behavior. All you need to do to make the feature work is to type in the lines shown in Code III.

If you have followed the logic this far, you should be able to create your own Undo parent scripts for other undoable actions that you might need in your projects.

Chained Lists and Upside-Down Ancestors

The Paint Line and Select Color Undo scripts are simple because the real complexity is in the Undo Broker movie script and the Undo Step parent script.

An instance of the Undo Step script adopts an instance of the Paint Line or the Select Color Undo script as its ancestor. The Undo Step instances provide the links that chain the multiple undo actions together. Each instance contains a pointer to the previous undoable action. When an action is undone, its Undo Step

instance becomes the first in a chain of actions to redo.

Normally, ancestors are used the other way around: the generic script is the ancestor of the specific script. Here, I make the specific script (Select Color Undo, for instance) the ancestor of the generic Undo Step. The idea is to make the specific scripts as simple as possible. There is no need, for instance, to declare an ancestor property.

they give no feedback to indicate that the user's click has been registered.

The final version of this movie is CanvasPro.dir. This uses the OSControl xtra to provide platform-specific buttons, and to provide button menus that allow you to choose how many steps to undo or redo at a time. The behaviors have been adapted to work with OSControl sprites, but you can use them on your own buttons if you prefer. The behaviors in the CanvasPro.dir movie are more robust and end-user

Setting Up the Undo Action

been a standard feature menu-driven applications"

By inverting standard practice, I make it easier for you to create your own undo action scripts.

Brokers, or Self-Instantiating Movie Scripts

The Undo Broker is a movie script. It thus responds to global calls to UndoAction(), Undo(), and Redo(). The first time any of its public methods is called, the Undo Broker creates an instance of itself, and stores that instance as a property of itself. All public methods are diverted to private methods in this script instance. The properties of the script instance are kept private.

The Undo Broker thus provides a movie-wide feature without using any global variables. You can add it to any of your movies without needing to worry about clashes with the names of your global variables.

Not only that, but the action of the Undo Broker is limited to the movie itself. If you have several document windows open in your Paint application, each with its own Undo Broker, each paint document will perform its own undo actions independently.

A Professional-Strength Version

The movie you have been working on contains simplified scripts to make it easy to see the wood for the trees. The buttons are just areas to click on;

friendly than the ones you have been working with. To encourage you to use them in your own projects, they are organized into two separate external casts, which you can add to the Libs folder alongside your copy of Director. When you next open the Library Palette, the new behaviors will be there for you to use.

Conclusion

This article has dealt with Imaging Lingo, behaviors, and parent scripts. It has shown you an unexpected use of movie scripts, and turned the notion of ancestors on its head. Some of these concepts may be new to you, and you may need to study the movies in more detail on your own to come to terms with them. But whatever your level of competence with Director, you should come away with something useful: an understanding of the power and flexibility of Lingo, two new sets of behaviors in your Library Palette, and the perception of how Director MX can help you make compelling, user-friendly applications. ⌒⌒

James Newton wrote many behaviors for Director's Library Palette. His book, Director Shockwave Studio Developer's Guide *(ISBN 0-07-213265-5), covers Imaging Lingo and 3D. OpenSpark Interactive Ltd develops bespoke applications for research and industrial clients, and also commercializes the OSControl xtra.*
james.newton@openspark.com
http://xtras.openspark.com/

How does the movie know to create a new instance of the Paint Line script, and how does it know what values to use to create that instance? The answer is in the Paint behavior.

As the user drags the mouse to paint a line, the mPaint() handler of the Paint behavior continuously checks the position of the mouse, and determines the smallest rectangle that encloses all the positions of the mouse since the drag started.

While the user is dragging the mouse, the line is painted directly into the image of the stage. This acts as a separate layer above the sprite that the user is painting on. When the user releases the mouse after painting the line, the bitmap still looks the way it did before the user started painting, and (the stage).image contains the "after" image. The mUpdateBitmap() handler copies the appropriate rect of the image of the stage into the appropriate rectangle of the image of the member.
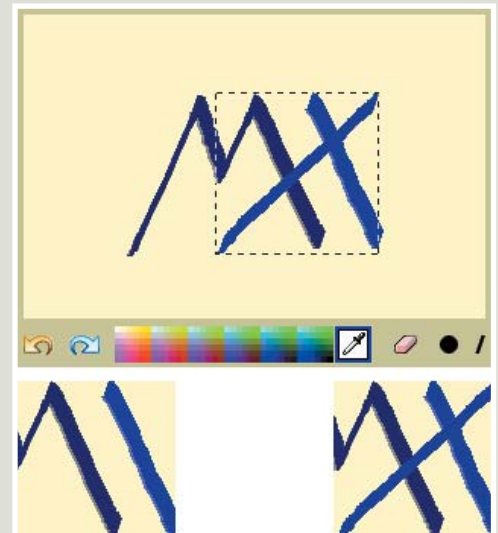
At this point, all the ingredients are prepared for the undo action: the bitmap member, the dirtied rectangle, and the before and after images. All that remains is to assemble the ingredients into a list and to tell the Undo Broker which script to use to treat the ingredients.

```
property pMember -- Bitmap member used for canvas
property pRect   -- rect of altered area
property pBefore -- image of altered rect before
--                  alteration
property pAfter  -- image of altered rect after alter-
ation


on new(me, aDataList) -------------------------------
  -- INPUT: <aDataList> must be a property list with the
  --           structure:
  --           [#member: <bitmap member>,
  --            #rect: <rect>,
  --            #before: <image>,
  --            #after: <image>]
  -- ACTION: Stores a pointer to the member whose image
is
  --           altered, along with the rect of the dirtied
  --           area and the image of that rect before and
  --           after the change
  -- OUTPUT: Returns a pointer to this instance
  -------------------------------------------------------

  pMember = aDataList.member
  pRect   = aDataList.rect
  pBefore = aDataList.before
  pAfter  = aDataList.after

  return me
end new
```

```
property pColor
on Paint_SetBrushColor(me, aColor)
 if pColor = aColor then
  -- The color has not changed
  exit
 end if

 pColor = aColor
 sprite(me.spriteNum).color = pColor
end Paint_SetBrushColor
```

```
on Paint_SetBrushColor(me, aColor, isUndo)
  if pColor = aColor then
    exit
  end if

  -- Add your lines here
  if not isUndo then
  tUndoData = [:]
  tUndoData[#before] = pColor
  tUndoData[#after] = aColor
```

```
    UndoAction("Select Color Undo", tUndoData)
  end if


 pColor = aColor
  sprite(me.spriteNum).color = pColor
end Paint_ShowBrushColor
```

```
property pBefore -- color before alteration
property pAfter  -- color after alteration


on new(me, aDataList) -------------------------------
  -- INPUT: <aDataList> must be a property list with the
  --           structure:
  --           [#before: <color object>,
  --            #after: <color object>]
  -- ACTION: stores the color before and after the color
  --         change
  -- OUTPUT: a pointer to this instance
  -------------------------------------------------------

  pBefore = aDataList.before
  pAfter = aDataList.after
  return me
end new



on undo(me) ------------------------------------------
  -- SENT BY the mDo() handler of the Undo Step instance
  --         that this instance is the ancestor of
  -- ACTION: restores the brush to its original color
  -------------------------------------------------------

  isUndo = TRUE
  sendAllSprites(#Paint_SetBrushColor, pBefore, isUndo)
end undo



on redo(me) ------------------------------------------
  -- SENT BY the mDo() handler of the Undo Step instance
  --         that this instance is the ancestor of
  -- ACTION: re-adopts the altered brush color
  -------------------------------------------------------

  isUndo = TRUE
  sendAllSprites(#Paint_SetBrushColor, pAfter, isUndo)
end redo
```

Organism

multimedia developer Adam Montandon, a researcher in digital futures at the Institute of Digital Arts Technology (see "Tangled in the Machine," page 54) designed this as an "organism chatroom" where virtual AI agents form an artificial language, and words live and die in an organic pattern. The project was a collaborative effort with Zoë Kennard.

You can find more on Adam's latest work at www.onceuponawebsite.com.

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.