

MX

developer's journal

THE LEADING MAGAZINE FOR
MACROMEDIA MX DEVELOPMENT,
DESIGN, & PRODUCTION TOOLS

volume 3 issue 12 2005 www.mxdj.com

DESIGNING WITH CSS

UNDERSTANDING



CSS DESIGN
CONCEPTS



FLASH
Using Bitmap
Caching in Flash



FIREWORKS
Programmer's Guide to
Fireworks Commands



COLDFUSION
Validation of Using
DOA, DG, and VO



CAPTIVATE
Using Captivate to
Retouch Images



FLEX
Using the Flex
Trace Panel



Presorted
Standard
US Postage
PAID
St. Croix Press



Rev Up Your Flash Video

Stay Ahead of the Competition With VitalStream and the Enhanced Video Features in Flash 8

With over two years of experience in delivering much of today's most popular media, VitalStream® is the first and most experienced Flash™ video streaming service provider.

Enhanced Flash 8 Video Features:

- New VP6 codec delivers higher quality video at the same bit rate
- 8-bit alpha channel transparency enables you to blend video with other elements
- Improved live video capabilities

VitalStream Complete Toolset for Flash:

- MediaConsole®
- MediaOps™ SDK
- Flash Authentication
- Reporting Dashboard



*Integrate Streaming Media
Into Your Flash Projects*

Take Advantage of the Enhanced Video Features in Macromedia Flash 8
Call (800) 254-7554 or Download Tutorials at www.vitalstream.com/go/mxdj



Call (800) 254-7554
Visit www.vitalstream.com





Xmas Mega Pack

**SO MANY FEATURES
WE HAD TO BRING 2 SANTAS**



459 Server side Features for Dreamweaver 8

Build any dynamic website you might think of with Xmas Mega Pack 2005. This special bundle includes 21 Dreamweaver extensions that help you easily develop and maintain dynamic applications saving hundreds of hours of work. In a nutshell - all your Christmas wishes are fulfilled. Xmas Mega Pack 2005 costs \$899 and this offer is available till December 31st. Visit www.interaktonline.com/xmas/ for detailed information.



MX File Upload

Upload files and images, create thumbnails



MX RSS Reader-Writer

Use RSS feeds in your sites



MX Query Builder

Visual SQL query builder for Dreamweaver



MX User Login

Professional user login for your sites



MX Coder Pack

Code completion for Dreamweaver's programmers



MX Looper

Nested Repeats and Horizontal Loopers in Dreamweaver



MX Form Validation

Form Validation made easy



MX CSS Dynamic Menus

Create CSS pull down menus from Dreamweaver Recordsets



MX Includes

Functional Server-Side Includes in Dreamweaver



MX Send E-mail

Send e-mails from web forms



MX Dynamic Charts

Create pie, bar and line charts from Dreamweaver Recordsets



NeXTensio

Dreamweaver Extension for dynamic lists and forms



MX Kart

Shopping cart extension for Dreamweaver



MX Site Search

Search database sites



MX Widgets

Form controls for validation and enhanced usability



KTML

Online Visual HTML editor



MX CSV Import-Export

Easy CSV Import Export



XML Import-Export

Easy Data transfer with XML



MX Shop

Customizable e-commerce web application for Dreamweaver



MX Calendar

Build web calendars from Dreamweaver



MX Table Sorter

Sort your dynamic tables

work smart

www.interaktonline.com/xmas

Interakt



7 Designing with CSS

*Understanding CSS
Design Concepts*
by **adrian senior**



-  **18** Flash Animation Learning Guide
Part 2
by **jen dehaan & chris georgenes**

-  **28** Using Bitmap Caching in Flash
Developers and designers use Flash to do a lot more than just animation
by **guy watson**

-  **34** Flash Is Your Friend in Web 2.0
High order bit
by **kevin lynch**

-  **36** Programmer's Guide to Fireworks Commands
Getting started with your first command
by **dustin dupree**

-  **40** Validation of Using DOA, DG, and VO
ColdFusion development
by **scott barnes**

-  **42** Using Captivate to Retouch Images
Recorded demos and simulations
by **mark fletcher**

-  **48** Using the Flex Trace Panel
Developing Flex applications
by **dirk eismann**

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

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

Surround SCM lets you...

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

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

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

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

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

Remotely access your source code repository from Dreamweaver MX.

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

Features:

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

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

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

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

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

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

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

Successful Web Project Management

Delivering Web sites on time and within budget may seem like an impossible task. Download Seapine's **Beyond the Wild, Wild West: Successfully Managing Web Development** white paper, and learn how to easily and cost effectively manage your next Web development project using version control, issue management, and automated testing tools.



www.seapine.com/mxww



macromedia
ALLIANCE PARTNER



Intermedia.NET...

Now serving the hottest ColdFusion.

At Intermedia.NET we go beyond the industry standard by supporting the hottest new ColdFusion software, offering power like never before. For nearly a decade, we've been providing reliable, secure hosting to thousands of companies across the globe. We can do the same for you.

Intermedia.NET's premier hosting services include:

- ColdFusion MX hosting
- Custom tag registration
- Competitive plans
- Verity collections search engine
- Security sandboxes
- Guaranteed service levels

Unprecedented power, unmatched reputation...
Intermedia.NET is your hosting solution.

Call us at: 1.888.379.7729

e-mail us at: sales@intermedia.NET

Visit us at: www.intermedia.NET



INTERMEDIA.NET

Group Publisher Jeremy Geelan
Art Director Louis F. Cuffari

EDITORIAL BOARD
Dreamweaver Editor
 Dave McFarland
Flash Editor
 Brian Eubanks
Fireworks Editor
 Joyce J. Evans
FreeHand Editor
 Ron Rockwell
 Louis F. Cuffari
Director Editor
 Andrew Phelps
Captivate Editor
 Tom Green

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

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

Associate Editor
 Seta Papazian, 201 802-3052
 seta@sys-con.com

Technical Editors
 Jesse Warden • Sarge Sargent

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

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

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

Worldwide Newsstand Distribution
 Curtis Circulation Company, New Milford, NJ

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

Promotional Reprints
 Dorothy Gil, 201 802-3024
dorothy@sys-con.com

Copyright © 2005
 by SYS-CON Publications, Inc. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy or any information storage and retrieval system, without written permission.

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

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

MX
 developer's journal

Designing with CSS

Understanding CSS Design Concepts
 by adrian senior

This article has been updated for Dreamweaver 8. If you are still using Dreamweaver MX 2004, please read the version of this article series for Dreamweaver MX 2004. The CSS features in Dreamweaver have been vastly improved in Dreamweaver 8. You can learn about those changes in Julie Hallstrom's article, "An Overview of CSS in Dreamweaver 8."

This article is the first in a series of tutorials about Cascading Style Sheets (CSS). The aim in Part 1 is to familiarize you with some of the basics of CSS. I'll review some of the problems you may run into. I'll also cover the shorthand and longhand versions of the CSS syntax.

I would like to make it clear that I do not advocate you dropping the use of tables as design elements. It is a case of what suits you best. Whatever you feel comfortable with is a good way to go. What this series of tutorials will do is provide you with the ability to create CSS layouts using Dreamweaver 8. Once you have some of the basics down, you will move on to Part 3 to create a simple but effective CSS layout.

All exercises you undertake in this series are available as downloads, so you can work alongside, or just review the code as you read through. I will keep the same structure as you work throughout the series so you can keep everything structured locally for easy reference if required.

Creating a Cascading Style Sheet

First, I'd just like to touch on the outmoded use of font tags. By default, Dreamweaver 8 uses styles rather than

font tags to redefine the appearance of your text content. This is a good move in the right direction that it is tantamount to sacrilege to make the changes to revert back to font tags. Let me explain why – you can see the argument in progress in a little while.

To begin with, font tags are deprecated, which means they may not be supported by browsers in the future. They are no longer a part of the XHTML specification. That is a good enough reason not to use them.

The real problem with font tags begins when your design client requests changes. Maybe the font color needs to be changed or perhaps the font face needs to change from Verdana to Arial. If the content has been generated using font tags, making these changes can take a huge amount of time. By contrast, you can make changes such as these—and indeed far more complicated ones—in a very short time by editing an external style sheet because it applies the changes on a sitewide basis instantly. You also get easier document manipulation and lighter, faster-loading pages. The benefits are many.

Taking the time to learn CSS, even if you wish to use it only for redefining elements rather than full-fledged page layout, is well worth enduring the small learning curve it takes to get you there. Dreamweaver 8 has made giant strides in its implementation of CSS. Take advantage of these changes. I guarantee you won't be sorry!

How Do You Use CSS?

Let's start at the beginning of the CSS trail and look at the methods available to



you when you want to apply a Cascading Style Sheet to your documents. You can use an external style sheet, an embedded style sheet, or inline styles. External style sheets are best because they give you the most control over styles.

Using an External Style Sheet

By using an external (linked) style sheet, you exercise global control over the appearance of every page in your site that is linked to that CSS file. This is a very powerful method of using CSS. By using a linked style sheet, you have the ability to make sitewide changes on countless numbers of pages from a single location. This can be completed in what may amount to no more than a few seconds' work. Powerful indeed!

Dreamweaver makes setting up an external style sheet just about as easy as it can be. In the first exercise, I'll review the procedure of creating an external style sheet and how to link it to your web pages:

- 1 Select File > New or press Control+N to open the New Document dialog box (see Figure 1)
- 2 In the Category column, select Basic Page
- 3 In the Basic Page column, select CSS
- 4 Click the Create button

Your newly created Cascading Style Sheet will open in Dreamweaver. If you

are completely unfamiliar with CSS, you will notice that style sheets don't have a Design view. Your style sheet is little more than a text file that contains your CSS rules and their properties and values.

Alternatively, if you select the CSS Style Sheets option in the Category column of the New Document dialog box, the Basic Page column will show a list of "starter" style sheets that already contain some of the CSS rules you may use. You won't be using these starter pages at the moment. Instead, you will be building your own style sheet, and I will discuss why you are setting the properties and values you will use.

Now that you have created your first CSS file, use the following steps to save it in a defined site:

- 1 If you haven't already defined a site, do so now. If you need help defining a site, see "How to Define a Site in Dreamweaver" (TechNote 14028)
- 2 Create a new folder called CsstFiles in the root of your site
- 3 Save the CSS file in the new CsstFiles folder and name it external.css

Note: Normally, I suggest saving CSS files in their own folder just for the sake of good organization. This keeps your site neat and tidy as it grows.

In this section, you will create two pages that will demonstrate the power of using an external style sheet:

- 1 Open the New Document dialog box (File > New)
- 2 Select Basic Page from the Category column
- 3 Select HTML from the Basic Page column
- 4 (Optional step) Click the Make Document XHTML Compliant option
- 5 Click the Create button
- 6 Save the page to your site root and name it external.html
- 7 Repeat Steps 1–5 to create a second page
- 8 Save the second page to your site root and name it external2.html

You now have all the documents you need to complete the first section of this tutorial.

Linking the Cascading Style Sheet

You should now have Dreamweaver open and have the following three documents displayed, saved, and ready to work on:

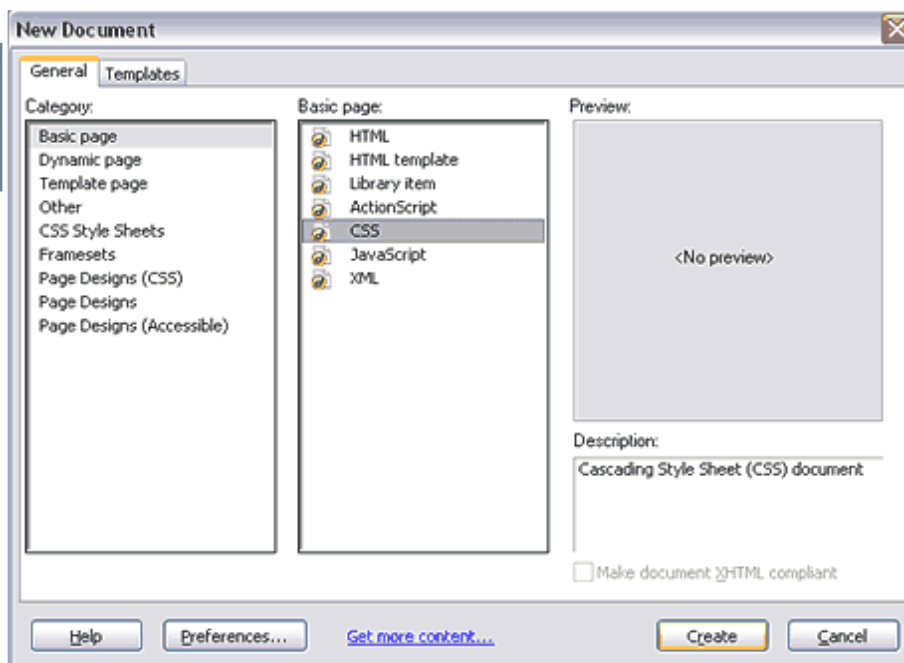
- external.css
- external.html
- external2.html

If you don't have these files open in Dreamweaver, open them now.

Here are the steps to the process of linking your style sheet to your HTML documents:

- 1 Open external.html, external2.html, and external.css in Dreamweaver 8.
- 2 Press Shift+F11 to open the CSS Styles panel.
- 3 Make sure external.html is the active document, and click the Attach Style Sheet button in the lower right corner of the CSS Styles panel. The Attach External Style Sheet dialog box opens. Dreamweaver remembers your last selection.
- 4 Select the Link option.
- 5 Click the Browse button.
- 6 The Select Style Sheet File dialog opens.
- 7 Select Document from the Relative To pop-up menu.
- 8 Select the external.css file you created earlier. The path is inserted in the File/URL field.
- 9 Click OK to add your style sheet name to the CSS Styles panel.
- 10 Click the Code view button. You will

figure 1



see that Dreamweaver has inserted the necessary code linking your document to your style sheet.

- 11 Select File > Save to save the file.
- 12 Repeat the process to link external2.html to your style sheet.

Before you begin to add rules into your style sheet, there are one or two things you should cover first. In the next section, you will learn why it is important to use a complete doc type. I will then give a quick introduction to margins and padding before discussing browser default settings and why it is a good practice to zero off these defaults so you start on a level playing field.

Margins, Padding, and Doc Types

Dreamweaver 8 (and MX 2004) adds a complete doc type to every page you create. This was another big step in the right direction from Macromedia because an incomplete doc type can cause many problems with your CSS.

A complete doc type is shown below. In this case, the doc type is a Transitional XHTML doc type:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

An incomplete doc type will cause your browser to drop into what is called "quirks mode." This will very quickly cause you to lose your hair and any sanity you may have if you are aiming for a pixel-perfect design. Ensure that your doc type is a complete doc type at all times. If you are using a version of Dreamweaver prior to MX 2004, you may want to view the available doc types you can use at the World Wide Web Consortium (W3C) (www.w3.org/QA/2002/04/valid-dtd-list.html).

You should also be aware that if you have anything above the doc type in Internet Explorer – even a comment – the browser will revert to quirks mode. This, of course, does not include the use of any server-side code you may have above the opening XHTML tag. Server-side code is, of course, processed on the server and

not returned to the browser in the manner I am talking about here.

In Figure 2, you are looking at Internet Explorer 6 (IE6). When this browser – and earlier versions – drops into quirks mode, it has a unique problem in that it gets the CSS box model completely wrong. It includes the padding within the defined width rather than adding the padding to the width.

You can clearly see the results in Figure 2. While this may not mean too much to you at the moment, you will be able to see the importance of ensuring your pages are rendered in standards mode. The small box in Figure 2 shows IE6 in quirks mode. In this instance, quirks mode was triggered by placing a simple HTML comment above the doc type.

What Are Margins and Padding?

Margins and padding exist within many of the XHTML elements you will use within your design work. Figure 3 represents a <p> element. The text is clearly visible and the red area surrounding it indicates the presence of a padding value. The black border around the red area indicates the boundary of the <p> element. Increasing or decreasing the padding value causes the red area to expand or contract to suit the value you provide in your CSS rule for the <p> element.

The blue area represents the margin value. The margin value pushes away other elements surrounding the <p> element as long as they are in the document flow. (I will talk more about the document flow later in this series.) Increasing and decreasing the margin value determines how closely the elements surrounding the <p> element are allowed to encroach upon it.

So to recap: Padding resides within an element and margins reside outside the element.

The Syntax of CSS

CSS is a simple language. It is easy to read and takes very little time to grab the basics and start building your own style sheets. After you complete this section of the tutorial, you will have a good understanding of the syntax and various elements that exist within CSS rules. I begin by creating a simple rule that clearly lays

SYS-CON MEDIA
President & CEO
Fuat Kircaali, 201 802-3001
fuat@sys-con.com
Group Publisher
Jeremy Geelan, 201 802-3040
jeremy@sys-con.com

ADVERTISING
Senior Vice President, Sales & Marketing
Carmen Gonzalez, 201 802-3021
carmen@sys-con.com
Vice President, Sales & Marketing
Miles Silverman, 201 802-3029
miles@sys-con.com
Advertising Sales Director
Robyn Forma, 201 802-3022
robyn@sys-con.com
Advertising Sales & Marketing Manager
Dennis Leavey, 201 802-3023
dennis@sys-con.com
Advertising Sales Manager
Megan Mussa, 201 802-3023
megan@sys-con.com
Associate Sales Managers
Kerry Mealia, 201 802-3026
kerry@sys-con.com

PRODUCTION
Production Consultant
Jim Morgan, 201 802-3033
jim@sys-con.com
Lead Designer
Louis F. Cuffari, 201 802-3035
louis@sys-con.com
Art Director
Alex Botero, 201 802-3031
alex@sys-con.com
Associate Art Director
Tami Beatty, 201 802-3038
tami@sys-con.com
Assistant Art Directors
Andrea Boden, 201 802-3034
andrea@sys-con.com
Abraham Addo, 201 802-3037
abraham@sys-con.com
Video Production
Frank Moricco, 201 802-3036
frank@sys-con.com

SYS-CON.COM
Consultant, Information Systems
Robert Diamond, 201 802-3051
robert@sys-con.com
Web Designers
Stephen Kilmurray, 201 802-3053
stephen@sys-con.com

ACCOUNTING
Financial Analyst
Joan LaRose, 201 802-3081
joan@sys-con.com
Accounts Payable
Betty White, 201 802-3002
betty@sys-con.com
Accounts Receivable
Gail Naples, 201 802-3062
gailn@sys-con.com

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





out the structure for you to see:

```
selector{
property: value;
}
```

Selector

The first part of the rule is called a selector. The selector represents a structure that can be used as a condition to define how an element looks in the browser.

Property

The property section of the CSS rule defines a specific area of the structure, such as padding or margin properties.

Value

The value section of the CSS rule defines a measurement – in general terms. If you are defining how a <p> element may look, the property might be a reference to the <p> element's padding and the value might be 10 pixels:

```
p {
padding: 10px;
}
```

A CSS rule may, of course, contain many property and value pairs. Each would perform a specific task within the structure to provide you with the look and feel you want for the rule.

Property and value pairs are separated by a colon (:). The space after the colon is optional. Immediately following the value, you have a semicolon (;). The semicolon is said to be optional after the last property/value pair in your CSS rule. However, I prefer to leave it in place.

That is really all there is to CSS syntax; it is as simple as that. While it is always good to know your code, it is also good to know that Dreamweaver writes the CSS rules for you, as you shall see shortly.

Longhand or Shorthand?

When you write your CSS rules, you have an option of two different writing styles: longhand and shorthand. Dreamweaver allows you to set your own preference for the writing style. Here are the steps to access the Dreamweaver CSS preferences:

Select Edit > Preferences, or use the Control+U shortcut. In the Preferences dialog box, select the CSS Styles option in the Category list (see Figure 4).

By selecting all the Use Shorthand For options, as well as the According to Settings Above option, you tell Dreamweaver to write your CSS rules in shorthand. That's fine, but what is the difference between longhand and shorthand writing styles?

The obvious answer is that shorthand creates a smaller file. Shorthand allows you to condense your rules in such a way that you can assign multiple properties and values in a single line. More importantly, you can write your rules and modify them faster – less work all around.

To show you how this is achieved, I will write the same CSS rule in longhand and then again in shorthand. The difference will be obvious. Once you are familiar with shorthand, I will show you how it can be condensed further to reduce the amount of required information.

Let's begin by creating a CSS rule that redefines the margins of the body element, first in longhand and then in shorthand. Go to your external.css file and type the following longhand code to set the body margins:

```
body{
margin-top: 0;
margin-right: 10px;
margin-bottom: 0;
margin-left: 10px;
}
```

Right under that code, type the following shorthand version of the same code:

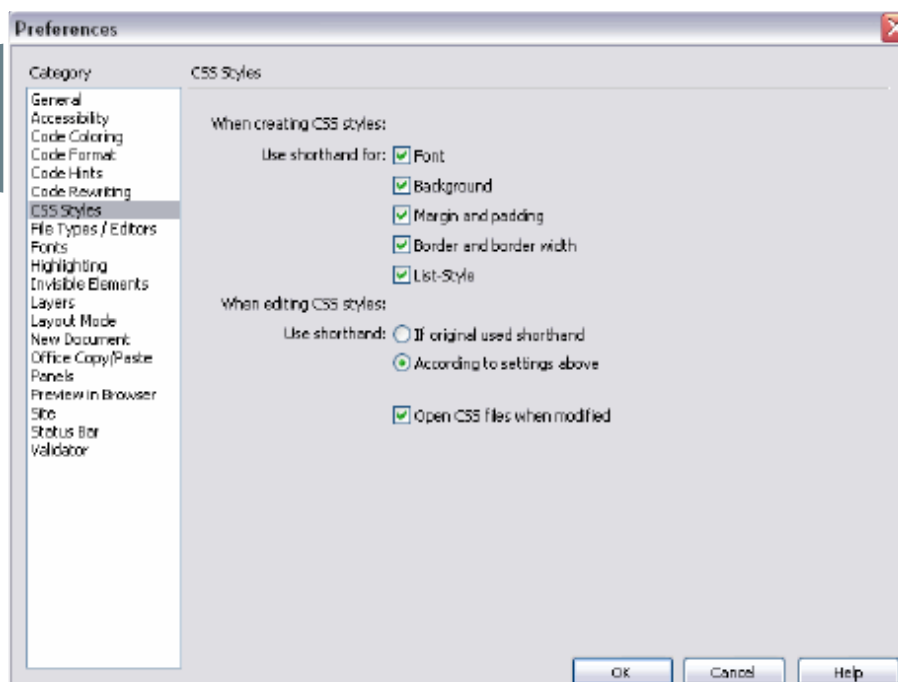
```
body{
margin: 0 10px 0 10px;
}
```

There's quite a bit of difference here. Using the shorthand method, you eliminate the need to describe each side of the body you want to affect in the property side of the rule. By simply declaring "margin," you avoid having to declare each margin property within your CSS rule. You also are able to apply the values in a single line of code. This reduces the CSS property from a four-line affair to a single line.

Delete both the longhand and shorthand code you just wrote.

How does the browser know which value is for which side of the body element? Simple, the browser takes the order of the values into consideration and then applies them accordingly. Reading from left to right, the values are applied to the top, right, bottom, and left margins. In this scenario you would have a 0 margin to the top and bottom of your page, while the left and right margins

figure 2





STREAM57



Customized Flash-based media solutions

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

visit us
stream57.com

e-mail us
streamline@stream57.com

call us
212.909.2550 x1012



would have a value of 10 pixels.

You can further reduce the shorthand rule as follows:

```
body{
margin: 0 10px;
}
```

When the browser comes across a rule like this one, it knows that the values are pairs of values. The first value is taken and applied to the top and bottom margins while the second value – 10px in this instance – is applied to the left and right margins.

If all the margins of the body are the same value, you set the value only once:

```
body{
margin: 0;
}
```

In the rule shown above, a value of 0 is applied to all four sides of the body element. Zeroing the margins on the body element is a good practice, at least initially. Browsers tend to apply default margins on the body (and other elements) if they are not explicitly stated. Explicitly zeroing the defaults and setting the margins you require is the best way to guarantee that browsers apply the margins you want. Leaving the defaults can result in unexpected displacements within your page because default margins vary greatly from browser to browser.

The Opera browser is a special case because it also applies a default padding setting on the body element. This means you need to zero off the padding on the

body as well as the margins to get a consistent cross-browser starting point. To achieve this, your final body rule would look like the one below:

```
body{
margin: 0;
padding: 0;
}
```

The above rule provides you with a cross-browser zeroed body element – a good base from which to start when laying out pages.

Now that you understand about zeroing your page margins and the importance of a full doc type, return to your external.css file and add the body rule to your style sheet using the Dreamweaver user interface. Here are the steps in the process:

- 1 Click the New CSS Style button in the CSS Styles panel. The New CSS Style dialog box opens.
- 2 Select the Tag Selector Type option.
- 3 Select Body from the Tag pop-up menu.
- 4 Select the body element.
- 5 Ensure that Dreamweaver has selected the correct style sheet in the Define In pop-up menu.
- 6 Click OK to open the CSS Style Definition dialog box.
- 7 Select the Box category option.
- 8 Ensure that the Same for All options are selected for both Padding and Margin.
- 9 Enter 0 in the Padding field.
- 10 Enter 0 to the Margin field.
- 11 Click the OK button.

- 12 Look at the code that Dreamweaver enters in the external.css file. Although Dreamweaver has added a value for the measurement (px), this is not necessary for a zero value.
- 13 Select the body rule and delete it. You will next use the Dreamweaver code hints to write your CSS body rule.
- 14 Type `body {`. As you begin typing, the code hints appear.
- 15 Select Margin from the code hint list.
- 16 Enter a value of 0 after the margin property: `margin: value`.
- 17 Press Enter to go to the next line of code and type `p`.
- 18 Select Padding from the code hint list.
- 19 Enter a value of 0 after the padding property: `padding: value`.
- 20 Select File > Save to save the file.

It is possible to reduce the amount of characters in a color value within your style sheet rules. If a color is defined in three pairs (for instance, red is #FF0000), that could be written as #F00, where each character represents a matching pair. The F represents FF and the single zero in the second and third positions of the value indicate 00 and 00. This technique can only be applied if the color value contains three pairs. Other examples of this would be #FFFFFF (white) written as #FFF and #000000 (black) written as #000.

Alternatively, each color value could be set using the color's name:

- color: red;
- color: black;
- color: white;

The syntax of CSS is a flexible one. In many cases, it allows you to use different syntax to achieve the same results.

In the next section, we will look at how you can lay out your web pages correctly by using semantic markup.

Semantic Markup

“The Semantic Web approach develops languages for expressing information in a machine processable form” —(W3C) The line above is a quote, so it resides in a pair of `<blockquote>` tags. This tag lets a screen reader know that the text is a quote and not just another paragraph. Screen readers are machines that enable visually impaired people to surf the web. Creating a semantically correct document in its simplest form is just one case of

figure 3

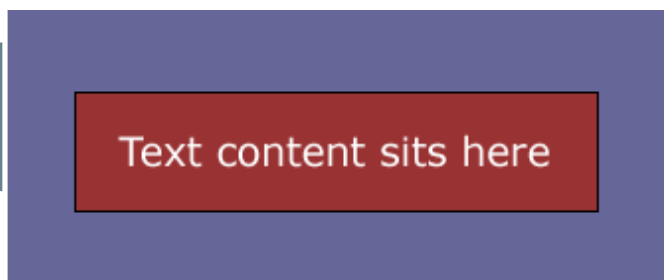


figure 4

```
<body>
<h1>Use h1 elements for your main title</h1>
<p>We should use p tags for creating paragraph text</p>
<h2>Use the correct element for sub headings</h2>
<p>and the content should still be inside p tags</p>
</body>
```

using the XHTML elements supplied with the specification to lay out your document correctly.

Each element provided in the XHTML specification has been designed to be used in a specific way and each has a specific meaning. An h element is a title and any text between title tags is instantly recognized as a title by a machine reader, just as text within <p> tags is recognized as being a paragraph. Not all tags are semantic in their makeup. Take the tag, for instance. A tag has no meaning associated with it; it is simply a container element. If you were using a tag and styled it to emphasize text within a paragraph, you would be better off using the tag. This is what it was designed to do. It says to the machine reader this text should be emphasized; in turn, the machine reader emphasizes the text within the tags.

Semantic Improvements in Dreamweaver 8

This is another area in which Dreamweaver progressed greatly. In earlier versions of Dreamweaver (prior to MX 2004), pressing Control+I or clicking the I button in the Property inspector inserted a pair of <i> tags to italicize the text. While this looked fine to the human eye, the <i> tag means absolutely nothing to a machine reader. Dreamweaver 8 (and MX 2004) now inserts tags when you press Control+I or click the I button in the Property inspector. A machine reader understands that the tag means to emphasize. In much the same way, the Control+B shortcut (or B button in the Property inspector) now inserts tags rather than tags. The resetting of these keyboard shortcuts in Dreamweaver MX 2004 was a move in the right direction for creating semantically correct documents.

Hierarchical Structure

Creating text in a p element and then using font tags – or CSS for that matter – to increase the text size so it looks like a heading and then applying a bold tag to it does not make it a header. The finished result might look like a header to the human eye. However, that's where the association of the text with a header ends.

If you want to create a header, you should use one of the header elements. The header element you use depends on where you are in your document. The main heading on your page would be an h1. Subheadings would be h2, and so on. You have no need to use all the headings within your document but they are available if you require them:

- h1
- h2
- h3
- h4
- h5
- h6

Each header element is hierarchical in nature, with h1 being the most important header and h6 being the least important header. All of them have a natural place within your documents. Commonly you will see designers typing text directly into a <td> tag set within a table – not good. This practice gives machine readers no indication at all as to how to read the text. If you are dealing with copy text below a header, then you should probably set it within a p element. Once the text is set within the <p> tags, machine readers know what type of content they are looking at and how it should be read. The idea is to set your document out in a hierarchical manner that is easily read not only by humans but by machine readers (see Figure 5).

I can hear you saying, "OK, that sounds fine but h1 elements are huge! I understand how making semantically correct documents is a good thing, but I also want my documents to look good."

Enter CSS

CSS provides you with the ability to redefine elements to take on the appearance you specify. This is cool because you can set the size of h1 to whatever you want. You can set the font face, margins, padding, color, background color, and a whole list of other properties all from within a single CSS rule. This keeps your document semantically correct.

Not only can you redefine the h1 element, but once its rule is safely in your style sheet and the pages in your website are linked to that style sheet, you can change its appearance globally from a single location in a matter of seconds. The demo below walks you through add-

ing an h1 element and demonstrates the ease with which you can control your documents from an external style sheet.

Here are the steps in the process:

- 1 Click the New CSS style button in the CSS Styles panel. The New CSS Style dialog box opens.
- 2 Ensure that the Tag Selector Type option is selected.
- 3 Select h1 from the Tag pop-up menu.
- 4 Select external.css from the Define In pop-up menu.
- 5 Click OK.
- 6 Select "Georgia, Times New Roman, Times, Serif" from the Font pop-up menu.
- 7 Enter 100 in the Size text box and select % from the unit of measure pop-up menu.
- 8 Enter #003366 in the Color text box.
- 9 Select the Box category.
- 10 Click the Same for All option in both the Padding and Margin sections.
- 11 Enter a value of 0 for both the Margin and Padding.
- 12 Click OK.
- 13 Open external.css. You can see your newly added h1 rule.
- 14 Select File > Save to save the external.css file.

As you can see in the CSS Styles panel, your h1 rule is now shown with the body element in the external.css tree.

Zoe Gillenwater wrote an excellent article for Community MX that goes into more detail on semantic markup (www.communitymx.com/abstract.cfm?cid=A1A37). It also includes information on how semantic markup can help your search engine positioning.

ID, Class, and Descendant Selectors

In the previous section I reviewed the parts of a CSS rule. I now take you through some of the more commonly used selectors and look at how they are used.

The ID Selector

I begin by defining a div with an ID selector. The div you define here is a container or wrapper div. Using a wrapper div is a common practice in CSS positioning, and we will look deeper into how they work later in Part 3. The ID selector is preceded by the pound sign (#) within



your style sheet (see Listing 1). The margins you set should be somewhat familiar to you from the way you set up your body margins earlier. I am using a pairs value to declare the settings. As you can see, the top and bottom will be zero.

Listing 1: Wrapper ID selector

```
#wrapper{
width: 770px;
margin: 0 auto;
}
```

The auto value may be new to you, so let me explain. When you progress to laying out your first CSS positioning document in Part 3, you will create a fixed-width layout. The width of your container or wrapper div will be 770 pixels to ensure that you have no horizontal scrolling for your viewers who have set their browsers to an 800-pixel-wide resolution. The margin value of auto is applied to the left and right sides of the wrapper div. This means that regardless of the users' browser window width, the page content is always centered horizontally. The space on either side of the fixed-width wrapper div is distributed evenly by the auto pairs value in your style sheet. You will see how these values are set in the Creating an ID Selector demo at the end of this section.

The pound sign (#) in your style sheet indicates that your wrapper div is an ID selector. However, it is not used when you reference the ID selector from within your XHTML code. If you were inserting the wrapper div into your XHTML document, you would reference it as shown in Listing 2.

Listing 2: ID referenced in HTML, as opposed to how it is written in the CSS file

```
<div id="wrapper">
Our wrapper div content
</div>
```

As you can see, you reference the wrapper by using `id="wrapper"`. The properties and values you set for your wrapper div within your style sheet are now

applied to the wrapper div in your XHTML code.

Caution: You can only use an ID selector name once in your XHTML document. For example, if you defined an ID selector in your style sheet and then reused that selector name on an image swap behavior within the same document, you will get some very odd behavior.

Here are the steps in the process:

- 1 Click the New CSS Style button in the CSS Styles panel. The New CSS Style dialog box opens.
- 2 For Selector Type, click the Advanced option.
- 3 Name the ID selector #wrapper. When you create an ID selector, you type the name into the Selector field. An ID selector is always preceded by the pound (#) sign.
- 4 Ensure that you have external.css selected in the Define In pop-up menu.
- 5 Click OK.
- 6 Select the Box category.
- 7 Deselect the Same for All option in the Margin section.
- 8 In the Top text box, enter 0.
- 9 In the Right text box, enter auto.
- 10 In the Bottom text box, enter 0.
- 11 In the Left text box, enter auto.
- 12 Select the Border category.
- 13 Deselect the Same For All option in the Style area. You are going to add a 1 pixel solid black border to the left and right sides of your #wrapper ID.
- 14 Select Solid from the Right and Left pop-up menus for the border style.
- 15 Deselect the Same For All option in the Width section.
- 16 Enter 1 in the Right and Left text boxes for the border width.
- 17 Deselect the Same For All option in the Color section.
- 18 Enter #000000 in the Right and Left text boxes for the border color.
- 19 Select the Positioning category.
- 20 Select Relative from the Type pop-up menu.
- 21 Click OK.
- 22 Open the external.css file. You will now see the newly created #wrapper rule.

23 Select the File > Save to save the external.css file.

Your #wrapper selector is now in the CSS Styles panel.

The Class Selector

Unlike the ID selector, the class selector can be used as often as you need within your XHTML document. The class selector is preceded by a period (.) within your style sheet (see Listing 3).

Listing 3: Class selector

```
.leftimage {
margin-right: 15px;
margin-bottom: 5px;
}
```

The period (.) in the style sheet indicates that the rule is a class selector. However, the period is not used when you reference the class selector from within your XHTML code. For example, if you were inserting the leftimage class into your XHTML document, you would reference it as shown in Listing 4.

Listing 4: Referencing a class selector within XHTML

```

```

The final outcome is the same whether you use a class selector or an ID selector. Because you will need to apply the settings you have declared within your rule on many occasions, and often more than once in the same document, you really need this rule to be a class selector instead of an ID selector.

Here are the steps in the process:

- 1 Click the New CSS style button in the CSS Styles panel. The New CSS Style dialog opens.
- 2 For Selector Type, click the Class option.
- 3 Name your class selector .leftimage. A class name is always preceded by a period.

“You can only use an ID selector name once in your XHTML document”

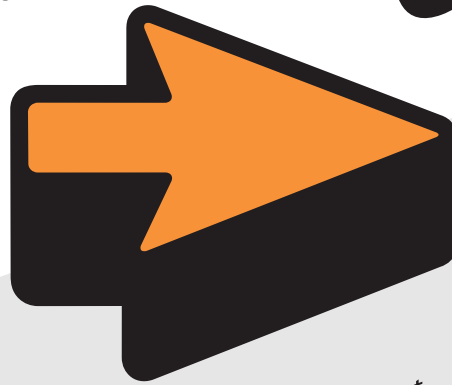
FLASHFORWARD

CONFERENCE & FILM FESTIVAL

Washington State Convention & Trade Center

SEATTLE
Feb 27 - Mar 2

106



Join the 15th
Flashforward
Conference &
Film Festival

- Keep your skills current
- Learn from experts
- Be inspired by the Film Festival
- Network at the job fair
- See the latest products
- Attend industry parties
- See Adobe/Macromedia keynote
- Visit an awesome new city
- Live, breath, eat, sleep Flash® for four days!

Confirmed Topics:

Workshops covering ActionScript, Flash Video, Flex 2, and Flash for Educators.
Sessions on Flash 8 and Animation, Games, Portfolios, ActionScript 3, Flex 2, Open Source, Mobile Devices, Audio, Art, Creative Commons, User Interfaces, Accessibility, Installations, Education, Security, JSAPI, Fuse, Physical Computing, Video Compositing, Color, Testing, and Effects.

Other sessions cover related technologies such as After Effects, Ajax, Breeze, Captivate, ColdFusion, FlashBlock, Flex, MySQL, PHP, and Photoshop.
As always we will deconstruct the best Flash websites, showcase inspirational designers, and more.

For more information, visit :
flashforwardconference.com

Save \$50

with this special discount code:

FFWDVMX6

For group discounts
(over four people)
and educational pricing,
please contact:

registration@flashforwardconference.com

or call

877-435-2744

toll free

(001-410-386-0646 outside the US)
during East coast business hours

Produced by:



lynda.com

Platinum Sponsor:



macromedia

Silver Sponsor:



Bookstore Sponsor:



Peachpit Press

Bronze Sponsor:



introNetworks

Media Sponsors:



MX
developer's journal

creativepro
.com



“The CSS rules you have created in this tutorial will be put to good use as you work your way through this series”

- 4 Ensure that the external.css style sheet is selected.
- 5 Click OK.
- 6 Select the Box category.
- 7 Deselect the Same For All option in the Margin section.
- 8 Give the Right margin a value of 15.
- 9 Give the Bottom margin a value of 5.
- 10 Click OK.
- 11 Open the external.css file and view your .leftimage class rule.
- 12 Select File > Save to save the external.css file.

The .leftimage class appears in the external.css tree in the CSS Styles panel.

Pattern Matching and the Descendant Selector

The descendant selector is a powerful way to target specific elements within specific areas of your XHTML documents. Using a descendant selector, you could, for instance, easily target any em elements that reside within p elements. There is no need to create a separate class and then apply it to the tags. You can simply target them from the style sheet. There is a space between each element within the descendant selector (see Listing 5). A descendant selector works in much the same way as the forward slash (/) does in a folder hierarchy within a URL.

Listing 5: Descendant selector applying to em elements within p elements

```
p em{
color: #990000;
}
```

The rule shown in Listing 5 says, Find p elements and if those p elements contain em elements, change the text color to #990000.

For any em elements that exist outside a p element, or within a different element, the pattern is broken and the text color change is not applied. The descendant selector allows you to pattern-match your XHTML documents and to be very specific as to how you affect elements with any given rule.

Here are the steps in the process:

- 1 Click the New CSS style button in the CSS Styles panel. The New CSS Style dialog box opens.
- 2 For Selector Type, click the Advanced option.
- 3 In the Selector text box, enter p em. Notice the space between the two elements. This is important; this space is known as a descendant combinator.
- 4 Ensure that the external.css style sheet is selected.
- 5 Click OK.
- 6 Select the Type category in the CSS Style Definition dialog box that opens.
- 7 Enter #990000 in the Color text box.
- 8 Click OK. Your newly created descendant selector appears in the external.css file.
- 9 Select File > Save to save the external.css file.

The descendant selector has been added to the external.css tree in the CSS Styles panel.

Redefining a Selector

Review the Defining the h1 Type Selector demo and then create a new rule in your external.css file. Redefine the p element and give one property and value pair. Set the property to font-size and set the value to 80% (see Listing 6).

Listing 6: Redefined p rule

```
p{
font-size: 80%;
}
```

Conclusion

If you were totally new to CSS at the beginning of this tutorial, you have come far. You have seen how you can use Dreamweaver MX 2004 to create an external style sheet and link that style sheet to documents within your website. You have learned about selectors and how to create them.

Specifically, you have looked at:

- Redefining elements with the Type selector
- Creating ID selectors

- Creating class selectors
- Creating descendant selectors
- Zeroing off your body element

In Part 2 you will explore CSS further and look at how your document interacts with the relevant CSS Styles panel to make editing CSS a breeze in the Design view. You will look at avoiding a common pitfall or two and lay the foundations further to create your first CSS positioning layout, which I cover in Part 3.

The CSS rules you have created in this tutorial will be put to good use as you work your way through this series. I hope that you now have a good understanding of how to create CSS rules using the Dreamweaver design panels.

Part 2 of this article can be found at www.macromedia.com/devnet/dreamweaver/articles/css_concepts_pt2.html and Part 3 is at www.macromedia.com/devnet/dreamweaver/articles/css_concepts_pt3.html.

Adrian Senior owns the UK-based web design agency Webade, which has been in business since 1998. He is also a member of Team Macromedia and a partner at Community MX. The year 2004 saw Adrian's first trip to America, where he visited Orlando and delivered two sessions at the TODCon conference. Before becoming involved with website design and development, Adrian's working life centered around the shipyards of the River Mersey, and oil production units in the North Sea. Adrian also provides training courses for companies who need to train their designers how to build compliant, accessible web sites using CSS and xhtml. He's been married to his wife, Janette, for 24 years and has two children, Antony and Eleanor.

This article originally appeared in the Macromedia Developer Center, www.macromedia.com/devnet/dreamweaver/articles/css_concepts_mx2004.html.

CFDynamics

Dedicated Server Packages Starting at \$189/mo.

All dedicated servers include:

- ▶ FREE STATS SOFTWARE!
- ▶ No long term commitments!
- ▶ FREE SQL server access!
- ▶ FREE MAIL SOFTWARE!
- ▶ Fair and simple pricing!
- ▶ Optional server maintenance!

As one of the premier ColdFusion hosting community leaders, CFDynamics is constantly looking for ways to provide better service to ensure your satisfaction. Not only do we offer the finest in shared hosting plans, but we now offer the finest in 100% dedicated server plans! Now you can afford the freedom of having your own dedicated server!

When your needs have outgrown shared hosting look to CFDynamics for **total freedom**. With dedicated server packages they're not offering an oxymoron; "virtually private" or "virtually dedicated" is NEITHER private nor dedicated. CFDynamics offers a solution that is **100% completely dedicated**. They don't play games with the fake stuff; CFDynamics only offers the real deal. Real Service. Real Satisfaction. Real Value.


Real Freedom.



Visit us online or call to order!

 macromedia
ALLIANCE PARTNER

Paper|Thin Partner

 BlueDragon Certified
V.I.P. Hosting Partner



Flash Animation Learning Guide

By tweening shapes, you can create an effect similar to morphing - Part 2
by jen dehaan & chris georgenes

by tweening shapes, you can create an effect similar to morphing, making one shape appear to change into another shape over time. Flash can also tween the location, size, color, and opacity of shapes.

Tweening one shape at a time usually yields the best results. If you tween multiple shapes at one time, and want them to morph together, all the shapes must be on the same layer. Otherwise, for some effects, you should shape tween each shape on separate layers if you do not want them to affect each other. Each rectangle—the curtain effect in the background and the rectangle on top—tweens on its own separate layer.

To apply shape tweening to groups, instances, or bitmap images, you must first break these elements apart (Modify > Break Apart). To apply shape tweening to text, you must break the text apart twice to convert the text to objects.

Note: To control more complex or improbable shape changes, you use shape hints, which control how parts of the original shape move into the new shape. For information, see the following section, Using Shape Hints with Shape Tweens.

To tween a shape:

- 1 Create a new file and call it shape-between.fla.
- 2 Select Frame 1 of Layer 1. This is where the animation will start.
- 3 Create a graphic with the Pen, Oval, Rectangle, Pencil, or Brush tool. For best results when you're starting out, the frame should contain only one drawing (such as a single oval shape).

4 Select Frame 10 of Layer 1 and create a second keyframe (F6).

5 Select the artwork on the Stage at Frame 10 and do one of the following:

- Modify the shape, color, opacity, or position of the artwork.
- Delete the artwork and place new artwork in the second keyframe (it should be a raw graphic drawing as well).

6 Select a frame in Timeline between Frame 1 and Frame 10.

7 In the Property inspector (Window > Properties > Properties), select Shape from the Tween pop-up menu.

8 Select an option for Blend:

- Distributive creates an animation in which the intermediate shapes are smoother and more irregular.
- Angular creates an animation that preserves apparent corners and straight lines in the intermediate shapes.

Note: Angular is appropriate only for blending shapes with sharp corners and straight lines. If the shapes you select do not have corners, Flash reverts to distributive shape tweening.

Using Shape Hints with Shape Tweens

To control more complex or improbable shape changes, you can use shape hints. Shape hints identify points that should correspond to starting and ending shapes. For example, if you are tweening the letter M into the letter N, you can use a shape hint to mark corner of the letter's shape. Then, instead of the letters becoming a jumble of lines while the shape

change takes place, each letter remains recognizable and changes separately during the shift.

To use shape hints:

- 1 Create a shape tween (see the previous section, Creating Shape Tweens).
- 2 Select Frame 1 of the layer with the animation on the Timeline.
- 3 Select Modify > Shape > Add Shape Hint.
- 4 Move the shape hint to an edge or corner that you want to mark.
- 5 Select the next keyframe in the tweening sequence.
- 6 Move the shape hint to the edge or corner in the ending shape that should correspond to the first location you marked.

Repeat this process to add additional shape hints. New hints appear with the letters that follow (b, c, and so on).

The SWF files animate the letter M into the letter N. The top SWF file does not use shape hints to morph the two letters, while the bottom SWF file uses shape hints to improve the morphing. Refer to the sample file shape_hints.fla, which is part of the animation_samples.zip archive that accompanies this article.

Shape hints contain letters (a through z) for identifying which points correspond in the starting and ending shape. You can use up to 26 shape hints. Shape hints are yellow in a starting keyframe, green in an ending keyframe, and red when within a fill area or outside the shape (Flash ignores red shape hints).

For best results when tweening shapes, follow these guidelines:

- In complex shape tweening, create intermediate shapes and tween them

instead of just defining a starting and ending shape.

- Make sure that shape hints are logical. For example, if you're using three shape hints for a triangle, they must be in the same order on the original triangle and on the triangle to be tweened. The order cannot be abc in the first keyframe and acb in the second.
- Shape hints work best if you place them in counterclockwise order beginning at the top left corner of the shape.

Creating Frame-by-Frame Animations

A frame-by-frame animation changes the contents of the Stage in every frame and is best suited to a complex animation in which an image changes in every frame instead of simply moving across the Stage. This type of animation increases the file size more rapidly than tweened animation because Flash stores the values for each keyframe.

To create a frame-by-frame animation, you define each frame as a keyframe and create a different (typically modified) image for each frame. Each new keyframe on a layer typically contains the same contents as the keyframe preceding it because the contents of a frame are copied to the next keyframe when you select a frame and press F6. By selecting a frame and pressing F6, you can modify each new keyframe in the animation incrementally. The bee's wing moves only slightly so that a frame-by-frame animation was used to move the wing very slightly on sequential frames.

Often, you use the onion skin feature to view incremental changes between each keyframe. A motion tween has been applied to the lemur's arm and head. The onion skin tool enables you to view multiple frames of the animation at once.

To turn on onion skinning, click the Onion Skin or Onion Skin Outlines button near the bottom of the Timeline. Drag the markers above the Timeline to view multiple frames at once. The onion skin outlines are enabled for an animation on the Stage.

To create a frame-by-frame animation:

- 1 Create a new file and call it frameby-frame fla.
- 2 Select Frame 1 of Layer 1. This is where the animation will start.

- 3 Create a graphic with the Pen, Oval, Rectangle, Pencil, or Brush tool. You can also paste graphics from the Clipboard or import a file (such as an Adobe Illustrator illustration).
- 4 Select the next frame on Layer 1 and create a second keyframe (F6). The contents of Frame 2 are the same as Frame 1 at this time.
- 5 Alter the contents of this frame on the Stage to develop the next increment of the animation. You might select the graphic and move it a couple pixels to the right, add some new lines, or bend a line on a shape.
- 6 To complete your frame-by-frame animation sequence, repeat Steps 4 and 5 until you've built the motion you want.
- 7 To test the animation sequence, select Control > Play or Control > Test Movie.

Editing Your Animations

In Flash 8 you can use various tools to edit your animations, such as commands to insert frames, modify keyframes, onion-skinning tools, and the ability to move your animations around timelines.

After you create a frame or a keyframe, you can move it elsewhere in the active layer or to another layer, remove it, and make other changes. Only keyframes are editable. You can view tweened frames, but you can't edit them directly. To edit tweened frames, you change one of the defining keyframes or insert a new keyframe between the beginning and ending keyframes. You can drag items from the Library panel onto the Stage to add the items to the current keyframe.

To display and edit more than one frame at a time, use onion skinning, covered next.

figure 1

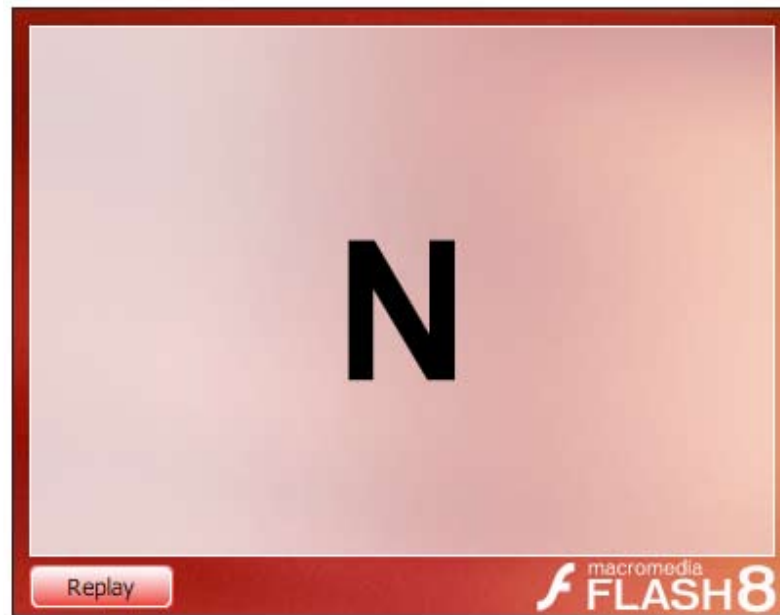
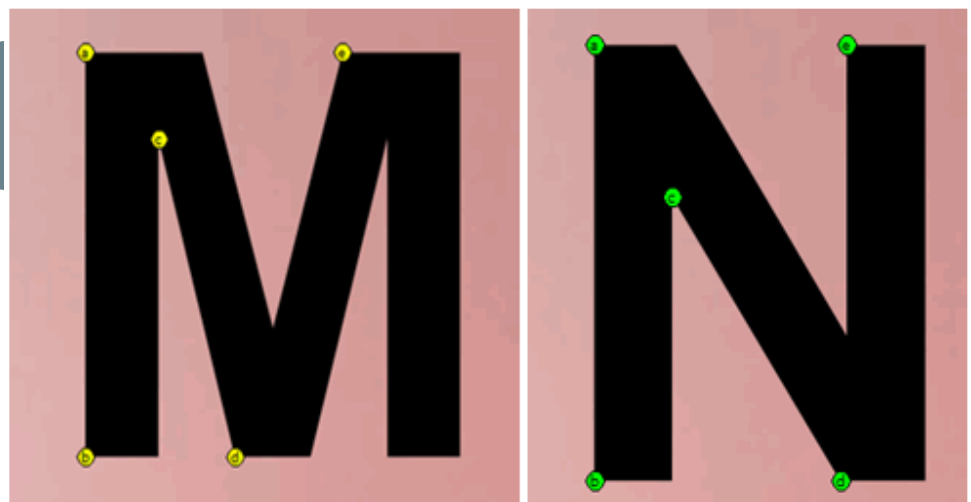


figure 2





Inserting and Modifying Frames

To insert frames in the Timeline, do one of the following:

- To insert a new frame, select Insert > Timeline > Frame.
- To create a new keyframe, select Insert > Timeline > Keyframe, or right-click (Windows) or Control-click (Macintosh) the frame where you want to place a keyframe, and select Insert Keyframe from the context menu.
- To create a new blank keyframe, select Insert > Timeline > Blank Keyframe, or right-click (Windows) or Control-click (Macintosh) the frame where you want to place the keyframe, and select Insert Blank Keyframe from the context menu.

To delete or modify a frame or keyframe, do one of the following:

- To delete a frame, keyframe, or frame sequence, select the frame, keyframe, or sequence and right-click (Windows) or Control-click (Macintosh) the frame, keyframe, or sequence and select Remove Frames from the context menu. Surrounding frames remain unchanged.
- To move a keyframe or frame sequence and its contents, select the keyframe or sequence, then drag to the desired location.
- To extend the duration of a keyframe, Alt-drag (Windows) or Option-drag

(Macintosh) the keyframe to the final frame of the new sequence.

- To copy a keyframe or frame sequence by dragging, select the keyframe or sequence, then ALT-drag (Windows) or Option-drag (Macintosh) to the new location.
- To copy and paste a frame or frame sequence, select the frame or sequence and select Edit > Timeline > Copy Frames. Select a frame or sequence that you want to replace, and select Edit > Timeline > Paste Frames.
- To convert a keyframe to a frame, select the keyframe and select Modify > Timeline > Clear Keyframe, or right-click (Windows) or Control-click (Macintosh) the keyframe and select Clear Keyframe from the context menu. The cleared keyframe and all frames up to the subsequent keyframe are replaced with the contents of the frame preceding the cleared keyframe.
- To change the length of a tweened sequence, drag the beginning or ending keyframe left or right. To change the length of a frame-by-frame sequence, see Creating frame-by-frame animations.
- To add a library item to the current keyframe, drag the item from the Library panel onto the Stage.
- To reverse an animation sequence, select the appropriate frames in one or more layers and select Modify >

Timeline > Reverse Frames. There must be keyframes at the beginning and end of the sequence.

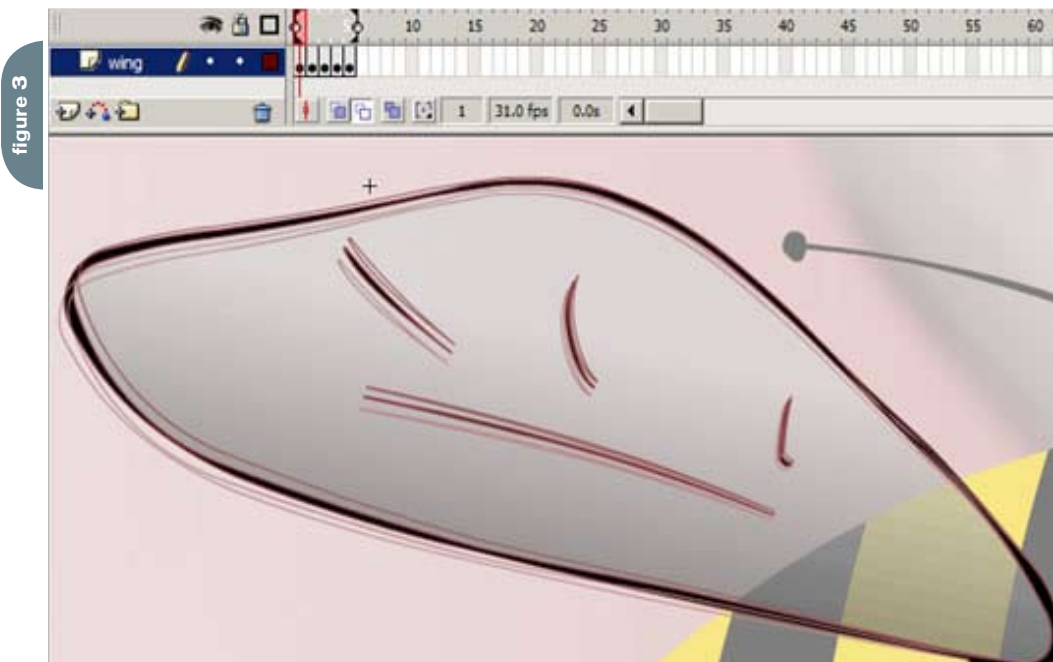
Using Onion Skinning

Normally, Flash displays one frame of the animation sequence at a time on the Stage. To help you position and edit a frame-by-frame animation, you can view two or more frames on the Stage at once. The frame under the playhead appears in full color, while surrounding frames are dimmed, making it appear as if each frame were drawn on a sheet of translucent onion-skin paper and the sheets were stacked on top of each other. Dimmed frames cannot be edited.

To simultaneously see several frames of an animation on the Stage, click the Onion Skin button. All frames between the Start Onion Skin and End Onion Skin markers (in the Timeline header) are superimposed as one frame in the Document window.

To control onion skinning display, do any of the following:

- To display onion skinned frames as outlines, click the Onion Skin Outlines button.
- To change the position of either onion skin marker, drag its pointer to a new location. (Normally, the onion skin markers move in conjunction with the current frame pointer.)
- To enable editing of all frames between onion skin markers, click the





> You don't have to mortgage the farm for a **web content manager** that's **Powerful** and **Affordable**.

Our newest partners:

CFMX Hosting, Fulgen Technology, Harbour Light Strategic Marketing and ipXperts

> **Call for more info:**
1.866.870.6358

www.BeSavvy.com

Savvy Software Inc.



macromedia®
ALLIANCE PARTNER

What's your PDF?



Precise Document Formatting



With activePDF Server, you gain full control over your PDF output with conversion options that allow you to specify page size, compression and resolution options, embed text, create bookmarks, concatenate to existing files, and more. Licensed per server, you can easily add PDF generation to virtually any Windows application.



Populate Dynamic Forms



With activePDF Toolkit's form-filling capabilities, you can dynamically populate PDF forms with data and images from a database, allow users using only Adobe Reader to fill-in and save forms and use PDF forms as document templates to precisely control image placement and resizing. With Toolkit's robust API, the automation of virtually any PDF manipulation task becomes possible - append, stamp, stitch, merge, paint, secure PDF and more.



Promote Digital Fidelity



Do you need to standardize PDF output within your enterprise? With DocConverter, you can easily use built-in support for "watched" folders to implement server-side PDF generation in a matter of minutes, with full control over the PDF output at the server level. Or, use DocConverter's programmable COM object to integrate convert-to-PDF functionality within your enterprise application.



Present Data Fashionably



Ensuring precise layout of an HTML document can be a nightmare, especially when printing. PDF guarantees pixel-perfect layout every time as what you see is what you print. With activePDF WebGrabber, you can dynamically convert any URL, HTML stream, or HTML file to PDF on the fly, while maintaining embedded styles.

Download your
free trial version today at
www.activePDF.com

Copyright © 2004, activePDF, Inc. All Rights Reserved.
"ACTIVEPDF", "Leading the iPaper Revolution" and the activePDF logo are registered trademarks of activePDF, Inc. All activePDF product names are trademarks of activePDF, Inc.





Edit Multiple Frames button. Usually onion skinning lets you edit only the current frame. However, you can display the contents of each frame between the onion skin markers normally, and make each available for editing, regardless of which is the current frame.

Note: Locked layers (those with a padlock icon) aren't displayed when onion skinning is turned on. To avoid a multitude of confusing images, you can lock or hide the layers you don't want onion skinned.

To change the display of onion skin markers, click the Modify Onion Markers button and select an item from the menu:

- Always Show Markers displays the onion skin markers in the Timeline header whether or not onion skinning is on.
- Anchor Onion locks the onion skin markers to their current position in the Timeline header. Normally, the Onion Skin range is relative to the current frame pointer and the Onion Skin markers. By anchoring the Onion Skin markers, you prevent them from moving with the current frame pointer.
- Onion 2 displays two frames on either side of the current frame.
- Onion 5 displays five frames on either side of the current frame.
- Onion All displays all frames on either side of the current frame.

Moving an Entire Animation

If you need to move an entire animation on the Stage, you must move the graphics in all frames and layers at once to avoid realigning everything.

To move the entire animation to another location on the Stage:

- 1 Unlock all layers. To move everything on one or more layers but nothing on other layers, lock or hide all the layers you don't want to move.
- 2 Click the Edit Multiple Frames button in the Timeline.
- 3 Drag the onion skin markers so that they enclose all the frames you want to select, or click Modify Onion Markers and select Onion All.
- 4 Select Edit > Select All.
- 5 Drag the entire animation to the new location on the Stage.

Using Timeline Effects

Flash includes prebuilt Timeline effects that enable you to create complex animations with a minimal number of steps. You can apply Timeline effects to the following objects:

- Text
- Graphics, including shapes, groups, and graphic symbols
- Bitmap images
- Button symbols

Note: Timeline effects share some of the same names as filter effects; however, they are completely different features. Timeline effects are automated vector animations you apply to the previously mentioned objects. Filter effects are static effects you apply to objects and animate in various ways by applying code or motion tweens.

Adding a Timeline Effect

When you add a Timeline effect to an object, Flash creates a layer and transfers the object to the new layer. The object is placed inside the effect graphic, and all tweens and transformations required for the effect reside in the graphic on the newly created layer. The new layer automatically receives the same name as the effect, appended with a number that represents the order in which the effect is applied, out of all effects in your document.

When you add a Timeline effect, a folder with the effect's name is added to the Library, containing elements used in creating the effect.

To add an effect to an object:

- 1 Do one of the following to add a Timeline effect:
 - Select the object to which you're adding the Timeline effect. Select Insert > Timeline Effects. Then select Assistants, Effects, or Transition/Transform from the submenu, and select an effect from the list.
 - Right-click (Windows) or Control-click (Macintosh) the object to which you're adding the Timeline effect. From the context menu, select Timeline Effects. Then select Assistants, Effects, or Transition/Transform from the submenu, and select an effect from the list.
- 2 Effects available for the type of object you've selected appear as active menu choices.

- 3 In the dialog box that appears for the effect, view the effect preview based on default settings. Modify the default settings as desired, and then click Update Preview to view the effect with the new settings.
- 4 When the Timeline effect appears as desired in the preview window, click OK.

Editing a Timeline Effect

You edit Timeline effects using the Effect Settings dialog box:

- 1 Select the object associated with the effect on the Stage.
- 2 To open the Effect Settings dialog box, do one of the following:
 - In the Property inspector, click Edit.
 - Right-click (Windows) or Control-click (Macintosh) the object and select Timeline Effects > Edit Effect from the context menu.
- 3 In the Effect Settings dialog box, edit the settings you want to edit, and click OK to apply your settings.

Deleting a Timeline Effect

You use the context menu to delete Timeline effects. Right-click (Windows) or Control-click (Macintosh) the object on the Stage that has the Timeline effect you want to remove, and select Timeline Effects > Remove Effect from the context menu.

About Scripting Animation

You can use ActionScript 2.0 to add animation to your Flash applications instead of using motion or shape tweens on a timeline. The following sections show you how to use code to animate instances, such as changing the transparency and appearance of the instance, and moving the instance around the Stage.

For information on using the Tween and TransitionManager classes to further automate code-based animations, see the following section, Using the Tween and TransitionManager Classes. These classes help you add advanced easing equations and transition animations to movie clip instances in your application. Many of these effects are difficult to recreate using ActionScript without these prebuilt classes because the code involves writing complex mathematical equations to achieve the effect.

figure 4



The animation uses code to tween the bee horizontally across the Stage. Notice the easing that has been applied to the motion as well. This animation uses very few lines of code to achieve this effect.

Note: Issues regarding frame rate discussed in the earlier section, About Frame Rate and Animation, also apply to scripted animation.

Fading a Movie Clip

When you work with movie clips on the Stage, you might want to fade the movie clip in or out instead of toggling its `_visible` property. The following examples, taken from the Flash documentation, show you a variety of simple ways to use ActionScript to create animation.

The following procedure demonstrates how to use an `onEnterFrame` event handler to animate a movie clip. To fade a movie clip by using code:

- 1 Create a new Flash document called `fade1.fla`.
- 2 Draw some graphics on the Stage using the drawing tools, or import an image to the Stage (File > Import > Import to Stage).
- 3 Select the content on the Stage and select Modify > Convert to Symbol.
- 4 Select the Movie Clip option and click OK to create the symbol.
- 5 Select the movie clip instance on the Stage and type `img1_mc` in the Instance Name text box in the Property inspector.
- 6 Select Frame 1 of the Timeline and add the following code to the Actions panel:

```
img1_mc.onEnterFrame = function() {
    img1_mc._alpha -= 5;
    if (img1_mc._alpha <= 0) {
        mg1_mc._visible = false;
        delete img1_mc.onEnterFrame;
    }
};
```

This code uses an `onEnterFrame` event handler, which is invoked repeatedly at the frame rate of the SWF file. The number of times per second that the event handler is called depends on the frame rate at which the Flash document is set. If the frame rate is 12 fps, the `onEnterFrame` event handler is invoked 12 times per second. Likewise, if the Flash document's frame rate is 30 fps, the event handler is invoked 30 times per second.

- 7 Select Control > Test Movie to test the document. The movie clip you added to the Stage slowly fades out.

You can modify the `_alpha` property by using the `setInterval()` function instead of an `onEnterFrame` event handler, as the next procedure shows. To fade an object by using the `setInterval()` function:

- 1 Create a new Flash document called `fade2.fla`.
- 2 Draw some graphics on the Stage, or import an image to the Stage (File > Import > Import to Stage).
- 3 Select the content on the Stage and select Modify > Convert to Symbol.
- 4 Select the Movie Clip option and click OK to create the symbol.
- 5 Select the movie clip instance on the Stage and type `img1_mc` in the Instance Name text box in the Property inspector.

- 6 Select Frame 1 of the Timeline and add the following code to the Actions panel:

```
var alpha_interval:Number =
setInterval(fadeImage, 50, img1_mc);
function fadeImage(target_mc:
MovieClip):Void {
    target_mc._alpha -= 5;
    if (target_mc._alpha <= 0) {
        target_mc._visible = false;
        clearInterval(alpha_interval);
    }
}
```

The `setInterval()` function behaves slightly differently than the `onEnterFrame` event handler because the `setInterval()` function tells Flash precisely how frequently the code should call a particular function. In this code example, the user-defined `fadeImage()` function is called every 50 milliseconds (20 times per second). The `fadeImage()` function decrements the value of the current movie clip's `_alpha` property. When the `_alpha` value is equal to or less than 0, the interval is cleared and the `fadeImage()` function stops executing.

- 7 Select Control > Test Movie to test the document. The movie clip you added to the Stage slowly fades out.

To move an instance on the Stage by using code:

- 1 Create a new Flash document called `moveClip.fla`.
- 2 Change the frame rate of the document to 24 fps in the Property inspector. The animation is much smoother if you use a higher frame rate such as 24 fps.
- 3 Select Frame 1 of the Timeline and add the following code to the Actions panel:

```
// Create a movie clip instance.
this.createEmptyMovieClip("img1_mc",
10);
var mcl_obj:Object = new Object();
mcl_obj.onLoadInit = function (target_mc:MovieClip):Void {
    target_mc._x = Stage.width;
    target_mc.onEnterFrame = function() {
        target_mc._x -= 3; // decrease
current _x position by 3 pixels
        if (target_mc._x <= 0) {
```



```

        target_mc._x = 0;
        delete target_
mc.onEnterFrame;
    }
    };
};
var img_mc1:MovieClipLoader = new
MovieClipLoader();
img_mc1.addListener(mc1_obj);
// Load an image into the movie
clip
img_mc1.loadClip("http://www.helpex-
amples.com/flash/images/image1.jpg",
img1_mc);

```

This code example loads an external image from a remote web server and, when the image is fully loaded, animates it horizontally across the Stage. Instead of using an onEnterFrame event handler, you could use the setInterval() function to animate the image.

4 Select Control > Test Movie to test the document. The image loads and then animates from the right side of the Stage to the upper-left corner of the Stage.

Using the Tween and TransitionManager Classes

When you install Flash Basic 8 or Flash Professional 8, you also install two powerful classes: the Tween and TransitionManager classes. This section describes how to use these classes with movie clips and Macromedia V2 components (included with Flash MX 2004 and Flash 8) to add animation easily to your SWF files.

If you create a slide presentation or form application with Flash Professional 8 (ActionScript 2.0 only), you can select behaviors that add different kinds of transitions between slides, which is similar to when you create a PowerPoint presentation. You add this functionality into a screen application by using the Tween and TransitionManager classes, which generate ActionScript that animates the screens depending on the behavior you choose.

You can also use the Tween and TransitionManager classes outside of a screen-based document in either Flash Basic 8 or Flash Professional 8. For example, you can use the classes with the component set of version 2 of the Macromedia Component Architecture, or

with movie clips. If you want to change the way a ComboBox component animates, you can use the TransitionManager class to add some easing when the menu opens. You can also use the Tween and TransitionManager classes, instead of creating motion tweens on the Timeline or writing custom code, to create your own animated menu system.

Note: The Tween and TransitionManager classes are available only in ActionScript 2.0, but these classes are available in both Flash Basic 8 and Flash Professional 8.

Tween Class

The Tween class enables you to use ActionScript to move, resize, and fade movie clips easily on the Stage by specifying a property of the target movie clip to be tween-animated over a number of frames or seconds. The Tween class also enables you to specify a variety of easing methods. Easing refers to gradual acceleration or deceleration during an animation, which helps your animations appear more realistic. For example, the options on a drop-down list component you create might gradually increase their speed near the beginning of an animation as the options appear, but slow down before the options come to a full stop at the end of the animation as the list is extended. Flash provides many easing methods that contain equations for this acceleration and deceleration, which change the easing animation accordingly.

The Tween class also invokes event handlers so your code may respond when an animation starts, stops, or resumes or increments its tweened property value. For example, you can start a second tweened animation when the first tween invokes its Tween.onMotionStopped event handler, indicating that the first tween has stopped.

TransitionManager Class

The TransitionManager class and the effect-defining transition-based classes enable you to apply impressive transition animation effects quickly to slides and movie clips.

As its name implies, the TransitionManager class manages transitions by implementing animation events. It enables you to apply one of 10 animation effects to movie clips. The transition

effects are defined in a set of transition classes that all extend the base class mx.transitions.Transition.

Combining Animation, Filters, and the Tween Class

You can use ActionScript, such as the Tween class, to animate filters at runtime, which enables you to apply interesting, animated effects to your Flash applications. In the following example, you see how to combine the Blur filter with the Tween class to create an animated blur that modifies the Blur filter between a value of 0 and 10 at runtime.

To animate blurs using the Tween class:

- 1 Create a new Flash document and save it as animatedfilter fla.
- 2 Add the following ActionScript to Frame 1 of the Timeline:

```

import flash.filters.BlurFilter;
import mx.transitions.Tween;
import mx.transitions.easing.*;

this.createEmptyMovieClip("holder_mc",
10);
holder_mc.createEmptyMovieClip("img_
mc", 20);

var mc1Listener:Object = new Object();
mc1Listener.onLoadInit =
function(target_mc:MovieClip) {
    target_mc._x = (Stage.width - tar-
get_mc._width) / 2;
    target_mc._y = (Stage.height - tar-
get_mc._height) / 2;
    var myTween:Tween = new
Tween(target_mc, "blur", Strong.easeI-
nOut, 0, 20, 3, true);
    myTween.onMotionChanged = func-
tion() {
        target_mc.parent.filters =
[new BlurFilter(target_mc.blur, tar-
get_mc.blur, 1)];
    };
    myTween.onMotionFinished = func-
tion() {
        myTween.yoyo();
    }
};
var my_mc1:MovieClipLoader = new
MovieClipLoader();
my_mc1.addListener(mc1Listener);
my_mc1.loadClip("http://www.helpex-
amples.com/flash/images/image1.jpg",
holder_mc.img_mc);

```

Macromedia Instructional Media Development (IMD) is a team of instructional designers, developers, technical writers, technical editors, publishing engineers, and multimedia artists. IMD authors the online help systems, manuals, tutorials, and examples that help customers learn how to use Macromedia products. Jen deHaan, a rather awkward and uncool Canadian, likes robots and pirates (as well as robotic pirates). Jen works on documentation at Macromedia in San Francisco. She also maintains a blog at weblogs.macromedia.com/dehaan and believes that root and low-carb diets are unusually evil. Chris Georgenes is a full-time freelance artist, animator, and all-around designer for the web, CD-ROM, and television. His clients include Pileated Pictures, LucasArts, Universal Records, Plot Developers, and AOL, among others. He maintains www.mud-bubble.com as his online portfolio and www.key-framer.com as his Flash tutorial website. Chris is also a member of Team Macromedia.

BALANCE

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

www.communitymx.com



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

COMMUNITY
MX



The preceding code is separated into three sections. The first section imports the required classes and packages.

The second section creates a nested movie clip that is used to load an image and apply filters to the holder movie clip. The final section creates a new `MovieClipLoader` instance and a listener for the movie clip loader.

The listener object defines a single event handler function, `onLoadInit`, which is started once the image successfully loads and is available on the Stage. First the image is repositioned to the center of the Stage and then a new `Tween` object is created that animates the movie clip and applies a `Blur` filter of 0 and 10.

3 Select **Control > Test Movie** to test the Flash document.

Combining Animation and the Drawing API

You can combine the Drawing API with the `Tween` and `TransitionManager` classes to create some excellent animated results, and you only have to write a small amount of ActionScript. For more information on the `Tween` and `TransitionManager` classes, see the previous section, *Using the Tween and TransitionManager Classes*.

The following procedure loads a JPEG image and dynamically masks the image so you can reveal the image slowly after it loads by tweening the image's mask.

- 1 Create a new Flash document and save it as `dynmask fla`.
- 2 Add the following ActionScript to Frame 1 of the Timeline:

```
import mx.transitions.Tween;
import mx.transitions.easing.*;
var mclListener:Object = new Object();
mclListener.onLoadInit =
function(target_mc:MovieClip) {
    target_mc._visible = false;
    // Center the image on the Stage.
    target_mc._x = (Stage.width - target_mc._width) / 2;
    target_mc._y = (Stage.height - target_mc._height) / 2;
    var maskClip:MovieClip = target_mc.createEmptyMovieClip("mask_mc", 20);
    with (maskClip) {
        // Draw a mask that is the same size as the loaded image.
        beginFill(0xFF00FF, 100);
        moveTo(0, 0);
       .lineTo(target_mc._width, 0);
       .lineTo(target_mc._width, target_mc._height);
       .lineTo(0, target_mc._height);
       .lineTo(0, 0);
        endFill();
    }
    target_mc.setMask(maskClip);
    target_mc._visible = true;
    var mask_tween:Object = new Tween(maskClip, "_yscale", Strong.
```

```
easeOut, 0, 100, 2, true);
};
this.createEmptyMovieClip("img_mc", 10);
var img_mcl:MovieClipLoader = new MovieClipLoader();
img_mcl.addListener(mclListener);
img_mcl.loadClip("http://www.helpexamples.com/flash/images/image1.jpg", img_mc);
```

This code example imports the `Tween` class and each of the classes in the `easing` package. Next it creates an object that acts as the listener object for a `MovieClipLoader` instance that's created in a later section of the code. The listener object defines a single event listener, `onLoadInit`, which centers the dynamically loaded JPEG image on the Stage. After the code repositions the image, a new movie clip instance is created within the `target_mc` movie clip (which contains the dynamically loaded JPEG image). The Drawing API code draws a rectangle with the same dimensions as the JPEG image within this new movie clip. The new movie clip masks the JPEG image by calling the `MovieClip.setMask()` method. After the mask is drawn and set up, the mask uses the `Tween` class to animate, which causes the image to slowly reveal itself.

3 Save the Flash document and select **Control > Test Movie** to test the SWF file.

Note: To animate `_alpha` in the previous example instead of `_yscale`, tween the `target_mc` movie clip directly instead of the mask movie clip.

You can use alpha masks if you apply runtime bitmap caching. See an sample FLA file demonstrate this feature on the Flash Samples page (http://macromedia.com/support/documentation/en/flash/fl8/samples.html#alpha_mask).

*Instructional Media Development (IMD)
Macromedia Documentation
Editor: Jen deHaan
www.flash8forums.com*

*Reviewer: Chris Georgenes
www.mudbubble.com*

figure 5





M E T A L I Q

MetalIQ Components (mCOM)

Creating data-driven interfaces for online applications can be complicated and time-consuming. Developers and designers worldwide need an extensible solution that saves time and headaches. There needs to be a stable, innovative approach to solving common UI problems, such as navigation, forms, filtering and sorting immense amounts of data.

Start your development with MetalIQ Components (**mCOM**). Completely new and developed from the ground up specifically for RIA (Rich Internet Applications) and online experiences, **mCOM** are easy to integrate, lightweight and built on the same industrial strength code deployed across some of the largest websites, portals and web applications in the world.

mCOM allow you to quickly create stable, lightweight data-driven interfaces that are extensible by designers and developers alike, both through extending classes and a simpler visual skinning process. **mCOM** include a fully implemented advanced focus manager that works seamlessly with all of our components, and allows for easy creation of tab groups. For anyone familiar with Flash components, there is no learning curve. In most cases, Flash components can be easily updated to **mCOM** with little or no modifications to your project.

Component List:

- Accordion
- Button
- Check Box
- Color Pickers
- Combo Box
- Data Grid
- Field Set
- List
- Menu
- Numeric Stepper
- Radio Button
- Scroll Bar
- Scrolling Pane
- Slider
- Tab Box
- Text Area
- Text Input
- And more...



mCOM
MetalIQ Components
in sample interface



Key Features:

- Up to 75% smaller file size
- Faster download
- Lower CPU usage
- Simplified customization
- Flash Player 6, 7 and 8 compatible
- Reusable
- Easy to integrate

Integrated Technology:

- Effective, understandable code architecture
- Advanced tabbing management
- Improved event model
- Written in ActionScript 2.0
- Advanced focus manager

MetalIQ is a solutions company, specializing in industrial strength creative, code and consulting. Identifying, creating, and implementing innovative solutions that are usable, intuitive and appealing. MetalIQ offers a suite of solutions for video (**mVID**), dashboards (**mDAT**), applications (**mCAP**) and components (**mCOM**). Contact us by phone at 415.642.3332, email us at mCOM@metalIQ.com or visit us at metalIQ.com/mCOM.



Using Bitmap Caching in Flash

Developers and designers use Flash to do a lot more than just animation
by guy watson

As most of you know, Flash began as a tool for creating vector animations on the web. Flash Player was designed specifically as a lightweight animation viewer to display moving vector objects, which are, in their simplest form, mathematical equations that describe complex shapes made up of points, lines, curves, and fills.

However, today, developers and designers use Flash to do a lot more than just animation; today, we use Flash for everything from interactive banner ads to games and large applications with complex user interfaces. We are now pushing Flash to its limits, and our frame rates are starting to suffer as we develop content that is more application-centric.

Previous versions of Flash Player showed some obvious performance limitations. The vector renderer inside the player generally coughed and spluttered when it attempted to play Flash applications with a lot of objects on the Stage. This is because the player was not optimized to deal with large amounts of static content. By design, on each frame, Flash Player 7 and previous versions had to update and redraw all the vector objects on the Stage, even if they had not changed. This was an intensive and unnecessary process.

Flash Player 8 addresses these visual performance issues with various improvements and optimizations to the renderer. There are also various new performance-enhancing authoring features in Flash Professional 8, which developers can use to take advantage of these significant changes.

This article will show you how to effectively use these new performance-related features to increase the frame rates of your Flash applications. An intermediate to advanced familiarity with

ActionScript and Flash Player functionality is required.

Bitmap Caching

One of the most significant additions to Flash Player 8 has to be bitmap caching. This feature gives developers that are experiencing poor frame rates the power to greatly increase the speed at which large amounts of objects are updated and drawn onto the Stage by the renderer. The renderer is a very important part of Flash Player; it is responsible for everything that you see when you view a Flash application, as it draws all the vector and bitmap data onto the Stage. On each frame, the renderer has to update the Stage and the various objects it contains to reflect any changes that have occurred since the last frame. This process can become quite intensive when the application updates a large amount of information on any given frame. Using this feature gives the developer some control over the amount of work that the renderer has to perform in each frame – the less work you give the renderer, the faster and smoother your Flash application will run.

How Bitmap Caching Works

When you turn on bitmap caching for any given movie clip, the player converts the contents of the movie clip into a bitmap, which it generates and then stores in memory alongside the original vector data equivalent. The renderer then displays this bitmap in the place of the vector data by copying the image from memory onto the Stage.

This process essentially makes the renderer's life easier, because it doesn't have to update the movie clip each frame. Instead the process only has to draw the bitmap it generated once, and

from then on it simply copies the bitmap from memory onto the Stage. If you change the movie clip or its contents, Flash regenerates the bitmap. There is little or no visual difference when a movie clip has bitmap caching turned on. You may notice a very slight difference because the vector data is snapped to the nearest whole pixel when the bitmap is generated. Bitmap caching also works perfectly well with nested movie clips (movie clips inside movie clips).

To put it in simple terms, by turning on bitmap caching for a movie clip, you are essentially telling Flash Player, "Hey Renderer, I'll make your life a little easier. Freeze this movie clip and display it as a bitmap instead, because this movie clip or its contents are not going to change very often, if at all. They are static."

Turning Bitmap Caching On and Off

Bitmap caching comes in the form of an additional movie clip property that can be switched on or off at author-time using the Property inspector and at runtime using ActionScript, on a per movie clip basis.

Using the Authoring Environment

You can turn bitmap caching on or off for any movie clip in the authoring environment using the Property inspector. Selecting this option disables bitmap caching for all movie clips by default.

To turn bitmap caching on select the desired movie clip instance on Stage by clicking on it. Then, open the Property Inspector from the Window menu. Select Window > Properties > Properties, or use the keyboard shortcut Control + F3.

In the lower right corner of the Property inspector underneath the Blend

field, select the Use Runtime Bitmap Caching option for the selected movie clip by clicking the checkbox. Bitmap caching is deselected or turned off by default for all movie clips.

Using ActionScript

You can also turn bitmap caching on and off at runtime using ActionScript. Every movie clip object now has a new ActionScript property called `cacheAsBitmap`. To turn bitmap caching on for a movie clip, you simply need to set the value of its `cacheAsBitmap` property to true using the following code:

```
someMovieClip.cacheAsBitmap=true;
//Turn Bitmap Caching ON
```

Similarly, to turn bitmap caching off for a movie clip using ActionScript, you simply need to set the value of its `cacheAsBitmap` property to false as follows:

```
someMovieClip.cacheAsBitmap=false;
//Turn Bitmap Caching OFF
```

You can also determine if a movie clip has bitmap caching turned on by retrieving the value of the `cacheAsBitmap` property:

```
isCached=someMovieClip.cacheAsBitmap;
```

When to Use Bitmap Caching

If used correctly, bitmap caching can dramatically reduce the amount of updating instructions that the renderer has to process in every frame, and your applications should, in practice, run a lot quicker and smoother. However, in certain circumstances bitmap caching could have a detrimental effect on the performance of your Flash application, therefore you should choose the movie clips you cache wisely.

Bitmap caching works best for movie clips whose visual appearance doesn't change often or not at all. This is because when a cached movie clip or its contents change, Flash regenerates the bitmap with new data for the area or region of vector data that changed and updates

the bitmap held in memory. The renderer then displays the new bitmap.

Bitmap caching also works well for movie clips that contain complex vector data (for example, shapes with lots of curves or gradient fills), as it is quicker for the renderer to copy a bitmap from memory onto the Stage than to draw all those vectors to the Stage.

Every time you rotate, scale, or change the alpha of a cached movie clip, the whole bitmap has to be regenerated. So, turning on bitmap caching for a movie clip that is constantly rotating, changing in size, or contains an animation doesn't make much sense because in every frame the renderer has to update the bitmap to reflect the new appearance of the movie clip as well as redraw it to the Stage, which adds overhead.

On the other hand, bitmap caching works well for both static movie clips and movie clips that move, as long as they don't change visually. It is perfectly fine to move a cached movie clip around the Stage, either with ActionScript or with Timeline animation, because moving it around won't change its visual appearance. The cached bitmap doesn't have to be updated, and the renderer will merely draw the bitmap at its new position.

Valid Use-Case Scenario

Consider the following use-case scenario: you have a large application that contains multiple Window components; each Window component contains numerous other components. Each Window component is draggable. When the user drags one of the Window components around the application, he notices a slowdown—the Window component has to catch up to the mouse position, which causes jerky movement. To fix this problem, the developer turns on bitmap caching for each Window component instance. Now the player only has to update the internal bitmap representation of each Window component when the visual appearance of one of the UI components

inside it changes. The rest of the time – even when the Window component is being dragged around – the player simply needs to move the bitmaps around the Stage, as opposed to constantly updating the Stage from the vector data in each movie clip. The result is a much smoother dragging experience for the user.

When Are Surfaces Regenerated?

As stated previously, changes to a movie clip that make the player regenerate the internal bitmap representation should be used sparingly; otherwise you defeat the purpose of the feature. With bitmap caching on for a movie clip, Flash Player will regenerate the bitmap every time you change any of the following MovieClip ActionScript properties:

- `_xscale`
- `_yscale`
- `_rotation`
- `_alpha`
- `_width`
- `_height`
- `filters`
- `blendMode`
- `opaqueBackground`
- `transform`

So, when using bitmap caching, try to avoid changing any of these ActionScript properties on a regular basis.

The bitmap will also be regenerated when:

- The Timeline playhead changes inside the movie clip
- When the outer boundaries of the movie clip change
- When you draw something inside the movie clip with the Drawing API
- When you attach an image or symbol from the Library into the movie clip
- Any of the above occur within a nested movie clip (child movie clip)
- The movie's viewing window changes (for example, the viewer zooms in on the movie by right-clicking and choosing Zoom)

“We are now pushing Flash to its limits, and our frame rates are starting to suffer as we develop content that is more application-centric”



Bitmap Caching and Memory Usage

Bitmap caching naturally makes Flash Player use more memory, because for every cached movie clip the player has to store the vector data and the additional bitmap equivalent in memory. When you turn bitmap caching off for any particular movie clip, Flash removes its bitmap representation from memory.

You should be concerned about the amount of memory that your Flash file uses, because it can affect the performance of other applications that are running on the same computer. The more memory Flash Player uses, the less memory that is available for other programs to run effectively.

Computers only have so much memory available to them in the form of RAM. Most computers nowadays have at least 256MB of RAM. The operating system may provide more memory when required in the form of virtual memory. Flash Player should never use that much memory, but now, with the various new bitmap-related features added to the player, Flash files can consume large amounts of memory at a time, so you should make an effort to minimize memory usage.

Before we talk about the specifics of bitmap caching and memory usage, here's a little background information. As you probably know, a bitmap is made up of pixels. It can be thought of as a grid of color values, which designate a particular color for each and every pixel. Each pixel is a cell in the grid. A 100 x 100 pixel bitmap can be described by a grid of 10,000 color values, one for each pixel.

Each color value in a bitmap is a binary number. A binary number is made up of bits, whose values can be either 0 or 1. This binary number will differ in length, depending upon the color depth of the bitmap. The color depth of a bitmap determines the range of possible color values that can be used in each pixel. For example, each pixel in a 24-bit

image can be one of roughly 16.8 million colors. Those colors are formed by mixing together varying quantities of three primary colors: red, green, and blue. The three main colors are called channels. It follows that:

- Each channel can have 256 possible values (0–255).
- $256 * 256 * 256 = 16.8$ million
- 256 decimal is 11111111 in binary.
- This binary number is 8 bits long. 8 bits is 1 byte.

Therefore each channel in a color uses 1 byte. The bitmaps that are created by Flash Player when it converts a movie clip into a surface have a 32-bit color depth. 32-bit images have four channels: red, green, blue, and an additional alpha channel.

Therefore the color value for each pixel in a surface created by Flash Player is 32 bits long, or 4 bytes.

$$4 * 8 = 32 \text{ bit}$$

The bitmap that is created by Flash Player to represent the visual state of a movie clip when you turn bitmap caching on will have the same dimensions (width and height) as the movie clip.

A cached movie clip that is 100 x 100 pixels has 10,000 pixels.

$$100 * 100 = 10,000 \text{ pixels}$$

Each of those pixels will be 32 bits or 4 bytes. Therefore the movie clip will use an extra 40,000 bytes of memory.

$$10,000 * 4 = 40,000 \text{ bytes}$$

There are 1024 bytes in 1 kilobyte (K). So, 40,000 bytes can also be said to be roughly 40 kilobytes (40K).

Built-in Rules for Bitmap Caching

Flash guides you through best practices of using bitmap caching through some

built-in restrictions. Basically, Flash Player takes care of you by either turning on or turning off bitmap caching automatically in several important circumstances.

Size Limits

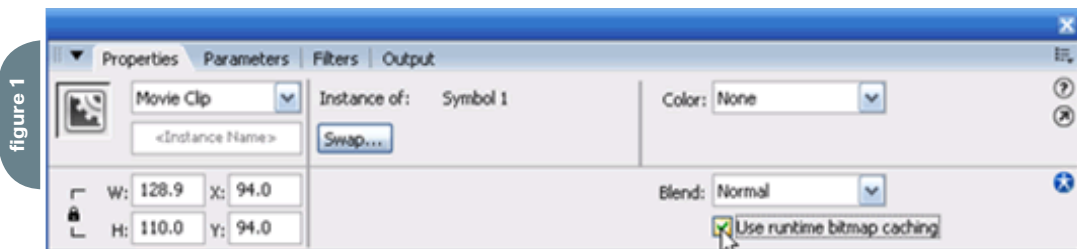
Because it can use excessive amounts of memory, bitmap caching will not work if the dimensions of a cached movie clip are larger (or become larger) than 2880 pixels in either width or height. Flash Player restricts you from using bitmap caching for larger-than-2880 bitmaps to minimize excessive memory use. Why? Because a 2880 x 2880 pixel movie clip that is cached will use up roughly 32MB of memory. Four of these cached movie clips with those same dimensions could potentially fill up a computer's memory and crash the machine.

Filters

It is worth noting here, that if you apply a filter effect to a movie clip either in the Flash authoring environment or through ActionScript, then bitmap caching is automatically turned on, and the `cacheAsBitmap` property will always return true. This happens even if you turned bitmap caching off—either in the authoring environment or with ActionScript. To prove this, create a new movie clip, give it an instance name of `scrollingMovieclip`, and add the following code to the first frame of the main Timeline:

```
var blur=new flash.filters.
BlurFilter(3,3,3); //Gaussian blur
scrollingMovieclip.filters=[blur];
trace(scrollingMovieclip.cacheAsBitmap); //outputs 'true'
```

When all of the filters effects are removed from a movie clip, the `cacheAsBitmap` property will return to its previous state. So for example, if bitmap caching is turned off for a movie clip but it has filters applied to it, then removing all of those filters from the movie clip will turn bitmap caching off again:



BY NOW THERE ISN'T A SOFTWARE DEVELOPER ON EARTH WHO ISN'T AWARE OF THE COLLECTION OF PROGRAMMING TECHNOLOGIES KNOWN AS AJAX!

REAL-WORLD
AJAX
ONE DAY SEMINAR

www.ajaxseminar.com

March 13, 2006

Marriott

Marriott Marquis Times Square
New York City



For more information
Call 201-802-3022 or
email events@sys-con.com

How, in concrete terms, can you take advantage in your own projects of this newly popular way of delivering online content to users without reloading an entire page?

How soon can you be monetizing AJAX?

This "Real-World AJAX" one-day seminar aims to answer these exact questions...

Led by "The Father of AJAX" himself, the charismatic Jesse James Garrett of Adaptive Path, "Real-World AJAX" has one overriding organizing principle: its aim is to make sure that delegates fortunate enough to secure themselves a place – before all seats are sold out – will leave the seminar with a sufficient grasp of Asynchronous JavaScript and XML to enable them to build their first AJAX application on their own when they get back to their offices.

HURRY!
LIMITED SEATING
THIS SEMINAR WILL
SELL-OUT
CALL 201-802-3022
TO REGISTER!

Jeremy Geelan
Conference Chair, Real-World AJAX
jeremy@sys-con.com



Jesse James Garrett
Father of AJAX



Scott Dietzen
Creator of WebLogic,
Ph.D., President and
CTO, Zimbra



Bill Scott
AJAX Evangelist
of Yahoo!



David Heinemeier Hansson
Creator of Ruby on Rails



Satish Dharmaraj
Father of JSP, Co-
Founder & CEO, Zimbra



Rob Gonda
Best-Selling AJAX
Author, CTO,
iChameleon Group



Dion Hinchcliffe
Co-founder & CTO,
Sphere of Influence Inc.



Ross Dargahi
Well-known AJAX
Evangelist & Co-founder
and VP of Engineering,
Zimbra

Early Bird	\$995
(Before January 31, 2006)	
Discounted Price	\$1,195
(Before February 28, 2006)	
Seminar Price	\$1,295
(After February 28, 2006, and if any seat is available)	

MEDIA SPONSOR



LIVE SIMULCAST!
AROUND THE WORLD ON SYS-CON TV

PRODUCED BY
SYS-CON
EVENTS



“When you load an external Flash movie or image into a cached movie clip using ActionScript, bitmap caching is automatically turned off”

```
scrollingMovieClip.filters=undefined;
trace(scrollingMovieClip.cacheAsBitmap); //outputs 'false'
```

Loading External Content

When you load an external Flash movie or image into a cached movie clip using ActionScript, bitmap caching is automatically turned off. This is because when Flash loads a movie clip, it totally resets it, deleting all variables inside it, removing all child movie clips, and setting all movie clip properties back to their default values.

To prove it, try the following code:

```
/*

This code fixes the onLoad bug, that is that anyMovieclip.onLoad event handler is deleted when loading external content into a movie clip:

*/

_global.s_onLoad=function(f)
{
    if(onLoadManager == undefined)
    {
        _global.onLoadManager={};
    }
    onLoadManager[this] =f;
}

_global.g_onLoad=function()
{
    return onLoadManager[this];
}

MovieClip.prototype.
addProperty('onLoad', g_onLoad, s_
onLoad);

this.createEmptyMovieClip("scrollingMovieclip",this.getNextHighestDepth());
scrollingMovieclip.onLoad=function()
{
    trace(this.cacheAsBitmap);
    //when the photo is loaded, show that
    bitmap caching was turned off
}

```

```
scrollingMovieclip.cacheAsBitmap=true;
scrollingMovieclip.loadMovie("photo.jpg");
```

Collision Detection

Finally, results from hit-testing code using `MovieClip.hitTest` will not be affected by bitmap caching, as hit-testing is still calculated based upon the vector data of a movie clip, not the generated bitmap that you actually see. To prove this, create a new movie clip and draw a circle inside it, drag it onto the first frame of the main Timeline, and give it an instance name of `circle_mc`. Now add the following code to the first frame of the main Timeline:

```
circle_mc.onMouseMove=function()
{
    trace("hit: "+this.hitTest(_root._xmouse,_root._ymouse,true));
}

circle_mc.cacheAsBitmap=true;
```

Where to Go from Here

As you can see, if used correctly, bitmap caching is a powerful tool for managing graphics performance. Here are some more resources for you to check out as you start playing around more with creating graphic effects using Flash 8.

- Flash Graphic Effects Center: www.macromedia.com/devnet/flash/graphic_effects.html
- Flash Graphic Effects Learning Guide: www.macromedia.com/devnet/flash/articles/graphic_effects_guide.html

As you become more advanced, there are a few more articles you should check out, including a couple that I've written:

- Introducing the Image API in Flash 8: www.macromedia.com/devnet/flash/articles/image_api.html
- Webcam Motion Detection: www.macromedia.com/devnet/flash/articles/webcam_motion.html

Also, Grant Skinner has written a couple of very useful articles you can dive into:

- Varicose-g Example: www.macromedia.com/devnet/flash/articles/varicose_g.html
- Using Bitwise Operators to Manipulate Bits and Colors: www.macromedia.com/devnet/flash/articles/bitwise_operators.html

Grant and I also both regularly update our blogs, which contain useful information for Flash developers:

- My FlashGuru Consulting Blog: www.flashguru.co.uk/
- Grant Skinner's Blog: www.gskinner.com/blog/

Finally, Flash engineer Tinic Uro often posts critical information for those developers who want to get inside what makes Flash tick and how to best take advantage of it:

- Tinic Uro's blog: www.kaourantin.net/

I hope this article helps you manage your memory and performance and gets you on the way to optimizing your frame rates. Stay tuned to the Developer Center (www.macromedia.com/devnet/) and to my website (www.flashguru.co.uk/) for more articles about Flash and graphic effects.

This article originally appeared in the Macromedia Developer Center, www.macromedia.com/devnet/flash/articles/bit-map_caching.html.

Guy Watson is the managing director of FlashGuru LTD, a Flash development consultancy company based in London, England. Guy has delivered presentations at various Flash industry events worldwide and has also won numerous industry awards for his work. He maintains the popular Flash resource website, FlashGuru's Knowledge Base (<http://www.flashguru.co.uk/>). www.flashguru.co.uk

Welcome to the Future

REGISTER NOW!
www.iTVcon.com

CALL FOR PAPERS NOW OPEN!

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

of Video on the Web!

iTVCON.COM
INTERNET TV CONFERENCE & EXPO 2006

Coming in 2006 to New York City!

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



**For More Information, Call 201-801-8023
or Email itvcon@sys-con.com**

Welcome to the Future!

Did you already purchase your “.tv” domain name?
You can’t afford not to add Internet TV to your Website in 2006!

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

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

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

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

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

PRODUCED BY
SYS-CON
EVENTS

List of Topics:

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

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



Flash Is Your Friend in Web 2.0

High order bit
by kevin lynch

I recently presented a “high order bit” at the Web 2.0 conference about how Flash and HTML work together and made some announcements about building Flash applications in a way developers can relate to more easily.

Tim O’Reilly wrote a great overview of what he means by Web 2.0, which are essentially design patterns and business models for the next generation of software. The shorter term results are things like easier sharing of photos with your friends and family, finding the most interesting things to read, and getting new insights on information by combining data like rental listings with their locations on a map. Longer term, this “architecture of participation” could mean harnessing collective intelligence across the Internet to solve increasingly difficult problems around the world, as also envisioned by Doug Engelbart.

In terms of building applications for Web 2.0, I believe the key underlying theme is the separation of data and user

interface through open data formats, RSS/Atom feeds, and programming interfaces made publicly available. This enables not only a revolution in machine to machine communication, as all the excitement about web services has been about, but also human to machine as we’re seeing with remixing applications and new user interfaces on data.

There is clearly a resurgence in how HTML can be used to deliver application user interfaces and terrific progress has been made on that. In addition, Flash brings capabilities that HTML doesn’t currently have, and they can be used together to great benefit – in fact, Flash has already been architected to fit perfectly in the Web 2.0 model. For example, Adaptive Path has been working on a great new application called MeasureMap that helps people track traffic on their blogs and is being built with a combination of HTML and Flash on the client. Another is how Flickr is using both HTML and Flash, for example implementing the organizer and slideshow with Flash

and the photo index with HTML. The language in Flash is ActionScript which is the same as JavaScript, both ECMA standard languages, and it’s very simple to call between code in HTML and Flash, enabling smooth integration with a free open-source integration kit. This is not about Flash vs. HTML or Ajax. It’s using Flash + HTML with the Ajax approach to build Web 2.0 applications (to be fully buzzword compliant).

There are many examples of applications built in Flash, though building them is not what a lot of people would call easy as the current Flash authoring tool and programming model were optimized for more creative uses such as animations. What we’re working on now is a set of technologies to make building applications and components much easier and for the results to run much faster.

These technologies include the upcoming Flash Player 8.5 which has a new virtual machine that runs several times faster – it has been in development for over two years and is aimed squarely

“There is clearly a resurgence in how HTML can be used to deliver application user interfaces and terrific progress has been made on that”

“The new Flash Player 8.5 has even stronger enterprise data connectivity, including client support for Flex Enterprise Services”

at providing a high performance Web 2.0 client runtime with a just in time compiler, runtime error checking, support for E4X (which makes XML a first class data type in scripting so you can easily use XML in code) and compliance with the standard ECMA language definition. The second technology is the Flex Framework, a programming model that enables developers to use an XML based language to build apps much more intuitively along with integrated scripting. And third, a tool called Flex Builder that is being developed on the Eclipse open source framework, is designed for developers, combines code editing with visual layout of applications, and has a compiler built right in. For a demo of how quickly you'll be able to build applications with this, please check out the video where you can see a photo search app built in about five minutes.

A major advantage of using the Flash Player for Web 2.0 applications is consistent development across operating systems and browsers and a lot less overhead programming around differences and needing to debug and test on every configuration. The Flash Player has more reach than any browser or operating system, and is being distributed faster than any other technology I know of on the Internet today which means innovation on client technology can be deployed to over 80% of people on the Web in about a year and then reach 98% a little while later.

This transformation of Flash from purely an animation engine to a runtime for rich media and rich internet applications has been happening for several years now, though many people aren't yet aware of these capabilities. Some things I find many people don't real-

ize about Flash include: an active open source community around Flash; support for more sophisticated local storage than what the browser provides along with more control for users; Google does in fact index Flash; two-way live audio/video communication is built in along with support for synchronized data transmission for collaborative apps; Flash supports accessible applications including integration with screen readers; it's actually the most widely distributed video player on the web; it supports sockets (which enable push of data from servers rather than the request-only model) in either XML or with the new client in any binary format.

Also, there is currently support for integration with backend infrastructure through web services or REST interfaces from the Flash Player today. The new Flash Player 8.5 has even stronger enterprise data connectivity, including client support for Flex Enterprise Services, which enables use of message queues, integration with JMS, remote procedure calls, and data synchronization. This enables not only simple applications like photo viewers, but also sophisticated business applications.

We're making this next generation available in early stages so we can collaborate with the community around it and make sure we're all building the right stuff. Alpha releases of Flash Player 8.5 and Flex Builder will be available for hacking Web 2.0 applications starting at the MAX developer conference on October 17 and posted at macromedia.com/go/web2.

This blog is reprinted with the author's permission and appeared originally on www.klynch.com.

As senior vice president and chief software architect, Kevin Lynch leads Adobe's Platform Business Unit, which is focused on advancing the company's software platform for the creation and delivery of engaging applications and content to any desktop or device. Lynch is responsible for the company's ubiquitous Portable Document Format (PDF), Adobe® Reader®, and Macromedia® Flash® Player, as well as alignment of Adobe's servers and tools with the company's technology platform. Lynch also oversees Adobe's developer relations program, including the integration of customers and partners in the development process through Adobe Labs and customer advisory councils.

Lynch joined Adobe through the company's 2005 acquisition of Macromedia, Inc., where he served as chief software architect and president of product development. He headed up the creation of the company's mobile and devices group and served as general manager of the web publishing group. Lynch also oversaw the initial development of Macromedia Dreamweaver®, a leading web development product.

Before joining Macromedia in 1996, Lynch worked for General Magic, where he pioneered a navigational user interface for handheld communicators. Previously, he designed the user interface and developed the first Macintosh release of FrameMaker® software for Frame Technology, later acquired by Adobe. While at the University of Illinois, Lynch developed early Macintosh applications, including a desktop publishing program that introduced user interface elements in common use today.

Lynch holds three patents with others currently pending, and he is actively involved in Adobe's international standards efforts with organizations such as the W3C, ECMA and ISO. In 2003, he was named one of CRN's "Top 25 Innovators," and was honored as one of the "First Annual Web Innovators" by CNET in 1998. Lynch studied interactive computer graphics at the University of Illinois, working with artists and engineers in the Electronic Visualization Laboratory.



Programmer's Guide to Fireworks Commands

Getting started with your first command
by **dustin dupree**

As with almost any program you work with, there are going to be things you wish you could change, add, improve, or remove. It might be a feature you are used to from another program, or one you have devised on your own. "If only I could run to the developers and tell them my great idea, the program would be perfect," you say. I've often wanted to change things in my favorite software but am usually stuck waiting and hoping it will be fixed in the next version.

Thankfully, sometimes there are alternatives. Extending Fireworks enables you to create new features, tweaks, and modifications. They can be as simple as setting a few preferences or as complex as being an entirely separate application. There are several ways to extend Fireworks but they all work towards the same goal: extending the capabilities of Fireworks and making it work the way you want it to.

This article is intended for someone with an interest in making commands for Fireworks. It is a guide to help anyone who has even a small amount of programming knowledge to learn the procedure for creating commands. The first part of the article explains what commands are, where they are stored, and the different ways you can execute them. It also explains some of the basic concepts and structures of most Fireworks commands. The middle part examines a practical command example and discusses where to look for more help. The last section provides some advice for troubleshooting your commands as well as some information about packaging and distributing them. After completing this article you should have a good idea of what is possible with Fireworks commands and have an understanding of how to create and package them.

Getting Started with Your First Command

The fastest and simplest way to extend Fireworks is by creating commands. These are JavaScript files that give you control over the internal functions of Fireworks, extending the capabilities of the program to suit your needs. You control Fireworks by calling a set of custom JavaScript functions from within your commands using the Fireworks Application Programming Interface (API). Commands work by utilizing the Fireworks API to control the functionality of the program. They tap into all Fireworks tools and features and allow you to control them through code. View the Fireworks 8 API documentation to get a more complete overview of the Fireworks Object Model and the custom JavaScript functions that are built into Fireworks.

Now that you have a basic understanding of what commands are and how they work, it's time to get started making some of your own.

Hello World

The first command you make will simply create a dialog box in Fireworks that says "Hello World!" If you are familiar with JavaScript, then the `alert()` function is nothing new to you, but you will still learn where to store your custom commands and how to run them. First open Fireworks and create a new document, approximately 500 pixels wide and 400 pixels tall.

Create a new text file in your text editor of choice. Place this Fireworks command in it:

```
alert("Hello World!");
```

That's it! Now save the command and you're done.

Saving Your Commands

There are a couple options available for saving a command. If you are on a single-user machine or want everyone who uses your computer in a multiuser environment to be able to access the command, save it into your installation folder's Configuration\Commands sub-folder:

```
(Windows) C:\Program Files\Macromedia\Fireworks 8\Configuration\Commands
```

```
(Mac OS) Macintosh HD:Applications:Macromedia:Fireworks 8:Configuration:Commands
```

If you want to limit access to just to your user name, save it here:

```
(Windows) C:\Documents and Settings\  
<User Name>\Application Data\  
Macromedia\Fireworks 8\Commands
```

```
(Mac OS) Macintosh HD:Users:<User>:
```

```
Library:Application Support:  
Macromedia:Fireworks 8:Commands
```

Save the file as `Hello World.jsf`.

Testing Your Commands

After your command is saved, switch to Fireworks to test it. If you go to the Commands pop-up menu, there should be a new entry for the "Hello World" command you just saved. Click on it now to run the command. If you see a message box that says "Hello World!" then the command ran successfully. If you see a message box that says "Can not run the script. An error occurred." that means there is a problem with the script and you should switch back to your text editor and make

sure you wrote the code correctly.

There are a couple of other options available for running commands in Fireworks:

- You can drag and drop .jsf files into Fireworks to run them; just make sure you drop them anywhere except the active document workspace.
- In Windows you can run commands by setting .jsf files to open with Fireworks (assuming they aren't already set by default). Right-click the "Hello World.jsf" file in the Fireworks 8\Configuration\Commands folder and select Open With from the context menu. A dialog box should open, showing Fireworks as the recommended application. If not, click the Browse button to find Fireworks.exe in your installation folder (usually it's C:\Program Files\Macromedia\Fireworks 8). Make sure to select the option that says "Always use the selected program to open this kind of file." Now simply double-click "Hello World.jsf" to run it on the active Fireworks document.

Using the Fireworks API for Your Second Command

Hello World wasn't all that much fun but at least now you know how to create, save, and run a command. This time let's utilize some more interesting functions in the Fireworks API to make a simple command that contains more Fireworks-specific code.

Hello Universe

Make a new document in your text editor and save it to the commands folder as "Hello Universe.jsf". Copy the following command code:

```
var dom = fw.getDocumentDOM();
dom.addNewRectangle({left:10, top:10,
right: 150, bottom: 45}, .25);
fw.selection[0].pathAttributes.fill-
Color = "#FF0000";
var stored_selection = new Array();
stored_selection.push(fw.selec-
tion[0]);
dom.addNewText({left:25, top:20,
right: 140, bottom: 30}, false);
var tr = fw.selection[0].textRuns;
tr.textRuns = [{changedAttrs:
{fillColor:"#FFFFFF", size: "16pt"},
characters: "Hello Universe!"}];
fw.selection[0].textRuns = tr;
stored_selection.push(fw.selec-
```

```
tion[0]);
fw.selection = stored_selection;
dom.group();
```

The first line of code uses the getDocumentDOM() method of the fw object. You can think of the fw object as a code representation of the Fireworks application itself. It contains the functionality and methods for the common tasks that Fireworks performs, such as Save Document, Save As, Export, and even Quit. The getDocumentDOM() method returns the Document Object Model (DOM) for the active document.

The document's DOM is stored into the dom variable to be used later in the command. The document's DOM can be thought of as a code representation of the active document in Fireworks. Any action you would perform on a document in Fireworks is handled by methods of the document's DOM, such as creating a rectangle, adding a text object, and arranging layers.

Once the document's DOM is stored into a variable, you can start using it to control the active document. In the second line of code, the addNewRectangle() method creates a new rectangle in the current document. The first argument being passed is an object conforming to the Rectangle data type, as specified in the Fireworks 8 API documentation. The Rectangle data type provides the information to place the rectangle (top, left) and define its dimensions (right, bottom). The second argument defines the roundness of the rectangle's corners—a number between 0 (no roundness) and 1 (maximum roundness).

Working with Selections

Working with selections is a very important part of writing Fireworks commands. Selections enable the users of your command to choose which object(s) inside a document to affect, and they enable you as a developer to work with the objects that you create programmatically. When a command is invoked, the fw.selection array can be used to access each selected object inside the active document. At any given point in the command, the fw.selection array holds the currently selected objects, whether there are zero or 20. When a new object is created, like the rectangle created in line 2, it is automatically selected.

It is easy to change the properties of a selected object by accessing them through fw.selection[0], as in line 3 where the rectangle's fill color is set to bright red. After creating a new Text object on line 6, it can be manipulated the same way by accessing it with fw.selection[0].

You can set a selection by assigning an object to fw.selection. To select multiple objects, create an array like stored_selection on line 4. You can add objects to the array and easily select the object's later in the command by overwriting the fw.selection array with the array of objects. There are other ways to make selections, too, by using the dom.selectAll() and dom.selectAllOnLayer() functions.

The API Documentation

If you haven't done so already, you should look over the Fireworks 8 API documentation. It would be a good idea to download a copy of the PDF version (ZIP, 1.2 MB) to your hard drive for reference purposes as you make your own commands. It explains all the objects and functions available to you as a Fireworks command developer. Here is a list of parts in the API documentation to pay close attention to:

- Fireworks Object Model > Objects Within Fireworks Documents: This section describes the objects that can be inside a Fireworks document. It is a great reference for looking up object structures. Pay particular attention to the "Notes" column where you can find more information about each property.
- Fireworks Object Model > Core Objects: This section includes object descriptions for the six core objects: Dialogs, Document, Errors, Files, Find, and System. These objects are accessible from anywhere within your command.
- Fireworks Object Model > Working with Selected Objects: This section covers what properties can be accessed from selected elements.
- Fireworks JavaScript API > Document Functions: This section lists all methods available to an instance of the Document Object Model retrieved by fw.getDocumentDOM(). The arguments section of each method description mentions any special objects that the method is expecting as an argu-



ment (see “Objects Within Fireworks Documents” for reference). These methods are specific to an individual document within Fireworks.

- Fireworks JavaScript API > Fireworks Functions: This section lists all methods of the fw object. These methods perform similar tasks to those found in the top-level menus in Fireworks, including saving documents, setting preferences, and exporting images.

Looking at Real-World Commands

After reading the Fireworks API documentation (or at least skimming through it) you should have a good idea of what you can achieve with Fireworks commands. Looking through the methods of the DOM and the fw object will show you exactly what functionality you have control over using commands. With a basic understanding of how commands work and what options are available, you can accomplish a lot.

Copy Primitive Fill

The Copy Primitive Fill command is a simple command that copies the fill color of a selected primitive to the Clipboard. It is called “Copy Primitive Fill.jsf” in the sample files. The command looks like this:

```
var dom = fw.getDocumentDOM();
if(fw.selection.length > 0) {
    var sel = fw.selection[0];
    if(sel.pathAttributes.fill) {
        copyToClipboard(sel.pathAttributes.fillColor);
    } else {
        alert("There is no fill color on this object!");
    }
    fw.selection = sel;
} else {
    alert("There must be an object selected to run this command.");
}

function copyToClipboard(txt){
    dom.addNewText({left:1, top:1, right:1, bottom:1}, true);
    var tr = fw.selection[0].textRuns;
    tr.textRuns = [{changedAttrs: {}, characters: txt}];
    fw.selection[0].textRuns = tr;
    dom.clipCut(fw.selection[0]);
}
```

The Copy Primitive Fill command begins by getting the active document’s DOM to use later on. After that it checks to see whether the user has made a selection. It then either executes the command or alerts the user that an object must be selected. This is a common structure for many commands that require a selection to be made before the command can be executed.

Sometimes things don’t go according to plan when you create a command. The Copy Primitive Fill command is a case in point. Because there is no copyToClipboard() function in the API, a function had to be created using a combination of whatever functions were available. As luck would have it, using clipCut() on a text object copies the object’s text to the Clipboard; otherwise this Copy Primitive Fill command wouldn’t be possible. For a more in-depth version of this command, which copies the fill from more complex objects, see the “Copy Fill Color.jsf” command in the sample files.

Learning by Example

One of the best ways to learn how to make a Fireworks command is by looking at the different ways that other people have done it. When you install commands from .mxd extensions, they are just copied into your Fireworks 8/Configuration/Commands folder as .jsf files. What this means is that you can open the .jsf files and view or edit the command’s code the same way you would your own! If you ever find yourself stuck creating part of a command, chances are good that someone has already found a solution for a similar problem. You might be able to find a better way to accomplish a certain task by looking at other people’s commands, even if those commands don’t necessarily produce the exact same results that you want, they may offer some alternative ideas or helpful pointers. If you are creating a command that requires grouping objects and you have used a command that also groups objects, it is very likely that the code for grouping objects can be found within the command.

Using the History Palette

One great way to learn a few

things about commands is by using the History palette. Many people who use it don’t realize how helpful it can be when writing your own Fireworks commands. Clicking the Copy Steps to Clipboard icon in the lower right corner of the History palette copies the actual code for performing the selected task to the Clipboard. If your command requires something like creating a slice, then instead of digging through the API to find the correct function, you can just create a slice in Fireworks, copy the “slice tool” entry from the history palette, and paste it into your code. After you know the proper function for inserting a slice, you can easily mold it to suit your needs. By holding down the Shift key, you can select multiple History palette entries and copy them to the Clipboard. This is a great way to start creating a new command or exploring how to reproduce tasks in your commands.

Troubleshooting Your Command

One thing that Fireworks lacks is a decent error-reporting mechanism for testing commands. Instead, a script error simply generates the very uninformative message, “Can not run the script. An error occurred.” After looking over your code to make sure there are no syntax errors, you have to start troubleshooting. Here are a few tips to help you troubleshoot your code.

Commenting

Start by looking at unfamiliar functions and objects. Comment out suspected errors with the block comment tags (/* */) or single-line dual forward slashes (//) and then run the command again. If you find the error, the command should execute without any error message. If not, keep trying.

Revisiting the Alert Function

Use the alert() function on certain variables to make sure they are returning what you expect. This method isn’t always reliable by itself, however, because alert() won’t work if the code error still persists. But you can use it in conjunction with the commenting route just mentioned to find out exactly what is going wrong.

Fireworks Command Prompt

Senocular of senocular.com has created an amazing third-party tool called Fireworks Command Prompt that can help you troubleshoot errors. This Fireworks extension returns data from JavaScript code while inside the Fireworks environment. This is a helpful learning tool, too, because Fireworks contains some pretty complex object structures that take awhile to get used to.

Fireworks Command Prompt is great for testing if you need to make sure you are accessing properties correctly. Typing something like `fw.selection[0].pathAttributes` in the upper text box and executing it (pressing Control+Enter) returns the `pathAttributes` object of the selection for you to examine in the lower text box. This is often a much better solution than using the `alert()` function because you can isolate code without making a new command or worrying about unrelated errors stopping the `alert()` function from executing.

After you find the error, you can reference the API documentation to see what went wrong and how to correct it.

Packaging and Distributing Your Command

The best way to distribute commands is with an `.mxp` (Macromedia Extension Package) file that you can easily install and uninstall using the Macromedia Extension Manager. Not only does the `.mxp` file provide a nice little package that is easy to download and organize, it also ensures that the commands are installed correctly without the user having to work manually with `.jsf` files.

Setting up the Extension

You will find a file called `Blank.mxi` in the `C:\Program Files\Macromedia\Extension Manager\Samples\Fireworks` folder. You can use this sample XML file as a general guide for creating your `.mxi` file—which is what the Extension Manager uses to compile an `.mxp` file.

Make a copy of the `Blank.mxi` file and save it to the same folder as the Fireworks commands you want to package. After that you can open the file with your text editor and start to fill in the blanks. Most attributes should be self-explanatory; Macromedia has also provided some

comments to help you out. Read the Macromedia Extension Information File Format document if you need more specific information or help with XML.

The most important part of the `.mxi` file is the `<files>` tree. This is where you define the command files to include as well as their destination upon installation. You can use as many `<file />` nodes as you need, using one for each `.jsf` file you want to include. A sample `<file>` tree that packages three commands would look something like this:

```
<files>
  <file name="Command1.jsf"
    destination="$fireworks/configuration/
    commands/" />
  <file name="Command2.jsf"
    destination="$fireworks/configuration/
    commands/" />
  <file name="Command3.jsf"
    destination="$fireworks/configuration/
    commands/" />
</files>
```

For additional references, see Sample `.mxi` included in the sample files for the article. It is set up to generate an `.mxp` extension containing the sample scripts, which are also provided.

Packaging the Extension

After you finish editing the `.mxi` file, open the Macromedia Extension Manager and choose File > Package Extension. When the browse dialog box appears, find your `.mxi` file and open it. Another dialog box asks where you want to save the generated `.mxp` file. Choose an output folder and a name for your extension, and click Save. Now your `.mxp` file is ready for you to distribute or submit to the Macromedia Exchange!

Using Third-Party Tools

If you have trouble creating your own packages, there are a couple tools that can help you with the packaging process. MXI File Creator from Muzak (www.muza-kdeezign.com/mxi_creator/download.asp) and MXI Wizard (www.linecraft.com/products.php) from Alex July are both free applications that provide a GUI for creating your `.mxi` files. The previous section, "Packaging the Extension," explains how to finish the process once you create the `.mxi` file.

Where to Go from Here

Writing commands for Fireworks can be fun and rewarding. Even very simple commands can make working with Fireworks easier and more productive than before. Now that you know all the essential parts of creating a command, you can start writing your own. What parts of Fireworks do you want to extend? The possibilities are endless, and creating `.jsf` commands is only the tip of the iceberg. Commands are not the only way to extend Fireworks. Building more advanced extensions that require unique user input can be produced using Macromedia Flash in conjunction with the Fireworks API. You'll need to have a good understanding of the Fireworks API before you can start making Flash panel extensions, though. You can also use custom commands while batch-processing images if you want more control over the process. Here is a list of links to help you on your path to extending Fireworks, commands, and beyond:

- Extending Fireworks by Kleanthis Economou (good general guide that covers all the different ways to extend Fireworks)
- 10 Minutes with Flash: Creating a Custom Panel by Robert Hoekman (page that guides you through the creation of a simple Flash panel)
- Creating Macromedia Fireworks Auto Shapes by Senocular (excellent guide to creating Auto Shape objects)
- Senocular.com Fireworks Extensions (many examples and quality extensions)
- John Dunning Fireworks Extensions (some more great commands just waiting to be studied!)

This article originally appeared in the Macromedia Developer Center, www.macromedia.com/devnet/fireworks/articles/extending_fireworks.html

Dustin DuPree currently attends Milwaukee Area Technical College in Milwaukee, WI, studying visual communications/multimedia. He also freelances as a web developer and designer. His interests include Flash, Fireworks, programming, design, photography, playing guitar, computer gaming, film, politics, and reading. www.dujodu.com



Validation of Using DAO, DG, and VO

ColdFusion development
by scott barnes

In the more modern day ColdFusion Development, we tend to rely heavily on various Design Patterns to automate or “template” our daily coding chores. The patterns most frequently touched on are the Data Access Objects (DAO), DataGateway (DG) and VO (Value Object) – or – Bean (VO with setter/getter routines).

I’ve often used them, but can’t but help get the feeling that they are probably not suited for the ColdFusion Development world as to me; their reasons for being used aren’t valid enough to warrant using them blindly.

I mean, why Use a DAO?

I’ll attack DAOs first, as these are one of the core primitives found in most frameworks.

The concept of a DAO is as most already know, to allow the four key actions to take place (CREATE, UPDATE, DELETE, READ). In doing this, along the way we are supposed to accept “arguments” of each specific type (i.e., CustomerID is Numeric?).

If that succeeds, the logic to perform

one of the above actions commences, and as a result we have a success result.

Yet, what are we really achieving by having a DAO in place? In that is it about strict typed variables or is it about housing logic inside a CFC for safekeeping. At the very least that’s all we are really doing is preventing data that has the wrong type, being inserted into a database or xml packet.

What if we were to say, use an XML Schema to dictate this?

In that using a “TableAdapter” object, we feed in arguments or serialized object, which the TableAdapter dissects, validates and allows passing otherwise it throws an exception.

Immediately one thing springs to mind, in that it breaks encapsulation of knowing what variables are passed in and out of an object, in that they become “unknown.”

Which is true, how are we to know that CustomerID is String or Numeric?

We, the developer would look this up under the various documentation tools out there such as CFCDoc, which tells us

exactly what’s coming in and out (i.e., how to implement).

Yet, are we not getting in danger of coupling our entire business logic with say our database? What would happen if down the road we change CustomerID from Numeric to String (i.e., AutoNumber gets replaced with UUID) what then? How do we adjust our rules and policies to cope?

We could make a ruling, which simply states that all “ID” fields are “Strings” regardless, and only when they get tucked away inside our nominated database, that actual type conversion kicks in. Yeah, that’d work but where does that rule end?

This is probably a problem used in most other languages, and there are other ways to overcome this (i.e., overloading a method for one) – yet – we are in ColdFusion, not one of these languages.

This line of reasoning is the foundation of why I disliked in many ways the concept of DAOs as they are “too” specific in terms of data typing. It’s too easy to let

“The concept of a DAO is as most already know, to allow the four key actions to take place”

“Its hard post to swallow, I do admit that as every bone in your OOP bodies would be fighting tooth and nail that this is totally against the OOP grain”

the compiler take care of the “rulings” on what type of variable should be sent to the view.

How about an alternative solution? In that what if we put together an object, which we could ask should we need to know what type of variable it is?

In that, what if an object called “DataTable” existed, and inside that we had another object (composition) called “DataRow,” which essentially we ask to know what “type” a column is.

This concept is loosely borrowed from Windows .NET 2.0’s “DataSets.”

Essentially all your “Database” activities are stored inside an XML Schema, which essentially defines how and where data gets stored and what “type” they are expected to be stored in.

Furthermore, it’s not limited to “Databases.” It’s also used for “Files” (XML, CSV, MS Access etc.).

The point is this that by hard coding our persistence storage via DAO, we are really shifting our approach in terms of defining the rules required to be in place before data is persisted. If these rules change, it can be an achievement unto itself to go through code and adjust properties/arguments to suite.

Then there is the risk of having a mutipersist-based application, which results in duplication of code in the way of DAO just because there are different types for each persistence technology (MySQL versus Oracle, etc.).

Its hard post to swallow, I do admit that as every bone in your OOP bodies would be fighting tooth and nail that this is totally against the OOP grain, which I do agree, except we are in “ColdFusion” – a point that I cannot stress enough. ColdFusion isn’t sophisticated enough to cope with the rulings of its foundation (J2EE) unless

you introduce specific Java hacks. If you introduce JAVA into the equation, then yes we are in a whole new discussion, but most don’t use JAVA as their buffer between ColdFusion and a Database.

This brings me to my original question, why use a DAO? Same question goes for a DG, why Use them?

In most cases, what’s the difference between a “GetPersonByEmail” and “GetPersonByID” and how would you store that principal concept in a DG component? Smart answer would probably be two separate queries, with two separate “WHERE” statements in SQL terms.

Yet, what happens if we were to drop a field from the equation or better yet, introduce one?

Those of you who have used DGs and DAOs in large applications will know what I’m talking about, in that it’s “a lot of work” to get all those arguments in place throughout the code base.

To me, there is an alternative and I’m currently hacking away at it. It simply involves XML to describe how my persistence gets stored and what constraints are in place should they arise are declared via XML.

A Value Object though is a hard one, because they really only serve an “emulated” approach, in that the CFPROPERTY tag really a has purpose other than to describe what potentially the variables inside a CFC will be – yet – they don’t enforce such rulings.

Nope, to me ColdFusion is great, but it’s not advanced enough to warrant the patterns such as DAO and DG.

How’s that sit with you?

This blog is reprinted with the author’s permission and appeared originally on www.mossyblog.com/archives/553.cfm.

Advertising Index

Advertiser	URL	Phone	Page
ActivePDF	www.activepdf.com	866-468-6733	21
CFDynamics	www.cfdynamics.com	866-233-9626	17
Community MX	www.communitymx.com		25
Flashforward	www.flashforwardconference.com		15
InterAKT	www.interAKTonline.com/macromedia	4031 401.68.19	3
Intermedia	www.intermedia.net	888-379-7729	6
Macromedia	www.macromedia.com/go/8_studio8	415-252-2000	52
Metalq	www.metalq.com	415-642-3332	27
Savvy	www.besavvy.com	866-870-6358	21
Seapine Software	www.seapine.com/mxww	888-683-6456	5
Stream57	www.stream57.com	212-909-2550x1012	11
Vitalstream	www.vitalstream.com	800-254-7554	2
WebAppCaberet	www.webappcaberet.com/jdj.jsp	1-866-256-7973	51

Advertiser index is provided as an additional service to our readers. Publisher does not assume any liability for omissions and/or misprints in this listing since this listing is not part of any insertion order.



Using Captivate to Retouch Images

Recorded demos and simulations
by mark fletcher

have you ever worked on an e-learning project, only to find that you need to alter some of the screen action you have captured?

In projects I have completed recently, I found myself in this exact situation. Had I not discovered Macromedia Captivate, I would have had no choice but to either re-record a number of screens or the entire demonstration or simulation again – both of which would have been an extremely time-consuming affair.

In this article I show how you can update your Captivate project files using an external image editor and insert new image files into your Captivate content. I also explain the way Captivate handles mouse movements, which is crucial when performing these tasks.

For the last seven years, I spent the majority of my time creating video-based tutorials for VTC (Virtual Training Company) online and on CD. I've also created tutorials for Macromedia, Adobe Systems Inc., and the leading Macromedia extension developer, WebAssist. Historically I've always used an AVI recording tool to create my demonstrations. One of the problems that plagued me over the years, however, is how to alter the recorded screens.

Below are some examples of when I have found it necessary recapture a screen area:

- The recording application crashes while recording your screen action.
- Artifacts are present on certain frames in your movie.
- An object is present on a screen that shouldn't be there.
- The version of the application you are recording is still in development. You have created the movies and have sent them to the client for approval, only to find that there is a new feature that needs to be included, such as a new entry in a pop-up menu.
- You have an icon somewhere on the screen that needs to be removed, such as another vendor software application.
- The version number of the program has been updated and you need to change it or remove the version number altogether in the demonstration.
- You have typed some text on the screen, only to find that you have spelled a word or phrase incorrectly.

Attempting to change demonstrations using an AVI recording tool can be both a time-consuming and complex task. Most of the time this requires that you record the whole demonstration again. Even if you decide to try to record just the screens that need to be amended, you may still encounter mouse posi-

tion issues, which you may not be able to resolve successfully.

Why is this such a big deal? The main reason that editing demonstrations created using an AVI recording tool can be an arduous task is the way it captures your screens. Video files created using an AVI recording application are made up of many still images combined sequentially into one file. With video files, each screen with mouse action appears at a number of frames per second (typically a frame rate of between 5 and 10 frames per second), thus creating the illusion of movement.

Video files use a delicate balance of key and delta frames. Key frames contain all the information that is required to display the frame. Delta frames contain only what has changed from the previous frame. Remember that the more movement there is in a video file, the more area of the screen is altered, which results in larger delta frames and, therefore, a much larger resulting file size.

Some AVI recording tools give you the ability to overlay new graphics into your movies. However, because the mouse pointer has to be captured when the screen area is recorded, inserting a new graphic image will typically result in the pointer appearing behind the inserted image, so you are still left having to reshoot your movies!

“Attempting to change demonstrations using an AVI recording tool can be both a time-consuming and complex task”

“Macromedia Captivate uses a completely different approach. In general, when capturing a screen area, Captivate takes a series of static screen shots”

Capturing Screens Using Macromedia Captivate

Macromedia Captivate uses a completely different approach. In general, when capturing a screen area, Captivate takes a series of static screen shots. This means that most of the time you don't have to worry about capturing a screen area again. But what about the mouse pointer? Surely if Captivate takes static screen shots, it must include the pointer on each screen shot it takes, right?

Actually, no. Captivate does not include the mouse pointer when it takes a screen shot. Instead, during capture, only the position of the mouse pointer (among other things) is stored. Only after you have finished capturing your screens does Captivate add a fully editable mouse pointer and curved motion path to your Captivate project file. Captivate gives you an incredibly flexible working environment.

Editing Screens in Captivate

There are two ways you can make changes to the screens that make up your Captivate project files:

- Copy a background image and edit it in an image editor, such as Macromedia Fireworks MX 2004
- Overlay a new graphic file on top of the portion of the screen you wish to alter

Retouching Captivate Background Images in an External Image Editor

Being able to edit each of the background images that make up your Captivate project files gives you an incredibly versatile working environment. For example, one of the projects I just finished involved creating a series of interactive simulated tasks for the e-commerce software application WA eCart developed by WebAssist. Having created a Captivate project demonstrating how to install the

software, I received an e-mail requesting that one of the screens should be altered because it displayed the software version number, as shown in Figure 1.

Because Captivate enables you to copy the background image of a slide, altering a screen is incredibly simple:

1. Select the slide you wish to edit.
(Optional steps)
Because you cannot undo pasting an image background in Captivate, you may also want to do the following:
 - a. Select Slide > Duplicate Slide3.
 - b. Select the original slide and choose Slide > Hide Slide.
2. Select Edit > Copy Background for your duplicated slide.
3. Switch to an image editor such as Macromedia Fireworks MX 2004.
4. Select File > New in your image editor and leave the default values.
5. Select Edit > Paste.
6. Change the image as you deem necessary.
7. Save the file.
8. Select the entire image and choose Edit > Copy. (If you're in Fireworks, make sure you've selected the Pointer tool.)
9. Switch back to Captivate.
10. Select Edit > Paste as Background.
11. Save the Captivate file.

Using this copying slide background technique, I was able to completely remove all traces of “WA eCart 2.1.0” (see Figure 2).

Inserting New Images into Captivate Projects

As well as being able to alter the background image of each of your slides, Captivate also enables you to overlay a new image over a certain region of your captured screen area.

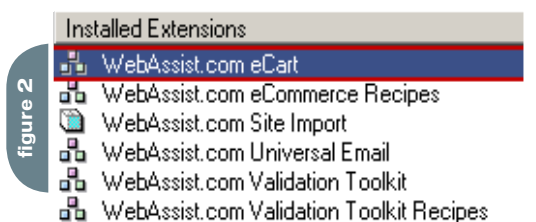
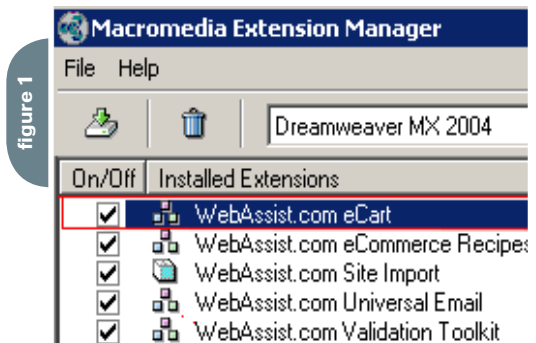
Again, while working on the WA

eCart project, the day the software and interactive simulations were going to be launched, I received another e-mail telling me that, at the last minute, a new feature (an e-mail friendly cart) had been added to the application which needed to be somehow incorporated into one of the existing tutorials. Figure 3 shows what the original screen looked like.

To resolve this issue, I simply took a screen shot of the new pop-up menu and placed this over the top of the existing menu. You can use a screen shot tool or simply Print Screen and crop the image (using an image editor such as Fireworks) to simply display the change. Once I was happy with the way it looked on all the relevant slides, I incorporated (merged) it into the slide background. See the result in Figure 4.

To insert a new image into a Captivate project, follow these steps:

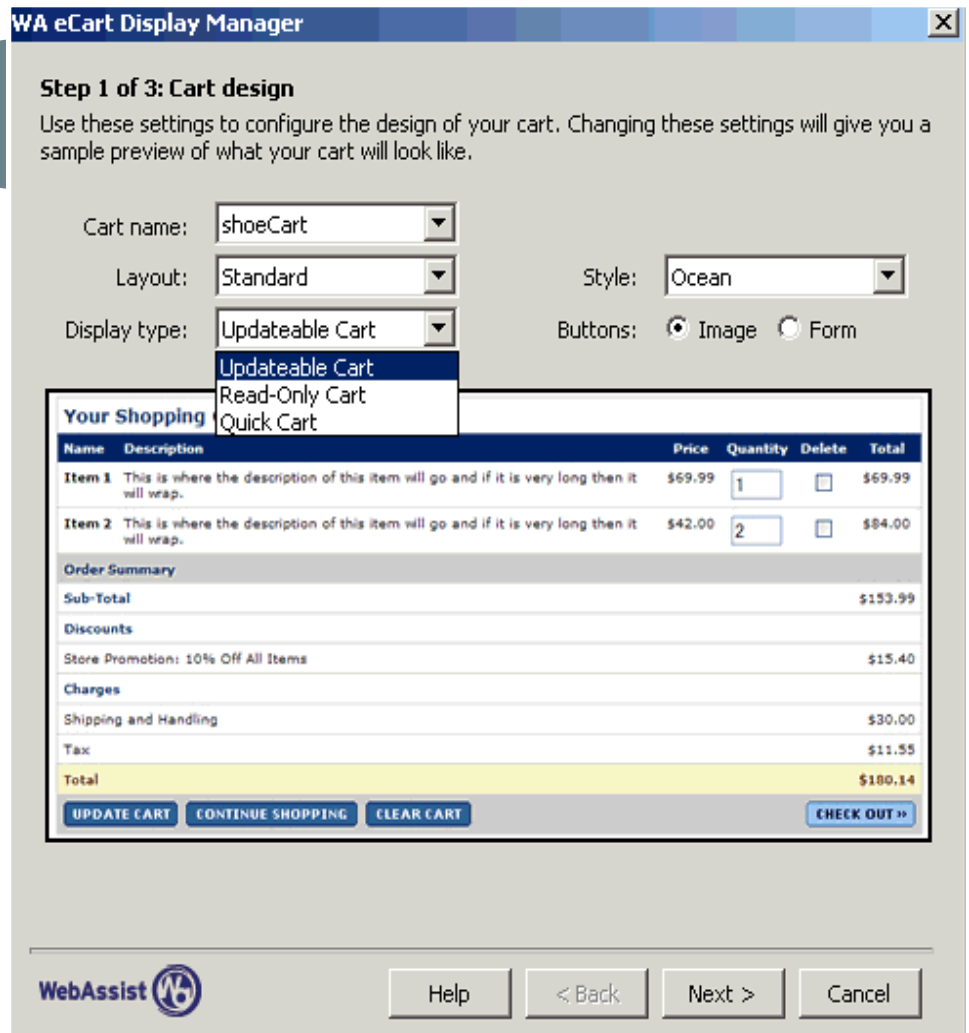
1. In Captivate, select the slide into which you wish to insert the image



t breaks the eCommerce barrier with unparalleled ch
perty designed shopping
These snippets ar



figure 3



2. Select Insert > Image

3. Browse to select the image you wish to insert

Note: All image file formats (apart from JPEG and GIF) are automatically converted to BMP when inserted into a Captivate project. JPEG and GIF are also converted to BMP if the image needs to be cropped or resized.

4. From the Image dialog box (see Figure 5), you can set the image properties.

The dialog box consists of three tabs:

- Image lets you change the appearance of the image
- Options enables you to set timing and transition effects for the image
- Audio lets you add audio to an image or edit audio that is already associated with an image

5. Insert and position the image in the desired location on each slide you

want to amend.

6. Preview the movie.

Merging Objects into a Slide

Merging images into the slide background can be very useful, especially when working with a large image. To do this, use the following steps:

- 1 Select the image object that you wish to permanently become a part of the background.
- 2 Select Edit > Merge into Background. A warning appears, stating that merging cannot be undone.
- 3 Click Yes to confirm the merge.

Editing Slides That Contain Mouse Movement

So far, all the examples you have seen consist of screens that don't include any mouse movement. Attempting to edit movies that include a mouse pointer using a traditional screen capture tool is probably one of

the most complex tasks you are likely to encounter.

In this section you will see how Captivate deals with this troublesome issue and the problems that result when attempting to record a demonstration using a screen-capture tool such as Camtasia Studio 3.0.

Consider the following scenario: You create a demonstration that includes both mouse movement and a mouse click to show a pop-up menu. You then find that there are a number of items present (in this case, the FlashPaper tool bar in a Microsoft Word document) that you must remove from the project.

In this test case, I recorded a project in both Captivate and Camtasia Studio 3.0 from TechSmith. The aim of this demonstration is for you to learn how to apply a heading style to a block of selected text, using the Style menu in Word:

1. Select a block of text in a document
2. Choose a heading style from the Style

ENGAGE AND EXPLORE...

The Technologies, Solutions and Applications that are Driving Today's Initiatives and Strategies...

CALL FOR PAPERS **NOW OPEN!**

SOA 10th International
WebServices
Edge conference+expo
Edge 06



The Sixth Annual SOA Web Services Edge 2006 East - International Web Services Conference & Expo, to be held June 2006, announces that its Call for Papers is now open. Topics include all aspects of Web services and Service-Oriented Architecture

Suggested topics...

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

CALL FOR PAPERS **NOW OPEN!**

2006
ENTERPRISE>
OPENSOURCE
CONFERENCE+EXPO



The first annual Enterprise Open Source Conference & Expo announces that its Call for Papers is now open. Topics include all aspects of Open Source technology. The Enterprise Open Source Conference & Expo is a software development and management conference addressing the emerging technologies, tools and strategies surrounding the development of open source software. We invite you to submit a proposal to present in the following topics. Case studies, tools, best practices, development, security, deployment, performance, challenges, application management, strategies and integration.

Suggested topics...

- > Open Source Licenses
- > Open Source & E-Mail
- > Databases
- > ROI Case Studies
- > Open Source ERP & CRM
- > Open-Source SIP
- > Testing
- > LAMP Technologies
- > Open Source on the Desktop
- > Open Source & Sarbanes-Oxley
- > IP Management

Submit Your Topic Today! www2.sys-con.com/events

Sponsored by

WebServices JOURNAL

XML JOURNAL

NET JOURNAL

eclipse developer's journal

WebSphere JOURNAL

information STORAGE+SECURITY journal

wildj

JDJ

LinuxWorld

MX developer's journal

asp.netPRO

SDTimes

CoDe

Software Test & Performance

*Call for Papers email: jimh@sys-con.com



Attention Exhibitors:

An Exhibit-Forum will display leading Web services and OpenSource products, services, and solutions

For Exhibit and Sponsorship Information > **Call 201 802-3066**

Produced by **sys-con** EVENTS

© 2005 WEB SERVICES EDGE. ALL RIGHTS RESERVED



figure 4

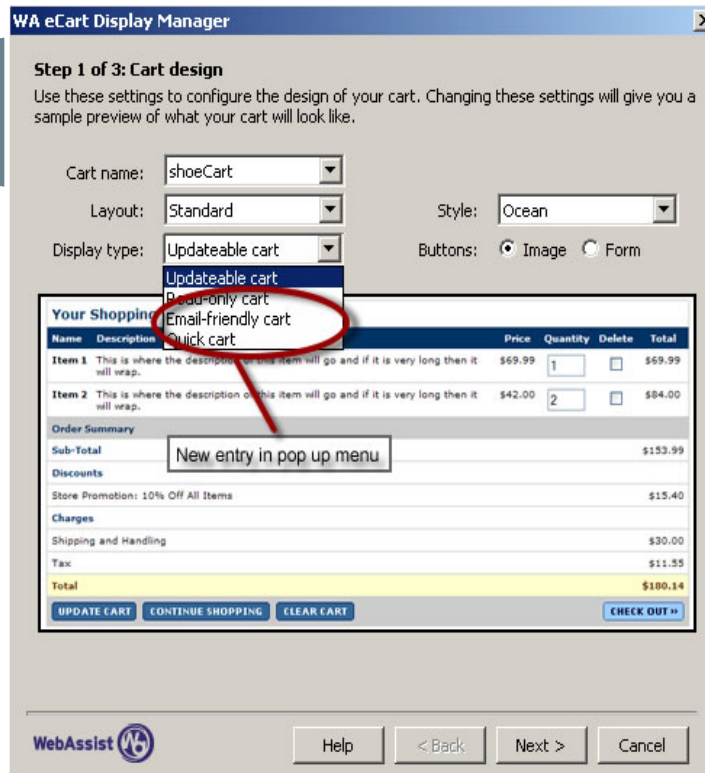
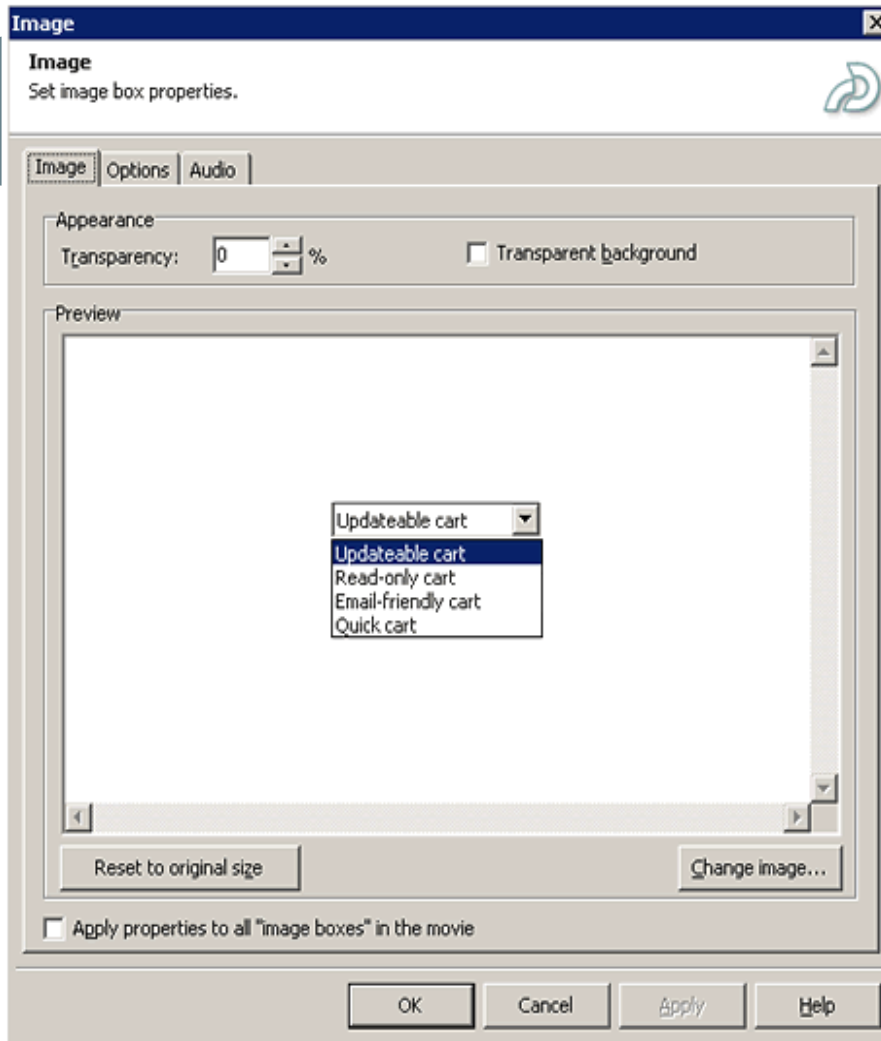


figure 5



- menu
- 3. Save the document

Remove the FlashPaper tool bar by inserting a new graphic. Because the demonstration took less than one minute to capture, the task should be completed in less than two minutes' time.

Findings with Camtasia Studio 3.0

In Camtasia Studio 3.0 I was able to insert a new graphic image using a feature called custom callouts. However, this meant that I had to insert multiple custom callouts on the Timeline. This was time-consuming and became extremely frustrating because I couldn't use the cursor keys to reposition my graphic elements.

With Camtasia Studio 3.0 the process took over 12 minutes and yet there were still some instances when the FlashPaper tool bar was visible. However, a much bigger problem was when the mouse had to travel over the newly added graphics. Because Camtasia is a screen-capture application that includes the mouse at capture time, attempting to overlay a new image results in the mouse pointer traveling behind the image rather than in front of it, as shown in Figure 6. Because this made the demonstration look unrealistic, I would have had no choice but to record the whole task again.

Findings with Macromedia Captivate

Using Captivate I was able to correct my project easily by overlaying the new tool bar image onto each of my slides. In Captivate the mouse movement is a different layer, so I can choose whether the mouse travels behind or on top of the inserted tool bar image. When the mouse traveled past the new graphic, I had only to make sure that the image object was below the mouse object on the Slide Timeline, as shown in Figure 7.

Below are the steps I followed to remove the FlashPaper tool bar in Captivate:

1. Select Image > Insert.
2. Reposition the graphic using the cursor keys, which allows pixel-level precision.
3. Move the graphic so that it is below the mouse object on the Slide Timeline.

figure 6



4. Select the graphic and show the image by removing fade in/out effects.

Note: The image transition must be set to None. Otherwise it will be apparent that a new image has been inserted on top of the original region of the screen you are trying to alter.

5. Copy the inserted image.
6. Select the next slide to which you want to add the image.
7. Select Edit > Paste.

Note: When copying an image, Captivate captures the position of the graphic. You don't need to go through the process of repositioning it again, nor is it necessary to change its position on the Slide Timeline.

8. Repeat Steps 4 and 5 on the desired slides.

Success! Using Captivate I completely removed all traces of the FlashPaper tool bar. The whole process took just over 44 seconds and, most importantly, the mouse pointer travels in front of the inserted graphics, as you can see in the Captivate movie.

Conclusion

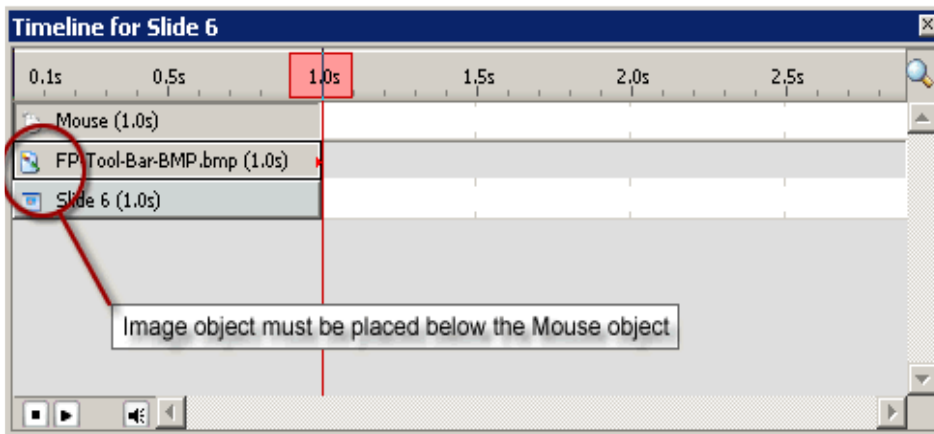
In this article I covered how you can create demonstrations and interactive simulated content, safe in the knowledge that you can change virtually any screen at any time, even if your screens include complex mouse movement. With Macromedia Captivate, having to recapture a screen is finally a thing of the past.

This article originally appeared in the Macromedia Developer Center, www.macromedia.com/devnet/captivate/articles/retouch_imgs.html.

Mark Fletcher is one of the most experienced authors at the Virtual Training Company, where he specializes in creating training CDs and online tutorials on Macromedia products, such as UltraDev, Fireworks, HomeSite, Sitespring, and Dreamweaver, including Dreamweaver MX Fundamentals, Dreamweaver MX 2004 Fundamentals, and Dreamweaver MX Web Applications. Mark also develops online tutorials for WebAssist.com, a Dreamweaver and UltraDev extension developer. www.vtc.com, mark-fletcher.co.uk

“With Camtasia Studio 3.0 the process took over 12 minutes and yet there were still some instances when the FlashPaper tool bar was visible”

figure 7





Using the Flex Trace Panel

Developing Flex applications
by dirk eismann

Since a Rich Internet Application is not a stateless client, you typically have a lot of data in the client, especially when you load dynamically over time. Often, while developing Flex applications, you might need to peek at data structures during runtime—this is where my utility, the Flex Trace Panel, comes in handy.

Requirements

To make this most of this tutorial, you need to install the following software and files:

- Flex 1.5
- Flex Trace Panel and accompanying files (which can be found at <http://macromedia.com/devnet/flex/articles/tracepanel.html>)

Prerequisite Knowledge

This article assumes that you are familiar with the basics of Flex.

The Problem

When you debug a web application that uses HTML in the presentation tier, you typically write debug messages to either the standard log file of your application server or directly into the HTML output to track the flow of your application or to inspect data. For Flex-powered applications you can leverage the Flex Debugger integrated in Flex Builder 1.5 to inspect your application. However, there are situations where you do not want to run your application under the debugger or inside Flex Builder 1.5 but directly in the browser. In those cases you must use the `trace()` function or add some sort of sophisticated debug text area to the application to output debug messages.

While the `trace()` statement works fine for simple debugging tasks, one major

drawback of it is that there is no console window to display the messages. Instead, you must browse your local machine for the log file that Flash Player writes to when executing a `trace()` statement. This may not even work at all if you haven't installed the Flash Debug Player and configured it to enable file logging. Finally, to get some sort of real-time logging experience you must open the log file with some text editor that automatically reloads the file after its content change.

One other technique is to add a logging console to the application itself. This may work, but can compromise the design of your application's user interface.

Introducing the Flex Trace Panel

To improve the debugging experience, I wrote a simple but powerful add-on for your daily Flex development needs: the Flex Trace Panel. It's a standalone application that runs outside the browser and listens for debug messages sent from your Flex application. Received messages display in real time. There are several features such as different log levels and nested object introspection that help you develop and debug your Flex application. Technically, the Flex Trace Panel is a SWF file made with Macromedia Flash Professional 8, wrapped into a standalone shell by using the third-party tool Zinc v2.5 by Multimedia Limited.

Once you start the Flex Trace Panel, it creates a new `LocalConnection` instance and listens for incoming messages. After the Flex Trace Panel receives a message, it writes the data to the output text area. If the data is complex (such as an object), the Trace Panel recursively introspects the structure and prints the hierarchy while avoiding cyclical references that could

cause the Trace Panel to freeze. In short, your Flex applications send data to the Flex Trace Panel by using the Dumper class, an ActionScript 2.0 class that comes with the Flex Trace Panel download in the Requirements section.

Using the Flex Trace Panel

To use the Flex Trace Panel, unzip the downloaded ZIP file to a directory of your choice. The directory contains an `src` and a `bin` folder, with the Flex Trace Panel executable in the `bin` folder. You may want to copy this file to another location on your machine.

The `src` folder contains a directory structure for the Dumper class. Copy the contents of the `src` folder to your Flex server, so that the Dumper class is available to your Flex application. In general, place the Dumper class in the class path of your Flex application. To make the Dumper class available to all of your Flex applications (recommended for a testing or development system) simply copy the directory structure to the `user_classes` directory of your Flex server. With a default, out-of-the-box Flex 1.5 installation, this is at:

```
C:\Program Files\Macromedia\Flex\
jrun4\servers\default\flex\WEB-INF\
flex\user_classes
```

If you only want to use the Dumper class in a single application, copy the `de.richinternet.util.*` package to the root of your Flex application. After you install the source files, it's time to start the show.

Create a new Flex application and add the following code:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.
macromedia.com/2003/mxml">
```


A new tool for MX professional developers and designers...



ADVERTISE

Contact: Robyn Forma
robyn@sys-con.com
(201) 802-3022
for details on rates
and programs

SUBSCRIBE

[www.sys-con.com/
mx/subscription.cfm](http://www.sys-con.com/mx/subscription.cfm)
1 (888) 303-5282





```

<mx:Script>
<![CDATA[
    import de.richinternet.utils.
Dumper;

    private function sayHello():
Void {
        Dumper.dump("Hello World!");
    }
}]>
</mx:Script>
<mx:Button label="Say Hello!"
click="sayHello()" />
</mx:Application>

```

Start the Flex Trace Panel and run your Flex application. When you click the button, the following message will appear in the output window.

The [INFO] text printed left to the message indicates the log level of the message, while (String) tells you the type of data sent. This is extremely helpful when passing complex data such as Arrays or nested Objects to the Trace Panel.

As mentioned before, you can set up your applications to send any data type to the Flex Trace Panel. For example, if you want to output the content of the DataProvider attached to the DataGrid myGrid you simply write:

```
Dumper.dump(myGrid.dataProvider);
```

The Dumper class provides several methods to send data with different log levels to the Trace Panel. This is convenient if you for example want to indicate whether a certain message is an error message or just simple information. The log levels are:

- INFO
- WARN
- ERROR

Here's a listing of all available methods of the Dumper class:

- Dumper.dump(message:Object):Void takes the passed Object (any type) and routes it to the Flex Trace Panel with log level INFO
- Dumper.info(message:Object):Void takes the passed Object (any type) and routes it to the Flex Trace Panel with log level INFO
- Dumper.warn(message:Object):Void takes the passed Object (any type) and

- routes it to the Flex Trace Panel with log level WARN
- Dumper.error(message:Object):Void takes the passed Object (any type) and routes it to the Flex Trace Panel with log level ERROR

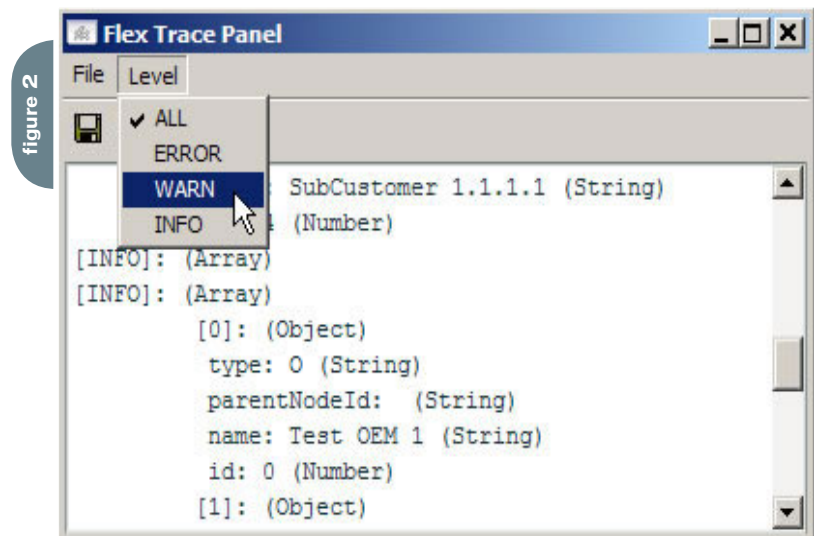
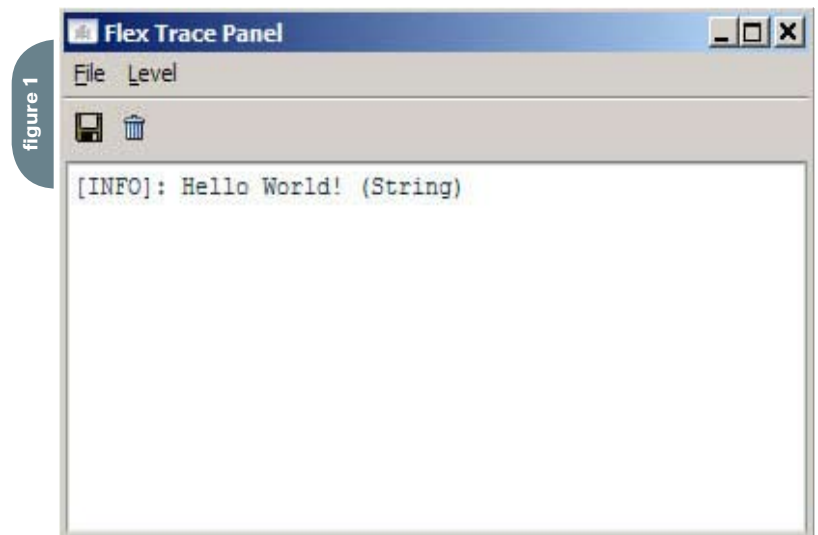
To distinguish between messages with different log levels, you can use the Flex Trace Panel to filter messages on their log level. By using the Level menu of the Trace Panel, you can filter all received messages. Internally, the Flex Trace Panel uses a message buffer to filter messages dynamically.

To save the content of the output text field to a text file, click the save icon or select Save from the File menu. To clear the text field, select File > Clear or click the trash can icon.

This article explained how you can use the Flex Trace Panel while you develop Flex applications. As with any other

software, the Flex Trace Panel will keep evolving. Especially, I'd like to update it so that it runs with Flex 2 and the new Flex 2 Logging API. Stay tuned on developments with the Flex Trace Panel category on my blog.

Dirk Eismann is a software engineer at the Hannover, Germany-based software company Herrlich & Ramuschkat where he develops and implements web-based applications. Currently he focuses on Flex application architecture and the integration of Flex applications into Java and .NET environments. Dirk has been working with Flex since it hit the streets and now fluently speaks ActionScript 2.0 and MXML. He is also a Macromedia Certified Instructor for Flex and Flash, and an active contributor to the Flex community. He runs richinternet.blog, a blog dedicated to Flash-powered Internet applications. www.richinternet.de



Complex JAVA J2EE Hosting made easy.

WebAppCabaretsm

<http://www.webappcabaret.com/jdj.jsp>
1.866.256.7973



JAVA J2EE-Ready Managed Dedicated Hosting Plans:

Xeon I

SAMEDAY
SETUP

Dual 2.8 GHz Xeons
2GB RAM
Dual 73GB SCSI
1U Server
Firewall
Linux
Monitoring
NGASI Manager

\$279
monthly

Pentium 4 I

SAMEDAY
SETUP

FREE
SETUP

2.4 GHz P4
2GB RAM
Dual 80GB ATA
1U Server
Firewall
Linux
Monitoring
NGASI Manager

\$199
monthly
2nd month
FREE

4Balance I

1 Database Server
and 2 Application
Servers connected
to 1
dedicated
load
balancing device.
Dual Xeons.
High-Availability.

\$1724
monthly

At **WebAppCabaret** we specialize in **JAVA J2EE Hosting**, featuring **managed dedicated servers** preloaded with most open source JAVA technologies.

PRELOADED WITH:

JDK1.4 . JDK1.5 . Tomcat . JBoss . Struts . ANT . Spring . Hibernate
Apache . MySQL . PostgreSQL . Portals . CRM . CMS . Blogs . Frameworks
All easily manage via a web based control panel.

Details:

- All Servers installed with the latest Enterprise Linux
- Firewall Protection
- Up to 60 GB daily on site backup included at no extra charge per server.
- Database on site backup every 2 hours
- Daily off site database backup
- A spare server is always available in case one of the server goes down
- Intrusion detection.
- 24x7 Server and application monitoring with automatic self healing
- The Latest Bug fixes and Security updates.
- Tier 1 Data Center. 100% Network Uptime Guarantee
- Guaranteed Reliability backed by industry-leading Service Level Agreements

Log on now at <http://www.webappcabaret.com/jdj.jsp> or call today at
1.866.256.7973

WebAppCabaretsm

JAVA J2EE Hosting

Prices, plans, and terms subject to change without notice. Please log on to our website for the latest price and terms. Copyright © 1999-2005 WebAppShowcase • All rights reserved • Various trademarks held by their respective owners.





WEB DEV GURU seeks an Integrated Software Suite that speaks my language (XHTML) and won't cramp my style. Must play well with XML, CSS and others. I'm not superficial, but I like a nice code view. Clutter isn't cute.



Different people. Different needs. One suite solution.

With the latest versions of Macromedia Dreamweaver®, Flash® Professional, Fireworks®, Contribute™, and FlashPaper™, the new Studio 8 is quite a catch. To meet Studio 8 and find all the web design and development tools you need, visit www.macromedia.com/go/8_studio8.

macromedia®
STUDIO 8