

The eLearning Guild's  
**LEARNING SOLUTIONS**<sup>SM</sup>

*Practical Applications of Technology for Learning* e-Magazine

**THIS WEEK: Development Techniques**

## Make Learners (and Editors) Happy With Small Flash 8 Files

**By Thomas Toth**

**T**he age of high speed Internet connections – at home, in the office, and on the road – has certainly diminished developer angst over the creation of high quality, multimedia-intensive online learning. While the concerns that we have about quality versus file size are less now than they were in the days when dial-up was dominant, there are still benefits for keeping file sizes down when building in Flash 8.

Embedding video segments, audio segments, and graphic files into the final Flash file (SWF) has been a standard practice for a while, and splitting your movies up into smaller chunks is a best practice. However, as the demands for higher quality video and audio over the Web keep increasing, the break-up strategy has an impact on the final e-Learning project. Higher quality means larger file size, which means longer load times. A CD-quality audio file ripped to please the most demanding audiophile will be much larger than an audio file optimized down to mono and a lower quality.

For several years, my solution to this problem has been to keep the final Flash files (the SWF files) small and to dynamically load only the requested files. There are at least three advantages to doing things this way:

- 1) The developer doesn't have to manage and maintain a huge source file (FLA). When all the graphic and media files are imported into the source file, this single file can become quite large. Opening it to make edits or additions can become tedious as Flash spends time loading and crunching the library files.
- 2) A single "control" Flash file (the SWF) that is responsible for loading menus, toolbars, help bars, and for individual module control allows your

*Even with high-speed internet connections as common as they are, it is still important to keep Flash file sizes small. By storing your large media and text files outside the Flash file, you can improve your learners' experience of your course, facilitate updates and maintenance of content, and make the lives of your subject matter experts and editors much more pleasant. Read this week's article to learn how to efficiently use Flash 8 features that make this possible!*

A publication of



team to develop the content independently, and to control it all using this single SWF.

- 3) The learner doesn't want to wait for content to load. Even with the use of "pre-loaders" (which is mandatory even for smaller SWF files), some learners will rate a program as being lower quality if there are several long content loads.

Let's set the file size issue aside for just a second. Projects that contain lots of media, graphical, and text elements can be a real pain to edit and maintain, especially for non-Flash users (the typical content authors and subject matter experts). Using the traditional embedded method, it doesn't matter whether or not you have broken the files into smaller, loadable SWF movies. (See Sidebar 1 on page 3.) Breaking them up doesn't address the edit issues. To make changes, the person doing the editing must crack open the source file (the FLA), make the edits, re-publish the file (back to SWF), and upload the published Flash file back to the final location.

While this is standard practice (not necessarily best practice), it is not the best way to build your e-Learning. As e-Learning developers and producers, we tweak content and materials constantly –

text gets updates, animations change based on updated text, and, in some instances, we use and re-use content multiple times. It is with this in mind that I suggest keeping certain text and media files out of the final SWF file. By uploading the graphic, text, and media files to the same directory as your Flash file, you can keep the overall file sizes smaller, make download times shorter and make editing your Flash files easier – even for non-Flash developers.

The rest of this article will show you the process for loading text (including HTML-formatted text), graphics, other SWF files, and media files into your project at run time. These files will stay separate from the running SWF, but will come into the file using ActionScript and components.

### Loading external JPG images into Flash

JPG (or JPEG) images are a common Web site feature, especially if you are using color photographs in your site or e-Learning project. Odds are you have a bunch of them already created. Why recreate them or bring them into the source file when they are perfectly happy sitting outside on the Web server? I'll start by showing you how to use a com-

*Projects that contain lots of media, graphical and text elements can be a real pain to edit and maintain, especially for non-Flash users (the typical content authors and subject matter experts).*

The eLearning Guild's  
**LEARNING SOLUTIONS**<sup>SM</sup>  
Practical Applications of Technology for Learning **e-Magazine**

**Publisher** David Holcombe

**Editorial Director** Heidi Fisk

**Editor** Bill Brandon

**Copy Editor** Charles Holcombe

**Design Director** Nancy Marland Wolinski

**The eLearning Guild™ Advisory Board**

Ruth Clark, Lance Dublin, Conrad Gottfredson, Bill Horton, Bob Mosher, Eric Parks, Brenda Pfau, Marc Rosenberg, Allison Rossett

Copyright 2002 to 2006.

**Learning Solutions e-Magazine™** (formerly **The eLearning Developers' Journal™**). Compilation copyright by The eLearning Guild. All rights reserved. Please contact **The eLearning Guild** for reprint permission.

**Learning Solutions e-Magazine™** is published weekly for members of **The eLearning Guild**, 525 College Avenue, Suite 215, Santa Rosa, CA 95404. Phone: +1.707.566.8990 [www.eLearningGuild.com](http://www.eLearningGuild.com)

**Learning Solutions e-Magazine™** serves as a catalyst for innovation and as a vehicle for the dissemination of new and practical strategies, techniques, and best practices for e-Learning design, development and management professionals. It is not intended to be THE definitive authority ... rather, it is intended to be a medium through which e-Learning professionals can share their knowledge, expertise, and experience. As in any profession, there are many different ways to accomplish a specific objective. **Learning Solutions** will share many different perspectives and does not position any one as "the right way," but rather we position each article as "one of the right ways" for accomplishing an objective. We assume that readers will evaluate the merits of each article and use the ideas they contain in a manner appropriate for their specific situation.

The articles in **Learning Solutions** are all written by people who are actively engaged in this profession – not by journalists or freelance writers. Submissions are always welcome, as are suggestions for future topics. To learn more about how to submit articles and/or ideas, please visit our Web site at [www.eLearningGuild.com](http://www.eLearningGuild.com).

ponent called “Loader” to allow you to leave those images on the server. (See Sidebar 2 below for an important note.)

Flash components are custom-made movie clips with the complex ActionScript created for you. Components allow you to quickly add cool elements to your project without all the hair-pulling and debugging involved in programming it yourself. The advanced functionality hides behind a simple user interface. Some examples of the most popular components include text scrollers, radio buttons, and media players (which we’ll also use in this article).

For this section, we are going to use the Loader component. This container can display a Flash movie (a SWF) or a JPG image. It pulls external content into a Flash movie. This means you can play a Flash movie inside a Flash movie, as well as showing diagrams, photos, logos, and other graphic content that is not in the Flash file (as long as it is saved as a JPG).

### Step by step instructions

Begin by creating an empty folder on your desktop. We will be storing the source file and the JPG file in this folder.

Open Flash 8 and create a new Flash document. You can keep all of the default settings. Save this file into the folder you just created on your desktop.

If your component panel isn’t open, choose Window > Components from within Flash. The Components panel will open on the right side of the screen. The Loader component is located in the User Interface section of the Component Panel. Click on the Loader component and drag an instance of this component to your stage. A box with the text “mx.controls.Loader” in it will appear. (See Figure 1 on page 4.)

With the Selection Tool (black arrow) select your Loader instance and access the Properties panel. Click on the “parameters” tag and the panel will change, allowing you to alter the properties of the component.

Move a JPG image from your hard drive into the newly created folder on your desktop. Remember the name of this image file – you are going to use it in the component inspector from within Flash.

Back in Flash, edit the “contentPath” field and put the name of your image into that spot. Click on the empty location and it will allow you to type in the name of your image, including the extension (JPG). In Figure 2 on page 4, my image is called smSpot001.JPG.

Once you have this completed, save and test your movie. (Control > Test Movie). Your image will load into Flash.

The image will load into the component regardless of pixel dimensions. If your component is 100 x 100 and your image is 250 x 250, the image will display at runtime as 100 x 100. You have to change the Width and Height values from the component inspector to match the size of your original image. (The scale tool will work just as well.) However, if you don’t know the original size of your image, change the `scaleContent` parameter to “false” – the image will load in at its original size.

If your e-Learning project requires you to use the same images in multiple projects, this technique can be very helpful. Also, if subject matter experts or authors are supplying you with images, you can use this strategy when those images will arrive at a later date. Knowing the pixel size of the images to be loaded allows you to create a placeholder image until the real one arrives. Replace the placeholder image with the final image and you don’t

*... I suggest keeping certain text and media files out of the final SWF file. By uploading the graphic, text, and media files to the same directory as your Flash file, you can keep the overall file sizes smaller, make download times shorter and make editing your Flash files easier ...*

### Sidebar 1 FLA and SWF

The author has assumed that readers will have some basic knowledge of Flash. For the sake of less-experienced readers, here are two explanations to keep in mind.

A SWF (“filename.SWF”) is a Flash file. This is the one that the learner downloads and “runs” on his or her computer. It is not editable (see the following explanation of FLA). Flash files are often called “movies” but this can be misleading. A SWF may represent animation, plus sound, delivered to the user. A SWF may also contain applets that control the delivery. In this article, the author shows how to create a SWF that delivers text, media, and other SWFs.

A FLA (“filename.FLA”) is an editable source file. It often contains all the text, animation, graphics, and ActionScript that will go into the Flash file. You must have Flash MX 8 installed on your computer to manage and edit these source files. However, by following the instructions in this article, you can separate your content (the text, animation, graphics, and so on) and place it in external text and media files that your authors and Subject Matter Experts can edit without having to republish the FLA as a new SWF.

### Sidebar 2 Progressive JPG files don’t work here

One thing you have to keep in mind as you apply the procedures in this article – you cannot save your images as “progressive” JPG files. A progressive JPG image is one that loads into the browser window in “stages” so users see a pixilated version of the image that becomes clearer and clearer as it loads into the browser. Making your images progressive was standard practice in the days of slow connections. It allowed your users to see “something” before the image finished downloading to their browser window.

In most image editing software, like Photoshop or Fireworks, you must manually assign the image to be saved as progressive, so I would suspect that most of your images are saved as standard JPGs. If you encounter errors with the technique explained in this article, you may have images saved as progressive. Simply open them up in your image editing software of choice, and re-save them as non-progressive JPGs.

need to open Flash. You could also provide graphic designers with a list of file names and pixel dimensions, and then send the developer all the images at one time. Uploading the images to the Web server enables Flash to find them and dynamically place them in these components at run time.

### ActionScript gives additional flexibility

You can also use ActionScript to pass the name of your image to Flash. This allows you to turn the image name into a variable, or manipulate the image name in a number of other ways. To accomplish this, we have to give our component instance a unique name. Once it has a unique name, you can manipulate the instance of the component using ActionScript. For this test, let's give our component the name "imageLoader". In this test, we are going to load our image through ActionScript. Make sure that you remove the name of your image from the `contentPath` parameter as well. (See Figure 3, below.)

Click once on the only Frame in the timeline and access the Action panel. We are going to edit the property of the `contentPath` parameter using the following code:

```
imageLoader.contentPath =
"smSpot001.JPG";
```

The dot notation above tells Flash to load the graphic file into the clip named `imageLoader`. If you save and test your movie, your image should load as expected.

You could also use this technique with variables so that you can use external text files to name your images.

```
imageLoader.contentPath =
imageVariable;
```

Then, from within your external text file, you declare the `imageVariable` value and the custom image displays. While this is getting a little bit ahead of ourselves, it is a great introduction to the next tutorial.

### Loading external text files into Flash

For those developers who are building text-based content in Flash, edits are going to occur more often than you expect. While I am not a big fan of paragraphs of text appearing in online learning programs, the reality is that sometimes it occurs, so why not make it easier to edit and manipulate? It is infinitely easier for a subject matter expert who is not experienced with Flash to edit external text files in a text editor than it for that same expert to open the source file to make changes.

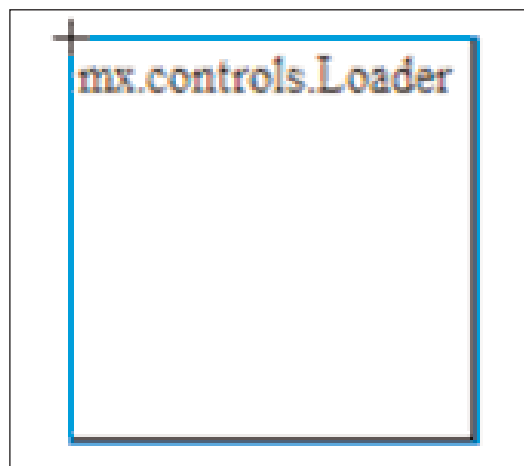
Of course, this procedure also works if you are loading content dynamically from a database or

server-side application. Text and image data (as you saw above) can come into Flash from a database, LMS, or other application using this method as well.

To begin, open your text editor (MS NotePad, for example) and create a simple string that we will use in our Flash movie. In the text editor, type:

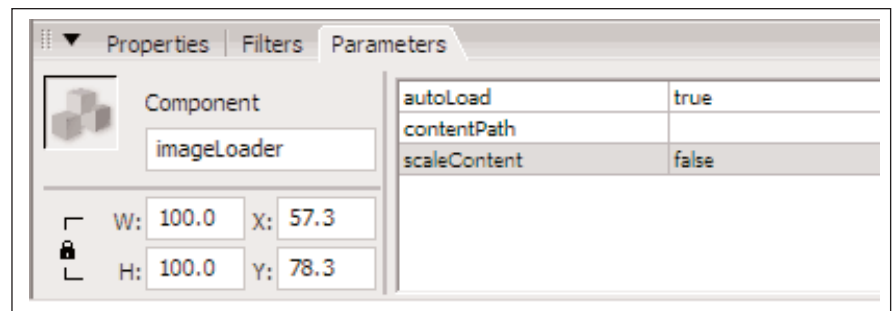
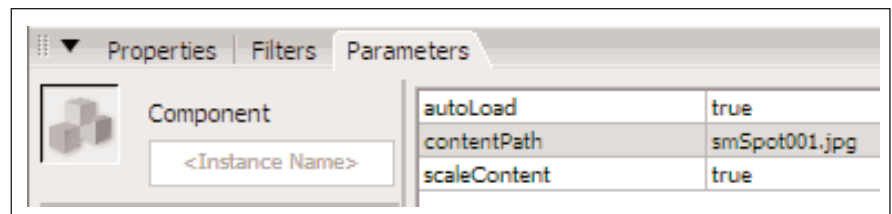
```
textVariable=Testing...testing...is
this thing on?
```

Save this file as `sampleText.txt` file in the directory you created on your desktop. That's it for

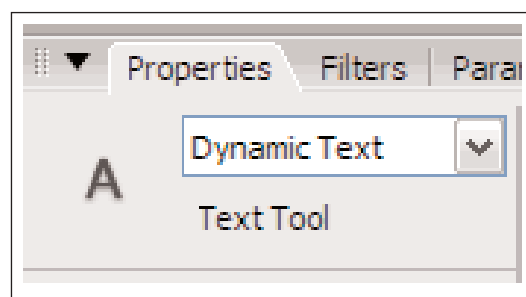


← **Figure 1** The Loader component looks like this on your stage.

↙ **Figure 2** Enter your image filename in the `contentPath` field of the Parameters tab.



↖ **Figure 3** The component's new name is "imageLoader" and the name of the image in the `contentPath` parameter is now blank



← **Figure 4** Creating a dynamic text field in the Properties Inspector

the text file. Go back into Flash and create a new Flash document. Save this source document as `text.FLA`.

The next step is to create a dynamic text field. To do this, click on the Text tool, and, in the Properties Inspector, change the drop down from Static to Dynamic Text. (See Figure 4 on page 4.)

With the text tool still active, click and drag a text box on the stage. This will be the area where your external text file content will load!

A quick click is next – with the text box still active, change the drop down that displays Single Line to Multiline. This will ensure that our text wraps from line to line in our text box.

Just like our image example, we need to provide our text box with a unique name. To do so, while your text field is still selected, click in the Properties Inspector once again. Give your text field an instance name of `text001`. (See Figure 5 on page 6.) Just like our image example, an object's instance name is unique and ActionScript uses it to manipulate any object.

Click once on the only frame and add the following ActionScript:

```
myData = new LoadVars();
myData.onLoad = function() {
    text001.text = this.textVariable;
}
myData.load("sampleText.txt");
```

`LoadVars` is instanced into an object that we can call in future lines of ActionScript and therefore control. `myData` is the name we have chosen for our instance of `LoadVars`.

The second line creates the function that will tell the movie to do something once the file is loaded. What the function will accomplish is that any loaded text that has the `textVariable` variable will display in the `text001` instance.

The last line of code tells Flash which file to load (`sampleText.txt`). The function looks for the `textVariable` variable and displays the text after `sampleText.txt` loads.

### HTML formatted text

In addition to loading plain text, it is possible to load HTML formatted text. Not all HTML tags are supported, but most of the common ones are. After all, you are most likely only going to apply text formatting to your loaded text, but the `<a>` tag to create hyperlinks is also used quite a bit. Yes, you can embed an HTML hyperlink into text dynamically loaded into Flash.

## THE ELEARNING GUILD'S ONLINE FORUMS<sup>SM</sup>

**June 15 & 16, 2006**

### Advanced Topics in e-Learning Instructional Design

- Learn new ID techniques and models
- Explore approaches for expediting the ID process
- Discover new and proven e-Learning ID models
- Examine some of the latest research in instructional design
- Ensure that your e-Learning instructional design has impact

**Register Today! +1.707.866.8990**  
**[www.eLearningGuild.com](http://www.eLearningGuild.com)**

Hosted by:



Technology Sponsor:



To load an HTML formatted text file, you will need to modify your existing text file with HTML tags and save it. The second step is to modify the ActionScript to allow for the proper display of HTML inside the text box:

```
myData = new LoadVars();
myData.onLoad = function() {
    text001.html=true;
    text001.htmlText = this.textVariable;
}
myData.load("sampleText.txt");
```

Adding these lines enables the HTML code to be interpreted and display correctly in the loaded text.

Going forward, edits to this text file will dynamically change the text in the Flash project every time the program is run. The changes don't require the Flash project to re-publish or have to be placed back onto the Web server. Editing the text file is easy – any user will be able to modify the text and your project will update without any additional work from you.

### Loading external SWF files into Flash

Breaking your larger Flash files into smaller ones is a great best-practice. There are two different ways to accomplish this. Both use ActionScript and both are easy to use.

The first is to use `loadMovie()`. The entire string of ActionScript is:

```
loadMovie("movie.SWF", load_mc);
```

What this command accomplishes is to load the file `movie.SWF` into a movie clip named `load_mc`. Let's do that one first.

Create a new Flash document and save it in the folder you created on your desktop. With the new movie open, choose `Insert > New Symbol`. When the dialog box opens, name your symbol `loader` and ensure that `Type` is set to `Movie clip`. Click `OK`.

The movie clip `loader` is in your library. Drag an instance of `loader` onto your stage.

Click on the tiny target icon to bring up the attributes in the Properties Inspector. This symbol needs a unique name in order to manipulate it with ActionScript. Call this symbol `loader001`. (See Figure 6 at right.)

On the only frame of the movie, add the following ActionScript:

```
loadMovie("text.SWF", loader001);
```

This loads the external SWF file (in my example, I chose `text.SWF` – the Flash file we created dur-

ing the last section on loading text dynamically) into the movie clip instance named `loader001`.

Save and test your movie. Note that the Flash comes in with the 0,0 point of the new file in the upper corner of the movie clip, so you have to move the movie clip around until it is positioned where you'd like it.

The second method is to use `loadMovieNum()`. The ActionScript command for loading a movie using this method is:

```
LoadMovieNum("yourMovie.SWF", Level);
```

The file `yourMovie.SWF` is obviously replaced by the name of your valid SWF file, but the `Level` field is completed by adding a number (a level) to this field.

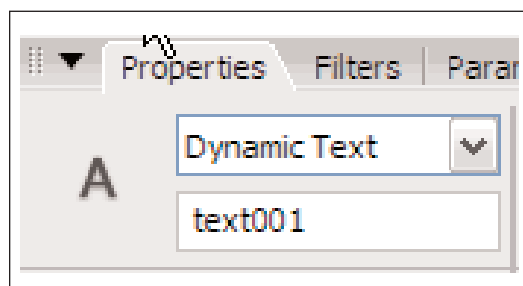
### Levels

Let's take a second to talk about this `Level` thing. Think of `Level` numbers a little like the layers on your timeline. Objects located at the top of the timeline appear "above" objects located at the bottom. This `Level` attribute determines the location of the loaded movie: the higher the number, the further "above" the other levels objects appear. (See Figure 7 on page 7.)

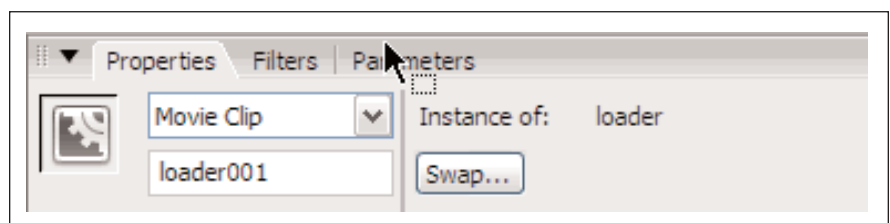
Your main movie level is considered to be `Level 0` (`_level0`). `Level 0` defines the stage size, the frame rate and background color for all the levels loaded above it. The levels loaded above `Level 0` are transparent: background colors **do not** apply. However, all buttons and symbols are active on all layers. A button on `Level 0` can interact with a movie on `Level 1` and so on.

When you wish to load a movie, you use the `LoadMovieNum` ActionScript command and set the parameters accordingly. So, if I have my movie root

*For this section, we are going to use the Loader component. This container can display a Flash movie (a SWF) or a JPG image. It pulls external content into a Flash movie. This means you can play a Flash movie inside a Flash movie, as well as showing diagrams, photos, logos, and other graphic content that is not in the Flash file (as long as it is saved as a JPG).*



← **Figure 5** Name the text field in the Properties Inspector.



↘ **Figure 6** Name your new symbol in the Properties Inspector.

with background imaging on Level 0, and I have an audio SWF and an interactive menu SWF I want to load. I would use the following commands:

```
LoadMovieNum("audio.SWF", 1);
LoadMovieNum("menu.SWF", 2);
```

From a user perspective, they are seeing one movie, but you have segmented it into pieces that are more manageable.

If you wanted to use the buttons in the `menu.SWF` file (on Level 2) to alter the timeline of the movie on Level 0, you can simply call the level and frame name or number like this:

```
on (release){
    _level0.gotoAndPlay("intro");
}
```

This tells Flash that when the user releases the mouse button, play the frame named "intro" on Level 0. You can give instructions to named movie symbols, named text fields, frames, and scenes from different levels.

The location of the new layers is directly above or beneath the other layers. Layout is determined by the loaded SWF file, so if objects appear at 50,50 in the loaded layer, they will appear in 50,50 of the `_level0` layer. Positioning objects using this method is a little easier than when using the regular `loadMovie()` command.

### Loading external multimedia files into Flash

Flash 8 has made using external media files much easier to manage by asking you how you'd like to import your media. One of the options now is to keep the media as an external file.

Importing a video into Flash 8 starts with `File > Import Video`. This new command opens the video import wizard. (See Figure 8 at right.)

From the first screen, you browse to the video file you wish to import into your Flash movie.

The first choice on the second screen is called Progressive Download from a Web Server. Choosing this option does several things to your video:

- 1) Converts your multimedia file to a FLV (Flash Video) file. The Flash Video file is an optimized version of your chosen video format. It doesn't necessarily mean that your final movie file will be smaller than the original, but converting to an FLV provides you with more flexibility with the Flash movie.

- 2) Automatically streams your video using HTTP streaming. You don't need a streaming server

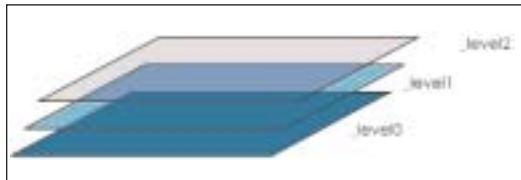
or any special hardware to make this happen. Flash handles it for you with the player and the Flash Video file.

The next screen allows you to encode, edit, and add cue points to your movie. You accomplish this by clicking on the Show Advanced Setting button in the wizard. You can play with these settings to optimize your video, reducing the overall file size and quality of the audio and the video image. There are some default settings at the top, but if you want to tweak the video itself, the Advance Setting button will allow you to do just that.

Clicking the Next button moves you to the skinning screen. This is where you get to choose the player look, feel, and type from a long list of drop downs. Choosing a skin determines how your movie will appear to the user, and also provides controls like stop, start, and rewind.

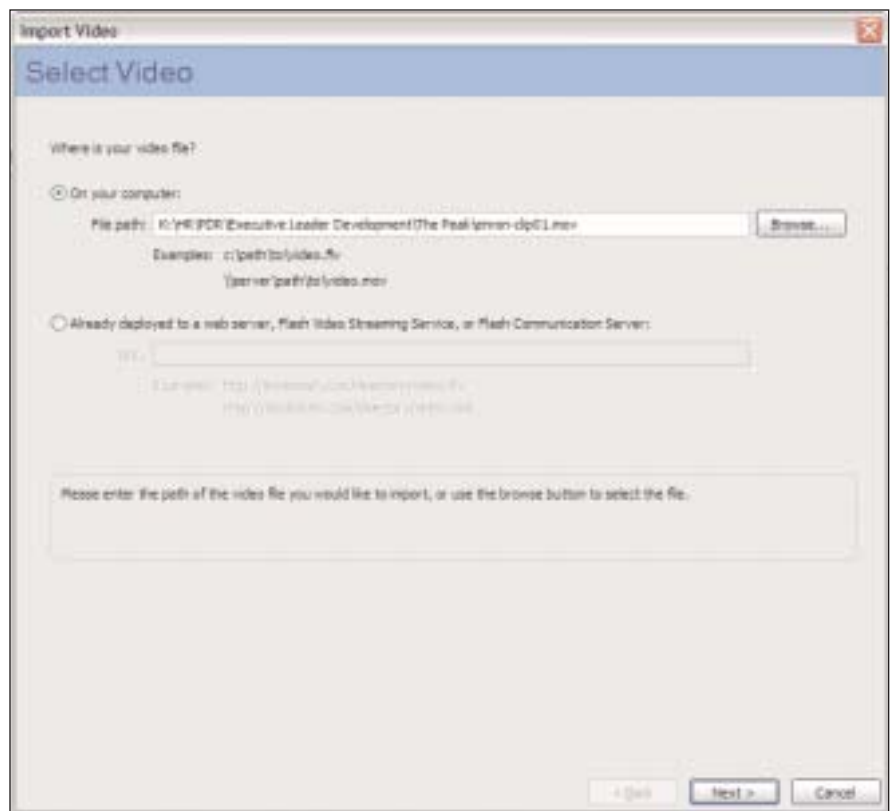
One downside to all video is the fact that it can take a long time to import and convert. If your movie is 10 minutes long, plan to spend 20-30

*One downside to all video is the fact that it can take a long time to import and convert. If your movie is 10 minutes long, plan to spend 20-30 minutes of idle time while your computer optimizes and compresses the movie. For higher quality movies, expect to spend more time.*



← Figure 7 Level numbers in Flash


↘ Figure 8 The Video Import wizard.



minutes of idle time while your computer optimizes and compresses the movie. For higher quality movies, expect to spend more time.

Once the video has been converted to an FLV, the player will appear on the stage. From here you now can position it on the stage and you are ready to watch the video. The FLV is separate from the SWF, but both need to be loaded onto the Web server for the video to play. You can accomplish edits to the original video file using Quicktime Pro and then export to FLV from the same program. This works on both PCs and Macs, so there isn't a need to get back into Flash for simple cuts and edits to the final video.

### Conclusion

As I stated at the beginning of this article, keeping content out of the final Flash file helps keep download times short, but the greatest benefit to the e-Learning developer is in the area of edits and updates. By keeping your media, text, and graphic files out of the final Flash file, you can edit and tweak these elements without completely re-publishing it. The time you spend putting these techniques into action pays for itself after you have to make edits to your first e-Learning project. 

### Author Contact



Thomas Toth is a designer with a decade of training, management, and design experience in the computer and technical education industries. He is a dedicated Web designer, and has personally created and maintained over three

dozen professional Internet and intranet Web sites. In the field of e-Learning, Thomas has designed and programmed several dozen on-line and CBT training courses, from simple Flash-based product demonstrations to complex, multi-modular on-line courses. He has been recognized by ASTD as a leader and expert in the field of e-Learning. Thomas wears the hats of Instructional Designer, Project Manager, HTML Programmer, Graphic Designer, Flash Programmer, and Stand-Up Trainer. He and other members of TothWeb teach Dreamweaver, Fireworks, Flash, Photoshop, and PageMaker, bringing students real-world knowledge and experience on how to use these products.

Contact Thomas by email to [ttoth@dWebstudios.com](mailto:ttoth@dWebstudios.com).

*Additional information on the topics covered in this article is also listed in the Guild Resource Directory.*

## DO YOU HAVE AN INTERESTING STRATEGY OR TECHNIQUE TO SHARE?

Get It Published in...



This publication is by the people, for the people.

That means it's written by YOU the readers and members of **The eLearning Guild!** We encourage you to submit articles for publication in **Learning Solutions e-Magazine**.

Even if you have not been published before, we encourage you to submit a query if you have a great idea, technique, case study or practice to share with your peers in the e-Learning community. If your topic idea for an article is selected by the editors, you will be asked to submit a complete article on that topic. Don't worry if you have limited experience writing for publication. Our team of editors will work with you to polish your article and get it ready for publication in **Learning Solutions**.

By sharing your expertise with the readers of **Learning Solutions**, you not only add to the collective knowledge of the e-Learning community, you also gain the recognition of your peers in the industry and your organization.

### How to Submit a Query

If you have an idea for an article, send a plain-text email to our editor, Bill Brandon, at [bbrandon@eLearningGuild.com](mailto:bbrandon@eLearningGuild.com), with the following information in the body of the email:

- **A draft of the first paragraph**, written to grab the reader's attention and identify the problem or issue that will be addressed.
- **A short outline of your main points** addressing the problem or resolving the issue. This could be another paragraph or it could be a bulleted list.
- **One paragraph on your background or current position** that makes you the one to tell this story.
- **A working title** for the article.
- **Your contact information:** name, job title, company, phone, email. This information is to be for the writer of the article. We are unable to accept queries from agents, public relations firms, or other third parties.

All of this information should fit on one page. If the topic fits our editorial plan, Bill will contact you to schedule the manuscript deadline and the publication date, and to work out any other details.

Refer to [www.eLearningGuild.com](http://www.eLearningGuild.com) for Author Guidelines.



## A Worldwide Community of Practice for e-Learning Professionals

The eLearning Guild is a Community of Practice for e-Learning design, development, and management professionals. Through this member driven community we provide high-quality learning opportunities, networking services, resources, and publications. Members represent a diverse group of managers, directors, and executives focused on training and learning services, as well as e-Learning instructional designers, content developers, Web developers, project managers, contractors, and consultants. Guild members work in a variety of settings including corporate, government, and academic organizations.

Guild membership is an investment in your professional development and in your organization's future success with its e-Learning efforts. Your membership provides you with learning opportunities and resources so that you can increase your knowledge and skills. That's what the Guild is all about ... putting the resources and information you need at your fingertips so you can produce more successful e-Learning.

The eLearning Guild offers four levels of membership. Each level provides members with benefits commensurate

with your investment. In the table you will find a comprehensive summary of benefits offered for each membership level. To learn more about Group Membership and pricing, go to [www.eLearningGuild.com](http://www.eLearningGuild.com).

Guild Benefits	Associate	Member	Member+	Premium
eLearning Insider	✓	✓	✓	✓
Annual Salary Survey	✓	✓	✓	✓
Past Conference Handouts	✓	✓	✓	✓
Resource Directory – Access & Post	✓	✓	✓	✓
Info Exchange – Access & Post	✓	✓	✓	✓
Job Board – Access Jobs & Resumes	✓	✓	✓	✓
Job Board – Post Resumes	✓	✓	✓	✓
Job Board – Post Jobs	✗	✓	✓	✓
Guild Research – Online Briefings	✓	✓	✓	✓
Guild Research – Reports	✗*	✓	✓	✓
Guild Research – Archives	✗	✓	✓	✓
Learning Solutions e-Magazine	✗*	✓	✓	✓
Online Events Archive	✗	✗	✓	✓
Online Forums	\$	\$	✓	✓
Face-to-Face Conferences	\$	\$	\$	✓*
Pre-Conference Workshops	\$	\$	\$	✓*
Event Fee Discounts	✗	20%	20%	20%
Online Event Site License Discounts	✗	✗	20%	20%

\*See [www.eLearningGuild.com](http://www.eLearningGuild.com) for details

✓ = Included in Membership   ✗ = Not available   \$ = Separate fee required

The eLearning Guild organizes a variety of important industry events...



April 18 - 21, 2006  
BOSTON



April 18 - 21, 2006  
BOSTON



April 18 - 21, 2006  
BOSTON



July 13 & 14, 2006  
ONLINE



October 10 - 13, 2006  
SAN FRANCISCO



October 10 - 13, 2006  
SAN FRANCISCO